On the initialization of the DNMF algorithm

Ioan Buciu, Nikos Nikolaidis and Ioannis Pitas
Department of Informatics
Aristotle University of Thessaloniki
GR-54124, Thessaloniki, Box 451, Greece
Email: {nelu,nikolaid,pitas}@zeus.csd.auth.gr

***Abstract*— A subspace supervised learning algorithm named Discriminant Non-negative Matrix Factorization (DNMF) has been recently proposed for classifying human facial expressions. It decomposes images into a set of basis images and corresponding coefficients. Usually, the algorithm starts with random basis image and coefficient initialization. Then, at each iteration, both basis images and coefficients are updated to minimize the underlying cost function. The algorithm may need several thousands of iterations to obtain cost function minimization. We provide a way to significantly improve the speed of the algorithm convergence by constructing initial basis images that meet the sparseness and orthogonality requirements and approximate the final minimization solution. To experimentally evaluate the new approach, we have applied DNMF using the random and the proposed initialization procedure to recognize six basic facial expressions. While fewer iteration steps are needed with the proposed initialization, the recognition accuracy remains within satisfactory levels.**

I. Introduction

A supervised Non-negative Matrix Factorization (NMF) algorithm, especially designed to cope with pattern recognition task, called Discriminant-NMF (DNMF), has been recently proposed in [1]. The algorithm is a modified version of LNMF [2] that decomposes an image database $\mathbf{X}$ into basis images $\mathbf{Z}$ and a coefficient matrix $\mathbf{H}$, in such a way, that it incorporates class information into the coefficient matrix. As expected, DNMF has shown superiority over NMF [3] and LNMF when applied on a facial expression recognition task, the method based on DNMF yielding the highest recognition rate. The cost function associated with DNMF algorithm leads to a modified update procedure for $\mathbf{Z}$ and $\mathbf{H}$ by using the same expectation-maximization (EM) strategy utilized by NMF and LNMF algorithms. Divergence minimization between the image data set and the product of the decomposition factors, sparseness of basis images, redundant information reduction between basis images (by imposing orthogonality) and feature selection based on class information lead to an underlying cost function that is composed of five objective terms. The algorithm starts with random values for $\mathbf{Z}$ and $\mathbf{H}$ and, at each iteration, it modifies them in order to find the minimum of the cost function. However, since these objective terms are coupled and must be simultaneously minimized, one caveat of this algorithm is that it may need several thousands of iterations to converge to its minimum. In this paper we provide a way to speed up the convergence of DNMF. This is performed by constructing initial basis images, whose values are not randomly chosen but contain information taken from the original image database. The redundant information of these initial basis images is then minimized by imposing orthogonality "a priori" (see Sections II and III). Additionally, the initial basis images are projected into a sparse and non-negative feature space. Finally, by using a least square approach, we give a first approximation to $\mathbf{X} \approx \mathbf{ZH}$ that brings us closer to the final solution. Thus, instead of using initial random factors $\mathbf{Z}$ and $\mathbf{H}$ as input to the DNMF algorithm the factors derived by a new initialization procedure are used.

II. Discriminant Non-negative Matrix Factorization

Let us suppose that we have an $m\times$ n non-negative matrix $\mathbf{X}$ whose column vectors are formed by the pixel values obtained by lexicographically scanning an image, i.e. $\mathbf{x}_j = [x_1, x_2, \ldots, x_m]^T$, where $j = 1, \ldots, n$ and m is the number of pixels in the image. We want to approximate $\mathbf{X}$ by a product of two non-negative matrices (factors) $\mathbf{Z}$ and $\mathbf{H}$ of size $m \times p$ and $p \times n$, respectively, i.e. $\mathbf{X} \approx \mathbf{ZH}$. The columns of $\mathbf{Z}$ form the basis images and $\mathbf{H}$ contains in its rows the decomposition coefficients. Let us now further suppose that we have $\mathcal{Q}$ distinctive image classes and n_c is the number of image samples in a certain class $\mathcal{Q}$, $c = 1, \ldots, \mathcal{Q}$. Each image from the image database corresponds to one column of matrix $\mathbf{X}$ and belongs to one of these classes. Therefore, each column of the $p \times n$ matrix $\mathbf{H}$ can be considered as an image representation coefficient vector $\mathbf{h}_{(c)l}$, where $c = 1, \ldots, \mathcal{Q}$ and $l = 1, \ldots, n_{(c)}$. The total number of coefficient vectors is $n = \sum_{c=1}^{\mathcal{Q}} n_{(c)}$. We denote the mean coefficient vector of class c by $\boldsymbol{\mu}_{(c)} = \frac{1}{n_c}\sum_{l=1}^{n_{(c)}} \mathbf{h}_{(c)l}$ and the global mean coefficient vector by $\boldsymbol{\mu} = \frac{1}{n}\sum_{c=1}^{\mathcal{Q}}\sum_{l=1}^{n_{(c)}} \mathbf{h}_{(c)l}$. If we express the within-class scatter matrix by $\mathbf{S}_w = \sum_{c=1}^{\mathcal{Q}}\sum_{l=1}^{n_{(c)}}(\mathbf{h}_{(c)l} - \boldsymbol{\mu}_{(c)})(\mathbf{h}_{(c)l} - \boldsymbol{\mu}_{(c)})^T$ and the between-class scatter matrix by $\mathbf{S}_b = \sum_{c=1}^{\mathcal{Q}}(\boldsymbol{\mu}_{(c)} - \boldsymbol{\mu})(\boldsymbol{\mu}_{(c)} - \boldsymbol{\mu})^T$, the cost function $cost_{DNMF}$ associated with DNMF algorithm is written as [1]:

$$cost_{DNMF} = KL(\mathbf{X}||\mathbf{ZH}) + \alpha\sum_{i,j} u_{ij} - \beta\sum_{i} v_{ii} + \\ + \gamma\mathbf{S}_w - \delta\mathbf{S}_b, \quad (1)$$

subject to $\mathbf{Z}, \mathbf{H} \geq 0$. Here $\mathbf{U} = \mathbf{Z}^T\mathbf{Z}$, $\mathbf{V} = \mathbf{H}\mathbf{H}^T$, α, β, γ and δ are constants. The other terms appearing in the cost function have the following meaning. The first term $KL(\mathbf{X}||\mathbf{ZH}) =$

$\sum_{i,j}\left(x_{ij}\ln\frac{x_{ij}}{\sum_k z_{ik}h_{kj}}+\sum_k z_{ik}h_{kj}-x_{ij}\right)$ is the Kullback-Leibler divergence ($k=1,\ldots,p$) and ensures that the product $\mathbf{ZH}$ approximates as much as possible the original data $\mathbf{X}$. The second term can be further split in two as $\sum_{i,j}u_{ij}=\sum_{i\neq j}u_{ij}+\sum_i u_{ii}$, where the minimization of the first sum forces the columns of $\mathbf{Z}$ to be orthogonal in order to reduce the redundancy between basis images, while the minimization of the second term guarantees the generation of sparse features in the basis images. The third term $\sum_i v_{ii}$ aims at maximizing the total "energy" on each retained component.

Starting at iteration $t=0$ with random positive matrices $\mathbf{Z}^{(0)}$ and $\mathbf{H}^{(0)}$, the algorithm updates their values according to an Expectation-Maximization (EM) approach, leading to the following updating rules for each iteration $t>0$ [1]:

$$(i)\quad h_{kl(c)}^{(t)}=\frac{2\mu_c-1}{4\xi}+ +\frac{\sqrt{(1-2\mu_c)^2+8\xi h_{kl(c)}^{(t-1)}\sum_i z_{ki}^{(t)}\frac{x_{ij}}{\sum_k z_{ik}^{(t)}h_{kl(c)}^{(t-1)}}}}{4\xi} \tag{2}$$

The elements h_{kl} are then concatenated for all $\mathcal{Q}$ classes as:

$$(ii)\qquad h_{kj}^{(t)}=[h_{kl(1)}^{(t)}\,|\,h_{kl(2)}^{(t)}\,|\,\ldots\,|\,h_{kl(\mathcal{Q})}^{(t)}] \tag{3}$$

where "|" denotes concatenation. The basis images are updated as:

$$(iii)\qquad z_{ik}^{(t)}=\frac{z_{ik}^{(t-1)}\sum_j\frac{x_{ij}}{\sum_k z_{ik}^{(t-1)}h_{kj}^{(t)}}h_{jk}^{(t)}}{\sum_j h_{kj}^{(t)}} \tag{4}$$

$$(iv)\qquad z_{ik}^{(t)}=\frac{z_{ik}^{(t)}}{\sum_i z_{ik}^{(t)}},\quad \text{for all } k \tag{5}$$

The final $\mathbf{Z}^{(f)}$ and $\mathbf{H}^{(f)}$ found at the last iteration are such that $\mathbf{X}\approx\mathbf{Z}^{(f)}\mathbf{H}^{(f)}$, S_w is as small as possible and S_b is as large as possible.

III. DNMF INITIALIZATION

Instead of starting with random $\mathbf{Z}^{(0)}$ we provide a way to build the initial basis $\mathbf{Z}^{(0)}$ in a such a way that they are already approximately orthogonal and sparse, before entering the DNMF iteration presented in (2) - (5). As a consequence, the convergence speed of the DNMF algorithm can increase substantially. Before presenting the proposed initialization algorithm, we present the following measure of sparseness σ of a vector $\mathbf{z}=[z_1,z_2,\ldots,z_m]^T$ as provided by Hoyer [4]:

$$\sigma(\mathbf{z})=\frac{\sqrt{m}-\frac{L_1}{L_2}}{\sqrt{m}-1}, \tag{6}$$

m being the number of image pixels. This definition of sparseness is expressed in terms of the relationship between the L_1 and the L_2 norm of $\mathbf{z}$, respectively, where $L_1=\sum_{i=1}^m|z_i|$ and $L_2=\sqrt{\sum_{i=1}^m z_i^2}$. Expression (6) equals to zero if all elements of $\mathbf{z}$ are equal (in which case the vector corresponds to an holistic image representation) and to unity, if only one element is nonzero (involving an extreme local image representation). A σ value between these two extremes corresponds to an image representation with varying degrees of sparseness. The measure of sparseness given in this form is useful, because it enables us to explicitly enforce the sparseness of a vector by modifying its L_1 norm, as will be seen below.

Given an image database $\mathbf{X}$, a random positive matrix $\mathbf{Z}^{(0)}$ and a parameter σ that, explicitly, controls the sparseness of basis images, the DNMF's initialization algorithm consists of the following steps:

1) The algorithm starts by projecting the image database onto the random basis images $\mathbf{H}_e=\mathbf{Z}^{(0)+}\mathbf{X}$, where "+" denotes matrix pseudoinversion.

2) The reconstructed images can then be written as $\mathbf{X}_{rec}=\mathbf{Z}^{(0)}\mathbf{Z}^{(0)+}\mathbf{X}$.

3) By subtracting the reconstructed images from the original images the residual matrix is formed as $\mathbf{R}=\mathbf{X}-\mathbf{X}_{rec}$.

The reasoning for applying these first three steps is that we wish the initial basis images to contain some information from the original database. This is done by constructing the aforementioned residual matrix. Moreover, as we shall see in the Section IV, employing the residual matrix will guide the features to be close to the final ones (mostly representing fiducial points) through the sparseness process (step 7). The algorithm continues as follows:

4) Sort the columns $\mathbf{r}_j$ ($j=1,\ldots,n$) of $\mathbf{R}$ so that $s_1<s_2<\ldots<s_n$, where $s_j=\sum_i r_{ij}$, and keep only $p<n$ columns, thus forming the matrix $\mathbf{R}_p$.

5) Given the set of vectors $\mathbf{r}_k$ (the columns of $\mathbf{R}_p$), $k=1,\ldots,p$, derive an orthogonal set of vectors $\mathbf{d}_k$ that has the same span as the original set of vectors $\mathbf{r}_k$. The span of a set of S vectors is the set of all linear combinations that can be formed using vectors in the set S. Collect these vectors in the columns of a matrix $\mathbf{D}$.

6) Set the L_1-norm for the columns of $\mathbf{D}$ as $L_1=\left[\sqrt{m}-(\sqrt{m}-1)\sigma\right]L_1$, so as to incorporate the desired sparseness σ into it.

7) For each column $\mathbf{d}_k$ of matrix $\mathbf{D}$, find the closest non-negative vector $\mathbf{z}_k^\star$ (in the Euclidean sense) corresponding to each column $\mathbf{d}_k$. This operation is performed by applying a projection operator technique proposed by Hoyer [4] and consists of the following steps applied to each column $\mathbf{d}_k$: a) $z_{ik}^\star=d_{ik}+L_1-\sum_i\mathbf{z}_{ik}/m$; for all i, b) set $P=\{\}$; c) $w_{ik}=L_1/(m-\text{size}(P))$ if $k\notin P$ or $w_{ik}=0$ otherwise; d) $\mathbf{z}_k^\star=\mathbf{w}_k+\lambda(\mathbf{z}_k^\star-\mathbf{w}_k)$; e) if $z_{ik}^\star\geq 0$ for all i, end; else f) set $P=P\cup\{i,\ z_{ik}^\star<0\}$; g) $z_{ik}^\star=0$ for all $i\in P$; h) compute $b=(\sum_i z_{ik}-L_1)/(m-\text{size}(P))$; i) set $z_{ik}^\star=z_{ik}^\star-b$, for all $i\notin P$; j) repeat a) - e) until all columns are non-negative. Basically, through the seventh step, the vector $\mathbf{d}_k$ is projected onto L_1 (step 7. a.) . Within this space $\mathbf{z}_k^\star$ is projected to the closest point on the joint constraint hypersphere, i.e. the intersection of the sum and the the L_2 constraint (step 7. d.). It also enforces non-negativity constraints by repeating the steps until all elements are non-

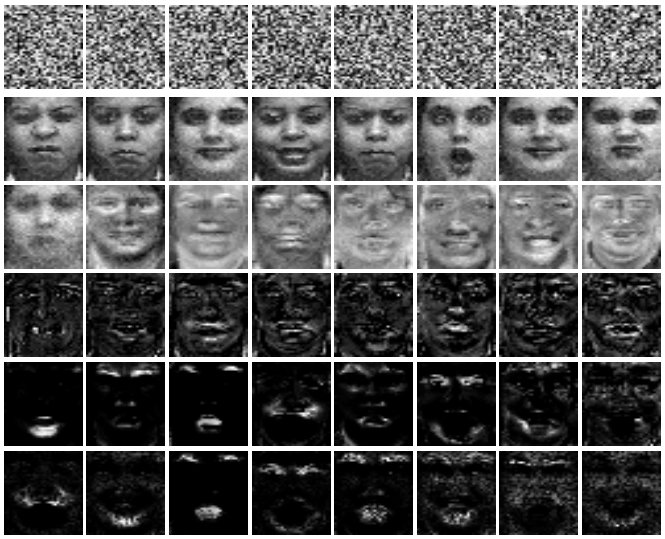

Fig. 1. Formation of several basis images through the proposed and random initialization.

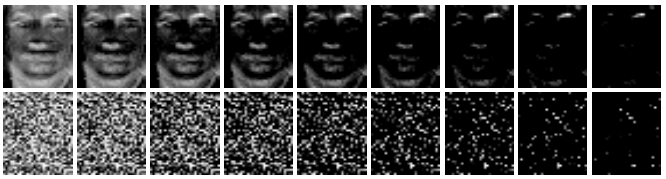

Fig. 2. Generation of one basis image (step 7) with $\sigma = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The first row depicts the case when this process is applied to one column of the matrix $\mathbf{D}$, while the second row shows the case when the process is applied to a column of the random matrix $\mathbf{Z}^{(0)}$.

negative. The parameter $\lambda \geq 0$ from step 7) d is selected such that the resulting $\mathbf{z}_k^\star$ satisfies the L_2 norm constraint. This is done by solving a quadratic equation [4].

8) Set $\mathbf{H}^\star = \mathbf{Z}^{\star T}\mathbf{X}$. The last step approximates $\mathbf{H}^\star$ through the product $\mathbf{Z}^{\star T}\mathbf{X}$. Once we have formed $\mathbf{Z}^\star$, a direct solution would be $\mathbf{H}^\star = \mathbf{Z}^{\star +}\mathbf{X}$. Involving the pseudoinverse could generate unwanted negative values for the coefficients. To avoid this situation we chose to use $\mathbf{Z}^{\star T}$ that gives us a satisfactory approximation of $\mathbf{Z}^{\star +}$ since the columns of $\mathbf{Z}$ are approximately orthogonal to each other.

IV. Experimental results and discussions

DNMF has been tested with the random and the proposed initialization methods for the recognition of six basic facial expressions namely, anger, disgust, fear, happiness, sadness and surprise from face images. The Cohn-Kanade AU-coded facial expression image database [5] was used in the experiments. The preprocessing steps and the training procedure that involves finding $\mathbf{Z}^{(f)}$ and $\mathbf{H}^{(f)}$ are described in [1].

The basis image generation process is depicted in Figure 1 for eight samples. Each row represents eight columns of the following matrices: random matrix $\mathbf{Z}^{(0)}$ (1st row), residual matrix $\mathbf{R}_p$ (2nd row), orthogonal columns of $\mathbf{D}$ (3rd row), initial proposed $\mathbf{Z}^\star$ (4th row), final $\mathbf{Z}^{(f)}$ when the input is $\mathbf{Z}^\star$ (5th row), and final $\mathbf{Z}^{(f)}$ when the input is $\mathbf{Z}^{(0)}$ (last row). Both 5-th and 6-th rows contain sparse basis images that are ordered according to their decreasing degree of sparseness.

As mentioned in Section III, employing the residual matrix can guide the selection of the features which will appear on the final $\mathbf{Z}^{(f)}$, when $\mathbf{Z}^\star$ is chosen. This is graphically illustrated in Figure 2, where the first row represents the seventh step from Section III for $\sigma = \{0.1, \ldots, 0.9\}$ applied to one column of $\mathbf{D}$, while the second row depicts the case when the the seventh step was applied to one column of $\mathbf{Z}^{(0)}$ for the same values of σ. Notice that, as σ grows, the facial fiducial points corresponding to mouth, chin, eyes or eyebrows are emphasized, whilst the selected features corresponding to the random matrix have rather random positions.

After the training process, $\mathbf{Z}^{(f)}$ and $\mathbf{H}^{(f)}$ are available. The zero mean image data are projected onto the image basis yielding a feature vector $\mathbf{F} = \mathbf{Z}^{(f)T}\mathbf{X}$. In the test phase, for each zero mean test face image $\mathbf{x}_{test}$, a test feature vector $\mathbf{f}_{test}$ is then formed as $\mathbf{f}_{test} = \mathbf{Z}^{(f)T}\mathbf{x}_{test}$. If we construct a classifier whose class label output for a test sample $\mathbf{f}_{test}$ is $\widetilde{l}$ then, the error rate is defined as the percentage of the misclassified labels (expressions) when $\{\widetilde{l}(\mathbf{f}_{test}) \neq l(\mathbf{f}_{test})\}$, where $l(\mathbf{f}_{test})$ is the correct class label. Once we have formed $\mathcal{Q}$ classes of new feature vectors (or prototype samples), a nearest neighbor classifier is employed to classify the new test sample, by using an Euclidean distance between a training feature and a test one. The Euclidean distance from $\mathbf{f}_{test}$ to $\mathbf{f}_c$ is expressed as $\|\mathbf{f}_{test} - \mathbf{f}_c\|^2 = -2g_c(\mathbf{f}_{test}) + (\mathbf{f}_{test})^T\mathbf{f}_{test}$, where $g_c(\mathbf{f}_{test}) = \mathbf{f}_c^T\mathbf{f}_{test} - \frac{1}{2}\|\mathbf{f}_c\|^2$ is a linear discriminant function of $\mathbf{f}_{test}$. Minimizing the distance is equivalent with maximize (due to the "minus" sign) the correlation between the test sample and training sample (hence we call this distance measure as Maximum Correlation Classifier - MCC). A test image is classified by this classifier by computing $\mathcal{Q}$ linear discriminant functions and choosing $l_{MCC} = \mathrm{argmax}_c\{g_c(\mathbf{f}_{test})\}$.

We ran DNMF for $p = \{16, 25, 36, 49, 64\}$ basis images and for $\sigma = \{0.1, 0.2, 0.3, 0.4\}$. The parameter ξ was kept fixed at 0.4999.

The number of iterations t for both initialization methods along with the initial and final values for KL, S_w and S_b are listed in Table I for $\sigma = 0.3$ and 16,25,36,49 and 64 basis images. Here "M" stands for the initialization method where "I" and "II" represent the random and the proposed method, respectively. Comparing the two initialization methods in terms of convergence, we can observe that random initialization gives slightly smaller value for $S_w^{(f)}$ than the proposed method. On the other hand $S_b^{(f)}$ is much greater for the proposed method than for random initialization method. Furthermore, the initial $KL^{(0)}$ for the proposed initialization is closer to the final $KL^{(f)}$ and it gets closer as p increases. This is a consequence of the step VIII and it indicates how precise is the approximation. However, the most important aspect is the number of iterations for both methods. Regardless of p, the DNMF algorithm needs less iterations when initialized

TABLE I

NUMBER OF ITERATIONS t, INITIAL AND FINAL VALUES FOR KL, $\mathbf{S}_w(h)$ AND $\mathbf{S}_b(h)$ FOR INITIALIZATION METHODS "I" (RANDOM) AND "II" (PROPOSED) AND FOR VARIOUS NUMBERS p OF BASIS IMAGES.

p	M	t	$KL^{(0)}$	$KL^{(f)}$	$S_w^{(0)}(h)$	$S_w^{(f)}(h)$	$S_b^{(0)}(h)$	$S_b^{(f)}(h)$	Execution time (seconds)
16	I	5588	6.58e+008	5.42e+006	208.93	137.69	0.25	2.85e+007	1425
	II	4971	5.66e+007	5.40e+006	4.06e+009	148.61	5.37e+005	4.46e+007	1258
25	I	4804	6.28e+008	5.41e+006	329.50	217.71	0.46	1.78e+007	1450
	II	3845	3.60e+007	5.40e+006	6.18e+009	232.82	6.44e+005	2.43e+007	1160
36	I	4206	6.05e+008	5.41e+006	477.59	313.79	0.58	1.53e+007	1558
	II	2805	2.17e+007	5.42e+006	8.80e+009	322.83	8.30e+005	1.27e+007	991
49	I	3733	5.84e+008	5.41e+006	646.47	425.83	0.81	1.24e+007	1609
	II	1773	1.25e+007	5.44e+006	1.20e+010	421.42	1.15e+006	4.97e+006	712
64	I	3350	5.66e+008	5.41e+006	833.12	552.80	0.98	9.05e+006	1740
	II	722	8.36e+006	5.46e+006	1.58e+010	533.42	1.32e+006	2.31e+006	375

with $\mathbf{Z}^\star$ and $\mathbf{H}^\star$. The last column tabulates the execution time necessary for DNMF to converge. Each iteration takes longer time to be executed as the number of basis images increases. The gain in the execution time, by employing the proposed initialization, is more and more significant as the number of basis images raises. For example, in the case of 36 basis images, the proposed initialization helped the algorithm to converge with 10 minutes earlier than the case when the random initialization was used. The difference is around 15 and 23 minutes when the number of basis images is 49 and 64, respectively. We must note that the execution time shown in the Table corresponds to a vector $\mathbf{x}$ of dimension 2700, i.e. an image of 60×45 pixels. A higher image resolution will lead to a longer execution time.

Although the main purpose of the paper was to provide an initialization method in order to reduce the number of iterations (and implicitly the convergence time) we also investigated how this initialization affects the classification error rate. Table II lists the classification error rate (%) corresponding to the proposed $(\mathbf{Z}^\star, \mathbf{H}^\star)$ and random $(\mathbf{Z}^{(0)}, \mathbf{H}^{(0)})$ initialization methods for various numbers of basis images p and sparseness degrees σ. For all σ and all p the proposed initialization approach leads to a classifier performance comparable or equal to the situation where the random initialization is chosen, but requires less number of iterations. In the light of the classification error rate, we found that, by applying the random initialization method, the DNMF algorithm reached a minimum error rate of 17.15 % for 3350 iterations (executed in 1740 seconds), while the same minimum error rate was obtained by the proposed initialization method for 722 iterations (executed in 375 seconds). We must notice that the total time necessary to process the steps I - VIII is very small, i.e. equivalent with the time for executing approximately five iteration steps of the basic DNMF algorithm.

TABLE II

CLASSIFICATION ERROR RATE (%) CORRESPONDING TO THE PROPOSED $(\mathbf{Z}^\star, \mathbf{H}^\star)$ AND RANDOM $(\mathbf{Z}^{(0)}, \mathbf{H}^{(0)})$ INITIALIZATION METHOD FOR VARIOUS NUMBERS OF BASIS IMAGES p AND SPARSENESS DEGREES σ.

		Number of basis images p				
		16	25	36	49	64
$Error\ rate$ $(\mathbf{Z}^\star, \mathbf{H}^\star)$	$\sigma = 0.1$	22.86	18.58	17.5	24.58	22.86
	$\sigma = 0.2$	20	17.5	21.43	21.43	30
	$\sigma = 0.3$	18.58	18.58	22	20	17.15
	$\sigma = 0.4$	21.43	18.43	22.86	21.43	20
$Error\ rate$ $(\mathbf{Z}^{(0)}, \mathbf{H}^{(0)})$		21.43	18.58	18.58	17.15	17.15

V. CONCLUSION

The supervised learning DNMF algorithm involves the derivation of factors that satisfy several objectives: their product to approximate the image database, basis images to contain as less as possible redundant information and as much as possible sparse elements and the coefficients to incorporate image class information. These requirements lead to an algorithm that has a slow convergence rate. We proposed an approach to speed up the learning rate by replacing the random initialization of the factors with a new initialization method. This method, along with random initialization, was tested on the case when DNMF was applied to recognize six basic facial expressions.

ACKNOWLEDGMENT

This work has been conducted in conjunction with the "SIMILAR" European Network of Excellence on Multimodal Interfaces of the IST Programme of the European Union (www.similar.cc).

REFERENCES

[1] I. Buciu and I. Pitas, "A new sparse image representation algorithm applied to facial expression recognition," *Proc. IEEE Workshop on Machine Learning for Signal Processing*, pp. 539–548, 2004.

[2] S. Z. Li, X. W. Hou, and H. J. Zhang, "Learning spatially localized, parts-based representation," *Int. Conf. Computer Vision and Pattern Recognition*, pp. 207–212, 2001.

[3] D. D. Lee and H. S. Seung, "Learning the parts of the objects by non-negative matrix factorization," *Nature*, vol. 41, pp. 788–791, 1999.

[4] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *Journal of Machine Learning Research*, vol. 5, pp. 1457–1469, 2004.

[5] T. Kanade, J. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," *Proc. Fourth IEEE Int. Conf. Face and Gesture Recognition*, pp. 46–53, March 2000.