Algorithms for Weighted Non-Negative Matrix Factorization

Vincent Blondel, Ngoc-Diep Ho*and Paul Van Dooren †

Abstract

In this paper we introduce a new type of weighted non-negative matrix factorization and we show that the popular algorithms of Lee and Seung can easily be adapted to also incorporate such a weighting. We then prove that for appropriately chosen weighting matrices, the weighted Euclidean distance function and the weighted Kullback-Leibler divergence function are essentially identical. We finally show that the weighting can be chosen to emphasize parts of the data matrix to be approximated and this is applied successfully to the low rank fitting of a face image database.

Keywords Non-negative matrix factorization, weighting, Euclidean distance, Kullback-Leibler divergence

1 Introduction

Non-Negative Matrix Factorizations (NNMF's) are popular for the problem of approximating non-negative data in a parts-based context. The classical example is that of approximating a given image by a linear combination of other "parts" (i.e. simpler images) with the additional constraint that all images must be represented by a matrix with non-negative elements: each matrix element gives the grey level of an image pixel, and is constrained to be non-negative.

If the simpler images are non-negative matrices of rank one then they can be written as a product $u_i v_i^T$ where both u_i and v_i are non-negative vectors of appropriate length. The approximation problem of a $m \times n$ matrix A by a linear combination of $k < m, n$ such products then reduces to

$$A \approx \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

where the non-negative elements σ_i are the weighting factors of the linear combination. When there is no constraint on the vectors u_i and v_i it is well known that the best rank k approximation in the Euclidean norm is given by the Singular Value Decomposition (SVD), that is $\min \|A - \sum_{i=1}^{k} \sigma_i u_i v_i^T\|$ is achieved for $u_i^T u_j = 0$ and $v_i^T v_j = 0$, $\forall i \neq j$ and $u_i^T u_i = v_i^T v_i = 1$, $\forall i$. Moreover there are good algorithms available to compute the optimal approximation in a computing time that is cubic in the dimensions m and n of the matrix A [4]. But imposing the non-negativity constraint makes the low-rank approximation problem non convex and even NP-hard. We point out here that in many applications this is a crucial property that one wants to preserve. In polynomial time one can still look for a local minimum of a particular error function of the low rank approximation, and iterative

[*] Corresponding author

[†] V. Blondel, N.-D. Ho and P. Van Dooren are with CESAME, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium. E-mail addresses : blondel@inma.ucl.ac.be, ho@inma.ucl.ac.be and vdooren@inma.ucl.ac.be.

algorithms for obtaining such local minima were proposed in [7]. In this paper we show that one can include a weighting matrix in the approximation problem and we show that the popular algorithm of Lee and Seung can then be adapted to also incorporate this weighting. We then prove in Section 4 that for appropriately chosen weighting matrices, the weighted Euclidean distance function and the weighted Kullback-Leibler divergence function become essentially identical. Section 5 describes some experiments of the weighted approximation in order to emphasize parts of the data matrix to be approximated and this is applied successfully to the low rank fitting of a face image database.

2 Non-Negative Matrix Factorization

The *Non-Negative Matrix Factorization* problem imposes non-negativity conditions on the factors (i.e. $A \approx \sum_{i=1}^{k} u_i v_i^T$, $u_i, v_i \geq 0$) and can be stated as follows:

Given a non-negative $(m \times n)$ matrix A, find two non-negative matrices $U(m \times k)$ and $V(k \times n)$ with $k \ll m, n$ that minimize $F(A, UV)$, where $F(A, B)$ is a cost function defining the "nearness" between matrices A and B.

The choice of cost function F of course affects the solution of the minimization problem. One popular choice in matrix problems is the Euclidean Distance (or the *Frobenius norm*)

$$F(A, UV) = \frac{1}{2}\|A - UV\|^2 := \frac{1}{2}\sum_{ij}(A_{ij} - [UV]_{ij})^2 = \frac{1}{2}\sum_{ij}[A - UV]_{ij}^2. \tag{1}$$

Another popular choice in parts based image approximation problems is the *Kullback-Leibler Divergence*

$$F(A, UV) = D(A\|UV) := \sum_{ij}\left(A_{ij}\log\frac{A_{ij}}{[UV]_{ij}} - A_{ij} + [UV]_{ij}\right) = \sum_{ij}\left[A \circ \log\frac{[A]}{[UV]} - A + UV\right]_{ij}, \tag{2}$$

where $log(X)$ is the element-wise logarithm of X, $X \circ Y$ is the Hadamard product (or element by element product) of the matrices X and Y, and $\frac{[X]}{[Y]}$ is the Hadamard division (or element by element division) of the matrices X and Y.

In [7, 8], Lee and Seung propose two algorithms for finding local minimizers of these two cost functions, based on multiplicative updating rules which are simple but quite elegant. We will derive below two similar algorithms for the problem of *Weighted Non-Negative Matrix Factorization (WNNMF)* which minimize the following weighted cost functions: the *Weighted Euclidean Distance*

$$F_W(A, UV) = \frac{1}{2}\|A - UV\|_W^2 := \frac{1}{2}\sum_{ij}[W \circ (A - UV) \circ (A - UV)]_{ij} \tag{3}$$

and the *Weighted Kullback-Leibler Divergence*

$$F_W(A, UV) = D_W(A\|UV) := \sum_{ij}\left[W \circ \left(A \circ \log\frac{[A]}{[UV]} - A + UV\right)\right]_{ij}, \tag{4}$$

where $W = \{W_{ij}\} \geq 0$ is a non-negative weight matrix. Clearly, the two earlier versions are just particular cases of the weighted ones where all the weights are equal to 1.

The problem of Weighted Non-Negative Matrix Factorization was first stated in [10] where the cost function was the Weighted Euclidean Distance (3). Several algorithms including Newton-related methods were used to solve the problem, but they have a high complexity. Simpler algorithms were introduced by Lee and Seung [7, 8] based on a set of multiplicative updating rules but these algorithms were presented for the *unweighted* Euclidean Distance and KL Divergence.

Recently [5], a particular type of weighting was proposed for the divergence cost function, in order to vary the importance of each column of the matrix A in the approximation $UVD \approx AD$, where D is a non-negative diagonal scaling matrix. One can easily see that this non-negative weight matrix is equivalent to a rank-one weighting matrix W in our weighted KL divergence.

An approach that allows to use weighting matrices in a more general context is given in [11], where an Expectation-Maximization algorithm is used in an iterative scheme that produces an *unweighted* low-rank approximation of a *weighted* combination of a previously computed approximation :

$$(U_{k+1}, V_{k+1}) = LowRank(W \circ A + (1 - W) \circ (U_k V_k)). \tag{5}$$

Here there are *no* constraints of non-negativity, but the same idea can also be used to incorporate weights in an algorithm for non-negative matrix factorizations. This implies that one has to solve an unweighted low-rank non-negative approximation at each step of the iteration, and this can become quite inefficient in computing time.

3 The Lee-Seung approach

In this paper we propose variants of the algorithms of Lee and Seung, which allow to solve the weighted cases with arbitrary weighting matrices. Therefore, we first briefly recall in this section the basic ideas of the Lee-Seung approach.

Although the cost functions $\frac{1}{2}\|A - UV\|^2$ and $D(A\|UV)$ are not convex in the two matrix variables U and V (one can show that there are many local minimizers), it has been shown that for a fixed U the cost function is convex in V, and vice-versa. A simple strategy to find a local minimizer is therefore to alternate between minimizations in U and V while keeping the other matrix variable fixed.

The minimization of $F(A, UV)$ under the constraints $U, V \geq 0$, requires the construction of the gradients ∇_U and ∇_V of the cost function $F(A, UV)$. For the Euclidean Distance, these are:

$$\nabla_U \frac{1}{2}\|A - UV\|^2 = -(A - UV)V^T \ , \qquad \nabla_V \frac{1}{2}\|A - UV\|^2 = -U^T(A - UV) \ . \tag{6}$$

The corresponding Kuhn-Tucker optimality conditions for constrained optimization are then:

$$U \circ (AV^T - UVV^T) = 0 \ , \qquad V \circ (U^T A - U^T UV) = 0. \tag{7}$$

For the KL divergence, the gradients are also easy to construct:

$$\nabla_U D(A\|UV) = -\left(\frac{[A]}{[UV]} - \mathbf{1}_{m \times n}\right) V^T \ , \qquad \nabla_V D(A\|UV) = -U^T \left(\frac{[A]}{[UV]} - \mathbf{1}_{m \times n}\right) \tag{8}$$

where $\mathbf{1}_{m \times n}$ is a $m \times n$ matrix with all elements equal to 1. The Kuhn-Tucker conditions are then:

$$U \circ \left[\left(\frac{[A]}{[UV]} - \mathbf{1}_{m \times n}\right) V^T\right] = 0 \ , \qquad V \circ \left[U^T \left(\frac{[A]}{[UV]} - \mathbf{1}_{m \times n}\right)\right] = 0. \tag{9}$$

Lee and Seung [7] use these identities to propose simple updating rules to converge to a local minimum of the cost function. Their convergence results are described in the following two theorems [7, 8]:

Theorem 1. *The Euclidean distance* $\frac{1}{2}\|A-UV\|^2$ *is non-increasing under the updating rules:*

$$V \leftarrow V \circ \frac{[U^T A]}{[U^T UV]}\ , \qquad U \leftarrow U \circ \frac{[AV^T]}{[UVV^T]}. \tag{10}$$

The Euclidean distance $\frac{1}{2}\|A-UV\|^2$ *is invariant under these updates iff* U *and* V *are at a stationary point of the distance.*

Theorem 2. *The divergence* $D(A\|UV)$ *is non-increasing under the updating rules:*

$$V \leftarrow \frac{[V]}{[U^T \mathbf{1}_{m\times n}]} \circ \left(U^T \frac{[A]}{[UV]} \right)\ , \qquad U \leftarrow \frac{[U]}{[\mathbf{1}_{m\times n} V^T]} \circ \left(\frac{[A]}{[UV]} V^T \right), \tag{11}$$

where $\mathbf{1}_{m\times n}$ *is a* $m \times n$ *matrix with all elements equal to* 1*. The divergence* $D(A\|UV)$ *is invariant under these updates iff* U *and* V *are at a stationary point of the divergence.*

The proofs of these theorems can be found in [8], and will be extended for the weighted cases in the next section. The above updating rules are the same as in [7, 8] but are rewritten into matrix form using the Hadamard product and Hadamard division, in order to allow an easy comparison with the updating rules for the weighted cases. Notice also that the non-negativity constraints on the matrices U and V are automatically satisfied by these updating rules if the starting matrices U_0 and V_0 are non-negative.

4 Weighted Non-Negative Matrix Factorization

In this section we extend the results of Lee and Seung to the weighted case. We treat the different cases separately.

4.1 The weighted Euclidean distance

The following theorem generalizes Theorem 1 to the weighted case:

Theorem 3. *The weighted Euclidean distance* $\frac{1}{2}\|A-UV\|_W^2$ *is non-increasing under the updating rules:*

$$V \leftarrow V \circ \frac{[U^T(W\circ A)]}{[U^T(W\circ(UV))]}\ , \qquad U \leftarrow U \circ \frac{[(W\circ A)V^T]}{[(W\circ(UV))V^T]}. \tag{12}$$

The weighted Euclidean distance $\frac{1}{2}\|A-UV\|_W^2$ *is invariant under these updates iff* U *and* V *are at a stationary point of the distance.*

Let $D_x = diag(x)$ denote a diagonal matrix with the elements of the vector x as diagonal entries. Then the following lemma will help constructing the updating equations in the above theorem.

Lemma 1. *Let* A *be a symmetric non-negative matrix and* v *be a positive vector, then the matrix* $\hat{A} = diag\left(\frac{[Av]}{[v]}\right) - A$ *is positive semi-definite.*

Proof. It is easy to see that $diag\left(\frac{[Av]}{[v]}\right) = D_v^{-1} D_{Av}$. The scaled version $\hat{A}_s := D_v \hat{A} D_v$ of $\hat{A}$ satisfies $\hat{A}_s = D_{Av}D_v - D_v A D_v$ and is a *diagonally dominant* matrix since $\hat{A}_s \mathbf{1}_m = (Av)\circ v - v \circ (Av) = 0$ and its off-diagonal elements are negative. Therefore, the matrix $\hat{A}_s$ is positive semi-definite, and so is $\hat{A}$. □

We can now use this lemma to prove the above theorem.

Proof. (Theorem 3) We only treat the updating rule for V since that of U can be proven in a similar fashion. First, we point out that the cost $F(A, UV)$ splits in n independent problems related to each column of the error matrix. We can therefore consider the partial cost function for a single column of A, V and W, which we denote by a, v and w, respectively:

$$F(v) = F_w(a, Uv) = \frac{1}{2}\sum_i (w_i(a_i - [Uv]_i)^2) = \frac{1}{2}(a - Uv)^T D_w (a - Uv) \tag{13}$$

where $D_w = diag(w)$. Let v^k be the current approximation of the minimizer of $F(v)$ then one can rewrite $F(v)$ as the following quadratic form:

$$F(v) = F(v^k) + (v - v^k)^T \nabla_v F(v^k) + \frac{1}{2}(v - v^k)^T U^T D_w U (v - v^k) \tag{14}$$

where $\nabla_v F(v^k)$ is explicitly given by

$$\nabla_v F(v^k) = -U^T D_w (a - Uv^k). \tag{15}$$

Next, we approximate $F(v)$ by a simpler quadratic model:

$$G(v, v^k) = F(v^k) + (v - v^k)^T \nabla_v F(v^k) + \frac{1}{2}(v - v^k)^T D(v^k)(v - v^k) \tag{16}$$

where $G(v^k, v^k) = F(v^k)$ and $D(v^k)$ is a diagonal matrix chosen to make $D(v^k) - U^T D_w U$ positive semi-definite implying that $G(v, v^k) - F(v) \geq 0, \forall v$. The choice for $D(v^k)$ is similar to that proposed by Lee and Seung:

$$D(v^k) = diag\left(\frac{[U^T D_w U v^k]}{[v^k]}\right). \tag{17}$$

Lemma 1 assures the positive semi-definiteness of $D(v^k) - U^T D_w U$. As a result, we have

$$F(v^k) = G(v^k, v^k) \geq \min_v G(v, v^k) = G(v^{k+1}, v^k) \geq F(v^{k+1}) \tag{18}$$

where v^{k+1} is found by solving $\frac{\partial G(v, v^k)}{\partial v} = 0$:

$$\begin{aligned} v^{k+1} &= v^k - D(v^k)^{-1} \nabla F(v^k) && (19)\\ &= v^k + diag\left(\frac{[v^k]}{[U^T D_w U v^k]}\right) U^T D_w (a - Uv^k) && (20)\\ &= v^k + v^k \circ \frac{[U^T D_w (a - Uv^k)]}{[U^T D_w U v^k]} && (21)\\ &= v^k \circ \frac{[U^T D_w a]}{[U^T D_w U v^k]} && (22)\\ &= v^k \circ \frac{[U^T (w \circ a)]}{[U^T (w \circ (Uv^k))]}. && (23) \end{aligned}$$

Putting together the updating rules for all the columns of V yields the desired result for the whole matrix V in (12). The relation (18) shows that the weighted Euclidean distance is non increasing under the updating rule for V, and (19) show that $v^{k+1} = v^k$ if and only if $v^k \circ \nabla F(v^k) = 0$. □

4.2 The weighted KL divergence

The following theorem generalizes Theorem 2 to the weighted case:

Theorem 4. *The weighted divergence $D_W(A\|UV)$ is non-increasing under the updating rules:*

$$V \leftarrow \frac{[V]}{[U^T W]} \circ \left(U^T \frac{[W \circ A]}{[UV]} \right), \qquad U \leftarrow \frac{[U]}{[WV^T]} \circ \left(\frac{[W \circ A]}{[UV]} V^T \right). \tag{24}$$

The weighted divergence $D_W(A\|UV)$ is invariant under these updates iff U and V are at a stationary point of the divergence.

Proof. Again, we prove the theorem only for V and we also split the divergence into partial divergences corresponding to one column of V, W and A, denoted by v, w and a.

$$F(v) = D_w(a\|Uv) = \sum_i w_i \left(a_i \log a_i - a_i + \sum_j U_{ij} v_j - a_i \log \sum_j U_{ij} v_j \right). \tag{25}$$

This partial divergence is approximated by the following auxiliary function:

$$G(v, v^k) = \sum_i w_i \left(a_i \log a_i - a_i + \sum_j U_{ij} v_j - a_i \sum_j \frac{U_{ij} v_j^k}{\sum_l U_{il} v_l^k} \left(\log U_{ij} v_j - \log \frac{U_{ij} v_j^k}{\sum_l U_{il} v_l^k} \right) \right). \tag{26}$$

Because of the convexity of the function $-log(x)$ and since $\sum_j \frac{U_{ij} v_j^k}{\sum_l U_{il} v_l^k} = 1$, we have that $G(v, v^k) \geq F(v), \forall v$. Moreover $G(v^k, v^k) = F(v^k)$, so we obtain:

$$F(v^k) = G(v^k, v^k) \geq \min_v G(v, v^k) = G(v^{k+1}, v^k) \geq F(v^{k+1}) \tag{27}$$

To obtain the updating rule, it is sufficient to construct the minimizer of G with respect to v, given by:

$$\frac{\partial G(v, v^k)}{\partial v_j} = \sum_i w_i U_{ij} - \frac{v_j^k}{v_j} \sum_i w_i a_i \frac{U_{ij}}{\sum_l U_{il} v_l^k} = 0. \tag{28}$$

Then the minimizer of $G(v, v^k)$ is chosen as the next value of v:

$$v^{k+1} = \frac{[v^k]}{[U^T w]} \circ \left(U^T \frac{[a \circ w]}{[Uv^k]} \right). \tag{29}$$

Putting together the updating rules for all the columns of V gives the desired updating rule for the whole matrix V as in (24). The relation (27) shows that the weighted divergence is non increasing under the updating rule for V. Using (29) and the fact that

$$\nabla F(v^k) = U^T w - U^T \frac{[a \circ w]}{[Uv^k]} \tag{30}$$

we can easily see that that $v^{k+1} = v^k$ if and only if $v^k \circ \nabla F(v^k) = 0$. □

4.3 Linking the two cost functions

One can rewrite the updating rule for V in the weighted KL divergence case as follows:

$$V \leftarrow \frac{[V]}{[U^TW]} \circ \left(U^T \frac{[W \circ A]}{[UV]}\right) = V \circ \left(\frac{\left[U^T \frac{[W\circ A]}{[UV]}\right]}{\left[U^T \frac{[W\circ (UV)]}{[UV]}\right]} \right) = V \circ \left(\frac{[U^T(W_{UV} \circ A)]}{[U^T(W_{UV} \circ (UV))]} \right), \tag{31}$$

where $W_{UV} = \frac{[W]}{[UV]}$. This shows that each update in the weighted KL divergence is equivalent to an update in the weighted Euclidean distance with the weight matrix W_{UV}. This is an adaptive weighting since the weights change after each update. And at the stationary point of this minimization, V and U converge to the minimizer of the weighted Euclidean distance for which the weight matrix is exactly W_{UV}.

Conversely, one can see that each update in the weighted Euclidean distance with the weight matrix W is equivalent to an update in the weighted KL divergence with the weight matrix $W_{UV} = W \circ (UV)$. And again, at the stationary point of this minimization, U and V converge to the minimizer of the weighted KL divergence for which the weight matrix is exactly W_{UV}.

Moreover, if we look at the optimality conditions in the two cases

$$V \circ (U^T(W_1 \circ (UV - A))) = 0 \tag{32}$$

and

$$V \circ (U^T(W_2 \circ (\mathbf{1}_{m\times n} - \frac{[A]}{[UV]}))) = 0, \tag{33}$$

it is easy to see that if $W_1 = \frac{[W_2]}{[UV]}$, these two conditions are identical.

We summarize all the updating rules and the link between the two minimizations in the following table. In the unweighted case, the matrix $\mathbf{1}_{m\times n}$ is included to make it easier to compare it with the matrices W_1 and W_2 of the weighted case. With our updating rules for weighted case, we have thus shown that even though the two cost functions are very different, their minimizations are closely related.

Table 1: Summary of algorithms for Weighted Non-Negative Matrix

	Euclidean Distance (ED)	KL Divergence (KLD)
NNMF	$V \leftarrow V \circ \frac{\left[U^T(\mathbf{1}_{m\times n}\circ A)\right]}{[U^T(\mathbf{1}_{m\times n}\circ (UV))]}$	$V \leftarrow \frac{[V]}{[U^T\mathbf{1}_{m\times n}]} \circ \left(U^T \frac{[\mathbf{1}_{m\times n}\circ A]}{[UV]}\right)$
WNNMF	$V \leftarrow V \circ \frac{\left[U^T(\mathbf{W_1}\circ A)\right]}{[U^T(\mathbf{W_1}\circ (UV))]}$	$V \leftarrow \frac{[V]}{[U^T\mathbf{W_2}]} \circ \left(U^T \frac{[\mathbf{W_2}\circ A]}{[UV]}\right)$
ED ⇔ KLD	$\mathbf{W_1} = \frac{[\mathbf{W_2}]}{[UV]}$	

4.4 Other weighted NNMF methods

Here we consider other NNMF methods found in the literature and we briefly show how to incorporate weighting matrices in these methods as well.

The *Non-Negative Matrix Factorization with Sparseness Constraint* of [6] imposes sparseness constraints on the matrices U and V. The algorithm uses two separate steps to achieve this: a gradient-descent step and and a sparseness control step. Clearly weights can be easily added in the gradient-descent step by setting the cost function to be the weighted Euclidean distance instead of the unweighted one. The sparseness control step is kept unchanged.

A second method is the *Local Non-Negative Matrix Factorization* [9]. It uses a modified KL divergence cost by adding two more terms in order to force the sparseness of the columns of U. This results in updating rules consisting of three steps:

$$\begin{aligned} V &\leftarrow \sqrt{V \circ \left(U^T \frac{[A]}{[UV]} \right)}, \qquad (34)\\ U &\leftarrow \frac{[U]}{[\mathbf{1}_{m\times n} V^T]} \circ \left(\frac{[A]}{[UV]} V^T \right), \qquad (35)\\ U &\leftarrow \frac{[U]}{[\mathbf{1}_{m\times m} U]}. \qquad (36) \end{aligned}$$

It is proven in [9] that with this updating scheme the algorithm converges to a local minima of the following modified cost function:

$$D_W(A\|UV) = \sum_{ij} \left[A \circ \log \frac{[A]}{[UV]} - A + UV \right]_{ij} + \alpha \sum_{ij} [U^T U]_{ij} - \beta \sum_i [VV^T]_{ii}, \qquad (37)$$

where α and β are appropriately chosen non-negative constants.

We can clearly add weights to this algorithm by using the weighted version of the KL divergence. The original convergence proof in [9] can be modified to yield the following updating rules:

$$\begin{aligned} V &\leftarrow \sqrt{\frac{[V]}{[U^T \mathbf{W}]} \circ \left(U^T \frac{[A \circ \mathbf{W}]}{[UV]} \right)}, \qquad (38)\\ U &\leftarrow \frac{[U]}{[\mathbf{W} V^T]} \circ \left(\frac{[A \circ \mathbf{W}]}{[UV]} V^T \right), \qquad (39)\\ U &\leftarrow \frac{[U]}{[\mathbf{1}_{m\times m} U]}. \qquad (40) \end{aligned}$$

where W is the non-negative weight matrix of the following weighted cost function:

$$D_W(A\|UV) = \sum_{ij} \left[W \circ \left(A \circ \log \frac{[A]}{[UV]} - A + UV \right) \right]_{ij} + \alpha \sum_{ij} [U^T U]_{ij} - \beta \sum_i [VV^T]_{ii}, \qquad (41)$$

where α and β are appropriately chosen non-negative constants. Both unweighted and weighted versions of the Local NNMF produce sparse bases, i.e. columns of U. In addition, the weighted one puts more emphasis on elements with higher weight. This effect will be illustrated in numerical experiments in the next section.

A third method is the *Fisher Non-Negative Matrix Factorization* [12] that imposes Fisher constraints on the NNMF algorithms by using a priori information of class relation between data. This method is very similar to the Local NNMF and weights can also be added in a similar fashion.

5 Numerical experiments

In [7] Lee and Seung argued that there is a link between human perception and non-negative data representation. The intuition behind this is that perception is based on a representation that is additive and tends to expose parts of the data. Since then, many researchers have tried to use non-negative representations of data – such as NNMF – in many application areas.

One of the major application of NNMF is the representation of human faces. In this section, we show the results of two numerical experiments on human faces. These experiments also illustrate the effect of weights on the obtained approximation.

5.1 Experiment settings

The experiments use the Cambridge ORL face database as the input data. This contains 400 images of 40 persons (10 images per person). The size of each image is 112×92 with 256 gray levels per pixel representing a front view of the face of a person. As was also done in earlier papers, we chose here to show the images in negative because visibility is better. Pixels with higher intensity are therefore darker. Ten randomly chosen images are shown in the first row of Figure 1.

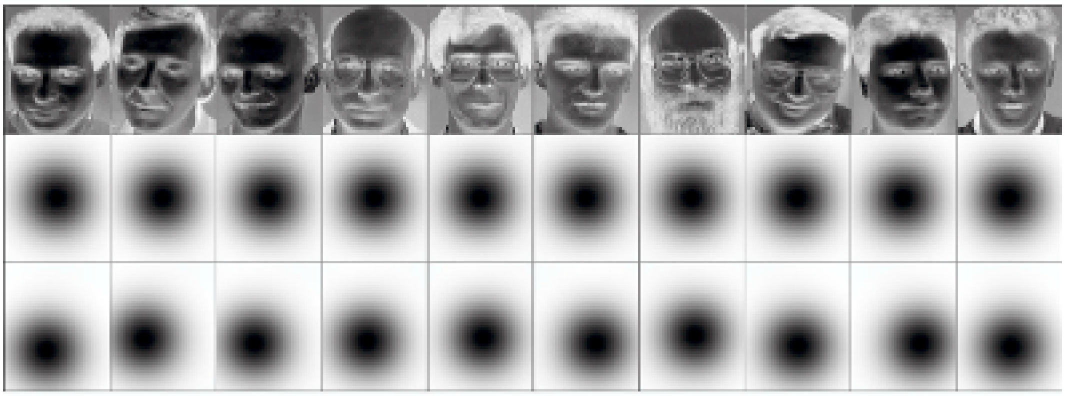

Figure 1: Original faces (first row), their image-centered weights W_2 (second row) and their face-centered weights W_3 (last row)

The images are then transformed into 400 “face vectors” in $\mathbb{R}^{10304}$ ($112 \times 92 = 10304$) to form the data matrix A of size 10304×400. We used three weights matrices:

- **Uniform weight** W_1: a matrix with all elements equal to 1 (i.e. the unweighted case)
- **Image-centered weight** W_2: a non-negative matrix whose columns are identical, i.e. same weights are applied to every images. For each image, the weight of each pixel is given by $w_d = e^{-\frac{d^2}{\sigma^2}}$ where $\sigma = 30$ and d is the distance of the pixel to the *center of the image* $(56.5, 46.5)$.

This weight matrix has rank one. Ten columns of this matrix are shown in the second row of Figure 1

- **Face-centered weight** W_3: a non-negative matrix whose columns are *not* identical, i.e. different weights are applied to different images. For each image, the weight of each pixel is given by $w_d = e^{-\frac{d^2}{\sigma^2}}$ where $\sigma = 30$ and d is the distance of the pixel to the *center of the face* in that image. The rank of this matrix is not restricted to one. Ten columns of this matrix are shown in the last row of Figure 1.

Next, the matrix A is approximated by non-negative matrices U and V. The rank chosen for the factorization is 49, the matrices U and V will thus be of dimension 10304×49 and 49×400 respectively. Each column of U is considered as a non-negative basis vector.

5.2 NNMF versus Weighted NNMF

In this experiment, all three weight matrices W_1, W_2 and W_3 are used in a NNMF based on the weighted KL divergence. For each weight matrix, 49 non-negative bases, i.e. columns of U, are calculated and shown in Figure 2.

Each image in the database can be reconstructed as a weighted sum of these non-negative bases with non-negative weights determined by the corresponding column of V. In Figure 3, ten selected images are compared with the reconstructed images from the three experiments. The pixel-wise KL divergence averages from the three experiments are shown in Figure 4.

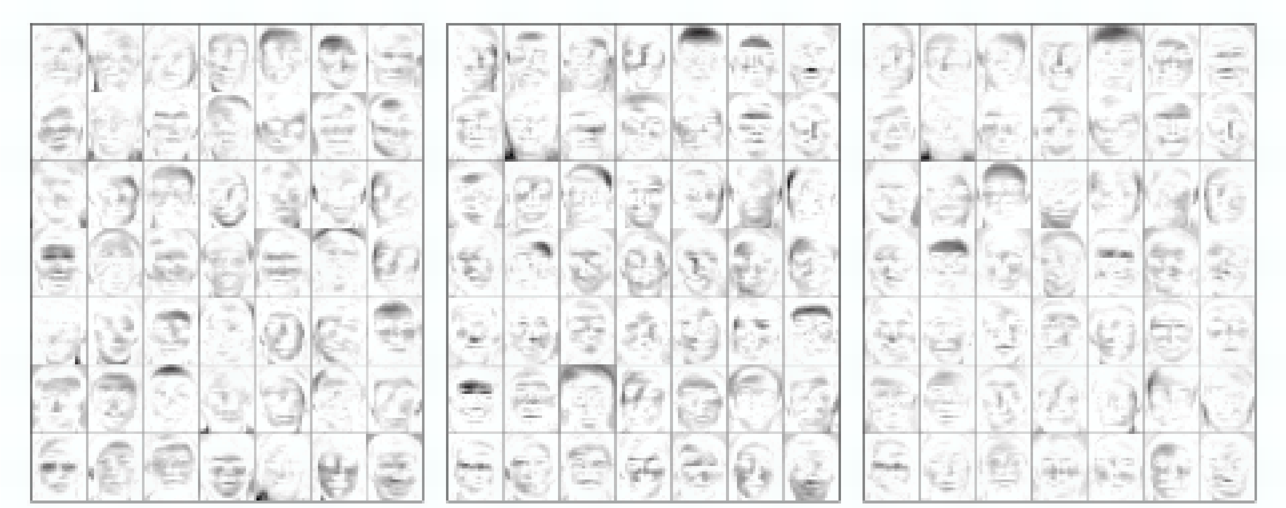

Figure 2: WNNMF Bases when using: uniform weights (left), image-centered weights (middle) and face-centered weights (right)

It can be seen from the results that more important pixels (i.e. those with higher weight, at the center of images or at the center of faces in our example) are better reconstructed than less important ones. This improvement can be seen in both reconstructed images and the pixel-wise average divergence of all the images. In figure 4, the darker colors correspond to larger errors, which means that the algorithm pays more attention to the center of the images (or to the center of the faces) and that the details at the center are privileged in the approximation. For the face-centered case, the improvement on the pixel-wise average errors are less visible due to the fact that the centers of faces are not fixed.

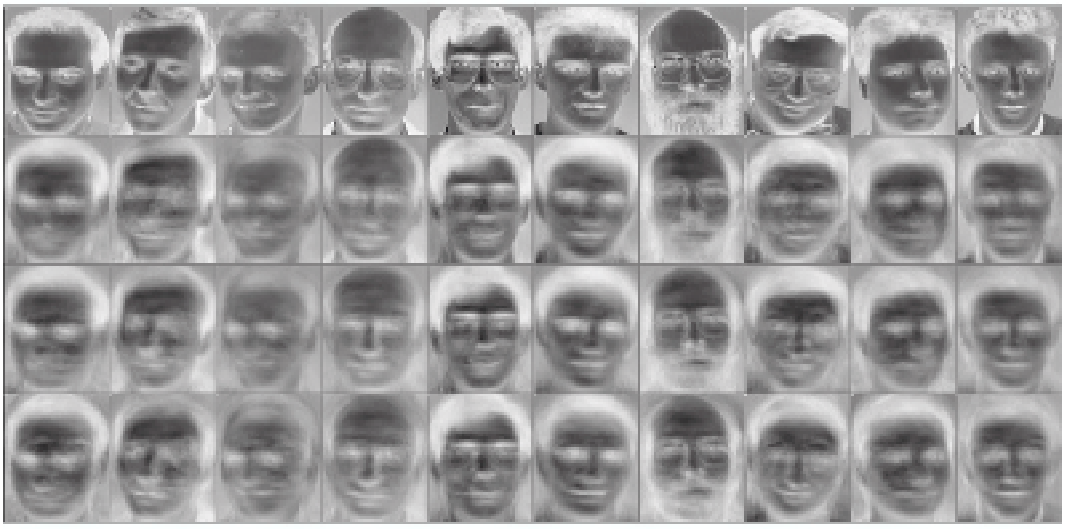

Figure 3: Original and reconstructed faces: original (top), using uniform weights (second line), using image-centered weights (third line) and using face-centered weights (bottom)

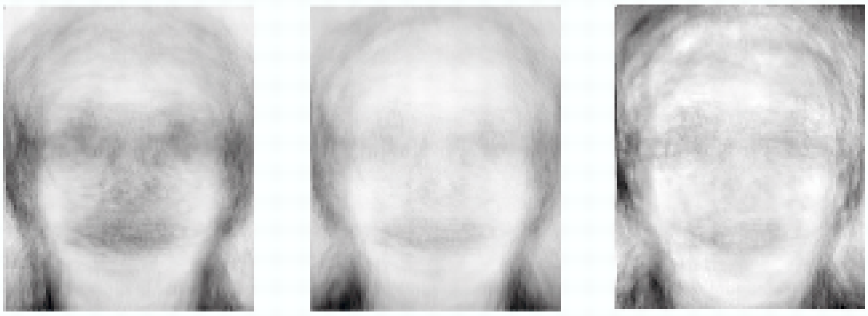

Figure 4: Pixel-wise average divergence: unweighted (left), image-centered (middle) and face-centered (right)

However, more details can be seen on the reconstructed faces when face-centered weights are applied, especially when the center of a face is further away from the center of the image.

The results also show that our algorithms can deal with weight matrices without rank restriction. And weights can be adapted to each data vector in order to yield better approximations.

5.3 Local NNMF versus Weighted Local NNMF

This second experiment shows the effect of adding weights into the LNNMF. Figure 5 shows two sets of 49 non-negative bases obtained by LNNMF with uniform weight (left) and with face-centered weight W_3 (right).

The Local NNMF is often used to extract local and independent features on faces. As weights are more centered, more features at the center of faces are retained. This allows us to tune the Local NNMF algorithm to more relevant parts to to give more useful information about the data.

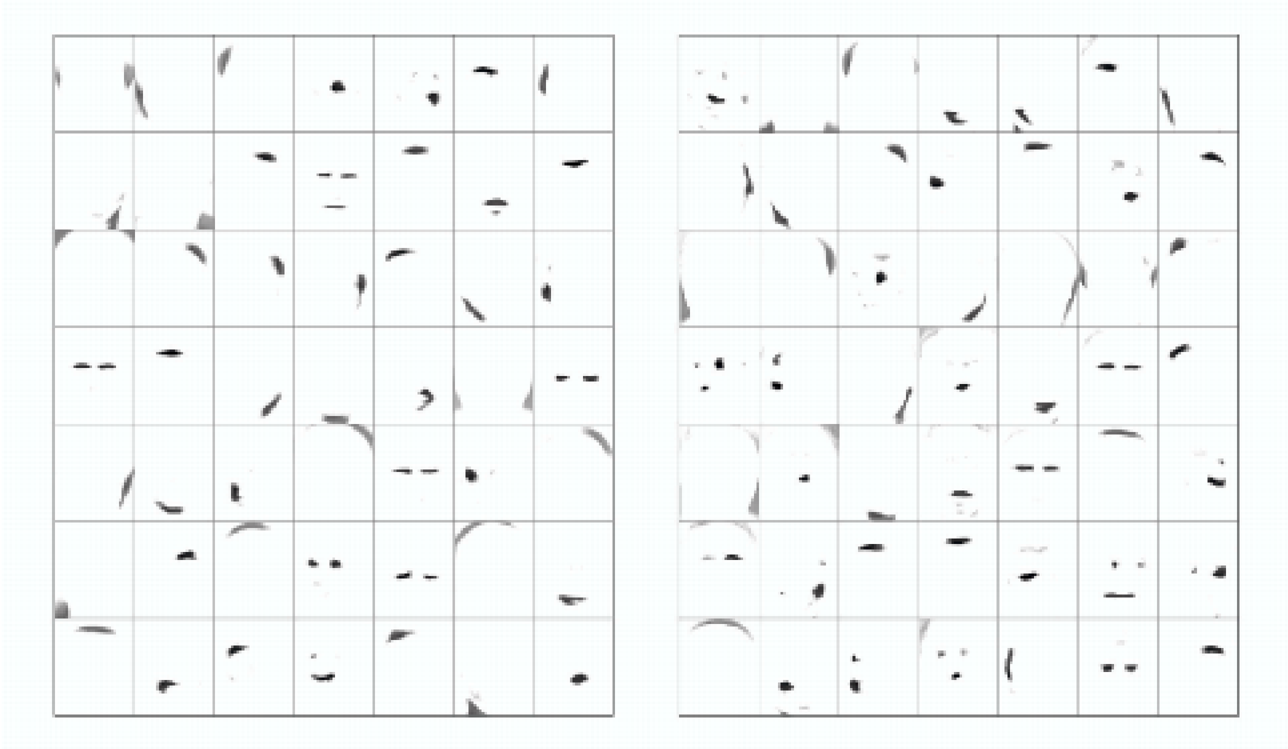

Figure 5: LNNMF Bases: unweighted (left) and face-centered (right)

6 Conclusion

In this paper, we extended several Non-Negative Matrix Factorization (NNMF) algorithms in order to incorporate weighting matrices and we derived weighted iterative schemes for which we proved convergence results that are similar to the unweighted counterparts. We showed that the inclusion of weights allowed us to link the different algorithms in a certain manner and we showed that weighting yields an important flexibility allowing to emphasize better certain features in image approximation problems. This was illustrated in the approximation of faces extracted from a database that is often used as benchmark.

Acknowledgements

This paper presents research supported by the Concerted Research Action (ARC) "Large Graphs and Networks" of the French Community of Belgium, and by the Belgian Programme on Inter-university Poles of Attraction, initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture. The scientific responsibility rests with the authors. Ngoc-Diep Ho is a FRIA fellow.

References

[1] M. Catral, L. Han, M. Neumann and R. Plemmons, *On Reduced Rank Nonnegative Matrix Factorizations for Symmetric Matrices, preprint, Lin. Alg. and Applications*, 2004.

[2] M. Chu, *Structured Low Rank Approximation - Lecture V: Nonnegative Matrix Factorization, XXII School of Computational Mathematics, Num. Lin. Alge. and Its Applications*, 2004.

[3] L. Finesso, P. Spreij *Approximate Nonnegative Matrix Factorization via Alternating Minimization*, 2004.

[4] G. Golub, C. Van Loan. *Matrix Computations. Patter Recognition Letters 24, 2447-2454*, 2003.

[5] D. Guillamet, J. Vitri, B. Schiele. *Introducing a weighted non-negative matrix factorization for image classification. Patter Recognition Letters 24, 2447-2454*, 2003.

[6] P. Hoyer, *Non-negative matrix factorization with sparseness constraints. to find source*, 2004.

[7] D. D. Lee and H. S. Seung, *Learning the parts of objects by non-negative matrix factorization. Nature 401, 788-791*, 1999.

[8] D. D. Lee and H. S. Seung, *Algorithms for non-negative matrix factorization. In Advances in Neural Information Processing 13*, 2001.

[9] S. Z. Li, X. W. Hou, H. J. Zhang and Q. S. Cheng, *Learning Spatially Localized, Parts-based Representation. IEEE*, 2001.

[10] P. Paatero, *Least Squares Formulation of Robust, Non-Negative Factor Analysis*, *Chemom. Intell. Lab. Syst. 37:23-35*, 1997.

[11] N. Srebro and T. Jaakkola, *Weighted low rank approximation. In 20th International Conference on Machine Learning*, 2003.

[12] Y. Wang, Y. Jia, C. Hu and M. Turk, *Fisher Non-Negative Matrix Factorization For Learning Local Features*, *Asian Conference on Computer Vision, Korea, January 27-30*, 2004.