Appendix B

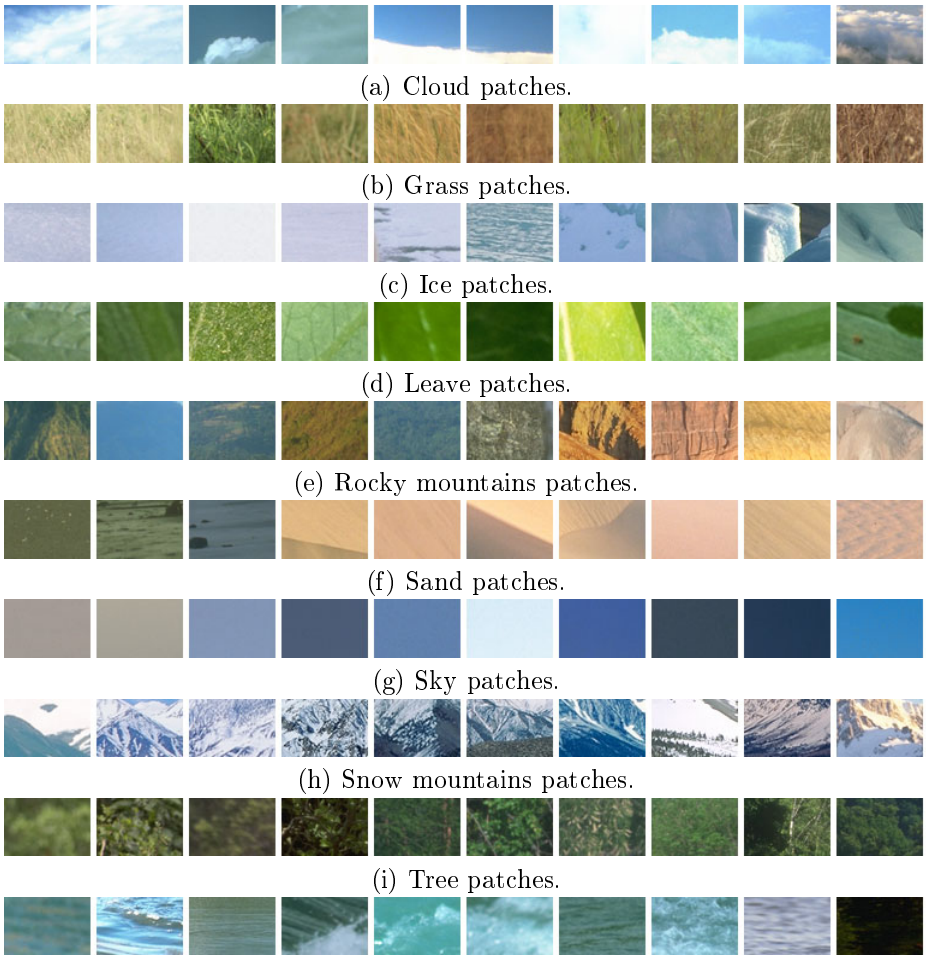
Experiments using WNMF

Two different experiments are shown in this appendix. The first one evaluates the performance of a Weighted version of NMF with respect the classical approach in the context of object recognition of natural patch classes. A second experiment evaluates the reconstruction error of WNMF and NMF in order to evaluate which technique performs faster in terms of the number of iterations.

B.1 Outperformance of WNMF with respect to NMF

In this experiment, input vector samples consist of 512-dimensional color histograms made from the three 8-bin histograms that correspond to each color spectrum. We use the RGB color space. These histograms were extracted from different representative regions of 948 images belonging to the corel database [33] (see appendix C.2 where we show some images of the corel database). We automatically divided each image into 10×10 local regions of $3456(48 \times 72)$ RGB pixels. The regions belong to ten different classes corresponding to clouds, grass, ice, leaves, rocky mountains, sand, sky, snow mountains, trees and water. Some of these representative regions, for all ten classes, are illustrated in figure (B.1). In this figure we can already predict the high confusion between classes derived from only using color signature. Grass, trees and leaves can all present similar green patterns; sky, ice and water have strong blue distributions; rock and sand are also similar, at least visually. In addition, there are portions of the clouds which are only sky, or portions of the snow mountains which are mainly rock. Of course, additional information such as textural information would aid discriminability.

From each class, we randomly extracted 1000 local color histograms for building the training set and the same for the testing set. After this, we learned a NMF and a WNMF model for each class. Random initial conditions for both techniques are the same, so that, we can directly compare the obtained results. When we explained WNMF and we presented an initial visual comparison between NMF and WNMF, we noted that WNMF starts to perfom better than NMF at a given dimensionality of the subspace created ($r = 45$). Now, we have 10 different classes each of them represented using a NMF and a WNMF model, so that, it is expected that the performance of



(a) Cloud patches.

(b) Grass patches.

(c) Ice patches.

(d) Leave patches.

(e) Rocky mountains patches.

(f) Sand patches.

(g) Sky patches.

(h) Snow mountains patches.

(i) Tree patches.

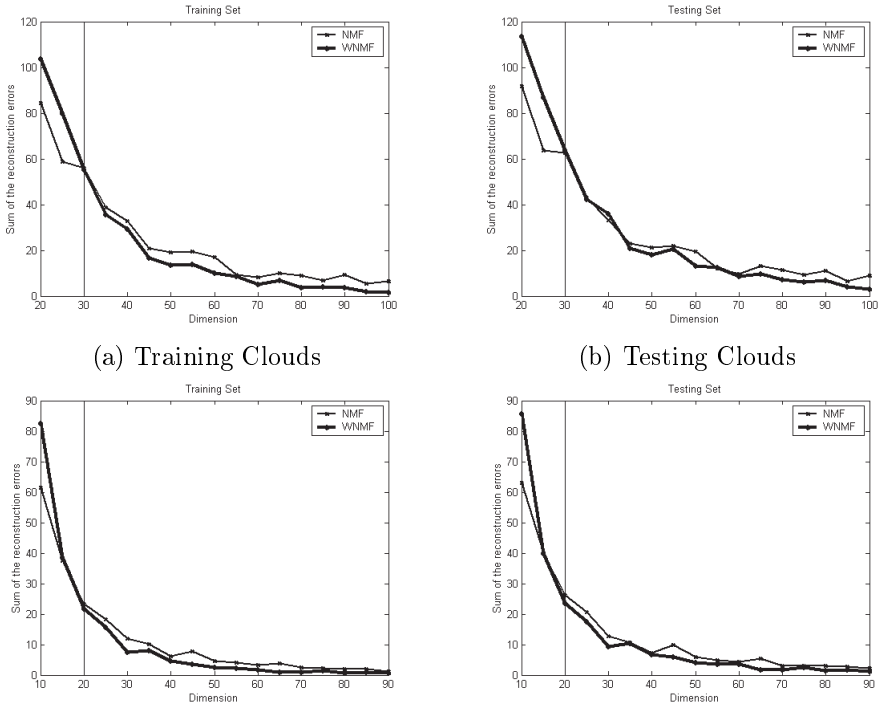
(j) Water patches.

Figure B.1: Sample image regions and their corresponding class labels (cloud, grass, ice, leave, rocky mountains, sand, sky, snow mountains, tree and water) extracted from the corel image database.

WNMF starts to be better than the one of NMF at a different subspace dimension.

Using the reconstruction error of the NMF and WNMF models (see expression (3.7)) as a possible way to compare both algorithms, we present the reconstruction error of both techniques against the dimensionality of the positive subspace created (r). Figures (B.2,B.3,B.4, B.5) show the reconstruction error of our 10 data classes according to the dimensionality of the subspace. Also, we show results for the training dataset (vectors used to create our NMF/WNMF models) and the testing dataset (new vector instances of each class that we do not use for learning).

Figures (B.2,B.3,B.4,B.5) provide a graphical result of the reconstruction error of all 10 classes from $r = 20$ to $r = 100$ dimensions of the subspace created by NMF



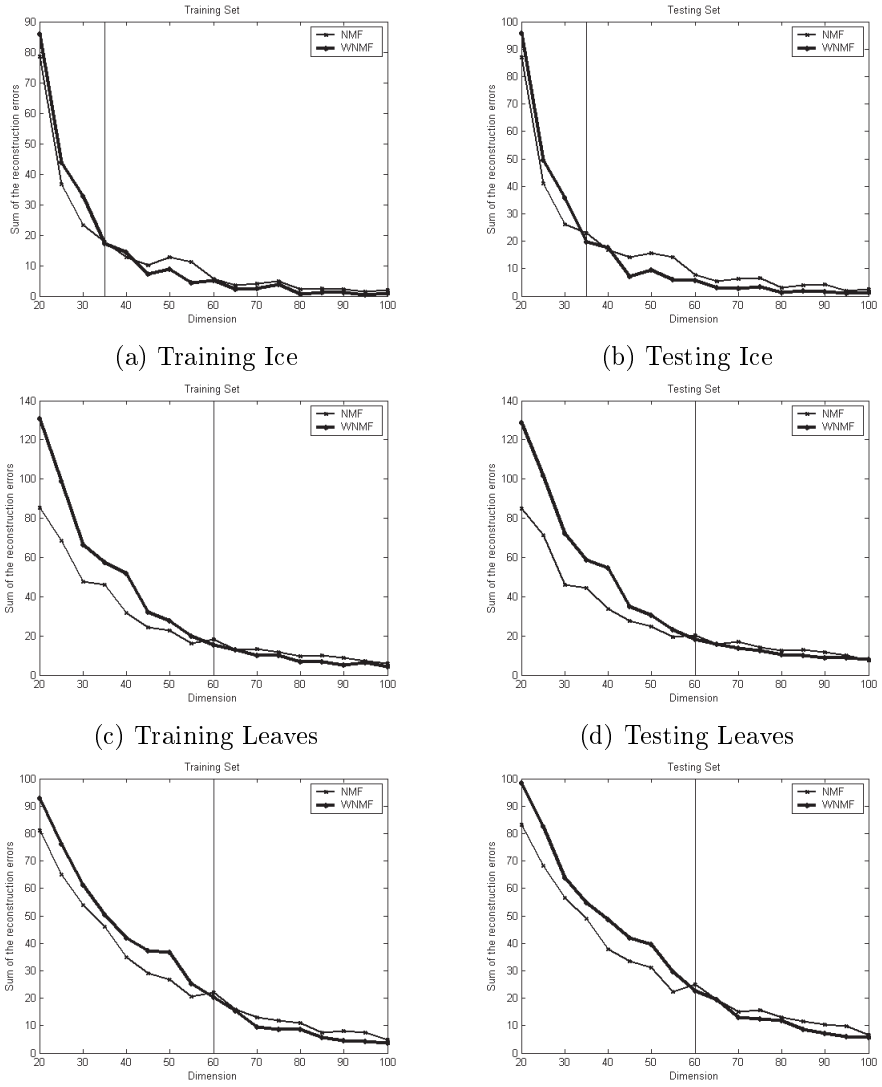
(a) Training Clouds (b) Testing Clouds

(c) Training Grass (d) Testing Grass

Figure B.2: Evolution of the reconstruction error of NMF (thin line) and WNMF (thick line) techniques through dimension 20 to 100 when we consider 2 data classes (Clouds, Grass). Both training and testing results are presented independently. In the specific case of the Grass class, dimension interval goes from 10 to 90.

and WNMF. NMF and WNMF have been trained starting from the same random conditions, so that, the reconstruction error is comparable. We divided our experiments into two datasets (the training and the testing datasets) and we show the reconstruction error because we want to know if the learned model can be generalized to new unseen data. As expected, when we increase the dimensionality (r) of the NMF/WNMF subspaces, reconstruction error of both sets drops because both models are able to represent more information. Each graphic of figures (B.2,B.3,B.4,B.5) show a vertical line that corresponds to the specific moment where WNMF starts to perform better than NMF. As seen in these figures, when a data class is complex, we can observe some indecision when determinig a threshold dimension. One clear example of this indecision is the sky class because WNMF is better than NMF when using a $r = 45$ dimensional subspace. However, this scenario changes at dimension $r = 60$ where NMF is better than WNMF. This is due to the complexity of the class and possibly due to the initial random conditions of NMF/WNMF techniques.

Again, we have chosen two particular situations where our data classes are better represented by NMF and vice versa. We chose to use dimension $r = 40$ and



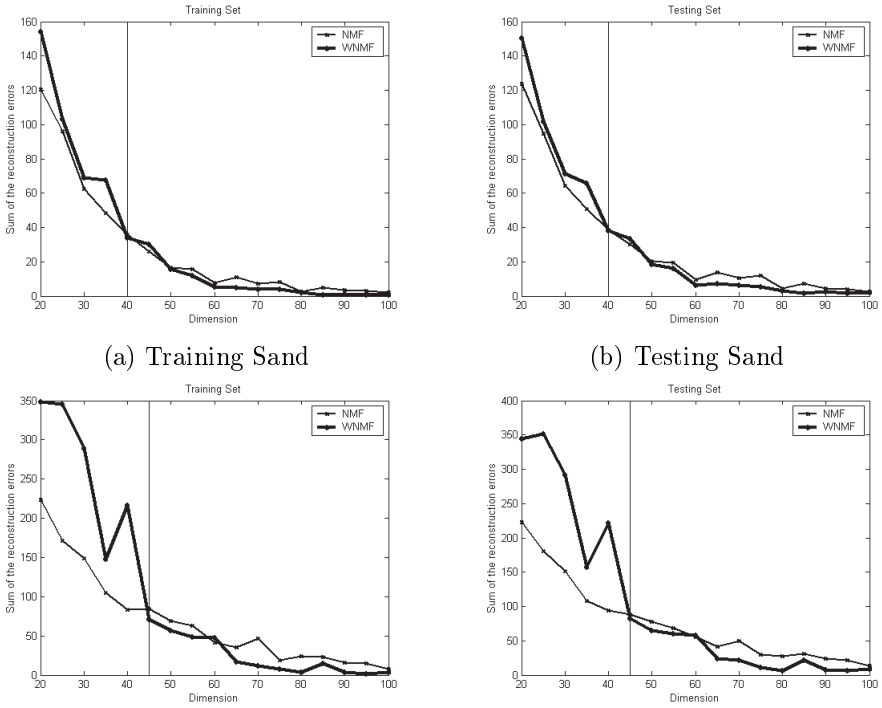
(a) Training Ice (b) Testing Ice

(c) Training Leaves (d) Testing Leaves

(e) Training Rocky (f) Testing Rocky

Figure B.3: Evolution of the reconstruction error of NMF (thin line) and WNMF (thick line) techniques through dimension 20 to 100 when we consider 3 data classes (Ice, Leaves and Rocky Mountains).

data classes Grass and Leaves. Figure (B.6) shows the obtained $\mathbf{W}$ bases of both NMF/WNMF techniques. As noted in figure (B.2), data vectors corresponding to the grass class are better represented using WNMF in the selected dimension of $r = 40$ but data vectors of the leaves class are better represented using NMF using this specific dimension of $r = 40$. This means that we can analyze the bases presented

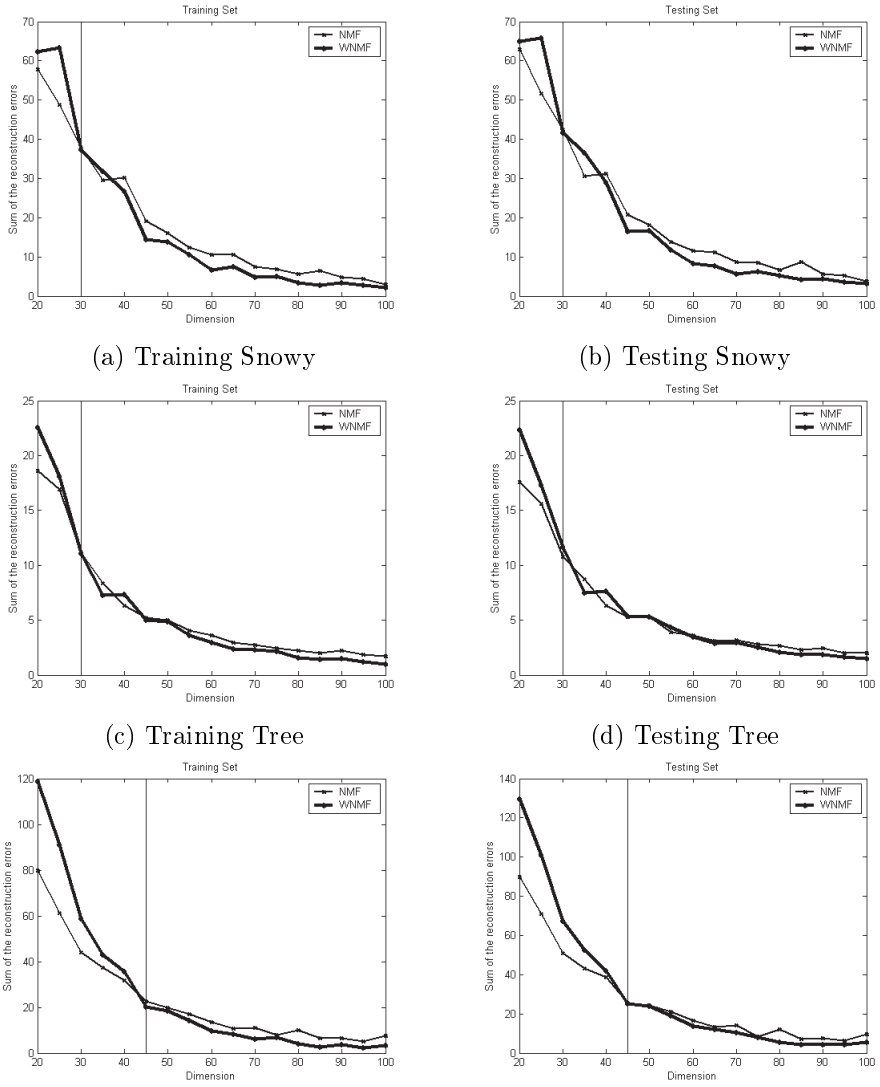


(a) Training Sand (b) Testing Sand

(c) Training Sky (d) Testing Sky

Figure B.4: Evolution of the reconstruction error of NMF (thin line) and WNMF (thick line) techniques through dimension 20 to 100 when we consider 2 data classes (Sand, Sky). Both training and testing results are presented independently.

in figure (B.6) using some measure function in order to evaluate the redundancy of bases **W**. A rapid visual analysis of figure (B.6) should be enough to realize that figure (B.6.b) contains less redundant bases than figure (B.6.a) and that figure (B.6.c) also contains more redundant bases with respect to figure (B.6.d). Redundancy evaluation of **W** bases is a difficult task and we can not guarantee that our perception is able to distinguish when one method performs better than the other. So that, in order to evaluate the redundancy of a given set of NMF/WNMF **W** bases, we extract a similarity matrix from this set of bases. Since the elements of matrix **W** are also color histograms we can evaluate the similarity between two data vectors of matrix **W** using some of the well-known histogram metrics (see section 2.1.3). We use the histogram intersection measure between two bases of matrix **W** in order to evaluate how redundant can be our bases. Why this measure? Since the obtained bases of matrix **W** are also color histograms, we can also evaluate whether two bases contain the same color using the histogram intersection. It can be the case to have one color histogram with both white and green tonalities, and another color histogram with both white and red tonalities. Using the histogram intersection, we will be able to know that both color histograms contain a white tonality. Thus, using the histogram



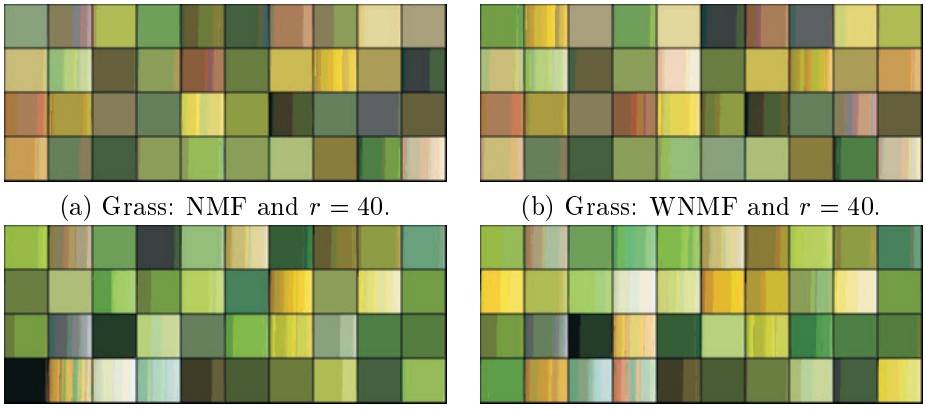
(a) Training Snowy (b) Testing Snowy

(c) Training Tree (d) Testing Tree

(e) Training Water (f) Testing Water

Figure B.5: Evolution of the reconstruction error of NMF (thin line) and WNMF (thick line) techniques through dimension 20 to 100 when we consider 3 data classes (Snow Mountains, Tree and Water). Both training and testing results are presented independently.

intersection is equivalent to use a perceptual measure of similarity because we understand what we are measuring and what are the results that we obtain. Figure (B.7) presents the similarity matrices for the particular cases of the four set of bases of figure (B.6).



(a) Grass: NMF and $r = 40$.

(b) Grass: WNMF and $r = 40$.

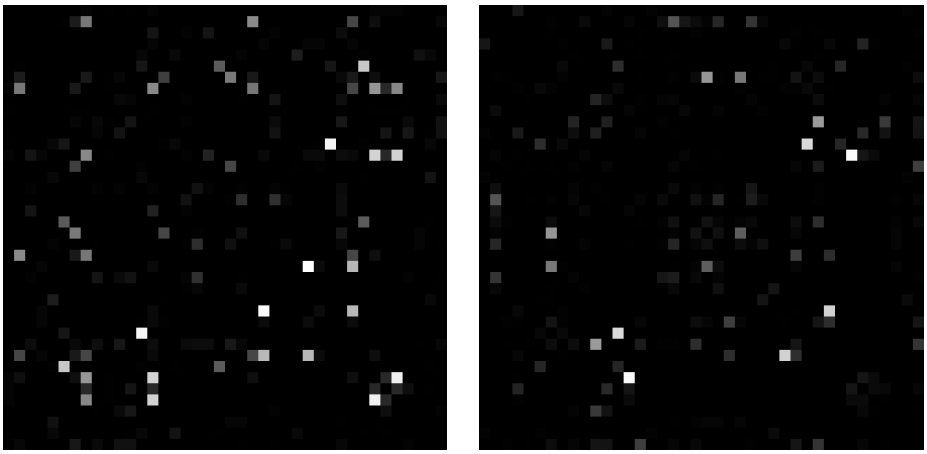
(c) Leaves: NMF and $r = 40$.

(d) Leaves: WNMF and $r = 40$.

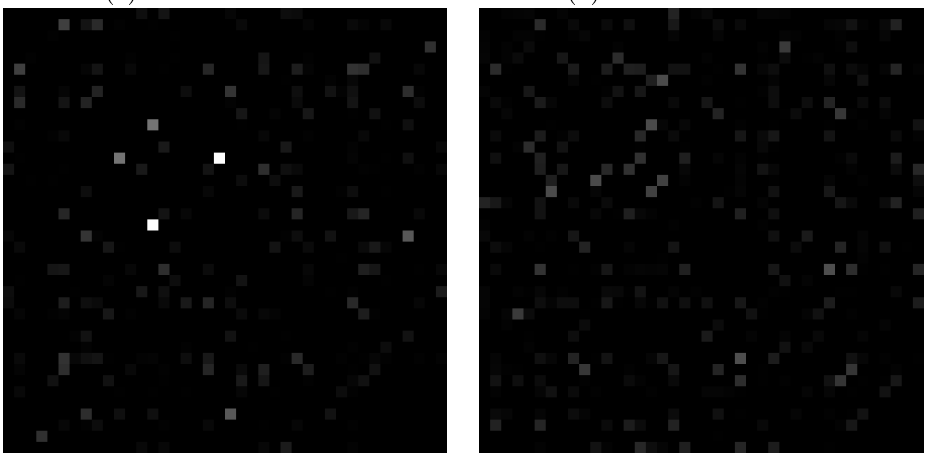
Figure B.6: Graphical representation of the obtained NMF and WNMF bases of Grass and Leaves classes in a $r = 40$ dimensional subspace (NMF and WNMF).

The similarity matrices presented in figure (B.7) can be understood assuming that a white zone corresponds to a high degree of similarity between two bases. Of course, this matrix presents symmetry and diagonal entries have been removed. So that, the way to evaluate the redundancy present in our set of bases is considering the white zones of the similarity matrices. If a matrix contains a considerable amount of white zones, this particular set of bases is very redundant, because they contain duplicated data. For example, if we visually analyze figure (B.7) we are able to appreciate that the use of NMF with the grass class generates more redundancy than considering the use of WNMF. We just have to count the number of white zones of both similarity matrices to realize about this.

Once we have that a similarity matrix of bases $\mathbf{W}$ reflects how a given set of bases is correlated showing some degree of redundancy, we can take advantage of this and evaluate this behaviour through the number of dimensions of the subspace created (r). We evaluate the degree of correlation between two set of bases through dimensionality and it would be expected to find a threshold dimension where WNMF starts to perform better (less redundant bases) than NMF. It is expected that this threshold dimension should be localized around the threshold dimension found in the previous experiment shown in figures (B.2,B.3,B.4,B.5). The reason is that a redundant set of bases should imply to increase the reconstruction error when compared with a set of bases with a low level of redundancy. A set of bases with a low degree of redundancy should be able to represent more "colors" of the original space and the reconstruction error for this particular case should be lower than having a set of bases with a high degree of redundancy. The evaluation of the similarity matrices is shown in figures (B.8,B.9) where we see the degree of correlation between the set of bases through the number of subspace dimensions (r). We should note that given a similarity matrix, we take all the values of the matrix in order to obtain a value that reflects the level of redundancy. We simply sum the values of the similarity matrix and this is the value that we show in figures (B.8,B.9).



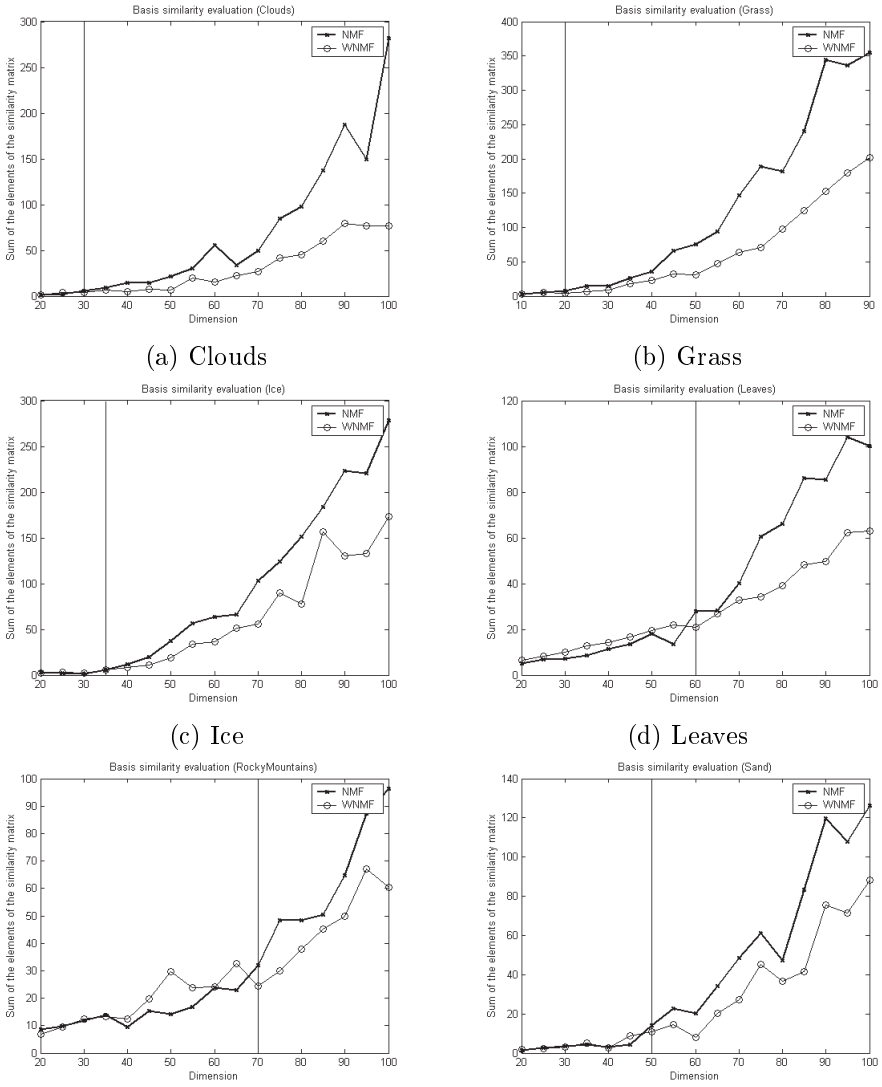
(a) NMF Grass class (b) WNMF Grass class



(c) NMF Leaves class (d) WNMF Leaves class

Figure B.7: Graphical representation of the similarity matrices of bases **W** of grass and leaves classes in a $r = 40$ dimensional subspace. These similarity matrices are obtained using the bases shown in figure (B.6). White zones correspond to high similarity bases and black zones to non similar ones.

Figures (B.2,B.3,B.4,B.5) showed that for a given dimension, WNMF started to perfom better than NMF. Here, with figures (B.8,B.9) we find the same behaviour and, the most important thing, the threshold dimensions are nearly the same as in the previous analysis. Figures (B.8,B.9) show that NMF is always a redundant technique with respect to WNMF when using a high dimensional subspace. Thus, we can conclude stating that the behaviour of NMF in a high dimensional subspace is not the most appropiate if we want to have a correct subspace description of our original data. Analyzing the threshold dimensions of each data class of figures (B.8,B.9), we can see a high correlation with the threshold dimensions of figures (B.2,B.3,B.4,B.5).



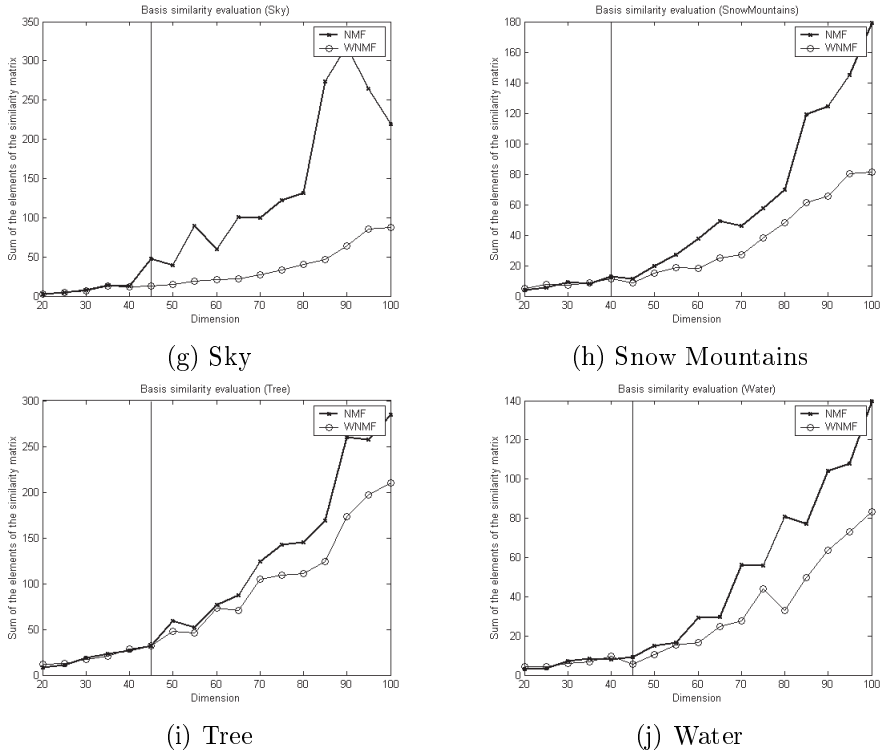
(a) Clouds (b) Grass

(c) Ice (d) Leaves

(e) Rock Mountains (f) Sand

Figure B.8: Evaluation of the similarity matrices of both NMF and WNMF techniques through 20 to 100 dimensions in order to determine when the WNMF starts to obtain a less redundant set of bases. We show classes Clouds, Grass, Ice, Leaves, Rock Mountains and Sand. These results are nearly the same as the ones obtained in figures (B.2,B.3,B.4,B.5).

In the most complex data classes, threshold dimensions are not exactly the same as in the previous experiment but we can find a very close approximation. For example, in the case of rocky mountains, if we analyze the reconstruction error of both NMF



(g) Sky (h) Snow Mountains

(i) Tree (j) Water

Figure B.9: Evaluation of the similarity matrices of both NMF and WNMF techniques through 20 to 100 dimensions in order to determine when the WNMF starts to obtain a less redundant set of bases. We show classes Sky, Snow Mountains, Tree and Water. These results are nearly the same as the ones obtained in figure (B.2,B.3,B.4,B.5).

and WNMF techniques, the threshold dimension is localized in dimension 60 but if we analyze the similarity matrix of their bases, this threshold dimension is localized at 70. Also, we can see in figure (B.3.e) a problematic interval from 60 to 70 dimensions where the reconstruction error does not show which is the best method for this particular case.

From this experiment, we can conclude stating that WNMF always outperforms NMF if we are dealing with local data representations (which contain not uniformly data distributions) and high-dimensional subspaces. We have experimented with 10 different data classes and we are able to obtain a threshold dimension that represents the specific moment where WNMF starts to perform better with respect to the classical NMF approach. We have seen that this threshold dimension is directly related to the complexity of the class and that can be obtained through the analysis of the **W** bases or through the analysis of the reconstruction error of both NMF and WNMF techniques.

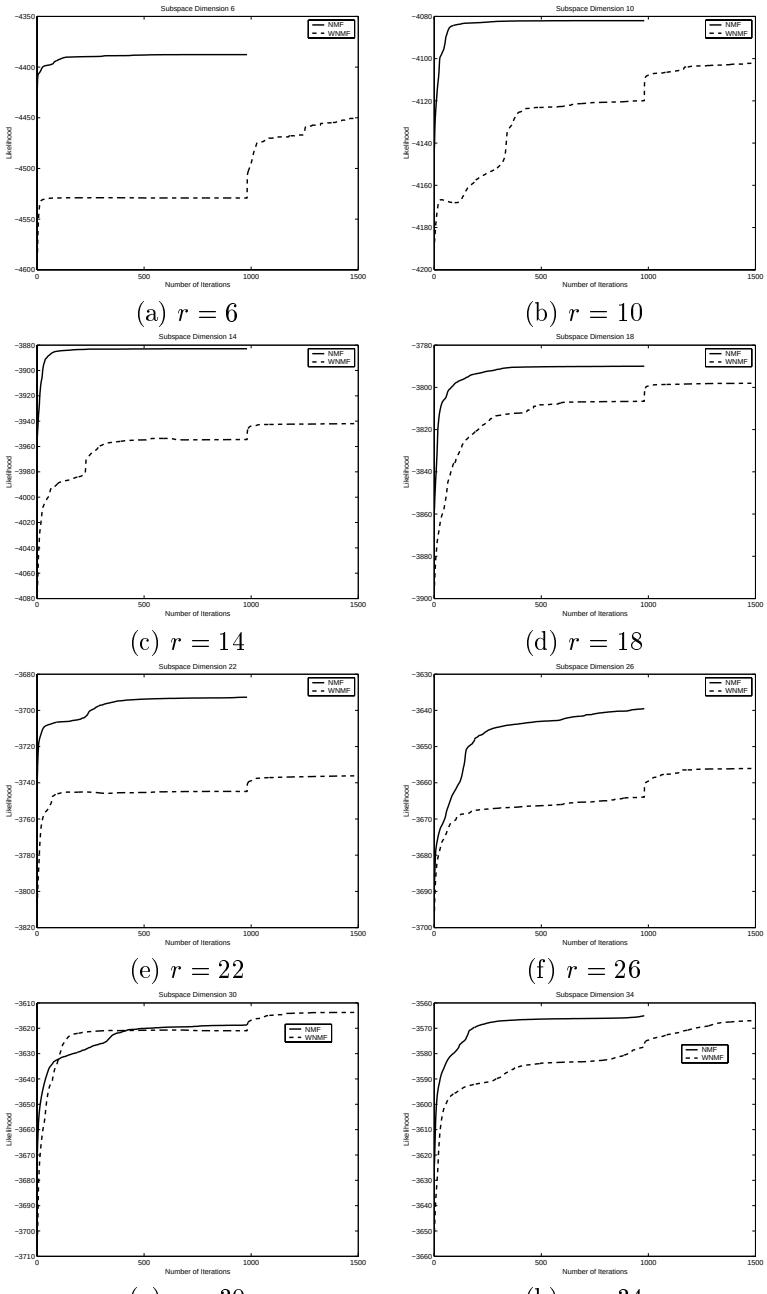
B.2 Evolution Curves of NMF and WNMF

Once WNMF has been presented as a good alternative to the typical NMF technique, one may think if it is an important improvement with respect to NMF in terms of computational resources and time. It is clear, for example, that the iterative update rules of WNMF are exactly the same as the rules of NMF but with a weight matrix. Thus, the computational time required for both techniques is practically the same if we assume that we require the same amount of iterations. So that, in this section we present some graphical results based on the previous experiments where we show the evolution of the likelihood expression defined in expression (3.15).

From the previous experiments where we compared NMF and WNMF, we realized that the performance of WNMF with respect to NMF directly depends on the subspace dimensions. Low dimensional subspaces tend to produce bad WNMF models with respect to NMF since no redundancy is present. When we increase the dimensionality of the subspace, WNMF starts to perform better. This behaviour is shown in the previous experiments. Taking as a reference a particular set of local data vectors (color histograms) belonging to one object, we show the evolution of the likelihood of NMF and WNMF with respect to the number of iterations. We initialize matrix $\mathbf{W}$ of NMF and WNMF with the same random conditions, so that, we are able to compare the evolution error of both techniques. Figures (B.10,B.11) show the evolution error of both techniques for this particular example. We analyzed different subspace dimensions (from $r = 6$ to $r = 58$ with steps of 4 units) and, as expected, we find that WNMF starts to outperform NMF in dimension 38.

We can find two error curves in figures (B.10,B.11). The solid curve shows the evolution error of NMF and the dotted line the evolution error of WNMF. We performed 1000 iterations of NMF and 1000 iterations of WNMF. As seen in these figures, the WNMF curve has 1500 iterations. That is, after applying 1000 iterations of WNMF, we performed 500 iterations of NMF using the local solution of WNMF as a starting point. Surprisingly, we see that after finding a good solution with WNMF, we can continue iterating this initial solution with NMF and find a better approximation of the problem. This is due to the fact that WNMF is a constrained framework that tends to find a completely different local solution with respect to NMF. Once this local solution is found, we can unconstrain the problem using the typical NMF. Using this approach, we are able to find a better solution to the problem. In these figures, we see that for low-dimensional subspaces, WNMF performs very bad even using a posterior NMF. When the subspace dimensionality is about $r = 30$ and $r = 34$, performances of WNMF start to be comparable with the ones of NMF as seen in figure (B.10). We see in figure (B.11) that $r = 38$ is the threshold dimension because WNMF performs better than NMF. But, as seen in this figure, we can iterate our WNMF solution using the typical NMF technique to obtain a better solution to the problem.

Up to $r = 26$ dimensions WNMF is not able to outperform NMF even using a huge amount of iterations. Using $r = 30$ and $r = 34$ dimensions, WNMF can outperform NMF but it requires a relative amount of iterations, usually, more than 1000. When we use more than $r = 34$ dimensions, WNMF always outperforms NMF needing less than 100 iterations to find a stable solution. Since NMF and WNMF are two techniques that require a lot of computational resources, we can use WNMF



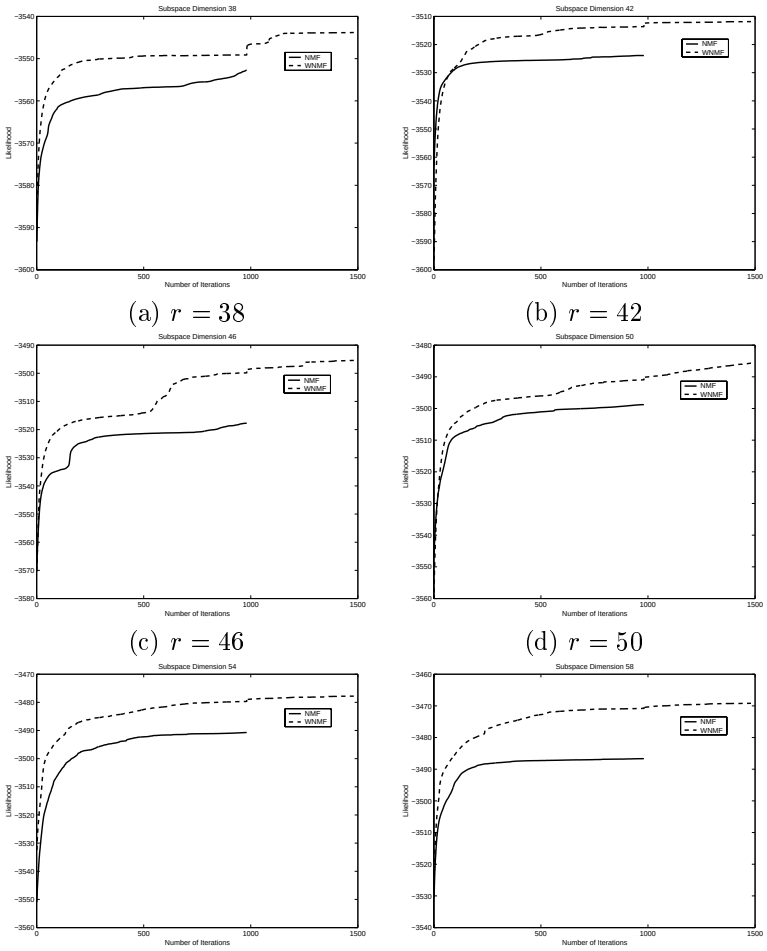
(a) $r = 6$ (b) $r = 10$

(c) $r = 14$ (d) $r = 18$

(e) $r = 22$ (f) $r = 26$

(g) $r = 30$ (h) $r = 34$

Figure B.10: Evolution of the error curves of NMF (solid line) and WNMF (dotted line) in different subspaces (from $r = 6$ to $r = 34$ in steps of 4 units).



(a) $r = 38$ (b) $r = 42$

(c) $r = 46$ (d) $r = 50$

(e) $r = 54$ (f) $r = 58$

Figure B.11: Evolution of the error curves of NMF (solid line) and WNMF (dotted line) in different subspaces (from $r = 38$ to $r = 58$ in steps of 4 units).

to find a reliable solution using only 100 iterations instead of needing 1000 iterations with NMF. However, this can only be done if we previously know that WNMF will outperform NMF. Thus, it is clear that when WNMF outperforms NMF, the number of required iterations is not very demanding (less than 100 iterations).