Non-Negative Matrix Factorization, Convexity and Isometry *

Nikolaos Vasiloglou Alexander G. Gray David V. Anderson†

Abstract

Traditional Non-Negative Matrix Factorization (NMF) [19] is a successful algorithm for decomposing datasets into basis function that have reasonable interpretation. One problem of NMF is that the original Euclidean distances are not preserved. Isometric embedding (IE) is a manifold learning technique that traces the intrinsic dimensionality of a data set, while preserving local distances. In this paper we propose a hybrid method of NMF and IE IsoNMF. IsoNMf combines the advantages of both NMF and Isometric Embedding and it gives a much more compact spectrum compared to the original NMF.

1 Introduction.

Maximum Variance Unfolding (MVU) [29] and its variant Maximum Furthest Neighbor Unfolding (MFNU) [22] are very successful manifold learning methods that reduce significantly the dimension of a dataset. It has been proven experimentally that they can recover the intrinsic dimension of a dataset very effectively, compared to other methods like ISOMAP [27] Laplacian Eigen-Maps [1] and Diffusion Maps [5]. In some toy experiments the above methods manage to decompose data in meaningful dimensions. The statue dataset for example consists of images of a statue photographed from different horizontal and vertical angles. After manifold learning with any of the above methods the initial dimension is reduced to two, where each of them represents the horizontal and the vertical camera angle. For more complex datasets it is not possible to find an interpretation of the dimensions in the low dimensional space.

Non-Negative Matrix Factorization (NMF) is another dimensionality reduction method [19]. Although NMF is targeted for non-negative data, in reality it is an additive component model, the sign doesn't really matter as long as the components have the same sign. As we will prove later NMF can never give better dimensionality reduction than Singular Value Decomposition although the principal components of NMF are more meaningful than SVD. Another drawback of NMF is that points that are close in the original domain may actually land up far away after NMF.

In this paper we present a hybrid of NMF and MFNU in one algorithm that we call IsoNMF and show that it combines the advantages of both (local neighborhood preservation and interpretability of the results) plus it gives a more sparsity compared to traditional NMF.

2 Convexity in Non Negative Matrix Factorization.

Given a non-negative matrix $V \in \Re_+^{N\times m}$ the goal of NMF is to decompose it in two matrices $W \in \Re_+^{N\times k}$, $H \in \Re_+^{k\times m}$ such that $V = WH$. Such a factorization always exists for $k \geq m$. The factorization has a trivial solution where $W = V$ and $H = I_m$. Determining them minimum k is a difficult problem and no algorithm exists for finding it. In general we can show that NMF can be cast as a Completely Positive (CP) Factorization problem [2].

DEFINITION 2.1. *A matrix $A \in \Re_+^{N\times N}$ is Completely Positive if it can be factored in the form $A = BB^T$, where $B \in \Re_+^{N\times k}$. The minimum k for which $A = BB^T$ holds is called the CP rank of A.*

Up to now there is no algorithm of polynomial complexity that can decide if a given positive matrix is CP. A simple observation can show that A has to be positive definite, but this is a necessary and not a sufficient condition.

THEOREM 2.1. *If A is CP then $rank(A) \leq cp\text{-}rank(A) \leq \frac{k(k+1)}{2} - 1$*

The proof can be found in [2]p.156. It is also conjectured that the upper bound can be tighter $\frac{k^2}{4}$.

THEOREM 2.2. *if $A \in \Re_+^{N\times N}$ is diagonally dominant[1] then it is also CP.*

The proof of the theorem can be found in [16].

Next we prove that non-trivial NMF always exists.

THEOREM 2.3. *Every non-negative matrix $V \in \Re_+^{N\times M}$ has a non-trivial, non-negative factorization of the form $V = WH$.*

*Supported by Google Grants

†Georgia Institute of Technology

[1] A matrix is diagonally dominant if $a_{ii} \geq \sum_{j\neq i} |a_ij|$

Proof. Consider the following matrix:

$$Z = \begin{bmatrix} D & V \\ V^T & E \end{bmatrix} \tag{2.1}$$

We want to prove that there exists $B \in \Re_+^{N \times k}$ such that $Z = BB^T$. If this is true then B can take the form:

$$B = \begin{bmatrix} W \\ H^T \end{bmatrix} \tag{2.2}$$

Notice that D and E are arbitrary positive matrices. We can always adjust them so that Z is diagonally dominant and according to theorem 2.2 Z is always CP. Since Z is CP then B exists so do W and H $\square$

Although theorem 2.2 also provides an algorithm for constructing the CP-factorization, the cp-rank is usually high. A corollary of theorems 2.1 (cp-rank$(A) \geq$ rank(A)) and 2.3(existence of NMF) is that SVD has always a more compact spectrum than NMF since the.

There is no algorithm known yet for computing an exact NMF despite its existence. In practice, scientists try to minimize the norm of the factorization error.

$$\min_{W,H} ||V - WH||_2 \tag{2.3}$$

2.1 Solving the optimization problem of NMF. Although in the current literature it is widely believed that NMF is a non-convex problem and only local minima can be found, we will show in the following subsections that a convex formulation does exist. Despite the existence of the convex formulation, we also show that a formulation of the problem as a generalized geometric program could give us a better approach to the global optimum.

2.1.1 NMF as a convex conic program.

THEOREM 2.4. *The set of Completely Positive Matrices $\mathcal{K}^{\mathcal{CP}}$ is a convex cone.*

Proof. See [2].71.

Finding the minimum rank NMF can be cast as the following optimization problem:

$$\min_{\mathcal{W},\mathcal{H}} \quad \text{rank}(Z) \tag{2.4}$$

subject to:

$$\begin{aligned} &\mathcal{W} \in \mathcal{K}^{\mathcal{CP}} \\ &\mathcal{H} \in \mathcal{K}^{\mathcal{CP}} \\ &Z = \begin{bmatrix} \mathcal{W} & V \\ V^T & \mathcal{H} \end{bmatrix} \end{aligned} \tag{2.5}$$

Since minimizing the function rank(Z) is non-convex we can use it's convex envelope that according to [25] is the trace of the matrix. So a convex relaxation of the above problem is:

$$\min_{\mathcal{W},\mathcal{H}} \quad \text{trace}(Z) \tag{2.6}$$

subject to:

$$\begin{aligned} &\mathcal{W} \in \mathcal{K}^{\mathcal{CP}} \\ &\mathcal{H} \in \mathcal{K}^{\mathcal{CP}} \\ &Z = \begin{bmatrix} \mathcal{W} & V \\ V^T & \mathcal{H} \end{bmatrix} \end{aligned} \tag{2.7}$$

After determining $\mathcal{W}, \mathcal{H}$, W and H can be recovered by CP factorization of $\mathcal{W}, \mathcal{H}$, which again is not an easy problem. In fact there is no practical barrier function known yet for the CP cone so that Interior Point Methods can be employed. Finding a practical description of the CP cone is an open problem. So although the problem is convex there is no algorithm known for solving it.

2.2 Convex relaxations of the NMF problem. In the following subsections we investigate convex relaxations of the NMF problem with the Positive Semidefinite Cone [23].

2.2.1 A simple convex upper bound with Singular Value Decomposition.

Singular Value Decomposition (SVD) can decompose a matrix in two factors:

$$A = UV \tag{2.8}$$

Unfortunately the sign of the SVD components of $A \geq 0$ cannot be guaranteed to be non-negative except for the first eigenvector [21]. However if we project U, V on the nonnegative orthant ($U, V \geq 0$) we get a very good estimate (upper bound) for NMF. We will call it clipped SVD, (CSVD). CSVD was used as a benchmark for the relaxations that follow. It has also been used as an initializer for NMF algorithms [18].

2.2.2 Relaxation with a positive semidefinite cone.

In the minimization problem of eq. 2.3 where the cost function is the L_2 norm, the nonlinear terms $w_{il}h_{lj}$ appear. A typical way to get these terms [23] would be to generate a large vector $\vec{z} = [W'(:); H(:)]$, where we use the MATLAB notation ($H(:)$ is the column-wise unfolding of a matrix). If $Z = zz^T$ and $z > 0$ is true then the terms appearing in $||V - WH||_2$ are linear in Z. In the following example eq. 2.9, 2.10 (see next page) where $V \in \Re^{2\times 3}, W \in \Re^{2\times 2}, H \in \Re^{2\times 3}$ we show the structure of Z. Terms in bold are the ones we need to

express the constraint $V = WH$.

$$z = \begin{bmatrix} w_{11} \\ w_{12} \\ w_{21} \\ w_{22} \\ h_{11} \\ h_{21} \\ h_{12} \\ h_{22} \\ h_{13} \\ h_{23} \end{bmatrix} \tag{2.9}$$

Now the optimization problem is equivalent to:

$$\begin{aligned} &\min \sum_{i=1,j=1}^{i=N,j=m} (\sum_{l=1}^{k} Z_{ik+l,Nk+jk+l} - V_{ij})^2) \\ &\text{subject to:} \\ &\text{rank(Z)} = 1 \end{aligned} \tag{2.11}$$

This is not a convex problem but it can be easily approximated by

$$\begin{aligned} &\min \text{Trace}(Z) \\ &\text{subject to:} \\ &A \bullet Z = V_{ij} \\ &Z \succeq 0 \\ &Z \succeq zz^T \\ &Z \geq 0 \end{aligned} \tag{2.12}$$

where A is a matrix that selects the appropriate elements from Z. Here is an example for a matrix A that selects the elements of Z that should sum to the V_{13} element:

$$A_{13} = \begin{bmatrix} & 0 & 0 & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 1 \\ \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & & & & \mathbf{0} & & \end{bmatrix} \tag{2.13}$$

In the second formulation (2.12) we have relaxed $Z = zz^T$ with $Z \succeq zz^T$. The objective function tries to minimize the rank of the matrix, while the constraints try to match the values of the given matrix V. After solving the optimization problem the solution can be found on the first eigenvector of Z. The quality of the relaxation depends on the ratio of the first eigenvalue to the rest. The positivity of Z will guarantee that the first eigenvector will have elements with the same sign according to the Peron Frobenious Theorem [21]. Ideally if the rest of the eigenvectors are positive they can also be included. One of the problem of this method is the complexity. Z is $(N+m)k \times (N+m)k$ and there are $\frac{((N+m)k)((N+m)k-1)}{2}$ positivity constraints. Very quickly the problem becomes unsolvable.

In practice the problem as posed in (**??**) always gives W and H matrices that are rank one. After testing the method exhaustively with random matrices V that either had a product $V = WH$ representation or not the solution was always a rank one on both W and H. This was always the case with any of the convex formulations presented in this paper. This is because there is a missing constraint that will let the energy of the dot products spread among dimensions. This is something that should characterize the spectrum of H. The H matrix is often interpreted as the basis vectors of the factorization and W as the matrix that has the coefficients. It is widely known that in nature spectral analysis is giving spectrum that decays either exponentially $e^{-\lambda f}$ or more slowly $1/f^{\gamma}$. Depending on the problem we can try different spectral functions. In our experiments we chose the exponential one. Of course the decay parameter λ is something that should be set adhoc. We experimented with several values of λ, but we couldn't come up with a systematic, heuristic and practical rule. In some cases the reconstruction error was low but in some others not. Another relaxation that was necessary for making the optimization tractable was to reduce the the non-negativity constraints only on the elements that are involved in the equality constraints.

2.2.3 Approximating the SDP scone with smaller ones. A different way to deal with the computational complexity of SDP is to approximate the big SDP cone $(N+m)k \times (N+m)k$ with smaller ones. Let W_i be the ith row of W and H_j the j_{th} column of H. Now $z_{ij} = [W_i(:); H_j(:)]$ ($2k$ dimensional vector) and $Z_{ij} = z_{ij} z_{ij}^T$ ($2k \times 2k$ matrix), or

$$Z_{ij} = \begin{bmatrix} W_i^T W_i & W_i^T H_j \\ W_i^T H_j & H_j H_j^T \end{bmatrix} \tag{2.14}$$

or it is better to think it in the form:

$$Z_{ij} = \begin{bmatrix} \mathcal{W}_i & \mathcal{Z}_{WH} \\ \mathcal{Z}_{WH} & \mathcal{H}_j \end{bmatrix} \tag{2.15}$$

and once $\mathcal{W}, \mathcal{H}$ are found then W_i, H_j can be found from SVD decomposition of $\mathcal{W}, \mathcal{H}$ and again the quality of the relaxation will be judged upon the magnitude of the first eigenvalue compared to the sum of the others.

Now the optimization problem becomes:

$$\min \sum_{i=1}^{Nm} \text{Trace}(Z_{ij}) \tag{2.16}$$

$$
(2.10)\quad Z = \begin{bmatrix}
w_{11}^2 & w_{11}w_{12} & w_{11}w_{21} & w_{11}w_{22} & \mathbf{w_{11}h_{11}} & w_{11}h_{21} & \mathbf{w_{11}h_{12}} & w_{11}h_{22} & \mathbf{w_{11}h_{13}} & w_{11}h_{23} \\
w_{12}w_{11} & w_{12}^2 & w_{12}w_{21} & w_{12}w_{22} & w_{12}h_{11} & \mathbf{w_{12}h_{21}} & w_{12}h_{12} & \mathbf{w_{12}h_{22}} & w_{12}h_{13} & \mathbf{w_{12}h_{23}} \\
w_{21}w_{11} & w_{21}w_{12} & w_{21}^2 & w_{21}w_{22} & \mathbf{w_{21}h_{11}} & w_{21}h_{21} & \mathbf{w_{21}h_{12}} & w_{21}h_{22} & \mathbf{w_{21}h_{13}} & w_{21}h_{23} \\
w_{22}w_{11} & w_{22}w_{12} & w_{22}w_{21} & w_{22}^2 & w_{22}h_{11} & \mathbf{w_{22}h_{21}} & w_{22}h_{12} & \mathbf{w_{22}h_{22}} & w_{22}h_{13} & \mathbf{w_{22}h_{23}} \\
h_{11}w_{11} & h_{11}w_{12} & h_{11}w_{21} & h_{11}w_{22} & h_{11}^2 & h_{11}h_{21} & h_{11}h_{12} & h_{11}h_{22} & h_{11}h_{13} & h_{11}h_{23} \\
h_{21}w_{11} & h_{21}w_{12} & h_{21}w_{21} & h_{21}w_{22} & h_{21}h_{11} & h_{21}^2 & h_{21}h_{12} & h_{21}h_{22} & h_{21}h_{13} & h_{21}h_{23} \\
h_{12}w_{11} & h_{12}w_{12} & h_{12}w_{21} & h_{12}w_{22} & h_{12}h_{11} & h_{12}h_{21} & h_{12}^2 & h_{12}h_{22} & h_{12}h_{13} & h_{12}h_{23} \\
h_{22}w_{11} & h_{22}w_{12} & h_{22}w_{21} & h_{22}w_{22} & h_{22}h_{11} & h_{22}h_{21} & h_{22}h_{12} & h_{22}^2 & h_{22}h_{13} & h_{22}h_{23} \\
h_{13}w_{11} & h_{13}w_{12} & h_{13}w_{21} & h_{13}w_{22} & h_{13}h_{11} & h_{13}h_{21} & h_{13}h_{12} & h_{13}h_{22} & h_{13}^2 & h_{13}h_{23} \\
h_{23}w_{11} & h_{23}w_{12} & h_{23}w_{21} & h_{23}w_{22} & h_{23}h_{11} & h_{23}h_{21} & h_{23}h_{12} & h_{23}h_{22} & h_{23}h_{13} & h_{23}^2
\end{bmatrix}
$$

$$
\begin{aligned}
Z_{ij} &\geq 0 \\
Z_{ij} &\succeq 0 \\
A_{ij} \bullet Z_{ij} &= v_{ij}
\end{aligned}
$$

The above method has Nm constraints. In terms of storage it needs

- $(N + m)$ symmetric positive definite $k \times k$ matrices for every row/column of W, H, which is $\frac{(N+m)k(k+1)}{2}$
- Nm symmetric positive definite $k \times k$ matrices for every $W_i H_j$ product, which is $\frac{(Nm)k(k+1)}{2}$

In total the storage complexity is $(N+m+Nm)\frac{k(k+1)}{2}$ which is significantly smaller by an order of magnitude from $\frac{(N+m)k((N+m)k-1)}{2}$ which is the complexity of the previous method. There is also significant improvement in the computational part. The SDP problem is solved with interior point methods [23] that require the inversion of a symmetric positive definite matrix at some point. In the previous method that would require $O((N+m)^3k^3)$ steps, while with this method we have to invert Nm $2k \times 2k$ matrices, that would cost $Nm(2k)^3$. Because of their special structure the actual cost is $(Nm)k^3+\max(N,m)k^3 = (Nm+\max(N,m))k^3$.

We know that $\mathcal{W}_i, \mathcal{H}_j \succeq 0$. Since Z_{ij} is PSD and according to Schur's complement on eq. 2.15:

$$
(2.17)\qquad \mathcal{H}_j - \mathcal{Z}_{WH}\mathcal{W}_i^{-1}\mathcal{Z}_{WH} \succeq 0
$$

So instead of inverting (2.15) that would cost $8k^3$ we can invert 2.17. This formulation gives similar results with the big SDP cone and most of the cases the results are comparable to the CSVD.

2.2.4 NMF as a convex multi-objective problem. A different approach would be to find a convex set in which the solution of the NMF lives and search for it over there. Assume that we want to match $V_{ij} = W_i H_j = \sum_{l=1}^{m} W_{il}H_{lj}$. Define $W_{il}H_{lj} = V_{ij,l}$ and $\sum_{l=1}^{k} V_{ij,l} = V_{ij}$. We form the following matrix that we require to be PSD:

$$
(2.18)\qquad \begin{bmatrix} 1 & W_{il} & H_{lj} \\ W_{il} & t_{il} & V_{ij,l} \\ H_{lj} & V_{ij,l} & t_{jl} \end{bmatrix} \succeq 0
$$

If we use the Schur complement we have:

$$
(2.19)\quad \begin{bmatrix} t_{il} - W_{il}^2 & V_{ij,l} - W_{il}H_{lj} \\ V_{ij,l} - W_{il}H_{lj} & t_{jl} - H_{lj}^2 \end{bmatrix} \succeq 0
$$

An immediate consequence is that

$$
\begin{aligned}
(2.20)\qquad t_{il} &\geq W_{il}^2 \\
(2.21)\qquad t_{jl} &\geq H_{ll}^2 \\
(2.22)\qquad (t_{il} - W_{il}^2)(t_{jl} - H_{lj}^2) &\geq (V_{ij,l} - W_{il}H_{lj})^2
\end{aligned}
$$

In the above inequality we see that the mean square error becomes zeros if $t_{il} = W_{il}^2$ or $t_{jl} = H_{il}^2$. In general we want to minimize t while maximizing $||W||^2$ and $||H||^2$. L_2 Norm maximization is not convex, but instead we can maximize $\sum W_{il}, \sum H_{lj}$ which are equal to the L_1 norms since everything is positive. This can be cast as convex multi-objective problem [2] on the second order cone [3].

$$
\begin{aligned}
\min \quad & \begin{bmatrix} \sum_{i=1}^{i=N}\sum_{l=1} kt_{il} + \sum_{j=1}^{j=m}\sum_{l=1} kt_{lj} \\ -\sum_{i=1}^{i=N}\sum_{l=1} kW_{il} - \sum_{j=1}^{j=m}\sum_{l=1} kH_{lj} \end{bmatrix} \\
(2.23)\quad & \text{subject to :} \\
& \begin{bmatrix} t_{il} - W_{il}^2 & V_{ij,l} - W_{il}H_{lj} \\ V_{ij,l} - W_{il}H_{lj} & t_{jl} - H_{lj}^2 \end{bmatrix} \succeq 0
\end{aligned}
$$

Unfortunately multi-objective optimization problems even when they are convex they have local minima that are not global too. An interesting direction would be to test the robustness of existing multi-objective algorithms on NMF.

2.2.5 Local solution of the non-convex problem. In the previous sections we show several convex formulations and relaxations of the NMF problem that

[2] also known as vector optimization

unfortunately are either unsolvable or they give trivial rank one solutions that are not useful at all. In practice the the non-convex formulation of eq. 2.3 along with other like the KL distance between V and WH are used in practice. All of them are non-convex and several methods have been recommended, such as alternating least squares, gradient decent or active set methods [17]. In our experiments we used the L-BFGS method that scales very well for large matrices.

2.2.6 NMF as a Generalized Geometric Program and it's Global Optimum. The objective function can be written in the following form:

$$(2.24)\quad ||V - WH||_2 = \sum_{i=1}^{N}\sum_{j=1}^{m}\sum_{l=1}^{k}(V_{ij} - W_{il}H_{lj})^2$$

The above function is twice differential so according to [10] the function can be cast as the difference of convex (d.c.) functions. The problem can be solved with general of the shelf global optimization algorithms. The problem can also e formulated as a special case of dc programming, the generalized geometric programming. With the following transformation $W_{il} = e^{\tilde{w}_{il}}, H_{lj} = e^{\tilde{h}_{lj}}$ the objective becomes:

$$\begin{aligned}(2.25)\quad ||V - WH||_2 &= \sum_{i=1}^{N}\sum_{j=1}^{m}\sum_{l=1}^{k}\left(V_{ij} - e^{\tilde{w}_{il}+\tilde{h}_{lj}}\right)^2 \\ &= \sum_{i=1}^{N}\sum_{j=1}^{m}V_{ij}^2 + \sum_{i=1}^{N}\sum_{j=1}^{m}\left(\sum_{l=1}^{k}e^{\tilde{w}_{il}+\tilde{h}_{lj}}\right)^2 \\ &-2\sum_{i=1}^{N}\sum_{j=1}^{m}V_{ij}\left(\sum_{l=1}^{k}e^{\tilde{w}_{il}+\tilde{h}_{lj}}\right)\end{aligned}$$

The first term is constant and it can be ignored for the optimization. The other two terms:

$$(2.26) f(\tilde{w}_{il},\tilde{h}_{lj}) = \sum_{i=1}^{N}\sum_{j=1}^{m}\left(\sum_{l=1}^{k}e^{\tilde{w}_{il}+\tilde{h}_{lj}}\right)^2$$

$$(2.27) g(\tilde{w}_{il},\tilde{h}_{lj}) = 2\sum_{i=1}^{N}\sum_{j=1}^{m}V_{ij}\left(\sum_{l=1}^{k}e^{\tilde{w}_{il}+\tilde{h}_{lj}}\right)$$

are convex functions also known as the exponential form of posynomials [3] [3]. For the global solution of the problem

$$(2.28)\qquad \min_{\tilde{W},\tilde{H}} = \quad f(\tilde{W},\tilde{H}) - g(\tilde{W},\tilde{H})$$

the algorithm proposed in [6] can be employed.

[3]Posynomial is a product of positive variables exponentiated in any real number

Since the above method is impractical in its form as it requires too many iterations to converge, it is worthwhile to compare it with the local non-convex NMF solver on a small matrix. We tried to do NMF of order 2 on the following random matrix:

$$\begin{bmatrix} 0.45 & 0.434 & 0.35 \\ 0.70 & 0.64 & 0.43 \\ 0.22 & 0.01 & 0.3 \end{bmatrix}$$

After 10000 restarts of the local solver the best error we got was 2.7% while the global optimizer very quickly gave 0.015% error, which is 2 orders of magnitude less than the local optimizer.

Another direction that is not investigated in this paper is the recently developed algorithm for Difference Convex problems by Tao [26] that has been applied successfully to other data mining applications such as Multidimensional Scaling. [9].

3 Isometric Embedding

The key concept in Manifold Learning (ML)is to represent a dataset in a lower dimensional space by preserving the local distances. The differences between methods Isomap [27], Maximum Variance unfolding [29], Laplacian EigenMaps [1] and Diffusion Maps [5] is how they treat distances between points that are not in the local neighborhood. For example IsoMap preserves exactly the geodesic distances, while Diffusion Maps preserves distances that are based on the diffusion kernel. Maximum Furthest Neighbor Unfolding (MFNU) [22] that is a variant of Maximum Variance Unfolding (MVU), preserves local distance and it tries to maximize the distance between furthest neighbors. In this section we are going to present the MFNU method as it will be the basis for building IsoNmf.

3.1 Convex Maximum Furthest Neighbor Unfolding. Weinberger formulated the problem of isometric unfolding as a Semidefinite Programming algorithm [29]. In [22] Vasiloglou presented a variance of MVU the MFNU. The latest formulation tends to be more robust and scalable than MVU, this is why we will employ it as the basis of IsoNMF. Both methods can be cast as a semidefinite programming problem [28].

Given a set of data $X \in \Re^{N\times d}$, where N is the number of points and d is the dimensionality. The dot product or Gramm matrix is defined as $G = XX^T$. The goal is to find a new Gramm matrix K such that $rank(K) < rank(G)$ in other words $K = \hat{X}\hat{X}^T$ where $\hat{X} \in \Re^{N\times d'}$ and $d' < d$. Now the dataset is represented by $\hat{X}$ which has fewer dimensions that X. The requirement of isometric unfolding is that the euclidian distances in the $\Re^{d'}$ for a given neighborhood

around every point have to be the same as in the $\Re^d$. This is expressed in:

$$K_{ii} + K_{jj} - K_{ij} - K_{ji} = G_{ii} + G_{jj} - G_{ij} - G_{ji}, \forall i, j \in I_i$$

where I_i is the set of the indices of the neighbors of the *ith* point. From all the K matrices MFNU chooses the one that maximizes the distances between furthest neighbor pairs. So the algorithm is presented as an SDP:

$$\begin{aligned} \max_K & \sum_{i=1}^{N} A_{ij} \bullet K \\ \text{i, j furthest neighbors} & \\ \text{subject to} & \\ A_{ij} \bullet K &= d_{ij} \quad \forall j \in I_i \end{aligned}$$

where the $A \bullet B = \text{Trace}(AB^T)$ is the dot product between matrices. A_{ij} has the following form:

$$\begin{bmatrix} 1 & 0 & \dots & -1 & \dots & 0 \\ 0 & \ddots & 0 & \dots & 0 & 0 \\ \vdots & 0 & \ddots & 0 & \dots & 0 \\ -1 & \dots & 0 & 1 & \dots & 0 \\ \vdots & 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & \dots & 0 & \dots & 0 \end{bmatrix} \tag{3.29}$$

and

$$d_{ij} = G_{ii} + G_{jj} - G_{ij} - G_{ji} \tag{3.30}$$

The last condition is just a centering constraint for the covariance matrix. The new dimensions $\hat{X}$ are the eigenvectors of K. In general MFNU gives Gram matrices that have compact spectrum at least more compact than traditional linear Principal Component Analysis (PCA). Unfortunately this method can handle datasets of no more than hundreds of points because of its complexity.

3.2 The Non Convex Maximum Furthest Neighbor Unfolding. By replacing the constraint $K \succeq 0$ [4] with an explicit rank constraint $K = RR^T$ the problem becomes non-convex and it is reformulated to

$$\begin{aligned} \max \quad & \sum_{i=1}^{N} A_{ij} \bullet RR^T \\ & \text{i, j furth. neighbors} \\ & \text{subject to:} \\ & A_{ij} \bullet RR^T = d_{ij} \end{aligned} \tag{3.31}$$

The above problem can be solved with the augmented Lagrangian method [24].

$$\begin{aligned} \mathcal{L} &= -\sum_{i=1}^{N} A_{ij} \bullet RR^T \\ &\quad -\sum_{i=1}^{N} \sum_{\forall j \in I_i} \lambda_{ij} (A_{ij} \bullet RR^T - d_i j) \\ &\quad +\frac{\sigma}{2} \sum_{i=1}^{N} \sum_{\forall j \in I_i} (A_{ij} \bullet RR^T - d_{ij})^2 \end{aligned} \tag{3.32}$$

Our goal is to minimize the Lagrangian that's why the objective function is $-RR^T$ and not RR^T

The derivative of the augmented Lagrangian is:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial R} &= -2 \sum_{i=1}^{N} A_{ij} \bullet R \\ &\quad -2 \sum_{i=1}^{N} \sum_{\forall j \in I_i} \lambda_{ij} A_{ij} R \\ &\quad 2\sigma \sum_{i=1}^{N} \sum_{\forall j \in I_i} (A_{ij} \bullet RR^T - d_i j) A_{ij} R \end{aligned} \tag{3.33}$$

Gradient descent is a possible way to solve the minimization of the Lagrangian, but it is rather slow. The Newton method is also prohibitive. The Hessian of this problem is a sparse matrix although the cost of the inversion might be high it is worth investigating. In our experiments we used the limited memory BFGS (L-BFGS) method [20, 24] that is known to give a good rate for convergence.

4 Isometric NMF.

NMF and MFNU are optimization problems. The goal of IsoNMF is to combine these optimization problems in one optimization problem. MFNU has a convex and a non-convex formulation, while for NMF only a non-convex formulation that can be solved is known.

4.1 Convex IsoNMF. By using the theory presented in section 2.1.1 we can cast IsoNMF as a convex problem:

$$\begin{aligned} \max_{\tilde{W}, \tilde{H}} \quad & \sum_{i=1}^{N} A_{ij} \bullet Z \\ & \text{i, j furthest neighbors} \\ & \text{subject to:} \\ & A_{ij} \bullet \tilde{W} = d_{ij} \\ & Z = \begin{bmatrix} \tilde{W} & V \\ V^T & \tilde{H} \end{bmatrix} \end{aligned} \tag{4.34}$$

$$Z \in \mathcal{K}^{\mathcal{CP}}$$
$$\tilde{W} \in \mathcal{K}^{\mathcal{CP}}$$
$$\tilde{H} \in \mathcal{K}^{\mathcal{CP}}$$

Then W, H can be found by the complete factorization of $\tilde{W} = WW^T, \tilde{H} = HH^T$. Again this problem although it is convex, there is no polynomial algorithm known for solving it.

4.2 Non-convex IsoNMF. The non convex IsoNMF can be cast as the following problem:

$$\begin{aligned} (4.35) \qquad \max \quad & \sum_{i=1}^{N} A_{ij} \bullet WW^T \\ & \text{i, j furthest neighbors} \\ & \text{subject to:} \\ & A_{ij} \bullet WW^T = d_{ij} \\ & WH = V \\ & W \geq 0 \\ & H \geq 0 \end{aligned}$$

The augmented lagrangian with quadratic penalty function is the following:

$$\begin{aligned} (4.36)\ \mathcal{L} \quad = \quad & -\sum_{i=1}^{N} A_{ij} \bullet WW^T \\ & -\sum_{i=1}^{N} \sum_{\forall j \in I_i} \lambda_{ij} (A_{ij} \bullet WW^T - d_i j) \\ & -\sum_{i=1}^{N} \sum_{j=1}^{m} \mu_{ij} (\sum_{l=1}^{k} (W_{ik} H_{kj} - V_{ij})) \\ & +\frac{\sigma_1}{2} \sum_{i=1}^{N} \sum_{\forall j \in I_i} (A_{ij} \bullet WW^T - d_{ij})^2 \\ & +\frac{\sigma_2}{2} \sum_{i=1}^{N} \sum_{j=1}^{m} (\sum_{l=1}^{k} (W_{il} H_{lj} - V_{ij})^2 \end{aligned}$$

The non-negativity constraints are missing from the Lagrangian. This is because we can enforce them through the limited bound BFGS also known as L-BFGS-B. The derivative of the augmented Lagrangian is:

$$\begin{aligned} (4.37) \qquad \frac{\partial \mathcal{L}}{\partial W} = & -2 \sum_{i=1}^{N} A_{ij} W \\ & -2 \sum_{i=1}^{N} \sum_{\forall j \in I_i} \lambda_{ij} A_{ij} W \\ & -\sum_{i=1}^{N} \sum_{j=1}^{m} \mu_{ij} W \\ & +2\sigma_1 \sum_{i=1}^{N} \sum_{\forall j \in I_i} (A_{ij} \bullet WW^T - d_i j) A_{ij} W \\ & +2\sigma_2 \sum_{i=1}^{N} \sum_{j=1}^{m} (\sum_{l=1}^{k} (W_{il} H_{lj} - V_{ij}) W \end{aligned}$$

$$\begin{aligned} (4.38) \qquad \frac{\partial \mathcal{L}}{\partial H} = & -\sum_{i=1}^{N} \sum_{j=1}^{m} \mu_{ij} H \\ & +2\sigma_2 \sum_{i=1}^{N} \sum_{j=1}^{m} (\sum_{l=1}^{k} (W_{il} H_{lj} - V_{ij}) H \end{aligned}$$

4.3 Computing the local neighborhoods. As already discussed in previous section MFNU and IsoNMF require the computation of all-nearest and all-furthest neighbors. The all-nearest neighbor problem is a special case of a more general class of problems called N-body problems [8]. In the following sections we give a sort description of the nearest neighbor computation. The actual algorithm is a four-way recursion. More details can be found in [8].

4.4 Kd-tree. The kd-tree is a hierarchical partitioning structure for fast nearest neighbor search [7]. Every node is recursively partitioned in two nodes until the points contained are less than a fixed number. This is a leaf. Nearest neighbor search is based on a top down recursion until the query point finds the closest leaf. When the recursion hits a leaf then it searches locally for a candidate nearest neighbor. At this point we have an upper bound for the nearest neighbor distance, meaning that the true neighbor will be at most as far away as the candidate one. As the recursion backtracks it eliminates (prunes) nodes that there are further away than the candidate neighbor. Kd-trees provide on the average nearest neighbor search in $O(\log N)$ time, although for pathological cases the kd-tree performance can asymptotically have linear complexity like the naive method.

4.5 The Dual Tree Algorithm for nearest neighbor computation. In the single tree algorithm the reference points are ordered on a kd-tree. Every nearest neighbor computation requires $O(\log(N))$ computations. Since there are N query points the total cost is $O(N \log(N))$. The dual-tree algorithm [8] orders the query points on a tree too. If the query set and the reference set are the same then they can share the same tree. Instead of querying a single point at a time the dual-tree algorithm always queries a group of points that live in the same node. So instead of doing the top-down recursion individually for every point it does

it for the whole group at once. Moreover instead of computing distances between points and nodes it computes distances between nodes. This is the reason why most of the times the dual-tree algorithm can prune larger portions of the tree than the single tree algorithm. The complexity of the dual-tree algorithm is empirically $O(N)$. If the dataset is pathological then the algorithm can be of quadratic complexity too. The pseudo-code for the algorithm is described in fig. 1.

```
recurse(q : KdTree, r : KdTree) {
  if (max_nearest_neighbor_distance_in_node(q)
        < distance(q, r) {
     /* prune */
  } else if (IsLeaf(q)==true and IsLeaf(r)==true) {
    /* search for every point in q  node  */
    /* its nearest neighbor in the r node */
    /* at leaves we must resort to        */
    /* exhaustive search O(n^2)           */
    /*update the maximum_neighbor_distance_in_node(q)*/
  } else if (IsLeaf(q)==false and IsLeaf(r)=true {
    /* choose the child that is closer to r */
    /* and recurse first                    */
    recurse(closest(r, q.left, q.right), r)
    recurse(furthest(r, q.left, q.right), r)
  } else if (IsLeaf(q)==true and IsLeaf(r)==false) {
    /* choose the child that is closer to q  */
    /*  and recurse first */
    recurse(q, closest(q, r.left, r.right))
    recurse(q, furthest(q, r.left, r.right))
  } else {
    recurse(q.left,closest(q.left, r.left, r.right));
    recurse(q.left,furthest(q.left, r.left, r.right));
    recurse(q.right,closest(q.right, r.left, r.right));
    recurse(q.right,furthest(q.right, r.left, r.right));
  }
}
```

Figure 1: Pseudo-code for the dual-tree all nearest neighbor algorithm

4.6 The Dual Tree Algorithm for all furthest neighbor algorithm. Computing the furthest neighbor with the naive computation is also of quadratic complexity. The use of trees can speed up the computations too. It turns out that furthest neighbor search for a single query point is very similar to the nearest neighbor search presented in the original paper of kd-tree [7]. The only difference is that in the top-down recursion the algorithm always chooses the furthest node. Similarly in the bottom up recursion we prune a node only if the maximum distance between the point and the node is smaller than the current furthest distance. The pseudo code is presented in fig. 2.

```
recurse(q : KdTree, r : KdTree) {
  if (min_furthest_neighbor_distance_in_node(q)
        < distance(q, r) {
     /* prune */
  } else if (IsLeaf(q)==true and IsLeaf(r)==true) {
    /* search for every point in q node its
    /* furthest neighbor in the r node */
    /* at leaves we must resort to     */
    /* exhaustive search O(n^2) */
    /*update the minimum_furthest_distance_in_node(q)*/
  } else if (IsLeaf(q)==false and IsLeaf(r)=true {
    /*choose the child that is furthest to r */
    /* and recurse first */
    recurse(furthest(r, q.left, q.right), r)
    recurse(closest(r, q.left, q.right), r)
  } else if (IsLeaf(q)==true and IsLeaf(r)==false) {
    /* choose the child that is furthest to q  */
    /* and recurse first */
    recurse(q, furthest(q, r.left, r.right))
    recurse(q, closest(q, r.left, r.right))
  } else {
    recurse(q.left,furthest(q.left, r.left, r.right));
    recurse(q.left,closest(q.left, r.left, r.right));
    recurse(q.right,furthest(q.right, r.left, r.right));
    recurse(q.right,closest(q.right, r.left, r.right));
  }
}
```

Figure 2: Pseudo-code for the dual-tree all furthest neighbor algorithm

5 Experimental Results

In order to evaluate and compare the performance of IsoNMF with traditional NMF we picked 3 benchmark datasets that have been tested in the literature:

1. The CBCL faces database fig. 3(a,b) [12], used in the experiments of the original paper on NMF [19]. It consists of 2429 grayscale 19×19 images that they are hand aligned. The dataset was normalized as in [19].

2. The isomap statue dataset fig. 3(c) [13] consists of 698 64×64 synthetic face photographed from different angles. The data was downsampled to 32×32 with the Matlab *imresize* function (bicubic interpolation).

3. The ORL faces [14] fig. 3(d) presented in [11]. The set consists of 472 19×19 gray scale images that are not aligned. For visualization of the results we used the *nmfpack* code available on the web [15].

The results for classic NMF and IsoNmf with k-neighborhood equal to 3 are presented in fig. 4 and tables 1, 2. We observe that classic NMF gives always lower reconstruction error rates that are not that far away from the IsoNMF. Classic NMF fails to preserve distances contrary to IsoNMF that always does a good

job in preserving distances. Another observation is that IsoNMF gives more sparse solution than classic NMF. The only case where NMF has a big difference in reconstruction error is in the CBCL-face database when it is being preprocessed. This is mainly because the preprocessing distorts the images and spoils the manifold structure. If we don't do the preprocessing fig. 4(f), the reconstruction error of NMF and IsoNMF are almost the same.

In fig. 6 we see a comparison of the energy spectrums of classic NMF and IsoNMF. We define the spectrum as

$$s_i = \frac{\sum_{l=1}^{N} W_{li}^2}{\sqrt{\sum_{l=1}^{M} H_{il}^2}}$$

This represents the energy of the component normalized by the energy of the prototype image generated by NMF/IsoNMF. Although the results show that IsoNMF is much more compact than NMF, it is not a reasonable metric. This is because the prototypes (rows of the H matrix are not orthogonal to each other. So in reality $\sum_{i=1}^{k} s_i < \sum_{i=1}^{N} \sum_{j=1}^{m} (WH)_{ij}^2$ and actually much smaller. This is because the dot product between the rows is not zero.

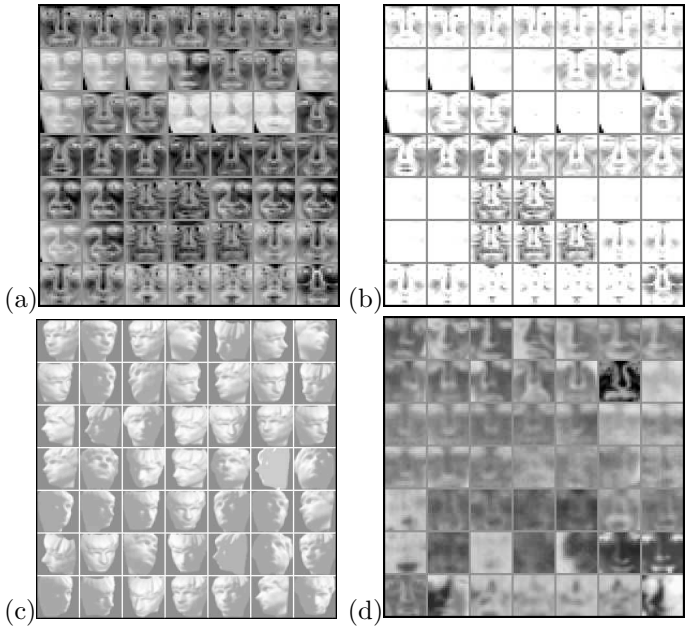


Figure 3: (a)Some images from the cbcl face database (b)The same images after variance normalization, mean set to 0.25 and thresholding in the interval [0,1] (c)The synthetic statue dataset from the isomap website [13] (d)472 images from the orl faces database [14]

classic NMF	cbcl norm.	cbcl	statue	orl
rec. error	22.01%	9.20%	13.62%	8.46%
sparsity	63.23%	29.06%	48.36%	46.80%
dist. error	92.10%	98.61%	97.30%	90.79%

Table 1: Classic NMF, the relative root mean square error, sparsity and distance error for the four different datasets (cbcl normalized and plain, statue and orl)

IsoNMF	cbcl norm.	cbcl	statue	orl
rec error	33.34%	10.16%	16.81%	11.77%
sparsity	77.69%	43.98%	53.84%	54.86%
dist. error	4.19%	3.07%	0.03%	0.01%

Table 2: IsoNMF, the relative root mean square error, sparsity and distance error for the four different datasets (cbcl normalized and plain, statue and orl)

6 Summary

In this paper we presented a study on the optimization schemes, convex and non-convex, global and local of Non Negative Matrix Factorization (NMF). Despite the existence of convex formulations there is no algorithm to solve them, so local non-convex optimizers are preferred. A global optimization scheme was presented too that outperforms local optimizers, but it cannot scale yet to larger matrices. We also presented a variant of NMF the isometric NMF (IsoNMF), that preserves local distances between points in the original dimensions. Our experiments on benchmark datasets indicate that IsoNMF except for maintaining the original distances also gives more sparse prototypes with the cost of a slightly higher reconstruction error. We also showed that if the dataset doesn't have a manifold structure then IsoNMF fails.

References

[1] M. Belkin and P. Niyogi. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, 2003.

[2] A. Berman and N. Shaked-Monderer. *Completely Positive Matrices*. World Scientific, 2003.

[3] S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

[5] R.R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[6] C.A. Floudas. *Deterministic Global Optimization:*

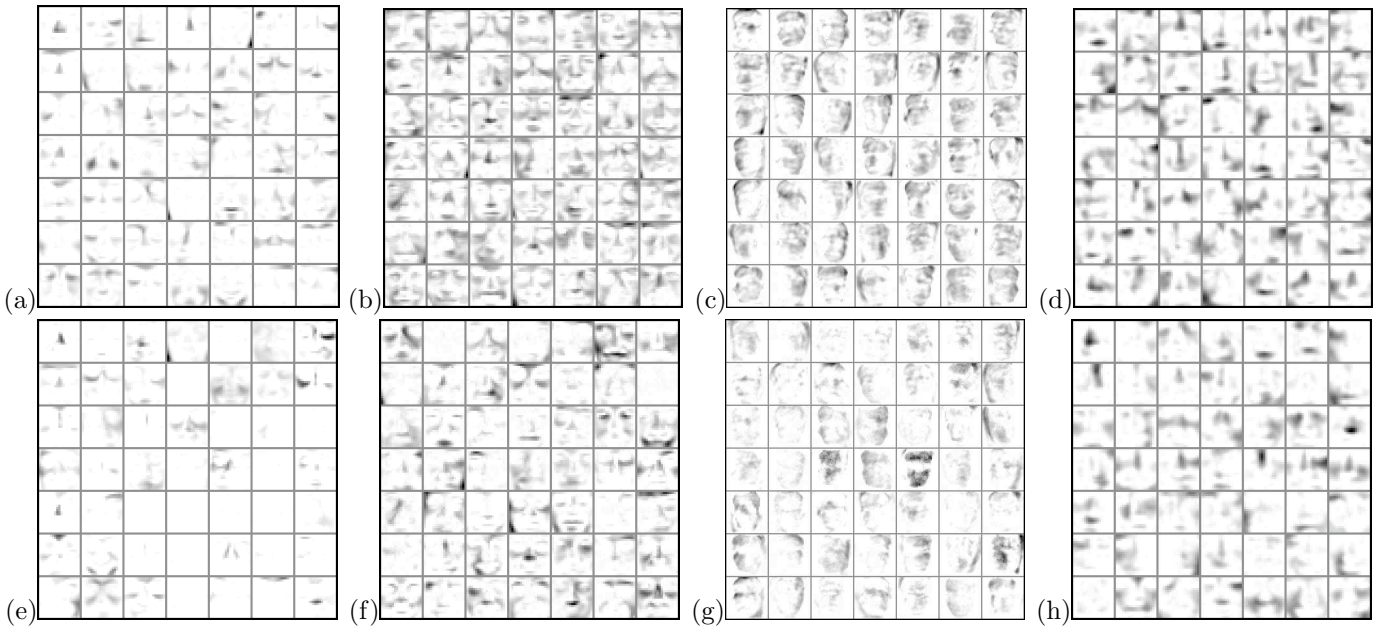


Figure 4: Top row: 49 Classic NMF prototype images. Bottom row: 49 IsoNMF prototype images (a, e) CBCL-face database with mean variance normalization and thresholding, (b, f) CBCL face database without preprocessing, (c, g) Statue dataset (d, h)ORL face database

Theory, Methods and Applications. Kluwer Academic Pub, 2000.

[7] J.H. Friedman, J.L. Bentley, and R.A. Finkel. An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

[8] A. Gray and A.W. Moore. N-Body problems in statistical learning. *Advances in Neural Information Processing Systems*, 13, 2001.

[9] An Le Thi Hoai and P.D. Tao. DC Programming Approach and Solution Algorithm to the Multidimensional Scaling Problem.

[10] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer, 1996.

[11] P.O. Hoyer. Non-negative Matrix Factorization with Sparseness Constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[12] http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html.

[13] http://isomap.stanford.edu/face_data.mat.Z.

[14] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[15] http://www.cs.helsinki.fi/u/phoyer/software.html.

[16] M. Kaykobad. On nonnegative factorization of matrices. *Linear Algebra and its Applications*, 96:27–33, 1987.

[17] H. Kim and H. Park. Non-Negative Matrix Factorization Based on Alternating Non-Negativity Constrained Least Squares and Active Set Method.

[18] A.N. Langville, C.D. Meyer, and R. Albright. Initializations for the nonnegative matrix factorization. *Proc. of the 12 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[19] D.D. Lee and H.S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[20] D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

[21] H. Minc. *Nonnegative matrices*. Wiley New York, 1988.

[22] D. Anderson N. Vasiloglou, A. Gray. Scalable Semidefnite Manifold Learning. *IEEE Machine Learning in Signal Processing*, 2008.

[23] Nemirovski A. *Lectures on Modern Convex Optimization*.

[24] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 1999.

[25] B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization. *Arxiv preprint arXiv:0706.4138*, 2007.

[26] P.D. Tao and L.T.H. An. Difference of convex functions optimization algorithms (DCA) for globally minimizing nonconvex quadratic forms on Euclidean balls and spheres. *Operations Research Letters*, 19(5):207–216, 1996.

[27] J.B. Tenenbaum, V. Silva, and J.C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction, 2000.

[28] L. Vandenberghe and S. Boyd. Semidefinite Programming. *SIAM Review*, 38(1):49–95, 1996.

[29] K.Q. Weinberger and L.K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. *Proceedings of the Twenty First National Conference on Artificial Intelligence (AAAI-06)*, 2006.

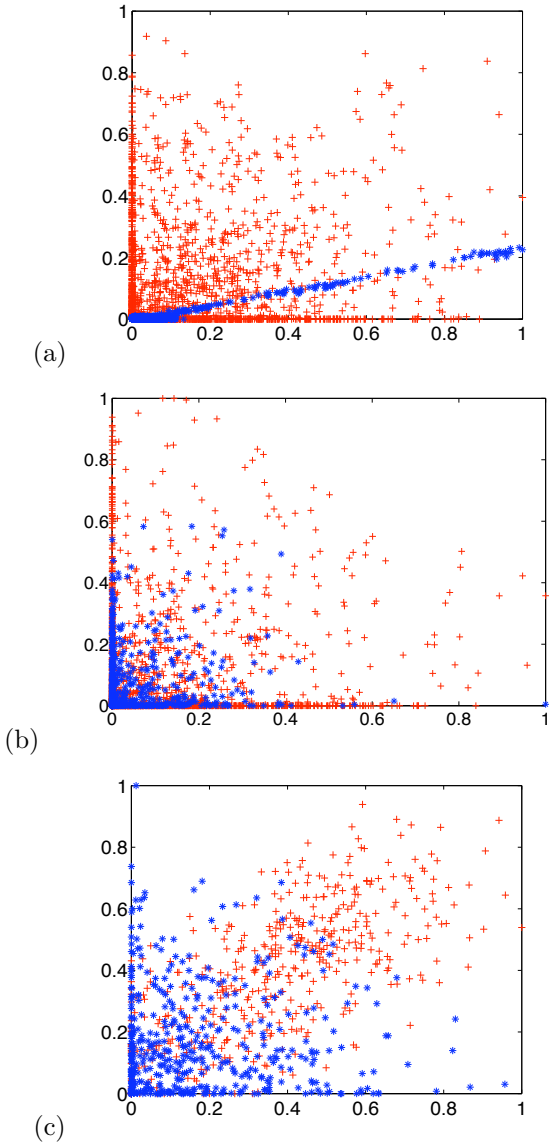
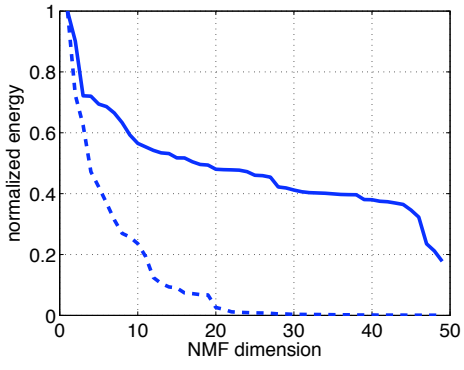
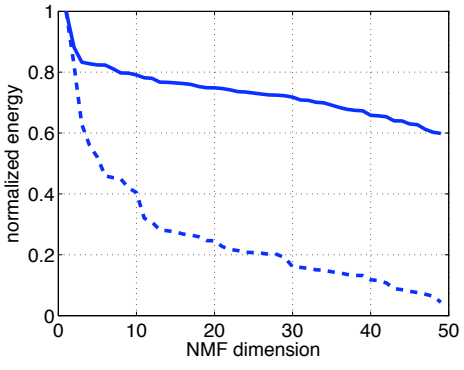


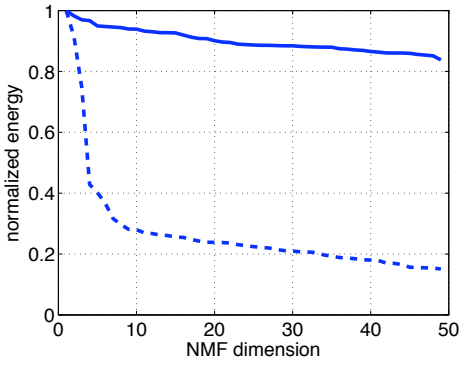
Figure 5: Scatter plots of two largest components of classic NMF(in blue) and Isometric NMF(in red) for (a)cbcl faces (b)isomap faces (c)orl faces



(a)



(b)



(c)

Figure 6: In this set of figures we show the spectrum of classic NMF (solid line) and Isometric NMF (dashed line) for the three datasets (a)cbcl face (b)isomap statue (c)orl faces. Although IsoNMf gives much more compact spectrum we have to point that the basis functions are not orthogonal, so this figure is not comparable to SVD type spectrums