PARALLEL VECTOR FIELD REGULARIZED NON-NEGATIVE MATRIX FACTORIZATION FOR IMAGE REPRESENTATION

Yong Peng*, Rixin Tang, Wanzeng Kong, Feiwei Qin

School of Computer Science and Technology
Hangzhou Dianzi University
Hangzhou 310018, China

Feiping Nie

Center for OPTIMAL
Northwestern Polytechnical University
Xi'an 710072, China

ABSTRACT
Non-negative Matrix Factorization (NMF) is a popular model in machine learning, which can learn parts-based representation by seeking for the two non-negative matrices whose product can best approximate the original matrix. However, the manifold structure is not considered by NMF and many of the existing work use the graph Laplacian to ensure the smoothness of the learned representation coefficients on the data manifold. Further, beyond smoothness, it is suggested by recent theoretical work that we should ensure second order smoothness for the NMF mapping, which measures the linearity of the NMF mapping along the data manifold. Based on the equivalence between the gradient field of a linear function and a parallel vector field, we propose to find the NMF mapping which minimizes the approximation error, and simultaneously requires its gradient field to be as parallel as possible. The continuous objective function on the manifold can be discretized and optimized under the general NMF framework. Extensive experimental results suggest that the proposed parallel field regularized NMF provides a better data representation and achieves higher accuracy in image clustering.

Index Terms— Non-negative matrix factorization, Vector field, Image representation, Clustering

1. INTRODUCTION
Matrix factorization is a class of effective methods to learn data representation. Among existing matrix factorization methods such as LU-decomposition, QR-decomposition, Cholesky decomposition, and Singular Value Decomposition (SVD), the Non-negative Matrix Factorization (NMF) is very popular one which can learn parts of objects like human faces and text documents [1, 2]. Though the parts-based representation have got physiological evidence from previous studies [3] and shown great performance in face recognition [4] and document clustering [5], NMF fails to consider the geometrical structure of the data space which has been proven to be essential for data clustering and classification. To compensate the limitation, Cai et al. proposed to encode the geometrical information of the data space by constructing a nearest neighbor graph and formulated the Graph regularized Non-negative Matrix Factorization (GNMF) model [6, 7], which shows superior performance to NMF. However, such graph Laplacian-based method only enforces the smoothness of data representation on data manifold in first order manner. Based on the conclusions that 1) the second smoothness measures the linearity of the mapping function, 2) the gradient field of a linear function has to be a parallel vector field, which were shown in [8, 9], we propose a novel matrix factorization algorithm, called Parallel Field Regularized NMF (PFNMF), which enforces the learned codes to vary linearly along the data manifold. Specifically, we firstly estimate parallel vector fields on manifold from data samples, and then learn the codes by requiring the gradient fields of NMF mapping are as close to the estimated parallel fields as possible.

The remainder of this paper is organized as follows. Section 2 introduces the formulation of the proposed PFNMF model and its optimization. Experimental setting and results are presented in Section 3. Section 4 concludes the paper.

2. PARALLEL FIELD REGULARIZED NMF

2.1. NMF
NMF is a matrix factorization algorithm that focuses on analysis of data matrices whose elements are non-negative. Given a data matrix $X = [x_1, \cdots, x_n] \in \mathbb{R}^{m \times n}$, NMF aims at finding two non-negative matrices $U = [u_{ik}] \in \mathbb{R}^{m \times K}$ and $H = [h_{jk}] \in \mathbb{R}^{n \times K}$, whose product can approximate $X$. Based on the square of the Euclidean distance between two matrices, the objective function of NMF can be formulated as

$$
\mathcal{O}_\text{NMF} = \|X - U H^T\|^2.
$$

(1)

The $K$ columns of $U$ are basis vectors, and each column of $H$ is an encoding of a sample vector in $X$ and is an one-to-one mapping. The non-negative property enforced on both $U$ and
allows only addictive combinations among different bases, which makes NMF learn parts-based representation.

2.2. Parallel Fields and Linear Functions

In geometry and vector calculus, a vector field is a mapping from a manifold \( \mathcal{M} \) to tangent spaces [10]. As shown in [8], the parallel vector field has close relationship to a linear function on the manifold, which can be described below.

**Definition 1.** (Parallel Field [10]). A vector field \( X \) on manifold \( \mathcal{M} \) is a parallel field if \( \nabla X \equiv 0 \), where \( \nabla \) is the covariant derivative on \( \mathcal{M} \).

**Definition 2.** (Linear Function [10]). A continuous function \( f : \mathcal{M} \to \mathbb{R} \) is said to be linear if \( (f \circ \gamma)(t) = f(\gamma(0)) + ct \) for each geodesic \( \gamma \).

In this paper, a function \( f \) is linear means that it varies linearly along the geodesics of the manifold. This definition is a natural extension of linear functions on Euclidean space. The following proposition reveals the relationship between a parallel vector field and a linear function on the data manifold. We will not strictly distinguish between the concepts of covariant derivative and gradient field in this paper.

**Proposition 1.** [10] Let \( V \) be a parallel field on the manifold. If it is also a gradient field for function \( f \), \( V = \nabla f \), then \( f \) is a linear function on the manifold.

2.3. PFNMF Model Formulation

As discussed before, we aim to ensure the linearity of NMF embedding with respect to the data manifold, which is equivalent to the parallelism of the gradient field of the NMF mapping. However, it is difficult to design constraints on the gradient field of a mapping function directly [8]. Therefore, we propose to 1) learn vector fields on manifold from data samples to approximate the gradient field of the NMF mapping function, and encourage the vector fields to be as parallel as possible; 2) learn NMF embedding while enforcing the encodes to be as close to the estimated parallel fields as possible.

Since the manifold \( \mathcal{M} \) is unknown, the mapping function in NMF \( f_k(x_i) = b_{jk} \equiv f_j^{(k)}(x_i), j = 1, \ldots, n \) has no explicit form. Following [8], we introduce how to estimate the tangent space of each data point, which is important for discretizing the continuous objective function form when estimating parallel vector fields and learning NMF embedding.

Let \( W \) be the corresponding affinity matrix of graph \( \mathcal{G} \) and \( W \) is simply defined by 0-1 weight. Then, for each \( x_i \), we can estimate its tangent space \( T_{x_i} \mathcal{M} \) by performing PCA on its local neighborhood. We choose the eigenvectors corresponding to the \( d \) largest eigenvalues since \( T_{x_i} \mathcal{M} \) is \( d \)-dimensional. Let \( T_i \in \mathbb{R}^{m \times d} \) be the matrix whose columns constitute an orthogonal basis for \( T_{x_i} \mathcal{M} \). It is easy to show that \( P_i = T_i T_i^T \) is the unique orthogonal projection from \( \mathbb{R}^m \) onto the tangent space \( T_{x_i} \mathcal{M} \). That is, for any vector \( a \in \mathbb{R}^m \), we have \( P_i a \in T_{x_i} \mathcal{M} \) and \((a - P_i a) \perp P_i a \).

**Estimating the Parallel Vector Field.** Firstly, we try to find vector fields which are as parallel as possible on the manifold. Let \( V \) be a smooth vector field on \( \mathcal{M} \). By definition, the covariant derivative of \( V \) should be zero. That is, \( \nabla V \equiv 0 \). Naturally, the objective for estimating parallel vector field is

\[
E(V) = \int_{\mathcal{M}} \| \nabla V \|_{\text{HS}}^2 dx, \tag{2}
\]

where \( \| \cdot \|_{\text{HS}} \) denotes Hilbert-Schmidt tensor norm [11].

For each point \( x_i \), let \( V_{x_i} \) denote the value of the vector field \( V \) at \( x_i \), and \( \nabla V_{x_i} \) denote the value of \( \nabla V \) at \( x_i \). According to its definition, \( V_{x_i} \) should be a vector in the tangent space \( T_{x_i} \mathcal{M} \). Therefore, it can be represented by the local coordinates of the tangent space, \( V_{x_i} = T_i v_i \), where \( v_i \in \mathbb{R}^d \).

By discretizing (2), the parallel field \( V \) can be obtained by solving the following optimization problem (Here we use the notation \( V \) instead of \( V \) while the specific definition of \( V \) will be given next):

\[
\min_V E(V) = \sum_{i=1}^n \| \nabla V_{x_i} \|_{\text{HS}}^2. \tag{3}
\]

Then according to the analysis in [8], the optimization problem of (3) reduces to

\[
\min_V E(V) = \sum_{i,j=1}^n w_{ij} \| P_i T_j v_j - T_i v_i \|_2^2. \tag{4}
\]

The compact form of \( E(V) \) can be written as \( E(V) = \sum_{i,j=1}^n w_{ij} (Q_{ij}^2 + I_d) \).

\[
B = \begin{pmatrix} B_{11} & \cdots & B_{1n} \\ \vdots & \ddots & \vdots \\ B_{n1} & \cdots & B_{nn} \end{pmatrix}, \tag{5}
\]

where \( B \) is a \( dn \times dn \) sparse block matrix. If we index each \( d \times d \) block by \( B_{ij} \), then for \( j = 1, 2, \ldots, n \), we have

\[
B_{jj} = \sum_{x_i \in N_k(x_j)} w_{ij} (Q_{ij}^2 + I_d), \tag{6}
\]

\[
B_{ij} = \begin{cases} -2w_{ij}Q_{ij}, & \text{if } x_i \sim x_j \\ 0, & \text{otherwise}. \end{cases} \tag{7}
\]

Obviously, \( V \) is a \( dn \)-dimensional big column vector concatenating all the \( v_i \)’s.

We impose a global normalization constraint \( \| V \|_2^2 = 1 \) on vector field; thus, the Lagrangian function w.r.t. \( E(V) \) is

\[
\mathcal{L}_{E(V)} = E(V) - \lambda (V^T V - 1). \tag{8}
\]

The parallel vector field \( V \) can be estimated by solving the following eigenvalue decomposition problem

\[
BV = \lambda V. \tag{9}
\]

The multiple vector fields are eigenvectors corresponding to the smallest eigenvalues of the matrix \( B \). Specifically, the dimensionality of learned NMF embedding is \( K \); thus, we use \( V \in \mathbb{R}^{dn \times K} \) to denote the stack of \( K \) eigenvectors.

**Learning NMF Embedding.** Once the parallel vector fields \( V_i \) are obtained, the embedding functions \( f^{(k)} : \mathcal{M} \to \mathbb{R} \) can be constructed by requiring their gradient fields to be as close as \( V_i \) as possible. This can be attained in the least square sense and thus minimizing the following objective

\[
R(f^{(k)}) = \int_{\mathcal{M}} \| \nabla f^{(k)} - V \|^2 dx \tag{10}
\]
Based on the analysis in [8], \( R(f(k)) \) can be discretized as
\[
R(f(k)) = \sum_{i,j=1}^{n} w_{ij} ((x_j - x_i)^2 T_i v_i - f_j + f_i)^2. \quad (11)
\]

By incorporating this as a regularizer into the objective of NMF, we obtain the objective of PFNMF as
\[
O_{PFNMF} = \|X - U H^T\|^2 + \alpha \sum_k R(f(k)) \quad (12)
\]

Let \( L \) be the graph Laplacian. Then we have
\[
R(f(k)) = 2 f(k)^T L f(k) + \sum_{i,j=1}^{n} w_{ij} ((x_j - x_i)^2 T_i v_i - f_j + f_i)^2 - 2 \sum_{i,j=1}^{n} w_{ij} (x_j - x_i)^T T_i v_i s_{ij} f(k),
\]
where \( s_{ij} \in \mathbb{R}^n \) is a selection vector of all zero elements except for the \( i \)-th element being -1 and the \( j \)-th element being 1. We further construct a \( dn \times dn \) block diagonal matrix \( G \), and a \( dn \times n \) block matrix \( C = [C_1^T, C_2^T, \ldots, C_n^T]^T \). Let \( G_{ii} \) denote the \( i \)-th \( d \times d \) diagonal block of \( G \), and \( C_i \) denote the \( i \)-th \( d \times n \) block of \( C \), we define:
\[
G_{ii} = \sum_{j=1}^{n} w_{ij} T_j (x_j - x_i)^T T_i, \quad C_i = \sum_{j=1}^{n} w_{ij} T_j (x_j - x_i) s_{ij}.
\quad (13)
\]

With simple algebra transformations, it is easy to check that \( R(f(k)) \) can be rewritten as follows:
\[
R(f(k)) = 2 f(k)^T L f(k) + \nu^T G \nu - 2 \nu^T C f(k). \quad (14)
\]

Similar to the basic NMF, we need to minimize \( O_{PFNMF} \) by updating one variable while fixing the others.

Let \( \psi_{ik} \) and \( \phi_{jk} \) be the Lagrange multipliers for constraints \( u_{ik} \geq 0 \) and \( h_{jk} \geq 0 \). We define matrix \( \Psi = [\psi_{ik}] \) and \( \Phi = [\phi_{jk}] \), then the Lagrangian function \( L_{PFNMF} \) is
\[
L_{PFNMF} = O_{PFNMF} + \sum_{ik} \psi_{ik} u_{ik} + \sum_{jk} \phi_{jk} h_{jk}.
\quad (15)
\]

Fixing \( H \), the partial derivative of \( L_{PFNMF} \) w.r.t. \( U \) is:
\[
\frac{\partial L_{PFNMF}}{\partial U} = -2 (X - U H^T) H + \Psi.
\]

Using the KKT condition \( \psi_{ik} u_{ik} = 0 \), we can get
\[- (X H)_{ik} u_{ik} + (U H^T H)_{ik} u_{ik} = 0, \]
which leads to the following updating rule for \( u_{ik} \):
\[
u_{ik} \leftarrow u_{ik} (X H)_{ik} / (U H^T H)_{ik}. \quad (16)
\]

Based on the NMF definition, the mapping function \( F = \{ f^{(1)}, f^{(2)}, \ldots, f^{(K)} \} \in \mathbb{R}^{n \times K} \) is defined as \( f^{(k)} = h_{ik} \in \mathbb{R}^{n \times 1} \), \( f(k) = h_{ik} \) and thus we have
\[
\frac{\partial L_{PFNMF}}{\partial H} = -2 (X - U H^T) U + \alpha R / \partial F + \Phi
\]
\[
= -2 X^T U + 2 H U^T U + 4 \alpha L H - 2 \alpha C^T \tilde{V} + \Phi
\]

Using KKT conditions \( \phi_{jk} h_{jk} = 0 \), we have
\[- (X^T U)_{jk} h_{jk} + (H U^T)_{jk} h_{jk} + 2 \alpha L H)_{jk} h_{jk} - \alpha (C^T \tilde{V})_{jk} h_{jk} = 0. \quad (17)
\]

Since NMF only allows additive combinations, we decompose \( L \) and \( V \) as \( L = D - W \) and \( V = \tilde{V} + \tilde{V} - \tilde{V} + \tilde{V} = (|V| + \tilde{V}) / 2, \tilde{V} = (|V| - \tilde{V}) / 2 \), then we obtain the following updating rule for \( h_{jk} \):
\[
h_{jk} \leftarrow h_{jk} \left( (X^T U + \alpha C^T \tilde{V} + 2 \alpha W H)_{jk}
\quad (17)
\]

We can easily find that PFNMF will boil down to NMF when \( \alpha = 0 \) and GNMF [7] if we do not consider the vector fields \( \tilde{V} \).

The workflow of PFNMF is summarized in Algorithm 1.

### Algorithm 1 PFNMF

**Input:** Data samples \( X = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^{m \times n} \) and \( \alpha \);
**Output:** The basis matrix \( U \) and coefficient matrix \( H \).

for \( i = 1 \) to \( n \) do

Compute tangent spaces \( T_x \mathcal{M} \) for each data sample by performing PCA on neighborhood of \( x_i \);
end for

Construct matrix \( B \) according to (7);
Do eigen-decomposition on (9) to estimate \( \tilde{V} \);
// Iteratively optimize PFNMF model
while not converged do

Update \( U \) according to (16) with \( H \) fixed;
Update \( H \) according to (17) with \( U \) fixed;
end while

### 3. EXPERIMENTS

We investigate the effectiveness of PFNMF on image clustering.

#### 3.1. Data Sets and Experimental Settings

**Data Sets.** Three data sets are used in this experiment. The important statistics of these data sets are summarized below (see also Table 1). **COIL20.** It contains 20 objects. The images of each object were taken 5 degrees apart as the object is rotated on a turntable and each object has 72 images. The size of each image is 32×32 pixels, with 256 gray levels per pixel. Thus, each image is represented by a 1024-dimensional vector. **ORL.** There are ten different images of each of 40 distinct subjects. All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position. We crop the original 112×92 images into 64×64 gray scale images. **CMU PIE.** It contains 32×32 gray scale images of 68 subjects. Each person has 42 facial images under different light and illumination conditions.

**Evaluation Metrics.** The clustering result is evaluated by comparing the obtained label of each sample with the label provided by the data set. Two standard clustering metrics, the accuracy (ACC) and normalized mutual information (NMI), are used to measure the clustering performance.

**Table 1.** Properties of the used data sets.

<table>
<thead>
<tr>
<th>dataset</th>
<th>size</th>
<th>dimensionality</th>
<th>#class</th>
</tr>
</thead>
<tbody>
<tr>
<td>COIL20</td>
<td>1440</td>
<td>1024</td>
<td>20</td>
</tr>
<tr>
<td>ORL</td>
<td>400</td>
<td>4096</td>
<td>40</td>
</tr>
<tr>
<td>PIE</td>
<td>2856</td>
<td>1024</td>
<td>68</td>
</tr>
</tbody>
</table>

#### 3.2. Clustering Results

To show the improvement of the clustering performance by our method, we compared PFNMF with the following three popular algorithms: K-means, NMF [1], and GNMF [7]. The
evaluations were conducted with different numbers of clusters. For each given cluster numbers, 20 test runs were conducted on different randomly chosen clusters. The final performance is recorded by averaging the performance of the 20 tests. For fair comparison, we record the randomly selected cluster indices, and fix them for all competing algorithms.

Table 2. Clustering performance on COIL20.

<table>
<thead>
<tr>
<th>K</th>
<th>Accuracy (mean±std-dev%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>84.11±12.13 79.77±14.26 92.73±11.98 95.97±8.74</td>
</tr>
<tr>
<td>6</td>
<td>78.07±12.16 75.53±11.87 86.66±12.67 92.82±8.20</td>
</tr>
<tr>
<td>8</td>
<td>72.96±10.23 72.06±9.64 90.43±7.75 95.29±6.48</td>
</tr>
<tr>
<td>10</td>
<td>68.87±6.17 68.89±8.88 81.41±7.58 87.44±6.51</td>
</tr>
<tr>
<td>12</td>
<td>62.80±4.02 67.87±5.45 79.13±5.96 86.17±5.32</td>
</tr>
<tr>
<td>14</td>
<td>67.11±5.65 66.27±4.60 82.49±4.61 84.67±4.09</td>
</tr>
<tr>
<td>16</td>
<td>65.23±4.42 65.56±4.69 79.09±4.10 80.90±3.72</td>
</tr>
<tr>
<td>18</td>
<td>62.74±3.85 63.05±3.65 78.97±3.49 80.41±4.29</td>
</tr>
<tr>
<td>20</td>
<td>60.49 58.37 80.69 85.14</td>
</tr>
<tr>
<td>Avg</td>
<td>69.75 68.65 83.31 87.65</td>
</tr>
</tbody>
</table>

Table 3. Clustering performance on ORL.

<table>
<thead>
<tr>
<th>K</th>
<th>Accuracy (mean±std-dev%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>75.03±16.69 70.73±17.67 90.32±13.14 94.55±10.96</td>
</tr>
<tr>
<td>6</td>
<td>74.83±12.79 71.99±11.91 87.91±8.84 92.54±7.75</td>
</tr>
<tr>
<td>8</td>
<td>74.23±7.65 72.02±7.57 91.29±5.72 95.36±5.23</td>
</tr>
<tr>
<td>10</td>
<td>72.68±5.95 71.72±7.49 86.80±5.07 90.41±4.68</td>
</tr>
<tr>
<td>12</td>
<td>73.22±3.25 72.36±3.95 86.83±3.43 90.85±3.41</td>
</tr>
<tr>
<td>14</td>
<td>74.19±3.93 72.92±3.75 89.29±2.80 91.33±2.51</td>
</tr>
<tr>
<td>16</td>
<td>73.88±2.71 72.61±3.38 88.36±2.10 89.73±2.58</td>
</tr>
<tr>
<td>18</td>
<td>73.25±2.54 72.28±2.41 88.36±1.57 89.50±1.61</td>
</tr>
<tr>
<td>20</td>
<td>73.86 71.51 89.12 90.50</td>
</tr>
<tr>
<td>Avg</td>
<td>73.91 72.02 88.70 91.64</td>
</tr>
</tbody>
</table>

Table 4. Clustering performance on PIE.

<table>
<thead>
<tr>
<th>K</th>
<th>Accuracy (mean±std-dev%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>31.15±4.00 56.62±5.16 85.01±9.28 91.39±7.38</td>
</tr>
<tr>
<td>20</td>
<td>27.20±1.99 55.32±4.46 80.69±6.20 83.72±5.64</td>
</tr>
<tr>
<td>30</td>
<td>26.35±2.12 58.87±2.18 81.60±3.08 84.19±3.51</td>
</tr>
<tr>
<td>40</td>
<td>25.34±1.31 59.96±3.37 77.51±3.84 79.81±3.18</td>
</tr>
<tr>
<td>50</td>
<td>24.62±1.08 58.27±2.10 76.59±3.90 78.83±4.41</td>
</tr>
<tr>
<td>60</td>
<td>24.17±1.10 56.64±2.42 74.69±2.89 76.34±3.29</td>
</tr>
<tr>
<td>80</td>
<td>24.54 56.79 70.52 73.01</td>
</tr>
<tr>
<td>Avg</td>
<td>26.30 55.92 78.09 81.04</td>
</tr>
</tbody>
</table>

Outperforms its counterpart GNMF by around 2%–3%, which means that the data representation learned by PFNMF show more respect to the underlying data manifold. The linearity of image data representation emphasized by the second order smoothness is beneficial to clustering task.

3.3. Basis Vectors and Image Encodings

In this test, we randomly select 25 subjects from the ORL data set and for each subject we randomly select 5 face images. Figure 1 shows the basis vectors respectively learned by NMF and PFNMF. Comparing the basis images obtained by PFNMF with the original face images. The basis learned by PFNMF is obviously much clearer than that learned by N-MF, which will naturally lead to better face image encodings for clustering.

![Fig. 1. Basis vectors learned from the ORL data set. Large values are illustrated with white pixels.](image)

4. CONCLUSIONS

We have presented a novel method for matrix factorization algorithm to enforce the second order smoothness of data representation called PFNMF. Experiments show that our proposed method performs better than other comparison methods in image clustering. In the future, we will extend our method to deal with data in noisy environments and thus develop a robust version.
5. REFERENCES


[6] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han, “Non-negative matrix factorization on manifold,” in *IEEE International Conference on Data Mining*, 2008, pp. 63–72.

[7] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang, “Graph regularized nonnegative matrix factorization for data representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2011.


