Efficient Orthogonal Non-negative Matrix Factorization over Stiefel Manifold

Wei Emma Zhang*, Mingkui Tan†*, Quan Z. Sheng*,Lina Yao‡, Qingfeng Shi*
*School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia
{wei.zhang01,michael.sheng, javen.shi}@adelaide.edu.au
†School of Software Engineering, South China University of Technology, Guangzhou, China, 510006
tanmingkui@gmail.com
‡School of Computer Science and Engineering, UNSW Australia, Sydney, NSW 2052, Australia
lina.yao@unsw.edu.au

ABSTRACT

Orthogonal Non-negative Matrix Factorization (ONMF) approximates a data matrix $\mathbf{X}$ by the product of two lower-dimensional factor matrices: $\mathbf{X} \approx \mathbf{U}\mathbf{V}^\mathsf{T}$, with one of them orthogonal. ONMF has been widely applied for clustering, but it often suffers from high computational cost due to the orthogonality constraint. In this paper, we propose a method, called Nonlinear *Riemannian* Conjugate Gradient ONMF (NRCG-ONMF), which updates $\mathbf{U}$ and $\mathbf{V}$ alternatively and preserves the orthogonality of $\mathbf{U}$ while achieving fast convergence speed. Specifically, in order to update $\mathbf{U}$, we develop a Nonlinear *Riemannian* Conjugate Gradient (NRCG) method on the *Stiefel* manifold using Barzilai-Borwein (BB) step size. For updating $\mathbf{V}$, we use a closed-form solution under non-negativity constraint. Extensive experiments on both synthetic and real-world data sets show consistent superiority of our method over other approaches in terms of orthogonality preservation, convergence speed and clustering performance.

Keywords

Orthogonal NMF; *Stiefel* Manifold; Clustering

1. INTRODUCTION

Given a matrix, Non-negative Matrix Factorization (NMF) aims to find two non-negative factor matrices whose product approximates that matrix. This enhances the interpretability initiated by Paatero and Tapper [25] as Positive Matrix Factorization and popularized by Lee and Seung [20]. NMF has enjoyed much success in text mining [26], image processing [11], recommendation systems [19] and many other areas, and has attracted much theoretical and practical attention.

*corresponding author.

CIKM'16 , October 24-28, 2016, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/2983323.2983761

Orthogonal NMF (ONMF), first introduced by Ding et al. [9], is a variant of NMF with an additional orthogonal constraint on one of the factor matrices. Without loss of generality, the problem can be written as follows:

$$\min_{\mathbf{U},\mathbf{V}} ||\mathbf{X} - \mathbf{U}\mathbf{V}^\mathsf{T}||_F^2, \text{ s.t. } \mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}_r, \mathbf{U} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}, \quad (1)$$

where $\mathbf{X}$ is an $m \times n$ matrix, $\mathbf{U} \in \mathbb{R}^{m\times r}$ and $\mathbf{V} \in \mathbb{R}^{n\times r}$ are two factor matrices, and $||\cdot||_F$ is the *Frobenius* norm. In Problem (1), $\mathbf{I}_r$ is an identity matrix of size $r \times r$. The factor matrix $\mathbf{U}$ is imposed on the orthogonal constraint. ONMF has been shown to be identical to k-means when the *Frobenius* norm is used as divergence/distance [9]. Thus one factor matrix is corresponding to the cluster centers and the other matrix is corresponding to the cluster membership indicators [7]. The orthogonality on the columns of $\mathbf{U}$ is identical to clustering rows of the input data matrix $\mathbf{X}$ [32] and makes the clusters more distinct than without orthogonal constraint [21]. Moreover, the orthogonal constraint reduces computation complexity of NMF approaches [34]. Due to these properties, ONMF becomes increasingly popular in clustering tasks [32]. Most existing ONMF methods either enforce the orthogonality directly on the factor matrix [9, 6] as constraints or in the objective function [21]. Besides that, the authors in [26] propose a projection gradient method leveraging the manifold constraint. Very recently, people try to approximate ONMF problem by the Non-negative Principle Component Analysis (NNPCA) problem [2] gaining good results. However these methods do not preserve orthogonality well (except for the work in [26]) and/or suffer from slow convergence (Section 5.4.2).

Another branch of approaches to improve clustering interpretation on NMF is to relax the non-negativity constraint on one of the factor matrices. Ding et al. [8] refer to this constrained NMF as Semi-NMF and prove its applicability in the clustering perspective where one factor matrix represents the cluster centroids and the other represents soft membership indicators for every data point.

However, limited works consider both orthogonality and relaxed non-negativity to improve the clustering performance of NMF. In this work, we formalize the problem as follows:

$$\min_{\mathbf{U},\mathbf{V}} ||\mathbf{X} - \mathbf{U}\mathbf{V}^\mathsf{T}||_F^2 + \frac{\lambda}{2}||\mathbf{U}\mathbf{V}^\mathsf{T}||_F^2, \text{ s.t. } \mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}_r, \mathbf{V} \geq \mathbf{0}, \quad (2)$$

where $\mathbf{U}$ is with orthogonality constraint only and λ is the

regularization parameter. The regularizer $\frac{1}{2}||\mathbf{U}\mathbf{V}^\mathsf{T}||_F^2$ is introduced to avoid over-fitting issue.

To solve Problem (2), we propose a Nonlinear *Riemannian* Conjugate Gradient ONMF (NRCG-ONMF), which updates $\mathbf{U}$ and $\mathbf{V}$ alternatively and iteratively: when updating $\mathbf{U}$ (resp., $\mathbf{V}$), $\mathbf{V}$ (resp., $\mathbf{U}$) is fixed. Specifically, when updating $\mathbf{U}$, a Nonlinear *Riemannian* Conjugate Gradient method (NRCG) is proposed. This method preserves the orthogonality of $\mathbf{U}$ in the setting of *Stiefel* manifold which is an embedded sub-manifold of the *Riemannian* manifold [1]. The NRCG method performs nonlinear search on *Stiefel* manifold by three steps: i) identifying the search direction on the tangent space of *Stiefel* manifold regarding $\mathbf{U}$ with Conjugate Gradient (CG) rules, ii) moving the point along the search direction with carefully determined step size, and iii) projecting the new point back to the *Stiefel* manifold to preserve the orthogonality with a *Retraction* operation. When updating $\mathbf{V}$, due to the nonnegative constraint, we update each entry of $\mathbf{V}$ separately in a coordinate descent manner by closed-form solutions.

The main contributions of our work are summarized as follows.

- We exploit the orthogonality and semi-nonnegativity constraints in NMF aiming at improving the clustering performance. We develop an efficient nonlinear *Riemannian* conjugate gradient method with Barzilai-Borwein (BB) step size to update $\mathbf{U}$ in order to preserve the orthogonality of $\mathbf{U}$ with fast convergence speed.
- Based on the NRCG method, we propose a NRCG-ONMF method to address Problem (2), where we update $\mathbf{U}$ and $\mathbf{V}$ alternatively. The convergence of NRCG-ONMF is also analyzed.
- Extensive experiments on both synthetic and real-world data sets show that our method outperforms the related works in terms of orthogonality, convergence speed and clustering performance.

The rest of this paper is organized as follows. We first provide the notations and preliminaries in Section 2. Then we introduce our proposed algorithm in Section 3. The experiments are presented in Section 5 followed by related works in Section 4 and conclusion in Section 6.

2. NOTATIONS AND PRELIMINARIES

2.1 Notations

Throughout the paper, we use bold uppercase and lowercase letters to represent the matrices and vectors respectively. We denote by the superscript $^\mathsf{T}$ the transpose of a vector/matrix. The *Frobenius* norm of $\mathbf{X}$ is defined as $||\mathbf{X}||_F$. The list of main notations can be found in Table 1.

2.2 Optimization on Stiefel Manifold

To begin with, we introduce some geometries regarding *Stiefel* manifold. The *Stiefel* manifold, denoted by St_r^m, is defined as

$$\mathrm{St}_r^m = \{\mathbf{U} \in \mathbb{R}^{m\times r} | \mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}_r\},$$

which is a set of m by r orthonormal matrices [1]. The *Stiefel* manifold is compact, and its dimension is $(mr - \frac{1}{2}r(r+1))$.

Table 1: Notation conventions

Notations	Descriptions
$\mathbf{U}, \mathbf{V}, \mathbf{G}, \mathbf{X}, \mathbf{Z}, \mathbf{I}, \mathbf{E}$	Matrices
$\boldsymbol{\eta}_k, \mathbf{x}_i, \mathbf{b}_j, \mathbf{v}_i$	Vectors
$\mathcal{M}_r, \mathrm{St}_r^m$	Manifolds
$T_\mathbf{U}\mathcal{M}_r$	Tangent spaces
$\lambda, \alpha_k, \beta_k, \delta, \epsilon, w_i, s_{jl}$	Parameters
m, n, r	Integers
$F(\mathbf{U}, \mathbf{V}), f(\mathbf{U}), g(\mathbf{V})$	Functions

For convenience of presentation, hereafter, we denote St_r^m by $\mathcal{M}_r$ which is a nonlinear manifold. The tangent space to $\mathcal{M}_r$ at $\mathbf{U}$, denoted by $T_\mathbf{U}\mathcal{M}_r$, is the set of all tangent vectors to $\mathcal{M}_r$ at $\mathbf{U}$. To be more specific, the tangent space of *Stiefel* manifold $\mathcal{M}_r$ at $\mathbf{U}$ is given as [1]:

$$\begin{aligned} T_\mathbf{U}\mathcal{M}_r &:= \left\{\mathbf{Z} \in \mathbb{R}^{m\times r} : \ \mathbf{Z}^\mathsf{T}\mathbf{U} + \mathbf{U}^\mathsf{T}\mathbf{Z} = \mathbf{0}\right\} \\ &= \left\{\mathbf{U}\mathbf{K} + (\mathbf{I}_m - \mathbf{U}\mathbf{U}^\mathsf{T})\mathbf{J} : \mathbf{K} = -\mathbf{K}^\mathsf{T} \in \mathbb{R}^{r\times r}, \mathbf{J} \in \mathbb{R}^{m\times r}\right\}. \end{aligned} \tag{3}$$

On the tangent space $T_\mathbf{U}\mathcal{M}_r$ for any $\mathbf{U} \in \mathcal{M}_r$, we introduce the standard inner product as a metric:

$$\langle \mathbf{Y}, \mathbf{Z}\rangle_\mathbf{U} := \mathrm{tr}(\mathbf{Y}^\mathsf{T}\mathbf{Z}), \ \forall \mathbf{Y}, \mathbf{Z} \in T_\mathbf{U}\mathcal{M}_r. \tag{4}$$

Then, we can view St_r^m (i.e. $\mathcal{M}_r$) as a sub-manifold of *Riemannian* manifold $\mathbb{R}^{m\times r}$. Given a smooth function $f(\mathbf{U})$ on $\mathcal{M}_r$, its *Riemannian* gradient is given as the orthogonal projection onto the tangent space of the gradient of $f(\mathbf{U})$. Specifically, the orthogonal projection of any $\mathbf{Z} \in \mathbb{R}^{m\times r}$ onto the tangent space at $\mathbf{U}$ is defined as [1]:

$$\begin{aligned} P_{T_\mathbf{U}\mathcal{M}_r}(\mathbf{Z}) :&= \mathbf{U}\left(\frac{\mathbf{U}^\mathsf{T}\mathbf{Z} - \mathbf{Z}^\mathsf{T}\mathbf{U}}{2}\right) + (\mathbf{I}_m - \mathbf{U}\mathbf{U}^\mathsf{T})\mathbf{Z} \\ &= \mathbf{Z} - \mathbf{U}\mathrm{Sym}(\mathbf{U}^\mathsf{T}\mathbf{Z}), \end{aligned} \tag{5}$$

where $\mathrm{Sym}(\mathbf{A}) := \frac{1}{2}(\mathbf{A} + \mathbf{A}^\mathsf{T})$. Let $\mathbf{G}_\mathbf{U} = \nabla f(\mathbf{U})$ be the gradient of $f(\mathbf{U})$, the *Riemannian* gradient of $f(\mathbf{U})$ on $\mathcal{M}_r$, denoted by $\mathrm{grad} f(\mathbf{U})$, can be calculated using:

$$\begin{aligned} \mathrm{grad} f(\mathbf{U}) &= P_{T_\mathbf{U}\mathcal{M}_r}(\mathbf{G}_\mathbf{U}) \\ &= \mathbf{G}_\mathbf{U} - \mathbf{U}\mathrm{Sym}(\mathbf{U}^\mathsf{T}\mathbf{G}_\mathbf{U}). \end{aligned} \tag{6}$$

Regarding our problem in (2), the gradient $\mathbf{G}_\mathbf{U}$ can be computed by

$$\mathbf{G}_\mathbf{U} = (\mathbf{U}\mathbf{V}^\mathsf{T} - \mathbf{X})\mathbf{V} + \lambda\mathbf{U}\mathbf{V}^\mathsf{T}\mathbf{V}. \tag{7}$$

By applying Equation (6) and Equation (7), $\mathrm{grad} f(\mathbf{U})$ can be computed by

$$\mathrm{grad} f(\mathbf{U}) = \frac{1}{2}\mathbf{U}\mathbf{V}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{U} - \frac{1}{2}\mathbf{X}\mathbf{V}. \tag{8}$$

3. ONMF ON STIEFEL MANIFOLD

For convenience, we define the objective function as

$$F(\mathbf{U}, \mathbf{V}) = \frac{1}{2}||\mathbf{X} - \mathbf{U}\mathbf{V}^\mathsf{T}||_F^2 + \frac{\lambda}{2}||\mathbf{U}\mathbf{V}^\mathsf{T}||_F^2. \tag{9}$$

So Problem (2) can be written as

$$\min_{\mathbf{U},\mathbf{V}} \ F(\mathbf{U}, \mathbf{V}), \quad \text{s.t.}, \mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}_k, \mathbf{V} \geq \mathbf{0}.$$

To address this problem, we propose the NRCG-ONMF method as in Algorithm 1, which NRCG-ONMF updates

$\mathbf{U}$ and $\mathbf{V}$ alternatively until reaching the stopping criteria. The details of updating $\mathbf{U}$ and $\mathbf{V}$ are discussed in following sections. In Section 3.1, we introduce the NRCG, a *Riemannian* optimization scheme, for updating the orthogonal factor matrix $\mathbf{U}$. We discuss the closed-form update of the non-negative factor matrix $\mathbf{V}$ in Section 3.2. The convergence analysis of NRCG-ONMF is given in Section 3.3.

Algorithm 1 NRCG-ONMF

Given $\mathbf{X}$, initialize $\mathbf{U}_0$,$\mathbf{V}_0$. Set $k = 1$.
1: Update $\mathbf{U}$ by NRCG method in Section 3.1.
2: Update $\mathbf{V}$ according to Algorithm 3 in Section 3.2.
3: Quit if stopping conditions achieve.
4: Let $k = k + 1$ and go to step 1.

3.1 Update U via NRCG

When updating $\mathbf{U}$, the variable $\mathbf{V}$ is fixed, and we equivalently address the following optimization problem:

$$\min_{\mathbf{U}} \quad f(\mathbf{U}), \quad \text{s.t.}, \mathbf{U}^\top\mathbf{U} = \mathbf{I}_k, \tag{10}$$

where $f(\mathbf{U}) = \frac{1}{2}||\mathbf{X} - \mathbf{U}\mathbf{V}^\top||_F^2 + \frac{\lambda}{2}||\mathbf{U}\mathbf{V}^\top||_F^2$ with $\mathbf{V}$ being fixed and $\mathbf{U}$ being restricted on the *Stiefel* manifold. Problem (10) is a non-convex and nonlinear optimization problem. To address it, we develop a NRCG method using BB step size for minimizing $f(\mathbf{U})$.

On Euclidean space, the classical CG method does the search with two major steps, i.e., *finding a conjugate search direction* and *determining the step size*. However, unlike classical CG method, when doing the optimization on the *Riemannian* manifold, two additional geometric operations, namely *Vector Transport* and *Retraction*, are required.

3.1.1 Conjugate Gradient Descent on $\mathcal{M}_r$

Similar to gradient based optimization methods on the Euclidean space, at any point $\mathbf{U}$, the optimization on *Stiefel* manifold requires identifying a search direction, which is a tangent vector to $\mathcal{M}_r$ at $\mathbf{U}$. A direct choice would be the steepest descent direction, i.e., the negative of the gradient of the objective function. However, the steepest descent method may incur slow convergence speed. To avoid this, we seek to apply the conjugate search direction.

On the Euclidean space $\mathbb{R}^n$, the conjugate search direction $\boldsymbol{\eta}_k$ in nonlinear CG is calculated by

$$\boldsymbol{\eta}_k = -\text{grad}f(\mathbf{U}_k) + \beta_k\boldsymbol{\eta}_{k-1}, \tag{11}$$

where the initial direction $\boldsymbol{\eta}_0$ is set to the steepest descent direction, and β_k is calculated by, for example, the Hestenes-Stiefel's rule (HS) [15, 14] as follows:

$$\beta_k = \frac{\langle \text{grad}f(\mathbf{U}_k), \text{grad}f(\mathbf{U}_k)\rangle - \langle \text{grad}f(\mathbf{U}_k), \text{grad}f(\mathbf{U}_{k-1})\rangle}{\langle \boldsymbol{\eta}_{k-1}, \text{grad}f(\mathbf{U}_k)\rangle - \langle \boldsymbol{\eta}_{k-1}, \text{grad}f(\mathbf{U}_{k-1})\rangle}. \tag{12}$$

Different from methods on the Euclidean space, the search direction on a manifold is adapted to follow a path on the manifold [1]. Since $\text{grad}f(\mathbf{U}_k) \in T_{\mathbf{U}_k}\mathcal{M}_r$, $\text{grad}f(\mathbf{U}_{k-1}) \in T_{\mathbf{U}_{k-1}}\mathcal{M}_r$, and $\boldsymbol{\eta}_{k-1}$ are in different tangent spaces of the manifold, Equations (11) is not applicable on *Riemannian* manifolds. To address this, we need to introduce the *Vector Transport*.

Vector Transport. The *Vector Transport* $\mathcal{T}$ on a manifold $\mathcal{M}_r$ is a smooth mapping that transports tangent vectors from one tangent space to another [31]. Specifically, let $\mathcal{T}_{\mathbf{U}\to\mathbf{Y}}(\boldsymbol{\zeta}_\mathbf{U})$ denote the transport from one tangent space $T_\mathbf{U}\mathcal{M}_r$ to another tangent space $T_\mathbf{Y}\mathcal{M}_r$, where $\boldsymbol{\zeta}_\mathbf{U}$ denotes the tangent vector on $T_\mathbf{U}\mathcal{M}_r$, the conjugate direction can be calculated by

$$\boldsymbol{\eta}_k = -\text{grad}f(\mathbf{U}_k) + \beta_k\mathcal{T}_{\mathbf{U}_{k-1}\to\mathbf{U}_k}(\boldsymbol{\eta}_{k-1}), \tag{13}$$

where β_k can be computed according to the HS rule (Equation (12)). Here, $\text{grad}f(\mathbf{U}_k)$ is the *Riemannian* gradient of $f(\mathbf{U}_k)$ on the *Riemannian* manifold $\mathcal{M}_r$, which can be computed by Equation (8). For the calculation of $\mathcal{T}_{\mathbf{U}\to\mathbf{Y}}(\boldsymbol{\zeta}_\mathbf{U})$, we adopt the method in [31] and readers are referred to this paper for details.

Given a search direction $\boldsymbol{\eta}_k$ at the k-th iteration, one may move the point $\mathbf{U}_k$ along the search direction to a new point $(\mathbf{U}_k+\alpha\boldsymbol{\eta}_k)$, where α is the step size. However, this new point will no longer stay on the manifold when $\alpha > 0$. To address this, we need to introduce the *Retraction* operation.

Retraction. *Retraction* is a projection mapping from the tangent bundle onto the manifold to keep the new point on the manifold [1]. In other words, with the retraction mapping, one can move points in the direction of a tangent vector and stay on the manifold. Given any $\boldsymbol{\xi} \in T_\mathbf{U}\mathcal{M}_r$, the retraction on *Stiefel* manifold $\mathcal{M}_r$, denoted by $R_\mathbf{U}(\boldsymbol{\xi})$, can be computed as

$$R_\mathbf{U}(\boldsymbol{\xi}) := qf(\mathbf{U} + \boldsymbol{\xi}), \quad s.t. \ \ \boldsymbol{\xi} \in T_\mathbf{U}\mathcal{M}_r, \tag{14}$$

where $qf(\mathbf{A})$ denotes the Q factor of the decomposition of $\mathbf{A}$ as $\mathbf{A} = \mathbf{Q}\mathbf{R}$ [1]. Apparently, we have $R_\mathbf{U}(\boldsymbol{\xi}) \in \mathcal{M}_r$, which is set as the new point. Given a search direction $\boldsymbol{\eta}_k$ at $\mathbf{U}_k$, set $\boldsymbol{\xi} = \alpha\boldsymbol{\eta}_k$, then the retraction can be performed by

$$\mathbf{U}_{k+1} = R_{\mathbf{U}_k}(\alpha\boldsymbol{\eta}_k) := qf(\mathbf{U}_k + \alpha\boldsymbol{\eta}_k), \tag{15}$$

where α is the step size to be determined.

3.1.2 Determination of the Step Size

A good step size would guarantee the convergence of a search algorithm and accelerate its speed without increasing much cost. Many search rules (e.g., Armijo-Wolfe rule [24]) have been proposed to find a suitable step size along a given search direction. In this paper, we choose the BB step size [3] as it can significantly reduce the total number of iterations through empirical studies on optimization problems that subject to spherical constraints [12].

BB adjusts the step size α by considering second order information (similar to Newton method) but without computing the second derivative of objective function. We adopt a non-monotone line search method from [35], where α_k satisfies non-monotone Wolfe conditions:

$$f(R_{\mathbf{U}_k}(\alpha_k\boldsymbol{\eta}_k)) \le C_k + \delta\alpha_k\langle \text{grad}f(\mathbf{U}_k), \boldsymbol{\eta}_k\rangle, \tag{16}$$

and

$$\begin{aligned} C_{k+1} &= (\sigma Q_k C_k + f(\mathbf{U}_{k+1}))/Q_{k+1}, \\ Q_{k+1} &= \sigma Q_k + 1, \end{aligned} \tag{17}$$

where $C_0 = f(\mathbf{U}_0)$ and $Q_0 = 1$. Once the condition in (16) is fulfilled, set $\alpha_{k+1} = \tau\alpha_k$. The parameters $\delta, \sigma, \tau \in (0, 1)$. Recall $\boldsymbol{\eta}_k \in T_{\mathbf{U}_k}\mathcal{M}_r$ is the search direction. The existence of α_k is guaranteed according to the following lemma.

LEMMA 1. *Let $\mathbf{U}_k \in \mathcal{M}_r$, and $\boldsymbol{\zeta}_k \in T_{\mathbf{U}_k}\mathcal{M}_r$ be a descent direction. Then there exists an α_k that satisfies the condition in (16).*

PROOF. Recall that $\mathcal{M}_r$ is a compact manifold. Since $\boldsymbol{\zeta}_k$ is a descent direction, it follows that $\mathbf{0} \neq \text{grad}f(\mathbf{U}_k)$, which implies that $\langle \text{grad}f(\mathbf{U}_k), \boldsymbol{\zeta}_k \rangle < 0$. Since $C_k \geq f(\mathbf{U}_k)$ (see [35]) and $R_{\mathbf{U}_k}(\alpha\boldsymbol{\eta}_k))$ is continuous in α, there must exist an $\widehat{\alpha}$ such that the inequality in (16) holds $\forall \alpha \in (0, \widehat{\alpha}]$. □

The update of $\mathbf{U}$ is summarized in Algorithm 2.

Algorithm 2 Nonlinear *Riemannian* Conjugate Gradient (NRCG) Update for $\mathbf{U}$

1: Given $\mathbf{U}_k$, $\mathbf{X}$, $\mathbf{V}_k$, $\alpha > 0$, $\delta, \sigma, \tau \in (0,1)$, $k \geq 1$.
2: Compute the gradient $\text{grad}f(\mathbf{U}_k)$ by (8).
3: Compute a conjugate direction $\boldsymbol{\eta}_k$ according to (13).
4: Choose a BB step size α_k to satisfy conditions (16) and (17), and set $\mathbf{U}_{k+1} = R_{\mathbf{U}_k}(\alpha_k \boldsymbol{\eta}_k)$.
5: Output $\mathbf{U}_{k+1}$ in order to update $\mathbf{V}$.

3.2 Update V

When updating $\mathbf{V}$, we fix $\mathbf{U}$ as a constant. Due to the non-negative constraint of $\mathbf{V}$, updating all the elements in $\mathbf{V}$ is difficult. Therefore, we adopt a coordinate-decent update for $\mathbf{V}$, namely we update each entry of $\mathbf{V}$ in a random order. For example, to update V_{jl}, we have

$$(\mathbf{U}, \mathbf{V}) \rightarrow (\mathbf{U}, \mathbf{V} + s_{jl}E_{jl}), \tag{18}$$

where $\mathbf{E}$ is a $n \times r$ matrix with all elements zero except that $E_{j,l} = 1$, and s_{jl} is regarded as the step size when updating V_{jl}. The problem of finding s_{jl} in (18) can be cast as the following optimization problem:

$$\min_{s_{jl}: \mathbf{V}_{jl} + s_{jl} \geq 0} g_{jl}(s_{jl}) = g(\mathbf{V} + s_{jl}E_{jl}), \tag{19}$$

where $g(\mathbf{V}) = \frac{1}{2}||\mathbf{X} - \mathbf{U}\mathbf{V}^\top||_F^2 + \frac{\lambda}{2}||\mathbf{U}\mathbf{V}^\top||_F^2$ with $\mathbf{U}$ being fixed. Similar to [16], we can rewrite $g_{jl}(s_{jl})$ as

$$g_{jl}(s_{jl}) = \frac{1}{2}g_{jl}^{''}(0)s_{jl}^2 + g_{jl}^{'}(0)s_{jl} + g_{jl}(0), \tag{20}$$

where the $g_{jl}^{'}(0)$ and $g_{jl}^{''}(0)$ denote the first derivative and second derivative of $g_{jl}(s_{jl})$ at $s_{jl} = 0$, respectively. It follows that

$$\begin{aligned} g_{jl}^{'}(0) &= (\frac{\partial g}{\partial \mathbf{V}})_{jl} \\ &= ((\mathbf{U}\mathbf{V}^\top - \mathbf{X})^\top\mathbf{U} + \lambda(\mathbf{U}\mathbf{V}^\top)^\top\mathbf{U})_{jl} \\ &= (1+\lambda)V_{jl} - (\mathbf{X}^\top\mathbf{U})_{jl}, \end{aligned} \tag{21}$$

and

$$g_{jl}^{''}(0) = (\frac{\partial^2 g}{\partial \mathbf{V}^2})_{jl} = (1+\lambda). \tag{22}$$

By ignoring the non-negative constraint, one can get a closed form minimizer of $g_{jl}(s_{jl})$ as follows

$$s_{jl} = -\frac{g_{jl}^{'}(0)}{g_{jl}^{''}(0)}. \tag{23}$$

Accordingly, considering the non-negative constraint on $\mathbf{V}$, the computation of s_{jl} is modified as

$$s_{jl}^* = \max(0, V_{jl} - \frac{g_{jl}^{'}(0)}{g_{jl}^{''}(0)}) - V_{jl}. \tag{24}$$

Algorithm 3 Closed-form Update for $\mathbf{V}$

1: Given $\mathbf{U}_{k+1}$,$\mathbf{V}_k$, $\mathbf{X}$, $k \geq 1$.
2: for each $i \in (1, n)$, $j \in (1, r)$
  Compute $s_{i,j}^*$ by (25).
  Set $V_{k+1}^{i,j} = s_{i,j}^* + V_k^{i,j}$.
  end
3: Output $\mathbf{V}_{k+1}$ in order to update $\mathbf{U}$ in next iteration.

From Equations (21), (22) and (24), we get

$$\begin{aligned} s_{jl}^* &= \max(0, V_{jl} - \frac{(1+\lambda)V_{jl} - (\mathbf{X}^\top\mathbf{U})_{jl}}{1+\lambda}) - V_{jl}, \\ &= \max(0, \frac{(\mathbf{X}^\top\mathbf{U})_{jl}}{1+\lambda}) - V_{jl}. \end{aligned} \tag{25}$$

The detailed update of $\mathbf{V}$ is given in Algorithm 3.

3.3 Convergence Analysis

PROPOSITION 1. *Let $\{\mathbf{U}_k, \mathbf{V}_k\}$ be an infinite sequence of iterates generated by Algorithm 1. Then every accumulation point of $\{\mathbf{U}_k, \mathbf{V}_k\}$ is a critical point of f over the space $\mathcal{M}_r \times {\mathbb{R}_+}^{n \times r}$, namely $\lim_{k\to\infty} gradF(\mathbf{U}_k, \mathbf{V}_k) = \mathbf{0}$ and $\lim_{k\to\infty} \partial_{\mathbf{V}_k} F(\mathbf{U}_k, \mathbf{V}_k) = \mathbf{0}$, where $gradF(\mathbf{U}_k, \mathbf{V}_k) = gradf(\mathbf{U}_k)$.*

PROOF. Note that $\mathcal{M}_r$ is a compact manifold. Moreover, $\{\mathbf{V}_k\}$ is bounded; otherwise $F(\mathbf{U}, \mathbf{V})$ will go to infinity. Without loss of generality, suppose $\mathbf{V} \in [0, L]^{n \times r}$, where $L > 0$ is a finite number. Now both $\{\mathbf{U}_k\}$ and $\{\mathbf{V}_k\}$ stay in a closed and bounded subset.

We complete the proof by contradiction. Without loss of generality, suppose

$$\lim_{k\to\infty} ||\text{grad}F(\mathbf{U}_k, \mathbf{V}_k)||_F + ||\partial_{\mathbf{V}_k} F(\mathbf{U}_k, \mathbf{V}_k)||_F \neq 0,$$

and then there exists an $\epsilon > 0$, and a subsequence in $\{(\mathbf{U}_k, \mathbf{V}_k)\}$ such that $||\text{grad}F(\mathbf{U}_k, \mathbf{V}_k)||_F + ||\partial_{\mathbf{V}} F(\mathbf{U}_k, \mathbf{V}_k)||_F \geq 2\epsilon > 0$ for all k. Since $\mathcal{M}_r$ is closed and bounded, and $\mathbf{V}_k$ is constrained in $[0, L]^{n \times r}$, the subsequence $\{(\mathbf{U}_k, \mathbf{V}_k)\}_{k \in \Gamma}$ should have a limit point $(\mathbf{U}^*, \mathbf{V}^*)$ on $\mathcal{M}_r \times [0, L]^{n \times r}$, i.e. $\lim_{k\to\infty} F(\mathbf{U}_k, \mathbf{V}_k) = F(\mathbf{U}^*, \mathbf{V}^*)$. By the continuity of $\text{grad}F(\mathbf{U}, \mathbf{V})$ and $\partial_{\mathbf{V}} F(\mathbf{U}, \mathbf{V})$, it implies that either

$$\lim_{k\to\infty} ||\text{grad}F(\mathbf{U}_k, \mathbf{V}_k)||_F \geq \epsilon > 0$$

or

$$\lim_{k\to\infty} ||\partial_{\mathbf{V}} F(\mathbf{U}_k, \mathbf{V}_k)||_F \geq \epsilon > 0.$$

Without loss of generality, suppose $\lim_{k\to\infty} ||\text{grad}F(\mathbf{U}_k, \mathbf{V}_k)||_F \geq \epsilon > 0$. Based on (16), there exist a step size α such that $C_k = f(\mathbf{U}_k)$ and $f(\mathbf{U}_{k+1}) = F(\mathbf{U}_{k+1}, \mathbf{V}_k) \leq f(\mathbf{U}_k) + \delta\alpha\langle \text{grad}f(\mathbf{U}_k), \boldsymbol{\eta}_k \rangle$. Note that $F(\mathbf{U}_k, \mathbf{V}_k) = f(\mathbf{U}_k) > F(\mathbf{U}_{k+1}, \mathbf{V}_k)$, $\boldsymbol{\eta}_k$ is a descent direction (such as the steepest descent direction), and $\lim_{k\to\infty} ||\text{grad}F(\mathbf{U}_k, \mathbf{V}_k)||_F \geq \epsilon > 0$. Then $\forall k$, it follows that $|F(\mathbf{U}_k, \mathbf{V}_k) - F(\mathbf{U}_{k+1}, \mathbf{V}_{k+1})| \geq |F(\mathbf{U}_k, \mathbf{V}_k) - F(\mathbf{U}_{k+1}, \mathbf{V}_k)| \geq |\delta\alpha\langle \text{grad}f(\mathbf{U}_k), \boldsymbol{\eta}_k \rangle| = \nu > 0$, where ν is some constant. This contradicts that $\{(\mathbf{U}_k, \mathbf{V}_k)\}$ has a limit point. In addition, we have similar result for $\mathbf{V}$. We therefore conclude that $\lim_{k\to\infty} \text{grad}F(\mathbf{U}_k, \mathbf{V}_k) = \mathbf{0}$ and $\lim_{k\to\infty} \partial_{\mathbf{V}} F(\mathbf{U}_k, \mathbf{V}_k) = \mathbf{0}$. □

4. RELATED WORKS

In this section, we review some of the representative efforts, as well as the most recent solutions of the ONMF problem. We also briefly overview the works that introduce manifold into the NMF solutions.

Ding et al. [9] are the first to explicitly propose the concept of ONMF. They impose the orthogonality constraint on factor matrix by considering Lagrangian multiplier which can be solved as an unconstrained optimization problem. Thus they apply the standard multiplicative update rules on each of the factor matrices. Choi [6] simplifies Ding's algorithm by turning the orthogonality constraint into *Stiefel* manifold. They directly use the gradient in *Stiefel* manifold in the multiplicative update of the orthogonal matrix. The Euclidean distance based ONMF method proposed in [21] also embeds Lagrangian multiplier in the solution. Moreover, they consider gradient descent in both the orthogonal subspace and the original space. However, these approaches produce heavy computation overhead as the Lagrange multiplier is a symmetrical matrix with many parameters.

Pompili et al. [26] propose two ONMF solutions, one of which is an EM-like ONMF algorithm based on the equivalence of ONMF and spherical k-means they proved. Very recently, Asteris et al. [2] propose an ONMF solution that relies on a novel approximation to the Non-negative Principle Component Analysis (NNPCA) problem, which jointly optimizes multiple orthogonal non-negative components and provably achieves an object value close to be optimal. However above works can only attain orthogonality to a limited extent. One exception is the method proposed by Pompili et al. in [26]. This method updates the orthogonal constrained matrix using projection step, which projects the matrix onto a feasible set of $St(k, n)$ of orthogonal matrices via a projection gradient method. The step size is chosen according to a backtracking line search using the Armijo rule. Our work shares similar idea with this method, but exploits more efficient nonlinear conjugate search algorithm with non-monotone step sizes.

Many research efforts introduce the manifold regularization into the NMF solutions [27, 5, 13, 17, 29]. Although they use the manifold concept in the matrix approximation problem, these approaches differ from our solution in that they identify the geometrical structure of the original data space by incorporating the geometrical structure into objective function (as the regularizer). Our solution consider the parameter constraint on factor matrix that is equivalent to the manifold definition.

5. EXPERIMENTS

In this section, we use both synthetic and real-world data sets to show the performance of our proposed method, NRCG-ONMF. Our method is implemented in Matlab and the source codes are available per request. The source codes of all the comparing methods are either publicly available or obtained from the corresponding authors. We perform the experiments on a PC with 64-bit Windows 7 operation system, 8GB RAM and 2.40GHZ Intel i7-3630QM CPU.

5.1 Implementation Details

Initialization. In general, optimization methods on nonlinear manifolds are guaranteed to converge to a local solution. Therefore, a good initialization of $\mathbf{U}_0$ and $\mathbf{V}_0$ is important. In our case, $\mathbf{U}_0$ and $\mathbf{V}_0$ can be obtained by applying truncated Singular Value Decomposition (SVD) of rank r on $\mathbf{X}$, namely $[\bar{\mathbf{U}}, \bar{\mathbf{S}}, \bar{\mathbf{V}}] = \text{svds}(\mathbf{X}, r)$, where $\bar{\mathbf{U}}$, $\bar{\mathbf{S}}$ and $\bar{\mathbf{V}}$ are the output of SVD. We then set $\mathbf{U}_0 = \bar{\mathbf{U}}$ and $\mathbf{V}_0 = \max(\bar{\mathbf{S}}\bar{\mathbf{V}}, 0)$.

Gradient Computation. Because $\mathbf{U}$ is imposed on orthogonality constraint, namely $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}_r$, the following equation has the same projection result on the tangent space $T_{\mathbf{U}}\mathcal{M}_r$ as Equation (7).

$$\mathbf{G}_\mathbf{U} = (\mathbf{U}\mathbf{V}^\mathsf{T} - \mathbf{X})\mathbf{V} \tag{26}$$

So to reduce calculation time, when calculating $\text{grad} f(\mathbf{U})$, we use Equation (26) to replace $\mathbf{G}_\mathbf{U}$ in Equation (6).

Objective Value Computation. Note that the original objective function is $\frac{1}{2}||\mathbf{X} - \mathbf{U}\mathbf{V}^\mathsf{T}||_F^2 + \frac{\lambda}{2}||\mathbf{U}\mathbf{V}^\mathsf{T}||_F^2$, which has $O(mnr + mn)$ complexity. However, we can rewrite it into the following form:

$$\frac{1}{2}||\mathbf{X}||_F^2 - trace(\mathbf{X}^\mathsf{T}\mathbf{U}, \mathbf{V}) + \frac{1+\lambda}{2}||\mathbf{V}||_F^2. \tag{27}$$

Note that $\frac{1}{2}||\mathbf{X}||_F^2$ is a constant, thus it can be pre-computed. Now, the calculation of Equation (27) takes $O(mnr + mr)$. As generally $r \ll n$, we can reduce the computation cost using Equation (27) to compute objective value.

5.2 Data sets

We use synthetic data to compare the orthogonality and convergence of various ONMF algorithms. Real-world data sets are used for the comparison of clustering performance.

5.2.1 Synthetic Data

We generate a synthetic data set in order to control the noise level. Specifically, we select five base vectors $\mathbf{b}_j, j \in [1, 5]$ randomly from the unit hypercube in n dimensions (here $n = 100$). The selected base vectors are independent to each other. Then, we generate data points $\mathbf{x}_i = w_i\mathbf{b}_j + \epsilon\mathbf{v}_i, i \in [1, m]$ (here $m = 100$ is the number of data points generated for a testing matrix), where $\epsilon \in [10^{-2}, 1]$ is a parameter controlling the noise level, $w_i \in (0, 1)$ is the random weight on base vectors and satisfies the uniform distribution. $\mathbf{v}_i$ is the random vector satisfying the multivariate standard Gaussian distribution. Negative entries of $\mathbf{x}_i$ are set to zero. For each $\epsilon \in [10^{-2}, 1]$, we generate m data points and vertically concatenate them into a $m \times n$ matrix $\mathbf{X}$, where each row represents a data point.

5.2.2 Real-world Data sets

We collect several publicly-available real-world data sets for our experimental studies. These data sets are described as follows:

COIL-20 [23]. It is an image data set of objects, which contains 1,440 gray scale images of 20 different objects. In our experiment, each image is resized to 32×32 pixels.

The Yale Face Database [28]. This data set contains 165 gray scale images of 15 people. Each one has 11 images with different facial expression or configuration. We resize each image to 32×32 pixels.

CLUTO. It is a data set collected for text mining [36]. We choose one large subset (k1b) and one medium subset (wap) from CLITO to showcase the performance on text data with various sizes.

Table 2: Data sets Description

Data	Instance#	Dimension#	Class#	Type
COIL20	1,440	1,024	20	image
Yale	165	1,024	15	image
CLUTO-k1b	2,340	21,839	6	text
CLUTO-wap	1,560	8,460	20	text
UCI-mfeat-fac	2,000	216	10	text
UCI-mfeat-fou	2,000	76	10	text
UCI-mfeat-pix	2,000	240	10	text
UCI-mfeat-zer	2,000	47	10	text

UCI-mfeat [30]. This data set consists of features of handwritten numerals extracted from a collection of Dutch utility maps. 200 patterns per class for a total of 2,000 patterns have been digitized in binary images files.

Table 2 shows the statistics of each of the data sets.

5.3 Metrics

5.3.1 ONMF Metrics

We evaluate the various ONMFs in terms of three metrics, namely relative *Frobenius* approximation error (relative error for short thereafter), orthogonality, computation time. They are defined as follows:

- Relative error: we define the relative error as in [2]:

$$RelFerr = \frac{||\mathbf{X} - \mathbf{U}\mathbf{V}^T||_F^2}{||\mathbf{X}||_F^2} \tag{28}$$

- Orthogonality: we leverage the orthogonality measurement in [6]: $||\mathbf{U}^\mathsf{T}\mathbf{U} - \mathbf{I}||$.

- Computation time: we record the time a algorithm takes to converge or reach the maximum iteration.

5.3.2 Clustering Metrics

To evaluate the clustering performance, we adopt three metrics, namely Clustering Accuracy (CA), Normalized Mutual Information (NMI) and Purity metrics from [33, 21, 18, 11]. The definitions of the three metrics are given as the follows:

- **CA** is defined as follows:

$$CA = \frac{\sum_{i=1}^{n} \delta(map(r_i), l_i)}{n} \tag{29}$$

  where r_i is the computed cluster label and l_i is the true cluster label. $\delta(x, y) = 1$ if $x = y$, otherwise, $\delta(x, y) = 0$. $map(r_i)$ is a mapping function that matches the computed label to the best true label. We adopt the Kuhn-Munkres [22] algorithm for the mapping. A larger CA value indicates a better clustering result.

- **NMI** is defined as follows:

$$NMI = \frac{I(R, L)}{(H(R) + H(L))/2} \tag{30}$$

  where R denotes the set of true cluster labels and L is a set of computed lables from the evaluated algorithm. $I(R, L)$ is the mutual information (see [4] for the definition) between R and L. $H(.)$ is the entropy function. A larger NMI value indicates a better clustering result.

- **Purity** measures the extent to which each cluster contained data points from primarily one cluster. The purity of a clustering algorithm is obtained by the weighted sum of individual cluster purity values [18], given as:

$$Purity = \frac{1}{n}\sum_{i=1}^{k} max_j(n_i^j) \tag{31}$$

  where n_i^j is the number of the i-th input cluster that is assigned to the j-th cluster, k is the number of clusters and n is the total number of the data points. A larger purity value indicates a better clustering solution.

5.4 Results

In the experiments, we use the stopping criteria suggested in [31]: when $|1-\frac{\sqrt{2*currentObj}}{\sqrt{2*preciousObj}}| < 1e{-}5$, the iteration stops. The *currentObj* is current objective value and *preciousObj* is the objective value from last iteration.

5.4.1 Convergence Comparison

We evaluate the convergence of NRCG-ONMF with three β solutions and ONPMF on synthetic data without noises ($\epsilon = 0$). The other two β solutions besides HS (Equation (12)) are as follows:

Polak-Ribiére (PR) rule[14] is:

$$\beta_k = \frac{\langle \mathrm{grad} f(\mathbf{U}_k), \mathrm{grad} f(\mathbf{U}_k)\rangle - \langle \mathrm{grad} f(\mathbf{U}_k), \mathrm{grad} f(\mathbf{U}_{k-1})\rangle}{\langle \mathrm{grad} f(\mathbf{U}_{k-1}), \mathrm{grad} f(\mathbf{U}_{k-1})\rangle}.$$

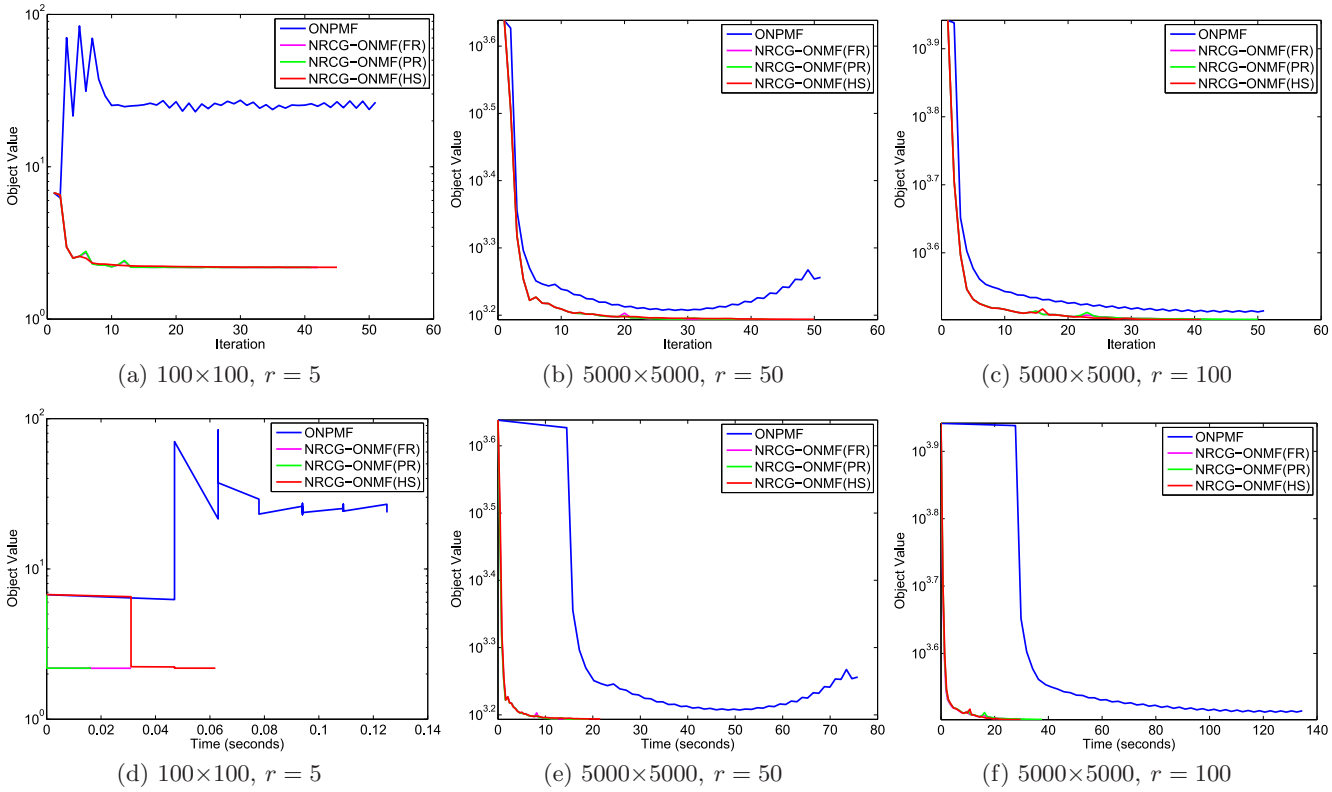
and the Fletcher-Reeves (FR) rule[10] is:

$$\beta_k = \frac{\langle \mathrm{grad} f(\mathbf{U}_k), \mathrm{grad} f(\mathbf{U}_k)\rangle}{\langle \mathrm{grad} f(\mathbf{U}_{k-1}), \mathrm{grad} f(\mathbf{U}_{k-1})\rangle}.$$

We choose ONPMF because it also considers *Stiefel* manifold when preserving orthogonality for $\mathbf{U}$. Three randomly generated matrices with different dimensions and ranks are used in our experiments and we set the maximum iteration number to 50 in order to show the different performance clearly. Figure 1 depicts the results. NRCG-ONMF algorithms are shown non-monotone because BB step does not necessarily decrease the objective value at every iteration, but this issue has been solved in our adopted non-monotone line search method in [35]. For the small and medium sized matrices (100×100 and $5,000 \times 5,000$), all variants of NRCG-ONMF (i.e., with different β) can converge within 50 iterations (Figure 1(a), 1(b) and 1(c)). The convergence time on small matrix is less than 0.07 seconds and the time on medium matrix is less than 30 seconds (Figure 1(d), 1(e) and 1(f)). For small matrix, NRCG-ONMF(HS) converges less quickly than NRCG-ONMF(FR) and NRCG-ONMF(PR), but is more quickly than these two algorithms when performed on medium matrix. Thus we only choose NRCG-ONMF(HS) in the following evaluations. ONPMF cannot converge within 50 iterations (more than 500 iterations when raising the iteration limit). For medium matrix with rank 50, it has a up trend on objective values. NRCG-ONMF algorithms outperform ONPMF in all testing matrices.

5.4.2 Orthogonality Comparison

In this section, we compare several state-of-the-art ONMF works with our method NRCG-ONMF(HS). We also show the comparison of their performance when introducing noises.



(a) 100×100, $r = 5$ (b) 5000×5000, $r = 50$ (c) 5000×5000, $r = 100$

(d) 100×100, $r = 5$ (e) 5000×5000, $r = 50$ (f) 5000×5000, $r = 100$

Figure 1: Convergence of NRCG-ONMFs and ONPMF on synthetic data sets of different sizes/ranks. Three NRCG-ONMF variants with different β choices are compared. The matrices are generated randomly, and the maximum iteration number is set to 50.

Table 3: Description of Orthogonal NMF algorithms discussed in this work

Algorithms	Object Function	Orthogonal part
ONMFEU [9]	$min\|\|\mathbf{X} - \mathbf{FG}^\mathsf{T}\|\|_F^2$	**F**
ONMFA [6]	$min\|\|\mathbf{X} - \mathbf{AS}\|\|_F^2$	**A**
NMFOS_ED [21]	$min\|\|\mathbf{V} - \mathbf{WH}\|\|_F^2 + \lambda\|\|\mathbf{W}^\mathsf{T}\mathbf{W} - I\|\|_F^2$	**W**
ONPMF [26]	$min\|\|\mathbf{M} - \mathbf{UV}\|\|_F^2$	**V**
EMONMF [26]	$min\|\|\mathbf{M} - \mathbf{UV}\|\|_F^2$	**V**
SPANONMF [2]	$min\|\|\mathbf{M} - \mathbf{WH}^\mathsf{T}\|\|_F^2 + \epsilon\|\|\mathbf{M}\|\|_F^2$	**W**
NRCG-ONMF(HS) (this work)	$min\|\|\mathbf{X} - \mathbf{UV}^\mathsf{T}\|\|_F^2 + \lambda\|\|\mathbf{UV}^\mathsf{T}\|\|_F^2$	**U**

Table 4: Performance comparison of various ONMF methods. RelFerr represents relative Frobenius approximation error. Orthogonality denotes the value of $||U^TU - I||_F$. Time is the computation time for 500 iterations or until converged. #Iter is the actual iteration number. A lower value of all metrics indicates a better performance. Bold font denotes the best performance. The values are the average of 100 tries on $100 \times 100, rank = 5$ randomly generated matrices.

Algorithms	RelFerr	Orthogonality	Time(sec)	#Iter
ONMFEU	231.1850	12906.1000	0.2362	500
ONMFA	0.0989	2.1696	0.1381	333
NMFOS_ED	0.2323	0.4600	0.2293	500
ONPMF	0.3451	5.5267e-15	0.7849	500
EMONMF	0.0836	49.0612	0.2868	**10**
SPANONMF	0.1179	1.7320	0.0893	500
NRCG-ONMF(HS)	**0.0236**	**1.0438e-15**	**0.0747**	55

It is worth noting that the parameters of each comparing methods are set as the values suggested in the corresponding papers. The descriptions of compared ONMF works are shown in the Table 3.

Table 4 gives the average value of comparison results based

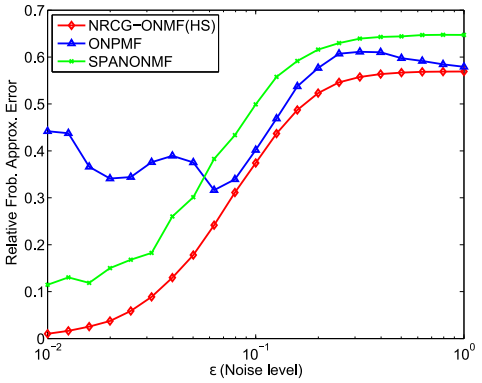


Figure 2: Relative error on synthetic data with noises. The values are the average of 100 tries on $100 \times 100, rank = 5$ randomly generated matrices.

on 100 tries. The values of bold font denote the best performance values. The iteration number equals to 500 indicates the algorithm cannot converge within 500 iteration. From this table we can see that NRCG-ONMF(HS) achieves the lowest values in three metrics except the iteration number. SPANONMF is the second fastest algorithm. However it fails to maintain the orthogonality constraint. ONPMF is the only one that has comparable performance with NRCG-ONMF(HS) in terms of keeping orthogonality. However it has one order higher relative error than NRCG-ONMF(HS), as well as higher computation time and number of iterations.

We further compare NRCG-ONMF(HS), SPANONMF and ONPMF when the noises are introduced. We run each algorithm for 100 times and record the average value. We choose ONMF because it maintains the only comparable performance on preserving orthogonality with our method NRCG-ONMF(HS). SPANONMF is chosen as it is the second best in terms of computation time (Table 4). Figure 2 shows that the relative error increases when the noise level increases. NRCG-ONMF(HS) continues to achieve the lowest relative error. SPANONMF shows smoothly increased relative error that is consistently higher than NRCG-ONMF(HS), while the value of ONPMF fluctuates at the beginning and gradually becomes smooth.

5.4.3 Clustering Performance Comparison

We compare various ONMF works on the clustering application. As EMONMF and ONMFEU performs the worst in keeping orthogonality and NMFOS_ED does not perform well in three metrics, we omit them in this evaluation. We also compare the well-known clustering algorithm k-means and some of the very recent matrix factorization works that aim to do clustering, namely CAPNMF [11], CVXNMF [8] and NMF_GCD [16].

We report the performance results in Table 5-7 for CA, NMI and Purity respectively with deviations after '±'. Values in bold denote the best performance. For most of the data sets, our algorithm NRCG-ONMF achieves the best in terms of accuracy, mutual information and purity. ONPMF and EMONMF also perform well in some data sets. For example, ONPMF gives the highest accuracy in the mfeat-fou data set and the highest purity in the COIL20 data set. EMONMF achieves the best NMI and purity for the Yale data set. k-means also shows good performance on some data sets. For example, it achieves the highest accuracy for UCI-mfeat-pix and UCI-mfeat-zer data sets.

6. CONCLUSIONS

In this paper, we have proposed a NRCG-ONMF method which alternatively updates the orthogonal factor $\mathbf{U}$ by doing nonlinear search on *Stiefel* manifold, and updates the nonnegative factor $\mathbf{V}$ in a coordinate manner with closed form solutions. The convergence of NRCG-ONMF has been analyzed. Our approach sheds lights on an promising way to efficiently perform ONMF and shows great potential to handle large scale problems. We evaluate the proposed method on clustering tasks. Extensive experiments on both synthetic and real-world data sets demonstrate that the proposed NRCG-ONMF method outperforms other ONMF methods in terms of the effectiveness on preservation of orthogonality, optimization efficiency and clustering performance.

7. ACKNOWLEDGMENTS

We thank all the people who have made open their source codes to the public or kindly share their codes with us.

This work was in part funded by the National Natural Science Foundation of China (NSFC) under Grant No. 61602185, the Data to Decisions Cooperative Research Centre, Australia, Australian Research Council grants DP140102270, DP160100703, DP140100104 and FT140101247.

8. REFERENCES

[1] P. Absil, R. E. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[2] M. Asteris, D. Papailiopoulos, and A. G. Dimakis. Orthogonal NMF through Subspace Exploration. In *Proc. of the 29th Annual Conference on Neural Information Processing Systems (NIPS 2015)*, pages 343–351, Montreal, Canada, December 2014.

[3] J. Barzilai and J. M. Borwein. Two-Point Step Size Gradient Methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[4] D. Cai, X. He, and J. Han. Document Clustering Using Locality Preserving Indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, 2005.

[5] D. Cai, X. He, J. Han, and T. S. Huang. Graph Regularized Nonnegative Matrix Factorization for Data Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2011.

[6] S. Choi. Algorithms for orthogonal nonnegative matrix factorization. In *Proc. of the International Joint Conference on Neural Networks (IJCNN 2008)*, pages 1828–1832, Hong Kong, China, June 2008.

[7] C. H. Q. Ding and X. He. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In *Proc. of the 2005 SIAM International Conference on Data Mining (SDM 2005)*, pages 606–610, Newport Beach, USA, April 2005.

[8] C. H. Q. Ding, T. Li, and M. I. Jordan. Convex and Semi-Nonnegative Matrix Factorizations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):45–55, 2010.

Table 5: Clustering performance comparison in terms of CA on different data sets.

Data set	K-Means	CAPNMF	CVXNMF	NMF_GCD
COIL20	0.5364±0.0422	0.6942±0.0295	0.6601±0.0360	0.6390±0.0195
Yale	0.4353±0.0138	0.4532±0.0301	0.4345±0.0198	0.3830±0.0325
CLUTO-k1b	0.7579±0.0072	0.5940±0.0012	0.6460±0.0021	0.7015±0.0474
CLUTO-wap	0.3638±0.0420	0.2186±0.0124	0.3631±0.0138	0.3446±0.0245
UCI-mfeat-fac	0.6213±0.0552	0.6005±0.0302	0.5308±0.0402	0.4766±0.0479
UCI-mfeat-fou	0.623±0.0694	0.6013±0.0303	0.6894±0.0313	0.6306±0.0353
UCI-mfeat-pix	**0.7582±0.0572**	0.6578±0.0245	0.6673±0.0289	0.7071±0.0489
UCI-mfeat-zer	**0.5181±0.0160**	0.4705±0.0235	0.5058±0.0280	0.4025±0.0688
Data set	ONPMF	EMONMF	SPANONMF	NRCG-ONMF
COIL20	0.6736±0.0345	0.6260±0.0358	0.6011±0.0317	**0.6982±0.0306**
Yale	0.4321±0.0284	0.4333±0.0285	0.3770±0.0117	**0.4588±0.0257**
CLUTO-k1b	0.6665±0.0322	0.5726±0.1204	0.6633±0.1025	**0.7590±0.0492**
CLUTO-wap	0.3623±0.0098	0.3687±0.0231	0.2804±0.0114	**0.4090±0.0221**
UCI-mfeat-fac	0.5449±0.0549	0.5386±0.0621	0.2338±0.0003	**0.6472±0.0068**
UCI-mfeat-fou	**0.7201±0.0024**	0.6083±0.0429	0.5560±0.0521	0.7197±0.0006
UCI-mfeat-pix	0.6949±0.0672	0.5845±0.0976	0.5323±0.0857	0.7372±0.0624
UCI-mfeat-zer	0.5094±0.0045	0.4788±0.0566	0.3760±0.0807	0.5157±0.0182

Table 6: Clustering performance comparison in terms of NMI on different data sets.

Data set	K-Means	CAPNMF	CVXNMF	NMF_GCD
COIL20	0.7186±0.0092	0.7556±0.0195	0.7696±0.0172	0.7425±0.0058
Yale	0.5011±0.0229	0.4935±0.0190	0.4973±0.0144	0.4242±0.0301
CLUTO-k1b	0.3732±0.0153	0.3027±0.0123	0.3727±0.0039	0.4347±0.0202
CLUTO-wap	0.3674±0.0277	0.3120±0.0106	0.4238±0.0154	0.3829±0.0141
UCI-mfeat-fac	0.6145±0.0244	0.5045±0.0356	0.5051±0.0157	0.4462±0.0698
UCI-mfeat-fou	0.6315±0.0318	0.5045±0.0308	0.6600±0.0081	0.5977±0.0105
UCI-mfeat-pix	0.6336±0.0301	0.5065±0.0284	0.6675±0.0172	0.6542±0.0187
UCI-mfeat-zer	0.4808±0.0097	0.3043±0.0403	0.4762±0.0151	0.3512±0.0589
Data set	ONPMF	EMONMF	SPANONMF	NRCG-ONMF
COIL20	0.7754±0.0242	0.7452±0.0224	0.6997±0.0194	**0.7826±0.0104**
Yale	0.4906±0.0173	**0.5050±0.0307**	0.4297±0.0233	0.5028±0.0231
CLUTO-k1b	0.4263±0.0123	0.2302±0.1350	0.3408±0.1223	**0.4638±0.0080**
CLUTO-wap	0.3946±0.0014	0.3727±0.0352	0.3927±0.0141	**0.4460±0.0142**
UCI-mfeat-fac	0.5192±0.0242	0.5134±0.0344	0.1936±0.0005	**0.6209±0.0061**
UCI-mfeat-fou	0.6672±0.0024	0.5704±0.0355	0.5339±0.0603	**0.6734±0.0018**
UCI-mfeat-pix	0.6568±0.0263	0.5492±0.0993	0.5128±0.0621	**0.6954±0.0242**
UCI-mfeat-zer	0.4728±0.0013	0.3992±0.0402	0.3008±0.0976	**0.4886±0.0028**

Table 7: Clustering performance comparison in terms of Purity on different data sets.

Data set	K-Means	CAPNMF	CVXNMF	NMF_GCD
COIL20	0.6722±0.0490	0.6320±0.0115	0.6967±0.0105	0.6796±0.0137
Yale	0.4339±0.0447	0.4215±0.0205	0.4400±0.0175	0.3770±0.0224
CLUTO-k1b	0.7937±0.0108	0.5957±0.0102	0.7792±0.0004	0.7938±0.0121
CLUTO-wap	0.4901±0.0186	0.4256±0.0175	0.5004±0.0165	0.4827±0.0231
UCI-mfeat-fac	0.6610±0.0601	0.5090±0.0754	0.5125±0.0125	0.5081±0.0827
UCI-mfeat-fou	0.6844±0.0433	0.6039±0.0195	0.6875±0.0159	0.6541±0.0261
UCI-mfeat-pix	0.7130±0.0647	0.6056±0.0246	0.7104±0.0208	0.7312±0.0339
UCI-mfeat-zer	0.5499±0.0237	0.6045±0.0308	0.5497±0.0164	0.4350±0.0602
Data set	ONPMF	EMONMF	SPANONMF	NRCG-ONMF
COIL20	**0.7162±0.0279**	0.6614±0.0325	0.6247±0.0433	0.7122±0.0212
Yale	0.4327±0.0289	**0.4558±0.0367**	0.4218±0.0406	0.4364±0.0283
CLUTO-k1b	0.7935±0.0092	0.6776±0.0905	0.7238±0.0740	**0.8021±0.0106**
CLUTO-wap	0.5354±0.0108	0.4945±0.0242	0.4794±0.0210	**0.5413±0.0099**
UCI-mfeat-fac	0.6871±0.0018	0.5791±0.0537	0.3002±0.0354	**0.6883±0.0092**
UCI-mfeat-fou	0.7016±0.0157	0.6186±0.0333	0.5917±0.0383	**0.7202±0.0013**
UCI-mfeat-pix	0.7056±0.0268	0.6024±0.0951	0.5910±0.0297	**0.7539±0.0505**
UCI-mfeat-zer	0.5491±0.0019	0.4936±0.0489	0.4024±0.0363	**0.5506±0.0121**

[9] C. H. Q. Ding, T. Li, W. Peng, and H. Park. Orthogonal Nonnegative Matrix Tri-factorizations for Clustering. In *Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 126–135, Philadelphia, USA, August 2006.

[10] R. Fletcher and C. M. Reeves. Function Minimization by Conjugate Gradients. *The Computer Journal*, 7(2):149–154, 1964.

[11] H. Gao, F. Nie, T. W. Cai, and H. Huang. Robust Capped Norm Nonnegative Matrix Factorization: Capped Norm NMF. In *Proc. of the 24th ACM*

International on Conference on Information and Knowledge Management (CIKM 2015), pages 871–880, Melbourne, Australia, October 2015.

[12] D. Goldfarb, Z. Wen, and W. Yin. A Curvilinear Search Method for p-Harmonic Flows on Spheres. *SIAM Journal on Imaging Sciences*, 2(1):84–109, 2009.

[13] N. Guan, D. Tao, Z. Luo, and B. Yuan. Manifold Regularized Discriminative Nonnegative Matrix Factorization With Fast Gradient Descent. *IEEE Transactions on Image Processing*, 20(7):2030–2048, 2011.

[14] W. W. Hager and H. Zhang. A Survey of Nonlinear Conjugate Gradient Methods. *Pacific Journal of Optimization*, 2(1):35–58, 2006.

[15] M. R. Hestenes and E. Stiefel. Methods of Conjugate Gradients for Solving Linear Systems. *Journal of the Research of the National Bureau of Standards*, 49(6):409–436, 1952.

[16] C. Hsieh and I. S. Dhillon. Fast coordinate descent methods with variable selection for non-negative matrix factorization. In *Proc. of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2011)*, pages 1064–1072, San Diego, USA, August 2011.

[17] J. Huang, F. Nie, H. Huang, and C. H. Q. Ding. Robust Manifold Nonnegative Matrix Factorization. *ACM Transactions on Knowledge Discovery from Data*, 8(3):11, 2013.

[18] D. Kong, C. H. Q. Ding, and H. Huang. Robust Nonnegative Matrix Factorization using L21-Norm. In *Proc. of the 20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, pages 673–682, Glasgow, United Kingdom, October 2011.

[19] Y. Koren, R. M. Bell, and C. Volinsky. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*, 42(8):30–37, 2009.

[20] D. D. Lee and H. S. Seung. Learning the Parts of Objects by Non-negative Matrix Factorization. *Nature*, 401(6755):788–791, 1999.

[21] Z. Li, X. Wu, and H. Peng. Nonnegative Matrix Factorization on Orthogonal Subspace. *Pattern Recognition Letters*, 31(9):905–911, 2010.

[22] L. Lovász and M. Plummer. *Matching Theory*. North Holland, Budapest, 1986.

[23] S. A. Nene, S. K. Nayar, H. Murase, et al. Columbia object image library (COIL-20). Technical report.

[24] J. Nocedal and J. Wright Stephen. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering, Springer, 2006.

[25] P. Paatero and U. Tapper. Positive Matrix Factorization: A Non-negative Factor Model with Optimal Utilization of Error Estimates of Data Values. *Environmetrics*, 5(2):111–126, 1994.

[26] F. Pompili, N. Gillis, P. Absil, and F. Glineur. Two algorithms for orthogonal nonnegative matrix factorization with application to clustering. *Neurocomputing*, 141:15–25, 2014.

[27] B. Shen and L. Si. Non-Negative Matrix Factorization Clustering on Multiple Manifolds. In *Proc. of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010)*, Atlanta, USA, July 2010.

[28] T. Sim, S. Baker, and M. Bsat. The CMU Pose, Illumination, and Expression Database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003.

[29] F. Sun, M. Xu, X. Hu, and X. Jiang. Graph Regularized and Sparse Nonnegative Matrix Factorization with Hard Constraints for Data Representation. *Neurocomputing*, 173:233–244, 2016.

[30] M. van Breukelen, R. P. W. Duin, D. M. J. Tax, and J. E. den Hartog. Handwritten digit recognition by combined classifiers. *Kybernetika*, 34(4):381–386, 1998.

[31] B. Vandereycken. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

[32] Y. Wang and Y. Zhang. Nonnegative Matrix Factorization: A Comprehensive Review. *IEEE Trans. Knowl. Data Eng.*, 25(6):1336–1353, 2013.

[33] W. Xu, X. Liu, and Y. Gong. Document Clustering Based on Non-Negative Matrix Factorization. In *Proc. of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 267–273, Toronto, Canada, July 2003.

[34] Z. Yang and E. Oja. Linear and Nonlinear Projective Nonnegative Matrix Factorization. *IEEE Transactions on Neural Networks*, 21(5):734–749, 2010.

[35] H. Zhang and W. W. Hager. A Nonmonotone Line Search Technique and Its Application to Unconstrained Optimization. *SIAM Journal on Optimization*, 14(4):1043–1056, 2004.

[36] S. Zhong and J. Ghosh. Generative Model-based Document Clustering: A Comparative Study. *Knowledge and Information Systems*, 8(3):374–384, 2005.