NON-NEGATIVE MATRIX FACTORIZATION BASED ON PROJECTED NONLINEAR CONJUGATE GRADIENT ALGORITHM

Wenwu Wang
Centre for Vision Speech and Signal Processing
University of Surrey
Guildford, GU2 7XH, U.K.
w.wang@surrey.ac.uk

Xuan Zou
Centre for Vision Speech and Signal Processing
University of Surrey
Guildford, GU2 7XH, U.K.
xuan.zou@surrey.ac.uk

ABSTRACT

The popular multiplicative algorithms in non-negative matrix factorization (NMF) are known to have slow convergence. Several algorithms have been proposed to improve the convergence of iterative algorithms in NMF, such as the projected gradient algorithms. However, these algorithms also suffer a common problem, that is, a previously exploited descent direction may be searched again in subsequent iterations which potentially leads to slow convergence of these algorithms. In this paper, we propose a projected non-linear conjugate gradient algorithm using orthogonal searching directions at each iteration which ensures each descent direction is different from others. The algorithm is shown to have better convergence performance as compared with both multiplicative algorithms and the projected gradient algorithms.

Keywords: Non-negative matrix factorization, projected non-linear conjugate gradient, projected gradient.

1 INTRODUCTION

Non-negative matrix factorization (NMF) attempts to decompose a non-negative data matrix into a product of non-negative matrices [1]. It is especially useful for finding latent structures or features from original data. Due to its interesting properties, this technique has found many applications in, for example, source separation, dimensionality reduction, clustering, and pattern recognition [1]-[7].

Given an $M \times N$ non-negative matrix $\mathbf{X} \in \mathbb{R}_+^{M \times N}$, the goal of NMF is to find nonnegative matrices $\mathbf{W} \in \mathbb{R}_+^{M \times R}$ and $\mathbf{H} \in \mathbb{R}_+^{R \times N}$, such that $\mathbf{X} \approx \mathbf{WH}$, where R is the rank of the factorization, generally chosen to be smaller than M (or N), or akin to $(M+N)R < MN$, which results in the extraction of some latent features whilst reducing some redundancies in the input data. A commonly used cost function for finding such $\mathbf{W}$ and $\mathbf{H}$ is the squared Euclidean distance based on the reconstruction error between $\mathbf{X}$ and $\mathbf{WH}$, given by

$$\mathcal{F}(\mathbf{W}, \mathbf{H}) = \frac{1}{2} \|\mathbf{X} - \mathbf{WH}\|_F^2 \quad (1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

To minimize criterion (1), Lee and Seung developed a multiplicative algorithm in [1]. However, this algorithm is known to have slow convergence, and easily get stuck in local minima [2]. Several methods have been proposed to improve the convergence of such an iterative algorithm, such as the projected gradient (PG) algorithms with a fixed step-size [4], or with a variable step-size controlled by the Armijo rule and the Lin rule [7], the projected alternating least squares (ALS) algorithms [6], and the projected Newton method [2].

In this paper, we propose a projected nonlinear conjugate gradient (PNCG) algorithm for the optimisation of (1) in an alternating manner. The proposed algorithm has the potential of overcoming a common problem with (projected) gradient descent algorithms, such as [4] [7], that is, the gradient search direction in an iteration may still be searched again in later iterations, which makes the gradient descent potentially slow in these algorithms. The proposed algorithm, however, avoids such problems by following orthogonal (conjugate) search directions, which guarantees that a searched direction will not be checked again in a later stage.

2 EXISTING GRADIENT-BASED ALGORITHMS

Many NMF algorithms based on the optimisation of the cost function $\mathcal{F}$ in (1) have been developed. In this section, we focus on a group of gradient-based algorithms. We will discuss both their advantages and limitations, with a particular interest on the convergence properties of these algorithms. The analysis motivates us to propose a new algorithm based on the projected nonlinear conjugate gradient (PNCG) technique, which will be presented in details in Section 3.

2.1 Multiplicative Learning Methods

The most well-known algorithm for the minimisation of criterion (1) is probably the multiplicative learning algorithm by Lee and Seung, presented in their seminal paper

[1]. The algorithm is summarized in Table 1, where q is the iteration index, and $\odot$ and $\oslash$ denote the element-wise product and division between matrices respectively. This elegant yet simple algorithm is easy to implement.

Table 1: Multiplicative method (Lee-Seung)

1. Initialize $\mathbf{H}^1 \in \mathbb{R}_+^{M\times R}$, $\mathbf{W}^1 \in \mathbb{R}_+^{R\times N}$
2. Iterations, for $q = 1, 2, ...$
$\mathbf{H}^{q+1} = \mathbf{H}^q \odot ((\mathbf{W}^q)^T\mathbf{X}) \oslash ((\mathbf{W}^q)^T\mathbf{W}^q\mathbf{H}^q)$
$\mathbf{W}^{q+1} = \mathbf{W}^q \odot (\mathbf{X}(\mathbf{H}^{q+1})^T) \oslash (\mathbf{W}^q\mathbf{H}^{q+1}(\mathbf{H}^{q+1})^T)$

If $\mathbf{W}$ and $\mathbf{H}$ are initialized to non-negative matrices, the non-negative property of $\mathbf{W}$ and $\mathbf{H}$ can be guaranteed in the following iterations due to the multiplicative operations. This algorithm is essentially derived from the gradient learning of (1), with the step-size being normalized for each elements of $\mathbf{W}$ and $\mathbf{H}$ at each iteration (see [1] for details). As will be seen in our numerical examples, this algorithm has relatively slow convergence rate.

2.2 Gradient Learning Approaches

The standard gradient descent techniques can be applied to NMF, as suggested in [3] and [4]. The algorithm is summarized in Table 2, where α_q is a fixed step-size, and $\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$ and $\nabla_{\mathbf{W}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^{q+1})$ are the gradients of $\mathcal{F}$ with respect to $\mathbf{H}$ and $\mathbf{W}$ respectively, given by

$$\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) = \mathbf{W}^{qT}(\mathbf{W}^q\mathbf{H}^q - \mathbf{V}) \tag{2}$$

$$\nabla_{\mathbf{W}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^{q+1}) = (\mathbf{W}^q\mathbf{H}^{q+1} - \mathbf{V})\mathbf{H}^{q+1T} \tag{3}$$

Due to the subtraction operation used in each iteration, the non-negative property of $\mathbf{H}$ and $\mathbf{W}$ cannot be guaranteed, and a projection orthant has to be introduced to project any negative elements back to non-negative regions. Therefore, this algorithm is essentially a simple PG method. For notational convenience, we denoted this algorithm as PG-Fix (*Fix* for using a *fixed* step-size). This algorithm also suffers the problem of slow convergence rate.

Table 2: Gradient descent method (PG-Fix)

1. Initialize $\mathbf{H}^1 \in \mathbb{R}_+^{M\times R}$, $\mathbf{W}^1 \in \mathbb{R}_+^{R\times N}$
2. Iterations, for $q = 1, 2, ...$
$\mathbf{H}^{q+1} = \max(0, \mathbf{H}^q - \alpha_q\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^q))$
$\mathbf{W}^{q+1} = \max(0, \mathbf{W}^q - \alpha_q\nabla_{\mathbf{W}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^{q+1}))$

2.3 Projected Gradient Methods With Armijo Rule

Instead of using a fixed step-size, the convergence rate of PG-Fix can be improved by using a variable step-size α_q selected by an Armijo rule [5] [9]. As shown in [7], this technique can be implemented for NMF based on the same alternating procedure adopted for Lee-Seung algorithm and PG-Fix algorithm, i.e., fixing one matrix, finding the other. The algorithm is summarized in Table 3, where $P(\mathbf{\Delta}) = \max(0, \mathbf{\Delta})$ is a projection function. The

Table 3: PG with Armijo rule (PG-Armijo)

1. Initialize $\mathbf{H}^1 \in \mathbb{R}_+^{M\times R}$, $\mathbf{W}^1 \in \mathbb{R}_+^{R\times N}$, $0 < \beta < 1$, and $0 < \sigma < 1$.
2. Iterations, for $q = 1, 2, ...$
(a) $\mathbf{H}^{q+1} = P(\mathbf{H}^q - \alpha_q\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$ where $\alpha_q = \beta^{t_q}$, and t_q is the first non-negative integer t for which eqn. (5) is statisfied.
(b) Let $\mathcal{F}$ be the form of (4), $\mathbf{X}^T$ and $(\mathbf{H}^{q+1})^T$ as input matrices. Then, $\mathbf{W}^{q+1}$ can be easily obtained by repeating the step similar to (a).

step 2(b) in Table 3 can be implemented using the same sub-routine as for step 2(a). In order to find $\mathbf{H}$, we can take $\mathbf{X}$ and $\mathbf{W}$ as constants and then solve the following least squares (LS) problem (1). Likewise, the same routine for solving the LS problem can be used for finding $\mathbf{W}$, when using the transformed criterion (4) and taking $\mathbf{X}^T$ and $\mathbf{H}^T$ as constants,

$$\mathcal{F}(\mathbf{W}) = \frac{1}{2}\left\|\mathbf{X}^T - \mathbf{H}^T\mathbf{W}^T\right\|_F^2 \tag{4}$$

In step 2(a), the Armijo rule uses the condition (5) to ensure a sufficient decrease at each iteration, therefore it provides a better convergence performance than the PG-Fix algorithm. Note that, $\langle\cdot,\cdot\rangle$ in (5) is the sum of the element-wise product of two matrices.

$$(1-\sigma)\left\langle\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{W}^q, \mathbf{H}^q), \mathbf{H}^{q+1} - \mathbf{H}^q\right\rangle + \frac{1}{2}\left\langle\mathbf{H}^{q+1} - \mathbf{H}^q, (\mathbf{W}^{qT}\mathbf{W}^q)(\mathbf{H}^{q+1} - \mathbf{H}^q)\right\rangle \leq 0 \tag{5}$$

2.4 Projected Gradient Methods With Lin Rule

To further improve the convergence speed of PG-Armijo algorithm, Lin [7] suggested to use α_{q-1} as an initial guess for α_q, which has the advantage of taking fewer steps to find α_q. The algorithm (PG-Lin in brevity) is detailed in Table 4, which is similar to PG-Armijo with the major difference lying in the choice of α_q.

3 PROJECTED NONLINEAR CONJUGATE GRADIENT METHOD

A common problem with gradient descent algorithms is their relatively slow convergence rate. The major reason for this problem is that steepest descent often takes many steps in the same direction that has been searched already. A notorious method to avoid such a problem is to use orthogonal searching direction at each step, which ensures that a descent direction, if being searched already, will not be considered in later iterations. This method is known as CG [8] [9]. In this section, we develop a projected nonlinear CG algorithm for NMF. Again, we use the same procedure as adopted in the aforementioned algorithms, i.e., solving the LS problems (1) and (4) in an alternating manner. Nevertheless, the gradient at each iteration

Table 4: PG with Lin rule (PG-Lin)

1. Initialize $\mathbf{H}^1 \in \mathbb{R}_+^{M\times R}$, $\mathbf{W}^1 \in \mathbb{R}_+^{R\times N}$, $0 < \beta < 1$, and $0 < \sigma < 1$.
2. Iterations, for q =1, 2, ...
   (a) Set $i = 0$, $k = 0$; Assign $\alpha_q \leftarrow \alpha_{q-1}$
   (b) If α_q satisfies (5), repeatedly do $\alpha_q \leftarrow \alpha_q/\beta$ until α_q does not satisfy (5) or $\mathbf{H}^q$ remains unchanged when updating α_q
   Else repeatedly decrease α_q by $\alpha_q \leftarrow \alpha_q \cdot \beta$ until α_q satisfies (5).
   (c) Set $\mathbf{H}^{q+1} = P(\mathbf{H}^q - \alpha_q \nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q))$
   (d) Let $\mathcal{F}$ be the form of (4), $\mathbf{X}^T$ and $(\mathbf{H}^{q+1})^T$ as input matrices. Then, $\mathbf{W}^{q+1}$ can be easily obtained by repeating steps similar to (a)-(c).

is taken to be orthogonal (conjugate) to all the previous gradients. Suppose we consider criterion (1), which is a quadratic function of $\mathbf{H}$. To minimise (1), $\mathbf{H}$ is updated iteratively as

$$\mathbf{H}^{q+1} = \mathbf{H}^q + \alpha_q \mathbf{D}^q \tag{6}$$

where $\mathbf{D}^q$ is the search direction. Different from the steepest descent method where $\mathbf{D}^q$ is taken as the negate of the gradient, that is, $\mathbf{D}^q = -\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$, $\mathbf{D}^q$ in conjugate gradient is updated iteratively as

$$\mathbf{D}^q = -\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) + \beta_q \mathbf{D}^{q-1} \tag{7}$$

which is a combination of the negated gradient at current iteration and the search direction obtained from the previous iteration [9]. For the choice of β_q in (7), we employ the Fletcher-Reeves formula

$$\beta_q = \frac{\mathbf{r}_{q+1}^T \mathbf{r}_{q+1}}{\mathbf{r}_q^T \mathbf{r}_q} \tag{8}$$

where $\mathbf{r}_q$ =vec$(-\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q))$, where vec is an operator for stacking the column vectors of a matrix into a single vector. Note that, the Polak-Ribiére formula can also be used for the adaptation of β_q, which generally converges more quickly but may also cycle infinitely in few cases [8]. The step-size α_q in (6) can be found by the minimisation of $\mathcal{F}$ when varying α, i.e.,

$$\alpha_q = \min_{\alpha} \mathcal{F}(\mathbf{H}^q + \alpha \mathbf{D}^q) \tag{9}$$

This can be achieved by performing line search based on the Armijo rule. Here, we use the Newton-Raphson method. That is,

$$\alpha_q = \frac{\mathbf{r}_q^T \mathbf{d}_q}{\mathbf{d}_q^T \nabla_{\mathbf{H}}^2 \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) \mathbf{d}_q} \tag{10}$$

where $\mathbf{d}_q$ = vec$(\mathbf{D}^q)$, and $\nabla_{\mathbf{H}}^2 \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$ is a Hessian matrix, given by

$$\nabla_{\mathbf{H}}^2 \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) = \mathbf{I} \otimes (\mathbf{W}^{qT} \mathbf{W}^q) \tag{11}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is an identity matrix, and $\otimes$ is the Kronecker product. As the update equation (6) cannot guarantee the non-negativity of $\mathbf{H}$, we need the same operation as in Table 3 to project the negative orthants back to non-negative regions. The proposed algorithm can be implemented in Table 5, where I_{MAX} and J_{MAX} are pre-defined maximum iteration numbers for performing line search in order to find β_q and α_q. Note that, in Table 5, we use the negate of the gradient to reset the search direction every K_{MAX} (pre-defined) inner-iterations, or when the search direction is not a descent direction. The algorithm stops iterations when the following condition is satisfied,

$$\left| \mathcal{F}(\mathbf{W}^{q+1}, \mathbf{H}^{q+1}) - \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) \right| < \tau \tag{12}$$

where τ is a small constant for measuring the convergence tolerance. The stopping condition (12) is also applied to the other four algorithms discussed in this paper.

Table 5: The proposed PNCG method

1. Initialize $\mathbf{H}^1 \in \mathbb{R}_+^{M\times R}$, $\mathbf{W}^1 \in \mathbb{R}_+^{R\times N}$, I_{MAX}, J_{MAX}, K_{MAX}, $0 < \varepsilon < 1$, $0 < \epsilon < 1$
2. Iterations, for q =1, 2, ..., set $i = 0$, $k = 0$
   (a) $\nabla_{\mathbf{H}}^2 \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) = \mathbf{I} \otimes (\mathbf{W}^{qT} \mathbf{W}^q)$, $\mathbf{I} \in \mathbb{R}^{N\times N}$
   (b) Set $\mathbf{R} = -\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$, $\mathbf{D} = \mathbf{R}$
   (c) Set $\mathbf{r}$ =vec$(\mathbf{R})$, and $\mathbf{d}$ =vec$(\mathbf{D})$
   (d) Set $\varphi_{new} = \mathbf{r}^T \mathbf{r}$, $\varphi_0 = \varphi_{new}$
   (e) While $i < I_{MAX}$ and $\varphi_{new} > \varepsilon^2 \varphi_0$ do
      Set $\phi = \mathbf{d}^T \mathbf{d}$
      For $j = 1 : J_{MAX}$
         $\mathbf{R} = -\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$, $\mathbf{r}$ =vec$(\mathbf{R})$
         $\alpha = \frac{\mathbf{r}^T \mathbf{d}}{\mathbf{d}^T \nabla_{\mathbf{H}}^2 \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q) \mathbf{d}}$
         $\mathbf{H}^q = \mathbf{H}^q + \alpha \mathbf{D}$, $\mathbf{H}^q = \max(0, \mathbf{H}^q)$
         if $\alpha\phi \le \epsilon^2$ break; end if
      End for
      $\mathbf{R} = -\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{W}^q, \mathbf{H}^q)$, $\mathbf{r}$ =vec$(\mathbf{R})$
      $\varphi_{old} = \varphi_{new}$, $\varphi_{new} = \mathbf{r}^T \mathbf{r}$, $\beta = \varphi_{new}/\varphi_{old}$
      $\mathbf{D} = \mathbf{R} + \beta \mathbf{D}$, $\mathbf{d}$ =vec$(\mathbf{D})$, $k = k + 1$
      If $k = K_{MAX}$ and $\mathbf{r}^T \mathbf{d} \le 0$
         $\mathbf{D} = \mathbf{R}$, $\mathbf{d}$ =vec$(\mathbf{D})$, $k = 0$
      End if
      $i = i + 1$
   End while
   (f) Set $\mathbf{H}^{q+1} = \mathbf{H}^q$
   (g) Let $\mathcal{F}$ be the form of (4), $\mathbf{X}^T$ and $(\mathbf{H}^{q+1})^T$ as input matrices. Then, $\mathbf{W}^{q+1}$ can be easily obtained by repeating steps similar to (a)-(f).

4 SIMULATIONS

In this section, we provide a numerical example to show the convergence performance of the proposed PNCG algorithm, as compared with Lee-Seung, PG-Armijo, PG-Lin and PG-Fix algorithms. The major parameters of these algorithms are set in Table 6. To perform the test, we generated synthetic non-negative data matrix $\mathbf{X}$ as a 30-by-20 matrix with its elements drawn from zero mean and unit norm Gaussian distribution. The iterations required for convergence of these algorithms were measured by (12). Figure 1 shows the evolution of the cost functions versus

the number of iterations from the initialization until the convergence. Figure 2 is a zoom-in plot of Figure 1 from the second iteration to the number of iterations required for convergence. It can be seen from these plots that the proposed algorithm takes much fewer iterations to converge, as compared with the Lee-Seung, PG-Fix, and PG-Armijo algorithms. At the same time, it takes as many iterations as required by PG-Lin algorithm. The result is promising and clearly suggests the benefits for further investigation.

Table 6: Initialization of parameters

PNCG					
I_{MAX}	1000	J_{MAX}	20	K_{MAX}	30
ε	0.5	ϵ	0.5	τ	0.0001
PG-Lin					
β	0.1	σ	0.01	α_0	1
PG-Armijo					
β	0.1	σ	0.01	α_0	1
PG-Fix					
α_q	0.01	-	-	-	-
Common parameters for all algorithms					
M	30	N	20	R	5

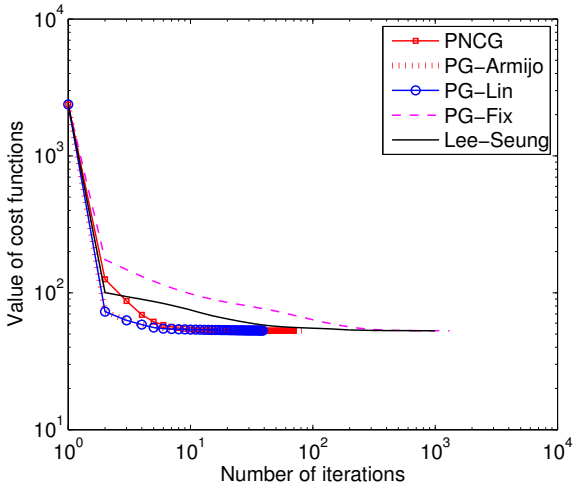

Figure 1: Convergence comparison between the proposed PNCG algorithm and the PG-Fix, PG-Armijo, PG-Lin and Lee-Seung algorithms.

5 CONCLUSIONS

Based on the analysis of several existing gradient descent algorithms for NMF, we have presented a novel algorithm, i.e., projected non-linear conjugate gradient algorithm, that has been shown to have better convergence property for synthetic data, as compared to the investigated descent methods such as the Lee-Seung, PG-Armijo, and PG-Fix algorithms. Its convergence is comparable to PG-Lin algorithm. Our future work should include applications of the proposed algorithm to real-world datasets, and further

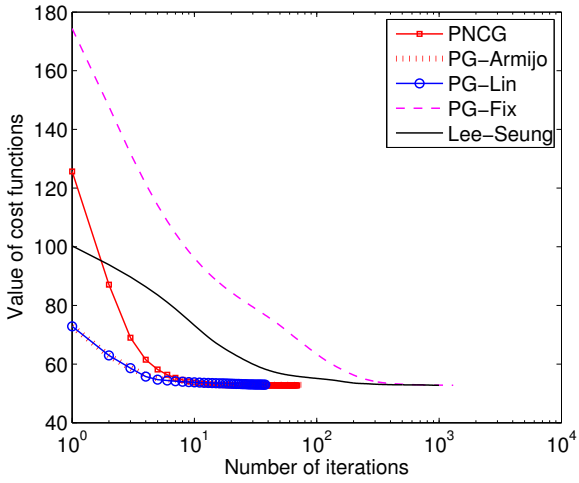

Figure 2: A zoom-in plot of Figure 1 excluding the first iteration.

investigation of its convergence affected by the key parameters used in the algorithm.

References

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

[2] R. Zdunek, A. Cichocki, "Non-negative matrix factorization with quadratic programming," *Neurocomputing*, vol. 71, pp. 2309-2320, 2007.

[3] Non-negative matrix factorization. Available online at http://www.simonshepherd.supanet.com/nnmf.htm

[4] M. Chu, F. Diele, R. Plemmons, and S. Ragni, "Optimality, computation and interpretation of non-negative matrix factorizations," Wake Forest University, 2004.

[5] D. P. Bertsekas, "On the Goldstein-Levitin-Polyak gradient projection method," *IEEE Trans. on Automatic Control*, vol. 21, pp. 174-184, 1976.

[6] M. Berry, M. Browne, A. Langville, P. Pauca, and R. Plemmons, "Algorithms and applications for approximate non-negative matrix factorization," *Computational Statistics and Data Analysis*, vol. 52, no.1, pp.155-173, 2007.

[7] C.-J. Lin, "Projected gradient methods for non-negative matrix factorization," *Neural Computation*, vol. 19, pp. 2756-2779, 2007.

[8] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Technical Report, Carnegie Mellon University, 1994. Available online at http://www.cs.cmu.edu/quake-papers/painless-conjugate-gradient.pdf

[9] D. P. Bertsekas, *Nonlinear Programming* (2nd ed.). Belmont, MA: Athena Scientific.