# A Cyclic Weighted Median Method for $L_1$ Low-Rank Matrix Factorization with Missing Entries*

**Deyu Meng**[1], **Zongben Xu**[1], **Lei Zhang**[2], **Ji Zhao**[3]
[1]Institute for Information and System Sciences and
Ministry of Education Key Lab for Intelligent Networks and Network Security
Xian Jiaotong University, Xi'an 710049, China
[2]Department of Computing, The Hong Kong Polytechnic University
[3]The Robotics Institute, Carnegie Mellon University

**Abstract**

A challenging problem in machine learning, information retrieval and computer vision research is how to recover a low-rank representation of the given data in the presence of outliers and missing entries. The $L_1$-norm low-rank matrix factorization (LRMF) has been a popular approach to solving this problem. However, $L_1$-norm LRMF is difficult to achieve due to its non-convexity and non-smoothness, and existing methods are often inefficient and fail to converge to a desired solution. In this paper we propose a novel cyclic weighted median (CWM) method, which is intrinsically a coordinate decent algorithm, for $L_1$-norm LRMF. The CWM method minimizes the objective by solving a sequence of scalar minimization sub-problems, each of which is convex and can be easily solved by the weighted median filter. The extensive experimental results validate that the CWM method outperforms state-of-the-arts in terms of both accuracy and computational efficiency.

## Introduction

Many machine learning, computer vision and statistical problems can be posed as problems of learning low dimensional linear or multi-linear models, for example, social networks (Cheng et al. 2012b), structure from motion (Tomasi and Kanade 1992), face recognition (Cheng et al. 2012a; Wright et al. 2009), collaborative filtering (Koren 2008), information retrieval (Deerwester et al. 1990), object recognition (Turk and Pentland 1991), layer extraction (Ke and Kanade 2001) and plane-based pose estimation (Sturm 2000). Methods for learning linear models can be seen as a special case of subspace fitting. If there is no missing entries, efficient algorithms based on singular value decomposition (SVD) can be used. One drawback of SVD-type of approaches is that they are based on least-square-estimation techniques and hence fail to account for outliers which are common in realistic training sets. Moreover, in many applications the data contain missing entries due to analog-to-digital converter errors (Wright et al. 2009), faulty memory locations in hardware (Ji et al. 2010) or tracking failures (Tomasi and Kanade 1992). To deal with corrupted and incomplete data, recently the $L_1$-norm low-rank matrix factorization (LRMF) is commonly used (Kwak 2008; Ding et al. 2006; Ke and Kanade 2005; Eriksson and van den Hengel 2010).

The $L_1$-norm LRMF problem is formulated as follows. Given a matrix $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n) \in \Re^{d\times n}$, where each column $\mathbf{x}_i$ is a $d$-dimensional measurement. The missing entries of $\mathbf{X}$ is indicated by a matrix $\mathbf{W} \in \Re^{d\times n}$, whose element $w_{ij}$ is 0 if the corresponding element is missing, and 1 otherwise (Buchanan and Fitzgibbon 2005). $L_1$-norm LRMF minimizes the following error[1]:

$$\min_{\mathbf{U},\mathbf{V}} \left\| \mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T) \right\|_{L_1}, \tag{1}$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k] \in \Re^{d\times k}$, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k] \in \Re^{n\times k}$, $k < d, n$ and $\odot$ denotes the Hadamard product (component-wise multiplication). Unfortunately, the above $L_1$-norm minimization is very difficult to solve. On one hand, the optimization is non-convex and the global minimum is generally difficult to find. It is even shown to be an NP-hard problem in presence of missing entries (Gillis and Glineur 2011). On the other hand, the $L_1$-norm minimization is non-smooth, so it is hard to find an efficient closed-form iteration formula by standard optimization tools (Eriksson and van den Hengel 2010).

Many recent approaches to solving the $L_1$-norm LRMF problem use variants of the Wiberg method (Eriksson and van den Hengel 2010; Okatani and Deguchi 2007; Strelow 2012). However, these general purpose methods are often inefficient, requiring too much cost to reach the minimum, especially for high-dimensional data encountered in real world. This paper presents a surprisingly simple cyclic coordinate descent (Tseng 2001) algorithm which however shows outstanding performance on $L_1$-norm LRMF. The core idea is to break the original complex minimization problem into a series of elementary simple sub-problems, each having only one scalar parameter, and then recursively optimize them. Each of these small problems is convex and can be easily solved by weighted median filter, which makes our algorithm free of the time-consuming inner loop numerical optimization. The recursively employed weighted median filters further make the method be robust to outliers

*This research was supported by 973 Program (2013CB329404), NSFC (11131006) and Ph.D. Program (20090201120056).

[1]$\|\mathbf{A}\|_{L_1}$ refers to the $L_1$ norm of the matrix (the summarization of the absolute values of all components in $\mathbf{A}$).

and missing entries to a large extent. Although there is no guarantee of convergence to the global minima for the non-convexity of LRMF problems, empirically we found that the proposed algorithm converges more often to a desired solution. Experiments on extensive synthetic and real data show the effectiveness of our approach.

Throughout the paper, we use bold uppercase, bold lowercase and non-bold letters to denote matrices, vectors and scalars, respectively.

## Previous work

The problem of bilinear factorization with missing entries has been studied in statistical analysis in the early 80's. Gabriel and Zamir (Gabriel and Zamir 1979) proposed a weighted SVD technique that uses alternated minimization (or criss-cross regression) to find the principal subspace of the data. They minimize

$$\min_{\mathbf{U},\mathbf{V}} \left\| \mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T) \right\|_F . \qquad (2)$$

In general, Eq. (2) does not have a closed-form solution in terms of a generalized eigenvalue problem. Moreover, the problem of data factorization with arbitrary weights has several local minima depending on the structure of the weights (Buchanan and Fitzgibbon 2005). Shum et al. (Shum, Ikeuchi, and Reddy 1995) further used PCA with missing entries for polyhedral object modeling. Jacobs (Jacobs 1997) proposed a method by linearly fitting the data with missing entries. Aguiar et al. (Aguiar, Stosic, and Xavier 2008) proposed a closed-form solution to the data factorization problem, when the missing entries has a special structure.

In order to handle outliers, De la Torre and Black (De la Torre and Black 2003) proposed to change the Frobenious norm to a robust function and used iterative re-weighted least squares algorithms to solve it. Unfortunately, the problem is non-convex and it is sensitive to the initialization. Ding et al. (Ding et al. 2006) proposed to use the rotational invariant $R_1$ norm, defined by $\|\mathbf{X}\|_{R_1} = \sum_{i=1}^{n} (\sum_{i=1}^{d} x_{ji}^2)^{1/2}$, to replace the $L_1$ norm in Eq. (1). Like the $L_1$ norm, the $R_1$ norm is also capable of weakening the contributions from outliers. By maximizing the $L_1$ dispersion of the data matrix, i.e., $\max_{\mathbf{U}} \|\mathbf{U}^T\mathbf{X}\|_{L_1}$, instead of minimizing the objective in Eq. (1), Kwak (Kwak 2008) presented the PCA-$L_1$ approach to suppressing the influence of outliers in matrix factorization to some extent. These methods, however, always lose effectiveness for LRMF problems with missing entries. Ke and Kanade (Ke and Kanade 2005) proposed an alternated linear/quatratic programming (ALP/AQP) method to solve the $L_1$-norm LRMF problem with missing entries. Eriksson and Hengel (Eriksson and van den Hengel 2010; 2012) experimentally revealed that the alternated convex programming approach frequently converges to a point that is not a local minimum (typically, the evolution of the $L_1$-norm cost function stops after a small number of iterations), and thus they introduced a Wiberg-$L_1$ approach, which is an extension of Wiberg method for $L_1$ minimization. This method shows good performance on some synthetic simulations and structure from motion problems. Candés et al. (Candès et al. 2011; Wright et al. 2009) proposed a robust PCA method using recent advances in rank minimization. They modeled the observed data as the sum of a low-rank clean data matrix and a sparse outlier matrix. A major advantage of this approach is the convex formulation. However, the numerical algorithms (e.g., augmented lagrange, fix-point) require computing an SVD in each iteration which always makes its complexity comparatively high.

## Robust matrix factorization via cyclic weighted median

Recent work on compressed sensing (Friedman, Hastie, and Höfling 2007; Friedman, Hastie, and Tibshirani 2010) has shown how coordinate descent can be an effective algorithm to minimize convex non-smooth functions (e.g., Lasso). In multivariate minimization, coordinate descent methods minimize the objective by solving a sequence of scalar minimization subproblems. Each subproblem improves the estimate of the solution by minimizing along a selected coordinate with all other coordinates fixed. The method is analogous to the Gauss-Seidel iterative algorithm for solving linear systems of equations (Jeffreys and Jeffreys 1988). Coordinate descent is particularly attractive when the subproblems can be solved quickly. In our case, each subproblem in the cyclic coordinate descent can be solved easily by weighted median filter, and thus we call our method cyclic weighted median method or CWM in short.

### Subproblem decomposition

Let's first equivalently reformulate the objective function of (1) as the following two decomposed expressions:

$$\begin{aligned} \left\| \mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T) \right\|_{L_1} &= \left\| \mathbf{W} \odot (\mathbf{X} - \textstyle\sum_{j=1}^{k} \mathbf{u}_j \mathbf{v}_j^T) \right\|_{L_1} \\ = \left\| \mathbf{W} \odot (\mathbf{E}_i - \mathbf{u}_i \mathbf{v}_i^T) \right\|_{L_1} &= \textstyle\sum_{j=1}^{n} \left\| \mathbf{w}_j \odot (\mathbf{e}_j^i - \mathbf{u}_i v_{ij}) \right\|_{L_1} \\ = \textstyle\sum_{j=1}^{n} \left\| \mathbf{w}_j \odot \mathbf{e}_j^i - \mathbf{w}_j \odot \mathbf{u}_i v_{ij} \right\|_{L_1} , \end{aligned}$$

and

$$\begin{aligned} \left\| \mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T) \right\|_{L_1} &= \left\| \mathbf{W} \odot (\mathbf{X}^T - \textstyle\sum_{j=1}^{k} \mathbf{v}_j \mathbf{u}_j^T) \right\|_{L_1} \\ = \left\| \mathbf{W} \odot (\mathbf{E}_i^T - \mathbf{v}_i \mathbf{u}_i^T) \right\|_{L_1} &= \textstyle\sum_{j=1}^{d} \left\| \widetilde{\mathbf{w}}_j \odot (\widetilde{\mathbf{e}}_j^i - \mathbf{v}_i u_{ij}) \right\|_{L_1} \\ = \textstyle\sum_{j=1}^{d} \left\| \widetilde{\mathbf{w}}_j \odot \widetilde{\mathbf{e}}_j^i - \widetilde{\mathbf{w}}_j \odot \mathbf{v}_i u_{ij} \right\|_{L_1} , \end{aligned}$$

where

$$\mathbf{E}_i = \mathbf{X} - \sum_{j \neq i} \mathbf{u}_j \mathbf{v}_j^T , \qquad (3)$$

$\mathbf{w}_j$ and $\widetilde{\mathbf{w}}_j$ are the $j$-th column and row vectors of the indicator matrix $\mathbf{W}$, respectively, $\mathbf{e}_j^i$ and $\widetilde{\mathbf{e}}_j^i$ are the $j$-th column and row vectors of $\mathbf{E}_i$, respectively, and $u_{ij}$ and $v_{ij}$ are the $j$-th components of $\mathbf{u}_i$ and $\mathbf{v}_i$, respectively.

It is then easy to separate the original large minimization problem (1), which is with respect to $\mathbf{U}$ and $\mathbf{V}$, into a series of small minimization problems, which are each with respect to only one scalar parameter $u_{ij}$ ($i = 1, 2, \cdots, k, j = 1, 2, \cdots, d$) and $v_{ij}$ ($i = 1, 2, \cdots, k, j = 1, 2, \cdots, n$), expressed as

$$\min_{v_{ij}} \left\| \mathbf{w}_j \odot \mathbf{e}_j^i - \mathbf{w}_j \odot \mathbf{u}_i v_{ij} \right\|_{L_1} \qquad (4)$$

and

$$\min_{u_{ij}} \left\| \widetilde{\mathbf{w}}_j \odot \widetilde{\mathbf{e}}_j^i - \widetilde{\mathbf{w}}_j \odot \mathbf{v}_i u_{ij} \right\|_{L_1} . \qquad (5)$$

Fortunately, both the minimization problems in (4) and (5) are not only convex, but also can be easily solved by weighted median filter. This implies that it is possible to construct a fast coordinate decent algorithm for $L_1$-norm LRMF problem (1), as presented in the following sub-sections.

## Solving subproblems by weighted median filter

For the convenience of expression, we formulate Eq. (4) and Eq. (5) as the following form:

$$\min_v f_{\mathbf{e},\mathbf{u}}(v) = \|\mathbf{e} - \mathbf{u}v\|_{L_1}, \tag{6}$$

where $\mathbf{e}$ and $\mathbf{u}$ are $d$-dimensional vectors, and their $i$-th elements are $e_i$ and $u_i$, respectively. It is evident that $f_{\mathbf{e},\mathbf{u}}(z)$ is convex since each of its components $|e_i - u_i z|$ is convex, and the positive sum of convex functions is still convex.

Without loss of generality, we assume that all $u_i$s are nonzeros. Then let's reformulate Eq. (6) as follows:

$$\|\mathbf{e} - \mathbf{u}v\|_{L_1} = \sum\nolimits_{i=1}^{d} |e_i - u_i v| = \sum\nolimits_{i=1}^{d} \left( |u_i| \left| v - \frac{e_i}{u_i} \right| \right).$$

It is interesting to see that the problem actually corresponds to a weighted median problem (Brownrigg 1984), and can be easily solved by applying the weighted median filter on the sequence $\{\frac{e_i}{u_i}\}_{i=1}^{d}$ under weights $\{|u_i|\}_{i=1}^{d}$. The global solution of Eq. (6) (i.e., Eq. (4) and Eq. (5)) can then be exactly and efficiently obtained in time and space complexity $O(d)$ (Rauh and Arce 2010).

We can now construct the whole CWM algorithm for solving the $L_1$-norm LRMF problem (1).

## CWM algorithm for $L_1$-norm LRMF problem

The main idea of the CWM algorithm for solving the minimization in Eq. (1) is to recursively apply weighted median filter to update each element of $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k) \in \Re^{d\times k}$ and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k) \in \Re^{n\times k}$. The steps of the algorithm are described as follows:

- Cyclicly apply the weighted median filter to update each entry $v_{ij}$ $(i = 1, \cdots, k, j = 1, \cdots, n)$ of $\mathbf{V}$ with all the other components of $\mathbf{U}$ and $\mathbf{V}$ fixed by solving

  $$v_{ij}^{*} = \arg\min_{v_{ij}} \left\| (\mathbf{w}_j \odot \mathbf{e}_j^i - \mathbf{w}_j \odot \mathbf{u}_i v_{ij}) \right\|_{L_1},$$

  where $\mathbf{w}_j$ is the $j$-th column vector of $\mathbf{W}$ and $\mathbf{e}_j^i$ is the $j$-th column vector of $\mathbf{E}_i$ defined in (3).
- Cyclicly apply the weighted median filter to update each entry $u_{ij}$ $(i = 1, \cdots, k, j = 1, \cdots, d)$ of $\mathbf{U}$ with all the other components of $\mathbf{U}$ and $\mathbf{V}$ fixed by solving

  $$u_{ij}^{*} = \arg\min_{u_{ij}} \left\| \widetilde{\mathbf{w}}_j \odot \widetilde{\mathbf{e}}_j^i - \widetilde{\mathbf{w}}_j \odot \mathbf{v}_i u_{ij} \right\|_{L_1},$$

  where $\widetilde{\mathbf{w}}_j$ denotes the $j$-th row vector of $\mathbf{W}$ and $\widetilde{\mathbf{e}}_j^i$ denotes the $j$-th row vector of $\mathbf{E}_i$.

Through iteratively implementing the above procedures, the factorized matrices $\mathbf{U}$ and $\mathbf{V}$ can be recursively updated until the termination condition is satisfied. We summarize the aforementioned CWM algorithm in Algorithm 1.

As for the initialization of $\mathbf{U}$ and $\mathbf{V}$ in step 1 of the algorithm, in our experiments, we just simply use random initialization, and the proposed algorithm performs very well in all our experiments. And for the terminate condition in step 4 of the algorithm, since the objective function of Eq. (1) decreases monotonically in the iteration process, the algorithm can be reasonably terminated when the updating rate of $\mathbf{U}$ or $\mathbf{V}$ is smaller than some preset threshold, or the maximum number of iterations is reached.

**Algorithm 1**: CWM algorithm for solving Problem (1)

```
Given: $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n) \in \Re^{d\times n}$, $\mathbf{W}$
Execute:
  1. Randomly initialize $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_k) \in \Re^{d\times k}$
     and $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_k) \in \Re^{n\times k}$.
  2. Cyclicly apply weighted median to update each
     entry $v_{ij}$ $(i = 1, \cdots, k, j = 1, \cdots, n)$ of $\mathbf{V}$ with
     all the other components of $\mathbf{U}$ and $\mathbf{V}$ fixed.
  3. Cyclicly apply weighted median to update each
     entry $u_{ij}$ $(i = 1, \cdots, k, j = 1, \cdots, d)$ of $\mathbf{U}$ with
     all the other components of $\mathbf{U}$ and $\mathbf{V}$ fixed.
  4. Iterate steps 2 and 3 until convergence.
```

## Convergence and computational complexity

We first prove that the proposed algorithm converges to a coordinate-wise minimum point (Tseng 2001; Breheny and Huang 2011) of the problem. The coordinate-wise minimum point is defined as:

**Definition 1** *For function $f : \Re^d \to \Re$, we say that $\mathbf{z} = (z_1, z_2, \cdots, z_d)^T$ is a coordinate-wise minimum point of $f$ if for any $\alpha \in R$ and all $i = \{1, 2, \cdots, d\}$,*

$$f(\mathbf{z}) = \min_\alpha f(z_1, \cdots, z_{i-1}, \alpha, z_{i+1}, \cdots, z_d).$$

The convergence of the proposed algorithm is evaluated in the following theorem.

**Theorem 1** *Algorithm 1 converges to a coordinate-wise minimum point of the $L_1$-norm LRMF problem in (1).*

**Proof**: In steps 2 and 3 of Algorithm 1, we can exactly attain the global minimums of (4) and (5) with respect to one single entry of $\mathbf{V}$ and $\mathbf{U}$, with all the other ones fixed, respectively, by weighted median filter. Therefore, in each of the iterations between steps 2 and 3, the objective function of (1) is monotonically decreasing. Since this objective function is lower bounded by 0, the algorithm is guaranteed to be convergent to a point $(\mathbf{U}^*, \mathbf{V}^*)$.

Denote by

$$\begin{aligned} f(\mathbf{U}, \mathbf{V}) &= f(u_{11}, \cdots, u_{d,k}, v_{11}, \cdots, v_{n,k}) \\ &= \left\| \mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T) \right\|_{L_1}. \end{aligned}$$

Since for any entry $u_{ij}^*$ $(i = 1, \cdots, d, j = 1, \cdots, k)$ of $\mathbf{U}^*$ and $v_{ij}^*$ $(i = 1, \cdots, n, j = 1, \cdots, k)$ of $\mathbf{V}^*$, the updating in steps 2 and 3 of Algorithm 1 converges, we have

$$\begin{aligned} v_{ij}^* &= \arg\min_{v_{ij}} \left\| (\mathbf{w}_j \odot (\mathbf{e}_j^i)^* - \mathbf{w}_j \odot \mathbf{u}_i^* v_{ij}) \right\|_{L_1} \\ &= \arg\min_{v_{ij}} f(u_{11}^*, \cdots, u_{d,k}^*, v_{11}^*, \cdots, \\ &\qquad v_{i,j-1}^*, v_{ij}, v_{i,j+1}^*, \cdots, v_{n,k}^*), \end{aligned}$$

where $\mathbf{u}_i^*$ is the $j$-th column vector of $\mathbf{U}^*$ and $(\mathbf{e}_j^i)^*$ is the $j$-th column vector of $\mathbf{E}_i^* = \mathbf{X} - \sum_{j \neq i} \mathbf{u}_j^* \mathbf{v}_j^{*T}$, and

$$\begin{aligned} u_{ij}^* &= \arg\min_{u_{ij}} \left\| \widetilde{\mathbf{w}}_j \odot (\widetilde{\mathbf{e}}_j^i)^* - \widetilde{\mathbf{w}}_j \odot \mathbf{v}_i^* u_{ij} \right\|_{L_1} \\ &= \arg\min_{v_{ij}} f(u_{11}^*, \cdots, u_{i,j-1}^*, u_{ij}, u_{i,j+1}^*, \\ &\qquad\qquad \cdots, u_{d,k}^*, v_{11}^*, \cdots, v_{n,k}^*), \end{aligned}$$

where $\mathbf{v}_i^*$ is the $j$-th column vector of $\mathbf{V}^*$ and $(\widetilde{\mathbf{e}}_j^i)^*$ denotes the $j$-th row vector of $\mathbf{E}_i^*$. It is then easy to conduct that $(\mathbf{U}^*, \mathbf{V}^*)$ corresponds to a coordinate-wise minimum point of $f(\mathbf{U}, \mathbf{V})$ based on Definition 1. ■

We now discuss the space and time complexities of Algorithm 1. It is easy to see that both complexities are essentially determined by steps 2 and 3 of Algorithm 1, i.e., the cyclic utilization of the weighted median filter. To compute each element $v_{ij}$ of $\mathbf{V}$ ($i = 1, \cdots, k, j = 1, \cdots, n$) in step 2, the weighted median calculation needs around $O(d)$ time and space, respectively (Rauh and Arce 2010). Updating the entire $\mathbf{V}$ thus costs $O(knd)$ in time and $O(knd)$ in space, respectively. Similarly, updating $\mathbf{U}$ in step 3 also needs $O(knd)$ time and $O(knd)$ space costs, respectively. The entire time and space complexities of Algorithm 1 are thus $O(Tknd)$ and $O(knd)$, respectively, where $T$ is the number of iterations for convergence. That is, both the time and space complexities of the proposed algorithm are linear in both the size $n$ and the dimensionality $d$ of the input data.

A very interesting point is that when the input data contain a large extent of missing entries, the intrinsic computational complexity of the proposed algorithm, in terms of both time and space, is always much lower than the above evaluation. Let's denote by $d'$ the maximal number of nonzero elements in all column vectors (i.e., $\mathbf{w}_i$ in (4)) of $\mathbf{W}$ and by $n'$ that of all its row vectors (i.e., $\widetilde{\mathbf{w}}_i$ in (5)). It is easy to see that solving (4) and (5) by weighted median actually needs only $O(d')$ and $O(n')$ time and space complexities, respectively. If we denote $s = max(d', n')$, it is easy to obtain that the entire time/space complexity of Algorithm 1 is at most $O(Tk(d+n)s)$ and $O(k(d+n)s)$ (instead of $O(Tknd)$ and $O(knd)$ in non-missing entry cases), respectively. This means that when the input data are highly sparse, i.e., $s \ll d$ and $s \ll n$, the computational complexity of the algorithm can be further reduced. Such a computational complexity makes the proposed algorithm well-suited in solving $L_1$-norm LRMF problems with missing entries. This property also implies the great potentials of our algorithm in real big and sparse data applications, such as text mining (Li and Yeung 2009) and gene analysis (Sinha 2010).

## Difference with the ALP method

We would like to stress that although the idea of the proposed CWM method looks somewhat similar to the ALP method (Ke and Kanade 2005), the mechanisms of the two methods are very different. We briefly introduce the implementation of ALP as follows. First, ALP decomposes the objective function of $L_1$-norm LRMF into a series of independent sub-problems as:

$$\left\| \mathbf{w}_j \odot (\mathbf{x}_j - \mathbf{U}\mathbf{v}_j) \right\|_1, j = 1, 2, \cdots, n \quad (7)$$

and

$$\left\| \widetilde{\mathbf{w}}_j \odot (\widetilde{\mathbf{x}}_j - \mathbf{V}\mathbf{u}_j) \right\|_1, j = 1, 2, \cdots, d, \quad (8)$$

where $\mathbf{x}_j$ and $\widetilde{\mathbf{x}}_j$ are the $j$-th column and row vectors of $\mathbf{X}$, respectively, $\mathbf{v}_j$ and $\mathbf{u}_j$ are the $j$-th row vectors of $\mathbf{V}$ and $\mathbf{U}$, respectively. Then the ALP algorithm recursively updates each $\mathbf{v}_j$ and $\mathbf{u}_j$ (Ke and Kanade 2005).

Although both methods are constructed by alternatively optimizing the convex sub-problems of the original problem, ALP uses a linear programming or quadratic programming technique to approximately solve (7) and (8), and typically involves several inner loops and iteration, while our algorithm directly utilizes weighted median filter to exactly solve (4) and (5) without inner loops. This explains why the proposed method is always much faster and converges to a better solution than ALP in the experiments.

An even more important and intrinsic difference between the two methods is that their computational complexities will differ significantly in handling missing entries. Note that the indicator vector $\mathbf{w}_j(\widetilde{\mathbf{w}}_j)$ can be distributive over the terms $\mathbf{e}_j^i(\widetilde{\mathbf{e}}_j^i)$ and $\mathbf{u}_i(\mathbf{v}_i)$ in Eq. (4)(Eq. (5)) in the bracket, while $\mathbf{w}_j(\widetilde{\mathbf{w}}_j)$ cannot distributively multiply the terms $\mathbf{x}_j(\widetilde{\mathbf{x}}_j)$ and $\mathbf{U}(\mathbf{V})$ inside the bracket in Eq. (7)(Eq. (8)). This means that the sparsity of input data cannot be fully utilized to reduce the computational cost in minimizing (7) and (8). Actually, the LP process to solve (7) and (8) incurs in at least $O(dk)$ and $O(nk)$ computational cost, respectively. The overall cost of the algorithm will be at least $O(Tknd)$, where $T$ is the iteration number in updating (7) and (8), which is generally larger than the complexity ($O(Tk(d+n)s)$) of the proposed method in handling data with missing entries.

## Experiments

To evaluate the performance of the proposed CWM algorithm on $L_1$-norm LRMF problems, we conducted extensive experiments on various types of synthetic and real data with outliers and missing entries. All the methods were implemented in Matlab and run on a PC with Intel Core(TM) Q9300@2.50G CPU and 4GB memory.

### Synthetic data experiment

In this experiment, we synthetically generated 100 sets of low rank matrices, each with size $7 \times 12$ and rank 3. Each matrix was generated by $\mathbf{U}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are matrices of size $7 \times 3$ and $12 \times 3$, respectively. Each element of $\mathbf{U}$ and $\mathbf{V}$ was generated following Gaussian distribution with zero mean and unit variance. Then $10\%$ of the elements were randomly selected and designated as missing entries, and $10\%$ of the matrix elements were corrupted by uniformly distributed noises over $[-5, 5]$ (the same setting as (Eriksson and van den Hengel 2010)).

Four representative LRMF methods were employed for the performance comparison with the proposed CWM method. They include two classical methods for $L_2$-norm LRMF: WLRA (Srebro and Jaakkola 2003)[2] and Wiberg-$L_2$ (Okatani and Deguchi 2007)[3] and two state-of-the-art methods for

[2]We wrote the code for WLRA by ourselves.

[3]http://www.fractal.is.tohoku.ac.jp

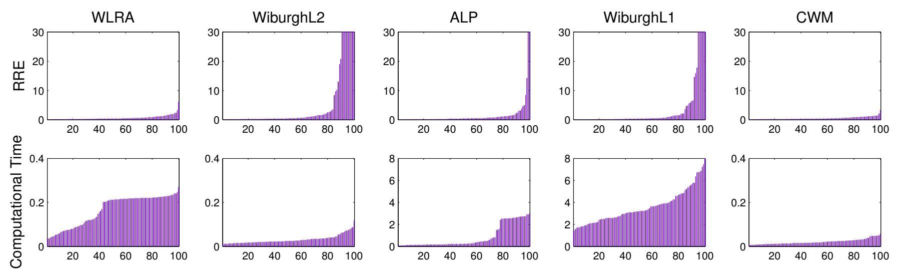

Figure 1: Comparison of relative reconstruction errors (RRE) and computational times obtained by the WLRA, Wilberg-$L_2$, ALP, Wiberg-$L_1$ and CWM methods on 100 series of synthetic data experiments. The values are sorted in an ascending order for easy visualization. For fair comparison, we plot all figures in the same scale (except for the computational time of ALP and Wiberg-$L_1$).

Table 1: Performance comparison of the competing methods in our experiments. The best result in each experiment is highlighted in bold.

| | Computational time (s) | | | | | Accuracy (RRE) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | WLRA | Wiberg-$L_2$ | ALP | Wiberg-$L_1$ | CWM | WLRA | Wiberg-$L_2$ | ALP | Wiberg-$L_1$ | CWM |
| Syntheticdata mean | 0.165 | 0.030 | 0.826 | 3.554 | **0.021** | 0.64 | 395 | 19.01 | 4.82 | **0.51** |
| Syntheticdata variance | 0.0048 | 0.0004 | 1.0592 | 2.2174 | **0.0001** | 0.63 | 14331 | 192.1 | 245.2 | **0.23** |
| Dinosaur sequence | **2.9447** | 127.37 | 24.93 | 132.79 | 10.071 | 0.4539 | 0.7749 | 0.0426 | 0.0405 | **0.0031** |
| Pingpong sequence | 224.91 | 13265 | 718.82 | 166424 | **70.950** | 0.3498 | 0.5624 | 0.0903 | 0.0755 | **0.0151** |
| Facedata1 | **0.5161** | 1380 | 32.8545 | 691.9 | 2.3120 | 0.0609 | 0.0539 | 0.0327 | 0.0417 | **0.0323** |
| Facedata2 | **0.7548** | 1069 | 44.0804 | 603.7 | 2.2234 | 0.2029 | 0.4594 | 0.1940 | 0.4185 | **0.1458** |

$L_1$-norm LRMF: ALP (Ke and Kanade 2005)[4] and Wiberg-$L_1$ (Eriksson and van den Hengel 2010)[5]. For each input matrix, the initialization was randomly generated and utilized by all competing methods. The maximum number of iterations were 100 for all methods. The means and variances of the accuracy and speed of all methods are compared in the first and second rows of Table 1. The accuracy is measured by relative reconstruction error (RRE) for the 100 realizations. The RRE (Luo, Ding, and Huang 2011; Lin, Chen, and Ma 2009; Wright et al. 2009) is calculated as: $\left\|\mathbf{X}_{ori} - \widetilde{\mathbf{U}}\widetilde{\mathbf{V}}^T\right\|_F / \left\|\mathbf{X}_{ori}\right\|_F$, where $\mathbf{X}_{ori}$ is the groundtruth data matrix. Figure 1 provides a more detailed visualization of the performance.

By observing Table 1 (the first two rows) and Figure 1, the advantage of CWM is clear in terms of both computational speed and accuracy. On one hand, our method always achieves the most accurate reconstructions of the groundtruth data matrices, and on the other hand, the computational cost of the proposed method is much lower than other competing methods. Also, from Figure 1, we see that the proposed method performs very stable and it has the smallest variance among all methods. This shows that the proposed method can always obtain a good solution to the original $L_1$-norm LRMF problem under different initializations.

## Structure from motion experiments

The structure from motion (SFM) problem can be viewed as a typical LRMF task (Ke and Kanade 2005; Eriksson and van den Hengel 2010). In this series of experiments, we employed two well known SFM sequences, the dinosaur sequence (http://www.robots.ox.ac.uk/~vgg/) and the pingpong ball sequence (http://vasc.ri.cmu.edu/idb/). The dinosaur and pingpong sequences contain projections of 336 and 839 points tracked over 36 and 226 frames, respectively, leading to a $336 \times 72$ matrix and an $839 \times 452$ matrix, respectively. Each matrix contains more than $80\%$ missing entries due to occlusions or tracking failures. In order to verify the robustness of competing methods, we further added $10\%$ outliers uniformly generated from $[-5000, 5000]$ to the two SFM data matrices. The WLRA, Wilberg-$L_2$, ALP, Wilberg-$L_1$ and CWM methods were employed for comparison. The results obtained by the competing methods are summarized in the third and fourth rows of Table 1 in terms of both computational time and accuracy. Since the groundtruth values of the missing entries in data are not known, the relative reconstruction error is instead calculated as $\left\|\mathbf{W} \odot (\mathbf{X}_{ori} - \widetilde{\mathbf{U}}\widetilde{\mathbf{V}}^T)\right\|_F / \left\|\mathbf{W} \odot \mathbf{X}_{ori}\right\|_F$, where $\mathbf{W}$ is the indicator matrix of missing entries.

From Table 1 we can see that the proposed CWM method performs the best among all competing methods in terms of both computational speed and accuracy (except for the computational time on the dinosaur sequence).

## Face reconstruction experiments

We then test our algorithm in face reconstruction problems. We generated some relatively small datasets and some relatively large datasets in the experiments. The small face datasets were generated by extracting the first subset of Extended Yale B (Georghiades, Belhumeur, and Kriegman 2001; Basri and Jacobs 2003), containing 64 face images of size $192 \times 168$. We downsampled the images to $24 \times 21$ and contaminated the images with missing entries and salt-pepper noise. The images were contaminated with $(10\%, 10\%)$ and $(40\%, 10\%)$ of (missing entries, salt-pepper noise), respectively, and then two data matrices of dimension $504 \times 64$ are

[4]We used the code "l1decode_pd.m" (Candès and Romberg 2005) for solving the linear programming problem in the iteration.

[5]http://cs.adelaide.edu.au/~anders/code/cvpr2010.html

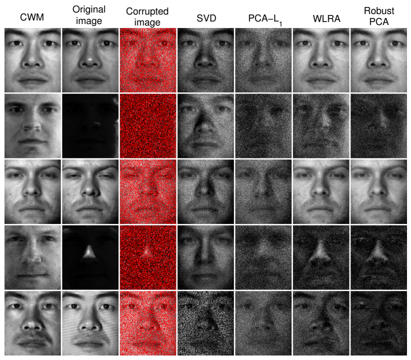

Figure 2: From left column to right column: faces reconstructed by CWM, original face images in Facedata8, the corrupted images (black and white pixels denote noises and red pixels denote missing entries), faces reconstructed by SVD, PCA-$L_1$, WLRA and robust PCA, respectively. The CWM results are put to the left for easy visualization of latent features discovered by the method from the images.

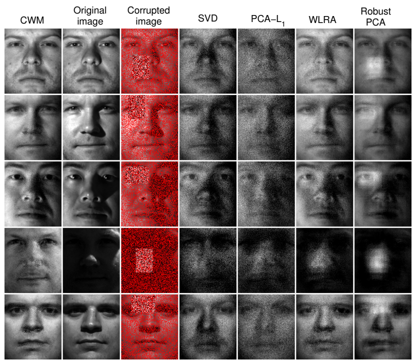

Figure 3: From left column to right column: faces reconstructed by CWM, original face images in Facedata14, the corrupted images, faces reconstructed by SVD, PCA-$L_1$, WLRA and robust PCA, respectively.

formed, called Facedata1 and Facedata2, respectively.

The larger data matrices were built by using the subsets 1-4 of the Extended Yale B database, containing 256 face images of size $192 \times 168$. 12 data matrices, denoted as Facedata3-Facedata14, were generated in the following way. Facedata3-Facedata8 are generated by setting $(0\%, 10\%)$, $(10\%, 10\%)$, $(20\%, 10\%)$, $(30\%, 10\%)$, $(40\%, 10\%)$, $(50\%, 10\%)$ of the randomly selected pixels of each image in the dataset as (missing entries, salt-pepper noise), respectively. Facedata9-Facedata14 are generated by first occluding each image with salt-pepper noise located in a rectangular of size $50 \times 70$ or $70 \times 50$, at a random position of the image, and then setting $0\%, 10\%, 20\%, 30\%, 40\%, 50\%$ of its pixels as missing entries. Each dataset corresponds to a matrix with size $32256 \times 256$. Typical images of Facedata8 and Facedata14 are depicted in Figures 2 and 3, respectively.

We first compared our CWM method against WLRA, Wilberg-$L_2$, ALP and Wilberg-$L_1$ on the small data matrices. The results are shown in the fifth and sixth rows of Table 1. For the large data matrices of Facedata3-Facedata14, Wilberg-$L_2$, ALP and Wilberg-$L_1$ could not execute in our computer. We thus employed another three methods for per-

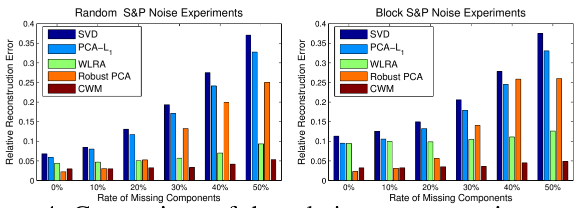

Figure 4: Comparison of the relative reconstruction errors obtained by the SVD, PCA-$L_1$, WLRA, robust PCA and CWM methods on Facedata3-Facedata14 (missing entry rate in data varies from $0\%$ to $50\%$).

formance comparison. They include SVD (Golub and Loan 1989), state-of-the-art for $L_2$-norm LRMF without missing entries, PCA-$L_1$ (Kwak 2008), a typical $L_1$-norm LRMF method without missing entries, and robust PCA (Wright et al. 2009), state-of-the-art for LRMF with sparse noises. Figures 2 and 3 show some original and reconstructed images by the used methods on Facedata8 and Facedata14, respectively. The RRE values obtained by these methods on Facedata3-Facedata14 are also depicted in Figure 4 for easy comparison.

From Table 1, it can be seen that the computational speed of the proposed CWM method is faster than other methods except for WLRA, which is designed for the smooth $L_2$-norm LRMF problem; and its computational accuracy is clearly higher than all the other competing methods in both Facedata1 and Facedata2. It can also be easily observed from Figure 4 that the CWM method has the highest accuracy among all competing methods (except that in the case of LRMF without missing entries, its accuracy is slightly lower than that of robust PCA), and its accuracy is only sightly reduced with the rate of missing entries increasing (from $0\%$ to $50\%$). It can be observed from Figures 2 and 3 that some latent features underlying the original faces can be recovered by the proposed method, e.g., the shadows and the stripe waves tend to be removed from the faces. What makes it interesting is that these reconstructions are obtained from the corrupted images, but not the clean faces. Such an advantage attributes to the cyclicly employed weighted median filters in our method, which is intrinsically insensitive to the impulsive noises and outliers and always helpful in avoiding overfitting on data corruptions (Brownrigg 1984). This shows that the CWM method is potentially useful for latent information retrieval from noisy measurements in practice.

## Conclusion

In this paper we developed a novel cyclic weighted median (CWM) method based on coordinate decent methodology to solve the $L_1$-norm low-rank matrix factorization (LRMF) problem in the presence of outliers and missing entries. Based on the extensive experimental results, it can be concluded that the proposed CWM method is very robust to outliers and missing entries, and it outperforms state-of-the-art methods in terms of both accuracy and efficiency. A prominent advantage of the proposed method is that it reduces the computational complexity of $L_1$-norm LRMF from the state-of-the-art speed $O(dn)$ to $O(d+n)$ in the case that the input data matrix is highly sparse, which makes it particularly useful in solving real problems with big and sparse data.

## References

Aguiar, P.; Stosic, M.; and Xavier, J. 2008. Spectrally optimal factorization of incomplete matrices. In *CVPR*.

Basri, R., and Jacobs, D. 2003. Lambertian reflection and linear subspaces. *IEEE Trans. PAMI* 25:218–233.

Breheny, P., and Huang, J. 2011. Coordinate desent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics* 5:232–253.

Brownrigg, D. R. K. 1984. The weighted median filter. *Commun. Assoc. Comput. Mach.* 27.

Buchanan, A. M., and Fitzgibbon, A. W. 2005. Damped newton algorithms for matrix factorization with missing data. In *CVPR*.

Candès, E. J., and Romberg, J. 2005. *l1-MAGIC: recovery of sparse signals via convex programming*. Technical Report, California Institute of Technology.

Candès, E.; Li, X.; Ma, Y.; and Wright, J. 2011. Robust principal component analysis? *Journal of the ACM* 58.

Cheng, C.; Yang, H.; King, I.; and Lyu, M. R. 2012a. Fused matrix factorization with geographical and social influence in location-based social networks. In *AAAI*.

Cheng, C.; Yang, H. Q.; King, I.; and Lyu, M. R. 2012b. Fused matrix factorization with geographical and social influence in location-based social networks. In *AAAI*.

De la Torre, F., and Black, M. J. 2003. A framework for robust subspace learning. *IJCV* 54:117–142.

Deerwester, S.; Dumais, S.; Furnas, G.; Landauer, T.; and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41.

Ding, C.; Zhou, D.; He, X.; and Zha, H. 2006. R1-PCA: Rotational invariant l1-norm principal component analysis for robust subspace factorization. In *ICML*.

Eriksson, A., and van den Hengel, A. 2010. Efficient computation of robust low-rank matrix approximations in the presence of missing data using the $l_1$ norm. In *CVPR*.

Eriksson, A., and van den Hengel, A. 2012. Efficient computation of robust weighted low-rank matrix approximations using the $l_1$ norm. *IEEE Trans. PAMI* 34:1681–1690.

Friedman, J.; Hastie, T.; and Höfling. 2007. Pathwise coordinate optimization. *The Annals of Applied Statistics* 1:302–332.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2010. Regularized paths for generalized linear models via coordinate descent. *Journal of Statistical Software* 33:1–22.

Gabriel, K. R., and Zamir, S. 1979. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics* 21:489–498.

Georghiades, A.; Belhumeur, P.; and Kriegman, D. 2001. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. PAMI* 23:643–660.

Gillis, N., and Glineur, F. 2011. Low-rank matrix approximation with weights or missing data is NP-hard. *SIAM Journal of Matrix Analysis and Applications* 32.

Golub, G., and Loan, C. V. 1989. *Matrix Computation*. Maryland: Johns Hopkins University Press.

Jacobs, D. 1997. Linear fitting with missing data: Applications to structure-from-motion and to characterizing intensity images. In *CVPR*.

Jeffreys, H., and Jeffreys, B. S. 1988. *Methods of Mathematical Physics*. Cambridge University Press.

Ji, H.; Liu, C.; Shen, Z.; and Xu, Y. 2010. Robust video denoising using low rank matrix completion. In *CVPR*.

Ke, Q., and Kanade, T. 2001. A subspace approach to layer extraction. In *CVPR*.

Ke, Q., and Kanade, T. 2005. Robust $l_1$ norm factorization in the presence of outliers and missing data by alternative convex programming. In *CVPR*.

Koren, Y. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *SIGKDD*.

Kwak, N. 2008. Principal component analysis based on l1-norm maximization. *IEEE Trans. PAMI* 30:1672–1680.

Li, W. J., and Yeung, D. Y. 2009. Relation regularized matrix factorization. In *IJCAI*.

Lin, Z.; Chen, M.; and Ma, Y. 2009. *The augmented lagrange multiplier method for exact recovery of corrupted low-Rank matrix*. Technical Report UILU-ENG-09-2215.

Luo, D.; Ding, C.; and Huang, H. 2011. Cluster indicator decomposition for efficient matrix factorization. In *IJCAI*.

Okatani, T., and Deguchi, K. 2007. On the Wiberg algorithm for matrix factorization in the presence of missing components. *IJCV* 72:329–337.

Rauh, A., and Arce, G. R. 2010. A fast weighted median algorithm based on quickselect. In *ICIP*.

Shum, H.; Ikeuchi, K.; and Reddy, R. 1995. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Trans. PAMI* 17:855–867.

Sinha, A. 2010. *Topics in the analysis of sparse and irregularly spaced time dependent gene expression data*. Columbia University, Doctoral Dissertation.

Srebro, N., and Jaakkola, T. 2003. Weighted low-rank approximations. In *ICML*.

Strelow, D. 2012. General and nested Wiberg minimization. In *CVPR*.

Sturm, P. 2000. Algorithms for plane-based pose estimation. In *CVPR*.

Tomasi, C., and Kanade, T. 1992. Shape and motion from image streams under orthography: A factorization method. *IJCV* 9:137–154.

Tseng, P. 2001. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications* 109:475–494.

Turk, M., and Pentland, A. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience* 3:71–86.

Wright, J.; Peng, Y.; Ma, Y.; Ganesh, A.; and Rao, S. 2009. Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *NIPS*.