# Robust Tucker Tensor Decomposition for Effective Image Representation

Miao Zhang
University of Texas at Arlington
701 S. Nedderman Drive, Arlington, TX 76019
miao.zhang@mavs.uta.edu

Chris Ding
University of Texas at Arlington
701 S. Nedderman Drive, Arlington, TX 76019
chqding@uta.edu

## Abstract

*Many tensor based algorithms have been proposed for the study of high dimensional data in a large variety of computer vision and machine learning applications. However, most of the existing tensor analysis approaches are based on Frobenius norm, which makes them sensitive to outliers, because they minimize the sum of squared errors and enlarge the influence of both outliers and large feature noises. In this paper, we propose a robust Tucker tensor decomposition model (RTD) to suppress the influence of outliers, which uses $L_1$-norm loss function. Yet, the optimization on $L_1$-norm based tensor analysis is much harder than standard tensor decomposition. In this paper, we propose a simple and efficient algorithm to solve our RTD model. Moreover, tensor factorization-based image storage needs much less space than PCA based methods. We carry out extensive experiments to evaluate the proposed algorithm, and verify the robustness against image occlusions. Both numerical and visual results show that our RTD model is consistently better against the existence of outliers than previous tensor and PCA methods.*

## 1. Introduction

Image or video storage and denoising problems are two important research topics in computer vision area, especially with the development of online social media, which provides tons of images and videos everyday. In a typical image storage problem, an image is represented as a 1-d long feature vector, and then this long vector denotes one data point in a high dimensional space. But as we all know, an image can be naturally represented as a 2-d matrix, with each element denoting the feature value on that specific spot. The 1-d vector denotation of an image makes it convenient for subspace learning, such as principal component analysis (PCA)[19] and linear discriminant analysis (LDA)[2] used in face recognition area.

Recently, some of other subspace learning algorithms applied on 1-d vector data are studied, such as locality preserving projection (LPP)[10] and localized linear models (LLM)[7], which are proven to be efficient. However, the 1-d vector denotation strategy as a whole ignores the neighborhood feature information within one image, while 2-d matrix denotation retains the important spatial relationship between features within one image.

Therefore, a lot of tensor decomposition techniques are studied in computer vision applications. For example, Shashua and Levine [16] adopted rank-one decomposition to represent images, which was described in detail in [18]. Yang et al. [22] introduced a two dimensional PCA (2DPCA), in which, one-side low-rank approximation was applied. Generalized Low Rank Approximation of Matrices (GLRAM) was proposed by Ye et al. [23], and the method projected the original images onto one two dimensional space. Ding and Ye proposed a two dimensional singular value decomposition (2DSVD) [6], which computes principal eigenvectors of row-row and column-column covariance matrices. Other tensor decomposition methods are also proposed and some of them are proven to be equivalent to 2DSVD and GLRAM in [11]. High order singular value decomposition (HOSVD) [14] were proposed for higher dimensional tensor by Vasilescu and Terzopoulos [20].

In above tensor analysis algorithms, an image is denoted by a 2-d matrix or second order tensor as itself, which retains the neighborhood information within the image itself, and then a set of images can be denoted by a third-order tensor. They minimize the sum of squared errors, which is known as frobenious norm, in which large errors due to outliers and feature noises such as occlusion, after being squared, dominate the error function and force the low rank approximation to concentrate on these few data points and features, while nearly ignoring most of other data points.

Over the years, there are many different approaches proposed to solve this problem both on 1-d vector data and 2-d matrix data. [24] [17] [8] [1] [4] [12] [13] [9] [5]. The approach using pure $L_1$-norm is used widely because it offers an simple and elegant formulation [1] [4] [12] [9] to suppress the impact coming from noisy data or features.

A difficulty of pure $L_1$-based methods is that the optimization tends to be hard. Several computational methods have been proposed [1] [4] [12] [9]. These methods are either complicated or difficult to scale to large problems.

In this paper, we propose a robust Tucker tensor decomposition (RTD) model to deal with images occluded by noisy information, and also propose a simple yet computationally efficient algorithm to solve the $L_1$-norm based Tucker tensor decomposition optimization. This method also provides some insights to the optimization problem such as the Lagrangian multiplier and KKT condition. We also carry out extensive experiments in face recognition, and verify the robustness of the proposed method to image occlusions. Both numerical and visual results demonstrate the effectiveness of our proposed method.

## 2. Robust Tucker Tensor Decomposition (RTD)

Standard Tucker tensor decomposition [14] uses reconstructed tensor $Y$ to approximate the original tensor $X$,

$$Y_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} U_{ip}V_{jq}W_{kr}S_{pqr} \tag{1}$$

where $Y$ is a third order tensor, $Y \in \Re^{n_i \times n_j \times n_k}$, $U \in \Re^{n_i \times P}$, $V \in \Re^{n_j \times Q}$, $W \in \Re^{n_k \times R}$, $S \in \Re^{P \times Q \times R}$ is a core tensor, which couples different 3rd order multi-linear polynomials. Therefore, mathematically, $Y$ can be expressed as the following (Eq.(2)), which simplifies the tensor constructing expressions in next sections.

$$Y = U \otimes_1 V \otimes_2 W \otimes_3 S \tag{2}$$

Tucker tensor decomposition has the following cost function [18],

$$\begin{aligned} &\min_{U,V,W,S} \|X - Y\|_F^2 = \sum_{i=1}^{n_i}\sum_{j=1}^{n_j}\sum_{k=1}^{n_k}\left(X_{ijk} - Y_{ijk}\right)^2 \\ &s.t. \quad U^TU = I, V^TV = I, W^TW = I \end{aligned} \tag{3}$$

It is well-known that the solution to the above optimization is given by high order singular value decomposition (HOSVD) [14], which will be introduced in the algorithm part. As we can see, the standard Tucker tensor decomposition uses Frobenius norm to decompose the original tensor. Frobenius norm is known for being sensitive to outliers and feature noises, because it sums the squared errors. While, $L_1$-norm just sums the absolute value of error, which reduces the influence of the outliers comparing to the Frobenius norm. So the more robust against outlier version of Tucker tensor decomposition is formulated using $L_1$-norm. $L_1$-norm of a third order tensor $A$ with size $n_i \times n_j \times n_k$ is defined as $\|A\|_1 = \sum_{i=1}^{n_i}\sum_{j=1}^{n_j}\sum_{k=1}^{n_k}|a_{ijk}|$. Therefore, the robust Tucker tensor decomposition (RTD) is formulated as,

$$\begin{aligned} &\min_{U,V,W,S} \|X - Y\|_1 = \sum_{i=1}^{n_i}\sum_{j=1}^{n_j}\sum_{k=1}^{n_k}\left|X_{ijk} - Y_{ijk}\right| \\ &s.t. \quad U^TU = I, V^TV = I, W^TW = I \end{aligned} \tag{4}$$

**Illustration**. Before going any further, we want to give a glance at the denoising effect by RTD first. Figure 1 and Figure 2 illustrate the reconstructed effect on AT&T data set, with existence of two different occlusion strategies, which will be explained in details in the experiment part. In both figures, images of the second row represent the reconstructed images by RTD and those of the fourth row represent images reconstructed by Tucker tensor decomposition. In both noise and corruption cases, Robust Tucker decomposition gives clearly better reconstruction.

## 3. Efficient Algorithm for Robust Tucker Tensor Decomposition

The standard Tucker decomposition can be efficiently solved using the HOSVD algorithm [14]. In this paper, we propose an efficient algorithm to solve robust Tucker tensor decomposition. We employ the Augmented Lagrange Multiplier (ALM) method [3] to solve this problem. ALM has been successfully used in other $L_1$ related problems [21].

One important finding of this paper is that ALM is extremely well suited to this RTD model. The algorithm iteratively solves two sub-problems: One is a simplified LASSO (see Eq.(7)) with simple exact solution; Another is a standard Tucker tensor decomposition of Eq.(3). This enables us to utilize existing software to efficiently solve the RTD.

**Outline of the algorithm**. We first rewrite the objective function of robust Tucker tensor decomposition equivalently as

$$\begin{aligned} &\min_{U,V,W,S,E} \|E\|_1 \\ &\quad s.t. \quad E = X - U \otimes_1 V \otimes_2 W \otimes_3 S \\ &\qquad\qquad U^TU = I, V^TV = I, W^TW = I \end{aligned} \tag{5}$$

Now we use ALM approach by enforcing the equality constraint $E = X - U \otimes_1 V \otimes_2 W \otimes_3 S$ using Lagrange multipliers (matrix $\Lambda$) and quadratic penalty. Then ALM becomes to solve the following problem,

$$\begin{aligned} &\min_{E,U,V,W,S} \|E\|_1 + \langle \Lambda, X - U \otimes_1 V \otimes_2 W \otimes_3 S - E \rangle \\ &\qquad + \frac{\mu}{2}\|X - U \otimes_1 V \otimes_2 W \otimes_3 S - E\|_F^2 \\ &s.t. \quad U^TU = I, V^TV = I, W^TW = I \end{aligned} \tag{6}$$

where scalar $\mu$ is the penalty parameter, $\langle P, Q \rangle$ is defined as $\sum_{ijk} P_{ijk}Q_{ijk}$.

Figure 1. Samples of occluded images and reconstructed images on AT&T face data. First row is the input occluded images; Second row is from RTD; Third row is from $L_1$PCA; Fourth row is from Tucker decomposition; Fifth row is from PCA.

The ALM is an iteratively updating algorithm. There are two major parts, solving the sub-problems and updating parameters, which will be presented in the following sections.

### 3.1. Solving the Sub-optimization Problems

The key step of the algorithm is solving the two subprograms of Eq.(6) for each set of parameter values of $\Lambda, \mu$. Fortunately, this can be solved in closed form solutions for $E$ and group of $(U, V, W, S)$.

**A**. Solve for $E$. First, we solve $E$ while fixing $U$, $V$, $W$ and $S$. From Eq.(6), the objective function becomes

$$\min_{E} \; \|E\|_1 + \frac{\mu}{2}\|E - P\|_F^2 \tag{7}$$

where $P$ is a constant matrix independent of $E$:

$$P = X - U \otimes_1 V \otimes_2 W \otimes_3 S + \frac{\Lambda}{\mu}. \tag{8}$$

This problem has closed form solution

$$E^*_{ijk} = sign(P_{ijk}) \max(|P_{ijk}| - 1/\mu, 0). \tag{9}$$

**B**. Solve for $(U, V, W, S)$. In this step, we solve $U$, $V$, $W$ and $S$ together while fixing $E$. From Eq.(6), the objective function becomes

$$\begin{aligned} \min_{U,V,W,S} \; & \frac{\mu}{2}||Q - U \otimes_1 V \otimes_2 W \otimes_3 S||_F^2, \\ s.t. \quad & U^T U = I, V^T V = I, W^T W = I \end{aligned} \tag{10}$$

where

$$Q = X - E + \frac{A}{\mu}; \tag{11}$$

This is exactly the usual Tucker tensor decomposition. This is solved by the known HOSVD algorithm [14]. HOSVD is an iterative algorithm. Given initial guess of

Figure 2. Samples of type 2 (mixed) occluded images and reconstructed images using different methods of AT&T data set. The first row is from input occluded images; the second row is from RTDreconstructed images; the third row is from $L_1$ PCA; the fourth row is from Tucker tensor; and the fifth row is from PCA. The cross corruptions can only be removed by RTD.

$U, V, W$ we update $U, V, W$ until convergence.
$U$ is given by the $P$ eigenvectors with largest eigenvalues of $F$, where

$$F_{ii'} = \sum_{jj'kk'} Q_{ijk} Q_{i'j'k'} (VV^T)_{jj'} (WW^T)_{kk'} \quad (12)$$

$V$ is given by the $Q$ eigenvectors with largest eigenvalues of $G$, where

$$G_{jj'} = \sum_{ii'kk'} Q_{ijk} Q_{i'j'k'} (UU^T)_{ii'} (WW^T)_{kk'} \quad (13)$$

$W$ is given by the $R$ eigenvectors with largest eigenvalues of $H$, where

$$H_{kk'} = \sum_{jj'ii'} Q_{ijk} Q_{i'j'k'} (VV^T)_{jj'} (UU^T)_{ii'}. \quad (14)$$

These steps are repeated until convergence. After $(U^*, V^*, W^*)$ are obtained, $S$ is given by

$$S_{pqr} = \sum_{ijk} Q_{ijk} U_{ip} V_{jq} W_{kr}. \quad (15)$$

### 3.2. Updating Parameters

In each iteration of ALM, after obtaining consistent $E$ and $(U, V, W, S)$, the parameters $\Lambda$ and $\mu$ are updated as the following

$$\Lambda \Leftarrow \Lambda + \mu(X - U \otimes_1 V \otimes_2 W \otimes_3 S - E) \quad (16)$$
$$\mu \Leftarrow \mu\rho \quad (17)$$

where $\rho > 1$ is a constant.
The complete algorithm is described in Algorithm 1.

> **Input**: $X, P, Q, R$
> **Output**: $U, V, W, S$
> Initialize $\mu = 1/||X||_F, \rho = 1.01, U_0, V_0, W_0$
> **repeat**
> Compute $E$ using Eq.(9)
> Compute $U, V, W, S$ using Eq.(12 - 15)
> $\Lambda = \Lambda + \mu(X - U \otimes_1 V \otimes_2 W \otimes_3 S - E)$
> $\mu = \min(\mu\rho, 10^{10})$
> **until** *Converge*

**Algorithm 1:** RTD Algorithm

We initialize $(U, V, W)$ either by random or by the solution to the standard Tucker decomposition. In all these cases the ALM algorithm did converge. The converged solutions from different initialization are very close to each other[15], and there are no visible differences in the reconstructed images.

**Convergence Analysis**. By taking derivative of the Lagrangian function w.r.t. $E$, we obtain the Karush-Kuhn-Tucker (KKT) condition,

$$\Lambda_{ijk} = \begin{cases} sign(E_{ijk}) & \text{if } E_{ijk} \neq 0 \\ \partial|E_{ijk}| & \text{if } E_{ijk} = 0 \end{cases} \tag{18}$$

where $\partial|E_{ijk}| \in [-1, 1]$ is the subgradient of function $f(x) = |x|$.

here we view $\Lambda_{ijk}$ as Lagrangian multipliers. We now verify the KKT condition of our algorithm. The following are examples from AT&T dataset, whose tensor size is 56x46x400. More detailed dataset information will be introduced in the experiment part.

At convergence, the first 25 elements of computed $E_{ijk}$ are,

$$E = \begin{pmatrix} 0.0012 & -0.0003 & 0.0000 & -0.0005 & 0 \\ 0.0005 & -0.0005 & 0 & -0.0007 & -0.0011 \\ -0.0001 & 0 & -0.0005 & -0.0008 & 0 \\ -0.0002 & 0 & -0.0015 & -0.0001 & 0 \\ 0 & 0 & -0.0012 & 0 & 0.0001 \end{pmatrix}$$

The corresponding 25 elements $\Lambda_{ijk}$ are

$$\Lambda = \begin{pmatrix} 1.0000 & -1.0000 & 1.0000 & -1.0000 & 0.2806 \\ 1.0000 & -1.0000 & -0.8213 & -1.0000 & -1.0000 \\ -1.0000 & -0.5164 & -1.0000 & -1.0000 & 0.3976 \\ -1.0000 & -0.2643 & -1.0000 & -1.0000 & -0.4540 \\ 0.0630 & 0.1762 & -1.0000 & 0.3274 & 1.0000 \end{pmatrix}$$

We see that the above KKT condition are satisfied for every elements. When $E_{ijk}$ is nonzero, $\Lambda_{ijk}$ is its sign. When $E_{ijk}$ is zero, $\Lambda_{ijk}$ is its subgradient (a value in $[-1, 1]$).

## 4. Efficient Algorithm for $L_1$-PCA

In standard computer vision problems, each image is converted to a vector and a set of images is represented by a matrix. Here PCA is mostly wide used. The advantage of tensor approach is that each image retains its 2D form in tensor representation and thus tensor analysis retains more information on image collections.

We need to compare the tensor approaches with matrix approaches. Thus we implement the algorithm for computing $L_1$PCA. $L_1$PCA is formulated as the following

$$\min_{U,V} \|X - UV\|_1 = \sum_{j=1}^{n} \sum_{i=1}^{p} |(X - UV)_{ij}|, \tag{19}$$

where $X = (x_1, \cdots, x_n)$ contain $n$ images. $X \in \Re^{p \times n}$ where $p = rc$ for $r$-by-$c$ images. The factor matrices $U, V$ have sizes of $U \in \Re^{p \times k}$, $V \in \Re^{k \times n}$.
Similarly with solving RTD, Eq.(19) can be rewritten equivalently as

$$\min_{E,U,V} \|E\|_1, \; s.t. \; E = X - UV, \tag{20}$$

ALM solves a sequence of sub-problems

$$\min_{E,U,V} \|E\|_1 + \langle A, X - UV - E\rangle + \frac{\mu}{2}||X - UV - E||_F^2 \tag{21}$$

where matrix $A$ is the Lagrange multipliers.

**A. Solve for** $E$. First, we solve $E$ while fixing $U$ and $V$. From Eq.(21), the objective function becomes

$$\min_{E} \|E\|_1 + \frac{\mu}{2}||E - (X - UV + \frac{A}{\mu})||_F^2 \tag{22}$$

This problem has closed form solution:

$$E^*_{ij} = sign(P_{ij})(|P_{ij}| - 1/\mu)_+, \; P = X - UV + \frac{A}{\mu}. \tag{23}$$

**B. Solve for** $U, V$. Next we solve $U$ and $V$ together while fixing $E$. From Eq.(21), the objective function becomes

$$\min_{U,V} \langle A, X - UV - E\rangle + \frac{\mu}{2}||X - UV - E||_F^2. \tag{24}$$

Which is is equivalent to

$$\min_{U,V} \frac{\mu}{2}||Q - UV||_F^2, \;\; Q = X - E + \frac{A}{\mu}; \tag{25}$$

The solution is given by standard PCA. Denote the singular value decomposition (SVD) of $Q$ as

$$Q = F\Sigma G^T \tag{26}$$

Only first $k$ largest singular values and associated singular vectors are needed. Then the solution of $U, V$ are given by

$$\begin{aligned} U &= F_k, \\ V &= \Sigma_k G_k^T \end{aligned} \tag{27}$$

In each iteration of ALM, after obtaining consistent $E$ and $(U, V)$, the parameters $A$ and $\mu$ are updated as the following

$$A \;\; \Leftarrow \;\; A + \mu(X - UV - E) \tag{28}$$

$$\mu \;\; \Leftarrow \;\; \mu\rho \tag{29}$$

where $\rho > 1$ is a constant.

# 5. Experiments

In this section, three benchmark face databases AT&T, YALE and CMU PIE are used to evaluate the effectiveness of our proposed RTD tensor factorization approach.

## 5.1. Data Description

The properties of the three data sets we used are summarized in Table 1, and the detailed information of each data set is given as the following.

Table 1. Description of Data sets

| Data set | #images $n_k$ | #Dimensions $n_i \times n_j$ | #Class $K$ |
|---|---|---|---|
| AT&T | 400 | $56 \times 46$ | 40 |
| YALE | 1984 | $48 \times 42$ | 31 |
| CMU PIE | 680 | $32 \times 32$ | 68 |

Table 2. Performance Comparison (Storage, Noise-free Error and Classification Accuracy) on AT&T data with Block Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 1,030,400 | $4.7269 \times 10^4$ | 0.6050 |
| RTD | 19,672 | $3.0457 \times 10^4$ | 0.7125 |
| $L_1$PCA | 119,040 | $3.1435 \times 10^4$ | 0.7025 |
| Standard Tensor | 19,672 | $3.3834 \times 10^4$ | 0.6775 |
| Standard PCA | 119,040 | $3.4959 \times 10^4$ | 0.6675 |

Table 3. Performance Comparison(Storage, Noise-free Error and Classification Accuracy) on Yale data with Block Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 3,999,744 | $6.9070 \times 10^4$ | 0.3766 |
| RTD | 64,204 | $4.3685 \times 10^4$ | 0.3896 |
| $L_1$PCA | 124,000 | $4.6886 \times 10^4$ | 0.3311 |
| Standard Tensor | 64,204 | $4.8164 \times 10^4$ | 0.3831 |
| Standard PCA | 124,000 | $5.0806 \times 10^4$ | 0.2989 |

**AT&T**: The AT&T face data contains 400 upright face images of 40 individuals, collected by AT&T Laboratories Cambridge. Each image is resized to 56x46 pixels in this experiment.

**YALE**: There are totally 38 classes (10 subjects in original database with 28 subjects in the extended database) under 576 viewing conditions (9 poses with 64 different illumination conditions). 64 images in different illumination conditions from 31 classes are selected for our experiment, so there are totally 1984 images.

Table 4. Performance Comparison(Storage, Noise-free Error and Classification Accuracy) on CMU PIE data with Block Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 696,320 | $2.4501 \times 10^4$ | 0.4735 |
| RTD | 47,840 | $0.8578 \times 10^4$ | 0.5294 |
| $L_1$PCA | 115,872 | $1.0388 \times 10^4$ | 0.5279 |
| Standard Tensor | 47,840 | $1.7610 \times 10^4$ | 0.4926 |
| Standard PCA | 115,872 | $1.8419 \times 10^4$ | 0.4882 |

Table 5. Performance Comparison(Storage, Noise-free Error and Classification Accuracy) on AT&T data with Mixed Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 1,030,400 | $2.9635 \times 10^4$ | 0.8725 |
| RTD | 19,672 | $1.8536 \times 10^4$ | 0.9450 |
| $L_1$PCA | 119,040 | $1.9924 \times 10^4$ | 0.9325 |
| Standard Tensor | 19,672 | $2.4942 \times 10^4$ | 0.8875 |
| Standard PCA | 119,040 | $2.5723 \times 10^4$ | 0.8800 |

Table 6. Performance Comparison(Storage, Noise-free Error and Classification Accuracy) on Yale data with Mixed Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 3,999,744 | $4.5618 \times 10^4$ | 0.3725 |
| RTD | 64,204 | $3.3482 \times 10^4$ | 0.4134 |
| $L_1$PCA | 124,000 | $3.6471 \times 10^4$ | 0.3916 |
| Standard Tensor | 64,204 | $4.1843 \times 10^4$ | 0.3678 |
| Standard PCA | 124,000 | $4.0981 \times 10^4$ | 0.3714 |

Table 7. Performance Comparison(Storage, Noise-free Error and Classification Accuracy) on CMU PIE data with Mixed Occlusion

| Methods | Storage | Noise-free Error | Class ACC |
|---|---|---|---|
| Corrupted $X$ | 696,320 | $2.4532 \times 10^4$ | 0.4562 |
| RTD | 47,840 | $1.7856 \times 10^4$ | 0.5332 |
| $L_1$PCA | 115,872 | $1.8442 \times 10^4$ | 0.5106 |
| Standard Tensor | 47,840 | $2.1427 \times 10^4$ | 0.4762 |
| Standard PCA | 115,872 | $2.1019 \times 10^4$ | 0.4632 |

**CMU PIE**: CMU PIE is a face database of 41,368 images of 68 people, collected by Carnegie Mellon Robotics Institute between October and December 2000. Each image is resized into 32x32 pixels in our experiment. We randomly select 10 images from each class with different combinations of pose, face expression and illumination condition.

### 5.2. Corrupted Images

For evaluation purpose, we generate occluded images from the above three image data sets. One added advantage of this approach is that we can compare the reconstructed images with the original uncorrupt images to assess the effectiveness of removing the corruption (occlusion).

We use two type of occlusions added to the original input images to evaluate the effectiveness of proposed RTD tensor method against outliers. First, square block occlusions with different size are added. The occlusion is generated as the following, given the size of occlusion $d$, we randomly pick up the $d \times d$ block position for each image, and we set pixels in this $d \times d$ area to zero. There are some examples of occluded images using this method in Figure 1.

Second, mixed occlusions with 3 different corrupting methods are added to the original images. First corruption methods are called cross occlusions, and the cross has specified length $l$ and width $w$. For each class, we randomly select $m$ images to add cross occlusions. We also randomly select the position of the cross, and set the pixels in the cross to the average pixel value of the whole data set. To make the occlusions realistic and diversified, for each class, on the basis of cross occlusions, we randomly select $m$ images to add square block occlusions introduced above. In the end, rectangular occlusions are added. Similarly, for each class, we randomly select $m$ images to add rectangular occlusions. We randomly set the sizes of each rectangle within a permitted range $[a, b]$, and within each rectangle, some of the pixels are set to 0, and the rest are set to 1. The first row in Figure 2 demonstrates this mixed occlusion method.

Figure 1 and Figure 2 only show 1 person of 400 people in AT&T data set due to space limitation. For AT&T data set, an $8 \times 8$ occlusion is added to every image of each class in the first type of occlusion. For the second type of occlusion, within each class of images, we first randomly select $m = 2$ images to add the cross, and for each selected image the length of cross is $l = 22$ and width is $w = 3$. Second, we randomly select $m = 2$ images to add the square block. Third, we randomly select $m = 2$ images to add the rectangle, and for each added rectangle, the sizes are random within a ranger of $[a, b] = [4, 10]$. Similarly, for Yaleb data set, $d = 8$ and $l = 20$, $w = 3$, $m = 12$, $[a, b] = [4, 10]$. For CMU PIE data set, we set $d = 6$ and $l = 15$, $w = 3$, $m = 3$, $[a, b] = [3, 10]$.

### 5.3. Experiment Results

In this section, we compare the performance of our RTD method with standard Tucker tensor method, $L_1$-norm PCA method ($L_1$PCA) and standard PCA method at storage space, the noise reduction effect and classification accuracy.

One of the biggest advantage of our proposed RTD method is to save image storage space, because for Tucker tensor decomposition methods, to reconstruct the images, we only need to store $U$, $V$ and $W$, the core tensor $S$ can be calculated using $U$, $V$, $W$. The sizes of $U$, $V$, $W$ are $n_i \times P$, $n_j \times Q$, $n_k \times R$, respectively. So the storage space for our $L_1$-norm tensor are

$$n_i \times P + n_j \times Q + n_k \times R$$

While for PCA based methods, $U$ and $V$ need to be stored, and the sizes of $U$ and $V$ are $p \times k$ and $k \times n$ respectively, and here $p = n_i \times n_j$ and $n = n_k$. So the storage space for PCA based methods would be

$$n_i \times n_j \times k + k \times n_k$$

The parameters we used in our experiment for each data set is given in Table 8. Accordingly, the needed storage space for each method on every data sets can be calculated, which are given in Table 2, 3, 4. **Noise-free Reconstruction Error**. Let $X$ be the original images and $O$ be the occlusion. Then $X + O$ are the input data to tensor decompositions and PCA. Let $Y$ be the reconstructed images from Eq.(1). All tensor analysis and PCA minimize $\|(X + O) - Y\|_F$. However, our goal is to recover the *true, noise-free* images. For occluded data, we take the original images as the approximation of the true noise-free images, and consider $\|X - Y\|_F$ as a measure of the ability to recover the noise-free images. We thus call $\|X - Y\|_F$ as the noise-free reconstruction error. It can be computed for PCA and tensor decompositions.

Table 8. Parameters of different Data sets

| Data set | $P \times Q \times R$ | $k$ |
|---|---|---|
| AT&T | $36 \times 36 \times 40$ | 40 |
| YALE | $30 \times 30 \times 31$ | 31 |
| CMU PIE | $25 \times 25 \times 68$ | 68 |

The noise-free error for each method is listed in Table 2, 3, 4 for the first type of occlusion and Table 5, 6, 7 for the second type of occlusion. We can see (1) the noise-free errors for RTD and $L_1$PCA are *always* smaller than those for Tucker decomposition and PCA; This shows the effectiveness of $L_1$ norm for removing corruptions. (2) Noise-free errors for RTD are always smaller than those for $L_1$PCA; This demonstrates the advantage of Tensor decomposition approach.

A byproduct of image denoising is improved classification accuracy. Here we perform classification as the demonstration and evaluation of denoising effectiveness of the proposed RTD. We use k nearest neighbor (kNN) (we use 1NN here) as the multi-class classifier. Classification accuracy on occluded image data are listed in Table 2, 3, 4 for the first type of occlusion and Table 5, 6, 7 for the second type of occlusion. All classification results are based on 2-fold cross-validation. For each class, we randomly split the images into 2 parts, and then we set each of the two parts as training set and the rest part as testing set. The reported accuracy is the average of 100 times of cross validations.

### 5.4. Reconstruction Images and Discussion

Figure 1 and Figure 2 demonstrate the sample occluded images and the corresponding reconstructed images from different methods. As we can see, the reconstructed images from our RTD method reduce the occlusion more successfully than other methods, which is also shown by the noise-free error in Table 2, 3, 4, 5, 6, 7, the noise-free error of our methods are smaller than other methods. Our method needs far less storage space than PCA based methods, for example, the storage for PCA based method is 119,040 for AT&T data set, while for our RTD method, the storage is only 19,672, that is to say, PCA based methods need 6 times bigger storage than tensor methods do on AT&T data set. Classification accuracies on the reconstructed images from RTD method are higher in most cases, which demonstrated the effectiveness our method.

## 6. Conclusion

In this paper, we propose an $L_1$-norm based robust Tucker tensor decomposition (RTD) method, which is effective for correcting corrupted images. Our method requires far less storage space than PCA based methods. We also propose a computationally efficient algorithm to solve the proposed RTD model. Extensive experiments are carried out to evaluate the proposed RTD. Both numerical and visual results are consistently better for images with outliers or noisy features than standard PCA, $L_1$PCA and standard Tucker tensor decomposition methods. This validates the effectiveness of the proposed RTD.

**Acknowledgement**. This research is partially supported by NSF-CCF-0917274 and NSF-DMS-0915228 grants.

## References

[1] A. Baccini, P. Besse, and A. D. Falguerolles. An L1-norm PCA and a heuristic approach. *Ordinal and Symbolic Data Analysis, edited by E. Diday, Y. Lechevalier and O. Opitz, Springer*, 1996.

[2] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):711–720, 1997.

[3] D. P. Bertsekas. *Nonlinear Programming, 2nd Ed.* MIT Press, 1998.

[4] J. Bolton and W. J. Krzanowski. A characterization of principal components for projection pursuit, Mar. 26 2001.

[5] E. J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis, Dec. 18 2009.

[6] C. H. Q. Ding and J. Ye. 2-dimensional singular value decomposition for 2d maps and images. In *SDM*, 2005.

[7] Y. Fu, Z. Li, J. Yuan, Y. Wu, and T. S. Huang. Locality versus globality: Query-driven localized linear models for facial image computing. *IEEE Trans. Circuits Syst. Video Techn.*, 18(12):1741–1752, 2008.

[8] J. Galpin and D. Hawkins. Methods of l1 estimation of a covariance matrix. *Computational Statistics and Data Analysis*, 5:305–319, 1987.

[9] J. Gao. Robust L1 principal component analysis and its bayesian variational inference. *Neural Computation*, 20(2), 2008.

[10] X. He, S. Yan, Y. Hu, and H. jiang Zhang. Learning a locality preserving subspace for visual recognition. In *in Proc. IEEE International Conference on Computer Vision*, pages 385–393, 2003.

[11] K. Inoue and K. Urahama. Equivalence of non-iterative algorithms for simultaneous low rank approximations of matrices. In *CVPR (1)*, pages 154–159, 2006.

[12] Q. Ke and T. Kanade. Robust l1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 592–599, 2004.

[13] N. Kwak. Principal component analysis based on L1-norm maximization. *IEEE Trans. Pattern Anal. Mach. Intell*, 30(9):1672–1680, 2008.

[14] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl*, 21:1253–1278, 2000.

[15] D. Luo, C. Ding, and H. Huang. Are tensor decomposition solutions unique? on the global convergence of hosvd and parafac algorithms. 2008.

[16] A. Shashua and A. Levin. Linear image coding for regression and classification using the tensor-rank principle. In *CVPR*, pages 42–49, 2001.

[17] F. D. Torre and M. J. Black. A framework for robust subspace learning. *Int'l J. Computer Vision*, pages 117–142, 2003.

[18] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966c.

[19] M. Turk and A. Pentland. Eigen faces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[20] M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *ECCV (1)*, pages 447–460, 2002.

[21] A. Y. Yang, A. Ganesh, Z. Zhou, S. Sastry, and Y. Ma. A review of fast l1-minimization algorithms for robust face recognition. *CoRR*, abs/1007.3753, 2010.

[22] J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang. Two-dimensional pca: A new approach to appearance-based face representation and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(1):131–137, 2004.

[23] J. Ye. Generalized low rank approximations of matrices. In *ICML*, 2004.

[24] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *J. Computational and Graphical Statistics*, 15:265–286, 2006.