# Face Image Modeling by Multilinear Subspace Analysis with Missing Values


Xin Geng[a,b], Kate Smith-Miles[a], Zhi-Hua Zhou[c], Liang Wang[d]

[a]School of Mathematical Sciences
Monash University, VIC 3800, Australia
{xin.geng, kate.smith-miles}
@sci.monash.edu.au

[b]School of Computer Science and Engineering
Southeast University, Nanjing 210096, China
xgeng@seu.edu.cn

[c]National Key Lab for Novel Software Technology
Nanjing University, Nanjing 210093, China
zhouzh@nju.edu.cn

[d]Dept. of Computer Science and Software Eng.
The University of Melbourne, VIC 3010, Australia
lwwang@csse.unimelb.edu.au



## ABSTRACT

The main difficulty in face image modeling is to decompose those semantic factors contributing to the formation of the face images, such as identity, illumination and pose. One promising way is to organize the face images in a higher-order tensor with each mode corresponding to one contributory factor. Then, a technique called Multilinear Subspace Analysis (MSA) is applied to decompose the tensor into the mode-$n$ product of several mode matrices, each of which represents one semantic factor. In practice, however, it is usually difficult to obtain such a complete training tensor since it requires a large amount of face images with all possible combinations of the states of the contributory factors. To solve the problem, this paper proposes a method named $M^2$SA, which can work on the training tensor with massive missing values. Thus $M^2$SA can be used to model face images even when there are only a small number of face images with limited variations (which will cause missing values in the training tensor). Experiments on face recognition show that $M^2$SA can work reasonably well with up to 70% missing values in the training tensor.



## Categories and Subject Descriptors

I.2.10 [**Computing Methodologies**]: Artificial Intelligence—*Vision and Scene Understanding*

## General Terms

Algorithms


## 1. INTRODUCTION

Face images are highly variable source of multimedia data. Each face image results from the interaction of multiple contributory factors. For instance, one particular face image might be obtained by imaging a certain person (factor 1: identity), under certain lighting conditions (factor 2: illumination), from a certain view angle (factor 3: pose). The difficulties of face image modeling exist in the complexity of the interaction of the contributory factors. Among these factors, usually only one is of interest in a particular problem, and all the others are regarded as interferences. For example, in face recognition, the only goal is the recognition of identity, regardless of other possible variations. The key issue is, how to distinguish the target factor from the interferential factors while they are complicatedly interlaced in the face image?

Many face modeling methods [2] [3] [4] [8] aim to find certain statistical properties of the face images that correspond to the target factor. Once such properties are found, modeling them is equivalent to modeling the target factor. As a typical example, the PCA-based Eigenface method [8] models the major variation in the training face images, which is assumed to be caused by the difference of identities. Unfortunately, the variation caused by the changes in illumination, pose, or other interferential factors could be larger than that caused by identity changes. The problem is, PCA models the input data according to only one factor, i.e., the variance of the data. This creates the gap between the statistical factor (variance) and the semantic factors (identity, illumination, pose, etc.). Similar problems happen in the LDA-based Fisherface [2], the ISS [3] [4] based on PCA+MCA, and many other face modeling methods. Generally speaking, a 'good' model for multi-factor problems should satisfy the following requirements:

- The model consists of several components, each explicitly representing one semantic factor contributing to the problem;
- In each component, there is an unique feature vector for each state of the corresponding semantic factor. Except for the related factor, this feature vector will not change with any other contributory factors;
- There is a way for the model to decompose a single data source into its inner components with semantic meanings.

Toward this end, the training face images can be organized in a higher-order tensor rather than matrices. Each *mode* (also referred to as *dimension* or *way* in the literatures of multiway data analysis [1]) of the tensor explicitly corresponds to one of the semantic factors that generate the images. *Multilinear Subspace Analysis* (MSA) [9] [10] was recently proposed to decompose such a tensor of image ensemble. Through the application of *N-mode SVD* [9] to the tensor, MSA separates and parsimoniously represents each of the semantic factors. Then each image can be represented by a set of coefficient vectors, one for each semantic factor. Given a state of one semantic factor, there is an unique coefficient vector associated to it. For a particular problem, only the coefficient vector accounting for the target factor is used (to find out the state of the target factor for the input image). Thus the influence of interferential factors can be filtered out. Clearly, MSA fulfills the checklist of a 'good' face image model mentioned above.

Despite the beauty in theory, there are practical problems in MSA. The most prominent one might be the possible missing values in the training tensor. Back to the history of linear PCA, the missing value problem has long been recognized as an important practical issue and intensively investigated [5] [7] [11]. Although MSA is a multilinear extension of PCA, to the best of our knowledge, not much work has been done so far to deal with missing values in the training tensor. In fact, the missing value problem in MSA is *much more common* than that in PCA. In addition to the same situation PCA might encounter when some of the values in the training samples are missing due to data acquisition, transmission or storage problems, the following reason makes the missing values more likely to appear in MSA. Instead of a set of samples, the training data of MSA is a single well organized tensor. To fill all the positions in the tensor, a large amount of samples with all combinations of the states of the contributory factors are needed. Unfortunately, in many real applications, it is very hard (or impossible) to obtain such a large 'complete' image ensemble. In some other applications, even when the collection of samples with all kinds of variations is possible, clients wish to reduce cost by using as few as possible training samples without noticeable performance deterioration. In such cases, the available training samples might only account for a small portion of the quantity required to compose a 'complete' training tensor. Thus the algorithm must be able to work on the tensor with massive missing values even when no missing value appears in individual training samples. The missing value problem is therefore crucial for the practicability of MSA.

To solve the problem, this paper proposes a new method called $M^2$SA (Multilinear Subspace Analysis with Missing values) to model face images. Instead of minimizing the reconstruction error of the *whole* training tensor, $M^2$SA finds the approximation that can best reconstruct the *available* values in the tensor. The missing values may appear anywhere in the tensor, and they may even account for the majority of the tensor.

The rest of the paper is organized as follows. Section 2 introduces tensor fundamentals. Section 3 proposes the $M^2$SA algorithm to decompose the tensors with missing values. Experiments on face recognition using $M^2$SA are reported in Section 4. Finally, conclusions are drawn in Section 5

## 2. TENSOR FUNDAMENTALS

Tensors are higher-order generalization of scalar (zero-order tensor), vector (first-order tensor), and matrix (second-order tensor). In this paper, lowercase italic letters ($a, b, \ldots$) denote scalars, bold lowercase letters ($\boldsymbol{a}, \boldsymbol{b}, \ldots$) denote vectors, bold uppercase letters ($\mathbf{A}, \mathbf{B}, \ldots$) denote matrices, and calligraphic uppercase letters ($\mathcal{A}, \mathcal{B}, \ldots$) denote tensors. The *order* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is $N$. An element of $\mathcal{A}$ is denoted by $\mathcal{A}_{i_1 i_2 \ldots i_N}$ or $a_{i_1 i_2 \ldots i_N}$, where $1 \leq i_n \leq I_n, n = 1, 2, \ldots, N$. The *mode-n vectors* of $\mathcal{A}$ are the $I_n$-dimensional vectors obtained from $\mathcal{A}$ by varying index $i_n$ while keeping other indices fixed to certain values. A tensor $\mathcal{A}$ can be *flattened* into matrices in different ways. The *mode-n flattened matrix* of $\mathcal{A}$, denoted by $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times (I_1 I_2 \cdots I_{n-1} I_{n+1} \cdots I_N)}$, is obtained by parallelly concatenating all the mode-$n$ vectors of $\mathcal{A}$. The *mode-n rank* of $\mathcal{A}$, denoted by $R_n$, is defined as the dimensionality of the vector space generated by the mode-$n$ vectors: $R_n = rank_n(\mathcal{A}) = rank(\mathbf{A}_{(n)})$. A tensor can be multiplied by a matrix. The *mode-n product* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_n \times \cdots \times I_N}$ and a matrix $\mathbf{M} \in \mathbb{R}^{J_n \times I_n}$, denoted by $\mathcal{B} = \mathcal{A} \times_n \mathbf{M}$, is a tensor of dimensionality $\mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$, whose entries are $\mathcal{B}_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} m_{j_n i_n}$. Alternatively, $\mathcal{B}$ can also be calculated by re-tensorizing the matrix $\mathbf{B}_{(n)} = \mathbf{M}\mathbf{A}_{(n)}$.

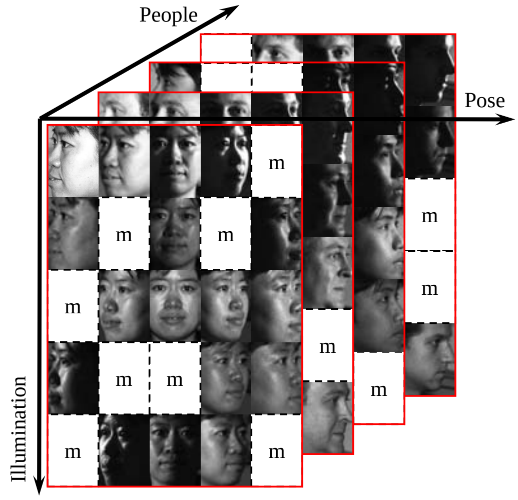


**Figure 1: Tensor representation of a subset from the CMU PIE database. Missing parts are labeled by 'm'.**

## 3. MULTILINEAR SUBSPACE ANALYSIS WITH MISSING VALUES

A tensor is a natural structure for data resulting from the interaction of multiple factors. Each mode of the tensor corresponds to one factor. A typical tensor representation of a set of face images is shown in Fig. 1. The face images are a subset of the CMU PIE database [6], which vary in identity, pose and illumination, but not all combinations of states of the three factors are available. They are assembled into a fourth-order tensor, with three modes shown in Fig. 1 corresponding to people, pose and illumination, respectively, and the fourth mode corresponding to the features extracted from the images (shown here as images for better display). The missing factor state combinations cause missing values in the tensor, marked by 'm' in the figure. The goal of $M^2$SA is to decompose such incomplete tensors into a set of parsimonious features representing the semantic factors respectively.

### 3.1 N-Mode Dimensionality Reduction

Suppose a tensor $\mathcal{D} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times I_{N+1}}$ consists of samples formed from $N$ factors. Note that the $(N+1)$-th mode is used to store the features extracted from the samples. The $N$-mode SVD algorithm [9] can be used to decompose $\mathcal{D}$ as the mode-$n$ product of $N$ orthogonal spaces:

$$\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \cdots \times_n \mathbf{U}_n \cdots \times_N \mathbf{U}_N, \qquad (1)$$

where $\mathcal{Z}$ is called the *core tensor*, and the mode matrices $\mathbf{U}_n (n = 1, 2, \ldots, N)$ contain the orthonormal vectors spanning the factor spaces, one for each contributory factor. Basically, $N$-mode SVD applies the matrix SVD to each of the mode-$n$ flattened matrices $\mathbf{D}_{(n)} (n = 1, 2, \ldots, N)$ of $\mathcal{D}$, obtains the left matrix of the SVD

**Algorithm 1**: $N$-Mode Dimensionality Reduction

**Input**: $\mathcal{D}$ and the target rank $(R_1, R_2, \ldots, R_N)$
**Output**: Rank-reduced approximation $\hat{\mathcal{D}}$

Apply $N$-mode SVD algorithm to $\mathcal{D}$;
Truncate each mode matrix $\mathbf{U}_n$ to $R_n$ columns, obtain the initial mode matrices $\mathbf{U}_1^0, \mathbf{U}_2^0, \ldots, \mathbf{U}_N^0$;
$i \leftarrow 0$;
**repeat**
  $i \leftarrow i + 1$;
  **for** $n \leftarrow 1$ **to** $N$ **do**
    $\tilde{\mathcal{U}}_n^i \leftarrow \mathcal{D} \times_1 (\mathbf{U}_1^i)^T \cdots \times_{n-1} (\mathbf{U}_{n-1}^i)^T \times_{n+1} (\mathbf{U}_{n+1}^{i-1})^T \cdots \times_N (\mathbf{U}_N^{i-1})^T$;
    Mode-$n$ flatten tensor $\tilde{\mathcal{U}}_n^i$ to obtain $\tilde{\mathbf{U}}_n^i$;
    Set $\mathbf{U}_n^i$ to the first $R_n$ columns of the left matrix of the SVD of $\tilde{\mathbf{U}}_n^i$;
  **end**
**until** $\|(\mathbf{U}_n^i)^T \mathbf{U}_n^{i-1}\| > (1-\varepsilon) R_n$ $(n = 1, 2, \ldots, N)$ ;
$\hat{\mathbf{U}}_n \leftarrow \mathbf{U}_n^i$ $(n = 1, 2, \ldots, N)$;
$\hat{\mathcal{Z}} \leftarrow \tilde{\mathcal{U}}_N^i \times_N \hat{\mathbf{U}}_N^T$;
$\hat{\mathcal{D}} \leftarrow \hat{\mathcal{Z}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_2 \cdots \times_N \hat{\mathbf{U}}_N$;

as $\mathbf{U}_n$ for each $n$, and then computes the core tensor

$$\mathcal{Z} = \mathcal{D} \times_1 \mathbf{U}_1^T \times_2 \mathbf{U}_2^T \cdots \times_n \mathbf{U}_n^T \cdots \times_N \mathbf{U}_N^T. \tag{2}$$

In order to get a compact representation of the contributory factors, the dimensionality of the decomposed orthogonal spaces can be reduced. However, the optimal dimensionality reduction in multilinear analysis is not as simple as that in PCA by directly removing those eigenvectors associated with the smallest eigenvalues. The *$N$-mode dimensionality reduction* algorithm [9] is summarized in Algorithm 1. The goal is to find a best rank-$(R_1, R_2, \ldots, R_N)$ approximation $\hat{\mathcal{D}} = \hat{\mathcal{Z}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_2 \cdots \times_N \hat{\mathbf{U}}_N$, with orthonormal mode matrices $\hat{\mathbf{U}}_n$ of lower rank $R_n < I_n$ for $n = 1, 2, \ldots, N$.

## 3.2 Dealing with Missing Values

The missing values in the tensor $\mathcal{D}$ prevent the direct application of Algorithm 1. To deal with this problem, here we propose the M$^2$SA algorithm. Suppose the index for the available values in $\mathcal{D}$ is $\mathcal{I}$, which is also a tensor of the same size. $\mathcal{I}_{i_1 i_2 \ldots i_N} = 1$ if $\mathcal{D}_{i_1 i_2 \ldots i_N}$ is available, otherwise, $\mathcal{I}_{i_1 i_2 \ldots i_N} = 0$. Instead of finding a best approximation for $\mathcal{D}$, the goal is changed into finding a best approximation for the available values, i.e., finding a low-rank $\hat{\mathcal{D}}$ which minimizes the reconstruction error of the available values

$$\Delta_a = \|(\mathcal{D} - \hat{\mathcal{D}}).\times\mathcal{I}\|, \tag{3}$$

where $.\times$ represents the element-wise multiplication, and $\|\cdot\|$ represents the Frobenius norm of a tensor. M$^2$SA uses an iterative process to gradually reduce $\Delta_a$. When initializing, each missing value is filled by the mean over all the available values sharing some contributory factors with the missing value. Then the $N$-mode dimensionality reduction algorithm is applied to the fulfilled tensor to obtain the initial mode matrices $\hat{\mathbf{U}}_n^0$ $(n = 1, 2, \ldots, N)$ and the core tensor $\hat{\mathcal{Z}}^0$. The initial reconstruction of $\mathcal{D}$ is therefore $\hat{\mathcal{D}}^0 = \hat{\mathcal{Z}}^0 \times_1 \hat{\mathbf{U}}_1^0 \times_2 \hat{\mathbf{U}}_2^0 \cdots \times_N \hat{\mathbf{U}}_N^0$. In the iteration $i$, the missing values of $\mathcal{D}$ are updated by the corresponding reconstructions:

$$\mathcal{D}^i = \mathcal{D}.\times\mathcal{I} + \hat{\mathcal{D}}^{i-1}.\times(\sim\mathcal{I}), \tag{4}$$

where $\sim$ is the boolean NOT operator. After that, the $N$-mode dimensionality reduction algorithm is applied to the updated tensor

**Algorithm 2**: M$^2$SA

**Input**: $\mathcal{D}$, $\mathcal{I}$, and the target rank $(R_1, R_2, \ldots, R_N)$
**Output**: Rank-reduced approximation $\hat{\mathcal{D}}$

Fill each missing value in $\mathcal{D}$ with the mean over all the available values sharing some contributory factors to obtain the initialized training tensor $\mathcal{D}^0$;
Apply Algorithm 1 to $\mathcal{D}^0$ to get the initial low-rank approximation $\hat{\mathcal{D}}^0 = \hat{\mathcal{Z}}^0 \times_1 \hat{\mathbf{U}}_1^0 \times_2 \hat{\mathbf{U}}_2^0 \cdots \times_N \hat{\mathbf{U}}_N^0$;
$i \leftarrow 0$;
**repeat**
  $i \leftarrow i + 1$;
  $\mathcal{D}^i \leftarrow \mathcal{D}.\times\mathcal{I} + \hat{\mathcal{D}}^{i-1}.\times(\sim\mathcal{I})$;
  Apply Algorithm 1 to $\mathcal{D}^i$ to obtain the new low-rank approximation $\hat{\mathcal{D}}^i = \hat{\mathcal{Z}}^i \times_1 \hat{\mathbf{U}}_1^i \times_2 \hat{\mathbf{U}}_2^i \cdots \times_N \hat{\mathbf{U}}_N^i$;
  $\Delta_a^i \leftarrow \|(\mathcal{D}^i - \hat{\mathcal{D}}^i).\times\mathcal{I}\|$;
**until** $\Delta_a^i < \varepsilon$ **or** $i > \tau$ ;
$\hat{\mathcal{D}} \leftarrow \hat{\mathcal{Z}}^i \times_1 \hat{\mathbf{U}}_1^i \times_2 \hat{\mathbf{U}}_2^i \cdots \times_N \hat{\mathbf{U}}_N^i$;

$\mathcal{D}^i$ to obtain the new mode matrices $\hat{\mathbf{U}}_n^i$ and the new core tensor $\hat{\mathcal{Z}}^i$. The whole procedure repeats until $\Delta_a$ becomes smaller than a predefined threshold $\varepsilon$ or the time of iteration reaches a maximum number $\tau$. The whole process is summarized in Algorithm 2.

As can be seen, the missing values are reconstructed by those available values sharing some factor values in the multilinear way. Technically speaking, a missing value can be reconstructed as long as there is one available value sharing one factor with it. This means the algorithm can work even when the majority of the tensor are missing values. However, too few available values will induce a poor approximation of the missing value. A relatively good approximation needs some available values on each sharing factor. According to the later experimental results, the algorithm can work well on the tensor with up to 70% missing values.

## 3.3 Test Sample Modeling

Given a previously unseen test sample $\boldsymbol{b}$. It can be modeled by the mode-$n$ product of a set of coefficient vectors:

$$\boldsymbol{b}^T = \mathcal{Z} \times_1 \boldsymbol{c}_1^T \times_2 \boldsymbol{c}_2^T \cdots \times_N \boldsymbol{c}_N^T, \tag{5}$$

where $\mathcal{Z}$ is the core tensor obtained by applying Algorithm 2 to the incomplete training tensor $\mathcal{D}$, and $\boldsymbol{c}_n (n = 1, 2, \ldots, N)$ is the coefficient vector corresponding to the state of factor $n$ associated to the test sample. Note that both $\mathcal{D}$ and $\mathcal{Z}$ are of order $N+1$, with the $(N+1)$-th mode corresponding to the image features. Then the response tensor $\mathcal{R}$ can be calculated by

$$\begin{aligned} \mathcal{R} &= \mathcal{Z}^{+(N+1)} \times_{(N+1)} \boldsymbol{b}^T \\ &= \boldsymbol{c}_1^T \circ \boldsymbol{c}_2^T \cdots \circ \boldsymbol{c}_N^T, \end{aligned} \tag{6}$$

where $\mathcal{Z}^{+(N+1)}$ is the *mode-(N+1) pseudo-inverse tensor* of $\mathcal{Z}$, which can be obtained by re-tensorizing the matrix $\mathbf{P} = \mathbf{Z}_{(N+1)}^{+T}$ (the transpose of the pseudo-inverse of the mode-$(N+1)$ flattened matrix of $\mathcal{Z}$, please refer to [10] for more details). Thus $\mathcal{R}$ is the outer product of all factor coefficient vectors associated with $\boldsymbol{b}$, and therefore is of rank-$(1, \ldots, 1)$. Then Algorithm 1 is used to find a best rank-$(1, \ldots, 1)$ approximation of $\mathcal{R}$, which leads to

$$\hat{\mathcal{R}} = \hat{\mathcal{T}} \times_1 \hat{\boldsymbol{c}}_1 \times_2 \hat{\boldsymbol{c}}_2 \cdots \times_N \hat{\boldsymbol{c}}_N, \tag{7}$$

where $\hat{\boldsymbol{c}}_n$ is the approximation of $\boldsymbol{c}_n$.

For classification, the coefficient vector $\hat{\boldsymbol{c}}_t$ corresponding to the target factor is compared with each row vector of $\mathbf{U}_t$, which cor-

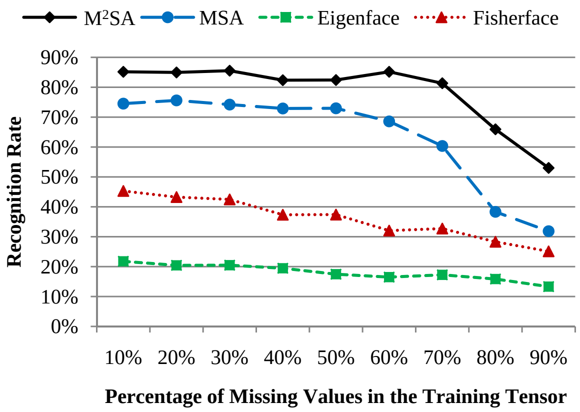


**Figure 2: Face recognition rate with different percentage of missing values in the training tensor.**

responds to one state of the $t$-th factor, and the most similar row vector will then indicate the predicted state of the target factor for the test sample. In this paper, the similarity between the coefficient vectors is measured by the angle between them.

## 4. EXPERIMENT

The experimental data is the 'illum' subset of the the CMU PIE database [6]. There are 68 individuals with 13 different poses and 21 different illumination conditions in this data set. The faces are normalized by fixing the positions of the two eyes (for those profile faces, the positions of one eye and the nose tip are used). The normalized face image has $67 \times 47$ pixels.

The face models are tested in the scenario of face recognition, i.e., the target factor is the identity. The 10-fold cross validation is used to evaluate the performance of the face models. In each fold, 10% of the images are randomly selected as the test set, the remaining are used as the training set. The final result is the average over the 10 folds. The training images are organized into a fourth-order tensor $\mathcal{D} \in \mathbb{R}^{68\times13\times21\times3149}$, where the four modes correspond to people, pose, illumination, and image pixels, respectively. Removing the test images leaves 10% missing values in $\mathcal{D}$. A typical portion of $\mathcal{D}$ is shown in Fig. 1. In order to test the capability of $M^2$SA to deal with missing values, the images in $\mathcal{D}$ are gradually reduced from 90% to only 10% of the total data set with the step 10%, while the test set remains the same 10% of the total data set.

The compared baseline methods are the standard MSA and two linear methods: Eigenface [8] and Fisherface [2]. For $M^2$SA and MSA, if not specified, the rank $R_n$ of the mode-$n$ subspace is set to $2/3$ of that of the original space $I_n$. In order to apply the standard MSA, the missing values in the tensor are filled with the mean of available values sharing some contributory factors. For Eigenface and Fisherface, there is no missing value problem, but just a decrease of the number of training samples. The subspace in Eigenface is set to explain 95% of the variance. Fisherface uses the same settings as in [2].

The face recognition rates of the algorithms are compared in Fig. 2. $M^2$SA achieves the best performance in all cases. It keeps relatively steady at a high level above 80% while the missing values in the training tensor gradually increase from 10% to 70%. Only when the missing values account for 80% or higher of the training tensor, the performance of $M^2$SA starts to notably deteriorate. The standard MSA performs better than the linear methods (Eigenface and Fisherface), but this superiority rapidly shrinks when the missing values become dominating. Note that even when 90% of the tensor are missing values, the recognition rate of $M^2$SA is still significantly higher than that of the linear methods.

## 5. CONCLUSION

Every face image is the result of the interaction of multiple factors. Therefore, the key issue of face image modeling is to explicitly and independently represent these contributory semantic factors. To this end, Multilinear Subspace Analysis (MSA) provides a promising way by decomposing a well organized training tensor with each mode corresponding to one contributory factor. Unfortunately, such a 'complete' training set is usually unavailable in many real applications. The $M^2$SA algorithm proposed in this paper makes it possible for MSA to work on the tensors with a large amount of missing values. Experiments on face recognition reveal that $M^2$SA can perform stably with massive missing values accounting for up to 70% of the total data. Besides face recognition, $M^2$SA can also be applied to many other multimedia applications, such as image reconstruction, facial age estimation, etc.

## 6. ACKNOWLEDGEMENT

This work was supported by ARC Discovery Grant (DP0987421), NSFC (60325207, 60721002), JiangsuSF (BK2008018), 863 Program (2007AA01Z169) and Jiangsu 333 Program.

## 7. REFERENCES

[1] E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE Trans. Knowledge and Data Engineering*, 21(1):6–20, 2009.

[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] X. Geng, Z.-H. Zhou, and H. Dai. Uncontrolled face recognition by individual stable neural network. In *Proc. PRICAI*, pages 553–562, Guilin, China, 2006.

[4] X. Geng, Z.-H. Zhou, and K. Smith-Miles. Individual stable space: An approach to face recognition under uncontrolled conditions. *IEEE Transactions on Neural Networks*, 19(8):1354–1368, 2008.

[5] S. T. Roweis. EM algorithms for PCA and SPCA. In *Advances in Neural Information Processing Systems 10*, pages 626–632, Denver, CO, 1997.

[6] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression database. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(12):1615–1618, 2003.

[7] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B: Statistical Methodology*, 61:611–622, 1999.

[8] M. A. Turk and A. Pentland. Eigenface for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[9] M. A. O. Vasilescu and D. Terzopoulos. Multilinear subspace analysis of image ensembles. In *Proc. CVPR*, pages 93–99, Madison, WI, 2003.

[10] M. A. O. Vasilescu and D. Terzopoulos. Multilinear projection for appearance-based recognition in the tensor framework. In *Proc. ICCV*, pages 1–8, Rio de Janeiro, Brazil, 2007.

[11] T. Wiberg. Computation of principal component when data are missing. In *Proc. the 2nd Symp. Computational Statistics*, pages 229–236, Berlin, Germany, 1976.