Multifactor Analysis for fMRI Brain Image Classification by Subject and Motor Task

Sung Won Park
Electrical and Computer Engineering Department
Carnegie Mellon University
sungwonp@cmu.edu

May 5, 2011

Abstract

FMRI brain images are generated by the variation of multiple factors, such as subject, motor task, and time frame. Just as this example demonstrates, in image analysis, much work has been aimed at analyzing a set of images generated by variation of multiple factors. To perform image analysis successfully, it is often necessary to model multiple factor frameworks found in image sets. One leading method for multifactor analysis is Multilinear Principal Component Analysis (MPCA), also called Tensorfaces, based on Higher-Order Singular Value Decomposition in tensor algebra. While traditional dimension reduction methods use a single low-dimensional vector to represent an original high-dimensional sample vector, MPCA uses multiple low-dimensional vectors associated with multiple factors. This project extends Multifactor Kernel PCA (MKPCA), a kernel-based version of MPCA, based on kernels that consist of multiple sub-kernels. Based on multifactor analysis provided by MKPCA, in this project, we conducted experiments using a set of 4D spatiotemporal fMRI brain images with two factors: 14 subjects and 4 motor tasks. Subject parameters and motor-task parameters, obtained by decomposing the influences of the two factors, were used for classifying fMRI images by subject and motor task, respectively. The classification accuracy obtained by our experiments demonstrates the advantages of our proposed method over current leading methods for multifactor analysis. In particular, MKPCA outperformed MPCA, as MKPCA was able to provide more reliable analysis for nonlinear structures in real-world data sets.

Contents

1 Introduction

In image analysis, we often aim to analyze a set of images generated by varying multiple factors. For example, face images are generated by variation of multiple factors, such as subject, viewpoint, and lighting condition. Another example includes fMRI brain images generated by the variation of subject and motor task. Thus, for successful image analysis, it is often necessary to model multiple factor frameworks found in image sets.

One leading method for multifactor analysis is Multilinear Principal Component Analysis (MPCA), also called Tensorfaces [11] [13], based on tensor frameworks and tensor algebra. MPCA has been used to analyze the interaction between multiple factors. After establishing multiple dimensions based on multiple factors, MPCA computes a linear subspace that represents the variation of each factor. By using MPCA, a data set is represented as a tensor product of multiple orthogonal matrices, where the columns of each matrix span the subspace constructed by the variation of each factor. Similarly, in MPCA, one image is represented as a tensor product of multiple factor-dependent parameters that describe the features of multiple factors. In other words, MPCA can decompose the characteristics of each sample vector into multiple parameters associated with multiple factors. For example, MPCA separately parameterizes the subject and the motor task of each fMRI image. In the context of the classification of fMRI images by subject, the main advantage of MPCA results from its ability to obtain each fMRI image's subject parameter. These subject parameters represent the invariant characteristics of each subject under other varying factors, such as motor task. Consequently, using MPCA, a single subject's fMRI images have identical subject parameters regardless of varying motor tasks, while different subjects' fMRI images have different subject parameters. Thus, these subject parameters have been applied to the classification of fMRI images by subject. Similarly, the motor-task parameters of fMRI images can be applied to classification by motor task. Recently, a kernel-based extension of MPCA called Multifactor Kernel Principal Component Analysis (MKPCA) has been introduced [6] [7].

The diagram in Figure 1 summarizes the historical evolution of dimensionality reduction methods from Principal Component Analysis (PCA) [9] to MKPCA. PCA is most often cited as the precursor to several subspace methods that have been applied to dimensionality reduction. The basic methodology of PCA is to project sample vectors into the linear subspace spanned by the principal directions with the largest variance of data distribution. Kernel Principal Component Analysis (KPCA) [8], a nonlinear variant of PCA, assumes that there exists a nonlinear map that makes mapped observations linearly separable, and then applies PCA to them. Thus, the subspace achieved by KPCA is nonlinear in the input space, which allows KPCA to be successfully applied to complex real-world data that are not linearly separable. Another variant of PCA is aforementioned MPCA, an extension of PCA to multiple

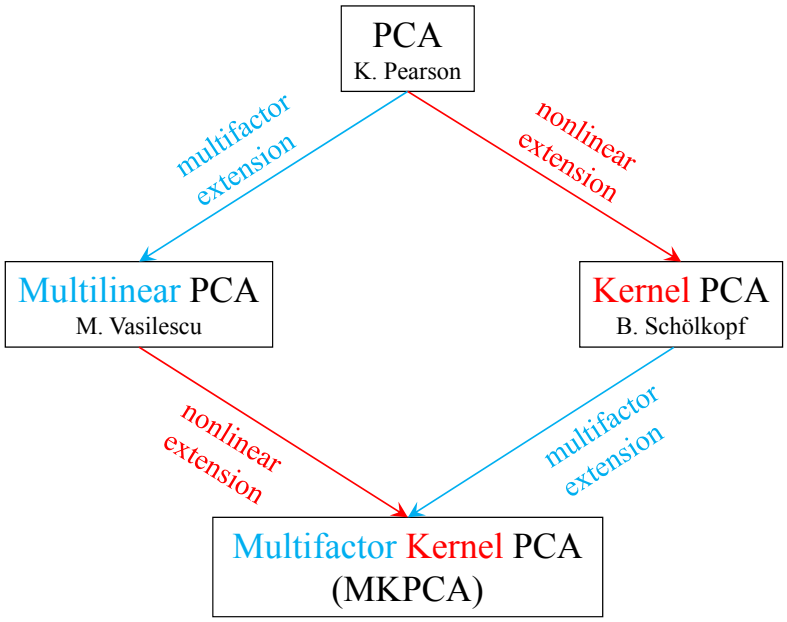


Figure 1: Historical evolution of dimensionality reduction techniques from PCA to MKPCA.

factor frameworks. Recently, by combining KPCA and MPCA, a more advanced subspace method has been introduced, called MKPCA. MKPCA can be thought of as both a kernel-based extension of MPCA and a multifactor extension of KPCA.

In this project, we propose an extension of MKPCA based on kernel functions that can be successfully applied to structured data, such as fMRI images. When data samples are originally represented as matrices or tensors, such as 2D or 3D images, MKPCA proposed in [6] and [7] merely vectorizes these sample matrices or tensors and applies kernel functions to these vectors. For example, each fMRI brain image is originally a 3rd-order tensor consisting of 3D voxels; however, traditional MKPCA based on vectorization cannot take advantage of this 3rd-order structure in each fMRI image. To overcome this limitation in previous work, we apply kernels that can make full use of structured data. Our experiments aimed to classify fMRI images by subject and motor task using subject parameters and motor-task parameters, respectively. We used the fMRI brain scan database from the fMRI Data Center (fMRIDC) [3] with 14 subjects, four motor tasks, and 66 time frames.

2 Multifactor Analysis Based on Multilinear Algebra

Multilinear Principal Component Analysis (MPCA) [11] [12], also called Tensorfaces, is the application of Higher-Order Singular Value Decomposition (HOSVD) [5] to multiple factor frameworks.

MPCA aims to decompose the characteristics of each data sample into multiple parameters associated with multiple factors. In this section, we review PCA, on which MPCA is based, and MPCA.

2.1 Principal Component Analysis (PCA)

PCA [9] can be achieved using Singular Value Decomposition (SVD) [2]. Let $\mathbf{X}$ denote the $d \times n$ data matrix whose columns represent n training sample vectors in $\mathbb{R}^d$. We assume that these data samples are centered at zero. As illustrated in Figure 2, the truncated SVD represents the data matrix $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T, \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{d \times d'}$ denotes the left singular vector matrix, $\mathbf{S} \in \mathbb{R}^{d' \times d'}$ denotes the singular value matrix, and $\mathbf{V} \in \mathbb{R}^{n \times d'}$ denotes the right singular vector matrix for $d' \leq \min(d, n)$. Here, $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, and $\mathbf{S}$ is a diagonal matrix.

Since $\mathbf{U}$ is an orthogonal matrix, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ denotes an identity matrix. Thus, we can see that $\mathbf{U}^+ = \mathbf{U}^T$, where $+$ denotes the Moore-Penrose pseudoinverse. PCA performs dimensionality reduction through the linear projection matrix $\mathbf{U}$:

$$\mathbf{Y} = \mathbf{U}^+\mathbf{X} = \mathbf{U}^T\mathbf{X} = \mathbf{S}\mathbf{V}^T \in \mathbb{R}^{d' \times n}. \tag{2}$$

The columns of $\mathbf{U}$ represent the eigenvectors of the covariance matrix $\mathbf{X}\mathbf{X}^T$. Traditionally, when PCA is used for classification, the last few eigenvectors are removed to improve performance; that is, the last few columns of $\mathbf{U}$ are discarded by truncation.

In Figure 2, the parts represented in red show that $\mathbf{v}_i$, each row of the matrix $\mathbf{V}$, can be thought of as the low-dimensional representation of $\mathbf{x}_i$, each column of $\mathbf{X}$:

$$\mathbf{x}_i = \mathbf{U}\mathbf{S}\mathbf{v}_i. \tag{3}$$

2.2 Multilinear Principal Component Analysis (MPCA)

MPCA is a multilinear extension of PCA. Just as PCA can be achieved using SVD, MPCA can be achieved using Higher-Order Singular Value Decomposition (HOSVD) [5], a multilinear extension of matrix SVD.

MPCA computes a linear subspace generated by varying each factor as well as the linear subspace of the sample space itself. In this paper, we consider a set of fMRI images with two factors: n_s subjects and n_m motor tasks. Since in general, MPCA vectorizes data samples, we assume that $n_x \times n_y \times n_z$

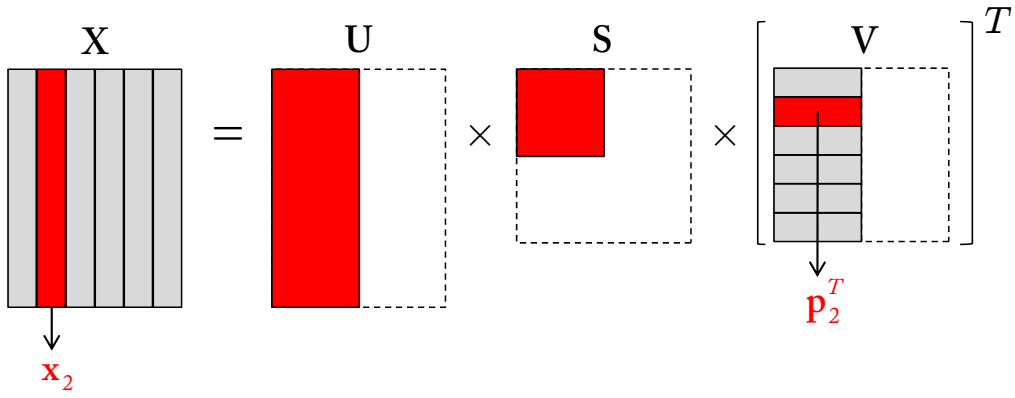


Figure 2: SVD of a data matrix $\mathbf{X}$.

fMRI images with $d(= n_x n_y n_z)$ voxels are vectored and reshaped as d-dimensional vectors. Let $\mathcal{X} \in \mathbb{R}^{d \times n_s \times n_m}$ denote a 3rd-order tensor consisting of these vectored fMRI images for n_s subjects and n_m motor tasks. For analyzing the multifactor structure, HOSVD represents a tensor $\mathcal{X}$ in terms of tensor products:

$$\mathcal{X} = \mathcal{Z} \times_1 \mathbf{U}_v \times_2 \mathbf{U}_s \times_3 \mathbf{U}_m. \tag{4}$$

In (4), $\mathcal{Z} \in \mathbb{R}^{d' \times n_s' \times n_m'}$ is called a core tensor, and $\mathbf{U}_v \in \mathbb{R}^{d \times d'}$, $\mathbf{U}_s \in \mathbb{R}^{n_s \times n_s'}$, and $\mathbf{U}_m \in \mathbb{R}^{n_m \times n_m'}$ are orthogonal matrices, where $d \geq d$, $n_s \geq n_s'$, and $n_m \geq n_m'$.

The matrix $\mathbf{U}_v$ can be calculated as the left singular vector matrix of $\mathbf{X}_{(v)}$, where the matrix $\mathbf{X}_{(v)} \in \mathbb{R}^{d \times n_s n_m}$ is obtained by flattening a tensor $\mathcal{X}$ in the mode-1, i.e., the voxel mode. In other words, if a vector $\mathbf{x}_{i,j} \in \mathbb{R}^d$ denotes a vectorized fMRI image of subject i with motor task j, as shown in Figure 3, a matrix $\mathbf{X}_{(v)}$ can be written as

$$\mathbf{X}_{(v)} = [x_{1,1}, x_{2,1}, \cdots, x_{n_s,1}, x_{1,2}, \cdots, x_{n_s,n_m}]\,. \tag{5}$$

Note that $\mathbf{U}_v$ is technically the same as $\mathbf{U}$ in (1), where $n = n_s n_m$.

Similarly, $\mathbf{U}_s$ and $\mathbf{U}_m$ can be calculated as the left singular vector matrices of $\mathbf{X}_{(s)} \in \mathbb{R}^{n_s \times n_m d}$ and $\mathbf{X}_{(m)} \in \mathbb{R}^{n_m \times n_s d}$, respectively. Here, $\mathbf{X}_{(s)}$ and $\mathbf{X}_{(m)}$ are the mode-2 (or the subject mode) and mode-3 (or the motor-task mode) flattening matrices of the tensor $\mathcal{X}$. Just like $\mathbf{X}_{(v)}$ in (5), the two matrices $\mathbf{X}_{(s)}$ and $\mathbf{X}_{(m)}$ can be formulated in terms of $\mathbf{x}_{i,j}$:

$$\mathbf{X}_{(s)} = \begin{bmatrix} \mathbf{x}_{1,1}^T & \mathbf{x}_{1,2}^T & \cdots & \mathbf{x}_{1,n_m}^T \\ \mathbf{x}_{2,1}^T & \mathbf{x}_{2,2}^T & \cdots & \mathbf{x}_{2,n_m}^T \\ \vdots & & & \\ \mathbf{x}_{n_s,1}^T & \mathbf{x}_{n_s,2}^T & \cdots & \mathbf{x}_{n_s,n_m}^T \end{bmatrix} \; and \tag{6}$$

$$\mathbf{X}_{(m)} = \begin{bmatrix} \mathbf{x}_{1,1}^T & \mathbf{x}_{2,1}^T & \cdots & \mathbf{x}_{n_s,1}^T \\ \mathbf{x}_{1,2}^T & \mathbf{x}_{2,2}^T & \cdots & \mathbf{x}_{n_s,2}^T \\ \vdots & & & \\ \mathbf{x}_{1,n_m}^T & \mathbf{x}_{2,n_m}^T & \cdots & \mathbf{x}_{n_s,n_m}^T \end{bmatrix}. \tag{7}$$

After calculating the three orthogonal matrices $\mathbf{U}_v$, $\mathbf{U}_s$, and $\mathbf{U}_m$, a core tensor $\mathcal{Z}$ can be calculated as

$$\mathcal{Z} = \mathcal{X} \times_1 \mathbf{U}_v^T \times_2 \mathbf{U}_s^T \times_3 \mathbf{U}_m^T. \tag{8}$$

In Figure 3(a), the parts represented in red show that each sample vector $\mathbf{x}_{i,j}$ can be formulated as

$$\mathbf{x}_{i,j} = \mathcal{Z} \times_1 \mathbf{U}_v \times_2 \mathbf{s}_i^T \times_3 \mathbf{m}_j^T, \tag{9}$$

where a vector $\mathbf{s}_i \in \mathbb{R}^{n'_s}$ is the i-th row vector of $\mathbf{U}_s$ and a vector $\mathbf{m}_j \in \mathbb{R}^{n'_m}$ is the j-th row vector of $\mathbf{U}_m$. These two vectors $\mathbf{s}_i$ and $\mathbf{m}_j$ represent the subject and motor-task parameters of $\mathbf{x}_{i,j}$, respectively.

While the HOSVD of $\mathcal{X}$ is formulated in terms of tensors and tensor products as shown in (4), the equivalent of (4) can be formulated in terms of matrices and Kronecker products:

$$\mathbf{X}_{(v)} = \mathbf{U}_v \mathbf{Z}_{(v)} (\mathbf{U}_s \otimes \mathbf{U}_m)^T, \tag{10}$$

where $\otimes$ denotes the Kronecker product and $\mathbf{Z}_{(v)}$ denotes the mode-1 (voxel mode) flattening matrix of a core tensor $\mathcal{Z}$. Similarly, the equivalent of (9) can be derived as

$$\mathbf{x}_{i,j} = \mathbf{U}_v \mathbf{Z}_{(v)} (\mathbf{s}_i \otimes \mathbf{m}_j) \tag{11}$$

using matrices and the Kronecker product instead of tensors and tensor products. Figure 3(a) and Figure 3(b) compare the two equivalent formulations of HOSVD in a tensor framework and in a matrix framework, respectively.

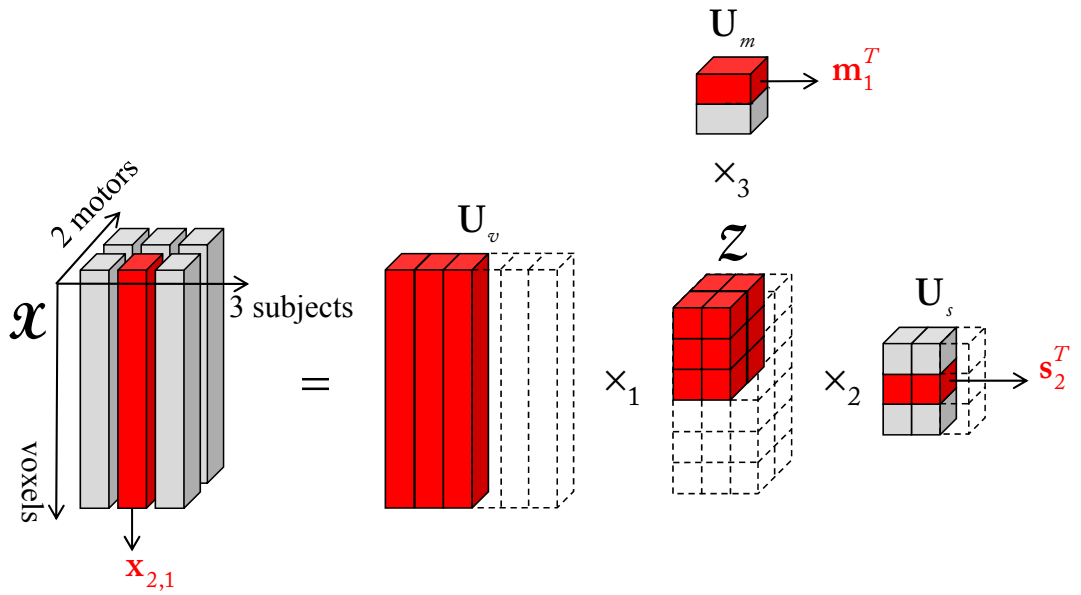
MPCA clearly defines the generalization of parameter decomposition to out-of-sample data. The representation of each labeled training sample defined in (11) is generalized to the representation of any unlabeled test sample $\mathbf{x}$:

$$\mathbf{x} = \mathbf{U}_v \mathbf{Z}_{(v)} (\mathbf{s} \otimes \mathbf{m}). \tag{12}$$

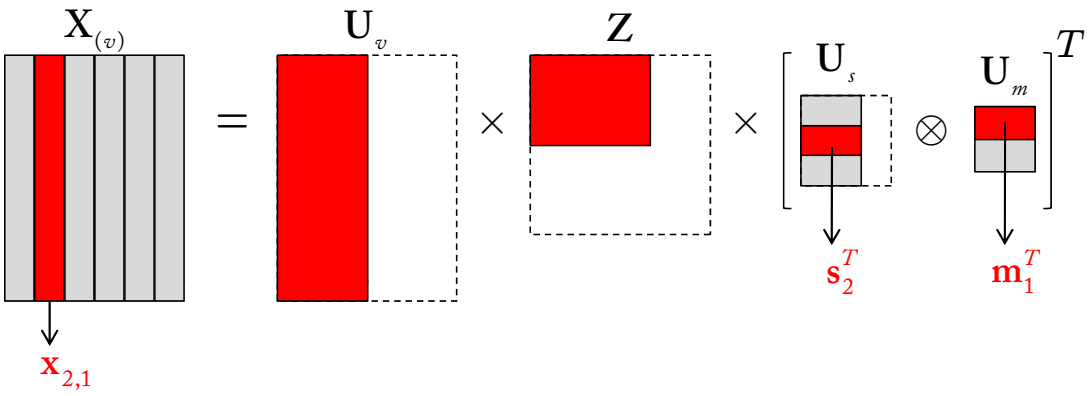
Note that $\mathbf{x}$'s labels for the two factors are not given, and thus, each factor-dependent parameter is unknown. However, although both $\mathbf{s}$ and $\mathbf{m}$ are unknown, we can calculate $\mathbf{s} \otimes \mathbf{m}$, the *joint parameter* of $\mathbf{x}$:

$$\mathbf{s} \otimes \mathbf{m} = \mathbf{Z}_{(v)}^{+} \mathbf{U}_v^T \mathbf{x} \tag{13}$$

using $\mathbf{U}_v$ and $\mathbf{Z}_{(v)}$ achieved by training. The joint parameter $\mathbf{s} \otimes \mathbf{m}$ in (13) can be easily decomposed by rank-$(1, 1, \cdots, 1)$ approximation using the alternating least squares method [13]. Rank-$(1, 1, \cdots, 1)$



(a) Tensor formulation in (4).



(b) Matrix formulation in (10).

Figure 3: Two equivalent formulations of Higher-Order SVD. $\mathbf{x}_{2,1}$ represents a vectorized fMRI image for subject 2 and motor task 1. $\mathbf{s}_2$ represents the subject parameter for subject 2, and $\mathbf{m}_1$ represents the motor-task parameter for motor task 1.

approximation aims to calculate $\mathbf{s} \in \mathbb{R}^{n'_s}$ and $\mathbf{m} \in \mathbb{R}^{n'_m}$ as the left and right singular vectors, respectively, corresponding to the largest singular value of a matrix $\mathbf{s}\mathbf{m}^T \in \mathbb{R}^{n'_s \times n'_m}$.

After decomposing the joint parameter $\mathbf{s} \otimes \mathbf{m}$ into $\mathbf{s}$ and $\mathbf{m}$, one can use only the subject parameter $\mathbf{s}$ for the classification of $\mathbf{x}$ by subject, assuming that $\mathbf{s}$ does not depend on the type of motor task. Similarly, $\mathbf{m}$ can be used for classifying $\mathbf{x}$ by motor task.

3 A Kernel-Based Extension of Multifactor Analysis

In this project, we propose the method for classifying fMRI images by subject and motor task using a kernel-based extension of multifactor analysis. We also propose kernel functions that can be successfully applied to structured data, such as 3D fMRI images.

3.1 Multifactor Kernel Principal Component Analysis (MKPCA)

MKPCA [6] [7] is a recently proposed variant of PCA. Since MKPCA is the combination of KPCA and MPCA, it offers the virtues of both KPCA and MPCA. Based on the observation that the covariance matrices in every factor space of MPCA are formulated with inner products, one can apply the kernel trick to MPCA and consequently obtain MKPCA.

(6) enables us to calculate the matrix $\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T$. If we let $\left[\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T\right]_{r,c}$ denote the (i,j) entry of $\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T$, this entry is calculated as

$$\left[\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T\right]_{r,c} = \sum_{i=1}^{n_m} \mathbf{x}_{r,i}^T \mathbf{x}_{c,i}. \tag{14}$$

Similarly, $\mathbf{X}_{(m)}\mathbf{X}_{(m)}^T$ can be calculated as

$$\left[\mathbf{X}_{(m)}\mathbf{X}_{(m)}^T\right]_{r,c} = \sum_{i=1}^{n_s} \mathbf{x}_{i,r}^T \mathbf{x}_{i,c}. \tag{15}$$

Since the left singular vector matrices of $\mathbf{X}_{(s)}$ and $\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T$ are identical, the orthogonal matrix $\mathbf{U}_s$ in (9) can be calculated as the left singular vector matrix of $\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T$, as well as $\mathbf{X}_{(s)}$. Similarly, the orthogonal matrix $\mathbf{U}_m$ in (9) can be calculated as the left singular vector matrix of $\mathbf{X}_{(m)}\mathbf{X}_{(m)}^T$, as well as $\mathbf{X}_{(m)}$.

In order to extend MPCA's formulations introduced in Section 2.2 to a nonlinear feature space, we let $\bar{\mathcal{X}}$ denote a data tensor whose mode-1 vectors represent nonlinear mappings of fMRI images: $\phi(\mathbf{x}_{1,1})$, $\phi(\mathbf{x}_{2,1})$, $\cdots$, and $\phi(\mathbf{x}_{n_s,n_m})$. [6] and [7]show that although the nonlinear map $\phi(\cdot)$ is not defined

explicitly, MPCA can be performed in the feature space if a kernel function $k(\cdot,\cdot)$, which defines the inner product in the feature space, is given. To review this idea, first, let us show that the nonlinear versions of factor-dependent parameters can be calculated only based on $k(\cdot,\cdot)$ without any explicit use of ϕ. From (14) and (15), we can see that both matrices $\mathbf{X}_{(s)}\mathbf{X}_{(s)}^T$ and $\mathbf{X}_{(m)}\mathbf{X}_{(m)}^T$ are formulated in terms of inner products of pairs of training images. Thus, we can apply the kernel trick and replace (14) and (15) with the flowing nonlinear versions:

$$\left[\bar{\mathbf{X}}_{(s)}\bar{\mathbf{X}}_{(s)}^T\right]_{r,c} = \sum_{i=1}^{n_m} k(\mathbf{x}_{r,i}, \mathbf{x}_{c,i}), \tag{16}$$

$$\left[\bar{\mathbf{X}}_{(m)}\bar{\mathbf{X}}_{(m)}^T\right]_{r,c} = \sum_{i=1}^{n_s} k(\mathbf{x}_{i,r}, \mathbf{x}_{i,c}). \tag{17}$$

Using (16) and (17), we can directly calculate $\bar{\mathbf{X}}_{(s)}\bar{\mathbf{X}}_{(s)}^T$ and $\bar{\mathbf{X}}_{(m)}\bar{\mathbf{X}}_{(m)}^T$, respectively, without any use of $\phi(\cdot)$. Thus, the subspace matrices $\bar{\mathbf{U}}_s$ and $\bar{\mathbf{U}}_m$ can be calculated by the SVD of $\bar{\mathbf{X}}_{(s)}\bar{\mathbf{X}}_{(s)}^T$ and $\bar{\mathbf{X}}_{(m)}\bar{\mathbf{X}}_{(m)}^T$, respectively.

While $\bar{\mathbf{U}}_s$ and $\bar{\mathbf{U}}_m$ can be directly calculated using (16) and (17), $\bar{\mathbf{U}}_v$ cannot be calculated in this manner. This is because $\bar{\mathbf{X}}_{(v)}\bar{\mathbf{X}}_{(v)}^T$ cannot be formulated only in terms of the inner product while $\bar{\mathbf{X}}_{(s)}\bar{\mathbf{X}}_{(s)}^T$ and $\bar{\mathbf{X}}_{(m)}\bar{\mathbf{X}}_{(m)}^T$ can, as shown in (16) and (17). However, using the kernel trick, the factor-dependent parameters of out-of-sample data can be calculated without solving for $\bar{\mathbf{U}}_v$ in MKPCA. The nonlinear version of (13) can be derived as

$$\bar{\mathbf{s}} \otimes \bar{\mathbf{m}} = \bar{\mathbf{Z}}_{(v)}^{+}\bar{\mathbf{U}}_v^T\phi(\mathbf{x}). \tag{18}$$

By flattening in the voxel mode, (8) can be revised as

$$\bar{\mathbf{Z}}_{(v)} = \bar{\mathbf{U}}_v^T\bar{\mathbf{X}}_{(v)}(\bar{\mathbf{U}}_s \otimes \bar{\mathbf{U}}_m). \tag{19}$$

Since $\bar{\mathbf{X}}_{(v)}$ can be decomposed as $\bar{\mathbf{X}}_{(v)} = \bar{\mathbf{U}}_v\bar{\mathbf{S}}_v\bar{\mathbf{V}}_v^T$ using SVD, $\bar{\mathbf{U}}_v$ can be denoted as

$$\bar{\mathbf{U}}_v = \bar{\mathbf{X}}_{(v)}\bar{\mathbf{V}}_v\bar{\mathbf{S}}_v^{-1}, \tag{20}$$

where $\bar{\mathbf{V}}_v$ and $\bar{\mathbf{S}}_v$ can be directly calculated by the SVD of the kernel matrix $\mathbf{K}$:

$$\mathbf{K} = \bar{\mathbf{X}}_{(v)}^T\bar{\mathbf{X}}_{(v)} = \bar{\mathbf{V}}_v\bar{\mathbf{S}}_v^2\bar{\mathbf{V}}_v^T. \tag{21}$$

Using (20) and (21), $\bar{\mathbf{Z}}_{(v)}$ in (19) can be revised as

$$\bar{\mathbf{Z}}_{(v)} = \bar{\mathbf{S}}_v^{-T}\bar{\mathbf{V}}_v^T\bar{\mathbf{X}}_{(v)}^T\bar{\mathbf{X}}_{(v)}(\bar{\mathbf{U}}_s \otimes \bar{\mathbf{U}}_m) = \bar{\mathbf{S}}_v^{-1}\bar{\mathbf{V}}_v^T\mathbf{K}(\bar{\mathbf{U}}_s \otimes \bar{\mathbf{U}}_m). \tag{22}$$

Note that $\bar{\mathbf{Z}}_{(v)}$ can be directly calculated because all of the matrices on the right side in (22) can be solved. Consequently, using (19) and (20), (18) can be derived as

$$\bar{\mathbf{s}} \otimes \bar{\mathbf{m}} = \bar{\mathbf{Z}}_{(v)}^{+} \bar{\mathbf{U}}_v^T \phi(\mathbf{x}) = \bar{\mathbf{Z}}_{(v)}^{+} \bar{\mathbf{S}}_v^{-T} \bar{\mathbf{V}}_v^T \bar{\mathbf{X}}_{(v)}^T \phi(x) = \bar{\mathbf{Z}}_{(v)}^{+} \bar{\mathbf{S}}_v^{-1} \bar{\mathbf{V}}_v^T \mathbf{k}, \tag{23}$$

where the column vector $\mathbf{k}$ is defined as $\mathbf{k} = [k(\mathbf{x}_{1,1}, \mathbf{x}), k(\mathbf{x}_{2,1}, \mathbf{x}), \cdots, k(\mathbf{x}_{n_s,n_m}, \mathbf{x})]^T$. Note that in MKPCA, the joint parameter $\bar{\mathbf{s}} \otimes \bar{\mathbf{m}}$ can be calculated using the kernel matrix $\mathbf{K}$ and the vector $\mathbf{k}$, without the use of ϕ. Just as in (13), rank-$(1, 1, \cdots, 1)$ approximation using the alternating least squares method [13] can be applied to parameter decomposition in (23).

In sum, although a nonlinear map is not defined, by using the kernel trick, factor-dependent parameters can be achieved for both training samples and test samples. Consequently, by combining KPCA and MPCA, the more advanced subspace method, MKPCA, is derived.

3.2 Kernels for Structured Objects

If the various parts of each sample vector have different meanings and should be dealt with differently, we need to define kernels on structured objects. For example, the elements of strings, trees, and graphs are discrete structures. If a sample vector $\mathbf{x}$ can be split into two parts $\mathbf{x}_1$ and $\mathbf{x}_2$, two different kernels can be used for these parts. In literature, kernel functions for structured data, such as R-convolution kernels [4] and ANOVA kernels [10], have been proposed for this purpose.

An fMRI brain tensor $\mathcal{X} \in \mathbb{R}^{n_x \times n_y \times n_z}$ consists of 3D voxels, and thus, this tensor can be decomposed into vectors in three different ways: mode-1, mode-2, and mode-3 vectors as shown in Figure 4. Let $\mathbf{x}_{i,j}$, $\mathbf{y}_{i,j}$, and $\mathbf{z}_{i,j}$ denote mode-1, mode-2, and mode-3 vectors of the tensor $\mathcal{X}$, respectively.

Each original fMRI brain image is a 3rd-order tensor consisting of 3D voxels. To make full use of the tensorial structures in fMRI images, we need to define a kernel function between a pair of fMRI brain tensors. However, traditional MKPCA merely vectorizes these 3rd-order tensors and ignores the original structures in fMRI images. In other words, MKPCA uses a kernel function $k(\cdot, \cdot)$ between two vectors to represent a kernel function $k_{\text{tensor}}(\cdot, \cdot)$ between two tensors:

$$k_{\text{tensor}}(\mathcal{X}, \mathcal{X}') = k(\text{vec}(\mathcal{X}), \text{vec}(\mathcal{X}')), \tag{24}$$

where $\text{vec}(\cdot)$ denotes vector unfolding, and $\mathcal{X}$ and $\mathcal{X}'$ denote 3D fMRI tensors.

In previous work, both R-convolution kernels [4] and ANOVA kernels [10] aim to define kernels between composite objects by building on similarity measures that assess their respective parts. The R-convolution is defined as

$$k_R(\mathbf{x}, \mathbf{x}') = \sum_R \prod_{d=1}^{D} k_d(\mathbf{x}_d, \mathbf{x}'_d), \tag{25}$$

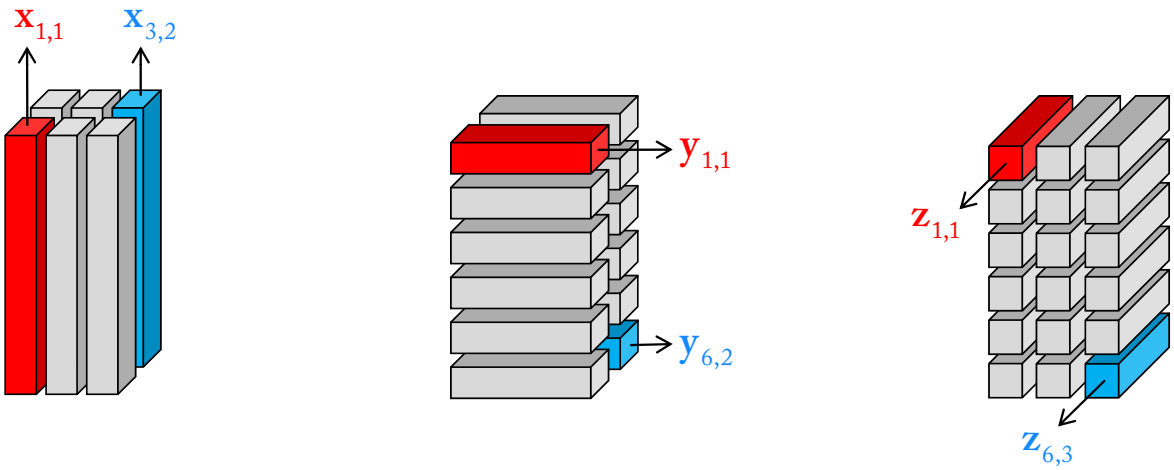


Figure 4: Three vector spaces of a 3rd-order tensor. (a) mode-1 vectors. (b) mode-2 vectors. (c) mode-3 vectors.

where the sum runs over all possible ways in which we can decompose $\mathbf{x}$ into $\mathbf{x}_1, \cdots, \mathbf{x}_D$ and $\mathbf{x}'$ into $\mathbf{x}'_1, \cdots, \mathbf{x}'_D$. (25) shows that during the process of hierarchical merging, larger structures are formed through the continuous merging of smaller structures. This continuous merging is performed by the following order of operators: the kernel functions $k_d(\cdot, \cdot)$, the products $\prod$, and the sums $\sum$. More specifically, in (25), D vectors are ultimately reduced to just a single scalar through a hierarchical merging process that is greedy, in that most similar pairwise vectors are merged by the kernel function during the earliest phase. Subsequently, in the second phase, most similar pairwise function outputs are merged as scalar products, and these scalar products are then all summed up in the final phase. ANOVA kernels provide similar observations. Based on these observations, in this project, we invent the following kernel $k_{\text{3rd}}(\cdot, \cdot)$ that aims to sufficiently represent the three vector spaces of 3rd-order tensors of fMRI images:

$$k_{\text{3rd}}(\mathcal{X}, \mathcal{X}') = \prod_{i}^{n_y} \prod_{j}^{n_z} k_x(\mathbf{x}_{i,j}, \mathbf{x}'_{i,j}) + \prod_{i}^{n_x} \prod_{j}^{n_z} k_y(\mathbf{y}_{i,j}, \mathbf{y}'_{i,j}) + \prod_{i}^{n_x} \prod_{j}^{n_y} k_z(\mathbf{z}_{i,j}, \mathbf{z}'_{i,j}), \tag{26}$$

where $\mathbf{x}'_{i,j}$, $\mathbf{y}'_{i,j}$, and $\mathbf{z}'_{i,j}$ are mode-1, mode-2, and mode-3 vectors of the tensor $\mathcal{X}'$, as shown in Figure 4.

4 Experimental Results

In this section, we demonstrate that the proposed method is appropriate for dimensionality reduction of fMRI data with multiple factors.

4.1 fMRIDC Database

We used the fMRI brain scan database from the fMRI Data Center (fMRIDC) [3] with 14 subjects, four motor tasks, and 66 time frames. Each frame image consists of $64 \times 64 \times 22$ voxels, and one of the images is shown in Figure 5. Each 3D fMRI scan in the original data set has the NIFTI format with a raw data file (.img) and a header (.hdr) file. We converted each scan to a 3D tensor using Spatial Transformations 8 (SPM8) [1], which is one of the most powerful tools to analyze biomedical images such as fMRI, PET, SPECT, EEG, and MEG.

For preprocessing, each 3D temporal fMRI image was vectored and normalized to zero mean and unit variance. After vectorization and normalization, we applied 4D discrete wavelet transform (DWT) using Haar wavelet to each 4D spatiotemporal fMRI image.

4.2 Classification by Subject and Motor Task

We compare the performance of MKPCA using our proposed kernel function $k_{3rd}(\cdot, \cdot)$ to other dimensionality reduction methods including PCA, MPCA, KPCA, and MKPCA using an RBF kernel $k_{\text{RBF}}(\cdot, \cdot)$. To test the quality of dimensionality reduction, we conducted two classification tests: the classification of test images by subject and motor task. We compared the classification rates of all of the methods based on rank-1 classification, i.e., 1-NN classification. To measure the distances in the parameter spaces, we used the Euclidean distance. In order to show the statistical significance of the differences between the results of the aforementioned methods, a One-Way Analysis of Variance test was conducted for the results of leave-one-out cross-validation.

For truncated SVD reviewed in Section 2.1, we used energy thresholding to retain top-m eigenvectors such that

$$\min_m \frac{\sum_{j=1}^{m} \lambda_j}{\sum_{j=1}^{n} \lambda_j} \geq t, \tag{27}$$

where λ_j is the eigenvalue corresponding to the j-th eigenvector for $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Here, n is the total number of eigenvectors and t is the user specified threshold. In other words, energy thresholding is for using a linear subspace that consists of the minimum numbers of basis vectors whose cumulative energy defined in (27) were above t. In our experiments, we set 0.95 as the threshold for PCA and

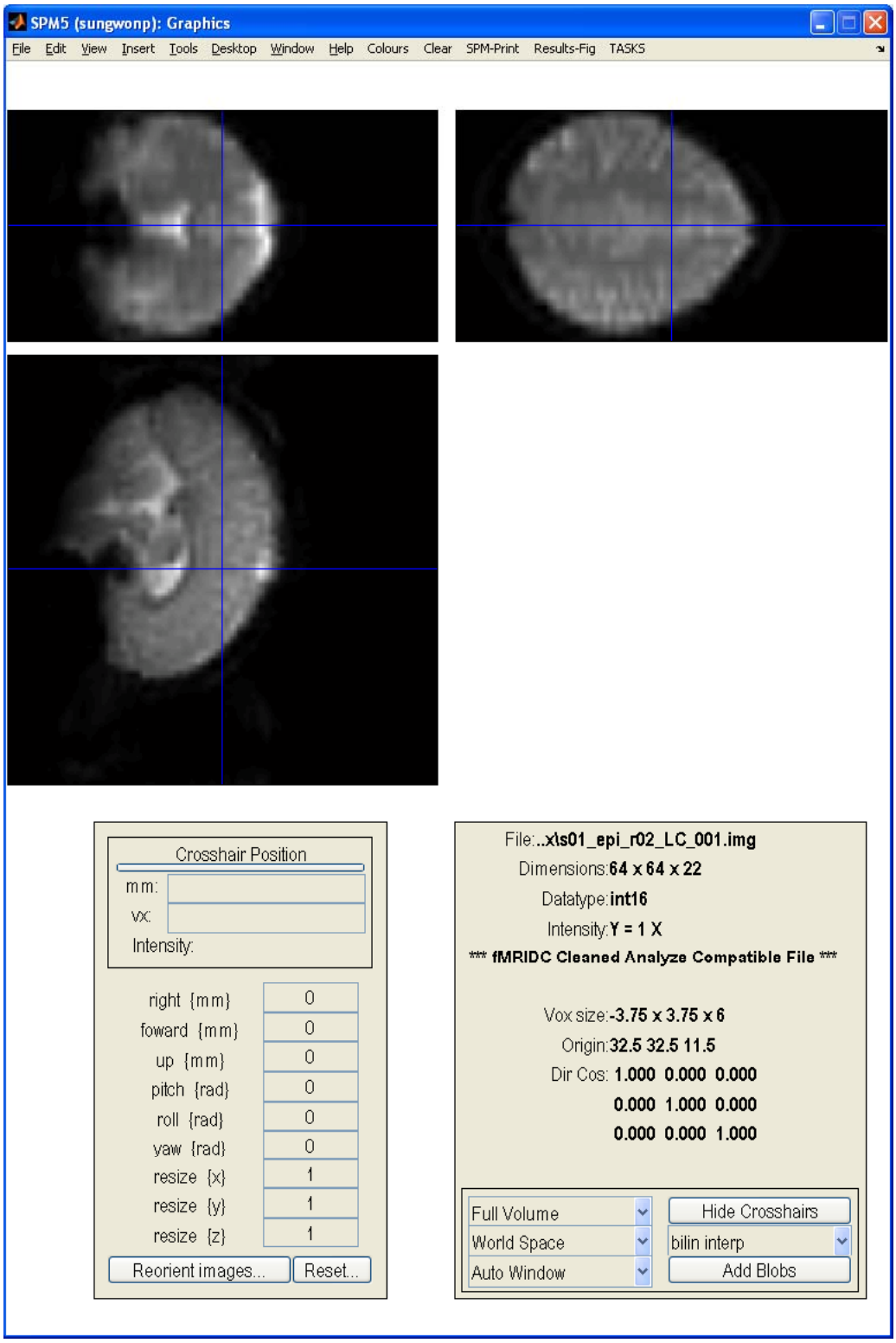


Figure 5: 3D fMRI data in the fMRIDC database

	Average accuracy	F	p-value
PCA	0.846	31.570	1.528E-7
MPCA	0.893		
KPCA using $k_{RBF}(\cdot,\cdot)$	0.853		
MKPCA using $k_{RBF}(\cdot,\cdot)$	0.909		
MKPCA using $k_{3rd}(\cdot,\cdot)$	0.922		

Table 1: Subject classification accuracy

	Average accuracy	F	p-value
PCA	0.890	52.081	2.539E-6
MPCA	0.934		
KPCA with $k_{RBF}(\cdot,\cdot)$	0.920		
MKPCA with $k_{RBF}(\cdot,\cdot)$	0.950		
MKPCA with $k_{3rd}(\cdot,\cdot)$	0.960		

Table 2: Motor-task classification accuracy

KPCA. For the three methods based on parameter decomposition, MPCA, MKPCA using $k_{\text{RBF}}(\cdot,\cdot)$, and MKPCA using $k_{\text{3rd}}(\cdot,\cdot)$, the percentages of cumulative energy in the voxel mode was set to 0.95 again, while the percentages of cumulative energy in the other modes were fixed at 1.0.

Only subject parameters were compared for all the three methods based on parameter decomposition, MPCA, MKPCA using $k_{\text{RBF}}(\cdot,\cdot)$, and MKPCA using $k_{\text{3rd}}(\cdot,\cdot)$, when classifying fMRI images by subject without use of motor-task parameters. Similarly, for classification by motor task, only motor-task parameters obtained by each method were compared without use of subject parameters. On the other hand, PCA and KPCA use a single low-dimensional vector to represent an fMRI image thus, these low-dimensional vectors were compared by the rank-1 recognition rate for classification.

For KPCA, we used the RBF kernel function defined in (28), and set σ for this function to 100:

$$k_{\text{RBF}}(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma}\right). \tag{28}$$

For MKPCA, we tested two different kernels: one is the RBF kernel with $\sigma = 100$ and the other is the kernel $k_{\text{3rd}}(\cdot,\cdot)$ defined in (26). Again, the RBF kernel with $\sigma = 100$ was applied to $k_x(\cdot,\cdot)$, $k_y(\cdot,\cdot)$, and $k_z(\cdot,\cdot)$ in (26).

Tables 1 and 2 show the results of classification by subject and motor task, respectively, using One-Way Analysis of Variance. If we set the significance level to 1%, since the p-values are less than 0.01

in both tables, we can think that the proposed method, MKPCA using $k_{3rd}(\cdot,\cdot)$, obtained statistically significantly higher accuracy than the comparative methods.

As shown in Tables 1 and 2, two versions of MKPCA using $k_{RBF}(\cdot,\cdot)$ and $k_{3rd}(\cdot,\cdot)$ outperformed all of the other methods for classification by both subject and motor task. This may be because MKPCA offers the combined virtues of both multifactor analysis and kernel-based methods. One advantage of MKPCA is MKPCA's ability to achieve subject parameters and motor-task parameters. Since ideally, these two types of parameters are independent from one another, subject and motor-task parameters can be successfully applied to classification by subject and motor task, respectively. Also, MKPCA uses nonlinear kernels, and thus MKPCA is appropriate for complex real-world data samples that are not linearly separable. The advantages of nonlinear methods are demonstrated in Tables 1 and 2 by the fact that KPCA and MKPCA outperformed PCA and MPCA, respectively.

Tables 1 and 2 also demonstrate that MKPCA using $k_{3rd}(\cdot,\cdot)$ outperformed MKPCA using $k_{RBF}(\cdot,\cdot)$. This is because the kernel function $k_{3rd}(\cdot,\cdot)$ sufficiently represents the hierarchical structures of fMRI samples, while the RBF kernel ignores these structures.

5 Conclusion and Future Work

We propose a kernel-based extension of multifactor analysis based on kernels on structured data. We applied our proposed method to fMRI brain scans and analyzed their classification accuracy. Since MKPCA offers the virtues of both multifactor analysis and kernel methods, MKPCA can obtain more reliable experimental results than other baseline methods. Therefore, the main advantage of MKPCA is that it can learn both a multiple factor framework and a nonlinear structure in a data distribution. Moreover, our proposed kernel function for structured data improved the performance of MKPCA; this is because fMRI images are represented by tensorial structures.

Since we do not need to limit the applications of MKPCA only to fMRI data, we will perform more experiments using other structured data with multiple factors, such as 2D face images generated by variation of subject, viewpoint, and illumination.

References

[1] http://www.fil.ion.ucl.ac.uk/spm/.

[2] Gene H. Golub and Charles V. Van Loan. *Matrix Computations*. Johns Hopkins Studies in Mathematical Sciences, third edition, 1996.

[3] Kathleen Y. Haaland, Catherine L. Elsinger, Andrew R. Mayer, Sally Durgerian, and Stephen M. Rao. Motor sequence complexity and performing hand produce differential patterns of hemispheric lateralization. *Journal of Cognitive Neuroscience*, (4):621 – 636, 2004.

[4] David Haussler. Convolutional kernels on discriete structures. In *Technical Report UCSC-CRL-99-10*. University of California at Santa Cruz, 1999.

[5] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, (4):1253 – 1278, 2000.

[6] Yang Li, Yangzhou Du, and Xueyin Lin. Kernel-based multifactor analysis for image synthesis and recognition. In *IEEE International Conference on Computer Vision*, volume 1, pages 114–119, 2005.

[7] Sung Won Park and Marios Savvides. Individual kernel tensor-subspaces for robust face recognition: a computationally efficient tensor framework without requiring mode factorization. *IEEE Transactions on System, Man, and Cybernetics - Part B: Cybernetics*, 37(5):1156–1166, 2007.

[8] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, pages 1299–1319.

[9] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[10] V. Vapnik. In *Statistical Learning Theory*. Wiley, 1998.

[11] M.A.O Vasilescu and Demetri Terzopoulos. Multilinear image analysis for facial recognition. In *International Conference on Pattern Recognition*, volume 1, pages 511–514, 2002.

[12] M.A.O Vasilescu and Demetri Terzopoulos. Multilinear independent components analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 547–553, 2005.

[13] M.A.O Vasilescu and Demetri Terzopoulos. Multilinear projection for appearance-based recognition in the tensor framework. In *ICCV*, pages 1–8, 2007.