Supervised Probabilistic Principal Component Analysis

Shipeng Yu[1,2], Kai Yu[2], Volker Tresp[2], Hans-Peter Kriegel[1], Mingrui Wu[3]
[1]Institute for Computer Science, University of Munich, Germany
[2]Siemens AG, Corporate Technology, Information and Communications, Munich, Germany
[3]MPI for Biological Cybernetics, Tübingen, Germany

spyu@dbs.ifi.lmu.de, kai.yu@siemens.com, volker.tresp@siemens.com, kriegel@dbs.ifi.lmu.de, mingrui.wu@tuebingen.mpg.de

ABSTRACT

Principal component analysis (PCA) has been extensively applied in data mining, pattern recognition and information retrieval for unsupervised dimensionality reduction. When labels of data are available, e.g., in a classification or regression task, PCA is however not able to use this information. The problem is more interesting if only part of the input data are labeled, i.e., in a semi-supervised setting. In this paper we propose a supervised PCA model called SPPCA and a semi-supervised PCA model called S^{2}PPCA, both of which are extensions of a probabilistic PCA model. The proposed models are able to incorporate the label information into the projection phase, and can naturally handle multiple outputs (i.e., in multi-task learning problems). We derive an efficient EM learning algorithm for both models, and also provide theoretical justifications of the model behaviors. SPPCA and S^{2}PPCA are compared with other supervised projection methods on various learning tasks, and show not only promising performance but also good scalability.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

General Terms

Algorithms, Theory, Measurement, Performance

Keywords

Dimensionality reduction, Principal component analysis, Supervised projection, Semi-supervised projection

1. INTRODUCTION

Data mining problems often suffer from the high dimensionality of the data, for the reason of learnability or computational efficiency. Therefore dimensionality reduction, which is also called *feature transformation* or *document indexing*, is of great importance and has been extensively studied (see, e.g., [8, 2]). The most popular method is probably the principal component analysis (PCA), which performs a singular value decomposition (SVD) to the data matrix and obtains the sub-eigenspace with large singular values [10].

Traditional dimensionality reduction methods are *unsupervised*, i.e., they only focus on observations or input data. However, in discriminant analysis where the prediction value or output is available, it would be more helpful to incorporate this information into the mapping and derive a *supervised projection* for input data. Since this projection is designed for the specific prediction problem, it could be substantially different from unsupervised projection. A more interesting setting is *semi-supervised projection* where we have only part of input data labeled, along with a large number of unlabeled data. This is often true in real world problems because labeling is expensive, or unlabeled data are very easy to obtain. An ideal projection method should be able to take into account both the observed labeling information and the unlabeled inputs. There exist many supervised projection algorithms in the literature such as linear discriminative analysis (LDA), partial least squares (PLS) and many others (see, e.g., [8] for an overview). However, these methods cannot incorporate the unlabeled data into the mapping, which will cause problems when we have only very few labeled points.

In this paper we propose a supervised PCA model called SPPCA, which extends the probabilistic PCA model [17] to incorporate label information into the projection. SPPCA takes into account not only the inter-covariance between inputs and outputs, but also the intra-covariance of both. More interestingly, the model can be further extended to model unlabeled data as well, which we call semi-supervised PCA or S^{2}PPCA. This model allows us to elegantly use all the available information to define the mapping. We derive an efficient EM learning algorithm for both models, and provide some theoretical justifications for the model behavior. Experimental results on various learning tasks show promising performance for both SPPCA and S^{2}PPCA models.

This paper is organized as follows. After reviewing previous work in Section 2, we formally introduce SPPCA and S^{2}PPCA models in Section 3 and derive an EM learning algorithm in Section 4. We then presents some theoretical justifications in Section 5 with further discussions. Finally Section 6 illustrates experimental results and Section 7 concludes the paper.

2. PREVIOUS WORK

In this section we review some previous work on unsupervised and supervised projections. In what follows we consider a set of N objects (e.g., images), and each object n is described by an M-dimensional feature vector $\mathbf{x}_n \in \mathcal{X} \subset \mathbb{R}^M$. For dimensionality reduction we aim to derive a mapping $\Psi : \mathcal{X} \mapsto \mathcal{Z}$ which maps the input features into a K-dimensional space ($K < M$).

2.1 Unsupervised Projection

Probably the most popular unsupervised projection is principal component analysis (PCA). Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$ denote the input matrix after centralization, i.e., we subtract the sample mean from each input. In PCA we want to find the principal components which illustrate the directions with maximal variances of the data. Let $\mathbf{X} = \mathbf{VDU}^\top$ be the singular value decomposition (SVD) of $\mathbf{X}$, where $\mathbf{V}$ and $\mathbf{U}$ are $N \times N$ and $M \times M$ column orthogonal matrices, respectively, and $\mathbf{D}$ is $N \times M$ diagonal matrix with singular values sorted in descending order along the diagonal. Then it is known that the first K columns of $\mathbf{U}$, which we denote $\mathbf{U}_K$, defines the mapping Ψ. The projections of $\mathbf{X}$ onto the principal component space are given as $\mathbf{V}_K\mathbf{D}_K$, where $\mathbf{V}_K$ contains the first K columns of $\mathbf{V}$, and $\mathbf{D}_K$ is the top left $K \times K$ sub matrix of $\mathbf{D}$.

Unlike PCA which is maximizing the global covariance of the data, there also exist "local" projection algorithms such as locally linear embedding [15], which tries to preserve the local structure of the data after projecting into a low dimensional space.

2.2 Supervised Projection

When each data point n is associated with not only low level features $\mathbf{x}_n$ but also some outputs $\mathbf{y}_n \in \mathcal{Y} \subset \mathbb{R}^L$ (i.e., classification or regression targets), unsupervised projection such as PCA may be not able to project the data into useful directions (see Figure 2). Therefore many *supervised projection* methods are introduced to make use of the output information. Linear discriminant analysis (LDA) focuses on multi-class classification and finds projection directions that separate the data best (see [4]). The number of projection dimensions is however limited by $L-1$. Partial least squares (PLS) [18] originates from regression analysis and finds the directions of maximal covariance between inputs and outputs sequentially. It however ignores the intra covariance of either inputs or outputs, and its generalization performance on new dimensions of outputs is restricted (see discussions in [6]). Other related works include [9, 12, 7, 1, 19, 20] which consider this problem from different perspectives.

In the situations where we have few labeled points and a large amount of unlabeled data, all these supervised projection methods are however not able to use the unlabeled data. This is often the case in computer vision, information retrieval and bioinformatics, where labeling is expensive and unlabeled data are sufficient and cheap to obtain. We call this setting the *semi-supervised projection*, and in the next section we will propose a model which can deal with semi-supervised projection naturally.

3. THE SPPCA MODEL

In this section we first review a probabilistic model for PCA in Section 3.1, and then present our supervised models.

3.1 Probabilistic PCA (PPCA)

While PCA originates from the analysis of data variances, in statistics community there exists a probabilistic explanation for PCA, which is called *probabilistic PCA* or *PPCA* in the literature [17, 14]. PPCA is a latent variable model and defines a generative process for each object $\mathbf{x}$ as (see Figure 1(a) for an illustration)

$$\mathbf{x} = \mathbf{W}_x\mathbf{z} + \boldsymbol{\mu}_x + \boldsymbol{\epsilon}_x,$$

where $\mathbf{z} \in \mathbb{R}^K$ are called the *latent variables*, and $\mathbf{W}_x$ is a $M \times K$ matrix called *factor loadings*. In this probabilistic model, latent variables $\mathbf{z}$ are conventionally assumed as a Gaussian distribution with zero mean and unit variance, i.e., $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_x$ defines a noise process which also takes an isotropic Gaussian form as $\boldsymbol{\epsilon}_x \sim \mathcal{N}(\mathbf{0}, \sigma_x^2\mathbf{I})$, with σ_x^2 the *noise level*. Additionally, we have parameters $\boldsymbol{\mu}_x \in \mathbb{R}^M$ which allow non-zero means for the data.

It is shown that PPCA has strong connections to PCA. In particular when $\sigma_x^2 \to 0$, the projections of data $\mathbf{x}$ onto the K-dimensional principal subspace in PCA are *identical* to the latent variables $\mathbf{z}$ up to a rotation and scaling factor [17]. We summarize the related results in the following proposition without proof, since this is simply a corollary of Theorem 2 in Section 5.

Proposition 1. *Let $\mathbf{S}_x = \frac{1}{N}\sum_{n=1}^N(\mathbf{x}_n - \boldsymbol{\mu}_x)(\mathbf{x}_n - \boldsymbol{\mu}_x)^\top$ be the sample covariance matrix for data $\{\mathbf{x}_n\}_{n=1}^N$, and $\lambda_1 \geq \ldots \geq \lambda_M$ be its eigenvalues with eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_M$, then if the latent space in PPCA model is K-dimensional,*

(i) *The maximum likelihood estimate of $\mathbf{W}_x$ is given as*

$$\mathbf{W}_x = \mathbf{U}_K(\boldsymbol{\Lambda}_K - \sigma_x^2\mathbf{I})^{\frac{1}{2}}\mathbf{R},$$

where $\boldsymbol{\Lambda}_K = \mathrm{diag}(\lambda_1, \ldots, \lambda_K)$, $\mathbf{U}_K = [\mathbf{u}_1, \ldots, \mathbf{u}_K]$, and $\mathbf{R}$ is an arbitrary $K \times K$ orthogonal matrix.

(ii) *The mean projections $\mathbf{z}^*$ for new input $\mathbf{x}^*$ is given as*

$$\mathbf{z}^* = \mathbf{R}^\top \left(\boldsymbol{\Lambda}_K - \sigma_x^2\mathbf{I}\right)^{\frac{1}{2}} \boldsymbol{\Lambda}_K^{-1}\mathbf{U}_K^\top(\mathbf{x}^* - \boldsymbol{\mu}_x).$$

As a probabilistic framework, PPCA provides additional benefits over PCA such as a fast EM learning procedure, a principled way of handling missing entries, and a possibility of considering mixture of PCA models. PPCA is also closely related to *factor analysis* models, but the modeling perspectives are different (see [17] for more discussions).

3.2 Supervised PPCA (SPPCA)

The key point of PPCA model is that all the M dimensions of $\mathbf{x}$ are *conditionally independent* given the latent variables $\mathbf{z}$, due to the isotropic property of the noise process. This indicates that the principal components in PPCA are *the K latent variables which best explain the data covariance.*

When supervised information is available, each object $\mathbf{x}$ is associated with an output value $y \in \mathcal{Y}$, e.g., $y \in \mathbb{R}$ for regression task and $y \in \{+1, -1\}$ for classification task. In general we believe there are *covariances* between input space $\mathcal{X}$ and output space $\mathcal{Y}$ (since otherwise the supervised learning task is not learnable), and it is reasonable to extend PPCA to model this covariance as well. Furthermore, when there are more than one learning tasks (i.e., in a multi-task learning setting [5]), the *covariances between different tasks* can also be modeled by latent variables.

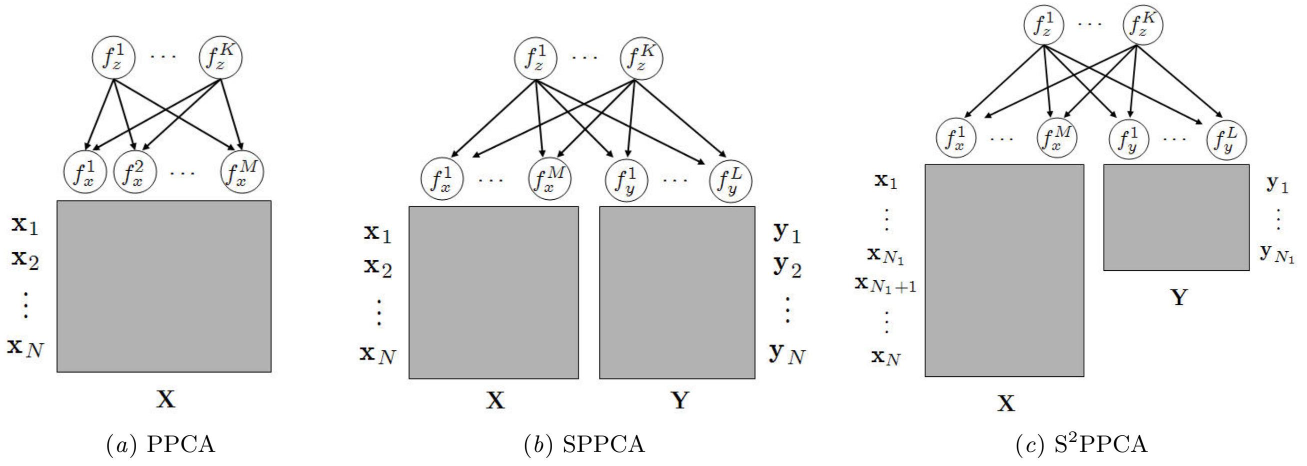
f_z^1 $\cdots$ f_z^K
f_x^1 f_x^2 $\cdots$ f_x^M
$\mathbf{x}_1$ $\mathbf{x}_2$ $\vdots$ $\mathbf{x}_N$
$\mathbf{X}$
(a) PPCA
f_x^1 $\cdots$ f_x^M f_y^1 $\cdots$ f_y^L
$\mathbf{y}_1$ $\mathbf{y}_2$ $\vdots$ $\mathbf{y}_N$
$\mathbf{Y}$
(b) SPPCA
$\mathbf{x}_{N_1}$ $\mathbf{x}_{N_1+1}$ $\mathbf{y}_{N_1}$
(c) S^{2}PPCA

Figure 1: Illustrations of the three models PPCA, SPPCA and S^{2}PPCA. $\mathbf{X}$ and $\mathbf{Y}$ denote respectively the input and output matrices, where each row is one data point. $f_x^1, \ldots, f_x^M$ are the M input features, and $f_y^1, \ldots, f_y^L$ are the L outputs. On the top $f_z^1, \ldots, f_z^K$ are the K latent variables in each model. They are all in circles because they are variables in the probabilistic models. The arrows denote probabilistic dependency.

We now formally describe our proposed model family which we call the *supervised probabilistic principal component analysis* (SPPCA). Let the number of outputs be L, and each object $\mathbf{x}$ be associated with an output vector $\mathbf{y} = [y_1, \ldots, y_L]^\top \in \mathcal{Y} \subset \mathbb{R}^L$. In SPPCA the observed data $(\mathbf{x}, \mathbf{y})$ is generated from a latent variable model as

$$\mathbf{x} = \mathbf{W}_x \mathbf{z} + \boldsymbol{\mu}_x + \boldsymbol{\epsilon}_x,$$
$$\mathbf{y} = \mathbf{f}(\mathbf{z}, \boldsymbol{\Theta}) + \boldsymbol{\epsilon}_y,$$

where $\mathbf{f}(\mathbf{z}, \boldsymbol{\Theta}) = [f_1(\mathbf{z}, \theta_1), \ldots, f_L(\mathbf{z}, \theta_L)]^\top$ encode the values of L deterministic functions $f_1, \ldots, f_L$ with parameters $\boldsymbol{\Theta} = \{\theta_1, \ldots, \theta_L\}$. Here $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ are the latent variables *shared by both inputs* $\mathbf{x}$ *and outputs* $\mathbf{y}$, and the two noise models are independent to each other and both defined as isotropic Gaussians: $\boldsymbol{\epsilon}_x \sim \mathcal{N}(\mathbf{0}, \sigma_x^2 \mathbf{I}), \boldsymbol{\epsilon}_y \sim \mathcal{N}(\mathbf{0}, \sigma_y^2 \mathbf{I})$. We use two noise levels σ_x^2 and σ_y^2 for inputs and outputs, respectively, and it is also straightforward to define different noise levels for different outputs if desired. See Figure 1(b) for an illustration of the model.

In SPPCA model we keep the nice property of *conditional independence*, i.e., all the input and output dimensions are conditionally independent to each other given the latent variables. If we integrate out the latent variables $\mathbf{z}$, the likelihood of observation $(\mathbf{x}, \mathbf{y})$ is obtained as

$$P(\mathbf{x}, \mathbf{y}) = \int P(\mathbf{x}, \mathbf{y}|\mathbf{z}) P(\mathbf{z})\, d\mathbf{z} = \int P(\mathbf{x}|\mathbf{z}) P(\mathbf{y}|\mathbf{z}) P(\mathbf{z})\, d\mathbf{z},$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and from the latent variable model,

$$\mathbf{x}|\mathbf{z} \sim \mathcal{N}(\mathbf{W}_x \mathbf{z} + \boldsymbol{\mu}_x, \sigma_x^2 \mathbf{I}), \quad \mathbf{y}|\mathbf{z} \sim \mathcal{N}(\mathbf{f}(\mathbf{z}, \boldsymbol{\Theta}), \sigma_y^2 \mathbf{I}). \quad (1)$$

After observing N pairs, the likelihood of all the observations $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$, with i.i.d. assumption, is simply $P(\mathcal{D}) = \prod_{n=1}^N P(\mathbf{x}_n, \mathbf{y}_n)$.

In the following we consider the simplest model in this family, i.e., we assume each function f_ℓ, $\ell = 1, \ldots, L$, is linear in $\mathbf{z}$:

$$f_\ell(\mathbf{z}, \theta_\ell) = {\mathbf{w}_y^\ell}^\top \mathbf{z} + \mu_y^\ell,$$

where the parameters $\theta_\ell = \{\mathbf{w}_y^\ell, \mu_y^\ell\}$ include the linear coefficients and intercept. Then we can group all the f_ℓ's and write

$$\mathbf{f}(\mathbf{z}, \boldsymbol{\Theta}) = \mathbf{W}_y \mathbf{z} + \boldsymbol{\mu}_y,$$

a similar form as the generative model for $\mathbf{x}$ where $\mathbf{W}_y = [\mathbf{w}_y^1, \ldots, \mathbf{w}_y^L]^\top$ and $\boldsymbol{\mu}_y = [\mu_y^1, \ldots, \mu_y^L]^\top$. The reason why we choose this form for $\mathbf{f}$ is that the EM learning is simple (see the next section), and we have closed form solution (see Section 5). We will discuss other forms of $\mathbf{f}$ in Section 5.3 which may need special approximation techniques.

Let us denote

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_x \\ \mathbf{W}_y \end{pmatrix}, \qquad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{pmatrix}, \qquad \boldsymbol{\Phi} = \begin{pmatrix} \sigma_x^2 \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \sigma_y^2 \mathbf{I} \end{pmatrix},$$

then based on the model assumption, it is easily seen that $(\mathbf{x}, \mathbf{y})$ are jointly Gaussian distributed, with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Phi} + \mathbf{W}\mathbf{W}^\top$. All the parameters for the SPPCA model are $\boldsymbol{\Omega} := \{\mathbf{W}_x, \mathbf{W}_y, \boldsymbol{\mu}_x, \boldsymbol{\mu}_y, \sigma_x^2, \sigma_y^2\}$.

3.3 Semi-Supervised PPCA (S²PPCA)

In SPPCA model, we assume we observe both the inputs $\mathbf{x}$ and outputs $\mathbf{y}$ for every data point. In many real world problems, however, we may only observe the outputs for a small portion of data, and have many *unlabeled data* in which only inputs $\mathbf{x}$ are known. This may be because some measurements are unobservable, the labeling cost is too high, or simply we have a large amount of unlabeled data available. Learning in this situation is in general called *semi-supervised learning*. For learning a projection, an ideal model would incorporate both the unlabeled inputs and the partially labeled outputs to define the mapping.

This can be easily done under the SPPCA framework. Let the number of labeled and unlabeled data points be N_1 and N_2, respectively, with $N = N_1 + N_2$. The whole observation is now $\mathcal{D} = \mathcal{D}_1 \bigcup \mathcal{D}_2 = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N_1} \bigcup \{\mathbf{x}_{n'}\}_{n'=N_1+1}^{N}$. The likelihood, with the independence assumption of all the data points, is calculated as

$$P(\mathcal{D}) = P(\mathcal{D}_1) P(\mathcal{D}_2) = \prod_{n=1}^{N_1} P(\mathbf{x}_n, \mathbf{y}_n) \prod_{n'=N_1+1}^{N} P(\mathbf{x}_{n'}),$$

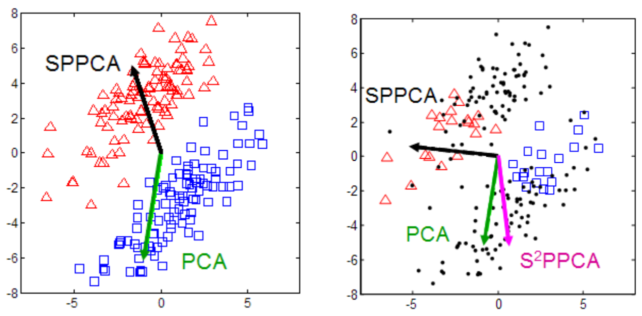

Figure 2: Projection directions for a toy data. They are fully labeled on the left, and only partially labeled on the right.

where $P(\mathbf{x}_n, \mathbf{y}_n)$ is calculated as in SPPCA model, and $P(\mathbf{x}_{n'}) = \int P(\mathbf{x}_{n'}|\mathbf{z}_{n'})P(\mathbf{z}_{n'})\,d\mathbf{z}_{n'}$. Due to its applicability to semi-supervised projection, we call it *semi-supervised PPCA* or S^{2}PPCA in this paper. Figure 1(c) illustrates this model.

Under the additional assumptions that all the f_ℓ's are linear, it can be easily checked that all the likelihood terms in this product are Gaussians. This makes the model easy to learn. Other forms of $\mathbf{f}$ will be discussed in Section 5.3.

When $N_2 = 0$, S^{2}PPCA degrades to SPPCA which is purely supervised. This means one can view SPPCA as a special case of S^{2}PPCA model with no unlabeled data. From the perspective of probabilistic modeling, S^{2}PPCA can also be viewed as an SPPCA model where all the $\mathbf{y}$'s for the N_2 unlabeled points are missing. Due to this close relationship, in the following we use SPPCA to denote both models unless clearly specified.

3.4 Projections in SPPCA Models

Analogous to the PPCA model, in SPPCA models the projection of data point $\mathbf{x}$ is directly given in the latent variables $\mathbf{z}$. If we know all the parameters $\mathbf{\Omega}$, calculating this projection is simply an *inference* problem. To do this we can apply Bayes' rule and calculate the *a posteriori* distribution of $\mathbf{z}$. Therefore we can obtain not only the mean projection vector, but also the *uncertainty* of the projection.

3.4.1 Projection for Fully Observed Data

When both inputs $\mathbf{x}$ and outputs $\mathbf{y}$ are observed, we can calculate the *a posteriori* distribution of $\mathbf{z}$ given $(\mathbf{x}, \mathbf{y})$ as

$$P(\mathbf{z}|\mathbf{x},\mathbf{y}) \propto P(\mathbf{x},\mathbf{y}|\mathbf{z})P(\mathbf{z}) = P(\mathbf{x}|\mathbf{z})P(\mathbf{y}|\mathbf{z})P(\mathbf{z}). \quad (2)$$

Since all the three terms on the right hand side are Gaussians, this distribution is also a Gaussian $\mathcal{N}(\boldsymbol{\mu}_\mathbf{z}, \mathbf{\Sigma}_\mathbf{z})$, with

$$\mu_\mathbf{z} = \mathbf{A}^{-1}\left[\frac{1}{\sigma_x^2}\mathbf{W}_x^\top(\mathbf{x}-\boldsymbol{\mu}_x) + \frac{1}{\sigma_y^2}\mathbf{W}_y^\top(\mathbf{y}-\boldsymbol{\mu}_y)\right], \quad \mathbf{\Sigma}_\mathbf{z} = \mathbf{A}^{-1},$$

where $\mathbf{A}$ is a $K \times K$ matrix defined as

$$\mathbf{A} = \frac{1}{\sigma_x^2}\mathbf{W}_x^\top\mathbf{W}_x + \frac{1}{\sigma_y^2}\mathbf{W}_y^\top\mathbf{W}_y + \mathbf{I}. \quad (3)$$

This means that the projection is $\boldsymbol{\mu}_\mathbf{z}$ with uncertainty $\mathbf{\Sigma}_\mathbf{z}$.

3.4.2 Projection for Pure Input Data

For a test data $\mathbf{x}^*$ that has no output information, what are the most likely latent variables $\mathbf{z}^*$? This can also be done using Bayes' rule

$$P(\mathbf{z}^*|\mathbf{x}^*) \propto P(\mathbf{x}^*|\mathbf{z}^*)P(\mathbf{z}^*). \quad (4)$$

This turns out again to be a Gaussian $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}|\mathbf{x}}, \mathbf{\Sigma}_{\mathbf{z}|\mathbf{x}})$, with

$$\boldsymbol{\mu}_{\mathbf{z}|\mathbf{x}} = (\mathbf{W}_x^\top\mathbf{W}_x + \sigma_x^2\mathbf{I})^{-1}\mathbf{W}_x^\top(\mathbf{x}^* - \boldsymbol{\mu}_x),$$
$$\mathbf{\Sigma}_{\mathbf{z}|\mathbf{x}} = \sigma_x^2(\mathbf{W}_x^\top\mathbf{W}_x + \sigma_x^2\mathbf{I})^{-1}.$$

This result looks similar as that in PPCA model, but the projection is now *supervised* because the learning of $\mathbf{W}_x$ is influenced by those observed outputs. This is clarified in the next section and will be theoretically justified in Section 5.

4. LEARNING IN SPPCA MODELS

Learning in probabilistic models reduces to maximizing the data (log) likelihood with respect to all the model parameters. In case of SPPCA model, the log likelihood of the whole observation $\mathcal{D}$ is

$$\mathcal{L} = \sum_{n=1}^N \log \int P(\mathbf{x}_n|\mathbf{z}_n)P(\mathbf{y}_n|\mathbf{z}_n)P(\mathbf{z}_n)\,d\mathbf{z}_n.$$

For SPPCA analytical solution exists, and we summarize it later in Theorem 2. For S^{2}PPCA model, however, there is no analytical solution since all the outputs for the unlabeled data are missing. Fortunately we can derive an EM algorithm [3] which is applicable to both models.

The EM algorithm iterates the two steps *expectation* (E-step) and *maximization* (M-step) until convergence, and it is guaranteed to find a local minima of the data likelihood. In the E-step, we fix the model parameters ($\mathbf{\Omega}$ for SPPCA models) and calculate the expected distributions of latent variables (all the $\mathbf{z}_n$'s for SPPCA models), and in the M-step we fix this distribution and maximize the *complete data likelihood* with respect to the model parameters. As will be discussed later, EM learning for SPPCA models is important because it can deal with very large data sets, and it has, in particular for SPPCA model with no unlabeled points, no local minima problem up to a rotation factor (see Section 5). For simplicity we only outline the update equations in the following and omit details (see [17] for a similar derivation).

4.1 EM Learning for SPPCA

We first consider the SPPCA model without unlabeled data. In the E-step, for each data point n we estimate the distribution of $\mathbf{z}_n$ given observation $(\mathbf{x}_n, \mathbf{y}_n)$. This is done using (2), and we calculate the sufficient statistics as

$$\langle\mathbf{z}_n\rangle = \mathbf{A}^{-1}\left[\frac{1}{\sigma_x^2}\mathbf{W}_x^\top(\mathbf{x}_n-\boldsymbol{\mu}_x) + \frac{1}{\sigma_y^2}\mathbf{W}_y^\top(\mathbf{y}_n-\boldsymbol{\mu}_y)\right], \quad (5)$$

$$\langle\mathbf{z}_n\mathbf{z}_n^\top\rangle = \mathbf{A}^{-1} + \langle\mathbf{z}_n\rangle\langle\mathbf{z}_n\rangle^\top, \quad (6)$$

where $\langle\cdot\rangle$ denotes the expectation with respect to the posterior distribution $P(\mathbf{z}_n|\mathbf{x}_n,\mathbf{y}_n)$ given in (2).

In the M-step, we maximize the complete log-likelihood

$$\tilde{\mathcal{L}} = \sum_{n=1}^N \int P(\mathbf{z}_n|\mathbf{x}_n,\mathbf{y}_n)\log\Big(P(\mathbf{x}_n|\mathbf{z}_n)P(\mathbf{y}_n|\mathbf{z}_n)P(\mathbf{z}_n)\Big)\,d\mathbf{z}_n$$

with respect to the model parameters, holding $P(\mathbf{z}_n|\mathbf{x}_n,\mathbf{y}_n)$ fixed from the E-step. This can be done by setting the partial derivatives with respect to each parameter to be zero.

Algorithm 1 Learning in SPPCA Model - Primal Form

```
Require: $N$ data points $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$ with inputs $\mathbf{x}_n \in \mathbb{R}^M$ and outputs $\mathbf{y}_n \in \mathbb{R}^L$. A desired dimension $K < M$.
 1: Calculate the sample means (7) and center the data via $\mathbf{x}_n \Leftarrow \mathbf{x}_n - \boldsymbol{\mu}_x$, $\mathbf{y}_n \Leftarrow \mathbf{y}_n - \boldsymbol{\mu}_y$.
 2: Initialize model parameters $\boldsymbol{\Omega}$ randomly.
 3: repeat
 4:   {E-step}
 5:   for $n = 1$ to $N$ do
 6:     Calculate sufficient statistics (5) and (6);
 7:   end for
 8:   {M-step}
 9:   Update $\mathbf{W}_x$ and $\mathbf{W}_y$ via (8);
10:   Update $\sigma_x^2$ and $\sigma_y^2$ via (9) and (10);
11: until the change of $\boldsymbol{\Omega}$ is smaller than a threshold.
Output: Parameters $\boldsymbol{\Omega}$ and projection vectors $\{\mathbf{z}_n\}_{n=1}^N$ which are obtained from E-step. For test data $\mathbf{x}^*$, the mean projection $\mathbf{z}^* = (\mathbf{W}_x^\top \mathbf{W}_x + \sigma_x^2 \mathbf{I})^{-1} \mathbf{W}_x^\top (\mathbf{x}^* - \boldsymbol{\mu}_x)$.
```

For means of $\mathbf{x}$ and $\mathbf{y}$ we have

$$\tilde{\boldsymbol{\mu}}_x = \frac{1}{N}\sum_{n=1}^N \mathbf{x}_n, \qquad \tilde{\boldsymbol{\mu}}_y = \frac{1}{N}\sum_{n=1}^N \mathbf{y}_n, \tag{7}$$

which are just the sample means. Since they are always the same in all EM iterations, we can *center* the data by subtracting these means in the beginning and ignore these parameters in the learning process. So for simplicity we change the notations $\mathbf{x}_n$ and $\mathbf{y}_n$ to be the *centered vectors* in the following.

The mapping matrices $\mathbf{W}_x$ and $\mathbf{W}_y$ are updated as

$$\widetilde{\mathbf{W}}_x = \mathbf{X}^\top \mathbf{Z} \mathbf{C}^{-1}, \qquad \widetilde{\mathbf{W}}_y = \mathbf{Y}^\top \mathbf{Z} \mathbf{C}^{-1}, \tag{8}$$

where for clarity we use matrix notations $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$, $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^\top$ and $\mathbf{Z} = [\langle \mathbf{z}_1 \rangle, \ldots, \langle \mathbf{z}_N \rangle]^\top$. Matrix $\mathbf{C}$ is defined to be the sum of all second-order sufficient statistics of the data, i.e., $\mathbf{C} = \sum_{n=1}^N \langle \mathbf{z}_n \mathbf{z}_n^\top \rangle$. Finally the noise levels are updated as

$$\tilde{\sigma}_x^2 = \frac{1}{MN}\left[\sum_{n=1}^N \|\mathbf{x}_n\|^2 + \mathrm{tr}(\widetilde{\mathbf{W}}_x^\top \widetilde{\mathbf{W}}_x \mathbf{C}) - 2\,\mathrm{tr}(\mathbf{X}\widetilde{\mathbf{W}}_x \mathbf{Z}^\top)\right] \tag{9}$$

$$\tilde{\sigma}_y^2 = \frac{1}{LN}\left[\sum_{n=1}^N \|\mathbf{y}_n\|^2 + \mathrm{tr}(\widetilde{\mathbf{W}}_y^\top \widetilde{\mathbf{W}}_y \mathbf{C}) - 2\,\mathrm{tr}(\mathbf{Y}\widetilde{\mathbf{W}}_y \mathbf{Z}^\top)\right] \tag{10}$$

where $\|\cdot\|$ denotes vector 2-norm, and $\mathrm{tr}(\cdot)$ denotes matrix trace. The whole algorithm is summarized in Algorithm 1.

4.2 EM Learning for S^{2}PPCA

The log likelihood of the observations in S^{2}PPCA model is a sum of two parts: $\mathcal{L}_1 = \log P(\mathcal{D}_1)$ which contains all the labeled points, and $\mathcal{L}_2 = \log P(\mathcal{D}_2)$ which includes all unlabeled points. Therefore in E-step we need to deal with them differently. For a labeled points $(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{D}_1$, the latent variables $\mathbf{z}_n$ are estimated via (5) and (6), the same as in SPPCA model. For an unlabeled point $\mathbf{x}_{n'} \in \mathcal{D}_2$, the distribution of $\mathbf{z}_{n'}$ is only conditioned on input $\mathbf{x}_{n'}$, which can be calculated via (4), with sufficient statistics (the data are assumed centered already):

$$\langle \mathbf{z}_{n'} \rangle = (\mathbf{W}_x^\top \mathbf{W}_x + \sigma_x^2 \mathbf{I})^{-1} \mathbf{W}_x^\top \mathbf{x}_{n'}, \tag{11}$$

$$\langle \mathbf{z}_{n'} \mathbf{z}_{n'}^\top \rangle = (\mathbf{W}_x^\top \mathbf{W}_x + \sigma_x^2 \mathbf{I})^{-1} + \langle \mathbf{z}_{n'} \rangle \langle \mathbf{z}_{n'} \rangle^\top, \tag{12}$$

Algorithm 2 Learning in S^{2}PPCA Model - Primal Form

```
Require: $N_1$ labeled data points $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N_1}$ and $N_2$ unlabeled points $\{\mathbf{x}_{n'}\}_{n'=N_1+1}^N$, with inputs $\mathbf{x} \in \mathbb{R}^M$ and observed outputs $\mathbf{y} \in \mathbb{R}^L$. A desired dimension $K < M$.
 1: Calculate the sample means (7) and center the data via $\mathbf{x}_n \Leftarrow \mathbf{x}_n - \boldsymbol{\mu}_x$, $\mathbf{y}_n \Leftarrow \mathbf{y}_n - \boldsymbol{\mu}_y$, $\mathbf{x}_{n'} \Leftarrow \mathbf{x}_{n'} - \boldsymbol{\mu}_x$.
 2: Initialize model parameters $\boldsymbol{\Omega}$ randomly.
 3: repeat
 4:   {E-step}
 5:   for $n = 1$ to $N_1$ do
 6:     Calculate (5) and (6) for labeled data $n$;
 7:   end for
 8:   for $n' = N_1 + 1$ to $N$ do
 9:     Calculate (11) and (12) for unlabeled data $n'$;
10:   end for
11:   {M-step}
12:   Update $\mathbf{W}_x$ and $\mathbf{W}_y$ via (13) and (14);
13:   Update $\sigma_x^2$ and $\sigma_y^2$ via (15) and (16);
14: until the change of $\boldsymbol{\Omega}$ is smaller than a threshold.
Output: Parameters $\boldsymbol{\Omega}$ and projection vectors $\{\mathbf{z}_n\}_{n=1}^N$ which are obtained from E-step. For test data $\mathbf{x}^*$, the mean projection $\mathbf{z}^* = (\mathbf{W}_x^\top \mathbf{W}_x + \sigma_x^2 \mathbf{I})^{-1} \mathbf{W}_x^\top (\mathbf{x}^* - \boldsymbol{\mu}_x)$.
```

where here $\langle \cdot \rangle$ denotes the expectation with respect to the posterior distribution $P(\mathbf{z}_{n'}|\mathbf{x}_{n'})$ given in (4).

The M-step is similarly obtained by setting the partial derivatives of the complete log likelihood with respect to each parameter to zero. For the two mapping matrices, we have the updates

$$\widetilde{\mathbf{W}}_x = (\mathbf{X}_1^\top \mathbf{Z}_1 + \mathbf{X}_2^\top \mathbf{Z}_2)(\mathbf{C}_1 + \mathbf{C}_2)^{-1}, \tag{13}$$

$$\widetilde{\mathbf{W}}_y = \mathbf{Y}^\top \mathbf{Z}_1 \mathbf{C}_1^{-1}, \tag{14}$$

where $\mathbf{X}_1$, $\mathbf{Z}_1$, $\mathbf{C}_1$ are defined for labeled data, i.e., $\mathbf{X}_1 = [\mathbf{x}_1, \ldots, \mathbf{x}_{N_1}]^\top$, $\mathbf{Z}_1 = [\langle \mathbf{z}_1 \rangle, \ldots, \langle \mathbf{z}_{N_1} \rangle]^\top$, $\mathbf{C}_1 = \sum_{n=1}^{N_1} \langle \mathbf{z}_n \mathbf{z}_n^\top \rangle$, and $\mathbf{X}_2$, $\mathbf{Z}_2$, $\mathbf{C}_2$ are similarly defined for unlabeled data. It is seen that the update for $\mathbf{W}_x$ depends on both labeled data and unlabeled data, while $\mathbf{W}_y$ only depends on the labeled data. Updates for the noise levels are similar to those in SPPCA model, except that for σ_x^2 we need to consider both labeled data and unlabeled data:

$$\tilde{\sigma}_x^2 = \frac{1}{MN}\left[\sum_{n=1}^N \|\mathbf{x}_n\|^2 + \mathrm{tr}\left(\widetilde{\mathbf{W}}_x^\top \widetilde{\mathbf{W}}_x (\mathbf{C}_1 + \mathbf{C}_2)\right) \right.$$
$$\left. -2\,\mathrm{tr}\left(\widetilde{\mathbf{W}}_x (\mathbf{Z}_1^\top \mathbf{X}_1 + \mathbf{Z}_2^\top \mathbf{X}_2)\right)\right], \tag{15}$$

$$\tilde{\sigma}_y^2 = \frac{1}{LN_1}\left[\sum_{n=1}^{N_1} \|\mathbf{y}_n\|^2 + \mathrm{tr}(\widetilde{\mathbf{W}}_y^\top \widetilde{\mathbf{W}}_y \mathbf{C}_1) - 2\,\mathrm{tr}(\mathbf{Y}\widetilde{\mathbf{W}}_y \mathbf{Z}_1^\top)\right]. \tag{16}$$

The whole algorithm is summarized in Algorithm 2. When $N_2 = 0$, i.e., we have no unlabeled data, the learning algorithm reduces to SPPCA learning.

4.3 EM Learning in Dual Form

It is known that when the number of data points is less than the number of features, i.e., $N < M$, it is more efficient to consider the *dual* solution for PCA in which we perform SVD to the *Gram matrix* $\mathbf{K} = \mathbf{X}\mathbf{X}^\top$. The canonical PCA is sometimes called the *primal* solution. For SPPCA we have a similar dual solution, and it can be directly derived from the

Algorithm 3 Learning in S^{2}PPCA Model - Dual Form

```
Require: $N_1$ labeled data points $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N_1}$ and $N_2$ unlabeled points $\{\mathbf{x}_{n'}\}_{n'=N_1+1}^{N}$, with inputs $\mathbf{x} \in \mathbb{R}^M$ and observed outputs $\mathbf{y} \in \mathbb{R}^L$. A desired dimension $K < M$.
1: Calculate Gram matrix $\mathbf{K}$ with $\mathbf{K}_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$ and center it using (27). Center the outputs via $\mathbf{y}_n \Leftarrow \mathbf{y}_n - \boldsymbol{\mu}_y$.
2: Initialize $\mathbf{Z}$, $\mathbf{C}$ and model parameters $\boldsymbol{\Omega}$ randomly.
3: repeat {The EM-step}
4:     Calculate $\mathbf{Z}_1$ and $\mathbf{C}_1$ using (21) and (22);
5:     Calculate $\mathbf{Z}_2$ and $\mathbf{C}_2$ using (23) and (24);
6:     Update $\sigma_x^2$ and $\sigma_y^2$ via (25) and (26);
7: until the change of $\boldsymbol{\Omega}$ is smaller than a threshold.
Output: Parameters $\boldsymbol{\Omega}$ and projection vectors $\{\mathbf{z}_n\}_{n=1}^N$. The mean projection $\mathbf{z}^*$ for test data $\mathbf{x}^*$ is $\mathbf{z}^* = \mathbf{C}\left(\mathbf{Z}^\top \mathbf{K}\mathbf{Z} + \sigma_x^2 \mathbf{C}^2\right)^{-1} \mathbf{Z}^\top \mathbf{k}(\mathbf{X}, \mathbf{x}^*)$ where $\mathbf{k}(\mathbf{X}, \mathbf{x}^*) = [\mathbf{x}_1^\top \mathbf{x}^*, \ldots, \mathbf{x}_N^\top \mathbf{x}^*]^\top$ and is centered via (28).
```

EM learning in previous subsections. To avoid the tedious mathematics in the main text, we put the derivation details into Appendix and summarize the algorithm in Algorithm 3. Since SPPCA can be viewed as a special case of S^{2}PPCA, here we only give the algorithm for S^{2}PPCA model.

One important observation in the dual solution is that all the calculation involving input data $\mathbf{X}$ can be done via inner-product, e.g., in the Gram matrix $\mathbf{K}$ we have $\mathbf{K}_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$. This motivates us to consider *non-linear PCA* where we first map the data into a new feature space (via, e.g., basis functions), and then perform PCA in that space with a proper definition of inner-product. This is the idea behind kernel PCA [16], and we put detailed discussion into Appendix.

4.4 Computational Issues

In the primal form (i.e., Algorithm 1 and 2), the time complexity for both algorithms is $\mathcal{O}\big(m(M+L)NK\big)$, with m the number of iterations.[1] It is linear in the number of data points N and the input dimension M. The space complexity is $\mathcal{O}\big((M+L)N\big)$, which is also linear in both N and M. The projection for a test data point is just a linear operation and costs $\mathcal{O}(MK)$ time.

In the dual form, the time complexity is $\mathcal{O}(mN^2K)$ plus $\mathcal{O}(N^2M)$ which is the one-time calculation of Gram matrix, and the space complexity is $\mathcal{O}\big(N^2\big)$. Both of them are now quadratic in the number of data points N. The time for projecting a test data point is now $\mathcal{O}(NM)$. Similar to the case for PCA, in situations where $M > N$, i.e., we have more features than the number of data points, the dual form is more efficient than the primal form.

5. THEORETICAL JUSTIFICATION

In this section we provide some theoretical analysis for SPPCA model and show how the supervised information influences the projection. The proofs are given in Appendix.

5.1 Primal Form Solution

Recall that matrix $\boldsymbol{\Phi}$ is a $(M+L)\times(M+L)$ diagonal matrix with all the noise levels in diagonal, i.e., $\boldsymbol{\Phi} = \mathrm{diag}(\sigma_x^2, \ldots, \sigma_x^2, \sigma_y^2, \ldots, \sigma_y^2)$. For SPPCA model we obtain the following analytical solutions for mapping matrix $\mathbf{W}_x$ and $\mathbf{W}_y$. This makes it easier to compare SPPCA with related models such as PCA.

[1]Note that we only need to calculate the diagonal entries for matrix trace in the updates for noise levels.

Theorem 2. *Let $\mathbf{S}$ denote the normalized sample covariance matrix for centered observations $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$, i.e.,*

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^N \boldsymbol{\Phi}^{-\frac{1}{2}} \begin{pmatrix}\mathbf{x}_n\\ \mathbf{y}_n\end{pmatrix}\begin{pmatrix}\mathbf{x}_n\\ \mathbf{y}_n\end{pmatrix}^\top \boldsymbol{\Phi}^{-\frac{1}{2}} = \begin{pmatrix} \frac{1}{\sigma_x^2}\mathbf{S}_x & \frac{1}{\sigma_x\sigma_y}\mathbf{S}_{xy} \\ \frac{1}{\sigma_x\sigma_y}\mathbf{S}_{yx} & \frac{1}{\sigma_y^2}\mathbf{S}_y \end{pmatrix},$$

and $\lambda_1 \ge \ldots \ge \lambda_{(M+L)}$ be its eigenvalues with eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_{(M+L)}$, then if the latent space in SPPCA model is K-dimensional, the following results hold:

(i) *The maximum likelihood estimates of $\mathbf{W}_x$ and $\mathbf{W}_y$ are*

$$\mathbf{W}_x = \sigma_x \mathbf{U}_x(\boldsymbol{\Lambda}_K - \mathbf{I})^{\frac{1}{2}}\mathbf{R}, \tag{17}$$

$$\mathbf{W}_y = \sigma_y \mathbf{U}_y(\boldsymbol{\Lambda}_K - \mathbf{I})^{\frac{1}{2}}\mathbf{R}, \tag{18}$$

where $\boldsymbol{\Lambda}_K = \mathrm{diag}(\lambda_1, \ldots, \lambda_K)$, $\mathbf{U}_x$ ($\mathbf{U}_y$) contains the first M (last L) rows of $[\mathbf{u}_1, \ldots, \mathbf{u}_K]$, and $\mathbf{R}$ is an arbitrary $K\times K$ orthogonal rotation matrix.

(ii) *Projection $\mathbf{z}^*$ for centered new input $\mathbf{x}^*$ is given as*

$$\mathbf{z}^* = \frac{1}{\sigma_x}\mathbf{R}^\top(\boldsymbol{\Lambda}_K - \mathbf{I})^{-\frac{1}{2}}\left[\mathbf{U}_x^\top \mathbf{U}_x + (\boldsymbol{\Lambda}_K - \mathbf{I})^{-1}\right]^{-1}\mathbf{U}_x^\top \mathbf{x}^*.$$

In the special case that $L = 0$, the model is unsupervised and $\mathbf{S} = \frac{1}{\sigma_x^2}\mathbf{S}_x$ holds. Then (17) degrades to $\sigma_x \mathbf{U}_x(\boldsymbol{\Lambda}_K - \mathbf{I})^{\frac{1}{2}}\mathbf{R}$, which recovers the PPCA solution. $\mathbf{U}_x$ is seen to be column orthogonal in this case, and the mapping $\mathbf{z}^*$ of $\mathbf{x}^*$ is (scaled) standard PCA mapping when $\sigma_x^2 \to 0$ and $\mathbf{R} = \mathbf{I}$. This proves Proposition 1 which is a corollary of this theorem.

When $L > 0$, SPPCA solutions explain not only the sample covariance of inputs $\mathbf{S}_x$, but also the *intra-covariance* of outputs $\mathbf{S}_y$ (if $L > 1$) and the *inter-correlations* between inputs and outputs, $\mathbf{S}_{xy}$ and $\mathbf{S}_{yx}$. Therefore, one column of $\mathbf{W}_x$ is the direction that best explains the whole system from the perspective of inputs, and thus are *biased* by the outputs. Unlike the case of PCA, the learned $\mathbf{W}_x$ in SPPCA needs not to be column orthogonal. This means we are only learning an *affine* mapping for $\mathbf{x}$. If necessary, it is straightforward to find the orthogonal basis by performing SVD to matrix $\mathbf{W}_x$.

In both SPPCA and PPCA the learned $\mathbf{W}_x$ has an arbitrary rotation factor $\mathbf{R}$. This means the mapping is invariant under a rotation of latent space, as can be seen from the equation for $\mathbf{z}^*$. Therefore the SPPCA model can only find the *latent principal subspace*, which has been mentioned in [17] for PPCA. Thus the EM algorithm in Section 4 can find different mappings with different initializations, but they define the same subspace and do not change the structure of projected data. If necessary, this ambiguity can be removed by eigen-decomposing

$$\mathbf{W}_x^\top \mathbf{W}_x + \mathbf{W}_y^\top \mathbf{W}_y = \mathbf{R}^\top(\boldsymbol{\Lambda}_K - \mathbf{I})\mathbf{R}$$

and uncovering the rotation factor $\mathbf{R}$.

A final comment is that Theorem 2 may be not applicable to large-scale problems since we have to form a big square matrix of size $M+L$. This is however not necessary for the EM algorithm.

5.2 Dual Form Solution

In the dual form, we do not obtain the mapping matrix $\mathbf{W}_x$, but the projected vectors directly. The following theorem gives the analytical solution in the dual form.

Theorem 3. *Let $\widehat{\mathbf{K}} = \frac{1}{\sigma_x^2}\mathbf{K} + \frac{1}{\sigma_y^2}\mathbf{Y}\mathbf{Y}^\top$, and $\lambda_1 \geq \ldots \geq \lambda_N$ be its eigenvalues with eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_N$, then if the latent space in SPPCA model is K-dimensional, the following results hold:*[2]

(i) *The projection vectors of training data, which are encoded in rows of matrix $\mathbf{Z}$, are calculated as*

$$\mathbf{Z} = \sqrt{N}\mathbf{V}_K\mathbf{D}^{\frac{1}{2}}\mathbf{R}, \tag{19}$$

where $\mathbf{D} := \mathbf{I} - N\mathbf{\Lambda}_K^{-1}$, $\mathbf{\Lambda}_K = \mathrm{diag}(\lambda_1, \ldots, \lambda_K)$, $\mathbf{V}_K = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$, and $\mathbf{R}$ is an arbitrary $K \times K$ orthogonal rotation matrix.

(ii) *Projection $\mathbf{z}^*$ for new input $\mathbf{x}^*$ is given as*

$$\mathbf{z}^* = \sqrt{N}\mathbf{R}^\top\mathbf{D}^{-\frac{1}{2}}\left(\mathbf{V}_K^\top\mathbf{K}\mathbf{V}_K + \mathbf{D}\right)^{-1}\mathbf{V}_K^\top\mathbf{k}(\mathbf{X}, \mathbf{x}^*),$$

with $\mathbf{k}(\mathbf{X}, \mathbf{x}^*)$ centered via (28).

It is seen from this theorem that when there is no output in SPPCA, i.e., $\widehat{\mathbf{K}} = \frac{1}{\sigma_x^2}\mathbf{K}$, SPPCA reduces to the dual form of PCA as desired. This theorem directly applies for non-linear mappings if the inner-product is defined in a reproducing kernel Hilbert space (RKHS) [16], and leads to the kernel PCA solution when $L = 0$.

Theorem 3 presents a nice explanation for SPPCA model: we just use the outputs to modify the Gram matrix of input data, and control the trade-off via the ratio of noise levels. The model complexity remains the same (i.e., quadratic in N) no matter how many output dimensions we have. Our previous work [20] shares this same property and derives the supervised projection via an eigenvalue problem. But it cannot be elegantly extended to semi-supervised projections, and has problems to deal with large-scale data sets.

5.3 Discussions

Previous two subsections give some theoretical results for SPPCA model. There exists however no such a closed-form solution for S^{2}PPCA. One can only empirically analyze the behavior of this model.

In the EM learning algorithm we are learning the maximum likelihood (ML) estimates for the two mapping matrices $\mathbf{W}_x$ and $\mathbf{W}_y$. In the probabilistic framework we can also assign a prior to them to reduce overfitting. For instance, we can assign an isotropic Gaussian prior for each column of $\mathbf{W}_x$, and if we consider the *maximum a posteriori* (MAP) estimate this prior corresponds to a smooth term in the update equations. For simplicity we do not consider this prior here.

6. EXPERIMENTS

In this section we empirically investigate the performance of SPPCA models. The supervised tasks here are multi-class classification and multi-label classification. Our basic setting is that we train a supervised projection model using the input features and label information, and then test the classification performance for test data using the projected features. Since the test data are assumed known in the training phase, for S^{2}PPCA we will be able to use these unlabeled data to train the mapping.

[2]We use the same notation for eigenvalues as in Theorem 2 because it can be proved that they are identical up to a scaling factor of N.

Table 1: Statistics of the multi-class data sets (top) and multi-label data sets (bottom)

	Category	# Data	# Dim	# Class
Yale	Face	165	1024	15
ORL	Face	400	1024	40
PIE	Face	11554	1024	68
YaleB	Face	2414	1024	38
11_Tumors	Gene	174	12533	11
14_Tumors	Gene	308	15009	26
Lung_Cancer	Gene	203	12600	5
20Newsgroup	Text	19928	25284	20
TDT2	Text	8692	35452	20

	Category	# Data	# Dim	# Class
Yeast	Gene	2417	103	14
RCV1	Text	23149	15500	103

6.1 Data Sets

We test the proposed model on 9 multi-class and 2 multi-label classification problems. These problems include face recognition, gene classification and text categorization. Some statistics of these data sets are shown in Table 1.

For face recognition we use four data sets Yale, ORL, PIE and YaleB (the extended Yale Face Database B).[3] The Yale data set contains 165 gray-scale images in GIF format of 15 individuals. There are 11 images per subject, one per different facial expression or configuration such as center-light, left-light, happy or surprised. The ORL database contains 10 different images of each of 40 distinct subjects. For some subjects, the images were taken at different times with varying lighting and facial details. The PIE databases we use contains 170 images for each of 68 people. These images are the five near frontal poses under different illuminations and expressions. For YaleB we have 38 individuals and around 64 near frontal images under different illuminations per individual. All the face images are manually aligned, cropped and resized to 32×32 pixels. We then normalize each image to have Euclidean distance 1.

We consider three gene expression data sets 11_Tumors, 14_Tumors and Lung_Cancer for gene classification.[4] The 11_Tumors describes 11 various human tumor types, and 14_Tumors describes 14 tumor types with 12 normal tissue types. For Lung_Cancer we need to classify 4 lung cancer types and normal tissues. The characteristic of these data is that the number of data points is small, but the input dimensionality is very high.

The two textual data sets we use are taken from 20Newsgroup and TDT2. 20Newsgroup contains 20,000 news articles posted in 20 news groups. We remove the words that occur less than 5 times, and obtain 19,928 documents with 25,284 words. The TDT2 corpus we use consists of the documents collected during the first half of 1998 and taken from 6 sources, including 2 newswires (APW, NYT), 2 radio programs (VOA, PRI) and 2 television programs (CNN, ABC). It consists of 11,021 documents which are classified into 96 semantic categories. In our experiments, we keep the largest 20 categories and remove those documents that are assigned to more than one categories. This leaves us 8,692 documents with totally 35,452 words. For both of these data sets we

[3]See `http://www.ews.uiuc.edu/~dengcai2/Data/data.html`.
[4]They are available at `http://www.gems-system.org`.

Table 2: Results for Multi-class Classification Tasks. Bold face indicates lowest error rate. Symbols $\star$ indicate that the best method is significantly better than the competitors (p-value 0.01 in Wilcoxon rank sum test).

Task	Full	PCA	LDA	PLS	SPPCA	S^{2}PPCA
Yale	0.5656 ± 0.0394	0.6690 ± 0.0333	$\mathbf{0.6133 \pm 0.0471}$	0.6440 ± 0.0383	0.7007 ± 0.0402	0.7121 ± 0.0393
ORL	0.3308 ± 0.0347	0.5593 ± 0.0263	0.5302 ± 0.0444	0.5505 ± 0.0294	0.5459 ± 0.0305	$\mathbf{0.5287 \pm 0.0286}$
PIE	0.6988 ± 0.0085	0.9325 ± 0.0032	$\mathbf{0.7066 \pm 0.0177}$	0.8781 ± 0.0058	0.8780 ± 0.0116	0.8452 ± 0.0037
YaleB	0.6360 ± 0.0160	0.9895 ± 0.0023	$\star\mathbf{0.5328 \pm 0.0251}$	0.9546 ± 0.0066	0.9701 ± 0.0088	0.9800 ± 0.0034
11_Tumors	0.3161 ± 0.0566	0.5409 ± 0.0490	$\mathbf{0.4505 \pm 0.0755}$	N/A	0.5226 ± 0.0636	0.5130 ± 0.0491
14_Tumors	0.6084 ± 0.0360	0.7363 ± 0.0286	0.7161 ± 0.0481	N/A	0.7312 ± 0.0371	$\mathbf{0.7138 \pm 0.0296}$
Lung_Cancer	0.3680 ± 0.1148	0.3768 ± 0.0939	$\mathbf{0.3225 \pm 0.1658}$	N/A	0.4287 ± 0.1338	0.3896 ± 0.0923
20Newsgroup	0.6135 ± 0.0155	0.9070 ± 0.0177	0.9140 ± 0.0208	N/A	$\mathbf{0.9030 \pm 0.0162}$	0.9126 ± 0.0116
TDT2	0.1875 ± 0.0233	0.6664 ± 0.0657	0.7834 ± 0.0782	N/A	0.6236 ± 0.0739	$\star\mathbf{0.3686 \pm 0.0349}$

(a) Projection dimension $K = 5$.

Task	Full	PCA	LDA	PLS	SPPCA	S^{2}PPCA
Yale	0.5656 ± 0.0394	0.5993 ± 0.0312	$\mathbf{0.5279 \pm 0.0460}$	0.5698 ± 0.0386	0.6101 ± 0.0447	0.5916 ± 0.0433
ORL	0.3308 ± 0.0347	0.4049 ± 0.0293	0.3625 ± 0.0468	0.4048 ± 0.0349	0.3832 ± 0.0409	$\mathbf{0.3509 \pm 0.0287}$
PIE	0.6988 ± 0.0085	0.8573 ± 0.0051	$\mathbf{0.5496 \pm 0.0185}$	0.8062 ± 0.0068	0.7105 ± 0.0161	0.6942 ± 0.0047
YaleB	0.6360 ± 0.0160	0.9308 ± 0.0046	$\star\mathbf{0.3846 \pm 0.0282}$	0.8762 ± 0.0108	0.7976 ± 0.0242	0.7986 ± 0.0117
11_Tumors	0.3161 ± 0.0566	0.3682 ± 0.0655	0.3926 ± 0.0667	N/A	0.3801 ± 0.0624	$\star\mathbf{0.3297 \pm 0.0664}$
14_Tumors	0.6084 ± 0.0360	0.6868 ± 0.0288	0.6212 ± 0.0430	N/A	0.6322 ± 0.0363	$\mathbf{0.6120 \pm 0.0331}$
Lung_Cancer	0.3680 ± 0.1148	0.3493 ± 0.0996	$\mathbf{0.3225 \pm 0.1658}$	N/A	0.6235 ± 0.1520	0.3517 ± 0.1063
20Newsgroup	0.6135 ± 0.0155	0.9039 ± 0.0172	0.8943 ± 0.0292	N/A	0.8931 ± 0.0242	$\mathbf{0.8548 \pm 0.0138}$
TDT2	0.1875 ± 0.0233	0.5531 ± 0.0742	0.6878 ± 0.1068	N/A	0.5346 ± 0.0885	$\star\mathbf{0.2794 \pm 0.0327}$

(b) Projection dimension $K = 10$.

Task	Full	PCA	LDA	PLS	SPPCA	S^{2}PPCA
Yale	0.5656 ± 0.0394	0.5437 ± 0.0414	0.5793 ± 0.0438	0.5216 ± 0.0435	0.5093 ± 0.0391	$\mathbf{0.5001 \pm 0.0589}$
ORL	0.3308 ± 0.0347	0.3323 ± 0.0310	0.2944 ± 0.0398	0.3366 ± 0.0331	0.3271 ± 0.0372	$\mathbf{0.2755 \pm 0.0286}$
PIE	0.6988 ± 0.0085	0.7999 ± 0.0060	$\mathbf{0.4352 \pm 0.0186}$	0.7454 ± 0.0092	0.5912 ± 0.0146	0.5361 ± 0.0090
YaleB	0.6360 ± 0.0160	0.8304 ± 0.0096	$\star\mathbf{0.3004 \pm 0.0227}$	0.7695 ± 0.0148	0.5619 ± 0.0276	0.5652 ± 0.0172
11_Tumors	0.3161 ± 0.0566	0.3267 ± 0.0635	0.3926 ± 0.0667	N/A	0.4470 ± 0.0691	$\mathbf{0.3012 \pm 0.0582}$
14_Tumors	0.6084 ± 0.0360	0.6379 ± 0.0360	0.5822 ± 0.0388	N/A	$\mathbf{0.5669 \pm 0.0347}$	0.5674 ± 0.0372
Lung_Cancer	0.3680 ± 0.1148	0.3584 ± 0.0953	$\mathbf{0.3225 \pm 0.1658}$	N/A	0.6487 ± 0.1540	0.4092 ± 0.1107
20Newsgroup	0.6135 ± 0.0155	0.9160 ± 0.0220	0.8001 ± 0.0425	N/A	$\mathbf{0.6254 \pm 0.0420}$	0.6568 ± 0.0146
TDT2	0.1875 ± 0.0233	0.4582 ± 0.1441	0.1524 ± 0.0622	N/A	0.1566 ± 0.0509	$\mathbf{0.1520 \pm 0.0210}$

(c) Projection dimension $K = 20$.

use TF-IDF features and normalize each document to have Euclidean distance 1.

For multi-label classification we use Yeast and RCV1. The Yeast data set is formed by micro-array expression data and phylogenetic profiles with 2,417 genes in total and 103 input dimensions. There are 14 groups and each gene can belong to multiple groups. The other data is a subset of the RCV1-v2 text data set, provided by Reuters and corrected by Lewis et al. [11]. We use the training set provided by Lewis, which contains 103 labels, 23,149 documents with 15,500 words after we remove words that occur less than 5 times. We also extract TF-IDF features and normalize each document to have Euclidean distance 1.

6.2 Experimental Setting

For the multi-class classification tasks, we randomly pick up a small number of labeled data points for training (2 for those data sets with less than 500 data points, and 5 for the others), and test the classification error rate on the unlabeled data. We will in general compare the following six algorithms if applicable:

- PCA: Unsupervised projection. Note that we use both the training and test data to derive the mapping.
- LDA: Linear discriminant analysis.
- PLS: Partial least squares.
- SPPCA: Supervised probabilistic PCA.
- S^{2}PPCA: Semi-supervised probabilistic PCA. We allow S^{2}PPCA to use the test data to train the mapping.
- Full: All the features are used without projection.

For all the projection methods, we project the data into a space of 5, 10 and 20 dimensions, and train a nearest-neighbor classifier for the test points using new features with Euclidean distance. For Full we directly train the nearest-neighbor classifier using original features. For PLS, SPPCA and S^{2}PPCA, we translate the one column output to the "One of C" setting, i.e., each class has one column with binary labels.

For multi-label classification, we pick up 5 positive examples from each label to obtain the training data. For all projection methods we project to 5, 10 and 20 dimensions, and then train a linear SVM classifier for each label. The comparison metrics are F1-Macro, F1-Micro and AUC (Area Under ROC Curve) score. The candidate algorithms are almost the same as multi-class setting, except LDA which is not applicable to this task. The C in SVM is fixed as 100, and from our experience it is not sensible for all algorithms.

In all these comparisons, the iteration number for SPPCA and S^{2}PPCA is set to 1000. Both the noise levels σ_x^2 and σ_y^2 are set to 10^{-5} initially. It turns out that PLS gets memory problems when applied to large dimensions. We repeat each experiments 50 times independently,[5] and the results are illustrated in .

6.3 Analysis of Results

The first observation is that in most cases the supervised

[5]For the four face recognition tasks we use the split versions available from the web site.

Table 3: Results for Multi-label Classification Tasks. Bold face indicates best performance.

K	Model	Yeast			RCV1		
		F1-Macro	F1-Micro	AUC	F1-Macro	F1-Micro	AUC
5	Full	0.3813 ± 0.0102	0.5161 ± 0.0154	0.5571 ± 0.0094	0.2796 ± 0.0055	0.5053 ± 0.0095	0.6030 ± 0.0063
	PCA	0.2318 ± 0.0354	0.5600 ± 0.0220	0.5279 ± 0.0108	0.0467 ± 0.0063	0.3540 ± 0.0106	0.5208 ± 0.0072
	PLS	0.3432 ± 0.0231	0.5795 ± 0.0233	0.5556 ± 0.0094	N/A	N/A	N/A
	SPPCA	0.3823 ± 0.0120	0.5332 ± 0.0188	0.5641 ± 0.0087	0.1155 ± 0.0071	0.4433 ± 0.0089	0.5568 ± 0.0079
	S^2PPCA	$\mathbf{0.3927 \pm 0.0134}$	$\mathbf{0.5890 \pm 0.0126}$	$\mathbf{0.5842 \pm 0.0104}$	$\mathbf{0.1312 \pm 0.0118}$	$\mathbf{0.4620 \pm 0.0236}$	$\mathbf{0.5762 \pm 0.0162}$
10	Full	0.3813 ± 0.0102	0.5161 ± 0.0154	0.5571 ± 0.0094	0.2796 ± 0.0055	0.5053 ± 0.0095	0.6030 ± 0.0063
	PCA	0.3113 ± 0.0304	$\mathbf{0.5916 \pm 0.0146}$	0.5493 ± 0.0101	0.0843 ± 0.0124	0.4003 ± 0.0162	0.5347 ± 0.0072
	PLS	0.3756 ± 0.0154	0.5517 ± 0.0177	0.5610 ± 0.0095	N/A	N/A	N/A
	SPPCA	0.3924 ± 0.0117	0.5459 ± 0.0180	0.5685 ± 0.0084	0.1797 ± 0.0112	0.4474 ± 0.0124	0.5872 ± 0.0087
	S^2PPCA	$\mathbf{0.3985 \pm 0.0103}$	0.5914 ± 0.0106	$\mathbf{0.5896 \pm 0.0107}$	$\mathbf{0.1956 \pm 0.0110}$	$\mathbf{0.4735 \pm 0.0198}$	$\mathbf{0.6012 \pm 0.0098}$
20	Full	0.3813 ± 0.0102	0.5161 ± 0.0154	0.5571 ± 0.0094	0.2796 ± 0.0055	0.5053 ± 0.0095	0.6030 ± 0.0063
	PCA	0.3723 ± 0.0171	0.5537 ± 0.0204	0.5614 ± 0.0097	0.1320 ± 0.0086	0.4419 ± 0.0095	0.5504 ± 0.0061
	PLS	0.3799 ± 0.0123	0.5208 ± 0.0158	0.5585 ± 0.0102	N/A	N/A	N/A
	SPPCA	0.3859 ± 0.0133	0.5517 ± 0.0151	0.5640 ± 0.0097	0.2297 ± 0.0119	0.4690 ± 0.0126	0.6044 ± 0.0054
	S^2PPCA	$\mathbf{0.3976 \pm 0.0142}$	$\mathbf{0.6012 \pm 0.0190}$	$\mathbf{0.5921 \pm 0.0119}$	$\mathbf{0.2536 \pm 0.0117}$	$\mathbf{0.4921 \pm 0.0102}$	$\mathbf{0.6090 \pm 0.0076}$

PCA model is better than unsupervised PCA model. This means by using the output information, we are able to derive a more meaningful projection for the supervised tasks. When the dimensionality is larger (e.g., 20), SPPCA and S^2PPCA obtain the best results for most of the tasks.

When we compare SPPCA model with other supervised projection methods, SPPCA is consistently better than PLS, but in some tasks worse than LDA (e.g., for YaleB). The reasons may be that SPPCA models are still based on the PCA assumptions for input features, so the mapping is strongly biased by PCA. When PCA projection directions are almost useless for classification, like for YaleB dataset, the SPPCA projections are also not informative enough. In this case discriminative methods like LDA can often do a good job. In other situations where PCA does help, SPPCA can in general be better than pure discriminative methods.

When we compare SPPCA and S^2PPCA, in most cases S^2PPCA gets better results. For some tasks the difference is very big (e.g., for TDT2). This indicates that by incorporating the unlabeled data we can learn a better mapping. But S^2PPCA is in general slower than SPPCA because it has to consider all the test data in the training phase. In this case 1000 iterations may be not enough to get the algorithm converge. This may also be part of the reason why S^2PPCA is inferior to other methods for some tasks like PIE.

Most of the supervised projection methods can get a better performance than Full even if they only project the data into a very low dimensional space. This is important because we can not only speed up the system, but also improve the performance. PCA in our experiments uses the input features of both labeled and unlabeled data, thus it sometimes can get better results than Full method (e.g., for Yale and Lung_Cancer).

The results for multi-label classification show that S^2PPCA is consistently better than other methods. S^2PPCA also shows very good scalability in our experiments, since for 20Newsgroup and RCV1 it need to handle 20,000 documents with more than 15,000 features. Most of the other algorithms fail on these large data sets.

7. CONCLUSION

We proposed a supervised and a semi-supervised PCA in this paper, and derived an efficient EM algorithm for model learning. Empirical results show that the proposed model obtains good performance and scales well for large data sets.

In this paper we mainly focus on the Gaussian noise model for the outputs $\mathbf{y}$. One can define other likelihood models for specific tasks, e.g., the *probit* likelihood for classification, but then we lose the nice closed-form solutions as described in Theorem 2 and 3. This is because in E-step of the EM learning the *a posteriori* distribution of $\mathbf{z}$ is no longer a Gaussian (see (2)). To solve this problem we can apply the expectation-propagation (EP) [13] algorithm to sequentially approximate each likelihood term $P(y_\ell|\mathbf{z})$ as a Gaussian for $\mathbf{z}$. Then the approximated *a posteriori* distribution of $\mathbf{z}$ is still a Gaussian, and the EM algorithm can still be applied to find the optimal projection matrices. Empirically comparing this algorithm with the basic ones would be part of the future work.

8. REFERENCES

[1] E. Bair, T. Hastie, D. Paul, and R. Tibshirani. Prediction by supervised principal components. Technical report, 2004.

[2] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2000.

[5] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of SIGKDD*, 2004.

[6] K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5(Jan):73–99, 2004.

[7] T. Hastie and R. Tibshirani. Discriminant analysis by Gaussian mixtures. *Journal of the Royal Statistical Society series B*, 58:158–176, 1996.

[8] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Verlag, 2001.

[9] H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:321–377, 1936.

[10] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2002.

[11] D. D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[12] R. G. Miller. *Beyond Anova: Basics of Applied Statistics*. John Wiley, 1986.

[13] T. P. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[14] S. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computaion*, 11(2):305–345, 1999.

[15] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[17] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statisitical Scoiety*, B(61):611–622, 1999.

[18] H. Wold. Soft modeling by latent variables; the nonlinear iterative partial least squares approach. *Perspectives in Probability and Statistics, Papers in Honour of M.S. Bartlett*, 1975.

[19] J. Ye, R. Janardan, Q. Li, and H. Park. Feature extraction via generalized uncorrelated linear discriminant analysis. In *Proceedings of ICML*, 2004.

[20] K. Yu, S. Yu, and T. Volker. Multi-label informed latent semantic indexing. In *Proceedings of 27th Annual International ACM SIGIR Conference*, 2005.

APPENDIX

A. DERIVATION OF DUAL FORM

We only focus on S^{2}PPCA model here. Let $\mathbf{C} = \mathbf{C}_1 + \mathbf{C}_2$, and

$$\mathbf{X} = \begin{pmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{pmatrix},\ \mathbf{Z} = \begin{pmatrix}\mathbf{Z}_1\\ \mathbf{Z}_2\end{pmatrix},\ \mathbf{K} = \mathbf{X}\mathbf{X}^\top = \begin{pmatrix}\mathbf{X}_1\mathbf{X}^\top\\ \mathbf{X}_2\mathbf{X}^\top\end{pmatrix} = \begin{pmatrix}\mathbf{K}_1\\ \mathbf{K}_2\end{pmatrix}$$

then (13) can be written as $\widetilde{\mathbf{W}}_x = \mathbf{X}^\top\mathbf{Z}\mathbf{C}^{-1}$. This leads to

$$\widetilde{\mathbf{W}}_x^\top\widetilde{\mathbf{W}}_x = \mathbf{C}^{-1}\mathbf{Z}^\top\mathbf{K}\mathbf{Z}\mathbf{C}^{-1},\ \widetilde{\mathbf{W}}_x^\top\mathbf{x} = \mathbf{C}^{-1}\mathbf{Z}^\top\mathbf{k}(\mathbf{X},\mathbf{x}),\quad (20)$$

which are the building blocks for the dual form. The matrix $\mathbf{K} = \mathbf{X}\mathbf{X}^\top$ has each entry the inner-product of data points of corresponding row and column, $\mathbf{K}_{ij} = \mathbf{x}_i^\top\mathbf{x}_j$, and $\mathbf{k}(\mathbf{X},\mathbf{x}) = \mathbf{X}\mathbf{x}$ is a N-dimensional column vector with the i-th entry $\mathbf{x}_i^\top\mathbf{x}$.

In the E-step, we first rewrite $\mathbf{A}$ as

$$\widetilde{\mathbf{A}} = \frac{1}{\sigma_x^2}\mathbf{C}^{-1}\mathbf{Z}^\top\mathbf{K}\mathbf{Z}\mathbf{C}^{-1} + \frac{1}{\sigma_y^2}\mathbf{C}_1^{-1}\mathbf{Z}_1^\top\mathbf{Y}\mathbf{Y}^\top\mathbf{Z}_1\mathbf{C}_1^{-1} + \mathbf{I}.$$

Applying (20) in sufficient statistics (5), we get

$$\widetilde{\mathbf{Z}}_1 = \left[\frac{1}{\sigma_x^2}\mathbf{K}_1\mathbf{Z}\mathbf{C}^{-1} + \frac{1}{\sigma_y^2}\mathbf{Y}\mathbf{Y}^\top\mathbf{Z}_1\mathbf{C}_1^{-1}\right]\widetilde{\mathbf{A}}^{-1},\quad (21)$$

by collecting $\langle\mathbf{z}_n\rangle$ in columns and transposing it. Sufficient statistics (6) can be written in terms of $\mathbf{C}_1$:

$$\widetilde{\mathbf{C}}_1 = \sum_{n=1}^{N_1}\langle\mathbf{z}_n\mathbf{z}_n^\top\rangle = N_1\widetilde{\mathbf{A}}^{-1} + \widetilde{\mathbf{Z}}_1^\top\widetilde{\mathbf{Z}}_1.\quad (22)$$

Similarly, we obtain the following two updates for unlabeled data:

$$\widetilde{\mathbf{Z}}_2 = \frac{1}{\sigma_x^2}\mathbf{K}_2\mathbf{Z}\mathbf{C}^{-1}\left[\frac{1}{\sigma_x^2}\mathbf{C}^{-1}\mathbf{Z}^\top\mathbf{K}\mathbf{Z}\mathbf{C}^{-1} + \mathbf{I}\right]^{-1},\quad (23)$$

$$\widetilde{\mathbf{C}}_2 = N_2\left[\frac{1}{\sigma_x^2}\mathbf{C}^{-1}\mathbf{Z}^\top\mathbf{K}\mathbf{Z}\mathbf{C}^{-1} + \mathbf{I}\right]^{-1} + \widetilde{\mathbf{Z}}_2^\top\widetilde{\mathbf{Z}}_2.\quad (24)$$

In M-step, we only need to update variances σ_x^2 and σ_y^2 as

$$\tilde{\sigma}_x^2 = \frac{1}{MN}\left[\mathrm{tr}(\mathbf{K}) - \mathrm{tr}(\widetilde{\mathbf{Z}}^\top\mathbf{K}\widetilde{\mathbf{Z}}\widetilde{\mathbf{C}}^{-1})\right],\quad (25)$$

$$\tilde{\sigma}_y^2 = \frac{1}{LN}\left[\mathrm{tr}(\mathbf{Y}\mathbf{Y}^\top) - \mathrm{tr}(\widetilde{\mathbf{Z}}^\top\mathbf{Y}\mathbf{Y}^\top\widetilde{\mathbf{Z}}\widetilde{\mathbf{C}}^{-1})\right],\quad (26)$$

which can be easily verified from (9) and (10).

Therefore, it is seen that all interesting terms in the EM algorithm take input data into account *only* via the inner-product. This nice property allows us to extend the linear SPPCA model to non-linear mappings by first mapping the input data into a new feature space (via, e.g., basis functions) and then performing SPPCA in that space. This can also be done via *kernel trick* for which we only need to define a *kernel function* $\kappa(\cdot,\cdot)$ for each pair of input data [16].

Since we are now working on *centered* data in the feature space, we can achieve this by modifying $\mathbf{K}$ as

$$\widetilde{\mathbf{K}} = \mathbf{K} - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\mathbf{K} - \frac{1}{N}\mathbf{K}\mathbf{1}\mathbf{1}^\top + \frac{1}{N^2}\mathbf{1}\mathbf{1}^\top\mathbf{K}\mathbf{1}\mathbf{1}^\top,\quad (27)$$

where $\mathbf{1}$ denotes the all one column vector of length N [16]. For kernel vector $\mathbf{k}(\mathbf{X},\mathbf{x}^*)$, it can also be centered by

$$\tilde{\mathbf{k}} = \mathbf{k} - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\mathbf{k} - \frac{1}{N}\mathbf{K}\mathbf{1} + \frac{1}{N^2}\mathbf{1}\mathbf{1}^\top\mathbf{K}\mathbf{1}.\quad (28)$$

B. PROOF OF THEOREM 2

We give a sketch here. The mapping matrices are obtained by finding a fixed point in the EM algorithm in Section 4. For this proof we use notation $\mathbf{W} := (\frac{1}{\sigma_x}\mathbf{W}_x^\top, \frac{1}{\sigma_y}\mathbf{W}_y^\top)^\top$, and (3) can be rewritten as $\mathbf{A} = \mathbf{W}^\top\mathbf{W} + \mathbf{I}$. Plugging this and (5), (6) into (8) yields an update equation only related to $\mathbf{W}$: $\widetilde{\mathbf{W}} = \mathbf{S}\mathbf{W}\left[\mathbf{W}^\top\mathbf{S}\mathbf{W} + \mathbf{W}^\top\mathbf{W} + \mathbf{I}\right]^{-1}\left(\mathbf{W}^\top\mathbf{W} + \mathbf{I}\right)$. At the fixed point, this simplifies to $\mathbf{S}\mathbf{W} = \mathbf{W}\mathbf{W}^\top\mathbf{W} + \mathbf{W}$. Let $\mathbf{W} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ be the SVD of $\mathbf{W}$. Then each column $\mathbf{u}$ of $\mathbf{U}$ satisfies $d\mathbf{S}\mathbf{u} = (d + d^3)\mathbf{u}$, with d the corresponding singular value. Therefore solving an eigenvalue problem for $\mathbf{S}$ gives the mapping matrix $\mathbf{W}$, and plugging them into (4) gives the mapping $\mathbf{z}^*$ for $\mathbf{x}^*$. $\square$

C. PROOF OF THEOREM 3

We give a sketch here. We define $\mathbf{B} = \mathbf{Z}\mathbf{C}^{-1}$ and rewrite (21) and (22) using only $\mathbf{B}$. This leads to $N\mathbf{B}\left(\mathbf{I} + \mathbf{B}^\top\widehat{\mathbf{K}}\mathbf{B}\right) = \widehat{\mathbf{K}}\mathbf{B}$. To solve $\mathbf{B}$ we denote the SVD of $\mathbf{B}^\top\widehat{\mathbf{K}}\mathbf{B}$ as $\mathbf{Q}\mathbf{D}\mathbf{Q}^\top$, then we can obtain $\mathbf{B}^\top\mathbf{B} = \frac{1}{N}\mathbf{Q}\mathbf{D}\left(\mathbf{I} + \mathbf{D}\right)^{-1}\mathbf{Q}^\top$ after some mathematics. Then define $\mathbf{U} := \mathbf{B}\mathbf{Q}\sqrt{N}\mathbf{D}^{-1/2}\left(\mathbf{I} + \mathbf{D}\right)^{1/2}$, we have $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\mathbf{U}^\top\widehat{\mathbf{K}}\mathbf{U} = N\left(\mathbf{I} + \mathbf{D}\right)$. This clearly defines a SVD for $\widehat{\mathbf{K}}$, so by definition we have $\mathbf{\Lambda} = N\left(\mathbf{I} + \mathbf{D}\right)$. Then we can solve for $\mathbf{B}$ from $\mathbf{U}$ and $\mathbf{\Lambda}$, which yields $\mathbf{B} = \frac{1}{\sqrt{N}}\mathbf{U}\left(\mathbf{I} - N\mathbf{\Lambda}^{-1}\right)^{1/2}\mathbf{Q}^\top$. This recovers (19) with $\mathbf{R} = \mathbf{Q}^\top$, and the update equation for new test data can be easily obtained. $\square$