Linear Algebra and its Applications
MSc. Data Science

Joint Sparse Principal Component Analysis

Authors:

Shreyam Banerjee (MDS202236)

Shreyan Chakraborty (MDS202237)

Shubhangi Sanyal (MDS202238)

Shubhashish Chauhan (MDS202239)

Instructor:

Priyavrat Deshpande

Lecturer, Chennai Mathematical Institute

pdeshpande@cmi.ac.in

PROJECT REPORT

September 2, 2023

Work Contributions

Shreyam Banerjee	■ Understanding and explaining the Convergence Analysis proof ■ MATLAB demo of JSPCA ■ Editing the presentation slides
Shreyan Chakraborty	■ Comparison of PCA, SPCA, JSPCA ■ Objective Function of JSPCA ■ Python Implementation ■ Creating and editing the report and presentation slides
Shubhangi Sanyal	■ Optimal Solution (Proof and Procedure) ■ Algorithm and Computational Complexity ■ MATLAB implementation explanation ■ Creating and editing the report and presentation slides
Shubhashish Chauhan	■ Dimensionality Reduction ■ Introduction to PCA ■ Editing the presentation slides

Contents

1 Introduction

Principal Component Analysis (PCA) is a commonly known technique for dimensionality reduction. A lot of variants of this method have been proposed to overcome the drawbacks of PCA which is highly affected by outliers. This report discusses one such improvement of PCA which can be found in Joint Sparse Principal Component Analysis. Here, we discuss why JSPCA is needed and the improvements it brings to the table. Further, the objective function and algorithm has been given. Finally, using the optimal solution for the JSPCA algorithm, some experiments have been performed on a dataset and the performance has been compared with that of the experiments performed in the original paper.

1.1 Literature Review

Large datasets are increasingly common and are often difficult to interpret. Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables, i.e., the Principal Components (PCs), that successively capture maximal variance.
Although PCA works well to manage the size of large datasets by reducing the dimension, there can be a few problems arising. The primary problem occurs while interpreting the results as the PCs are formed by some combination of all the variables of the dataset; it is often difficult to make sense of what a PC represents. Also, PCA is sensitive to outliers since its covariance matrix is derived from ℓ_2-norm which is sensitive to outliers.
Sparse Principal Component Analysis (SPCA) is an extension of the PCA model which aims to reduce the loadings of the PCs. Generating sparsely loaded PCs helps in generating more consistent results and easier interpretations of them.

1.1.1 PCA

Suppose the given data matrix is $X = [x_1, x_2 \dots, x_n] \in R_{m\times n}$, where m denotes the original image space dimensionality and n denotes the number of training samples. Without loss of generality, $\{x_j\}_{j=1}^{n}$ is assumed to have zero mean. The problem of linear dimensionality reduction is to project the data from the high-dimensional original space into a low-dimen- sional subspace. That is, we need to find a transformation matrix $U = [u_1, u_2, \dots, u_d] \in R_{m\times d}$ with $d \leq m$, where each transformation vector u_k is with m loadings $(k = 1, 2, \dots, d)$.
Then, the transformed data denoted by Y can be shown as follows:

$$Y = U^T X \in R_{d\times n}$$

where each U_i represent the $i-th$ row of the transformation matrix U.

Our main objective is to project the data matrix X to some transformed matrix Y so as to reduce it's feature dimensions.

Suppose for feature vector x_i , we do the following transformation:

$$x_i^{'} = [proj_{u_i} x_i] = \frac{u_i \cdot x_i}{\|u_i\|^2 = 1} = u_i^T x_i \tag{1}$$

We also have :

$$\overline{x'} = u_i^T \overline{x} \tag{2}$$

where, $\overline{x'} = mean\{x_i^{'}\}_{i=1}^{n}$ and $\overline{x} = mean\{x_i\}_{i=1}^{n}$

We have to find $u_i \; s\cdot t \; Var\{proj_{u_i} x_i\}_{i=1}^{n}$ is maximal.
Now,

$$Var\{u_i^T x_i\}_{i=1}^{n} = \frac{1}{n}\sum_{i=1}^{n}(u_i^T x_i - u_i^T \overline{x'})^2 \tag{3}$$

Here, $X : ColumnStandardized$,

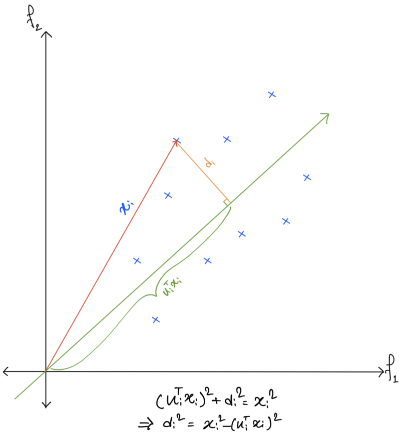
$$\therefore \overline{X'} = [0, 0, \dots, 0]$$

$Equation\,3$ breaks down as follows:

$$Var\{u_i^T x_i\}_{i=1}^n = \frac{1}{n}\sum_{i=1}^{n}(u_i^T x_i)^2 \tag{4}$$

The objective function breaks down as follows:

$$\begin{gathered} \max_{u_i} \frac{1}{n}\sum_{i=1}^{n}(u_i^T x_i)^2 \\ s \cdot t \;\; u_i^T u_i = 1 = \|u_i\|^2 \end{gathered} \tag{5}$$


f_2
x_i
d_i
$u_i^T x_i$
f_1
$(u_i^T x_i)^2 + d_i^2 = x_i^2$
$\Rightarrow d_i^2 = x_i^2 - (u_i^T x_i)^2$

From the above figure it is evident that the above objective can be rewritten as :

$$\begin{gathered} \min_{u_i} \sum_{i=1}^{n}(d_i^2 = \|x_i\|^2 - (u_i^T x_i)^2) \\ s \cdot t \;\; u_i^T u_i = 1 = \|u_i\|^2 \end{gathered} \tag{6}$$

1.1.2 SPCA

Sparse PCA (Principal Component Analysis) is a technique used in data analysis and machine learning to identify the most important features or components in a dataset. The objective function of Sparse PCA is to find a low-dimensional representation of the data while encouraging sparsity in the principal components. In other words, Sparse PCA aims to find the most important features in a dataset while ignoring the less important ones.

In general, sparse PCA can be formulated as an optimization problem over the vector of loadings, with constraints on the ℓ_0 norm of the vector. Since optimization problems involving ℓ_0 norms are in general NP-hard, most methods impose an $\ell_1$norm in the objective function to promote sparsity in the vector of loadings as a relaxation of the ℓ_0 norm.

The objective function of Sparse PCA can be formulated as:

$$\begin{gathered} \arg\min_{U} \|X - XUU^T\|_F^2 + \lambda\|U\|_1 \\ s \cdot t\|U\|^2 = 1 \end{gathered} \tag{7}$$

where X is the data matrix, U is the sparse principal component matrix, λ is the sparsity parameter.The first term represents the reconstruction error, which measures how well the low-dimensional representation of the data captures the original data. The second term encourages sparsity in the principal components by penalizing non-zero entries in U. The objective function is optimized using techniques such as gradient descent or alternating minimization.

1.1.3 Need for JSPCA

To facilitate interpretation, SPCA was proposed. However, SPCA has no ability to jointly select the useful features because the ℓ_1-norm is imposed on each transformation vector and ℓ_1-norm cannot select the consistent features. Moreover, SPCA still suffers from the effect of outliers because the ℓ_2-norm is imposed on loss.

Joint sparse principal component analysis (JSPCA) integrates feature selection into subspace learning to exclude the redundant features. Specifically, JSPCA imposes joint $\ell_{2,1}$-norm on both loss term and regularization term. In this way, our method can discard the useless features on one hand and reduce the effect of outliers on the other hand. The main contributions of the paper are described as follows:

1. JSPCA relaxes the orthogonal constraint of transformation matrix and introduces another transformation matrix to together recover the original data from the subspace spanned by the selected features, which makes JSPCA have more freedom to jointly select useful features for low-dimensional representation.
2. Unlike PCA and its existing extensions, JSPCA uses joint sparse constraints on the objective function, i.e., $\ell_{2,1}$-norm is imposed on the loss term and the transformation matrix, to do feature selection and learn the optimal transformation matrix simultaneously.
3. A simple yet effective optimal solution of JSPCA is provided. Furthermore, a series of theoretical analyses including convergence analysis, essence of JSPCA, and computational complexity are provided to validate the feasibility and effectiveness of JSPCA.

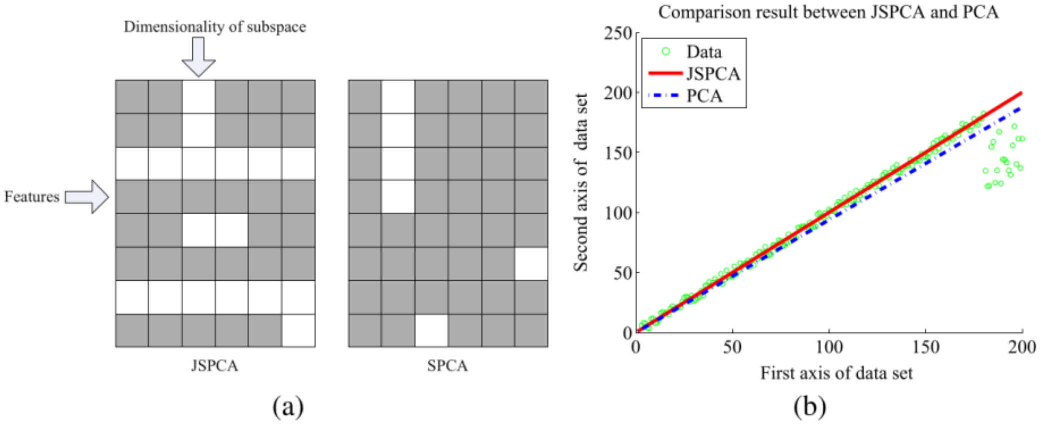


Figure 1: Motivation of JSPCA (a) JSPCA tells us useless features while SPCA cannot and (b) JSPCA is more robust to outliers

2 Methodology

2.1 Setting up the parameters

Before setting up the objective function we have to be clear about certain notations.

1. Given data matrix is $x = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{m \times n}$, where m denotes the features in m dimensional space, and n denotes the number of training samples. Here each x_i represents a column vector.
2. The matrix $l_{2,1}$ norm :
$$\|A\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{m} a_{ij}^2} \tag{8}$$
3. Frobenius norm:
$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{trace(A^T A)} = \sqrt{\sum_{i=1}^{min(m,n)} \sigma_i^2(A)} \tag{9}$$
where, $\sigma_i(A)$ are the singular values of A.
Some properties of the Frobenius norm that have been used in this report:

(a) $\|A\|_F = \|UA\|_F = \|AU\|_F$, for any unitary matrix U

(b) $\|AU\|_F^2 = trace((AU)^T AU) = trace(U^T A^T AU) = trace(UU^T A^T A) = trace(A^T A) = \|A\|_F^2$
Similarly, $\|UA\|_F^2 = \|A\|_F^2$ where, U has a unitary nature i.e., $U^T U = I = UU^T$

(c) $\|A + B\|_F^2 = \|A\|_F^2 + \|B\|_F^2 + 2Re(\langle A, B\rangle_F)$
where, $\langle A, B\rangle_F$ is the Frobenius inner product, and Re is the real part of a complex number (irrelevant for real matrices)

2.2 Objective function

The objective function is given by :

$$\arg\min_{Q,P} J(Q, P) = \arg\min_{Q,P} \left\|X - PQ^T X\right\|_{2,1} + \lambda\|Q\|_{2,1} \tag{10}$$

where transformation matrix $Q \in R^{m\times d}$ is first used to project the data matrix X onto a low-dimensional subspace and another transformation matrix $P \in R^{m\times d}$ is then used to recover the data matrix X. $\lambda \geq 0$ is the regularization parameter.

Directly solving the equation is tough, so the objective function is transformed in terms of the Frobenius norm. This is done by first simplifying the $l_{2,1}$ norm into the trace of matrices, and then introducing two diagonal matrices for further simplification of the objective function.

$$\begin{aligned}
\arg\min_{Q,P} J(Q, P) &= \arg\min_{Q,P} \left\|X - PQ^T X\right\|_{2,1} + \lambda\|Q\|_{2,1} \\
&= \arg\min_{Q,P} 2tr((X - PQ^T X)^T D_1(X - PQ^T X)) + 2\lambda tr(Q^T D_2 Q) \\
&= \arg\min_{Q,P} tr((X - PQ^T X)^T \sqrt{D_1}^T \sqrt{D_1}(X - PQ^T X)) + \lambda tr(Q^T \sqrt{D_2}^T \sqrt{D_2} Q) \\
&= \arg\min_{Q,P} tr((\sqrt{D_1}(X - PQ^T X))^T \sqrt{D_1}(X - PQ^T X)) + \lambda tr((\sqrt{D_2} Q)^T \sqrt{D_2} Q) \\
&= \arg\min_{Q,P} \left\|\sqrt{D_1}(X - PQ^T X)\right\|_F^2 + \lambda\left\|\sqrt{D_2} Q\right\|_F^2
\end{aligned} \tag{11}$$

Hence, the final objective function ,taken from eq.(11) is as follows :

$$\arg\min_{Q,P} J(Q, P) = \arg\min_{Q,P} \left\|\sqrt{D_1}(X - PQ^T X)\right\|_F^2 + \lambda\left\|\sqrt{D_2} Q\right\|_F^2 \ (11)$$

where

$$D_1 = \begin{bmatrix} \frac{1}{\sqrt{\sum_{i=1}^m (X - PQ^T X)_{1i}^2}} & & \\ & \ddots & \\ & & \frac{1}{\sqrt{\sum_{i=1}^m (X - PQ^T X)_{mi}^2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{2||(X - PQ^T X)^1||_2} & & \\ & \ddots & \\ & & \frac{1}{2||(X - PQ^T X)^m||_2} \end{bmatrix} \tag{12}$$

and

$$D_2 = \begin{bmatrix} \frac{1}{\sqrt{\sum_{i=1}^m Q_{1i}^2}} & & \\ & \ddots & \\ & & \frac{1}{\sqrt{\sum_{i=1}^m Q_{mi}^2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{2||Q^1||_2} & & \\ & \ddots & \\ & & \frac{1}{2||Q^m||_2} \end{bmatrix} \tag{13}$$

1. Here, D_1 & D_2 are two $m \times m$ diagonal matrices.
2. Note: $(X - PQ^T X)^i$ for $(i = 1, 2, ...m)$ means the $i - th$ row of matrix $(X - PQ^T X)$.
3. Similarly, Q^i for $(i = 1, 2, ...m)$ means the $i - th$ row of matrix Q.

It is evident that the smaller D_2^{ii} is, the more important the i-th feature is. Moreover, if $||(X-PQ^TX)^i||_2$ and $||Q^i||_2$ are small, then D_1 and D_2 are large. So, the minimization of $(X-PQ^T)^T D_1(X-PQ^TX))+2\lambda\mathrm{tr}(Q^TD_2Q)$ forces $||(X-PQ^TX)^i||_2$ and $||Q^i||_2$ to have very small values.
Thus, we obtain a joint sparse Q and a small reconstruction matrix P.

Next, let $\sqrt{D_1}P = \overline{P}$ and $\sqrt{D_1}^{-1}Q = \overline{Q}$. Then eq. (2) becomes

$$\underset{Q,P}{\arg\min} J(Q,P) = \underset{Q,P}{\arg\min}\ ||\sqrt{D_1}^T X - \overline{PQ}^T \sqrt{D_1}X)||_F^2 + 2\lambda||\sqrt{D_1}\sqrt{D_2}\overline{Q})||_F^2 \tag{14}$$

In order to reduce the feature redundancy, the orthogonality constraint $\overline{P}^T\overline{P} = I^{d\times d}$ is imposed.
Therefore, we have our final objective function

$$\begin{aligned} \underset{Q,P}{\arg\min} J(Q,P) = \underset{Q,P}{\arg\min}\ & ||\sqrt{D_1}^T X - \overline{PQ}^T \sqrt{D_1}X)||_F^2 \\ & + 2\lambda||\sqrt{D_1}\sqrt{D_2}\overline{Q})||_F^2 \\ & \text{s.t. } \overline{P}^T\overline{P} = I^{d\times d} \end{aligned} \tag{15}$$

where $\overline{Q} \in R^{m\times d}$ is first used to project the weighted data matrix $\sqrt{D_1}X$ and $\overline{P} \in R^{m\times d}$ is then used to recover it.

2.3 Optimal Solution and Algorithm

The procedure for deriving the optimal solution from the final objective function involves two steps:

1. Given $\overline{P}$, find an approximation to $\overline{Q}$ and ultimately to Q
2. Given $\overline{Q}$, find an approximation to $\overline{P}$ and ultimately to P

The above steps are explained in detail below.

2.3.1 Step 1: Solving for projection matrix

Here, the projection matrix is Q. We first find an approximation to $\overline{Q}$ and then using the definition of $\overline{Q}$, we find Q. Now, given $\overline{P}$, there exists an optimal matrix $\overline{P}_\perp$ such that $[\overline{P}, \overline{P}_\perp]$ is an $m \times m$ column orthogonal matrix. Here, $[\overline{P}, \overline{P}_\perp]$ is nothing but an extension of $\overline{P}$ with more orthogonal columns. With that, the optimization problem becomes,

$$\underset{\overline{Q}}{arg\,min}\ \left\|\sqrt{D_1}^T X - \overline{PQ}^T \sqrt{D_1}X)\right\|_F^2 + \lambda\left\|\sqrt{D_1}\sqrt{D_2}\overline{Q})\right\|_F^2 \tag{16}$$

The first part of the above equation becomes,

$$\begin{aligned} \left\|\sqrt{D_1}^T X - \overline{PQ}^T \sqrt{D_1}X)\right\|_F^2 &= \left\|X^T\sqrt{D_1} - X^T\sqrt{D_1}\overline{QP}^T\right\|_F^2 \\ &= \left\|X^T\sqrt{D_1}[\overline{P},\overline{P}_\perp] - X^T\sqrt{D_1}\overline{QP}^T[\overline{P},\overline{P}_\perp]\right\|_F^2 \\ &= \left\|X^T\sqrt{D_1}\overline{P} - X^T\sqrt{D_1}\overline{QP}^T\overline{P}\right\|_F^2 + \left\|X^T\sqrt{D_1}\overline{P}_\perp - X^T\sqrt{D_1}\overline{QP}^T\overline{P}_\perp\right\|_F^2 \\ &= \left\|X^T\sqrt{D_1}\overline{P} - X^T\sqrt{D_1}\overline{Q}\right\|_F^2 + \left\|X^T\sqrt{D_1}\overline{P}_\perp\right\|_F^2 \end{aligned}$$

Keeping $\overline{P}$ fixed, the optimization problem becomes,

$$\underset{\overline{Q}}{arg\,min}\ \left\|X^T\sqrt{D_1}\overline{P} - X^T\sqrt{D_1}\overline{Q}\right\|_F^2 + \lambda\left\|\sqrt{D_1}\sqrt{D_2}\overline{Q})\right\|_F^2 \tag{17}$$

In order to minimize $\overline{Q}$, the derivative of the above equation w.r.t $\overline{Q}$ has been set to 0. Then,

$$\overline{Q} = (\lambda\sqrt{D_1}D_2\sqrt{D_1} + \sqrt{D_1}XX^T\sqrt{D_1})^{-1}\sqrt{D_1}XX^T\sqrt{D_1}\overline{P} \tag{18}$$

Thus,

$$Q = (\lambda D_2 + XX^T)^{-1}XX^T\sqrt{D_1}\overline{P} \tag{19}$$

2.3.2 Step 2: Solving for recovery matrix

Given $\overline{Q}$, the optimization problem to compute $\overline{P}$ becomes,

$$\underset{\overline{P}}{arg\,min} \left\| \sqrt{D_1}X - \overline{PQ}^T \sqrt{D_1}X \right\|_F^2, \text{ s.t } \overline{PP}^T = I^{d\times d} \tag{20}$$

Using the properties of the Frobenius norm, the above equation can be rewritten as,

$$\begin{aligned}\left\| \sqrt{D_1}X - \overline{PQ}^T \sqrt{D_1}X \right\|_F^2 &= tr((\sqrt{D_1}X - \overline{PQ}^T \sqrt{D_1}X)^T(\sqrt{D_1}X - \overline{PQ}^T \sqrt{D_1}X)) \\ &= tr((X^T\sqrt{D_1} - X^T\sqrt{D_1}\overline{QP}^T)^T(\sqrt{D_1}X - \overline{PQ}^T\sqrt{D_1}X)) \\ &= tr(X^TD_1X - X^T\sqrt{D_1}\overline{PQ}^T\sqrt{D_1}X - X^T\sqrt{D_1}\overline{QP}^T\sqrt{D_1}X \\ &\quad + X^T\sqrt{D_1}\overline{QP}^T\overline{PQ}^T\sqrt{D_1}X) \\ &= tr(X^TD_1X + X^T\sqrt{D_1}\overline{QQ}^T\sqrt{D_1}X) - 2tr(\overline{Q}^T\sqrt{D_1}XX^T\sqrt{D_1}\overline{P})\end{aligned}$$

Since $\overline{Q}$ is given, the above equation becomes,

$$\underset{\overline{P}}{arg\,min}\; tr(\overline{Q}^T\sqrt{D_1}XX^T\sqrt{D_1}\overline{P}), \text{ s.t } \overline{PP}^T = I^{d\times d} \tag{21}$$

The original optimization problem for computing the recovery matrix (eq. 20) can also be written as:

$$\underset{\overline{P}}{\arg\min} \left\| X^T\sqrt{D_1} - X^T\sqrt{D_1}\overline{QP}^T \right\|_F^2 \text{ s.t. } \overline{P}^T\overline{P} = I^{d\times d} \tag{22}$$

Now, in order to compute $\overline{P}$, we introduce the following lemma:
Lemma 1: Let $Z^{n\times m}$ and $V^{n\times d}$ be two matrices. Consider the constrained minimization problem,

$$\underset{\overline{P}}{\arg\min} \left\| Z - V\overline{P}^T \right\|^2 \text{ s.t. } \overline{P}^T\overline{P} = I^{d\times d} \tag{23}$$

Suppose the SVD of Z^TV is EDU^T, then the optimal solution is $\overline{P} = EU^T$.

Proof:
We have to minimize $\left\| Z - V\overline{P}^T \right\|_F^2$ for an optimal $\overline{P}$ such that $\overline{P}^T\overline{P} = I$.
Now,

$$\left\| Z - V\overline{P}^T \right\|_F^2 = \text{tr}((Z - V\overline{P}^T)^T(Z - V\overline{P}^T)) \tag{24}$$

But,

$$\begin{aligned}(Z - V\overline{P}^T)^T(Z - V\overline{P}^T) &= (Z^T - \overline{P}V^T)(Z - V\overline{P}^T) \\ &= Z^TZ - Z^TV\overline{P}^T - \overline{P}V^TZ + \overline{P}V^TV\overline{P}^T\end{aligned}$$

Putting this in eq. (23), we get

$$\begin{aligned}\left\| Z - V\overline{P}^T \right\|_F^2 &= \text{tr}(Z^TZ - Z^TV\overline{P}^T - \overline{P}V^TZ + \overline{P}V^TV\overline{P}^T) \\ &= \text{tr}(Z^TZ) - \text{tr}(Z^TV\overline{P}^T) - \text{tr}(\overline{P}V^TZ) + \text{tr}((\overline{P}V^TV\overline{P}^T) \\ &= \|Z\|_F - \text{tr}(EDU^T\overline{P}^T) - \text{tr}(\overline{P}UDE^T) + \text{tr}(V^TV) \\ &(\because \overline{P}V^TV\overline{P}^T \sim V^TV \implies \text{tr}(\overline{P}V^TV\overline{P}^T) = \text{tr}(V^TV) \\ &= \|Z\|_F - \text{tr}(DU^T\overline{P}^TE) - \text{tr}(DE^T\overline{P}U) + \|V\|_F\end{aligned}$$

Here, $\|Z\|_F$ and $\|V\|_F$ are constant. So, we have to minimize the remaining terms.
Now, U, $\overline{P}$ and E are unitary matrices. We can claim

$$\text{tr}(AB) \le \text{tr}(A) \tag{25}$$

where A is a diagonal matrix with non-negative entries and B is unitary, since for arbitrary diagonal matrices D and U, we have

$$\begin{bmatrix} d_{11} & & & \\ & d_{22} & & \\ & & \ddots & \\ & & & d_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \cdots & u_{nn} \end{bmatrix} = \begin{bmatrix} d_{11}u_{11} & & & \\ & d_{22}u_{22} & & \\ & & \ddots & \\ & & & d_{nn}u_{nn} \end{bmatrix}$$

Since $|u_{ij}| \leq 1 \ \forall 1 \leq i, j \leq n$, hence the claim is true.
So, if we choose $\overline{P} = EU^T$, then $\operatorname{tr}(DU^T\overline{P}^T E) = \operatorname{tr}(D)$ and $\operatorname{tr}(DE^T\overline{P}U) = \operatorname{tr}(D)$, and we will get the optimal solution.
Using the definition of $\overline{P}$, we compute P as,

$$P = \sqrt{D_1}^{-1} EU^T \tag{26}$$

In fact, before we compute $\overline{Q}$, we need to compute the input of matrix $\overline{P}$, D_1, and D_2, which cannot be obtained directly. Therefore, we need to compute them in the designed iterative algorithm.

2.4 Algorithm

```
Input: Training sample set $X$, parameter $\lambda$, and dimensionality $d$

1. Initialize $D_1 = I^{m \times m}$, $D_2 = I^{m \times m}$, and random $\overline{P}^{m \times d}$.
2. while not converge do
    (a) Compute $\overline{Q}$ according to equation $\overline{Q} = (\lambda\sqrt{D_1}D_2\sqrt{D_1} + \sqrt{D_1}XX^T\sqrt{D_1})^{-1}\sqrt{D_1}XX^T\sqrt{D_1}\overline{P}$
    (b) Compute $Q$ according to equation $Q = (\lambda D_2 + XX^T)^{-1}XX^T\sqrt{D_1}\overline{P}$
    (c) Compute $\overline{P}$ according to equation $\overline{P} = EU^T$
    (d) Compute $P$ according to equation $P = \sqrt{D_1}^{-1}EU^T$
    (e) Compute $D_1$ according to equation
        $D_1 = \begin{bmatrix} \frac{1}{2||(X-PQ^TX)^1||_2} & & \\ & \ddots & \\ & & \frac{1}{2||(X-PQ^TX)^m||_2} \end{bmatrix}$
    (f) Compute $D_2$ according to equation
        $D_2 = \begin{bmatrix} \frac{1}{2||Q^1||_2} & & \\ & \ddots & \\ & & \frac{1}{2||Q^m||_2} \end{bmatrix}$
3. end while
4. Normalize each column vector of $Q$ to be identity vectors.

Output: Transformation matrix $Q$
```

2.5 Computation Complexity

In each iteration, two main steps are involved:

1. Computing $Q = (\lambda D_2 + XX^T)^{-1}XX^T\sqrt{D_1}P$ takes $\mathcal{O}(m^3)$ time.
2. Computing SVD of $\sqrt{D_1}XX^T\sqrt{D_1}\overline{Q} = EDU^T$ also takes $\mathcal{O}(m^3)$ time at most.

So, the computational complexity for one iteration will be up to $\mathcal{O}(m^3)$.
For t iterations, it will be $\mathcal{O}(tm^3)$.

3 Discussion

This section discusses the convergence proof of the JSPCA method as it is vital for the termination of the algorithm stated in the previous section.

3.1 Convergence Analysis

For any non-zero vectors $p, q \in R^c$, the following result holds:

$$\|p\|_2 - \frac{\|p\|_2^2}{\|q\|_2} \leq \|q\|_2 - \frac{\|q\|_2^2}{\|p\|_2} \tag{27}$$

Recall eq.

$$\arg\min_{Q,P} J(Q,P) = \arg\min_{Q,P} \left\| X - PQ^T X \right\|_{2,1} + \lambda \|Q\|_{2,1} \tag{28}$$

where transformation matrix $Q \in R^{m \times d}$ is first used to project the data matrix X onto a low-dimensional subspace and another transformation matrix $P \in R^{m \times d}$ is then used to recover the data matrix X. $\lambda \geq 0$ is the regularization parameter.

Theorem 1: Given all the variables in eq. (27) except P, Q, the optimization problem in eq. (27) will monotonically decrease the objective function value in each iteration and converge to the local optimal solution. We denote the objective function as $(J(Q,P) = J(Q,P,D_1,D_2)$. Suppose for the $(t-1)$-th iteration, we obtain $P^{(t-1)}, Q^{(t-1)}, D_1^{(t-1)}$ and $D_2^{(t-1)}$. From eq (18), we can find

$$J(Q^{(t)}, P^{(t-1)}, D_1^{(t-1)}, D_2^{(t-1)}) \leq J(Q^{(t-1)}, P^{(t-1)}, D_1^{(t-1)}, D_2^{(t-1)}) \tag{29}$$

Since the SVD gives the optimal $P^{(t)}$ that further decreases the objective value, we have

$$J(Q^{(t)}, P^{(t)}, D_1^{(t-1)}, D_2^{(t-1)}) \leq J(Q^{(t-1)}, P^{(t-1)}, D_1^{(t-1)}, D_2^{(t-1)}) \tag{30}$$

Once the optimal $P^{(t)}$ and $Q^{(t)}$ are obtained, we have

$$\begin{aligned}
&\mathrm{tr}((X - P^{(t)}Q^{(t)}X)^T D_1^{(t-1)} (X - P^{(t)}Q^{(t)}X) \\
&+ \lambda \mathrm{tr}((Q^{(t)T}) D_2^{(t-1)} Q^{(t)}) \\
&\leq \mathrm{tr}((X - P^{(t-1)}Q^{(t-1)}X)^T D_1^{(t-1)} (X - P^{(t-1)}Q^{(t-1)}X) \\
&+ \lambda \mathrm{tr}((Q^{(t-1)T}) D_2^{(t-1)} Q^{(t-1)})
\end{aligned}$$

That is,

$$\begin{aligned}
&\mathrm{tr}\left(\sum_{i=1}^{m} \frac{\left\| X - P_i^{(t)} {Q_i^{(t)}}^T X \right\| 2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2} \right) + \lambda \mathrm{tr}\left(\sum_{i=1}^{m} \frac{\left\| Q_i^{(t)} \right\|_2^2}{\left\| Q^{(t-1)} \right\|_2} \right) \\
&\leq \mathrm{tr}\left(\sum_{i=1}^{m} \frac{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\| 2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2} \right) + \lambda \mathrm{tr}\left(\sum_{i=1}^{m} \frac{\left\| Q_i^{(t)} \right\|_2^2}{\left\| Q^{(t-1)} \right\|_2} \right)
\end{aligned} \tag{31}$$

From Lemma 2, we have,

$$\begin{aligned}
&\left\| X - P_i^{(t)} {Q_i^{(t)}}^T X \right\|_2 - \frac{\left\| X - P_i^{(t)} {Q_i^{(t)}}^T X \right\|_2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2} \\
&\leq \left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2 - \frac{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2}
\end{aligned} \tag{32}$$

Using the matrix calculus on eq. (32), we have,

$$\begin{aligned}
&\sum_{i=1}^{m} \left(\left\| X - P_i^{(t)} {Q_i^{(t)}}^T X \right\|_2 - \frac{\left\| X - P_i^{(t)} {Q_i^{(t)}}^T X \right\|_2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2} \right) \\
&\leq \sum_{i=1}^{m} \left(\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2 - \frac{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2^2}{\left\| X - P_i^{(t-1)} Q_i^{(t-1)T} X \right\|_2} \right)
\end{aligned} \tag{33}$$

Again, from Lemma 2, we have,

$$\left\|Q_i^{(t)}\right\|_2 - \frac{\left\|Q_i^{(t)}\right\|_2^2}{\left\|Q_i^{(t-1)}\right\|_2} \leq \left\|Q_i^{(t-1)}\right\|_2 - \frac{\left\|Q_i^{(t-1)}\right\|_2^2}{\left\|Q_i^{(t-1)}\right\|_2} \tag{34}$$

Again, using matrix calculation, we have,

$$\sum_{i=1}^{m} \left(\left\|Q_i^{(t)}\right\| 2 - \frac{\left\|Q_i^{(t)}\right\|_2^2}{\left\|Q_i^{(t-1)}\right\|_2} \right) \leq \sum_{i=1}^{m} \left(\left\|Q_i^{(t-1)}\right\|_2 - \frac{\left\|Q_i^{(t-1)}\right\|_2^2}{\left\|Q_i^{(t-1)}\right\|_2} \right) \tag{35}$$

Combining eq. (31), (33) and (35), we have

$$\begin{aligned} & J(Q^{(t)}, P^{(t)}, D_1^{(t-1)}, D_2^{(t-1)}) = \left\|XP^{(t)}Q^{(t)T}X\right\|_{2,1} + \lambda\left\|Q^{(t)}\right\|_{2,1} \\ & \leq \left\|XP^{(t-1)}Q^{(t-1)T}X\right\|_{2,1} + \lambda\left\|Q^{(t-1)}\right\|_{2,1} \\ & = J(Q^{(t-1)}, P^{(t-1)}, D_1^{(t-1)}, D_2^{(t-1)}) \end{aligned} \tag{36}$$

That is,

$$\begin{aligned} & J(Q^{(t)}, P^{(t)}) = J(Q^{(t)}, P^{(t)}, D_1^{(t)}, D_2^{(t)}) \\ & \leq J(Q^{(t-1)}, P^{(t-1)}, D_1^{(t-1)}, D_2^{(t-1)}) = J(Q^{(t-1)}, P^{(t-1)}) \end{aligned} \tag{37}$$

4 Implementation

4.1 MATLAB code on the Breast-Cancer Wisconsin dataset

Using MATLAB, we implemented the JSPCA iterative algorithm. The code for which is given here. One iteration of JSPCA is captured as a MATLAB function, which we can run either for a fixed number of iterations or till the cost function converges. The code outputs the computed matrix Q as well as the decreasing cost-values across the iterations performed.

It is worth noting that since we have an iterative algorithm that relies on a regularization term to get sparse loadings, the values do not absolutely converge to 0. Instead, they reduce to small values on the order of 10^{-2}. We must set a threshold there to trim these values to absolute 0. This step is necessary, say, when we want to quantify the sparsity of Q.

Results

The dataset is used for investigating samples of breast-tissue to study the characteristics of cancerous breast cells. The features (columns) of the dataset are derived from the image data and describe the features of breast cells such as cell radius, smoothness, symmetry etc. There are 31 columns in the dataset and the highly correlated nature of the input features highlights the need for dimensionality reduction techniques.

We run JSPCA on this dataset with the parameters:

- `d = 6` (No. of output dimensions, i.e. No. of principal components)
- $\lambda = 3.0$ (Regularization parameter)
- `NumIterations = 50`
  We stop after 50 iterations are completed and assume convergence.

No. of total loadings	186	No. of input features	31	Total Variance (Normalized)	31
No. of 0 (sparse) loadings	152	No. of features removed	16	Variance of PCs	8.56
Sparsity	81.7%	Joint-Sparsity	51.6%	Variance Explained	27.6%

Table 1: Results of running JSPCA on MATLAB

PCA vs. JSPCA

We can compare the performance of PCA vs JSPCA by running both algorithms in MATLAB on the same Breast-Cancer dataset to find the first 6 PC's (principal components).

- JSPCA has eliminated 16 of the 31 input features entirely, resulting in a much smaller feature space. On the other hand, PCA results in no sparsity and cannot reduce the feature space at all.
- PCA was able to capture 88% of the variance in the first 6 PC's. JSPCA captured only 28% of the variance in the same number of PC's.

The PC's calculated by the vanilla PCA algorithm were oriented in very different directions compared to the PC's calculated by JSPCA.
We may reason that - since the algorithm and the corresponding cost function are different, along with removing the orthogonal constraint and asking for joint-sparsity to be there in the PC's, the resulting PC's found are bound to be different from the PC's found by vanilla PCA.
Hence, the components found by JSPCA are not principal components in a strict sense, but instead are components that can capture a moderate amount of the dataset's variance while simultaneously identifying redundant features through joint-sparsity.

4.2 Python code on the MNIST Dataset

This time we tried the JSPCA Algorithm in Python. The MNIST dataset is a large collection of handwritten digits and our goal was to see how easily PCA, SPCA, and JSPCA could process the images and identify the digits. The projection results have been given below.

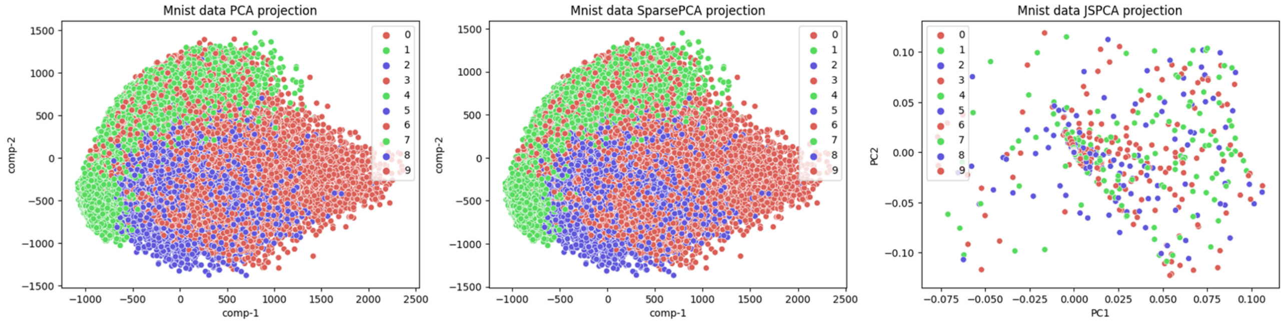


Figure 2: Projection comparison between PCA, SPCA, and JSPCA

For the python code implementation,the `SparsePCA` library from `sklearn` was heavily used.
Moreover, we see that while standard PCA and SparsePCA has almost no effect on reducing feature redundancy, JSPCA greatly reduced the number of key features required. In all the three cases the metric used for calibration is `Explained Variance` :

- PCA Explained Variance - approx. 0.73 out of a possible 1
- Sparse PCA Explained Variance - approx. 0.98 out of a possible 1
- JPCA Explained Variance - approx. 1 out of a possible 1

5 Conclusion

In the paper this report is based on, JSPCA is proposed to find representative features from the original high-dimensional space. The found representative features have been used for classification tasks. Although JSPCA outperforms the other PCA methods in most of the classification experiments, a series of PCA methods including JSPCA achieve low classification accuracy overall. This is because these PCA methods do not use class labels to extract discriminative features. Any dimensionality reduction method without using class labels does not always extract effective features for classification. In the future, this method can be extended to the supervised method to solve the skewed/imbalanced classification problem.

6 References

1. Yi, S., Lai, Z., He, Z., Cheung, Y.M. and Liu, Y., 2017. Joint sparse principal component analysis. Pattern Recognition, 61, pp.524-536.
2. F. Nie, H. Huang, X. Cai, C.H. Ding, Efficient and robust feature selection via joint l2,1-norms minimization, in: Advances in Neural Information Processing Systems, 2010, pp. 1813–1821.
3. Q. Gu, Z. Li, J. Han, Joint feature selection and subspace learning, in: Interna- tional Joint Conference on Artificial Intelligence, 2011, pp. 1294–1299.
4. H. Lu, K.N. Plataniotis, A.N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recognit. 44 (7) (2011) 1540–1551.
5. W. Ou, X. You, D. Tao, P. Zhang, Y. Tang, Z. Zhu, Robust face recognition via occlusion dictionary learning, Pattern Recognit. 47 (4) (2014) 1559–1572.
6. J. Huang, X. You, Y. Yuan, F. Yang, L. Lin, Rotation invariant iris feature ex- traction using Gaussian Markov random fields with non-separable wavelet, Neurocomputing 73 (4) (2010) 883–894.