

SBE 4032: Advanced Topics in Medical Informatics (2)

مُوْضُو عَاتٍ مُتَقْدِمَةٍ فِي الْمَعْلُومَاتِيَّةِ الطَّبِيَّةِ (2)

Lecture #2: Dimensionality Reduction — Linear Methods

Hisham Abdeltawab, Ph.D.
Assistant Professor,
Cairo University

Introduction

- *Dimensionality reduction:* is the process of finding a suitable lower dimensional space in which to represent the original data.
- One possible method for dimensionality reduction would be to just select subsets of the variables for processing and analyze them in groups. However, in some cases, that would mean throwing out a lot of useful information.
- An alternative would be to create new variables that are functions (e.g., linear combinations) of the original variables.
- The methods we describe, where we seek a mapping from the higher dimensional space to a lower-dimensional one, while keeping information on all of the available variables.
- In general, this mapping can be linear or nonlinear.

Introduction: Projection

- Since some of the methods in this lecture transform the data using projections, we take a moment to describe this concept before going on to explain how they work.
- A projection will be in the form of a matrix that takes the data from the original space to a lower-dimensional one. We illustrate this concept in the following example:
- In this example, we show how projection works when our data set consists of the two bivariate points

$$\mathbf{x}_1 = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} -4 \\ 5 \end{bmatrix},$$

and we are projecting onto a line that is θ radians from the horizontal or x axis. For this example, the projection matrix is given by

$$\mathbf{P} = \begin{bmatrix} (\cos\theta)^2 & \cos\theta\sin\theta \\ \cos\theta\sin\theta & (\sin\theta)^2 \end{bmatrix}$$

Introduction: Projection

The following MATLAB code is used to enter this information:

```
% Enter the data as rows of our matrix X.  
X = [4 3; -4 5];  
% Provide a value for theta.  
theta = pi/3;  
% Now obtain the projection matrix.  
c2 = cos(theta)^2;  
cs = cos(theta)*sin(theta);  
s2 = sin(theta)^2;  
P = [c2 cs; cs s2];
```

The coordinates of the projected observations are a weighted sum of the original variables, where the columns of P provide the weights. We project a single observation (represented as a column vector) as follows:

$$\mathbf{y}_i = \mathbf{P}^T \mathbf{x}_i \quad i = 1, \dots, n,$$

Introduction: Projection

We have to use some linear algebra to project the data matrix \mathbf{X} because the observations are *rows* of this matrix. Taking the transpose of both sides of our projection equation above, we have

$$\begin{aligned}\mathbf{y}_i^T &= (\mathbf{P}^T \mathbf{x}_i)^T \\ &= \mathbf{x}_i^T \mathbf{P}.\end{aligned}$$

Thus, we can project the data using this MATLAB code:

```
% Now project the data onto the theta-line.  
% Since the data are arranged as rows in the  
% matrix X, we have to use the following to  
% project the data.  
Xp = X*P;  
plot(Xp(:,1),Xp(:,2),'o') % Plot the data.
```

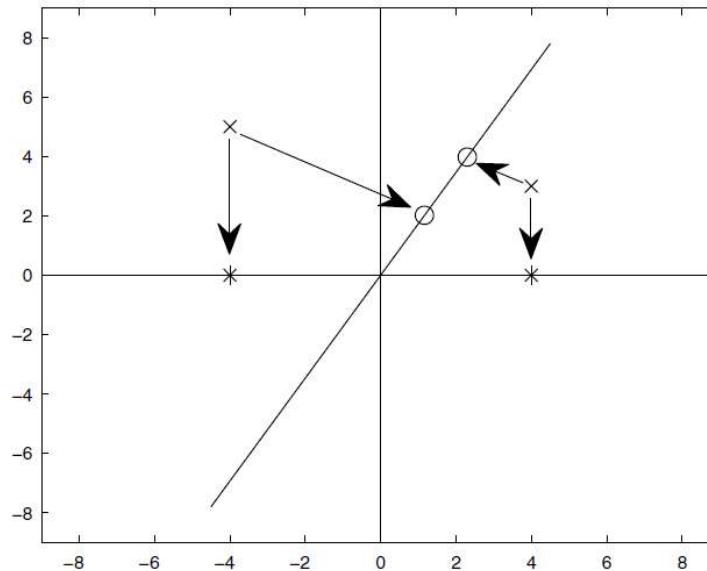
This projects the data onto a 1-D subspace that is at an angle θ with the original coordinate axes.

Introduction: Projection

As an example of a projection onto the horizontal coordinate axis, we can use these commands:

```
% We can project onto the 1-D space given by the  
% horizontal axis using the projection:  
Px = [1;0];  
Xpx = X*Px;
```

One could also use the projection matrix P with $\theta = 0$. These data now have only one coordinate value representing the number of units along the x-axis.



Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- We have surveyed *100 participants*. Each of these participants has a recorded answer for both the ‘happiness’ and ‘achievement’ survey questions.
- We represent this as the input matrix X , which is 100×2 long. Each row is a new training example (individual) and each column is that individual’s happiness (x_1) and achievement (x_2).

$$\begin{matrix} & \text{happiness} & \text{achievement} \\ \text{person 1} & x_1^1 & x_2^1 \\ \text{person 2} & x_1^2 & x_2^2 \\ & \vdots & \vdots \\ \text{person 100} & x_1^{100} & x_2^{100} \end{matrix}$$

100×2

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- **The Covariance Matrix:**

What we want to do is construct a 2×2 *covariance matrix* which represents the covariance between each variable.

$$\begin{bmatrix} \text{covariance of } x_1 \text{ and } x_1 & \text{covariance of } x_1 \text{ and } x_2 \\ \text{covariance of } x_2 \text{ and } x_1 & \text{covariance of } x_2 \text{ and } x_2 \end{bmatrix}$$

- A negative covariance between some variables a and b means that when a goes up, b goes down. A positive covariance means that when a goes up, so does b.
- Our covariance matrix:

$$\begin{bmatrix} \text{variance of happiness} & \text{cov. of achiev. / happy} \\ \text{cov. of achiev. / happy} & \text{variance of achievement} \end{bmatrix}$$

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- **Constructing the Covariance Matrix:**

The exact formula for calculating the covariance of two features (each with N examples):

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N - 1}$$

$cov_{x,y}$ covariance of features x and y

x_i specific training example from feature x

y_i specific training example from feature y

\bar{x} mean over all examples of feature x

\bar{y} mean over all examples of feature y

N total amount of examples

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- What we can do to ‘automate’ the process of subtracting the mean from each feature. We can normalize our matrix by subtracting the mean of each feature from each training example.

$$\bar{x}_1 = \frac{x_1^1 + x_1^2 + \dots + x_1^N}{N}$$

$$\bar{x}_2 = \frac{x_2^1 + x_2^2 + \dots + x_2^N}{N}$$

where N = 100

$$\begin{bmatrix} x_1^1 - \bar{x}_1 & x_2^1 - \bar{x}_2 \\ x_1^2 - \bar{x}_1 & x_2^2 - \bar{x}_2 \\ \vdots & \vdots \\ x_1^{100} - \bar{x}_1 & x_2^{100} - \bar{x}_2 \end{bmatrix} X_{norm}$$

normalize by subtracting means

$$\begin{bmatrix} x_1^1 - \bar{x}_1 & x_1^2 - \bar{x}_1 & \dots & x_1^{100} - \bar{x}_1 \\ x_2^1 - \bar{x}_2 & x_2^2 - \bar{x}_2 & \dots & x_2^{100} - \bar{x}_2 \end{bmatrix} \begin{bmatrix} x_1^1 - \bar{x}_1 & x_2^1 - \bar{x}_2 \\ x_1^2 - \bar{x}_1 & x_2^2 - \bar{x}_2 \\ \vdots & \vdots \\ x_1^{100} - \bar{x}_1 & x_2^{100} - \bar{x}_2 \end{bmatrix}$$

100 x 2

100 x 2

$$X_{norm}^T X_{norm}$$

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- We then form the sample covariance matrix S as:

$$= \frac{X_{norm}^T X_{norm}}{N - 1}$$

Eigenvectors of the Covariance Matrix and Spectral Theorem

The next step is to calculate the eigenvectors and eigenvalues of the matrix S . The eigenvalues are found by solving the following equation for each $l_j, j = 1, \dots, p$ where $p = 2$ in our example

$$|\mathbf{S} - l\mathbf{I}| = 0$$

where I is a $p \times p$ identity matrix and $|\cdot|$ denotes the determinant.

The eigenvectors are obtained by solving the following set of equations for a_j $(\mathbf{S} - l_j\mathbf{I})\mathbf{a}_j = 0 \quad j = 1, \dots, p$

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

subject to the condition that the set of eigenvectors is orthonormal. This means that the magnitude of each eigenvector is one, and they are orthogonal to each other:

$$\begin{aligned}\mathbf{a}_i \mathbf{a}_i^T &= 1 \\ \mathbf{a}_j \mathbf{a}_i^T &= 0,\end{aligned}$$

for $i, j = 1, \dots, p$ and $i \neq j$.

Because the covariance matrix is symmetric, the eigenvectors are orthogonal (perpendicular) to each other.

A major result in matrix algebra shows that any square, symmetric, nonsingular matrix can be transformed to a diagonal matrix using:

$$\mathbf{L} = \mathbf{A}^T \mathbf{S} \mathbf{A}$$

where the columns of \mathbf{A} contain the eigenvectors (orthogonal) of \mathbf{S} , and \mathbf{L} is a diagonal matrix with the eigenvalues along the diagonal.

Principal Component Analysis: The Eigenvectors of the Covariance Matrix Method

- We transform the observations to the PC coordinate system via the following equation:

$$Z = X_{norm}A$$

The matrix Z contains the principal component scores, and the columns of A are the eigenvectors.

- Note that these PC scores have zero mean (because we are using the data centered about the mean) and are uncorrelated.
- The dimensionality of the principal component scores in the last equation is still p (the original dimension); so **no dimensionality reduction has taken place**.
- The idea of dimensionality reduction with PCA is that one could include in the analysis only those PCs that have the highest eigenvalues, thus accounting for the highest amount of variation with fewer dimensions or PC variables.
- **We can reduce the dimensionality to d with the following:** $Z_d = X_{norm}A_d$

where A_d contains the first d eigenvectors. We see that Z_d is an $n \times d$ matrix (each observation now has only d elements), and A_d is a $p \times d$ matrix.

Principal Components Analysis: How Many Dimensions Should We Keep?

Cumulative Percentage of Variance Explained

This is a popular method of determining the number of PCs to use in PCA dimensionality reduction and seems to be implemented in many computer packages. The idea is to select those d PCs that contribute a specified cumulative percentage of total variation in the data, which is calculated using

$$t_d = 100 \frac{\sum_{i=1}^d l_i}{\sum_{j=1}^p l_j}$$

Choosing a value for t_d can be problematic, but typical values range between 70% and 95%.

Principal Components Analysis: How Many Dimensions Should We Keep?

Scree Plot

A graphical way of determining the number of PCs to retain is called the *scree plot*, and it is a plot of l_k (the eigenvalue) versus k (the index of the eigenvalue).

In some cases, we might plot the log of the eigenvalues when the first eigenvalues are very large. This type of plot is called a log-eigenvalue or LEV plot. To use the scree plot, one looks for the ‘elbow’ in the curve or the place where the curve levels off and becomes almost flat. The value of k at this elbow is an estimate for how many PCs to retain. Another way to look at this is by the slopes of the lines connecting the points. When the slopes start to level off and become less steep, that is the number of PCs one should keep.

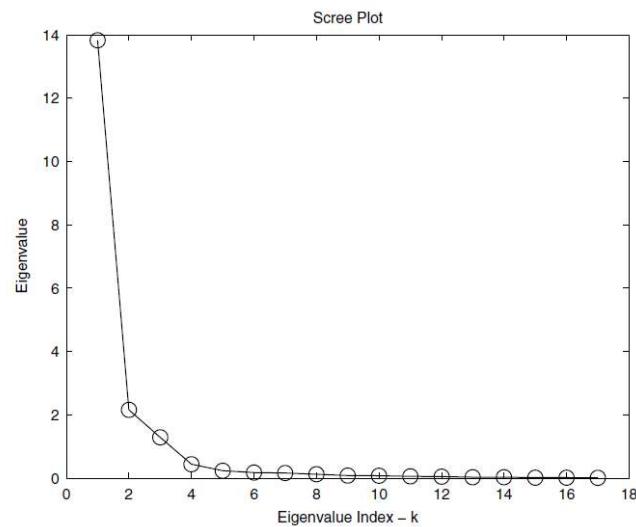


FIGURE 2.2

This is the scree plot for the **yeast** data. The elbow in the curve seems to occur at $k = 4$.

Principal Components Analysis: How Many Dimensions Should We Keep?

Size of Variance

- we would keep PCs if

$$l_k \geq 0.7\bar{l} \quad \text{or} \quad l_k \geq \bar{l},$$

where

$$\bar{l} = \frac{1}{p} \sum_{j=1}^p l_j.$$

PCA: Example

```
load yeast
[n,p] = size(data);
% Center the data.
datac = data - repmat(sum(data)/n,n,1);

% Find the covariance matrix.
covm = cov(datac);

[eigvec,eigval] = eig(covm);
eigval = diag(eigval); % Extract the
diagonal elements
% Order in descending order
eigval = flipud(eigval);
eigvec = eigvec(:,p:-1:1);
% Do a scree plot.
figure, plot(1:length(eigval),eigval,'ko-')
title('Scree Plot')
xlabel('Eigenvalue Index - k')
ylabel('Eigenvalue')
```

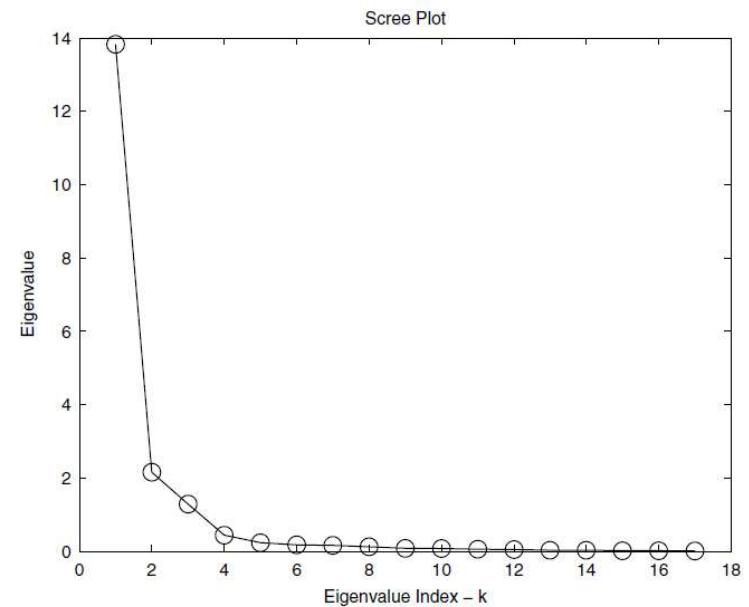


FIGURE 2.2

This is the scree plot for the **yeast** data. The elbow in the curve seems to occur at $k = 4$.

PCA: Example

```
% Now for the percentage of variance explained.  
pervar = 100*cumsum(eigval)/sum(eigval);
```

The first several values are:

```
73.5923 85.0875 91.9656 94.3217 95.5616
```

Depending on the cutoff value, we would keep four to five PCs (if we are using the higher end of the range of t_d)

```
% Now for the size of the variance.  
avgeig = mean(eigval);  
% Find the length of ind:  
ind = find(eigval > avgeig);  
length(ind)
```

According to this test, the first three PCs would be retained. So, we see that different values of d are obtained using the various procedures.

Because we want to easily visualize the data, we will use the first three PCs to reduce the dimensionality of the data, as follows.

PCA: Example

```
% Using d = 3, we will reduce the dimensionality.  
P = eigvec(:,1:3);  
Xp = datac*P;  
figure,plot3(Xp(:,1),Xp(:,2),Xp(:,3),'k*')  
xlabel('PC 1'),ylabel('PC 2'),zlabel('PC 3')
```

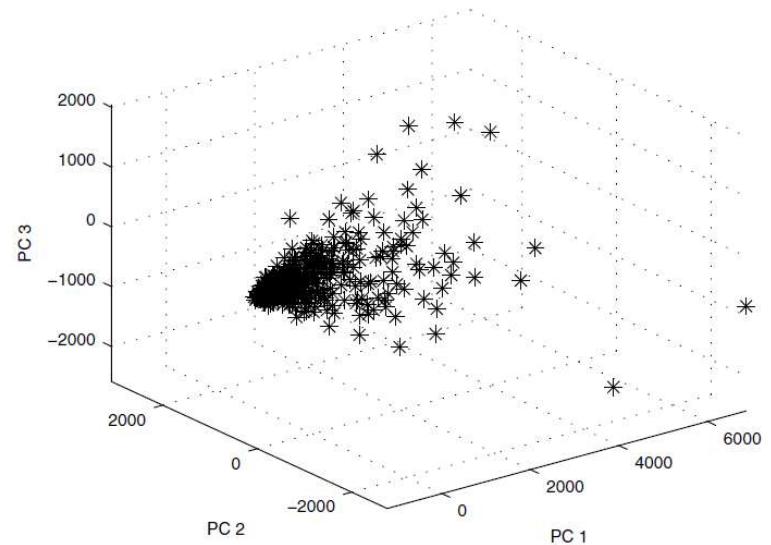


FIGURE 2.3

This shows the results of projecting the **yeast** data onto the first three PCs.

**Thank You
&
Questions**