CO902
Supporting Lecture Notes:
Principal Components Analysis Redux
Lecture 6 — 11 Feb 2013

Due to some inconsistencies in notations and one derivation malfunction, this handout summaries the key results for Principal Components Analysis (PCA) as a data reduction technique.

1 Preliminaries

First some notation and key results from linear algebra. Breaking from the slides and my own boardwork, here I will use bold captial Greek letters for matricies (e.g. $n \times n$ matrix $\mathbf{A}$) and bold lower case Greek letters for vectors (e.g. n-vector $\mathbf{u}$).

Eigendecomposition of a square matrix. For a $n \times n$ matrix $\mathbf{A}$, a length-n column vector $\mathbf{u}$ and scalar λ that satisfy

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$$

are an eigenvector-eigenvalue pair. For real symmetric $\mathbf{A}$ (the only kind we're concerned with) the eigenvectors can be choosen to be real, orthogonal and to have unit length, i.e. $\mathbf{u}_j^\top \mathbf{u}_{j'} = 1$ for $j \neq j'$ and $\mathbf{u}_j^\top \mathbf{u}_j = 1$. Collecting all n eigenvectors into a $n \times n$ matrix

$$\mathbf{U} = [\,\mathbf{u}_1\,\mathbf{u}_2\,\cdots\,\mathbf{u}_n\,],$$

and putting the corresponding eigenvalues into a $n \times n$ diagonal matrix $\mathbf{\Lambda}$ gives

$$\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n).$$

The set of eigenvectors and eigenvalues then gives

$$\mathbf{A}\mathbf{U} = \mathbf{U}\mathbf{\Lambda}.$$

The orthogonality and unit length[1] means that $\mathbf{U}$ is orthonormal and $\mathbf{U}^\top\mathbf{U} = \mathbf{U}\mathbf{U}^\top = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Premultiplying by $\mathbf{U}^\top$ shows that

$$\mathbf{U}^\top\mathbf{A}\mathbf{U} = \mathbf{\Lambda}.$$

the matrix $\mathbf{A}$ can be diagonalized with its eigenvectors, and postmultiplying by $\mathbf{U}^\top$ shows

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top = \sum_{i=1}^{n} \lambda_i \mathbf{u}\mathbf{u}^\top \qquad (1)$$

which explicitly shows that $\mathbf{A}$ is the sum of n rank-1 matrices.

[1] And an assumption of non-repeated eigenvalues, which is going to be the case for real data.

2 Unsupervised Learning - Notation

Let the data of interest be length-d column vectors x_i, $i = 1, \ldots, n$, assembed into a $d \times n$ data matrix $\mathbf{X}$,

$$\mathbf{X} = [\,\mathbf{x}_1\,\mathbf{x}_2\,\cdots\,\mathbf{x}_n\,].$$

The average data vector is

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = \frac{1}{n}\mathbf{X1},$$

where $\mathbf{1}$ is a length-n column vector of ones, and

$$\mathbf{X} - \bar{\mathbf{X}} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \mathbf{x}_2 - \bar{\mathbf{x}}, \ldots, \mathbf{x}_n - \bar{\mathbf{x}}] = \mathbf{X} - \frac{1}{n}\mathbf{X11}^\top$$

is the centered data matrix. The $d \times d$ sample covariance matrix of $\mathbf{X}$ is

$$\mathbf{S} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top = \frac{1}{n}(\mathbf{X} - \bar{\mathbf{X}})(\mathbf{X} - \bar{\mathbf{X}})^\top.$$

(Please make sure you're comfortable with all these alternative ways of writing the same expression!)

3 Principal Components Analysis - The crux of it

PCA amounts to finding a $k \times d$ matrix $\mathbf{U}$ such that the data projected into k dimensions,

$$\mathbf{y}_i = \mathbf{U}^\top\mathbf{x}_i,$$

is as "variable" as possible[2]. Here, variability is defined by the average squared L2 norm of $\mathbf{y}_i$ from its mean, or, equivalently, as the sum of variances in each of the d dimensions.

Writing the $k \times n$ reduced data matrix

$$\mathbf{Y} = [\,\mathbf{y}_1\,\mathbf{y}_2\,\cdots\,\mathbf{y}_n\,].$$

and its mean

$$\bar{\mathbf{y}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{y}_i = \frac{1}{n}\mathbf{Y1},$$

this aforementioned variance is

$$\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{y}_i - \bar{\mathbf{y}}\|^2 = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}})^\top(\mathbf{y}_i - \bar{\mathbf{y}}) = \frac{1}{n}\mathrm{tr}\left((\mathbf{Y} - \bar{\mathbf{Y}})(\mathbf{Y} - \bar{\mathbf{Y}})^\top\right).$$

Using the tricks shown in the class notes, you can show that the k dimensional transform $\mathbf{U}$ that maximizes this variance are the *first k eigenvectors* of $\mathbf{S}$, where eigenvectors are sorted by eigenvalues, largest to smallest. See, also, the class notes for the derivation of how this same choice of $\mathbf{U}$ minimizes the approximation error for a k dimensional basis set.

[2]Note that $\mathbf{y}_i$ no longer refers to "outcome" or response, as it did with supervised learning.

4 Principal Components Analysis - Useful Observations

The class notes show that the optimized variance measure for a k-dimensional PCA is

$$\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{y}_i-\bar{\mathbf{y}}\|^2=\sum_{j=1}^{k}\lambda_j.$$

That is, the contribution of the variance of the jth compontent is precisely λ_j. This tells us the about the relative importance of each component.

Eigenvalue plots. To understand the relative contribution of each component $\mathbf{u}_j$, we make plots of the ordered eigenvalues λ_j (largest to smallest) and their cumulative variance. The eigenvalue plot (sometimes called a "scree plot") shows how much variance explained by each component. The cumulative variance plot, a graph of

$$\sum_{j=1}^{k}\lambda_j\Big/\sum_{j=1}^{d}\lambda_j$$

versus k, shows the proportion of variance is explained by the first k components. This plot is useful when thinking of PCA as a k-dimensional representation of the full d dimensional data.

PCA as a transformation? What if we choose $k=d$, i.e., made a full-rank transoformation, no data reduction. Then all we've done is change coordinates. What is the $d\times d$ sample covariance matrix of this $\mathbf{Y}$? Using the fact $\bar{\mathbf{Y}}=\mathbf{Y}\mathbf{1}\mathbf{1}^\top/n=\mathbf{U}^\top\mathbf{X}\mathbf{1}\mathbf{1}^\top/n=\mathbf{U}^\top\bar{\mathbf{X}}$,

$$\begin{aligned}\frac{1}{n}(\mathbf{Y}-\bar{\mathbf{Y}})(\mathbf{Y}-\bar{\mathbf{Y}})^\top &= \frac{1}{n}(\mathbf{U}^\top\mathbf{X}-\mathbf{U}^\top\bar{\mathbf{X}})(\mathbf{U}^\top\mathbf{X}-\mathbf{U}^\top\bar{\mathbf{X}})^\top\\ &= \frac{1}{n}\mathbf{U}^\top(\mathbf{X}-\bar{\mathbf{X}})(\mathbf{X}-\bar{\mathbf{X}})^\top\mathbf{U}\\ &= \mathbf{U}^\top\mathbf{S}\mathbf{U}.\\ &= \mathbf{\Lambda},\end{aligned}$$

where, since $k=d$, $\mathbf{\Lambda}$ is the $d\times d$ diagonal matrix of eigenvectors, using result (1) above. This shows that when you use a full rank transofmation you "diagonalise" or "whiten" the data, as the covariances beteween the $k=d$ variables are now zero.

Approximating individual cases. While usually our interest will be working with the transformed k-dimensional data $\{\mathbf{y}_i\}$, we may want to see what we've lost, i.e. how close the PCA is approximating the data $\{\mathbf{x}_i\}$. The class notes show that data for the i-th case can be approximated

$$\hat{\mathbf{x}}_i=\bar{\mathbf{x}}+\sum_{j=1}^{k}\alpha_{ij}\mathbf{u}_j$$

where α_{ij} are the approximating coefficents for case i basis element j,

$$\alpha_{ij}=\mathbf{u}_j^\top\left(\mathbf{x}_i-\bar{\mathbf{x}}\right).$$

Putting these two together in "matrix mode" gives the approximation for all n cases

$$\hat{\mathbf{X}}=\bar{\mathbf{X}}+\mathbf{U}\mathbf{U}^\top(\mathbf{X}-\bar{\mathbf{X}})$$

5 PCA via SVD

The two "take away" messages about PCA are (1) it is the best k-dimensional approximation ($k \times n$ $\mathbf{Y}$) to d dimensional dataset ($d \times n$ $\mathbf{X}$), and (2) it is obtained through the eigenvectors of the $d \times d$ sample covariance matrix of $\mathbf{X}$,

$$\mathbf{S} = \frac{1}{n}(\mathbf{X} - \bar{\mathbf{X}})(\mathbf{X} - \bar{\mathbf{X}})^\top.$$

Notice that we haven't made any comment about whether $\mathbf{S}$ is invertable? That's because it doesn't matter; we never need to invert $\mathbf{S}$ and if $d \geq n$ $\mathbf{S}$ won't be full rank and one or more eigenvalues will be zero. No big deal... those would be the last eigenvectors to consider in an approximation anyway.

A practical issue arises, however, if $d \gg n$, particularlly when n is very small (10-20) and d is very large 100 or more. In that case, we are forming a gigantic $d \times d$ matrix $\mathbf{S}$ when we know the rank can be no more than $n-1$ (not n, due to the centering).

This is where a Singular Value Decomposition (SVD) comes in handy. Often people are sloppy and interchangeably refer to SVD and PCA, which is wrong. A SVD is a factorisation of an arbitrary (not necessarily square) matrix; for $m \times n$ matrix $\mathbf{A}$, the SVD is

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

where

U: Eigenvectors of $\mathbf{A}\mathbf{A}^\top$,

V: Eigenvectors of $\mathbf{A}^\top\mathbf{A}$, and

Σ: Diagonal matrix, common eigenvalues of $\mathbf{A}\mathbf{A}^\top$ & $\mathbf{A}^\top\mathbf{A}$ *squared.*

So, now consider the SVD of the centered data matrix $\mathbf{X} - \bar{\mathbf{X}}$,

$$\mathbf{X} - \bar{\mathbf{X}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

Plug this into the sample variance:

$$\begin{aligned}
\mathbf{S} &= \frac{1}{n}(\mathbf{X} - \bar{\mathbf{X}})(\mathbf{X} - \bar{\mathbf{X}})^\top \\
&= \frac{1}{n}(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top)(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top)^\top \\
&= \frac{1}{n}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top \\
&= \mathbf{U}\left(\frac{1}{n}\mathbf{\Sigma}^2\right)\mathbf{U}^\top.
\end{aligned}$$

This shows that the the $\mathbf{U}$ from the SVD of $\mathbf{X} - \bar{\mathbf{X}}$ is exactly the eigenvectors of $\mathbf{S}$, and the eigenvalues of $\mathbf{S}$ relate to the SVD as per

$$\mathbf{\Lambda}_{jj} = \mathbf{\Sigma}^2_{jj}/n.$$

For PCA, this means instead of computing $\mathbf{S}$, we can just submit $\mathbf{X} - \bar{\mathbf{X}}$ to a SVD *as long* as we get the details right:

1. Compute centered data matrix, $\mathbf{X} - \bar{\mathbf{X}}$;
2. Compute SVD of $\mathbf{X} - \bar{\mathbf{X}}$; only $\mathbf{U}$ and $\mathbf{\Sigma}$ are needed; then
3. $\mathbf{U}$ are the eigenvectors of $\mathbf{S}$, and
4. $\text{diag}(\mathbf{\Sigma}^2)/n$ are the eigenvalues of $\mathbf{S}$, $\text{diag}(\mathbf{\Lambda})$

In matlab-ese, this means you can do either

```
[U, D] = eig(S);
```

or

```
[U, Dc] = svd(Xcenter); D=Dc.^2/n;
```

Watch out! The function `eig` sorts the eigenvalues from smallest to largest, while `svd` sorts them largest to smallest.