An Introduction To Principal Component Analysis–STAT3690

Mohammad Jafari Jozani

Januray 7, 2021

Contents

1 Principal Component Analysis

Principal Component Analysis (PCA) is a multivariate statistical procedure concerned with representing the covariance structure of a set of p variables through a small number of their linear combinations.

Pearson(1901) and Hotelling (1933) defined principal components as a sequence of projections of the data, mutually uncorrelated and ordered in variance. This approach is concerned with explaining the variance-covariance structure through a few *linear combinations* of the original variables. Although p components are required to reproduce the total system variability, often much of this variability can be accounted by a small number $q < p$ of the principal components. If so, then there is almost as much information in q principle components as there is in original data with p variables. The q principle components can then replace the initial p variables, and the original data set which was consisting of n measurements on p variables is reduced to n measurements on q principal components. The idea is to make sure these linear projections keep *interesting* information in the original data. This is based on the assumption that the useful and interesting information in data is related to its variability

In general PCA is done for a variety of reasons. PCA is often the first step in an analysis to be followed by visualization, clustering, regression, classification, etc. One can perhaps consider its use in one of the following ways:

1. **Dimension reduction:** This is the task of transforming our data set to one with less features. These new features could be some of the old features or some linear or nonlinear combinations of old features.

(a) One way of dimension reduction is through variable selection (known as *feature selection*).

(b) Another way is to create a set of linear or nonlinear transformation of the inputs, using projection method (known as *feature extraction*). We want this transformation to preserve the main structure that is present in the feature space. PCA can be thought of as a method which provides a specific set of projections which represent a given data set in fewer dimensions. This has obvious advantages when it is possible to reduce dimensionality to two or three as visualization becomes very straightforward but it should be acknowledged that reduced dimensionality has advantages beyond visualization. Note that:

2. **Decorrelating data:** PCA is often used to *decorrelate the data*, i.e., transform correlated variables into uncorrelated ones (to sphere the data). Whilst univariate data can be standardized by centering and scaling, in a multivariate context one may also wish to "standardize" the correlation between variables to zero.

3. **Directions with most or least variability:** PCA can be used to find linear combinations of data which have relatively large (or relatively small) variability.

4. **Identifying important features:** PCA can be used to discover important features of the data. This can be done using graphical displays of the PC scores.

5. **Outlier detection:** PCA can be used to identify outliers, usual data points and clusters in the data points. The first few PC scores can reveal whether most of the data actually live on a linear subspace of the sample space. The last few PC scores show those linear linear projections of the original data that have the smallest variance. Any PC with zero or nearly zero variance is virtually a constant and hence can be used to detect collinearity as well as outliers that pop-up and alert the perceived dimensionality of the data.

6. **Interpreting data:** PCA can be used for interpretation of our data set as well. An analysis of principle components often reveals relationships that were not previously suspected and thereby allows interpretations that would not originally result.

As we will see shortly, the eigenvalues and eigenvectors of the covariance (correlation) matrix are the essence of a PCA. The eigenvectors determine the directions of maximum variabilities and the eigenvalues specify the variances. When the first few eigenvalues are much larger than the rest, most of the total variance can be "explained" in fewer than p dimension.

Remark 1. *We have said so many good things about PCA. However, PCA has its own problems. Emphasis on variance is where the weakness of PCA lies in:*

1. PCs depend heavily on measurement scaless. Where the data matrix contains measurements of vastly differing orders of magnitude, the PC will be greatly biased in the direction of larger measurement. It is therefore recommended to calculate PCs from correlation matrices instead of covariance matrices.

2. Robustness to outliers is also an issue. As we know, variance is affected by outliers therefore PCs can highly be affected by outliers.

3. Although PCs are uncorrelated, scatterplots sometimes reveal structures that are not necessarily revealed by linear combinations.

2 Projection onto unit vectors

Projection plays an important role in the development of PCA. For a unit norm vector $\mathbf{v} \in \mathbb{R}^p$ with $||\mathbf{v}||_2^2 = \mathbf{v}^\top \mathbf{v} = 1$, the projection of $\mathbf{x} \in \mathbb{R}^p$ onto $\mathbf{v}$ is

$$\text{Proj}_{\mathbf{v}}^{\mathbf{x}} = \frac{\mathbf{x}^\top \mathbf{v}}{\mathbf{v}^\top \mathbf{v}} \mathbf{v} = (\mathbf{x}^\top \mathbf{v})\mathbf{v} = c \cdot \mathbf{v},$$

where $c = \mathbf{x}^\top \mathbf{v}$ is like a score or a coefficient. For a matrix $\mathbf{X}_{n \times p} \in \mathbb{R}^{n \times p}$ representing a data set consisting on n observations of p variables, by projecting each row $\mathbf{x}_i \in \mathbb{R}^p$ onto $\mathbf{v}$ we get

$$\mathbf{X}\mathbf{v} = \begin{bmatrix} \mathbf{x}_1^\top \mathbf{v} \\ \mathbf{x}_2^\top \mathbf{v} \\ \vdots \\ \mathbf{x}_n^\top \mathbf{v} \end{bmatrix} \in \mathbb{R}^n$$

as the scores and the rows of $\mathbb{X}\mathbf{v}\mathbf{v}^\top \in \mathbb{R}^{n \times p}$ are the projected vectors. Note that $\mathbf{x}_j = [x_{j1}, \ldots, x_{jp}]^\top$.

As an example, below we have generated 100 observations in a 2-dimensional space and we would like to project them onto 3 directions

$$v_1 = (\frac{0.5}{\sqrt{1.25}}, \frac{1}{\sqrt{1.25}}), \quad v_2 = (\frac{1}{\sqrt{1.25}}, \frac{0.5}{\sqrt{1.25}}) \quad \text{and} \quad v_3 = (1, 0),$$

respectively.

```
set.seed(2021)
X <- matrix(rnorm(100, 0, 0.7), ncol=2)
v1 <- c(0.5/sqrt(1.25), 1/sqrt(1.25))
v2 <- c(1/sqrt(1.25), 0.5/sqrt(1.25))
v3 <-  c(1, 0)
plot(X, pch=20, xlab=expression(X[1]), ylab= expression(X[2]))
arrows(0, 0,0.5/sqrt(1.25), 1/sqrt(1.25), 0.1, lwd=2, col="darkgreen")
arrows(0, 0, 1/sqrt(1.25),0.5/sqrt(1.25), 0.1, lwd=2, col="red")
arrows(0, 0,1, 0, 0.1, lwd=2, col="blue")
text(0.65,0.10, expression(v[3]) )
text(0.65,0.45, expression(v[2]) )
text(0.25,0.85, expression(v[1]) )
```

After projecting the points onto each direction, we have depicted the projected points and you can see that not all projections are equal in terms of capturing the interesting feature of the data set in terms of its variability.

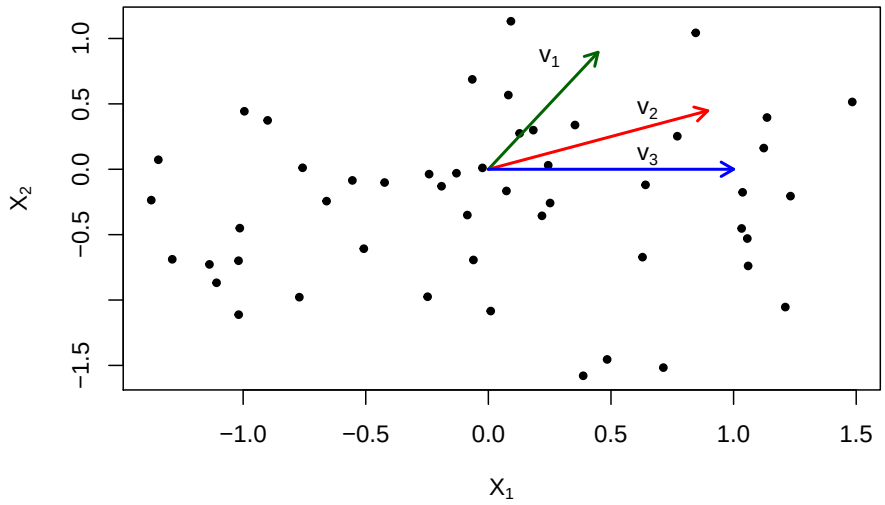

Figure 1: Scatterplot of 100 observations with 3 directions to project them onto.

```
par(mfrow=c(1, 3))
score1 <- X%*%v1
score2 <- X%*%v2
score3 <- X%*%v3

y1 <-score1%*%t(v1)
plot(y1, ylim=c(-1.5, 1.5), xlim=c(-1.5, 1.5),
     pch=10, col="darkgreen", xlab="", ylab="", main="Projected onto v1")

y2 <-score2%*%t(v2)
plot(y2, ylim=c(-1.5, 1.5), xlim=c(-1.5, 1.5),
     pch=5, col="red", xlab="", ylab="", main="Projected onto v2")

y3 <-score3%*%t(v3)
plot(y3, ylim=c(-1.5, 1.5), xlim=c(-1.5, 1.5),
     pch=8, col="blue", xlab="", ylab="", main="Projected onto v3")
```

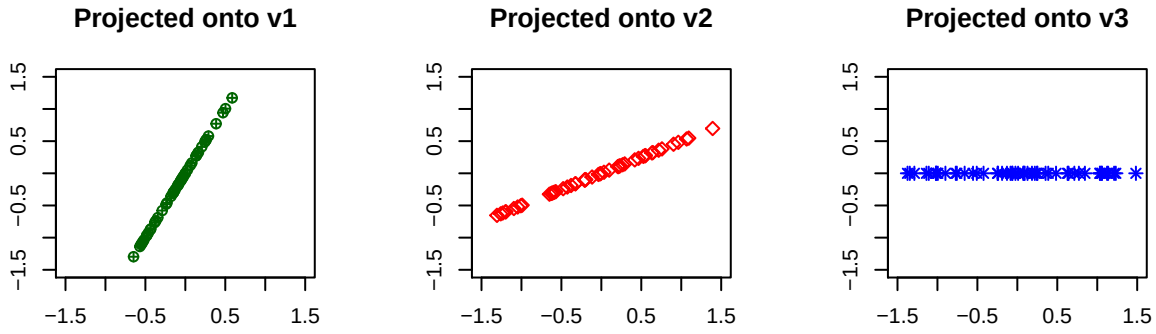

Figure 2: Projected points onto 3 directiosn depicted by different colours representinng directions

Also, the projection of $\mathbf{x} \in \mathbb{R}^p$ onto the spaces spanned by orthonormal vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{R}^p$ is given by

$$\sum_{j=1}^{k} (\mathbf{x}^\top \mathbf{v}_j) \times \mathbf{v}_j = \sum_{j=1}^{k} \text{score}_j \times \mathbf{v}_j.$$

As a generalization, consider a data matrix $\mathbb{X} \in \mathbb{R}^{n\times p}$ and suppose we want to project $\mathbf{X}$ onto columns of a matrix $\mathbf{V} \in \mathbb{R}^{p\times k}$ consisting of columns $\mathbf{v}_j \in \mathbb{R}^p$, $j = 1, \ldots, k$. The scores are given by $\mathbf{XV} \in \mathbb{R}^{n\times k}$ with jth column $\mathbf{X}v_j = (\mathbf{x}_1^\top \mathbf{v}_j, \ldots, \mathbf{x}_n^\top \mathbf{v}_j)^\top \in \mathbb{R}^n$ and the projections are the rows of $\mathbf{XVV}^\top \in \mathbb{R}^{n\times p}$.

As an example suppose we have the following 2000 observations in a 3 dimensional space. We would like to project these points onto 3 spaces created by horizontal and vertical planes denoted by xy, yx and xz, respectively. This is done below and after projecting the points onto the spaces created by two vectors we have depicted the projected points and you can easily see again not all projections capture the most interesting features in our data set.

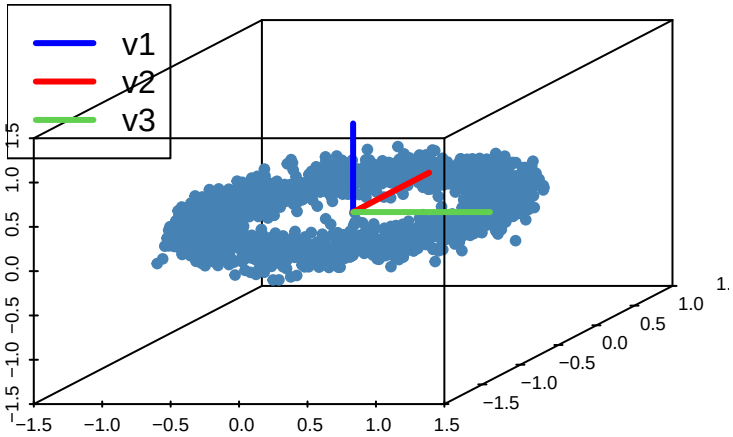

Figure 3: Scatterplot of 2000 points in a three dimensional space.

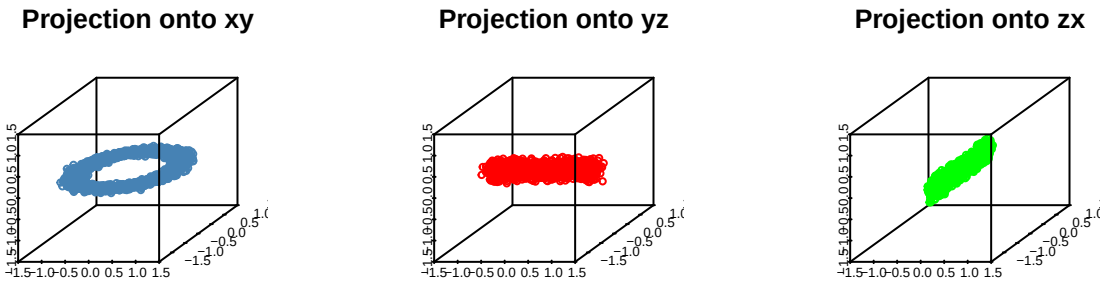

Figure 4: Projections of point in 3D onto 2D spaces.

In what follows we would like to find a way to project our data onto directions that capture the most variability among our data set.

3 Population principal components and PC scores

As we mentioned earlier, there are a number of ways in which one can reduce the dimensionality. Suppose we are dealing with a p-dimensional space representing the sample space of a p-variate random vector $\mathbf{X}_{p\times 1} = (X_1, \ldots, X_p)^\top$. In practice we will observe data (say n observations) from this space that can be represented in terms of a matrix of $n \times p$ dimension that can be used to estimate the variance-covariance structure of those variable in the p-dimensional space. But for now, suppose we have access to the true (population) variance-covariance structure of these random variables in the p-dimensional space and denote it by $\Sigma_{\mathbf{X}}$. In many practical problems p is often very large and we would like to move from a large p-dimensional space into another space that has a lower dimension q with $q < p$. There are different ways to find such a q-dimensional space. To do this, we need to find $\mathbf{W}_{q\times 1} = (W_1, \ldots, W_q)^\top$ such that the new

space has as much (or almost as much) information as the original space associated with $\mathbf{X}_{p\times 1}$, which we define it as follows

$$\begin{aligned}\text{trace}(\Sigma_{\mathbf{X}}) &= \text{Total variance of } \{X_1, \ldots, X_p\} \\ &= \sigma_1^2 + \cdots + \sigma_p^2 \\ &\approx \text{Total variance of } \{W_1, \ldots, W_q\} \\ &= \text{trace}(\Sigma_{\mathbf{W}}),\end{aligned}$$

where $\sigma_i^2 = Var(X_i)$, $i = 1, \ldots, p$.

On might want to finding this q-dimensional space such that its variance structure is very similar to the variance structure of the original space. In other words find $\mathbf{W}_{q\times 1}$ by forming linear combinations of the original random vector $\mathbf{X}$, i.e., $\mathbf{W}_{q\times 1} = A_{q\times p}^{\top}\mathbf{X}_{p\times 1}$ and construct the coefficient matrix $A_{q\times p}$ such that $\text{trace}(\Sigma_{\mathbf{X}}) \approx \text{trace}(\Sigma_{\mathbf{W}})$ and $\Sigma_{\mathbf{W}}$ is diagonal.

In PCA, the above mentioned problem is solved in a more general way. First, the original space associated with $\mathbf{X}_{p\times 1}$ is rotated by generating a new p-dimensional space consisting of linear combinations (called principal components) of $\mathbf{X}$ such that the coordinates of the new space are orthogonal (i.e., uncorrelated) and have a diagonal variance covariance matrix. Also, PCs are obtained such that they are ordered in terms of their variances. These linear combinations represent a new orthogonal coordinate system with p axes. The new axes represent the directions with maximum variability and provide a simpler and more parsimonious description of the variance-covariance structure. As coordinates in the new system are ordered based on their variability, one might decide to drop the last few ones and work with a system consisting of only q coordinates $q < p$ and still make sure that much of the variability in the original space is kept in this lower dimensional space. We now formulate the problem mathematically and show how PCs are developed.

Let $\mathbf{X}^{\top} = (X_1, \ldots, X_p)$ be a random vector having the covariance matrix $\Sigma := \Sigma_{\mathbf{X}}$ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$.

Exercise 1. *Show that any covariance matrix Σ is non-negative definite with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$.*

Consider the following linear combinations, where $a_i = (a_{i1}, \ldots, a_{ip})^{\top}$:

$$\begin{aligned}W_1 &= a_1^{\top}\mathbf{X} = a_{11}X_1 + a_{12}X_2 + \ldots + a_{1p}X_p, \\ W_2 &= a_2^{\top}\mathbf{X} = a_{21}X_1 + a_{22}X_2 + \ldots + a_{2p}X_p, \\ &\vdots \\ W_p &= a_p^{\top}\mathbf{X} = a_{p1}X_1 + a_{p2}X_2 + \ldots + a_{pp}X_p,\end{aligned}$$

with

$$Var(W_i) = a_i^{\top}\Sigma a_i, \quad i = 1, 2, \ldots, p,$$

and

$$Cov(W_i, W_j) = a_i^{\top}\Sigma a_j, \quad i, j = 1, 2, \ldots, p.$$

Principal components are uncorrelated linear combinations $W_1, \ldots, W_p$ whose variances are as large as possible.

- The first principal component is $W_1 = a_1^{\top}\mathbf{X}$ with maximum variance. So, one needs to choose $a_1^{\top} = (a_{11}, a_{12}, \ldots, a_{1p})$ such that $Var(W_1) = a_1^{\top}\Sigma a_1$ is maximized. As $a_1^{\top}\Sigma a_1$ can be maximized arbitrarily by multiplying $a_1^{\top}$ with some positive constant, it is convenient to restrict our attention to coefficient vectors $a_1^{\top}$ such that it is a unit vector (that is has length one, $a_1^{\top}a_1 = ||a||_2^2 = 1$). So, finding W_1 reduces to finding $a_1^{\top}$ such that $Var(W_1) = a_1^{\top}\Sigma a_1$ is maximized subject to $a_1^{\top}a_1 = 1$.

- We would like to find the second PC such that its variance is maximized but it is also uncorrelated with W_1. In other words, we would like to construct $W_2 = a_2^\top \mathbf{X}$ and select $a_2^\top = (a_{21}, a_{22}, \ldots, a_{2p})$ such that $Var(W_2) = a_2^\top \Sigma a_2$ is maximized subject to $a_2^\top a_2 = 1$ and $Cov(W_1, W_2) = a_1^\top \Sigma a_2 = 0$. Note that we are essentially finding a direction W_2 orthogonal to W_1 with the highest variance. This is done as we have already explained the variance in $\mathbf{X}$ along W_1, and now we want to look at variance in a different direction. Any direction not orthogonal to W_1 would necessarily have some overlap with W_1 and creates some redundancy in explaining the variance in $\mathbf{X}$.
- In general, to find the ith PC, $W_i = a_i^\top \mathbf{X}$, we would like to choose $a_i^\top = (a_{i1}, a_{i2}, \ldots, a_{ip})$ such that $Var(W_i) = a_i^\top \Sigma a_i$ is maximized subject to $a_i^\top a_i = 1$ and $Cov(W_i, W_j) = a_i^\top \Sigma a_j = 0$, $j = 1, \ldots, i-1$.

Theorem 1. *Suppose $Cov(\boldsymbol{X}_{p\times 1}) = \Sigma$ where $\boldsymbol{X}^\top = (X_1, \ldots, X_p)$. Let $(\lambda_1, \boldsymbol{e}_1), (\lambda_2, \boldsymbol{e}_2), \ldots, (\lambda_p, \boldsymbol{e}_p)$ be the eigenvalues and eignevectors of Σ such that $\lambda_1 \geq \lambda \geq \cdots \lambda_p \geq 0$. Then, the ith PC associated with $\boldsymbol{X}$ is given by*

$$W_i = \boldsymbol{e}_i^\top \boldsymbol{X} = e_{i1}X_1 + \ldots + e_{ip}X_p, \quad i = 1, 2, \ldots, p.$$

Also, for this choice of coefficients

$$Var(W_i) = \boldsymbol{e}_i^\top \Sigma \boldsymbol{e}_i = \lambda_i, \quad i = 1, 2, \ldots, p,$$

and

$$Cov(W_i, W_j) = \boldsymbol{e}_i^\top \Sigma \boldsymbol{e}_j = 0, \quad i \neq j.$$

Proof: Using the definition of PCs, we first need to obtain $W_1 = a_1^\top \mathbf{X}$ with $a_1^\top a_1 = 1$ and the goal is to

$$\max_{a_1} \; a_1^\top \Sigma a_1 = \max_{a_1} \; \frac{a_1^\top \Sigma a_1}{a_1^\top a_1}.$$

However, as we showed earlier in this course, this maximum happens at the first eigenvalue λ_1 which is attained at $\hat{a}_1 = \mathbf{e}_1$. Also, note that $\mathbf{e}_1^\top \mathbf{e}_1 = 1$ since eigenvectors are normalized. Thus

$$\max_{a} \; \frac{a_1^\top \Sigma a_1}{a_1^\top a_1} = \lambda_1 = \frac{\mathbf{e}_1^\top \Sigma \mathbf{e}_1}{\mathbf{e}_i^\top \mathbf{e}_1} = \mathbf{e}_1^\top \Sigma \mathbf{e}_1 = Var(W_1) = Var(\mathbf{e}_1^\top \mathbf{X})$$

Similarly, to find the ith PC we need to obtain $W_i = a_i^\top \mathbf{X}$ such that $a_i^\top a_i = 1$ and $a_i \perp a_j$, for $j = 1, 2, \ldots, i-1$. That is, finding a_i such through

$$\max_{a_i \perp a_i, \ldots, a_{i-1}} \; a_i^\top \Sigma a_i = \max_{a_i \perp a_i, \ldots, a_{i-1}} \; \frac{a_i^\top \Sigma a_i}{a_i^\top a_i}.$$

As we saw in Chapter 3:

$$\max_{a_i \perp a_i, \ldots, a_{i-1}} \; \frac{a_i^\top \Sigma a_i}{a_i^\top a_i} = \lambda_i, \quad i = 2, \ldots, p.$$

For the choice $a = \mathbf{e}_i$ where $\mathbf{e}_i^\top \mathbf{e}_j = 0$, $i = 2, \ldots, p$, and $j = 1, \ldots, i-1$, we have

$$\begin{aligned} \frac{\mathbf{e}_i^\top \Sigma \mathbf{e}_i}{\mathbf{e}_i^\top \mathbf{e}_i} &= \mathbf{e}_i^\top \Sigma \mathbf{e}_i \\ &= Var(W_i) \\ &= \mathbf{e}_i^\top (\lambda_i \mathbf{e}_i) \\ &= \lambda_i \mathbf{e}_i^\top \mathbf{e}_i^\top \\ &= \lambda_i. \end{aligned}$$

It remains to show that $Cov(W_i, W_j) = 0$ for $j \neq i$. To see this note that

$$\begin{aligned} Cov(W_i, W_j) &= \mathbf{e}_i^\top \Sigma \mathbf{e}_j \\ &= \mathbf{e}_i^\top (\lambda_j \mathbf{e}_j) \\ &= \lambda_j \mathbf{e}_i^\top \mathbf{e}_j \\ &= 0, \end{aligned}$$

which completes the proof. Note that the eigenvectors of Σ are orthogonal if $\lambda_1, \ldots, \lambda_p$ are distinct. If not, eigenvectors can be chosen to be diagonal. So, principal components are uncorrelated and have variances equal to the eigenvalues of Σ.

Note that $W_i = e_{i1}X_1 + \ldots + e_{ip}X_p$ and each of these e_{ij}'s represents the importance of X_j to the ith principal component, irrespective of other variables. Also, each e_{ij} is is proportional to the correlation coefficient between W_i and X_j. To be more specific

$$e_{ij} = \frac{\sigma_j}{\sqrt{\lambda_i}} \rho_{W_i, X_j}, \quad i, j = 1, 2, \ldots, p.$$

To see this, note that $X_j = l_j^\top \mathbf{X} = (0, 0, ..., 1, 0, ..., 0)\mathbf{X}$ where l_j has 1 as its jth element and 0 otherwise. So,

$$\begin{aligned} Cov(X_j, W_i) &= Cov(l_j \mathbf{X}, \mathbf{e}_i \mathbf{X}) \\ &= l_j \Sigma \mathbf{e}_i \\ &= l_j \lambda_i \mathbf{e}_i \\ &= \lambda_i e_{ij} \end{aligned}$$

and

$$\rho_{W_i, X_j} = \frac{Cov(X_j, W_i)}{\sqrt{Var(X_j)}\sqrt{Var(W_i)}} = \frac{\lambda_i e_{ij}}{\sigma_j \sqrt{\lambda_i}},$$

which results in what we claimed. Note that ρ_{W_i, X_j}'s help to interpret the components as they only measure the univariate contribution of an individual X_j to a principal component W_i. That is, they do not indicate the importance of X_j to W_i in the presence of other X_k's. So, it is good in practice to consider both e_{ij} and ρ_{W_i, X_j}. Also, if X_js are standardized with $\sigma_j = 1$ then $\rho_{W_i, X_j} = \sqrt{\lambda_i} e_{ij}$.

Theorem 2. *Suppose $Cov(\boldsymbol{X}_{p\times 1}) = \Sigma$ where $\boldsymbol{X}^\top = (X_1, \ldots, X_p)$. Let $(\lambda_1, \boldsymbol{e}_1), (\lambda_2, \boldsymbol{e}_2), \ldots, (\lambda_p, \boldsymbol{e}_p)$ be the eigenvalues and eignevectors of Σ such that $\lambda_1 \geq \lambda \geq \cdots \lambda_p \geq 0$. Let the ith PC associated with $\boldsymbol{X}$ be given by*

$$W_i = \boldsymbol{e}_i^\top \boldsymbol{X},$$

with

$$Var(W_i) = \lambda_i.$$

Then, PC components $\boldsymbol{W}^\top = (W_1, \ldots, W_p)$ preserve the total variance in $\boldsymbol{X}^\top = (X_1, \ldots, X_p)$, i.e.:

$$trace(\Sigma) = \sum_{i=1}^{p} Var(X_i) = \sum_{i=1}^{p} Var(W_i) = \sum_{i=1}^{p} \lambda_i$$

Proof: To show this result, note that by definition

$$\text{trace}(\Sigma) = \sum_{i=1}^{p} Var(X_i).$$

Also, one can write Σ as $\Sigma = P\Lambda P^\top = \sum_{i=1}^p \lambda_i \mathbf{e}_i \mathbf{e}_i^\top$ such that $P = [\mathbf{e}_1, \ldots, \mathbf{e}_p]$ contains the eigenvectors and $PP^\top = P^\top P = \mathbb{I}$. So, as $Var(W_i) = \lambda_i$, we have

$$\text{trace}(\Sigma) = \text{trace}(P\Lambda P^\top) = \text{trace}(\Lambda PP^\top) = \text{trace}(\Lambda) = \sum_{i=1}^p \lambda_i = \sum_{i=1}^p Var(W_i),$$

and this completets the proof.

The proportion of the total variation that is explained by the ith PCs is given by

$$\frac{\lambda_i}{\sum_{j=1}^p \lambda_j}, \quad i = 1, 2, \ldots, p.$$

Similarly, the total variance explained by the first q PCs is

$$\frac{\lambda_1 + \ldots + \lambda_q}{\lambda_1 + \ldots + \lambda_p}.$$

If most (e.g., 80% or 90%) of the total population variance can be attributed to the first q PCs then these components can replace the original p variables without much loss of information. In particular, if $\frac{\lambda_1+\ldots+\lambda_q}{\lambda_1+\ldots+\lambda_p} \approx 1$, then we do not lose much by transforming the original variables into fewer new (principal component) variables.

Example: As a toy example, consider the following variance-covariance matrix associated with $\mathbf{X}^\top = (X_1, X_2, X_3)$.

```
sigma<-matrix(c(1, -2, 0, -2, 5, 0, 0, 0, 2), ncol=3, byrow=TRUE)
sigma
```

```
##      [,1] [,2] [,3]
## [1,]    1   -2    0
## [2,]   -2    5    0
## [3,]    0    0    2
```

We can calculate the principal components in R. To this end, we can simply use `eigen()`.

```
results<- eigen(sigma)
results$vectors  # gives the eigenvector(weigth/contribution of each variable to PCs)
```

```
##            [,1] [,2]      [,3]
## [1,] -0.3826834    0 0.9238795
## [2,]  0.9238795    0 0.3826834
## [3,]  0.0000000    1 0.0000000
```

```
results$values  # gives the eignevalues
```

```
## [1] 5.8284271 2.0000000 0.1715729
```

Now, PCs $\mathbf{W} = (W_1, W_2, W_3)$ associated with $\mathbf{X}$ are obtained as

$$W_i = e_{i1}X_1 + e_{i2}X_2 + e_{i3}X_3, \quad i = 1, 2, 3,$$

with $\mathbf{e}_i = (e_{i1}, e_{i2}, e_{i3})$ being the ith eigenvector associated with $Cov(\mathbf{X}) = \Sigma$.

As we see, principal components are given by:

$$\begin{aligned} W_1 &= -0.383X_1 + 0.924X_2, \\ W_2 &= X_3, \\ W_3 &= 0.924X_1 + 0.383X_2, \end{aligned}$$

with the following variances

$$\begin{aligned} Var(W_1) &= \lambda_1 = 5.8284271, \\ Var(W_2) &= \lambda_2 = 2.0000000, \\ Var(W_3) &= \lambda_3 = 0.1715729. \end{aligned}$$

It is easily seen that $\text{trace}(\Sigma) = 8 = 5.8284271 + 2 + 0.1715729 = 8 = \lambda_1 + \lambda_2 + \lambda_3$. Also, it can be seen that Y_1 explains about 0.7285534%, and (W_1, W_2) about 0.9785534% of the variations in $\mathbf{X}$.

4 Centered and normalized PCs

Centered PCs are used to make PCs centered around zero. To this end, we first center the $\mathbf{X}$ using $\boldsymbol{\mu}$ and define

$$W_i = \mathbf{e}_i^\top (\mathbf{X} - \boldsymbol{\mu}).$$

Let

$$\boldsymbol{\rho} = \begin{bmatrix} 1, \rho_{12}, \ldots, \rho_{1p} \\ \rho_{12}, 1, \ldots, \rho_{2p} \\ \vdots, \vdots, \ddots, \vdots \\ \rho_{1p}, \rho_{2p}, \ldots, 1 \end{bmatrix},$$

represent the correlation matrix associated with $\mathbf{X}^\top = (X_1, \ldots, X_p)$ then the PCs obtained from $\boldsymbol{\rho}$ correspond to the PCs of standardized variables

$$\mathbf{Z} = \Omega^{-\frac{1}{2}}(\mathbf{X} - \boldsymbol{\mu}),$$

where

$$\Omega = \text{diag}(\sigma_1^2, \ldots, \sigma_p^2).$$

In this case, normalized PCs are obtained as

$$\begin{aligned} \mathbf{W}_i^* &= \mathbf{e}_i^{*\top} \mathbf{Z} \\ &= \mathbf{e}_i^{*\top} \Omega^{-\frac{1}{2}} (\mathbf{X} - \boldsymbol{\mu}) \end{aligned}$$

where $(\lambda_i^*, \mathbf{e}_i^*)$, $i = 1, \ldots, p$ are the eigenvalues and eigenvectors obtained from $\boldsymbol{\rho}$. In this setting we have

$$\sum_{i=1}^{p} Var(W_i^*) = p = \sum_{i=1}^{p} Var(Z_i),$$

and

$$Cov(W_i^*, Z_j) = e_{ij}^* \sqrt{\lambda_i^*}.$$

Note that the proportion of (standardized) population variance due to the ith principal component is λ_i^*/p where λ_i^* is the ith eigenvalue of $\boldsymbol{\rho}$.

Remark 2. *In general the PCs derived from the covariance matrix Σ are different from those derived from $\boldsymbol{\rho}$. Furthermore, one set of principal components is not a simple function of the others. This suggests that the standardization is not inconsequential. Variables should probably standardized if they are measured on scales with widely different ranges or if the measurement units are not common among variables.*

Example: Consider the following covariance matrix:

```
sigma <- matrix(c(1, 4, 4, 100), nrow=2, byrow=TRUE)
sigma
```

```
##      [,1] [,2]
## [1,]    1    4
## [2,]    4  100
```

with an associated correlation matrix $\boldsymbol{\rho}$:

```
rho <-  sigma/sqrt( (diag(sigma)) %*% t(diag(sigma)) )
rho
```

```
##      [,1] [,2]
## [1,]  1.0  0.4
## [2,]  0.4  1.0
```

The eigenvalues and eigen vectors of Σ are given by

```
eigen(sigma)
```

```
## eigen() decomposition
## $values
## [1] 100.1613532   0.8386468
##
## $vectors
##            [,1]        [,2]
## [1,] 0.04030552 -0.99918740
## [2,] 0.99918740  0.04030552
```

which results in PCs of Σ to be

$$W_1 = 0.040X_1 + 0.999X_2$$
$$W_2 = -0.999X_1 - 0.040X_2$$

Due to its large variance, we observe that X_2 completely dominates the first PC determined from Σ. This PC also explains $\frac{\lambda_1}{\lambda_1+\lambda_2} = 0.992$ of the total population variance. Based on $\boldsymbol{\rho}$, one can easily get the following eigenvalues and eigenvectors

```r
eigen(rho)
```

```
## eigen() decomposition
## $values
## [1] 1.4 0.6
##
## $vectors
##           [,1]       [,2]
## [1,] 0.7071068 -0.7071068
## [2,] 0.7071068  0.7071068
```

which results in

$$\begin{aligned} W_1^* &= 0.707Z_1 + 0.707Z_2 \\ &= 0.707(\frac{X_1 - \mu_1}{\sigma_1}) + 0.707(\frac{X_2 - \mu_2}{\sigma_2}) \\ &= 0.707(X_1 - \mu_1) + 0.0707(X_2 - \mu_2) \end{aligned}$$

and

$$W_2^* = -0.707(X_1 - \mu_1) + 0.0707(X_2 - \mu_2)$$

Here, we see that when X_1 and X_2 are both standardized the resulting variables equally contribute to the PCs determined from $\boldsymbol{\rho}$. Here, the first PC explains around $\frac{\lambda_1^*}{p} = \frac{1.4}{2} = 0.7$ of the total standardizes population variance.

We can also calculate the correlation between original variables and the PCs. These for PCs based on Σ are given by

```r
cor.WX <- function(mat){
    vals= mat
    rownames(vals) = paste("X", 1:nrow(mat), sep="")
    colnames(vals) = paste("W", 1:nrow(mat), sep="")

    for(i in 1: ncol(vals)){
        val= t(eigen(mat)$vectors[,i])* sqrt(eigen(mat)$values[i])
        vals[,i] = val/sqrt(diag(mat))
        }
    return(vals)
}

cor.WX(sigma)  # gives the correlation of W_i and X_k, for i,k = 1,2
```

```
##           W1           W2
## X1 0.4033802 -0.915032462
## X2 0.9999932  0.003691085
```

while based on $\boldsymbol{\rho}$ are

```r
cor.WX(rho)    # gives the correlation of W_i and Z_k, for i,k = 1,2
```

```
##          W1         W2
## X1 0.83666 -0.5477226
## X2 0.83666  0.5477226
```

5 Sample PCA

In practice, we do not know the population variance-covariance Σ or correlation matrices $\boldsymbol{\rho}$. Instead we have a sample matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ where n is the number of observations which is obtained on p variables. Sample PCs are defined as before except we replace Σ with $\mathbf{S}$ the sample variance-covariance matrix and $\boldsymbol{\rho}$ with $\mathbf{R}$ the sample correlation coefficient matrix, respectively. Sample PCs can be very useful in summarizing the observed variability in the data set and this is indeed what most people will end up using in practice.

To be more specific, consider a random sample $\mathbf{X}_1, \ldots, \mathbf{X}_n$ from the underlying population which make a matrix of observation in $\mathbb{R}^{n\times p}$ given by

$$\mathbf{X} = \text{Data}_{n\times p} = \begin{bmatrix} X_{11}, \ldots, X_{1p} \\ X_{21}, \ldots, X_{2p} \\ \vdots, \ddots, \vdots \\ X_{n1}, \ldots, X_{np} \end{bmatrix},$$

Suppose $\mathbf{S}$ is the sample covariance matrix associated with $\mathbf{X}$, given by

$$\mathbf{S}_{p\times p} = \frac{1}{n-1}\mathbf{Z}^\top\mathbf{Z} = \frac{1}{n-1}(\mathbf{X} - \mathbf{1}^\top\bar{\mathbf{X}})^\top(\mathbf{X} - \mathbf{1}^\top\bar{\mathbf{X}}),$$

where

$$\mathbf{Z} = \begin{bmatrix} X_{11} - \bar{X}_1, & \ldots & , X_{1p} - \bar{X}_p \\ X_{21} - \bar{X}_1, & \ldots & , X_{21} - \bar{X}_p \\ \vdots, & \ddots & , \vdots \\ X_{n1} - \bar{X}_1, & \ldots & , X_{np} - \bar{X}_1 \end{bmatrix} = \mathbf{X} - \mathbf{1}^\top\bar{\mathbf{X}},$$

$\mathbf{1} \in \mathbb{R}^n$ is a vector of 1s and $\bar{\mathbf{X}} = \frac{1}{n}\mathbf{X}^\top\mathbf{1} \in \mathbb{R}^p$ is the sample mean vector. Note that $\text{trace}(\mathbf{S}) = \sum_{i=1}^p S_{ii}$ is the total sample variance, where S_{ii} is the sample variance of X_i.

Let $\widehat{\lambda}_1 \geq \ldots \geq \widehat{\lambda}_p$ be the eigenvalues of $\mathbf{S}$ and denote their corresponding eigenvectors with $\widehat{\mathbf{e}}_1, \ldots, \widehat{\mathbf{e}}_p$. Then, the ith sample principal component is given by

$$\widehat{W}_i = \widehat{\mathbf{e}}_i^\top\mathbf{X} = \widehat{e}_{i1}X_1 + \ldots + \widehat{e}_{ip}X_p.$$

Properties of sample PCs are also obtained similar to the population PCs. For example

1. Sample variance of each PC $\widehat{W}_i$ is $\widehat{\lambda}_i$.
2. Sample PCs $\widehat{W}_i$ and $\widehat{W}_j$ are uncorrelated for $i \neq j$.
3. Total sample variance is given by $\sum_{i=1}^p S_{ii} = \sum_{i=1}^p \widehat{\lambda}_i$.
4. Sample correlation between $\widehat{W}_i$ and X_j is given by $\frac{\widehat{e}_{ij}\sqrt{\widehat{\lambda}_i}}{\sqrt{S_{jj}}}$
5. Centered sample PCs are given by $\widehat{W}_i = \widehat{\mathbf{e}}_i^\top(\mathbf{X} - \mathbf{1}^\top\bar{\mathbf{X}})$.

5.1 Singular value decomposition for PCA calculation

let $\mathbf{Z}_{n\times p} = \mathbf{X} - \mathbf{1}^\top \bar{\mathbf{X}}$ be the centered data matrix. Then

$$(n-1)\mathbf{S} = \mathbf{Z}^\top \mathbf{Z}.$$

Recall the singular value decomposition (SVD) for $\mathbf{Z}_{n\times p} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$, where

1. $\mathbf{U}$ is an $n \times p$ matrix consisting of observations scores. Columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{Z}\mathbf{Z}^\top$ and form an orthogonal basis for the observation profiles, so that $\mathbf{U}\mathbf{U}^\top = \mathbf{I}_{n\times n}$.
2. $\mathbf{D}$ is a $p \times p$ matrix of singular values. It is diagonal and its first $r = \min(n, p)$ singular values are non-zero and equal to the square roots of the eigenvalues of both $\mathbf{Z}^\top\mathbf{Z}$ and $\mathbf{Z}\mathbf{Z}^\top$.
3. $\mathbf{V}^\top$ is also a $p \times p$ matrix of variable weights. Columns of $\mathbf{V}$ are eigenvectors of $\mathbf{Z}^\top\mathbf{Z}$ and form an orthonormal basis for the variables, as $\mathbf{V}^\top\mathbf{V} = \mathbf{I}_{p\times p}$. Note that columns of $\mathbf{V}$ are the $\widehat{\mathbf{e}}_i$'s from before.

Using the definition of SVD, we have

$$\begin{aligned}(n-1)\mathbf{S} &= \left(\mathbf{U}\mathbf{D}\mathbf{V}^\top\right)^\top \left(\mathbf{U}\mathbf{D}\mathbf{V}^\top\right) \\ &= \mathbf{V}\mathbf{D}^\top\mathbf{U}^\top\mathbf{U}\mathbf{D}\mathbf{V}^\top \\ &= \mathbf{V}\mathbf{D}^\top\mathbf{D}\mathbf{V}^\top \quad (\text{as } \mathbf{U}^\top\mathbf{U} = \mathbf{I}_{n\times n}) \\ &= \mathbf{V}\mathbf{D}^2\mathbf{V}^\top.\end{aligned}$$

This essentially means that the eigenvalues of $\mathbf{S}$ are the diagonal entries of $\frac{1}{n-1}\mathbf{D}^2$ and the columns of the orthogonal matrix $\mathbf{V}$ are the eigenvectors of $\mathbf{S}$.

This is a very useful in practice as finding the eigenvalues and eigenvectors of $\mathbf{S}$ when we are dealing with a very high dimensional setting is challenging. For example, suppose we have $n = 100$ images with a resolution of 180×200. Each image constitutes vector intensities of length $p = 180 \times 200 = 36000$. To find the PCs one needs to work with an $\mathbf{S}$ of dimension 36000 times 36000 and this is computationally unrealistic. In `R` a 36000×36000 matrix requires $36000 \times 36000 \times 8 = 10368000000$ bytes that needs 10.4 GB RAM memory. To resolve this issue, one can use SVD as mentioned above. Another solution is to note that when p is very large and n is relatively small then $\frac{1}{n-1}\mathbf{Z}\mathbf{Z}^\top$ is a matrix of $n \times n$ and it is much easier to work with such a matrix than $\mathbf{S}$ and one can find the eigenvalues and eigenvectors of $\frac{1}{n-1}\mathbf{Z}\mathbf{Z}^\top$ say $(\gamma_i, \mathbf{u}_i)$, $i = 1, \ldots, n$ which satisfy

$$\frac{1}{n-1}\mathbf{Z}\mathbf{Z}^\top\mathbf{u}_i = \gamma_i\mathbf{u}_i.$$

Multiplying both sides with $\mathbf{Z}^\top$ we have

$$\mathbf{Z}^\top\left(\frac{1}{n-1}\mathbf{Z}\mathbf{Z}^\top\mathbf{u}_i\right) = \gamma_i\mathbf{Z}^\top\mathbf{u}_i,$$

or, equivalently

$$\left(\frac{1}{n-1}\mathbf{Z}^\top\mathbf{Z}\right)\mathbf{Z}^\top\mathbf{u}_i = \gamma_i\mathbf{Z}^\top\mathbf{u}_i,$$

or

$$\mathbf{S}\,\mathbf{e}_i = \gamma_i\,\mathbf{e}_i.$$

In other words, the first n eigenvalues λ_i of $\mathbf{S}$ are the same as the eigenvalues of $\frac{1}{n-1}\mathbf{Z}\mathbf{Z}^\top$ and their corresponding eigenvectors are obtained as $\mathbf{e}_i = \mathbf{Z}^\top\mathbf{u}_i$.

This has a huge computational benefit in particular for the cases where the original data set is very high dimensional and the number of observations n is much less than p.

```
set.seed(10)
n <- 5
p <- 8
N <- n*p
X<- matrix(sample(1:N, N, replace=FALSE), nrow=n, byrow=TRUE)
Z <- scale(X, center=TRUE, scale=FALSE)
#Find cov(X). You could also use cov(Z) or cov(X)
S <- t(Z)%*%Z/(n-1)
#eigenvalues of the covariance matrix
eigen(S)$value
```

```
## [1] 5.091027e+02 4.518035e+02 2.679258e+02 3.166797e+01 3.410605e-13
## [6] 5.332015e-14 1.720230e-14 2.752261e-15
```

```
#eigenvectors of the covariance matrix
eigen(S)$vectors
```

```
##            [,1]          [,2]       [,3]        [,4]        [,5]        [,6]
## [1,] -0.1957053  3.228876e-01 -0.4188824  0.60488158  0.56222665  0.00000000
## [2,]  0.1471564 -5.557864e-01 -0.3958612  0.45750170 -0.41673198 -0.32776059
## [3,] -0.3668402  1.867140e-01 -0.2321313 -0.28659960 -0.09952769 -0.38530596
## [4,] -0.3714390 -5.488858e-01 -0.1028906 -0.01477059  0.12516520  0.63839328
## [5,]  0.1675007  4.726489e-01 -0.1160733  0.24333109 -0.56140927  0.54338367
## [6,] -0.1407832 -1.188520e-01  0.2638103  0.05057669  0.16138765  0.10430350
## [7,]  0.2332888  7.960938e-05 -0.7141753 -0.52828837  0.11743769  0.17041711
## [8,]  0.7518016 -1.148242e-01  0.1012167  0.03995424  0.36006335  0.03713899
##              [,7]        [,8]
## [1,]  0.000000000  0.00000000
## [2,]  0.009817105  0.14914322
## [3,]  0.716977826  0.14872211
## [4,]  0.310318609 -0.17447893
## [5,]  0.204530222  0.15357313
## [6,] -0.048751588  0.92444496
## [7,] -0.259731965  0.21481870
## [8,]  0.527142067  0.02940866
```

```
#Use SVD of Z and find the
#eigenvalues of S as diagonal elemenst of D^2/(n-1)
##Also eigen vectors as columns of V
svd(Z)$d^2/(n-1)
```

```
## [1] 5.091027e+02 4.518035e+02 2.679258e+02 3.166797e+01 3.385360e-30
```

```r
svd(S)$v
```

```
##            [,1]          [,2]       [,3]        [,4]        [,5]       [,6]
## [1,] -0.1957053 -3.228876e-01 -0.4188824 -0.60488158  0.28836871  0.1251185
## [2,]  0.1471564  5.557864e-01 -0.3958612 -0.45750170 -0.09223384  0.1547098
## [3,] -0.3668402 -1.867140e-01 -0.2321313  0.28659960  0.23322032 -0.1686299
## [4,] -0.3714390  5.488858e-01 -0.1028906  0.01477059 -0.17885347 -0.5844178
## [5,]  0.1675007 -4.726489e-01 -0.1160733 -0.24333109 -0.67585603 -0.4364944
## [6,] -0.1407832  1.188520e-01  0.2638103 -0.05057669 -0.51212729  0.5440617
## [7,]  0.2332888 -7.960938e-05 -0.7141753  0.52828837 -0.21344051  0.2181507
## [8,]  0.7518016  1.148242e-01  0.1012167 -0.03995424  0.23946511 -0.2372969
##            [,7]        [,8]
## [1,]  0.1682144  0.43473159
## [2,] -0.3470440 -0.38800886
## [3,] -0.7792768  0.06664748
## [4,]  0.1846247  0.37727163
## [5,] -0.1368159 -0.09917749
## [6,] -0.2873663  0.50303430
## [7,]  0.1992798  0.15356539
## [8,] -0.2624793  0.47667577
```

```r
#Now when n<p use ZZT=Zt(Z)/(n-1) instead  of S
# and use the first n eigenvalues as those of S
# also use the t(Z)*eigenvectors(ZZt) as eigenvectors of S
ZZt <- Z%*%t(Z)/(n-1)
eigen(ZZt)$values[1:n]
```

```
## [1] 5.091027e+02 4.518035e+02 2.679258e+02 3.166797e+01 7.958079e-13
```

```r
U<- eigen(ZZt)$vectors

t(Z)%*%U
```

```
##            [,1]          [,2]      [,3]       [,4]          [,5]
## [1,]  -8.831519 -13.726384126 13.712904  6.8078580 -4.352074e-14
## [2,]   6.640667  23.627224213 12.959259  5.1491180 -3.197442e-14
## [3,] -16.554253  -7.937462845  7.599254 -3.2256385  1.154632e-14
## [4,] -16.761783  23.333868779  3.368316 -0.1662410 -8.826273e-15
## [5,]   7.558738 -20.092937856  3.799878  2.7386575 -1.372513e-14
## [6,]  -6.353068   5.052559633 -8.636327  0.5692336  2.220446e-16
## [7,]  10.527534  -0.003384301 23.379870 -5.9458121  2.309264e-14
## [8,]  33.926258   4.881330331 -3.313518  0.4496794  8.881784e-15
```

5.2 Matrix Approximation

Let $\mathbf{X}_{n\times p}$ be an $n\times p$ data matrix such that $\text{rank}(\mathbf{X}) = r$ and a singular value decomposition of $\mathbf{X} = \mathbf{UDV}^\top$. If $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r > 0$ be the singular values of $\mathbf{X}$, then one can find a rank q approximation of $\mathbf{X}$ that minimizes

$$||\mathbf{X} - \widehat{\mathbf{X}}(q)||$$

as follows

$$\widehat{\mathbf{X}}_{n\times p}(q) = \sum_{i=1}^{q} \lambda_i \begin{bmatrix} u_{i1} \\ \vdots \\ u_{in} \end{bmatrix} \begin{bmatrix} v_{i1} & \dots v_{ip} \end{bmatrix} = \lambda_1 \mathbf{u}_1 \mathbf{v}_1^\top + \dots + \lambda_q \mathbf{u}_q \mathbf{v}_q^\top,$$

where the variance in $\mathbf{X}$ accounted for each term in the above approximation is λ_i^2. This has many applications and one of the famous one is in image compression which is left as a project for you to discover this more. Here we give an example to see how matrix approximation using SVD works:

```
mm<- matrix(
c(20.03,  2.06,  1.64,  2.71, 20.33,
 17.59,  1.76, 20.09, 19.63, 18.45,
 19.48,  1.44, 21.95, 22.01, 19.24,
 21.42,  1.40,  0.32,  0.25, 23.30,
 19.74, 20.63, 16.24,  2.22, 16.65,
 18.45, 15.88, 16.91,  2.09, 18.52,
 20.17, 18.80, 19.53,  1.88, 18.85,
 20.13,  2.54,  2.72,  2.37, 22.18), nrow=8, byrow=TRUE)

plot.image <- function(img, m = ""){
image(t(img)[, nrow(img):1], col = gray.colors(100), main = m)
}

# This calualtes X.hat(q)
mm.app.q <- function(q, im=mm){
l <- svd(im)$d
u <- svd(im)$u
v <- svd(im)$v
a.m <- matrix(0, nrow=dim(im)[1], ncol=dim(im)[2])
for(i in 1:q){
    a.m <- a.m + l[i]*(u[,i] %*% t(v[,i] ))
}
return(a.m)
}

par(mfrow=c(2, 2))
plot.image(mm, m="Original image")
plot.image(mm.app.q(1), m="Approximation with q=1")
plot.image(mm.app.q(2), m="Approximation with q=2")
plot.image(mm.app.q(3), m="Approximation with q=3")
```

6 Sampling distribution of sample eigenvalues

If $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \Sigma)$ with Σ being a positive definite variance-covariance matrix, then

$$\mathbf{W} = \mathbf{V}^\top (\mathbf{X} - \boldsymbol{\mu}) \sim N_p(\mathbf{0}, \Lambda),$$

where

$$\mathbf{V} = [\mathbf{e}_1 | \mathbf{e}_2 | \dots | \mathbf{e}_p]$$

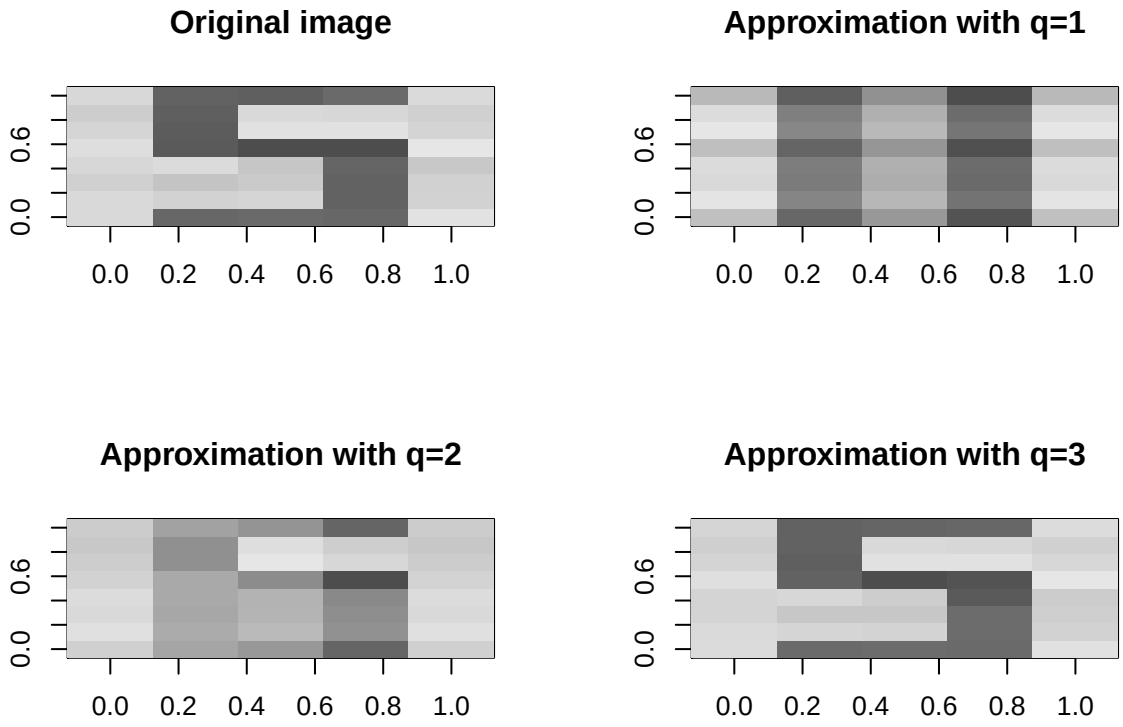

Figure 5: The original image and approximated ones using SVD.

and

$$\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_p).$$

Let $\widehat{\lambda} = (\widehat{\lambda}_1, \ldots, \widehat{\lambda}_p)$ and suppose n is large. Then

1. $\sqrt{n}(\widehat{\lambda} - \lambda) \sim N_p(\mathbf{0}, 2\Lambda^2)$. So, the large sample $100(1-\alpha)\%$ confidence interval for λ_i is obtained as follows:
$$\frac{\widehat{\lambda}_i}{1 + z_{\frac{\alpha}{2}}\sqrt{\frac{2}{n}}} \leq \lambda_i \leq \frac{\widehat{\lambda}_i}{1 - z_{\frac{\alpha}{2}}\sqrt{\frac{2}{n}}}.$$
2. $\sqrt{n}(\widehat{\mathbf{e}}_i - \mathbf{e}_i) \sim N_p(\mathbf{0}, \mathbf{E}_i)$ where $\mathbf{E}_i = \lambda_i \sum_{k \neq i} \frac{\lambda_k}{(\lambda_k - \lambda_i)^2} \mathbf{e}_k \mathbf{e}_k^\top$.
3. Each $\widehat{\lambda}_i$ is distributed independently of $\widehat{\mathbf{e}_i}$.

7 Real data applications of PCA and implementation in R

When you analyze real data, you can use the following criteria to choosing the number of PCs:

1. Retain the first q components which explain a large proportion of the total variation, say $70 - 90\%$.
2. If the correlation matrix is analyzed, retain only those components with variances greater than one.
3. Examine a scree plot. This is a plot of the PC variances versus the PC number. The idea is to look for an "elbow" which corresponds to the point after which the eigenvalues decrease more slowly.

4. Consider whether the PC has a sensible and useful interpretation.

We pursue with a few examples:

Example 1: The Nutritional Value of Food Nutritional data from 961 food items are listed alphabetically in the `food.txt` data set. The nutritional components of each food item are given by the following seven variables: fat (grams), food energy (calories), carbohydrates (grams), protein (grams), cholesterol (milligrams), weight (grams), and saturated fat (grams). Food items are listed according to very disparate serving sizes, which include teaspoon, tablespoon, cup, loaf, slice, cake, cracker, package, piece, pie, biscuit, muffin, spear, pat, wedge, stalk, cookie, and pastry. To equalize out the different types of servings for each food, we first divide each variable by weight of the food item (which leaves us with 6 variables).

```
#put the food.txt data in the folder you run your R
dir <- getwd()
food.path <- paste(dir,"/food.txt", sep="")
food <-read.table(file=food.path, header = TRUE, sep = "")

#After dividing each variable by weight we get the follwing variables
food.adjusted<-food/food[, 6]
food<-food.adjusted[,-6]
str(food)
```

```
## 'data.frame':    961 obs. of  6 variables:
##  $ Fat.grams           : num  0.1333 0.375 0.0353 0 0 ...
##  $ Food.energy.calories: num  1.667 3.75 3.175 3.175 0.303 ...
##  $ Carbohydrates.grams : num  0.1333 0.125 0.776 0.776 0.0303 ...
##  $ Protein.grams       : num  0 0 0.1411 0.1058 0.0303 ...
##  $ Cholesterol.mg      : num  0.133 0.25 0 0 0 ...
##  $ Saturated.fat.grams : num  0.01333 0.0625 0.00353 0.00353 0 ...
```

Because of wide variations in
variable scales, each variable is standardized by subtracting its mean and dividing the result by its standard deviation.

```
food <- scale(food)
summary(food)
```

```
##    Fat.grams       Food.energy.calories Carbohydrates.grams Protein.grams
##  Min.   :-0.5853   Min.   :-1.1641      Min.   :-0.9538     Min.   :-0.7779
##  1st Qu.:-0.5853   1st Qu.:-0.8529      1st Qu.:-0.7646     1st Qu.:-0.6852
##  Median :-0.4238   Median :-0.2124      Median :-0.4074     Median :-0.4073
##  Mean   : 0.0000   Mean   : 0.0000      Mean   : 0.0000     Mean   : 0.0000
##  3rd Qu.: 0.1019   3rd Qu.: 0.5971      3rd Qu.: 0.5767     3rd Qu.: 0.2752
##  Max.   : 4.5835   Max.   : 3.4981      Max.   : 3.0527     Max.   : 8.7527
##  Cholesterol.mg     Saturated.fat.grams
##  Min.   :-0.37895   Min.   :-0.5620
##  1st Qu.:-0.37895   1st Qu.:-0.5620
##  Median :-0.37895   Median :-0.4411
##  Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.05007   3rd Qu.: 0.1539
##  Max.   :18.16991   Max.   : 7.1068
```

In R, PCA can be done using the functions `princomp()` and `prcomp()` (both contained in the R package `stats`). Here we use `princomp()` and in the next example we show how PCA can be done using `prcomp()`.

1. The `princomp()` function carries out PCA via an eigendecomposition of the sample covariance matrix **S**.
2. When the variables are on very different scales, PCA is usually carried out on the correlation matrix. These components are not equal to those derived from S.

A PCA of the transformed data yields six principal components ordered by decreasing variances, which by using `summary()` we can get a very good understanding of each PC, its standard deviation, proportion of the variance in the original data explained by each PC, etc. For example, below we see that the first three principal components, PC1, PC2, and PC3, account for more than 83.3 percent of the total variance in the data.

```
fit <- princomp(food, cor=TRUE)
summary(fit)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
## Standard deviation     1.6274498 1.1533146 1.0100127 0.8246633 0.51626027
## Proportion of Variance 0.4414322 0.2216891 0.1700209 0.1133449 0.04442078
## Cumulative Proportion  0.4414322 0.6631213 0.8331422 0.9464871 0.99090792
##                            Comp.6
## Standard deviation     0.233564773
## Proportion of Variance 0.009092084
## Cumulative Proportion  1.000000000
```

The results of the analysis using `princomp()` is represented by a list containing coefficients (loading) defining each component, the PC scores, etc. The coefficients (loading) for the first 3 PCs are obtained below. Notice that PC1 puts little weight on carbohydrates, and PC2 puts little weight on fat and saturated fat.

```
fit$loadings[, 1:3] # pc loadings
```

```
##                           Comp.1      Comp.2     Comp.3
## Fat.grams              0.55723936  0.09870077  0.2750890
## Food.energy.calories   0.53615066  0.35676646 -0.1370762
## Carbohydrates.grams   -0.02455362  0.67163163 -0.5684779
## Protein.grams          0.23522713 -0.37384298 -0.6388770
## Cholesterol.mg         0.25250455 -0.52130441 -0.3256120
## Saturated.fat.grams    0.53135067 -0.01923360  0.2611169
```

It is easy to verify that each loading is unique up to a sign flip. Also, note that

```
e.1 <- fit$loadings[,1]
e.2 <- fit$loadings[,2]
#Norm of loadings
e.1%*%e.1
```

```
##      [,1]
## [1,]    1
```

```
#PCs are perpendicular
e.1%*%e.2
```

```
##              [,1]
## [1,] 1.734723e-17
```

We can re-scale coefficients (loading) so that coefficients for the most important components are larger than those for the less in important components. This can be done by setting $\mathbf{e}_j^\star = \sqrt{\lambda_j}\mathbf{e}_j$ for which $\mathbf{e}^{\star\top}\mathbf{e}^\star = \lambda_j$. For example, the re-scaled loading for PC1 are calculated as

```
(scaled.e1 <- e.1 * fit$sdev[1])
```

```
##            Fat.grams Food.energy.calories  Carbohydrates.grams
##           0.90687912           0.87255831          -0.03995979
##        Protein.grams       Cholesterol.mg  Saturated.fat.grams
##           0.38282035           0.41093848           0.86474657
```

```
plot(fit,type="lines", pch=20) # scree plot
```

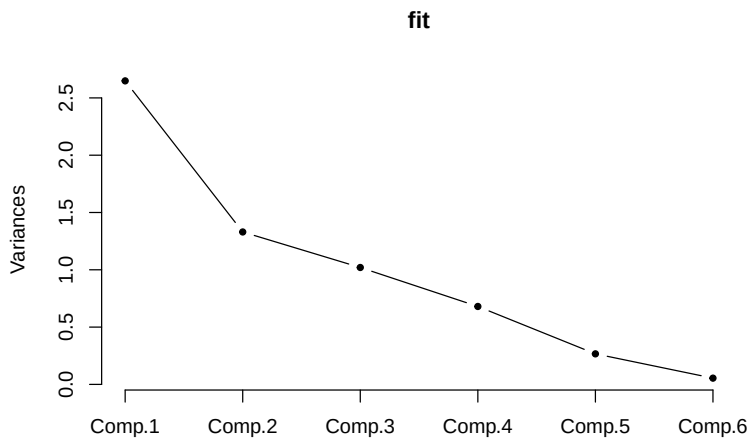

Figure 6: Variance explained by each PC

The scatter plot of the first two PCs is given below. The scatter plot appears to show a number of interesting features. Notice the almost straight-line edge to the plotted points at the upper left-hand corner.

```
biplot(fit, expand = 1.4,
       col=c("red","blue"),
       cex = c(0.35, 0.6),
       xlim = c(-1.5, 8),
       ylim = c(-11,4),
       scale = FALSE )
```

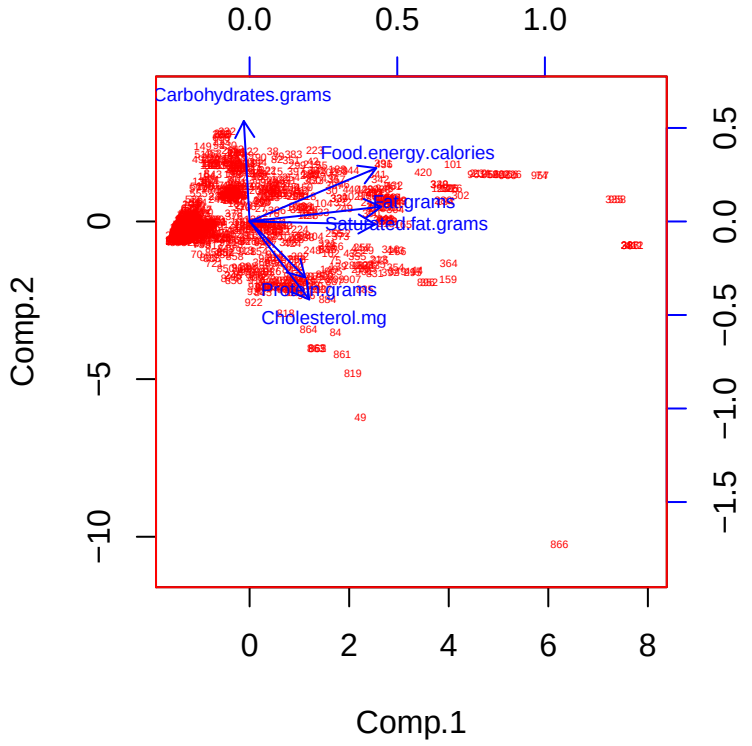

Figure 7: The scatterplot of the first two principal components

We also can identify various groups of points in this display, where the food items in each group have been ordered by magnitude of that nutritional component, starting at the largest value. For example, by looking at the raw data we see that the observations 866 (raw egg yolk), 49 (chicken liver), 861 (beef liver), 861 (fried egg), 833 (hard-cooked egg), 864 (poached egg), and 315 (scrambled egg) are leading items in PC1 which is reflecting Cholesterol.

One can see that Protein is associated with dry gelatin, raw seaweed, yeast, and Parmesan cheese. Saturated fat has the leading foods such as butter, lard, bitter chocolate, coconut, cooking fat, and cheddar cheese. etc. Most of these points are identified in the scatter plot, but some are covered too well to be displayed clearly. We see that food item raw egg yolk is an outlier along an imaginary cholesterol axis and butter and lard are outliers along an imaginary saturated-fat axis.

Example 2: In this example we provide an application by working with a data set, named USArrests, that contains the number of arrests per 100,000 residents for murder, assault, and rape for each of the 50 states in US in 1973. The data set also contains the percentage of people in the state who live in an urban area. The data set is pre-loaded in `R`, so you can load it directly as a data frame with 50 observations on 4 variables.

- Murder: numeric Murder arrests (per 100,000)
- Assault: numeric Assault arrests (per 100,000)
- UrbanPop: numeric Percent urban population
- Rape: numeric Rape arrests (per 100,000)

The rows of the data set contain the 50 states in alphabetical order and the columns contain the four variables.

```
data(USArrests)
help(USArrests)
```

You can look at the names of variables contained in this data object (or data frame, as R calls them) by typing:

```
names(USArrests)
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

```
head(USArrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

Let us examine the data first. The average of each variable is given by

```
apply(USArrests, 2, mean)
```

```
##   Murder  Assault UrbanPop     Rape
##    7.788  170.760   65.540   21.232
```

Also, the variance-covariance matrix is obtained as follows:

```
(S<-cov(USArrests))
```

```
##              Murder   Assault   UrbanPop      Rape
## Murder    18.970465  291.0624   4.386204  22.99141
## Assault  291.062367 6945.1657 312.275102 519.26906
## UrbanPop   4.386204  312.2751 209.518776  55.76808
## Rape      22.991412  519.2691  55.768082  87.72916
```

Now, we are ready to apply a PCA on this data set. Here we use `prcomp()` that instead of performing PCA via an eigendecomposition of the covariance matrix (as in `princomp()`), it does a singular value decomposition of the (centered and possibly scaled) data matrix. By default the `prcomp()` function centers the variable to have mean zero. By using the option `scale=TRUE`, one can scale the variables to have standard deviation one. This is what we use in this example:

```
fit <- prcomp(USArrests, scale=TRUE)
summary(fit) # print variance accounted for
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.5749 0.9949 0.59713 0.41645
## Proportion of Variance 0.6201 0.2474 0.08914 0.04336
## Cumulative Proportion  0.6201 0.8675 0.95664 1.00000
```

Let us look at the output of `prcomp()`:

```
names(fit)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```
fit
```

```
## Standard deviations (1, .., p=4):
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
##
## Rotation (n x k) = (4 x 4):
##                 PC1        PC2        PC3         PC4
## Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
## Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
## Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

When we matrix-multiply the data matrix matrix by `fit$rotation`, it gives the PC scores. Alternatively you can use `prcomp()$x` that is a matrix where its columns are the PC scores. For example, the first column will give the scores of PC1, which we show only a few of them below.

```
dim(fit$x)
```

```
## [1] 50  4
```

```
head(fit$x[,1])
```

```
##    Alabama     Alaska    Arizona   Arkansas California   Colorado
## -0.9756604 -1.9305379 -1.7454429  0.1399989 -2.4986128 -1.4993407
```

We have also provided the Scree plot and a plot showing the proportion of the total variance in the data set that is explained by the first k PCs.

```
par(mfrow=c(1, 2))
plot(fit,type="lines", pch=20, main="") # scree plot
plot(summary(fit)$importance[3,],
     type="b", pch=20,  col="green",
     ylab="Proportion of Variance explained")
```

Also, we can obtain the loading of the PCs using:

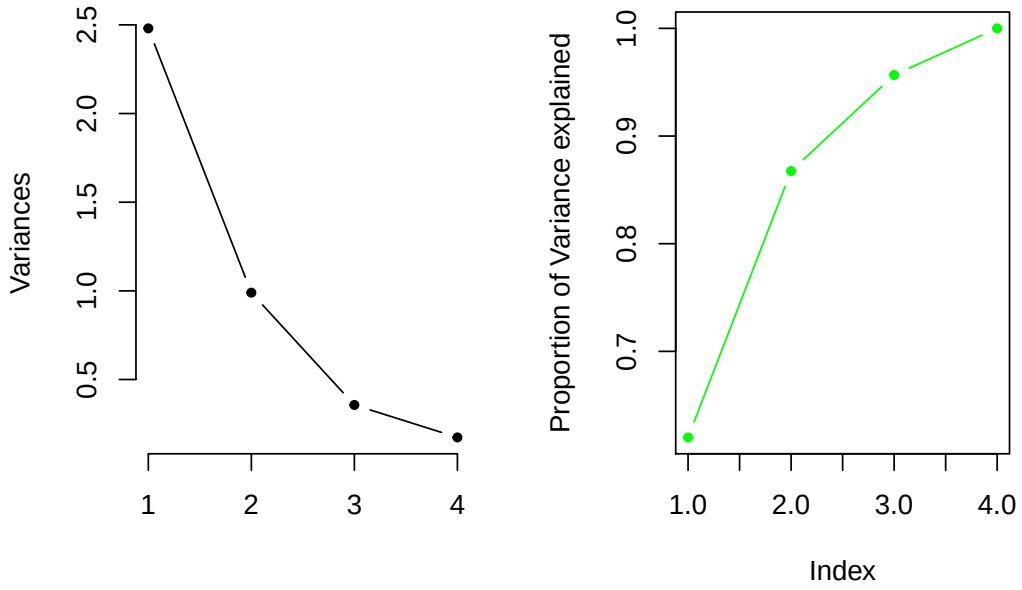

Figure 8: On the left we have a Scree plot to show the variance explained by each PC. On the right we have the plot of the proportion of the variance explained by PCs

```
fit$rotation
```

```
##                 PC1        PC2        PC3         PC4
## Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
## Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
## Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

By checking the weights of the first two principal components, we see that:

- The first loading vector places approximately equal weight on Assault, Murder, and Rape, with much less weight on UrbanPop. Hence this component roughly corresponds to a measure of overall rates of serious crimes.
- The second loading vector places most of its weight on UrbanPop and much less weight on the other three features. Hence, this component roughly corresponds to the level of urbanization of the state.

The below biplot shows that 50 states mapped according to the 2 principal components. The vectors of the PCA for 4 variables are also plotted.

```
biplot(fit,
       expand = 0.9,
       col=c("red","blue"),
       cex=c(0.5, 0.75),
       xlab="PC1 scores",
       ylab="PC2 scores")
```

The large positive scores on the first component, such as California, Nevada and Florida, have high crime rates, while states like North Dakota, with negative scores on the first component, have low crime rates.

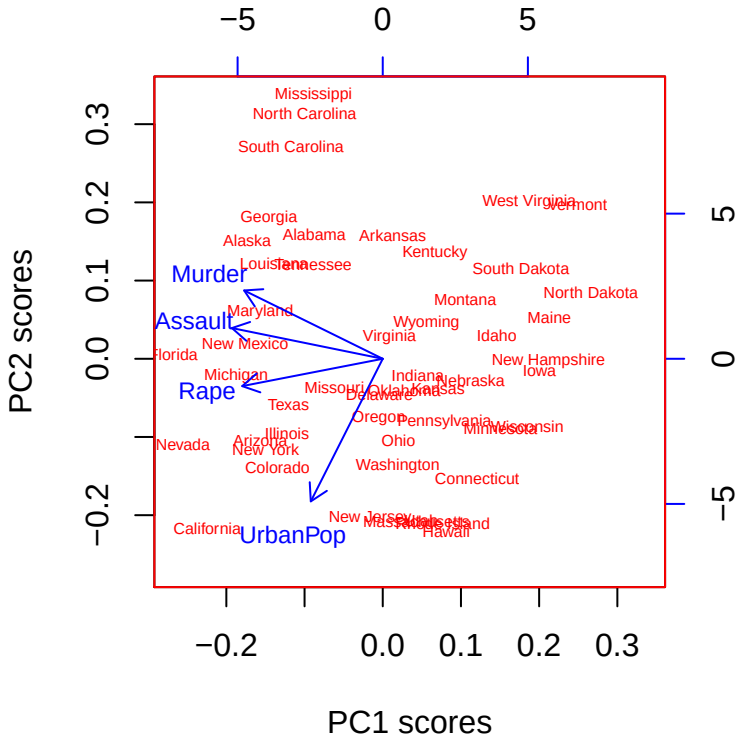

Figure 9: Biplot showing 50 states mapped according to the 2 principal components.

California also has a high score on the second component, indicating a high level of urbanization, while the opposite is true for states like Mississippi. States close to zero on both components, such as Indiana, have approximately average levels of both crime and urbanization.

In the following example we try to perform most of the calculation by writing functions. This is not necessary and you can use packages.

Example 3: We have data on blood pressure, age, weight, body surface area, duration of hypertension, basal pulse, stress index for 20 individuals with high blood pressure.

```
BP.data <- read.table(file="bp.txt", header=TRUE)[,-1]
BP.data
```

```
##      BP Age Weight  BSA  Dur Pulse Stress
## 1   105  47   85.4 1.75  5.1    63     33
## 2   115  49   94.2 2.10  3.8    70     14
## 3   116  49   95.3 1.98  8.2    72     10
## 4   117  50   94.7 2.01  5.8    73     99
## 5   112  51   89.4 1.89  7.0    72     95
## 6   121  48   99.5 2.25  9.3    71     10
## 7   121  49   99.8 2.25  2.5    69     42
## 8   110  47   90.9 1.90  6.2    66      8
## 9   110  49   89.2 1.83  7.1    69     62
## 10 114  48   92.7 2.07  5.6    64     35
## 11 114  47   94.4 2.07  5.3    74     90
## 12 115  49   94.1 1.98  5.6    71     21
## 13 114  50   91.6 2.05 10.2    68     47
## 14 106  45   87.1 1.92  5.6    67     80
```

```
## 15 125  52  101.3 2.19 10.0    76     98
## 16 114  46   94.5 1.98  7.4    69     95
## 17 106  46   87.0 1.87  3.6    62     18
## 18 113  46   94.5 1.90  4.3    70     12
## 19 110  48   90.5 1.88  9.0    71     99
## 20 122  56   95.7 2.09  7.0    75     99
```

```
n <- 20
```

Suppose we want to regress BP on other variables.

```
model1 <-  lm(BP~ Age + Weight + BSA + Dur + Pulse+ Stress, data=BP.data)
summary(model1)
```

```
##
## Call:
## lm(formula = BP ~ Age + Weight + BSA + Dur + Pulse + Stress,
##     data = BP.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93213 -0.11314  0.03064  0.21834  0.48454
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.870476   2.556650  -5.034 0.000229 ***
## Age           0.703259   0.049606  14.177 2.76e-09 ***
## Weight        0.969920   0.063108  15.369 1.02e-09 ***
## BSA           3.776491   1.580151   2.390 0.032694 *
## Dur           0.068383   0.048441   1.412 0.181534
## Pulse        -0.084485   0.051609  -1.637 0.125594
## Stress        0.005572   0.003412   1.633 0.126491
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4072 on 13 degrees of freedom
## Multiple R-squared:  0.9962, Adjusted R-squared:  0.9944
## F-statistic: 560.6 on 6 and 13 DF,  p-value: 6.395e-15
```

By checking the fitted regression model for multicollinearity using

```
library(car)
```

```
## Loading required package: carData
```

```
vif(model1)
```

```
##      Age   Weight      BSA      Dur    Pulse   Stress
## 1.762807 8.417035 5.328751 1.237309 4.413575 1.834845
```

We observe that some of the variance inflation factors (VIF) are quite large. Hence, we suspect that the model has multicollinearity. Let's check the correlations between covariates.

```
X= as.matrix(BP.data[2:7])      # exclude the response variable!!
round(cor(X),2)
```

```
##         Age Weight  BSA  Dur Pulse Stress
## Age    1.00   0.41 0.38 0.34  0.62   0.37
## Weight 0.41   1.00 0.88 0.20  0.66   0.03
## BSA    0.38   0.88 1.00 0.13  0.46   0.02
## Dur    0.34   0.20 0.13 1.00  0.40   0.31
## Pulse  0.62   0.66 0.46 0.40  1.00   0.51
## Stress 0.37   0.03 0.02 0.31  0.51   1.00
```

To predict BP using the covariates, we would like to use PCA first and then use a few PCs to fit a PCR (principal component regression model). First, we obtain the sample mean vector and sample covariance matrix of $\mathbf{X}$:

```
x.bar <-  apply(X,2,mean)
x.bar
```

```
##    Age Weight    BSA    Dur  Pulse Stress
## 48.600 93.090  1.998  6.430 69.600 53.350
```

```
S <- cov(X)
S
```

```
##                Age     Weight        BSA         Dur      Pulse       Stress
## Age     6.2526316  4.3747368 0.12915789  1.84421053  5.8842105 3.414737e+01
## Weight  4.3747368 18.4462105 0.51308421  1.84873684 10.7694737 5.472105e+00
## BSA     0.1291579  0.5130842 0.01862737  0.03822105  0.2412632 9.336842e-02
## Dur     1.8442105  1.8487368 0.03822105  4.60221053  3.2757895 2.479421e+01
## Pulse   5.8842105 10.7694737 0.24126316  3.27578947 14.4631579 7.141053e+01
## Stress 34.1473684  5.4721053 0.09336842 24.79421053 71.4105263 1.375397e+03
```

```
round(S, 3)
```

```
##           Age Weight   BSA    Dur  Pulse   Stress
## Age     6.253  4.375 0.129  1.844  5.884   34.147
## Weight  4.375 18.446 0.513  1.849 10.769    5.472
## BSA     0.129  0.513 0.019  0.038  0.241    0.093
## Dur     1.844  1.849 0.038  4.602  3.276   24.794
## Pulse   5.884 10.769 0.241  3.276 14.463   71.411
## Stress 34.147  5.472 0.093 24.794 71.411 1375.397
```

As we observe, the scales are quite different, suggesting that we may be better off using $\mathbf{R}$ to get the PCs nevertheless, let's continue with $\mathbf{S}$. We will redo the analysis with $\mathbf{R}$ later!. To get the PCs, first we find the eigenvalues and eigenvectors

```
Val <- eigen(S)$values
Vec <- eigen(S)$vectors
```

We have 6 PCs which are linear combinations of $X_1, \ldots, X_6$ with weights given by each eigenvector. We can obtain the data for PCs using:

```
W <-  X  # just to create a data matrix of the same size of X

# now fill in the entries by calculating sample PCs

for(i in 1:6){
    for(j in 1:20){
    W[j,i] <-  Vec[,i] %*% ( X[j,] -x.bar)  # centered PCs
 }}

colnames(W) <-  paste("W", 1:6, sep="")
# Principal Components have zero correlation:

plot(data.frame(W))
```

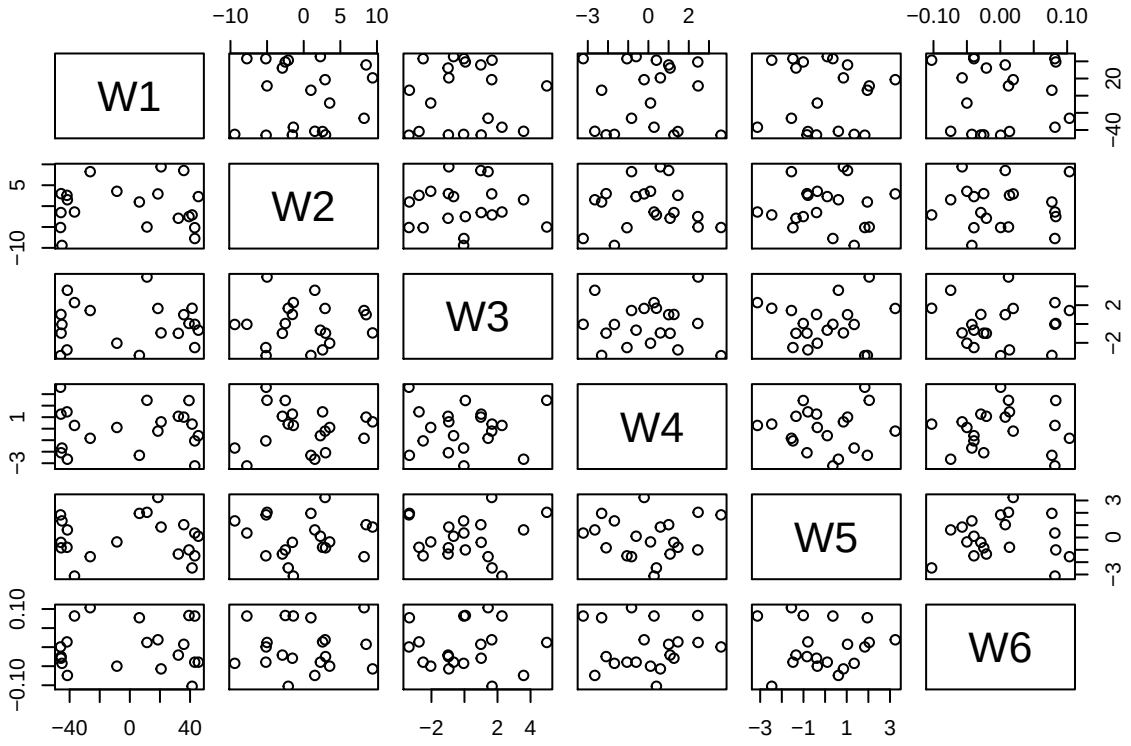

Figure 10: Scatterplot of PCs of centered variables

```
round(cor(W),3)
```

```
##    W1 W2 W3 W4 W5 W6
## W1  1  0  0  0  0  0
## W2  0  1  0  0  0  0
```

```
## W3  0  0  1  0  0  0
## W4  0  0  0  1  0  0
## W5  0  0  0  0  1  0
## W6  0  0  0  0  0  1
```

Notice that centered Principal Components have zero mean:

```
round(apply(W, 2, mean),3)
```

```
## W1 W2 W3 W4 W5 W6
##  0  0  0  0  0  0
```

How many components should we keep? To this end, we can use the scree plot:

```
plot(Val, type="b")  # suggests keeping the first PC only!
```

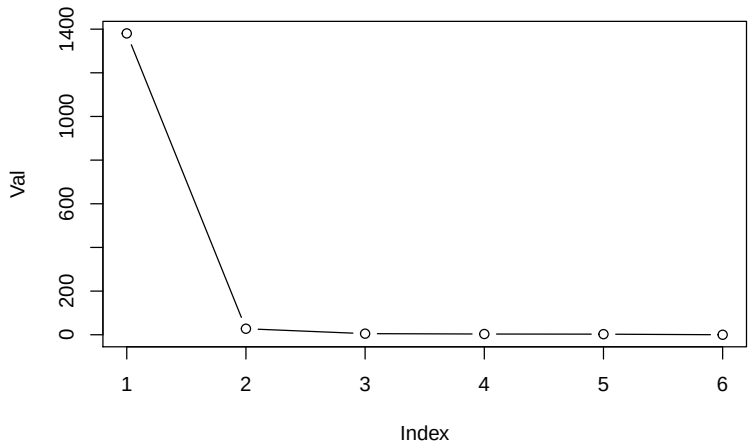

Figure 11: Variance of each PC

Also, it is easy to calculate the proportion of variation explained by each PC:

```
round(Val/sum(Val),3)  # 97.3 % of the sample variation in X is explained by the first PC.
```

```
## [1] 0.973 0.019 0.004 0.002 0.002 0.000
```

If you like, you can use built-in functions in R for a summary:

```
summary(prcomp(X))
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6
## Standard deviation     37.1548 5.25454 2.24839 1.83160 1.63689 0.05908
## Proportion of Variance  0.9727 0.01946 0.00356 0.00236 0.00189 0.00000
## Cumulative Proportion   0.9727 0.99218 0.99575 0.99811 1.00000 1.00000
```

Now let's run regression with the first PC as the explanatory variable:

```r
PC.model <- lm(BP.data$BP ~ W[,1])
summary(PC.model)  # W1 is not found significant! adj.R^2 is too low (negative!)
```

```
##
## Call:
## lm(formula = BP.data$BP ~ W[, 1])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6505 -3.3268  0.0889  2.2441  9.8861
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 114.00000    1.22926  92.739   <2e-16 ***
## W[, 1]       -0.02471    0.03394  -0.728    0.476
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.497 on 18 degrees of freedom
## Multiple R-squared:  0.02859,    Adjusted R-squared:  -0.02538
## F-statistic: 0.5297 on 1 and 18 DF,  p-value: 0.4761
```

However, if we add W_2 (or even more PCs) we would find them significant:

```r
PC.model.2 <- lm(BP.data$BP ~  W[,1]+ W[,2])
summary(PC.model.2)  # W1 is not significant!
```

```
##
## Call:
## lm(formula = BP.data$BP ~ W[, 1] + W[, 2])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2859 -0.8294 -0.0579  0.8103  3.3058
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 114.00000    0.36609 311.400  < 2e-16 ***
## W[, 1]       -0.02471    0.01011  -2.444   0.0257 *
## W[, 2]       -0.97474    0.07148 -13.636 1.39e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.637 on 17 degrees of freedom
## Multiple R-squared:  0.9186, Adjusted R-squared:  0.9091
## F-statistic: 95.96 on 2 and 17 DF,  p-value: 5.482e-10
```

One can also check the importance of each variable in the first two PCs:

```r
round(Vec[,1],3)   # Stress level dominates the first PC
```

```
## [1] -0.025 -0.005  0.000 -0.018 -0.052 -0.998
```

```r
round(Vec[,2],3)   # weight and pulse contribute to the second PC more than other variables.
```

```
## [1] -0.257 -0.779 -0.021 -0.118 -0.558  0.041
```

Let us now, redo the whole analysis using the sample correlation coefficient matrix.

```r
Z=X
for(i in 1:6){
Z[,i] <- (X[,i]-x.bar[i])/sqrt(diag(S)[i])
}

# obtain correlation matrix
R <- cor(X)
cov(Z)  # they should be the same!
```

```
##               Age     Weight        BSA       Dur     Pulse     Stress
## Age    1.0000000 0.40734926 0.37845460 0.3437921 0.6187643 0.36822369
## Weight 0.4073493 1.00000000 0.87530481 0.2006496 0.6593399 0.03435475
## BSA    0.3784546 0.87530481 1.00000000 0.1305400 0.4648188 0.01844634
## Dur    0.3437921 0.20064959 0.13054001 1.0000000 0.4015144 0.31163982
## Pulse  0.6187643 0.65933987 0.46481881 0.4015144 1.0000000 0.50631008
## Stress 0.3682237 0.03435475 0.01844634 0.3116398 0.5063101 1.00000000
```

We obtain eigenvalues and eigenvectors of $\mathbf{R}$

```r
Val.new <- eigen(R)$values
round(Val.new ,2)
```

```
## [1] 3.01 1.39 0.71 0.52 0.31 0.07
```

```r
Vec.new <-  eigen(R)$vectors
rownames(Vec.new) <-  colnames(X)
colnames(Vec.new) <-  c("PC1", "PC2", "PC3", "PC4", "PC5", "PC6")
round(Vec.new ,2)
```

```
##          PC1   PC2   PC3   PC4   PC5   PC6
## Age    -0.43  0.18  0.15  0.84  0.19 -0.10
## Weight -0.47 -0.44 -0.03 -0.19 -0.14 -0.73
## BSA    -0.42 -0.49  0.00 -0.17  0.53  0.52
## Dur    -0.29  0.39 -0.86 -0.10  0.10  0.00
## Pulse  -0.51  0.13  0.16 -0.11 -0.72  0.41
## Stress -0.26  0.60  0.45 -0.45  0.37 -0.17
```

Also, sample PC values are obtained below:

```
W.new <-  X  # just to create a data matrix of the same size of X
colnames(W.new) = c("PC1", "PC2", "PC3", "PC4", "PC5", "PC6")

# now fill in the entries by calculating sample PCs

for(i in 1:6){ # PC's
    for(j in 1:20){
    W.new[j,i] <-  Vec.new[,i] %*% Z[j,]
    # no need to center when using normalized PCCs
 }}
```

Note that PCs have correlation zero.

```
plot(data.frame(W.new))
```

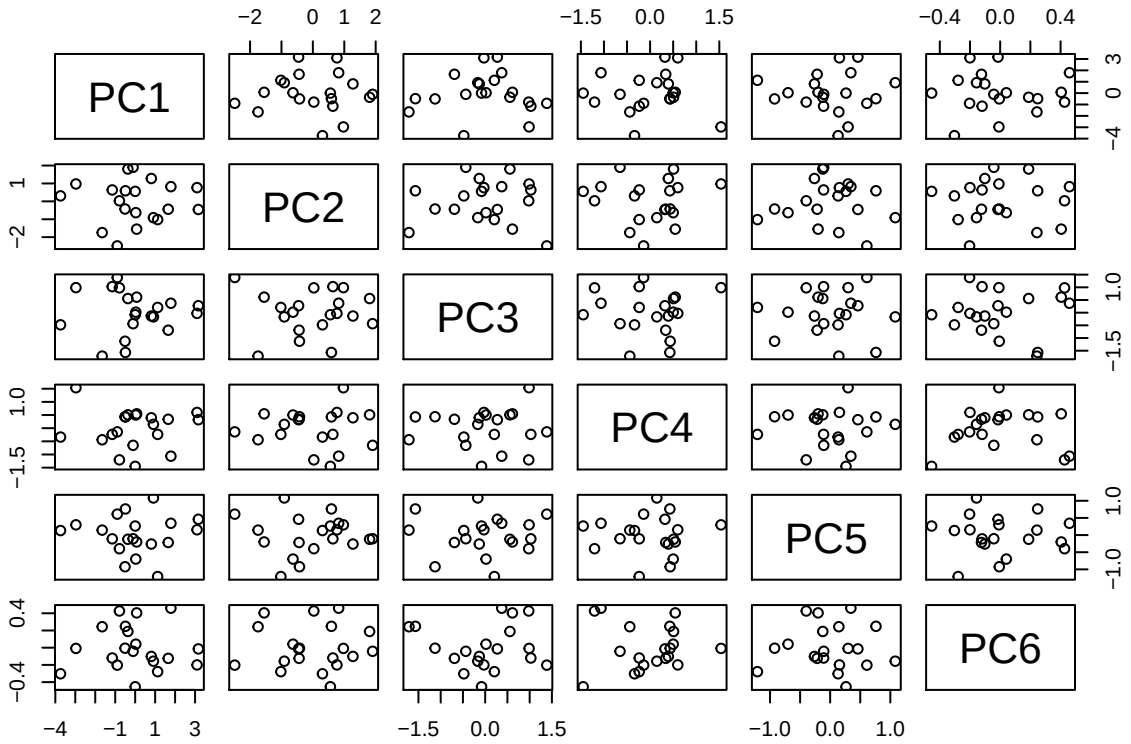

Figure 12: Scatterplots of PCs

```
round(cor(W.new),3)
```

```
##      PC1 PC2 PC3 PC4 PC5 PC6
## PC1    1   0   0   0   0   0
## PC2    0   1   0   0   0   0
## PC3    0   0   1   0   0   0
## PC4    0   0   0   1   0   0
## PC5    0   0   0   0   1   0
## PC6    0   0   0   0   0   1
```

Again, How many components should we keep?

```
plot(Val.new, type="b", pch=19, xlab="",ylab="Variances")
```

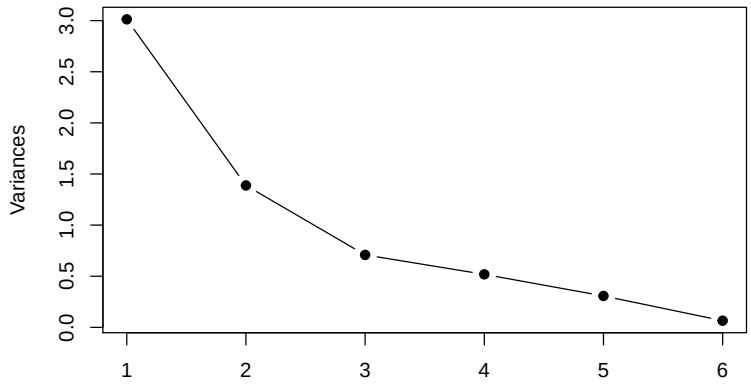

Figure 13: Variance explained by each PC

```
# suggests keeping the first 4 or 5 PCs.
```

Proportion of variation explained by each PC can also be obtained via `screeplot()` in `R`:

```
round(Val.new/sum(Val.new),3)
```

```
## [1] 0.502 0.231 0.118 0.086 0.051 0.011
```

```
summary( prcomp(Z))
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.7357 1.1781 0.8419 0.71993 0.55411 0.25528
## Proportion of Variance 0.5021 0.2313 0.1181 0.08638 0.05117 0.01086
## Cumulative Proportion  0.5021 0.7335 0.8516 0.93797 0.98914 1.00000
```

```
screeplot(prcomp(Z), npcs = 6, type = "lines", pch=20)
```

We can use these to perform a regression analysis with all standardized PCs as the explanatory variables

```
PC.model.new.1 <-  lm(BP.data$BP ~  W.new[,1] +
                      W.new[,2] + W.new[,3] + W.new[,4] + W.new[,5]+ W.new[,6])
summary(PC.model.new.1)  # W2 and W5 seem not to be significant
```

```
##
## Call:
## lm(formula = BP.data$BP ~ W.new[, 1] + W.new[, 2] + W.new[, 3] +
##     W.new[, 4] + W.new[, 5] + W.new[, 6])
##
```

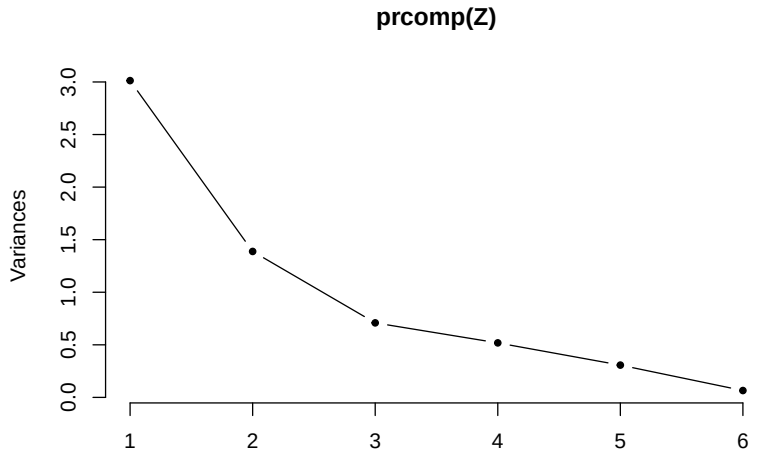

Figure 14: Scree plot to variance explained by each PC

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93213 -0.11314  0.03064  0.21834  0.48454
##
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 114.00000    0.09106 1251.934  < 2e-16 ***
## W.new[, 1]   -2.87295    0.05382  -53.376  < 2e-16 ***
## W.new[, 2]   -1.63040    0.07930  -20.560 2.68e-11 ***
## W.new[, 3]    0.04409    0.11097    0.397   0.6976
## W.new[, 4]    0.52438    0.12977    4.041   0.0014 **
## W.new[, 5]    0.34485    0.16860    2.045   0.0616 .
## W.new[, 6]   -3.09367    0.36597   -8.453 1.22e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4072 on 13 degrees of freedom
## Multiple R-squared:  0.9962, Adjusted R-squared:  0.9944
## F-statistic: 560.6 on 6 and 13 DF,  p-value: 6.395e-15
```

Let's remove W_3^* and W_5^*

```
PC.model.new.2 = lm(BP.data$BP ~  W.new[,1] + W.new[,2] + W.new[,4] + W.new[,6])
summary(PC.model.new.2)
```

```
##
## Call:
## lm(formula = BP.data$BP ~ W.new[, 1] + W.new[, 2] + W.new[, 4] +
##     W.new[, 6])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98824 -0.20787  0.04648  0.23649  0.58000
##
```

```
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 114.00000    0.09791 1164.355  < 2e-16 ***
## W.new[, 1]   -2.87295    0.05787  -49.642  < 2e-16 ***
## W.new[, 2]   -1.63040    0.08526  -19.122 6.04e-12 ***
## W.new[, 4]    0.52438    0.13953    3.758   0.0019 **
## W.new[, 6]   -3.09367    0.39349   -7.862 1.07e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4379 on 15 degrees of freedom
## Multiple R-squared:  0.9949, Adjusted R-squared:  0.9935
## F-statistic: 726.5 on 4 and 15 DF,  p-value: < 2.2e-16
```

Note that removing PCs from the model does not change the coefficients! We can check the importance of each variable in standardized PCs:

```
round(Vec.new[,1],3)   # all variables contribute to the first PC
```

```
##    Age Weight    BSA    Dur  Pulse Stress
## -0.430 -0.472 -0.425 -0.290 -0.509 -0.263
```

```
round(Vec.new[,2],3)   # stress vs (weight and BSA)
```

```
##    Age Weight    BSA    Dur  Pulse Stress
##  0.183 -0.440 -0.494  0.389  0.135  0.599
```

```
round(Vec.new[,3],3)   # Dur dominates the third PC  (perhaps vs stress)
```

```
##    Age Weight    BSA    Dur  Pulse Stress
##  0.153 -0.033 -0.004 -0.864  0.164  0.450
```

```
round(Vec.new[,4],3)   # Age dominates the fourth PC (perhaps vs stress)
```

```
##    Age Weight    BSA    Dur  Pulse Stress
##  0.844 -0.193 -0.166 -0.097 -0.108 -0.449
```

```
round(Vec.new[,5],3)   # Pulse vs (BSA and stress)
```

```
##    Age Weight    BSA    Dur  Pulse Stress
##  0.190 -0.140  0.527  0.095 -0.719  0.374
```

```
round(Vec.new[,6],3)   # Weight vs (BSA and pulse)
```

```
##    Age Weight    BSA    Dur  Pulse Stress
## -0.100 -0.725  0.519 -0.001  0.409 -0.166
```

The correlations between PCs and variables also indicate importance:

```
##            W1         W2           W3          W4          W5            W6
## X1 -0.7455676  0.2155651  0.128832599  0.60772061  0.10532396 -0.0254015989
## X2 -0.8188220 -0.5189090 -0.027604490 -0.13863089 -0.07762341 -0.1850891720
## X3 -0.7371247 -0.5825774 -0.003015128 -0.11941094  0.29228069  0.1324941525
## X4 -0.5037900  0.4577906 -0.727318923 -0.06968866  0.05266996 -0.0002002768
## X5 -0.8832188  0.1587580  0.138214375 -0.07759309 -0.39834375  0.1044927419
## X6 -0.4573168  0.7057680  0.378544379 -0.32345613  0.20746323 -0.0423340469
```

8 Exercises

1. Consider the following sample covariance matrix:

$$\mathbf{S} = \begin{pmatrix} 5 & 2 \\ 2 & 2 \end{pmatrix}.$$

Without using R and by calculations with hand

(a) Determine the sample principal components (PCs) from $\mathbf{S}$

(b) Calculate the proportion of the total sample variance explained by the first PC.

(c) Convert the covariance matrix to a correlation matrix and determine it PCs.

(d) Compute the correlations between the original variables and the two PCs obtained in (c).

2. Find the principal components and the proportion of the total variance explained by each component when the covariance matrix is

$$\sigma = \begin{pmatrix} \sigma^2 & \sigma^2\rho & 0 \\ \sigma^2\rho & \sigma^2 & \sigma^2\rho \\ 0 & \sigma^2\rho & \sigma^2 \end{pmatrix} \quad -\frac{1}{\sqrt{2}} < \rho < \frac{1}{\sqrt{2}}.$$

3. Multivariate methods are often used in the analysis of genomic data. In particular, PCA and Cluster Analysis are popular tools. In this exercise, we illustrate these techniques on the NCI60 cancer cell line microarray data that are contained in the R package ISLR. The format of NCI60 is a list containing two elements: data and labs, where data is a 64 by 6380 matrix of the 6380 expression measurements on 64 cancer cell lines and labs is a vector listing the cancer types for the 64 cell lines.

(a) Run the following R code:

```
library(ISLR)
nci.labs <- NCI60$labs
nci.data <- NCI60$data
dim(nci.data)
length(nci.labs)
```

(b) Examine the cancer types for the cell lines.

(c) Use the function `prcomp()` to perform PCA on the `nci.data` after scaling the variables (genes) to have standard deviation one.

(d) Verify that the sum of squares of the elements of the vectors of coefficients (loading) of the first and second principal component, respectively, is equal to one and that both vectors are orthogonal to each other.

(e) The following function can be used to assign a color to each of the 64 cell lines, based on the cancer type to which it corresponds. Note that the `rainbow()` function takes as its argument a positive integer and returns a vector containing that number of distinct colors. Use the function below to plot the scores of the first three principal components (PCs).

```
Cols <- function(vec){
        cols <- rainbow(length(unique(vec)))
        return(cols[as.numeric(as.factor(vec))])
}
```

(f) By making use of the `summary()` method for a `prcomp` object, obtain a summary of the proportion of variance explained (PVE) by the PCs. Scale the loading on the first two PCs so that the elements of the rescaled vectors represent the correlations between the variables and the first and second PC, respectively.

(g) Provide a plot of the PVE of each component (i.e. a scree plot) and a plot of the cumulative PVE of each component.

4. To simplify interpretation of principal components, rotation can be used with the objective of making the rotated components as simple as possible to interpret. However, after rotation, one or both of the properties of PCA, that is, the orthogonality of loading vectors and the uncorrelatedness of component scores, disappears. Verify this statement using a simulation in `R`.

5. (a) Let X and Y be jointly normally distributed and uncorrelated random variables. Are X and Y independent? Justify your answer!

(b) Now suppose that X and Y are not jointly normally distributed but each one alone is marginally normally distributed, and X and Y are uncorrelated. Can you make a statement whether X and Y are independent? Justify your answer!

6. Download the file `digits.Rdata` from the course website and load it into your `R` session with `load("digits.Rdata")`. Now you should have a matrix threes that has dimension 658×256. (This data set was taken from the data page on `http://www-stat.stanford.edu/~tibs/ElemStatLearn/`.) Each row of the matrix corresponds to an image of a "3" that was written by a different person. Hence each row vector is of length 256, corresponding to a 16 by 16 pixels image that has been unraveled into a vector, and each pixel takes gray scale values between -1 and 1. You can use the following function to show any of these threes:

```
plot.digit = function(x,zlim=c(-1,1)) {
  cols = gray.colors(100)[100:1]
  image(matrix(x,nrow=16)[,16:1],col=cols,
        zlim=zlim,axes=FALSE)
}
```

For example you can type `plot.digit(threes[1,])`.

(a) Compute the principal component directions and principal component scores of threes. Plot the first two principal component scores (the x-axis being the first score and the y-axis being the second score). Note that each point in this plot corresponds to an image of a "3".

(b) For each of the first two principal component scores, compute the following percentiles: $5\%, 25\%, 50\%, 75\%, 95\%$. Draw these values as vertical and horizontal lines on top of your plot (i.e., vertical for the percentiles of the first principal component score, and horizontal for those of the second.). You can use `quantile` for the percentiles, and `abline` to draw the lines

(c) Now you want to identify a point (i.e., an image of a "3") close to each of the vertices of the grid on your plot. This can be done by using the `identify` function with `n=25`, which allows you to click on the plot 25 times (since there are 25 vertices). Each time you click, it will print the index of the point that is closest to your click's location. Make sure you click left-to-right, and top-to-bottom, and record the indices in that order. (Note: although the `identify` function returns a vector of indices, and it claims that this vector is ordered by the order of your clicks, it actually gives them in sorted order. This isn't what you want—you want them in the order that you clicked, so you may have to build this vector manually.)

(d) Plot all of the images of "3"s that you picked out in part (c), in an order that corresponds to the vertices of the grid. For example, if you saved the vector of indices that you built in (c) as `inds`, and you built them by clicking left-to-right and top-to-bottom as instructed, this can be done with:

```
par(mfrow=c(5,5))            # allow for 5 x 5 plots
par(mar=c(0.2,0.2,0.2,0.2)) # set small margins
for (i in inds) {
  plot.digit(threes[i,])
}
```

(e) Looking at these digits, what can be said about the nature of the first two principal component scores? (The first principal component score is increasing as you move from left-to-right in any of the rows. The second principal component score is decreasing as you move from top-to-bottom in any of the columns.) In other words, explain what changes with respect to changes in each of the component scores.

(f) Plot the proportion of variance explained by the first k principal component directions, as a function of $k = 1, \ldots, 256$. How many principal component directions would we need to explain 50% of the variance? How many to explain 90% of the variance?

7. In a study of the anatomy of the North America marten, four bone dimensions were measured on 92 male specimens. The following wing bone dimensions were of particular interest:

X_1 : Humerus length

X_2: Femur length

X_3: Humerus width

X_4: Femur width

From the gathered data, the following sample covariance and correlation matrices were obtained:

$$\mathbf{S} = \begin{pmatrix} 1.1544 & 1.0330 & 0.9109 & 0.7993 \\ 1.0330 & 1.2100 & 0.7056 & 0.7953 \\ 0.9109 & 0.7056 & 2.0381 & 1.4083 \\ 0.7993 & 0.7953 & 1.4083 & 2.0277 \end{pmatrix} \quad \text{and} \quad \mathbf{R} = \begin{pmatrix} 1.0000 & 0.8740 & 0.5939 & 0.7993 \\ 0.8740 & 1.0000 & 0.4493 & 0.7953 \\ 0.5939 & 0.4493 & 1.0000 & 1.4083 \\ 0.5224 & 0.5077 & 0.6928 & 1.0000 \end{pmatrix}$$

IN an attempt to summarize and better interpret their findings, the people involved in this study carried through a PCA working from the sample covariance matrix **S**.

(a) In this case, a principal components analysis carried from either **S** or **R** can be seen to yield fairly similar results. Why do you think this is?

(b) Still, the results would differ somehow. In what way do you expect them to differ?

(c) Perform a PCA using both **S** and **R** and write a report. Justify why retaining 2 or 3 principal components would be reasonable in summarizing these data. Give at least 2 different arguments.

(d) Interpret the first three principal components.