# Chapter 13

# Principal Component Analysis

## Contents



---

**Principal component analysis** (PCA) is a multivariate technique for understanding variation, and for summarizing measurement data possibly through **variable reduction**. Principal components (the variables created in PCA) are sometimes used in addition to, or in place of, the original variables in certain analyses. I will illustrate the use and misuse of principal components in a series of examples.

---

Prof. Erik B. Erhardt

Given data on $p$ variables or features $X_1$, $X_2$, ..., $X_p$, PCA uses a rotation of the original coordinate axes to produce a **new** set of $p$ **uncorrelated** variables, called principal components, that are **unit-length linear combinations** of the original variables. A unit-length linear combination $a_1X_1+a_2X_2+\cdots+a_pX_p$ has $a_1^2 + a_2^2 + \cdots + a_p^2 = 1$.

The principal components have the following properties. The **first principal component**

$$\text{PC1} = a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p$$

has the **largest variability** among all unit-length linear combinations of the original variables. The **second principal component**

$$\text{PC2} = a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p$$

has the largest variability among all unit-length linear combinations of $X_1$, $X_2$, ..., $X_p$ that are uncorrelated with PC1. In general, the $j^{th}$ principal component PC$j$ for $j = 1, 2, \ldots, p$, has the largest variability among all unit-length linear combinations of the features that are uncorrelated with PC1, PC2, ..., PC$(j-1)$. The **last** or $p^{th}$ **principal component** PC$p$ has the **smallest variability** among all unit-length linear combinations of the features.

I have described PCA on the raw or unstandardized data. This method is often called PCA on the sample covariance matrix, because the principal components are computed numerically using a **singular value decomposition** of the sample covariance matrix for the $X_i$s. The variances of the PCs are **eigenvalues** of the sample covariance matrix. The coefficients in the PCs are **eigenvectors** of the sample covariance matrix. The sum of the variances of the principal components is equal to the sum of the variances in the original features. An alternative method for PCA uses standardized data, which is often called PCA on the correlation matrix.

The ordered principal components are uncorrelated variables with progressively less variation. Principal components are often viewed as separate dimensions corresponding to the collection of features. The variability of each

component divided by the total variability of the components is the proportion of the total variation in the data captured by each component. If data reduction is your goal, then you might need only the first few principal components to capture most of the variability in the data. This issue will be returned to later.

The unit-length constraint on the coefficients in PCA is needed to make the maximization well-defined. Without this constraint, there does not exist a linear combination with maximum variation. For example, the variability of an arbitrary linear combination $a_1X_1 + a_2X_2 + \cdots + a_pX_p$ is increased by 100 when each coefficient is multiplied by 10!

The principal components are unique only up to a change of the sign for each coefficient. For example,

$$\text{PC1} = 0.2X_1 - 0.4X_2 + 0.4X_3 + 0.8X_4$$

and

$$\text{PC1} = -0.2X_1 + 0.4X_2 - 0.4X_3 - 0.8X_4$$

have the same variability, so either could play the role of the first principal component. This non-uniqueness does not have an important impact on the analysis. Choose the direction of the component for a more natural interpretation and multiply by $-1$ if necessary.

# 13.1 Example: Temperature Data

The following temperature example includes mean monthly temperatures in January and July for 64 U.S. cities.

```
library(tidyverse)

# load ada functions
source("ada_functions.R")

#### Example: Temperature of cities
## The Temperature data file is in "fixed width format",
##    an older but very efficient data file format.
## Each field is specified by column ranges.
## Below I've provided numbers to help identify the column numbers
##   as well as the first three observations in the dataset.
## 12345678901234567890123456
## [  14 char   ][ 5 ][ 5 ]
#  mobile         51.2 81.6
#  phoenix        51.2 91.2
#  little rock    39.5 81.4

dat_temp <-
  read_fwf(
    "http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch13_temperature.dat"
  , col_positions =
      fwf_widths(
        widths = c(14, 5, 5)
      , col_names = c("city", "january", "july") # col names go here
      )
  ) %>%
  mutate(
    id = 1:n()    # Add an ID number for plotting
  )
```

```
## Parsed with column specification:
## cols(
##   city = col_character(),
##   january = col_double(),
##   july = col_double()
## )
```

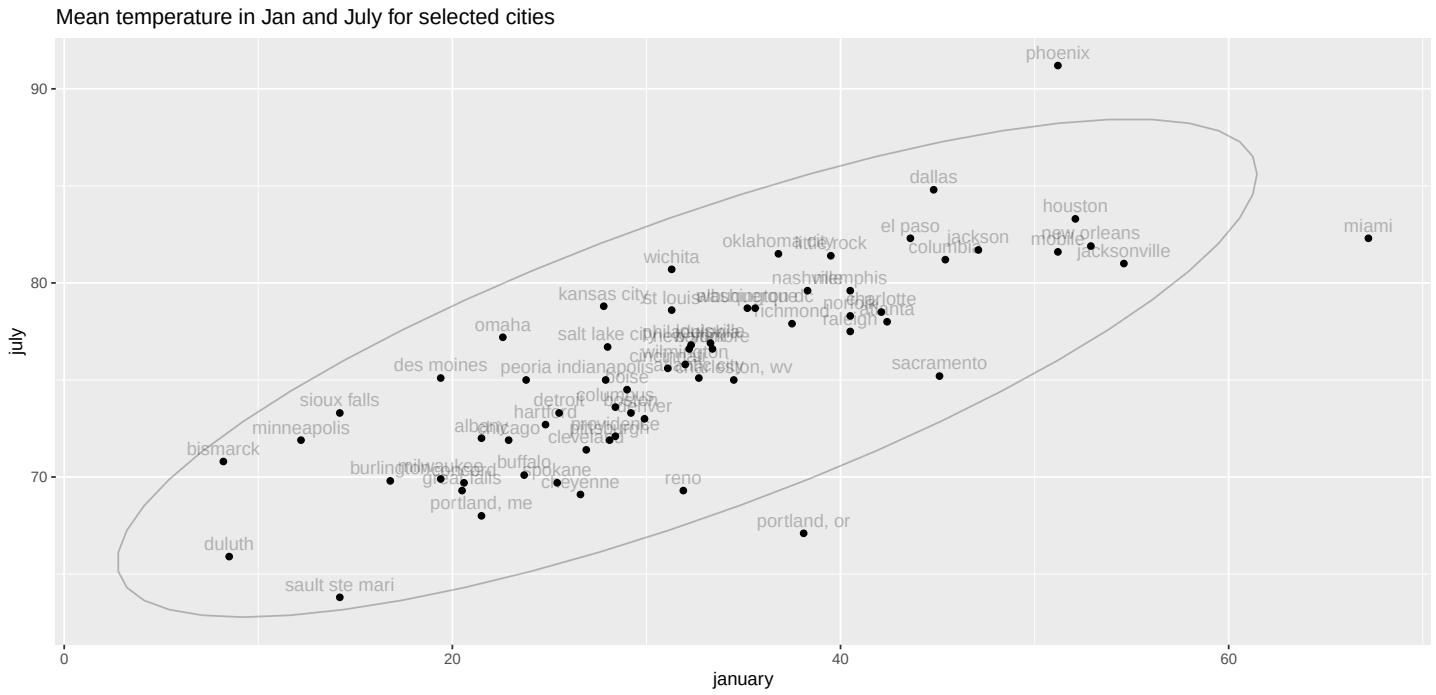
```
str(dat_temp)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 64 obs. of  4 variables:
##  $ city   : chr  "mobile" "phoenix" "little rock" "sacramento" ...
##  $ january: num  51.2 51.2 39.5 45.1 29.9 24.8 32 35.6 54.6 67.2 ...
##  $ july   : num  81.6 91.2 81.4 75.2 73 72.7 75.8 78.7 81 82.3 ...
##  $ id     : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
head(dat_temp)
```

```
## # A tibble: 6 x 4
##   city        january  july    id
##   <chr>         <dbl> <dbl> <int>
## 1 mobile         51.2  81.6     1
## 2 phoenix        51.2  91.2     2
## 3 little rock    39.5  81.4     3
## 4 sacramento     45.1  75.2     4
## 5 denver         29.9  73       5
## 6 hartford       24.8  72.7     6
```

```
# plot original data
library(ggplot2)
p1 <- ggplot(dat_temp, aes(x = january, y = july))
p1 <- p1 + geom_point()            # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                   # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = city), vjust = -0.5, alpha = 0.25) # city labels
p1 <- p1 + stat_ellipse(type = "norm", alpha = 1/4)
p1 <- p1 + labs(title = "Mean temperature in Jan and July for selected cities")
print(p1)
```




The `princomp()` procedure is used for PCA. By default the principal components are computed based on the covariance matrix. The correlation matrix may also be used (it effectively $z$-scores the data first) with the `cor = TRUE` option. The **principal component scores** are the values of the principal components across cases. The principal component scores PC1, PC2, ..., PC$p$ are centered to have mean zero.

Output from a PCA on the covariance matrix is given. Two principal components are created because $p = 2$.

```
# perform PCA on covariance matrix
pca_temp <-
  princomp(
    ~ january + july
  , data = dat_temp
  )
# standard deviation and proportion of variation for each component
pca_temp %>% summary()
```

```
## Importance of components:
##                             Comp.1    Comp.2
## Standard deviation     12.3217642 3.0004557
## Proportion of Variance  0.9440228 0.0559772
## Cumulative Proportion   0.9440228 1.0000000
```

```
# coefficients for PCs
pca_temp %>% loadings()
```

```
##
## Loadings:
##         Comp.1 Comp.2
## january  0.939  0.343
## july     0.343 -0.939
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

```
# scores are coordinates of each observation on PC scale
head(pca_temp$scores)
```

```
##      Comp.1     Comp.2
## 1 20.000106  0.9239612
## 2 23.291460 -8.0941867
## 3  8.940669 -2.8994977
## 4 12.075589  4.8446790
## 5 -2.957414  1.7000283
## 6 -7.851160  0.2333138
```

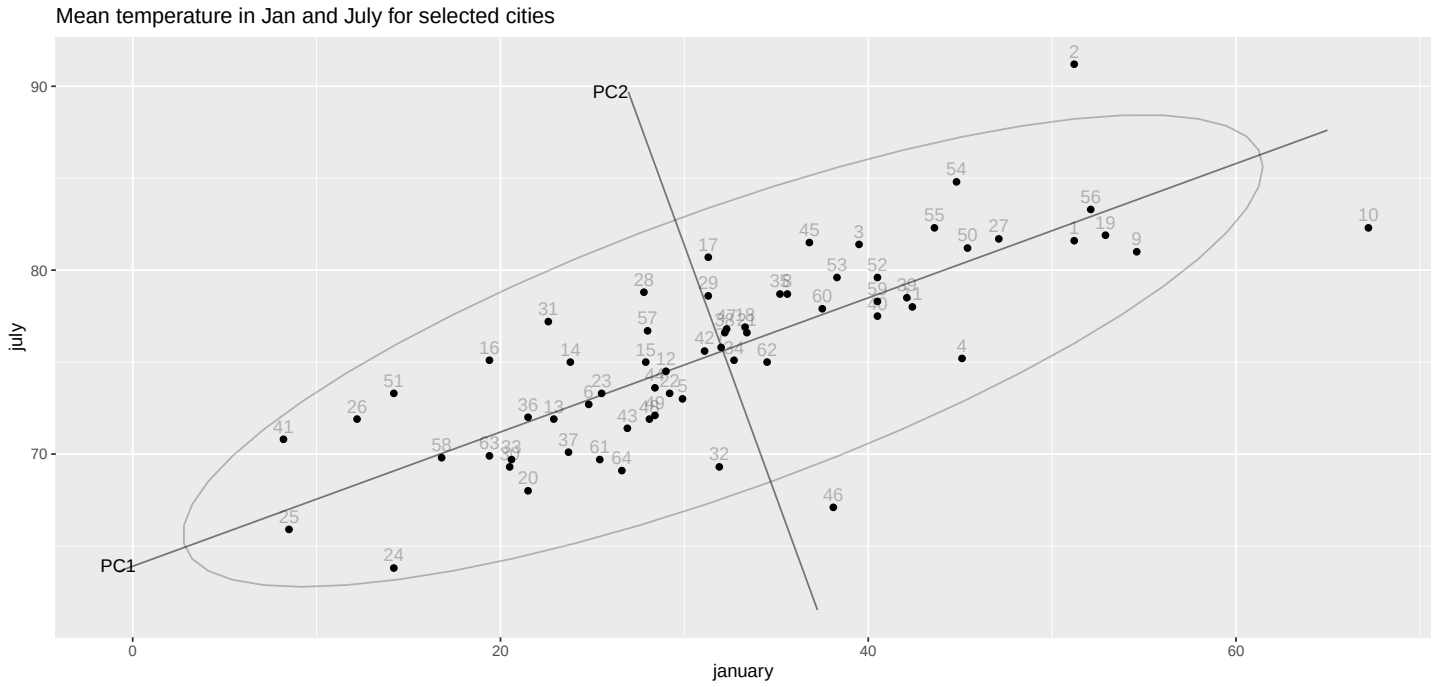
PCA is effectively doing a location shift (to the origin, zero) and a rotation of the data. When the correlation is used for PCA (instead of the covariance), it also performs a scaling so that the resulting PC scores have unit-variance in all directions.

```
# create small data.frame with endpoints of PC lines through data
line_scale <- c(35, 15) # length of PCA lines to draw
# endpoints of lines to draw
```

```
pca_temp_line_endpoints <-
  data.frame(
    PC = c(rep("PC1", 2), rep("PC2", 2))
  , x  = c(pca_temp$center[1] - line_scale[1] * pca_temp$loadings[1, 1]
         , pca_temp$center[1] + line_scale[1] * pca_temp$loadings[1, 1]
         , pca_temp$center[1] - line_scale[2] * pca_temp$loadings[1, 2]
         , pca_temp$center[1] + line_scale[2] * pca_temp$loadings[1, 2])
  , y  = c(pca_temp$center[2] - line_scale[1] * pca_temp$loadings[2, 1]
         , pca_temp$center[2] + line_scale[1] * pca_temp$loadings[2, 1]
         , pca_temp$center[2] - line_scale[2] * pca_temp$loadings[2, 2]
         , pca_temp$center[2] + line_scale[2] * pca_temp$loadings[2, 2])
  )
pca_temp_line_endpoints
```

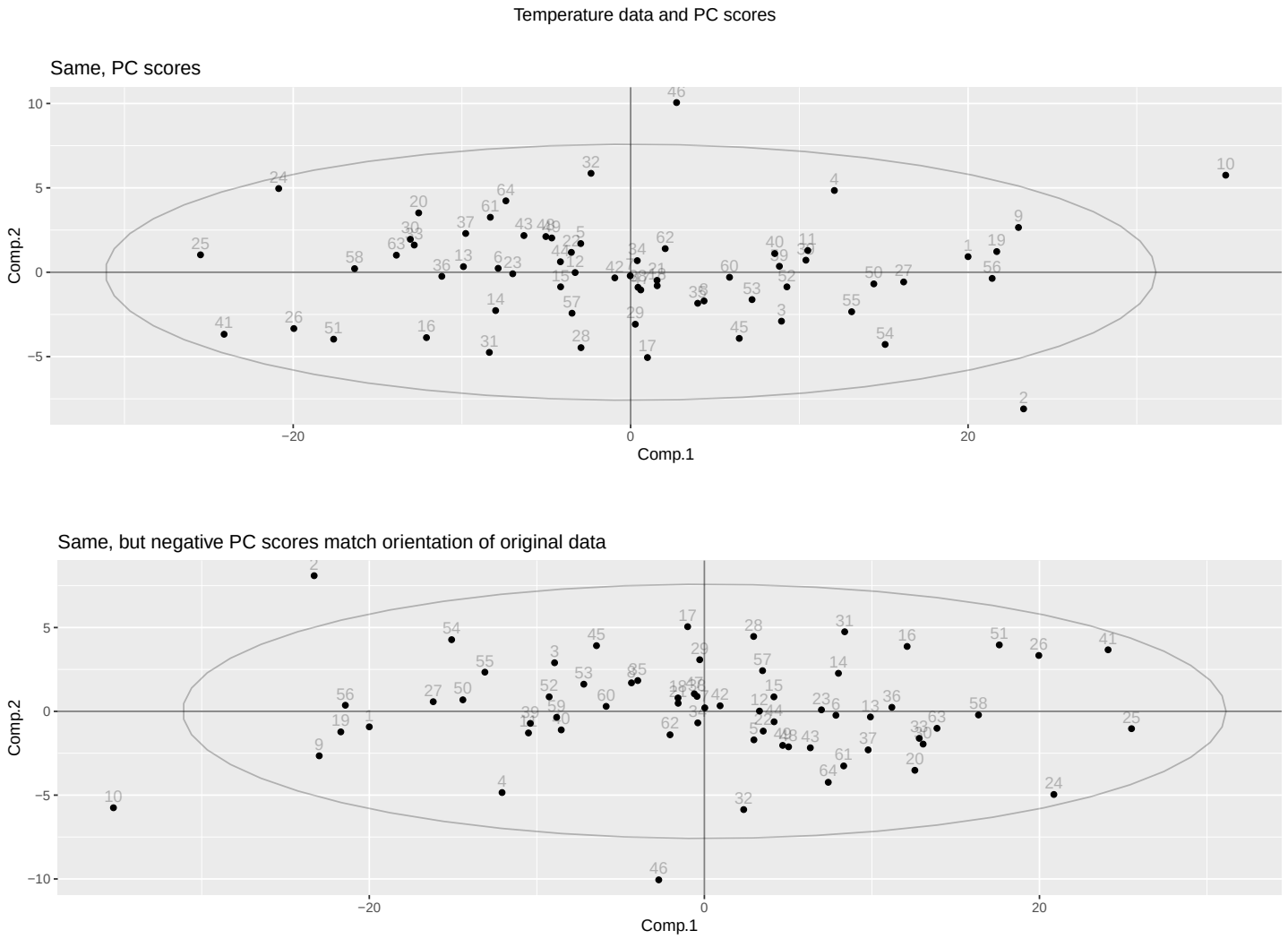
```
##    PC          x        y
## 1 PC1 -0.7833519 63.61121
## 2 PC1 64.9739769 87.61066
## 3 PC2 26.9525727 89.70179
## 4 PC2 37.2380523 61.52008
```

```
# plot original data with PCA vectors overlayed
library(ggplot2)
p1 <- ggplot(dat_temp, aes(x = january, y = july))
p1 <- p1 + geom_point()           # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = id), vjust = -0.5, alpha = 0.25) # city labels
p1 <- p1 + stat_ellipse(type = "norm", alpha = 1/4)
# plot PC lines
p1 <- p1 + geom_path(data = subset(pca_temp_line_endpoints, PC=="PC1"), aes(x=x, y=y)
                     , alpha=0.5)
p1 <- p1 + geom_path(data = subset(pca_temp_line_endpoints, PC=="PC2"), aes(x=x, y=y)
                     , alpha=0.5)
# label lines
p1 <- p1 + annotate("text"
                  , x     = pca_temp_line_endpoints$x[1]
                  , y     = pca_temp_line_endpoints$y[1]
                  , label = as.character(pca_temp_line_endpoints$PC[1])
                  , vjust = 0) #, size  = 10)
p1 <- p1 + annotate("text"
                  , x     = pca_temp_line_endpoints$x[3]
                  , y     = pca_temp_line_endpoints$y[3]
                  , label = as.character(pca_temp_line_endpoints$PC[3])
                  , hjust = 1) #, size  = 10)
p1 <- p1 + labs(title = "Mean temperature in Jan and July for selected cities")
print(p1)
```




```
# plot PCA scores (data on PC-scale centered at 0)
library(ggplot2)
p2 <- ggplot(as.data.frame(pca_temp$scores), aes(x = Comp.1, y = Comp.2))
p2 <- p2 + geom_point()            # points
p2 <- p2 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                   # good idea since both are in the same units
p2 <- p2 + geom_text(aes(label = rownames(pca_temp$scores)), vjust = -0.5, alpha = 0.25) # ci
p2 <- p2 + stat_ellipse(type = "norm", alpha = 1/4)
# plot PC lines
p2 <- p2 + geom_vline(xintercept = 0, alpha=0.5)
p2 <- p2 + geom_hline(yintercept = 0, alpha=0.5)
p2 <- p2 + labs(title = "Same, PC scores")
#print(p2)

# plot PCA scores (data on (negative) PC-scale centered at 0)
library(ggplot2)          # negative pca_temp$scores
p3 <- ggplot(as.data.frame(-pca_temp$scores), aes(x = Comp.1, y = Comp.2))
p3 <- p3 + geom_point()            # points
p3 <- p3 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                   # good idea since both are in the same units
p3 <- p3 + geom_text(aes(label = rownames(pca_temp$scores)), vjust = -0.5, alpha = 0.25) # ci
p3 <- p3 + stat_ellipse(type = "norm", alpha = 1/4)
# plot PC lines
p3 <- p3 + geom_vline(xintercept = 0, alpha=0.5)
p3 <- p3 + geom_hline(yintercept = 0, alpha=0.5)
p3 <- p3 + labs(title = "Same, but negative PC scores match orientation of original data")
#print(p3)

library(gridExtra)
```

```
grid.arrange(grobs = list(p2, p3), ncol=1, top="Temperature data and PC scores")
```




Some comments on the output:

1. You can visualize PCA when $p = 2$. In the temperature plot, the direction of maximal variability corresponds to the first PC axis. The PC1 score for each city is obtained by projecting the temperature pairs perpendicularly onto this axis. The direction of minimum variation corresponds to the second PC axis, which is perpendicular to the first PC axis. The PC2 score for each city is obtained by projecting the temperature pairs onto this axis.
2. The **total variance** is the sum of variances for the monthly temperatures: $163.38 = 137.18 + 26.20$.

```
# variance of data (on diagonals, covariance of off-diags)
dat_temp[,c("january","july")] %>% var()
```

```
##           january     july
## january 137.1811 46.72910
## july     46.7291 26.20035

# sum of variance
dat_temp[,c("january","july")] %>% var() %>% diag() %>% sum()

## [1] 163.3814

# variance of PC scores
pca_temp$scores %>% var()

##                Comp.1        Comp.2
## Comp.1  1.542358e+02 -1.831125e-15
## Comp.2 -1.831125e-15  9.145635e+00

# sum is same as original data
pca_temp$scores %>% var() %>% diag() %>% sum()

## [1] 163.3814
```

3. The **eigenvalues of the covariance matrix** are variances for the PCs. The variability of

$$\text{PC1} = +0.939\ \text{JAN} + 0.343\ \text{JULY}$$

is 154.236. The variability of

$$\text{PC2} = -0.343\ \text{JAN} + 0.939\ \text{JULY}$$

is 9.146. The proportion of the total variability due to PC1 is $0.944 = 154.23/163.38$. The proportion of the total variability due to PC2 is $0.056 = 9.146/163.38$.

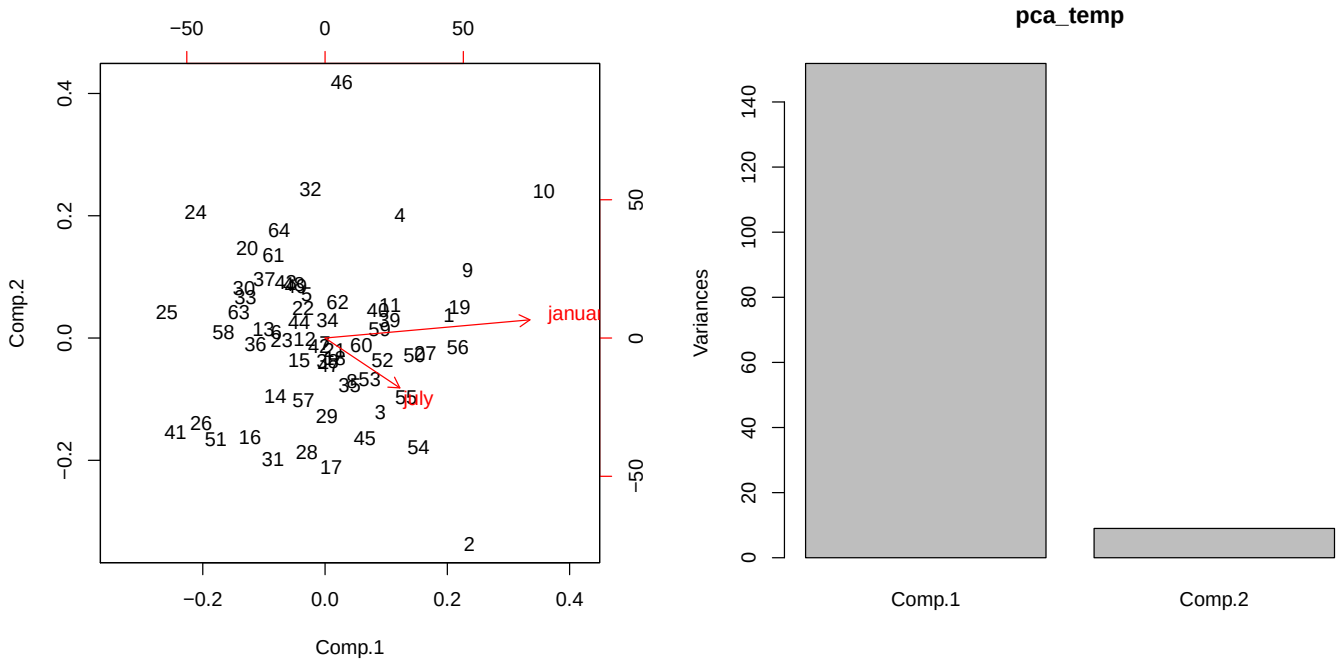
```
# eigenvalues and eigenvectors of covariance matrix give PC variance and loadings
dat_temp[,c("january","july")] %>% var() %>% eigen()

## eigen() decomposition
## $values
## [1] 154.235808   9.145635
##
## $vectors
##            [,1]       [,2]
## [1,] -0.9393904  0.3428493
## [2,] -0.3428493 -0.9393904
```

4. Almost all of the variability (94.4%) in the original temperatures is captured by the first PC. The second PC accounts for the remaining 5.6% of the total variability.

5. PC1 weights the January temperature about three times the July temperature. This is sensible because PC1 maximizes variation among linear combinations of the January and July temperatures. January temperatures are more variable, so they are weighted heavier in this linear combination.
6. The PCs PC1 and PC2 are standardized to have mean zero. This explains why some PC1 scores are negative, even though PC1 is a weighted average of the January and July temperatures, each of which is non-negative.

The built-in plots plot the scores and original data directions (biplot) and the screeplot shows the relative variance proportion of all components in decreasing order.

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(pca_temp)
screeplot(pca_temp)
```



## 13.2 PCA on the Correlation Matrix

The coefficients in the first principal component reflect the relative sizes of the feature variances. The features with large variances have larger coefficients or loadings. This might be considered a problem, because variability is **scale dependent** and the principal component analysis on the raw data does not take scale into account. For example, if you measure height in meters but then change height to centimeters, the variability increases by a factor of $100^2 = 10000$. This might have a dramatic effect on the PCA.

You might prefer to standardize the features when they are measured on different scales, or when the features have wildly different variances. The features are standardized to have mean zero and variance one by using the $Z$-score transformation: (Obs − Mean)/Std Dev. The PCA is then performed on the standardized data.

```
dat_temp_z <-
  dat_temp %>%
  mutate(
    # manual z-score
    january = (january - mean(january)) / sd(january)
    # z-score using R function scale()
  , july    = scale(july)
  )

# the manual z-score and scale() match
all.equal(
    dat_temp_z$january
  , scale(dat_temp_z$january) %>% as.vector()
  )
```

```
## [1] TRUE
```

```
# scale() includes attributes for the mean() and sd() used for z-scoring
str(dat_temp_z)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 64 obs. of  4 variables:
##  $ city   : chr  "mobile" "phoenix" "little rock" "sacramento" ...
##  $ january: num  1.631 1.631 0.632 1.11 -0.187 ...
##  $ july   : num [1:64, 1] 1.1701 3.0456 1.131 -0.0803 -0.5101 ...
##   ..- attr(*, "scaled:center")= num 75.6
##   ..- attr(*, "scaled:scale")= num 5.12
##  $ id     : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
head(dat_temp_z)
```

```
## # A tibble: 6 x 4
```

```
##   city         january july[,1]    id
##   <chr>          <dbl>    <dbl> <int>
## 1 mobile         1.63     1.17      1
## 2 phoenix        1.63     3.05      2
## 3 little rock    0.632    1.13      3
## 4 sacramento     1.11    -0.0803    4
## 5 denver        -0.187   -0.510     5
## 6 hartford      -0.623   -0.569     6
```

```
# z-scored data has mean 0 and variance 1
dat_temp_z[,c("january","july")] %>% colMeans()
```

```
##       january          july
##  1.228943e-16 -1.214842e-15
```

```
dat_temp_z[,c("january","july")] %>% var()
```

```
##           january      july
## january 1.0000000 0.7794472
## july    0.7794472 1.0000000
```

```
# the correlation is used to contruct the PCs
#   (correlation is the same as covariance for z-scored data)
dat_temp_z[,c("january","july")] %>% cor()
```

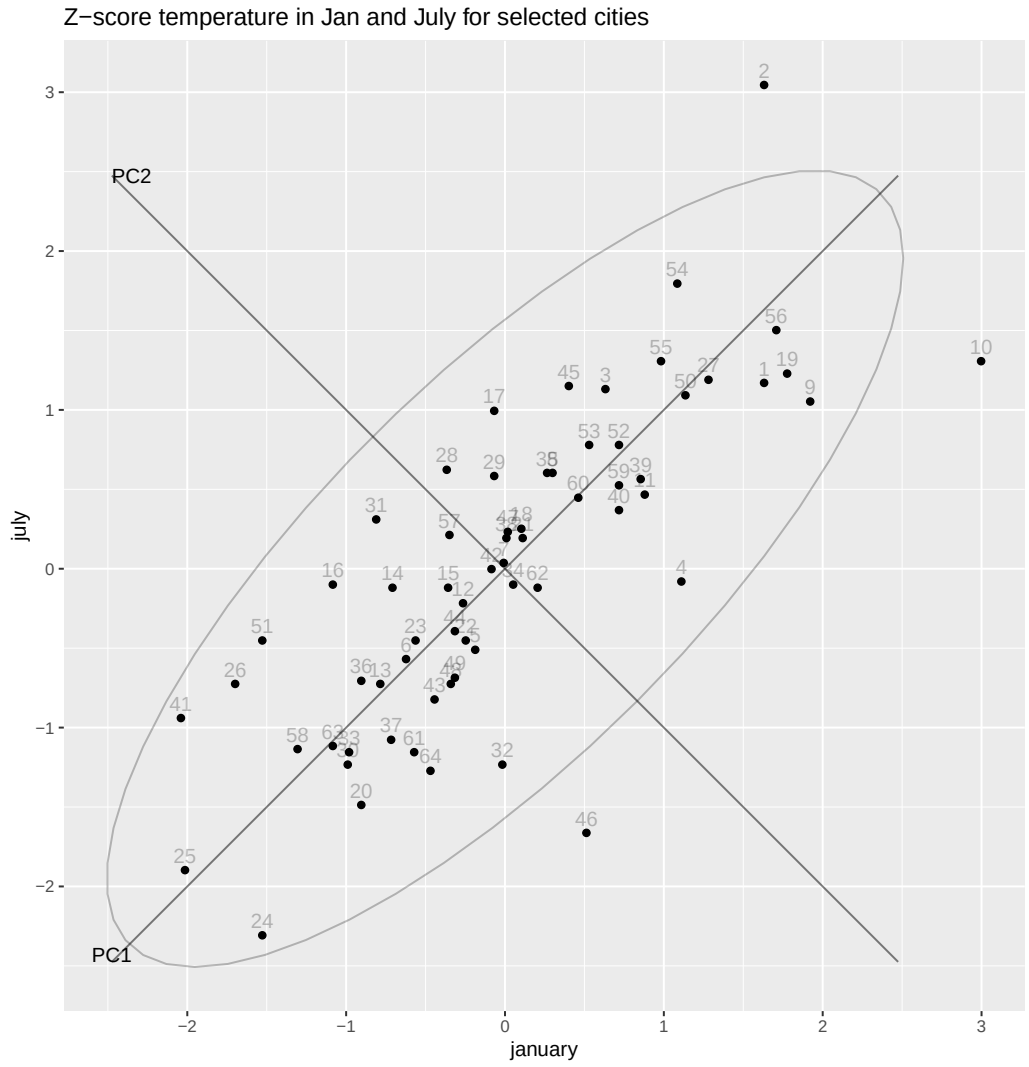
```
##           january      july
## january 1.0000000 0.7794472
## july    0.7794472 1.0000000
```

```
## Plot z-scored data
pca_temp_z <-
  princomp(
    ~ january + july
  , data = dat_temp_z
  )

# create small data.frame with endpoints of PC lines through data
line_scale <- c(3.5, 3.5) # length of PCA lines to draw
# endpoints of lines to draw
pca_temp_z_line_endpoints <-
  data.frame(
    PC = c(rep("PC1", 2), rep("PC2", 2))
  , x  = c(pca_temp_z$center[1] - line_scale[1] * pca_temp_z$loadings[1, 1]
         , pca_temp_z$center[1] + line_scale[1] * pca_temp_z$loadings[1, 1]
         , pca_temp_z$center[1] - line_scale[2] * pca_temp_z$loadings[1, 2]
         , pca_temp_z$center[1] + line_scale[2] * pca_temp_z$loadings[1, 2])
  , y  = c(pca_temp_z$center[2] - line_scale[1] * pca_temp_z$loadings[2, 1]
         , pca_temp_z$center[2] + line_scale[1] * pca_temp_z$loadings[2, 1]
         , pca_temp_z$center[2] - line_scale[2] * pca_temp_z$loadings[2, 2]
         , pca_temp_z$center[2] + line_scale[2] * pca_temp_z$loadings[2, 2])
  )
pca_temp_z_line_endpoints
```

```
##    PC        x        y
## 1 PC1 -2.474874 -2.474874
## 2 PC1  2.474874  2.474874
## 3 PC2 -2.474874  2.474874
## 4 PC2  2.474874 -2.474874
```

```
# plot original data with PCA vectors overlayed
library(ggplot2)
p1 <- ggplot(dat_temp_z, aes(x = january, y = july))
p1 <- p1 + geom_point()           # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = id), vjust = -0.5, alpha = 0.25) # city labels
p1 <- p1 + stat_ellipse(type = "norm", alpha = 1/4)
```

```
# plot PC lines
p1 <- p1 + geom_path(data = subset(pca_temp_z_line_endpoints, PC=="PC1"), aes(x=x, y=y), alpha=0.5)
p1 <- p1 + geom_path(data = subset(pca_temp_z_line_endpoints, PC=="PC2"), aes(x=x, y=y), alpha=0.5)
# label lines
p1 <- p1 + annotate("text"
                , x     = pca_temp_z_line_endpoints$x[1]
                , y     = pca_temp_z_line_endpoints$y[1]
                , label = as.character(pca_temp_z_line_endpoints$PC[1])
                , vjust = 0) #, size  = 10)
p1 <- p1 + annotate("text"
                , x     = pca_temp_z_line_endpoints$x[3]
                , y     = pca_temp_z_line_endpoints$y[3]
                , label = as.character(pca_temp_z_line_endpoints$PC[3])
                , hjust = 0) #, size  = 10)
p1 <- p1 + labs(title = "Z-score temperature in Jan and July for selected cities")
print(p1)
```




The covariance matrix computed from the standardized data is the correlation matrix. Thus, principal components based on the standardized data are computed from the correlation matrix. This is implemented by adding the `cor = TRUE` option on the `princomp()` procedure statement.

```
# perform PCA on correlation matrix
pca_temp2 <-
```

```
  princomp(
    ~ january + july
  , data = dat_temp
  , cor  = TRUE
  )
# standard deviation and proportion of variation for each component
pca_temp2 %>% summary()
```

```
## Importance of components:
##                           Comp.1    Comp.2
## Standard deviation     1.3339592 0.4696305
## Proportion of Variance 0.8897236 0.1102764
## Cumulative Proportion  0.8897236 1.0000000
```

```
# coefficients for PCs
pca_temp2 %>% loadings()
```

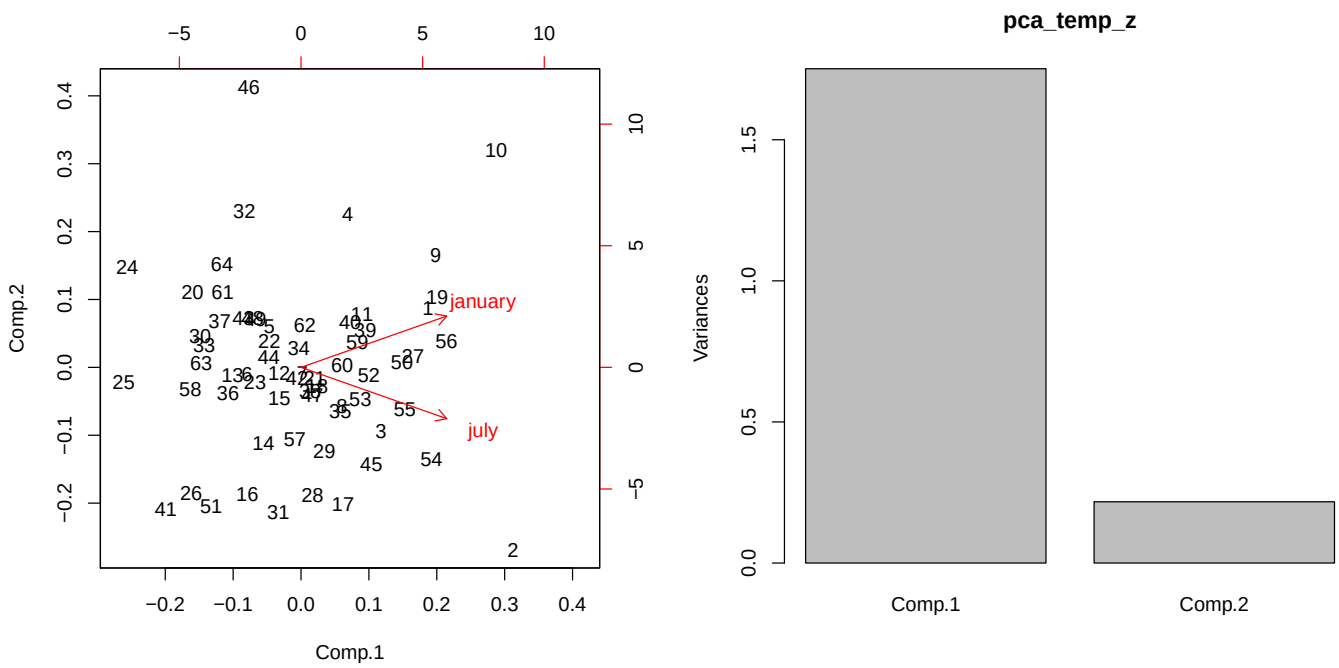
```
##
## Loadings:
##         Comp.1 Comp.2
## january  0.707  0.707
## july     0.707 -0.707
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

```
# scores are coordinates of each observation on PC scale
head(pca_temp2$scores)
```

```
##       Comp.1     Comp.2
## 1  1.9964045  0.3286199
## 2  3.3330689 -1.0080444
## 3  1.2566173 -0.3554730
## 4  0.7341125  0.8485470
## 5 -0.4971200  0.2299523
## 6 -0.8492236 -0.0386098
```

This plot is the same except for the top/right scale around the biplot and the variance scale on the screeplot.

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(pca_temp_z)
screeplot(pca_temp_z)
```




The standardized features are dimensionless, so the PCs are not influenced by the original units of measure, nor are they affected by the variability in the features. The only important factor is the correlation between the features, which is not changed by standardization.

The PCs from the correlation matrix are

$$\text{PC1} = +0.707\ \text{JAN} + 0.707\ \text{JULY}$$

and

$$\text{PC2} = -0.707\ \text{JAN} + 0.707\ \text{JULY}.$$

PCA is an exploratory tool, so neither a PCA on the covariance matrix nor a PCA on the correlation matrix is always the "right" method. I often do both and see which analysis is more informative.

## 13.3 Interpreting Principal Components

You should try to interpret the linear combinations created by multivariate techniques. The interpretations are usually non-trivial, and some degree of creativity is involved.

The **coefficients** or **loadings** in a principal component reflect the **relative** contribution of the features to the linear combination. Most researchers focus more on the **signs** of the coefficents than on the **magnitude** of the coefficients. The principal components are then interpreted as weighted averages or comparisons of weighted averages.

A **weighted average** is a linear combination of features with non-negative loadings. The simplest case of a weighted average is an arithmetic average, where each feature has the same coefficient.

The difference $Z = X - Y$ is a **comparison** of $X$ and $Y$. The sign and magnitude of $Z$ indicates which of $X$ and $Y$ is larger, and by how much. To see this, note that $Z = 0$ if and only if $X = Y$, whereas $Z < 0$ when $X < Y$ and $Z > 0$ when $X > Y$.

In the temperature data, PC1 is a weighted average of January and July temperatures (signs of the loadings: JAN is + and JULY is +):

$$\text{PC1} = +0.94\ \text{JAN} + 0.34\ \text{JULY}.$$

PC2 is a comparison of January and July temperatures (signs of the loadings: JAN is − and JULY is +):

$$\text{PC2} = -0.34\ \text{JAN} + 0.94\ \text{JULY}.$$

Principal components often have positive and negative loadings when $p \geq 3$. To interpret the components, group the features with + and − signs together and then interpret the linear combination as a comparison of weighted averages.

You can often simplify the interpretation of principal components by **mentally eliminating** features from the linear combination that have relatively small (in magnitude) loadings or coefficients. This strategy does not carry over to all multivariate analyses, so I will be careful about this issue when necessary.

## 13.4 Example: Painted turtle shells

Jolicouer and Mosimann gave the length, width, and height in mm of the carapace (shell) for a sample of 24 female painted turtles. I perform a PCA on the original data and on the standardized data.

```
#### Example: Painted turtle shells
dat_shells <-
  read_table2(
    "http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch13_shells.dat"
  )
```

```
## Parsed with column specification:
## cols(
##   length = col_double(),
##   width = col_double(),
##   height = col_double()
## )
```

```
str(dat_shells)
```

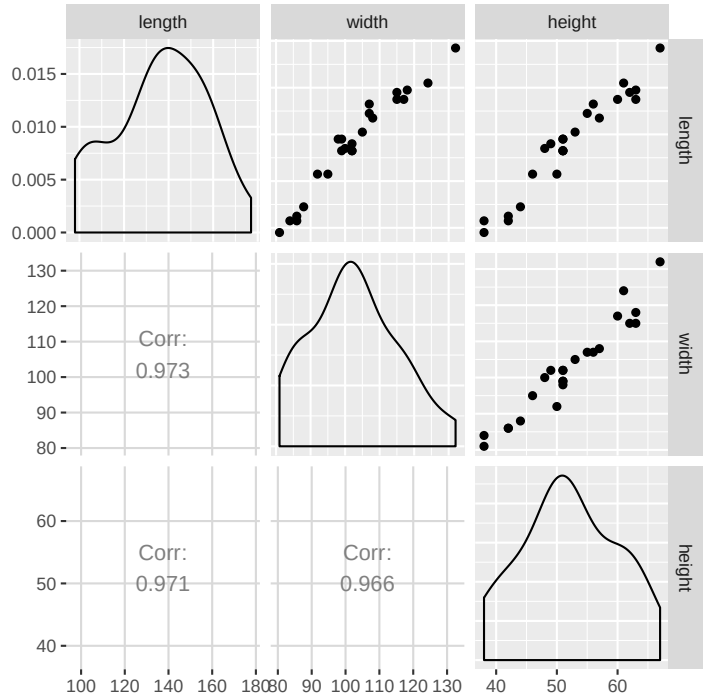
```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 24 obs. of  3 variables:
##  $ length: num  98 103 103 105 109 123 123 133 133 133 ...
##  $ width : num  81 84 86 86 88 92 95 99 102 102 ...
##  $ height: num  38 38 42 42 44 50 46 51 51 51 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   length = col_double(),
##   ..   width = col_double(),
##   ..   height = col_double()
##   .. )
```

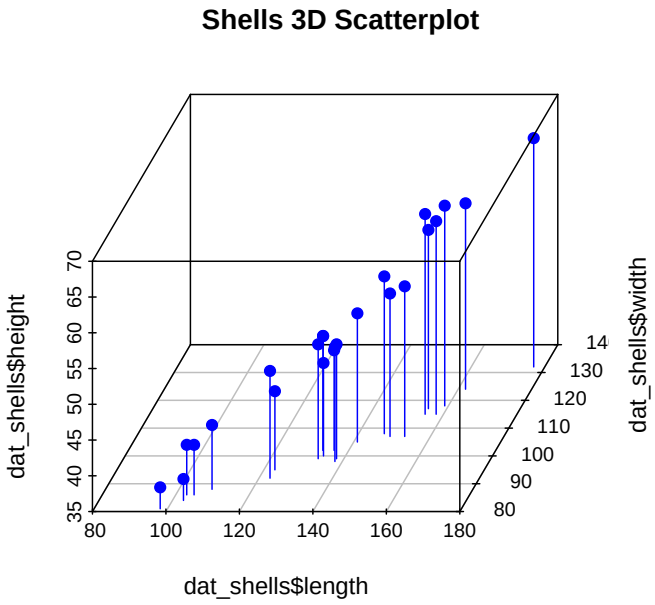
```
head(dat_shells)
```

```
## # A tibble: 6 x 3
##   length width height
##    <dbl> <dbl>  <dbl>
## 1     98    81     38
## 2    103    84     38
## 3    103    86     42
## 4    105    86     42
## 5    109    88     44
## 6    123    92     50
```

```
## Scatterplot matrix
library(ggplot2)
library(GGally)
# put scatterplots on top so y axis is vertical
p <-
  ggpairs(
    dat_shells
  , upper = list(continuous = "points")
```

```
  , lower = list(continuous = "cor")
  , progress = FALSE
  )
print(p)

## 3D scatterplot
library(scatterplot3d)
par(mfrow=c(1,1))
scatterplot3d(
    x     = dat_shells$length
  , y     = dat_shells$width
  , z     = dat_shells$height
  , main  = "Shells 3D Scatterplot"
  , type  =  "h"    # lines to the horizontal xy-plane
  , color = "blue"  # filled blue circles
  , pch   = 19      # filled blue circles
  , angle = 70      # perspective
  #, highlight.3d  =  TRUE  # makes color change with z-axis value
  )

#### For a rotatable 3D plot, use plot3d() from the rgl library
# ##   This uses the R version of the OpenGL (Open Graphics Library)
# #library(rgl)
# rgl::plot3d(x = dat_shells$length, y = dat_shells$width, z = dat_shells$height)
```



### 13.4.1 PCA on shells covariance matrix

The plots show that the shell measurements are strongly positively correlated, which is not surprising. Let us perform a PCA on the covariance matrix and interpret the results.

```
# perform PCA on covariance matrix
pca_shells <-
  princomp(
    ~ length + width + height
  , data = dat_shells
  )
# standard deviation and proportion of variation for each component
pca_shells %>% summary()
```

```
## Importance of components:
##                             Comp.1      Comp.2      Comp.3
## Standard deviation     25.4970668 2.547081962 1.653745717
## Proportion of Variance  0.9860122 0.009839832 0.004148005
## Cumulative Proportion   0.9860122 0.995851995 1.000000000
```

```
# coefficients for PCs
pca_shells %>% loadings()
```

```
##
## Loadings:
##        Comp.1 Comp.2 Comp.3
## length  0.814  0.555  0.172
## width   0.496 -0.818  0.291
## height  0.302 -0.151 -0.941
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

The three principal components from the raw data are given below. Length and width are grouped in PC3 because they have negative loadings.

$$\begin{aligned} \text{PC1} &= 0.81\ \text{Length} + 0.50\ \text{Width} + 0.30\ \text{Height} \\ \text{PC2} &= -0.55\ \text{Length} + (0.82\ \text{Width} + 0.15\ \text{Height}) \\ \text{PC3} &= -(0.17\ \text{Length} + 0.29\ \text{Width}) + 0.94\ \text{Height}. \end{aligned}$$

PC1 is a weighted average of the carapace measurements, and can be viewed as an overall measure of shell size. Jolicouer and Mosimann interpreted the second and third principal components as measures of **shape**, for they appear to

be a comparison of length with an average of width and height, and a comparison of height with length and width, respectively.

Jolicouer and Mosimann argue that the size of female turtle shells can be characterized by PC1 with little loss of information because this linear combination accounts for 98.6% of the total variability in the measurements. The form of PC1 makes sense conceptually. The carapace measurements are positively correlated with each other, so larger lengths tend to occur with larger widths and heights. The primary way the shells vary is with regards to their overall size, as measured by a weighted average of length, width, and height.

**Question:** Can PC2 and PC3, which have relatively little variation, be used in any meaningful way? To think about this, suppose the variability in PC2 and PC3 was zero.

### 13.4.2 PCA on shells correlation matrix

For the analysis on the correlation matrix, add the `cor = TRUE` option.

```
# perform PCA on correlation matrix
pca_shells <-
  princomp(
    ~ length + width + height
  , data = dat_shells
  , cor  = TRUE
  )
# standard deviation and proportion of variation for each component
pca_shells %>% summary()
```

```
## Importance of components:
##                           Comp.1    Comp.2      Comp.3
## Standard deviation     1.714584 0.1853043 0.160820482
## Proportion of Variance 0.979933 0.0114459 0.008621076
## Cumulative Proportion  0.979933 0.9913789 1.000000000
```

```
# coefficients for PCs
pca_shells %>% loadings()
```

```
##
## Loadings:
##        Comp.1 Comp.2 Comp.3
## length  0.578  0.137  0.804
## width   0.577  0.628 -0.522
## height  0.577 -0.766 -0.284
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

The three principal components for the standardized data are

$$\begin{aligned} \text{PC1} &= 0.58\ \text{Length} + 0.58\ \text{Width} + 0.58\ \text{Height} \\ \text{PC2} &= -(0.14\ \text{Length} + 0.63\ \text{Width}) + 0.77\ \text{Height} \\ \text{PC3} &= -0.80\ \text{Length} + (0.52\ \text{Width} + 0.28\ \text{Height}). \end{aligned}$$

The first principal component accounts for 98% of the total variability in the standardized data. The total variability for correlation is always the number $p$ of features because it is the sum of the variances. Here, $p = 3$. Little loss of information is obtained by summarizing the standardized data using PC1, which is essentially an average of length, width, and height. PC2 and PC3 are measures of shape.

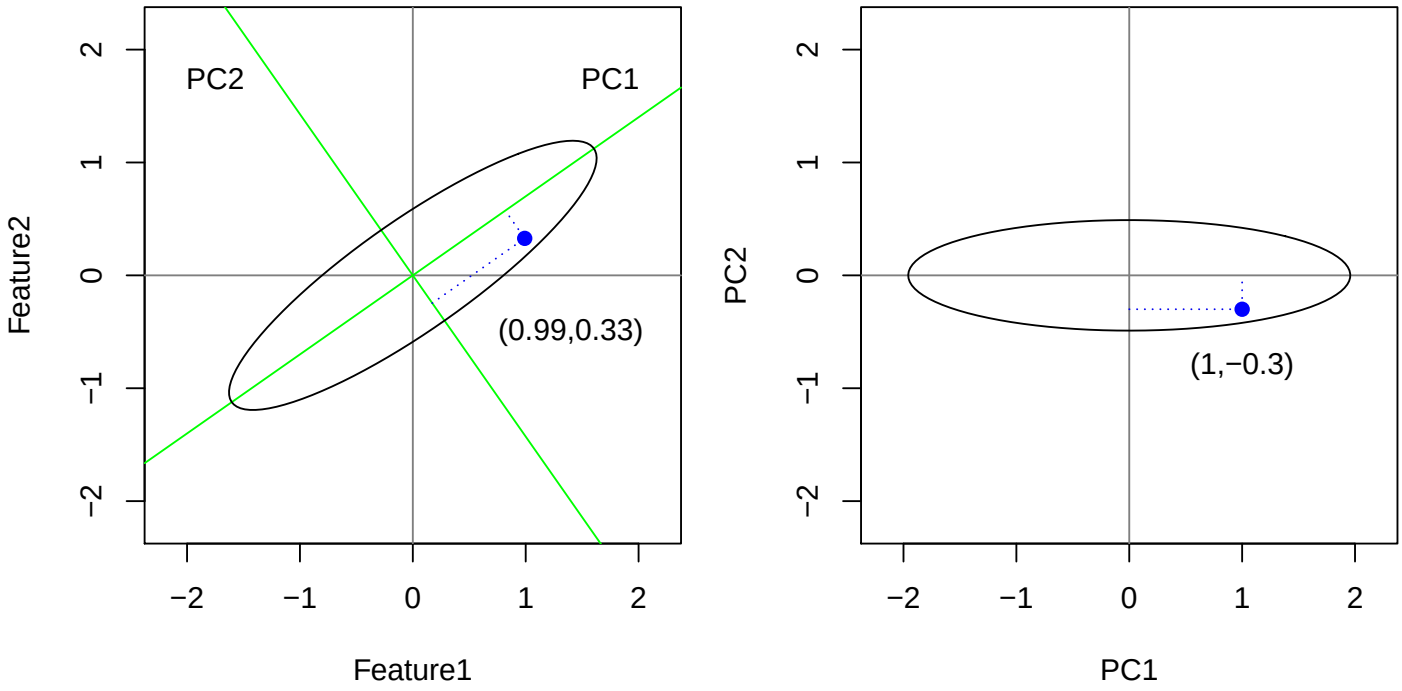
The loadings in the first principal component are approximately equal because the correlations between pairs of features are almost identical. The standardized features are essentially interchangeable with regards to the construction of the first principal component, so they must be weighted similarly. True, but not obvious.

## 13.5 Why is PCA a Sensible Variable Reduction Technique?

My description of PCA provides little insight into why principal components are reasonable summaries for **variable reduction**. Specifically, why does it make sense for researchers to consider the linear combination of the original features with the **largest variance** as the best single variable summary of the data?

There is an alternative description of PCA that provides more insight into this issue. For simplicity, consider the following data plot of two features, and the implied principal components. The PC scores for an observation are obtained by projecting the feature scores onto the axes of maximal and minimal variation, and then rotating the axes appropriately.

One can show mathematically that PC1 is the best (in some sense) linear combination of the two features to predict the original two features simultaneously. Intuitively, this is plausible. In a PCA, you know the direction for the axis of maximal variation. Given the value of PC1, you get a good prediction of the original feature scores by moving PC1 units along the axis of maximal variation in the feature space.



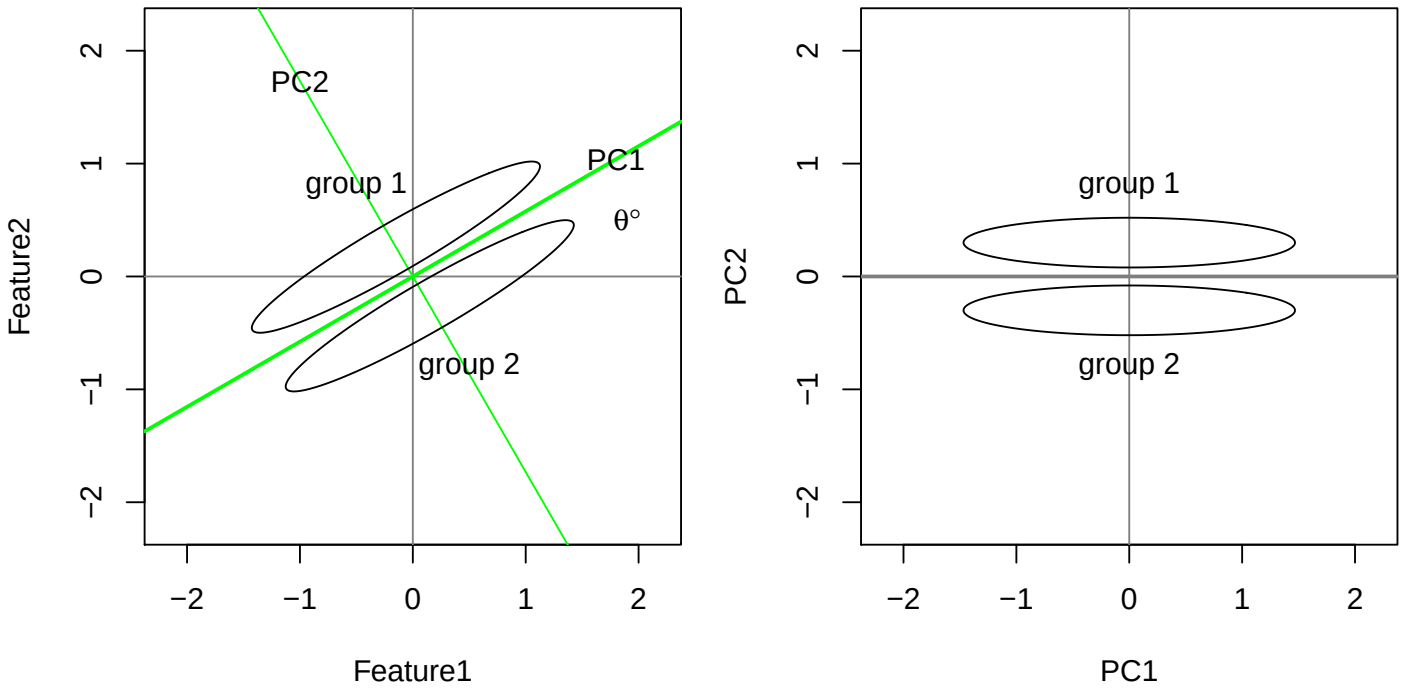

The LS line from regressing feature 2 on feature 1 gives the best prediction for feature 2 scores when the feature 1 score is known. Similarly, the LS line from regressing feature 1 on feature 2 gives the best prediction for feature 1 scores when the feature 2 score is known. PC1 is the best linear combination of features 1 and 2 to predict both features *simultaneously*. Note that feature 1 and feature 2 are linear combinations as well!

This idea generalizes. The first $k$ principal components give the best simultaneous prediction of the original $p$ features, among all possible choices of $k$ uncorrelated unit-length linear combinations of the features. Prediction of the original features improves as additional components are added, but the improvement is slight when the added principal components have little variability. Thus, summarizing the data using the principal components with maximum variation is a sensible strategy for data reduction.

### 13.5.1 A Warning on Using PCA as a Variable Reduction Technique

Some researchers view PCA as a "catchall" technique for reducing the number of variables that need to be considered in an analysis. They will replace the original variables with a small number of principal components that explain most of the variation in the original data and proceed with an analysis on the principal components.

This strategy is not always sensible, especially if a primary interest in the analysis is a comparison of heterogeneous groups. If the group structure was ignored in the PCA analysis, then the linear combinations retained by the researcher may contain little information for distinguishing among groups. For example, consider the following data plot on two features and two groups, and the implied principal components.

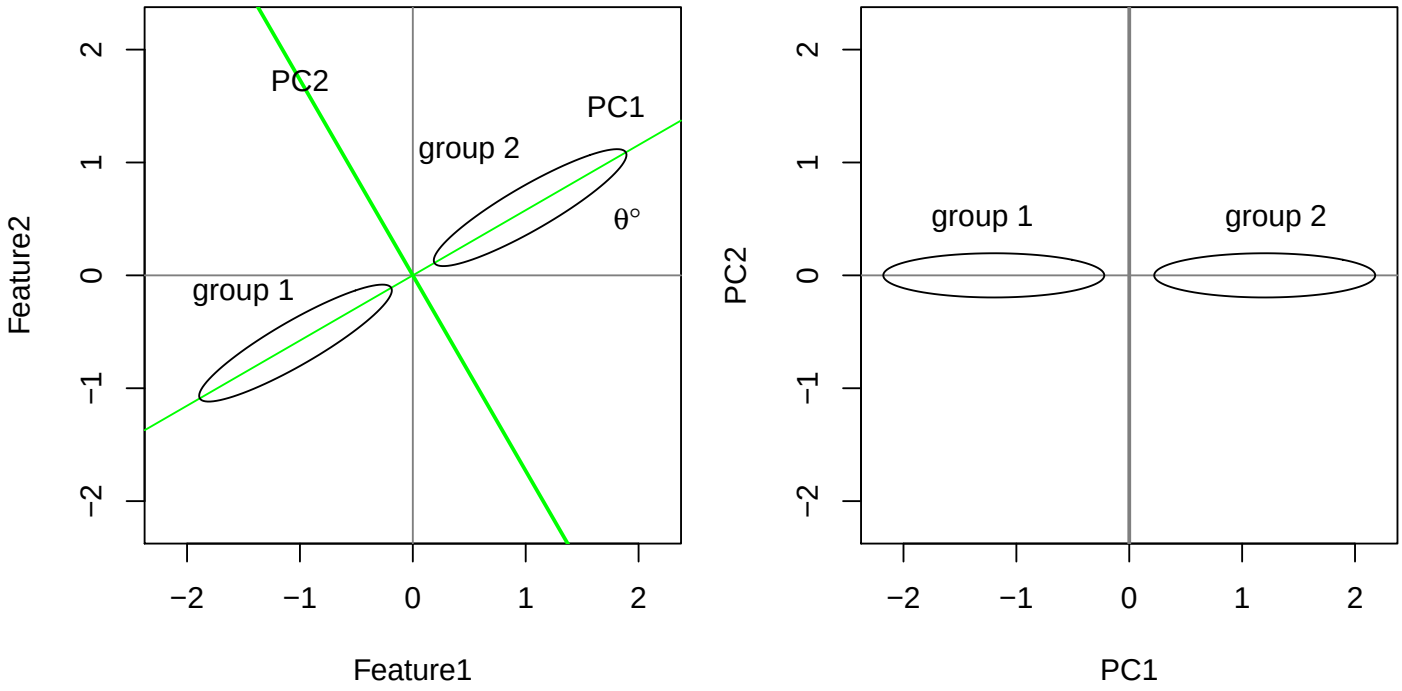



Although PC1 explains most of the variability in the two features (ignoring the groups), little of the total variation is due to group differences. If the

researcher reduced the two features to the first principal component, he would be throwing away most of the information for distinguishing between the groups. PC2 accounts for little of the total variation in the features, but most of the variation in PC2 is due to group differences.

If a comparison of the two groups was the primary interest, then the researcher should use discriminant analysis instead. Although there is little gained by reducing two variables to one, this principle always applies in multivariate problems. In discriminant analysis, a stepwise selection of variables can be implemented to eliminate features that have no information for distinguishing among groups. This is data reduction as it should be practiced — with a final goal in mind.

Variable reduction using PCA followed by group comparisons might be fruitful if you are fortunate enough to have the directions with large variation correspond to the directions of group differences. For example, in the plot below, the first principal component is a linear combination of the features that distinguishes between the groups. A comparison of the groups based on the first principal component will lead to similar conclusions as a discriminant analysis.




There is nothing unreasonable about using principal components in a comparison of groups, provided you recognize that the principal component scores with the largest variability need not be informative for group comparisons!

In summary, PCA should be used to summarize the variation **within a homogeneous group**, and should not, in general, be used as a data reduction tool prior to a comparison across groups. The same concern applies to using PCA for identifying groups of similar objects (use cluster analysis instead), or when **factor analysis** is used prior to a group comparison.

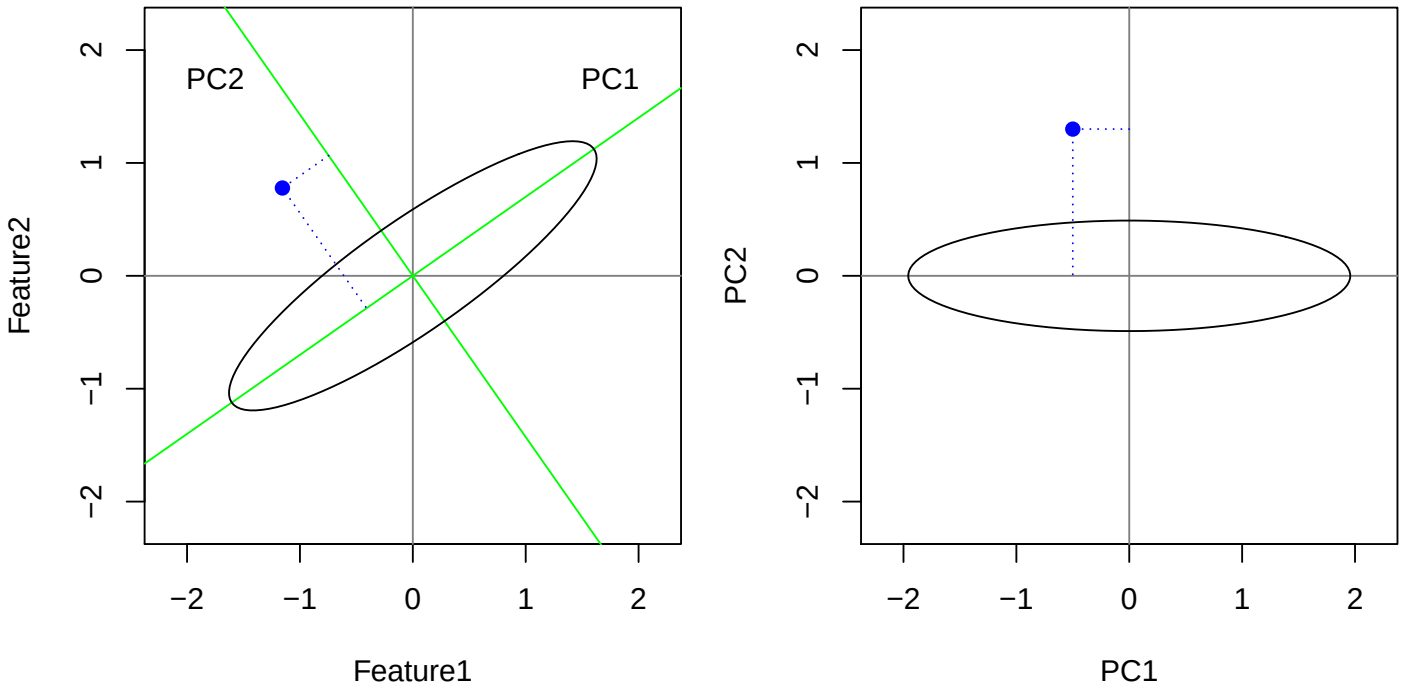
Additionally, Prof. Ed Bedrick[1] has recently derived the statistical properties of the two-step technique of PCA followed by a two-sample $t$-test. He showed that the size of the test (the Type-I error rate) can be distorted based on the mean and covariance structure and that the test has poor power. He provides a simple method to correct for the size of the test and provides recommendations.

[1]Bedrick, EJ. (2019). Data reduction prior to inference: Are there consequences of comparing groups using a $t$-test based on principal component scores? Biometrics. `https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.13159`

### 13.5.2 PCA is Used for Multivariate Outlier Detection

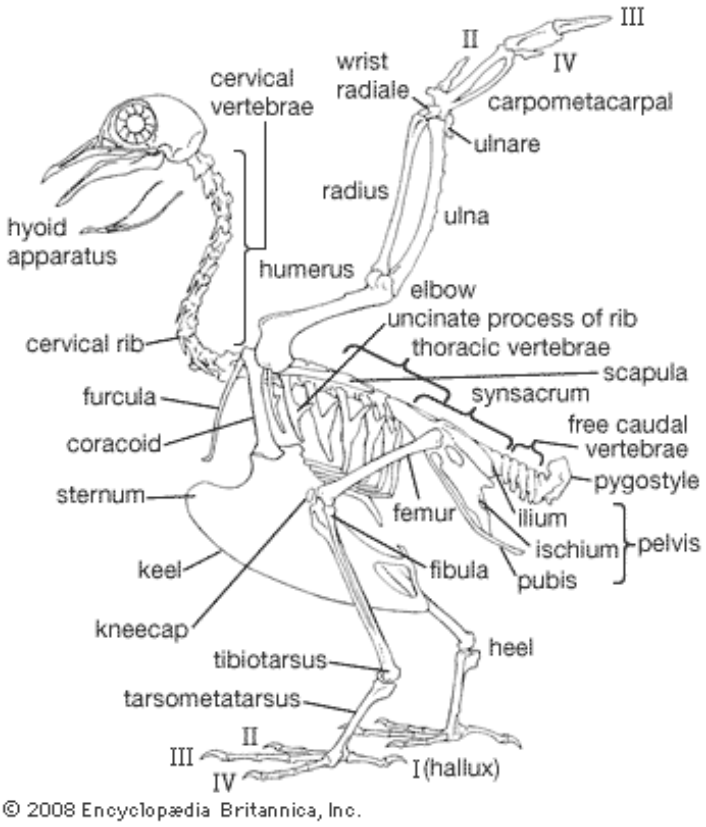
An outlier in a multidimensional data set has atypical readings on one or on several features *simultaneously*. These observations can often be found in univariate plots of the lead principal component scores, even when outliers are not extreme in any individual feature.

In the figure below, on the left, the illustrative point is within the range of the data in both the Feature1 and Feature2 directions. On the right, when the data has been rotated to align with the PC axes, the point stands out as extreme in the PC2 direction.



# 13.6 Example: Sparrows, for Class Discussion

After a severe storm in 1898, a number of sparrows were taken to the biological laboratory at the University of Rhode Island. H. Bumbus[2] measured several morphological characteristics on each bird. The data here correspond to five measurements on a sample of 49 females. The measurements are the total length, alar extent (wing-spread), beak-head length, humerus length (part of the wing), and length of keel of sternum.




http://media-2.web.britannica.com/eb-media/46/51946-004-D003BC49.gif

Let us look at the output, paying careful attention to the interpretations of the principal components (zeroing out small loadings). How many components seem sufficient to capture the total variation in the morphological measurements?

[2]Bumpus, Hermon C. 1898. Eleventh lecture. The elimination of the unfit as illustrated by the introduced sparrow, Passer domesticus. (A fourth contribution to the study of variation.) Biol. Lectures: Woods Hole Marine Biological Laboratory, 209–225.

```
#### Example: Sparrows
dat_sparrows <-
  read_table2(
    "http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch13_sparrows.dat"
  )
```

```
## Parsed with column specification:
## cols(
##   Total = col_double(),
##   Alar = col_double(),
##   BeakHead = col_double(),
##   Humerus = col_double(),
##   Keel = col_double()
## )
```

```
str(dat_sparrows)
```

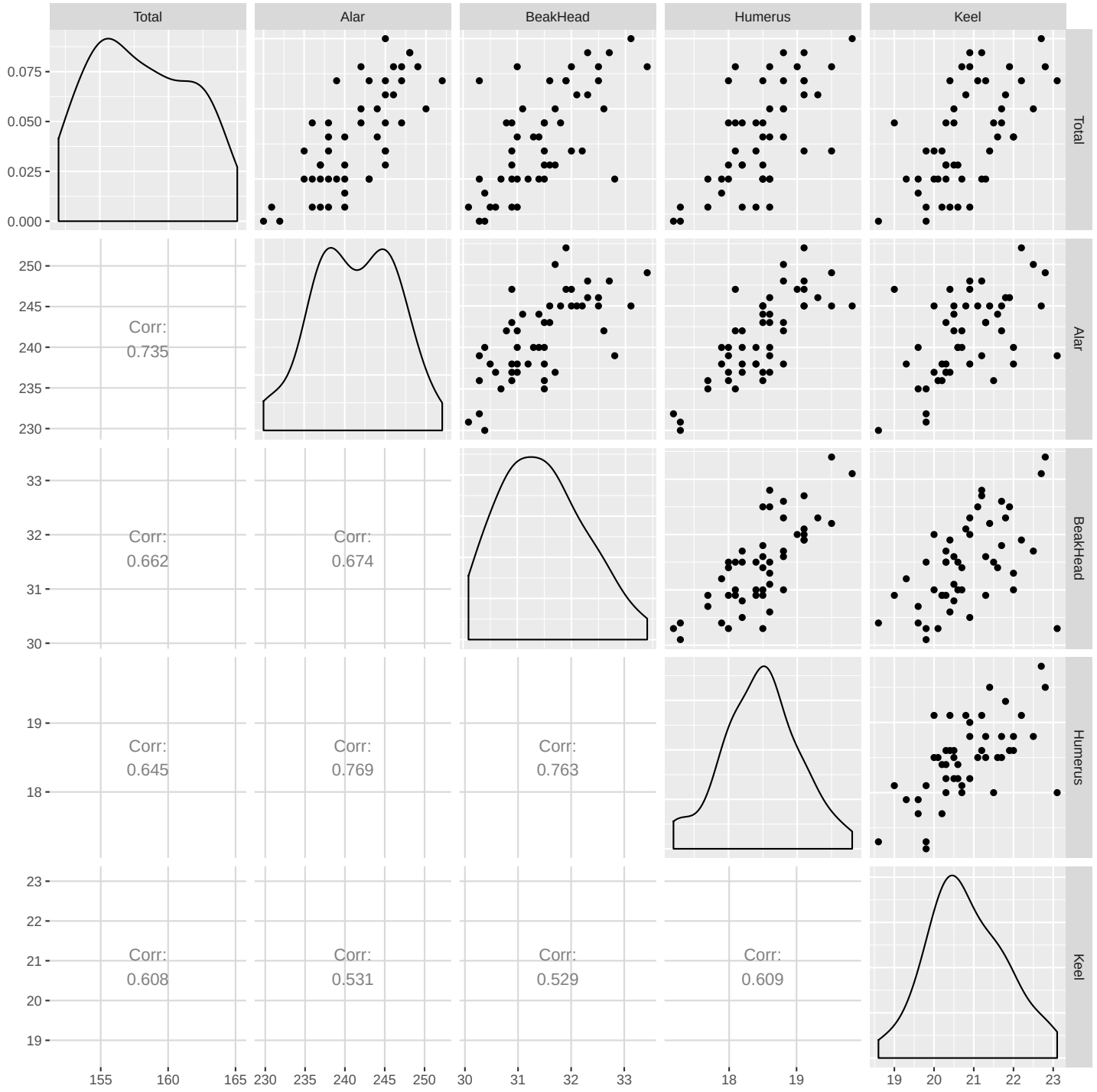
```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 49 obs. of  5 variables:
##  $ Total   : num  156 153 155 157 164 158 161 157 158 155 ...
##  $ Alar    : num  245 240 243 238 248 240 246 235 244 236 ...
##  $ BeakHead: num  31.6 31 31.5 30.9 32.7 31.3 32.3 31.5 31.4 30.3 ...
##  $ Humerus : num  18.5 18.4 18.6 18.4 19.1 18.6 19.3 18.1 18.5 18.5 ...
##  $ Keel    : num  20.5 20.6 20.3 20.2 21.2 22 21.8 19.8 21.6 20.1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Total = col_double(),
##   ..   Alar = col_double(),
##   ..   BeakHead = col_double(),
##   ..   Humerus = col_double(),
##   ..   Keel = col_double()
##   .. )
```

```
head(dat_sparrows)
```

```
## # A tibble: 6 x 5
##   Total  Alar BeakHead Humerus  Keel
##   <dbl> <dbl>    <dbl>   <dbl> <dbl>
## 1   156   245     31.6    18.5  20.5
## 2   153   240     31      18.4  20.6
## 3   155   243     31.5    18.6  20.3
## 4   157   238     30.9    18.4  20.2
## 5   164   248     32.7    19.1  21.2
## 6   158   240     31.3    18.6  22
```

```
## Scatterplot matrix
library(ggplot2)
library(GGally)
# put scatterplots on top so y axis is vertical
p <-
  ggpairs(
    dat_sparrows
  , upper = list(continuous = "points")
  , lower = list(continuous = "cor")
```

```
  , progress = FALSE
  )
print(p)
```




```
# perform PCA on covariance matrix
pca_sparrows <-
  princomp(
    ~ Total + Alar + BeakHead + Humerus + Keel
  , data = dat_sparrows
  )
```

```
# standard deviation and proportion of variation for each component
pca_sparrows %>% summary()
```

```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3      Comp.4
## Standard deviation     5.8828991 2.1280701 0.78468836 0.552957190
## Proportion of Variance 0.8623099 0.1128372 0.01534175 0.007618397
## Cumulative Proportion  0.8623099 0.9751472 0.99048891 0.998107311
##                            Comp.5
## Standard deviation     0.275612798
## Proportion of Variance 0.001892689
## Cumulative Proportion  1.000000000
```

```
# coefficients for PCs
# loadings(pca_sparrows)  # print method for loadings() uses cutoff = 0.1 by default
pca_sparrows %>% loadings() %>% print(cutoff = 0) # to show all values
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## Total     0.536  0.828  0.157  0.039  0.018
## Alar      0.829 -0.551  0.058  0.069 -0.040
## BeakHead  0.096  0.034 -0.241 -0.897 -0.357
## Humerus   0.074 -0.015 -0.205 -0.306  0.927
## Keel      0.101  0.100 -0.934  0.310 -0.110
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

In the PCA on the covariance matrix above,

1. How many components do you need to capture at least 95% of the total variability?
2. How would you interpret these few components?[3]

How does the PCA on the correlation matrix below compare?

```
# perform PCA on correlation matrix
pca_sparrows <-
  princomp(
    ~ Total + Alar + BeakHead + Humerus + Keel
  , data = dat_sparrows
```

[3] 1. The cumulative proportion of variance explained by the 5 components are 0.862, 0.975, 0.99, 0.998, 1, thus, we need only 2 components.
2a. The first PC has all positive values with large values for Total, Alar, and Keel; PC1 is the size of the bird.
2b. The second PC has a positive Total and negative Alar; PC2 is the contrast between the length of the bird (Total) and wing-spread (Alar).

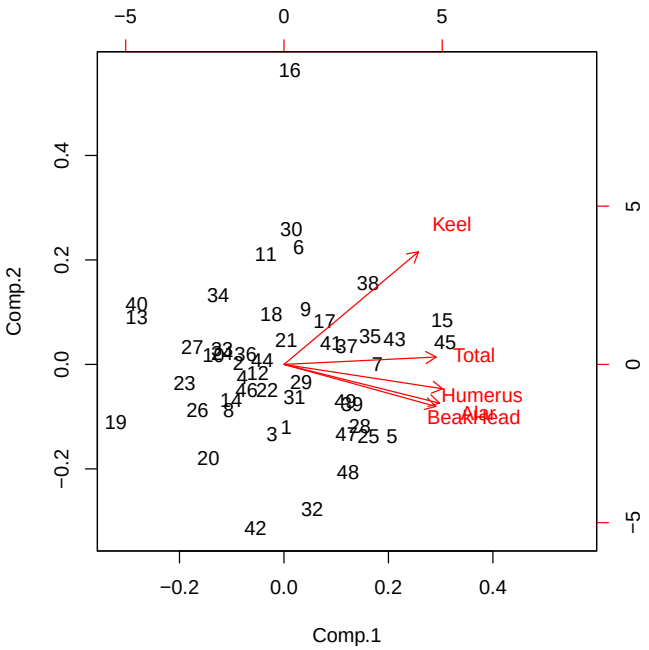
```
  , cor  = TRUE
  )
# standard deviation and proportion of variation for each component
pca_sparrows %>% summary()
```

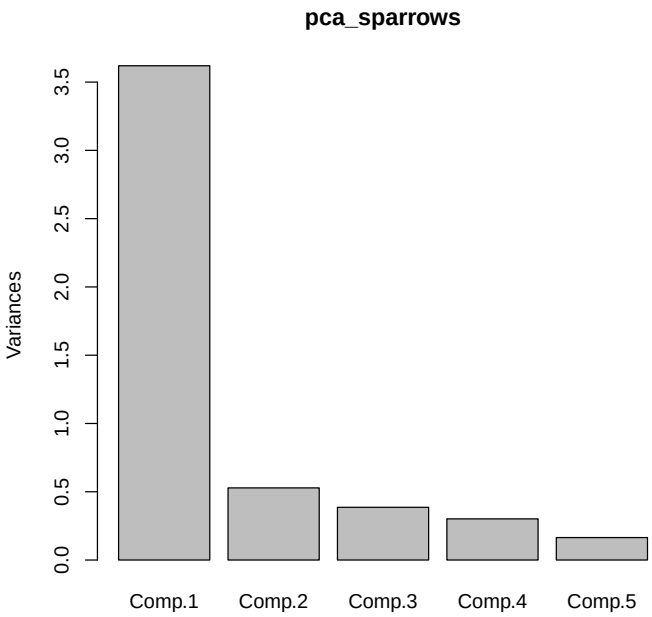
```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3     Comp.4
## Standard deviation     1.9025587 0.7267974 0.62139498 0.54902221
## Proportion of Variance 0.7239459 0.1056469 0.07722634 0.06028508
## Cumulative Proportion  0.7239459 0.8295928 0.90681917 0.96710425
##                            Comp.5
## Standard deviation     0.40555980
## Proportion of Variance 0.03289575
## Cumulative Proportion  1.00000000
```

```
# coefficients for PCs
#loadings(pca_sparrows)
pca_sparrows %>% loadings() %>% print(cutoff = 0) # to show all values
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## Total     0.452  0.058  0.689  0.422  0.375
## Alar      0.461 -0.301  0.345 -0.545 -0.530
## BeakHead  0.450 -0.326 -0.453  0.607 -0.342
## Humerus   0.470 -0.189 -0.409 -0.390  0.651
## Keel      0.399  0.874 -0.184 -0.073 -0.194
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(pca_sparrows)
screeplot(pca_sparrows)
```


Keel
Total
Humerus
Beak.Head
Alar
Comp.1
Comp.2


pca_sparrows
Variances
Comp.1
Comp.2
Comp.3
Comp.4
Comp.5

## 13.7 PCA for Variable Reduction in Regression

I will outline an analysis where PCA was used to create predictors in a regression model. The data were selected from the Berkeley Guidance Study, a longitudinal monitoring of children born in Berkeley, California, between 1928 and 1929. The variables selected from the study are

ID an identification number,
WT2 weight at age 2 in kg,
HT2 height at age 2 in cm,
WT9 weight at age 9,
HT9 height at age 9,
LG9 leg circumference at age 9 in cm,
ST9 a composite measure of strength at age 9 (higher is stronger),
WT18 weight at age 18,
HT18 height at age 18,
LG18 leg circumference at age 18,
ST18 a composite measure of strength at age 18, and
SOMA somatotype on a 7-point scale, as a measure of fatness (1=slender to 7=fat) determined using a photo taken at age 18.

Data on 26 boys are given below.

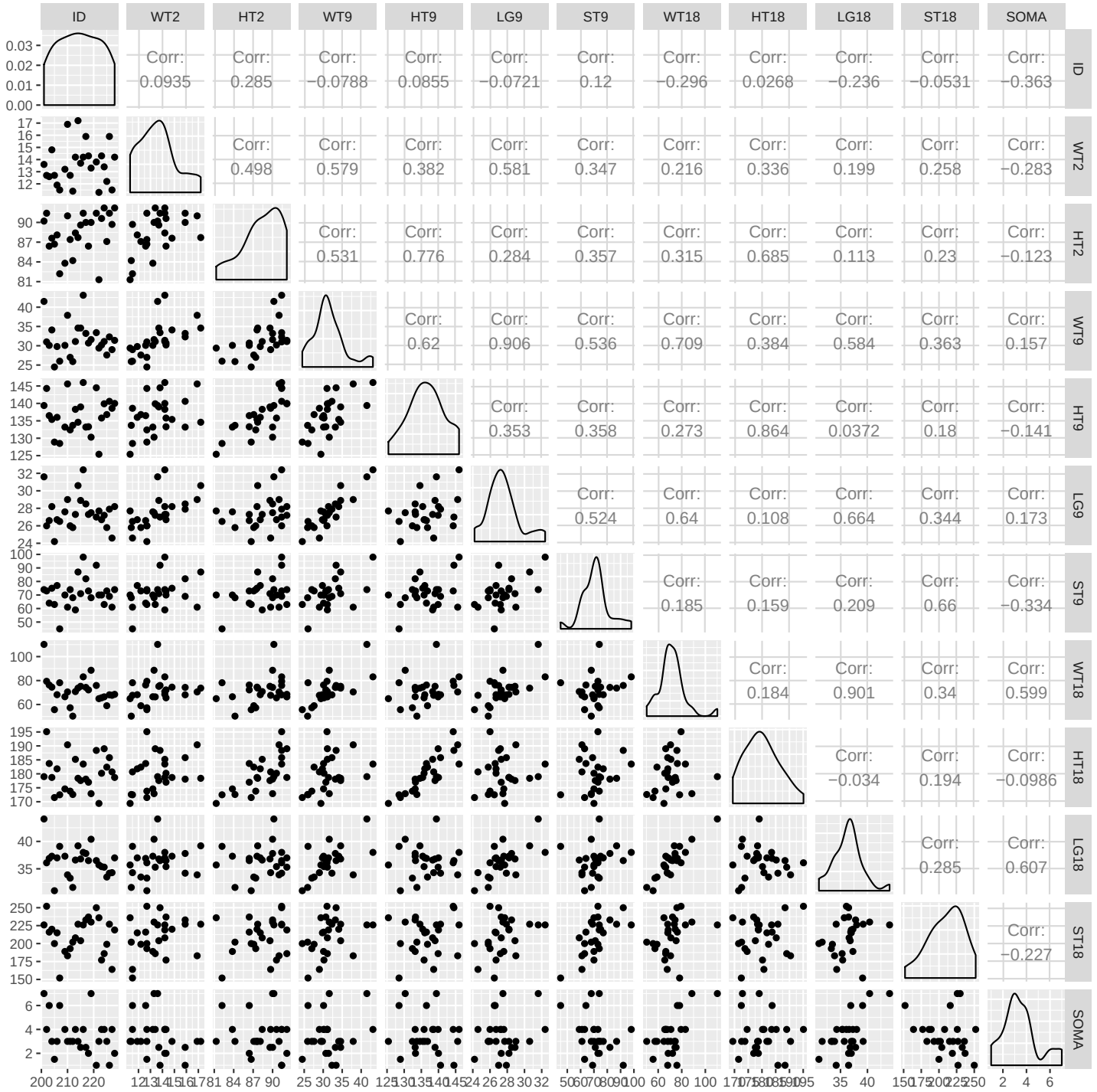
We are interested in building a regression model to predict somatotype (SOMA) from the other variables.

```
#### Example: BGS (Berkeley Guidance Study)
dat_bgs <-
  read_table2(
    "http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch13_bgs.dat"
  )

## Parsed with column specification:
## cols(
##   ID = col_double(),
##   WT2 = col_double(),
##   HT2 = col_double(),
##   WT9 = col_double(),
##   HT9 = col_double(),
```

```
##  LG9 = col_double(),
##  ST9 = col_double(),
##  WT18 = col_double(),
##  HT18 = col_double(),
##  LG18 = col_double(),
##  ST18 = col_double(),
##  SOMA = col_double()
## )
str(dat_bgs)
```

```
head(dat_bgs)
```

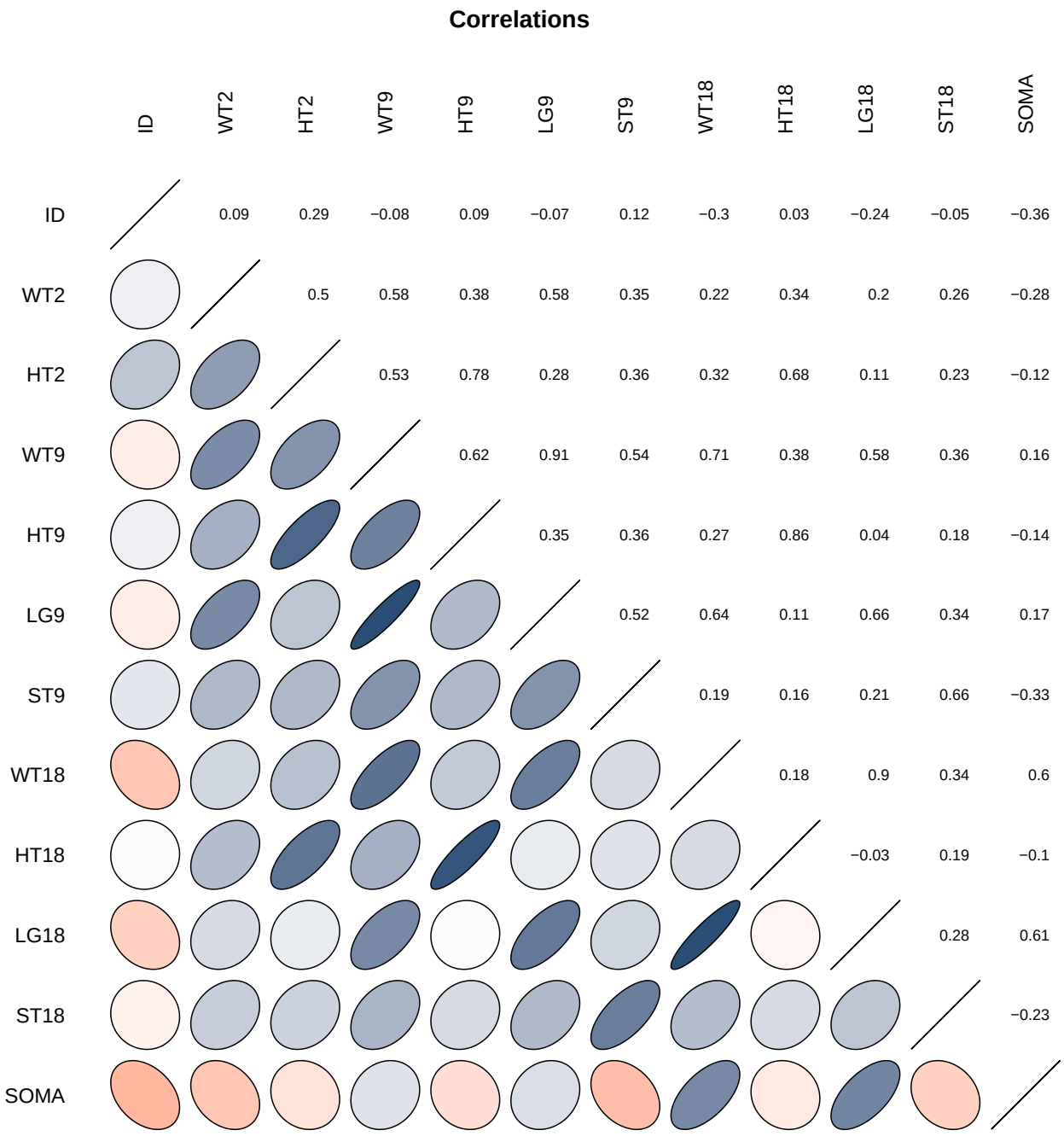
```
## Scatterplot matrix
library(ggplot2)
library(GGally)
p <-
  ggpairs(
    dat_bgs
  , progress = FALSE
  )
print(p)
```



As an aside, there are other ways to visualize the linear relationships more quickly. The `ellipse` library has a function `plotcorr()`, though it's output is less than ideal. An improvement has been made with an updated version[4] of the `plotcorr()` function.

```
## f_plot_corr_ellipse example, see ada_functions.R file for f_plot_corr_ellipse() function

# calculate correlations to plot
corr_bgs <- cor(dat_bgs)

# Change the column and row names for clarity
colnames(corr_bgs) = colnames(dat_bgs)
rownames(corr_bgs) = colnames(corr_bgs)

# set colors to use (blue positive, red negative)
col_sc   = c(rgb(241, 54, 23, maxColorValue=255), "white", rgb(0, 61, 104, maxColorValue = 255
col_ramp = colorRampPalette(col_sc, space = "Lab")
colors   = col_ramp(100)

# plot correlations, colored ellipses on lower diagonal, numerical correlations on upper
f_plot_corr_ellipse(
    corr_bgs
  , col   = colors[((corr_bgs + 1) / 2) * 100]
  , diag  = "ellipse"
  , upper.panel = "number"
  , mar   = c(0, 2, 0, 0)
  , main  = "Correlations"
  )
```

[4] http://hlplab.wordpress.com/2012/03/20/correlation-plot-matrices-using-the-ellipse-library




It is reasonable to expect that the characteristics measured over time, for example HT2, HT9, and HT18 are strongly correlated. Evidence supporting this hypothesis is given in the following output, which summarizes correlations within subsets of the predictors. Two of the three subsets include measures over time on the same characteristic.

```
dat_bgs[,c("WT2", "WT9", "WT18")]         %>% cor()
##            WT2       WT9      WT18
```

```
## WT2  1.0000000 0.5792217 0.2158735
## WT9  0.5792217 1.0000000 0.7089029
## WT18 0.2158735 0.7089029 1.0000000
```

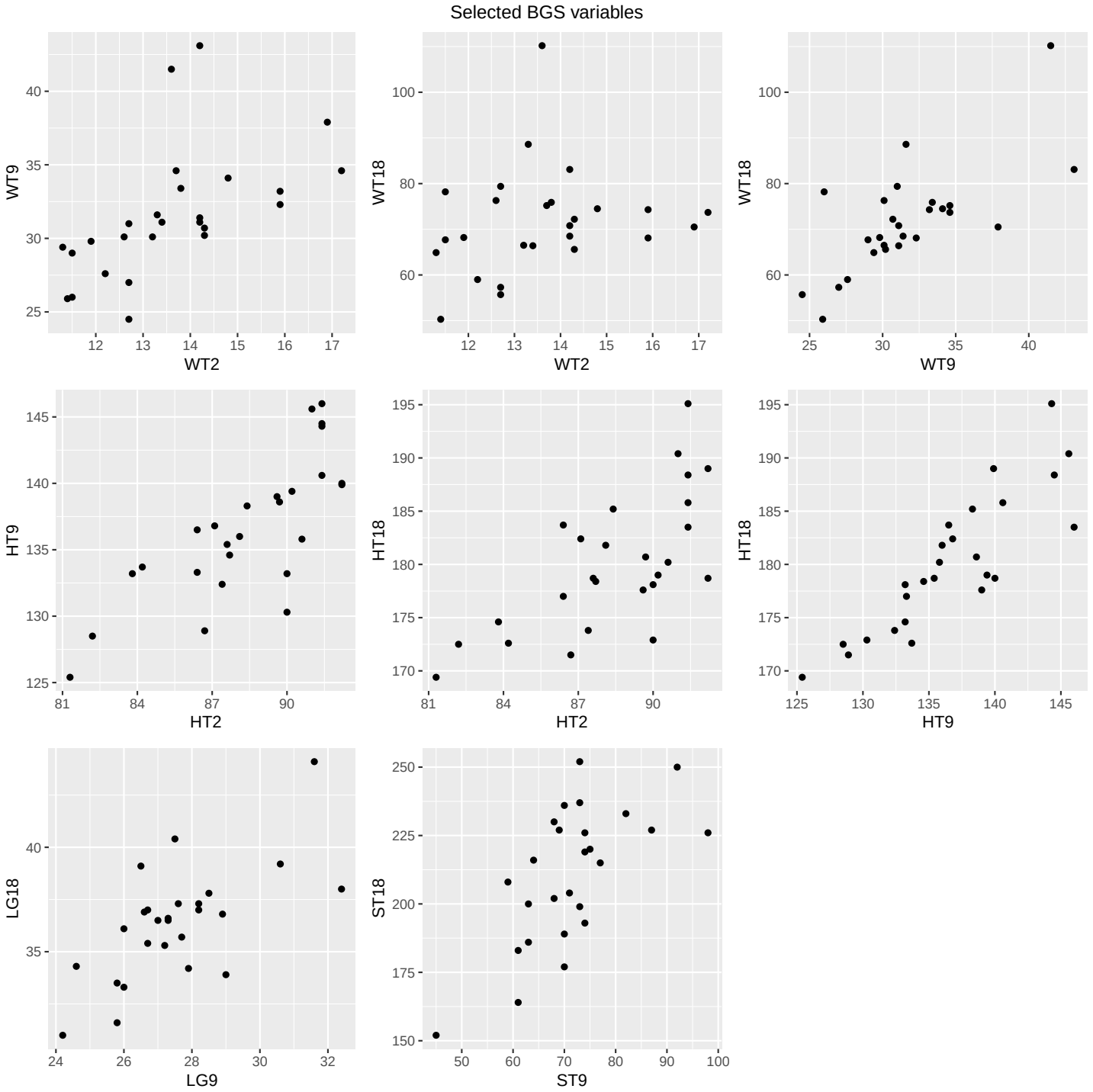
```
dat_bgs[,c("HT2", "HT9", "HT18")]         %>% cor()
```

```
##            HT2       HT9      HT18
## HT2  1.0000000 0.7758332 0.6847144
## HT9  0.7758332 1.0000000 0.8644624
## HT18 0.6847144 0.8644624 1.0000000
```

```
dat_bgs[,c("LG9", "LG18", "ST9", "ST18")] %>% cor()
```

```
##            LG9      LG18       ST9      ST18
## LG9  1.0000000 0.6644753 0.5239476 0.3440756
## LG18 0.6644753 1.0000000 0.2085289 0.2845414
## ST9  0.5239476 0.2085289 1.0000000 0.6595947
## ST18 0.3440756 0.2845414 0.6595947 1.0000000
```

```
library(ggplot2)
p1 <- ggplot(dat_bgs, aes(x = WT2 , y = WT9 )) + geom_point()
p2 <- ggplot(dat_bgs, aes(x = WT2 , y = WT18)) + geom_point()
p3 <- ggplot(dat_bgs, aes(x = WT9 , y = WT18)) + geom_point()
p4 <- ggplot(dat_bgs, aes(x = HT2 , y = HT9 )) + geom_point()
p5 <- ggplot(dat_bgs, aes(x = HT2 , y = HT18)) + geom_point()
p6 <- ggplot(dat_bgs, aes(x = HT9 , y = HT18)) + geom_point()
p7 <- ggplot(dat_bgs, aes(x = LG9 , y = LG18)) + geom_point()
p8 <- ggplot(dat_bgs, aes(x = ST9 , y = ST18)) + geom_point()

library(gridExtra)
grid.arrange(
    grobs = list(p1, p2, p3, p4, p5, p6, p7, p8)
  , ncol  = 3
  , top   = "Selected BGS variables"
  )
```




Strong correlation among predictors can cause collinearity problems in regression. The presence of collinearity makes the interpretation of regression effects more difficult, and can reek havoc with the numerical stability of certain algorithms used to compute least squares summaries. A natural way to avoid collinearity and improve interpretation of regression effects is to use uncorrelated linear combinations of the original predictors, such as principal components, to

build a model.

The interpretation of regression effects changes when linear combinations of predictors are used in place of the original predictors. However, two models will be equivalent in the sense of giving identical fitted values when $p$ linearly independent combinations of $p$ predictors are used. The fitted model can be expressed in terms of the original predictors, or in terms of the linear combinations. Similarly, standardizing the original predictors does not change the significance of individual regression effects nor does it change the interpretation of the model.

A reasonable strategy with the Berkeley data might be to find principal components for the three height variables separately, the three weights separately, and so on. It may not make sense to combine the four different types of measures (HT, WT, ST, and LG) together because the resulting linear combinations would likely be uninterpretable.

Output from a PCA on the four sets of standardized measures is given below. Two sets (ST9, ST18) and (LG9, LG18) have two predictors. The PCs on the strength and leg measures are essentially the sum and difference between the two standardized features. The loadings have magnitude $0.707 = (1/2)^{1/2}$ to satisfy the unit-length restriction.

Here are some comments on how I might use the PCA to build a regression model to predict somatotype. First, I would not necessarily use the given PCS, but would instead use interpretable linear combinations that were nearly identical to the PCs. For example, the first principal component of the heights is roughly an unweighted average of the weights at ages 2, 9, and 18. I would use (WT2+WT9+WT18)/3 instead. The overall sum of squared loadings is not important in the regression analysis. Only the relative sizes are important. Following this idea, what linear combinations of the heights are reasonable?

Second, you should not assume that principal components with low variance are unimportant for predicting somatotype. Recall our earlier discussion on potential problems with ignoring low variability components when group comparisons are the primary interest. The same problem is possible here.

Feel free to explore these ideas at your leisure.

```
# WT
pca_bgs_WT <-
  princomp(
    ~ WT2 + WT9 + WT18
  , data = dat_bgs
  , cor = TRUE
  )
pca_bgs_WT %>% summary()
```

```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3
## Standard deviation     1.4241276 0.8882456 0.42764499
## Proportion of Variance 0.6760465 0.2629934 0.06096008
## Cumulative Proportion  0.6760465 0.9390399 1.00000000
```

```
pca_bgs_WT %>% loadings() %>% print(cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2 Comp.3
## WT2   0.492  0.781  0.384
## WT9   0.665 -0.053 -0.745
## WT18  0.562 -0.622  0.545
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

```
# HT
pca_bgs_HT <-
  princomp(
    ~ HT2 + HT9 + HT18
  , data = dat_bgs
  , cor = TRUE
  )
pca_bgs_HT %>% summary()
```

```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3
## Standard deviation     1.5975991 0.5723653 0.34651866
## Proportion of Variance 0.8507743 0.1092007 0.04002506
## Cumulative Proportion  0.8507743 0.9599749 1.00000000
```

```
pca_bgs_HT %>% loadings() %>% print(cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2 Comp.3
## HT2   0.554  0.800  0.231
## HT9   0.599 -0.190 -0.778
## HT18  0.578 -0.570  0.584
```

```
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

```
# LG
pca_bgs_LG <-
  princomp(
    ~ LG9 + LG18
  , data = dat_bgs
  , cor = TRUE
  )
pca_bgs_LG %>% summary()
```

```
## Importance of components:
##                            Comp.1    Comp.2
## Standard deviation     1.2901455 0.5792449
## Proportion of Variance 0.8322377 0.1677623
## Cumulative Proportion  0.8322377 1.0000000
```

```
pca_bgs_LG %>% loadings() %>% print(cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2
## LG9   0.707  0.707
## LG18  0.707 -0.707
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

```
# ST
pca_bgs_ST <-
  princomp(
    ~ ST9 + ST18
  , data = dat_bgs
  , cor = TRUE
  )
pca_bgs_ST %>% summary()
```

```
## Importance of components:
##                            Comp.1    Comp.2
## Standard deviation     1.2882526 0.5834426
## Proportion of Variance 0.8297974 0.1702026
## Cumulative Proportion  0.8297974 1.0000000
```

```
pca_bgs_ST %>% loadings() %>% print(cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2
```

```
## ST9   0.707  0.707
## ST18  0.707 -0.707
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```