Covariance Matrix

Alexander McLain

Calculating the Covariance and Correlation matrix

First, let's load in the led data.

```
wide_lead <- read.csv("wide_lead.csv",header = TRUE, na.strings = "", stringsAsFactors = FALSE)
head(wide_lead)
```

```
##   ID TRT  PB1  PB2  PB3  PB4
## 1  1   P 30.8 26.9 25.8 23.8
## 2  2   A 26.5 14.8 19.5 21.0
## 3  3   A 25.8 23.0 19.1 23.2
## 4  4   P 24.7 24.5 22.0 22.5
## 5  5   A 20.4  2.8  3.2  9.4
## 6  6   A 20.4  5.4  4.5 11.9
```

To calculate the covariance and correlation matrix we'll want to use the **wide** version of the data.

Calculating the Covariance Matrix

We'll take off the 1st and 2nd column

```
round(cov(wide_lead[,-c(1,2)]),4)
```

```
##         PB1     PB2     PB3     PB4
## PB1 24.9891 18.1607 18.9215 21.7822
## PB2 18.1607 75.2249 59.2410 37.4869
## PB3 18.9215 59.2410 65.3854 36.5424
## PB4 21.7822 37.4869 36.5424 60.1590
```

Calculating the Correlation Matrix

```
round(cor(wide_lead[,c("PB1","PB2","PB3","PB4")]),4)
```

```
##        PB1    PB2    PB3    PB4
## PB1 1.0000 0.4189 0.4681 0.5618
## PB2 0.4189 1.0000 0.8447 0.5572
## PB3 0.4681 0.8447 1.0000 0.5826
## PB4 0.5618 0.5572 0.5826 1.0000
```

Same with the Six Cities example

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
## v ggplot2 3.3.3     v purrr   0.3.4
```

```
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
long_Six_cities <- read.csv("Six_cities.csv", header = TRUE)
head(long_Six_cities,6)
```

```
##   ID Height     Age INI_Height INI_Age Log_FEV1
## 1  1   1.20  9.3415        1.2  9.3415  0.21511
## 2  1   1.28 10.3929        1.2  9.3415  0.37156
## 3  1   1.33 11.4524        1.2  9.3415  0.48858
## 4  1   1.42 12.4600        1.2  9.3415  0.75142
## 5  1   1.48 13.4182        1.2  9.3415  0.83291
## 6  1   1.50 15.4743        1.2  9.3415  0.89200
```

```
long_Six_cities <- long_Six_cities %>% mutate(Age_R = floor(Age))
head(long_Six_cities,6)
```

```
##   ID Height     Age INI_Height INI_Age Log_FEV1 Age_R
## 1  1   1.20  9.3415        1.2  9.3415  0.21511     9
## 2  1   1.28 10.3929        1.2  9.3415  0.37156    10
## 3  1   1.33 11.4524        1.2  9.3415  0.48858    11
## 4  1   1.42 12.4600        1.2  9.3415  0.75142    12
## 5  1   1.48 13.4182        1.2  9.3415  0.83291    13
## 6  1   1.50 15.4743        1.2  9.3415  0.89200    15
```

```
wide_Six_cities <- long_Six_cities %>% pivot_wider(id_cols = c(ID, INI_Height, INI_Age),
                              names_from = Age_R, values_from = c(Log_FEV1,Height),
                              values_fn = min)
head(wide_Six_cities)
```

```
## # A tibble: 6 x 29
##      ID INI_Height INI_Age Log_FEV1_9 Log_FEV1_10 Log_FEV1_11 Log_FEV1_12
##   <int>      <dbl>   <dbl>      <dbl>       <dbl>       <dbl>       <dbl>
## 1     1        1.2    9.34      0.215       0.372       0.489       0.751
## 2     2       1.13    6.59     NA          NA          NA           0.756
## 3     3       1.18    6.91      0.751      NA           0.967      NA
## 4     4       1.15    6.76      0.445       0.577       0.673       0.723
## 5     5       1.11    6.50     NA          NA          NA          NA
## 6     6       1.24    6.90      0.713      NA           0.775       0.900
## # ... with 22 more variables: Log_FEV1_13 <dbl>, Log_FEV1_15 <dbl>,
## #   Log_FEV1_16 <dbl>, Log_FEV1_6 <dbl>, Log_FEV1_7 <dbl>, Log_FEV1_14 <dbl>,
## #   Log_FEV1_17 <dbl>, Log_FEV1_8 <dbl>, Log_FEV1_18 <dbl>, Height_9 <dbl>,
## #   Height_10 <dbl>, Height_11 <dbl>, Height_12 <dbl>, Height_13 <dbl>,
## #   Height_15 <dbl>, Height_16 <dbl>, Height_6 <dbl>, Height_7 <dbl>,
## #   Height_14 <dbl>, Height_17 <dbl>, Height_8 <dbl>, Height_18 <dbl>
```

Let's just look at a small subset of the data:

```
wide_Six_cities_sub <- wide_Six_cities[,c("Log_FEV1_9", "Log_FEV1_10",
                                          "Log_FEV1_11", "Log_FEV1_12")]
head(wide_Six_cities_sub)
```

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

```
##   Log_FEV1_9 Log_FEV1_10 Log_FEV1_11 Log_FEV1_12
##        <dbl>       <dbl>       <dbl>       <dbl>
## 1      0.215       0.372       0.489       0.751
## 2     NA          NA          NA           0.756
## 3      0.751      NA           0.967      NA
## 4      0.445       0.577       0.673       0.723
## 5     NA          NA          NA          NA
## 6      0.713      NA           0.775       0.900
```

The problem with estimating the covariance or correlation matrix from this data are the NA values. We have to remove the NA values to use the **cov** or **cor** functions. So we'll remove them below.

```
wide_Six_cities_sub <- wide_Six_cities_sub %>% na.omit
head(wide_Six_cities_sub)
```

```
## # A tibble: 6 x 4
##   Log_FEV1_9 Log_FEV1_10 Log_FEV1_11 Log_FEV1_12
##        <dbl>       <dbl>       <dbl>       <dbl>
## 1      0.215       0.372       0.489       0.751
## 2      0.445       0.577       0.673       0.723
## 3      0.560       0.802       0.971       1.09
## 4      0.507       0.571       0.693       0.875
## 5      0.577       0.688       0.756       0.802
## 6      0.560       0.732       0.908       0.850
```

Now we can estimate the covariance and correlation matrices

```
round(cov(wide_Six_cities_sub),3)
```

```
##             Log_FEV1_9 Log_FEV1_10 Log_FEV1_11 Log_FEV1_12
## Log_FEV1_9       0.018       0.017       0.016       0.015
## Log_FEV1_10      0.017       0.021       0.020       0.019
## Log_FEV1_11      0.016       0.020       0.027       0.022
## Log_FEV1_12      0.015       0.019       0.022       0.025
```

```
round(cor(wide_Six_cities_sub),2)
```

```
##             Log_FEV1_9 Log_FEV1_10 Log_FEV1_11 Log_FEV1_12
## Log_FEV1_9        1.00        0.89        0.73        0.71
## Log_FEV1_10       0.89        1.00        0.83        0.82
## Log_FEV1_11       0.73        0.83        1.00        0.84
## Log_FEV1_12       0.71        0.82        0.84        1.00
```

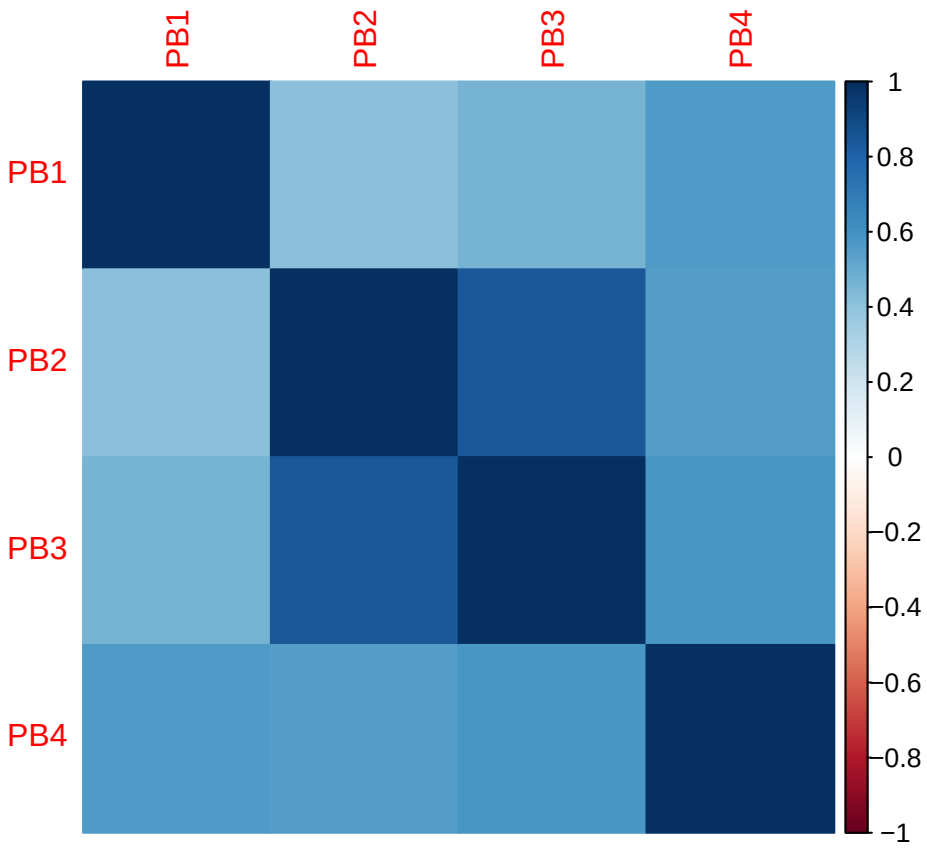
A plot of the correlation

We can also plot the correlation with the help of the **ggplot2** package.

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
M <- cor(wide_lead[,-c(1,2)])
corrplot(M, method="color")
```


PB1
PB2
PB3
PB4
PB1
PB2
PB3
PB4
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1