STAT3064/STAT4067 Statistical Learning/Statistical Data Science

Computer Lab 1 for Weeks 1 and 2, 2021

Things you may need to know/do.

- Relates to R, visualisation and Lecture 1.
- Q1—Q3 should be attempted in Week 1; the remaining questions (including Q4 and Q5) to be answered in Week 2, so after Lecture 1.
- Libraries ggplot2, tidyverse, MASS, GGally may be useful. Others may be mentioned below in the hints. Install the libraries MASS, ggplot2, tidyverse and GGally before you start with the questions.
- You might like to set up a project for each lab if you are using RStudio. Then you can copy a .Rmd file into that directory and write your answers in that file.

Q1

Consider the aircraft data (aircraft.csv) with variables Year, Period, Power, Span, Length, Weight, Speed and Range. The data are available in the Data Sets folder on LMS.

Data description (from the R help for package sm)

These data record six characteristics of aircraft designs which appeared during the twentieth century. The variables are: - Year: year of first manufacture - Period: a code to indicate one of three broad time periods - Power: total engine power (kW) - Span: wing span (m) - Length: length (m) - Weight: maximum take-off weight (kg) - Speed: maximum speed (km/h) - Range: range (km)

Source: The data were collected by P. Saviotti and are described in detail in Saviotti (1996), "Technological Evolution, Variety and Economy", Edward Elgar: Cheltenham.

a) Read the data into R, and summarise it to understand the inputs. Are there any missing values? (Hint: read.csv, head and summary.)

b) Show separate histogram of Power and log10(Power). Describe the shape of the histograms. (Hint: truehist (from MASS) or ggplot with geom_histogram.)

c) Construct smoothed histograms of Power and log10(Power). (Hint: ggplot + geom_density.)

d) Create new variables logPower equal to log10(Power) and similarly for the remaining variables, apart from Period and Year. Do this by creating a new dataframe with these variables plus Period and Year. Make Period a factor on this new dataframe. Why might the log transformation be a good idea to help in understanding these data? (Note that we have specified log10 rather than the natural log, because this is easier to interpret in the circumstances we have.) (Hint: Can be done directly but could use mutate from dplyr, which is part of tidyverse.)

e) Show pairwise plots of the logged variables. Which pair has the largest absolute correlation? (Hint: pairs or ggpairs.)

f) The code chunk below generates a 3D scatterplot of the variables logPower, logSpan and logLength. Comment on the scatterplots in relation to what it says about the variables, and discuss the shortcomings of this sort of display. Experiment with changing the viewing angles phi and theta.

g) The code chunk below makes a parallel coordinate plot of the 6 logged variables, coloured by the value of Period, and with boxplots and suitable transparency. It is arranged so that the values are not scaled. Comment on what this figure says about the data, and discuss any shortcomings of the plotting method.

Code chunk for 3d scatterplots.

Remove the ", = eval = FALSE" before running knitting in RStudio.

```
library( plot3D )
scatter3D (aircraft$logWeight, aircraft$logSpan, aircraft$logLength,
           phi = 20, theta = 80,
           col = NULL, NAcol = "white", breaks = NULL,
           colkey = NULL, panel.first = NULL,
           clim = NULL, clab = NULL,
           bty = "b", CI = NULL, surf = NULL,
           add = FALSE, plot = TRUE)
```

Code chunk for parallel co-ordinate plot.

Remove the ", eval = FALSE" before running knitting in RStudio.

```
library(GGally)
ggparcoord( aircraft,
            groupColumn = "Period",
            columns =  3:8,
            alpha = 0.1,
            scale = "globalminmax",
            boxplot = TRUE ) +
  theme( legend.position = "bottom")
```

Q2

Continue with the aircraft data of Q1 (using the logged variables).
The data are from three different periods.

(Hints for these: summary, xtabs, sapply, filter.)

a) How many observations belong to each period?

b) Which years belong to each period?

c) Which variable has the largest interquartile range? Which has the largest median? For which variables are the means larger than 1.04 times the median? (Leave out Period for this question, as it is a factor, not a numeric variable. Don't do the second part of this question by just looking at the output from say the summary command. Do it by writing a small amount of code that will tell you the answer directly.)

d) Divide the dataframe into the three period groups. For all the data and separately for period 1, give the histograms and density plots of logPower. Compare the results for the complete data with those obtained from period 1. Comment. (Hint, for interest and because it will be useful in the first assignment: It is fairly easy to get 3 separate density plots. The following on the other hand will generate a plot more useful for comparisons.)

```
# density plots
  ggplot( aircraft, aes( logPower, group = Period, colour = Period ) ) +
  geom_density( aes( fill = Period ), alpha = 0.4 ) +
  ggtitle("All data"))
```

e) Compare the pairs or ggpairs results for the logged variables for the whole data and for period 1. Comment.

Q3

Continue with the aircraft data of Q2.

a) Construct contour plots of the 2D smoothed histograms for the pair of variables logLength and logWeight, and also for logSpeed and logSpan for all records.
(Hint: This code chunk does the second one.)

```
ggplot( aircraft, aes( logSpeed, logSpan ) )  +
  geom_density_2d( ) +
  geom_point( aes( colour = Period ), alpha = 0.6 ) +
  theme( legend.position = "bottom")
```

b) Describe the shape of these density plots (eg bimodal, skewed, etc).

c) What do the shapes of the contours in these figures say about the variable pairs? Hint: look at the correlations you calculated earlier.

Q4

In this question we work with the matrix Σ_0 and want to calculate powers of Σ_0 as in Lecture 1. Remember to remove the "`, eval = FALSE`" before using the code.

```
Sigma0 = matrix(c(
 3.1386518, 0.38872659, 0.6178228, 1.7576847, 0.77433973, 0.7508789,
 0.3887266, 1.21417949, 0.1941453, 0.4518920, 0.01236855, 0.2155466,
 0.6178228, 0.19414529, 1.2437919, 0.5970320, 0.15151088, 0.2665018,
 1.7576847, 0.45189196, 0.5970320, 1.7083497, 0.52685668, 0.7109476,
 0.7743397, 0.01236855, 0.1515109, 0.5268567, 0.53406192, 0.2299193,
 0.7508789, 0.21554658, 0.2665018, 0.7109476, 0.22991933, 0.6642375),
 byrow = TRUE, nrow = 6 )
```

a) Use `eigen` to find the eigenvalues and eigenvectors of the covariance matrix Σ_0 and list them.

b) Show a plot of the eigenvalues against the index.

c) Calculate the square of the eigenvalues, and show a plot of these.

d) Calculate Σ_0^m , for $m = 2$.

Hint: use the fact that Σ_0 and Σ_0^m can be written as products. You may find the following code chunk useful.

```
S0 = Sigma0
spectral1 = eigen(S0)
V1 = spectral1$vectors
lambda1 = spectral1$values
diaglambda1 = diag( lambda1 )
S1 = V1 %*% diaglambda1 %*% t( V1 )
```

```
# now try m=2
diaglambda2 = diag( lambda1 )^2
S2 = V1 %*% diaglambda2 %*% t( V1 )
```

e) Repeat parts c) and d) for m=-1 and m=-2/3. Show plots of the powers of these eigenvalues and display the matrices Σ_0^{-1} and $\Sigma_0^{-2/3}$.

Q5

The purpose of this question is to offer you a template for writing simulations. This is not the only approach, but perhaps a helpful suggestion.

After the example and code for you to play with, there are tasks for you to practise at the end.

Our goal is to simulate multiple times the process of drawing a sample from a population, computing some items of interest and then examining the distribution of these items.

A simple outline is therefore to do the following: choose the number of replications `Nreps`, and the sample size `n`.

Repeat the following steps `Nreps` times (to the end of the loop):

1. Generate a sample of size `n`.
2. From this sample compute the items of interest and store them somewhere.
3. End of the loop.

Then analyse the stored results.

Some details need to be sorted out, which is easiest with an example.

Suppose we are interested in how sample estimates of the eigenvalues of a covariance matrix can vary. Suppose we are also interested in the so-called **condition number** of the sample covariance matrix, since this is a measure of how difficult it is numerically to invert the matrix. (If you like, it is a measure of how close the matrix is to being singular.) The condition number is defined to be the largest eigenvalue divided by the smallest.

We pick a particular population matrix, called `Sigma0` in the code below, the same matrix as in Q3.

Since we start with a particular population covariance matrix of full rank, the eigenvalues of that will be positive. Since we are going to choose a sample size greater than the dimension, the sample covariance matrix will also have full rank in each sample we draw (apart from possible rounding errors). But in some samples, the smallest eigenvalue may be close to 0, and the condition number may be large. We want to see how often this happens.

Since the question is about sample covariance matrices, we can fix the population mean at zero. (Samples will usually have different sample means, of course.)

The following code computes the eigenvalues and condition numbers of the sample covariance matrices generated from 200 samples. From these 200 results, summaries and plots can be produced in order to understand the distributions of the values across samples. Examples of such analyses are included in the code-chunk below.

Simulation code chunk

Remember to remove the "`, eval = FALSE`" before using the code.

```r
Sigma0 = matrix(c(
 3.1386518, 0.38872659, 0.6178228, 1.7576847, 0.77433973, 0.7508789,
 0.3887266, 1.21417949, 0.1941453, 0.4518920, 0.01236855, 0.2155466,
 0.6178228, 0.19414529, 1.2437919, 0.5970320, 0.15151088, 0.2665018,
 1.7576847, 0.45189196, 0.5970320, 1.7083497, 0.52685668, 0.7109476,
 0.7743397, 0.01236855, 0.1515109, 0.5268567, 0.53406192, 0.2299193,
 0.7508789, 0.21554658, 0.2665018, 0.7109476, 0.22991933, 0.6642375),
 byrow = TRUE, nrow = 6 )

Nreps = 200 # number of replications
n = 100 # sample size
results = data.frame() # an object to store the results in
set.seed( 19501111 ) # make the process reproducible to assist with debugging if needed

for( ii in 1:Nreps ) {
  Samp = mvrnorm(n = n, mu = rep(0,6), Sigma = Sigma0 ) # generate sample
  pca.XX = prcomp( Samp )
  eigvals = pca.XX$sdev ^ 2 # prcomp gives us the standard deviations, so square them
  cond = eigvals[1]/eigvals[6]
  res.incr = data.frame( replication = ii,    # So we can find problems
                         eig1 = eigvals[1],  # We are constructing one row of a dataframe
                         eig2 = eigvals[2],  # at a time so have to do this clunky bit
                         eig3 = eigvals[3],
                         eig4 = eigvals[4],
                         eig5 = eigvals[5],
                         eig6 = eigvals[6],
                        cond.number = cond   # The condition number.
                        )
  results = rbind( results, res.incr ) # add the new dataframe at the bottom of the old one
}
summary( results ) # see what we have got

# Some graphs to try to understand some relationships
ggplot( results, aes( eig1 ) ) + geom_density()
ggplot( results, aes( eig6 ) ) + geom_density()
ggplot( results, aes( eig1, eig6 ) ) + geom_point()

ggplot( results, aes( eig1, eig6 ) ) + geom_point() + geom_density_2d()

# Density of the condition number
ggplot( results, aes( cond.number ) ) + geom_density() +
  geom_vline( xintercept = cond.number0, colour = "grey" )
# cond.number0 is the condition number found for Sigma0 in Q3
```

Comments on this code

The question we have asked is quite a complicated function of the sample `Samp`, so it is hard to know what to expect. This suggests some general principles for this sort of simulation:

- It is advisable to try the loop out with only a small number of replications at first (1, then 10, maybe) before going to the full number wanted.
- Consider writing functions for the two parts:

  - generate the data;
  - compute the statistics needed for the loop output `res.incr`.

  In this way the structure of the code may be made clearer, so that others (including future you) can understand it more easily.

- In situations where the computations based on `Samp` (for `res.incr`) in each replication are more complicated than here, you may want to create and output complicated R-objects (matrices, dataframes, entire PCA objects, ...) which will not fit into a dataframe easily. An alternative is to use `lists` instead of the dataframes `results` and `res.incr`.

Learning to use a `list` is worth the effort if you are doing a lot of this sort of thing. Use of a `list` instead of a dataframe for `results` would help avoid the "clunky" part of the code above, for example, where we have copied `eigvals` out by hand into `res.incr`. We will not discuss them further here.

- Sometimes there will be errors generated by failure to satisfy some conditions (for example, dividing by an unexpected 0). The R help on `try` shows a way to deal with occasional failures in simulations.

Carry out the following tasks:

a) For each eigenvalue calculate the mean across all 200 simulations and compare this mean with the population value. Comment: is the mean larger or smaller than the population values? How does it change from the first to the sixth eigenvalue?

b) Show separate density plots (smoothed histograms) for all 6 eigenvalues and comment on the shape of each plot. Show the population value as a vertical line in each of these plots.

c) Your plots will have a symmetric look to them, which introduces an appearance of greater variability than is really present. Why does this happen? Modify the plots so that the near-symmetry is avoided, without losing any data.

d) [Advanced] If you feel like trying out your programming skills, try also to report the covariance matrix for each `Samp` and display the variability in a thoughtful way.