MAST90105: Lab and Workshop Problems for Week 7

The Lab and Workshop this week covers problems arising from Module 5, Section 2 onwards and the first part of Module 6. In the lab this week, you will enter some commands from this sheet into R-studio and see the results. The problems will not be assigned to groups this week.

1 Lab

Introduction

Data: In this lab we will analyse Tasmanian Rainfall data which are observations representing the maximum daily rainfall in each year from 1995 to 2014 recorded by 34 weather stations in Tasmania. Data source: Australian Bureau of Meteorology (`http://www.bom.gov.au/`)

Goals: (i) Explore data numerically and graphically; (ii) Remind you of probability, density calculations and random number generation;

Task 1 - Exploratory data analysis

You can import the data with the command:

```
tasmania=read.csv("L:/MAST90105MethodsofMathematicalStatistics/EditedRainfall.csv")
```

Please note that if you cut and paste this command from this file, you must paste it into *one* line of code in RStudio. RStudio will not accept file names spread across more than one line and you will not be able to read in the file.

Another tip before you start the commands is to open the File menu, and go to New File and then R script. If you cut and paste the commands from this file into the R Script, you can select and use the Run button to execute code. If you save the file to your desktop, for example as "Rainfall.R", you can then save it on a USB stick - if you have one. Or you can email it to yourself before closing the session. This script can then be opened from RStudio.

There is a copy of this file "EditedRainfall.csv" on the LMS in the Workshop/Labs area, so you can work on this at home if you transfer the file to your computer and change the address of the file to the appropriate one on your computer. Or you could save it on a USB stick from the L: drive or you could email a copy of the file to yourself.

You can view the data and create variables with the following commands:

```
View(tasmania)
dim(tasmania)  # check data dimension
names(tasmania[, 1:5])  # names of the first 5 columns
year = tasmania[, 1]  # create a vector for year
s1 = tasmania[, 2]  # create a vector for station 1 (Burnie)
s2 = tasmania[, 3]  # create a vector for station 2 (Cape Grim)
```

Explore the distribution of extreme rainfall in Burnie and Cape Grim by computing some basic summary statistics:

```
summary(s1)  # 5-number summary plus sample mean
sd(s1)
IQR(s1)  # IQR and sample standard deviaion
# sample percentiles using Type 6 and 7
# approximations
quantile(s1, type = 6)
quantile(s1, type = 7)
summary(s2)
sd(s2, na.rm = TRUE)
IQR(s2, na.rm = TRUE)
```

The sample mean and median suggest that the location for the distribution `s1` is larger than that of `s2`. One observation is missing for `s2` (corresponding to year 2014). Note that we need to use the option `na.rm = TRUE` to remove NAs. Measures of spread are obtained by computing the sample interquartile range (IQR) and sample standard deviation. The variability in `s2` is slightly smaller than that of `s1`.

Next, explore the data by graphical summaries. The following gives an annotated density histogram for `s1` and a smooth density estimate:

```
hist(s1, freq = FALSE, xlab = "Extreme rainfall (Burnie, Tas)")
smooth.density = density(s1)  # fits a smooth curve
points(smooth.density, type = "l", lty = 2, col = 2)  # adds a smooth curve
```

By adding the argument `nclass` in `hist` you can control the number of bins for the histogram (e.g. try to add the argument `nclass=15` or `breaks=15`). Note that this is indicative – R will choose a nearby number which makes a pretty histogram. From the above plot it is clear that that the sample distribution for `s1` is not symmetric.

We can compare the main features of two or more distributions by multiple boxplots on the same plotting window:

```
boxplot(s1, s2, names = c("Burnie Is", "Cape Grim"),
    col = c("yellow", "orange"))
```

The following commandsplots empirical cdfs:

```
ecdf1 = ecdf(s1)
ecdf2 = ecdf(s2)
plot(ecdf1)
plot(ecdf2, col = 2, add = TRUE)
```

The QQ plot compares the empirical distribution of the data against some theoretical distribution. The following compares the sample distribution with the theoretical normal distribution:

```
qqnorm(s1, main = "Normal QQ plot for S1")  # normal QQ plot
qqline(s1)
```

Although the central part of the data distribution is compatible with the normal distribution, note that the right tail deviates from the straight line. Probability theory suggests that extremes such as rainfall maxima follow the so-called Extreme Value distribution with inverse cdf $\mu + \sigma F^{-1}(p)$, where $F^{-1}(p) = -\log(-\log(p))$ is the inverse cdf of the standard EV distribution.

```
Finv = function(p) {
    -log(-log(p))
}  # quantile function
p = (1:20)/21
y = sort(s1)  # order statistics
x = Finv(p)  # theoretical quantiles
plot(x, y, ylab = "Sample quantiles", xlab = "EV quantiles")
# the next command computes and plots the 'best
# fitting line' (more details in next weeks)
fit = lm(y ~ x)
abline(fit)
```

From the last QQ plot the EV model seems to be more appropriate than the normal model, since the points in EV QQ plot are a little closer to the straight line compared to the previous normal QQ plot. As an exercise, repeat the same analysis for other weather stations.

Task 2 - Distributions and random numbers

We look at some of the basic operations associated with probability distributions. There are a large number of probability distributions available, but we only look at a few. If you would like to know what distributions are available you can do a search using the command `help.search("distribution")`.

Here we give details about the commands associated with the normal distribution and briefly mention the commands for other distributions. The functions for different distributions are very similar where the differences are noted below. To get a full list of the distributions available in R you can use the following command:

```
>help(Distributions)
```

For every distribution there are four commands. The commands for each distribution are determined by a prefix indicating the functionality: `d-` returns the height of the probability density function (or probability mass function for discrete data), `p-` returns the cumulative density function `q-` returns the inverse cumulative density function (quantiles), and `r-` returns randomly generated numbers.

For simplicity, in the rest of the task we will examine the familiar normal distribution, but similar calculations can be applied to other distributions.

Given a set of values `dnorm` returns the height of the probability distribution at each point. If you only give the points it assumes you want to use a mean of zero and standard deviation of one. There are options to use different values for the mean and standard deviation; try the following:

```
dnorm(0)
dnorm(0) * sqrt(2 * pi)
dnorm(0, mean = 4, sd = 2)
dnorm(c(-1, 0, 1))
x = seq(-5, 5, by = 0.1)
y = dnorm(x)
plot(x, y, typ = "l")
y = dnorm(x, mean = 2.5, sd = 0.5)
plot(x, y, typ = "l")
```

The second function we examine is `pnorm`. Given a number or a list it computes the probability that a normally distributed random number will be less than that number (i.e. it returns the normal cdf). It accepts the same options as dnorm:

```
pnorm(0)  # Lower tail probability (cdf)
pnorm(1)  #
pnorm(1, lower.tail = FALSE)  # upper tail probability
pnorm(0, mean = 2, sd = 3)
x = seq(-12, 12, by = 0.1)
y = pnorm(x)
plot(x, y, typ = "l")
y = pnorm(x, mean = 3, sd = 4)
plot(x, y, typ = "l")
```

The next function we look at is `qnorm` which is the inverse of `pnorm`. The idea behind qnorm is that you give it a probability, and it returns the number whose cumulative distribution matches the probability. For example, try

```
qnorm(c(0.25, 0.5, 0.75), mean = 1, sd = 2)  # quartiles for N(1,2)
x = seq(0, 1, by = 0.05)
y = qnorm(x)
plot(x, y)
y = qnorm(x, mean = 3, sd = 2)
plot(x, y)
y = qnorm(x, mean = 3, sd = 0.1)
plot(x, y)
```

The last function we examine is the `rnorm` function which can generate random numbers whose distribution is normal. The argument that you give it is the number of random numbers that you want, and it has optional arguments to specify the mean and standard deviation:

```
rnorm(4)
rnorm(4, mean = 3, sd = 3)
rnorm(4, mean = 3, sd = 3)
y = rnorm(200)
hist(y)
y = rnorm(200, mean = -2)
hist(y)
qqnorm(y)
qqline(y)
```

2 Workshop

1. Let $X_1, X_2, \cdots, X_9$ be a random sample from a normal distribution $N(60, 16)$.

   Compute:

   a. $P(58 < \bar{X} < 62)$.

   b. $P(58.8 < \bar{X} < 62.4)$.

2. Suppose you take a random sample of 9 nails from a population of nails used for making decks. Suppose further that the population weights (in grams) are normally distributed with mean 8.78 and variance 0.16. Let S^2 be the sample variance of the nine weights. Find constants a, b so that $P(a \leq S^2 \leq b) = 0.90$.

3. Let $T = \frac{\sqrt{r}Z}{\sqrt{R}}$ have a t-distribution with r degrees of freedom.

   a. What are the distribuions of Z, R?

   b. What are:

      i. $E(Z)$

      ii. $E(Z^2)$,

      iii. $E(\frac{1}{\sqrt{R}})$,

      iv. $E(\frac{1}{R})$?

   c. Find $E(T)$.

   d. Find $E(T^2)$.

   e. Find $Var(T)$.

4. Suppose there is a random sample of 9 observations from a normal population with mean μ and variance σ^2. Let T be defined as:

$$T = \frac{3(\bar{X} - \mu)}{S}$$

where $\bar{X}$ is the sample mean and S^2 is the sample variance.

   a. Find $t_{0.025}$ such that $P(-t_{0.025} < T < t_{0.025}) = 0.95$.

   b. Solve the inequality $[-t_{0.025} < T < t_{0.025}]$ so that μ is in the middle.