Introduction to R

Exercise 1

1. Start R . Select R from the `Start -> All Programs -> R` menu.
2. On the command line, type `demo(graphics)`. Follow prompts. You might prefer to re-size the windows.
3. On the command line, type `demo(image)`. This demonstration is concerned with representations of 3D data.
4. Get help on the function `q`, by typing `?q`.
5. Quit R , by typing `q()`.

It will be useful to create a location for files for this course. Your course account includes a network drive, drive `H:`. Create a new directory on this drive, called `rcourse` say. Follow these instructions to create a shortcut on the desktop for R to open using this location.

1. Right click on the desktop and select `New -> Shortcut`
2. Select `browse` in the dialogue box.
3. Find `c:\Program Files\R\rw2001\bin\Rgui.exe`
4. Click `Next`.
5. Enter a suitable name for the shortcut, and click `finish`.

Now, modify the properties of the shortcut as follows

1. Right click the R shortcut, and select `Properties`.
2. Change the `Start in` entry to the network directory you created, and click `Ok`.

Now, start R using this new shortcut. Type `shell("dir")`. Which directory is referred to? Finally, quit this R session.

Introduction to R

Exercise 2

Start R using the shortcut created earlier. Download the file

`http://stats.ma.ic.ac.uk/das01/public_html/RCourse/hills.txt`

and save it in the directory created earlier. This file contains the Scottish hill races data set. Now, enter the following commands

1. Read the file, assigning the result to the object `hills`

   ```
   hills <- read.table("hills.txt")
   ```

2. Examine the object. Note column and row names

   ```
   hills
   ```

3. Construct a scatterplot matrix

   ```
   pairs(hills)
   ```

   How many variables do you think such plots are suitable for?

4. Make the columns of the hills object available by name

   ```
   attach(hills)
   ```

5. Construct a scatter plot. The function call in this way means the first argument is the horizontal axis

   ```
   plot(dist,time)
   ```

6. Interact with the plot to label points - right click to finish

   ```
   identify(dist,time,row.names(hills))
   ```

7. Compute a linear regression of time against distance

   ```
   lm(time~dist)
   ```

8. Obtain more information about the regression

   ```
   summary(lm(time ~ dist))
   ```

9. Add the least squares regression line - note anonymous function call

```
abline(lm(time~dist))
```

10. Obtain some diagnostics plots - note the different arguments to the `plot` function. Be aware of the prompt in the Console.

```
plot(lm(time~dist))
```

11. there are many pre-defined system objects. Display the value of pi - note that this is a reserved word

```
pi
```

12. List objects in current working space

```
ls()
```

13. Display the object ls

```
ls
```

14. Create a copy of the `hills` object

```
hill.cp <- hills
```

15. List objects in current working space

```
ls()
```

16. Delete the copy. Note that there is `no` undelete functionality.

```
rm(hill.cp)
```

Finally, quit this session.

Introduction to R

Exercise 3

This sheet is primarily concerned with vectors and arithmetic.

1. Create a vector of coefficients for a quadratic equation, using the `sample` function. Here, we draw a sample of size 3 from $-20, -19, \ldots, 19, 20$ *with* replacement

   ```
   coeffs <- sample(-20:20,3,replace=T)
   ```

2. Determine the class of the object `coeffs`.

   ```
   class(coeffs)
   ```

3. Determine the length of the object `coeffs`

   ```
   length(coeffs)
   ```

4. Determine the names associated with the vector

   ```
   names(coeffs)
   ```

   What was the result?

5. Assign some names. Note the function call occurring on the right hand of the assignment operator.

   ```
   names(coeffs) <- c("a","b","c")
   ```

   What is the class of the names of the vector `coeffs`?

6. Prepare to plot the equation, by constructing a regularly spaced vector for the horizontal axis

   ```
   x <- seq(-3,3,length=200)
   ```

7. Evaluate the quadratic at each point in vector `x`

   ```
   y <- coeffs[1]*x^2+coeffs[2]*x+coeffs[3]
   ```

8. Construct the plot

   ```
   plot(x,y)
   ```

   ```
   plot(y=y,x=x)
   ```

   Note the different styles of function call.

9. Does the equation have real roots? Compute the discriminant

```
coeffs[2]^2-4*coeffs[1]*coeffs[3]
```

10. Oops, we didn't retain the value! R stores the last unassigned object in the system object .Last.value

```
disc <- .Last.value
```

11. Create a vector of type character, and display the second element

```
chr.vec <- sample(letters,5); chr.vec[2]
```

Problems

1. Compute the real roots of the quadratic equation

$$x^2 + x + 1 = 0$$

Recall the formula for the roots of a quadratic

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

And use `sqrt` to compute a (positive) square root.

2. Without using R , determine the result of the following computation

```
x <- c(1,2,3)
x[1]/x[2]^3-1+2*x[3]-x[2-1]
```

3. Generate a regular grid between -50 and 50. Construct separate plots of $\log(x)$, $\exp(x)$, $\sin(x)$, $\sin(2x)$, $\sin(x)/\cos(x)$. Examine the cumulative sum of the final function. Experiment with the argument `type` of the plot function.

Introduction to R

Exercise 4

This sheet is concerned with logical operations and factors.

1. Create a logical vector

   ```
   x <- seq(-3,3,length=200) > 0
   ```

2. negate this vector

   ```
   !x
   ```

3. Compute the truth table for logical AND

   ```
   c(T,T,F,F) & c(T,F,F,T)
   ```

4. Explore arithmetic with logical and numeric

   ```
   1:3 + c(T,F,T)
   ```

5. Compute the intersection of $\{1, 2, \ldots, 10\}$ and $\{5, 6, \ldots, 15\}$

   ```
   intersect(1:10,5:15)
   ```

6. Create a factor

   ```
   drinks <- factor(c("beer","beer","wine","water")
   ```

7. Examine the representation of the factor

   ```
   unclass(drinks)
   ```

Problems

1. Compute the truth table for logical OR. The function R computes the logical EXCLUSIVE-OR. What is the difference between the two?
2. Consider the vector `1:K`, where `K` is a positive integer. Write an R command that determines how many elements in the vector are exactly divisible by 3.
3. Write an R command to evaluate the proportion of `beer` in the `drinks` factor object.

Introduction to R

Exercise 5

This sheet is concerned with data frames and matrices.

1. Examine the row and column names of the hills object

```
row.names(hills)
names(hills)
```

2. Create data frames. Note row and column names

```
x1.df <- data.frame(1:10,I(letters[1:10]),factor(letters[1:10]))
x2.df <- data.frame(X1=1:10,X2=I(letters[1:10]),X3=factor(letters[1:10]))
```

3. Compute the mean of column X1

```
mean(x2.df$X1)
```

4. Create a matrix

```
x.mat <-matrix(1:12,nrow=3,ncol=4)
```

5. Examine the default dimension names of the matrix

```
dimnames(x.mat)
```

6. Assign some dimnames

```
dimnames(x.mat) <- list(letters[1:3],letters[1:4])
```

7. Combine matrices

```
xx <- cbind(x.mat,x.mat)
xxx <- rbind(x.mat,x.mat)
rbind(xx,xxx)
```

8. Explore indexing

```
x <- 1:10
names(x) <- letters[x]
x[1:3]
x[c(1,10)]
x[c(-1,-2)]
x[ x > 5]
x[c("a","d")]
x[]
jj1 <- matrix(1:100,ncol=10)
jj1[1:5,]
jj1[1:4,x[x <3]]
```

9. Compute row and column sums of a data frame

```
x <- matrix(1:10,ncol=2)
lapply(x,sum)
sapply(log(x),sum)
```

10. Create a list, and examine elements

```
x.lis <- list(a=1:10,b=letters[1:3],b=matrix(1:10,ncol=2))
x.lis$1
x.lis[[2]]
```

11. Element-wise arithmetic with matrices

```
x.mat <- matrix(1:10,ncol=2)
x.mat+1
x.mat + x.mat
```

12. Matrix multiplication

```
x.mat %*% t(x.mat)
```

13. Compute row and column sums of a matrix

```
apply(x,1,sum)
apply(x,2,sum)
```

Problems

1. Construct a 2×2 data frame, `X` say. Experiment with `X^(1:K)`, where `K` takes values 1:4. How does the recycling rule behave? What happens if you remove the brackets from the command?
2. The function `system.time` returns timings for R operations. Examine the help system about this function. For a $10^7 \times 2$ matrix, X, and vector y of length $10^7/2$ compute (a number of times) $X^t y$ using matrix multiplication and the function `crossproduct`. Which is quicker?

Introduction to R

Exercise 6

This sheet is concerned with usage issues.

1. Determine what objects are in the current workspace

   ```
   ls()
   objects()
   ```

2. Create and edit a new data frame

   ```
   a.df <- data.frame()
   fix(a.df)
   ```

3. Create, then delete some objects. Note the multiple assignment

   ```
   jj1 <- jj2 <- jj3 <- a.df
   rm(a.df)
   rm(list=objects(pattern="jj*"))
   ```

4. Examine the search path

   ```
   search()
   attach(hills)
   search()
   ```

5. Write the `hills` object to a file

   ```
   write.table("test.dat",hills)
   ```

   Where is this file situated on the file system. Examine the default output format by viewing the file with your favourite text editor.

6. Get some help

   ```
   ?mean
   help.start()
   ```

7. Do a silly large computation.

   ```
   sin(matrix(0,nrow=5000,ncol=5000))
   ```

   Use ⟨ESC⟩ to interrupt. Note that this might take some time to return control to the console.

Problems

1. Generate a matrix of size $n \times p$. Use the function `as.data.frame` to coerce the matrix to a data frame. Which object requires more storage space?

Introduction to R

Exercise 7

This sheet is concerned with basic statistics and graphics.

1. Generate a sample of random normal deviates, and a sample of random exponential deviates.

```
x<- rnorm(50)
y <- rnorm(50,0,1)
```

2. Compute some summaries

```
mean(x)
sqrt(var(x))
cor(x,y)
cor(cbind(x,y))
```

3. Try the `summary` function

```
summary(x)
summary(cbind(x,y))
```

4. Let $X \sim N(0,1)$ and $Y \sim Exp(2)$. Compute $P(X > 1.644)$ and find q such that $P(Y < q) = 0.75$.

```
1-pnorm(1.644)
qexp(0.75,2)
```

5. Use the `sample` function to obtain a random sample of 10 realisations in a biased coin experiment

```
sample(c("Head","Tail"), 10,probs=c(0.3,0.7),replace=T)
```

6. Load the package `SuppDists` and examine the help

```
help(package="SuppDists")
```

7. Experiment with `set.seed`

```
set.seed(1)
runif(10)
set.seed(1)
runif(10)
runif(10)
```

8. Test to see if sample `x` is consistent with an Exp(1) distribution using a QQ plot

```
plot(qexp(ppoints(x),1),sort(x))
abline(0,1)
```

Examine the help for the function `qqnorm`.

9. Compare the two samples with a QQ plot

```
qqplot(x,y)
abline(0,1)
```

10. Compare the two samples with box plots

```
boxplot(x,y)
```

11. A simple alternative to display the two samples

```
plot(c(x,y),rep(0:1,c(length(x),length(y))),xlab="",ylab="")
```

12. Plot a histogram of `x` and a box plot of `y`, in the same figure.

```
par(mfrow=c(2,1))
hist(x)
boxplot(y)
```

13. Consider the Pima Indians data: a collection of variables observed on a particular group of native American Indians who are either healthy or diabetic. This data includes measurements of tricep, skinfold and blood glucose level.

    First, load the `mlbench` package.

```
attach(PimaIndiansDiabetes)
plot(triceps,glucose,type="n")
plot(triceps[diabetes=="neg"],glucose[diabetes=="neg"],xlab="Tricep",ylab="glucose")
points(triceps[diabetes=="pos"],glucose[diabetes=="pos"],xlab="Tricep",ylab="glucose",col=2,pch=2)
legend(40,50,c("Diabetes","Healthy"),pch=1:2)
```

Problems

1. Examine the built in `ChickWeight` data (the help gives background about the data). The function `split` will prove useful to do the following (as will a script)

   (a) Construct a plot of weight against time for chick number 34.

   (b) For chicks in diet group 4, display box plots for each time point.

   (c) Compute the mean weight for chicks in group 4, for each time point. Plot this mean value against time.

   (d) Repeat the previous computation for group 2. Add the mean for group 2 to the existing plot.

   (e) Add a legend and a title.

   (f) Copy and paste the graph into Word.

   Note that some of these steps will be easier once we have some more programming expertise.

Introduction to R

Exercise 8

This sheet is concerned with control structures.

1. Write an R expression to determine if two sets, A and B, represented as integer vectors are disjoint. If they are disjoint, display elements of set A otherwise display elements of set B. (Examine the help for functions `print` and `cat`).

2. Write R codes that takes the coefficients of a quadratic equation, and outputs an appropriate message for the cases of (i). two distinct roots ($b^2 - 4ac > 0$) (ii) coincident roots ($b^2 = 4ac$) or (iii). complex roots ($b^2 < 4ac$).

3. Let vector `y` be the logarithm of a random sample from a standard normal distribution, $N(0, 1)$. Use the `ifelse` function to replace missing values with the value 9999.

4. Let n be a large integer. Compute a vector `x` containing n random uniform deviates. Embed the following code in the `system.time` function

   ```
   for (i in 1:n) y[i] <- sin(x[i])
   ```

   Do this a few times. Now, time the call

   ```
   y <- sin(x)
   ```

   Which is faster?

5. Compound interest can be computed using the formula

   $$A = P \times (1 + R/100)^n$$

   where P is the original money lent, A is what it amounts to in n years at R percent per year interest.

   Write R code to calculate the amount of money owed after n years, where n changes from 1 to 15 in yearly increments, if the money lent originally is 5000 pounds and the interest rate remains constant throughout the period at 11.5%.

6. Write a loop structure to scan through an integer vector to determine the index of the maximum value. The loop should terminate as soon as the index is obtained. (Don't worry about ties).

   Of course, this is not a good way to do this! Examine the help for the `rank`, `sort` and `order` functions.