Exercises on using R for Statistics and Hypothesis Testing

Dr. Wenjia Wang

School of Computing Sciences, UEA University of East Anglia

Brief Introduction to R

R is a free open source statistics and mathematical computing environment. The R project website is www.r-project.org. A brief introduction to R can be found from its web site: http://cran.at.r-project.org/doc/manuals/R-intro.html

- ### Installing R for Windows

You can download and install the current Windows version of R (the latest versions is 2.9.1) from above web site (or clink the link below). You can either run the program, or save the program to your computer and then run it to install R. When installing, you can probably just accept the default settings. Current Windows Version of R

- **Further Details**

You can see the following link with more details about the latest version of R. The link called **r-patched release** allows you to install a version that contains minor patches since the current official release (but note that the patched version is not the official version). The link called **r-devel release** is a version that is under development and likely contains bugs.

  - CRAN Page with latest R version and associated files and instructions

- **Start R**

Once you have installed it, you will get a window with a prompt "R 2.9.0", clink it and R windows will be up running.
You can then enter `help.start()`at the prompt to open a html help page. Help is available in R for most functions using the `help` command, e.g. for help on `setwd` type `help(setwd)`. If you type a function name without the (), the R code for the function is displayed, if you type an object name the contents of the object are displayed.

Enter `getwd()` at the prompt. It tells you the default directory R will load files from and save them to. Create a directory in your C drive, call it `rfiles`. Now enter `setwd("C:/rfiles")` at the R prompt, or in RGui, click File and choose Change dir to change the working directory. Enter `getwd()` to check. Note that in R you use forward slash `/` as separator, not backslash \ as in Windows.

- **Quit R**

Under the R window, you can stop the R by either
(1) clicking File and then Exit,
(2) or issuing a command: >q()
You will be prompted for saving your work environment, save it if you wish.

1. Basic Statistics and Analysis of Variance

For this example, we will use one of R's built in datasets, `GrowthData`.

```
> help(PlantGrowth)
```

The help page will tell you that this data shows yields (as measured by dried weight of plants) obtained under a control and two different treatment conditions. Let's look at the data:

```
> PlantGrowth
   weight group
1    4.17  ctrl
2    5.58  ctrl
...
11   4.81  trt1
12   4.17  trt1
...
29   5.80  trt2
30   5.26  trt2
> summary(PlantGrowth)
     weight       group
 Min.   :3.590   ctrl:10
 1st Qu.:4.550   trt1:10
 Median :5.155   trt2:10
 Mean   :5.073
 3rd Qu.:5.530
 Max.   :6.310
```

As you can see, there are only 30 data points, 10 in each of the three categories. This is a bit small, but anyway, we use it as a data set to learn basic functions of R.

Note: you can use command `help(datasets) or library(help="datasets")` to see all the built-in data sets in R Datasets Package.

Standard Deviations

What are the standard deviations of the plant weights - overall and for the three groups?

- Using the `sd()` function to compute the standard deviation of a given attribute,
- `Using the tapply(x, group, FUN)` function to apply the provided function FUN to each of the groups, e.g. `tapply(PlantGrowth$weight, PlantGrowth$group, sd)` to apply the function sd to calculate the standard deviation for the weight values of each group.

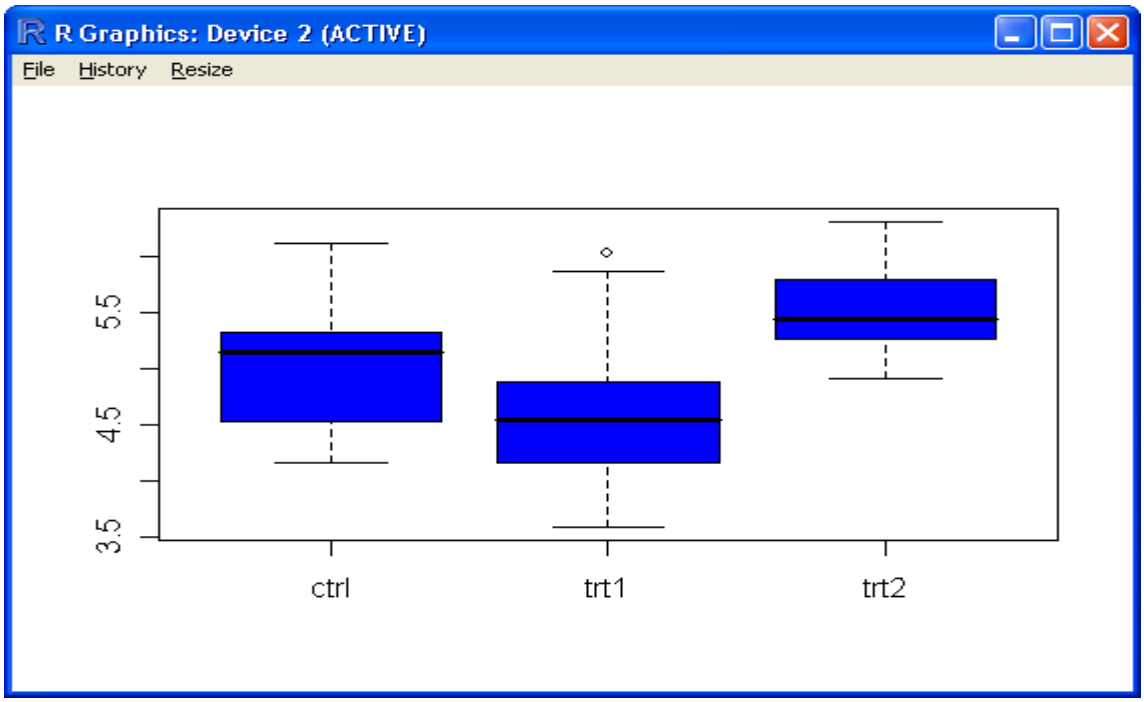
```
> sd(PlantGrowth$weight)
[1] 0.7011918
> tapply(PlantGrowth$weight, PlantGrowth$group, sd)
     ctrl      trt1      trt2
0.5830914 0.7936757 0.4425733
```

Can we assume a constant spread across the groups?

Simple Plots

Using the boxplot90 function to plot the **box and whiskers plot** for the data**:**

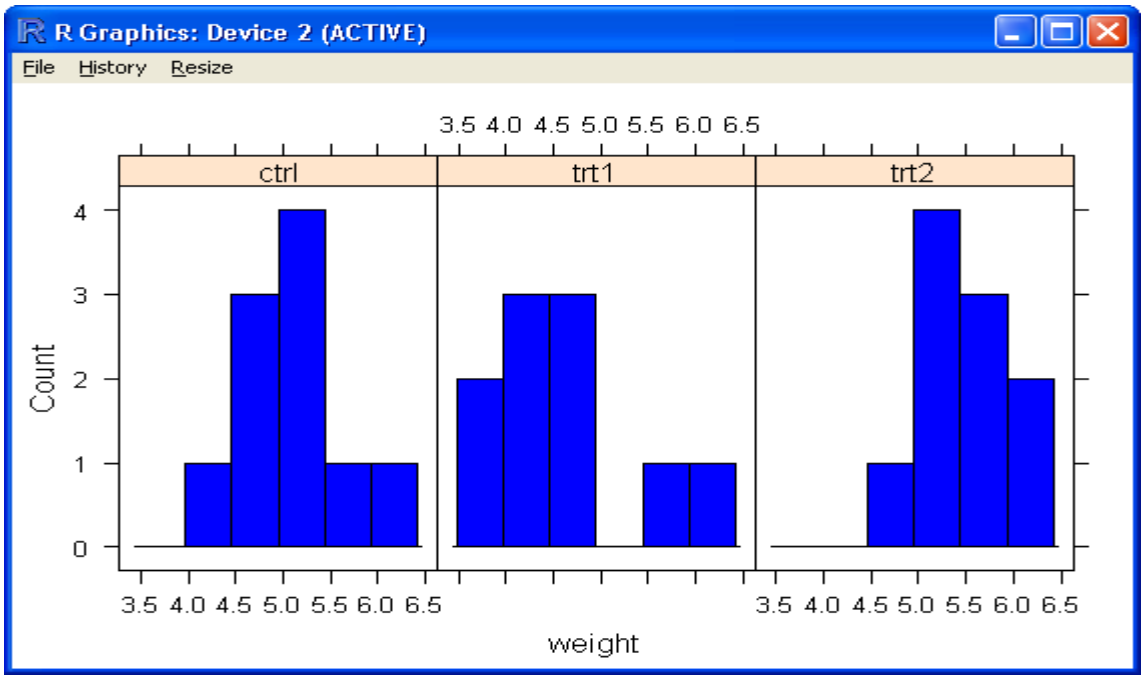
```
> boxplot(weight ~ group, data=PlantGrowth, col="blue")
```



Notice that the largest weight in Treatment One has been shown as an outlier in the box plot (above).

Lets draw a conditional histogram:

```
> library(lattice)
> histogram(~ weight | group, data=PlantGrowth, col="blue", type="count")
```



Can we assume we have normal distributions here? There are some skews, and Treatment One might actually be bimodal.

Hypothesis Testing

The experimental hypothesis would be that the treatment affects the plant growth. Looking at the plots this looks reasonable - Treatment One (`trt1`) seems to have reduced growth relative to the control (`ctrl`) while Treatment Two (`trt2`) has improved it. But can you show this with a statistical analysis?

Our null hypothesis is that the three groups have the same mean. Let's investigate this with an ANOVA/F-test.

This first example uses the `aov()` function to build a model and analyse it.

We write `weight ~ group` to say the response variable `weight` depends on the predictor variable `group`. Type `help("~")` and `help(formula)` for more details.

```
> analysis <- aov(weight ~ group, data=PlantGrowth)
> summary(analysis)
            Df  Sum Sq Mean Sq F value  Pr(>F)
group        2  3.7663  1.8832  4.8461 0.01591 *
Residuals   27 10.4921  0.3886
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the next example, we first build a linear model (using the function `lm()` command) and then use the `anova()` command to analyse that.

```
> fit <- lm(weight ~ group, data=PlantGrowth)
> anova(fit)
Analysis of Variance Table
```

```
Response: weight
          Df  Sum Sq Mean Sq F value  Pr(>F)
group      2  3.7663  1.8832  4.8461 0.01591 *
Residuals 27 10.4921  0.3886
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Reassuringly both methods give the same answer.

The p-value for the F-statistic is 0.01591 (i.e. 1.6%), which R has marked with a single asterix because it is in the range 0.01 to 0.05. As a rule of thumb this would be interpreted as only "weak" evidence to reject the null hypothesis (that the three groups have equal means).

i.e. There appears to be only weak statistical support for the experimental hypothesis that the treatment affects the plant growth. Maybe they should grow some more plants (use a bigger dataset).

However, is there any reason to doubt the validity conclusion? Does the data satisfy the model requirements? We have to assume the treatment groups are independent. What about normality? Looking at the graphs maybe not, but its hard to say from just ten data points. Looking at the standard deviations, can we assume a constant spread (homogeneity of variance)? Treatment One (0.79) is almost double Treatment Two (0.44)...

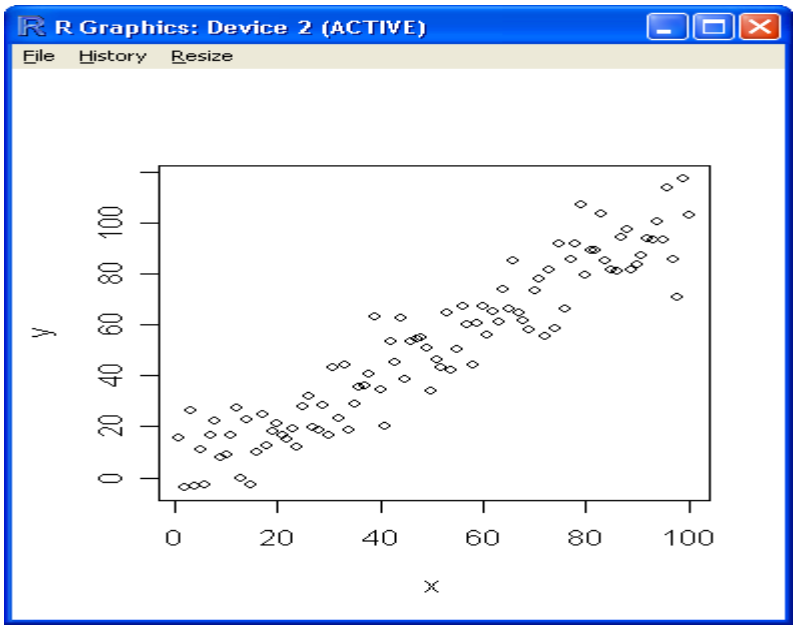
2. Scatter Plots and Linear Regressions

Given two vectors x and y describing a set of points, `plot(x,y)` is all you need to draw an x-y scatter plot.

To demonstrate this, first of all I'm going to create some fake data. For x I will simply have the integers 1 to 100, while for y I am going to use x plus some random noise. To do that, I generate 100 random numbers using a normal distribution with mean zero and standard deviation ten:

```
> x <- c(1:100)
> y <- x + rnorm(100, mean=0, sd=10)
> plot(x,y)
```

If you already had a graph on display, this will replace it; otherwise a second window will appear:



There are several ways of calculating a best fit line (linear regression), for example using the least squares fit `lsfit()` function:

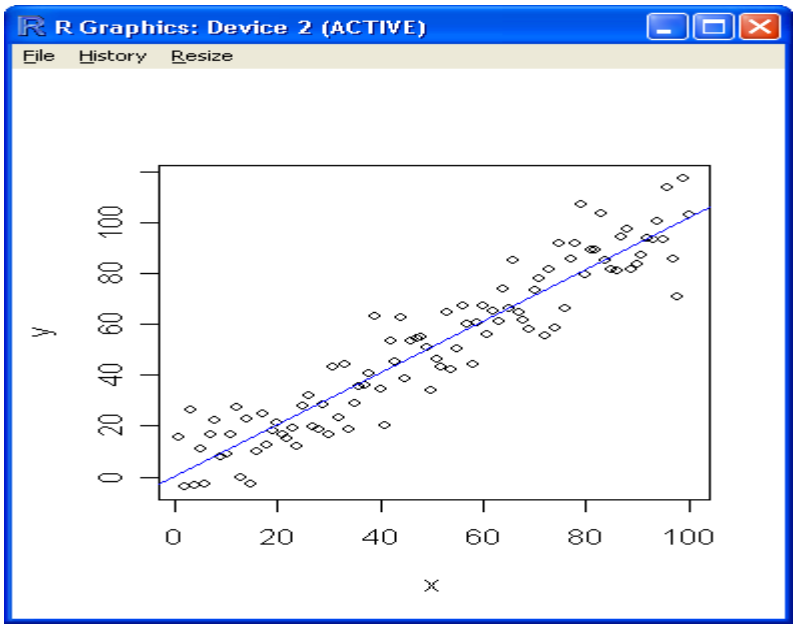
```
> fit <- lsfit(x,y)
> fit$coefficients
 Intercept           X
-0.2385198 1.0007447
```

These values are very close to what you would expect for this artificial data (intercept zero and gradient one).

Once you have the coefficients, use the `abline()` function to actually draw it:

```
> abline(fit, col="blue")
```

This will update the existing graph (assuming you didn't close that window!):



An equivalent result is obtained using the `lm()`, or linear model, function:

```
> fit <- lm(y~x)
> fit$coefficients
(Intercept)         x
0.04596275 1.01859341
```

This uses R's model definition syntax, where we write `y ~ x` to say y depends on x. Type help("~") and help(formula) for more details.

3. R Scripts

You can put your commands in the order you wish to run them in the command lines in R Console into a simple text file and save it as an R script file, and run it later any time. e.g.

- **Create or Edit R Script:**

In R console, RGui, click File, then choose New Script, then R-Editor window appears (you may use any other text editor), you can write commands there and save it in your R work directory, e.g. you may put some the above commands into an R script and save it as ScatterPlotLRegr.R

```
# any things after # are the comments
# An R script for Scater plot and linear regression
# script file: ScatterPlotLRegr.R

x <- c(1:100)
y <- x + rnorm(100, mean=0, sd=10)
plot(x,y)
fit <- lsfit(x,y)
abline(fit, col="blue”)
```

- **Run a R script:**

You can then run it in R console by using the function
`>source("ScatterPlotLRegr.R")`

4. Perform statistical tests on a real data set

In this exercise, you will use R to perform some simple statistical analysis and hypothesis testing on a real data set, i.e. twins data.

The data were generated from a study[1] that was carried out by Jensen in 1974 on the topic that if a person's intelligence is primarily influenced by genetics or the environment, a long standing debatable issue. The study was set up to investigate the IQ scores of twins who were raised either with their biological parents or in foster families.

Copy the data set `twins.txt` file (it will be provided) to your C:/rfiles folder.

4.1. Load the data set

Now load the twins data by entering the following:

```
twins.data<-read.table("twins.txt",header=TRUE,row.names=1)
```

(Note that the file name is enclosed in `""`. You can read files from any directory by using the full path name, for this exercise we will only use the default directory.)

The `read.table` command loads the file as a *data.frame* object called *twins.data*. This is like a table, and we can perform functions on the data in it. We have used the options `header=TRUE` to tell R that the first row contains the header, and `row.names = 1` to tell R that the first column is not data, but the names of the rows (in this particular dataset they are numbers).

Enter `twins.data` to see its contents. You can also look at part of a data.frame, e.g. enter `twins.data[1:5,]` just to see the first 5 rows. (In R we refer to the rows and columns of data.frames using `[row,column]`, in this example the column section is empty, which means *all* columns are included. Series of rows and columns are referred to using a colon, here 1:5 means rows 1 to 5. The order can be reversed, e.g. 5:1. If several are required use c(…), e.g. `twins.data[c(1,2,7,5),]` will display rows 1, 2, 7 and 5. Any order can be used.)

The columns in a data.frame can also be named. We did this when we loaded twins.txt. We can refer to the columns directly by their names, by entering the object name joined with `$name` (try entering `twins.data$Home` and `twins.data$Foster`). Note that all object and function names in R are case-sensitive.

4.2 Plotting data.

R is very good for plotting graphs. A simple scatter plot can be produced by entering `plot(twins.data)`, and a boxplot with `boxplot(twins.data)`.

A plot can be copied directly into programs such as Word as follows: right-click the plot, select "Copy as metafile", open a Word document, then use the Word paste command.

[1] Jensen, A.R. *Behavioral Genetics*, 4 (1), pp1-28, 1974.

(To use R plots in LaTeX right-click the plot, select “Save as postscript…” type a name for the file and save it. Then you can include the file in a LaTeX document in the usual way.)

4.3. Statistical tests: t-test.

Most statistical functions are included in the basic R installation. In addition many additional packages providing additional functions have been published and are available via the R project website.

Now we will perform a t-test. The command in R for the t-test is `t.test()`.

Perform the t-test by entering `t.test(...)`.
(You will need to replace `...` by the parameters for your data etc. between the `()`. Use the R help command, as described previously, to open the help page for the t-test, to find out how to use it.)

If you simply enter the command `t.test(...)` the results will be printed at the prompt. It is also possible to store the results of an R function as an object.
Enter:
`twins.ttest<-t.test(...)` to store the results of the function t.test in an object which is to be called twins.ttest.

Now enter `twins.ttest` to print the results. The t-test in R returns an object of class htest. This type of object is produced by several of the statistical tests in R and can contain several pieces of information. The parts of an R object can be given names (as we did with the columns in `twins.data`).
To see the names for `twins.ttest` enter `names(twins.ttest)`

It is possible to get just part of an object by using the name. To do this enter the object name joined with `$name`, e.g. enter `twins.ttest$statistic` This allows you to extract specific parts of an object, and is most useful if you write your own methods. For example, if you need to batch process multiple data sets, but you only need part of the outputs of the functions you are using.

4.4 Write R Scripts

Write an R script to perform above tasks, i.e. load the data, plot the data and do t-test, and save it with a name `twinsDataAnalsis.r`, into your rfiles folder. Try to run it to see if it works.