Beginning R

Marc H. Mehlman

29 April 2017

1 Working Directory of R

Before you start R, create a working directory where you will work with R (where you put your datasets and save your work). In my case, I made the directory "/home/mhm/Rwork". When you start R, you can find your working directory by giving the "getwd()" command. If the current working directory is not the directory where you want to work, you can go to your desired working directory using a "setwd(working/directory)" command.

```
> getwd()
[1] "/home/mhm"
> setwd("/home/mhm/Rwork")
> getwd()
[1] "/home/mhm/Rwork"
```

Keep in mind that windows is case insensitive (it does not care if a letter is capitalized or not), but R and some operating systems are case sensitive.

2 Datasets

2.1 Included Datasets

see

http://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

One can also view the included datasets when using R by typing

```
> library(help="datasets")
> trees
```

Here we see that "trees" is a R database. One types "q" to get out of the list of R databases. Then I typed "trees" to see the variables and values of the database "trees".

2.2 Inputing Data By Hand

```
> dat=c(1, 2, 3, 4, 5)
> dat
[1] 1 2 3 4 5
```

2.3 From an ASCII File

The format of the file is one of columns – one variable per column. The first line is the names of variables (if there is a space in the name, put the entire name in double quotes – I don't like spaces in variable names). For instance, the file "data.dat" could be:

quantity	resident	size	color
1	T	5.3	blue
4	F	2.1	green
3	T	.	yellow
7	T	-1.2	white

Notice missing data is represented ".". To input this file into the R dataset "dat" one uses

```
> dat=read.table("data.dat",h=TRUE)
> dat
  quantity resident size  color
1        1     TRUE  5.3   blue
2        4    FALSE  2.1  green
3        3    FALSE    . yellow
4        7     TRUE -1.2  white
```

Notice that "quantity" is read in as an integer, "resident" as TRUE/FALSE, "size" as a real number and "color" as text.

If one is using Windows or MacOS, once can use the command

```
> dat=read.table(file.choose(),h=TRUE)
```

instead of the read.table command above. One will then be prompted in a system window to select a file to be imported.

2.4 From an Spreadsheet File

The format of the spreadsheet should be the same as for the ASCII files; each column a variable and the first row contains the variable names. To read the data into R you have two options:

Option #1: One can copy and paste the data from an open spreadsheet into an ASCII file and then read the ASCII file into R.

Option #2: One can open the spreadsheet and then save it in the "*.csv" file format. Then one can read it into R as follows:

```
> dat=read.csv("data.csv",h=TRUE,sep=",")
> dat
```

where sep defines the delimiting symbol (it does not have to be a ",").

3 Saving your Results from R

To save ASCII from R, just copy and then paste into whatever other document you are working on (text, MicroSoft Word, etc.). For graphical output, use

```
> pdf("boxplot.pdf") # save graph to "boxplot.pdf".
> boxplot(trees$Height)
> dev.off() # stop writing to the file.
```

When writing to a file, the output is not simultaneously being written to the screen. The file "boxplot.pdf" will appear in your current directory. One can use the command jpg("boxplot.jpg") instead to produce a jpeg output. A list of possible formats is

Command	Type	Comments
JPG	jpeg	Can be used anywhere, but doesn't resize
PNG	png	Can be used anywhere, but doesn't resize
WMF	win.metafile	Windows only; best choice with Word; easily resizable
PDF	pdf	Best choice with pdflatex; easily resizable
Postscript	postscript	Best choice with latex and Open Office; easily resizable

4 Help

```
?hist               # help for hist (histogram)
apropos("test")     # lists functions having to do with test.
help(t.test)        # gives help page for t.test.
```

5 Manipulation

```
> x=5+sqrt(3)
> if(x>=6) x=9 else x+8
> x
[1] 9
```

6 A Descriptive Statistics Sample R Session

```
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> library(help="datasets")
> trees

> Girth=trees$Girth
> hist(Girth) # histogram
> stem(Girth) # stem and leaf plot

  The decimal point is at the |

   8 | 368
  10 | 57800123447
  12 | 099378
  14 | 025
  16 | 03359
```

```
  18 | 00
  20 | 6

> stem(Girth, scale=2)

  The decimal point is at the |

   8 | 368
   9 |
  10 | 578
  11 | 00123447
  12 | 099
  13 | 378
  14 | 025
  15 |
  16 | 03
  17 | 359
  18 | 00
  19 |
  20 | 6

> mean(Girth)
[1] 13.24839
> median(Girth)
[1] 12.9
> quantile(Girth) # notice quantile interpolates
   0%   25%   50%   75%  100%
 8.30 11.05 12.90 15.25 20.60
> quantile(Girth,0.90) # percentiles
 90%
 > length(Girth[Girth<=14.2])/length(Girth)*100 # percentile of 14.2
[1] 70.96774
> var(Girth) # sample variance
[1] 9.847914
> sd(Girth) # sample standard deviation
[1] 3.138139
> sqrt(var(Girth))
[1] 3.138139
> fivenum(trees$Girth) # five number summary
[1]  8.30 11.05 12.90 15.25 20.6
> IQR(Girth)
[1] 4.2
> Girth[Girth>median(Girth)+1.5*IQR(Girth) | Girth<median(Girth)-1.5*IQR(Girth)] # outliers
[1] 20.6

> boxplot(Girth) # boxplot
> boxplot(USJudgeRatings$DMNR,USJudgeRatings$DILG)
> pdf("hist.pdf") # save graph to "hist.pdf".
> boxplot(trees$Height)
> dev.off() # stop writing to the file.

> q() # quit
```

7 Distributions

r random, d density, p distribution, q quantile

```
binom(n,p)  # binomial
chisq(df)   # chi squared
f(df1,df2)  # f
geom(p)     # geometric
norm(m,sd)  # normal
pois(m)     # poisson
t(df)       # t
unif        # continuous uniform
```

Sample R Session:

```
> dbinom(5,10,0.5)  # X ~ BIN(10,1/2)    P(X = 5)
[1] 0.2460938
> ppois(3,2)        # X ~ POI(2)         P(X <= 3)
[1] 0.8571235
> pnorm(7,5,2)      # X ~ N(10,1/2)      P(X <= 7)
[1] 0.8413447
> pnorm(1,0,1)      # Z ~ N(0,1)         P(Z <= 1)
[1] 0.8413447
> pt(0,23)          # T ~ t(23)          P(T <= 0)
[1] 0.5
```

Notice that

```
> qqnorm(trees$Girth); qqline(trees$Girth, col="red")
```

will produce a Normal Q–Q Plot for trees$Girth.

8 Tests and Confidence Intervals

8.1 One Sample proportion test

```
> binom.test(42,900,p=0.06,alt="less")          # Exact Binomal test
```

```
        Exact binomial test

data:  42 and 900
number of successes = 42, number of trials = 900, p-value = 0.0493
alternative hypothesis: true probability of success is less than 0.06
95 percent confidence interval:
 0.00000000 0.05994393
sample estimates:
probability of success
            0.04666667
```

8.2 One Sample z test: means

```
> install.packages("TeachingDemos")
> library(TeachingDemos)
> set.seed(1234)
> ntab=rnorm(99,5,1)

> z.test(ntab,mu=4.6,sd=1)
        One Sample z-test
data:  ntab
z = 2.1913, n = 99.000, Std. Dev. = 1.000, Std. Dev. of the sample mean
= 0.101, p-value = 0.02843
alternative hypothesis: true mean is not equal to 4.6
95 percent confidence interval:
 4.623246 5.017213
sample estimates:
mean of ntab
    4.820229

> z.test(ntab,mu=4.6,sd=1,alternative="greater")
        One Sample z-test
data:  ntab
z = 2.1913, n = 99.000, Std. Dev. = 1.000, Std. Dev. of the sample mean
= 0.101, p-value = 0.01422
alternative hypothesis: true mean is greater than 4.6
95 percent confidence interval:
 4.654915      Inf
sample estimates:
mean of ntab
    4.820229
```

8.3 One Sample t test: means

```
> dat1=rnorm(25,0.9,2.5)
> t.test(dat1,mu=0,alt="greater",conf.level=0.9)  # one sample t test

        One Sample t-test

data:  dat1
t = 2.1258, df = 24, p-value = 0.022
alternative hypothesis: true mean is greater than 0
```

```
90 percent confidence interval:
 0.4723136       Inf
sample estimates:
mean of x
 1.242669
```

8.4 Two Sample t test: means

```
> dat1=rnorm(25,0.9,2.5)
> dat2=rnorm(43,0.6,1.3)
> t.test(dat1,dat2,mu=0.0)                          # Welch Two Sample t test

        Welch Two Sample t-test

data:  dat1 and dat2
t = 1.2977, df = 29.947, p-value = 0.2043
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.4611103  2.0679355
sample estimates:
mean of x mean of y
1.2426692 0.4392566

> power.t.test(sig.level=0.05,power=0.95, delta=0.5)

     Two-sample t test power calculation

              n = 104.928
          delta = 0.5
             sd = 1
      sig.level = 0.05
          power = 0.95
    alternative = two.sided

 NOTE: n is number in *each* group
```

8.5 Matched Pair t test: means

```
> a = c(12.9, 13.5, 12.8, 15.6, 17.2, 19.2, 12.6, 15.3, 14.4, 11.3)
> b = c(12.0, 12.2, 11.2, 13.0, 15.0, 15.8, 12.2, 13.4, 12.9, 11.0)
> t.test(a,b, paired=TRUE, alt="greater")         # Matched Pair t test

        Paired t-test

data:  a and b
t = 5.2671, df = 9, p-value = 0.0002579
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 1.049675      Inf
sample estimates:
mean of the differences
                  1.61
```

8.6 F test: variances

```
> dat1=rnorm(25,0.9,2.5)
> dat2=rnorm(43,0.6,1.3)
> var.test(dat1,dat2,ratio=1.5)                   # F test on variances

        F test to compare two variances

data:  dat1 and dat2
F = 3.1832, num df = 24, denom df = 42, p-value = 0.0009988
alternative hypothesis: true ratio of variances is not equal to 1.5
95 percent confidence interval:
  2.400867 10.202515
sample estimates:
ratio of variances
           4.77476
```

9 Chi Squared Goodness of Fit

9.1 Goodness of Fit

```
> obs=c(173,220,256,274,357,732,488)            # observed
> pprob=c(1/11,1/11,1/11,1/11,2/11,3/11,2/11)   # expected probatility
> chisq.test(obs,p=pprob)

        Chi-squared test for given probabilities

data:  obs
X-squared = 53.5202, df = 6, p-value = 9.217e-10
> exp=chisq.test(obs,p=pprob)$expected
> exp
[1] 227.2727 227.2727 227.2727 227.2727 454.5455 681.8182 454.5455
```

```
> (obs-exp)^2/exp
[1] 12.9603273  0.2327273  3.6311273  9.6071273 20.9332545  3.6933818  2.4622545
```

9.2 Independence

```
> row1 = c(91,90,51)                    # or col1 = c(91,150,109)
> row2 = c(150,200,155)                 # and col2 = c(90,200,198)
> row3 = c(109,198,172)                 # and col3 = c(51,155,172)
> obs = rbind(row1,row2,row3)   # and obs = cbind(col1,col2,col3)
> obs
     [,1] [,2] [,3]
row1   91   90   51
row2  150  200  155
row3  109  198  172
> chisq.test(obs)

        Pearson's Chi-squared test

data:  obs
X-squared = 25.086, df = 4, p-value = 4.835e-05
> exp
          [,1]      [,2]      [,3]
row1  66.77632  93.10526  72.11842
row2 145.35362 202.66447 156.98191
row3 137.87007 192.23026 148.89967
> (obs-exp)^2/exp
          [,1]       [,2]       [,3]
row1 8.7873503 0.10356728 6.18410250
row2 0.1485265 0.03503041 0.02502173
row3 6.0454073 0.17317702 3.58379030
```

10 Correlation and Regression

10.1 Correlation

```
> plot(trees$Girth~trees$Height,main="girth vs height")

> cor(trees$Girth,trees$Height)
[1] 0.5192801

> cor.test(trees$Girth,trees$Height) # option: "greater"/"less"

        Pearson's product-moment correlation

data:  trees$Girth and trees$Height
t = 3.2722, df = 29, p-value = 0.002758
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.2021327 0.7378538
```

```
sample estimates:
      cor
0.5192801
```

10.2 Regression

```
> plot(trees$Girth~trees$Height,main="girth vs height")
> abline(lm(trees$Girth ~ trees$Height))

> g.lm=lm(Girth~Height,data=trees)
> coef(g.lm)
 (Intercept) trees$Height
  -6.1883945    0.2557471

> residuals(g.lm)
         1          2          3          4          5          6          7
-3.4139043 -1.8351687 -1.1236745 -1.7253986 -3.8271227 -4.2386170  0.3090842
         8          9         10         11         12         13         14
-1.9926400 -3.1713756 -1.7926400 -2.7156285 -1.8483871 -1.8483871  0.2418428
        15         16         17         18         19         20         21
-0.9926400  0.1631072 -2.6501112 -2.5058584  1.7303485  3.6205784  0.2401187
        22         23         24         25         26         27         28
-0.0713756  1.7631072  3.7746014  2.7958658  2.7728773  2.7171301  3.6286244
        29         30         31
 3.7286244  3.7286244  4.5383945

> plot(trees$Height, residuals(g.lm), xlab="tree height", ylab="residuals", main="residual plot")
> abline(0, 0)

> summary(g.lm)

Call:
lm(formula = trees$Girth ~ trees$Height)

Residuals:
     Min       1Q   Median       3Q      Max
-4.23862 -1.92051 -0.07138  2.74500  4.53839

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -6.18839    5.96020  -1.038  0.30772
trees$Height  0.25575    0.07816   3.272  0.00276 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 2.728 on 29 degrees of freedom
Multiple R-squared: 0.2697,        Adjusted R-squared: 0.2445
F-statistic: 10.71 on 1 and 29 DF,  p-value: 0.002758

> predict(g.lm,newdata=data.frame(Height=c(74,83,91)),interval="prediction",level=.90)
       fit       lwr      upr
1 12.73689  8.020516 17.45327
2 15.03862 10.238843 19.83839
3 17.08459 11.971691 22.19750
```

10.3 Multivariate Regression

```
> g.lm=lm(mpg~disp+hp+wt+qsec, data=mtcars)
> summary(g.lm)

Call:
lm(formula = mpg ~ disp + hp + wt + qsec, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8664 -1.5819 -0.3788  1.1712  5.6468

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 27.329638   8.639032   3.164  0.00383 **
disp         0.002666   0.010738   0.248  0.80576
hp          -0.018666   0.015613  -1.196  0.24227
wt          -4.609123   1.265851  -3.641  0.00113 **
qsec         0.544160   0.466493   1.166  0.25362
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 2.622 on 27 degrees of freedom
Multiple R-squared: 0.8351,     Adjusted R-squared: 0.8107
F-statistic: 34.19 on 4 and 27 DF,  p-value: 3.311e-10
```

11 ANOVA

11.1 One–Factor Design

```
> cdat=read.table("carpet.dat",h=TRUE)

> cdat
   Durability Carpet Composition
1       18.95      1           A
2       12.62      1           B
3       11.94      1           A
4       14.42      1           B
5       10.06      2           A
6        7.19      2           B
7        7.03      2           A
8       14.66      2           B
9       10.92      3           A
10      13.28      3           B
11      14.52      3           A
12      12.51      3           B
13      10.46      4           A
14      21.40      4           B
15      18.10      4           A
16      22.50      4           B

> Carpet.F=as.factor(cdat$Carpet) # make variable categorical
```

```
> g.lm=lm(cdat$Durability~Carpet.F)

> anova(g.lm)
Analysis of Variance Table

Response: cdat$Durability
          Df Sum Sq Mean Sq F value  Pr(>F)
Carpet.F   3 146.37  48.791  3.5815 0.04674 *
Residuals 12 163.48  13.623
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

> kruskal.test(cdat$Durability~Carpet.F) # Kruskal--Wallis Test

        Kruskal-Wallis rank sum test

data:  cdat$Durability by Carpet.F
Kruskal-Wallis chi-squared = 5.2059, df = 3, p-value = 0.1573
```

11.2 Two–Factor Additive Design

```
> cdat=read.table("carpet.dat",h=TRUE)
> Carpet.F=as.factor(cdat$Carpet)  # make variable categorical
> Composition.F=as.factor(cdat$Composition)  # make variable categorical
> g.lm=lm(cdat$Durability~Carpet.F+Composition.F)
> anova(g.lm)
Analysis of Variance Table

Response: cdat$Durability
              Df  Sum Sq Mean Sq F value  Pr(>F)
Carpet.F       3 146.374  48.791  3.6697 0.04719 *
Composition.F  1  17.222  17.222  1.2953 0.27925
Residuals     11 146.254  13.296
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

11.3 Two–Factor Design

```
> cdat=read.table("carpet.dat",h=TRUE)
> Carpet.F=as.factor(cdat$Carpet)  # make variable categorical
> Composition.F=as.factor(cdat$Composition)  # make variable categorical
> g.lm=lm(cdat$Durability~Carpet.F+Composition.F+Carpet.F:Composition.F)
> anova(g.lm)
Analysis of Variance Table

Response: cdat$Durability
                      Df  Sum Sq Mean Sq F value  Pr(>F)
Carpet.F               3 146.374  48.791  4.0981 0.04912 *
Composition.F          1  17.222  17.222  1.4466 0.26347
Carpet.F:Composition.F 3  51.007  17.002  1.4281 0.30462
Residuals              8  95.247  11.906
```

```
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```