STAT:5400 (22S:166) Computing in Statistics

Introduction to R

Lecture 5
September 6, 2017

Kate Cowles
374 SH, 335-0727
kate-cowles@uiowa.edu

What R is

- "an integrated suite of software facilities for data manipulation, calculation, and graphics display" (*An Introduction to R*, Venables, Ripley, and the R Core team)
  - data handling and storage capabilities
  - operators for calculations on arrays and matrices
  - data analysis tools
  - graphical capabilities
  - programming language
  - planned and coherent system

- an implementation of S language
  - S language was developed at AT&T-Bell Labs
    - first version 1976
  - S-Plus is a commercial version of S (begin in 1987)
    - sold and supported by Insightful Corp.
    - GUI
    - many formats supported for graphics export and data input/output
    - runs on Windows, UNIX, Linux (not Macintosh)

- advantages of S
  - extendible
    - users write new functions in S language — just as developers do
    - excellent documentation for adding functions to system
    - users can create their own data types
    - huge international community of users constantly contribute new capabilities
    - contrast with SAS
      - very hard to write new SAS procedures
      - users write in different language (SAS macro or IML) than developers
  - high-level language
    - only a few commands required to do complex things

– language is connected to data while executing
example (from *Statistical Computing and Graphics* course notes by Frank Harrell)

```
if(is.factor(x) | is.character(x) |
  (is.numeric(x) & length(unique(x)) < 20))
  table (x) else quantile(x)
```

computes quantiles of `x` if `x` is numeric and has at least 20 distinct values, requency table otherwise

– object-oriented
  * fewer commands to learn because the same command can be applied to different types of objects

– Harrell: "best scientific graphics available"
  * Harrell: "SAS graphics are ugly, inflexible, have poor defaults, difficult to program"

R

- international team of statisticians started developing R in early 1990's
  – to provide open source alternative to S-Plus
  – to provide S implementation on Linux (not supported by S-Plus then)
- easy to download and install from web sites
- excellent documentation
- user-contributed libraries called packages expand capabilities
- runs on Windows, UNIX, Linux, Macintosh
- no GUI on most platforms
- fewer data import/export capabilities than S-Plus
  – although add-on packages provide more
  – no export specifically to Powerpoint

Starting and running R interactively on Linux

- strongly recommended to use a separate subdirectory for each major R project. You might want one subdirectory for your homework assignments, and another for your group project.
- begin by creating the subdirectory
- copy in or download any needed data files
- then invoke R in that subdirectory

```
[kcowles@p-lnx402 ~]$ mkdir examples166
[kcowles@p-lnx402 ~]$ cd examples166
[kcowles@p-lnx402 ~/examples166]$ ls -a
.  ..
[kcowles@p-lnx402 ~/examples166]$  R
```


Running R interactively from within Emacs

- this method is convenient when you want to record your R commands and output in a document (such as a homework assignment)
- in your subdirectory, start Emacs as a background process

  `emacs &`

- once Emacs is up, start a second window by choosing File / New frame
- start R in one of the frames by typing

  `Alt-X R`

  You will be prompted to enter the directory in which you want R to start. Just press Enter to accept the default (if it's what you want).
- in the other window, use File /Visit new file and type in a file name when prompted

- now you can copy R commands and output from the R window into your emacs file

Reading in data from external files

- Use Firefox to download `Cars.dat` from "Datasets" section of course web page into this directory.

```
[kcowles@p-lnx402 ~/examples166]$ ls
Cars.dat
```

- Use a text editor to look at this file. Note that the separators between columns are tabs (You can tell because the cursor jumps) and that the decimal point in numbers is indicated by periods.
- We need to read this file into an `object` in R to analyze it. R has several functions that read in data files in different formats.
- We will use R's built-in help facility to figure out which one to use.


```
[kcowles@p-lnx402 ~/examples166]$ R

R version 2.7.1 (2008-06-23)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
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.
```



```
> help(read.delim)

read.table               package:utils               R Documentation

Data Input

Description:

     Reads a file in table format and creates a data frame from it,
     with cases corresponding to lines and variables to fields in the
     file.

Usage:

     read.table(file, header = FALSE, sep = "", quote = "\"'",
                dec = ".", row.names, col.names,
                as.is = !stringsAsFactors,
                na.strings = "NA", colClasses = NA, nrows = -1,
                skip = 0, check.names = TRUE, fill = !blank.lines.skip,
                strip.white = FALSE, blank.lines.skip = TRUE,
                comment.char = "#",
                allowEscapes = FALSE, flush = FALSE,
                stringsAsFactors = default.stringsAsFactors(),
                encoding = "unknown")
     read.csv(file, header = TRUE, sep = ",", quote="\"", dec=".",
              fill = TRUE, comment.char="", ...)

     read.csv2(file, header = TRUE, sep = ";", quote="\"", dec=",",
               fill = TRUE, comment.char="", ...)

     read.delim(file, header = TRUE, sep = "\t", quote="\"", dec=".",
                fill = TRUE, comment.char="", ...)

     read.delim2(file, header = TRUE, sep = "\t", quote="\"", dec=",",
                 fill = TRUE, comment.char="", ...)


      .......   lots of additional detail ........
```

```
> Cars <- read.delim("Cars.dat")      #  <- is assignment operator

> str(Cars)              # find out structure of the object
'data.frame':   38 obs. of  8 variables:
 $ Country     : Factor w/ 6 levels "France","Germany",..: 6 6 6 6 6 4 4 6 2 5 ...
 $ Car         : Factor w/ 38 levels "AMC Concord D/L",..: 6 21 11 12 8 34 14 18 3 35 ...
 $ MPG         : num  16.9 15.5 19.2 18.5 30 27.5 27.2 30.9 20.3 17 ...
 $ Weight      : num  4.36 4.05 3.60 3.94 2.15 ...
 $ Drive_Ratio : num  2.73 2.26 2.56 2.45 3.7 3.05 3.54 3.37 3.9 3.5 ...
 $ Horsepower  : int  155 142 125 150 68 95 97 75 103 125 ...
 $ Displacement: int  350 351 267 360 98 134 119 105 131 163 ...
 $ Cylinders   : int  8 8 8 8 4 4 4 4 5 6 ...
```


Data frames

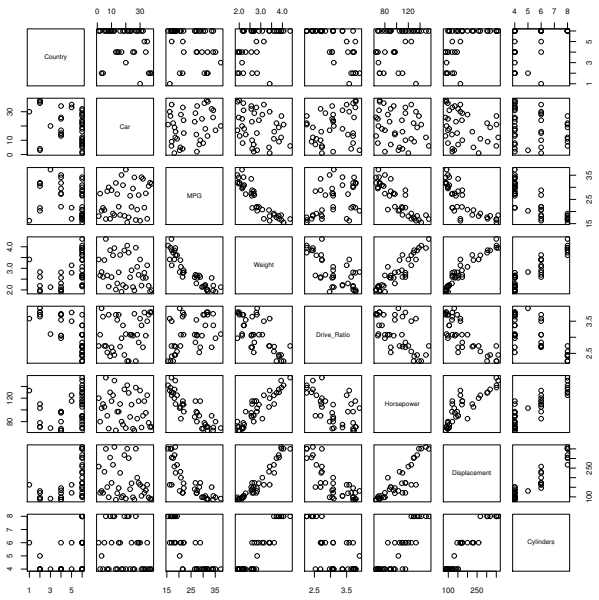
- special kind of object
- like a table of data
  - row for each observation
  - column for each variable
- variables (columns) can be of different data types
  - numeric, character, factors


```
> Cars
   Country                      Car  MPG Weight Drive_Ratio Horsepower
1     U.S.        Buick Estate Wagon 16.9  4.360        2.73        155
2     U.S. Ford Country Squire Wagon 15.5  4.054        2.26        142
3     U.S.        Chevy Malibu Wagon 19.2  3.605        2.56        125
4     U.S.    Chrysler LeBaron Wagon 18.5  3.940        2.45        150
5     U.S.                 Chevette 30.0  2.155        3.70         68
6    Japan             Toyota Corona 27.5  2.560        3.05         95
7    Japan                Datsun 510 27.2  2.300        3.54         97
8     U.S.                Dodge Omni 30.9  2.230        3.37         75
9  Germany                 Audi 5000 20.3  2.830        3.90        103
10  Sweden              Volvo 240 GL 17.0  3.140        3.50        125
11  Sweden               Saab 99 GLE 21.6  2.795        3.77        115
12  France            Peugeot 694 SL 16.2  3.410        3.58        133
....
   Displacement Cylinders
1           350         8
2           351         8
3           267         8
4           360         8
5            98         4
6           134         4
.......
```



```
>   summary(Cars)
    Country                      Car           MPG            Weight
 France : 1   AMC Concord D/L      : 1   Min.   :15.50   Min.   :1.915
 Germany: 5   AMC Spirit           : 1   1st Qu.:18.52   1st Qu.:2.208
 Italy  : 1   Audi 5000            : 1   Median :24.25   Median :2.685
 Japan  : 7   BMW 320i             : 1   Mean   :24.76   Mean   :2.863
 Sweden : 2   Buick Century Special: 1   3rd Qu.:30.38   3rd Qu.:3.410
 U.S.   :22   Buick Estate Wagon   : 1   Max.   :37.30   Max.   :4.360
              (Other)              :32
  Drive_Ratio      Horsepower     Displacement     Cylinders
 Min.   :2.260   Min.   : 65.0   Min.   : 85.0   Min.   :4.000
 1st Qu.:2.695   1st Qu.: 78.5   1st Qu.:105.0   1st Qu.:4.000
 Median :3.080   Median :100.0   Median :148.5   Median :4.500
 Mean   :3.093   Mean   :101.7   Mean   :177.3   Mean   :5.395
 3rd Qu.:3.625   3rd Qu.:123.8   3rd Qu.:229.5   3rd Qu.:6.000
 Max.   :3.900   Max.   :155.0   Max.   :360.0   Max.   :8.000
```

```
> plot(Cars)
```




```
> hist(Cars$MPG)     # can use $ to refer to column name in data frame
```


Subsetting rows and columns of a data frame

```
> Cars[1:10,]
   Country                     Car  MPG Weight Drive_Ratio Horsepower
1     U.S.         Buick Estate Wagon 16.9  4.360        2.73        155
2     U.S. Ford Country Squire Wagon 15.5  4.054        2.26        142
3     U.S.         Chevy Malibu Wagon 19.2  3.605        2.56        125
4     U.S.     Chrysler LeBaron Wagon 18.5  3.940        2.45        150
5     U.S.                 Chevette 30.0  2.155        3.70         68
6    Japan             Toyota Corona 27.5  2.560        3.05         95
7    Japan               Datsun 510 27.2  2.300        3.54         97
8     U.S.               Dodge Omni 30.9  2.230        3.37         75
9  Germany                Audi 5000 20.3  2.830        3.90        103
10  Sweden              Volvo 240 GL 17.0  3.140        3.50        125
   Displacement Cylinders
1           350         8
2           351         8
3           267         8
4           360         8
5            98         4
6           134         4
7           119         4
8           105         4
9           131         5
10          163         6
```



```
> Cars[, c("Country","MPG","Weight")]
   Country  MPG Weight
1     U.S. 16.9  4.360
2     U.S. 15.5  4.054
3     U.S. 19.2  3.605
4     U.S. 18.5  3.940
5     U.S. 30.0  2.155
6    Japan 27.5  2.560
7    Japan 27.2  2.300
8     U.S. 30.9  2.230
9  Germany 20.3  2.830
10  Sweden 17.0  3.140
11  Sweden 21.6  2.795
12  France 16.2  3.410
13    U.S. 20.6  3.380
14    U.S. 20.8  3.070
15    U.S. 18.6  3.620
16    U.S. 18.1  3.410
17    U.S. 17.0  3.840
18    U.S. 17.6  3.725
19    U.S. 16.5  3.955
20    U.S. 18.2  3.830
21    U.S. 26.5  2.585
22    U.S. 21.9  2.910
23   Japan 34.1  1.975
24   Japan 35.1  1.915
25    U.S. 27.4  2.670
26 Germany 31.5  1.990
27   Japan 29.5  2.135
28    U.S. 28.4  2.670
29    U.S. 28.8  2.595
30    U.S. 26.8  2.700
31    U.S. 33.5  2.556
32    U.S. 34.2  2.200
33   Japan 31.8  2.020
34   Italy 37.3  2.130
35 Germany 30.5  2.190
36   Japan 22.0  2.815
37 Germany 21.5  2.600
38 Germany 31.9  1.925
```



```
> Cars[ Cars$Country=="U.S.",]
   Country                     Car  MPG Weight Drive_Ratio Horsepower
1     U.S.         Buick Estate Wagon 16.9  4.360        2.73        155
2     U.S. Ford Country Squire Wagon 15.5  4.054        2.26        142
3     U.S.         Chevy Malibu Wagon 19.2  3.605        2.56        125
4     U.S.     Chrysler LeBaron Wagon 18.5  3.940        2.45        150
5     U.S.                 Chevette 30.0  2.155        3.70         68
8     U.S.               Dodge Omni 30.9  2.230        3.37         75
13    U.S.      Buick Century Special 20.6  3.380        2.73        105
14    U.S.             Mercury Zephyr 20.8  3.070        3.08         85
15    U.S.                Dodge Aspen 18.6  3.620        2.71        110
16    U.S.            AMC Concord D/L 18.1  3.410        2.73        120
17    U.S.      Chevy Caprice Classic 17.0  3.840        2.41        130
18    U.S.                  Ford LTD 17.6  3.725        2.26        129
19    U.S.      Mercury Grand Marquis 16.5  3.955        2.26        138
20    U.S.             Dodge St Regis 18.2  3.830        2.45        135
21    U.S.             Ford Mustang 4 26.5  2.585        3.08         88
22    U.S.          Ford Mustang Ghia 21.9  2.910        3.08        109
25    U.S.                AMC Spirit 27.4  2.670        3.08         80
28    U.S.              Buick Skylark 28.4  2.670        2.53         90
29    U.S.             Chevy Citation 28.8  2.595        2.69        115
30    U.S.                 Olds Omega 26.8  2.700        2.84        115
31    U.S.            Pontiac Phoenix 33.5  2.556        2.69         90
32    U.S.           Plymouth Horizon 34.2  2.200        3.37         70
   Displacement Cylinders
1           350         8
2           351         8
3           267         8
4           360         8
5            98         4
8           105         4
13          231         6
14          200         6
15          225         6
16          258         6
17          305         8
18          302         8
19          351         8
20          318         8
21          140         4
22          171         6
```

21

```
25          121         4
28          151         4
29          173         6
30          173         6
31          151         4
32          105         4
```

22

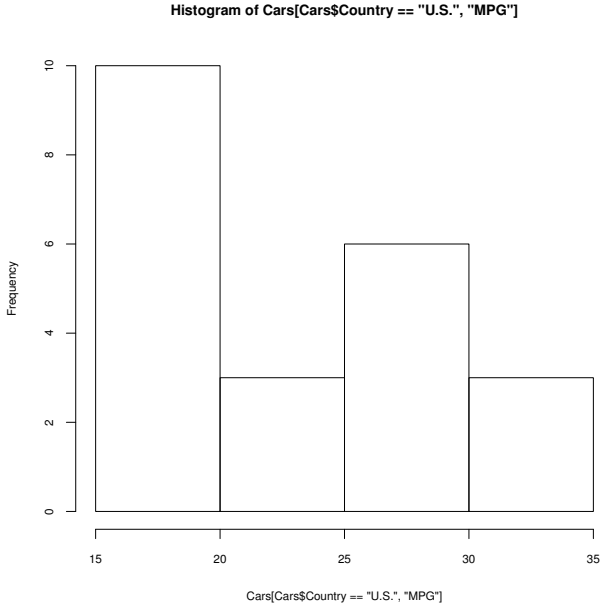
```
> Cars[ Cars$Country=="Japan",]
   Country              Car  MPG Weight Drive_Ratio Horsepower Displacement
6    Japan   Toyota Corona 27.5  2.560        3.05         95          134
7    Japan      Datsun 510 27.2  2.300        3.54         97          119
23   Japan       Mazda GLC 34.1  1.975        3.73         65           86
24   Japan      Dodge Colt 35.1  1.915        2.97         80           98
27   Japan Honda Accord LX 29.5  2.135        3.05         68           98
33   Japan      Datsun 210 31.8  2.020        3.70         65           85
36   Japan      Datsun 810 22.0  2.815        3.70         97          146
   Cylinders
6          4
7          4
23         4
24         4
27         4
33         4
36         6
```

23

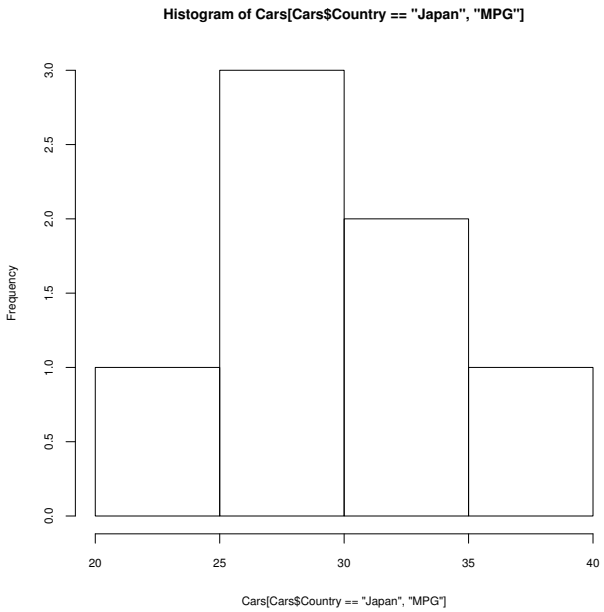
```
> summary(Cars[ Cars$Country=="U.S.","MPG"])
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.50   18.12   20.70   23.00   28.15   34.20

> summary(Cars[ Cars$Country=="Japan","MPG"])
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  22.00   27.35   29.50   29.60   32.95   35.10
```

24

```
> hist(Cars[ Cars$Country=="U.S.","MPG"])
```




```
> hist(Cars[ Cars$Country=="Japan","MPG"])
```




```
> help(t.test)

t.test                  package:stats                  R Documentation

Student's t-Test

Description:

     Performs one and two sample t-tests on vectors of data.

Usage:

     t.test(x, ...)

     ## Default S3 method:
     t.test(x, y = NULL,
            alternative = c("two.sided", "less", "greater"),
            mu = 0, paired = FALSE, var.equal = FALSE,
            conf.level = 0.95, ...)

     ## S3 method for class 'formula':
     t.test(formula, data, subset, na.action, ...)

Arguments:

       x: a (non-empty) numeric vector of data values.

       y: an optional (non-empty) numeric vector of data values.

alternative: a character string specifying the alternative hypothesis,
           must be one of '"two.sided"' (default), '"greater"' or
           '"less"'.  You can specify just the initial letter.

      mu: a number indicating the true value of the mean (or difference
           in means if you are performing a two sample test).

  paired: a logical indicating whether you want a paired t-test.

var.equal: a logical variable indicating whether to treat the two
           variances as being equal. If 'TRUE' then the pooled variance
           is used to estimate the variance otherwise the Welch (or
```



```
           Satterthwaite) approximation to the degrees of freedom is
           used.

conf.level: confidence level of the interval.

Value:

     A list with class '"htest"' containing the following components:

statistic: the value of the t-statistic.

parameter: the degrees of freedom for the t-statistic.

 p.value: the p-value for the test.

conf.int: a confidence interval for the mean appropriate to the
           specified alternative hypothesis.

estimate: the estimated mean or difference in means depending on
           whether it was a one-sample test or a two-sample test.

null.value: the specified hypothesized value of the mean or mean
           difference depending on whether it was a one-sample test or a
           two-sample test.

alternative: a character string describing the alternative hypothesis.

  method: a character string indicating what type of t-test was
           performed.

data.name: a character string giving the name(s) of the data.
```



```
> t.test( Cars[ Cars$Country=="U.S.","MPG"])
        One Sample t-test

data:  Cars[Cars$Country == "U.S.", "MPG"]
t = 17.8153, df = 21, p-value = 3.739e-14
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 20.31116 25.67975
sample estimates:
mean of x
 22.99545

> t.test( Cars[ Cars$Country=="U.S.","MPG"])
        One Sample t-test

data:  Cars[Cars$Country == "U.S.", "MPG"]
t = 17.8153, df = 21, p-value = 3.739e-14
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 20.31116 25.67975
sample estimates:
mean of x
 22.99545
```

```
> UStout <- t.test( Cars[ Cars$Country=="U.S.","MPG"])

> names(UStout)
[1] "statistic"   "parameter"   "p.value"     "conf.int"    "estimate"
[6] "null.value"  "alternative" "method"      "data.name"

> UStout$conf.int
[1] 20.31116 25.67975
attr(,"conf.level")
[1] 0.95

> UStout$estimate
mean of x
 22.99545
> UStout$statistic
       t
17.81533
```



```
> Japantout

        One Sample t-test

data:  Cars[Cars$Country == "Japan", "MPG"]
t = 17.2771, df = 6, p-value = 2.409e-06
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 25.40782 33.79218
sample estimates:
mean of x
     29.6

> t.test( Cars[ Cars$Country=="U.S.","MPG"], Cars[ Cars$Country=="Japan","MPG"])

        Welch Two Sample t-test

data:  Cars[Cars$Country == "U.S.", "MPG"] and Cars[Cars$Country == "Japan", "MPG"]
t = -3.0789, df = 13.502, p-value = 0.008466
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -11.221244  -1.987847
sample estimates:
mean of x mean of y
 22.99545  29.60000
```


Reading directly from web into dataframes

You can use the URL of a dataset from the web as the argument to a `read.table` or `read.delim` function.

```
read.delim("http://www.stat.uiowa.edu/~kcowles/Datasets/Cars.dat")
```