Reading in other sources of data in R

library() or require() functions

Loading and Listing of Packages

library()

list all the installed packages

library(package name) or require(package name)

Loading the specified package.

Example:

```
library("UsingR")
```

data()

Loads specified data sets, or list the available data sets.

data()

List all available data sets in loaded package.

data(name of data set)

load the specified data eset

Example:

```
data(survey,package="MASS")
# will not load help files for data set or the rest of package
library("MASS")
data(survey)   #better
```

Accessing the variables in a data set:

attach() and with():

Attach Set of R Objects vto Search Path.

Important note:

cannot change variable values in attached dataset

Example: (continued)

```
summary(women$height)   # refers to variable 'height' in the data frame (data set women)
attach(women)
summary(height)         # The same variable now available by name
detach()

with(data.frame,command)  # attach & detach
```

example in p26

```
names(Sitka)
Sitka
data(Sitka)
library(MASS)
data(Sitka)
names(Sitka)
length(tree)
length(Sitka$tree)
Sitka$size[tree>78]
Error: object "tree" not found
Sitka$size[Sitka$tree>78]
[1] 2.99 3.61 4.48 4.91 5.06
with(Sitka,list(a=range(tree),b=table(treat),c=max(Time)))
attach(Sitka)
Sitka$size[tree>78]
[1] 2.99 3.61 4.48 4.91 5.06
summary(Sitka)
detach(Sitka)
tree
attach(Sitka)
detach(Sitka)
```

easy example to create dataframe

```
weight = c(150, 135, 210, 140)
height = c(65, 61, 70, 65)
 gender = c("Fe","Fe","M","Fe")
 study = data.frame(weight,height,gender) # make the data frame
 study
row.names(study)<-c("Mary","Alice","Bob","Judy")
study
rm(weight) # clean out an old copy
weight
Error: Object "weight" not found
attach(study)
weight
```

UsingR package

Write these commands in R

>where="http://www.math.csi.cuny.edu/UsingR"
>install.packages("UsingR",contriburl=where)

OR

> install.packages("UsingR")
--- Please select a CRAN mirror for use in this session ---
trying URL 'http://cran.wustl.edu/bin/windows/contrib/2.3/UsingR_0.1-4.zip'
Content type 'application/zip' length 1419692 bytes
opened URL
downloaded 1386Kb

package 'UsingR' successfully unpacked and MD5 sums checked

The downloaded packages are in
C:\Documents and Settings\ÃæíÓ\Local Settings\Temp\RtmpQCZ8ub\downloaded_packages
updating HTML package descriptions

Import data into R:

- **Cut or copy and paste.**
- **Scan()**

Read data into a vector or list from the console or file.

Example:

```
> w=scan()
1: 3 4 6 7 8 7 9 9
9:                  # press enter (blank line)
Read 8 items
```

- **dump()**

The function dumb() can be used to write values of R object to a text file. This function takes a vector of names of R objects and produces text representations of the objects on a file or connection. A 'dump' file can usually be 'source'd into another R (or S) session.

Example:

```
dump("x","filename.txt") # or can write a vector of objects in one file
dump("w","infile.txt")
```

- **source**

Read R Code (commands) from a File or a Connection 'source' causes R to accept its input from the named file or URL (the name must be quoted) or connection. Input is read and 'parse'd by from that file until the end of the file is reached, then the parsed expressions are evaluated sequentially in the chosen environment.

Examples:

```
Source("infile.txt")
```

Examples:

```
whales=scan()
1: 74 122 235 111 292 111 211 133 156 79
11:
Read 10 items
dump("whales","f1.txt")
```

In a new session of R

```
source("f1.txt")  # open  file name f1.txt and get the data from it or any R commands stored
whales
```

reading data from formatted data source:

If we have a data in txt file with spaces between them, we can read them by

```
scan(file="f2.txt")
scan(file="f2.txt",sep=",")  #if comma between values

read.table("filename",header=T)  #read data frame or tables with column name

read.cvs()  # cvs files

read.table(file=file.choose())    # choose the file u need to read with out writing its name
```

Read file from anywhere see p29

```
read.table(file=site,header=T)
```

Function `read.spss` can read files created by the `save' and `export' commands in SPSS. It returns a list with one component for each variable in the saved data set. SPSS variables with value labels are optionally converted to R factors.

subset:

Returns subsets of vectors or data frames that meet specific requirements

Example:

```
library(MASS)
 data(Cars93)
 attach(Cars93)    # unnecessary in this case
 Vans <- subset(Cars93,Type=="Van")
 detach(Cars93)
Vans <- subset(Cars93,Type=="Van")
```

transform:

Transforms elements of an object

Example:

```
Cars93T <- transform(Cars93,WeightT=Weight/1000)
names(Cars93)
names(Cars93T)
```

Grouped data and data frames

Example:

```
attach(mtcars)
?mtcars
mtcars$mpg[mtcars$cyl==4]
 # same as
mtcars$mpg[cyl==4]
Another way
split(mtcars$mpg,mtcars$cyl)
```

Order() Function:

If you want to sort more than one variable, it is best to use the order function. The following is a short example of this.

Example:

```
> exame1 = c(16,18,12,15,17,14,15,13,15)
> exame2 = c(19,18,15,17,15,17,16,14,18)
> scores = data.frame(exame1,exame2)
> scores
```

the items are not sorted. Values in a single vector may be sorted directly using the sort function, as in

```
> sort ( scores[ ,1])
[1] 12 13 14 15 15 15 16 17 18

> ord = order(scores[,1],scores[,2])
> ord
[1] 3 8 6 7 4 9 1 5 2
> ordered.scores = scores[ord,]
> ordered.scores
```

	exame1	exame2
3	12	15
8	13	14
6	14	17
7	15	16
4	15	17
9	15	18
1	16	19
5	17	15
2	18	18

Sample function:

Random Samples and Permutations

sample(x, size, replace = FALSE, prob = NULL)

Arguments:

x: Either a (numeric, complex, character or logical) vector of more than one element from which to choose, or a positive integer.

size: non-negative integer giving the number of items to choose.

replace: Should sampling be with replacement?

prob: A vector of probability weights for obtaining the elements of the vector being sampled.

Examples:

```
sample(1:30,10,F)  # without replacement

sample(1:30,10,T)

sample(1:30,31,F)
Error in sample(length(x), size, replace, prob) :
        can't take a sample larger than the population
 when replace = FALSE

sample(10,5)

sample(10)

sample(1e6,40)   #  sample of 40 from 1,000,000

sample(c(1,3,7,9),10,T,prob=c(0.3,0.2,0.1,0.5))

sample(0:1,100,T,c(0.3,0.7))  #  Binomial(100,0.7)
sample(0:1,1,T,c(0.3,0.7))  # Bernolli(0.7)
```