Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

MCS 507 Lecture 35
Mathematical, Statistical and Scientific Software
Jan Verschelde, 10 November 2023

Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

R

R is a language and environment for statistical computing and graphics, released under the GNU General Public License.

R is an implementation of the S language developed at Bell Labs by Rick Becker, John Chambers, and Allan Wilks.
R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `https://www.R-project.org/`.

We run `R` explicitly in SageMath via

1. the class `R`, do `help(r);` or
2. opening a Terminal Session with R,
   type `r_console();` in a SageMath terminal.

`rpy2` is a Python interface to R, developed by Laurent Gautier.

no multiprecision

R is NOT a computer algebra package:

```
> choose(5, 2)
[1] 10
> choose(52, 5)
[1] 2598960
> choose(100, 30)
[1] 2.937234e+25
```

Instead of displaying an exact integer of 25 decimal places, the outcome of `choose` switched to a float.

Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

estimating π

We estimate $\int_0^1 \sqrt{1-x^2}dx = \frac{\pi}{4}$ using 1,000,000 samples:

```
> u <- runif(1000000, min=0, max=1)
> 4*mean(sqrt(1-u^2))
[1] 3.141879
```

To check, we can use `integrate`:

```
> integrand <- function(x) { sqrt(1-x^2) }
> integrate(integrand, lower=0, upper=1)
0.7853983 with absolute error < 0.00011
> pi/4
[1] 0.7853982
```

Monte Carlo integration

We estimate $\int_a^b f(x)dx$ by $(b-a)\frac{1}{n}\sum_{x\in[a,b]} f(x)$.

Applied to $\int_0^{\pi/2} 4\sin(2x)e^{-x^2}dx$:

```
> u <- runif(1000000, min=0, max=pi/2)
> pi/2*mean(4*sin(2*u)*exp(-u^2))
[1] 2.191348
```

Checking:

```
> integrand <- function(x) { 4*sin(2*x)*exp(-x^2) }
> integrate(integrand, lower=0, upper=pi/2)
2.190752 with absolute error < 2.4e-14
```

Statistical Computing with R

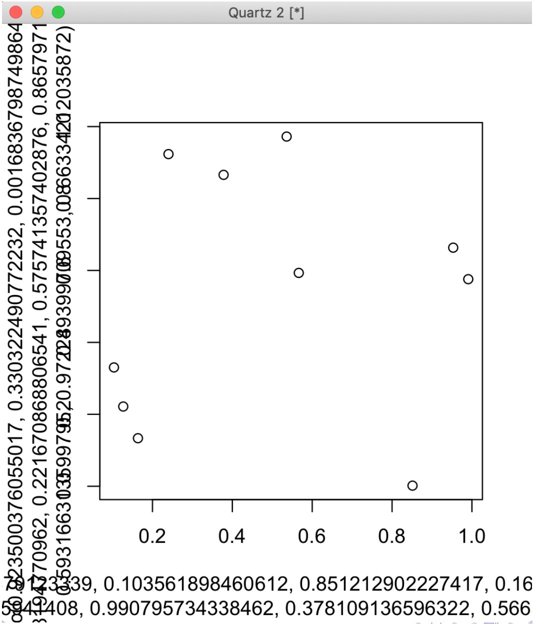
1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

R.plot() with ipython —pylab

After generating 10 points with coordinates $x, y \in [0, 1]$,
uniformly distributed, we make a plot:

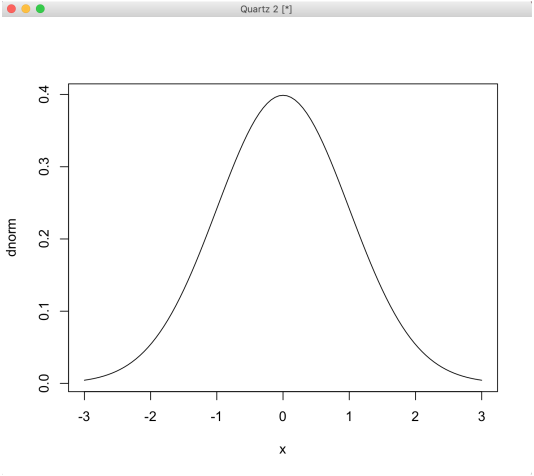
```
$ ipython --pylab
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
Type 'copyright', 'credits' or 'license' for more informati
IPython 7.8.0 -- An enhanced Interactive Python. Type '?' f
Using matplotlib backend: MacOSX

In [1]: from rpy2.robjects import r as R

In [2]: x = R("runif(10)")

In [3]: y = R("runif(10)")

In [4]: R.plot(x, y)
Out[4]: rpy2.rinterface.NULL
```

the plot



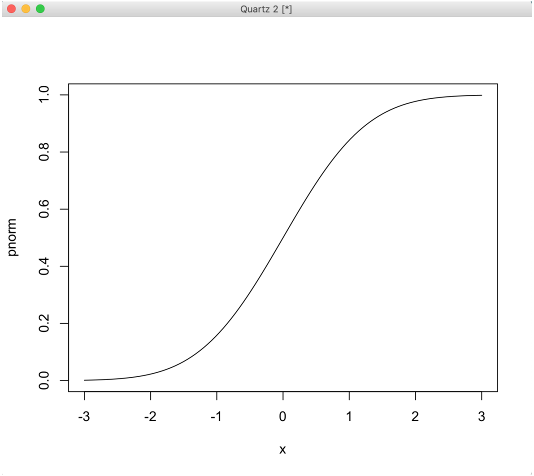
the plot of the normal density function

At the R prompt we type `plot(dnorm, -3, 3)` and see:



the plot of the normal distribution function

At the R prompt we type `plot(pnorm, -3, 3)` and see:

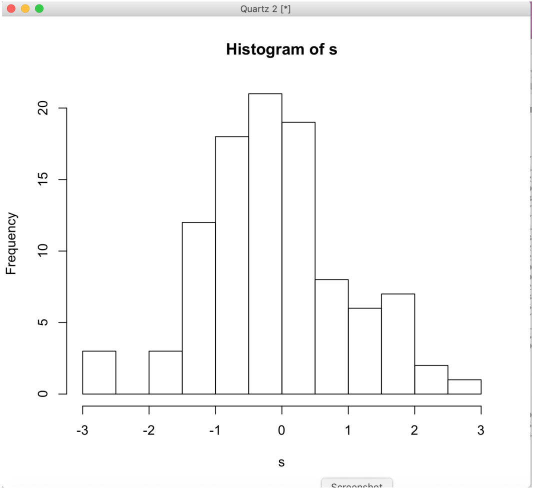


Statistical Computing with R

sample, sort, stem and leaf plot

```
> s = rnorm(100)
> sort(s)
  [1] -2.969379337 -2.784188808 -2.659379575 -1.734289431 -1.646096620
 ...
 [96]  1.884726237  1.918530401  2.059068927  2.081241693  2.524788866
> stem(s)

  The decimal point is at the |

  -3 | 0
  -2 | 87
  -1 | 766543333221100 0
  -0 | 9998888888877765555554444433333332 1110
   0 | 00011122223333344556666 7788
   1 | 112244556 8999
   2 | 115
```

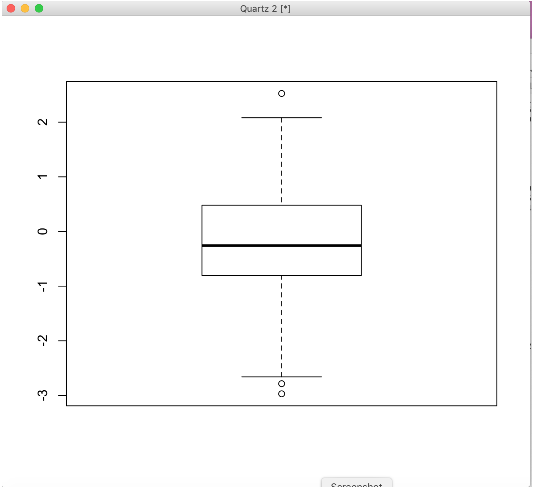
the histogram plot with `hist(s)`



the histogram data

```
> h = hist(s)
> h
$breaks
 [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0

$counts
 [1]  3  0  3 12 18 21 19  8  6  7  2  1

$density
 [1] 0.06 0.00 0.06 0.24 0.36 0.42 0.38 0.16 0.12 0.14 0.04 0.02

$mids
 [1] -2.75 -2.25 -1.75 -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25  2.75

$xname
[1] "s"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"
```

the boxplot with `boxplot(s)`



the boxplot data

```
> b = boxplot(s)
> b
$stats
           [,1]
[1,] -2.6593796
[2,] -0.8053552
[3,] -0.2579011
[4,]  0.4808079
[5,]  2.0812417

$n
[1] 100

$conf
            [,1]
[1,] -0.46111487
[2,] -0.05468734

$out
[1]  2.524789 -2.969379 -2.784189

$group
[1] 1 1 1

$names
[1] "1"
```

Statistical Computing with R

calling `lm`

For x we generate 50 points in $[0, 10]$
and y equals x plus some random noise:

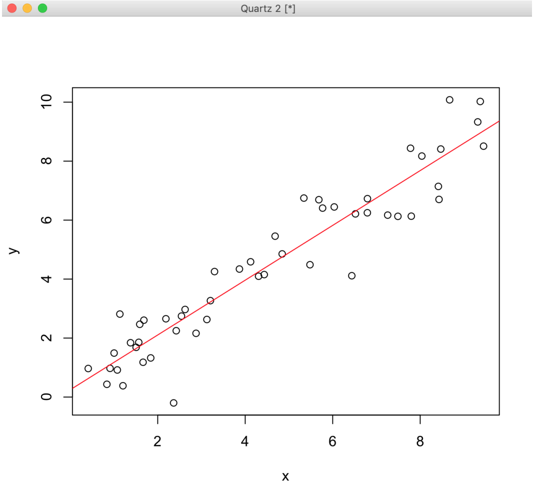
```
> x <- 10*runif(50)
> y <- x + rnorm(50)
> gx <- lm(y ~ x)
```

`lm` is used to fit linear models,
the method applied is QR decomposition.

```
> gx$coefficients
(Intercept)           x
  0.2457075   0.9292204
```

plotting the data

```
> plot(x, y)
> abline(a=coef(gx)[1], b = coef(gx)[2], col="red")
```



summary statistics of the linear fit

```
> summary(gx)

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-2.6415 -0.5732  0.1214  0.5072  1.7743

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.24571    0.24052   1.022    0.312
x            0.92922    0.04556  20.395   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.899 on 48 degrees of freedom
Multiple R-squared:  0.8965,Adjusted R-squared:  0.8944
F-statistic:   416 on 1 and 48 DF,  p-value: < 2.2e-16
```

Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

the package datasets

Typing `data()` at the `R` prompt
shows a list of data sets in the package `'datasets'`.

The last one in the list is `women`, which contains
`"Average Heights and Weights for American Women"`

The output of `help(women)` shows that the data set appears to have been taken from the American Society of Actuaries for some unknown year earlier than 1975.

printing the entire data set

```
> print(women)
   height weight
1      58    115
2      59    117
3      60    120
4      61    123
5      62    126
6      63    129
7      64    132
8      65    135
9      66    139
10     67    142
11     68    146
12     69    150
13     70    154
14     71    159
15     72    164
```

a summary of the data

```
> summary(women)
     height          weight     
 Min.   :58.0   Min.   :115.0  
 1st Qu.:61.5   1st Qu.:124.5  
 Median :65.0   Median :135.0  
 Mean   :65.0   Mean   :136.7  
 3rd Qu.:68.5   3rd Qu.:148.0  
 Max.   :72.0   Max.   :164.0  
```

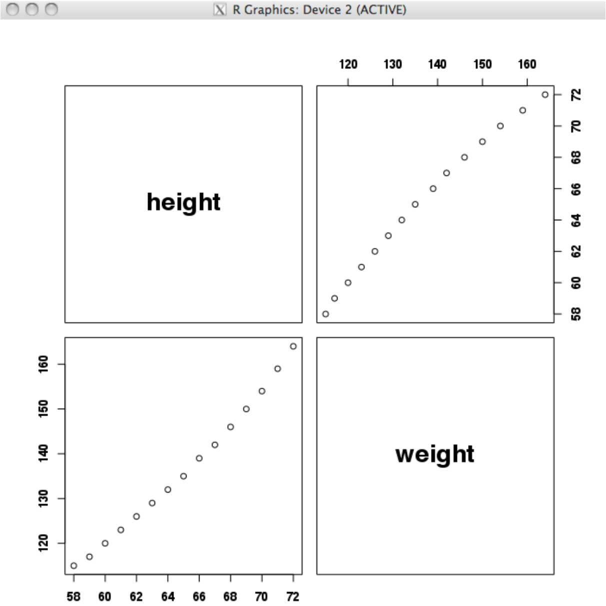
The correlation matrix:

```
> cor(women)
          height    weight
height 1.0000000 0.9954948
weight 0.9954948 1.0000000
```

To make a scatterplot:

```
> pairs
```

a scatterplot



Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

data about tornadoes

NOAA = National Oceanic and Atmospheric Administration
NOAA's National Weather Service provides free data.

Downloaded from `http://www.spc.noaa.gov/wcm/#data`
the file `2011_torn.csv` is a comma separated file with data about tornadoes in 2011.

The data file comes without a header in its first row.
Information about the data in the columns is provided in a separate .pdf file.

The file `2011_torn.csv` with an extra header row is renamed into `tornadoes.csv`.

reading data into R

```
> torn <- read.csv("tornadoes.csv")
> attributes(torn)
$names
 [1] "number"            "year"               "month"
 [4] "day"               "date"               "time"
 [7] "timezone"          "state"              "statefips"
[10] "statenum"          "scale"              "injuries"
[13] "fatalities"        "loss"               "croploss"
[16] "startinglatitude"  "startinglongitude"  "endinglatitude"
[19] "endinglongitude"   "length"             "width"
[22] "statesaffected"    "statenum.1"         "tornsegment"
[25] "county1fips"       "county2fips"        "county3fips"
[28] "county4fips"
```

These `names` are the headers for the 28 columns.
There are 1778 tornadoes recorded on file.

selecting data

```
> torn[,8]
```
shows all states in column 8

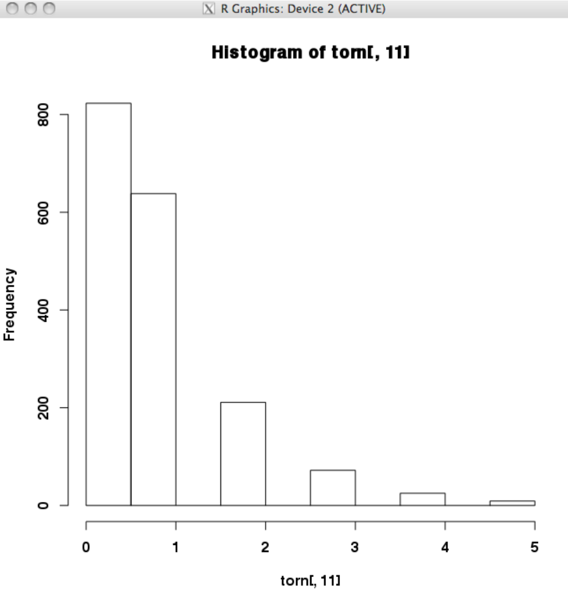
```
> torn[8,]
```
shows row 8.

```
> torn[8,8]
```
shows the state: row 8 and column 8.

To select specific rows and columns:

```
> torn[c(5,20),c(3,8,11)]
   month state scale
5      1    MS     2
20     2    HI     0
> hist(torn[,11])
```

a histogram of the scales of tornadoes



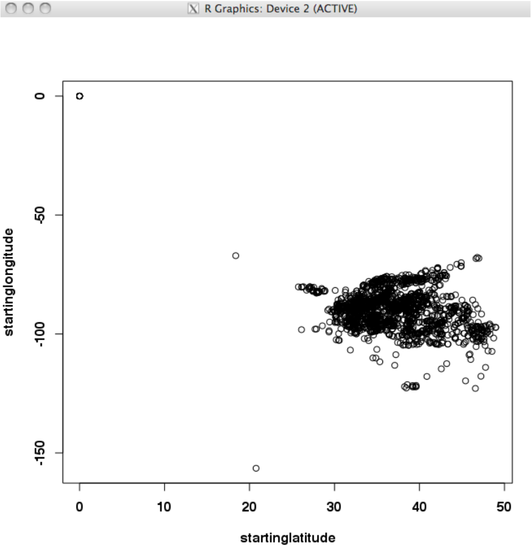
correlating scale, injuries, fatalities, and loss

```
> cor(torn[,c(11,12,13,14)])
               scale  injuries fatalities      loss
scale      1.0000000 0.2433426  0.3104764 0.2249695
injuries   0.2433426 1.0000000  0.7678119 0.8700983
fatalities 0.3104764 0.7678119  1.0000000 0.9204848
loss       0.2249695 0.8700983  0.9204848 1.0000000
```

Plotting starting latitude and longitudes:

```
> plot(torn[,c(16,17)])
```

starting latitudes and longitudes



Statistical Computing with R

1. The Language R
   - the language and environment R
   - Monte Carlo integration
   - making plots with R and ipython —pylab
2. Statistical Computing with R
   - histograms of data
   - fitting linear models
3. Data Sets
   - exploring available data sets
   - looking at a downloaded data set
4. Big Data
   - R and Hadoop

Large Data Files

Process big data with parallel and distributed computing:

- MySQL scales well.
- overview of Hadoop:

Hadoop has become the de facto standard for processing big data.

Hadoop is used by Facebook to store photos, by LinkedIn to generate recommendations, and by Amazon to generate search indices.

Hadoop works by connecting many different computers,
hiding the complexity from the user: work with one giant computer.

Hadoop uses a model called Map/Reduce.

RHadoop by Antonio Piccolboni is a project for R and Hadoop,
hosted at `https://github.com`.

the Map/Reduce model

Goal: help with writing efficient parallel programs.

Common data processing tasks: filtering, merging, aggregating data and many (not all) machine learning algorithms fit into Map/Reduce.

Two steps:

Map: Tasks read in input data as a set of records, process the records, and send groups of similar records to reducers. The mapper extracts a key from each input record. Hadoop will then route all records with the same key to the same reducer.

Reduce: Tasks read in a set of related records, process the records, and write the results. The reducer iterates through all results for the same key, processing the data and writing out the results.

Strength: the map and reduce steps run well in parallel.

example: predicting user behavior

Problem: how likely is a user to purchase an item from a website?
Suppose we have already computed (maybe using Map/Reduce) a set of variables describing each user:

- most common locations,
- the number of pages viewed,
- the number of purchases made in the past.

Calculate forecast using random forests:

- Random forests work by calculating a set of regression trees and then averaging them together to create a single model.
- It can be time consuming to fit the random trees to the data, but each new tree can be calculated independently.
- One way to tackle this problem is to use a set of map tasks to generate random trees, and then send the models to a single reducer task to average the results and produce the model.

Summary + Exercises

Available at `www.it-ebooks.info`:
Joseph Adler: *R in a Nutshell.* 2nd edition, O'Reilly, 2012.
Richard Cotton: *Learning R. A Step-by-Step Function Guide to Data Analysis.* O'Reilly, 2013.

1. Write a wrapper around `r.plot` which takes two arguments: a string with the argument for `r.plot` and the name of the file for the plot.
2. Annual U.S. Killer Tornado Statistics are listed at `http://www.spc.noaa.gov/climo/torn/fataltorn.html` Use python to get the annual data from the web page and into R data sets, making one data set for every year.