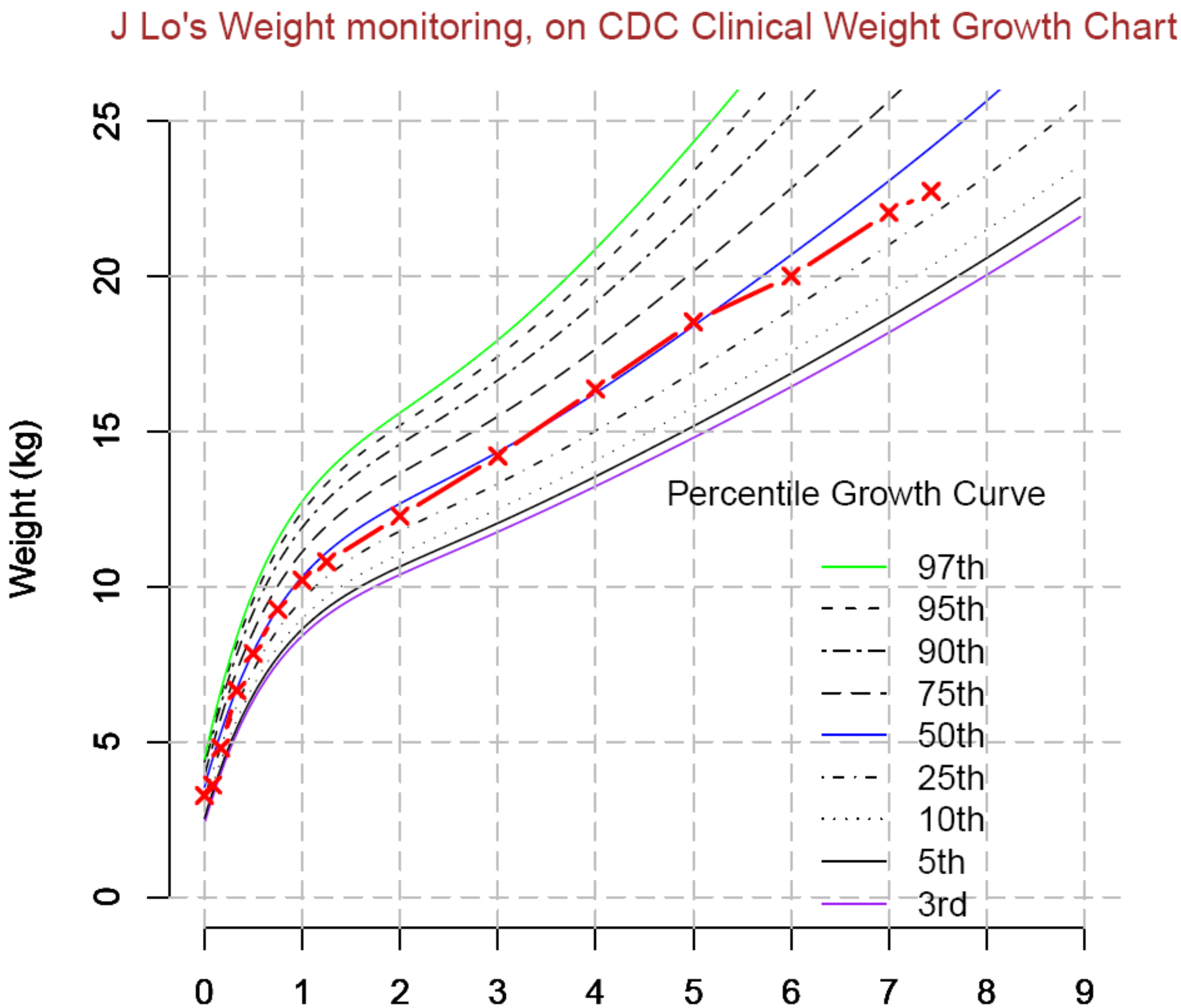
Genomics data analysis with R

Introduction to R language and graphs

教师： 李程（北大生命科学学院、统计科学中心）
网页：http://www.chenglilab.net/ （教学课程）
邮件：lch3000@gmail.com

Florence Yong's son's weight



Yered Pita-Juarez's 3D image

Today

- Introduction to R
  - syntax
  - flow control
  - descriptive statistics and graphics
  - probability functions

- Benefits of using R
  - many existing statistical functions
  - graphs
  - vector and matrix computing

R Language Essentials

- Basic mode is expression evaluation
  - Evaluation results are printed
  - Make plots or writing out files
  - All expression returns value (possibly NULL)
- Typically involves:
  - Variable references
  - Operators such as +
  - Function calls

Functions and arguments

- Many things in R are done using **function calls**
  - E.g. log(x)
  - A function name followed by parameters in ( )
  - Actual arguments vs. formal arguments
  - Positional matching
    - E.g. plot(height, weight)

Functions and arguments

- Most arguments have default values and can be omitted
- Arguments can also be specified in non-positional ways
  - plot(height, weight, pch=2)
  - “pch=2” as **named actual arguments**
  - plot(y=weight, x=height) is the same
  - Mixing positional and named arguments is OK

Vectors

- Character vectors (see code file)
  - A vector of text strings
  - Elements are specified and printed in quotes
- Logical vectors
  - Take value TRUE or FALSE (or NA)
  - Can abbreviate as T or F
  - Often result from **relational expression**
    - E.g. bmi > 25

Functions that create vectors

- **c(…), seq(…), rep(…)**
- All elements of a vector have the same type
- Conversion may happen
  - Logical values to 0/1 or “FALSE/TRUE”
  - Numbers to strings

Matrices and arrays

- Two or more dimensional array of values
- Represented as vectors with dimensions in R
  - dim(x) for the **dimension attribute** of x

Lists

- To combine a collection of objects into a composite object
- Construct from components using **list()**

- Many R functions compute multiple vectors of values, returned as a list

Data frame

- Correspond to “data matrix” or “data set” in other statistical software
- A list of vectors and/or factors of the same length
  - Related so that data in the same position come from the same experimental unit (subject, animal, etc.)
- Has a unique set of row names

	RE121024	RE121043	RE121056	baseline m	RE121004	RE121021	RE121054	RE121054	RE121065
Os.57551.	2.07	2.46	2.39	2.31	0	0	0	A	1.96
Os.56632.	2.23	1.93	3.33	2.5	0	0.49	1.69	P	0
Os.55858.	6.28	1.5	4.99	4.26	1.39	1.17	1.81	P	2.47
Os.9815.1	0.37	2.31	1.54	1.41	0	0.22	0	A	0
Os.45971.	2.77	3.43	0.36	2.19	0.21	0.86	2.92	A	0.35
OsAffx.201	1.37	4.8	2.81	2.99	2.99	1.28	0	A	0
OsAffx.116	2.23	2.22	2.28	2.24	0	1.64	0.01	A	0.88
OsAffx.909	2.59	3.2	1.69	2.49	0.59	0.92	1.45	P	2.39
OsAffx.831	4.16	2.67	4.06	3.63	2.88	0	0	A	3.61
OsAffx.916	4.01	4.05	3.31	3.79	1.46	1.76	1.28	A	1.99
OsAffx.958	4.89	5.43	5.92	5.41	3.99	1.34	2.24	A	2.4
OsAffx.223	2.32	3.1	0	1.81	3.64	0	0	A	0
OsAffx.290	2.86	2.86	3.74	3.15	0.27	2.45	2.04	A	0.79
OsAffx.397	2.78	4.05	5.78	4.21	2.48	0.5	2.51	A	3.23
OsAffx.242	0	4.26	3.01	2.42	1.85	1.5	0	A	0.78
OsAffx.210	6.01	5.73	5.62	5.78	3.09	0.86	0.99	P	4.17
OsAffx.212	2.92	2.39	0.19	1.83	0	0	0	A	2.16
OsAffx.162	2.66	3.38	3.85	3.3	0.62	1.28	0.47	A	3.39
OsAffx.281	1.75	3.4	4.03	3.06	0	2.36	0.76	A	2.24
OsAffx.142	2.57	3.8	2.64	3	0.48	1.93	4.03	P	0

Indexing

- Brackets are used for selection of data, known as **indexing** or **subsetting**

Conditional selection

- To extract data that satisfy certain criteria, such as from male patients
- Use **relational expression** instead of the index
- Indexing with a logical vector is to select values where the logical vector is TRUE
- Comparison operators: <, >, ==, <=, >=, !=
  - "==" is to avoid confusion with the "=" to match keywords with function argument
  - ! for negation

R programming

– Automate iterative tasks
– Handle more complex data and modeling
– Write custom functions
– Modify existing R functions

for() loop

- **for()** statement allows one to specify that a certain operation should be repeated a fixed number of times.

- Examples
  - compute the factorial of 20

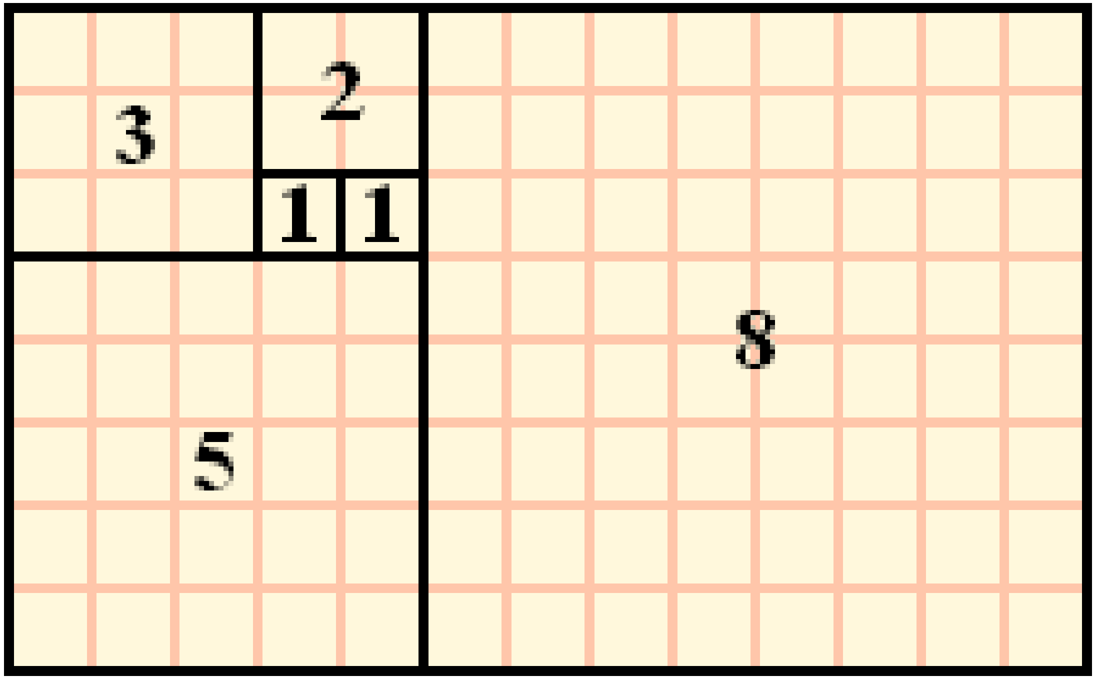
  - stochastic simulation are very repetitive; we want to see patterns of behavior from multiple, simulated instances.

Example: Fibonacci sequence

- Considers the growth of a rabbit population, assuming that:
  - At month 1 there is one pair of newborn rabbits
  - After two months they reach puberty and can give birth to a new pair
  - All mature pairs give birth to a new pair monthly
  - Rabbits never die

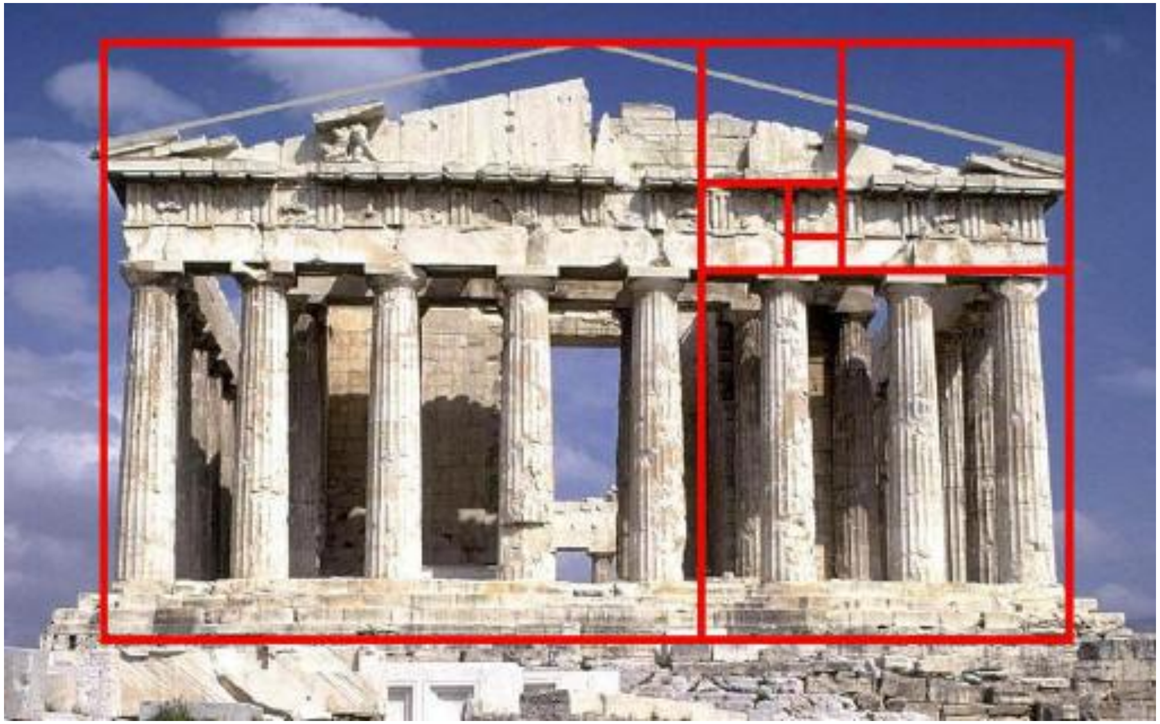
- Total pairs: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55…
- $F(n) = F(n-1) + F(n-2)$

Fibonacci sequence



A tiling with squares whose sides are successive Fibonacci numbers in length

Fibonacci sequence



for() loop

- Syntax

```
for (name in vector) {
      commands
 }
```

  - This sets a variable called **name** equal to each of the elements of **vector**, in sequence
  - For each value of **name**, the **commands** within the curly braces will be performed

Fibonacci sequence

R code for Fibonacci sequence, using for()

```
Fibonacci <- numeric(12)

Fibonacci[1] <- Fibonacci[2] <- 1

for (i in 3:12) {

   Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]

}
```

Another example: Interactive spinning 3D Scatterplot

if() statement

- The **if()** statement allows us to control which statements are executed, depending on the values of some input or variables.

- Examples

```
if (x > 2)
   y <- 2*x
else
    y <- 3*x
```

if() statement

- Syntax 1

```
if (condition) {
    commands when TRUE
}
```

- Syntax 2

```
if (condition) {
    commands when TRUE
} else {
    commands when FALSE
}
```

  - **condition** is logical expression of R, such as "x > 10"
  - Numerical values can be used as the value of **condition:** 0 is FALSE, non-zeros are TRUE.

while() loop

- We want to repeat statements, but the pattern of repetition is not known in advance.
  - We need to do some calculations and keep going as long as a condition holds.

- Examples

```
while (x.total < 100)
    x.total <- x.total + runif(1)
```

- From class: give an example of the while() loop

while() loop

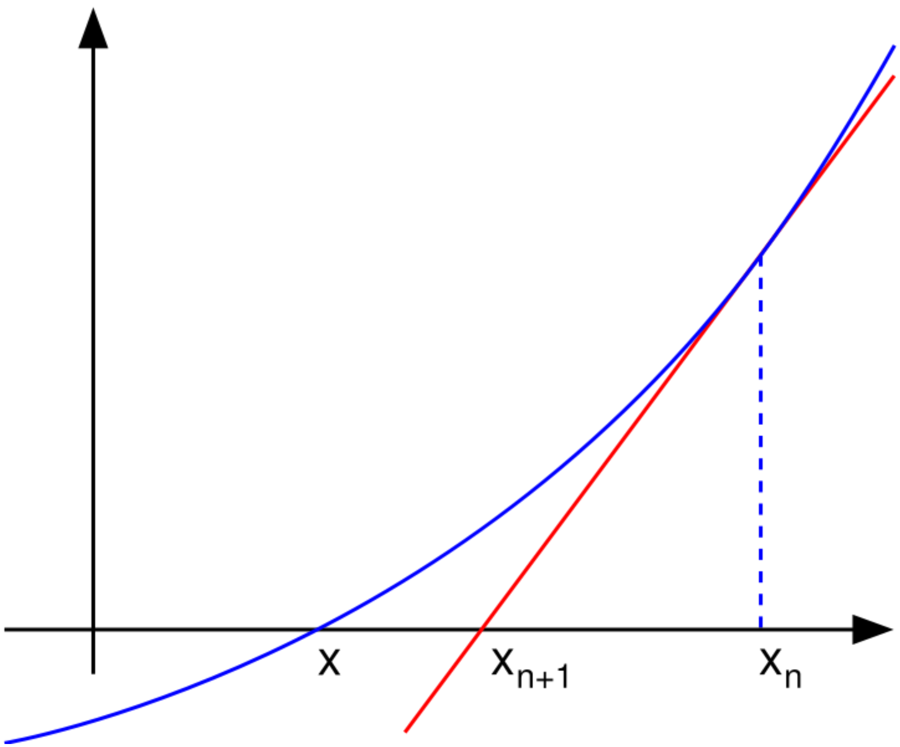
- Syntax

```
while (condition) {
    statements
}
```

  - The **condition** is evaluated, and if it evaluates to FALSE, nothing more is done;
  - If it evaluates to TRUE:
    - the statements are run
    - **condition** is evaluated again, and the process is repeated

Example: Newton's method for root finding

- Find the root of an algebraic equation: *$f(x)=0$*



$$x_0 = \text{initial guess}$$

$$f'(x_n) = \frac{f(x_n)}{x_n - x_{n+1}}$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Example: Newton's method for root finding

- if *f(x)* has derivative *f'(x)*, then the following iterations will converge to a root of *f(x) = 0* if started close enough to the root:

$$x_0 = \text{initial guess}$$

$$f'(x_n) = \frac{f(x_n)}{x_n - x_{n+1}}$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$f(x) = x^3 + 2x^2 - 7$$

$$x_{n+1} = x_n - \frac{x_n^3 + 2x_n^2 - 7}{3x_n^2 + 4x_n}$$

Tip: Write visually pleasing code

- Follow visually pleasing structure
  - Proper line indentation with tabs
  - More space

  - Informative variable names
  - Use variables of values rather than constants
  - Comments to key logic and tasks

Good example

```
###Personalized Homework
###Jia Weng

###Title: Proximation of pi value
###Some numbers are intriguing to me.
###Inspired by Normans's great example of exercise7 on class one,
###I wanted to use similar functions for a simulation to estimate Pi.


######Establish the pi estamation function, using n of random dots
Pi.approximation=function(n)
{

###Generate random x,y pairs in x(0,1) and y(0,1) area
   x = runif(n, 0, 1)
   y = runif(n, 0, 1)

#Visualize total counts in yellow
   plot(x,y,pch=".",col='yellow')

#Visualize pie boundary
   pie.boundary=function(x)
   {
   sqrt(1-x^2)
   }

   curve(pie.boundary, type='l', add=T)


###enumerate counts inside the radius boundary
   inside.pie=function(x,y)
   {
   x^2+y^2
   }
```

Bad example

```
# random walk Metropolis for standard normal

metrop = function(N,b){
x = rep(0,N)
for(i in 2:N){
y = rnorm(1,x[i-1],b)
r = exp((x^2-y^2)/2)
u = runif(1)
if(u < r)x[i] = y  else x[i] = x[i-1]
}
return(x)
}
N = 1000
par(mfrow=c(3,1))
x1 = metrop(N,0.5);x2 = metrop(N,0.1);
x3 = metrop(N,10);
plot(x1,type="l",xlab="",ylab="",xaxt="n",yaxt="n",bty="l")
plot(x2,type="l",xlab="",ylab="",xaxt="n",yaxt="n",bty="l")
plot(x3,type="l",xlab="",ylab="",xaxt="n",yaxt="n",bty="l")
```

Exercise 1

1. If R is not installed, download and install R 3.0 from this link:

http://cran.r-project.org/bin/windows/base/old/3.0.0/

2. Open this R code file in R (File/Open script), and use F5 to run line by line:
   ChengLi_genomics_analysis_with_R_01.R

   You can also select multiple lines to run by F5.

3. Use menu “Packages/Install packages from local zip files” to install “ISwR_2.0-6.zip”. (文件路径名需要是英文）

Exercise 2

- Referring to code section 1.1.3 (vectorized arithmetic), compute the mean and standard deviation of the **weight** variable, using the functions sum(), length(), sqrt(), but not mean(), sd() (use these two to confirm your calculation)
  - Use variable names to save intermediate results, the mean of **weight**
  - In the R console, recall and edit previous commands with the UP and DOWN keys
  - Try to edit and run in the R code file

Exercise 3: for() loop

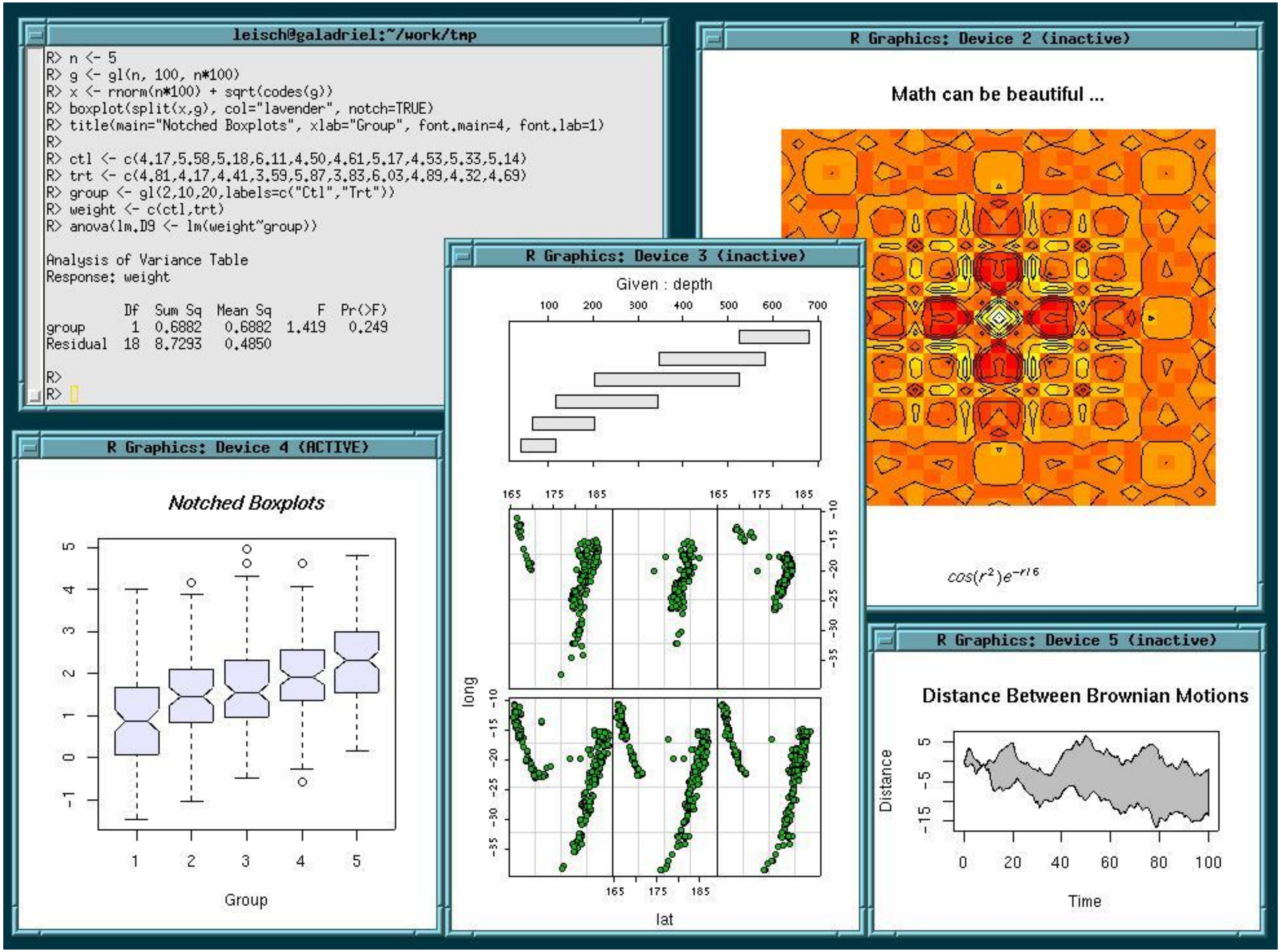
Let f_n denote the *n*th Fibonacci number.

1. Construct a sequence of ratios of the form f_{n+1} / f_n, n = 1, 2... 100. Does the sequence appear to be converging? (you can make a plot)
2. Add the golden ratio $(1+\sqrt{5})/2$ as a line in the plot. Is the sequence converging to this ratio?

hint:

```
> plot(x=1:100, y=(1:100)^2)
> abline(h=1000, col="blue")
```

R graphs


leisch@galadriel:~/work/tmp
R> n <- 5
R> g <- gl(n, 100, n*100)
R> x <- rnorm(n*100) + sqrt(codes(g))
R> boxplot(split(x,g), col="lavender", notch=TRUE)
R> title(main="Notched Boxplots", xlab="Group", font.main=4, font.lab=1)
R> ctl <- c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14)
R> trt <- c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69)
R> group <- gl(2,10,20,labels=c("Ctl","Trt"))
R> weight <- c(ctl,trt)
R> anova(lm.D9 <- lm(weight~group))
Analysis of Variance Table
Response: weight
Df Sum Sq Mean Sq F Pr(>F)
group 1 0.6882 0.6882 1.419 0.249
Residual 18 8.7293 0.4850
R Graphics: Device 2 (inactive)
Math can be beautiful ...
cos(r²)e^{-r/6}
R Graphics: Device 3 (inactive)
Given : depth
long
lat
R Graphics: Device 4 (ACTIVE)
Notched Boxplots
Group
R Graphics: Device 5 (inactive)
Distance Between Brownian Motions
Distance
Time

Why we use graphs?

- Help understand and solve a problem
- Help monitor and debug code. like print()
- Other examples and benefits of graphs?

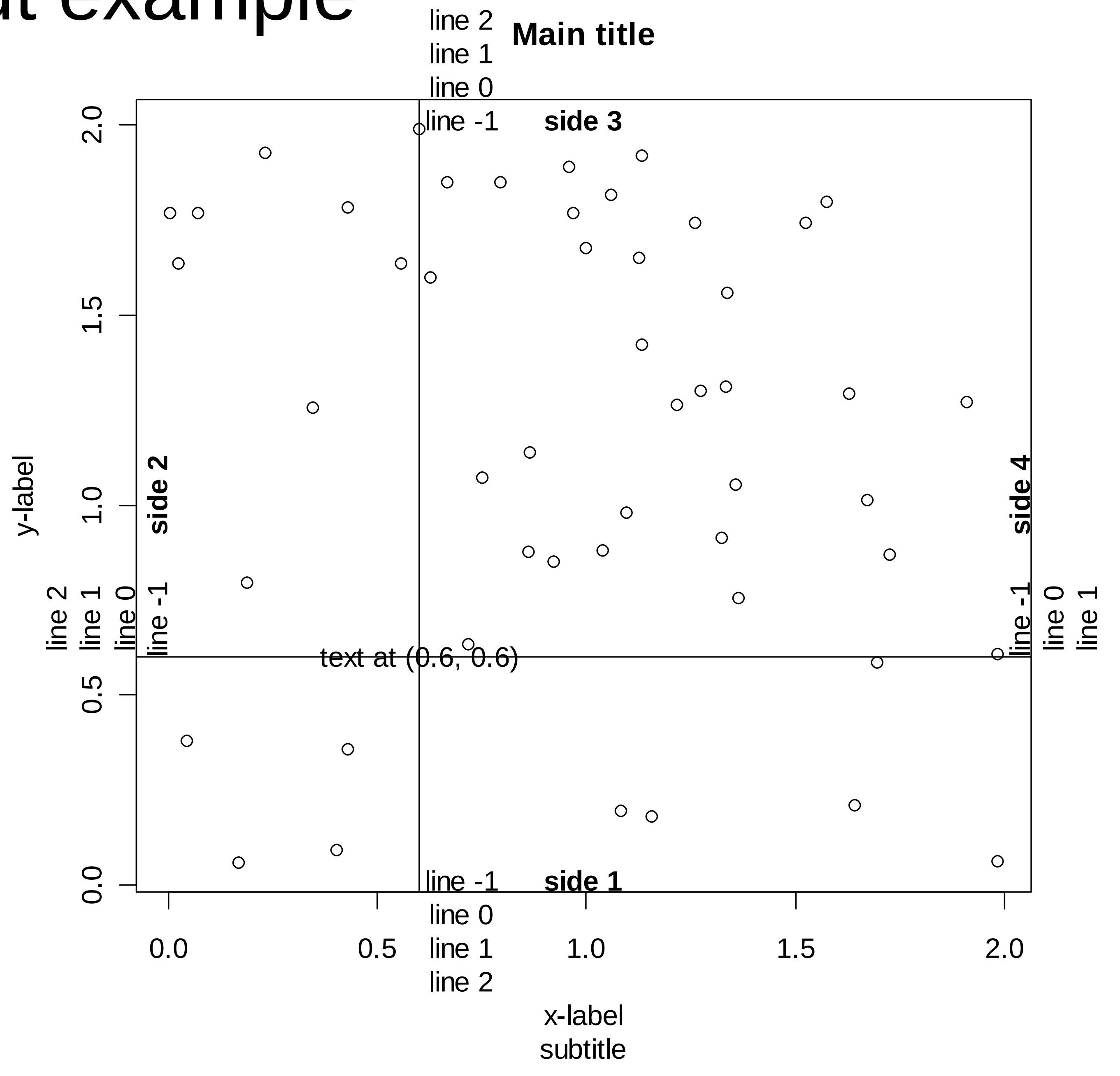
R graphs is flexible

- add annotations
- different axes
- labels
- irregular tick marks

Plot layout

- A central plotting region surrounded by margins
- Coordinates inside the plotting region are in data units along the side
- Coordinates in the margins are in **lines of text** perpendicular to a side

Layout example



Build a plot from pieces

- High-level plots are composed of elements
  - each element can be drawn separately
  - allows finer control of the elements

- **type="n"** is useful
  - a plot with different colors for different groups
- **par():** pick up a few useful tricks at a time
  - change margin size: par(mar=c(4,4,2,2))

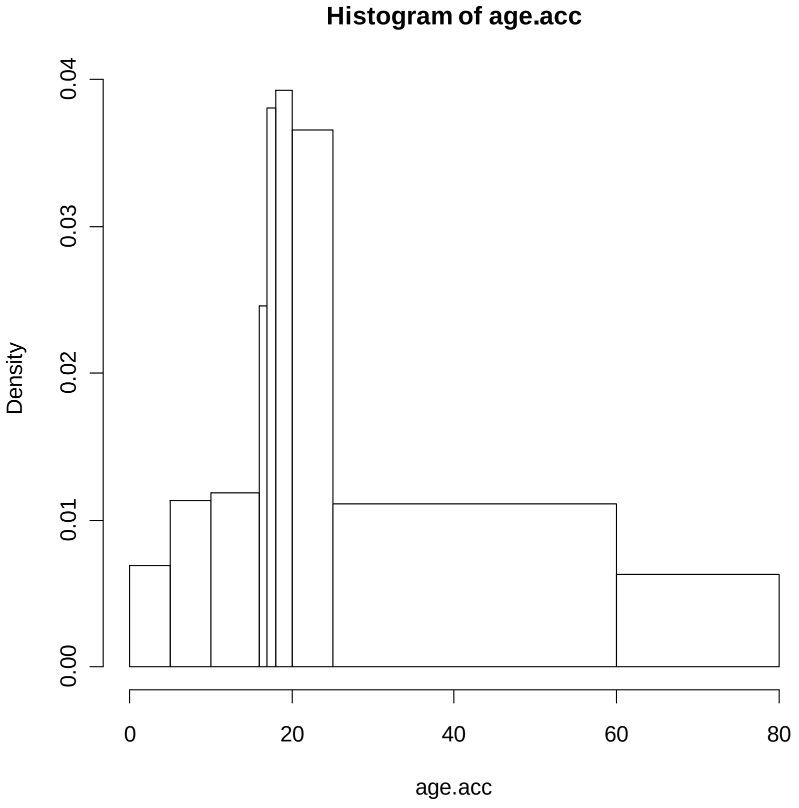
Summary statistics

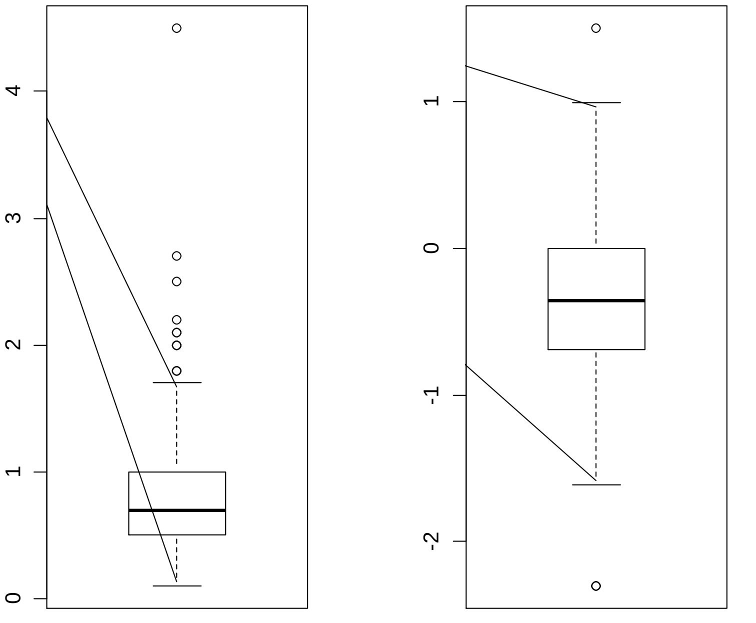
- mean(), sd(), var(), median()
- quantile()

- missing value handling

- summary()

Descriptive plots





Install the **ISwR** package

- A R package contains data sets and function code

- R menu "Packages/Install packages" to download. Select “ISwR”.

- menu "Packages/Load packages" to use it
  - or command "library(ISwR)"
  - search() to see ISwR is in search path
  - need to reload it next time using R

Display of distributions: histograms

- hist()
  - counting how many observations fall within bins of the x-axis
  - **breaks=n** argument
  - breaks as a vector for fine control
  - the area of a column is proportional to the number
    - freq=T is default for equidistant breakpoints

Q-Q plots

- Check whether data can be assumed normally distributed.
  - plot the k-th smallest observation against the expected value of the *k*-th smallest observation out of *n* in a standard normal distribution.
  - you would expect to obtain a straight line if data come from a normal distribution with any mean and standard deviation.

qqnorm(x)

Boxplots

- Box-and-whiskers plots, a graphical summary of a distribution
  - Box boundaries are quartiles (25^{th} and 75^{th} percentiles) and median
  - Lines (whiskers) show the largest or smallest data points that fall within 1.5 * box height (Inter Quartile Range) from the nearest box boundaries
  - Farther away points are shown separately as outliers
- mfrow=c(1,2): **m**ulti**f**rame, **row**wise, 1 by 2 layout

Basic graphics functions in R

Generic function ***plot*** and its methods:

plot	**Generic X-Y Plotting**
plot.data.frame	**Plot Method for Data Frames**
plot.default	**The Default Scatterplot Function**
plot.design	Plot Univariate Effects of a 'Design' or Model
plot.factor	**Plotting Factor Variables**
plot.formula	**Formula Notation for Scatterplots**
plot.histogram	**Plot Histograms**
plot.table	**Plot Methods for 'table' Objects**
plot.window	Set up World Coordinates for Graphics Window
plot.xy	Basic Internal Plot Function

Basic graphics functions in R

Axis	Generic function to add an Axis to a Plot
abline	**Add Straight Lines to a Plot**
arrows	Add Arrows to a Plot
assocplot	Association Plots
axTicks	Compute Axis Tickmark Locations
axis	Add an Axis to a Plot
axis.POSIXct	Date and Date-time Plotting Functions
barplot	**Bar Plots**
box	Draw a Box around a Plot
boxplot	**Box Plots**
bxp	Draw Box Plots from Summaries
cdplot	Conditional Density Plots
contour	Display Contours
coplot	Conditioning Plots
curve	Draw Function Plots
dotchart	**Cleveland Dot Plots**
filled.contour	Level (Contour) Plots
fourfoldplot	Fourfold Plots
frame	Create / Start a New Plot Frame
graphics-package	The R Graphics Package
grid	Add Grid to a Plot
hist	**Histograms**
hist.POSIXt	Histogram of a Date or Date-Time Object

Basic graphics functions in R

identify	**Identify Points in a Scatter Plot**
image	Display a Color Image
layout	Specifying Complex Plot Arrangements
legend	**Add Legends to Plots**
lines	**Add Connected Line Segments to a Plot**
locator	**Graphical Input**
matplot	**Plot Columns of Matrices**
mosaicplot	**Mosaic Plots**
mtext	Write Text into the Margins of a Plot
pairs	**Scatterplot Matrices**
panel.smooth	Simple Panel Plot
par	**Set or Query Graphical Parameters**
persp	Perspective Plots
pie	Pie Charts

Basic graphics functions in R

points	**Add Points to a Plot**
polygon	Polygon Drawing
rect	Draw One or More Rectangles
rug	Add a Rug to a Plot
screen	Creating and Controlling Multiple Screens on a Single Device
segments	Add Line Segments to a Plot
spineplot	Spine Plots and Spinograms
stars	Star (Spider/Radar) Plots and Segment Diagrams
stem	Stem-and-Leaf Plots
stripchart	**1-D Scatter Plots**
strwidth	Plotting Dimensions of Character Strings and Math Expressions
sunflowerplot	Produce a Sunflower Scatter Plot
symbols	Draw Symbols (Circles, Squares, Stars, Thermometers, Boxplots) on a Plot
text	**Add Text to a Plot**
title	Plot Annotation
xinch	Graphical Units

Probability and distributions

- Statistical methods: view data as coming from a statistical distribution

- R functions for random sampling and handling of theoretical distributions

Random sampling

- A random sample
  - dealing from a well-shuffled pack of cards
  - picking numbered balls from a well-stirred box

  - Other examples?

- **sample()**

Built-in distributions in R

- Replace traditional statistical tables
- Four values can be computed for a distribution:
  - density or point probability (**d**norm)
  - cumulated probability, distribution function (**p**norm)
  - quantiles (**q**norm)
  - pseudo-random numbers (**r**norm)

Densities

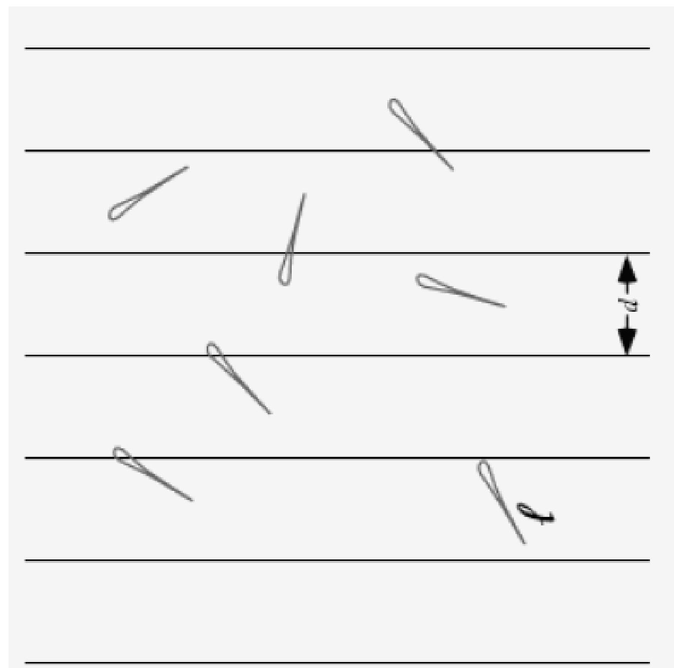
- continuous distribution:
  - measure of relative probability of "getting a value close to x"
  - probability of getting a value in an interval is the area under the corresponding density curve
- discrete distribution:
  - point probability of getting value x
- uses:
  - overlay theoretical density on histograms

Random numbers

- Computer algorithms are predictable and reproducible
- We can only generate "pseudo-random" numbers
  - for practical purpose behave **as if** they were drawn randomly
- Uses
  - create simulated data to test statistical methods
  - bootstrap, resampling methods

Cheng's Needle-throwing example

Buffon's needle problem asks to find the probability that a needle of length will land on a line, given a floor with equally spaced parallel lines a distance apart.



Your exercise to prove: if $l = d$, then $Prob(Intersection) = 2/\pi$.

Thus we can simulate the process, count the intersection events, and estimate π. Manually? Using R?

Tip: Save and load workspace

- See code
- save.image(), or menu "File/Save Workspace“
- load(".RData"), or menu "File/Load Workspace"

Exercise 4

- Plot the graph of the function

```
f(x) = 3x + 2,                x <= 3
       2x - 0.5 * x^2,         x > 3
```

on the interval [0, 6]. consider the function curve()
(use “?curve” in R to get more examples)

Hint: use this to set up the plotting window:
plot(x = c(), y = c(), xlim = c(0, 6), ylim = c(-10, 20));

Exercise 5: Three example plots

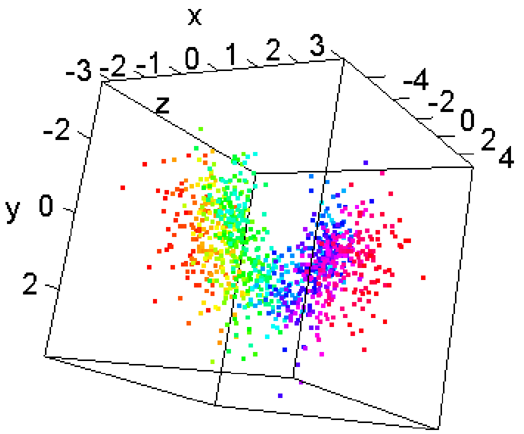
- Run the example code “3 example plots”

- use R help to understand and explain the data sets involved, as well as what the code does to produce the 3 plots.

Exercise 6: Interactive 3D scatterplots

- Refer to the code section "Interactive spinning 3D Scatterplot". (use the updated version sent by email)

- Use "data(package="ISwR")" to browse and select a data set from the ISwR package, and use Interactive 3D scatterplots to explore the relationships between the variables. Do you gain more insights compared to 2D plots?



Summary

- R data structures and flow controls
- Graph elements and descriptive plots
- Probability and random sampling

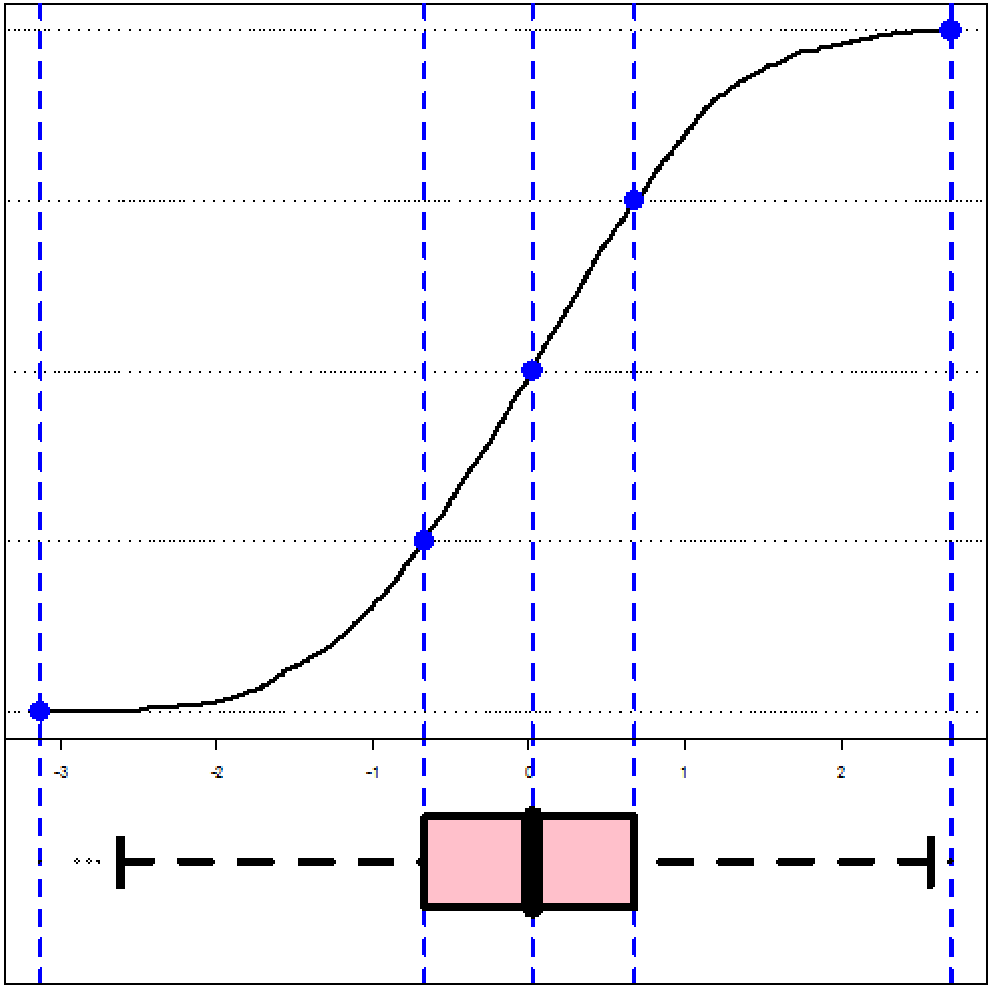
To do

- Finish the exercises.
- Review today's topics
  - Introductory_Statistics_With_R_Chapter_1-2.pdf
  - Introductory_Statistics_With_R_Chapter_3-5.pdf

Exercise 7: Graphs (p1 of 2)

Task 1: Combine the boxplot and culmulative plot. For a vector of values (e.g. rnorm(1000)), write R code to make the plot on the right:

the cumulative distribution (x-axis is the sorted observed values, y-axis is probability points from 0 to 1), the boxplot, and the vertical lines for quartiles.

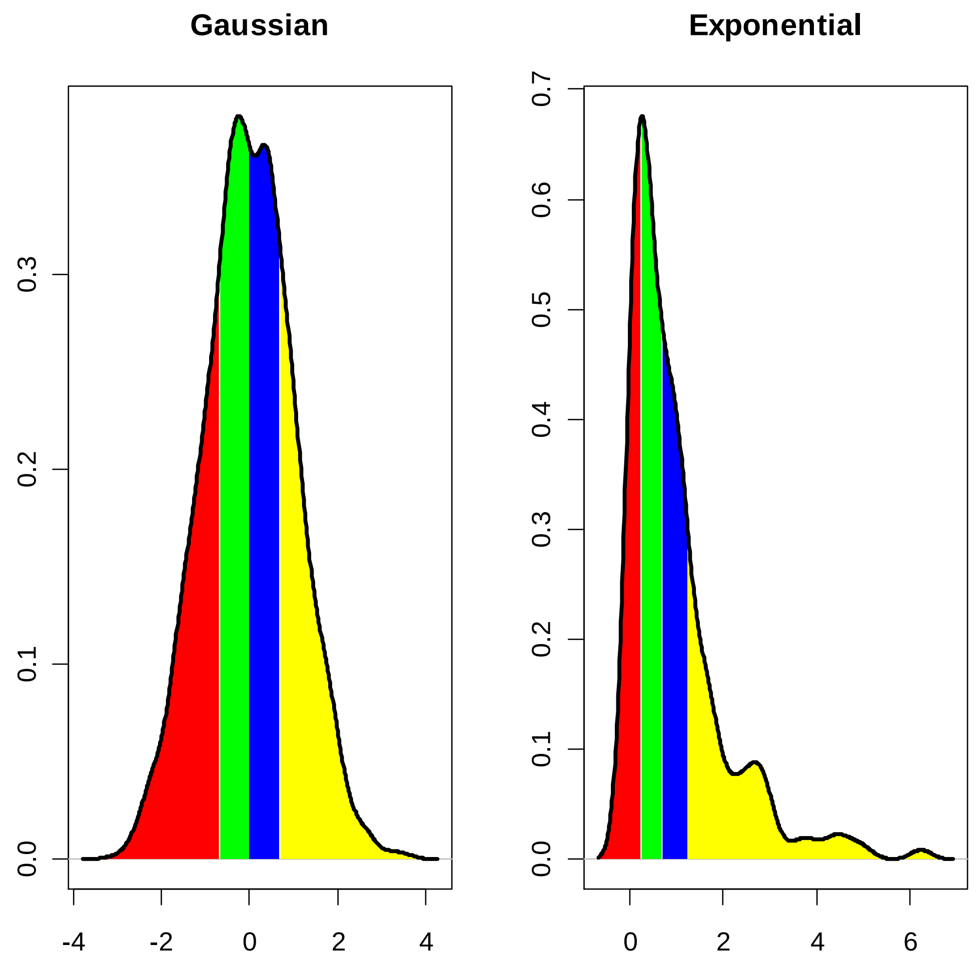


Hint: consider ppoints(), layout(), boxplot(x, horizontal = TRUE)

Graphs (p2 of 2)

One of main use of boxplots is assessing the symmetry of the data. This graphical representation of the quartiles uses areas of different colors. The four areas are equal; this highlights the often-claimed fact that the human eye cannot compare areas.

Task 2: read the example code and interpret what each line does.



Exercise 8. Your own R graphs

- Browse examples of R Graphics
  - http://zoonek2.free.fr/UNIX/48_R/03.html

- Can you think of an interesting visualization task from your life or courses, so that you can use R to make a graph or animation? (if needed, use R to analyze data first.)