Introduction to R

**Part 1: Getting Started**

*download, install, run, help*

<http://www.r-project.org>

Select a mirror site near you

Windows

Download and Install R

base

(click exe of latest version)

OSX

MacOS X

(click .pkg of latest version)

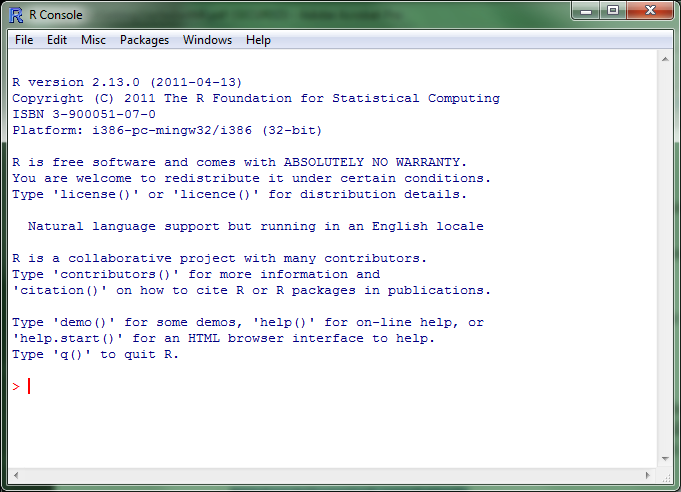
Optional: Download, install, and run Rstudio

<http://rstudio.org/download/desktop>

**Install R**

**Start R**

This is the R console



Type or paste code here.

**Working with R**

Typically when working with R, you will have some code you are working on in a text file in one window and the R will be open in a separate window. You will usually write your code in the text editor window and then copy / paste / send it into the R console to run. I will be showing you how to do this using Rstudio, but other options include Tinn-R or Wordpad (or your text editor of choice).

**Set up for this tutorial**

In this document, anything you see in red should be what you enter into the R console, and in blue will be the R output.

For the purposes of this tutorial, the following commands will create a folder on your H drive and copy some data to it that we'll be using in the tutorial.

dir.create("H:/introR")

file.copy(from="U:/mcm/presentations/CompTools/introR/data",to="H:/introR ",recursive=T)

Set your working directory

Your working directory will be where any files you generate end up. It can also be convenient (though not strictly required) to be the location of any data you might be reading in along with the location of any script you are currently working on.

setwd("H:/introR")

If you don’t explicitly set your working directory on windows, it will be set to

C:\Documents and Settings\mcm\My Documents *(Windows XP)*

or

C:\Users\mcm\Documents *(Windows Vista, Windows 7)*

**Get help on a function**

?mean

Browse Documentation

*Try asking for help on setwd*

help.start()

**Web resources**

R search engine - <http://rseek.org>

Stack Overflow - <http://stackoverflow.com>

R graph gallery - <http://addictedtor.free.fr/graphiques/>

**Entering commands**

You can use R like a calculator

1+1

[1] 2

To explain the result, the [1] before the 2 is because R considers the result of a calculation a vector with one element. So the first element of the vector is labeled with [1].

2 + 2

100 - 20

4 \* 6

10^2

100 / 4

sqrt(2)

log2(2)

Remember, to get help on a function, type question mark followed by function name

*What is 2 to the 12th?*

*What is the log (base 10) of 10000?*

?log2

**Create a vector and store it in a variable**

A vector in R can be a list of numbers, of strings (letters/words), or of TRUE/FALSE values. You can use the c() function to **combine** elements into a vector.

x<-c(1,3,6,8,13)

y<-c(2,5,4,7,12)

words<-c("hi","how","are","you")

sentences<-c("Hi, how are you?", "I am fine.")

torf<-c(TRUE,TRUE,FALSE,TRUE,TRUE)

If you mix strings and numbers, the numbers will be treated as strings.

v1<-c(6, 5,"hi")

A vector of two vectors gets flattened into one vector.

z<-c(x,y)

z

[1] 1 3 6 8 13 2 5 4 7 12

There is a shortcut in R, to create a vector with integers from one number to another.

1:10

[1] 1 2 3 4 5 6 7 8 9 10

to store that in a variable, you would do

m<-1:10

*create a vector of words of your choosing*

*create a vector of numbers from 5 to 20*

Calculate some basic statistics on vectors

Note: when you see a # in front of something, it's just a comment and is ignored by R.

mean(x)

median(x)

#sample standard deviation

sd(x)

[1] 4.658326

summary(x)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.0 3.0 6.0 6.2 8.0 13.0

#correlation

cor(x,y)

[1] 0.9442803

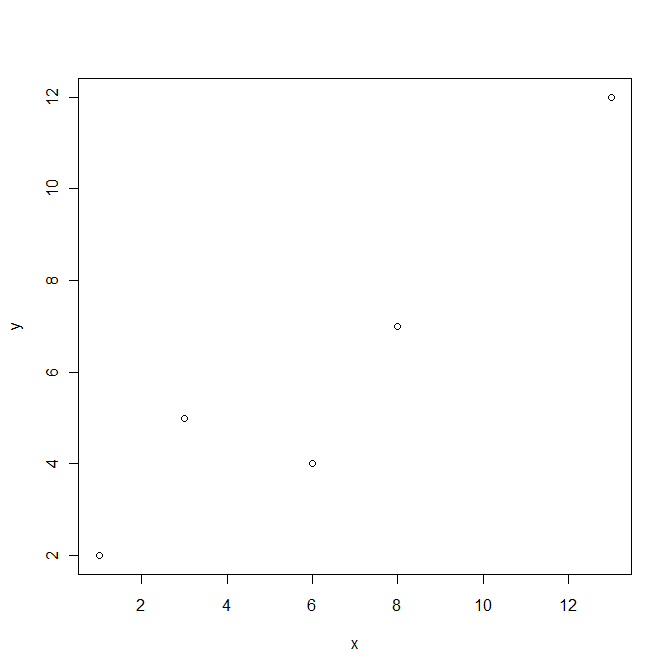
length(y)

[1] 5

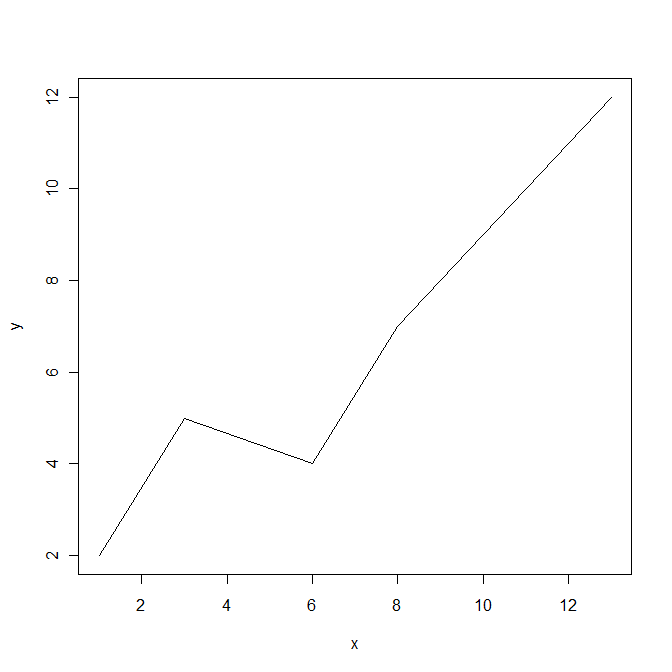
*what is the median of numbers 4 to 7?*

Plot x vs y

plot(x,y)



plot(x,y,type='l')



More on this later!

**Read in some data**

df <-read.table("H:/introR/data/yeast/gene\_relationships.txt",header=T,sep='\t')

In this case, data.txt must be a text file with the following properties:

* Each line contains one record
* Within each record, fields are separated by a tab, comma, or other single-character
* Each record contains the same number of fields

Data frames

A Data frame is a collection of columns. Each column can be a different type of data, numbers, strings, logical (TRUE/FALSE). If you use read.table, the result is already a data.frame, which you can check using

class(df)

[1] "data.frame"

Each column of the data frame is a vector, and each element of the data frame will be whatever type it is. You can access an individual row:

df[1,]

column:

df[,4]

or element:

df[1,7]

as well as little sections:

df[1:10,]

head(df)

You can also access columns by name, two different ways:

df$gene

df[,"gene"]

*Save the 4th column from the data frame into a new variable called "genes".*

Factors

Now might be a good time to mention factors. Factors are a feature of R used for categorical data, but will probably confuse you at first. When reading in a file with read.table, string data will automatically be converted to a factor unless you specify as.is=T (or stringsAsFactors=F).

df <-read.table("H:/introR/data/yeast/gene\_relationships.txt",header=T,sep='\t',as.is=T)

Scatter plot

Read in a data set from MyPyramid.gov listing some nutritional information for some common foods.

food<-read.table("H:/introR/data/food/food.txt",header=T,sep='\t',as.is=T,quote="")

plot(food$Calories,food$Saturated\_Fats)

Linear Regression

What is the relationship between two variables?

lm(food$Saturated\_Fats~food$Calories)

Call:

lm(formula = food$Saturated\_Fats ~ food$Calories)

Coefficients:

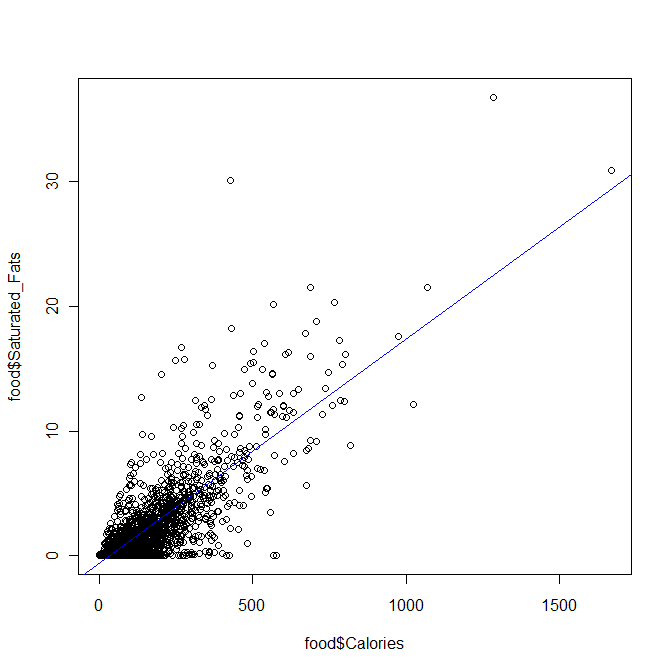
(Intercept) food$Calories

-0.64745 0.01802

Plot the regression line on the scatter plot.

plot(food$Calories,food$Saturated\_Fats)

abline(lm(food$Saturated\_Fats~food$Calories),col='blue')



#calc the correlation and put it on the plot too

correlation<- cor(food$Saturated\_Fats,food$Calories)

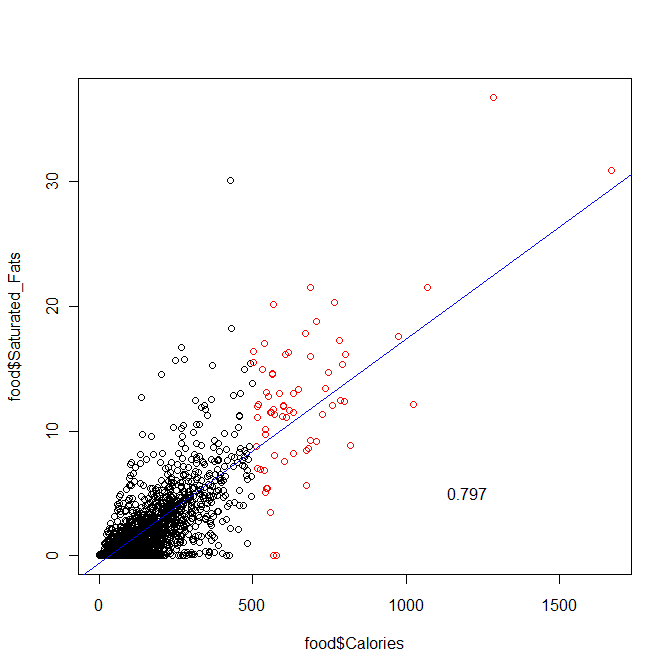
text(1200,5,paste("cor =",round(correlation,3)))

Highlight some points of interest on the plot

iv<-food$Calories > 800

iv is what we call an index vector, a way of selecting some part of your data in R. If you type *iv*, you see a lot of TRUE and FALSE values. iv is the same length as food$Calories, with a TRUE or FALSE for each element depending on whether or not it is > 800.

points(food$Calories[iv],food$Saturated\_Fats[iv],col='red')



Now that we have our index vector, we can use it to examine our data frame as well.

food[iv,]

*Read the following file into a variable called "microarray":*

*H:/introR/data/array/microarray1.txt*

*Is there a header on the file? Don't forget header=T*

*Once you have read in the file, try to make a scatter plot of Cy3 vs Cy5.*

*Create an index vector selecting points that have log2(Cy5/Cy3) > 1.*

*Highlight those points on the plot in blue.*