Sample R Markdown Document

Purvasha Chakravarti

September 6, 2018

R Markdown

R Markdown

R Markdown

This is an **R Markdown** document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

The default environment allows for text (potentially emphasized with fonts *like* **this**), lists, and LaTeX math.

Here is a list:

1. Item 1
2. Item 2
3. Item 3
    - Item 3a
    - Item 3b

You can embed LaTeX math like this:

$$a^2 + b^2 = c^2.$$

You may also want to embed math in-line like this: $1 + 1 = 2$.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

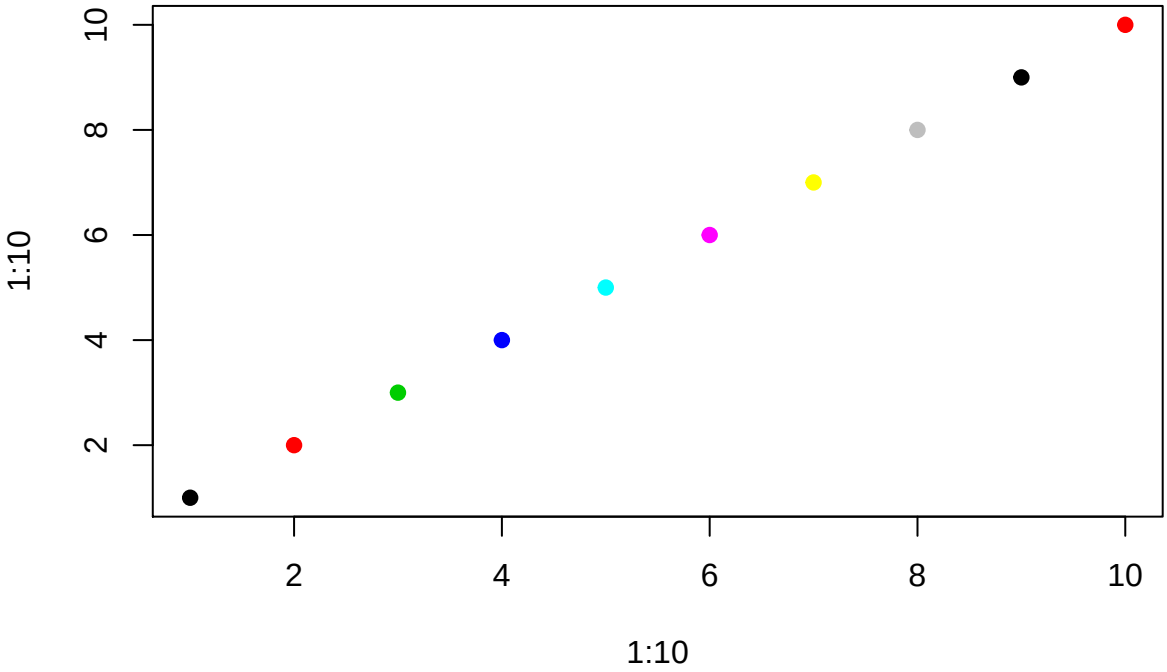
```
# comment
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Here we specifically requested to print the first and third lines of the R input chunk, thus keeping the second line from appearing in the Knit document.

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```
##########################
# R Tutorial
# Class: 36-401
# Date: September 6, 2018
# Author: Purvasha Chakravarti
##########################
```

Basic math

```
1 + 1
```

```
## [1] 2
```

```
10 / 4
```

```
## [1] 2.5
```

```
2 * 2
```

```
## [1] 4
```

```
15 %% 7
```

```
## [1] 1
```

```
2 ^ 4
```

```
## [1] 16
```

Variables

```
x <- 100
2 * x
```

```
## [1] 200
```

```
y <- 10
x / y
```

```
## [1] 10
```

```
z <- x + y
```

Vectors/Matrices

```r
1:10
```

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

```r
10:1
```

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

```r
c(1,2,5,0,1234)
```

```
## [1]    1    2    5    0 1234
```

```r
seq(0, 10, by = 0.5)
```

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0
```

```r
seq(0, 10, length = 7)
```

```
## [1]  0.000000  1.666667  3.333333  5.000000  6.666667  8.333333 10.000000
```

```r
rep(0, times = 10)
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0
```

```r
x <- 1:10
length(x)
```

```
## [1] 10
```

```r
x[5]
```

```
## [1] 5
```

```r
x[6:10]
```

```
## [1]  6  7  8  9 10
```

```r
y <- 11:20
z <- c(x,y)
z
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```r
matrix(1:16, ncol = 4, nrow = 4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

```r
matrix(1:16, ncol = 4, nrow = 4, byrow = TRUE)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
## [3,]    9   10   11   12
## [4,]   13   14   15   16
```

```r
A <- matrix(1:16, ncol = 4, nrow = 4, byrow = TRUE)
ncol(A)
```

```
## [1] 4
```

```r
nrow(A)
```

```
## [1] 4
```

```r
dim(A)
```

```
## [1] 4 4
```

```r
t(A)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

```r
#solve(A) # error: A is not invertible!
B <- matrix(16:1, ncol = 4, nrow = 4, byrow = TRUE)
A %*% B # matrix multiplication
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   80   70   60   50
## [2,]  240  214  188  162
## [3,]  400  358  316  274
## [4,]  560  502  444  386
```

```r
# WARNING! A * B and A ^ 2 perform elementwise operations!

A + B
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   17   17   17   17
## [2,]   17   17   17   17
## [3,]   17   17   17   17
## [4,]   17   17   17   17
```

```r
b <- 1:4
A %*% b # matrix-vector multiplication
```

```
##      [,1]
## [1,]   30
## [2,]   70
## [3,]  110
## [4,]  150
```

R Documentation

```r
# ? and help() are your friends!
?matrix
?nrow
```

Strings

```r
x <- "this is a string"
c("we","can","store","strings","in","a","vector","too")
```

```
## [1] "we"      "can"     "store"   "strings" "in"      "a"       "vector"
## [8] "too"
```

```r
paste("we","can","paste","strings","together", sep = " ")
```

```
## [1] "we can paste strings together"
```

```r
length(x)
```

```
## [1] 1
```

```r
nchar(x)
```

```
## [1] 16
```

```r
substr(x, 1, 4)
```

```
## [1] "this"
```

Some statistics

```r
x <- 1:10
min(x)
```

```
## [1] 1
```

```r
max(x)
```

```
## [1] 10
```

```r
sum(x)
```

```
## [1] 55
```

```r
mean(x)
```

```
## [1] 5.5
```

```r
range(x)
```

```
## [1]  1 10
```

Global Environment

```r
ls()
```

```
## [1] "A" "b" "B" "x" "y" "z"
```

```r
rm(x)
rm(list = ls(all = TRUE))
gc()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 405734 21.7     750400 40.1   592000 31.7
## Vcells 620380  4.8    1308461 10.0   889957  6.8
```

Working directory

```r
getwd()
```

```
## [1] "/Users/purvasha/Dropbox/36401 Fall 2018/R tutorial"
```

```r
setwd("~")
list.files()
```

```
##  [1] "anaconda"                          "Applications"
##  [3] "Calibre Library"                   "Desktop"
##  [5] "Documents"                         "Downloads"
##  [7] "DPP4_test_disguised.csv"           "DPP4_training_disguised_new.csv"
##  [9] "Dropbox"                           "Dropbox (Old)"
## [11] "gfortran-4.8.2-darwin13.tar.bz2" "graph_CNN"
## [13] "Library"                          "MHsampling.pdf"
## [15] "Mickey22.pdf"                     "Movies"
## [17] "Music"                            "Pictures"
## [19] "Public"
```

```r
setwd("~/Dropbox/36401 Fall 2018/R tutorial")
```

Comparisons & Index search

```r
x <- 1
x == 1
```

```
## [1] TRUE
```

```r
x != 1
```

```
## [1] FALSE
```

```r
# <, >, <=, >=, ==, !=

# Multiple comparisons
y <- 2
x == 1 & y == 2
```

```
## [1] TRUE
```

```r
x == 1 & y < 2
```

```
## [1] FALSE
```

```r
rm(list = ls(all = TRUE))
gc()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 406028 21.7     750400 40.1   605919 32.4
## Vcells 621689  4.8    1308461 10.0   889957  6.8
```

```r
# Index search
x <- 1:10
which(x < 5)
```

```
## [1] 1 2 3 4
```

```r
which(x > 4 & x < 6)
```

```
## [1] 5
```

```r
which(x <= 1 | x >= 10)
```

```
## [1]  1 10
```

```r
A <- matrix(1:16, ncol = 4, byrow = TRUE)
which(A == 12, arr.ind = TRUE)
```

```
##      row col
## [1,]   3   4
```

If/else statements

```r
if ( TRUE ){
  # code you want to run if TRUE
} else {
  # code you want to run if FALSE
}
```

```
## NULL
```

```r
x <- 1

if ( x < 0 ){
  cat("x is negative!")
} else {
  cat("x is nonnegative!")
}
```

```
## x is nonnegative!
```

Loops

```r
x <- 0
for ( itr in 1:10 ){
  x <- x + 1
}
x
```

```
## [1] 10
```

```
n <- 4
x <- 1
while ( n >= 1 ){
  x <- x * n
  n <- n - 1
}
```

Functions

```
myFactorial <- function(n){
  x <- 1
  while ( n >= 1 ){
    x <- x * n
    n <- n - 1
  }
  return(x)
}

cleanup <- function(){
  rm(list = setdiff(ls(all = TRUE, envir = globalenv()), "cleanup"), envir = globalenv())
  gc()
}
```

Random number generation

```
set.seed(1) # set the seed if you want to reproduce your work

# Sampling from well-known distributions
rnorm(50, mean = 0, sd = 1)
```

```
##  [1] -0.62645381  0.18364332 -0.83562861  1.59528080  0.32950777
##  [6] -0.82046838  0.48742905  0.73832471  0.57578135 -0.30538839
## [11]  1.51178117  0.38984324 -0.62124058 -2.21469989  1.12493092
## [16] -0.04493361 -0.01619026  0.94383621  0.82122120  0.59390132
## [21]  0.91897737  0.78213630  0.07456498 -1.98935170  0.61982575
## [26] -0.05612874 -0.15579551 -1.47075238 -0.47815006  0.41794156
## [31]  1.35867955 -0.10278773  0.38767161 -0.05380504 -1.37705956
## [36] -0.41499456 -0.39428995 -0.05931340  1.10002537  0.76317575
## [41] -0.16452360 -0.25336168  0.69696338  0.55666320 -0.68875569
## [46] -0.70749516  0.36458196  0.76853292 -0.11234621  0.88110773
```

```
runif(50, min = 0, max = 1)
```

```
##  [1] 0.65472393 0.35319727 0.27026015 0.99268406 0.63349326 0.21320814
##  [7] 0.12937235 0.47811803 0.92407447 0.59876097 0.97617069 0.73179251
## [13] 0.35672691 0.43147369 0.14821156 0.01307758 0.71556607 0.10318424
```

```
## [19] 0.44628435 0.64010105 0.99183862 0.49559358 0.48434952 0.17344233
## [25] 0.75482094 0.45389549 0.51116978 0.20754511 0.22865814 0.59571200
## [31] 0.57487220 0.07706438 0.03554058 0.64279549 0.92861520 0.59809242
## [37] 0.56090075 0.52602772 0.98509522 0.50764182 0.68278808 0.60154122
## [43] 0.23886868 0.25816593 0.72930962 0.45257083 0.17512677 0.74669827
## [49] 0.10498764 0.86454495
```

```
rbinom(50, size = 20, prob = 0.5)
```

```
##  [1] 11 10  9 10 10  8 10  7  9  8  9 13 10 12 13 10  7  9 11  9 11 12 12
## [24]  9  9 13 11 11 11 13  9  8 13 10 13  8 12 11 14 10 11  9  7 13  9 11
## [47]  7 12  9 12
```

```
rchisq(50, df = 10)
```

```
##  [1]  6.560435  6.576298  9.682946  6.437073  4.386400 13.275751 10.703760
##  [8] 17.551727  6.504879  9.108013  9.116945  3.283095  7.410606 10.480223
## [15] 11.221652  9.595237  8.503064 15.599908  8.112639  4.894931 12.277534
## [22]  8.690599  4.529266  7.676813  7.094928 11.396191  3.710760  5.401532
## [29]  7.768360  5.688085  8.760226  9.293017  7.678849  7.140837  6.037293
## [36] 20.033230  9.073955 17.173830  8.595138  5.488907  7.961876 13.742792
## [43]  6.557175  7.479884 10.893903  8.015982 17.540484 12.611370 12.280379
## [50]  6.844107
```

```
# others: rgamma, rbeta, rf

# Distribution functions
pnorm(-1.96, mean = 0, sd = 1)
```

```
## [1] 0.0249979
```

```
punif(0.2, min = 0, max = 1)
```

```
## [1] 0.2
```

```
pbinom(1, size = 10, prob = 0.5)
```

```
## [1] 0.01074219
```

```
# analogous functions for other distributions: pchisq, pgamma, pbeta, pf

# Density functions
dnorm(0, mean = 0, sd = 1)
```

```
## [1] 0.3989423
```

```
runif(0.5, min = 0, max = 1)
```

```
## numeric(0)
```

```r
dbinom(0, size = 10, prob = 0.5)
```

```
## [1] 0.0009765625
```

```r
dbinom(0, size = 10, prob = 0.5) + dbinom(1, size = 10, prob = 0.5) == pbinom(1, size = 10, prob = 0.5)
```

```
## [1] FALSE
```

```r
# similarly: dchisq, dgamma, dbeta, df

# Quantile functions
qnorm(0.975, mean = 0, sd = 1)
```

```
## [1] 1.959964
```

```r
qbinom(0.5, size = 10, prob = 0.5)
```

```
## [1] 5
```

```r
# again: qchisq, qgamma, qbeta, qf

# Sampling from discrete distributions
sample(1:10, size = 3, replace = TRUE)
```

```
## [1] 1 4 5
```

```r
sample(1:10, size = 10, replace = TRUE, prob = 1:10 / sum(1:10))
```

```
##  [1] 10  7  2  1 10  7  8  6  9  8
```

File I/O & Data Frames

```r
# File I/O
setwd("~/Dropbox/36401 Fall 2018/R tutorial")
x <- 1:10
save(x, file = "tmp.RData")
rm(x)
# x: Error (not found)
load("tmp.RData")
x
```

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

```r
write(x, file = "tmp.txt")
scan("tmp.txt")
```

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

```r
# Data frames
cleanup()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 408322 21.9     750400 40.1   750400 40.1
## Vcells 627252  4.8    1308461 10.0   978437  7.5
```

```r
x <- 1:100
y <- x + rnorm(100, sd = 10)
data <- data.frame(x,y)

View(data)
head(data)
```

```
##   x          y
## 1 1 -15.530938
## 2 2  -1.238143
## 3 3   3.273891
## 4 4   3.352623
## 5 5   9.724588
## 6 6  11.112738
```

```r
tail(data)
```

```
##       x         y
## 95   95  90.44547
## 96   96  99.41791
## 97   97  81.68477
## 98   98 105.69569
## 99   99 115.01689
## 100 100  85.26335
```

```r
write.table(data, file = "data.txt")
rm(data)
data <- read.table("data.txt")

write.csv(data, file = "data.csv", row.names = FALSE)
rm(data)
data <- read.csv("data.csv")
```

Installing packages

```r
install.packages("dplyr", repos = "http://cran.us.r-project.org")
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# combining data frames
cleanup()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 455829 24.4     940480 50.3   750400 40.1
## Vcells 679942  5.2    1308461 10.0  1048054  8.0
```

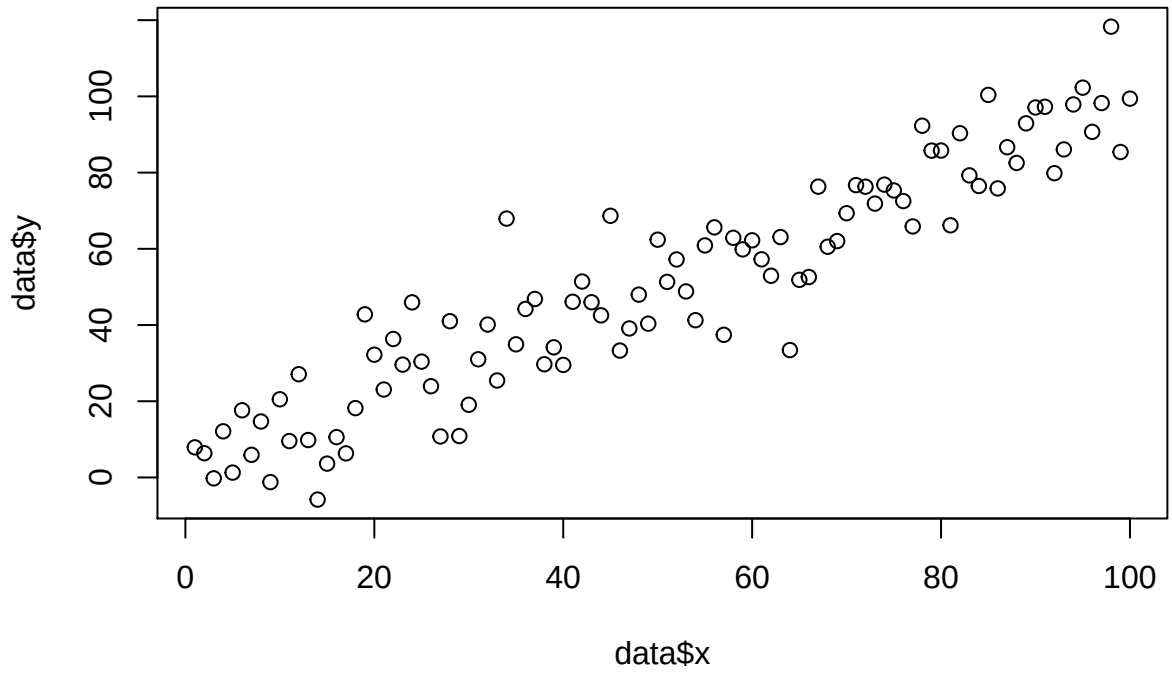
```
x <- 1:100
y <- x + rnorm(100, sd = 10)
data <- data.frame(x,y)
moredata <- data.frame(x = c(101,102), y = c(101,102) + rnorm(2, sd = 10))
```

Plotting & Linear Regression

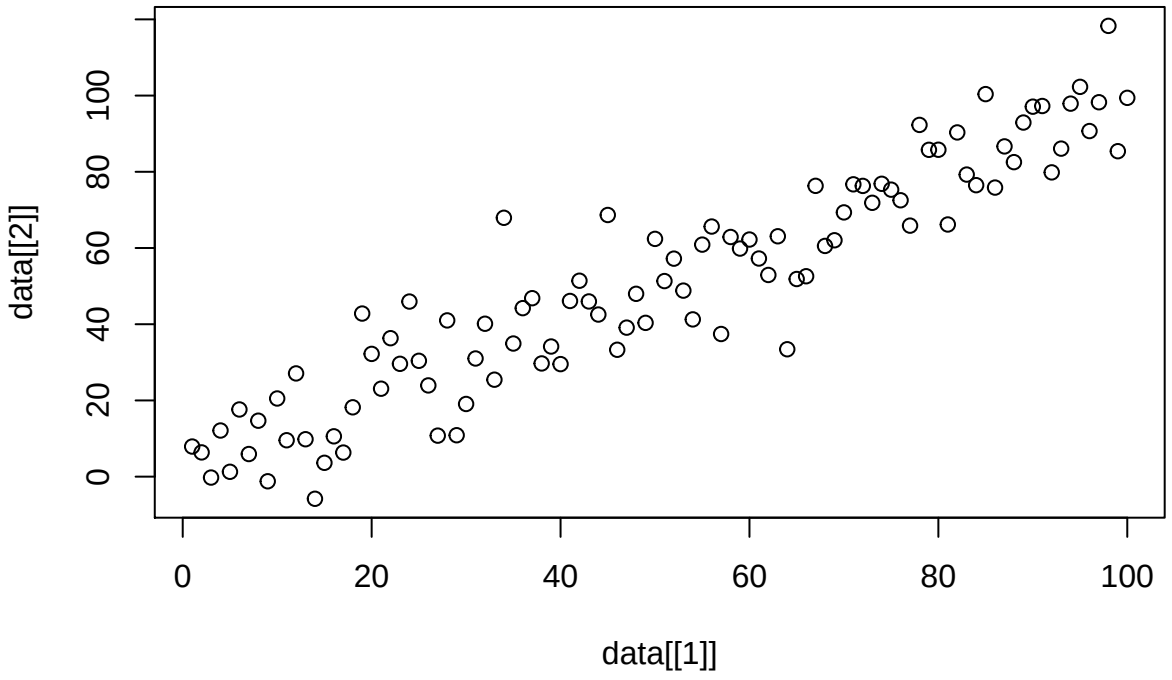
```
cleanup()
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 456428 24.4     940480 50.3   750400 40.1
## Vcells 682618  5.3    1308461 10.0  1048054  8.0
```

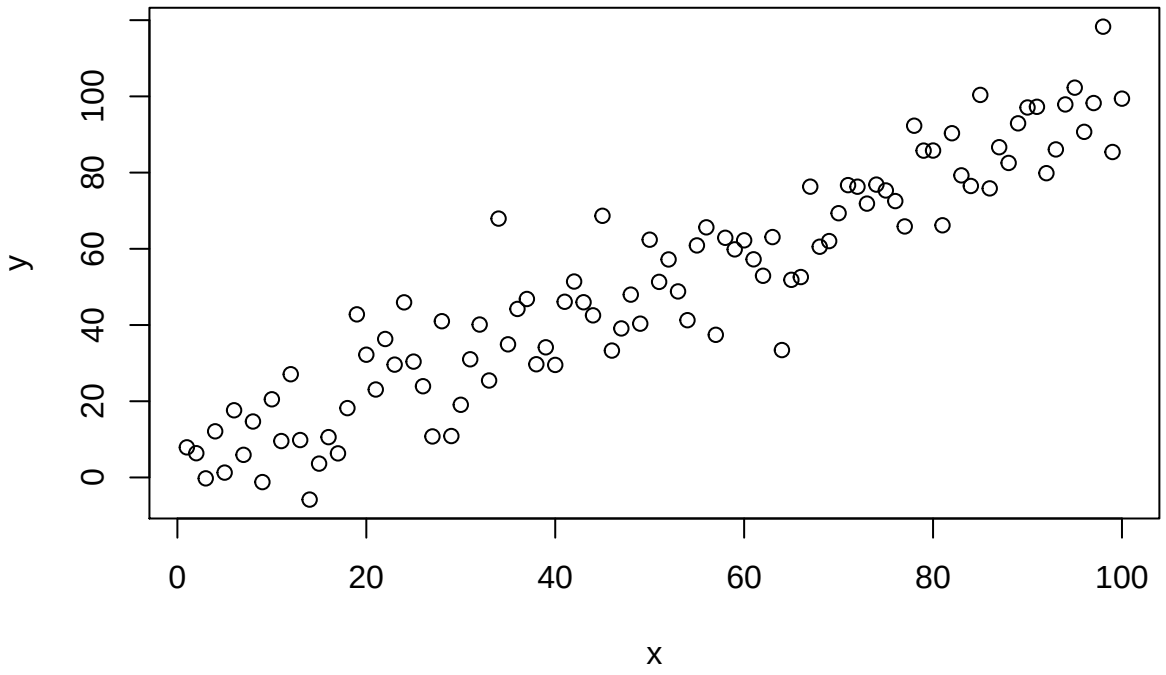
```
x <- 1:100
y <- x + rnorm(100, sd = 10)
data <- data.frame(x,y)

plot(data$x, data$y)
```



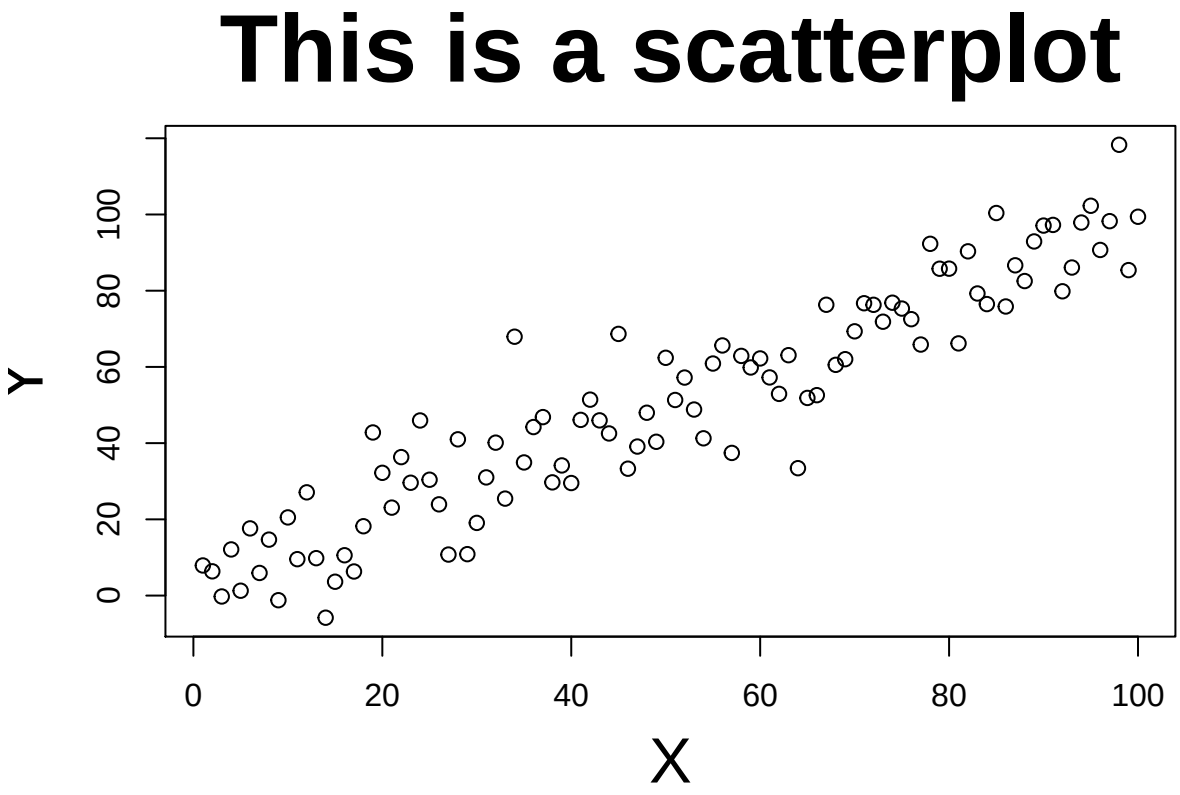

```
plot(data[[1]], data[[2]])
```



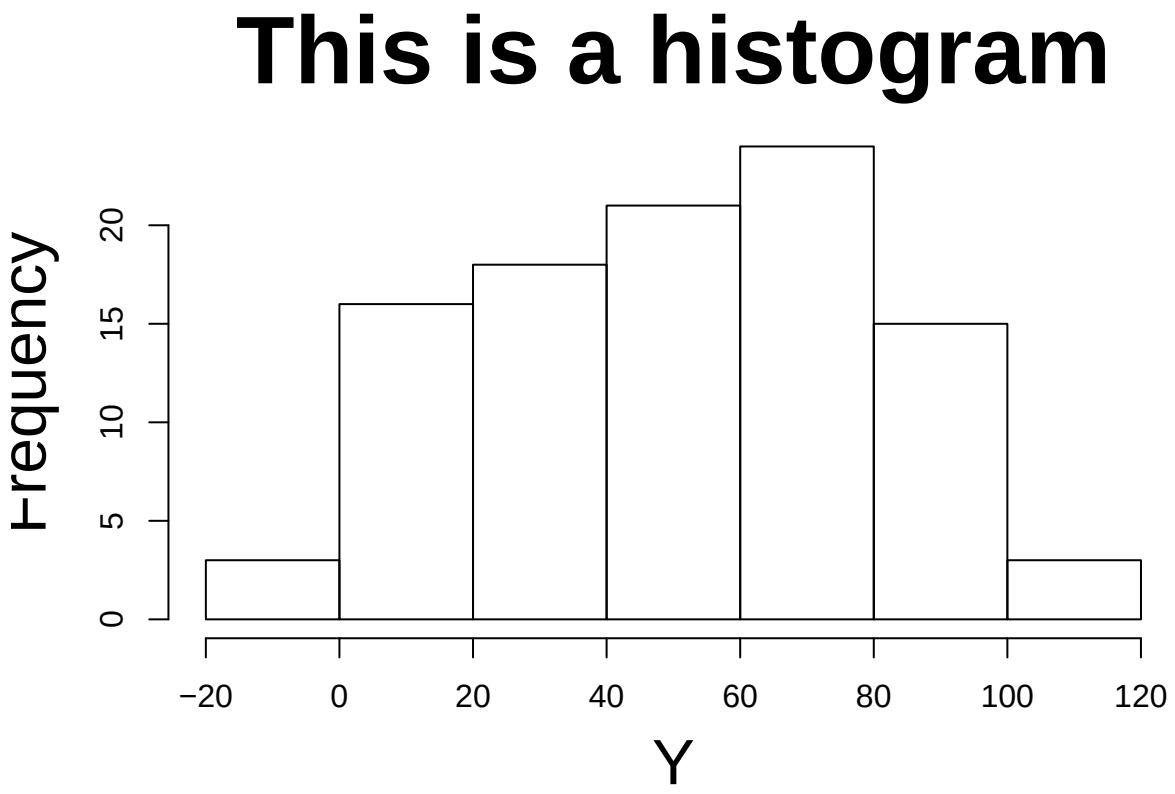

```
with(data, plot(x,y))
```




```r
# plot labels
with(data,
     plot(x, y, xlab = "X", ylab = "Y", main = "This is a scatterplot", cex.main = 3, cex.lab = 2)
     )
```



```r
hist(data$y, xlab = "Y", main = "This is a histogram", cex.main = 3, cex.lab = 2)
```

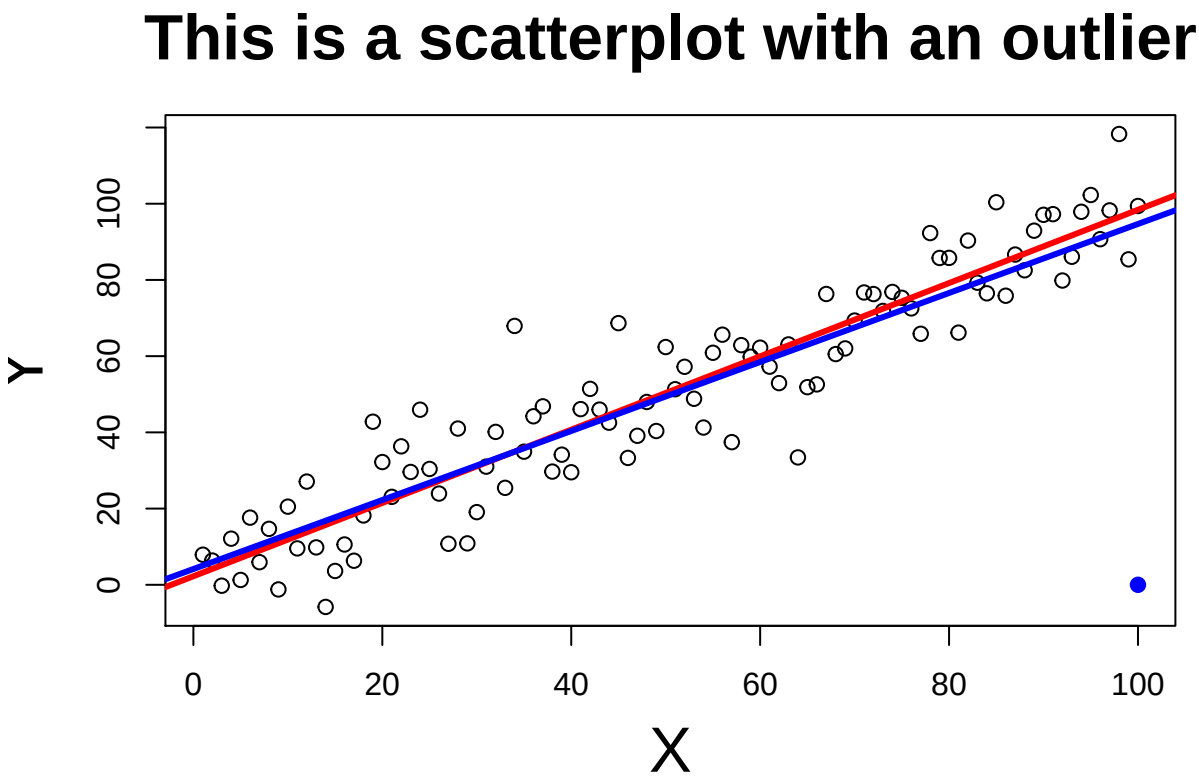



```r
# Simple linear regression
model <- lm(y ~ x, data = data)
names(model)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

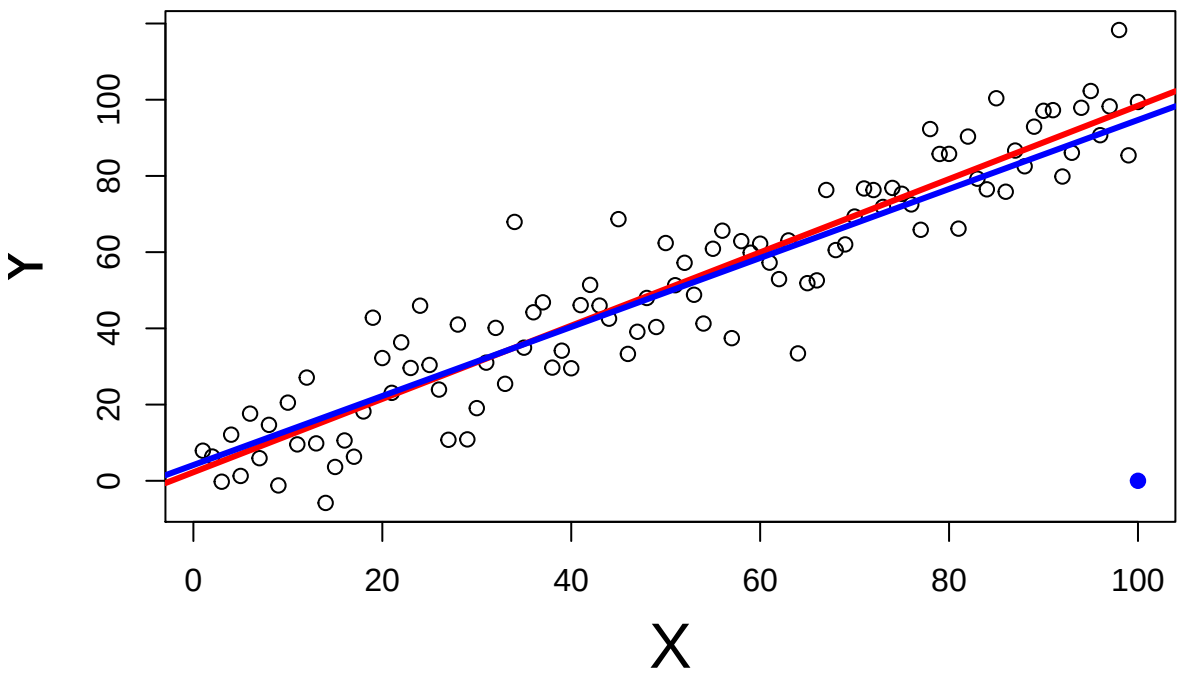
```r
with(data,
     plot(x, y, xlab = "X", ylab = "Y", main = "This is a scatterplot", cex.main = 3, cex.lab = 2)
)

abline(model, col = "red", lwd = 3)
```


```
# Add an outlier
extra.point <- data.frame(x = 100, y = 0)
data2 <- rbind(data, extra.point)

# refit the model
model2 <- lm(y ~ x, data = data2)

with(data,
     plot(x, y, xlab = "X", ylab = "Y", main = "This is a scatterplot with an outlier", cex.main = 2, c
)

# add the outlier
points(extra.point, col = "blue", pch = 19)

# Examine the effect of the outlier on the model
abline(model, col = "red", lwd = 3) # the linear fit before the outlier
abline(model2, col = "blue", lwd = 3) # the fit after the outlier
```



Saving plots

```
# If you have OS X
quartz(height = 7, width = 8)
with(data,
     plot(x, y, xlab = "X", ylab = "Y", main = "This is a scatterplot with an outlier", cex.main = 2, c
)
# add the outlier
points(extra.point, col = "blue", pch = 19)

# Examine the effect of the outlier on the model
abline(model, col = "red", lwd = 3) # the linear fit before the outlier
abline(model2, col = "blue", lwd = 3) # the fit after the outlier
```




```
quartz.save(file = "plot.pdf", type = "pdf")
```

```
## pdf
##   2
```

```
dev.off()
```

```
## pdf
##   2
```

```
# If you don't have a Mac... use dev.new() instead of quartz() and dev.copy() instead of quartz.save()

# Alternatively
?pdf
?png
```

```
# R Markdown packages

install.packages("rmarkdown")
```