Day 1: R introduction

Himanshu Yadav

14.12.2020

```
# Statistical data analysis workshop (organized by YUVA Germany)
# Day 1 - R introduction and graph plotting

########################################

### 0. Getting ready

# Try simple arithmetic operations

2+1      # Press CTRL + ENTER to see the output in the console
```

```
## [1] 3
```

```
2*4
```

```
## [1] 8
```

```
20/10
```

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

```
25/4
```

```
## [1] 6.25
```

```
25%/%4
```

```
## [1] 6
```

```
3^2       # Exponent
```

```
## [1] 9
```

```r
# Why there is "[1]" in front of output everytime

# 1. Vectors

x <- 2
x
```

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

```r
# A sequence of data elements of the same variable type
# Consider a sequence of integers from 1 to 4
x <- c(1,2,3,4)                 # c() is a function for concatenation
x
```

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

```r
x[1]
```

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

```r
x[3]
```

```
## [1] 3
```

```r
x[5] <- 2
x
```

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

```r
x[c(1,5)]
```

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

```r
y <- c("a","b")
y
```

```
## [1] "a" "b"
```

```r
y[2]
```

```
## [1] "b"
```

```r
y[3] <- 2
y
```

```
## [1] "a" "b" "2"
```

```r
y[4]
```

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

```r
z <- c(TRUE,FALSE,TRUE)
z
```

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

```r
z[2]
```

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

```r
# Vector is the fundamental data type in R
# All scalars are considered one-element vectors

x <- 2
x
```

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

```r
x[1]
```

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

```r
y <- "a"
y
```

```
## [1] "a"
```

```r
str(x)
```

```
##  num 2
```

```r
str(y)
```

```
##  chr "a"
```

```r
typeof(y)
```

```
## [1] "character"
```

```r
typeof(x)
```

```
## [1] "double"
```

```r
x <- 1:2
typeof(x)
```

```
## [1] "integer"
```

```r
# 1.1 Simple operations on vectors

# 1.1.1 Adding and deleting vector elements
#need to 'reassign' the vector for adding/deleting elements
#the size is determined at its creation

x <- c(88,5,12,13)
x
```

```
## [1] 88  5 12 13
```

```r
x[1]
```

```
## [1] 88
```

```r
x[2]
```

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

```r
x[1:3] # Give me elements of the vector at positions 1,.,3
```

```
## [1] 88  5 12
```

```r
x[c(1,3)]
```

```
## [1] 88 12
```

```r
# Combine two vectors
x <- c(1,6,5)
x
```

```
## [1] 1 6 5
```

```r
y <- 3:7
y
```

```
## [1] 3 4 5 6 7
```

```r
z <- c(x,y)
z
```

```
## [1] 1 6 5 3 4 5 6 7
```

```r
c(x,65,y)
```

```
## [1]  1  6  5 65  3  4  5  6  7
```

```r
c(x,65,y,1009)
```

```
##  [1]    1    6    5   65    3    4    5    6    7 1009
```

```r
#1.1.2 obtaining the length of a vector

x <- c(1,2,4)
length(x)
```

```
## [1] 3
```

```r
#1.1.3 declarations
#no prior declaration needed to use a variable

z <- 3

#if one needs to access specific element of a 'new' vector then
#R must be told beforehand that the variable is a vector

#W[1] <- 3  #will not work

W <- c(3, 2) # will work
```

```r
#or

W <- vector(length = 2)
W[1] <- 3
W[2] <- 2
W
```

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

```r
#1.1.4 vector arithmetic and logical operations

x <- c(1,2,4)
x + c(5,0,-1)
```

```
## [1] 6 2 3
```

```r
x * c(5,0,-1)
```

```
## [1]  5  0 -4
```

```r
x / c(5,0,-1)
```

```
## [1]  0.2  Inf -4.0
```

```r
x + c(5,0,-1,1) # You get an output 6, 2, 3, 2 Why? What is happening?
```

```
## Warning in x + c(5, 0, -1, 1): longer object length is not a multiple of shorter
## object length
```

```
## [1] 6 2 3 2
```

```r
# R duplicates the first value of vector x in the 4th position

x + c(5,0,-1,1,1) # Now first and second element of x are duplicated
```

```
## Warning in x + c(5, 0, -1, 1, 1): longer object length is not a multiple of
## shorter object length
```

```
## [1] 6 2 3 2 3
```

```r
c(x,x[1:2])+c(5,0,-1,1,1)
```

```
## [1] 6 2 3 2 3
```

```r
#1.1.5 vector indexing

y <- c(1.2,3.9,0.4,0.12)
y
```

```
## [1] 1.20 3.90 0.40 0.12
```

```r
y[c(1,3)]
```

```
## [1] 1.2 0.4
```

```r
y[2:3]
```

```
## [1] 3.9 0.4
```

```r
v <- 3:4
v
```

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

```r
y[v]
```

```
## [1] 0.40 0.12
```

```r
x <- c(4,2,17,5)
x[c(1,1,1)]
```

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

```r
y <- x[c(1,1,3)]
y
```

```
## [1]  4  4 17
```

```r
#-ve subscript mean that we want to exclude the given elements

z <- c(5,12,13)
z[-1]
```

```
## [1] 12 13
```

```r
z[-1:-2]
```

```
## [1] 13
```

```r
4:1
```

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

```r
1:4
```

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

```r
-1:-4
```

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

```r
# a:b gives a sequence that starts with a and increases/decreases by 1
1.1:6
```

```
## [1] 1.1 2.1 3.1 4.1 5.1
```

```r
6:1.1
```

```
## [1] 6 5 4 3 2
```

```r
#1.1.6: seq()

#a more general operator than :
seq(from=12,to=30,by=3)
```

```
## [1] 12 15 18 21 24 27 30
```

```r
seq(from=1,to=30,by=1)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30
```

```r
1:30
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30
```

```r
seq(from=1,to=100,by=10)
```

```
##  [1]  1 11 21 31 41 51 61 71 81 91
```

```r
seq(from=1.1,to=2,length=10)
```

```
##  [1] 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
```

```r
#1.1.7: rep()
x <- rep(8,4)
x
```

```
## [1] 8 8 8 8
```

```r
rep(c(5,12,13), 3)
```

```
## [1]  5 12 13  5 12 13  5 12 13
```

```r
rep(c(5,12,13), each=3)
```

```
## [1]  5  5  5 12 12 12 13 13 13
```

```r
k<-rep(c(5,12,13), each=3)
k
```

```
## [1]  5  5  5 12 12 12 13 13 13
```

```r
str(k)
```

```
##  num [1:9] 5 5 5 12 12 12 13 13 13
```

```r
#1.1.8: NA and NULL values
#NA: missing data
#NULL: value does not exist

x <- c(88,NA,12,168,13)
x
```

```
## [1]  88  NA  12 168  13
```

```r
mean(x)
```

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

```r
mean(x, na.rm=T)
```

```
## [1] 70.25
```

```r
x <- c(88,NULL,12,168,13)
mean(x)
```

```
## [1] 70.25
```

```r
u <- NULL
length(u)
```

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

```r
u <- NA
length(u)
```

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

```r
# 1.2 Analysis using vectors
# 1.2.1 filtering

#subset
(x <- c(6,1:3,NA,12))
```

```
## [1]  6  1  2  3 NA 12
```

```r
x
```

```
## [1]  6  1  2  3 NA 12
```

```r
x[x>5]
```

```
## [1]  6 NA 12
```

```r
subset(x,x>5)
```

```
## [1]  6 12
```

```r
#which
#to find position rather than the number itself
z <- c(5,2,-3,8)
which(z < 6)
```

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

```r
z[which(z < 6)]
```

```
## [1]  5  2 -3
```

```r
which(z*z<8)
```

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

```r
#ifelse()

x <- 1:10
y <- ifelse(x ==4,NA,x)
y
```

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

```r
y <- ifelse(x %% 2 == 0,5,12)
y
```

```
##  [1] 12  5 12  5 12  5 12  5 12  5
```

```r
#%% is the modulus operator: gives the remainder
#of a division
x %% 2
```

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

```r
(x %% 2 == 0)
```

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

```r
y
```

```
##  [1] 12  5 12  5 12  5 12  5 12  5
```

```r
z <- ifelse(x %% 2 == 0,'even','odd')
z
```

```
##  [1] "odd"  "even" "odd"  "even" "odd"  "even" "odd"  "even" "odd"  "even"
```

```r
#2.2 Mean, SDs
y <- c(34,67,209,55,69,100)
mean(y)
```

```
## [1] 89
```

```r
sd(y)
```

```
## [1] 62.58754
```

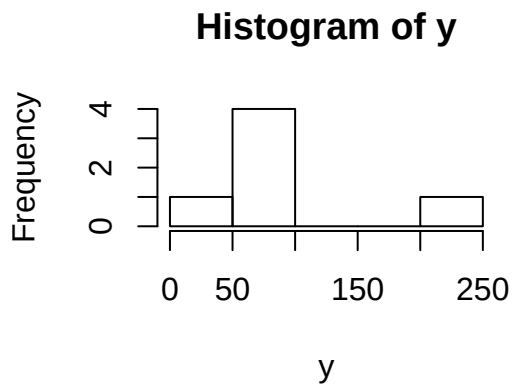
```r
var(y)
```

```
## [1] 3917.2
```

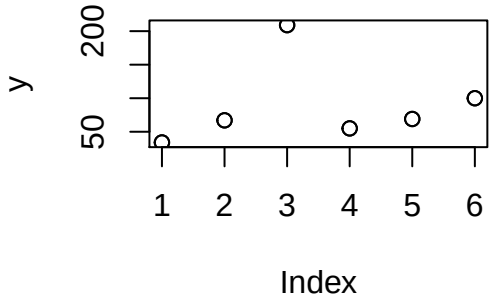
```r
#2.3 Hist, Plots
y
```

```
## [1]  34  67 209  55  69 100
```

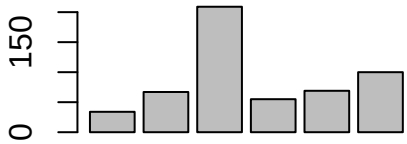
```r
hist(y)
```




```r
plot(y)
points(y)
```




```r
barplot(y)
```




```r
#Exercises

#(1) Manually create a vector with 10 numbers
#     (a) Get all the values that are greater than 5
#     (b) Get all the values that are odd
#     (provide two ways to do this)
#     (c) Get the index of all the odd values
#     (d) Check if all values are even
#     (e) Check if any value is even

#(2) Manually create a vector with 5 numbers
#     (a) Now create another vector where the 1st element is 1 more than the length of
#         and has a length of 10. The max value of this new vector should be 20
#     (b) What is the difference between the length of the two vectors?
```

```r
# 1.3 Factors

x <- c(5,12,13,12)
x
```

```
## [1]  5 12 13 12
```

```r
xf <- factor(x)
xf
```

```
## [1] 5  12 13 12
## Levels: 5 12 13
```

```r
str(xf)
```

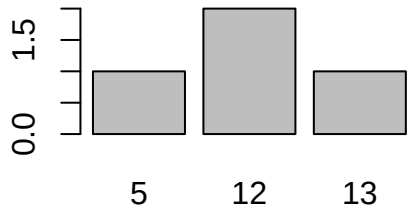
```
##  Factor w/ 3 levels "5","12","13": 1 2 3 2
```

```r
xf +x
```

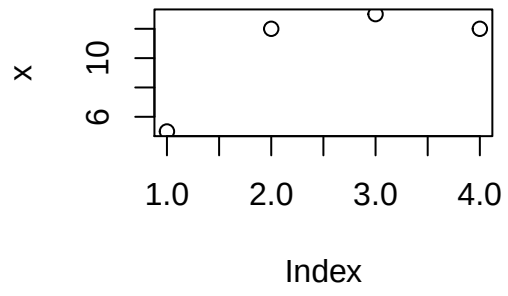
```
## Warning in Ops.factor(xf, x): '+' not meaningful for factors
```

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

```r
plot(xf)
```




```r
plot(x)
```




```
#####################################################################

# 2. Matrix
x <- matrix(1:4,nrow=2,ncol = 2)
x[1,]
```

```
## [1] 1 3
```

```
x[,1]
```

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

```
x[2,2]
```

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

```
y <- matrix(5:6,nrow=2,ncol = 1)
y
```

```
##      [,1]
## [1,]    5
## [2,]    6
```

```
# 2.1 Matrix operations
# Matrix multiplication
#x%*%y
#x*y
# Multiplication by a scalar
#x*2
```

```r
#Element wise multiplication
#x[,2]*y

# Matrix transpose
t(x)
```

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

```r
# Addition
#x+y
#x[,2]+y
y <- matrix(0:3,nrow = 2,ncol = 2)
y
```

```
##      [,1] [,2]
## [1,]    0    2
## [2,]    1    3
```

```r
x+y
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    3    7
```

```r
# Substraction
x-y
```

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

```r
# Matrix cross product
crossprod(x,y)
```

```
##      [,1] [,2]
## [1,]    2    8
## [2,]    4   18
```

```r
t(x)%*%y
```

```
##      [,1] [,2]
## [1,]    2    8
## [2,]    4   18
```

```r
crossprod(x,y)==t(x)%*%y
```

```
##      [,1] [,2]
## [1,] TRUE TRUE
## [2,] TRUE TRUE
```

```r
# Determinant

det(x)
```

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

```r
############################################################
# 3. Data frames
# Data-frames are collection of equal-length vectors in two dimesnions
# A data frame is like a matrix, but here each column may have
#different variable type.
#Technically, a data frame is a list, with the components of that list being
#equal-length vectors

#3.1 Representing and accessing data frames
kids <- c("Manish", "Abhay")
ages <- c(12, 10)
d <- data.frame(kids, ages)
d
```

```
##     kids ages
## 1 Manish   12
## 2  Abhay   10
```

```r
str(d)
```

```
## 'data.frame':    2 obs. of  2 variables:
##  $ kids: Factor w/ 2 levels "Abhay","Manish": 2 1
##  $ ages: num  12 10
```

```r
d <- data.frame(kids, ages, stringsAsFactors = FALSE)
d
```

```
##     kids ages
## 1 Manish   12
## 2  Abhay   10
```

```r
str(d)
```

```
## 'data.frame':    2 obs. of  2 variables:
##  $ kids: chr  "Manish" "Abhay"
##  $ ages: num  12 10
```

```r
kids <- c("Manish")
ages <- c(12, 10)
df <- data.frame(kids, ages)
df
```

```
##     kids ages
## 1 Manish   12
## 2 Manish   10
```

```r
#Accessing values

d
```

```
##     kids ages
## 1 Manish   12
## 2  Abhay   10
```

```r
d[1]
```

```
##     kids
## 1 Manish
## 2  Abhay
```

```r
d[2]
```

```
##   ages
## 1   12
## 2   10
```

```r
d[[1]]
```

```
## [1] "Manish" "Abhay"
```

```r
d[[2]]
```

```
## [1] 12 10
```

```r
d$kids
```

```
## [1] "Manish" "Abhay"
```

```r
d$ages
```

```
## [1] 12 10
```

```r
d[,1]
```

```
## [1] "Manish" "Abhay"
```

```r
studentID <- c(101:110)
marks <- c(40,12,34,56,78,90,87,56,89,60)
results <- data.frame(studentID,marks)
results
```

```
##    studentID marks
## 1        101    40
## 2        102    12
## 3        103    34
## 4        104    56
## 5        105    78
## 6        106    90
## 7        107    87
## 8        108    56
## 9        109    89
## 10       110    60
```

```r
# Average marks in class
mean(results$marks)
```

```
## [1] 60.2
```

```r
# How many students failed, marks<40
which(results$marks<40)
```

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

```r
length(which(results$marks<40))
```

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

```r
results$marks<40
```

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

```r
any(results$marks<40)  # Is "any" element of the vector TRUE
```

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

```r
all(results$marks<40) # Are all values of the vector true
```

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

```r
# What are IDs of students who failed?
which(results$marks<40)
```

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

```r
results$studentID[which(results$marks<40)]
```

```
## [1] 102 103
```

```r
# Filtering
# Only show results of passed students
subset(results,marks>=40)
```

```
##    studentID marks
## 1        101    40
## 4        104    56
## 5        105    78
## 6        106    90
## 7        107    87
## 8        108    56
## 9        109    89
## 10       110    60
```

```r
results[which(results$marks>=40),]
```

```
##    studentID marks
## 1        101    40
## 4        104    56
## 5        105    78
## 6        106    90
## 7        107    87
## 8        108    56
## 9        109    89
## 10       110    60
```

```r
# Average marks of students who passed the exam
subset(results,marks>=40)$marks
```

```
## [1] 40 56 78 90 87 56 89 60
```

```r
mean(subset(results,marks>=40)$marks)
```

```
## [1] 69.5
```

```r
#3.2: Reading from external files as data frames

#getwd() tells you where you are

new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
str(new.d)
```

```
## 'data.frame':    3 obs. of  4 variables:
##  $ Student: Factor w/ 3 levels "Abhay","Kavita",..: 1 3 2
##  $ Exam1  : int  100 57 100
##  $ Exam2  : int  98 75 99
##  $ Age    : int  10 12 11
```

```r
new.d$Student <- as.character(new.d$Student)
str(new.d)
```

```
## 'data.frame':    3 obs. of  4 variables:
##  $ Student: chr  "Abhay" "Manish" "Kavita"
##  $ Exam1  : int  100 57 100
##  $ Exam2  : int  98 75 99
##  $ Age    : int  10 12 11
```

```r
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, stringsAsFactors = FALS
str(new.d)
```

```
## 'data.frame':    3 obs. of  4 variables:
##  $ Student: chr  "Abhay" "Manish" "Kavita"
##  $ Exam1  : int  100 57 100
##  $ Exam2  : int  98 75 99
##  $ Age    : int  10 12 11
```

```r
# 3.2.1 'header' attribute
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
new.d <- read.table(file = "data-frame-file.dat", sep="\t")
new.d
```

```
##        V1    V2    V3  V4
## 1 Student Exam1 Exam2 Age
## 2   Abhay   100    98  10
## 3  Manish    57    75  12
## 4  Kavita   100    99  11
```

```r
new.d <- new.d[-c(1),]
new.d
```

```
##       V1  V2 V3 V4
## 2  Abhay 100 98 10
## 3 Manish  57 75 12
## 4 Kavita 100 99 11
```

```r
#I can add column names manually
colnames(new.d) <- c("Student","Exam1","Exam2", "Age")
new.d
```

```
##   Student Exam1 Exam2 Age
## 2   Abhay   100    98  10
## 3  Manish    57    75  12
## 4  Kavita   100    99  11
```

```r
# 3.2.2 'sep' attribute
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep=" ")
new.d
```

```
##   Student.Exam1.Exam2.Age
## 1         Abhay\t100\t98\t10
## 2         Manish\t57\t75\t12
## 3        Kavita\t100\t99\t11
```

```r
# 3.2.3 Accessing columns and rows
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
# Info
head(new.d)
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
dim(new.d)
```

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

```r
new.d$Student
```

```
## [1] Abhay  Manish Kavita
## Levels: Abhay Kavita Manish
```

```r
new.d$Exam1
```

```
## [1] 100  57 100
```

```r
# Columns
new.d[,c(1,2)]
```

```
##   Student Exam1
## 1   Abhay   100
## 2  Manish    57
## 3  Kavita   100
```

```r
new.d[,c(1,3)]
```

```
##   Student Exam2
## 1   Abhay    98
## 2  Manish    75
## 3  Kavita    99
```

```r
# Rows
new.d[c(1,2),]
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
```

```r
new.d[c(1,3),]
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 3  Kavita   100    99  11
```

```r
# 3.3 Removing rows and columns
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
new.d[-c(1),]
```

```
##   Student Exam1 Exam2 Age
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
new.new.d <- new.d[-c(1),]
new.new.d
```

```
##   Student Exam1 Exam2 Age
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
new.d[,-c(2)]
```

```
##   Student Exam2 Age
## 1   Abhay    98  10
## 2  Manish    75  12
## 3  Kavita    99  11
```

```r
new.new.d <- new.d[,-c(2)]
new.new.d
```

```
##   Student Exam2 Age
## 1   Abhay    98  10
## 2  Manish    75  12
## 3  Kavita    99  11
```

```
# 3.4 complete.cases function
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```
new.d[complete.cases(new.d),]
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```
new.d[!complete.cases(new.d),]
```

```
## [1] Student Exam1   Exam2   Age
## <0 rows> (or 0-length row.names)
```

```
# if data has NA values i.e., missing values
new.d$Exam1[2] <- NA
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    NA    75  12
## 3  Kavita   100    99  11
```

```
new.d[complete.cases(new.d),]
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 3  Kavita   100    99  11
```

```
new.d[!complete.cases(new.d),]
```

```
##   Student Exam1 Exam2 Age
## 2  Manish    NA    75  12
```

```r
new.d <- new.d[complete.cases(new.d),]

# 3.5 Operations on rows and columns
new.d <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
new.d$Exam1+new.d$Exam2
```

```
## [1] 198 132 199
```

```r
new.d$Exam1-new.d$Exam2
```

```
## [1]   2 -18   1
```

```r
mean(new.d$Exam1)
```

```
## [1] 85.66667
```

```r
sd(new.d$Exam1)
```

```
## [1] 24.82606
```

```r
max(new.d$Exam1)
```

```
## [1] 100
```

```r
min(new.d$Exam1)
```

```
## [1] 57
```

```r
is.numeric(new.d$Student)
```

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

```r
is.numeric(new.d$Exam1)
```

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

```r
# 3.6: Adding variables i.e., columns to the dataframes
new.d
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
# You can add a vector of same length
totalMarks <- c(198,132,199)
new.d$total <- totalMarks
new.d
```

```
##   Student Exam1 Exam2 Age total
## 1   Abhay   100    98  10   198
## 2  Manish    57    75  12   132
## 3  Kavita   100    99  11   199
```

```r
totalMarks <- new.d$Exam1+new.d$Exam2
totalMarks
```

```
## [1] 198 132 199
```

```r
new.d$total <- totalMarks
new.d
```

```
##   Student Exam1 Exam2 Age total
## 1   Abhay   100    98  10   198
## 2  Manish    57    75  12   132
## 3  Kavita   100    99  11   199
```

```r
new.d$total <- new.d$Exam1+new.d$Exam2

# 3.7 ifelse function
grade <- c("A","B","C")
new.d$grade <- grade
new.d
```

```
##   Student Exam1 Exam2 Age total grade
## 1   Abhay   100    98  10   198     A
## 2  Manish    57    75  12   132     B
## 3  Kavita   100    99  11   199     C
```

```r
new.d$grade <- ifelse(new.d$Exam1>80,"A","B")
new.d
```

```
##   Student Exam1 Exam2 Age total grade
## 1   Abhay   100    98  10   198     A
## 2  Manish    57    75  12   132     B
## 3  Kavita   100    99  11   199     A
```

```r
new.d$grade <- ifelse(new.d$Exam1>80,ifelse(new.d$Exam2>80,"A","B"),"B")
new.d
```

```
##   Student Exam1 Exam2 Age total grade
## 1   Abhay   100    98  10   198     A
## 2  Manish    57    75  12   132     B
## 3  Kavita   100    99  11   199     A
```

```r
#Exercises

#(1) Create a data frame as shown below

# Stimuli_id cond rt
# 1           a   300
# 1           b   420
# 2           a   330
# 2           b   412
# 3           a   250
# 3           b   523

# (2) Find the mean RT for each condition i.e., condition 'a' and condition 'b'
# Your output should look like:
# cond  mean.rt
#   a   293.33
#   b   451.66

# (3) Draw a barplot showing mean RT for each condition

# Load data from file 'dep_length.dat'
# (4) Compute the following
# (a) Number of variables and observations in the data
# (b) Number of observations having length=0
# (c) Overall Mean length, mean length for each direction
# (d) Draw histogram for lengths
# (e) Draw barplot showing mean length for each direction
# (f) Add a column "type" in the dataframe which has
#     value "adjacent" if length is 0, else "apart"
```

```r
# (g) Which of the directions has more number of higher lengths i.e., length>7

##############################################

# 4. Data frame manipulation

# 4.1 Combining multiple data frames

df1 <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
df1
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
df2 <- read.table(file = "data-frame-file.dat", header = TRUE, sep="\t")
df2
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
# rbind function
df <- rbind(df1,df2)
df
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
## 4   Abhay   100    98  10
## 5  Manish    57    75  12
## 6  Kavita   100    99  11
```

```r
# cbind
cbind(df1,df2)
```

```
##   Student Exam1 Exam2 Age Student Exam1 Exam2 Age
## 1   Abhay   100    98  10   Abhay   100    98  10
## 2  Manish    57    75  12  Manish    57    75  12
## 3  Kavita   100    99  11  Kavita   100    99  11
```

```r
examdiff <- df1$Exam1 - df2$Exam2
df1 <- cbind(df1, examdiff)
df1
```

```
##   Student Exam1 Exam2 Age examdiff
## 1   Abhay   100    98  10        2
## 2  Manish    57    75  12      -18
## 3  Kavita   100    99  11        1
```

```r
# merge function
df2 <- df2[,c(1,4)]
df1 <- df1[,c(1,2,3)]
df2
```

```
##   Student Age
## 1   Abhay  10
## 2  Manish  12
## 3  Kavita  11
```

```r
df1
```

```
##   Student Exam1 Exam2
## 1   Abhay   100    98
## 2  Manish    57    75
## 3  Kavita   100    99
```

```r
merge(df1,df2)
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Kavita   100    99  11
## 3  Manish    57    75  12
```

```r
df1$Age <- df2$Age[match(df1$Student, df2$Student)]
df1
```

```
##   Student Exam1 Exam2 Age
## 1   Abhay   100    98  10
## 2  Manish    57    75  12
## 3  Kavita   100    99  11
```

```r
####################################

# 4.2 "dplyr" package
```

```r
# COVID-19 data
# Reference: https://data.europa.eu/euodp/en/data/dataset/covid-19-coronavirus-data
# Description: daywise cases, deaths per country, total population

covid <- read.table("COVID19data.csv", header = TRUE,sep=",")
covid <- covid[,-1]
head(covid)
```

```
##      dateRep day month year cases deaths     country popData2019 continentExp
## 1 11/12/2020  11    12 2020    63     10 Afghanistan    38041757          Asia
## 2 10/12/2020  10    12 2020   202     16 Afghanistan    38041757          Asia
## 3 09/12/2020   9    12 2020   135     13 Afghanistan    38041757          Asia
## 4 08/12/2020   8    12 2020   200      6 Afghanistan    38041757          Asia
## 5 07/12/2020   7    12 2020   210     26 Afghanistan    38041757          Asia
## 6 06/12/2020   6    12 2020   234     10 Afghanistan    38041757          Asia
```

```r
dim(covid)
```

```
## [1] 61261     9
```

```r
library(dplyr)

# 4.2.1 select function

# Suppose I only want number of cases on every date, and want to throw rest columns
# Syntax is select(dataframeName,selectedColumnNames)
# Alternative syntax: dataframeName %>% select(selectedColumnNames)

d.cases <- select(covid,cases)
head(d.cases)
```

```
##   cases
## 1    63
## 2   202
## 3   135
## 4   200
## 5   210
## 6   234
```

```r
d.cases <- select(covid, dateRep, cases)
head(d.cases)
```

```
##      dateRep cases
## 1 11/12/2020    63
```

```
## 2 10/12/2020   202
## 3 09/12/2020   135
## 4 08/12/2020   200
## 5 07/12/2020   210
## 6 06/12/2020   234
```

```
# Select date, country, cases
d.cases <- select(covid, dateRep, country, cases)
head(d.cases)
```

```
##      dateRep     country cases
## 1 11/12/2020 Afghanistan    63
## 2 10/12/2020 Afghanistan   202
## 3 09/12/2020 Afghanistan   135
## 4 08/12/2020 Afghanistan   200
## 5 07/12/2020 Afghanistan   210
## 6 06/12/2020 Afghanistan   234
```

```
# Alternatively
d.cases <- covid %>% select(dateRep,country,cases)
head(d.cases)
```

```
##      dateRep     country cases
## 1 11/12/2020 Afghanistan    63
## 2 10/12/2020 Afghanistan   202
## 3 09/12/2020 Afghanistan   135
## 4 08/12/2020 Afghanistan   200
## 5 07/12/2020 Afghanistan   210
## 6 06/12/2020 Afghanistan   234
```

```
# 4.2.2 filter function

# Suppose I want to select only those rows
# where number of cases in day is greater than 2000
# If you can recall, you can do this using subset() function
highRate <- subset(covid, cases>2000)
head(highRate)
```

```
##         dateRep day month year cases deaths   country popData2019 continentExp
## 2023 11/12/2020  11    12 2020  6994    209 Argentina    44780675       America
## 2024 10/12/2020  10    12 2020  5303    213 Argentina    44780675       America
## 2025 09/12/2020   9    12 2020  3610    121 Argentina    44780675       America
## 2026 08/12/2020   8    12 2020  3216    118 Argentina    44780675       America
## 2027 07/12/2020   7    12 2020  3278    138 Argentina    44780675       America
## 2028 06/12/2020   6    12 2020  5201    120 Argentina    44780675       America
```

```r
# You can also do this using filter() function from "dplyr"
highRate <- filter(covid, cases>2000)
head(highRate)
```

```
##      dateRep day month year cases deaths   country popData2019 continentExp
## 1 11/12/2020  11    12 2020  6994    209 Argentina    44780675       America
## 2 10/12/2020  10    12 2020  5303    213 Argentina    44780675       America
## 3 09/12/2020   9    12 2020  3610    121 Argentina    44780675       America
## 4 08/12/2020   8    12 2020  3216    118 Argentina    44780675       America
## 5 07/12/2020   7    12 2020  3278    138 Argentina    44780675       America
## 6 06/12/2020   6    12 2020  5201    120 Argentina    44780675       America
```

```r
#Alternatively
highRate <- covid %>% filter(cases>2000)
head(highRate)
```

```
##      dateRep day month year cases deaths   country popData2019 continentExp
## 1 11/12/2020  11    12 2020  6994    209 Argentina    44780675       America
## 2 10/12/2020  10    12 2020  5303    213 Argentina    44780675       America
## 3 09/12/2020   9    12 2020  3610    121 Argentina    44780675       America
## 4 08/12/2020   8    12 2020  3216    118 Argentina    44780675       America
## 5 07/12/2020   7    12 2020  3278    138 Argentina    44780675       America
## 6 06/12/2020   6    12 2020  5201    120 Argentina    44780675       America
```

```r
#Show columns dateRep, country and cases only, when no. of cases>2000
#Apply both filter and select function
highRate <- covid %>% filter(cases>2000) %>% select(dateRep,country,cases)
head(highRate)
```

```
##      dateRep   country cases
## 1 11/12/2020 Argentina  6994
## 2 10/12/2020 Argentina  5303
## 3 09/12/2020 Argentina  3610
## 4 08/12/2020 Argentina  3216
## 5 07/12/2020 Argentina  3278
## 6 06/12/2020 Argentina  5201
```

```r
#List all the countries that had cases>50000 on any day
#in the last month
str(covid)
```

```
## 'data.frame':    61261 obs. of  9 variables:
##  $ dateRep     : Factor w/ 347 levels "01/01/2020","01/02/2020",..: 132 120 108 96 84
##  $ day         : int  11 10 9 8 7 6 5 4 3 2 ...
```

```
##  $ month       : int  12 12 12 12 12 12 12 12 12 12 ...
##  $ year        : int  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 ...
##  $ cases       : int  63 202 135 200 210 234 235 119 202 400 ...
##  $ deaths      : int  10 16 13 6 26 10 18 5 19 48 ...
##  $ country     : Factor w/ 214 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ popData2019 : int  38041757 38041757 38041757 38041757 38041757 38041757 38041757
##  $ continentExp: Factor w/ 6 levels "Africa","America",..: 3 3 3 3 3 3 3 3 3 3 ...
```

```
highRate <- covid %>% filter(cases>50000&month==11)
# & means 'AND'
# | means 'OR'
# Try: TRUE & FALSE, TRUE | FALSE
head(highRate)
```

```
##      dateRep day month year cases deaths country popData2019 continentExp
## 1 29/11/2020  29    11 2020 51922    587  Brazil   211049519      America
## 2 08/11/2020   8    11 2020 86852    304  France    67012883       Europe
## 3 07/11/2020   7    11 2020 60486    828  France    67012883       Europe
## 4 06/11/2020   6    11 2020 58046    363  France    67012883       Europe
## 5 03/11/2020   3    11 2020 52518    416  France    67012883       Europe
## 6 07/11/2020   7    11 2020 50356    577   India  1366417756         Asia
```

```
head(highRate$country)
```

```
## [1] Brazil France France France France India
## 214 Levels: Afghanistan Albania Algeria Andorra Angola ... Zimbabwe
```

```
unique(highRate$country)
```

```
## [1] Brazil                  France                  India
## [4] Spain                   United_States_of_America
## 214 Levels: Afghanistan Albania Algeria Andorra Angola ... Zimbabwe
```

```
# 4.2.3 summarise function

# Now suppose I want to calculate total number of cases in each country

totalCases <- covid %>% group_by(country) %>% summarise(total.cases=sum(cases))
head(totalCases)
```

```
## # A tibble: 6 x 2
##   country     total.cases
##   <fct>             <int>
## 1 Afghanistan       48116
```

```
## 2 Albania           46061
## 3 Algeria           90579
## 4 Andorra            7190
## 5 Angola            15925
## 6 Anguilla             10
```

```
totalCases <- as.data.frame(totalCases)
totalCases
```

```
##                                     country total.cases
## 1                                 Afghanistan       48116
## 2                                     Albania       46061
## 3                                     Algeria       90579
## 4                                     Andorra        7190
## 5                                      Angola       15925
## 6                                    Anguilla          10
## 7                          Antigua_and_Barbuda         146
## 8                                   Argentina     1482216
## 9                                     Armenia      146317
## 10                                      Aruba        5011
## 11                                  Australia       28000
## 12                                    Austria      311067
## 13                                 Azerbaijan      162774
## 14                                    Bahamas        7585
## 15                                    Bahrain       88632
## 16                                 Bangladesh      485965
## 17                                   Barbados         291
## 18                                    Belarus      154392
## 19                                    Belgium      600261
## 20                                     Belize        8805
## 21                                      Benin        3090
## 22                                    Bermuda         364
## 23                                     Bhutan         434
## 24                                    Bolivia      146385
## 25 Bonaire, Saint Eustatius and Saba         170
## 26                Bosnia_and_Herzegovina       98603
## 27                                   Botswana       12058
## 28                                     Brazil     6781799
## 29                 British_Virgin_Islands          75
## 30                      Brunei_Darussalam         152
## 31                               Bulgaria      174568
## 32                           Burkina_Faso        3579
## 33                                Burundi         721
## 34                               Cambodia         356
## 35                               Cameroon       25087
## 36                                 Canada      442069
```

```
## 37                                    Cape_Verde       11203
## 38  Cases_on_an_international_conveyance_Japan          696
## 39                                Cayman_Islands          292
## 40                     Central_African_Republic         4932
## 41                                          Chad         1739
## 42                                         Chile       566440
## 43                                         China        91972
## 44                                      Colombia      1399911
## 45                                       Comoros          617
## 46                                         Congo         6049
## 47                                    Costa_Rica       149815
## 48                                  Cote_dIvoire        21590
## 49                                       Croatia       163992
## 50                                          Cuba         9181
## 51                                      CuraÃ§ao         3325
## 52                                        Cyprus        14052
## 53                                       Czechia       569205
## 54            Democratic_Republic_of_the_Congo        13996
## 55                                       Denmark       100489
## 56                                      Djibouti         5717
## 57                                      Dominica           85
## 58                             Dominican_Republic       151721
## 59                                       Ecuador       200379
## 60                                         Egypt       120147
## 61                                   El_Salvador        40741
## 62                                Equatorial_Guinea         5183
## 63                                       Eritrea          656
## 64                                       Estonia        16598
## 65                                      Eswatini         6633
## 66                                      Ethiopia       115360
## 67                  Falkland_Islands_(Malvinas)           19
## 68                                Faroe_Islands          519
## 69                                          Fiji           44
## 70                                       Finland        29572
## 71                                        France      2337966
## 72                                French_Polynesia        15535
## 73                                         Gabon         9300
## 74                                        Gambia         3779
## 75                                       Georgia       183099
## 76                                       Germany      1272078
## 77                                         Ghana        52738
## 78                                     Gibraltar         1069
## 79                                        Greece       121253
## 80                                     Greenland           19
## 81                                       Grenada           43
## 82                                          Guam         7052
```

```
## 83                 Guatemala      127786
## 84                  Guernsey         289
## 85                    Guinea       13368
## 86             Guinea_Bissau        2444
## 87                    Guyana        5811
## 88                     Haiti        9465
## 89                  Holy_See          26
## 90                  Honduras      113207
## 91                   Hungary      271200
## 92                   Iceland        5524
## 93                     India     9796769
## 94                 Indonesia      598933
## 95                      Iran     1083023
## 96                      Iraq      571253
## 97                   Ireland       75203
## 98                Isle_of_Man         370
## 99                    Israel      352860
## 100                    Italy     1787147
## 101                  Jamaica       11443
## 102                    Japan      171542
## 103                   Jersey        1555
## 104                   Jordan      253121
## 105               Kazakhstan      183630
## 106                    Kenya       90305
## 107                   Kosovo       45004
## 108                   Kuwait      145495
## 109                Kyrgyzstan       76718
## 110                     Laos          41
## 111                   Latvia       23706
## 112                  Lebanon      142187
## 113                  Lesotho        2178
## 114                  Liberia        1676
## 115                    Libya       89183
## 116            Liechtenstein        1477
## 117                Lithuania       86949
## 118               Luxembourg       40037
## 119               Madagascar       17638
## 120                   Malawi        6053
## 121                 Malaysia       78499
## 122                 Maldives       13308
## 123                     Mali        5576
## 124                    Malta       10884
## 125          Marshall_Islands           4
## 126                Mauritania       10268
## 127                 Mauritius         515
## 128                    Mexico     1217126
```

```
## 129                                 Moldova      122685
## 130                                  Monaco         657
## 131                                Mongolia         905
## 132                              Montenegro       40165
## 133                              Montserrat          13
## 134                                 Morocco      391529
## 135                              Mozambique       16521
## 136                                 Myanmar      104487
## 137                                 Namibia       15773
## 138                                   Nepal      245650
## 139                             Netherlands      584980
## 140                           New_Caledonia          36
## 141                             New_Zealand        1736
## 142                               Nicaragua        5887
## 143                                   Niger        2126
## 144                                 Nigeria       71344
## 145                          North_Macedonia       70883
## 146                 Northern_Mariana_Islands         113
## 147                                  Norway       39524
## 148                                    Oman      125669
## 149                                Pakistan      432327
## 150                               Palestine      119420
## 151                                  Panama      185424
## 152                          Papua_New_Guinea         684
## 153                                Paraguay       90958
## 154                                    Peru      979111
## 155                             Philippines      445540
## 156                                  Poland     1102096
## 157                                Portugal      335207
## 158                             Puerto_Rico      101763
## 159                                   Qatar      140516
## 160                                 Romania      539107
## 161                                  Russia     2569126
## 162                                  Rwanda        6349
## 163                  Saint_Kitts_and_Nevis          26
## 164                             Saint_Lucia         274
## 165          Saint_Vincent_and_the_Grenadines          91
## 166                              San_Marino        1868
## 167                  Sao_Tome_and_Principe        1009
## 168                           Saudi_Arabia      359415
## 169                                Senegal       16766
## 170                                 Serbia      249224
## 171                              Seychelles         187
## 172                             Sierra_Leone        2435
## 173                                Singapore       58297
## 174                             Sint_Maarten        1201
```

```
## 175                                Slovakia      124921
## 176                                Slovenia       91921
## 177                         Solomon_Islands          17
## 178                                 Somalia        4579
## 179                            South_Africa      836764
## 180                             South_Korea       40786
## 181                             South_Sudan        3181
## 182                                   Spain     1720056
## 183                               Sri_Lanka       30613
## 184                                   Sudan       20468
## 185                                Suriname        5337
## 186                                  Sweden      312728
## 187                             Switzerland      367218
## 188                                   Syria        8787
## 189                                  Taiwan         725
## 190                               Tajikistan       12588
## 191                                 Thailand        4180
## 192                              Timor_Leste          31
## 193                                    Togo        3182
## 194                      Trinidad_and_Tobago        6833
## 195                                  Tunisia      107814
## 196                                   Turkey      558517
## 197                  Turks_and_Caicos_islands         768
## 198                                   Uganda       25730
## 199                                  Ukraine      858714
## 200                      United_Arab_Emirates      181405
## 201                           United_Kingdom     1787783
## 202                  United_Republic_of_Tanzania         509
## 203                     United_States_of_America    15616381
## 204                United_States_Virgin_Islands        1733
## 205                                  Uruguay        8487
## 206                                Uzbekistan       74664
## 207                                   Vanuatu           1
## 208                                 Venezuela      106280
## 209                                   Vietnam        1385
## 210                          Wallis_and_Futuna           3
## 211                             Western_Sahara         766
## 212                                     Yemen        2081
## 213                                    Zambia       18091
## 214                                  Zimbabwe       11081
```

```
# Total number of cases per month in each country
totalCases <- covid %>% group_by(country,year,month) %>% summarise(total.cases=sum(cases
totalCases <- as.data.frame(totalCases)
head(totalCases)
```

```
##       country year month total.cases
```

```
## 1 Afghanistan 2019    12           0
## 2 Afghanistan 2020     1           0
## 3 Afghanistan 2020     2           1
## 4 Afghanistan 2020     3         140
## 5 Afghanistan 2020     4        1808
## 6 Afghanistan 2020     5       12576
```

```
# 4.2.4 mutate function

# I want to add the total.cases column in the same covid dataframe

totalCases <-  covid %>% group_by(country) %>% mutate(total.cases=sum(cases))
totalCases <- as.data.frame(totalCases)
head(totalCases)
```

```
##      dateRep day month year cases deaths     country popData2019 continentExp
## 1 11/12/2020  11    12 2020    63     10 Afghanistan    38041757         Asia
## 2 10/12/2020  10    12 2020   202     16 Afghanistan    38041757         Asia
## 3 09/12/2020   9    12 2020   135     13 Afghanistan    38041757         Asia
## 4 08/12/2020   8    12 2020   200      6 Afghanistan    38041757         Asia
## 5 07/12/2020   7    12 2020   210     26 Afghanistan    38041757         Asia
## 6 06/12/2020   6    12 2020   234     10 Afghanistan    38041757         Asia
##   total.cases
## 1       48116
## 2       48116
## 3       48116
## 4       48116
## 5       48116
## 6       48116
```

```
# Can you create a dataframe sorted by total number of cases per country
# I want columns - country, total.cases, total.deaths, populaton

covid.table <- covid %>%
  group_by(country,popData2019) %>%
  summarise(Cases=sum(cases),Deaths=sum(deaths)) %>%
  select(country,Cases,Deaths,popData2019)
covid.table <- as.data.frame(covid.table)
head(covid.table)
```

```
##       country Cases Deaths popData2019
## 1 Afghanistan 48116   1945    38041757
## 2     Albania 46061    965     2862427
## 3     Algeria 90579   2564    43053054
## 4     Andorra  7190     78       76177
## 5      Angola 15925    362    31825299
## 6    Anguilla    10      0       14872
```

```r
# Why I did not use mutate here? Think about it!

sort(covid.table$Cases)
```

```
##   [1]       1       3       4      10      13      17      19      19
##   [9]      26      26      31      36      41      43      44      75
##  [17]      85      91     113     146     152     170     187     274
##  [25]     289     291     292     356     364     370     434     509
##  [33]     515     519     617     656     657     684     696     721
##  [41]     725     766     768     905    1009    1069    1201    1385
##  [49]    1477    1555    1676    1733    1736    1739    1868    2081
##  [57]    2126    2178    2435    2444    3090    3181    3182    3325
##  [65]    3579    3779    4180    4579    4932    5011    5183    5337
##  [73]    5524    5576    5717    5811    5887    6049    6053    6349
##  [81]    6633    6833    7052    7190    7585    8487    8787    8805
##  [89]    9181    9300    9465   10268   10884   11081   11203   11443
##  [97]   12058   12588   13308   13368   13996   14052   15535   15773
## [105]   15925   16521   16598   16766   17638   18091   20468   21590
## [113]   23706   25087   25730   28000   29572   30613   39524   40037
## [121]   40165   40741   40786   45004   46061   48116   52738   58297
## [129]   70883   71344   74664   75203   76718   78499   86949   88632
## [137]   89183   90305   90579   90958   91921   91972   98603  100489
## [145]  101763  104487  106280  107814  113207  115360  119420  120147
## [153]  121253  122685  124921  125669  127786  140516  142187  145495
## [161]  146317  146385  149815  151721  154392  162774  163992  171542
## [169]  174568  181405  183099  183630  185424  200379  245650  249224
## [177]  253121  271200  311067  312728  335207  352860  359415  367218
## [185]  391529  432327  442069  445540  485965  539107  558517  566440
## [193]  569205  571253  584980  598933  600261  836764  858714  979111
## [201] 1083023 1102096 1217126 1272078 1399911 1482216 1720056 1787147
## [209] 1787783 2337966 2569126 6781799 9796769 15616381
```

```r
sort(covid.table$Cases,decreasing = TRUE) # Returns values
```

```
##   [1] 15616381  9796769  6781799  2569126  2337966  1787783  1787147  1720056
##   [9]  1482216  1399911  1272078  1217126  1102096  1083023   979111   858714
##  [17]   836764   600261   598933   584980   571253   569205   566440   558517
##  [25]   539107   485965   445540   442069   432327   391529   367218   359415
##  [33]   352860   335207   312728   311067   271200   253121   249224   245650
##  [41]   200379   185424   183630   183099   181405   174568   171542   163992
##  [49]   162774   154392   151721   149815   146385   146317   145495   142187
##  [57]   140516   127786   125669   124921   122685   121253   120147   119420
##  [65]   115360   113207   107814   106280   104487   101763   100489    98603
##  [73]    91972    91921    90958    90579    90305    89183    88632    86949
##  [81]    78499    76718    75203    74664    71344    70883    58297    52738
```

```
##  [89]    48116    46061    45004    40786    40741    40165    40037    39524
##  [97]    30613    29572    28000    25730    25087    23706    21590    20468
## [105]    18091    17638    16766    16598    16521    15925    15773    15535
## [113]    14052    13996    13368    13308    12588    12058    11443    11203
## [121]    11081    10884    10268     9465     9300     9181     8805     8787
## [129]     8487     7585     7190     7052     6833     6633     6349     6053
## [137]     6049     5887     5811     5717     5576     5524     5337     5183
## [145]     5011     4932     4579     4180     3779     3579     3325     3182
## [153]     3181     3090     2444     2435     2178     2126     2081     1868
## [161]     1739     1736     1733     1676     1555     1477     1385     1201
## [169]     1069     1009      905      768      766      725      721      696
## [177]      684      657      656      617      519      515      509      434
## [185]      370      364      356      292      291      289      274      187
## [193]      170      152      146      113       91       85       75       44
## [201]       43       41       36       31       26       26       19       19
## [209]       17       13       10        4        3        1
```

```
order(covid.table$Cases,decreasing = TRUE) # Returns indices
```

```
##   [1] 203  93  28 161  71 201 100 182   8  44  76 128 156  95 154 199 179  19
##  [19]  94 139  96  53  42 196 160  16 155  36 149 134 187 168  99 157 186  12
##  [37]  91 104 170 138  59 151 105  75 200  31 102  49  13  18  58  47  24   9
##  [55] 108 112 159  83 148 175 129  79  60 150  66  90 195 208 136 158  55  26
##  [73]  43 176 153   3 106 115  15 117 121 109  97 206 144 145 173  77   1   2
##  [91] 107 180  61 132 118 147 183  70  11 198  35 111  48 184 213 119 169  64
## [109] 135   5 137  72  52  54  85 122 190  27 101  37 214 124 126  88  73  50
## [127]  20 188 205  14   4  82 194  65 162 120  46 142  87  56 123  92 185  62
## [145]  10  40 178 191  74  32  51 193 181  21  86 172 113 143 212 166  41 141
## [163] 204 114 103 116 209 174  78 167 131 197 211 189  33  38 152 130  63  45
## [181]  68 127 202  23  98  22  34  39  17  84 164 171  25  30   7 146 165  57
## [199]  29  69  81 110 140 192  89 163  67  80 177 133   6 125 210 207
```

```
covid.table[order(covid.table$Cases,decreasing = TRUE),]
```

```
##                                      country    Cases Deaths popData2019
## 203                 United_States_of_America 15616381 292179   329064917
## 93                                     India  9796769 142186  1366417756
## 28                                    Brazil  6781799 179765   211049519
## 161                                   Russia  2569126  45280   145872260
## 71                                    France  2337966  56940    67012883
## 201                            United_Kingdom  1787783  63082    66647112
## 100                                    Italy  1787147  62626    60359546
## 182                                    Spain  1720056  47344    46937060
## 8                                  Argentina  1482216  40431    44780675
## 44                                  Colombia  1399911  38484    50339443
## 76                                   Germany  1272078  20970    83019213
```

```
## 128                          Mexico 1217126 112326   127575529
## 156                          Poland 1102096  21630    37972812
## 95                             Iran 1083023  51496    82913893
## 154                            Peru  979111  36499    32510462
## 199                         Ukraine  858714  14470    43993643
## 179                    South_Africa  836764  22747    58558267
## 19                          Belgium  600261  17692    11455519
## 94                        Indonesia  598933  18336   270625567
## 139                     Netherlands  584980   9889    17282163
## 96                             Iraq  571253  12526    39309789
## 53                          Czechia  569205   9341    10649800
## 42                            Chile  566440  15774    18952035
## 196                          Turkey  558517  15531    82003882
## 160                         Romania  539107  12948    19414458
## 16                        Bangladesh  485965   6967   163046173
## 155                      Philippines  445540   8701   108116622
## 36                            Canada  442069  13109    37411038
## 149                         Pakistan  432327   8653   216565317
## 134                         Morocco  391529   6492    36471766
## 187                     Switzerland  367218   5273     8544527
## 168                    Saudi_Arabia  359415   6012    34268529
## 99                            Israel  352860   2962     8519373
## 157                         Portugal  335207   5278    10276617
## 186                           Sweden  312728   7354    10230185
## 12                           Austria  311067   4158     8858775
## 91                           Hungary  271200   6622     9772756
## 104                           Jordan  253121   3250    10101697
## 170                           Serbia  249224   2172     6963764
## 138                            Nepal  245650   1663    28608715
## 59                           Ecuador  200379  13850    17373657
## 151                           Panama  185424   3287     4246440
## 105                       Kazakhstan  183630   2550    18551428
## 75                           Georgia  183099   1694     3996762
## 200          United_Arab_Emirates  181405    602     9770526
## 31                         Bulgaria  174568   5405     7000039
## 102                           Japan  171542   2502   126860299
## 49                          Croatia  163992   2420     4076246
## 13                       Azerbaijan  162774   1793    10047719
## 18                          Belarus  154392   1238     9452409
## 58                Dominican_Republic  151721   2358    10738957
## 47                        Costa_Rica  149815   1882     5047561
## 24                           Bolivia  146385   9008    11513102
## 9                            Armenia  146317   2445     2957728
## 108                           Kuwait  145495    910     4207077
## 112                           Lebanon  142187   1170     6855709
## 159                             Qatar  140516    240     2832071
```

```
## 83                          Guatemala   127786   4345    17581476
## 148                              Oman   125669   1463     4974992
## 175                          Slovakia   124921   1104     5450421
## 129                           Moldova   122685   2500     4043258
## 79                             Greece   121253   3370    10724599
## 60                              Egypt   120147   6854   100388076
## 150                          Palestine   119420   1008     4981422
## 66                           Ethiopia   115360   1779   112078727
## 90                           Honduras   113207   2968     9746115
## 195                           Tunisia   107814   3758    11694721
## 208                         Venezuela   106280    938    28515829
## 136                           Myanmar   104487   2201    54045422
## 158                       Puerto_Rico   101763   1238     2933404
## 55                            Denmark   100489    918     5806081
## 26          Bosnia_and_Herzegovina      98603   3199     3300998
## 43                             China    91972   4739  1433783692
## 176                          Slovenia   91921   1387     2080908
## 153                          Paraguay   90958   1914     7044639
## 3                             Algeria   90579   2564    43053054
## 106                             Kenya   90305   1568    52573967
## 115                             Libya   89183   1273     6777453
## 15                            Bahrain   88632    347     1641164
## 117                         Lithuania   86949    764     2794184
## 121                          Malaysia   78499    396    31949789
## 109                         Kyrgyzstan   76718   1307     6415851
## 97                             Ireland   75203   2117     4904240
## 206                         Uzbekistan   74664    611    32981715
## 144                            Nigeria   71344   1190   200963603
## 145                   North_Macedonia    70883   2051     2077132
## 173                         Singapore    58297     29     5804343
## 77                              Ghana    52738    326    30417858
## 1                           Afghanistan   48116   1945    38041757
## 2                               Albania   46061    965     2862427
## 107                             Kosovo    45004   1161     1798506
## 180                         South_Korea   40786    572    51225321
## 61                          El_Salvador   40741   1186     6453550
## 132                         Montenegro    40165    566      622182
## 118                         Luxembourg    40037    384      613894
## 147                             Norway    39524    382     5328212
## 183                         Sri_Lanka     30613    146    21323734
## 70                              Finland   29572    442     5517919
## 11                            Australia   28000    908    25203200
## 198                             Uganda    25730    220    44269587
## 35                            Cameroon    25087    443    25876387
## 111                             Latvia    23706    304     1919968
## 48                        Cote_dIvoire    21590    133    25716554
```

```
## 184                                  Sudan 20468  1319 42813237
## 213                                 Zambia 18091   364 17861034
## 119                             Madagascar 17638   258 26969306
## 169                                Senegal 16766   343 16296362
## 64                                 Estonia 16598   141  1324820
## 135                             Mozambique 16521   139 30366043
## 5                                   Angola 15925   362 31825299
## 137                                Namibia 15773   158  2494524
## 72                        French_Polynesia 15535    91   279285
## 52                                 Cyprus 14052    72   875899
## 54  Democratic_Republic_of_the_Congo 13996   349 86790568
## 85                                 Guinea 13368    79 12771246
## 122                               Maldives 13308    47   530957
## 190                             Tajikistan 12588    88  9321023
## 27                               Botswana 12058    36  2303703
## 101                               Jamaica 11443   270  2948277
## 37                             Cape_Verde 11203   109   549936
## 214                              Zimbabwe 11081   305 14645473
## 124                                Malta 10884   164   493559
## 126                           Mauritania 10268   210  4525698
## 88                                 Haiti  9465   233 11263079
## 73                                 Gabon  9300    62  2172578
## 50                                  Cuba  9181   136 11333484
## 20                                Belize  8805   183   390351
## 188                               Syria  8787   476 17070132
## 205                             Uruguay  8487    90  3461731
## 14                              Bahamas  7585   163   389486
## 4                               Andorra  7190    78    76177
## 82                                 Guam  7052   115   167295
## 194                 Trinidad_and_Tobago  6833   122  1394969
## 65                             Eswatini  6633   126  1148133
## 162                              Rwanda  6349    53 12626938
## 120                              Malawi  6053   186 18628749
## 46                                Congo  6049    99  5380504
## 142                            Nicaragua  5887   162  6545503
## 87                                Guyana  5811   154   782775
## 56                              Djibouti  5717    61   973557
## 123                                Mali  5576   184 19658023
## 92                              Iceland  5524    28   356991
## 185                            Suriname  5337   117   581363
## 62                    Equatorial_Guinea  5183    85  1355982
## 10                                 Aruba  5011    46   106310
## 40          Central_African_Republic  4932    63  4745179
## 178                             Somalia  4579   121 15442906
## 191                            Thailand  4180    60 69625581
## 74                               Gambia  3779   123  2347696
```

```
## 32                                 Burkina_Faso     3579     69     20321383
## 51                                     CuraÃ§ao     3325      8       163423
## 193                                       Togo     3182     66      8082359
## 181                                South_Sudan     3181     62     11062114
## 21                                       Benin     3090     44     11801151
## 86                                Guinea_Bissau     2444     44      1920917
## 172                               Sierra_Leone     2435     74      7813207
## 113                                    Lesotho     2178     44      2125267
## 143                                      Niger     2126     80     23310719
## 212                                      Yemen     2081    606     29161922
## 166                                San_Marino     1868     49        34453
## 41                                        Chad     1739    102     15946882
## 141                              New_Zealand     1736     25      4783062
## 204              United_States_Virgin_Islands     1733     23       104579
## 114                                   Liberia     1676     83      4937374
## 103                                    Jersey     1555     32       107796
## 116                             Liechtenstein     1477     17        38378
## 209                                   Vietnam     1385     35     96462108
## 174                               Sint_Maarten     1201     26        42389
## 78                                   Gibraltar     1069      5        33706
## 167                     Sao_Tome_and_Principe     1009     17       215048
## 131                                  Mongolia      905      0      3225166
## 197                  Turks_and_Caicos_islands      768      6        38194
## 211                           Western_Sahara      766      1       582458
## 189                                   Taiwan      725      7     23773881
## 33                                    Burundi      721      1     11530577
## 38  Cases_on_an_international_conveyance_Japan      696      7           NA
## 152                         Papua_New_Guinea      684      7      8776119
## 130                                   Monaco      657      2        33085
## 63                                   Eritrea      656      0      3497117
## 45                                   Comoros      617      7       850891
## 68                             Faroe_Islands      519      0        48677
## 127                                Mauritius      515     10      1269670
## 202            United_Republic_of_Tanzania      509     21     58005461
## 23                                    Bhutan      434      0       763094
## 98                                Isle_of_Man      370     25        84589
## 22                                    Bermuda      364      9        62508
## 34                                   Cambodia      356      0     16486542
## 39                             Cayman_Islands      292      2        64948
## 17                                   Barbados      291      7       287021
## 84                                   Guernsey      289     13        64468
## 164                              Saint_Lucia      274      4       182795
## 171                               Seychelles      187      0        97741
## 25          Bonaire, Saint Eustatius and Saba      170      3        25983
## 30                       Brunei_Darussalam      152      3       433296
## 7                      Antigua_and_Barbuda      146      4        97115
```

```
## 146                  Northern_Mariana_Islands      113       2       57213
## 165          Saint_Vincent_and_the_Grenadines       91       0      110593
## 57                                  Dominica       85       0       71808
## 29                       British_Virgin_Islands       75       1       30033
## 69                                      Fiji       44       2      889955
## 81                                   Grenada       43       0      112002
## 110                                     Laos       41       0     7169456
## 140                          New_Caledonia       36       0      282757
## 192                            Timor_Leste       31       0     1293120
## 89                               Holy_See       26       0         815
## 163                    Saint_Kitts_and_Nevis       26       0       52834
## 67                 Falkland_Islands_(Malvinas)       19       0        3372
## 80                                Greenland       19       0       56660
## 177                         Solomon_Islands       17       0      669821
## 133                              Montserrat       13       1        4991
## 6                                  Anguilla       10       0       14872
## 125                       Marshall_Islands        4       0       58791
## 210                      Wallis_and_Futuna        3       0          NA
## 207                               Vanuatu         1       0      299882
```

```
# dataframe[rowIDs,]


# What is number of observations for each country in the data

covid %>%
  group_by(country) %>%
  summarise(count=n())
```

```
## # A tibble: 214 x 2
##    country             count
##    <fct>               <int>
##  1 Afghanistan           337
##  2 Albania               278
##  3 Algeria               342
##  4 Andorra               273
##  5 Angola                265
##  6 Anguilla              260
##  7 Antigua_and_Barbuda   267
##  8 Argentina             280
##  9 Armenia               338
## 10 Aruba                 264
## # ... with 204 more rows
```

```r
# More summaries

covid %>%
  group_by(country) %>%
  summarise(count=n(),
            total.cases=sum(cases),
            mean.cases=mean(cases),
            max.cases=max(cases),
            min.cases=min(cases))
```

```
## # A tibble: 214 x 6
##    country             count total.cases mean.cases max.cases min.cases
##    <fct>               <int>       <int>      <dbl>     <int>     <int>
##  1 Afghanistan           337       48116   143.          1063         0
##  2 Albania               278       46061   166.           873         0
##  3 Algeria               342       90579   265.          1133         0
##  4 Andorra               273        7190    26.3          299         0
##  5 Angola                265       15925    60.1          355         0
##  6 Anguilla              260          10     0.0385         2         0
##  7 Antigua_and_Barbuda   267         146     0.547         39         0
##  8 Argentina             280     1482216  5294.          18326         0
##  9 Armenia               338      146317   433.           4527         0
## 10 Aruba                 264        5011    19.0          176         0
## # ... with 204 more rows
```

```r
#4.3 "Reshape2" package
library(reshape2) #long-to-wide, wide-to-long

student.id <- c("s1", "s2", "s3", "s4", "s5", "s6")
m1 <- c(12, 10, 45, 22, 30, 27)
m2 <- c(55, 60, 72, 89, 45, 55)

data.wide <- data.frame(student.id, m1, m2, stringsAsFactors = FALSE)
data.wide
```

```
##   student.id m1 m2
## 1         s1 12 55
## 2         s2 10 60
## 3         s3 45 72
## 4         s4 22 89
## 5         s5 30 45
## 6         s6 27 55
```

```r
#this is the wide format, each variable is a column
```

```r
#this is the long format, variables are attributes with values
#(in another column)
#useful of a wide variety of R functions, e.g. ggplot, lm
melt(data.wide)
```

```
## Using student.id as id variables
```

```
##    student.id variable value
## 1          s1       m1    12
## 2          s2       m1    10
## 3          s3       m1    45
## 4          s4       m1    22
## 5          s5       m1    30
## 6          s6       m1    27
## 7          s1       m2    55
## 8          s2       m2    60
## 9          s3       m2    72
## 10         s4       m2    89
## 11         s5       m2    45
## 12         s6       m2    55
```

```r
melt(data.wide, id = "student.id")
```

```
##    student.id variable value
## 1          s1       m1    12
## 2          s2       m1    10
## 3          s3       m1    45
## 4          s4       m1    22
## 5          s5       m1    30
## 6          s6       m1    27
## 7          s1       m2    55
## 8          s2       m2    60
## 9          s3       m2    72
## 10         s4       m2    89
## 11         s5       m2    45
## 12         s6       m2    55
```

```r
data.long <- melt(data.wide, id = "student.id")
head(data.long)
```

```
##   student.id variable value
## 1         s1       m1    12
## 2         s2       m1    10
## 3         s3       m1    45
## 4         s4       m1    22
```

```
## 5         s5       m1    30
## 6         s6       m1    27
```

```
#Coverting long to wide
dcast(data.long,student.id~variable)
```

```
##   student.id m1 m2
## 1         s1 12 55
## 2         s2 10 60
## 3         s3 45 72
## 4         s4 22 89
## 5         s5 30 45
## 6         s6 27 55
```

```
#OR
reshape(data.long,idvar = "student.id",timevar="variable", direction = "wide")
```

```
##   student.id value.m1 value.m2
## 1         s1       12       55
## 2         s2       10       60
## 3         s3       45       72
## 4         s4       22       89
## 5         s5       30       45
## 6         s6       27       55
```

```
#Links
#http://seananderson.ca/2014/09/13/dplyr-intro.html
#http://seananderson.ca/2013/10/19/reshape.html
#-------------------------------------------------------------

############################################################

# 6. Graph plotting
library(ggplot2)

#A graph is made up of layers.
#Visual elements are known as geoms ('geometric objects')
#for example: bar, line, etc.
#Geoms have aesthetic properties (aes()) that control
#the appearance of the graph elements (or the graph as a whole)
#for example: color, size, style

#geoms: geom_bar(), geom_point(), geom_line(), geom_smooth()
#       geom_histogram(), geom_boxplot(), geom_text(), etc.
```

```r
data <- read.table("Howell.csv", header = T,sep=",")
data <- data[,-1]
head(data)
```

```
##    height   weight age male
## 1 151.765 47.82561  63    1
## 2 139.700 36.48581  63    0
## 3 136.525 31.86484  65    0
## 4 156.845 53.04191  41    1
## 5 145.415 41.27687  51    0
## 6 163.830 62.99259  35    1
```

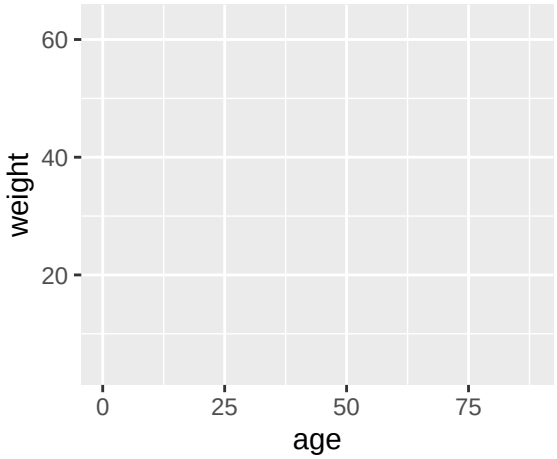
```r
data$Gender <- ifelse(data$male==1,"Male","Female")
head(data)
```

```
##    height   weight age male Gender
## 1 151.765 47.82561  63    1   Male
## 2 139.700 36.48581  63    0 Female
## 3 136.525 31.86484  65    0 Female
## 4 156.845 53.04191  41    1   Male
## 5 145.415 41.27687  51    0 Female
## 6 163.830 62.99259  35    1   Male
```

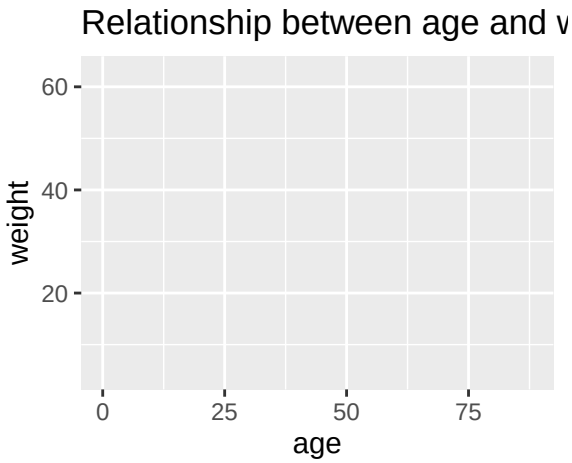
```r
#OR
data <- data %>% mutate(Gender=ifelse(male==1,"Male","Female"))
head(data)
```

```
##    height   weight age male Gender
## 1 151.765 47.82561  63    1   Male
## 2 139.700 36.48581  63    0 Female
## 3 136.525 31.86484  65    0 Female
## 4 156.845 53.04191  41    1   Male
## 5 145.415 41.27687  51    0 Female
## 6 163.830 62.99259  35    1   Male
```

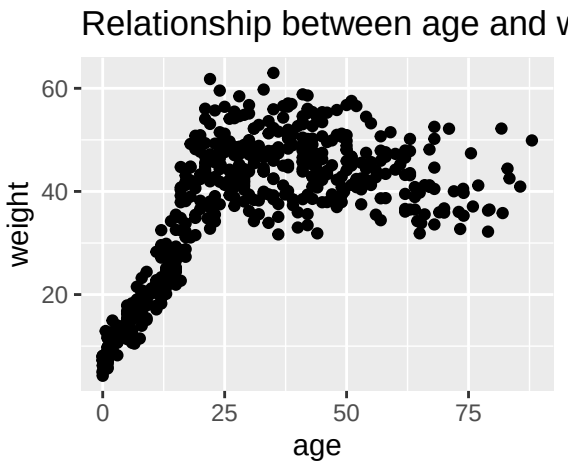
```r
# 6.1 Scatter plots

# Age on X axis and Weight on Y axis
base.plot<-ggplot(data=data, aes(x=age, y=weight))
base.plot
```



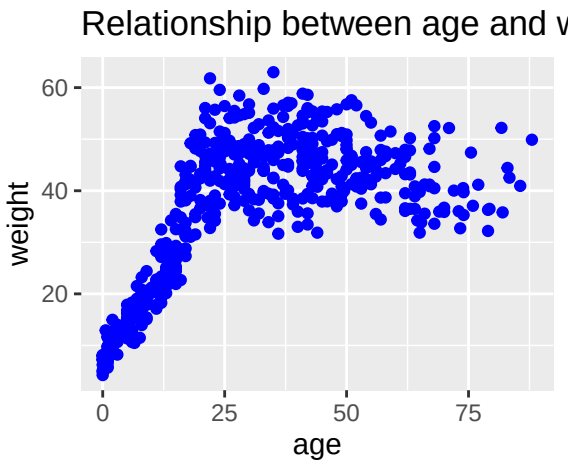

```
p <- base.plot +ggtitle("Relationship between age and weight")
p
```



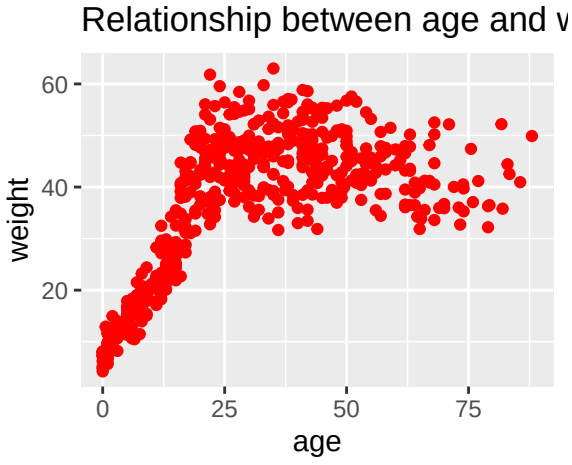

```
#plot the points

p + geom_point()
```




```
p + geom_point(color="Blue")
```




```
p + geom_point(color="red")
```

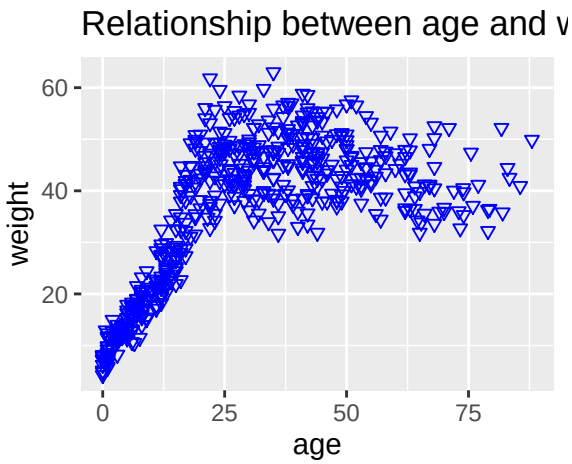


```
?geom_point
```

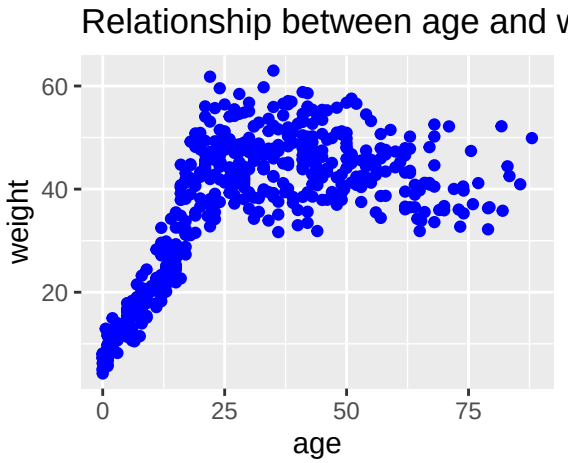
```
## starting httpd help server ...
```

```
##  done
```

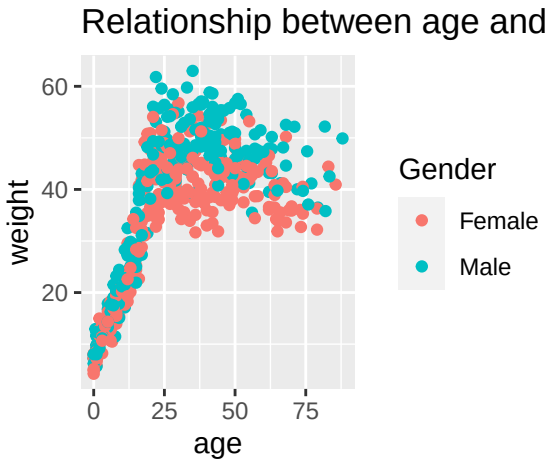
```
p + geom_point(color="blue",shape=6)
```



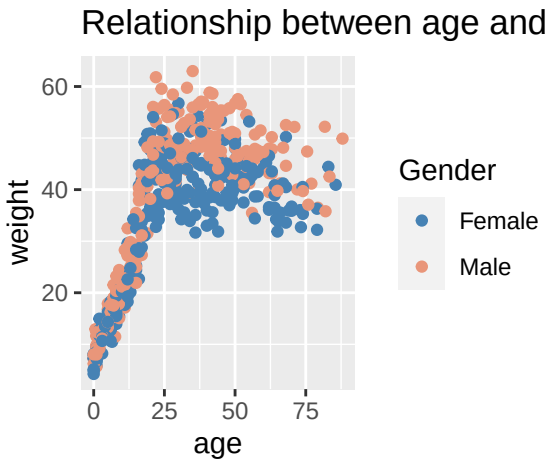

```
p + geom_point(color="blue",size=1.5)
```



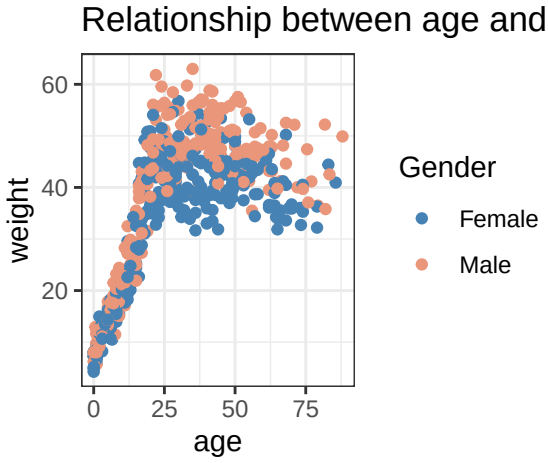

```
# Group datapoints by values of a third variable

p+geom_point(aes(colour=Gender))
```



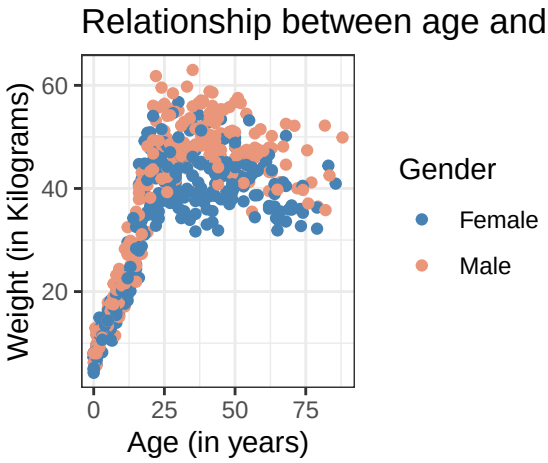

```
p <- p+geom_point(aes(colour=Gender))

p+scale_color_manual(values=c("#4682b4","#e9967a"))
```



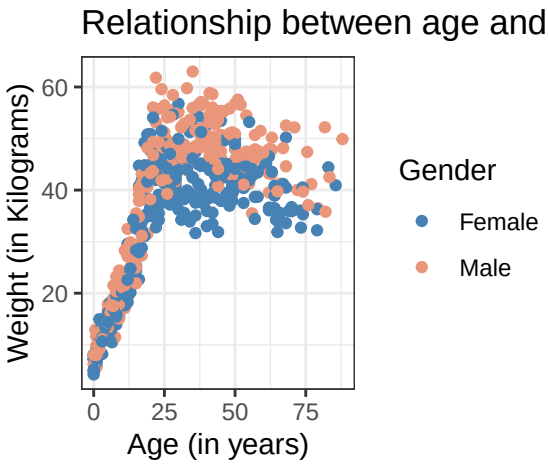

```
p+scale_color_manual(values=c("#4682b4","#e9967a"))+theme_bw()
```



```
p+scale_color_manual(values=c("#4682b4","#e9967a"))+theme_bw()+
  xlab("Age (in years)")+ylab("Weight (in Kilograms)")
```

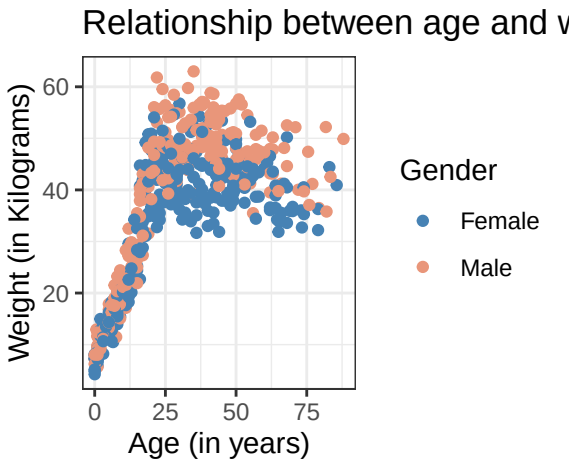



```
ggplot(data=data,aes(x=age,y=weight))+
  ggtitle("Relationship between age and weight")+
  geom_point(aes(colour=Gender))+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")
```

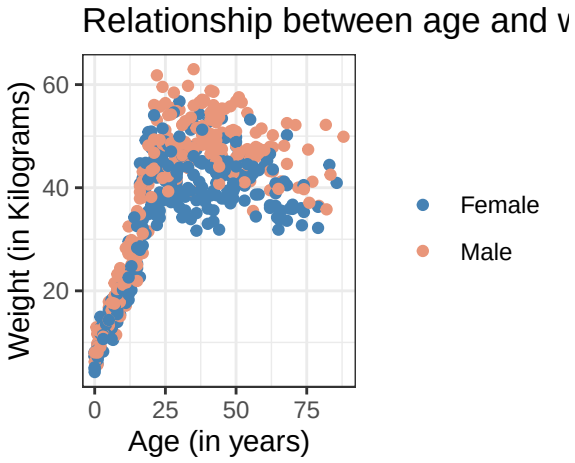



```
ggsave("Age_vs_weight_scatterplot.pdf",height=3,width=5)

ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
```

```
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")
```

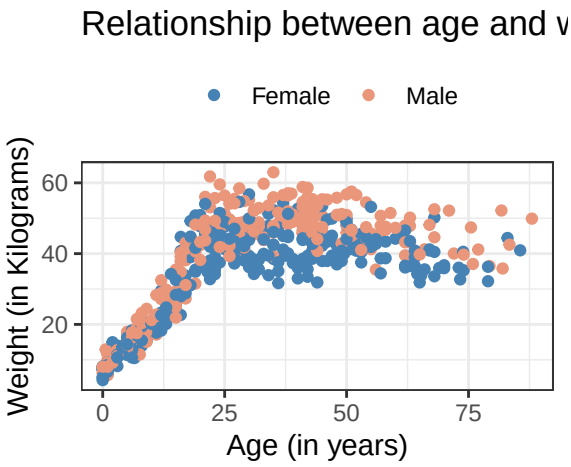



```
ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")+
  theme(legend.title = element_blank())
```

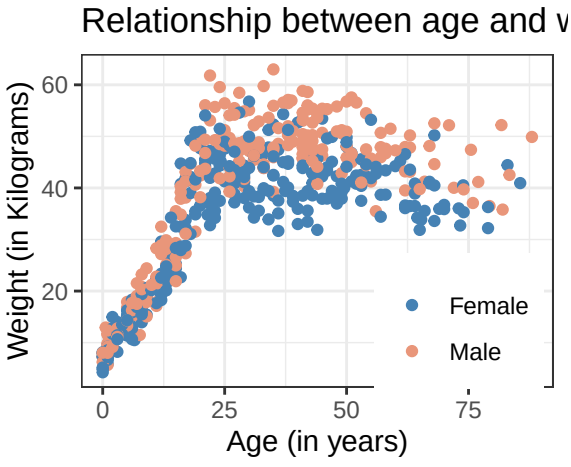



```
ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
```

```
  ylab("Weight (in Kilograms)")+
  theme(legend.title = element_blank(),
        legend.position = "top")
```

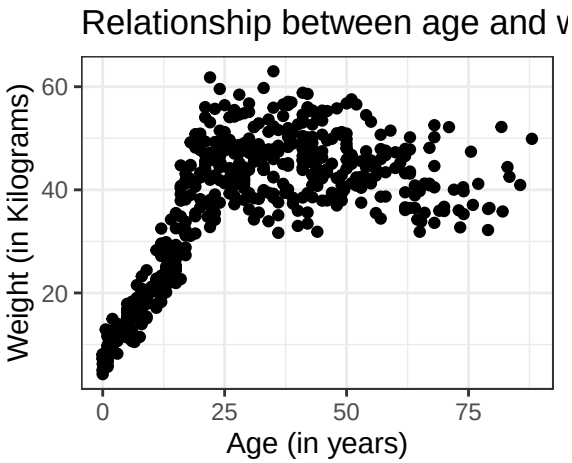



```
ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")+
  theme(legend.title = element_blank(),
        legend.position = c(0.8,0.2))
```

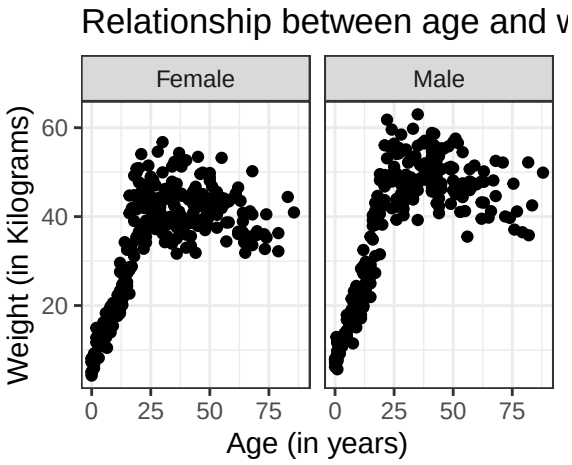



```
# Facet_wrap()
# Show and Male and Female data separately side by side

p <- ggplot(data=data,aes(x=age,y=weight))+
```

```
  ggtitle("Relationship between age and weight")+
  geom_point()+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")
p
```

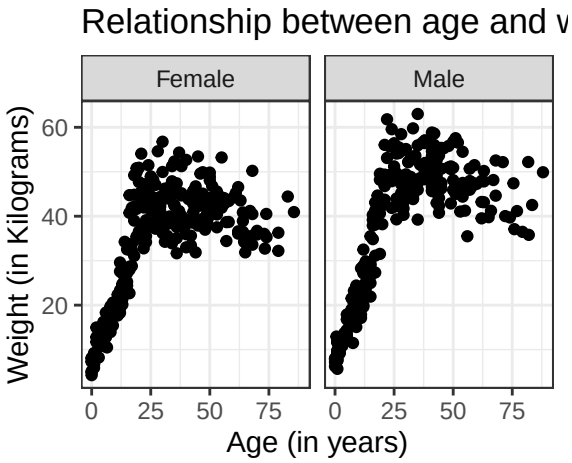



```
p + facet_wrap(~Gender)
```

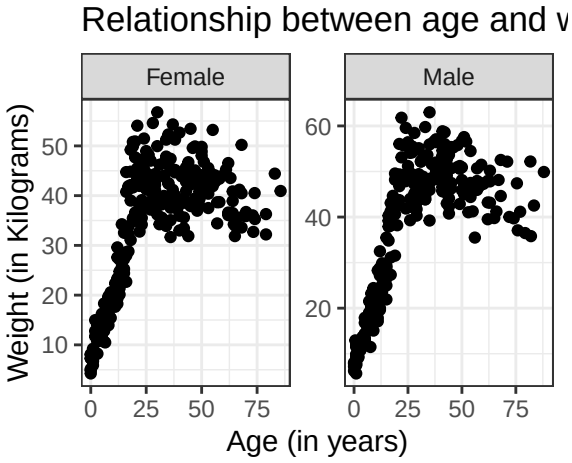



```
p + facet_wrap(~Gender,nrow=2)
```

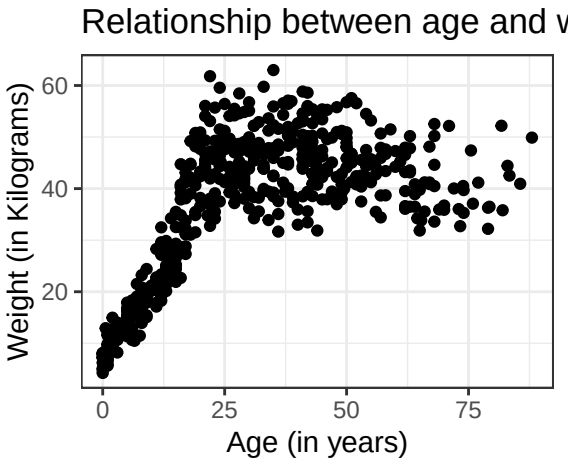

```
p + facet_wrap(~Gender,ncol=2)
```




```
p + facet_wrap(~Gender,scales = "free")
```

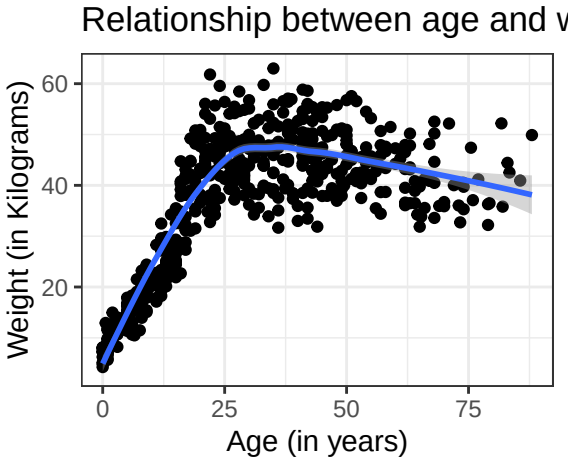


```
# 6.2 regression line
#the line summarizes the relationship between age and weight
#the shaded area is the 95% confidence interval around the line
p
```



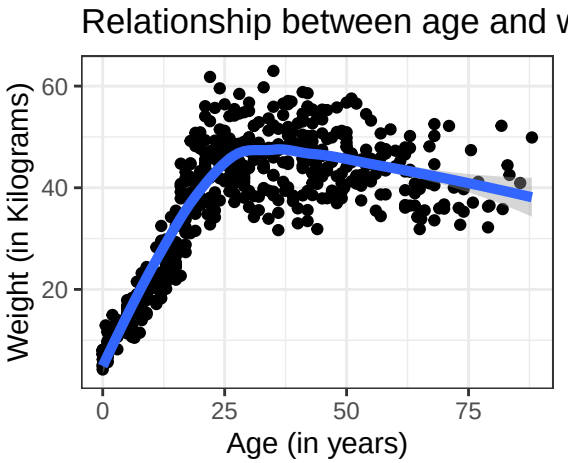

```
p+geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



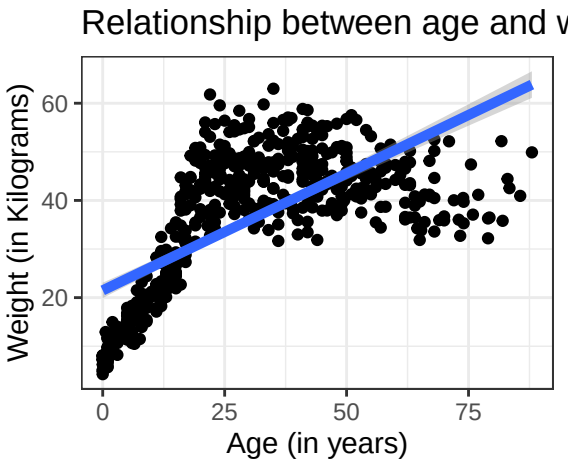

```
p+geom_smooth(size=1.8)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



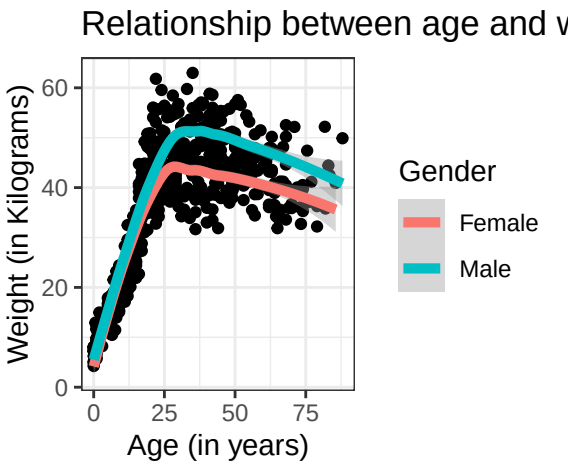

```
p+geom_smooth(size=1.8,method="lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



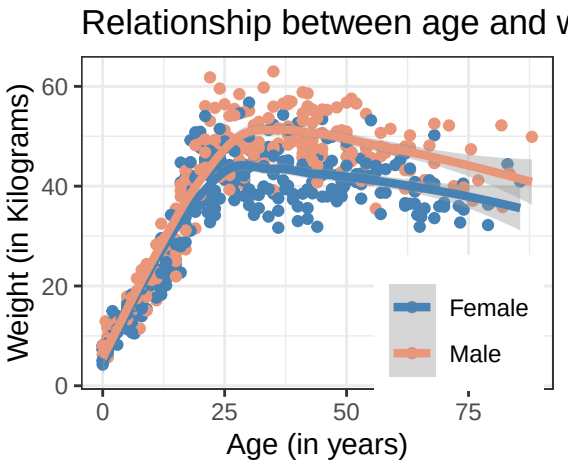

```
p+geom_smooth(size=1.8,aes(colour=Gender))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  geom_smooth(size=1.5)+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")+
  theme(legend.title = element_blank(),
        legend.position = c(0.8,0.2))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

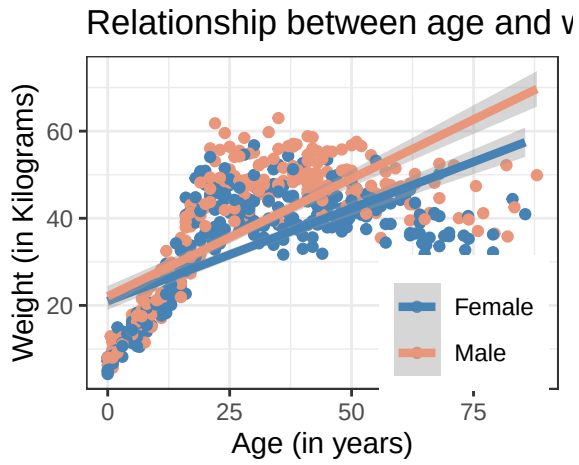



```
ggsave("Age_vs_Weight_regression.pdf",width=5,height=3)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
ggplot(data=data,aes(x=age,y=weight,colour=Gender))+
  ggtitle("Relationship between age and weight")+
  geom_point()+
  geom_smooth(size=1.5,method="lm")+
  scale_color_manual(values=c("#4682b4","#e9967a"))+
  theme_bw()+
  xlab("Age (in years)")+
  ylab("Weight (in Kilograms)")+
  theme(legend.title = element_blank(),
        legend.position = c(0.8,0.2))
```

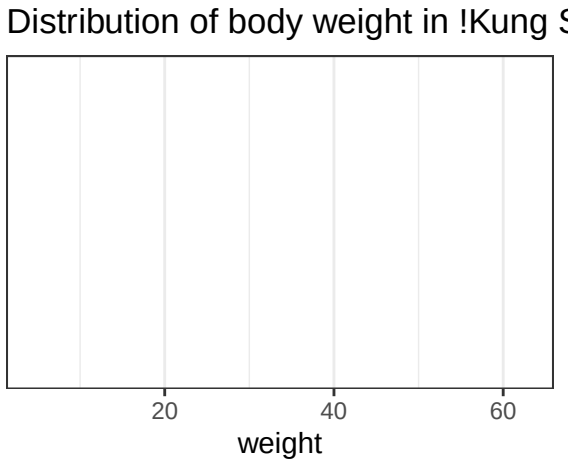
```
## `geom_smooth()` using formula 'y ~ x'
```




```
# 6.3 Histograms

head(data)
```

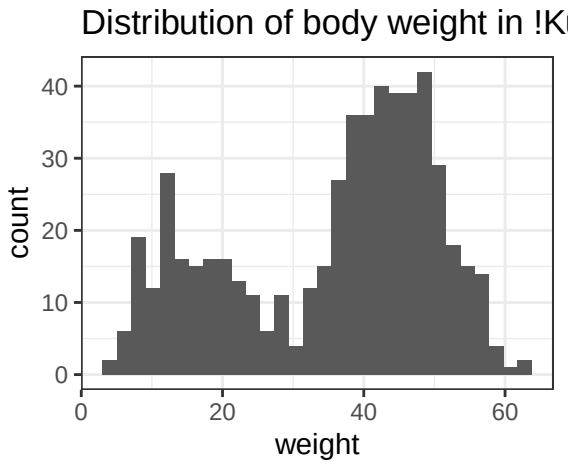
```
##    height   weight age male Gender
## 1 151.765 47.82561  63    1   Male
## 2 139.700 36.48581  63    0 Female
## 3 136.525 31.86484  65    0 Female
## 4 156.845 53.04191  41    1   Male
## 5 145.415 41.27687  51    0 Female
## 6 163.830 62.99259  35    1   Male
```

```
p <- ggplot(data,aes(x=weight))+theme_bw()+
  ggtitle("Distribution of body weight in !Kung San population")
p
```

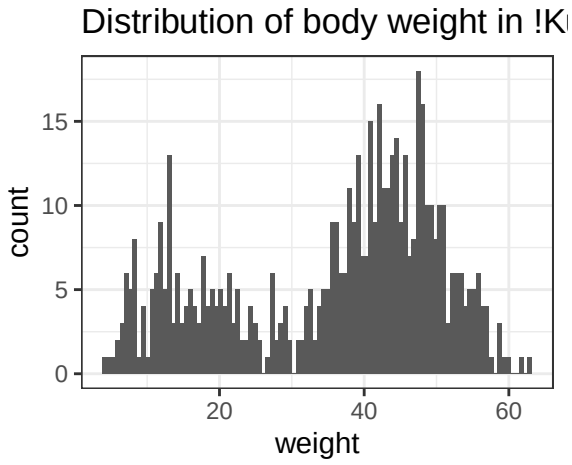



```
p+geom_histogram()
```

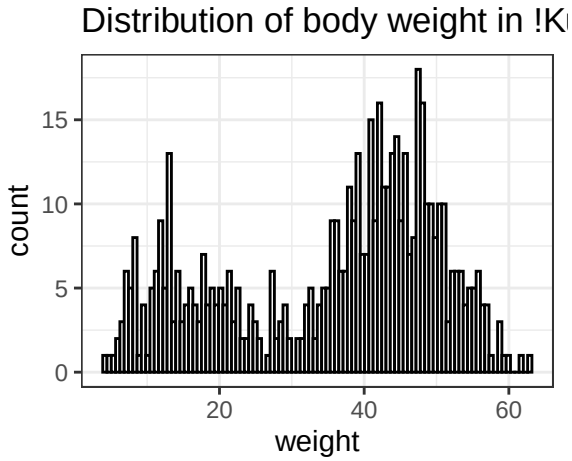
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



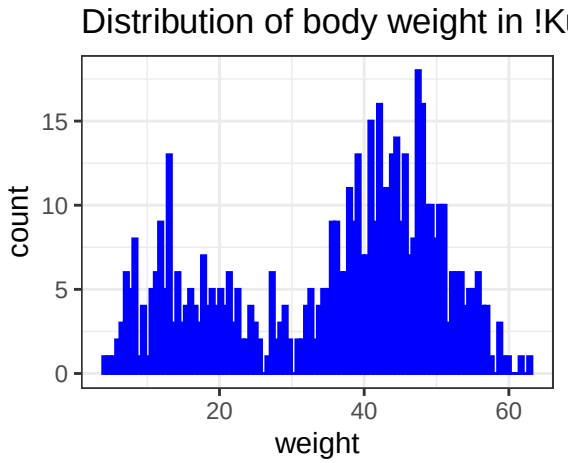

```
p+geom_histogram(bins=100)
```



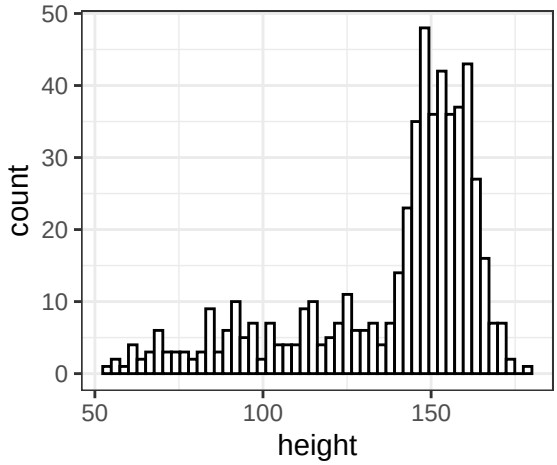
```
p+geom_histogram(bins=100,colour="black",fill="white")
```



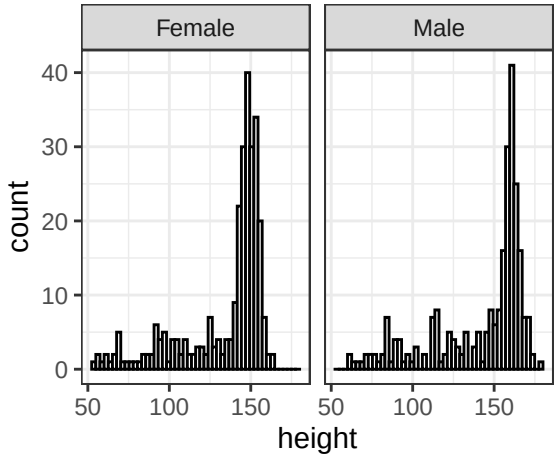

```
p+geom_histogram(bins=100,colour="blue",fill="blue")
```



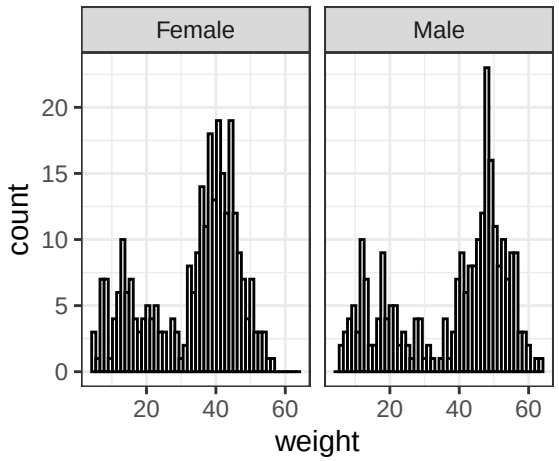

```
ggplot(data,aes(x=height))+theme_bw()+
  geom_histogram(bins=50,colour="black",fill="white")
```



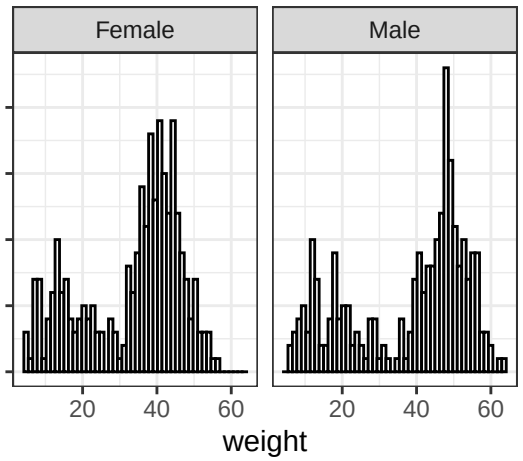

```
ggplot(data,aes(x=height))+theme_bw()+
  geom_histogram(bins=50,colour="black",fill="white")+
  facet_wrap(~Gender)
```



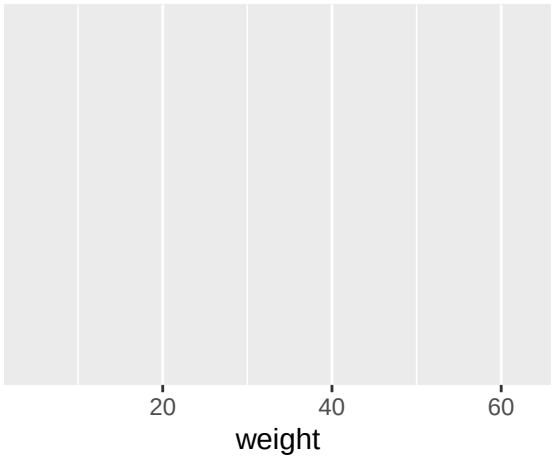

```
ggplot(data,aes(x=weight))+theme_bw()+
  geom_histogram(bins=50,colour="black",fill="white")+
  facet_wrap(~Gender)
```



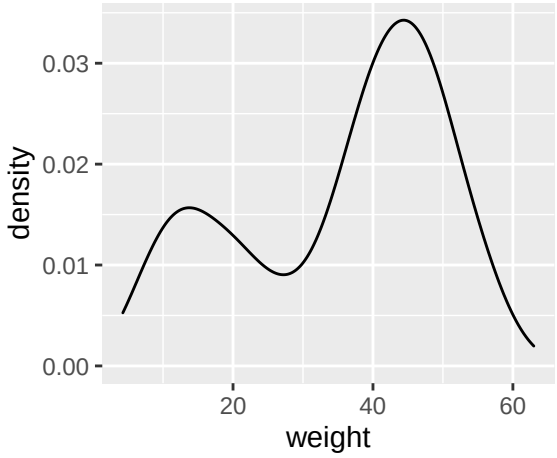

```
ggplot(data,aes(x=weight))+theme_bw()+
  geom_histogram(bins=50,colour="black",fill="white")+
  facet_wrap(~Gender)+
  ylab("")+
  theme(axis.text.y = element_blank())
```



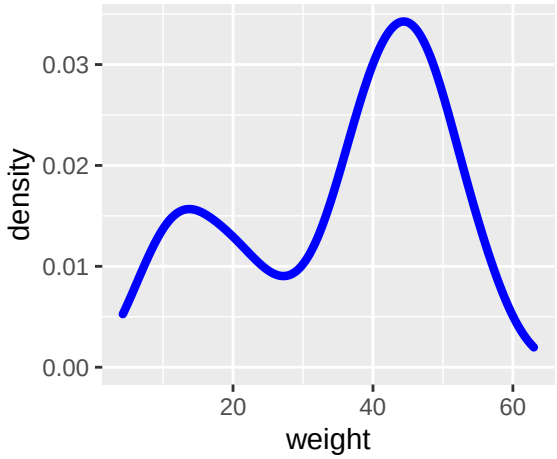

```
# 6.4 Density plots

p <- ggplot(data, aes(x=weight))
p
```



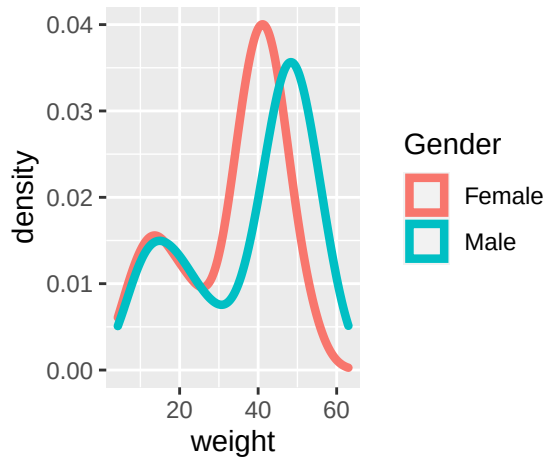

```
p+geom_density()
```



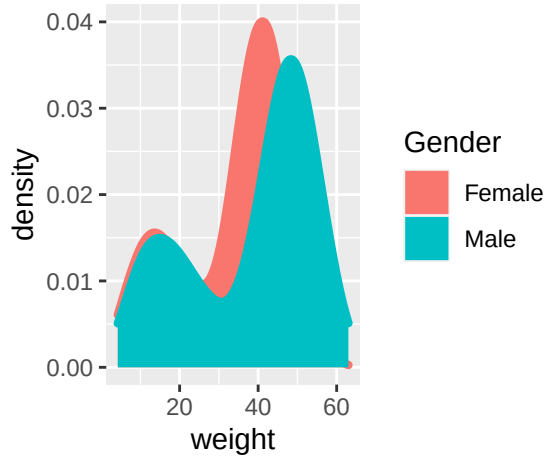

```
p+geom_density(colour="blue",size=1.5)
```



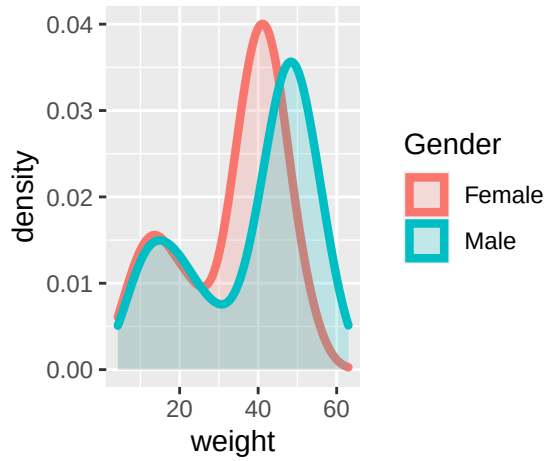
```
p+geom_density(aes(colour=Gender),size=1.5)
```



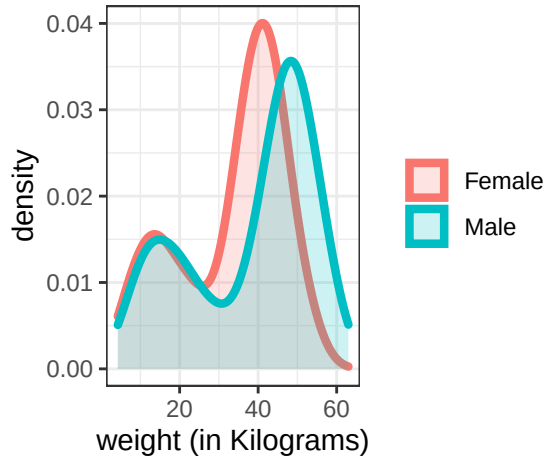

```
p+geom_density(aes(colour=Gender,
                   fill=Gender),
               size=1.5)
```




```
p+geom_density(aes(colour=Gender,
                   fill=Gender),
               size=1.5,
               alpha=0.2)
```

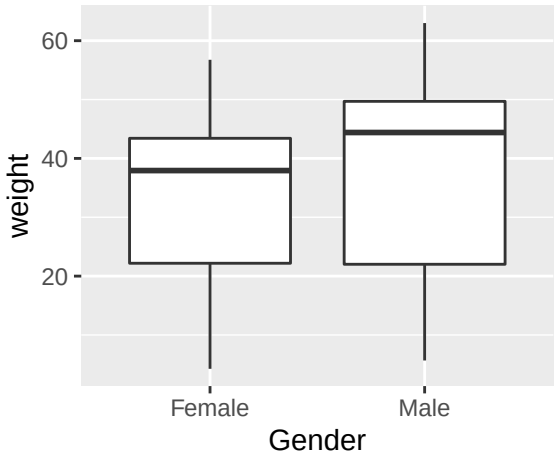



```
p+geom_density(aes(colour=Gender,
                   fill=Gender),
               size=1.5,
               alpha=0.2)+
  theme_bw()+
  theme(legend.title = element_blank())+
  xlab("weight (in Kilograms)")
```




```
ggsave("Weight_density_plot.pdf",width=5,height=4)

# 6.5 boxplot

# median at the centre
# sorrounded by a box; top/bottom: Upper/Lower quartile
# this covers the middle 50% of the data
# the vertical bar above and below (and the points) show
# the top and bottom 25% respectively
# the bars specifically: upper whisker = 3rd quartile + 1.5*IQR
```

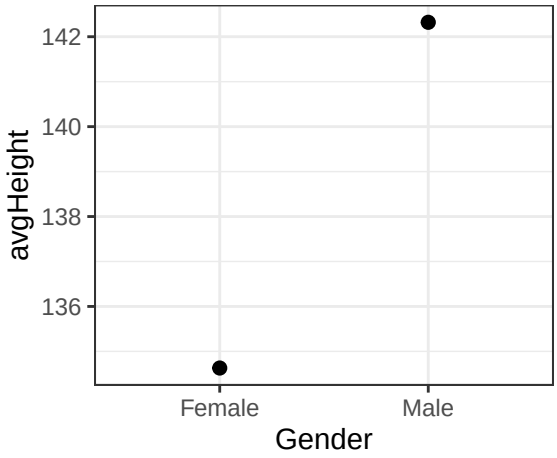
```r
# lower whisker = 1st quartile - 1.5*IQR

p <- ggplot(data, aes(x=Gender, y=weight))
p + geom_boxplot()
```



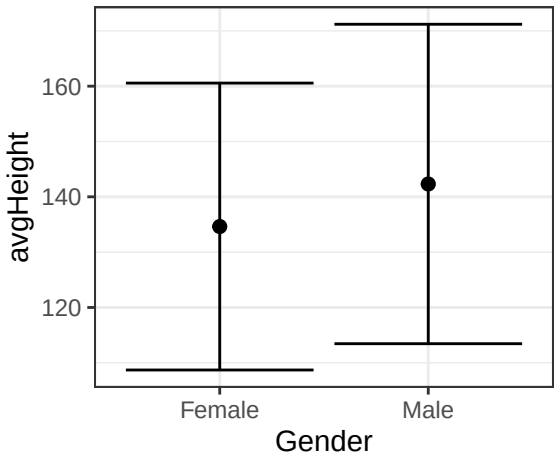

```r
# 6.6 Error bars

data.m <- data %>% group_by(Gender) %>% summarize(avgHeight=mean(height),SDheight=sd(he
data.m <- as.data.frame(data.m)
data.m
```

```
##   Gender avgHeight SDheight
## 1 Female  134.6303 25.93023
## 2   Male  142.3210 28.87132
```

```r
p <- ggplot(data=data.m,aes(x=Gender,y=avgHeight))+theme_bw()
p + geom_point(size=2)
```



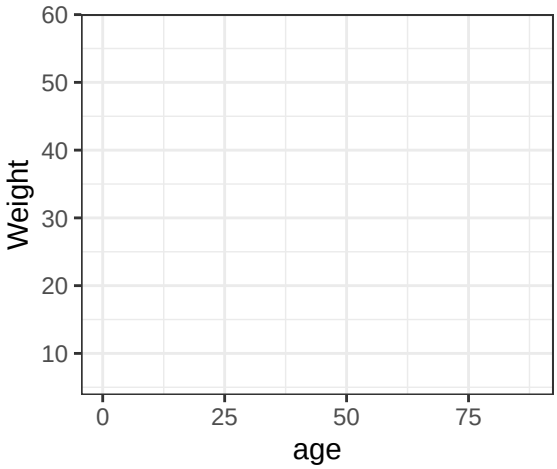
```
# I want to show the distribution of heights using
# an error bar that goes from one SD below average to one SD higher

p + geom_point(size=2)+
  geom_errorbar(aes(ymin=avgHeight-SDheight,ymax=avgHeight+SDheight))
```



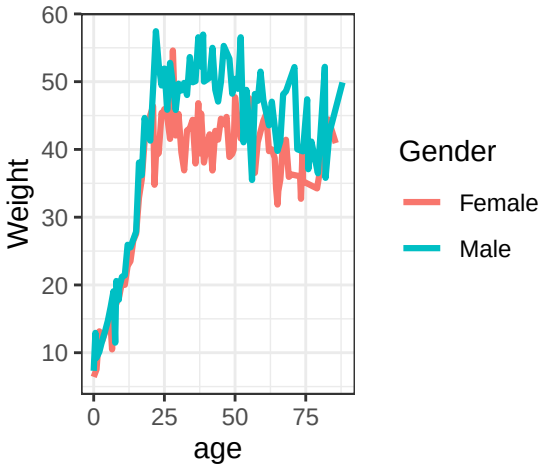

```
# 6.7 Line plots

# Could be more useful than scatter plots sometimes

data.m <- data %>%
  group_by(Gender,age) %>%
  summarize(Weight=mean(weight),upper.w=quantile(weight)[4],lower.w=quantile(weight)[2])
data.m <- as.data.frame(data.m)
head(data.m)
```

```
##   Gender age    Weight   upper.w   lower.w
## 1 Female   0  6.427237  7.668540  5.003687
## 2 Female   1  7.540967  8.136306  7.002327
## 3 Female   2 13.154168 13.877080 12.246984
## 4 Female   3 11.701256 12.842324 11.233489
## 5 Female   4 12.991158 13.579410 12.665139
## 6 Female   5 13.878655 15.223681 12.757275
```

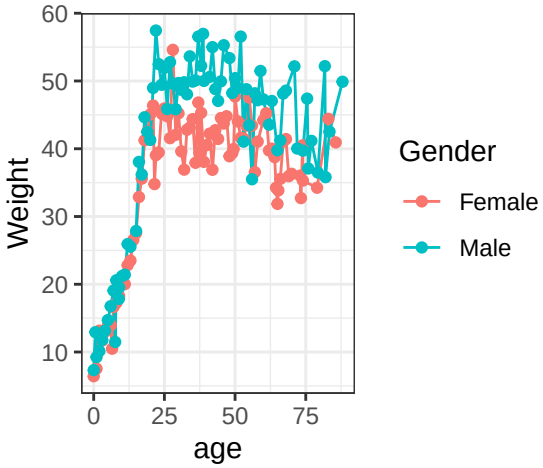
```
# Line plot between Age and weight

p <- ggplot(data.m,aes(x=age,y=Weight,group=Gender,color=Gender))+theme_bw()
p
```



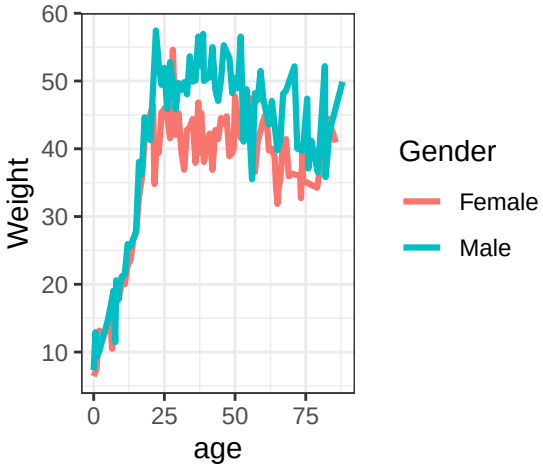

```
p + geom_line(size=1.1)
```



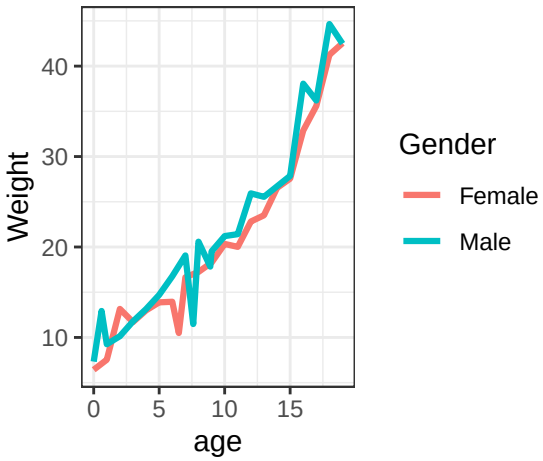

```
p + geom_line() + geom_point(size=1.5)
```



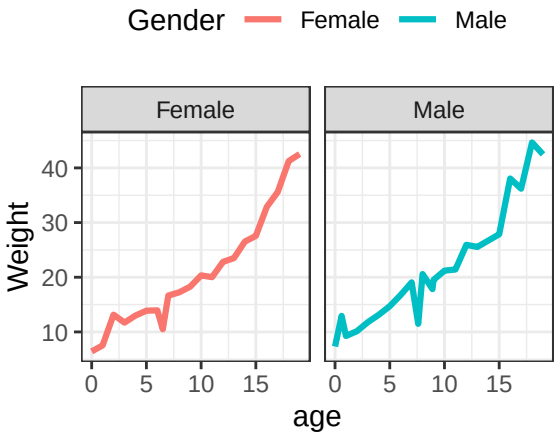
```
ggplot(data.m,aes(x=age,y=Weight,group=Gender,color=Gender))+
  theme_bw()+
  geom_line(size=1.1)
```



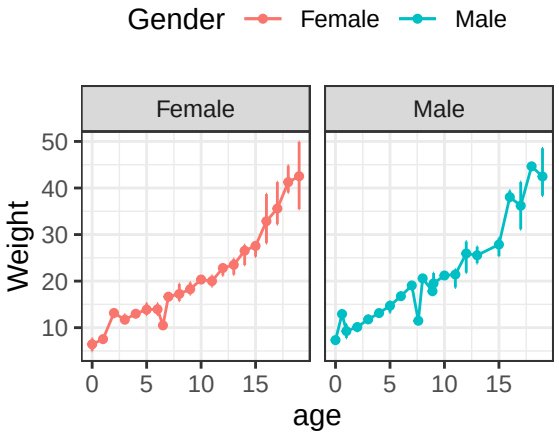

```
ggplot(subset(data.m,age<20),aes(x=age,y=Weight,group=Gender,color=Gender))+
  theme_bw()+
  geom_line(size=1.1)
```



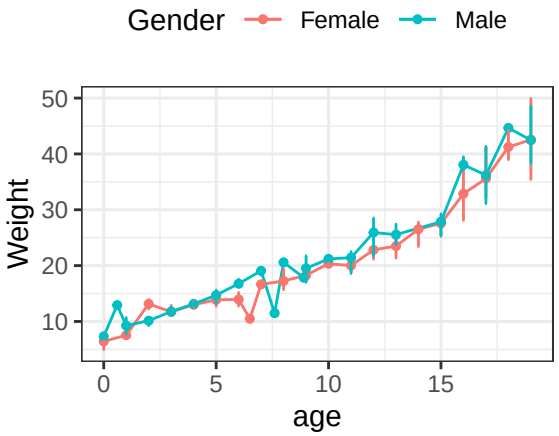

```
ggplot(subset(data.m,age<20),aes(x=age,y=Weight,group=Gender,color=Gender))+
  theme_bw()+
  geom_line(size=1.1)+
  facet_wrap(~Gender)+
  theme(legend.position = "top")
```



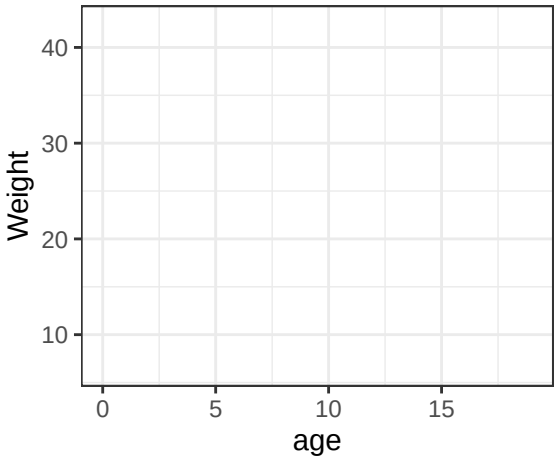

```
ggplot(subset(data.m,age<20),aes(x=age,y=Weight,group=Gender,color=Gender))+
  theme_bw()+
  geom_line()+
  facet_wrap(~Gender)+
  theme(legend.position = "top")+
  geom_point(size=1.1)+
  geom_errorbar(aes(ymax=upper.w,ymin=lower.w))
```



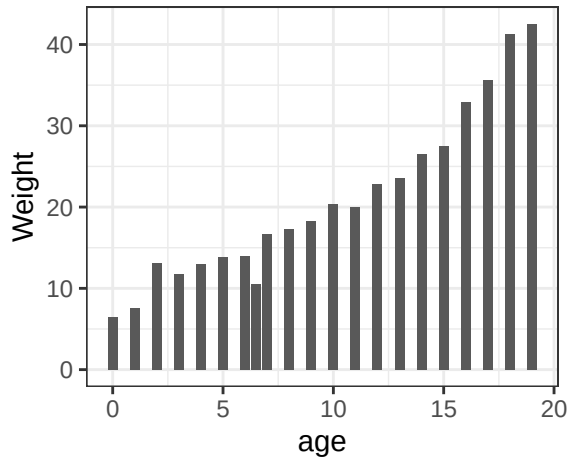

```
ggplot(subset(data.m,age<20),aes(x=age,y=Weight,group=Gender,color=Gender))+
  theme_bw()+
  geom_line()+
  theme(legend.position = "top")+
  geom_point(size=1.1)+
  geom_errorbar(aes(ymax=upper.w,ymin=lower.w))
```



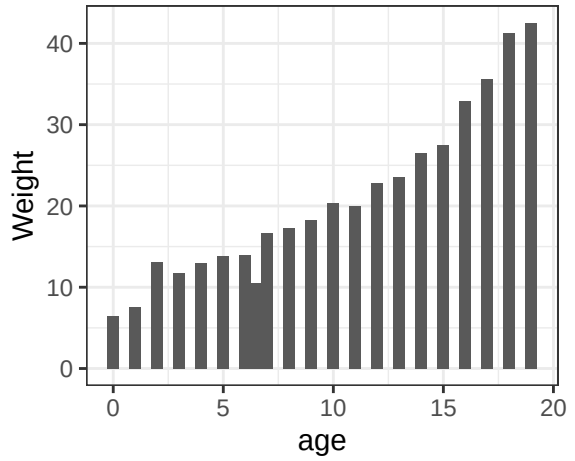

```
# 6.8 Barplots

p <- ggplot(subset(data.m,Gender=="Female"&age<20),
            aes(x=age,y=Weight))+theme_bw()
p
```




```
p+geom_bar(stat="identity")
```

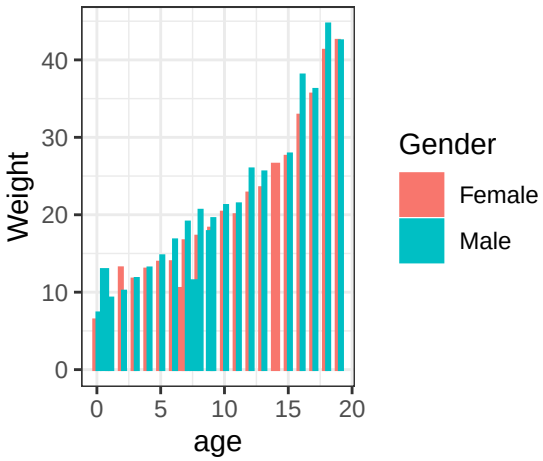



```
ggplot(subset(data.m,Gender=="Female"&age<20),
       aes(x=age,y=Weight))+
  theme_bw()+
  geom_bar(stat="identity",width=0.5)
```



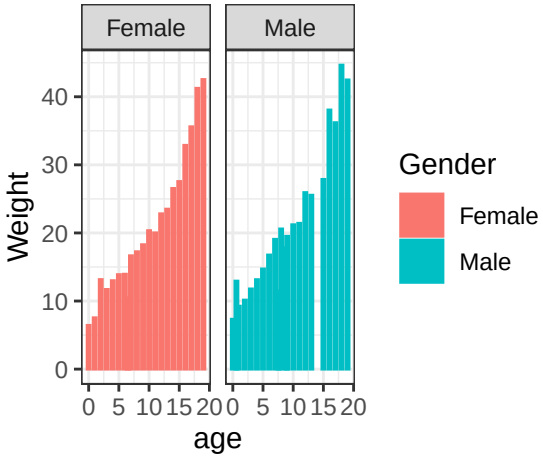

```
ggplot(subset(data.m,age<20),
       aes(x=age,y=Weight,group=Gender,colour=Gender,fill=Gender))+
  theme_bw()+
  geom_bar(stat="identity",position = "dodge",width=0.5)
```

```
## Warning: position_dodge requires non-overlapping x intervals
```




```
ggplot(subset(data.m,age<20),
       aes(x=age,y=Weight,group=Gender,colour=Gender,fill=Gender))+
  theme_bw()+
  geom_bar(stat="identity",width=0.5)+facet_wrap(~Gender)
```

```
## Warning: position_stack requires non-overlapping x intervals
```




```
# 6.9 Heat maps

# To visualize relationships between categorical data
head(covid)
```

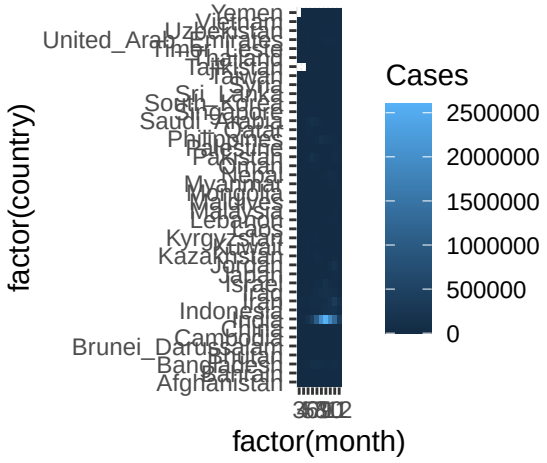
```
##      dateRep day month year cases deaths     country popData2019 continentExp
## 1 11/12/2020  11    12 2020    63     10 Afghanistan    38041757         Asia
## 2 10/12/2020  10    12 2020   202     16 Afghanistan    38041757         Asia
## 3 09/12/2020   9    12 2020   135     13 Afghanistan    38041757         Asia
## 4 08/12/2020   8    12 2020   200      6 Afghanistan    38041757         Asia
## 5 07/12/2020   7    12 2020   210     26 Afghanistan    38041757         Asia
```

```
## 6 06/12/2020    6    12 2020    234     10 Afghanistan    38041757         Asia
```

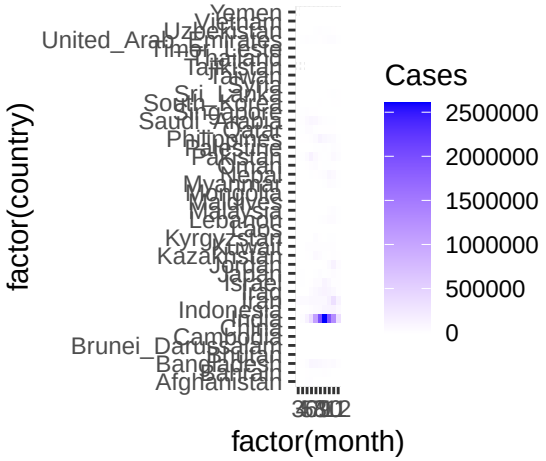
```
covid.m <- covid %>% filter(year==2020&continentExp=="Asia") %>% group_by(country,month)
covid.m <- as.data.frame(covid.m)
head(covid.m)
```

```
##       country month Cases
## 1 Afghanistan     1     0
## 2 Afghanistan     2     1
## 3 Afghanistan     3   140
## 4 Afghanistan     4  1808
## 5 Afghanistan     5 12576
## 6 Afghanistan     6 16713
```

```
# Number of cases per month in each country

p <- ggplot(subset(covid.m,month>2),aes(x=factor(month),y=factor(country),fill=Cases))
p + geom_tile()
```



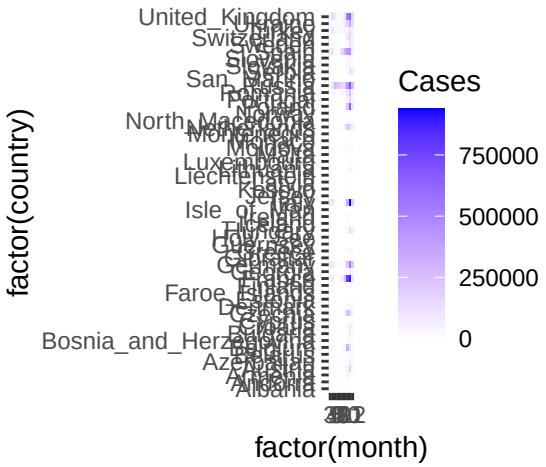

```
ggplot(subset(covid.m,month>2),
       aes(x=factor(month),y=factor(country),fill=Cases))+
  geom_tile()+
  scale_fill_gradient(low="white", high="blue")+
  theme(axis.text.x = )
```




```
# Month wise cases in Europe

covid.m <- covid %>% filter(year==2020&continentExp=="Europe") %>% group_by(country,mont
covid.m <- as.data.frame(covid.m)
head(covid.m)
```

```
##   country month Cases
## 1 Albania     3   223
## 2 Albania     4   543
## 3 Albania     5   356
## 4 Albania     6  1344
## 5 Albania     7  2731
## 6 Albania     8  4183
```

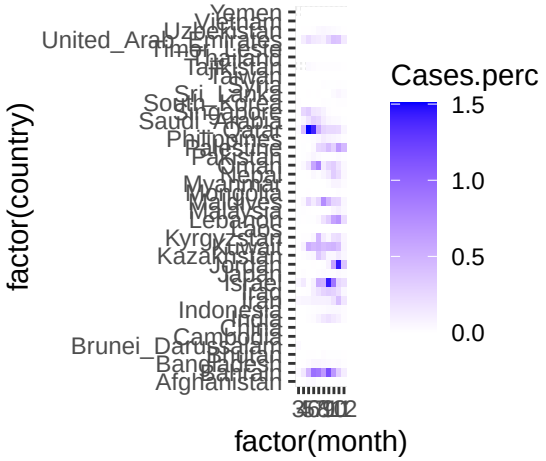
```
ggplot(subset(covid.m,month>2),
       aes(x=factor(month),y=factor(country),fill=Cases))+
  geom_tile()+
  scale_fill_gradient(low="white", high="blue")+
  theme(axis.text.x = )
```



```
# Case normalized by population in Asian counteries

covid.m <- covid %>% filter(year==2020&continentExp=="Asia") %>% group_by(country,month
covid.m <- as.data.frame(covid.m)
head(covid.m)
```

```
##       country month popData2019 Cases   Cases.perc
## 1 Afghanistan     1    38041757     0 0.000000e+00
## 2 Afghanistan     2    38041757     1 2.628690e-06
## 3 Afghanistan     3    38041757   140 3.680167e-04
## 4 Afghanistan     4    38041757  1808 4.752672e-03
## 5 Afghanistan     5    38041757 12576 3.305841e-02
## 6 Afghanistan     6    38041757 16713 4.393330e-02
```

```
ggplot(subset(covid.m,month>2),
       aes(x=factor(month),y=factor(country),fill=Cases.perc))+
  geom_tile()+
  scale_fill_gradient(low="white", high="blue")+
  theme(axis.text.x = )
```

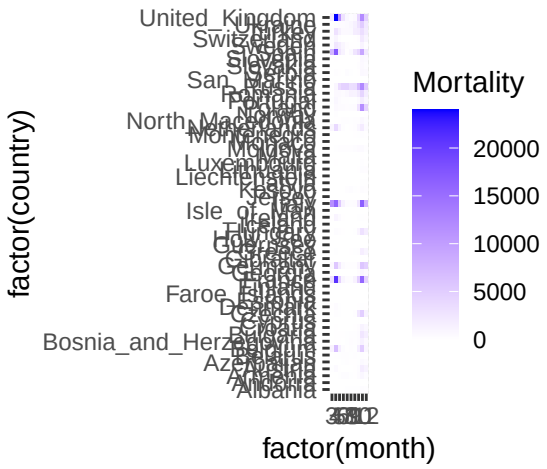



```
# Deaths in Europe

covid.m <- covid %>% filter(year==2020&continentExp=="Europe") %>% group_by(country,mont
covid.m <- as.data.frame(covid.m)
head(covid.m)
```

```
##   country month Mortality
## 1 Albania     3        12
## 2 Albania     4        19
## 3 Albania     5         2
## 4 Albania     6        25
```

```
## 5 Albania     7         96
## 6 Albania     8        126
```

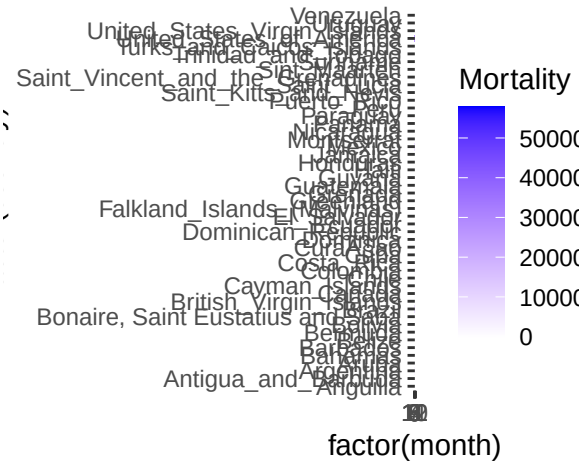
```
ggplot(subset(covid.m,month>2),
       aes(x=factor(month),y=factor(country),fill=Mortality))+
  geom_tile()+
  scale_fill_gradient(low="white", high="blue")+
  theme(axis.text.x = )
```




```
# America

covid.m <- covid %>% filter(year==2020&continentExp=="America") %>% group_by(country,mo
covid.m <- as.data.frame(covid.m)
head(covid.m)
```

```
##    country month Mortality
## 1 Anguilla     3         0
## 2 Anguilla     4         0
## 3 Anguilla     5         0
## 4 Anguilla     6         0
## 5 Anguilla     7         0
## 6 Anguilla     8         0
```

```
ggplot(subset(covid.m,month>2),
       aes(x=factor(month),y=factor(country),fill=Mortality))+
  geom_tile()+
  scale_fill_gradient(low="white", high="blue")+
  theme(axis.text.x = )
```




```
#########
## Homework
#Load the file 'Hindi-word-recognition-data.csv' as a dataframe
# The data are from a "word recognition experiment"
# About experiment: words are presented on a computer screen one by one
# A participant has to recognize the word as quickly as possible
# Our hypothesis is that
# "The words which are more frequent in everyday use will be read faster"
# i.e., the words with high frequency will have small reaction times
# "The words which are long in size will be read slower"
# i.e., the words with longer length will have larger reaction times
hindi <- read.table("Hindi-word-recognition-data.csv",sep=",",header=T)
head(hindi)
```

```
##   X word   label frequency length reactionTime
## 1 1    1 hfshort  4.857125      2     749.3158
## 2 2    2 hfshort  4.905634      2     694.6500
## 3 3    3 lfshort  2.033606      3     779.1500
## 4 4    4 hfshort  4.640838      2     659.5263
## 5 5    5  hflong  3.785533      5     691.4583
## 6 6    6 hfshort  5.391810      2     614.1923
```

```
#(1) What is the variable type of column "label"
#(2) Filter all the observations where label "hfshort"
#(3) Draw histogram of reactionTime in "hfshort" labeled words
#(4) Add a column "ReadingSpeed" in the dataframe such that
# it shows value "Fast" when reactionTime is less than 600 otherwise "Slow"
#(5) Summarize mean, SD of reactionTime by each label
#There are four labels hfshort, hflong, lfshort, lflong
#(6) Draw barplots showing mean reaction time for each label
#(7) Draw errorbar over barplots (ranging from mean-SD to mean+SD)
#(8) Draw density plots showing distribution of
```

```r
#reaction times in each of four labels
#(9) Draw scatter plot of frequency and reactionTime
#(10) Does scatter plot suggest any relationship between
# frequency and reaction time?
#(11) Draw reactionTime~frequency scatter plot in side by side for
# each label using facet_wrap
#(12) Draw regression line between reactionTime and frequency
#(13) Show scatter plot and regression line for reactionTime and length
#(14) Do this:
hindi$frequency <- as.integer(hindi$frequency)
hindi.m <- hindi %>% group_by(frequency,length) %>% summarise(count=n())
hindi.m <- as.data.frame(hindi.m)
head(hindi.m)
```

```
##   frequency length count
## 1         1    2.0     1
## 2         1    2.5     1
## 3         1    3.0    10
## 4         1    5.0    11
## 5         1    5.5     5
## 6         2    2.5     3
```

```r
# Use hindi.m dataframe and
# draw a heatmap between freuency and length where gradient colour is determined by "c
```