ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΥΠΟΥΡΓΕΙΟ ΕΣΩΤΕΡΙΚΩΝ

εκδδα
ΕΘΝΙΚΟ ΚΕΝΤΡΟ ΔΗΜΟΣΙΑΣ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΑΥΤΟΔΙΟΙΚΗΣΗΣ

ΥΠΟΕΡΓΟ: ΥΠΟΕΡΓΟ 2 «ΠΡΟΓΡΑΜΜΑΤΑ ΚΑΤΑΡΤΙΣΗΣ, ΑΝΑΠΤΥΞΗΣ ΔΕΞΙΟΤΗΤΩΝ, ΕΝΔΥΝΑΜΩΣΗΣ, ΠΙΣΤΟΠΟΙΗΣΗΣ - ΥΛΟΠΟΙΗΣΗ ΜΕ ΙΔΙΑ ΜΕΣΑ, ΕΠΙΜΟΡΦΩΣΗ ΑΠΟ ΤΟ ΕΚΔΔΑ» του Έργου «SUB4. Αναβάθμιση των δεξιοτήτων του ανθρώπινου δυναμικού του Δημόσιου Τομέα» με κωδικό ΟΠΣ ΤΑ 5150174
της Δράσης 16972 ΤΑΑ

ΤΙΤΛΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ:

«ΕΙΣΑΓΩΓΗ ΣΤΗ ΓΛΩΣΣΑ R ΓΙΑ ΑΝΑΛΥΣΗ ΔΕΔΟΜΕΝΩΝ (ΙΙ)»

ΕΚΠΑΙΔΕΥΤΙΚΟ ΥΛΙΚΟ

Κωδικός εκπαιδευτικού υλικού:

Κωδικός Πιστοποίησης προγράμματος: 690

Ελλάδα 2.0
ΕΘΝΙΚΟ ΣΧΕΔΙΟ ΑΝΑΚΑΜΨΗΣ ΚΑΙ ΑΝΘΕΚΤΙΚΟΤΗΤΑΣ

Με τη χρηματοδότηση της Ευρωπαϊκής Ένωσης
NextGenerationEU

ΥΠΟΕΡΓΟ: : ΥΠΟΕΡΓΟ 2 «ΠΡΟΓΡΑΜΜΑΤΑ ΚΑΤΑΡΤΙΣΗΣ, ΑΝΑΠΤΥΞΗΣ ΔΕΞΙΟΤΗΤΩΝ, ΕΝΔΥΝΑΜΩΣΗΣ, ΠΙΣΤΟΠΟΙΗΣΗΣ - ΥΛΟΠΟΙΗΣΗ ΜΕ ΙΔΙΑ ΜΕΣΑ, ΕΠΙΜΟΡΦΩΣΗ ΑΠΟ ΤΟ ΕΚΔΔΑ» του Έργου «SUB4. Αναβάθμιση των δεξιοτήτων του ανθρώπινου δυναμικού του Δημόσιου Τομέα» με κωδικό ΟΠΣ ΤΑ 5150174 της Δράσης 16972 ΤΑΑ

ΤΙΤΛΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ:

«ΕΙΣΑΓΩΓΗ ΣΤΗ ΓΛΩΣΣΑ R ΓΙΑ ΑΝΑΛΥΣΗ ΔΕΔΟΜΕΝΩΝ (ΙΙ)»

ΟΜΑΔΑ ΕΡΓΑΣΙΑΣ

Μέλη Ομάδας

Συντονιστής:

Ιωάννης Ματζαβάκης

Συγγραφείς:

Στέφανος Γιακουμάτος

Δημήτριος Τσέλιος

Ιωάννης Χαραλαμπόπουλος

Αξιολογητές:

Γεώργιος Παπαμιχαήλ

Αντώνιος Νείρος

Ελλάδα 2.0 ΕΘΝΙΚΟ ΣΧΕΔΙΟ ΑΝΑΚΑΜΨΗΣ ΚΑΙ ΑΝΘΕΚΤΙΚΟΤΗΤΑΣ

Με τη χρηματοδότηση της Ευρωπαϊκής Ένωσης NextGenerationEU

Περιεχόμενα

Κεφάλαιο 1. Βασικά γραφήματα εξερεύνησης δεδομένων

Σε αυτό το κεφάλαιο υπάρχουν πληροφορίες και τεχνικές για την οπτικοποίηση και τον μετασχηματισμό για την εξερεύνηση των δεδομένων με έναν συστηματικό τρόπο.

Το περιεχόμενο του κεφαλαίου είναι μια εισαγωγή στη διερευνητική ανάλυση δεδομένων (ΔΑΔ) στην αγγλική γλώσσα μεταφράζεται ως exploratory data analysis (EDA), και είναι μία επαναληπτική κυκλική διαδικασία με τα ακόλουθα βήματα:

1. Δημιουργία ερωτήσεων σχετικά με τα δεδομένα.
2. Αναζήτηση οπτικοποιημένων απαντήσεων, μεταμορφώνοντας και μοντελοποιώντας τα δεδομένα.
3. Χρήση των παραπάνω για τη βελτίωση των ερωτήσεων ή δημιουργία νέων ερωτήσεων.

Για την εκπαίδευση σε γραφήματα εξερεύνησης θα χρησιμοποιήσουμε τα δεδομένα που υπάρχουν στην R με το όνομα airquality

Για να δούμε τα δεδομένα:

```
# inspect the data in a tabular view
View(airquality)


# rename the dataset (to place the object in the Environment tab)
my_data <- airquality


# inspect the structure of the dataset
str(my_data)
```

```
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

1.1. Εξερεύνηση βασικών στατιστικών μεγεθών

Πριν προχωρήσουμε σε δημιουργία γραφημάτων, κάνουμε κάποιες βασικές εργασίες και πράξεις με τα δεδομένων του υποδείγματός μας.

```
# inspect a subset of the dataset
my_data[1:10,]
```

```
   Ozone Solar.R Wind Temp Month Day
1     41     190  7.4   67     5   1
2     36     118  8.0   72     5   2
3     12     149 12.6   74     5   3
```

```
4     18     313 11.5   62     5   4
5     NA      NA 14.3   56     5   5
6     28      NA 14.9   66     5   6
7     23     299  8.6   65     5   7
8     19      99 13.8   59     5   8
9      8      19 20.1   61     5   9
10    NA     194  8.6   69     5  10
```

```
# calculate the average of the air temperature column (Temp)
mean(my_data$Temp)
```

```
[1] 77.88235
```

```
# calculate the median aof the wind column (Wind)
median(my_data$Wind)
```

```
[1] 9.7
```

```
# calcute the quantiles of the air temperature column (Temp)
quantile(my_data$Temp)
```

```
  0%  25%  50%  75% 100%
  56   72   79   85   97
```

```
# Calculate the quantile 0, 10%....,90%
sep <- seq(0,1,0.1)

quantile(my_data$Temp,sep)
```

```
  0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
56.0 64.2 69.0 74.0 76.8 79.0 81.0 83.0 86.0 90.0 97.0
```

1.2. Δημιουργία βασικών γραφημάτων

Διάγραμμα διασποράς (scatter plot)

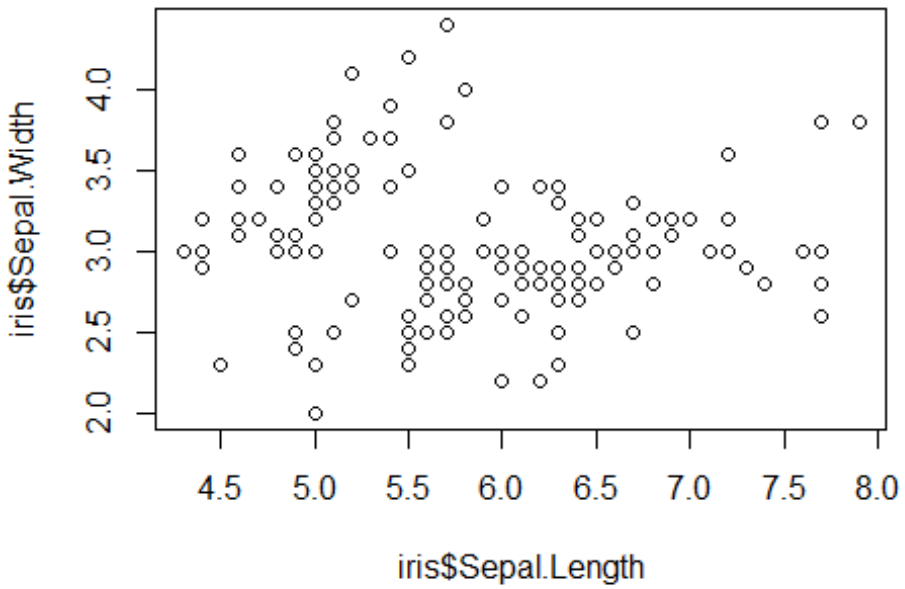
Τα διαγράμματα διασποράς, ειδικά όταν αυτά είναι ομαδοποιημένα, μπορούν να σώσουν άμεσα, χρήσιμες πληροφορίες και να κάνουν εύκολα κατανοητά τα χαρακτηριστικά των δεδομένων που χειριζόμαστε.

Για αυτή την ενότητα θα χρησιμοποιήσουμε τα δεδομένα iris (περιλαμβάνει μετρήσεις για το μήκος και πλάτος των πετάλων και σέπαλων φυτών ίριδας διαφορετικών ειδών).

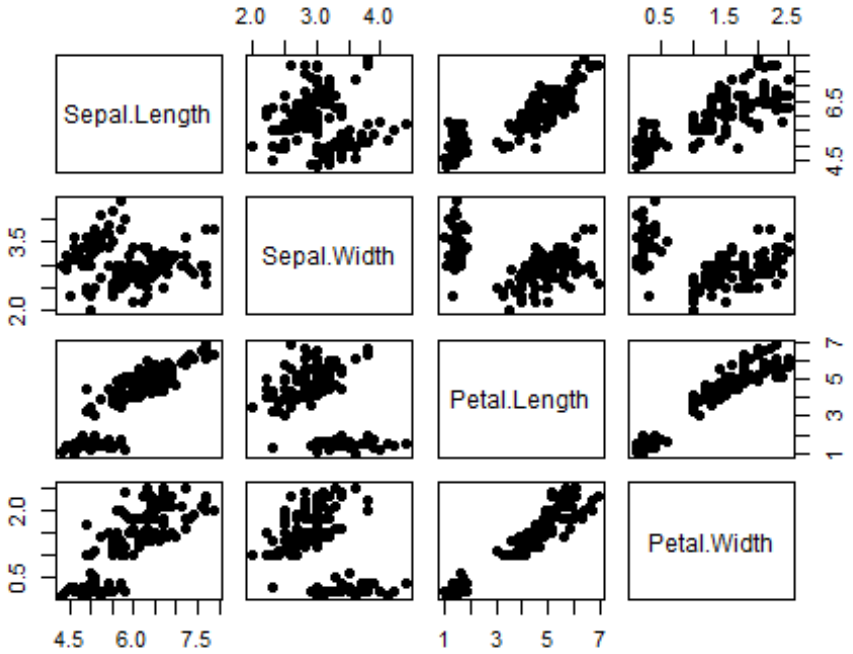
```
# Inspect the iris dataset
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

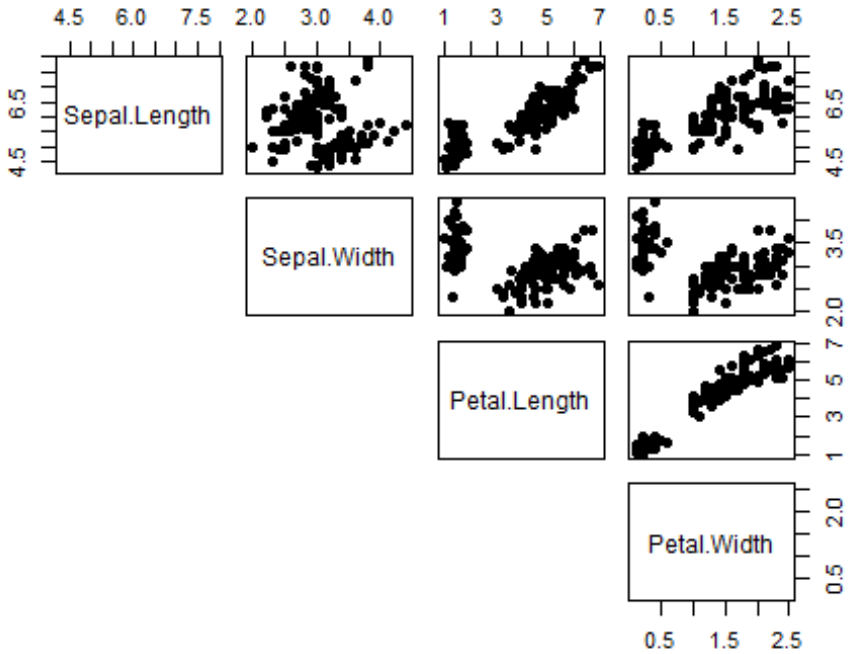
```r
# make a simple plot for Sepal.Length vs Sepal.Width
plot(iris$Sepal.Length, iris$Sepal.Width)
```




```r
# Create a basic scatter plot with pairs function with all the avail
able pairs
pairs(iris[, 1:4], # επιλογή μόνο των 4 πρώτων στηλών και όλων των γ
ραμμών*
      pch = 19) # το είδος του σημείου (κύκλος)
```

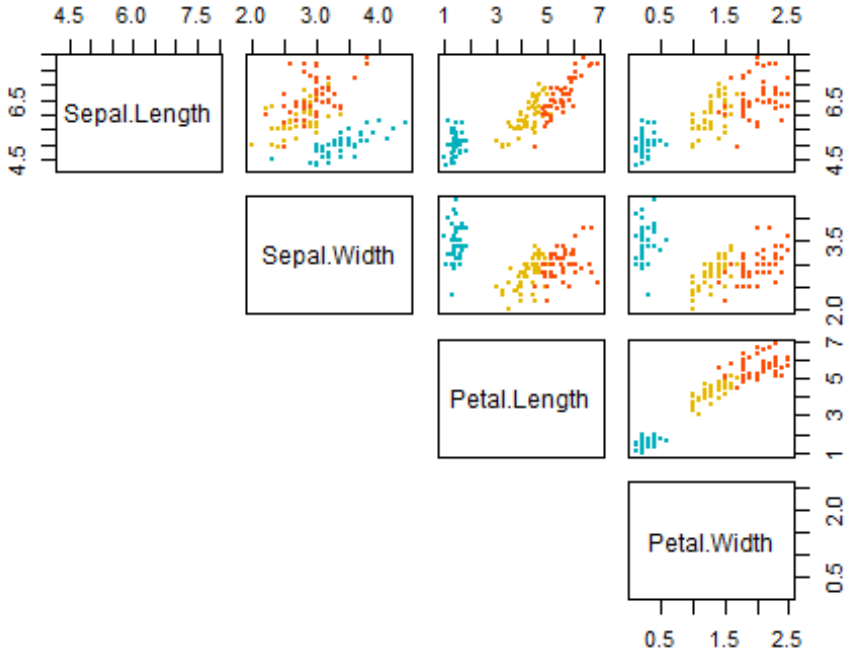



```
# show only the upper panel of the scatterplot
pairs(iris[, 1:4],
      pch = 19,
      lower.panel = NULL)# remove the lower panel
```




```
# color the different species
my_cols <- c("#00AFBB",
```

```
              "#E7B800",
              "#FC4E07")


# create pairs graph
pairs(
  iris[, 1:4], # the subset for the scatterplot
  pch = 19, # the type of the mark (19=cycle)
  cex = 0.5, # the size of the marks or rext
  col = my_cols[iris$Species], # change the colors according to the
Species
  lower.panel = NULL # remove the lowel panel
)
```

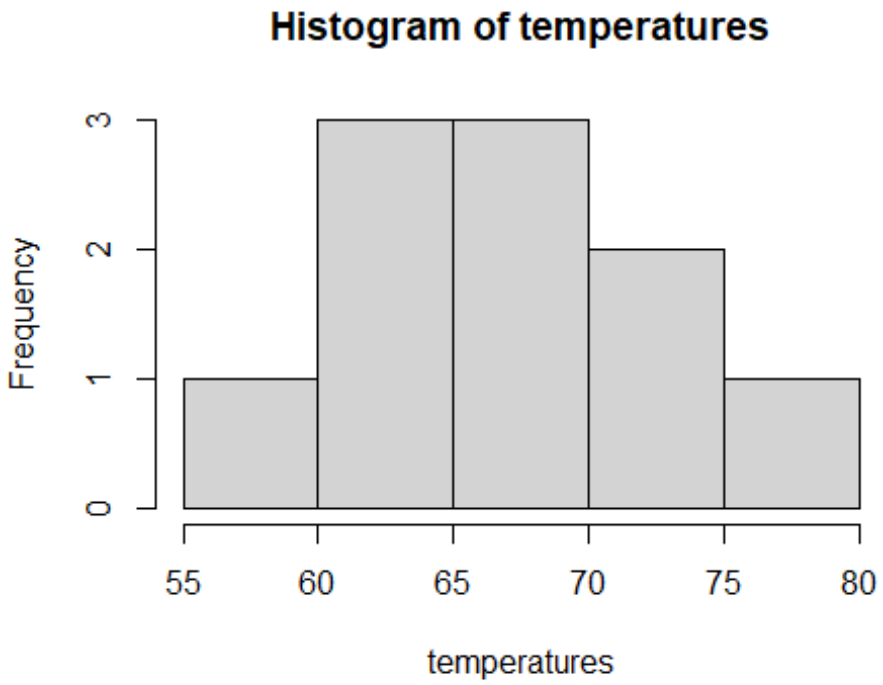


*Κατά τη συγγραφή κώδικα δεν είναι δόκιμο να μπαίνουν τα σχόλια με αυτό τον τρόπο, αλλά εξυπηρετούν την εκπαιδευτική διαδικασία.

Ιστόγραμμα (histogram)

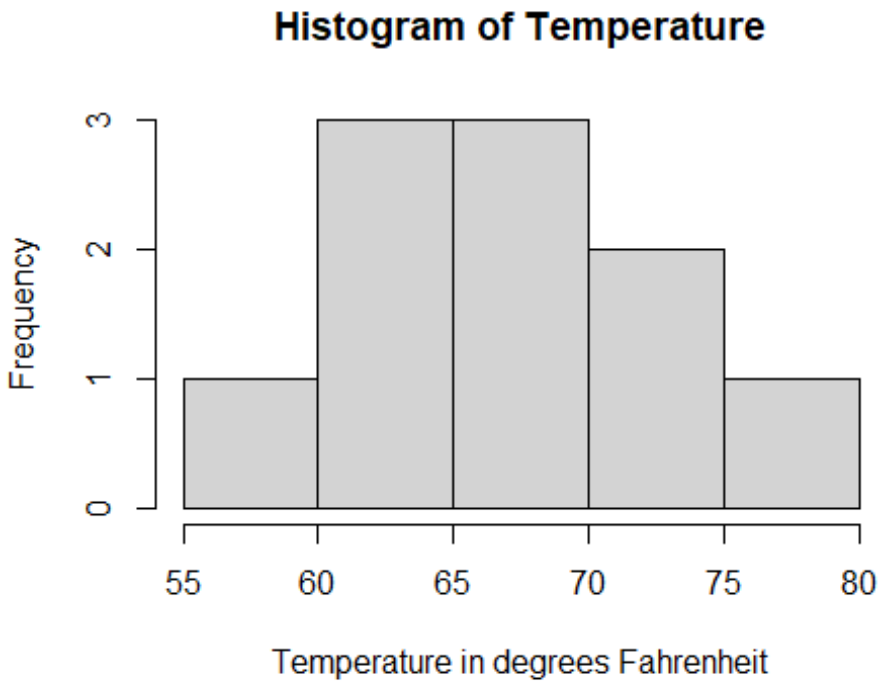
Το ιστόγραμμα είναι μια γραφική απεικόνιση στατιστικών συχνοτήτων περιοχών τιμών ενός συνόλου παρατηρήσεων. Είναι ένα γράφημα που παρουσιάζει εύκολα και κατανοητό παρατηρήσεις συνεχών μεταβλητών. Το ύψος κάθε ιστού δείχνει τη συχνότητα της περιοχής των παρατηρήσεων που αντιπροσωπεύει.

```
# create manually a dataset (vector)
temperatures <- c(67 , 72 , 74 , 62 , 76 , 66 , 65 , 59 , 61 , 69)


# histogram of temperatures vector
result_hist <- hist(temperatures)
```

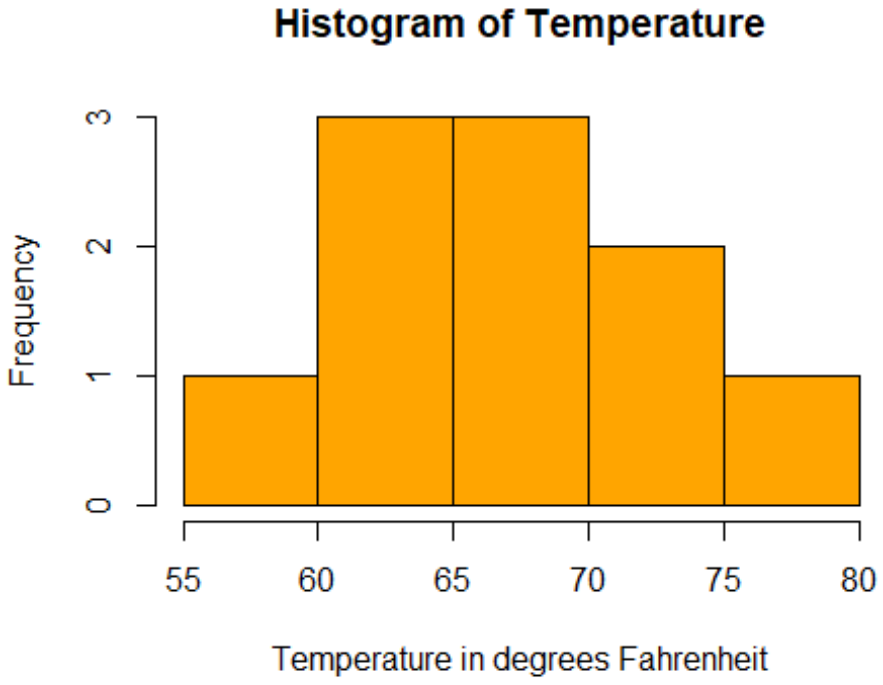



```
# print the histogram attributes
print(result_hist)

$breaks
[1] 55 60 65 70 75 80

$counts
[1] 1 3 3 2 1

$density
[1] 0.02 0.06 0.06 0.04 0.02

$mids
[1] 57.5 62.5 67.5 72.5 77.5

$xname
[1] "temperatures"

$equidist
[1] TRUE

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

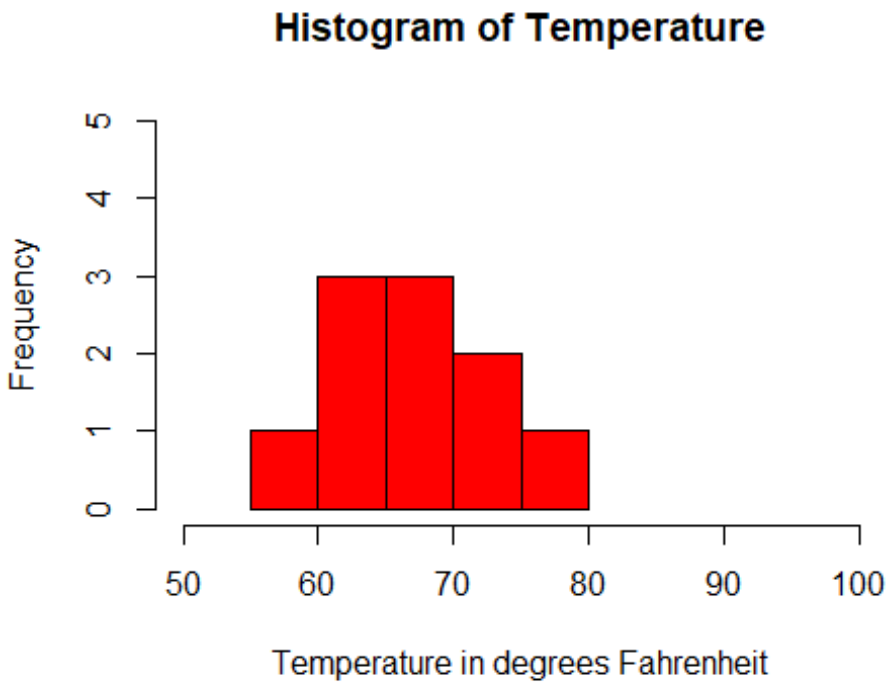
```
#  histogram of temperatures vector with title and x axis legend
my_hist2 <- hist(temperatures,
                 main = "Histogram of Temperature", # the main title
                 xlab = "Temperature in degrees Fahrenheit") # the x
axis label
```



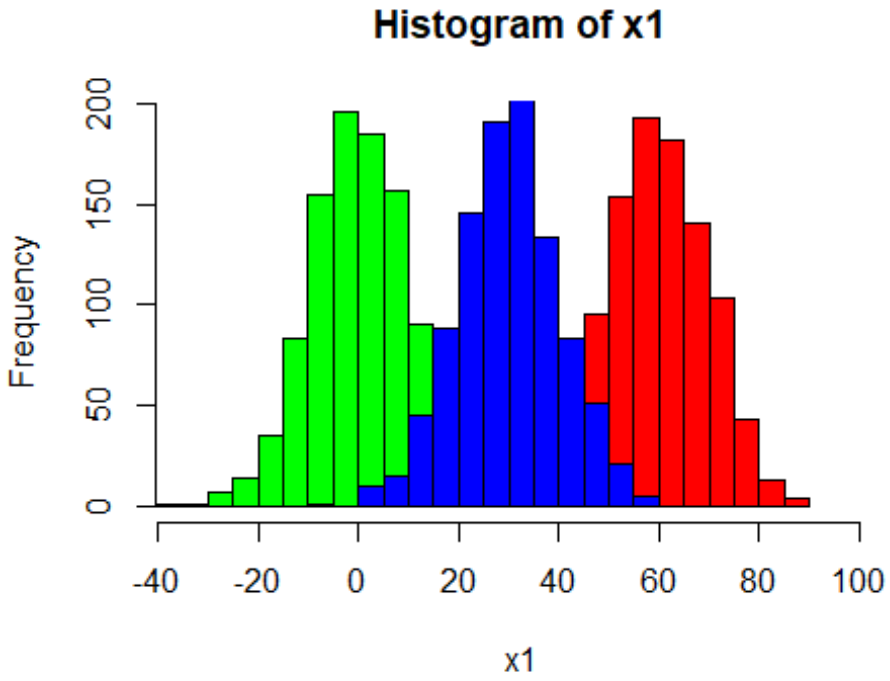

```
# paint the histogram with color
my_hist3 <- hist(temperatures,
                 main = "Histogram of Temperature",
                 xlab = "Temperature in degrees Fahrenheit",
                 col = "orange") # set the color
```



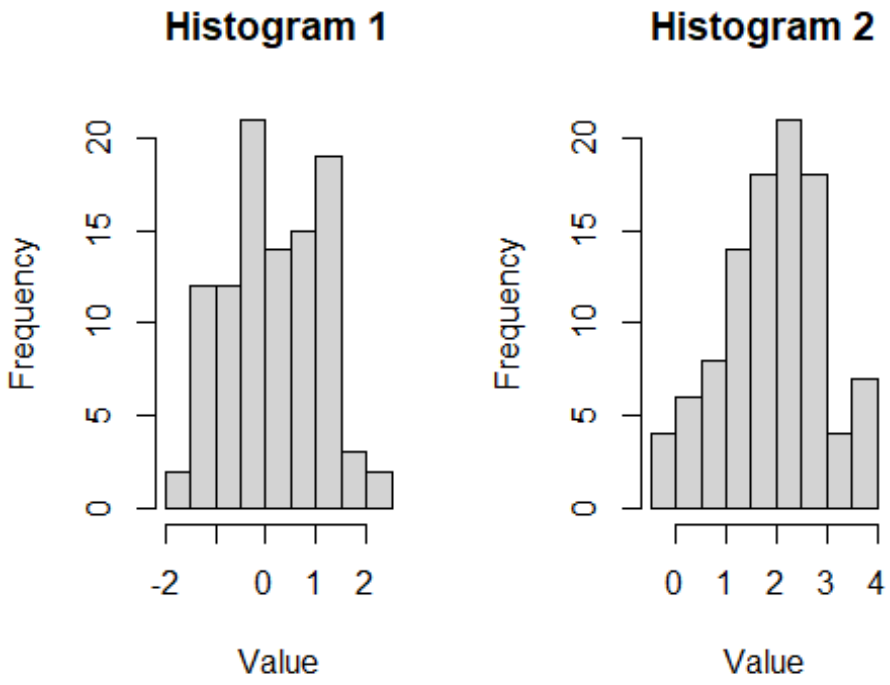
```r
# change the axes range
my_hist4 <- hist(
  temperatures,
  main = "Histogram of Temperature",
  xlab = "Temperature in degrees Fahrenheit",
  col = "red", # set the color as red
  xlim = c(50, 100), # set the limitis of the x axis
  ylim = c(0, 5) # set the limits of the y axis
)
```




```r
# create multiple histogramms in one plot

## create data vectors
x1 = rnorm(1000, mean=60, sd=10) # 1000 random numbers with mean = 6
0, standard deviation = 10
x2 = rnorm(1000, mean=0, sd=10)
x3 = rnorm(1000, mean=30, sd=10)


##create multiple histogram
hist(x1, col='red', xlim=c(-35, 100))
hist(x2, col='green', add=TRUE) # add the histogram inside the previ
ous plot
hist(x3, col='blue', add=TRUE)
```




```r
# Create two example datasets
data1 <- rnorm(100, mean = 0, sd = 1)
data2 <- rnorm(100, mean = 2, sd = 1)


# Set up a side-by-side layout
par(mfrow = c(1, 2)) # make o plot with 1 row of plots and 2 columns


# Create the first histogram
hist(data1, main = "Histogram 1",
     xlab = "Value",
     ylab = "Frequency")


# Create the second histogram
hist(data2,
     main = "Histogram 2",
     xlab = "Value",
     ylab = "Frequency")
```



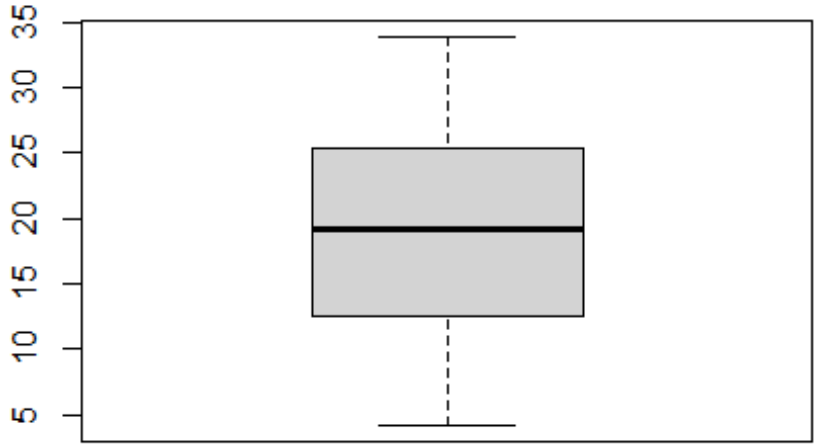
Θηκόγραμμα (boxplot)

Το θηκόγραμμα είναι ένα συνοπτικός τρόπο παρουσίασης πέντε στατιστικών μέτρων. Στην ουσία το θηκόγραμμα είναι ένα ορθογώνιο πλαίσιο στο οποίο παρουσιάζονται βασικές πληροφορίες (διάμεσος, μέση τιμή, ποσοστημόρια, μέγιστη και ελάχιστη τιμή) για το δείγμα δεδομένων που διερευνούμε.

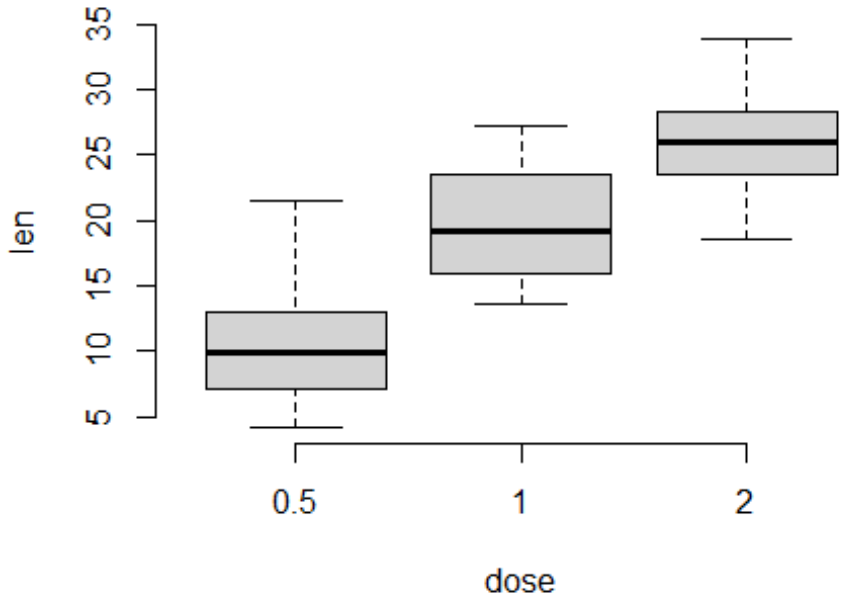
```
# use the ToothGrow dataset and ispect it
head(ToothGrowth)
```

```
   len supp dose
1  4.2   VC  0.5
2 11.5   VC  0.5
3  7.3   VC  0.5
4  5.8   VC  0.5
5  6.4   VC  0.5
6 10.0   VC  0.5
```

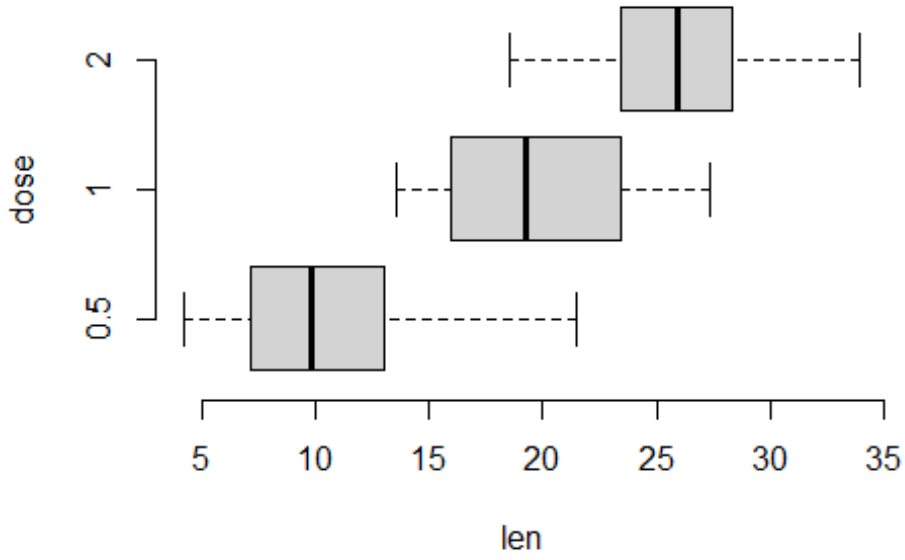
```
# Box plot of one variable
boxplot(ToothGrowth$len)
```



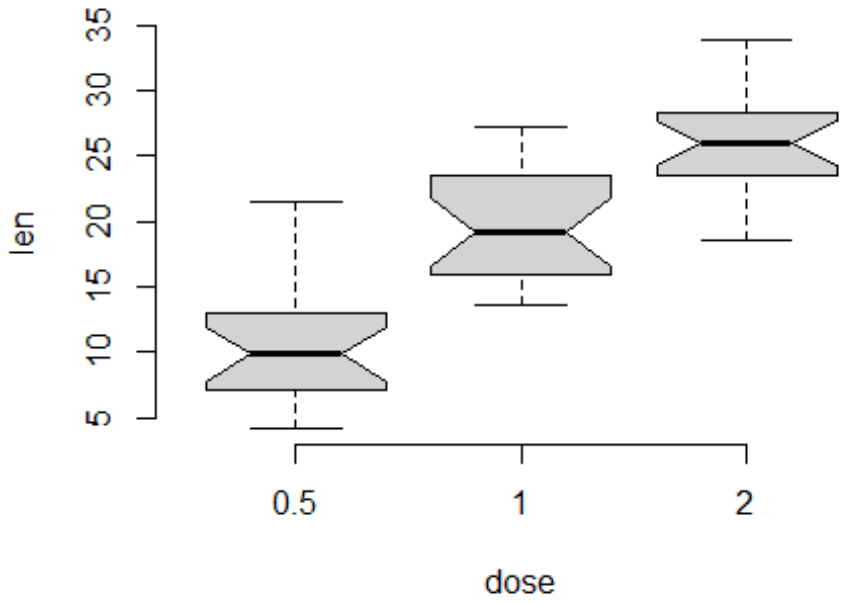

```
# Box plots by groups (dose)
# remove frame
boxplot(len ~ dose, # the vectors (variables) we want to illustrate
        data = ToothGrowth, # the name of the dataset
        frame = FALSE) # without frame
```



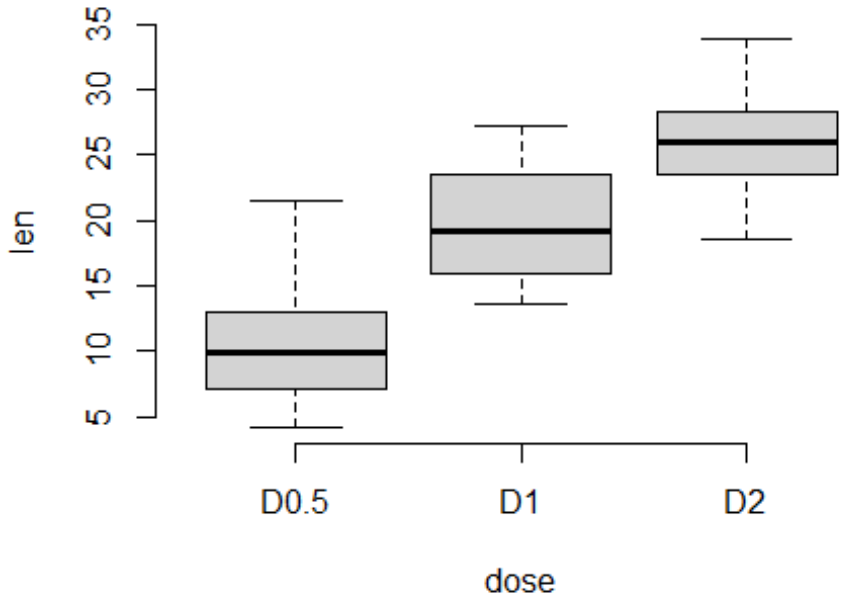
```
# Horizontal box plots (transpose)
boxplot(len ~ dose,
        data = ToothGrowth,
        frame = FALSE,
        horizontal = TRUE) # rotate horizontally
```



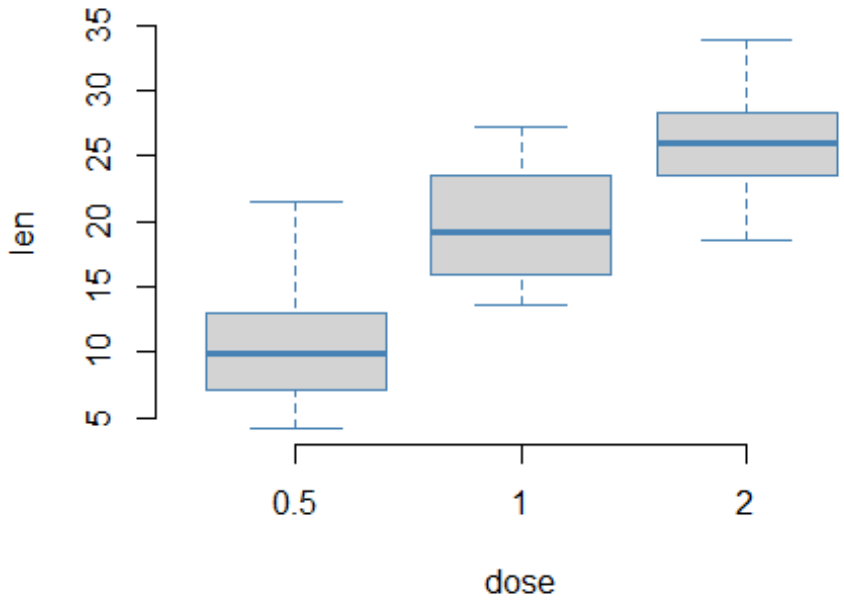

```
# Notched box plots
boxplot(len ~ dose,
        data = ToothGrowth,
        frame = FALSE,
        notch = TRUE) # reduse the shape in the median position
```



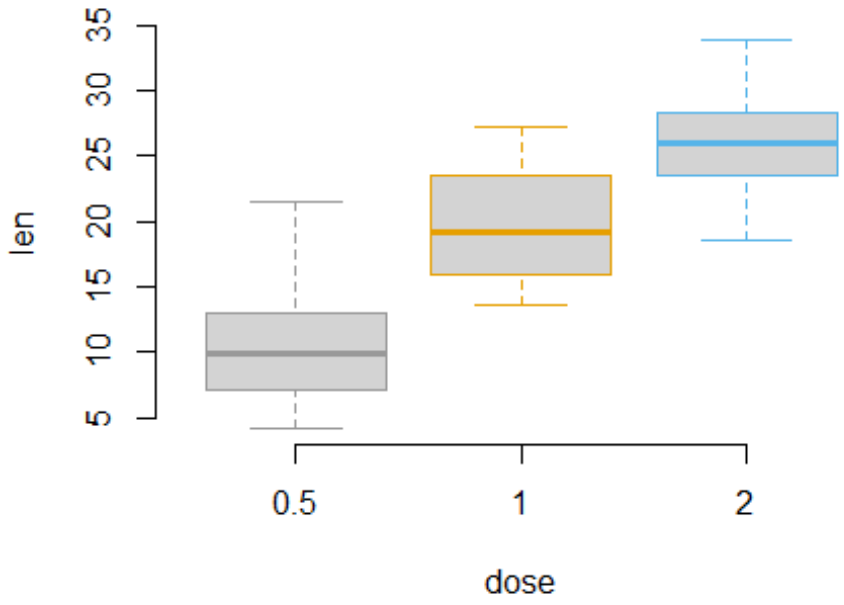

```
# Change group names
boxplot(len ~ dose,
        data = ToothGrowth,
        frame = FALSE,
        names = c("D0.5", "D1", "D2")) # set names
```



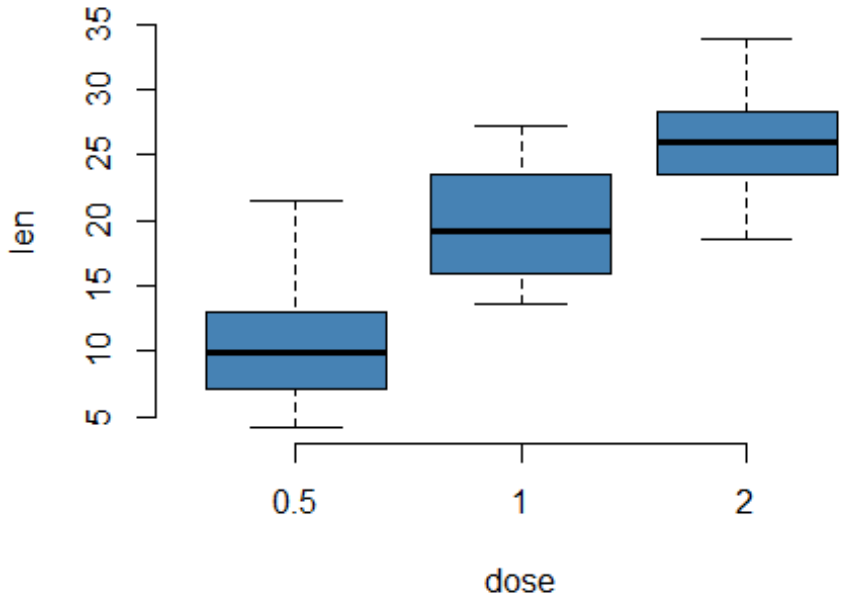
```
# Change the color of border using one single color
boxplot(
  len ~ dose,
  data = ToothGrowth,
  frame = FALSE,
  border = "steelblue" # set the perimeter of the boxes as steelblue
color
)
```



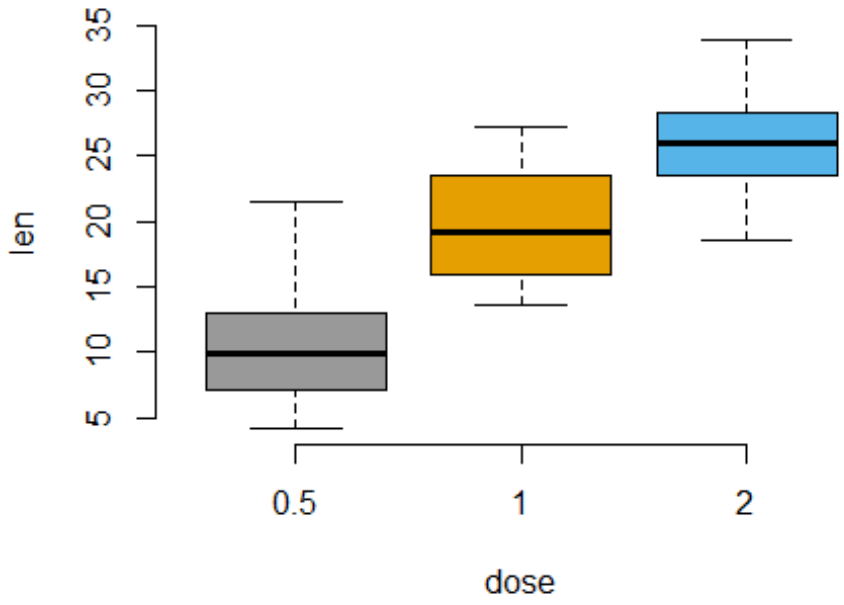

```
# Change the color of border.
#  Use different colors for each group
boxplot(
  len ~ dose,
  data = ToothGrowth,
  frame = FALSE,
  border = c("#999999",
             "#E69F00",
             "#56B4E9") # set manually the perimeter color of each b
ox
)
```




```
# Change fill color: single color
boxplot(len ~ dose,
        data = ToothGrowth,
        frame = FALSE,
        col = "steelblue") # set the fill color
```



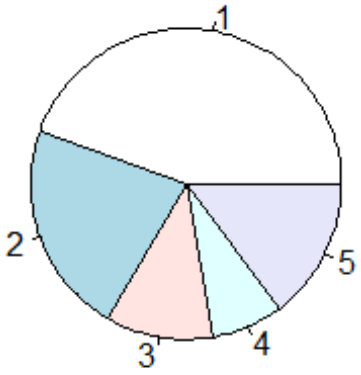
```r
# Change fill color: multiple colors
boxplot(
  len ~ dose,
  data = ToothGrowth,
  frame = FALSE,
  col = c("#999999",
          "#E69F00",
          "#56B4E9")
)
```



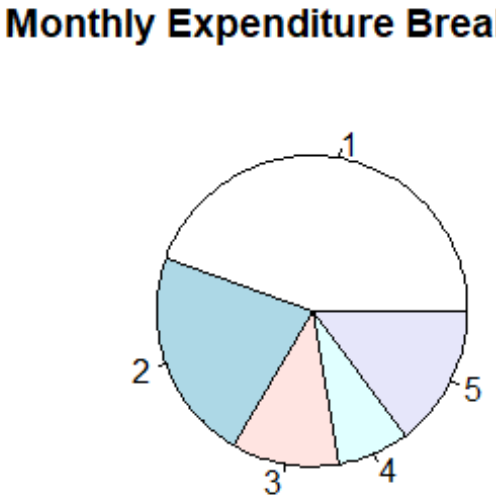
Διαγράμματα πίτας (Pie chart)

Το διάγραμμα πίτας είναι ένας κύκλος του οποίου οι φέτες υποδηλώνουν την αναλογία των διαφορετικών επιπέδων της παραμέτρου που μελετούμε.

```r
# Create a data vector manually
expenditure <- c(600, 300, 150, 100, 200)


# pie chart of of expenditure vector
pie_chart <- pie(expenditure)
```

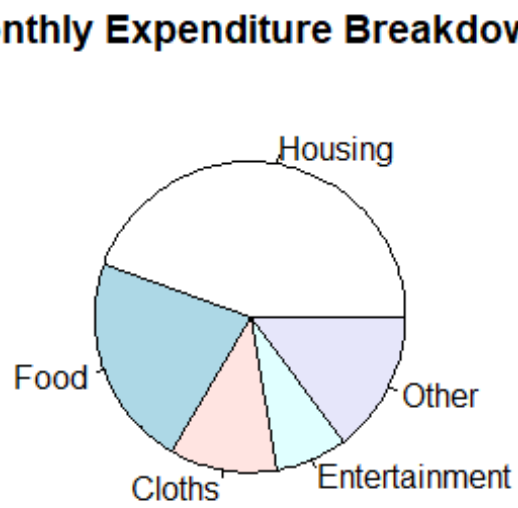



```
# create pie chart with title
pie_chart_title <-
  pie(expenditure,
      main = "Monthly Expenditure Breakdown")
```

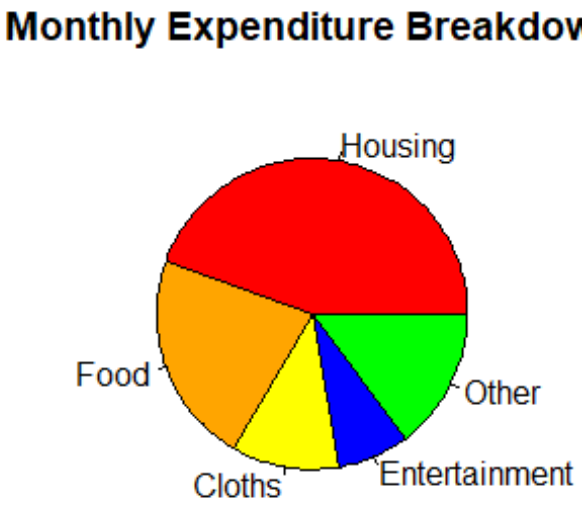



```
# add label for each part of the pie chart
pie_chart_title_partLabel <- pie(expenditure,
```

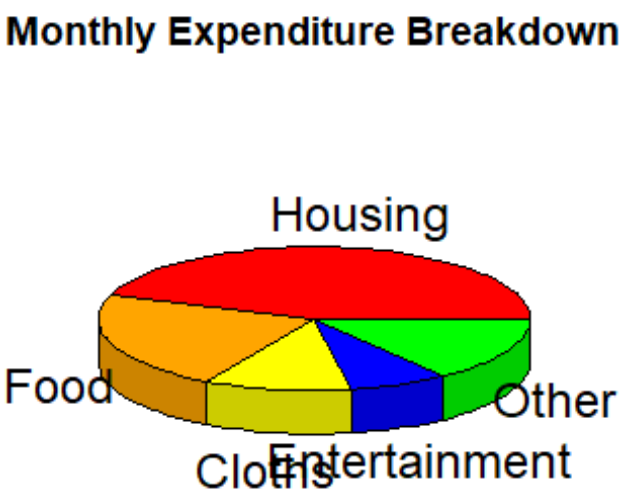
```
  main = "Monthly Expenditure Breakdown",
  labels = c("Housing",
             "Food",
             "Cloths",
             "Entertainment",
             "Other")
)
```




```
# define the color of pies parts
pie_colors <- pie(expenditure,
  main = "Monthly Expenditure Breakdown",
  labels = c("Housing",
             "Food",
             "Cloths",
             "Entertainment",
             "Other"),
  col = c("red",
          "orange",
          "yellow",
          "blue",
          "green")
)
```



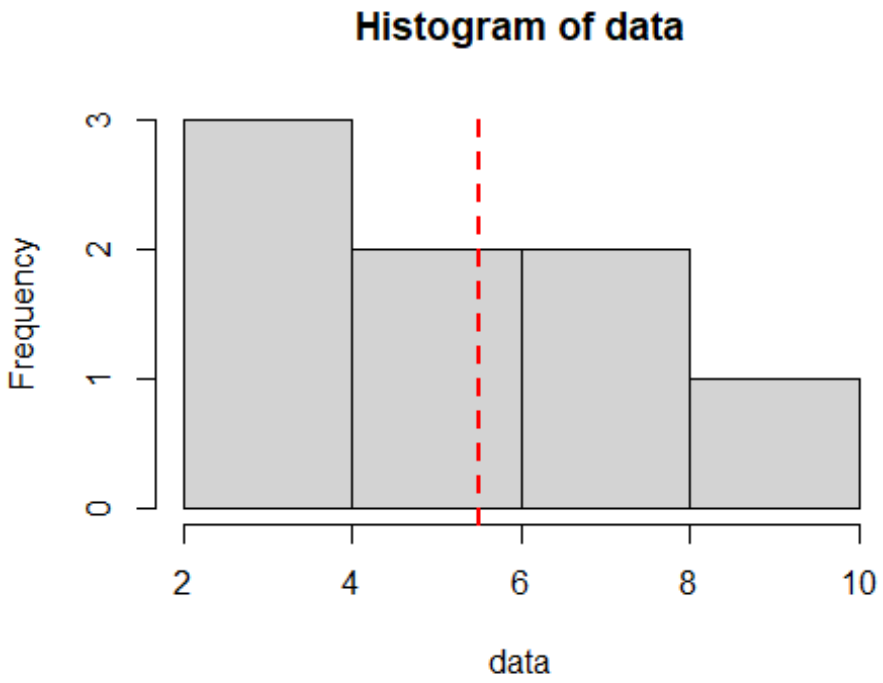

```
# in case we want a 3D pie we use plotrix::pie3D
pie_3D <- plotrix::pie3D(expenditure,
  main = "Monthly Expenditure Breakdown", # Main title
  labels = c("Housing", # define the names
             "Food",
             "Cloths",
             "Entertainment",
             "Other"),
  col = c("red",
          "orange",
          "yellow",
          "blue",
          "green")
)
```



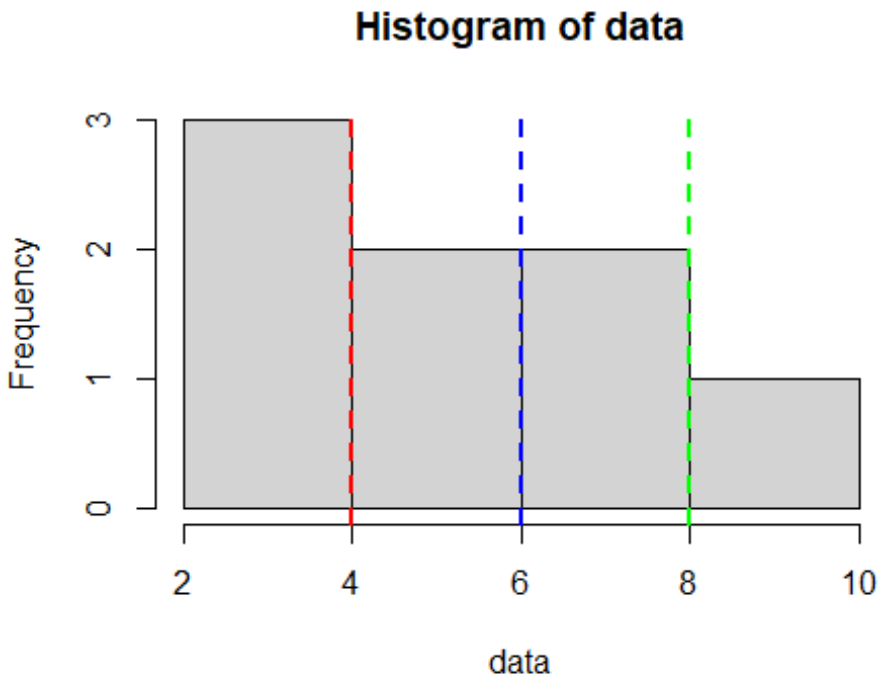
Πρόσθετες γραμμές (Abline and rugs)

Συχνά είναι επιθυμητό να προσθέτουμε γραμμές στα γραφήματα μας που υποδεικνύουν σημαντικές πληροφορίες (όπως η θέση της μέσης τιμής). Επίσης για να γίνουν ορατές οι θέσεις των τιμών των παραμέτρων μπορούν να προστεθούν σημεία στα γραφήματα μας.

```
# create a histogram
# Create a vector of data
data <- c(5, 7, 3, 9, 2, 6, 4, 8)


# Create a histogram to visualize the distribution of data
hist(data)


# Add a vertical line at the mean value of the data with a red dashe
d line
abline(
  v = mean(data), # set the line in the mean value position
  col = 'red', # set the line color
  lwd = 2, # line width
  lty = 'dashed' # line type
)
```



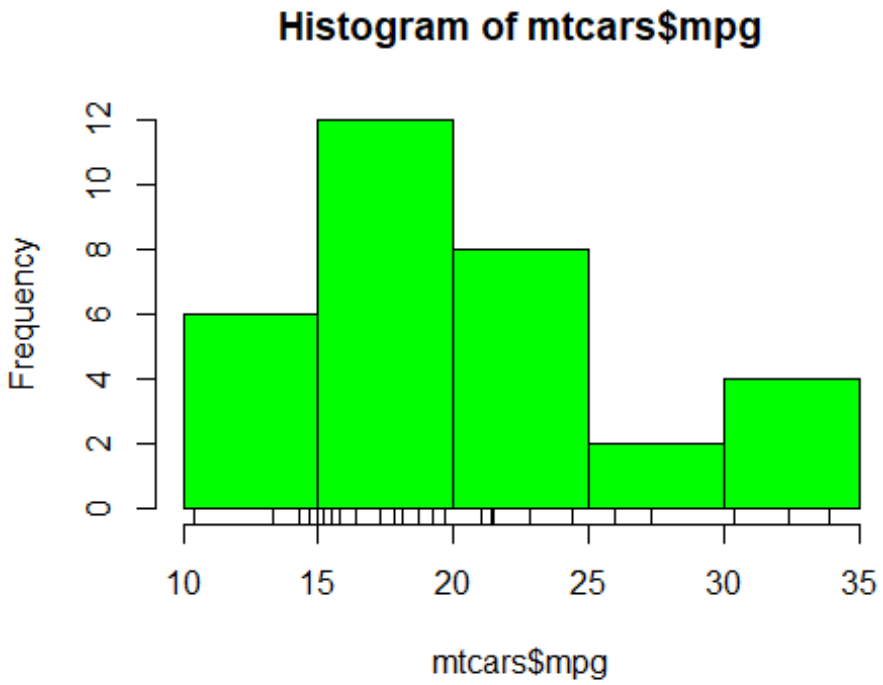

```
# Add specific lines ----
# Create a histogram to visualize the distribution of data
hist(data)


# Add multiple vertical lines at specific locations with different c
olors
abline(
  v = c(4, 6, 8), # set the positions
  col = c('red',
          'blue',
          'green'),
  lwd = 2,
  lty = 'dashed'
)
```




```
# Add rug in a mtcars$mpg data
head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

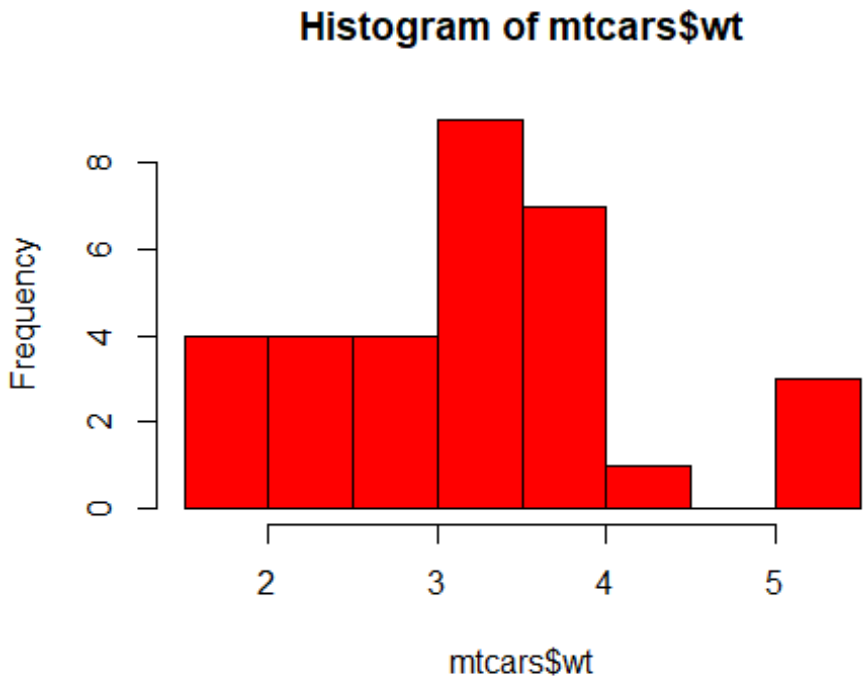
```
hist(mtcars$mpg,
     col="green")

rug(mtcars$mpg)
```



1.3. Εφαρμογές και ασκήσεις

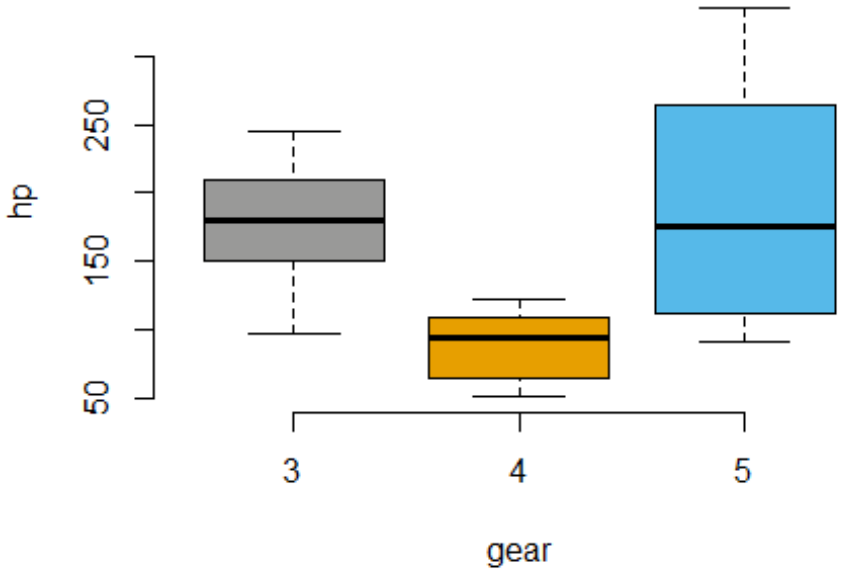
4. Να γίνει ραβδόγραμμα από το dataset mpg για την παράμετρο wt (βάρος) με χρώμα ιστών κόκκινο.

```
# create the histogram by mpg dataset for wt variable with red
color and the cars names as x axis label
hist(mtcars$wt,
     col = "red")
```



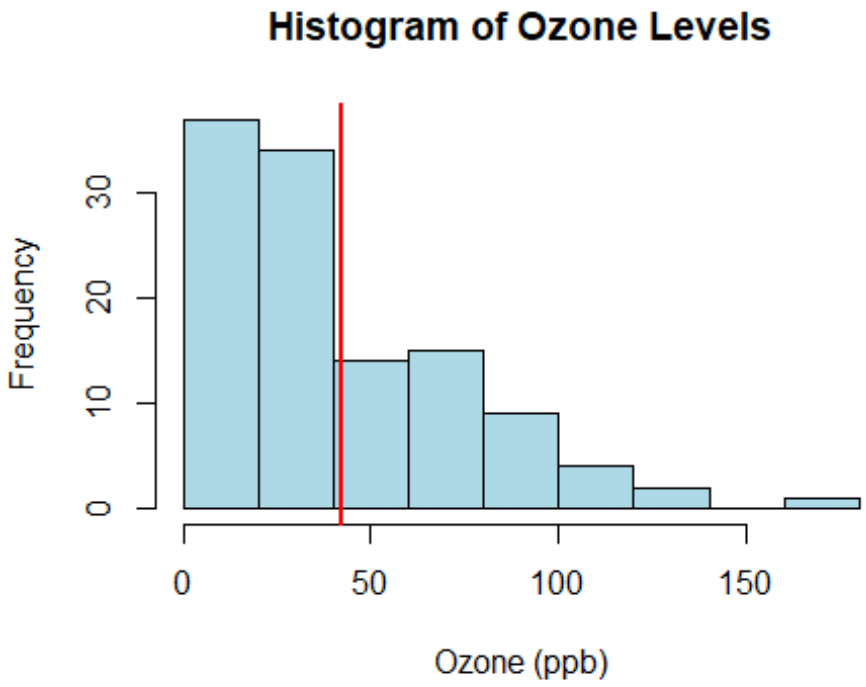
5. Να γίνει θηκόγραμμα από το σύνολο δεδομένων mtcars για τη σχέση ταχυτήτων (gear) με την ιπποδύναμη (hp) με διαφορετικό χρώμα για κάθε ταχύτητα.

```
# Create a boxplot with gear vs hp variables of the mtcars wit
h different color per gear level
boxplot(
  hp~gear, # the names of the columns
  data = mtcars, # the name of the dataset
  frame = FALSE, # disable the frame
  col = c("#999998",
          "#E79F00",
          "#56B9E9")
)
```



6. Να γίνει ιστόγραμμα με την διασπορά της παραμέτρου Ozone του συνόλου δεδομένων airquality και να εμφανίζεται γραμμή στη θέση της μέσης τιμές με χρώμα κόκκινο.

```
# Create a histogram for the Ozone (from airquality) with a red line
indicating the position of the mean value
hist(airquality$Ozone,
     main = "Histogram of Ozone Levels",
     xlab = "Ozone (ppb)",
     col = "lightblue",
     border = "black")


# Calculate the mean of the Ozone column
oz_mean <- mean(airquality$Ozone,
                na.rm = TRUE)


# draw the line in the mean position
abline(
  v = oz_mean, # set the line in the mean value position
  col = 'red', # set the color of the line
  lwd = 2 # set the color width
)
```



Ερωτήσεις αυτοαξιολόγησης

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

1. Ποιο όρισμα αφορά στον τίτλο του άξονα X;

A. main
B. Title
C. xlab
D. col

2. Ποια εντολή είναι κατάλληλη για την προσθήκη γραμμής σε ένα γράφημα;

A. main
B. abline
C. xlab
D. hist

3. Ποια εντολή είναι κατάλληλη για τη δημιουργία ιστογράμματος

A. hist
B. boxplot
C. abline
D. title

4. Ποια εντολή δημιουργεί διαγράμματα ζεύγους τιμών (scatterplots);

A. boxplot
B. pairs
C. hist
D. abline

5. Ποιο όρισμα προσδιορίζει το μέγεθος των σημείων του γραφήματος;
    A. ltw
    B. cex
    C. abline
    D. size

Πηγές - Αναφορές

Οι πηγές που ακολουθούν περιέχουν ενεργούς συνδέσμους (links)

- Summary Statistics and Graphs with R
- Scatter Plot Matrices - R Base Graphs
- Box Plots - R Base Graphs
- Histograms. Rug and abline

Κεφάλαιο 2. Σύνοψη δεδομένων - Πακέτα summarytools και descr

2.1. Υπολογισμός πινάκων συχνοτήτων

Μια από τις βασικές πληροφορίες για την κατανόηση των δεδομένων μας, είναι οι πίνακες συχνοτήτων, δηλαδή πίνακες που το κύριο περιεχόμενό τους είναι η συχνότητα εμφάνισης τιμών ή ομάδων τιμών (κλάσεων - ομάδων).

```
# use the mtcars
data(mtcars)


# inspect the mtcars dataset
str(mtcars)
```

```
'data.frame':   32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

```
# create a frequency table for the cyl column
table(mtcars$cyl)
```

```
 4  6  8
11  7 14
```

```
# create a two-way frequency table for cyl and disp
table(mtcars$cyl, mtcars$disp)
```

```
    71.1 75.7 78.7 79 95.1 108 120.1 120.3 121 140.8 145 146.7 160 167.6 225
  4    1    1    1  1    1   1     1     1   1     1   0     1   0     0   0
  6    0    0    0  0    0   0     0     0   0     0   1     0   2     2   1
  8    0    0    0  0    0   0     0     0   0     0   0     0   0     0   0

    258 275.8 301 304 318 350 351 360 400 440 460 472
  4   0     0   0   0   0   0   0   0   0   0   0   0
  6   1     0   0   0   0   0   0   0   0   0   0   0
  8   0     3   1   1   1   1   1   2   1   1   1   1
```

2.2. Βασικά στατιστικά μεγέθη και παράμετροι

Η γλώσσα R έχει στη βασική της εκδοχή και μέσω των πακέτων της πολλές συναρτήσεις - εντολές που προσφέρουν χρήσιμες πληροφορίες για τα δεδομένα που διαχειριζόμαστε. Μια πολύ συχνά χρησιμοποιούμενη συνάρτηση είναι η summary η οποία μας δίνει για το σύνολο των δεδομένων μας (ή για ένα υποσύνολο) την ελάχιστη τιμή, τη μέγιστη, τη διάμεσο και το πρώτο και τρίτο ποσοστιμόριο.

```r
# show the basic descriptive statistics for iris dataset
summary(iris)
```

```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
 setosa    :50
 versicolor:50
 virginica :50
```

```r
# show the above with one digit
summary(iris,
        digits = 1) # define the number of the digits
```

```
  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width        Species
 Min.   :4     Min.   :2    Min.   :1     Min.   :0.1   setosa    :50
 1st Qu.:5     1st Qu.:3    1st Qu.:2     1st Qu.:0.3   versicolor:50
 Median :6     Median :3    Median :4     Median :1.3   virginica :50
 Mean   :6     Mean   :3    Mean   :4     Mean   :1.2
 3rd Qu.:6     3rd Qu.:3    3rd Qu.:5     3rd Qu.:1.8
 Max.   :8     Max.   :4    Max.   :7     Max.   :2.5
```

```r
# show the descriptive statistics only for the Petal.Lenght column
summary(iris$Petal.Length)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.600   4.350   3.758   5.100   6.900
```

2.3. Εφαρμογές με το πακέτο summarytools

Ένα από τα βασικά πακέτα διερεύνησης των δεδομένων είναι το summarytools το οποίο περιλαμβάνει σημαντικές συναρτήσεις - εντολές που προσφέρουν δυνατότητες δημιουργίας αναφορών (reporting).

```r
# install the package if needed
#install.packages("summarytools")


# load the package
library(summarytools)


# freq()---------------
# create a frequence table from iris data
freq(iris$Species)
```

```
Frequencies
iris$Species
Type: Factor

                  Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
---------------- ------ --------- -------------- --------- --------------
          setosa     50     33.33          33.33     33.33          33.33
      versicolor     50     33.33          66.67     33.33          66.67
       virginica     50     33.33         100.00     33.33         100.00
            <NA>      0                               0.00         100.00
           Total    150    100.00         100.00    100.00         100.00
```

```r
# create a frequence table with rmarkdown style
freq(tobacco$smoker,
     style='rmarkdown')
```

```
### Frequencies
#### tobacco$smoker
**Type:** Factor

|       | Freq | % Valid | % Valid Cum. | % Total | % Total Cum. |
|-----------:|-----:|--------:|-------------:|--------:|-------------:|
|    **Yes** |  298 |   29.80 |        29.80 |   29.80 |        29.80 |
|     **No** |  702 |   70.20 |       100.00 |   70.20 |       100.00 |
```

```
| **\<NA\>** |    0 |         |               |    0.00 |       100.00 |
|  **Total** | 1000 |  100.00 |       100.00 |  100.00 |       100.00 |
```

```
# create a frequence table with rmarkdown style without plain.ascii characters
freq(iris$Species,
     plain.ascii = FALSE,
     style = "rmarkdown")
```

```
### Frequencies
#### iris$Species
**Type:** Factor

|           | Freq | % Valid | % Valid Cum. | % Total | % Total Cum. |
|---------------:|-----:|--------:|-------------:|--------:|-------------:|
|     **setosa** |   50 |   33.33 |        33.33 |   33.33 |        33.33 |
| **versicolor** |   50 |   33.33 |        66.67 |   33.33 |        66.67 |
|  **virginica** |   50 |   33.33 |       100.00 |   33.33 |       100.00 |
|     **\<NA\>** |    0 |         |              |    0.00 |       100.00 |
|      **Total** |  150 |  100.00 |       100.00 |  100.00 |       100.00 |
```

```
# ctable--------------------
# Create cross-tabulations table with tobacco data
head(tobacco)
```

```
  gender age age.gr      BMI smoker cigs.per.day diseased      disease
1      M  75   71 + 29.50225     No            0       No         <NA>
2      F  35  35-50 26.14989     No            0      Yes Neurological
3      F  70  51-70 27.53183     No            0       No         <NA>
4      F  40  35-50 24.05832     No            0       No         <NA>
5      F  75   71 + 22.77486     No            0      Yes      Hearing
6      M  38  35-50 21.46412     No            0       No         <NA>
  samp.wgts
1  1.062500
2  1.044177
3  1.049383
4  1.044177
5  1.062500
6  1.044177
```

```r
# create c-table for gender and smoker
ctable(tobacco$gender,
       tobacco$smoker)
```

```
Cross-Tabulation, Row Proportions
gender * smoker
Data Frame: tobacco

-------- -------- ------------ ------------ ---------------
          smoker          Yes           No           Total
  gender
       F           147 (30.1%)   342 (69.9%)    489 (100.0%)
       M           143 (29.2%)   346 (70.8%)    489 (100.0%)
    <NA>             8 (36.4%)    14 (63.6%)     22 (100.0%)
   Total           298 (29.8%)   702 (70.2%)   1000 (100.0%)
-------- -------- ------------ ------------ ---------------
```

```r
# Use with() to simplify syntax
with(tobacco,
     ctable(smoker, diseased))
```

```
Cross-Tabulation, Row Proportions
smoker * diseased
Data Frame: tobacco

-------- ---------- ------------ ------------ ---------------
          diseased          Yes           No           Total
  smoker
     Yes              125 (41.9%)   173 (58.1%)    298 (100.0%)
      No               99 (14.1%)   603 (85.9%)    702 (100.0%)
   Total              224 (22.4%)   776 (77.6%)   1000 (100.0%)
-------- ---------- ------------ ------------ ---------------
```

```r
with(tobacco,
     ctable(smoker, diseased, # define the variables
            prop = 'n', #Character. Indicates which
            #proportions to show: “r” (rows, default),
            #“c” #(columns),“t” (total),
```

```
            #or "n" (none).
            #Default value can be changed using st_options, option ctable.prop.
            totals = FALSE))
```

```
Cross-Tabulation
smoker * diseased
Data Frame: tobacco

-------- ---------- ----- -----
           diseased   Yes    No
  smoker
     Yes              125   173
      No               99   603
-------- ---------- ----- -----
```

```
ctable(x = tobacco$smoker,
       y = tobacco$diseased,
       prop = "r")   # Show row proportions
```

```
Cross-Tabulation, Row Proportions
smoker * diseased
Data Frame: tobacco

-------- ---------- ------------ ------------ ---------------
           diseased          Yes           No           Total
  smoker
     Yes             125 (41.9%)   173 (58.1%)    298 (100.0%)
      No              99 (14.1%)   603 (85.9%)    702 (100.0%)
   Total             224 (22.4%)   776 (77.6%)   1000 (100.0%)
-------- ---------- ------------ ------------ ---------------
```

```
# descr()-----------------
# descriptive statistics with descr(). Example with iris and exams dataset
descr(iris)
```

```
Descriptive Statistics
iris
N: 150
```

```
                  Petal.Length   Petal.Width   Sepal.Length   Sepal.Width
----------------- -------------- ------------- -------------- -------------
             Mean           3.76          1.20           5.84          3.06
          Std.Dev           1.77          0.76           0.83          0.44
              Min           1.00          0.10           4.30          2.00
               Q1           1.60          0.30           5.10          2.80
           Median           4.35          1.30           5.80          3.00
               Q3           5.10          1.80           6.40          3.30
              Max           6.90          2.50           7.90          4.40
              MAD           1.85          1.04           1.04          0.44
              IQR           3.50          1.50           1.30          0.50
               CV           0.47          0.64           0.14          0.14
         Skewness          -0.27         -0.10           0.31          0.31
      SE.Skewness           0.20          0.20           0.20          0.20
         Kurtosis          -1.42         -1.36          -0.61          0.14
          N.Valid         150.00        150.00         150.00        150.00
        Pct.Valid         100.00        100.00         100.00        100.00
```

```r
# descriptive statistics (mean and sd), transposed, without headings
descr(
  iris,
  stats = c("mean", "sd"),
  transpose = TRUE,
  headings  = FALSE
)
```

```
                    Mean   Std.Dev
------------------ ------ ---------
      Petal.Length   3.76      1.77
       Petal.Width   1.20      0.76
      Sepal.Length   5.84      0.83
       Sepal.Width   3.06      0.44
```

```r
# descriptive statistics with exams data
head(exams)
```

```
             student gender french math geography history economics english
1    Aubin, Éléonore   Girl   44.8 59.3      50.4    53.9        NA    58.3
2     Bennett, David    Boy   94.7 89.8      96.3    93.5      94.2    90.2
3     Bolduc, Pascal    Boy   76.9 93.2      79.1    86.1      89.1    80.9
4   Boucher, Roxanne   Girl   66.4 68.8      66.9    73.9      70.4    69.7
5        Fazl, Josué    Boy   70.5 67.0      69.6    68.6      68.8    70.9
6 Florent, Catherine   Girl   65.3 73.2      68.5    68.2      69.5    72.3
```

```
# use the 3rd to 5th columns with rmarkdown style
descr(exams[, 3:5],
      style = 'rmarkdown')
```

```
### Descriptive Statistics
#### exams
**N:** 30

|            | french | geography |  math |
|----------------:|-------:|----------:|------:|
|        **Mean** |  73.94 |     70.04 | 73.54 |
|     **Std.Dev** |  10.79 |     10.65 |  9.19 |
|         **Min** |  44.80 |     47.20 | 55.60 |
|          **Q1** |  68.20 |     65.90 | 66.95 |
|      **Median** |  73.60 |     68.50 | 73.75 |
|          **Q3** |  76.70 |     77.80 | 80.35 |
|         **Max** |  94.70 |     96.30 | 93.20 |
|         **MAD** |   7.56 |     12.31 |  9.93 |
|         **IQR** |   8.50 |     11.90 | 13.35 |
|          **CV** |   0.15 |      0.15 |  0.12 |
|    **Skewness** |   0.03 |      0.10 |  0.12 |
| **SE.Skewness** |   0.43 |      0.43 |  0.44 |
|    **Kurtosis** |   0.45 |     -0.03 | -0.58 |
|     **N.Valid** |  29.00 |     29.00 | 28.00 |
|   **Pct.Valid** |  96.67 |     96.67 | 93.33 |
```

```
# transposed table
descr(exams,
      style = 'rmarkdown',
      transpose = TRUE)
```

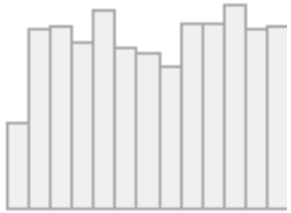
```
### Descriptive Statistics
#### exams
**N:** 30

|          |  Mean | Std.Dev |   Min |    Q1 | Median |    Q3 |   Max |   MAD |   IQR |
|--------------:|------:|--------:|------:|------:|-------:|------:|------:|------:|------:|
| **economics** | 73.91 |    8.62 | 60.50 | 68.80 |  71.60 | 77.00 | 94.20 |  5.49 |  8.20 |
|   **english** | 75.96 |    7.92 | 58.30 | 70.90 |  74.10 | 80.60 | 93.10 |  6.52 |  9.70 |
|    **french** | 73.94 |   10.79 | 44.80 | 68.20 |  73.60 | 76.70 | 94.70 |  7.56 |  8.50 |
| **geography** | 70.04 |   10.65 | 47.20 | 65.90 |  68.50 | 77.80 | 96.30 | 12.31 | 11.90 |
|   **history** | 72.77 |   10.20 | 53.90 | 68.20 |  72.75 | 76.50 | 93.50 |  6.45 |  8.15 |
|      **math** | 73.54 |    9.19 | 55.60 | 66.95 |  73.75 | 80.35 | 93.20 |  9.93 | 13.35 |

Table: Table continues below



|          |   CV | Skewness | SE.Skewness | Kurtosis | N.Valid | Pct.Valid |
|--------------:|-----:|---------:|------------:|---------:|--------:|----------:|
| **economics** | 0.12 |     0.75 |        0.43 |    -0.42 |   29.00 |     96.67 |
|   **english** | 0.10 |     0.28 |        0.43 |    -0.25 |   29.00 |     96.67 |
|    **french** | 0.15 |     0.03 |        0.43 |     0.45 |   29.00 |     96.67 |
| **geography** | 0.15 |     0.10 |        0.43 |    -0.03 |   29.00 |     96.67 |
|   **history** | 0.14 |     0.01 |        0.43 |    -0.60 |   30.00 |    100.00 |
|      **math** | 0.12 |     0.12 |        0.44 |    -0.58 |   28.00 |     93.33 |

# dfsummary()------
# create summary table with iris and tobacco data
view(dfSummary(iris))


# creates separate image (as html)
dfSummary(tobacco,
          plain.ascii  = FALSE, # when TRUE no markup characters used (printer friendly)
          style        = "grid",
          graph.magnif = 0.75,
          tmp.img.dir  = "/tmp") # the directory to store the temporary image
```

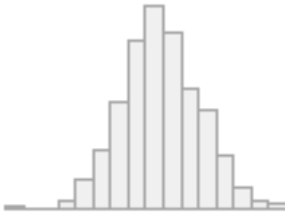
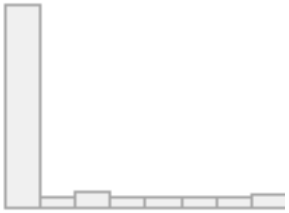
Data Frame Summary
tobacco
Dimensions: 1000 x 9
Duplicates: 2

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
1	gender\ [factor]	1\. F\ 2\. M	489 (50.0%)\ 489 (50.0%)		978\ (97.8%)	22\ (2.2%)
2	age\ [numeric]	Mean (sd) : 49.6 (18.3)\ min < med < max:\ 18 < 50 < 80\ IQR (CV) : 32 (0.4)	63 distinct values		975\ (97.5%)	25\ (2.5%)
3	age.gr\ [factor]	1\. 18-34\ 2\. 35-50\ 3\. 51-70\ 4\. 71 +	258 (26.5%)\ 241 (24.7%)\ 317 (32.5%)\ 159 (16.3%)		975\ (97.5%)	25\ (2.5%)
4	BMI\ [numeric]	Mean (sd) : 25.7 (4.5)\ min < med < max:\ 8.8 < 25.6 < 39.4\ IQR (CV) : 5.7 (0.2)	974 distinct values		974\ (97.4%)	26\ (2.6%)
5	smoker\ [factor]	1\. Yes\ 2\. No	298 (29.8%)\ 702 (70.2%)		1000\ (100.0%)	0\ (0.0%)
6	cigs.per.day\ [numeric]	Mean (sd) : 6.8 (11.9)\ min < med < max:\ 0 < 0 < 40\ IQR (CV) : 11 (1.8)	37 distinct values		965\ (96.5%)	35\ (3.5%)
7	diseased\ [factor]	1\. Yes\ 2\. No	224 (22.4%)\ 776 (77.6%)		1000\ (100.0%)	0\ (0.0%)
8	disease\ [character]	1\. Hypertension\ 2\. Cancer\ 3\. Cholesterol\ 4\. Heart\	36 (16.2%)\ 34 (15.3%)\ 21 (9.5%)\ 20 (9.0%)\		222\ (22.2%)	778\ (77.8%)

```
|    |               | 5\. Pulmonary\           | 20 ( 9.0%)\         |                           |          |         |
|    |               | 6\. Musculoskeletal\     | 19 ( 8.6%)\         |                           |          |         |
|    |               | 7\. Diabetes\            | 14 ( 6.3%)\         |                           |          |         |
|    |               | 8\. Hearing\             | 14 ( 6.3%)\         |                           |          |         |
|    |               | 9\. Digestive\           | 12 ( 5.4%)\         |                           |          |         |
|    |               | 10\. Hypotension\        | 11 ( 5.0%)\         |                           |          |         |
|    |               | [ 3 others ]             | 21 ( 9.5%)          |                           |          |         |
+----+---------------+--------------------------+---------------------+---------------------------+----------+---------+
| 9  | samp.wgts\    | Mean (sd) : 1 (0.1)\     | 0.86!: 267 (26.7%)\ | ![](/tmp/ds0270.png) \    | 1000\    | 0\      |
|    | [numeric]     | min < med < max:\        | 1.04!: 249 (24.9%)\ | \                         | (100.0%) | (0.0%)  |
|    |               | 0.9 < 1 < 1.1\           | 1.05!: 324 (32.4%)\ |                           |          |         |
|    |               | IQR (CV) : 0.2 (0.1)     | 1.06!: 160 (16.0%)\ |                           |          |         |
|    |               |                          | ! rounded           |                           |          |         |
+----+---------------+--------------------------+---------------------+---------------------------+----------+---------+
```

```
print(dfSummary(
  tobacco,
  varnumbers   = FALSE,
  graph.magnif = 0.76),
  method = 'render')
```

Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
gender [factor]	1. F	489	(50.0%	978 (97.8%)	22 (2.2%)
	2. M	489	(50.0%	)	
age [numeric]	Mean (sd) : 49.6 (18.3) min ≤ med ≤ max: 18 ≤ 50 ≤ 80 IQR (CV) : 32 (0.4)	63 distinct values		975 (97.5%)	25 (2.5%)

Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
age.gr [factor]	1. 18-34	258	(26.5%	975 (97.5%)	25 (2.5%)
	2. 35-50	241	(24.7%	)	
	3. 51-70	317	(32.5%	)	
	4. 71 +	159	(16.3%	)	
BMI [numeric]	Mean (sd) : 25.7 (4.5) min ≤ med ≤ max: 8.8 ≤ 25.6 ≤ 39.4 IQR (CV) : 5.7 (0.2)	974 distinct values		974 (97.4%)	26 (2.6%)
smoker [factor]	1. Yes	298	(29.8%	1000 (100.0%)	0 (0.0%)
	2. No	702	(70.2%	)	
cigs.per.day [numeric]	Mean (sd) : 6.8 (11.9) min ≤ med ≤ max: 0 ≤ 0 ≤ 40 IQR (CV) : 11 (1.8)	37 distinct values		965 (96.5%)	35 (3.5%)
diseased [factor]	1. Yes	224	(22.4%	1000 (100.0%)	0 (0.0%)
	2. No	776	(77.6%	)	

Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
disease [character]	1. Hypertension	36	(16.2%	222 (22.2%)	778 (77.8%)
	2. Cancer	34	(15.3%	)	
	3. Cholesterol	21	(9.5%	)	
	4. Heart	20	(9.0%	)	
	5. Pulmonary	20	(9.0%	)	
	6. Musculoskeletal	19	(8.6%	)	
	7. Diabetes	14	(6.3%	)	
	8. Hearing	14	(6.3%	)	
	9. Digestive	12	(5.4%	)	
	10. Hypotension	11	(5.0%	)	
	[3 others]	21	(9.5%	)	
samp.wgts [numeric]	Mean (sd) : 1 (0.1)	0.86 !	: (	1000 (100.0%)	0 (0.0%)
	min ≤ med ≤ max:	1.04 !	: (	24.9%)	
	0.9 ≤ 1 ≤ 1.1	1.05 !	: (	32.4%)	
	IQR (CV) : 0.2 (0.1)	1.06 !	: (	16.0%)	
		! rounded			

2.4. Βοηθητικές συναρτήσεις

```
# the unique() function----

# create a vector
vec <- c(11, 19, 11, 21, 46, 19)


# find the unique values of the matrix
unique(vec)
```

```
[1] 11 19 21 46
```

```
# create a matrix
my_data <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 4, 5, 6, 7, 8, 9)

mtrx <- matrix(my_data,
               nrow = 5,
               ncol = 3,
               byrow = TRUE)
print(mtrx)
```

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

```
# show the matrix with duplicate rows and columns removed
unique(mtrx)
```

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

```
# find the unique rows of a data frame
# create a data frame
df <- data.frame(
  col1 = c(1, 2, 3, 2, 3),
  col2 = c(4, 5, 6, 5, 6),
  col3 = c(7, 8, 9, 8, 9)
)


df
```

```
  col1 col2 col3
1    1    4    7
2    2    5    8
3    3    6    9
```

```
4    2    5    8
5    3    6    9
```

```
unique(df)
```

```
  col1 col2 col3
1    1    4    7
2    2    5    8
3    3    6    9
```

```
# find the unique values of specific column
unique(df$col3)
```

```
[1] 7 8 9
```

```
# the cut() function. Classdify the values------
# create a vector
data <- c(1, 2, 3, 4, 5)


# create bins with equal size
cut_data <- cut(data,
                breaks = 2)

cut_data
```

```
[1] (0.996,3] (0.996,3] (0.996,3] (3,5]     (3,5]
Levels: (0.996,3] (3,5]
```

```
# open and closed intervals
cut_data <- cut(data,
                breaks = 2,
                right = FALSE)

cut_data
```

```
[1] [0.996,3) [0.996,3) [3,5)     [3,5)     [3,5)
Levels: [0.996,3) [3,5)
```

```
# define the number of bins
cut_data <- cut(data,
                breaks = 4)

cut_data
```

```
[1] (0.996,2] (0.996,2] (2,3]     (3,4]     (4,5]
Levels: (0.996,2] (2,3] (3,4] (4,5]
```

2.5. Εφαρμογές με το πακέτο descr

```
# install the library if needed
# install.packages("descr")


# load library
```

```r
library(descr)


# create a matrix data
my_data <- matrix(c(23, 45, 21, 52),
                  nrow = 2,
                  ncol = 2)

my_data
```

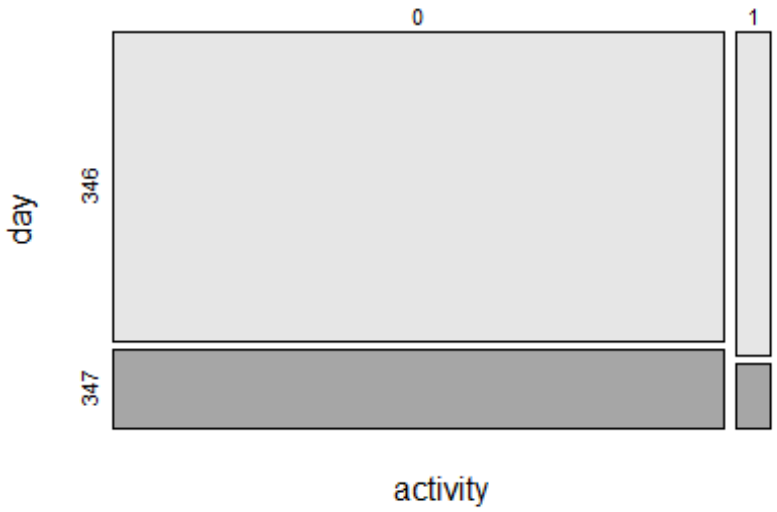
```
     [,1] [,2]
[1,]   23   21
[2,]   45   52
```

```r
descr::CrossTable(
  my_data,
  digits = 2, # the number of digits for my results
  chisq = T,  # print chisquare test results
  fisher = T, # print the results of Fisher Exact test
  resid = T,  # print residuals (Persons)
  sresid = T, # prints standarised residuals
  asresid = T, # adjusted standardized residual will be included
  format = "SPSS") # SAS default and SPSS format
```

```
   Cell Contents
|-------------------------|
|                   Count |
| Chi-square contribution |
|             Row Percent |
|          Column Percent |
|           Total Percent |
|                Residual |
|            Std Residual |
|            Adj Std Resid |
|-------------------------|

==================================
           [,1]     [,2]    Total
----------------------------------
[1,]         23       21       44
          0.15     0.14
         52.27%   47.73%   31.21%
         33.82%   28.77%
         16.31%   14.89%
          1.78    -1.78
          0.39    -0.37
          0.65    -0.65
----------------------------------
[2,]         45       52       97
          0.07     0.06
         46.39%   53.61%   68.79%
         66.18%   71.23%
         31.91%   36.88%
```

```
         -1.78     1.78
         -0.26     0.25
         -0.65     0.65
--------------------------------
Total       68       73      141
         48.23%   51.77%
================================

Statistics for All Table Factors

Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 = 0.419286      d.f. = 1      p = 0.52

Pearson's Chi-squared test with Yates' continuity correction
------------------------------------------------------------
Chi^2 = 0.216829      d.f. = 1      p = 0.64


Fisher's Exact Test for Count Data
------------------------------------------------------------
Sample estimate odds ratio: 1.26348

Alternative hypothesis: true odds ratio is not equal to 1
p = 0.59
95% confidence interval: 0.5830932 2.751215

Alternative hypothesis: true odds ratio is less than 1
p = 0.8
95%s confidence interval: % 0 2.450527

Alternative hypothesis: true odds ratio is greater than 1
p = 0.32
95%s confidence interval: % 0.6536088 Inf

        Minimum expected frequency: 21.21986
```

```
# Cross tabulation with a mosaic plot
descr::crosstab(beaver1$day, # the dependent variable
                beaver1$activ, # the independent variable
                beaver1$temp, # the cross tabulation weight variable
                chisq = T,
                xlab = "activity",
                ylab="day")
```




```
   Cell Contents
|-------------------------|
|                   Count |
|-------------------------|

==================================
               beaver1$activ
beaver1$day        0      1   Total
----------------------------------
346             3169    186    3355
----------------------------------
347              810     37     847
----------------------------------
Total           3979    223    4202
==================================

Statistics for All Table Factors

Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 = 1.859843      d.f. = 1      p = 0.173

Pearson's Chi-squared test with Yates' continuity correction
------------------------------------------------------------
Chi^2 = 1.633264      d.f. = 1      p = 0.201
        Minimum expected frequency: 44.95026
```

Ερωτήσεις αυτοαξιολόγησης

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

6. Ποια είναι η βασική εντολή από την οποία λαμβάνουμε τα βασικά στατιστικά μεγέθη

A. pairs
B. summary
C. freq
D. plot

7. Ποια εντολή μας δίνει τη συχνότητα των παρατηρήσεων

A. table
B. plot
C. summary
D. mean

8. Από το πακέτο summarytools ποια εντολή μας δίνει πίνακα συχνοτήτων;

A. max
B. freq
C. summary
D. mean

9. Από το πακέτο descr ποια εντολή δίνει πίνακα βασικών στατιστικών μεγεθών;

A. freq
B. CrossTable
C. head
D. max

10. Ποια εντολή δίνει τις πρώτες 10 γραμμές ενός πίνακα δεδομένων;

A. tail(dataname)
B. head(dataname)
C. head(dataname,10)
D. max(dataname)

Πηγές - Αναφορές

Οι πηγές που ακολουθούν περιέχουν ενεργούς συνδέσμους (links)

- Introduction to summarytools
- descr: Descriptive Statistics (r-project.org)

Κεφάλαιο 3. Εισαγωγή στη δημιουργία γραφημάτων με το πακέτου ggplot2

3.1. Εισαγωγή στη δημιουργία γραφημάτων με το πακέτο ggplot2

Η R γλώσσα διαθέτει ένα μεγάλο πλήθος επιλογών για τη δημιουργία γραφημάτων αλλά το πακέτο ggplot2 θεωρείται μια από τις πιο πολύπλευρες και καλαίσθητες από αυτές. Η βιβλιοθήκη ggplot2 ουσιαστικά υλοποιεί μια γραμματική κανόνων γραφικών η οποία είναι ένα συνεκτικό σύστημα περιγραφής και δημιουργία γραφημάτων. Χρησιμοποιώντας το ggplot2 μπορούμε να μάθουμε έναν τρόπο δημιουργίας γραφημάτων που θα εφαρμόζεται σε διάφορες περιπτώσεις. Για τη χρήση του ggplot2 και των παραδειγμάτων που ακολουθούν απαιτείται η χρήση της βιβλιοθήκης tidyverse.

Η λογική και η σύνταξη του πακέτου ggplot2 είναι συνυφασμένη με το πακέτο tidyverse. Εξάλλου, σήμερα το ggplot2 αποτελεί μέρος της συλλογής των πακέτων της tidyverse μαζί με άλλα χρήσιμα πακέτα όπως το dplyr, tibble και tidyr. Πρέπει να σημειωθεί πως το ggplot2 εφαρμόζεται σε δεδομένα που έχουν τη μορφή data frame ή tibble. Ακολουθεί η εντολή φόρτωσης της βιβλιοθήκης tidyverse.

```
library(tidyverse)
```

Προφανώς αν δεν έχει εγκατασταθεί ήδη το πακέτο tidyverse τότε θα απαιτηθεί η εγκατάσταση του με την προφανή ακόλουθη εντολή.

install.packages(“tidyverse”)

Το πακέτο ggplot2 παράγει στατιστικά γραφήματα τα οποία δομούνται σε επίπεδα. Ουσιαστικά χρησιμοποιεί μια δομή που χαρακτηρίζεται ως γραμματική η οποία κτίζει το γράφημα τμηματικά σε επίπεδα ρυθμίσεων ενώ το βασικό πακέτο δημιουργίας γραφημάτων της R παρέχει απλώς προκατασκευασμένα γραφήματα- αποτελέσματα κλήσεων συναρτήσεων. Επιπλέον η χρήση του πακέτου ggplot2 δεν απαιτεί από το χρήστη να εμπλακεί στο χειρισμό της γραμματικής αλλά σίγουρα αυτός που γνωρίζει τη σύνταξη της γραμματικής έχει ένα σημαντικό πλεονέκτημα. Με αυτόν τον τρόπο το πακέτο ggplot2 επιτρέπει στο χρήστη να δημιουργήσει ένα γράφημα με βάση τις έννοιες που θέλει να παρουσιάσει και όχι με την απλή ανάκληση πλήθους εντολών και επιλογών.

Το πρώτο γράφημα που θα παρουσιαστεί απαντά στο τυπικό ερώτημα εάν υπάρχει κάποια σχέση μεταξύ του συνολικού ημερήσιου χρόνου ύπνου και του χρόνου ύπνου rem των θηλαστικών. Η απάντηση είναι μάλλον είναι προφανής αλλά εμείς χρειαζόμαστε κάτι παραπάνω από τη διαίσθηση για να δείξουμε τη σχέση μεταξύ των δύο χρόνων. Θα χρησιμοποιήσουμε το data frame msleep που είναι ενσωματωμένο στο ggplot2 και το οποίο περιέχει τα δεδομένα 83 θηλαστικών. Ας δούμε το περιεχόμενο του.

```
msleep
```

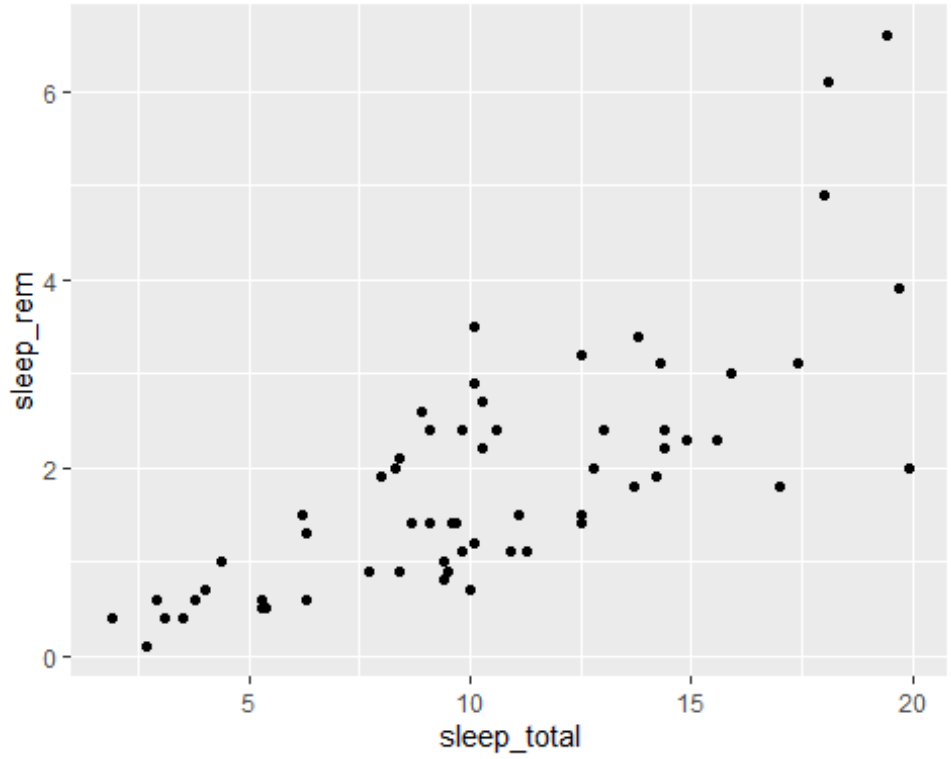
```
# A tibble: 83 × 11
   name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
   <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
 1 Cheet… Acin… carni Carn… lc                  12.1      NA         NA      11.9
 2 Owl m… Aotus omni  Prim… <NA>                17         1.8       NA       7
```

```
 3 Mount… Aplo… herbi Rode… nt                  14.4       2.4       NA        9.6
 4 Great… Blar… omni  Sori… lc                  14.9       2.3        0.133    9.1
 5 Cow    Bos   herbi Arti… domesticated        4          0.7        0.667   20
 6 Three… Brad… herbi Pilo… <NA>                14.4       2.2        0.767    9.6
 7 North… Call… carni Carn… vu                   8.7       1.4        0.383   15.3
 8 Vespe… Calo… <NA>  Rode… <NA>                 7        NA         NA       17
 9 Dog    Canis carni Carn… domesticated        10.1       2.9        0.333   13.9
10 Roe d… Capr… herbi Arti… lc                   3        NA         NA       21
# ℹ 73 more rows
# ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>
```

Οι μεταβλητές του msleep που μας ενδιαφέρουν είναι η sleep_total που εκφράζει το συνολικό ημερήσιο χρόνο ύπνου σε ώρες και η sleep_rem που εκφράζει το χρόνο ύπνου rem. Η αίσθηση μας είναι πως όσο αυξάνει ο ένας χρόνος τόσο αυξάνει και ο άλλος.

Ας παρουσιαστεί όμως ο κώδικας που δημιουργεί ένα απλό γράφημα με το ggplot2.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem))
```



Γράφημα 1

Όπως είναι προφανές, το γράφημα δείχνει την ύπαρξη μιας θετικής συσχέτισης μεταξύ του συνολικού ημερήσιου χρόνου ύπνου και του χρόνου ύπνου rem των θηλαστικών. Η κλήση της συνάρτησης ggplot() δημιουργεί ένα σύστημα συντεταγμένων στο οποίο μπορούμε να προσθέσουμε επίπεδα (layers). Συνεπώς η κλήση της ggplot(data=msleep) απλώς δημιουργεί ένα κενό γράφημα. Στη συνέχεια συμπληρώνουμε το γράφημα προσθέτοντας ένα ή περισσότερα επίπεδα. Η συνάρτηση geom_point() προσθέτει ένα επίπεδο γραφήματος σημείων το οποίο σχηματίζει το διάγραμμα διασποράς. Η ggplot2 διαθέτει πολλές συναρτήσεις geom οι οποίες προσθέτουν διαφορετικού τύπου γεωμετρικά αντικείμενα. Κάθε συνάρτηση geom δέχεται ένα όρισμα mapping το οποίο ορίζει το πως θα απεικονιστούν οι μεταβλητές ως οπτικές ιδιότητες. Το όρισμα mapping πάντα ζευγαρώνει με τη συνάρτηση aes() ενώ τα ορίσματα x και y

της aes() προσδιορίζουν ποιες μεταβλητές θα αναπαρίστανται στους δύο άξονες x, y του γραφήματος.

Η γενική μορφή της σύνταξης της απλής χρήσης της ggplot2 που χρησιμοποιήθηκε στο προηγούμενο παράδειγμα είναι η ακόλουθη:

ggplot(data = <Πηγή Δεδομένων>) + <Συνάρτηση Γεωμετρικού Αντικειμένου>(mapping = aes(<Αντιστοιχίσεις αισθητικών στοιχείων>))

3.2. Η γραμματική των γραφικών με επίπεδα

Όμως τι είναι μια γραμματική γραφημάτων; Όπως σε μια οποιαδήποτε γλώσσα η γραμματική ορίζει τους κανόνες δόμησης λέξεων και φράσεων έτσι ώστε να δημιουργούνται εκφράσεις με νόημα. Μια γραμματική των γραφημάτων ορίζει τους κανόνες της δόμησης μαθηματικών και αισθητικών στοιχείων για να συντίθεται ένα γράφημα με νόημα. Ποια είναι τα επιμέρους στοιχεία αυτής της γραμματικής;

Στις προηγούμενες παραγράφους είχε παρουσιαστεί ένα πρότυπο για τη γενική μορφή της απλής χρήσης της συνάρτησης ggplot. Το πρότυπο αυτό μπορεί να αλλάξει προσθέτοντας όλες τις δυνατότητες που θα γνωρίσουμε στην επόμενη ενότητα. Έτσι μια οποιαδήποτε έκφραση της γραμματικής αυτής περιγράφεται επαρκώς από την ακόλουθη γενική μορφή.

ggplot(data = <Πηγή Δεδομένων>) +

<Συνάρτηση Γεωμετρικού Αντικειμένου>(mapping = aes(<Αντιστοιχίσεις αισθητικών στοιχείων>), stat = <Στατιστική Συνάρτηση>, position = <Θέση>) +

<Συνάρτηση Συστήματος Συντεταγμένων> +

<Συνάρτηση ’Οψεων>

Στην πράξη σπάνια χρειαζόμαστε τη ρύθμιση και των επτά παραμέτρων του γενικού προτύπου γιατί το πακέτο ggplot διαθέτει χρήσιμες προκαθορισμένες τιμές εκτός των δεδομένων των αντιστοιχίσεων και της συνάρτησης του γεωμετρικού αντικειμένου. ’Όμως οι επτά παράμετροι του προτύπου συνθέτουν τη γραμματική των γραφικών δηλαδή ένα καλά δομημένο σύστημα για τη δημιουργία γραφικών. Η λογική της είναι πως ο χρήστης περιγράφει με έναν μοναδικό τρόπο οποιοδήποτε γράφημα ακολουθώντας συστηματικά τα εξής βήματα:

7. Επιλογή των δεδομένων
8. Επιλογή του κατάλληλου γεωμετρικού αντικειμένου
9. Αντιστοίχιση των δεδομένων στο γεωμετρικό αντικείμενο
10. Υπολογισμός του κατάλληλου στατιστικού μέτρου
11. Προσαρμογή της θέσης
12. Επιλογή του συστήματος συντεταγμένων
13. Καθορισμός όψεων

Με άλλα λόγια σταδιακά συνδυάζονται τα δεδομένα, δηλαδή οι μεταβλητές που αντιστοιχίζονται σε αισθητικά χαρακτηριστικά του, τα γεωμετρικά αντικείμενα δηλαδή τα αντικείμενα- σχήματα του γραφήματος, οι στατιστικοί μετασχηματισμοί που συνοψίζουν τα δεδομένα, οι κλίμακες που είναι οι αντιστοιχίσεις τιμών των μεταβλητών (τα υπομνήματα και οι άξονες οπτικοποιούν τις κλίμακες), τα συστήματα συντεταγμένων που είναι το επίπεδο όπου σχεδιάζεται το γράφημα και η δημιουργία όψεων που διαχωρίζουν τα δεδομένα σε υποσύνολα εμφανίζοντας πολλές εκφάνσεις του ίδιου γραφήματος.

3.3. Η συνάρτηση ggplot() και τα αισθητικά στοιχεία

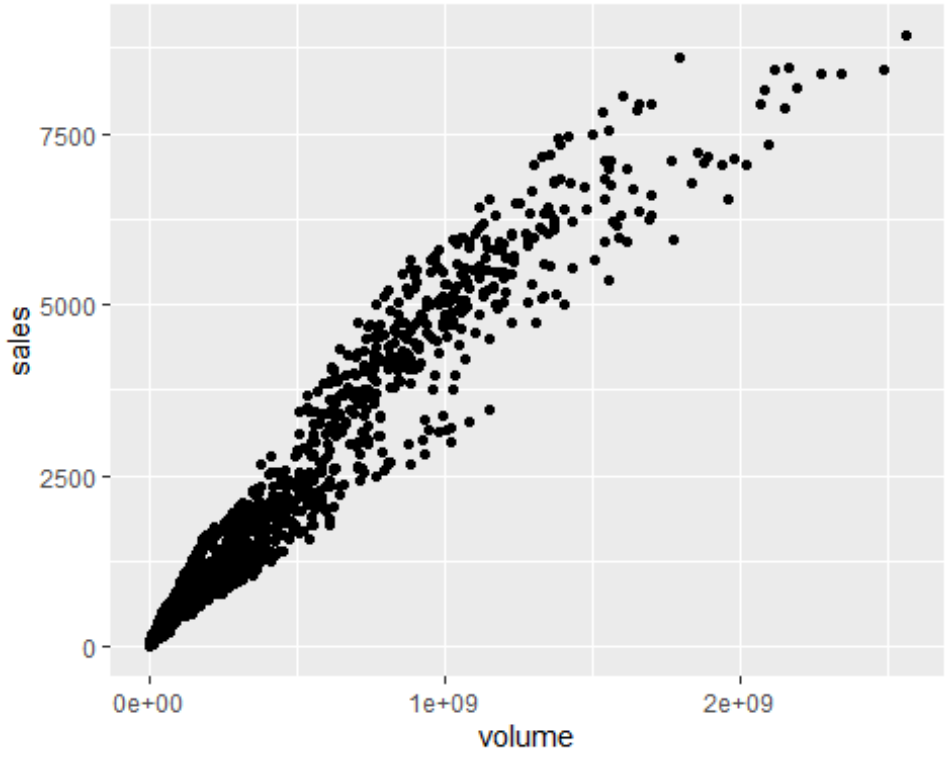
Όπως έχει ήδη γραφεί όλα τα γραφήματα δημιουργούνται μετά την κλήση της συνάρτησης ggplot(). Εδώ πρέπει να τονιστεί πως ενώ το πακέτο ονομάζεται ggplot2 η συνάρτηση καλείται απλώς ggplot() και δημιουργεί γραφήματα μίας μόνο διάστασης. Μέσα στη συνάρτηση προσδιορίζονται το σύνολο δεδομένων που περιέχει τις μεταβλητές οι οποίες θα αντιστοιχιστούν (mapping) στα αισθητικά στοιχεία (aesthetics), δηλαδή τις οπτικές ιδιότητες του γραφήματος. Πρέπει να σημειωθεί πως το σύνολο δεδομένων έχει τη δομή του data frame αντικειμένου. Η πιο απλή σύνταξη της εντολής είναι η ακόλουθη η οποία εμφανίζει ένα κενό γράφημα.

```
ggplot(data=txhousing)
```


Γράφημα 2

Ας παρουσιαστεί ένα παράδειγμα χρήσης της συνάρτησης για τη δημιουργία ενός γραφήματος από το ίδιο data frame.

```
ggplot(data=txhousing)+
  geom_point(aes(x=volume, y=sales))
```



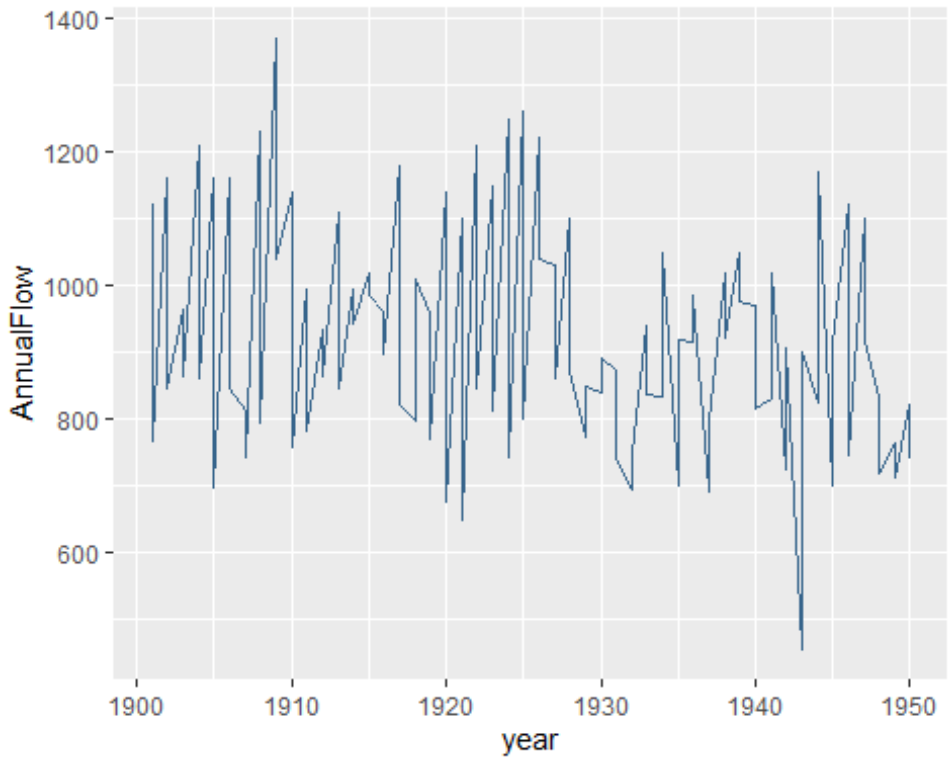
Γράφημα 3

Στο παράδειγμα το σύνολο δεδομένων είναι το txhousing ενώ η μεταβλητή volume αντιστοιχίζεται στον άξονα x και η μεταβλητή sales στον άξονα y. Έτσι το γεωμετρικό αντικείμενο δημιουργίας γραφήματος σημείων αποκτά αυτές τις ρυθμίσεις για τους άξονες.

Ένα απλό παράδειγμα δημιουργίας ενός γραφήματος

Ένα άλλο γεωμετρικό αντικείμενο, το αντικείμενο της γραμμής, το geom_line, είναι κατάλληλο για τη αναπαράσταση δεδομένων μίας διάστασης. Για παράδειγμα με τον παρακάτω κώδικα θα εξερευνήσουμε τη διακύμανση του όγκου της ετήσιας πλημμύρας του Νείλου τα πενήντα πρώτα χρόνια του εικοστού αιώνα. Στην πραγματικότητα θα χρησιμοποιήσουμε και μία δεύτερη διάσταση τη διάσταση του χρόνου. Τα δεδομένα βρίσκονται σε μία ιδιαίτερη μορφή (χρονοσειρά) αλλά στη συνέχεια μετατρέπονται σε μορφή data frame.

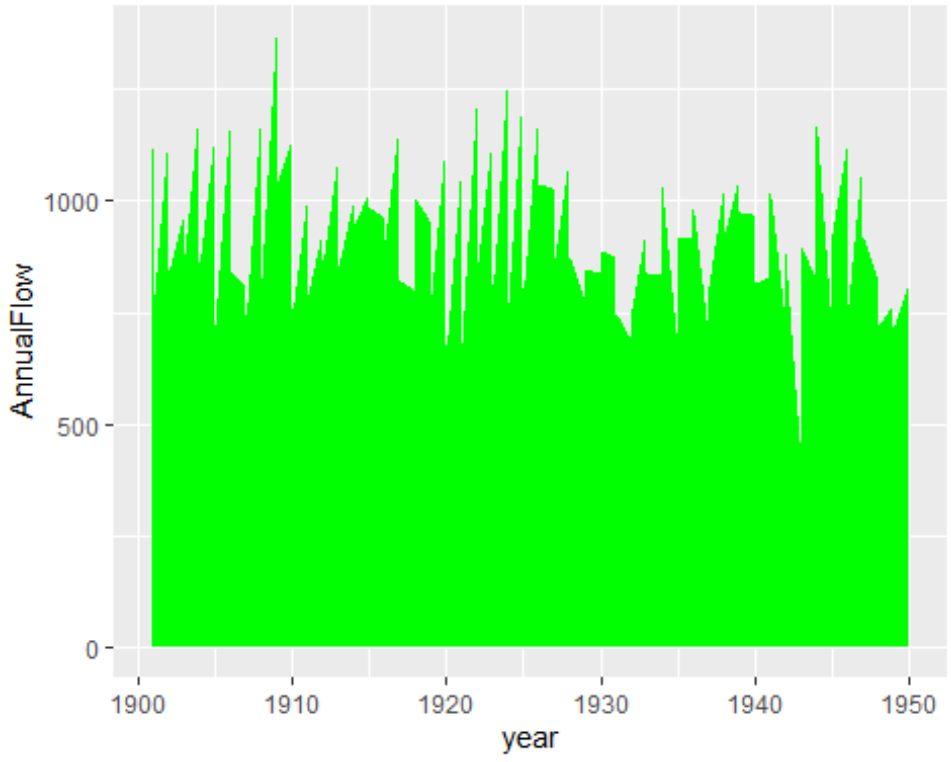
```
Floods <- data.frame("year"=1901:1950, "AnnualFlow"=as.numeric(Nile))
ggplot(data=Floods)+
  geom_line(aes(x=year, y=AnnualFlow), color="steelblue4")
```



Γράφημα 4

Ας χρησιμοποιήσουμε ένα άλλο γεωμετρικό αντικείμενο (geom_area) για την εναλλακτική παρουσίαση των δεδομένων σε μορφή επιφάνειας.

```
ggplot(data=Floods)+
  geom_area(aes(x=year, y=AnnualFlow), fill="green")
```



Γράφημα 5

3.4. Σύγκριση του πακέτου ggplot2 και του βασικού πακέτου δημιουργίας γραφημάτων της R.

Προφανώς όπως έχει ήδη κατανοητό σε ένα περιβάλλον ανοικτού κώδικα όπως είναι αυτό της R υπάρχουν πολλοί διαφορετικοί τρόποι για να επιτευχθεί το ίδιο αποτέλεσμα. Αυτό το επιχείρημα έχει ακόμη μεγαλύτερη ισχύ στην οπτικοποίηση των δεδομένων. Σε προηγούμενη ενότητα παρουσιάστηκε ένα σύνολο συναρτήσεων που διατίθενται με την τυπική έκδοση της R, το βασικό πακέτο γραφημάτων.

Όμως υπάρχουν αρκετά πακέτα που επεκτείνουν τη λειτουργικότητα του βασικού πακέτου και αναντίρρητα το πιο δημοφιλές είναι το ggplot2. Σήμερα οι περισσότεροι χρήστες επιλέγουν ένα από τα δύο, το βασικό πακέτο ή το πακέτο ggplot2. Κάποιοι ισχυρίζονται πως δεν μπορεί να θεωρηθεί κάποιο από τα δύο ως καλύτερο από το άλλο καθώς και τα δύο μπορούν να επιτύχουν το ίδιο αποτέλεσμα και με την ίδια ταχύτητα. Προφανώς υπάρχουν διαφορές που επηρεάζουν τον τρόπο δημιουργία των γραφημάτων καθώς και την παραγωγικότητα των χρηστών. Σις επόμενες παραγράφους ακολουθεί μια λίστα με τις εμφανείς διαφορές.

- Όπως παρουσιάστηκε το ggplot2 υλοποιεί μια γραμματική γραφικών που έχει ως βασική ιδέα ότι ο χρήστης διασπά το γράφημα σε επιμέρους γραφικά αντικείμενα τα οποία τα αντιμετωπίζει ως διακριτά επίπεδα. Συνεπώς το γράφημα που προκύπτει από το ggplot2 δημιουργείται όταν αυτά τα αντικείμενα "κουμπώσουν" μαζί. Ένας χρήστης που εργάζεται ήδη με το βασικό πακέτο θα δυσκολευτεί με αυτή τη διαφορετική λογική καθώς έχει συνηθίσει στη λογική της χρήσης μίας συνάρτησης για κάθε τύπο γραφήματος.

- Το πακέτο ggplot2 διαθέτει και ενσωματωμένες δυνατότητες διαχείρισης δεδομένων όπως την προκαθορισμένη χρήση στατιστικών για κάθε γεωμετρικό αντικείμενο ενώ στην τυπική R η επεξεργασία των δεδομένων γίνεται αυστηρά εκτός των γραφικών συναρτήσεων.

- Όταν το γράφημα χειρίζεται πολλές μεταβλητές τότε το πακέτο ggplot2 προσφέρει έτοιμες χρήσιμες επιλογές όπως το υπόμνημα (legend) ενώ το τυπικό πακέτο απαιτεί την κλήση μίας επιπλέον συνάρτησης.

- Αναμφισβήτητα το ggplot2 προσφέρει καλύτερη εμφάνιση στα γραφήματα με τις προκαθορισμένες επιλογές του και σίγουρα καλύτερη οργάνωση. Το μοτίβο του κώδικα είναι τυπικά ορισμένο και από τη στιγμή που γνωρίζει πως τα τμήματα του δένουν μεταξύ τους εύκολα ο χρήστης περνά από τον έναν τύπο γραφήματος στους επόμενους.

- Ένα από τα πλεονεκτήματα του ggplot2 είναι η δυνατότητα δημιουργίας όψεων (facets) που είναι πολύ χρήσιμες κατά την εξερευνητική ανάλυση των δεδομένων. Η ίδια δυνατότητα για να παραχθεί με το βασικό πακέτο απαιτεί αρκετές γραμμές κώδικα.

Βέβαια πρέπει να τονιστεί πως οτιδήποτε γίνεται με το ένα πακέτο μπορεί να γίνει και με το άλλο. Έτσι και αλλιώς και οι δύο προσεγγίσεις καταλήγουν σε κώδικα R.

Σύμφωνα με άλλους χρήστες της R υπάρχουν διαθέσιμα πολλά πακέτα που έχουν δημιουργηθεί ειδικά για κάποιους τύπους γραφημάτων. Ωστόσο, εάν δεν θέλουμε να φορτώνουμε πολλά πακέτα τότε και το βασικό πακέτο της R είναι πάντα μία καλή διαθέσιμη επιλογή. Ακολουθεί μια συνοπτική λίστα με τους λόγους επιλογής του πακέτου ggplot2 έναντι του βασικού πακέτου:

- αυτόματα υπομνήματα, χρώματα κ.α.
- το προκαθορισμένο αποτέλεσμα είναι πιο εμφανίσιμο από το αποτέλεσμα του βασικού πακέτου
- εύκολος συνδυασμός πολλών συνόλων δεδομένων σε ένα γράφημα
- μεγάλη ποικιλία πρόσθετων που επεκτείνουν το πακέτο ggplot2
- απλή διαδικασία εναλλαγής συστημάτων συντεταγμένων
- καλύτερη διεπαφή χρήστη
- ευελιξία και διαισθητικότητα

Κλείνοντας αυτή την υποενότητα πρέπει να επισημανθεί πως το πακέτο ggplot2 διαθέτει ένα σημαντικό πλεονέκτημα που είναι η δυνατότητα υλοποίησης στατιστικών αναλύσεων ταυτόχρονα με τη δημιουργία του γραφήματος. Ένα τυπικό παράδειγμα αυτής της δυνατότητας είναι το γεωμετρικό αντικείμενο geom_smooth το οποίο σχεδιάζει μία γραμμή που προσαρμόζεται στα δεδομένα και θα παρουσιαστεί αναλυτικά στην επόμενη ενότητα. Πιο συγκεκριμένα βασίζεται στο συνδυασμό πολυωνυμικής προσαρμογής και μέσης τιμής.

Ερωτήσεις αυτοαξιολόγησης για την ενότητα 4.14

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

11. Ποιο από τα παρακάτω δεν είναι χαρακτηριστικό του πακέτου ggplot2;

α. Χρησιμοποιεί επίπεδα για τη δημιουργία του γραφήματος

β. Βασίζεται σε συναρτήσεις δημιουργίας γραφημάτων

γ. Δίνει τη δυνατότητα εμπλοκής πολλών μεταβλητών σε ένα γράφημα

δ. Είναι φιλικό στο χρήστη (αισθητικά)

12. Ποιο επίπεδο επιτρέπει στο πακέτο ggplot2 να διαχωρίσει τα δεδομένα του γραφήματος σε υποσύνολα;

α. Σύστημα συντεταγμένων

β. Γεωμετρικό αντικείμενο

γ. Στατιστικό

δ. Όψεις (facets)

13. Έστω ότι επιθυμείτε να εμφανίσετε τέσσερα διαφορετικά σχήματα σε ένα γράφημα. Τότε θα πρέπει να χρησιμοποιήσετε:

α. τέσσερις φορές τη συνάρτηση ggplot2()

β. μία φορά τη συνάρτηση ggplot()

γ. μία φορά τη συνάρτηση ggplot2()

δ. τέσσερις φορές τη συνάρτηση ggplot()

14. Το πακέτο ggplot2 χειρίζεται δεδομένα που έχουν δομηθεί ως:

α. data frame

β. matrix

γ. vector

δ. Subset

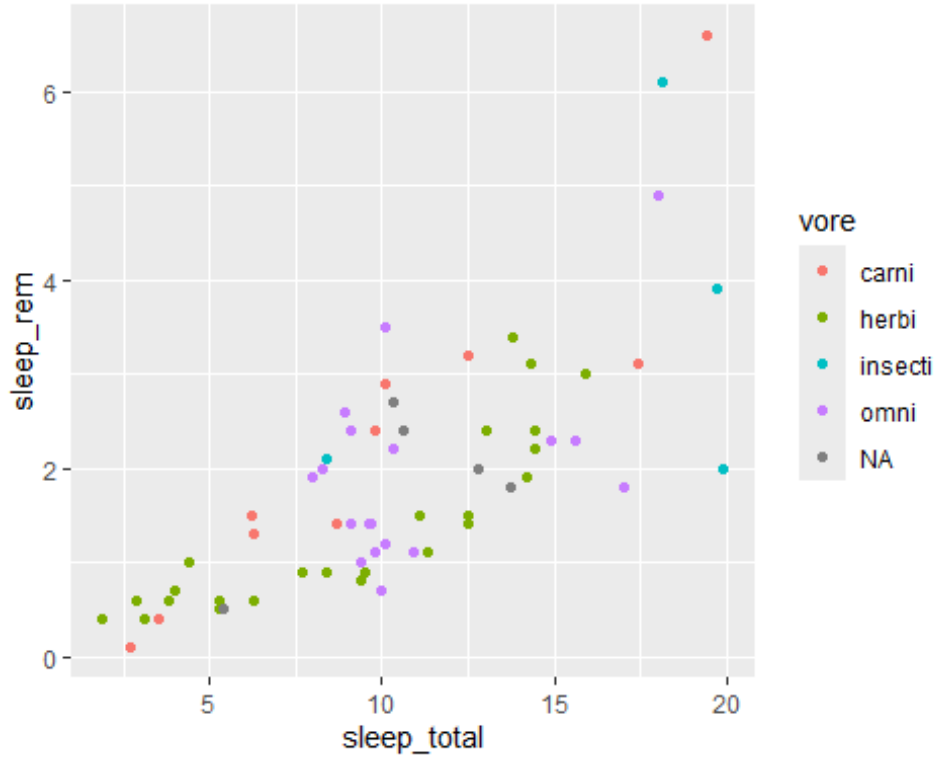
Κεφάλαιο 4. Χρήση του πακέτου γραφημάτων ggplot2

4.1. Χρήση του πακέτου γραφημάτων ggplot2

4.1.1. Αισθητικές αντιστοιχίσεις

Μπορούμε να προσθέσουμε στο πρώτο γράφημα που παρουσιάστηκε στην προηγούμενη ενότητα μια τρίτη μεταβλητή με μια αντιστοίχιση της σε ένα aesthetic. Αυτό είναι μια οπτική ιδιότητα των αντικειμένων που υπάρχουν στο γράφημα. Τέτοιες ιδιότητες είναι το μέγεθος, το σχήμα, το χρώμα των σημείων κλπ. Οι διαφορετικές τιμές αυτών των ιδιοτήτων μπορούν να ονομαστούν ως επίπεδα. Ας ξανασχεδιάσουμε το προηγούμενο γράφημα προσθέτοντας τη μεταβλητή vore (το είδος της διατροφής των θηλαστικών) ως χρώμα των σημείων.

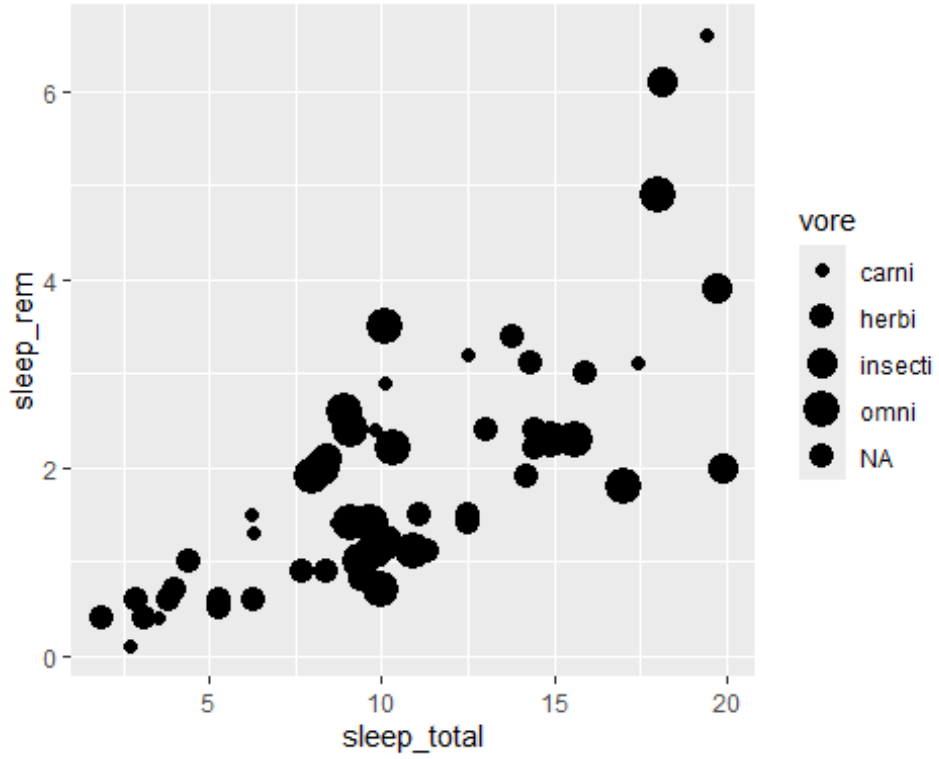
```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem, color = vore))
```



Γράφημα 6

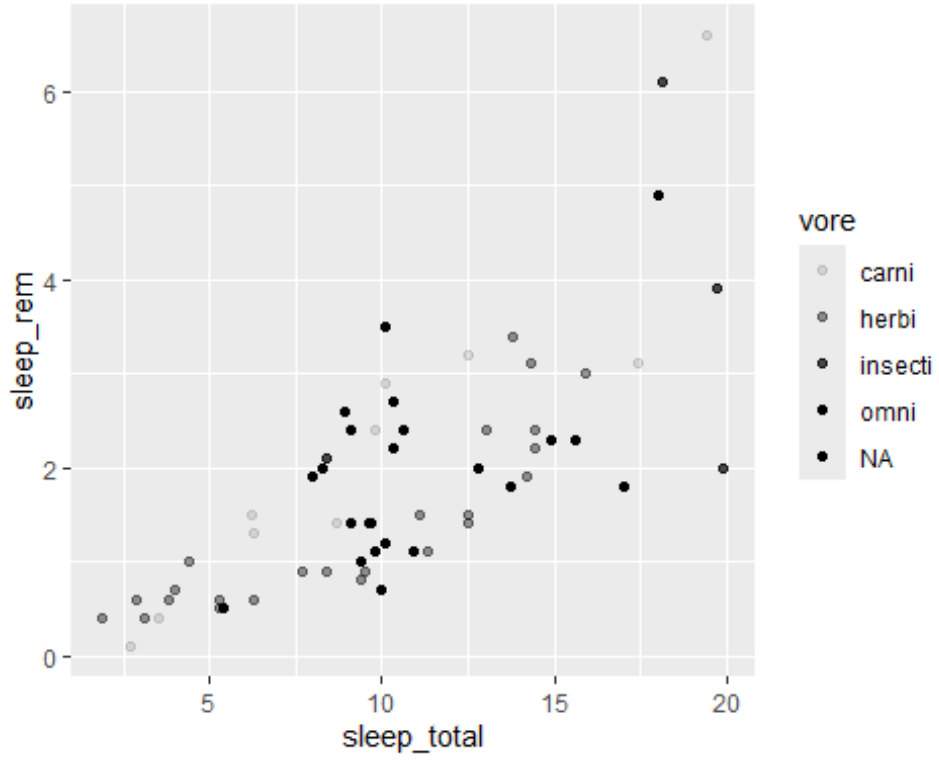
Για να γίνει η αντιστοίχιση ενός aesthetic σε μια μεταβλητή απλά τα συνδέουμε μέσα στη συνάρτηση aes(). Το πακέτο ggplot2 μέσα από μια αυτόματη διαδικασία που ονομάζεται scaling αποδίδει ένα μοναδικό επίπεδο του aesthetic. Εδώ αποδίδεται ένα μοναδικό χρώμα σε κάθε μοναδική τιμή της μεταβλητής ενώ προσθέτει και έναν τίτλο στο υπόμνημα χρωμάτων. Τα επόμενα τμήματα κώδικα δείχνουν πως μπορούμε να χρησιμοποιήσουμε άλλα aesthetics όπως το μέγεθος, το βαθμό διαφάνειας και το σχήμα των σημείων.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem, size = vore))
```



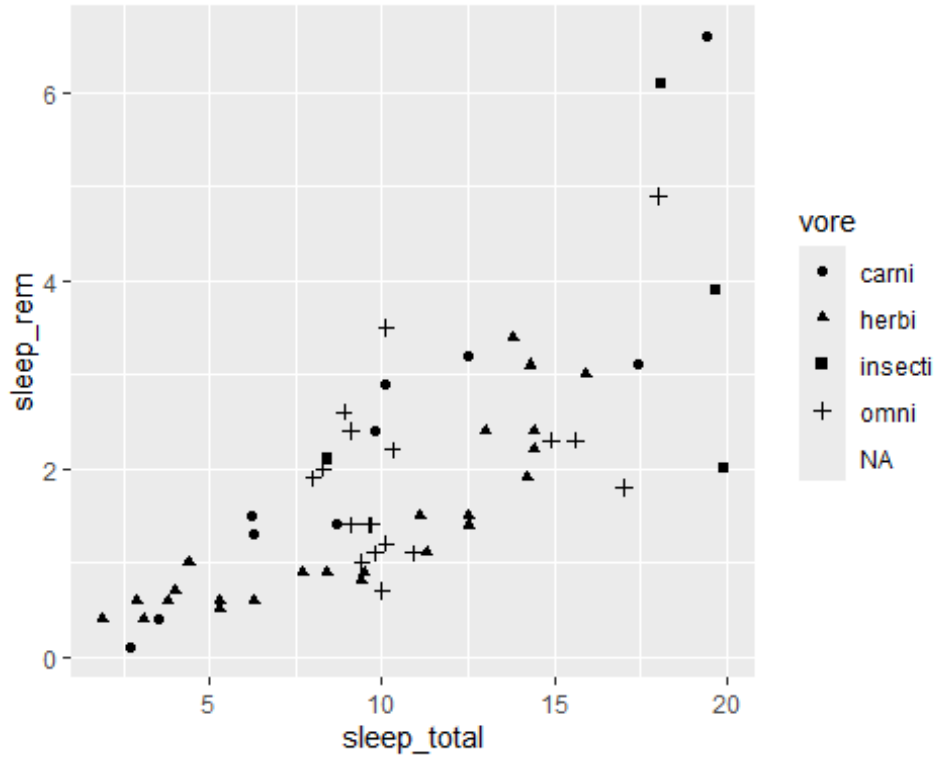
Γράφημα 7

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem, alpha = vore))
```



Γράφημα 8

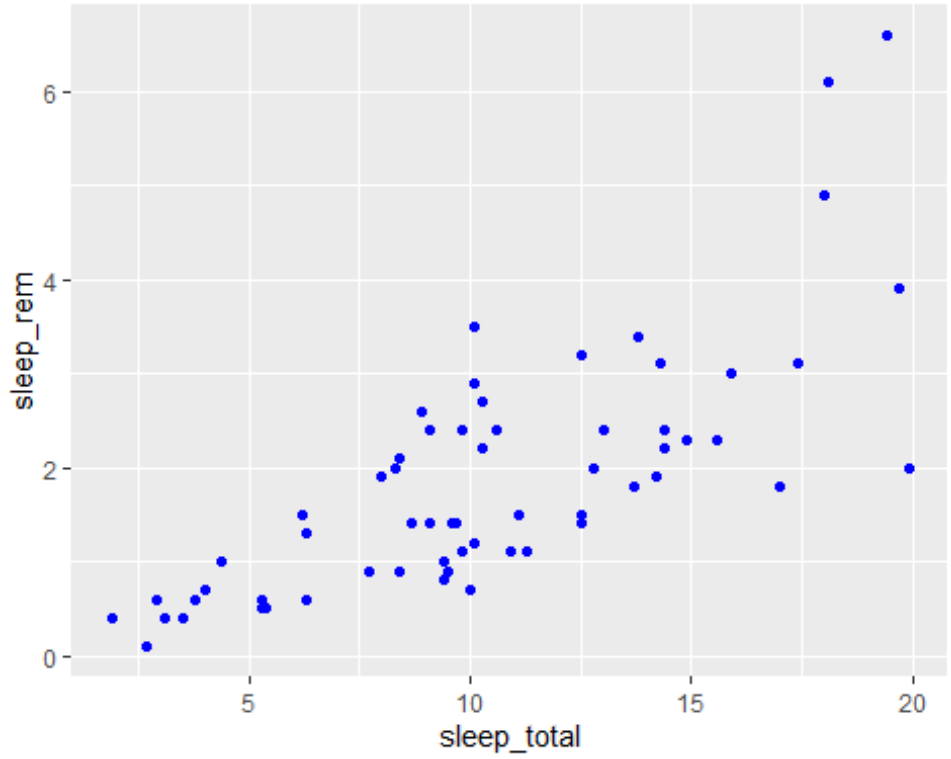
```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem, shape = vore))
```



Γράφημα 9

Εδώ πρέπει να σημειωθεί πως και οι θέσεις των αξόνων x και y είναι aesthetics. Το πλεονέκτημα με τη χρήση του ggplot2 είναι πως μετά την αντιστοίχιση του aesthetic το ggplot2 υλοποιεί όλες τις ρυθμίσεις αυτόματα. Φυσικά ο χρήστης μπορεί να θέσει τις ιδιότητες ενός aesthetic και χειροκίνητα (manually). Για παράδειγμα, ο παρακάτω κώδικας μετατρέπει το χρώμα όλων των σημείων σε μπλε. Συνήθως αυτός ο τρόπος δεν είναι τόσο χρήσιμος.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem), color = "blue")
```



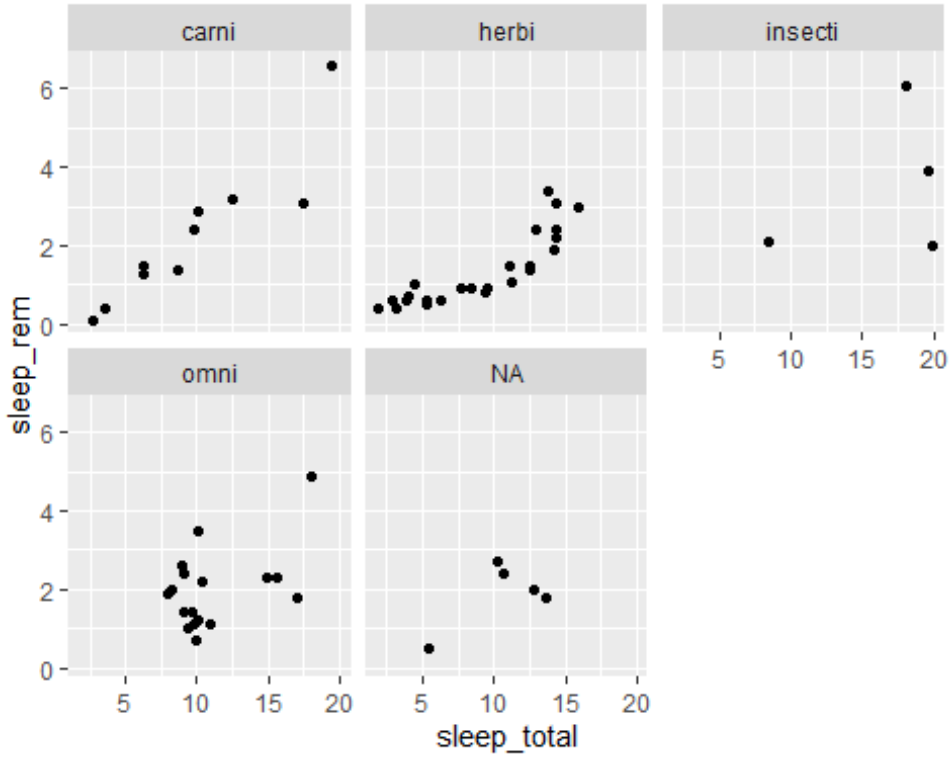
Γράφημα 10

Όπως παρατηρούμε σε αυτό το διάγραμμα το χρώμα δεν παίρνει τιμή από την τιμή κάποιας μεταβλητής αλλά αλλάζει συνολικά την εμφάνιση των σημείων καθώς η ρύθμιση έγινε εκτός της συνάρτησης aes().

4.1.2. Όψεις (facets)

Οι αισθητικές ιδιότητες είναι ένας τρόπος για να προσθέσουμε μεταβλητές στο διάγραμμα μας βελτιώνοντας το. Ένας άλλος τρόπος βελτίωσης, ιδιαίτερα χρήσιμος όταν έχουμε κατηγορηματικές μεταβλητές, είναι ο διαχωρισμός του διαγράμματος σε όψεις που είναι υπο-διαγράμματα των υποσυνόλων των δεδομένων με βάση την τιμή μίας κατηγορηματικής μεταβλητής. Για να υλοποιηθεί αυτή η τεχνική γίνεται χρήση της facet_wrap() όπως φαίνεται στο παρακάτω τμήμα κώδικα.

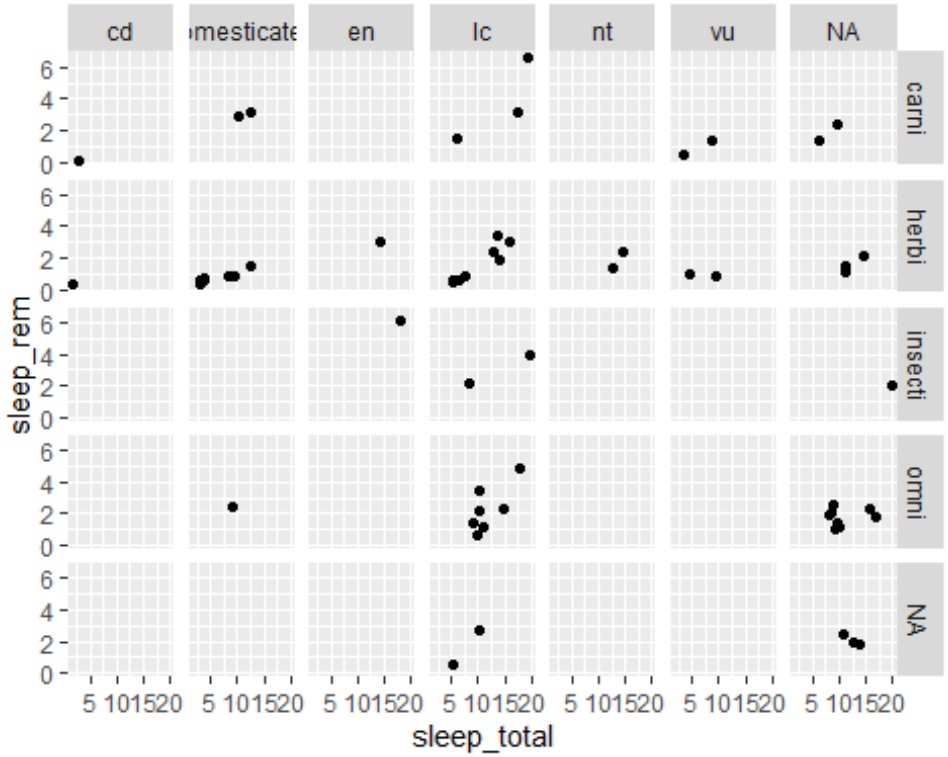
```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem)) +
  facet_wrap(~ vore, nrow = 2)
```



Γράφημα 11

Πρέπει να σημειωθεί πως η μεταβλητή διαχωρισμού θα πρέπει να είναι οπωσδήποτε κατηγορηματική αλλιώς θα εμφανιστεί ένα προειδοποιητικό μήνυμα. Εάν επιθυμούμε τη χρήση δύο μεταβλητών διαχωρισμού τότε χρησιμοποιούμε τη facet_grid(). Ακολουθεί κώδικας που δείχνει τη δημιουργία όψεων με συνδυασμό δύο μεταβλητών.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem)) +
  facet_grid(vore ~ conservation)
```



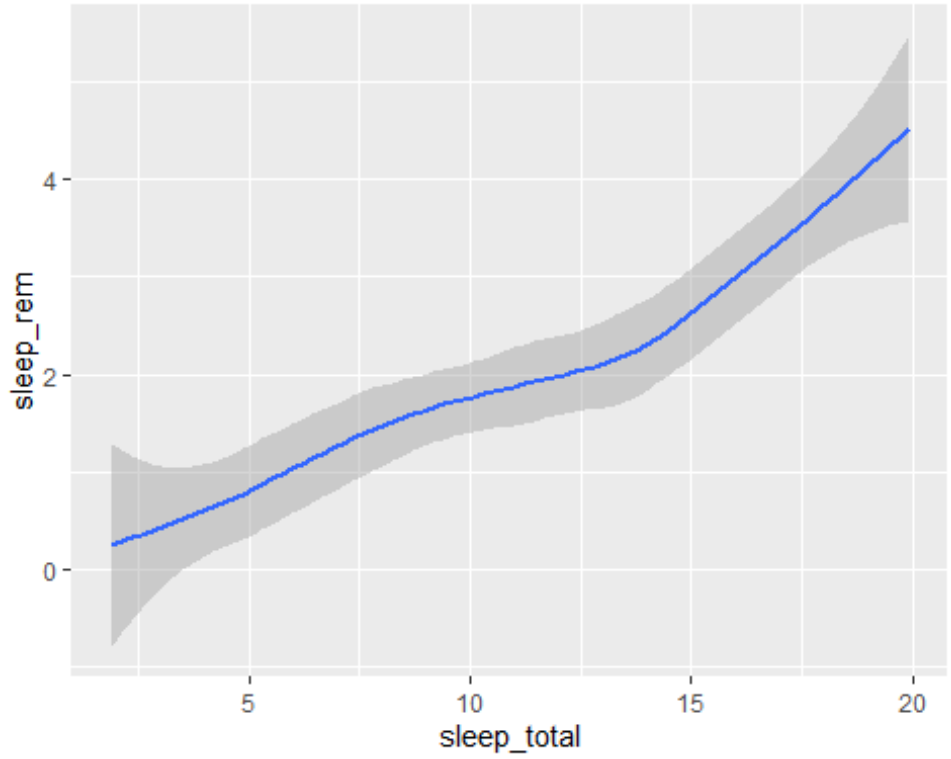
Γράφημα 12

Εάν θέλουμε να περιορίσουμε τη δημιουργία όψεων σε μια από τις δύο διαστάσεις τότε χρησιμοποιούμε στη θέση της μίας μεταβλητής το σύμβολο της τελείας.

4.1.3. Γεωμετρικά αντικείμενα

Ένα geom είναι το γεωμετρικό αντικείμενο που χρησιμοποιεί ένα γράφημα για να αναπαραστήσει τα δεδομένα. Συχνά περιγράφουμε τα γραφήματα με βάση το geom που χρησιμοποιήθηκε. Για να αλλάξουμε το geom σε ένα γράφημα απλώς αλλάζουμε τη συνάρτηση geom που προστίθεται στο ggplot(). Για παράδειγμα, γράφουμε τον παρακάτω κώδικα με χρήση του γεωμετρικού αντικειμένου geom_smooth και έχουμε το ακόλουθο γράφημα:

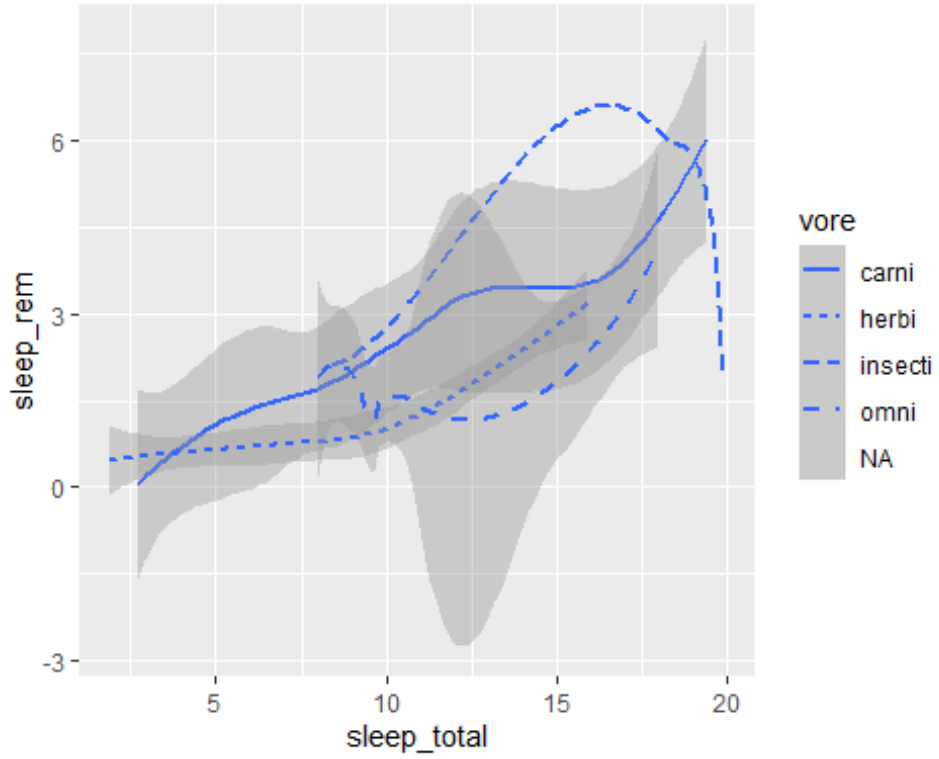
```
ggplot(data = msleep) +
  geom_smooth(mapping = aes(x = sleep_total, y = sleep_rem))
```



Γράφημα 13

Κάθε γεωμετρικό αντικείμενο geom απαιτεί ένα όρισμα για το mapping. Όμως δεν ταιριάζει το κάθε aesthetic με το κάθε geom. Για παράδειγμα, δεν μπορεί να χρησιμοποιηθεί η ιδιότητα του σχήματος με το αντικείμενο της γραμμής. Ας δούμε στον επόμενο κώδικα πως θέτουμε τον τύπο της γραμμής στο αντικείμενο geom_smooth().

```
ggplot(data = msleep) +
  geom_smooth(mapping = aes(x = sleep_total, y = sleep_rem, linetype
= vore))
```

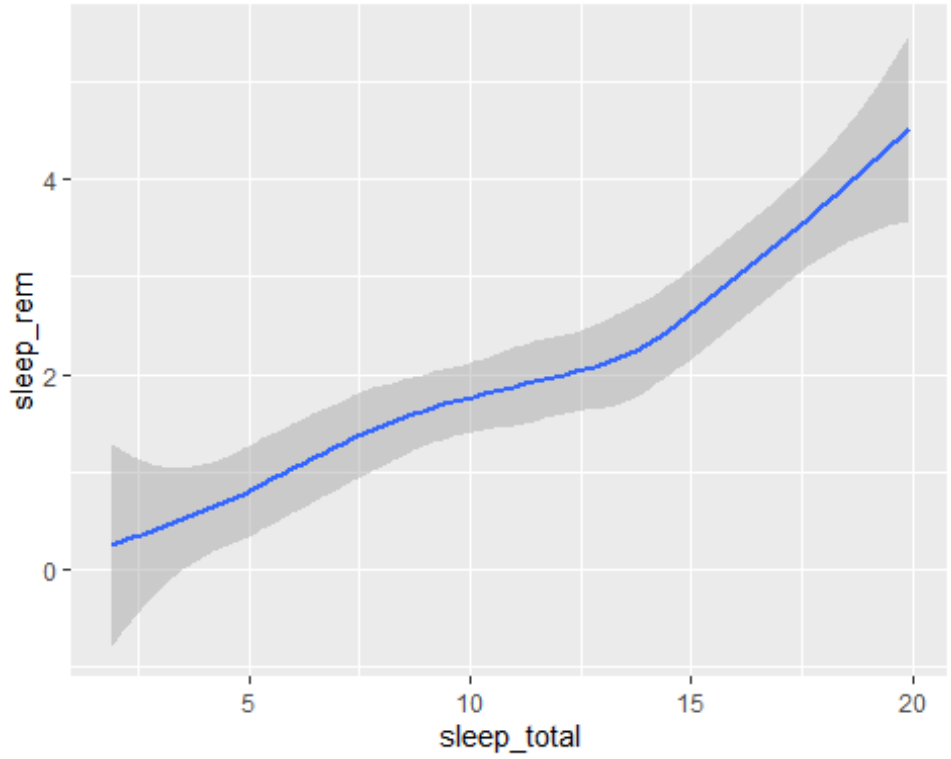


Γράφημα 14

Το διάγραμμα διαχωρίζει τα δεδομένα σε τρεις ξεχωριστές γραμμές ανάλογα με την τιμή της μεταβλητής vore. Το πακέτο ggplot2 διαθέτει πάνω από τριάντα γεωμετρικά αντικείμενα ενώ υπάρχουν και αρκετά πρόσθετα πακέτα που αυξάνουν σημαντικά αυτόν τον αριθμό. Πολλά από τα geom, όπως το geo_smooth(), χρησιμοποιούν ένα απλό γεωμετρικό αντικείμενο για να εμφανίσουν πολλές γραμμές δεδομένων. Για αυτά τα γεωμετρικά αντικείμενα μπορούμε να θέσουμε στο group aesthetic μια κατηγορηματική μεταβλητή έτσι ώστε να σχεδιάσουμε πολλαπλά αντικείμενα.

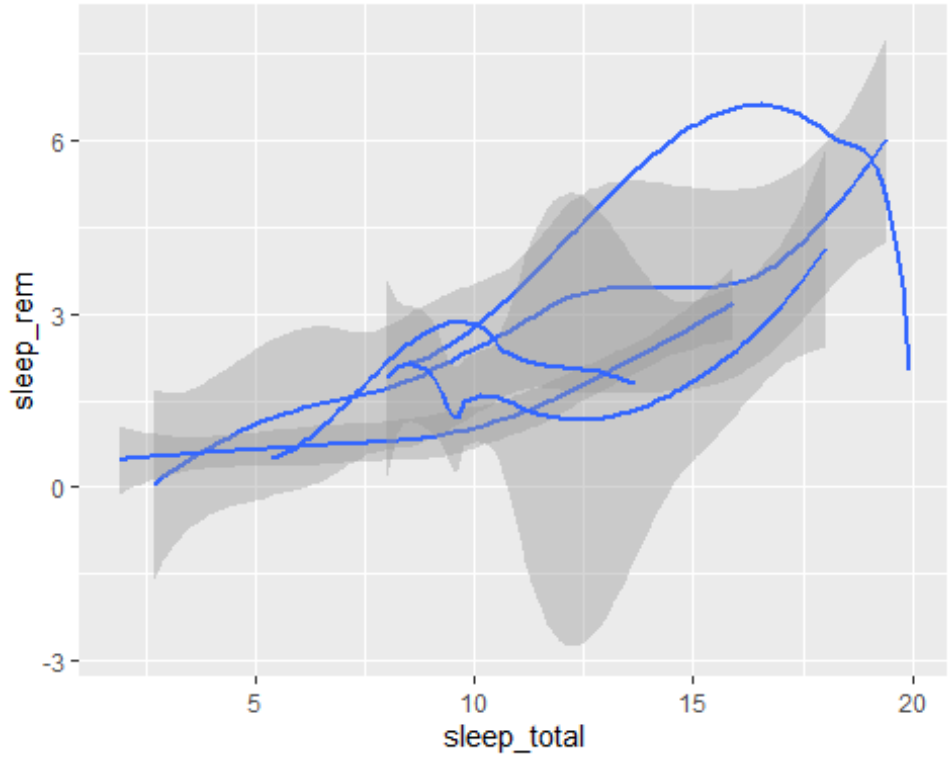
Στην πράξη, το ggplot2 αυτόματα ομαδοποιεί τα δεδομένα για εκείνα τα γεωμετρικά αντικείμενα για τα οποία αντιστοιχούμε μία αισθητική ιδιότητα σε μια διακριτή μεταβλητή. Είναι αρκετά βολικό να επαφιόμαστε σε αυτήν τη δυνατότητα επειδή το aesthetic της ομαδοποίησης από μόνο του δεν προσθέτει λεζάντα ή δεν διακρίνει τις δυνατότητες στα geom. Δείτε τα ακόλουθα τμήματα κώδικα:

```
ggplot(data = msleep) +
  geom_smooth(mapping = aes(x = sleep_total, y = sleep_rem, na.rm=TRUE))
```



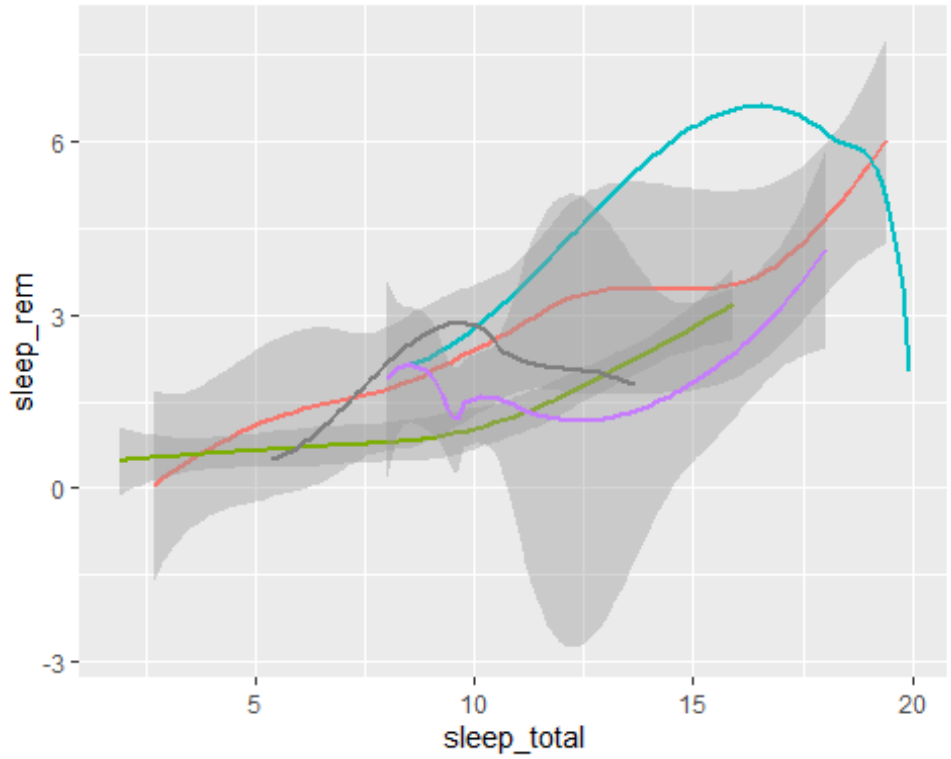
Γράφημα 15

```
ggplot(data = msleep) +
 geom_smooth(mapping = aes(x = sleep_total, y = sleep_rem, group = vore), na.rm=TRUE)
```



Γράφημα 16

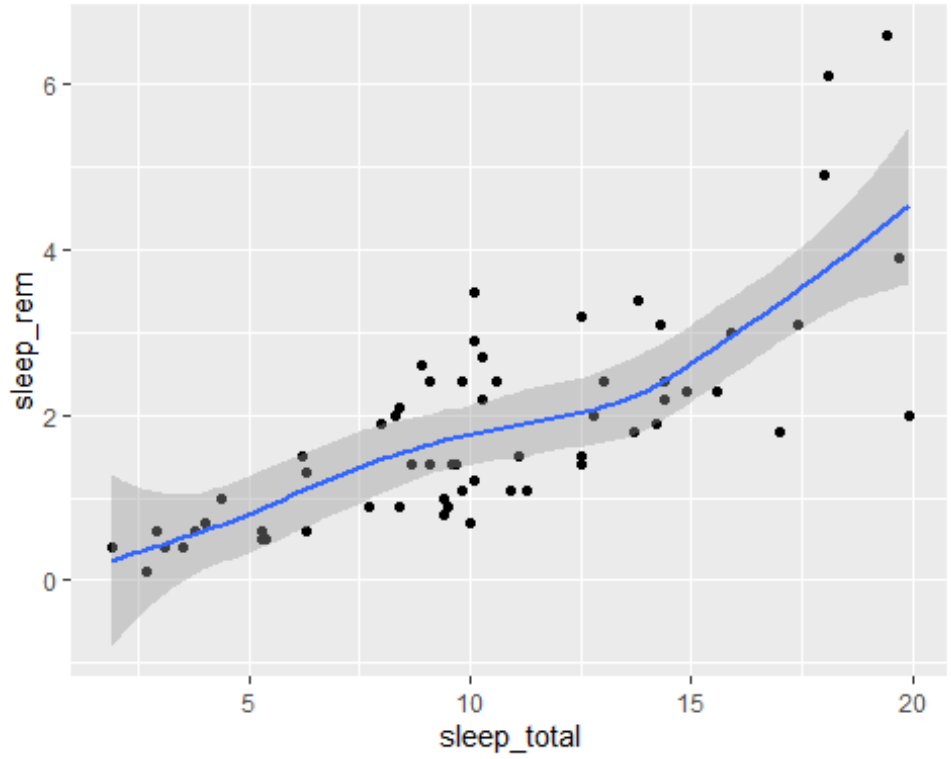
```
ggplot(data = msleep) + geom_smooth( mapping = aes(x = sleep_total,
y = sleep_rem, color = vore), show.legend = FALSE, na.rm=TRUE )
```



Γράφημα 17

Για να εμφανίσουμε πολλά γεωμετρικά αντικείμενα στο ίδιο γράφημα μπορούμε να προσθέσουμε πολλές συναρτήσεις geom στην ίδια κλήση της ggplot(), όπως στον παρακάτω κώδικα.

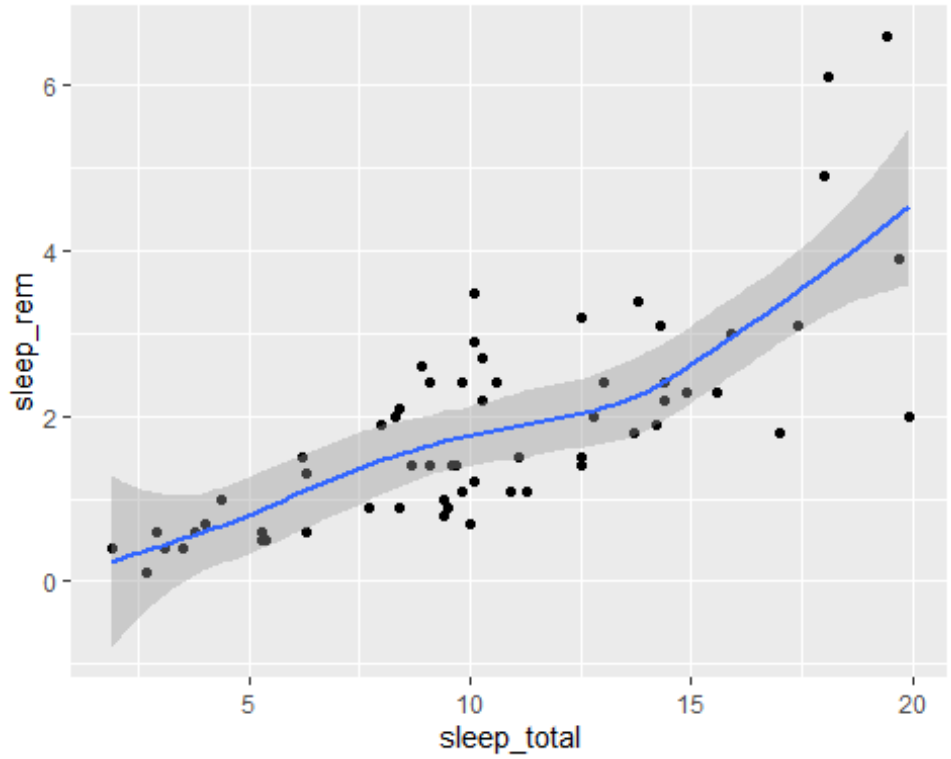
```
ggplot(data = msleep) + geom_point(mapping = aes(x = sleep_total, y
= sleep_rem)) + geom_smooth(mapping = aes(x = sleep_total, y = sleep
_rem))
```



Γράφημα 18

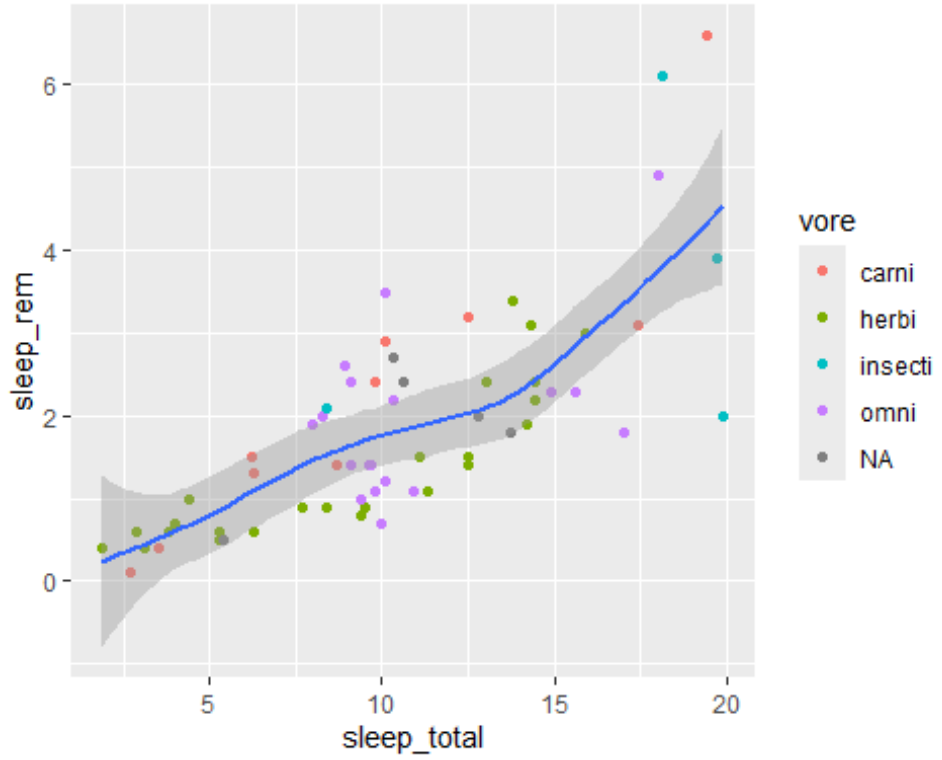
Επειδή ο προηγούμενος κώδικας έχει κάποια στοιχεία πλεονασμού μπορούμε να τον γράψουμε πιο συνοπτικά ως εξής:

```
ggplot(data = msleep, mapping = aes(x = sleep_total, y = sleep_rem))
+ geom_point() + geom_smooth()
```



Εάν τοποθετήσουμε αντιστοιχίσεις σε μια συνάρτηση geom, το πακέτο ggplot2 θα τις χειριστεί ως τοπικές αντιστοιχίσεις σε κάθε επίπεδο (layer). Αν χρησιμοποιήσουμε τις ίδιες αντιστοιχίσεις για να επεκτείνουμε ή υπερβούμε τις καθολικές αντιστοιχίσεις τότε αυτό ισχύει για αυτό το επίπεδο μόνο. Αυτό μας επιτρέπει τη δυνατότητα εμφάνισης διαφορετικών αισθητικών ιδιοτήτων σε διαφορετικά επίπεδα όπως στο γράφημα που δημιουργεί ο παρακάτω κώδικας.

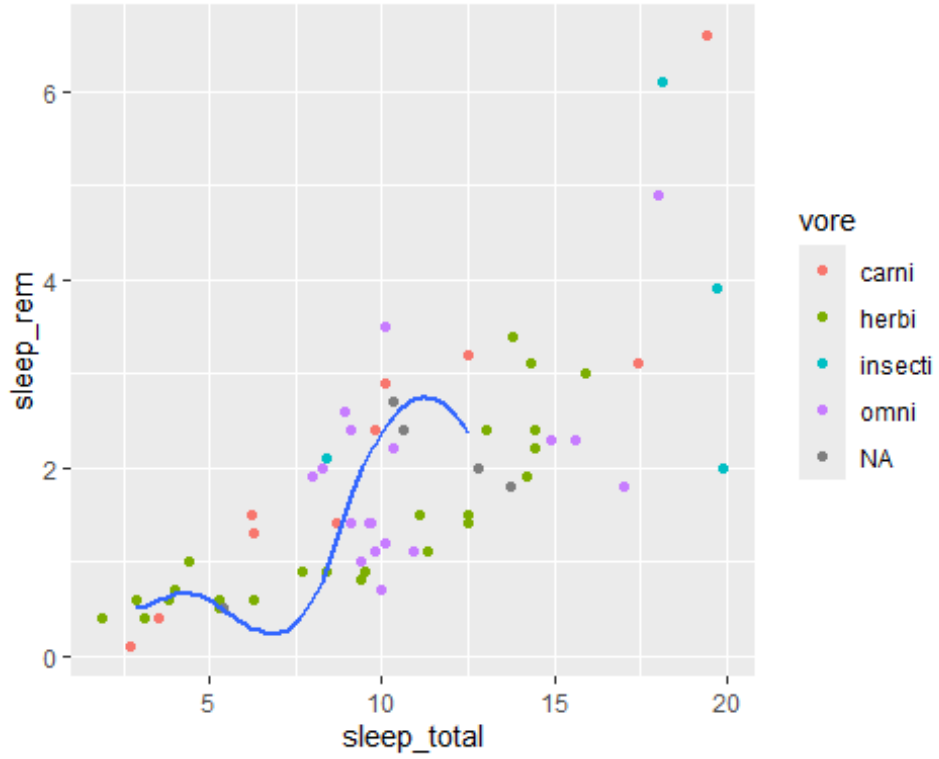
```
ggplot(data = msleep, mapping = aes(x = sleep_total, y = sleep_rem))
+ geom_point(mapping = aes(color = vore)) + geom_smooth()
```



Γράφημα 19

Μπορούμε να χρησιμοποιήσουμε την ίδια ιδέα για να προσδιορίσουμε διαφορετικά δεδομένα για κάθε επίπεδο όπως γίνεται στον παρακάτω κώδικα.

```
ggplot(data = msleep, mapping = aes(x = sleep_total, y = sleep_rem))
+ geom_point(mapping = aes(color = vore)) + geom_smooth(data = filte
r(msleep, conservation == "domesticated"), se = FALSE)
```

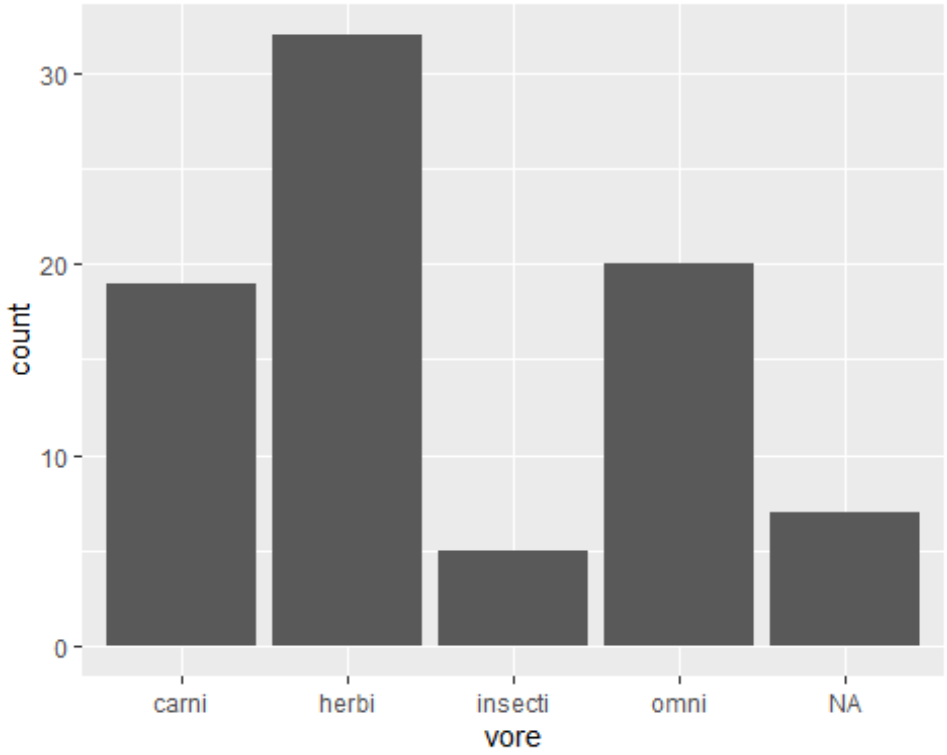


Γράφημα 20

4.2. Στατιστικές μετατροπές

Χρησιμοποιώντας ένα άλλο σύνολο δεδομένων και ένα άλλο γεωμετρικό αντικείμενο θα δημιουργήσουμε ένα ραβδόγραμμα (geom_bar) όπως φαίνεται μετά την εκτέλεση του ακόλουθου κώδικα.

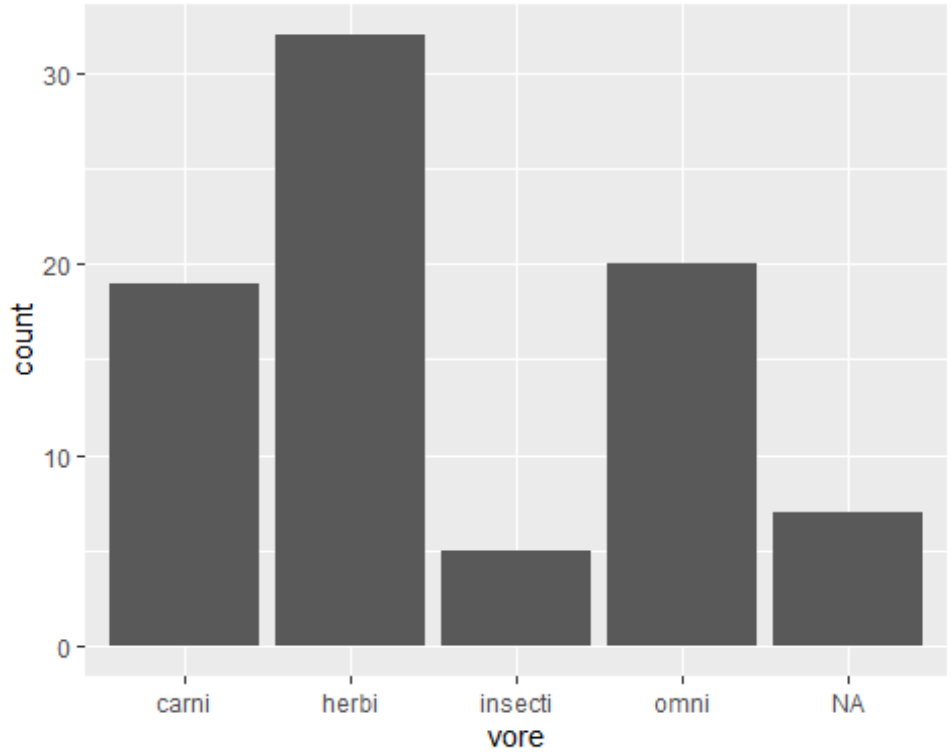
```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore))
```



Γράφημα 21

Όπως διαπιστώνουμε στον άξονα y εμφανίζεται ως τίτλος η λέξη count. Αυτό συμβαίνει γατί το γεωμετρικό αντικείμενο geom_bar χρησιμοποιεί το στατιστικό count που καταμετράει τη συχνότητα εμφάνισης των διακριτών τιμών της μεταβλητής vore. Το ίδιο συμβαίνει και για τα υπόλοιπα γεωμετρικά αντικείμενα, δηλαδή υπάρχουν αντίστοιχα στατιστικά μέτρα. Αν θέλουμε να βρούμε ποιο στατιστικό αντιστοιχεί σε ένα γεωμετρικό αντικείμενο τότε μπορούμε να χρησιμοποιήσουμε την αντίστοιχη εντολή βοήθειας ?geom_bar. Επιπλέον στο πακέτο gglpot2 το γεωμετρικό αντικείμενο και το στατιστικό είναι αλληλένδετα και μπορούμε να χρησιμοποιούμε τα geom και τα stat με τον ίδιο τρόπο όπως φαίνεται και στον επόμενο κώδικα.

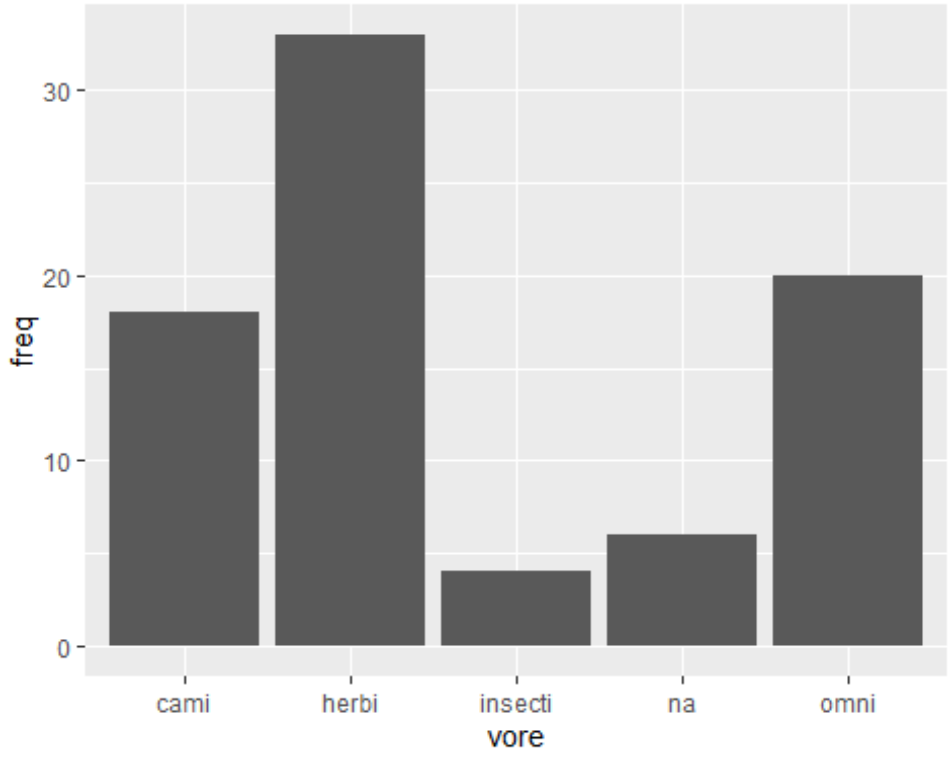
```
ggplot(data = msleep) + stat_count(mapping = aes(x = vore))
```



Γράφημα 22

Αυτό συμβαίνει γιατί σε κάθε geom υπάρχει ένα προκαθορισμένο συνδεδεμένο stat και κάθε stat έχει ένα προκαθορισμένο συνδεδεμένο geom. Ο παρακάτω κώδικας δείχνει πως αλλάζει το προκαθορισμένο stat με τη ρύθμιση του stat. Σε αυτό το παράδειγμα δεν μας δίνονται τα πρωτογενή δεδομένα αλλά μας δίνονται τα στοιχεία καταμέτρησης της κάθε διακριτής τιμής.

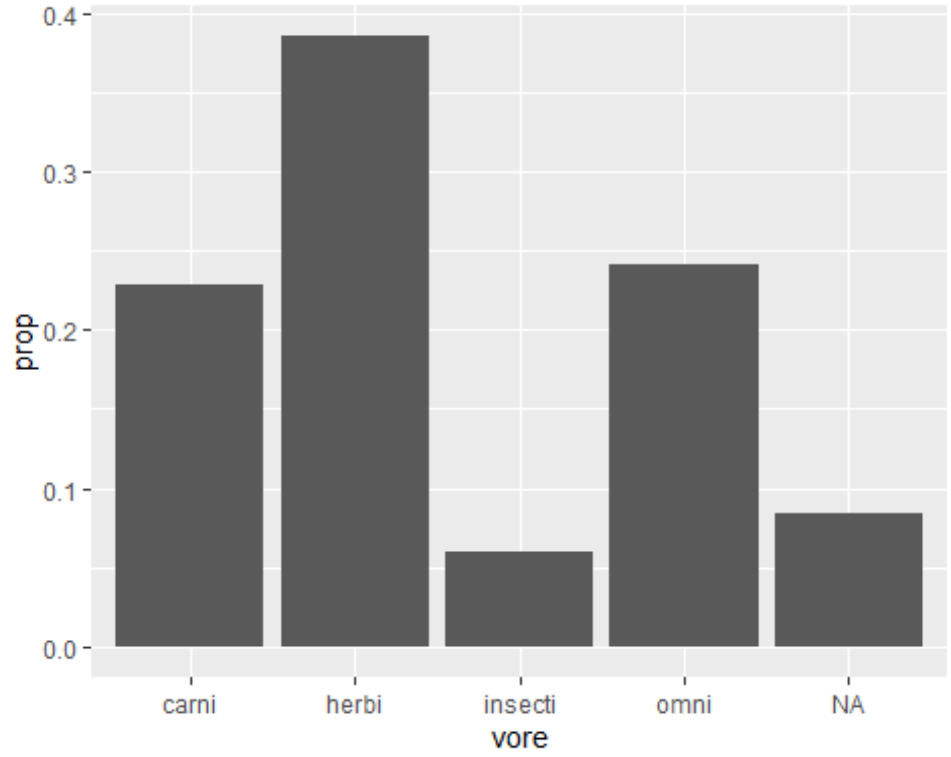
```
demo_bar <- tribble( ~vore, ~freq, "insecti", 4, "na", 6, "cami", 18
, "omni", 20, "herbi", 33)
ggplot(data = demo_bar) + geom_bar(mapping = aes(x = vore, y = freq)
, stat = "identity")
```



Γράφημα 23

Ο παρακάτω κώδικας αλλάζει την προκαθορισμένη αντιστοίχιση σε ένα άλλο stat το οποίο εμφανίζει τις ράβδους με αναλογική κλίμακα στον άξονα y.

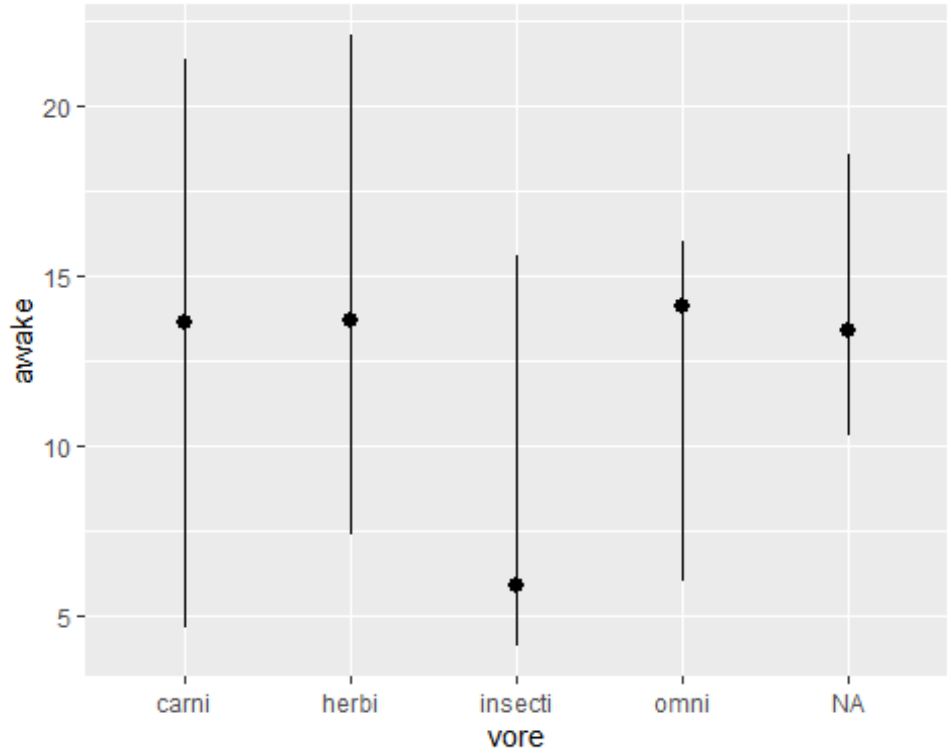
```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore, y = after_s
tat(prop), group = 1))
```



Γράφημα 24

Στο επόμενο τμήμα κώδικα γίνεται χρήση του stat summary.

```
ggplot(data = msleep) + stat_summary( mapping = aes(x = vore, y = aw
ake), fun.min = min, fun.max = max, fun = median)
```

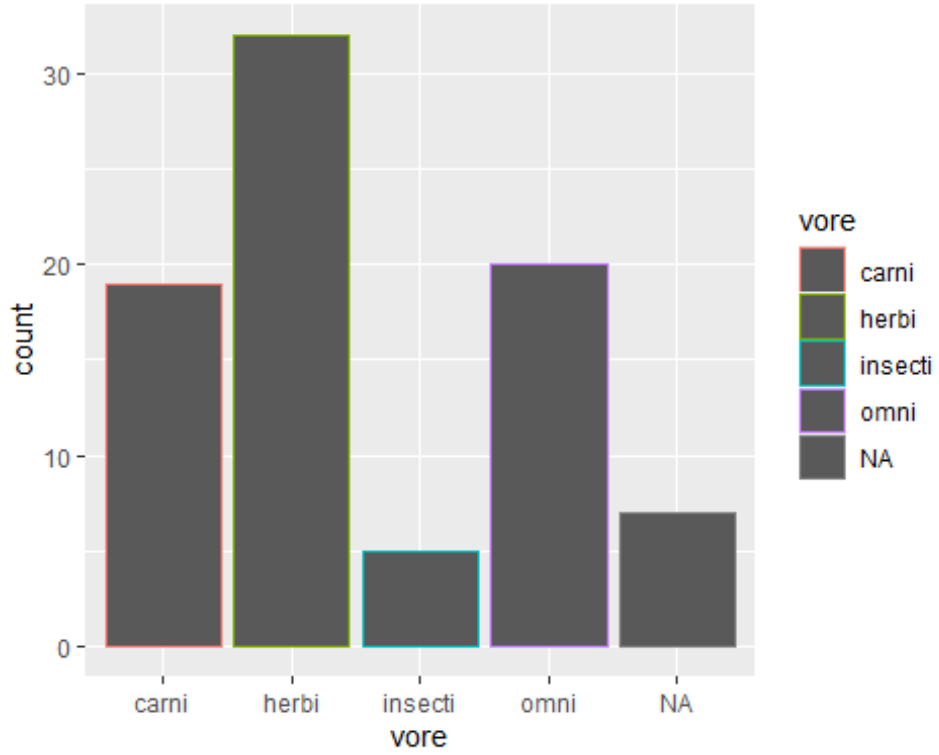


Γράφημα 25

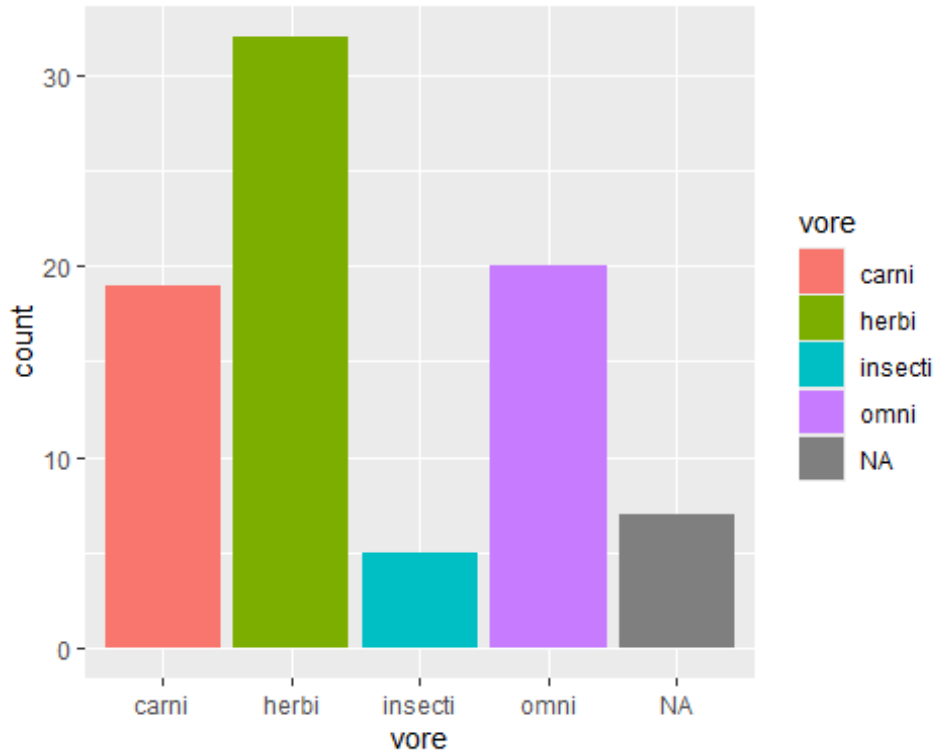
4.3. Ιδιότητες των γεωμετρικών αντικειμένων

Κάποια άλλα ενδιαφέροντα χαρακτηριστικά των ραβδογραμμάτων είναι οι ιδιότητες των color και του fill όπως φαίνονται στον παρακάτω κώδικα.

```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore, colour = vo
re))
```



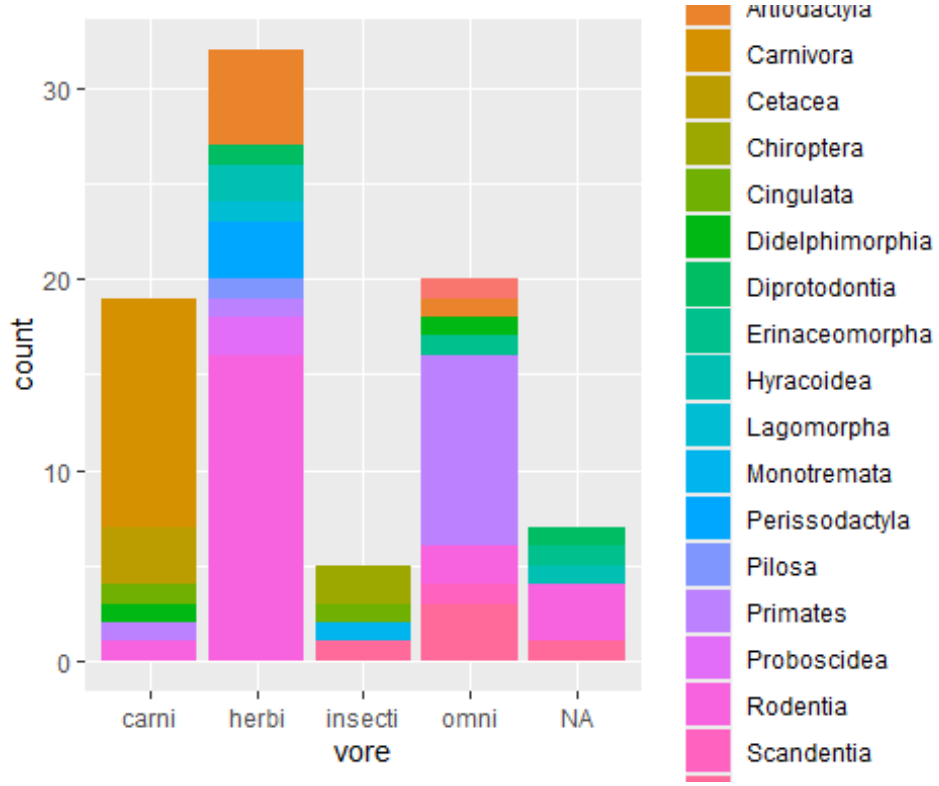

```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore, fill = vore
))
```



Γράφημα 26

Εάν χρησιμοποιηθεί μια άλλη μεταβλητή των δεδομένων για το fill τότε δημιουργείται ένα στοιβαγμένο ραβδόγραμμα όπως γίνεται με τον παρακάτω κώδικα.

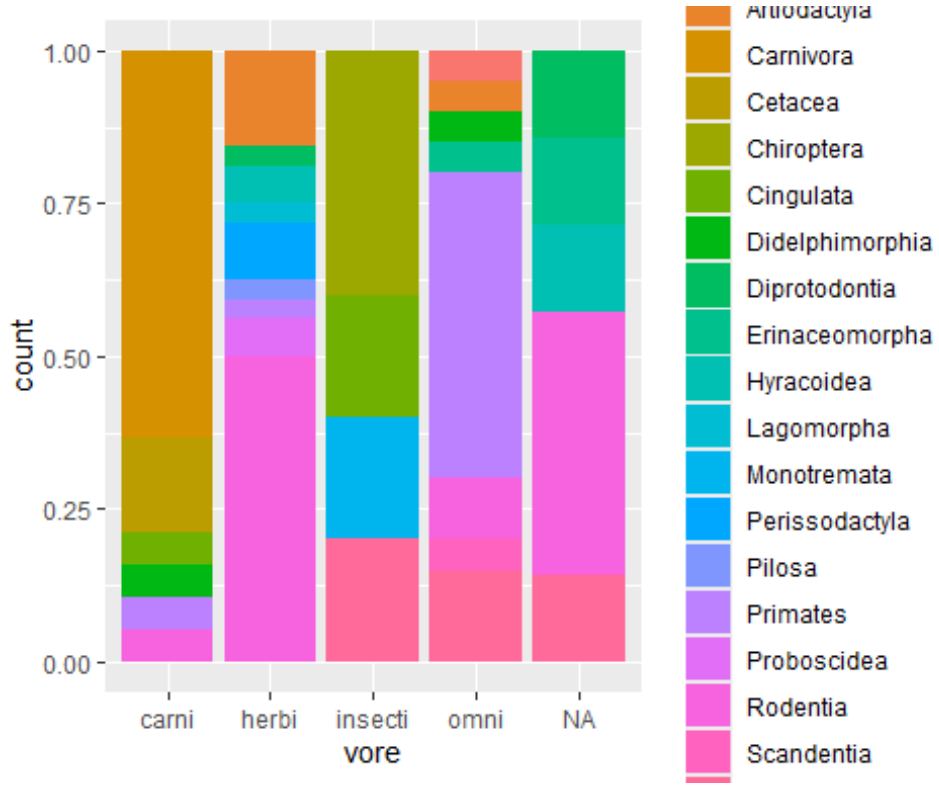
```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore, fill = orde
r))
```



Γράφημα 27

Αυτό επιτυγχάνεται αυτόματα με την προσαρμογή θέσης που καθορίζεται από το όρισμα της θέσης position. Το όρισμα μπορεί να πάρει επίσης τις τιμές identity, fill και dodge. Ο παρακάτω κώδικας χρησιμοποιεί την επιλογή fill που δίνει ένα ενδιαφέρον και χρήσιμο γράφημα.

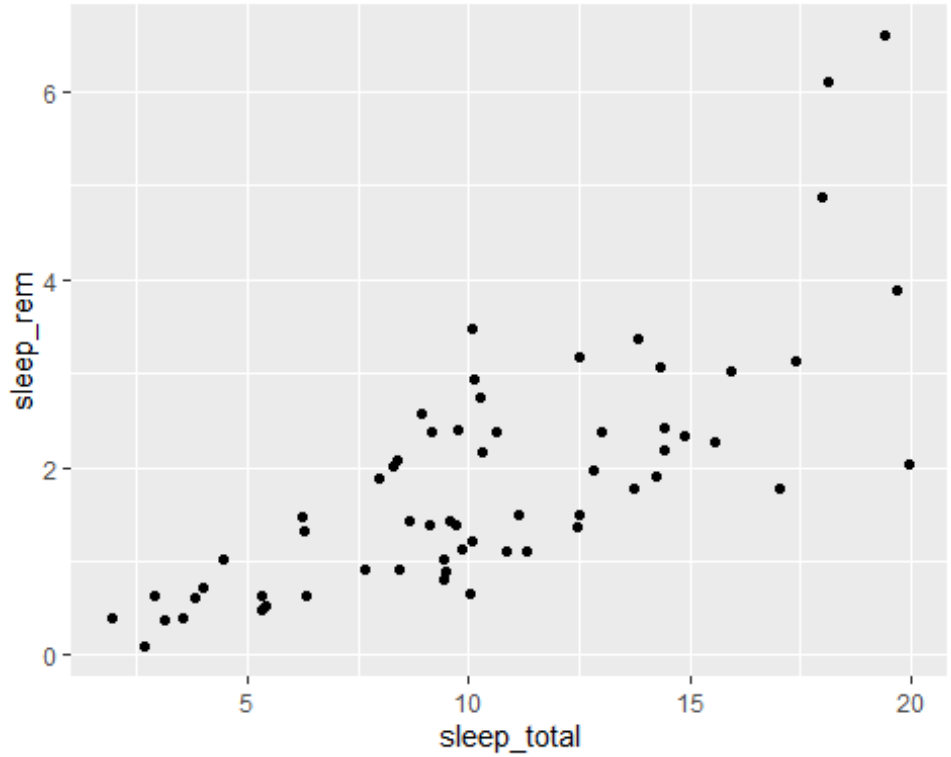
```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore, fill = orde
r), position = "fill")
```



Γράφημα 28

Όπως βλέπουμε αυτή η επιλογή δουλεύει όπως το στοίβαγμα αλλά δίνει σε κάθε στοίβα το ίδιο ύψος κάνοντας πιο εύκολη τη σύγκριση μέσα στην ομάδα. Ας δούμε όμως και μια προσαρμογή θέσης που λειτουργεί με το geom_point.

```
ggplot(data = msleep) + geom_point(mapping = aes(x = sleep_total, y
= sleep_rem), position = "jitter")
```

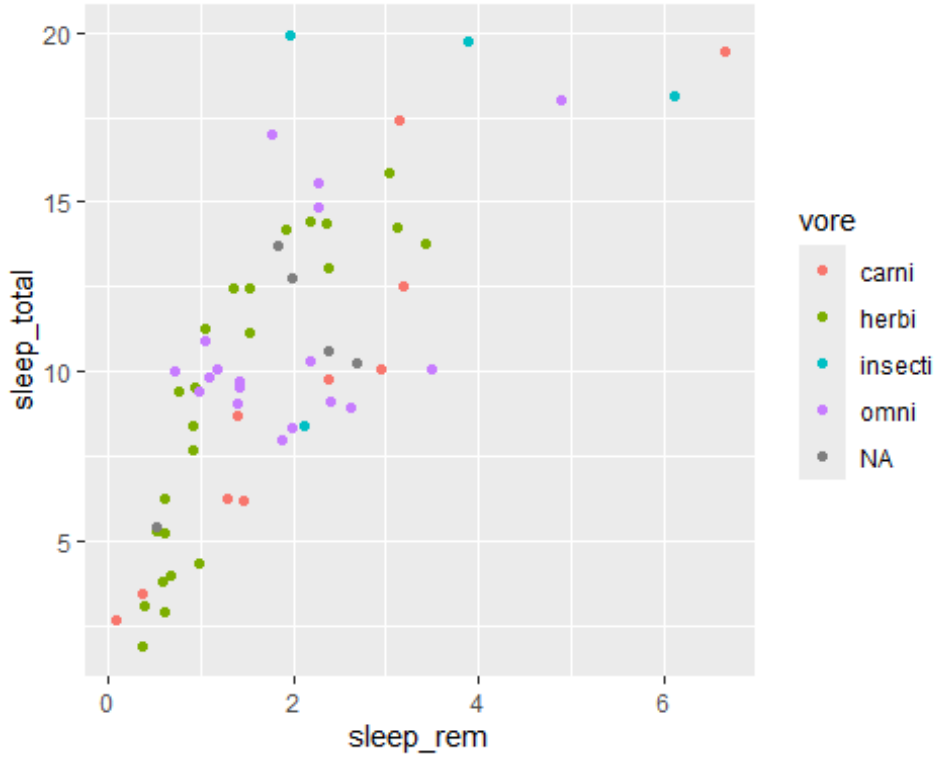


Γράφημα 29

Όπως βλέπουμε στο αποτέλεσμα αυτής της προσαρμογής παρακάμπτεται το πρόβλημα της επικάλυψης σημείων που έχουν τις ίδιες συντεταγμένες. Η επιλογή jitter προσθέτει τυχαίο θόρυβο σε κάθε σημείο και έτσι εξασφαλίζει πως δεν θα συμπίπτουν δύο σημεία στην ίδια θέση. Στα σημεία συγκέντρωσης πολλών σημείων θα εμφανίζεται μία μάζα σημείων που κολλούν μεταξύ τους.

Εναλλακτικά μπορούμε να κάνουμε χρήση του γεωμετρικού αντικειμένου geom_jitter το οποίο διασκορπίζει τα σημεία σε ένα σχετικό εύρος μέσα στο γράφημα όπως φαίνεται στο αποτέλεσμα του ακόλουθου κώδικα.

```
ggplot(data=msleep)+
  geom_jitter(aes(x=sleep_rem, y=sleep_total, colour=vore))
```

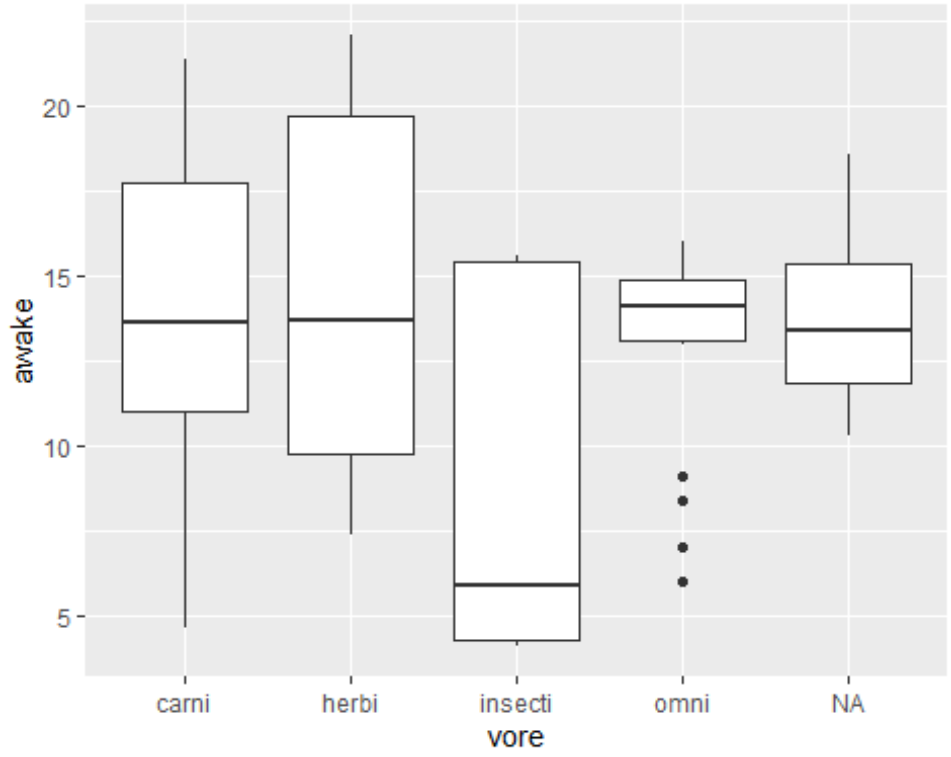


Γράφημα 30

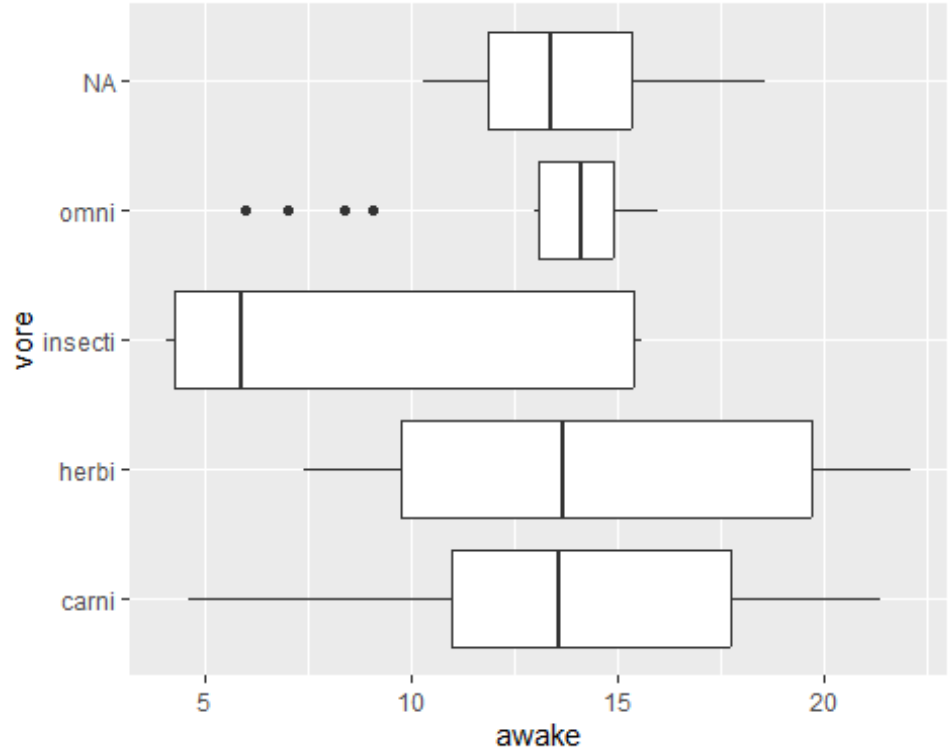
4.4. Συστήματα Συντεταγμένων

Στη διαχείριση των γραφημάτων ένα από τα πιο πολύπλοκα θέματα είναι το σύστημα συντεταγμένων που εφαρμόζεται κατά τη δημιουργία ενός γραφήματος. Το προκαθορισμένο σύστημα συντεταγμένων είναι προφανώς το καρτεσιανό. Υπάρχουν όμως και κάποιες εναλλακτικές επιλογές όπως η coord_flip, coord_quickmap και coord_polar που είναι αρκετά χρήσιμες. Η πιο απλή επιλογή είναι η coord_flip() που απλώς αντιμεταθέτει τους άξονες x και y όπως φαίνεται από το αποτέλεσμα του επόμενου κώδικα.

```
ggplot(data = msleep, mapping = aes(x = vore, y = awake)) + geom_box
plot()
```




```
ggplot(data = msleep, mapping = aes(x = vore, y = awake)) + geom_box
plot() + coord_flip()
```

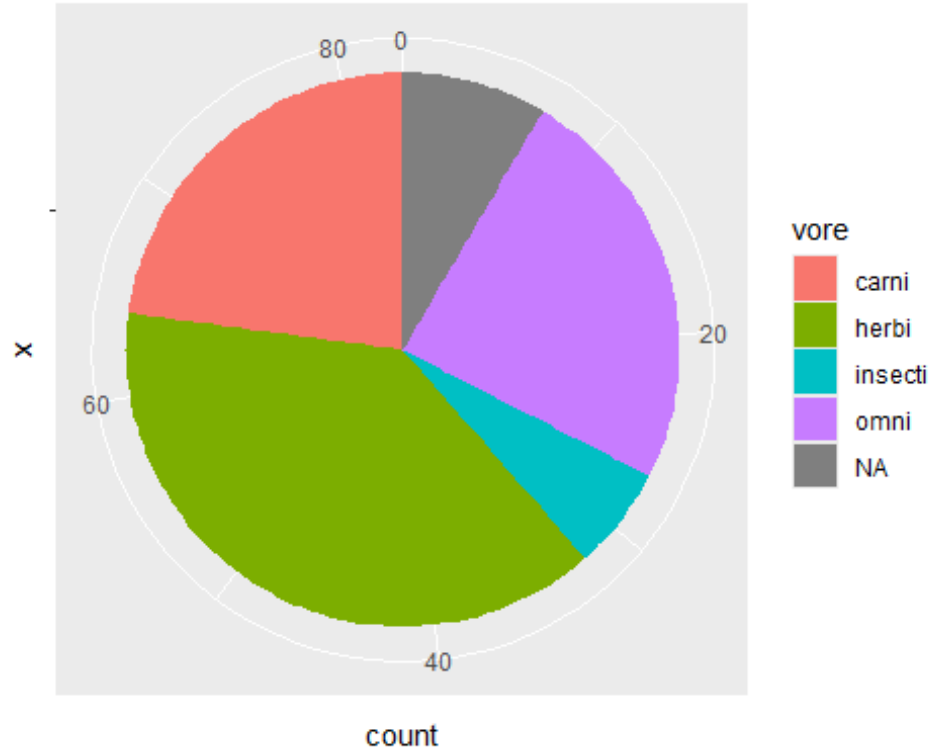


Γράφημα 31

Η πρώτη κλήση της συνάρτησης ggplot δημιουργεί μια σειρά από θηκογράμματα ανάλογα με την τιμή της μεταβλητής vore ενώ η δεύτερη κλήση της γίνεται σε συνδυασμό με την αλλαγή του συστήματος συντεταγμένων με την περιστροφή του καρτεσιανού κατά 90 μοίρες.

Το ακόλουθο παράδειγμα δείχνει πως με τη χρήση ενός στοιβαγμένου ραβδογράμματος και την αλλαγή του συστήματος συντεταγμένων σε πολικό σύστημα σχηματίζεται το πολύ δημοφιλές διάγραμμα πίτας (pie chart).

```
ggplot(data = msleep) +
     geom_bar(mapping = aes(x="", fill = vore))+coord_polar(theta =
"y")
```



Γράφημα 32

4.5. Διερευνητική ανάλυση δεδομένων με χρήση του ggplot2

Η διερευνητική ανάλυση αποτελεί ένα σημαντικό τμήμα της οποιαδήποτε ανάλυσης δεδομένων και συνήθως αποτελεί σημαντικό εργαλείο της πρώτης φάσης του καθαρισμού των δεδομένων καθώς προσφέρει οπτικοποίηση, μετατροπή και μοντελοποίηση. Συνήθη προβλήματα που επιλύονται με την εκτέλεση της διερευνητικής ανάλυση; είναι η μορφή της μεταβλητότητας των μεταβλητών και η ύπαρξης σχέσης μεταξύ δύο μεταβλητών.

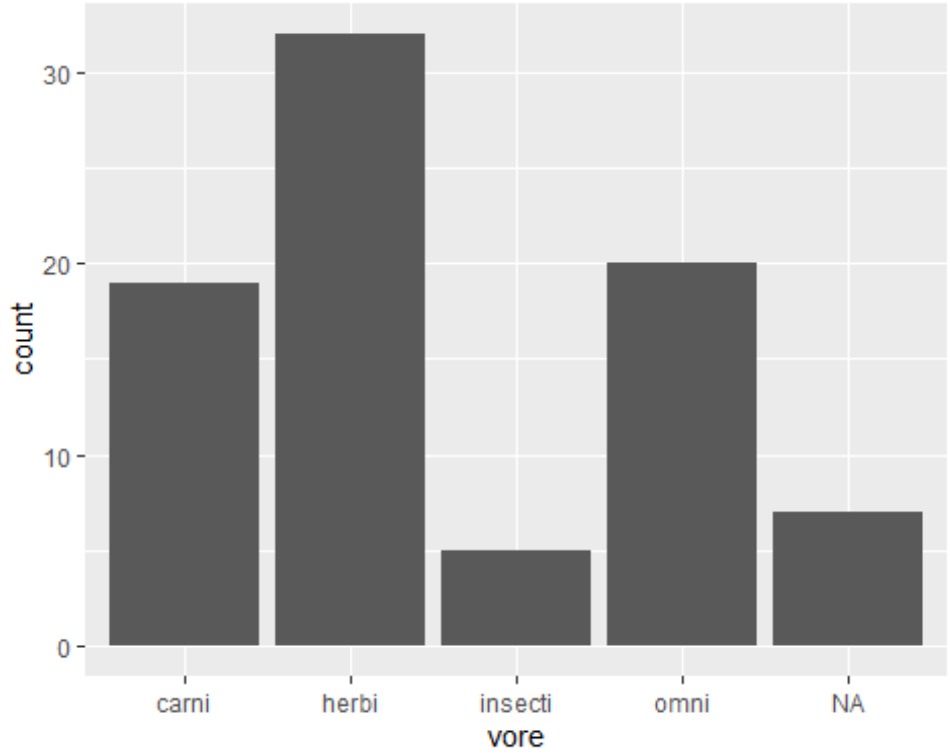
4.5.1. Μεταβλητότητα

Τι είναι όμως μεταβλητότητα; Είναι η τάση των τιμών μιας μεταβλητής να αλλάζουν από μία μέτρηση σε μία άλλη μέτρηση. Η μεταβλητότητα αφορά και τις συνεχείς και τις κατηγορηματικές μεταβλητές και ο εύκολος τρόπος κατανόησης της είναι η οπτικοποίηση της κατανομής των τιμών της μεταβλητής που μελετούμε.

4.5.2. Αναπαράσταση της κατανομής

Η αναπαράσταση της κατανομής μίας μεταβλητής και η επιλογή του κατάλληλου γεωμετρικού αντικειμένου εξαρτάται από το είδος της δηλαδή από το εάν η μεταβλητή είναι κατηγορηματική ή συνεχής. Στην R, οι κατηγορηματικές μεταβλητές είναι συνήθως τύπου factor ή διανύσματος χαρακτήρων και η αναπαράσταση τους γίνεται με ένα διάγραμμα ραβδογράμματος όπως αυτό που ακολουθεί και το έχουμε ήδη παρουσιάσει.

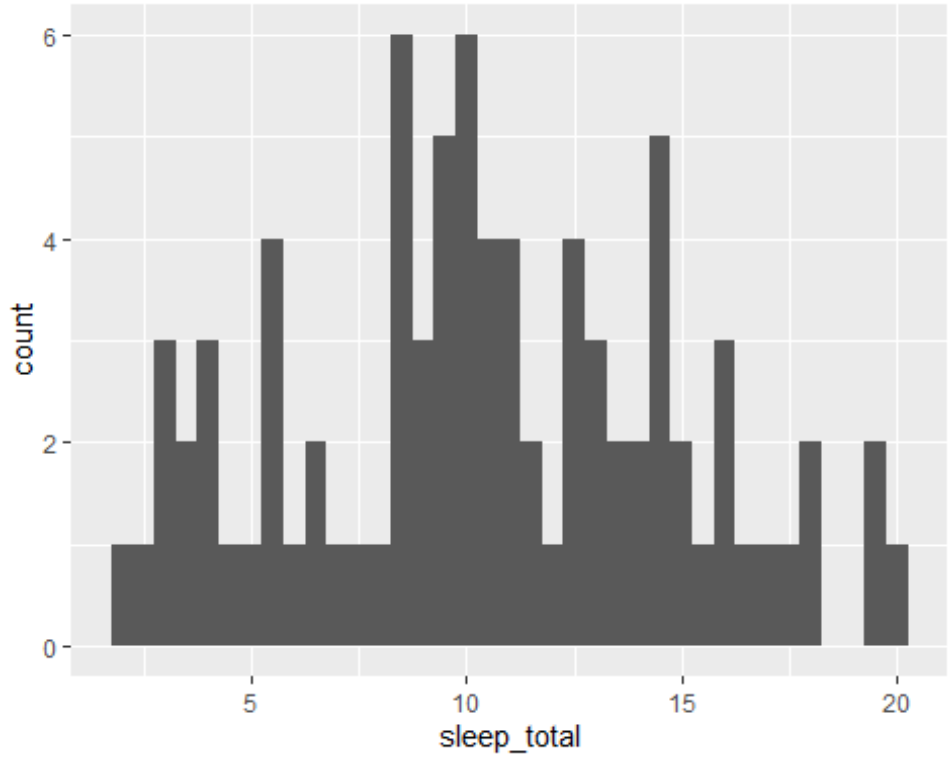
```
ggplot(data = msleep) + geom_bar(mapping = aes(x = vore))
```



Γράφημα 33

Σύμφωνα με τις βασικές γνώσεις μαθηματικών μία μεταβλητή είναι συνεχής αν μπορεί να πάρει μια οποιαδήποτε τιμή από ένα άπειρο σύνολο διατεταγμένων τιμών. Όταν η μεταβλητή είναι συνεχής τότε μπορεί να εξεταστεί ως προς την κατανομή της με τη χρήση ενός ιστογράμματος όπως αυτό που δημιουργείται από τον ακόλουθο κώδικα.

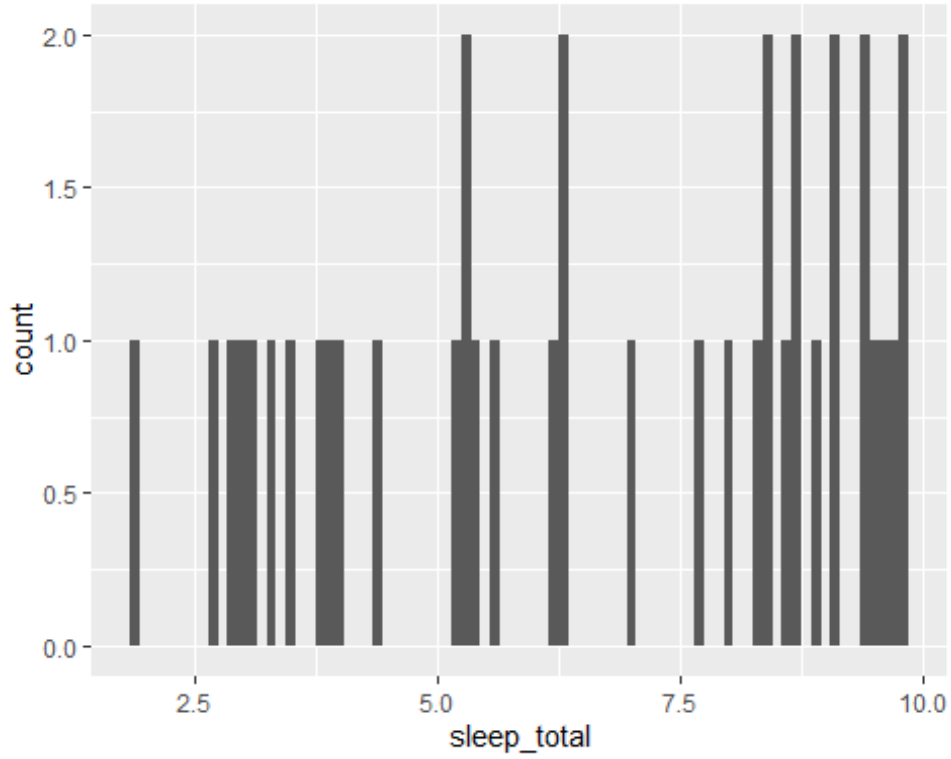
```
ggplot(data = msleep) + geom_histogram(mapping = aes(x = sleep_total
), binwidth = 0.5)
```



Γράφημα 34

Σημαντική παράμετρος του ιστογράμματος είναι ο ορισμός του μήκους των διαστημάτων στο ιστόγραμμα που γίνεται δίνοντας μία τιμή στο όρισμα binwidth. Συχνά χρειάζεται αρκετός πειραματισμός και δοκιμές με πολλές διαφορετικές τιμές του ορίσματος για να αποκαλυφθούν τα πιθανά διαφορετικά μοτίβα των τιμών της μεταβλητής και κατ' επέκταση η πιθανή κατανομή της. Για παράδειγμα μπορούμε να δούμε με περισσότερη ανάλυση το ιστόγραμμα για τα ζώα με λιγότερες από 10 ώρες ύπνου.

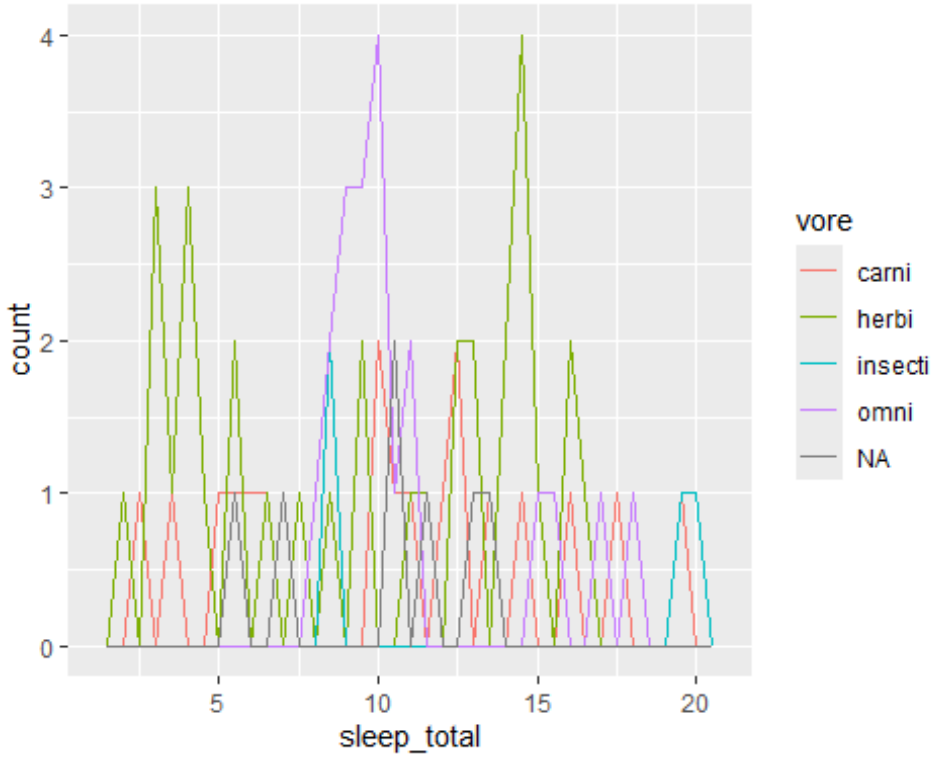
```
less <- msleep %>% filter(sleep_total < 10)
ggplot(data = less, mapping = aes(x = sleep_total)) + geom_histogram
(binwidth = 0.1)
```



Γράφημα 35

Εάν επιθυμούμε την ύπαρξη πολλαπλών ιστογραμμάτων στο ίδιο γράφημα τότε μπορούμε να χρησιμοποιήσουμε το geom_freqpoly() αντί του geom_histogram(), το οποίο κάνει τους ίδιους υπολογισμούς αλλά χρησιμοποιεί γραμμές.

```
ggplot(data = msleep, mapping = aes(x = sleep_total, colour = vore))
+ geom_freqpoly(binwidth = 0.5)
```

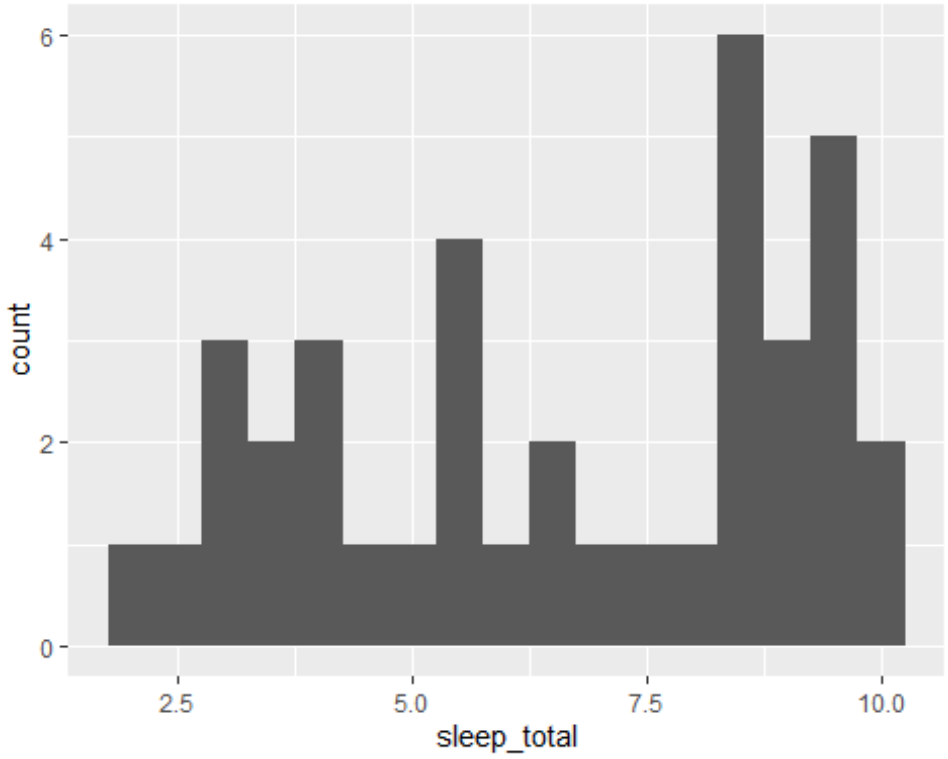


Γράφημα 36

4.5.3 Τυπικές- συνήθεις τιμές

Κάποια ζητήματα που προκύπτουν κατά τη διάρκεια της διερευνητικής εξέτασης είναι τα εξής: ποιες είναι οι πιο συχνά εμφανιζόμενες τιμές και ποιες οι αιτίες αυτής της εμφάνισης, ποιες είναι οι σπάνια εμφανιζόμενες τιμές και αν ταιριάζουν τα ευρήματα μας με τις τιμές που αναμέναμε, αν υπάρχουν ασυνήθιστα επαναλαμβανόμενα μοτίβα τιμών και ποια η πιθανή ερμηνεία τους; Ο επόμενος κώδικας μπορεί να βοηθήσει σε ανάλογες αναζητήσεις.

```
ggplot(data = less, mapping = aes(x = sleep_total)) + geom_histogram
(binwidth = 0.5)
```



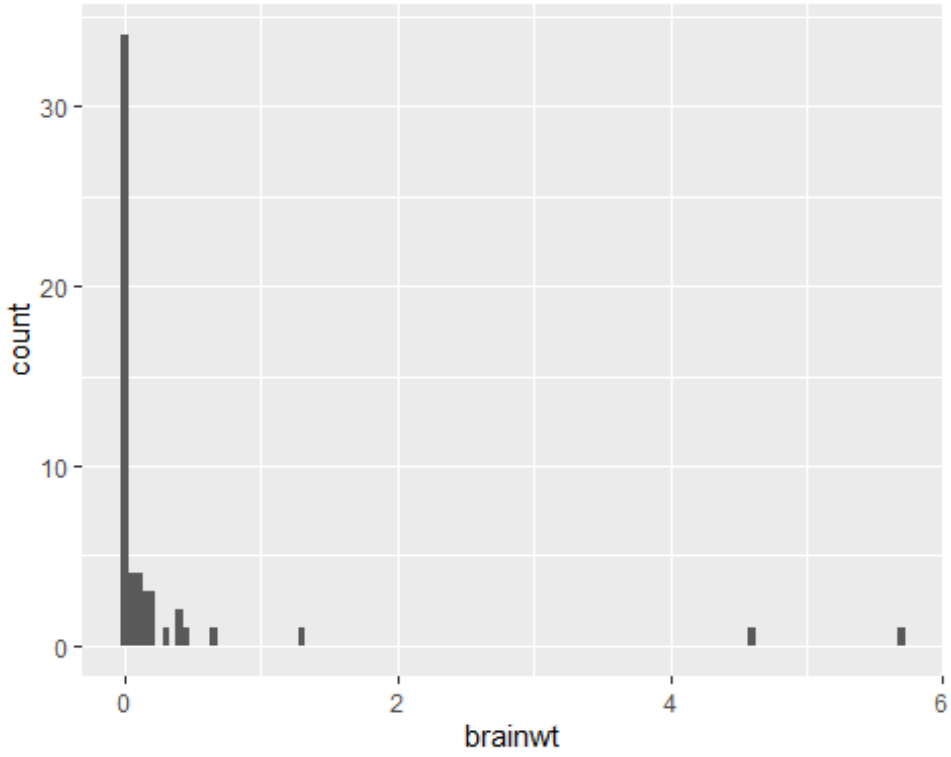
Γράφημα 37

Στο τελευταίο γράφημα διαπιστώνουμε πως υπάρχουν κάποιες συστάδες μετρήσεων. Γενικά, οι συστάδες όμοιων τιμών υποδεικνύουν την ύπαρξη ομαδοποιήσεων στα δεδομένα μας. Τότε προκύπτουν άλλα συνήθη ερωτήματα για τις συστάδες όπως τα εξής: Με ποιο τρόπο είναι οι παρατηρήσεις σε κάθε συστάδα όμοιες μεταξύ τους; Πως είναι οι παρατηρήσεις σε διαφορετικές συστάδες διαφορετικές μεταξύ τους; Γιατί η εμφάνιση των συστάδων μπορεί να μας παραπλανήσει; Ποια είναι η αιτία σχηματισμού των συστάδων;

4.5.4. Ασυνήθιστες τιμές

Ακραίες τιμές ονομάζονται οι ασυνήθιστες τιμές δηλαδή οι τιμές που δεν ταιριάζουν με το υπόλοιπο μοτίβο των τιμών μίας μεταβλητής. Όταν υπάρχουν πολλά δεδομένα, οι ακραίες τιμές δύσκολα εντοπίζονται στο ιστόγραμμα. Για παράδειγμα, ας δούμε το αποτέλεσμα του ακόλουθου κώδικα.

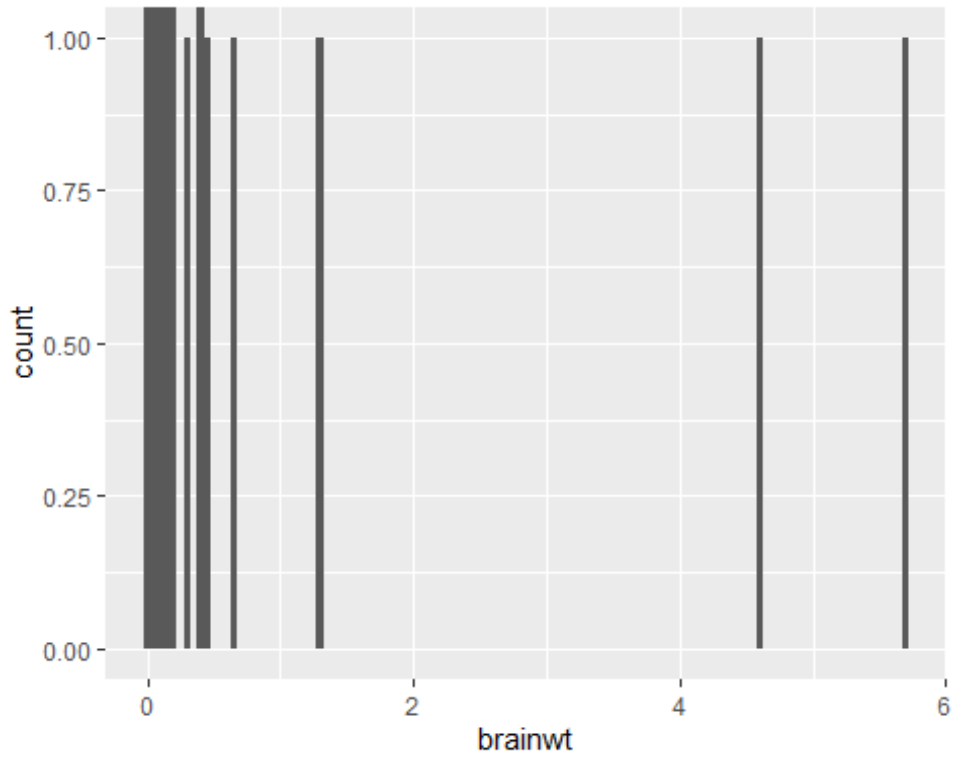
```
ggplot(msleep) + geom_histogram(mapping = aes(x = brainwt), binwidth
= 0.05)
```



Γράφημα 38

Η ένδειξη που παρουσιάζεται και υποδεικνύει τις ακραίες τιμές είναι τα ασυνήθιστα όρια του εύρους του άξονα x. Εάν επιθυμούμε να δούμε τις ασυνήθιστες τιμές τότε χρειάζεται να κάνουμε εστίαση στις μικρές τιμές του άξονα y με την κατάλληλη ρύθμιση του συστήματος coord_cartesian() ως εξής:

```
ggplot(msleep) + geom_histogram(mapping = aes(x = brainwt), binwidth
= 0.05) + coord_cartesian(ylim = c(0, 1))
```



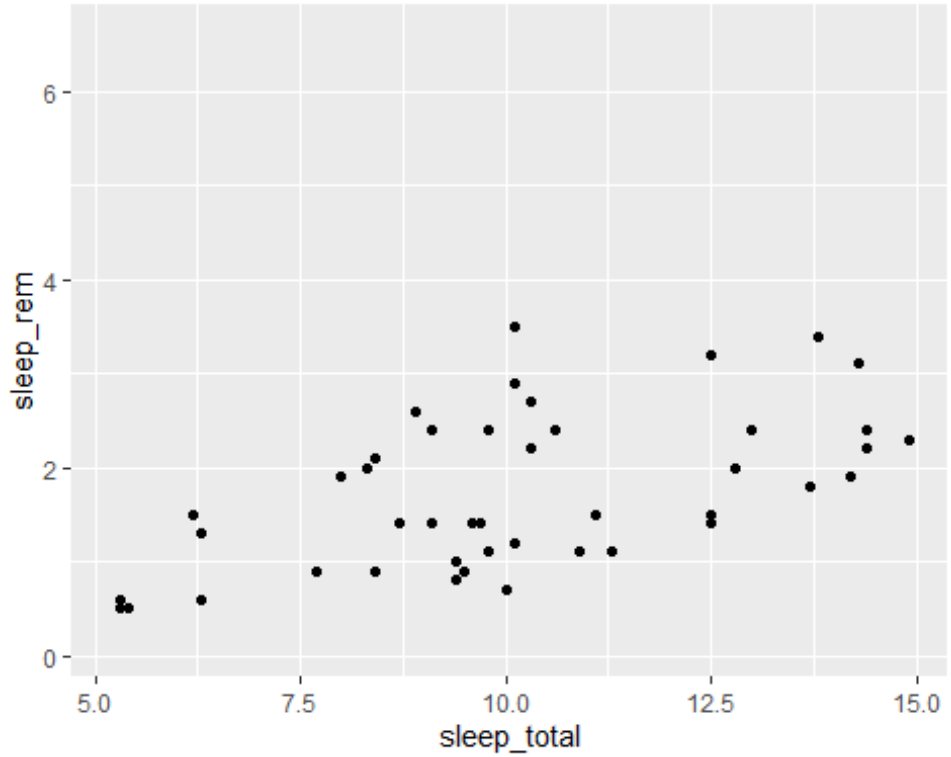
Γράφημα 39

Μια καλή τεχνική για να εντοπίσουμε τι συμβαίνει είναι η επανάληψη μιας ανάλυσης περιλαμβάνοντας τις ακραίες τιμές και μιας ανάλυσης χωρίς τη συμπερίληψη τους. Αν διαπιστώσουμε ότι δεν υπάρχει διαφορά στα αποτελέσματα της ανάλυσης τότε δεν έχει νόημα να τις χρησιμοποιούμε και μπορούμε να τις παραλείψουμε. Αν όμως έχουν σημαντική επίδραση στην ανάλυση τότε δεν μπορούμε να τις αγνοήσουμε χωρίς να διαθέτουμε επαρκή αιτιολόγηση.

4.5.5. Τιμές που λείπουν

Μια άλλη καλή πρακτική είναι η αντικατάσταση των ασυνήθιστων τιμών με τις τιμές που λείπουν (missing values). Ο ευκολότερος τρόπος είναι η χρήση της συνάρτησης mutate() για την αντικατάσταση της μεταβλητής με ένα τροποποιημένο αντίγραφο της όπως παρουσιάζεται στον ακόλουθο κώδικα.

```
msleep2 <- msleep %>% mutate(sleep_total = ifelse(sleep_total < 5 |
sleep_total > 15, NA, sleep_total))
ggplot(data = msleep2, mapping = aes(x = sleep_total, y = sleep_rem)
) + geom_point()
```



Γράφημα 40

Όπως και η R γενικά, και το πακέτο ggplot2 ενημερώνει πως οι τιμές που λείπουν δεν έχουν συμπεριληφθεί στο σχεδιαζόμενο γράφημα. Για παράδειγμα, ο παρακάτω κώδικας θα δώσει μια προειδοποίηση για τον αριθμό των τιμών που λείπουν. Φυσικά με τη ρύθμιση της παραμέτρου na.rm=TRUE η σχετική προειδοποίηση απαλείφεται.

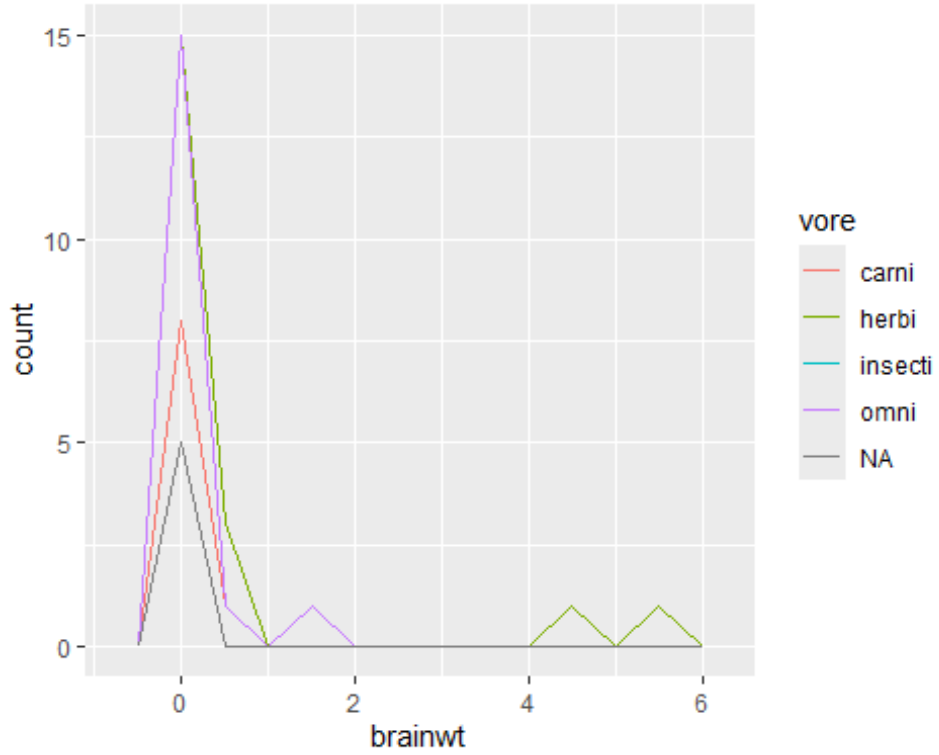
4.5.6. Συνδιακύμανση

Είναι η περιγραφή της συμπεριφοράς δύο μεταβλητών μεταξύ τους. Είναι η τάση των τιμών των δύο μεταβλητών να διαφέρουν συνδυαζόμενες μαζί με έναν σχετιζόμενο τρόπο. Ο πιο εύκολος τρόπος αναγνώρισης και εντοπισμού της συνδιακύμανσης είναι η απεικόνιση σε γράφημα της σχέσης μεταξύ δύο μεταβλητών. Η επιλογή αυτού του γραφήματος βασίζεται στον τύπο των εξεταζόμενων μεταβλητών, δηλαδή αν οι μεταβλητές είναι κατηγορηματικές ή συνεχείς. Στις προηγούμενες ενότητες ήδη έχουμε κάνει εκτεταμένη χρήση του διαγράμματος διασποράς που είναι το κατάλληλο εργαλείο για την ανίχνευση της σχέσης δύο συνεχόμενων μεταβλητών. Στη συνέχεια ακολουθεί η παρουσίαση των γραφημάτων για τους υπόλοιπους συνδυασμούς: α) μια κατηγορηματική με μια συνεχή μεταβλητή και β) δύο κατηγορηματικές μεταβλητές.

4.6. Σχέση μιας κατηγορηματικής με μια συνεχή μεταβλητή.

Αυτή η περίπτωση είναι αρκετά συνήθως και το geom_freqpoly() μας βοηθάει αλλά δεν μπορεί να αναδείξει γραφικά τη συνδιακύμανση χωρίς την τροποποίηση των χαρακτηριστικών του ειδικά όταν υπάρχουν μικρού μεγέθους ομάδες δεδομένων. Τότε δεν μπορούμε να δούμε τις διαφορές τους στο σχήμα.

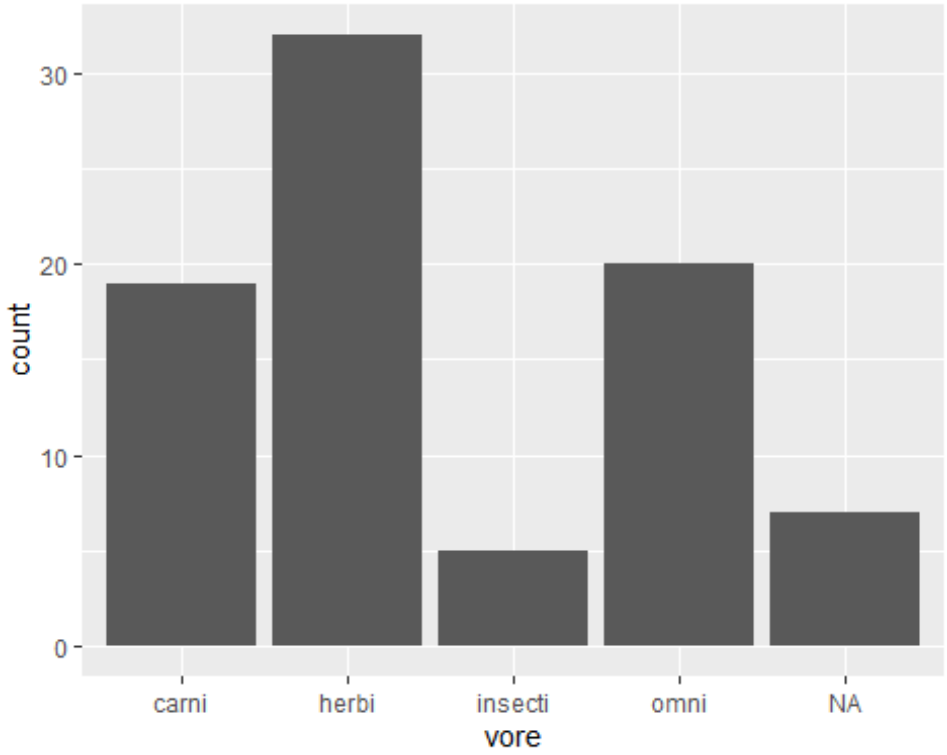
```
ggplot(data = msleep, mapping = aes(x = brainwt)) + geom_freqpoly(ma
pping = aes(colour = vore), binwidth = 0.5)
```



Γράφημα 41

Το διάγραμμα δεν βοηθά γιατί οι μετρήσεις διαφέρουν πάρα πολύ ανά κατηγορία όπως φαίνεται από το γράφημα που προκύπτει από τον προηγούμενο κώδικα. Είναι ορατό στο ακόλουθο διάγραμμα που έχει ξαναπαρουσιαστεί πως υπάρχει ανισότητα στο μέγεθος των εμφανίσεων στις διάφορες τιμές.

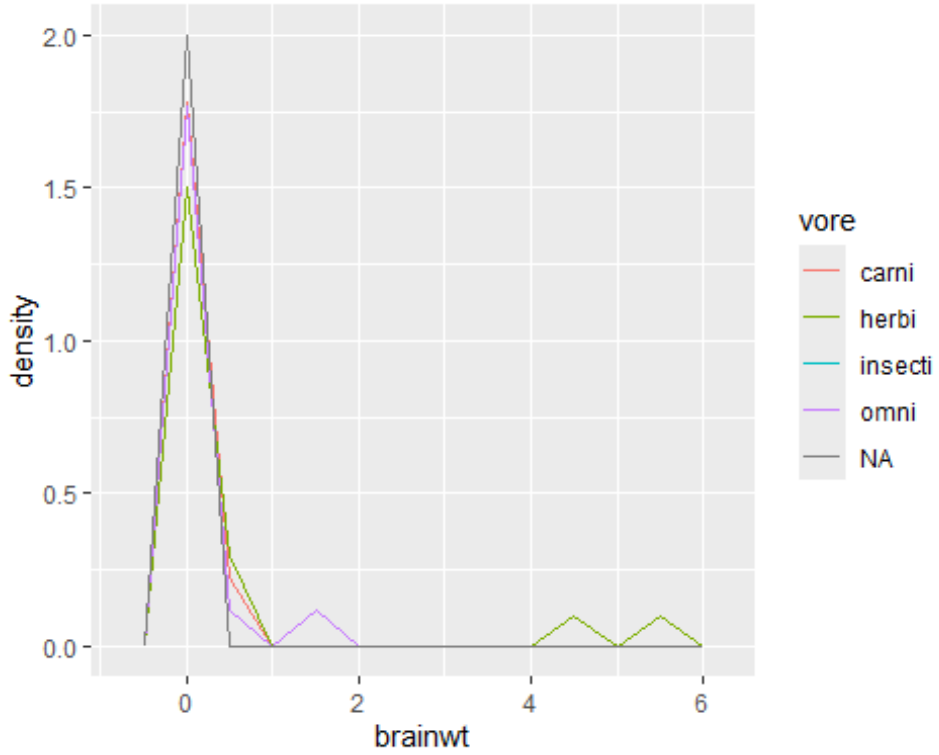
```
ggplot(msleep) + geom_bar(mapping = aes(x = vore))
```



Γράφημα 42

Ένας τρόπος για να γίνει η σύγκριση πιο ευκρινής είναι να αντικατασταθεί στον άξονα y το count από το density. Με αυτή την τροποποίηση γίνεται δυνατή η σύγκριση των διαφορετικών ομάδων που αντιστοιχούν στις τιμές της κατηγορηματικής μεταβλητής.

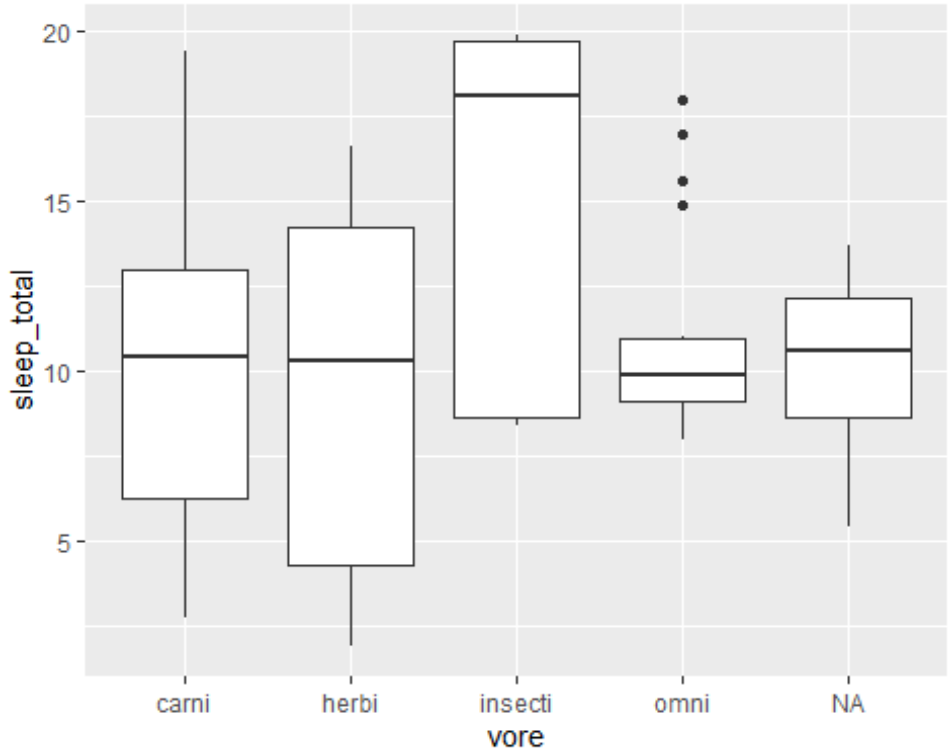
```
ggplot(data = msleep, mapping = aes(x = brainwt, y = ..density..)) +
geom_freqpoly(mapping = aes(colour = vore), binwidth = 0.5)
```



Γράφημα 43

Υπάρχει και μία άλλη εναλλακτική επιλογή για την εμφάνιση της κατανομής μιας συνεχούς μεταβλητής όπως διασπάται από τις τιμές μίας κατηγορηματική μεταβλητή. Αυτή είναι το boxplot το οποίο είναι μια γραφική σύνοψη της κατανομής. Ο παρακάτω κώδικας δημιουργεί το αναμενόμενο αποτέλεσμα όπως φαίνεται και στο διάγραμμα που ακολουθεί.

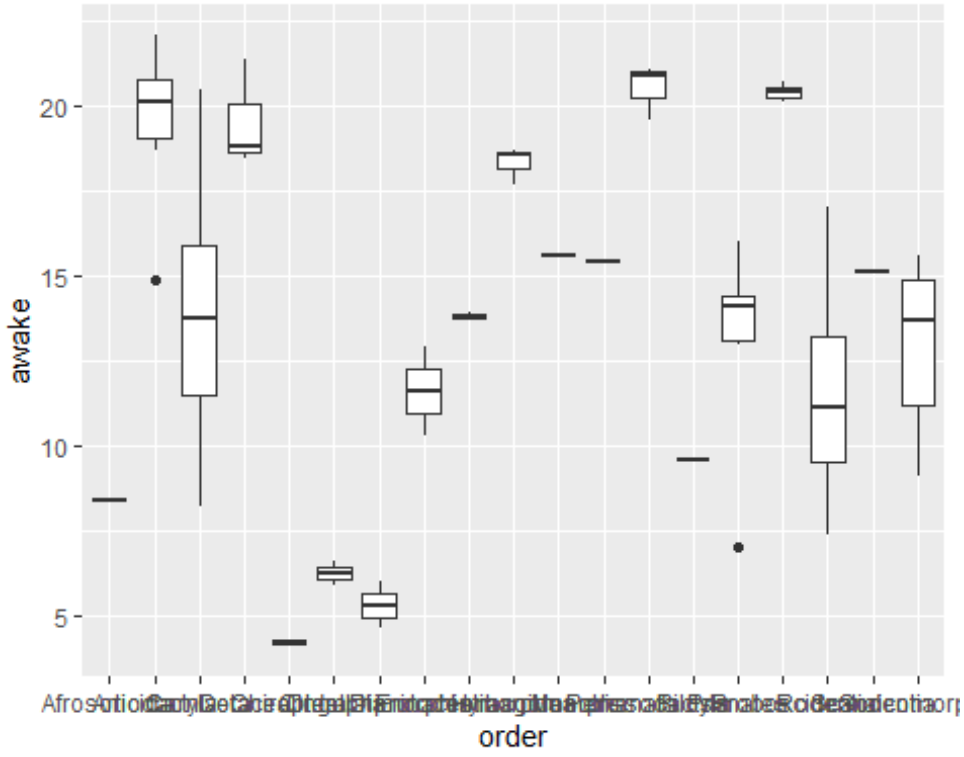
```
ggplot(data = msleep, mapping = aes(x = vore, y = sleep_total)) + ge
om_boxplot()
```



Γράφημα 44

Πολλές φορές επιδιώκουμε την αναδιάταξη της εμφάνισης του boxplot ή άλλων γεωμετρικών αντικειμένων σύμφωνα με κάποιο στατιστικό μέτρο. Για παράδειγμα, ο παρακάτω κώδικας δημιουργεί μια σειρά από boxplots χωρίς κάποια συγκεκριμένη διάταξη.

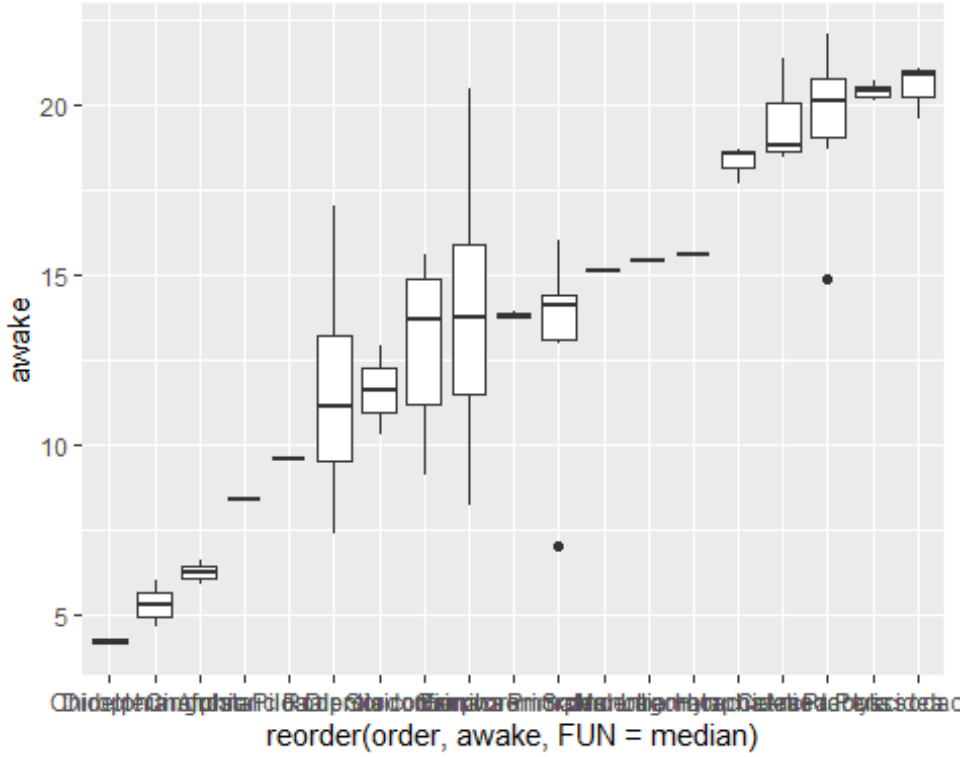
```
ggplot(data = msleep, mapping = aes(x = order, y = awake)) + geom_bo
xplot()
```



Γράφημα 45

Όμως εάν επιθυμούμε την αναδιάταξη του διαγράμματος σύμφωνα με την τιμή της διαμέσου τότε ο κώδικας ξαναγράφεται ως εξής:

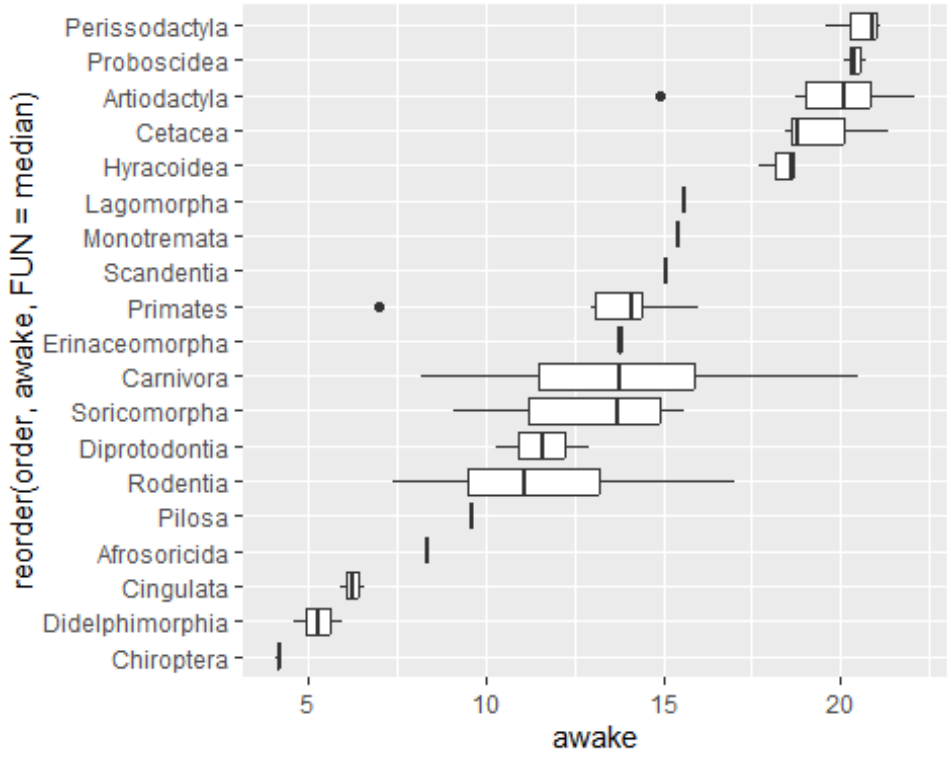
```
ggplot(data = msleep) + geom_boxplot(mapping = aes(x = reorder(order
, awake, FUN = median), y = awake))
```



Γράφημα 46

Ενώ με χρήση του συστήματος συντεταγμένων coord_flip() έχουμε το ακόλουθο πιο εμφανίσιμο διάγραμμα.

```
ggplot(data = msleep) + geom_boxplot(mapping = aes(x = reorder(order
, awake, FUN = median), y = awake)) + coord_flip()
```

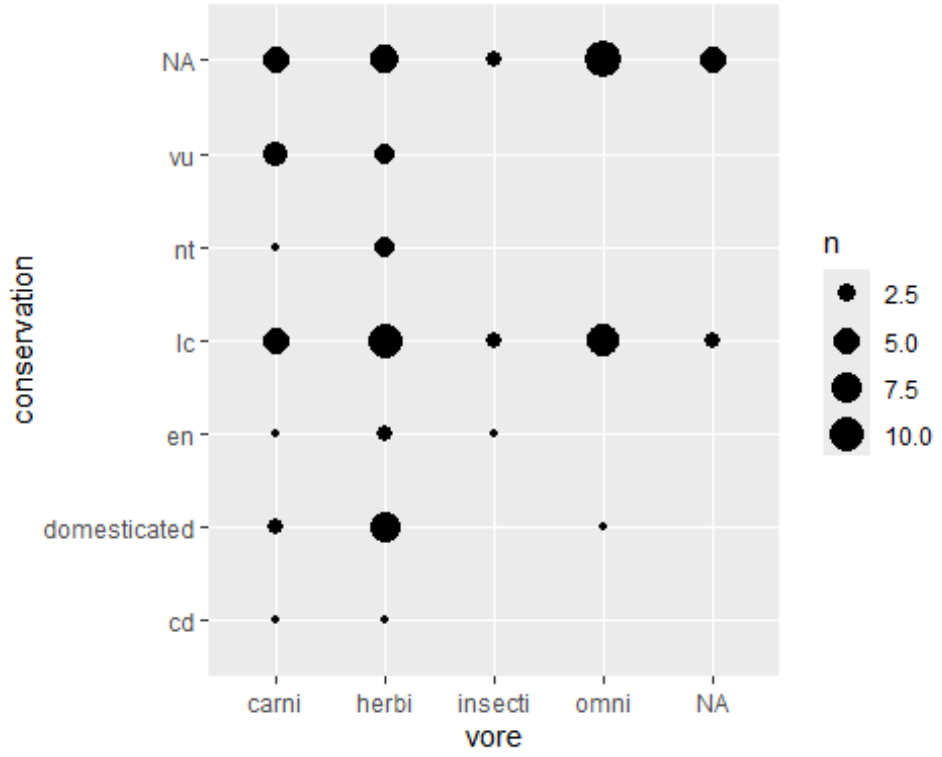


Γράφημα 47

4.7. Δύο κατηγορηματικές μεταβλητές

Παρόλο που αυτή η περίπτωση δεν είναι τόσο συχνή όσο η προηγούμενη υπάρχει μια σειρά γραφημάτων που μπορούν να βοηθήσουν. Για να απεικονίσουμε τη συνδιακύμανση μεταξύ των δύο κατηγορηματικών μεταβλητών θα χρειαστούμε την καταμέτρηση των παρατηρήσεων για κάθε συνδυασμό ζεύγους τιμών. Αυτό μπορεί να γίνει και με το geom_count() όπως υλοποιείται από τον ακόλουθο κώδικα.

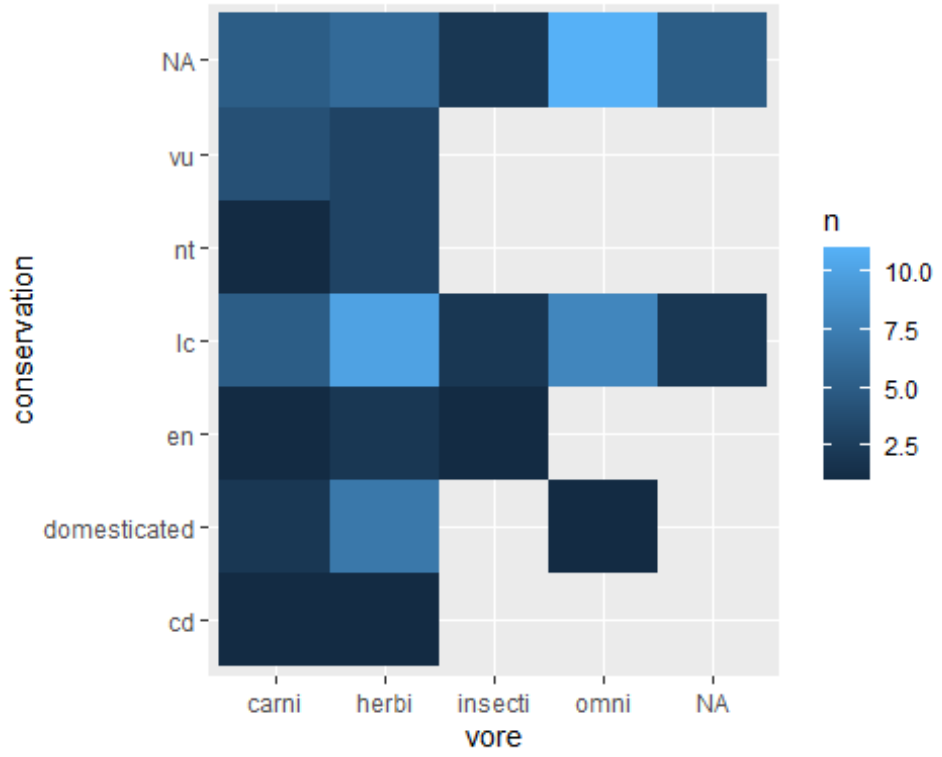
```
ggplot(data = msleep) + geom_count(mapping = aes(x = vore, y = conse
rvation))
```



Γράφημα 48

Το μέγεθος των κύκλων είναι ανάλογο του πλήθους των παρατηρήσεων για κάθε συνδυασμό ζεύγους τιμών. Η συνδιακύμανση θα εμφανιστεί ως ισχυρή συσχέτιση μεταξύ των δύο μεταβλητών όταν αυξάνεται η συχνότητα καθώς αυξάνονται οι τιμές εμφανίσεων του ζεύγους. Ένας άλλος τρόπος εμφάνισης αυτής της σχέσης είναι η χρήση του geom_tile() όπως υλοποιείται από τον ακόλουθο κώδικα.

```
msleep %>% count(vore, conservation) %>% ggplot(mapping = aes(x = vo
re, y = conservation)) + geom_tile(mapping = aes(fill = n))
```



Γράφημα 49

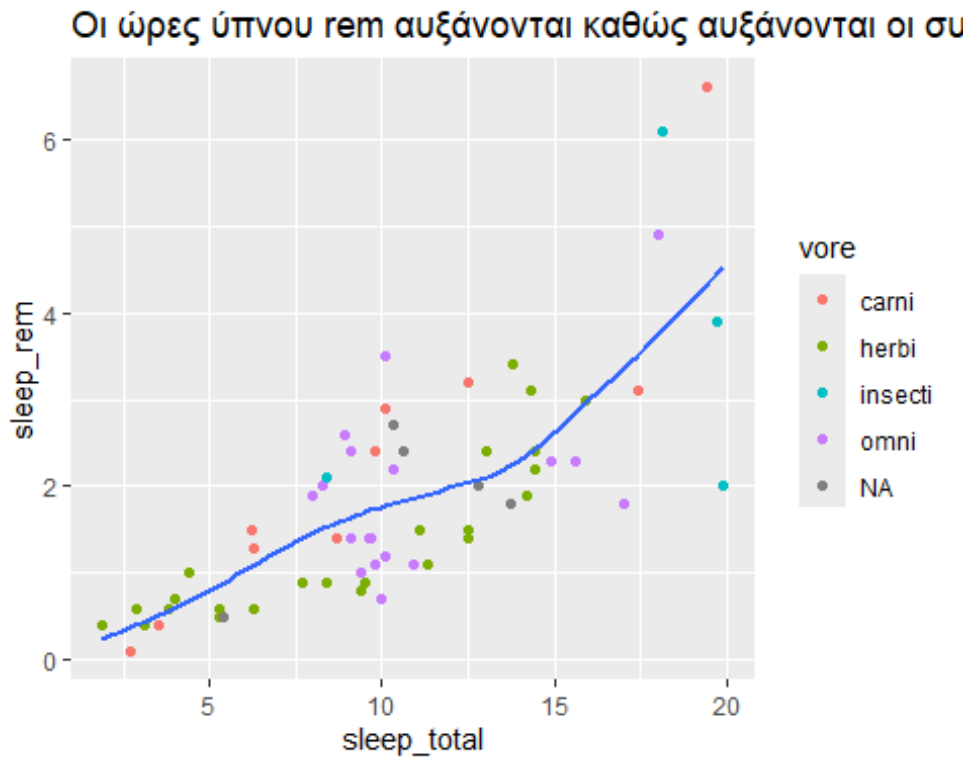
4.8. Μορφοποίηση των αποτελεσμάτων της ανάλυσης με το ggplot2

Όταν έχουμε κατανοήσει ικανοποιητικά τα δεδομένα τότε θα χρειαστεί να τα επικοινωνήσουμε και σε άλλους. Φυσικά αυτοί δεν γνωρίζουν ότι ξέρουμε ήδη εμείς για αυτά και για να τους βοηθήσουμε θα πρέπει να καταβάλουμε αρκετή προσπάθεια έτσι ώστε τα γραφήματα μας να ερμηνεύονται από μόνα τους. Αυτό σημαίνει πως θα χρειαστεί η μορφοποίηση του γραφήματος είτε με την τροποποίηση των δομικών στοιχείων του ή/ και με την προσθήκη επιπλέον στοιχείων.

4.8.1. Ετικέτα (label)

Το πιο απλό στοιχείο για να ξεκινήσουμε την προσπάθεια μετατροπής ενός στοιχειώδους εξερευνητικού διαγράμματος σε πιο επεξηγηματικό είναι η ετικέτα του τίτλου του. Μπορούμε να προσθέσουμε ετικέτες χρησιμοποιώντας τη συνάρτηση labs() όπως παρουσιάζεται στο ακόλουθο τμήμα κώδικα.

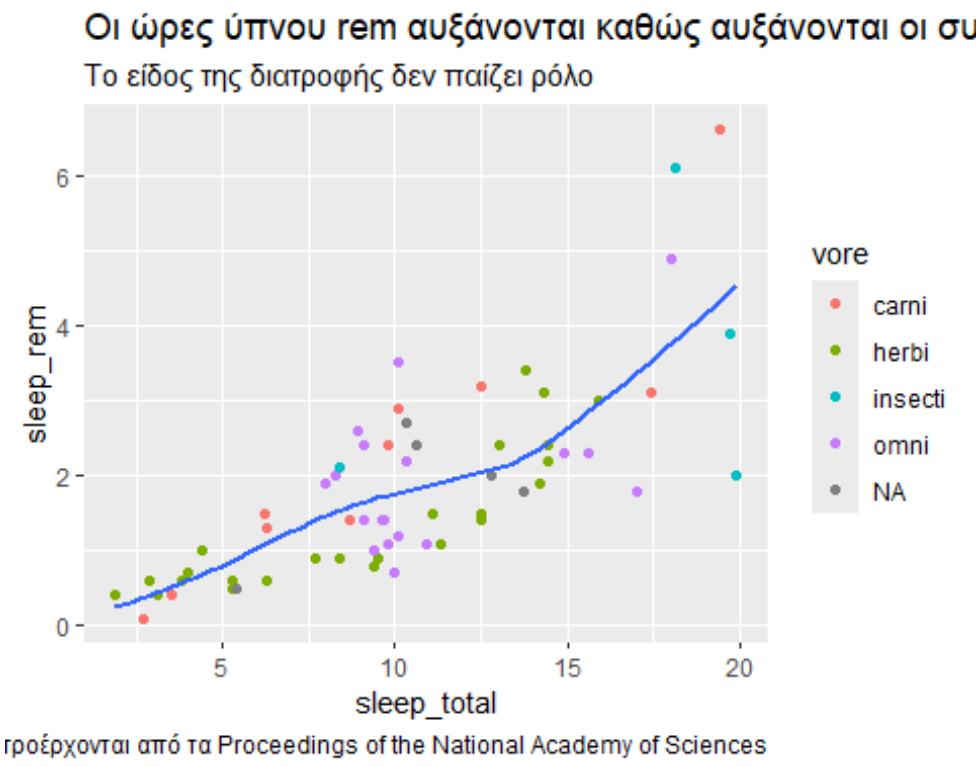
```
ggplot(msleep, aes(sleep_total, sleep_rem)) + geom_point(aes(color = vore)) + geom_smooth(se = FALSE) + labs(title = "Οι ώρες ύπνου rem αυξάνονται καθώς αυξάνονται οι συνολικές ώρες ύπνου")
```



Γράφημα 50

Ο τίτλος του γραφήματος συνήθως συνοψίζει το κύριο εύρημα αλλά κάποιες φορές δεν αρκεί. Οι νεότερες εκδόσεις του πακέτου ggplot2 προσφέρουν επιπλέον δυνατότητες όπως το subtitle, έναν μικρότερο υπότιτλο κάτω από τον κύριο και το caption που προσθέτει κείμενο κάτω δεξιά (όπου συνήθως αναγράφεται η πηγή της προέλευσης των δεδομένων). Η χρήση όλων αυτών μαζί φαίνεται στο ακόλουθο τμήμα κώδικα.

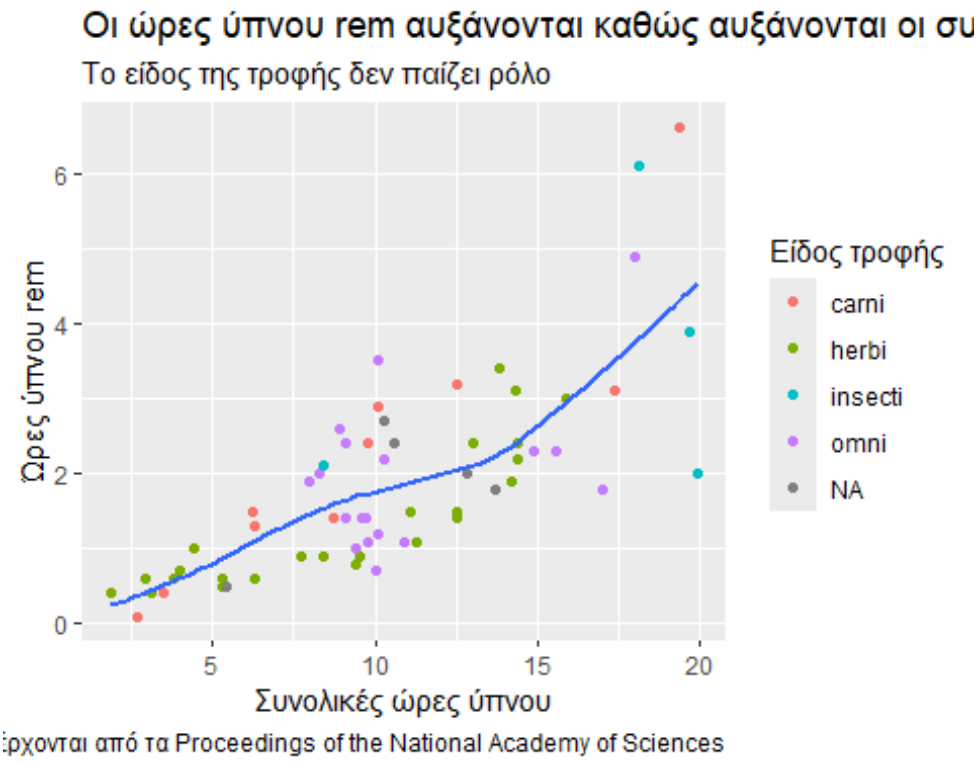
```
ggplot(msleep, aes(sleep_total, sleep_rem)) + geom_point(aes(color = vore)) + geom_smooth(se = FALSE) + labs( title = "Οι ώρες ύπνου rem αυξάνονται καθώς αυξάνονται οι συνολικές ώρες ύπνου", subtitle = "Το είδος της διατροφής δεν παίζει ρόλο", caption = "Τα δεδομένα προέρχονται από τα Proceedings of the National Academy of Sciences")
```



Γράφημα 51

Μπορούμε να χρησιμοποιήσουμε τη συνάρτηση labs() για να αντικαταστήσουμε τους προκαθορισμένους τίτλους των αξόνων και του υπομνήματος, δηλαδή τα ονόματα των μεταβλητών. Συνήθως αυτό γίνεται για να αντικατασταθούν τα σύντομα και ακατανόητα ονόματα των μεταβλητών με πιο λεπτομερείς περιγραφές τους.

```
ggplot(msleep, aes(sleep_total, sleep_rem)) + geom_point(aes(color = vore)) + geom_smooth(se = FALSE) + labs( title = "Οι ώρες ύπνου rem αυξάνονται καθώς αυξάνονται οι συνολικές ώρες ύπνου", subtitle = "Το είδος της τροφής δεν παίζει ρόλο", caption = "Τα δεδομένα προέρχονται από τα Proceedings of the National Academy of Sciences", x = "Συνολικές ώρες ύπνου", y = "Ώρες ύπνου rem", colour = "Είδος τροφής")
```

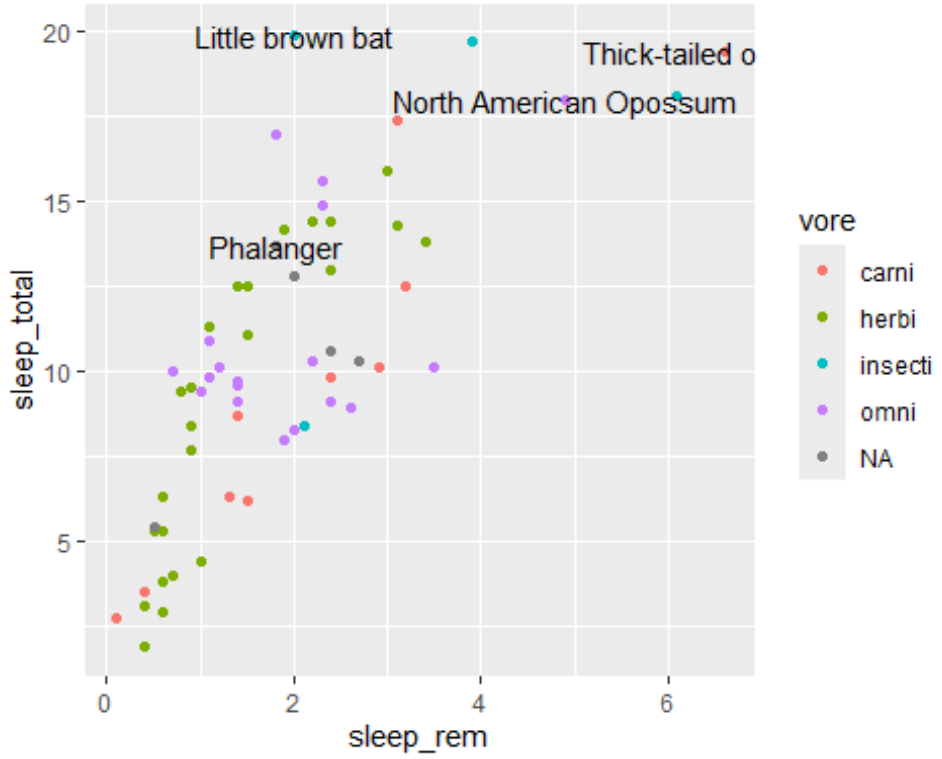


Γράφημα 52

4.8.2. Σχόλια

Κάποιες φορές επιθυμούμε να τοποθετήσουμε από μία ετικέτα μηνύματος σε μεμονωμένες παρατηρήσεις ή ομάδες τους. Το γεωμετρικό αντικείμενο geom_text() μπορεί να μας βοηθήσει σε αυτή την περίπτωση. Ας δούμε το ακόλουθο τμήμα κώδικα.

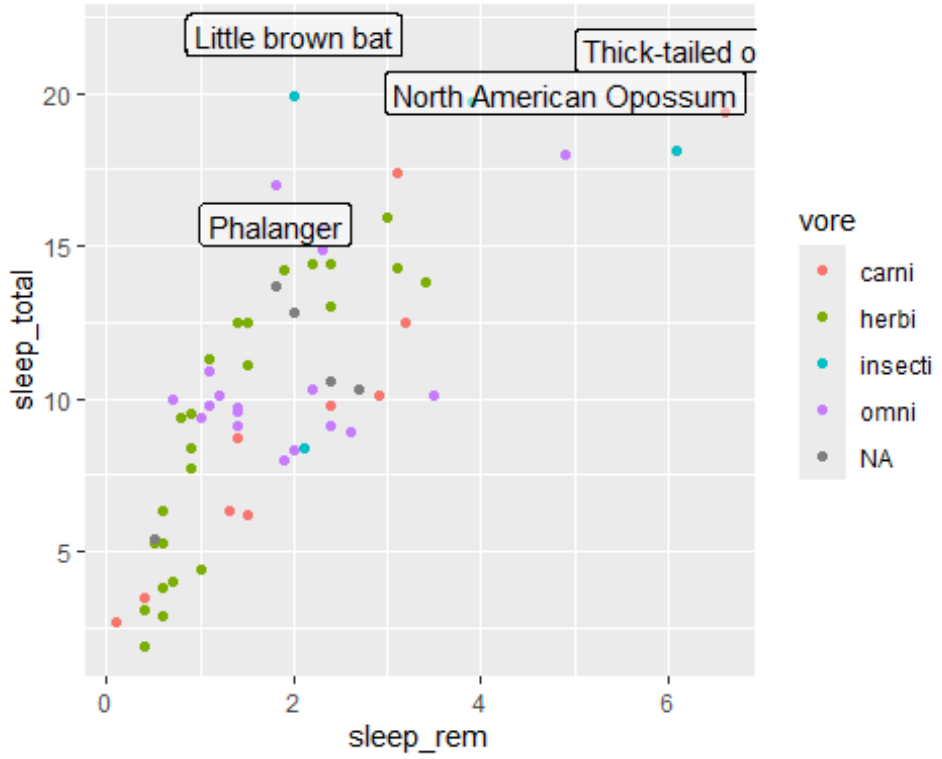
```
best_in_vore <- msleep %>% group_by(vore) %>% filter(row_number(desc
(sleep_total)) == 1)
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point(aes(colour
= vore)) + geom_text(aes(label = name), data = best_in_vore)
```



Γράφημα 53

Το πρόβλημα του γραφήματος είναι η επικάλυψη των σημείων. Για να βελτιωθεί η εμφάνιση του γραφήματος μπορούμε να χρησιμοποιήσουμε το geom_label που δημιουργεί ένα περίγραμμα στις ετικέτες των μεμονωμένων σημείων και χρησιμοποιώντας την παράμετρο nudge_y αποφεύγουμε τις επικαλύψεις σημείων με την ελαφριά μετακίνηση κάποιων που συμπίπτουν. Ακολουθεί το σχετικό τμήμα κώδικα.

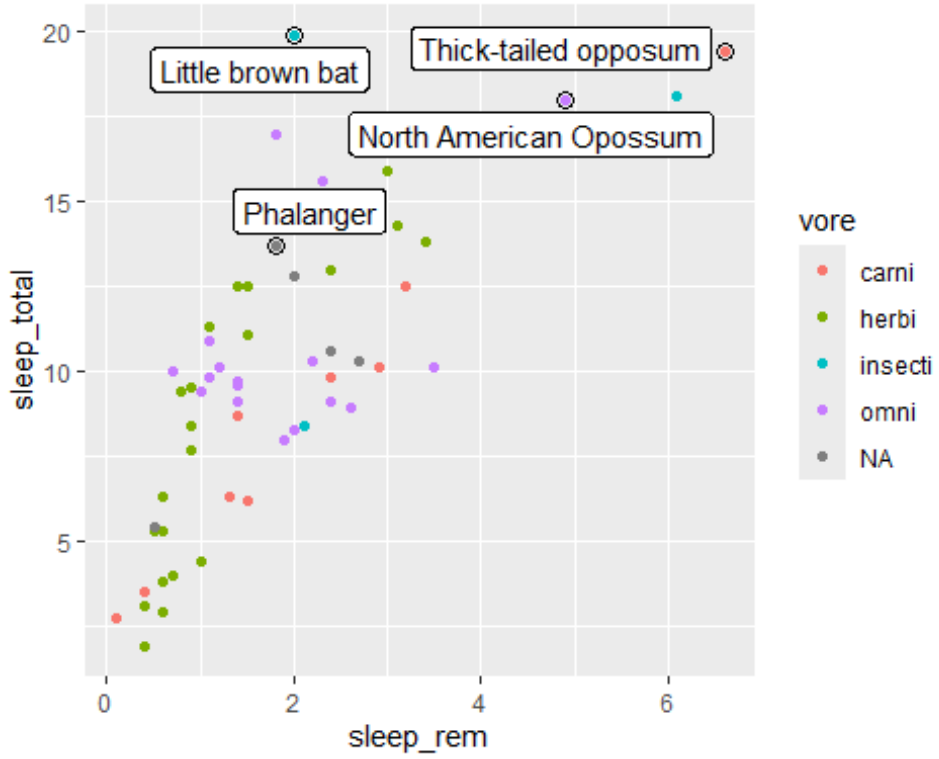
```
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point(aes(colour
= vore)) + geom_label(aes(label = name), data = best_in_vore, nudge_
y = 2, alpha = 0.5)
```



Γράφημα 54

Μια πιο αποτελεσματική λύση στο πρόβλημα των επικαλύψεων δίνει το πολύ χρήσιμο πρόσθετο πακέτο ggrepel() όπως φαίνεται στο παρακάτω τμήμα κώδικα.

```
library(ggrepel)
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point(aes(colour
= vore)) + geom_point(size = 3, shape = 1, data = best_in_vore) + gg
repel::geom_label_repel(aes(label = name), data = best_in_vore)
```



Γράφημα 55

4.8.3. Κλίμακες (scales)

Κάποιες φορές για να γίνει ένα γράφημα πιο εποπτικό είναι η προσαρμογή και ο έλεγχος μίας κλίμακας του. Στην πραγματικότητα το πακέτο ggplot2 αυτόματα προσθέτει τις κλίμακες που είναι οι εξής:

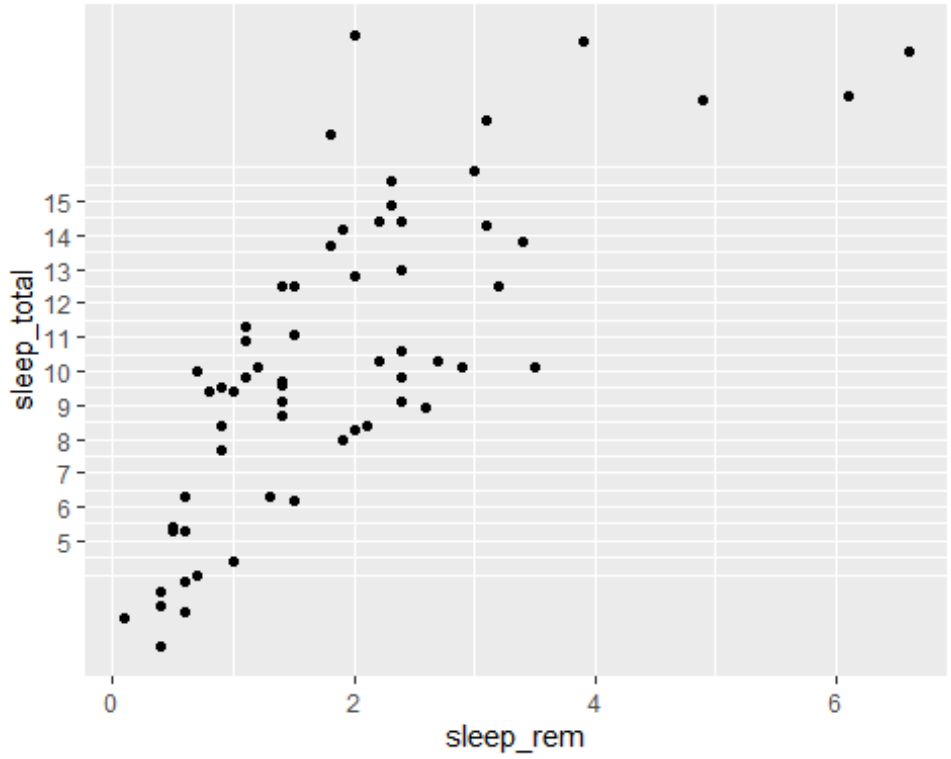
scale_x_continuous()

scale_y_continuous()

scale_colour_discrete()

Οι δύο πρώτες αντιπροσωπεύουν τις δύο διαστάσεις του επιπέδου ενώ η τρίτη τη διακριτή κλίμακα των χρωμάτων. Ανάλογα με τη μορφή του γραφήματος και τα δεδομένα του, προχωρούμε στην προσαρμογή των κλιμάκων για δύο λόγους: α) για να κάνουμε μικροδιορθώσεις κάποιων παραμέτρων μίας προκαθορισμένης κλίμακας και β) όταν θέλουμε να τις αντικαταστήσουμε πλήρως. Δύο παράμετροι που επηρεάζουν τις κλίμακες των διαστάσεων του επιπέδου είναι τα breaks και τα labels. Τα breaks καθορίζουν σημεία της κλίμακας (αρχή, τέλος, βήμα), όπως φαίνεται στο παρακάτω τμήμα κώδικα.

```
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point() + scale_y
_continuous(breaks = seq(5, 15, by = 1))
```

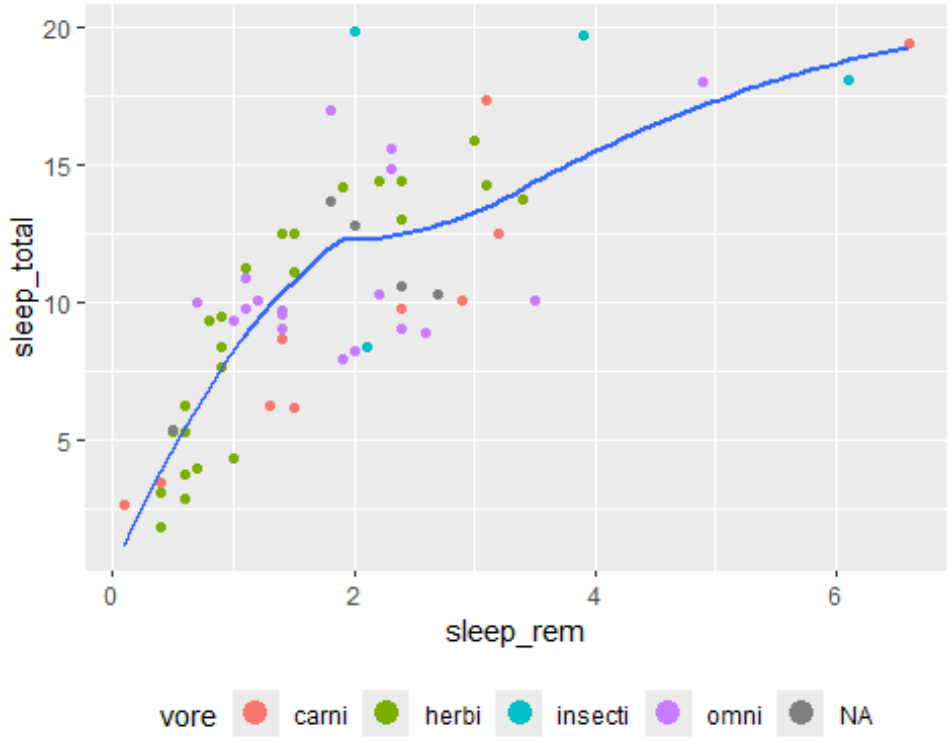


Γράφημα 56

4.8.4. Επίπεδο υπομνήματος

Για να ελέγχουμε τη συνολική τοποθέτηση ενός υπομνήματος θα χρειαστούμε τη ρύθμιση theme(). Το θέμα legend_position είναι υπεύθυνο για αυτή τη ρύθμιση όπως φαίνεται στον ακόλουθο κώδικα.

```
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point(aes(colour
= vore)) + geom_smooth(se = FALSE) + theme(legend.position = "bottom
") + guides(colour = guide_legend(nrow = 1, override.aes = list(size
= 4)))
```

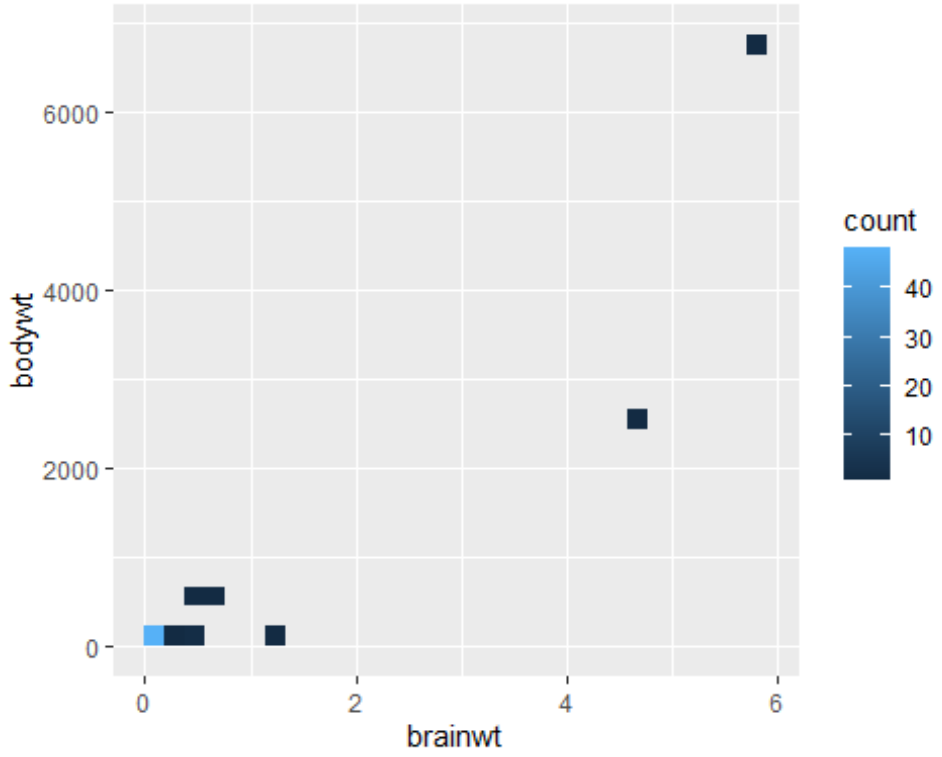


Γράφημα 57

4.8.5. Αντικατάσταση μιας κλίμακας

Αντί της ελαφράς τροποποίησης, μπορούμε να αντικαταστήσουμε πλήρως την κλίμακα. Υπάρχουν δύο τύποι κλίμακας που επιθυμούμε συνήθως να αλλάξουμε, τις κλίμακες θέσης και χρώματος. Φυσικά, λόγω ομοιομορφίας των ρυθμίσεων των aesthetics ότι ισχύει για τις δύο κλίμακες ισχύει και για τις υπόλοιπες. Πολλές φορές είναι χρήσιμη η μετατροπή μιας μεταβλητής σε λογαριθμική κλίμακα όπως στον παρακάτω κώδικα.

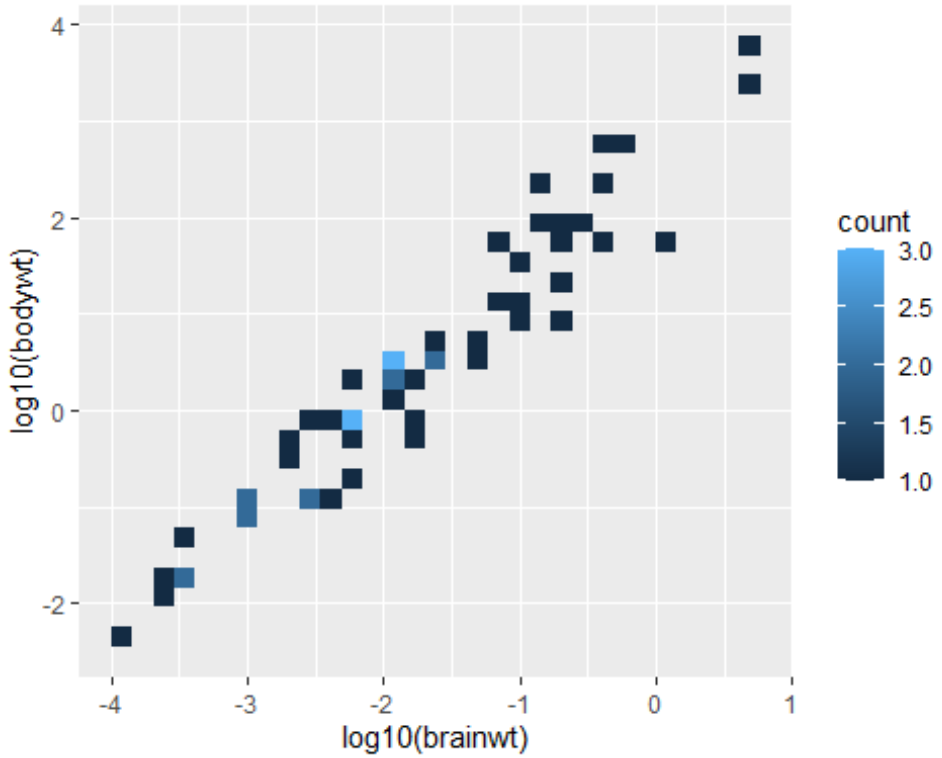
```
ggplot(msleep, aes(brainwt, bodywt)) + geom_bin2d()
```



Γράφημα 58

Όπως παρατηρείτε το παραπάνω γράφημα λόγω των μεγάλων αποκλίσεων των τιμών δεν βοηθά. Για αυτό στο επόμενο τμήμα κώδικα γίνεται μια λογαριθμική μετατροπή.

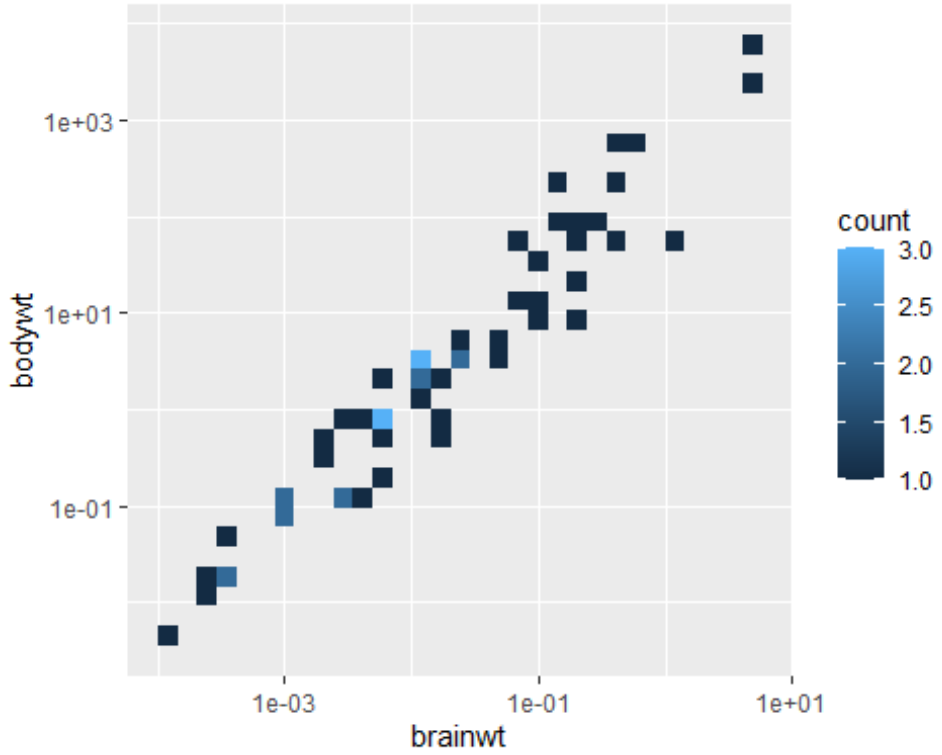
```
ggplot(msleep, aes(log10(brainwt), log10(bodywt))) + geom_bin2d()
```



Γράφημα 59

Δυστυχώς το μειονέκτημα αυτής της μετατροπής είναι πως οι άξονες έχουν ως ετικέτες αυτές των τροποποιημένων τιμών (λογαριθμικών συναρτήσεων) κάτι που δυσκολεύει την ερμηνεία του διαγράμματος. Εναλλακτικά, η βέλτιστη μετατροπή μπορεί να γίνει μέσω της συνολικής αλλαγής κλίμακας ως εξής:

```
ggplot(msleep, aes(brainwt, bodywt)) + geom_bin2d() + scale_x_log10(
) + scale_y_log10()
```

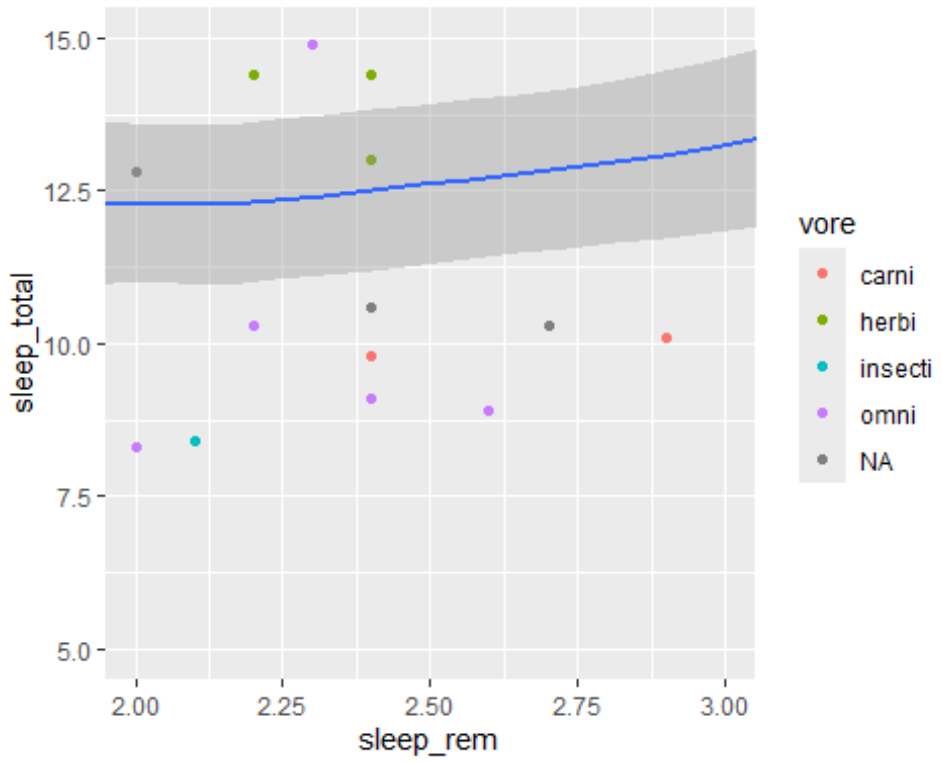


Γράφημα 60

4.8.6. Μεγέθυνση-Σμίκρυνση

Γενικώς υπάρχουν τρεις τρόποι για να ελεγχθούν τα όρια του γραφήματος και το ποσοστό μεγέθυνσης- σμίκρυνσης: α) προσαρμογή των ίδιων των δεδομένων, β) ρύθμιση των ορίων της κάθε κλίμακας των δύο διαστάσεων και γ) καθορισμός των xlim και ylim στη συνάρτηση coord_cartesian(). Για να μεγεθύνουμε σε μια περιοχή του γραφήματος είναι προτιμότερη η χρήση της coord_cartesian(). Ας δούμε όμως τον παρακάτω κώδικα:

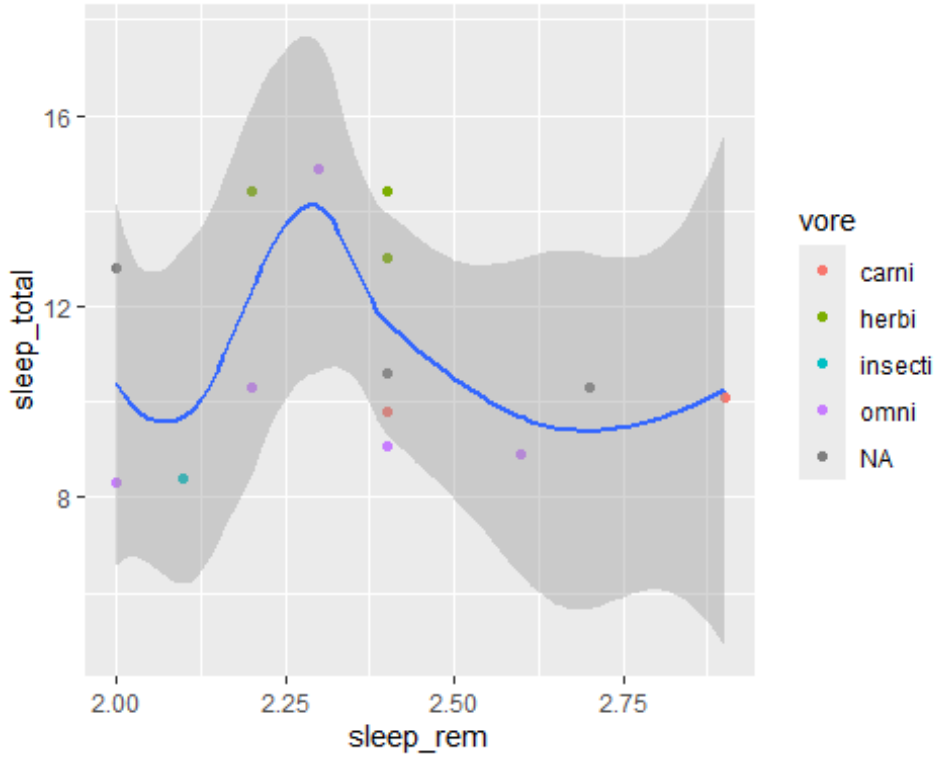
```
ggplot(msleep, mapping = aes(sleep_rem, sleep_total)) + geom_point(a
es(color = vore)) + geom_smooth() + coord_cartesian(xlim = c(2, 3),
ylim = c(5, 15))
```



Γράφημα 61

Το παραπάνω γράφημα έγινε με τον τρίτο τρόπο ενώ το ακόλουθο γίνεται με τον πρώτο τρόπο, το φιλτράρισμα δεδομένων. Η διαφορά τους είναι ορατή ενώ θεωρητικά θα έπρεπε να παράγουν το ίδιο αποτέλεσμα. Η αιτία είναι η εξαγωγή κάποιων δεδομένων από τον υπολογισμό της θέσης της γραμμής.

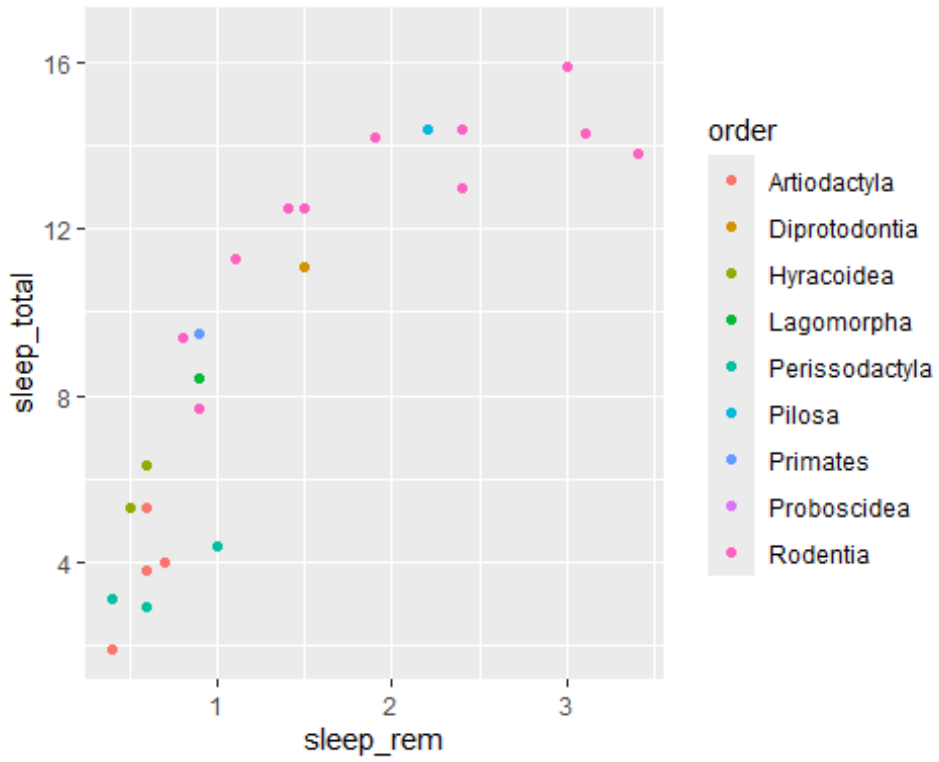
```
msleep %>% filter(sleep_rem >= 2, sleep_rem <= 3, sleep_total >= 5,
sleep_total <= 15) %>% ggplot(aes(sleep_rem, sleep_total)) + geom_po
int(aes(color = vore)) + geom_smooth()
```



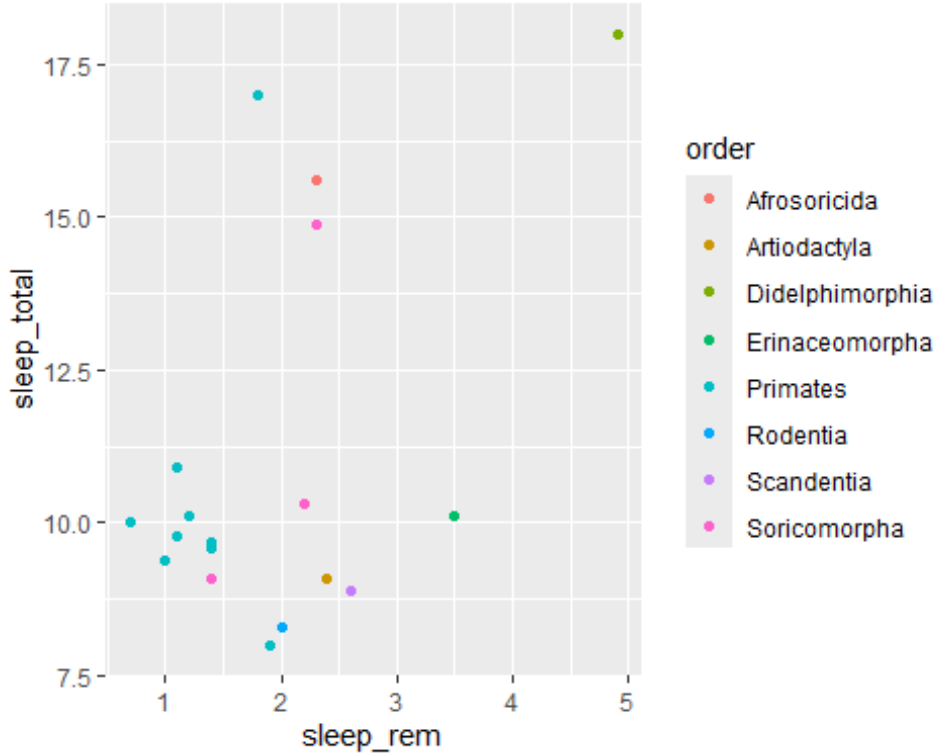
Γράφημα 62

Μπορούμε επίσης να θέσουμε απευθείας τα όρια των κλιμάκων. Η μείωση των ορίων ισοδυναμεί με την επιλογή ενός υποσυνόλου των δεδομένων. Μια χρήσιμη πρακτική είναι ο καθορισμός των κλιμάκων από τα όρια των δεδομένων όπως στο ακόλουθο παράδειγμα.

```
herbi <- msleep %>% filter(vore == "herbi")
omni <- msleep %>% filter(vore == "omni")
ggplot(herbi, aes(sleep_rem, sleep_total, colour = order)) + geom_po
int()
```

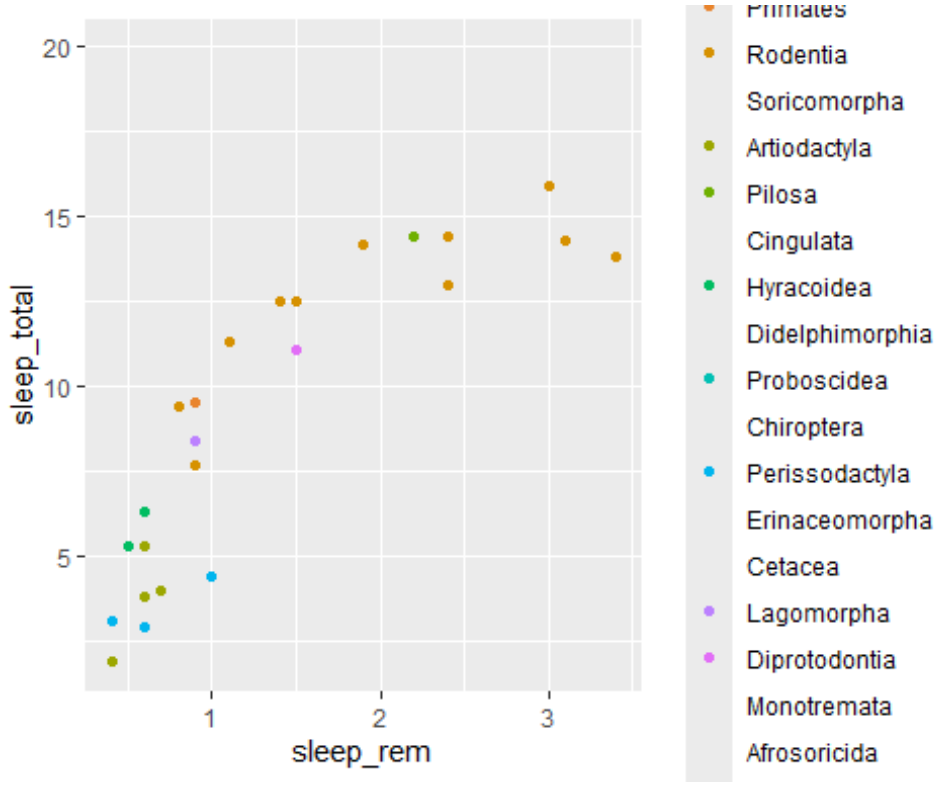



```
ggplot(omni, aes(sleep_rem, sleep_total, colour = order)) + geom_poi
nt()
```

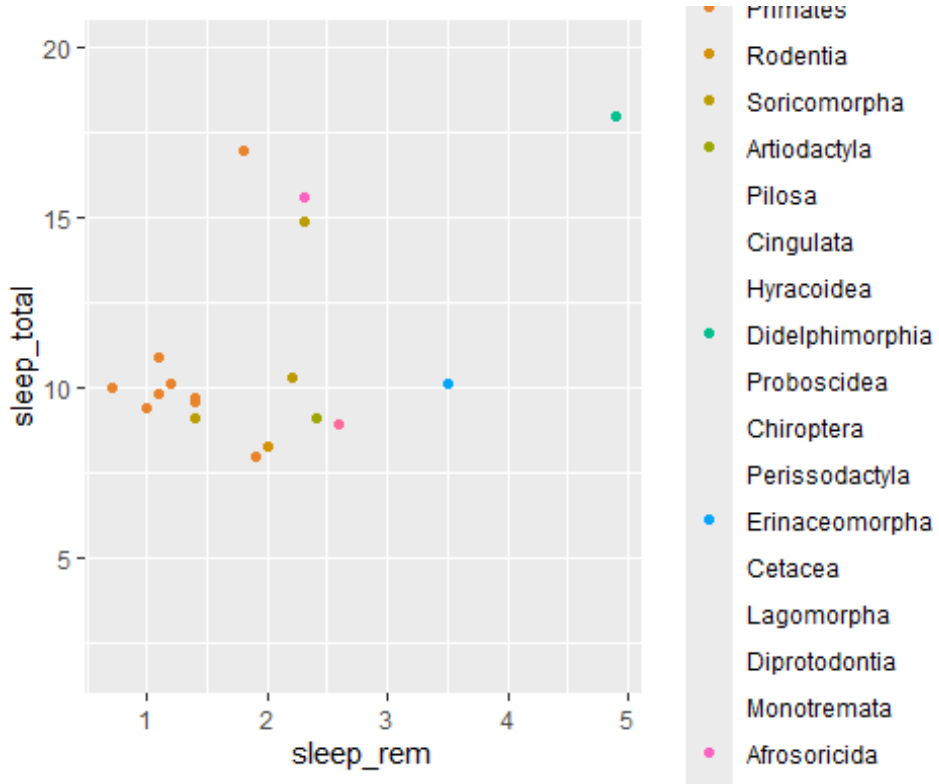



```
x_scale <- scale_x_continuous(limits = range(msleep$sleep_rem))
y_scale <- scale_y_continuous(limits = range(msleep$sleep_total))
col_scale <- scale_colour_discrete(limits = unique(msleep$order))
```

```
ggplot(herbi, aes(sleep_rem, sleep_total, colour = order)) + geom_po
int() + x_scale + y_scale + col_scale
```




```
 ggplot(omni, aes(sleep_rem, sleep_total, colour = order)) + geom_po
int() + x_scale + y_scale + col_scale
```



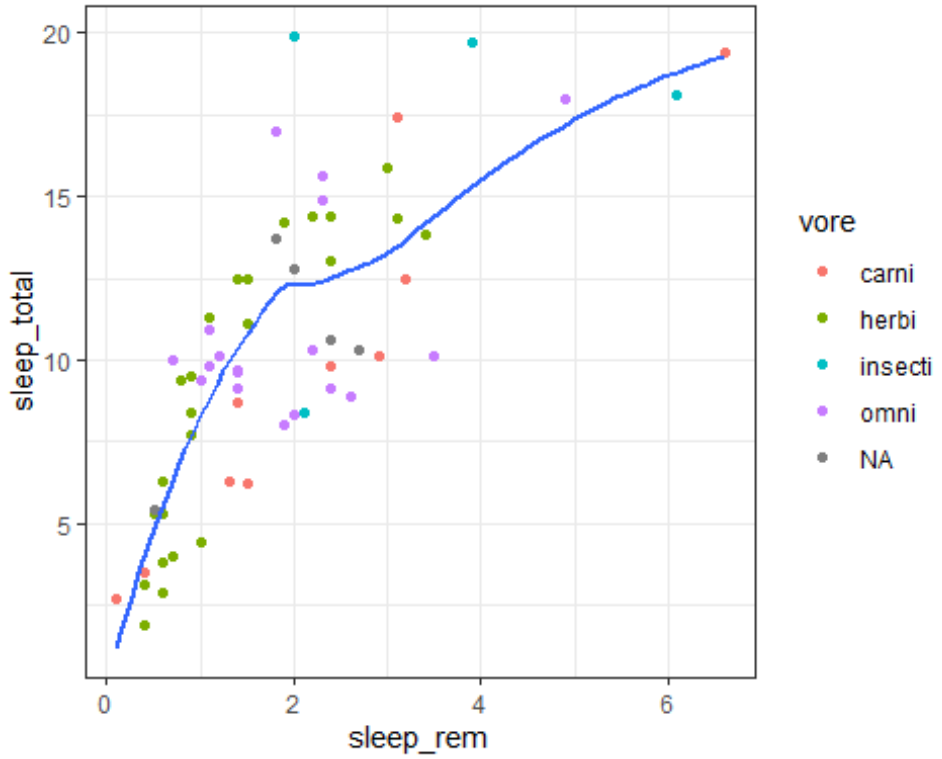
Γραφήματα 63

Το ίδιο αποτέλεσμα θα μπορούσε να επιτευχθεί και με χρήση των facets.

4.8.7. Θέματα

Το πακέτο ggplot2 διαθέτει ένα ακόμη ισχυρό εργαλείο για τη μαζική μορφοποίηση ενός γραφήματος, το θέμα (theme). Σύμφωνα με αυτό μπορούμε να μορφοποιήσουμε τις ρυθμίσεις που αφορούν τα δεδομένα του γραφήματος με τη χρήση ενός θέματος όπως στο παρακάτω τμήμα κώδικα.

```
ggplot(msleep, aes(sleep_rem, sleep_total)) + geom_point(aes(color = vore)) + geom_smooth(se = FALSE) + theme_bw()
```



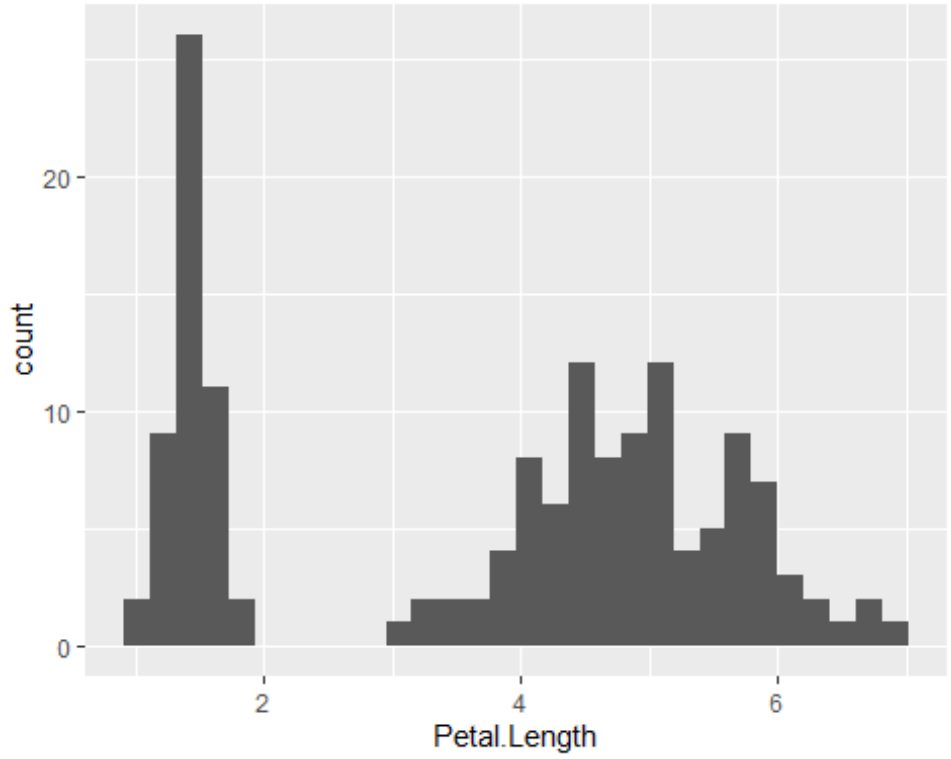
Γράφημα 64

Το βασικό πακέτο ggplot2 προσφέρει οκτώ προκαθορισμένα θέματα αλλά μπορούμε να βρούμε πολύ περισσότερα σε άλλα πρόσθετα πακέτα όπως το ggthemes. Το προκαθορισμένο θέμα έχει γκρίζες και λευκές διαχωριστικές γραμμές για λόγους καλύτερης εποπτικότητας του τελικού αποτελέσματος. Εδώ πρέπει να σημειωθεί πως μπορούμε να ελέγξουμε ξεχωριστά τμήματα του κάθε θέματος όπως το μέγεθος, το χρώμα κλπ. Επιπλέον, ως χρήστες μπορούμε να δημιουργήσουμε τα δικά μας θέματα.

4.8.7.1. Δημιουργία νέου θέματος με το πακέτο ggplot2.

Το πακέτο ggplot εκτός των έτοιμων θεμάτων δίνει στο χρήστη τη δυνατότητα δημιουργίας νέων θεμάτων με χρήση της συνάρτηση theme() όπως παρουσιάζεται στο ακόλουθο παράδειγμα. Αυτό που απαιτείται είναι απόδοση των ορθών ρυθμίσεων στη συνάρτηση theme(). Ας δούμε το παρακάτω τμήμα κώδικα που παράγει ένα τυπικό ιστόγραμμα.

```
ggplot(data=iris, aes(x=Petal.Length))+
  geom_histogram()
```

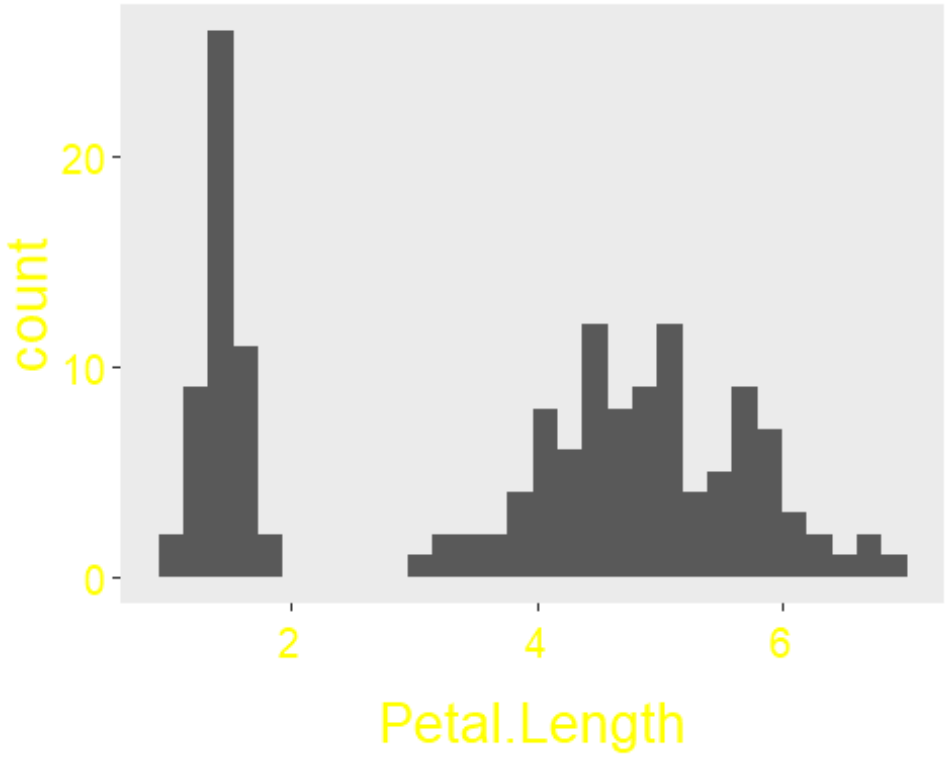


Γράφημα 65

Ας υποθέσουμε πως για τους δικούς μας λόγους επιθυμούμε να εξαφανίσουμε τις γραμμές πλέγματος, κύριες και δευτερεύουσες, να έχουμε σε κάθε στοιχείο του γραφήματος ως γραμματοσειρά την Arial και χρώμα να έχουμε το κίτρινο στους άξονες και στους τίτλους τους. Τότε το προηγούμενο τμήμα κώδικα τροποποιείται ως εξής προσθέτοντας ένα θέμα που συγκεντρώνει όλες τις μορφοποιήσεις που αφορούν τα κύρια στοιχεία που ρυθμίζονται από τη συνάρτηση labs():

```
ggplot(data=iris, aes(x=Petal.Length))+
  geom_histogram()+
  theme(
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),

      plot.title = element_text(
                  family = "Arial",
                  size = 20,
                  face = 'bold',
                  hjust = 0,
                  vjust = 2),

      plot.subtitle = element_text(
                  family = "Arial",
                  size = 14),

      plot.caption = element_text(
                  family = "Arial",
                  size = 9,
```

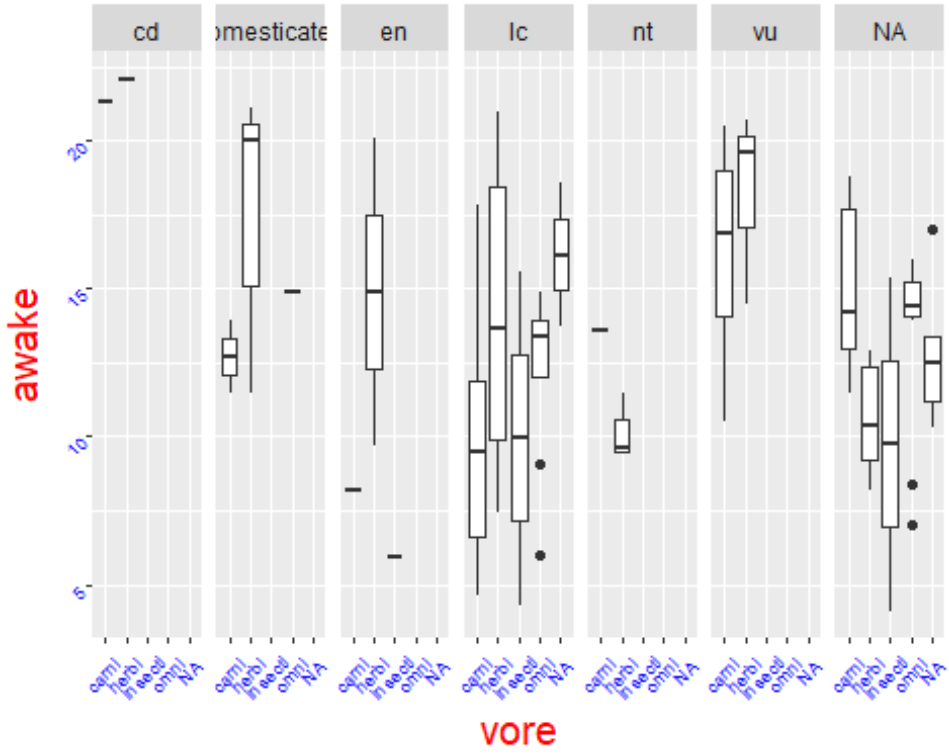
```
                    hjust = 1),

      axis.title = element_text(
                    family = "Arial",
                    color="yellow",
                    size = 20),

      axis.text = element_text(
                    family = "Arial",
                    color="yellow",
                    size = 15),

      axis.text.x = element_text(
                     margin=margin(5, b = 10))
    )
```



Γράφημα 66

Ακολουθεί ένα ακόμη παράδειγμα δημιουργία θέματος που παράγεται από το επόμενο τμήμα κώδικα.

```
ggplot(data=msleep)+
  geom_boxplot(aes(vore, awake))+
  facet_grid(.~ conservation)+
  theme(axis.text.y=element_text(color="blue", size=7, angle=45,vjust=0.8,
hjust=0.8),
        axis.text.x=element_text(color="blue", size=7, angle=45,vjust=0.8,
hjust=0.8),
        axis.title.x=element_text(color="red", size=15),
        axis.title.y=element_text(color="red", size=15),
        legend.title=element_text(size=20))
```



Γράφημα 67

Ερωτήσεις αυτοαξιολόγησης

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

15. Εάν επιθυμούμε να περιστρέψουμε το γράφημα κατά 90 μοίρες θα τροποποιήσουμε το προκαθορισμένο σύστημα συντεταγμένων επιλέγοντας την επιλογή:

α. coord_flip()

β. coord_quickmap()

γ. coord_polar()

δ. coord_cartesian()

16. Ποιο από τα παρακάτω γεωμετρικά αντικείμενα δεν χρησιμοποιείται για την αναπαράσταση μίας κατανομής τιμών μίας μεταβλητής;

α. geom_freqpoly()

β. geom_histogram()

γ. geom_bar()

δ. geom_tile()

17. Με ποιο γεωμετρικό αντικείμενο αποφεύγουμε τις επικαλύψεις μηνυμάτων (text) μέσα στο γράφημα;

α. geom_text()

β. ggrepel::geom_label_repel()

γ. geom_label_repel()

δ. geom_rect()

18. Ποιος είναι ο προτιμότερος τρόπος για να ρυθμιστεί η κλίμακα μεγέθυνσης ενός γραφήματος;

α. προσαρμογή των δεδομένων με φιλτράρισμα τους

β. ρύθμιση των ορίων της κάθε κλίμακας

γ. ρύθμιση των xlim και ylim της συνάρτησης coord_cartesian()

δ. κανένας από τους παραπάνω

19. Με ποια ρύθμιση αποφεύγεται η επικάλυψη σημείων σε ένα διάγραμμα διασποράς (scatter diagram);

α. δίνοντας στην παράμετρο position την τιμή jitter στο γεωμετρικό αντικείμενο geom_point

β. επιλέγοντας τη συνάρτηση coord_flip

γ. δίνοντας στην παράμετρο position την τιμή fill στο γεωμετρικό αντικείμενο geom_bar

δ. δίνοντας στην παράμετρο position την τιμή fill στο γεωμετρικό αντικείμενο geom_point

20. 6. Ποιο από τα παρακάτω δεν είναι ρύθμιση της συνάρτησης labs();

α. theme

β. subtitle

γ. x

δ. caption

Κεφάλαιο 5. Ασκήσεις με γραφήματα

5.1. Ασκήσεις ggplot2- Μέρος 1ο

1. Τρέξτε την κλήση της συνάρτησης ggplot(data=msleep). Τι παρατηρείτε; Στη συνέχεια φτιάξτε το διάγραμμα διασκόρπισης για τις μεταβλητές sleep_total και bodywt. Τι συμβαίνει αν φτιάξετε το διάγραμμα διασκόρπισης με τις μεταβλητές conservation και vore; Γιατί δεν είναι χρήσιμο αυτό το διάγραμμα;

2. Βρείτε ποιες μεταβλητές είναι κατηγορηματικές στο msleep και ποιες είναι συνεχείς. Αποδώστε στον προηγούμενο κώδικα μια συνεχή μεταβλητή στις ιδιότητες colour, size και shape. Τι παρατηρείτε;

3. Φτιάξτε για το msleep και το διάγραμμα του πρώτου ερωτήματος ένα facet_grid επιλέγοντας δύο κατηγορηματικές μεταβλητές.

4. Ποιος ο ρόλος της ρύθμισης se στο geom_smooth(); Δοκιμάστε και τις δύο εναλλακτικές τιμές TRUE και FALSE για να δείτε τη διαφορά στο αποτέλεσμα.

5. Ποια είναι η προκαθορισμένη τιμή για το position του geom_boxplot();

6. Μετατρέψτε ένα διάγραμμα bar σε διάγραμμα pie χρησιμοποιώντας τις πολικές συντεταγμένες coord_polar().

7. Εξερευνήστε την κατανομή των ωρών μη ύπνου των θηλαστικών (msleep). Χρησιμοποιείστε διάφορες τιμές του binwidth για να βρείτε την κατάλληλη.

8. Ποια μεταβλητή προβλέπει καλύτερα τις ώρες ύπνου των θηλαστικών (msleep);

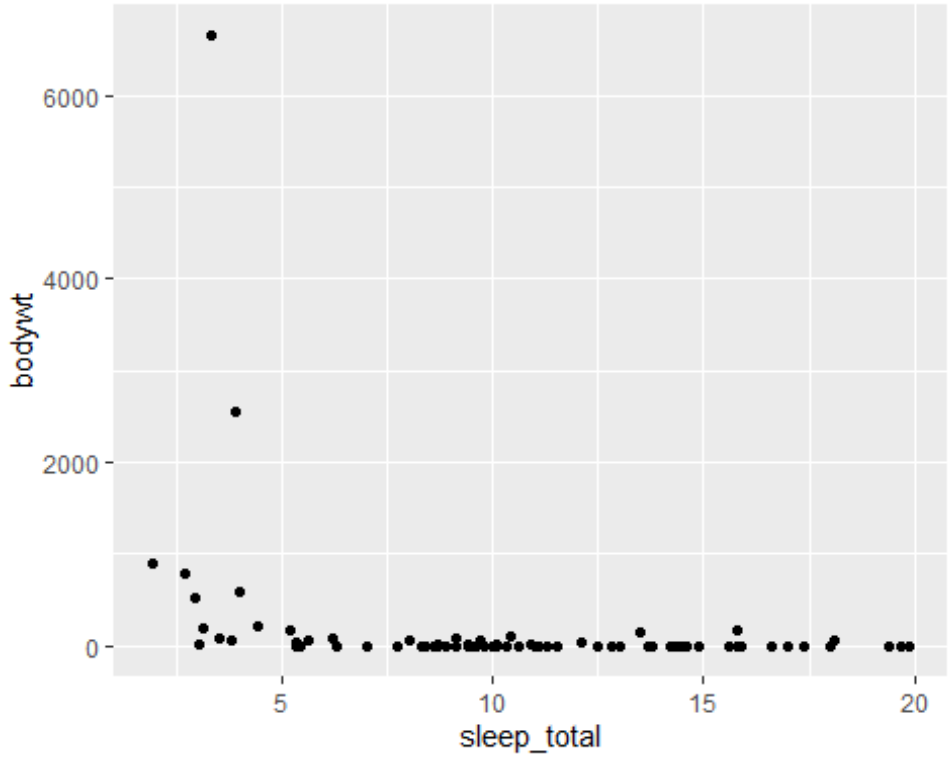
9. Χρησιμοποιείστε το geom_tile() για να εξερευνήσετε τη σχέση δύο κατηγορηματικών μεταβλητών του msleep.

10. Φτιάξτε ένα δικό σας διάγραμμα σε σχέση με τα δεδομένα ύπνου των θηλαστικών ρυθμίζοντας τους δικούς σας τίτλους, υπότιτλους κλπ.

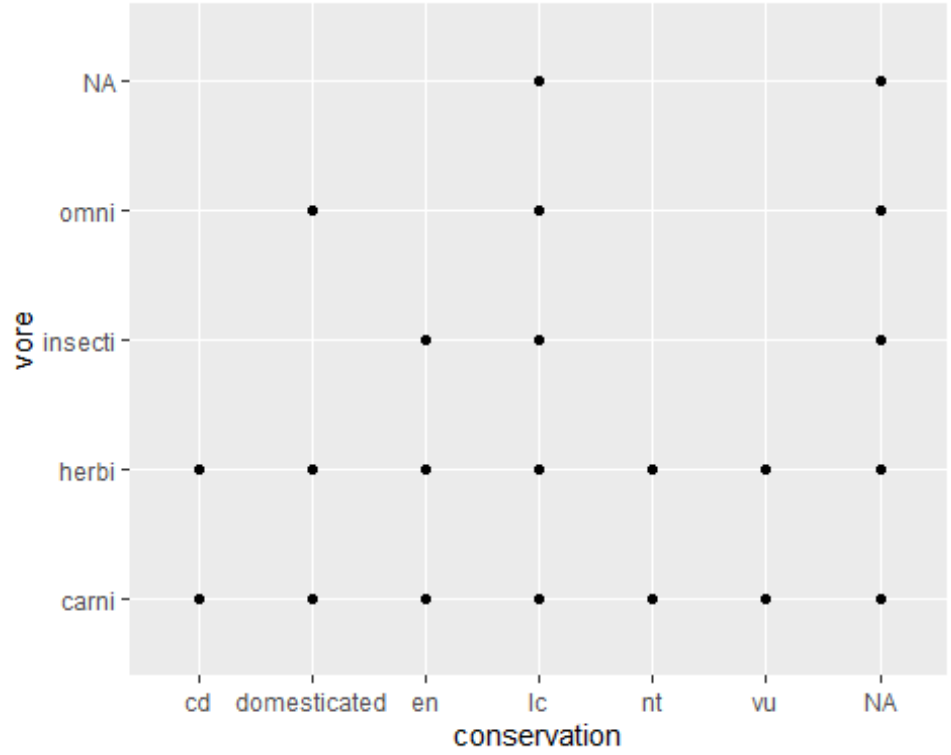
Ενδεικτική Λύση

1.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = bodywt))
```




```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = conservation, y = vore))
```



Γράφημα 68

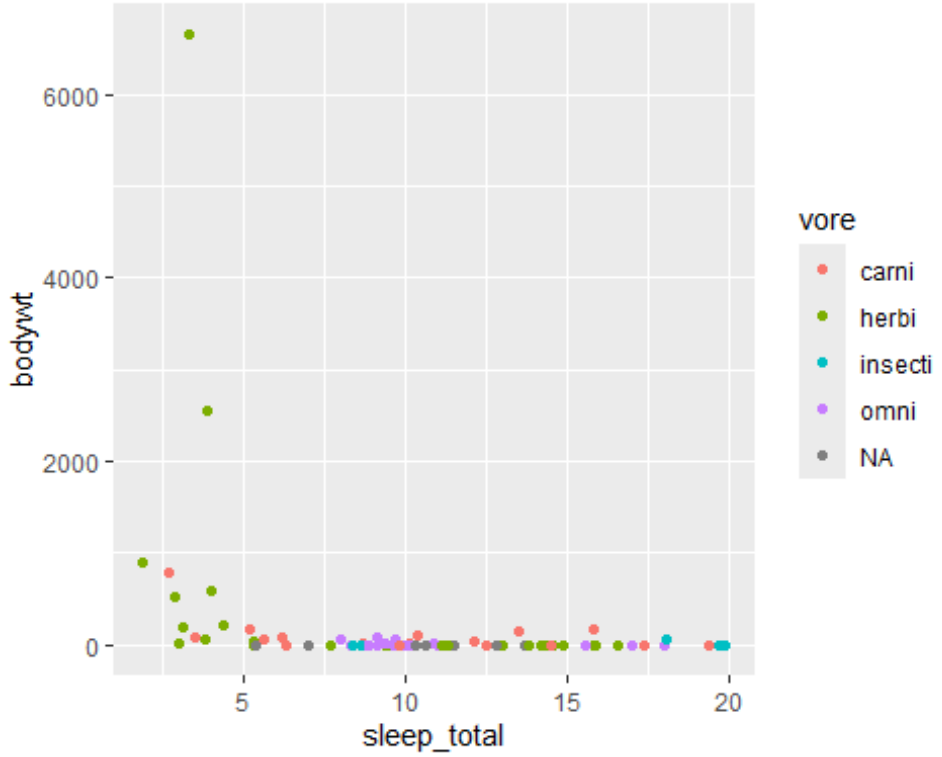
Ερμηνεία του κώδικα: αν εκτελεστεί η ggplot(data = msleep) απλώς εμφανίζεται ο καμβάς του διαγράμματος. Αν οι μεταβλητές conservation και vore δημιουργείται ένα διάγραμμα που δεν έχει νόημα γιατί και οι δύο μεταβλητές είναι κατηγορηματικές.

2.

```
str(msleep)
```

```
tibble [83 × 11] (S3: tbl_df/tbl/data.frame)
 $ name        : chr [1:83] "Cheetah" "Owl monkey" "Mountain beaver" "Great
er short-tailed shrew" ...
 $ genus       : chr [1:83] "Acinonyx" "Aotus" "Aplodontia" "Blarina" ...
 $ vore        : chr [1:83] "carni" "omni" "herbi" "omni" ...
 $ order       : chr [1:83] "Carnivora" "Primates" "Rodentia" "Soricomorpha
" ...
 $ conservation: chr [1:83] "lc" NA "nt" "lc" ...
 $ sleep_total : num [1:83] 12.1 17 14.4 14.9 4 14.4 8.7 7 10.1 3 ...
 $ sleep_rem   : num [1:83] NA 1.8 2.4 2.3 0.7 2.2 1.4 NA 2.9 NA ...
 $ sleep_cycle : num [1:83] NA NA NA 0.133 0.667 ...
 $ awake       : num [1:83] 11.9 7 9.6 9.1 20 9.6 15.3 17 13.9 21 ...
 $ brainwt     : num [1:83] NA 0.0155 NA 0.00029 0.423 NA NA NA 0.07 0.0982
...
 $ bodywt      : num [1:83] 50 0.48 1.35 0.019 600 ...
```

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = bodywt, colour = vore))
```

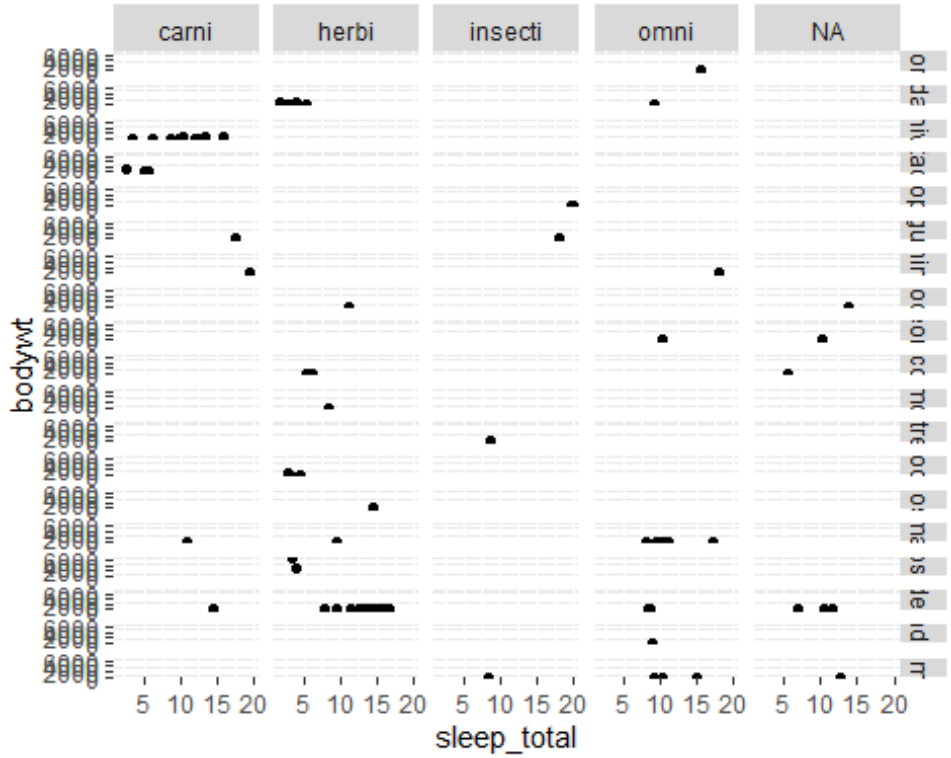


Γράφημα 69

Ερμηνεία του κώδικα: με την εντολή str() διαπιστώνουμε πως το msleep έχει μία ομάδα μεταβλητών τύπου chr που λογίζονται ως κατηγορηματικές και μία άλλη ομάδα τύπου num που είναι αριθμητικές. Συνεπώς για το δεύτερο βήμα της απάντησης επιλέγουμε μία μεταβλητή από την πρώτη ομάδα, έστω την vore.

3.

```
ggplot(data = msleep) +
 geom_point(mapping = aes(x = sleep_total, y = bodywt))+
 facet_grid(order ~ vore)
```



Γράφημα 70

Ερμηνεία του κώδικα: επιλέγουμε τις κατηγορηματικές μεταβλητές order και vore και χρησιμοποιώντας το χαρακτήρα ~ συμπληρώνουμε την παραμετροποίηση της συνάρτησης facet_grid().

4.

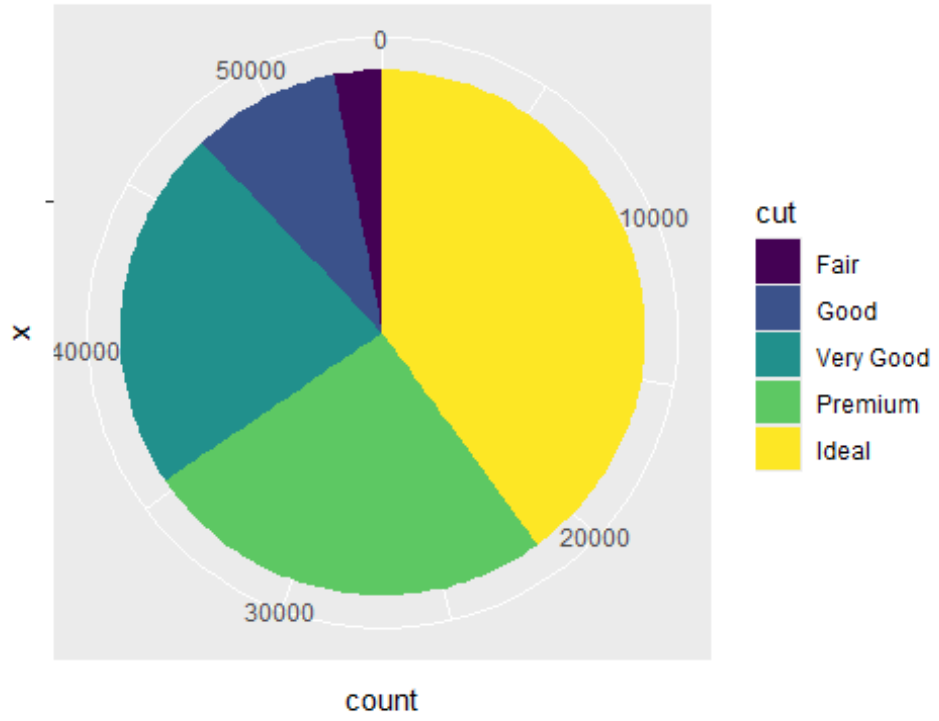
Η ρύθμιση της παραμέτρου se καθορίζει την εμφάνιση ή μη της γκρίζας περιοχής εκατέρωθεν της γραμμής που παράγεται από το γεωμετρικό αντικείμενο geom_smooth.

5.

Εκτελώντας την εντολή βοήθειας ? geom_boxplot θα εμφανιστεί το κείμενο βοήθειας με όλες τις ρυθμίσεις και παραμέτρους του γεωμετρικού αντικειμένου geom_boxplot. Ανάμεσα σε αυτές εντοπίζουμε ότι η προκαθορισμένη τιμή για το position είναι η dodge2.

6.

```
ggplot(data = diamonds) +
     geom_bar(mapping = aes(x="", fill = cut))+coord_polar(theta = "
y")
```



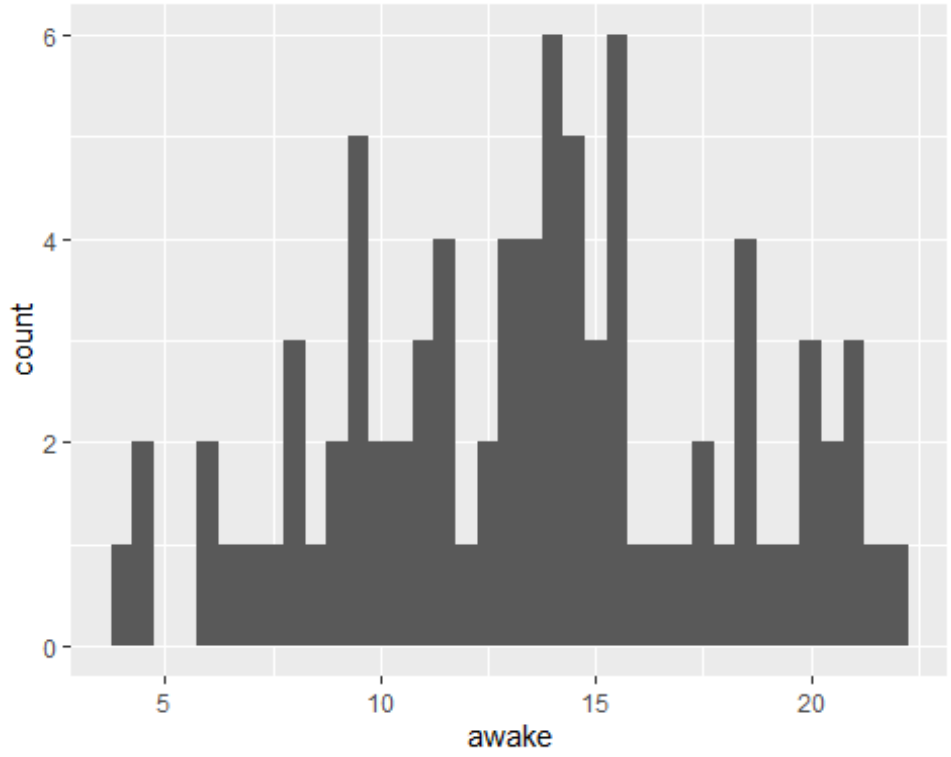
Γράφημα 71

Ερμηνεία του κώδικα: το προηγούμενο τμήμα κώδικα δείχνει πως αρχικά δημιουργείται ένα τυπικό στοιβαγμένο ραβδόγραμμα και μετά προχωρούμε στην αλλαγή του συστήματος συντεταγμένων σε πολικό σύστημα και έτσι σχηματίζεται το πολύ δημοφιλές διάγραμμα πίτας (pie chart). Πρέπει να σημειωθεί πως χρησιμποιήθηκε το data set diamonds που είναι ενσωματωμένο στο πακέτο ggplot2.

7.

Μετά από δοκιμές διαφόρων τιμών για το binwidth καταλήγουμε πως η κατάλληλη τιμή είναι το 0.5.

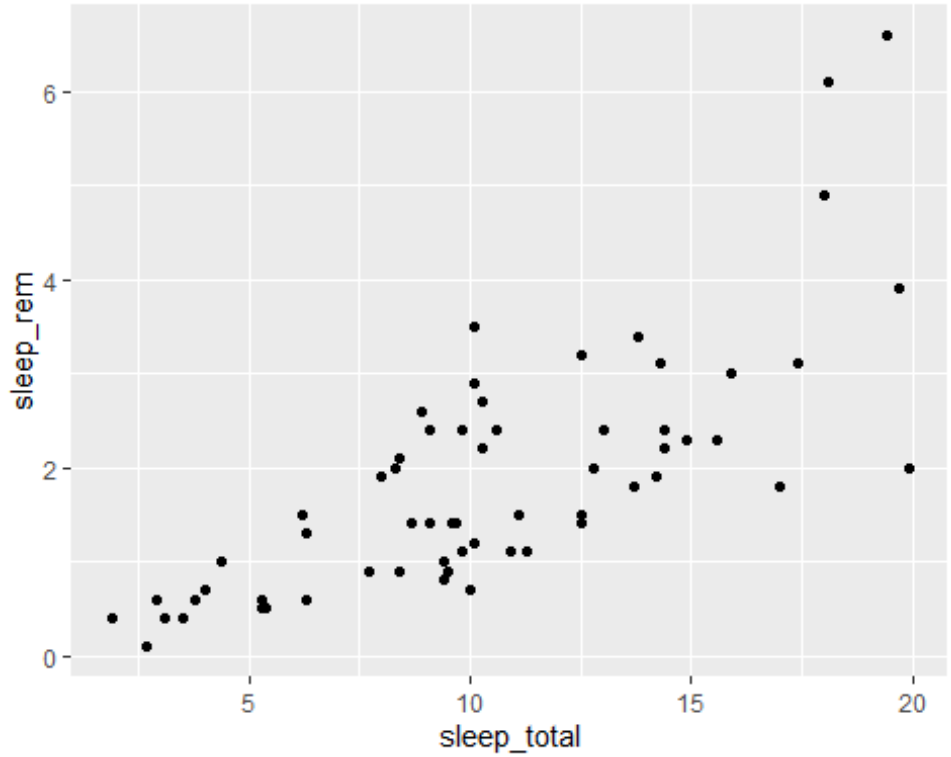
```
ggplot(msleep) + geom_histogram(mapping = aes(x =  awake), binwidth
= 0.5)
```



Γράφημα 72

8.

```
ggplot(data = msleep) +
  geom_point(mapping = aes(x = sleep_total, y = sleep_rem))
```

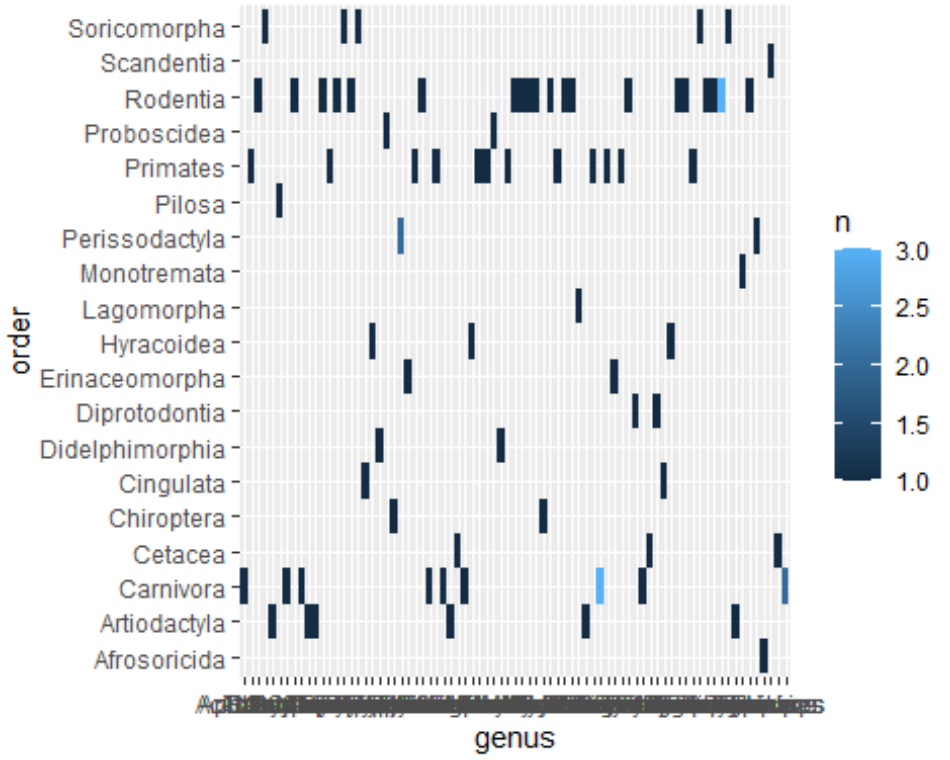


Γράφημα 73

Ερμηνεία του κώδικα: το γράφημα που θα χρησιμοποιήσουμε είναι το διάγραμμα διασποράς που παράγεται με χρήση του γεωμετρικού αντικειμένου geom_point. Η μία ελεγχόμενη μεταβλητή είναι η sleep_total και για την άλλη δοκιμάζουμε όλες τις υπόλοιπες συνεχείς μεταβλητές του msleep εκτός από τη μεταβλητή awake η οποία προκύπτει απλά από την πράξη 24-sleep_total και δεν έχει νόημα να σχηματίσουμε το διάγραμμα διασποράς γιατί θα μας δίνει την τέλεια αρνητική συσχέτιση. Τελικά η επιλογή της δεύτερης μεταβλητής που δίνει μια ρεαλιστική θετική συσχέτιση είναι επιλογή της sleep_rem.

9.

```
msleep %>% count(genus, order) %>% ggplot(mapping = aes(x = genus, y
= order)) + geom_tile(mapping = aes(fill = n))
```

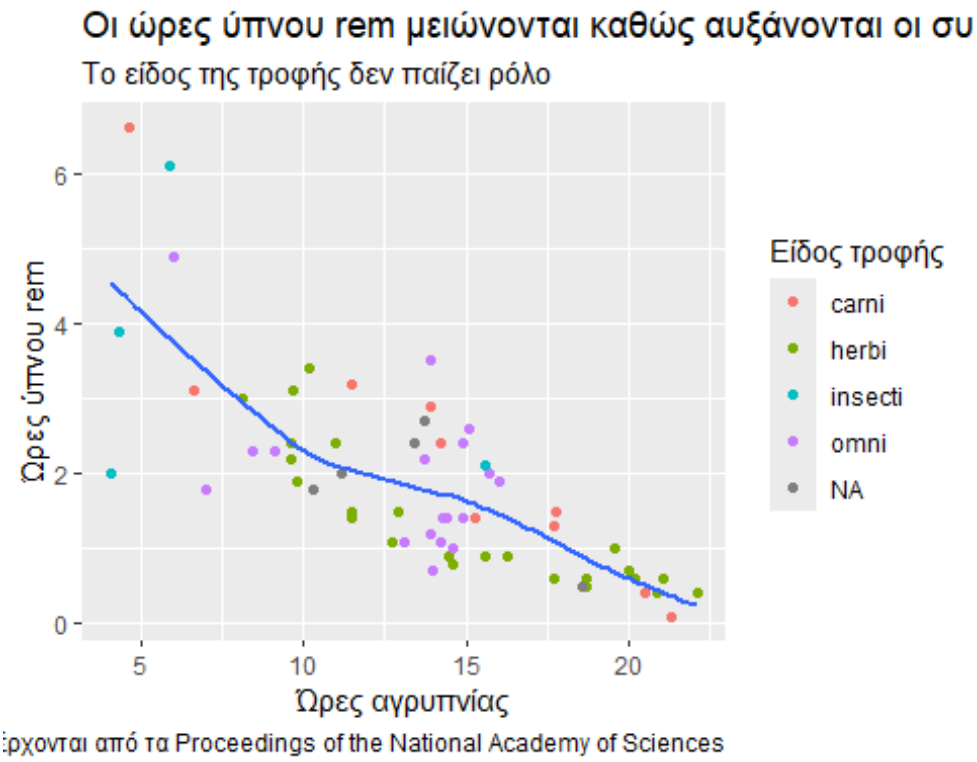


Γράφημα 74

Ερμηνεία του κώδικα: επιλέγουμε ως δύο κατηγορηματικές μεταβλητές τις genus και order και στη συνέχεια δημιουργείται το διάγραμμα το οποίο δεν αποκαλύπτει κάποιο ιδιαίτερο μοτίβο εμφάνισης ζευγαριών τιμών το οποίο να υποδεικνύει την ύπαρξη κάποιας σχέσης.

10.

```
ggplot(msleep, aes(awake, sleep_rem)) + geom_point(aes(color = vore)
) + geom_smooth(se = FALSE) + labs( title = "Οι ώρες ύπνου rem μειών
ονται καθώς αυξάνονται οι συνολικές αγρυπνίας των θηλαστικών", subti
tle = "Το είδος της τροφής δεν παίζει ρόλο", caption = "Τα δεδομένα
προέρχονται από τα Proceedings of the National Academy of Sciences",
x = "Ώρες αγρυπνίας", y = "Ώρες ύπνου rem", colour = "Είδος τροφής")
```



Γράφημα 75

Ερμηνεία του κώδικα: απλώς έχει ρυθμιστεί μία σειρά από τίτλους μέσα από τις κατάλληλες παραμέτρους της συνάρτησης labs().

Κεφάλαιο 6 Ασκήσεις στα γραφήματα

1. Κατασκευάστε (χρησιμοποιώντας το αρχείο με τα δεδομένα Covid19) ένα διάγραμμα γραμμής με τα στοιχεία των ημερήσιων θανάτων 4 χωρών της Ευρωπαϊκής Ένωσης με παρόμοιους πληθυσμούς. Προσθέστε τους κατάλληλους τίτλους και υπότιτλους. Φτιάξτε μια δεύτερη έκδοση με τον αθροιστικό αριθμό θανάτων.

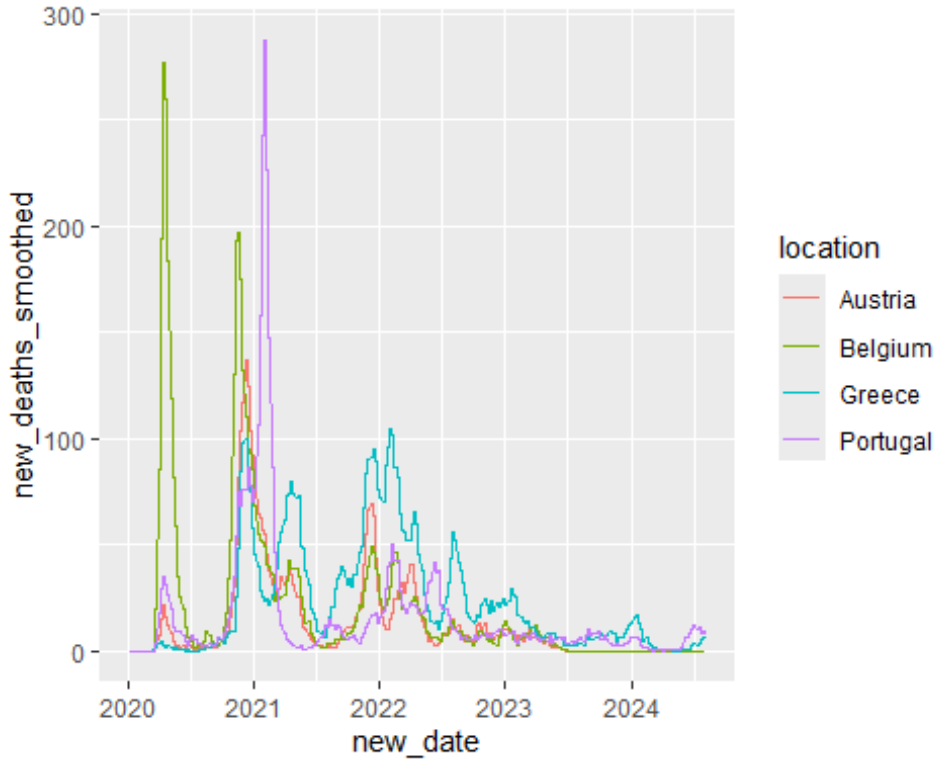
2. Βρείτε από το ίδιο αρχείο δεδομένων δύο μεταβλητές για να κατασκευάσετε ένα διάγραμμα διασκόρπισης που θα ελέγχει την ύπαρξη συνδιακύμανσης στα δεδομένα της Ελλάδας.

3.Βρείτε από το αρχείο δεδομένων μία μεταβλητή για να κατασκευάσετε ένα διάγραμμα τύπου bar. Ρυθμίστε διάφορα aesthetics που αφορούν το συγκεκριμένο διάγραμμα.

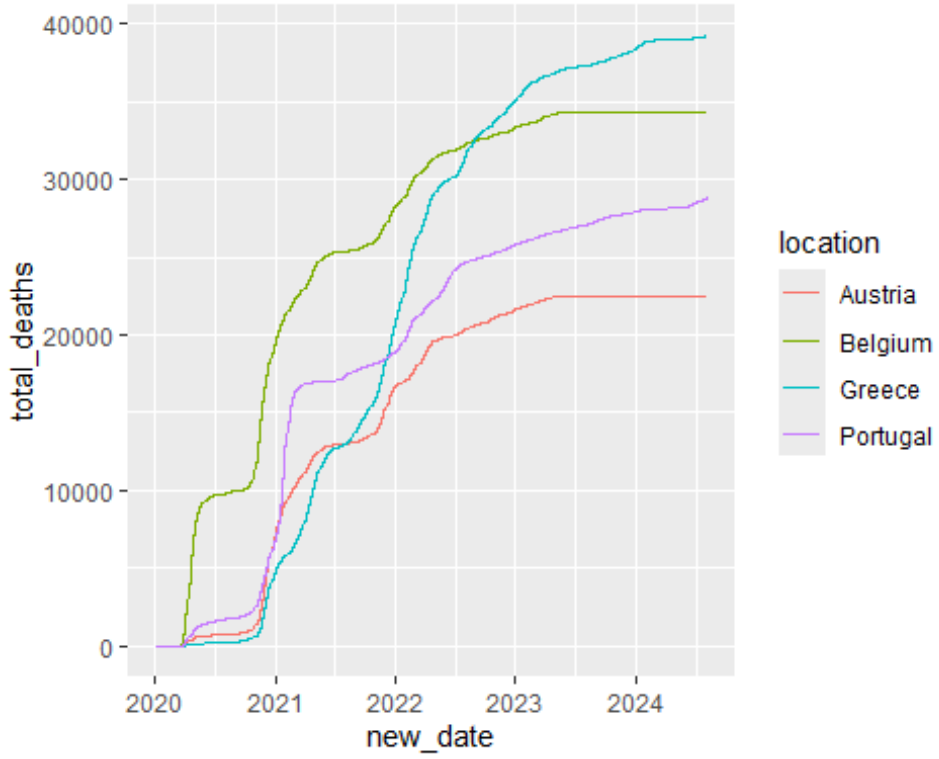
Ενδεικτική Λύση

1.

```
covid19 <- read.csv("https://covid.ourworldindata.org/data/owid-covid-data.
csv", stringsAsFactors = TRUE)
covid19$new_date <- as.Date(covid19$date)
# subsets
covid19_gr <- filter(covid19, location == "Greece")
covid19_four_countries <- filter(covid19, location == "Greece" | location =
= "Austria" | location == "Portugal" | location == "Belgium")
ggplot(data = covid19_four_countries) +
  geom_line(mapping = aes(x = new_date, y = new_deaths_smoothed, color=loca
tion))
```




```
ggplot(data = covid19_four_countries) +
  geom_line(mapping = aes(x = new_date, y = total_deaths, color=location))
```



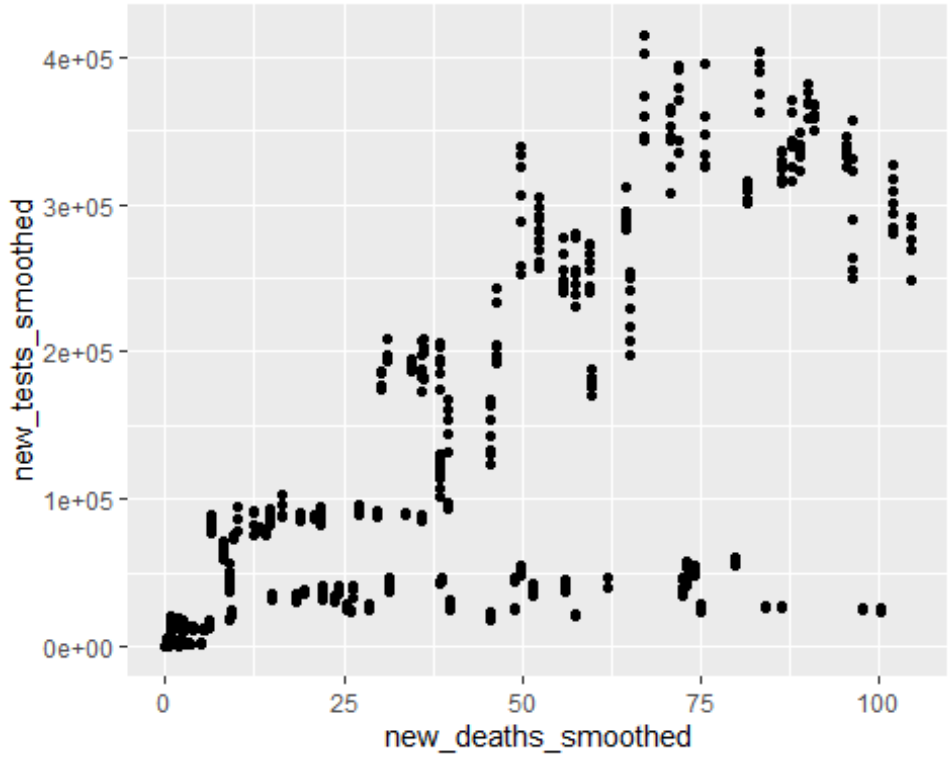
Γράφημα 76

Ερμηνεία του κώδικα: ο πρώτο μέρος του κώδικα εισάγει δεδομένα που σχετίζονται με την πανδημία **COVID19** και στη συνέχεια δημιουργείται ένα υποσύνολο των δεδομένων για την Ευρώπη. Τα δεδομένα είναι διαθέσιμα για χρήση στο σύνδεσμο **https://covid.ourworldindata.org/data/owid-covid-data.csv**. Αρχικά γίνεται κλήση της βιβλιοθήκης **tidyverse** και έπειτα γίνεται με τη **read.csv()** εισαγωγή των δεδομένων ρυθμίζοντας τη μετατροπή των αλφαριθμητικών δεδομένων (**string**) σε δεδομένα τύπου **factor**. Στη συνέχεια με την as.Date() μετατρέπεται η μεταβλητή **covid19$date** που περιέχει τις ημερομηνίες σε μία νέα μεταβλητή που θα έχει τύπο **Date**.

Έπειτα φιλτράρονται τα δεδομένα και δημιουργούνται δύο υποσύνολα τους, ένα για την Ελλάδα που θα χρησιμοποιηθεί στο 2ο ερώτημα και ένα για τις τέσσερις χώρες που έχουν παρόμοιο πλήθος κατοίκων: Ελλάδα, Αυστρία, Πορτογαλία και Βέλγιο. Μετά με το γεωμετρικό αντικείμενο γραμμής δημιουργούνται τα δύο γραφήματα που ζητούνται για τις μεταβλητές **new_deaths_smoothed** και **total_deaths**.

2.

```
ggplot(data = covid19_gr) +
  geom_point(mapping = aes(x = new_deaths_smoothed, y = new_tests_sm
oothed))
```

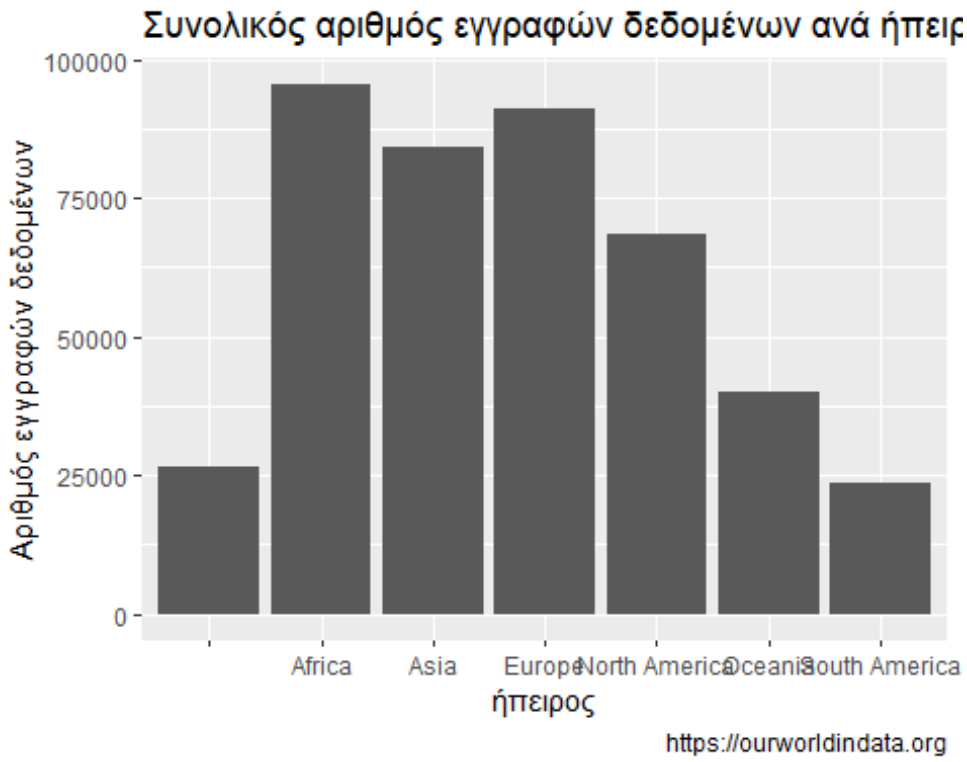


Γράφημα 77

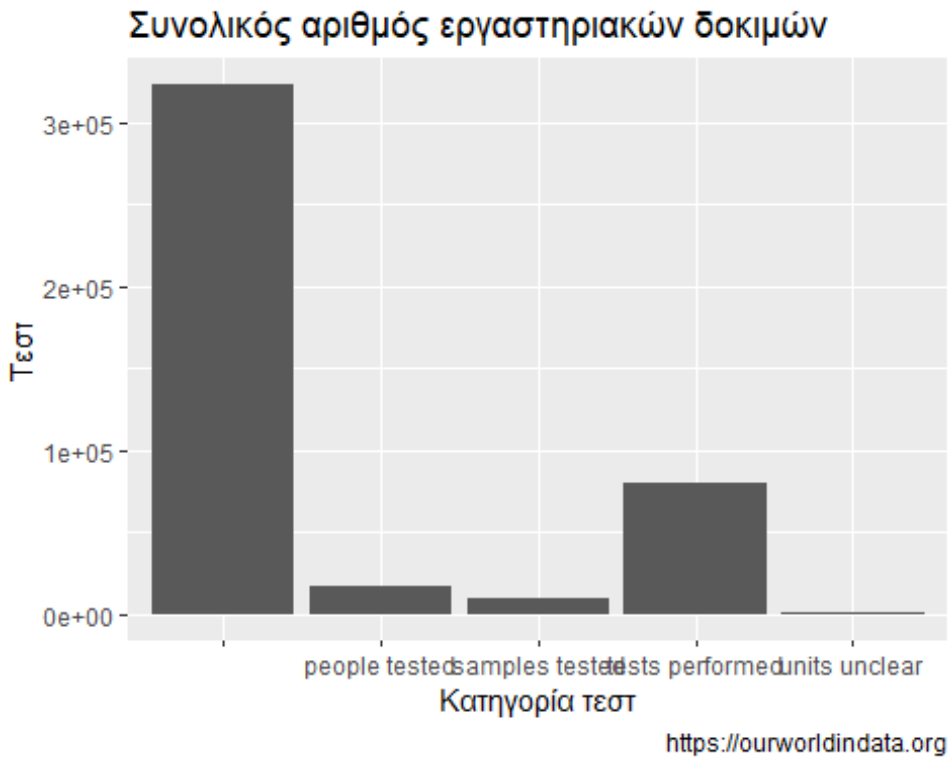
Ερμηνεία του κώδικα: Επιλέγουμε ως συνεχείς μεταβλητές τις new_deaths_smoothed και new_tests_smoothed και στη συνέχεια δημιουργείται ένα διάγραμμα διασκόρπισης με χρήση του geom_point.

3.

```
ggplot(data = covid19) +
  geom_bar(mapping = aes(x = continent))+
  labs(
    title = "Συνολικός αριθμός εγγραφών δεδομένων ανά ήπειρο",
    caption = "https://ourworldindata.org",
    y="Αριθμός εγγραφών δεδομένων",
    x="ήπειρος",
  )
```




```
ggplot(data = covid19) +
  geom_bar(mapping = aes(x = tests_units))+
  labs(
    title = "Συνολικός αριθμός εργαστηριακών δοκιμών",
    caption = "https://ourworldindata.org",
    y="Τεστ",
    x="Κατηγορία τεστ",
  )
```



Γράφημα 78

Ερμηνεία του κώδικα: Δίνονται δύο ενδεικτικά διαγράμματα στα οποία επιλέγουμε δύο διαφορετικές μεταβλητές, την continent και την tests_units που είναι κατηγορηματικές.

6.1 Εργασία (project) με το ggplot2

Χρησιμοποιώντας το dataset owid-covid-data.csv γράψτε τον κατάλληλο κώδικα για τη δημιουργία των διαγραμμάτων σύμφωνα με τις ακόλουθες περιγραφές.

1. Φτιάξτε το κατάλληλο γράφημα που θα εμφανίζει την πορεία (με βάση τον χρόνο) των αθροιστικών κρουσμάτων covid19 στις πέντε μεγαλύτερες χώρες της Ευρώπης. Υπόδειξη: πρέπει πρώτα να περιορίσετε το dataset στα δεδομένα της Ευρώπης και στη συνέχεια αφού βρείτε τις πέντε πιο πολυπληθείς χώρες της να φιλτράρετε τα δεδομένα έτσι ώστε να περιέχουν μόνο τα δικά τους. Μετά θα φτιαχτεί το διάγραμμα το οποίο θα πρέπει να υποστεί βελτίωση στους διάφορους τίτλους του προσθέτοντας τους όπου χρειάζεται. Όλος ο κώδικας της απομόνωσης των ορθών δεδομένων και της δημιουργίας του γραφήματος θα πρέπει να περαστούν στο script. Προαιρετικά μπορείτε να φτιάξετε συνάρτηση που θα παίρνει σαν μοναδικό όρισμα τον αριθμό των πολυπληθέστερων χωρών και θα εκτελεί όλα τα προηγούμενα.

2. Επιλέξτε από το υποσύνολο δεδομένων των πέντε χωρών μια κατηγορηματική και μία συνεχή μεταβλητή και κατασκευάστε το κατάλληλο διάγραμμα που θα αναδεικνύει την ύπαρξη πιθανής συνδιακύμανσης. Ρυθμίστε διάφορα aesthetics που αφορούν το συγκεκριμένο διάγραμμα.

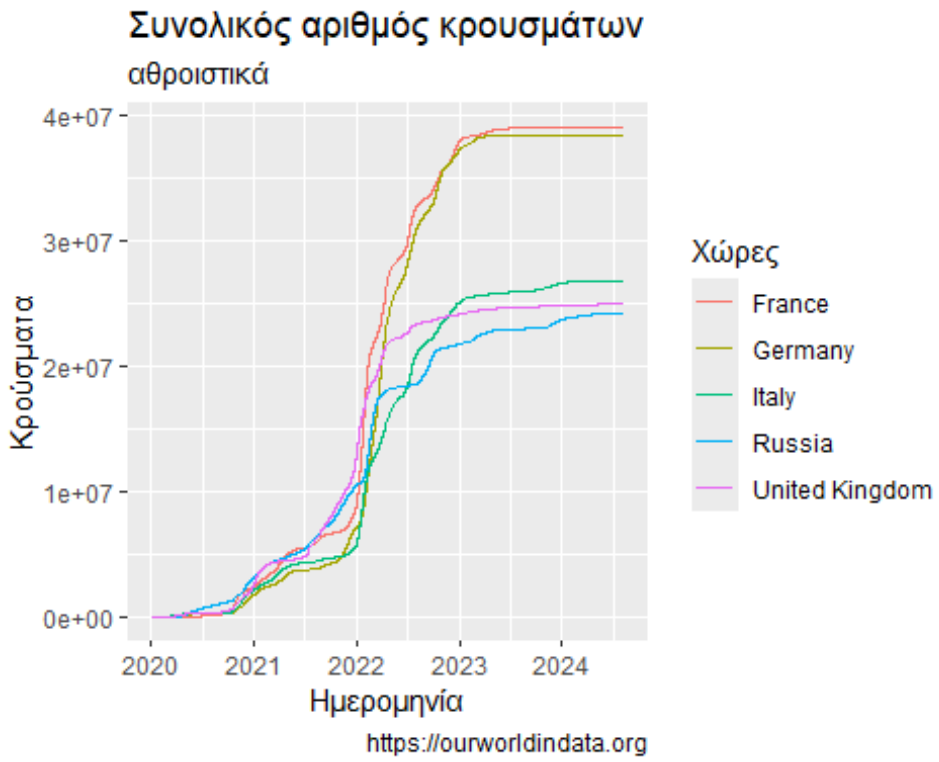
Ενδεικτική Λύση

1.

Πρώτα παρουσιάζεται η ενδεικτική λύση του κώδικα της συνάρτησης myDraw() η οποία υλοποιεί όλες τις απαραίτητες ενέργειες όσον αφορά τα δεδομένα και τη δημιουργία του

γραφήματος και μετά καλείται η συνάρτηση με τιμή παραμέτρου ίση με το 5 (για τις πέντε χώρες). Είναι προφανές πως το ίδιο αποτέλεσμα θα παρουσιαστεί και χωρίς να δώσουμε κάποια τιμή στην παράμετρο της συνάρτησης καθώς έχει ως προκαθορισμένη τιμή το 5.

```
myDraw <- function (num_countries=5){
library(tidyverse)
covid19 <- read.csv("https://covid.ourworldindata.org/data/owid-covi
d-data.csv", stringsAsFactors = TRUE)
covid19$new_date <- as.Date(covid19$date)
covid19_eu <- filter(covid19, continent == "Europe")
group_eu<- group_by(covid19_eu, location)
eu_popul<- summarise(group_eu, POPUL = mean(population, na.rm = TRUE
))

five_big_countries <- head(arrange(eu_popul,desc(eu_popul$POPUL)),nu
m_countries)

covid19_five_big_countries <- filter(covid19, location %in% pull(fiv
e_big_countries,location))

ggplot(data = covid19_five_big_countries) +
  geom_line(mapping = aes(x = new_date, y = total_cases, color=locat
ion))+
  labs(
    title = "Συνολικός αριθμός κρουσμάτων",
    subtitle = "αθροιστικά",
    caption = "https://ourworldindata.org",
    y="Κρούσματα",
    x="Ημερομηνία",
    color="Χώρες"
  )
}
myDraw(5)
```



Γράφημα 79

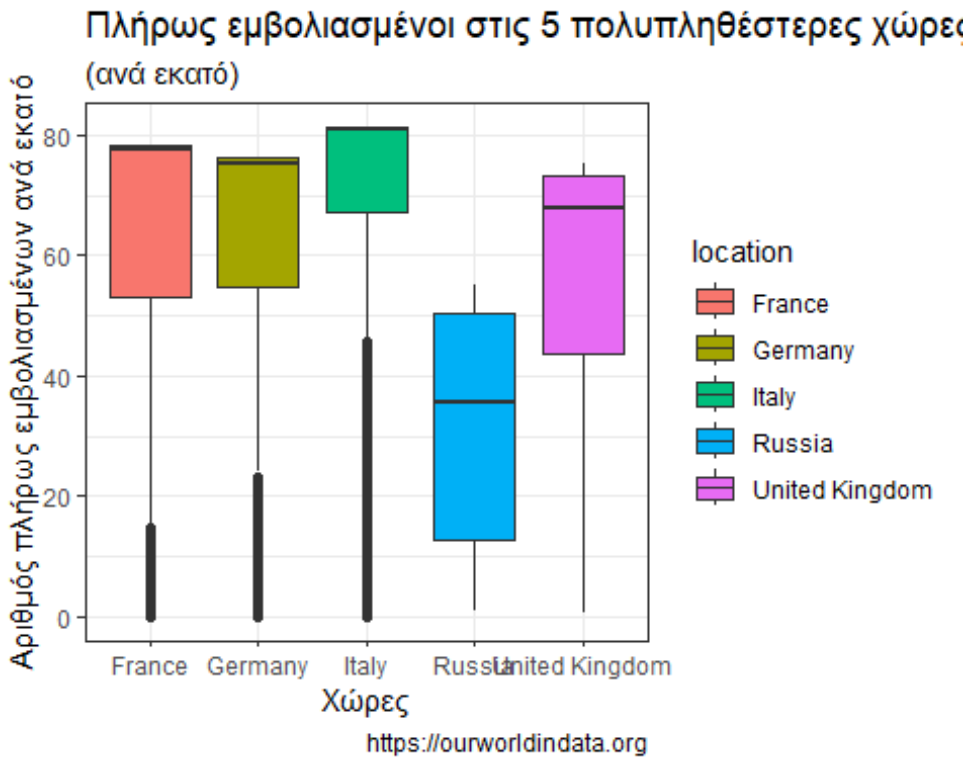
Ερμηνεία του κώδικα: το πρώτο μέρος του κώδικα εισάγει δεδομένα που σχετίζονται με την πανδημία **COVID19** και στη συνέχεια δημιουργείται ένα υποσύνολο των δεδομένων για την Ευρώπη. Τα δεδομένα είναι διαθέσιμα για χρήση στο σύνδεσμο **https://covid.ourworldindata.org/data/owid-covid-data.csv**. Αρχικά γίνεται κλήση της βιβλιοθήκης **tidyverse** και έπειτα γίνεται με τη **read.csv()** εισαγωγή των δεδομένων ρυθμίζοντας τη μετατροπή των αλφαριθμητικών δεδομένων (**string**) σε δεδομένα τύπου **factor**. Στη συνέχεια με την as.Date() μετατρέπεται η μεταβλητή **covid19$date** που περιέχει τις ημερομηνίες σε μία νέα μεταβλητή που θα έχει τύπο **Date**. Η τελευταία γραμμή κώδικα φιλτράρει τα δεδομένα επιλέγοντας αυτά που προέρχονται από την ήπειρο (**continent**) της Ευρώπης.

Έπειτα με κατάλληλη χρήση των συναρτήσεων του πακέτου dplyr απομονώνονται τα δεδομένα των χωρών που επιθυμούμε. Στη συνέχεια δημιουργείται μία ομάδα γραφημάτων γραμμής με κατάλληλη χρήση του γεωμετρικού αντικειμένου γραμμής ενώ ρυθμίζονται οι τίτλοι του γραφήματος.

2.

```
covid19 <- read.csv("https://covid.ourworldindata.org/data/owid-covid-data.
csv", stringsAsFactors = TRUE)
covid19$new_date <- as.Date(covid19$date)
covid19_eu <- filter(covid19, continent == "Europe")
group_eu<- group_by(covid19_eu, location)
eu_popul<- summarise(group_eu, POPUL = mean(population, na.rm = TRUE))

five_big_countries <- head(arrange(eu_popul,desc(eu_popul$POPUL)),5)

covid19_five_big_countries <- filter(covid19, location %in% pull(five_big_c
ountries,location))
```

```
ggplot(
  data = covid19_five_big_countries,
  mapping = aes(x = location, y = people_fully_vaccinated_per_hundred, fill
= location)
) +
  geom_boxplot() + labs(
    title = "Πλήρως εμβολιασμένοι στις 5 πολυπληθέστερες χώρες της Ευρώπης"
,
    y="Αριθμός πλήρως εμβολιασμένων ανά εκατό",
    x="Χώρες",
    subtitle="(ανά εκατό)",
    caption = "https://ourworldindata.org",
)+theme_bw()
```



Γράφημα 80

Ερμηνεία του κώδικα: Για την υλοποίηση της λύσης χρησιμοποιούνται δύο μεταβλητές: η people_fully_vaccinated_per_hundred που είναι συνεχής και η location που είναι κατηγορηματική. Ως γεωμετρικό αντικείμενο χρησιμοποιείται το geom_bar και ρυθμίζονται οι τίτλοι του γραφήματος και των αξόνων ενώ επιλέγεται το ασπρόμαυρο θέμα.

Πηγές γραφημάτων

- https://r4ds.hadley.nz
- https://flowingdata.com/2016/03/22/comparing-ggplot2-and-r-base-graphics
- https://jtr13.github.io/cc21fall2/base-r-vs.-ggplot2-visualization.html
- https://jtr13.github.io/cc20/base-r-vs-ggplot2.html
- https://www.youtube.com/watch?v=-kaZst0gdQ8
- https://emanuelaf.github.io/own-ggplot-theme.html

Κεφάλαιο 7 Συναρτήσεις στην R

7.1. Εισαγωγή

Η R είναι μια γλώσσα προγραμματισμού ειδικού σκοπού, με κύριο αντικείμενο την ανάλυση δεδομένων. Εκτός όμως από την χρήση της ως στατιστικό εργαλείο, ένα άλλο πλεονέκτημα της είναι ότι μας επιτρέπει να αυτοματοποιήσουμε τις εργασίες σας. Αυτό επιτυγχάνεται με τη χρήση των συναρτήσεων.

Η R παρέχει αρκετές build-in (δικές της) συναρτήσεις για να διευκολύνουν την εργασία σας, αλλά όπως κάθε γλώσσα προγραμματισμού σας δίνει την δυνατότητα να αναπτύξετε και δικές σας συναρτήσεις, τις οποίες μπορείτε να χρησιμοποιήσετε για αυτοματοποίηση συνθέτων διαδικασιών. Στο κεφάλαιο αυτό θα μάθετε πως να κατασκευάζετε τις δικές σας συναρτήσεις στην R.

7.2 Σενάρια (scripts) και Συναρτήσεις (functions)

Ως τώρα έχετε μάθει να κατασκευάζετε προγράμματα που περιέχουν μία σειρά εντολών, οι ποιες εκτελούνται διαδοχικά. Αυτές οι λίστες με διαδοχικές εντολές ονομάζονται σενάρια (scripts). Στα πλαίσια αυτού του κεφαλαίου θα μάθετε πώς να μετατρέπετε σενάρια σε συναρτήσεις. Η μετάβαση από ένα σενάριο (script) της R (δηλαδή μία σειρά από διαδοχικές εντολές) σε μια συνάρτηση - όπως θα δούμε- δεν απαιτεί ιδιαίτερη προσπάθεια.

Μια συνάρτηση είναι ουσιαστικά ένα κομμάτι (μπλοκ) κώδικα που εκτελείται διαδοχικά και χωρίς διακοπή. Με αυτόν τον τρόπο, μια συνάρτηση δεν διαφέρει πολύ από ένα σενάριο (script) της R.

Ωστόσο, οι συναρτήσεις έχουν δύο πολύ σημαντικά πλεονεκτήματα σε σχέση με τα σενάρια:

- Η συνάρτηση μπορεί να έχει μεταβλητή(ες) εισόδου, επομένως μπορείτε να τη χρησιμοποιείτε με διαφορετικά δεδομένα.
- Η συνάρτηση σάς επιστρέφει (ως έξοδο) ένα αντικείμενο, ώστε να μπορείτε να εργαστείτε με το αποτέλεσμα αυτής της συνάρτησης.

7.3. Δημιουργία του σεναρίου

Για να μπορέσουμε να εντρυφήσουμε στις συναρτήσεις, θα εστιάσουμε σε ένα παράδειγμα.

Ας υποθέσουμε ότι έχετε υπολογίσει τις σχετικές συχνότητες για μία κατηγορική μεταβλητή και θέλετε να παρουσιάσετε σχετικές συχνότητες (για παράδειγμα, 0.0412) σε μορφή ποσοστών, με ένα δεκαδικό ψηφίο και με το σύμβολο **%** προκειμένου να συμπεριλάβετε τα ποσοστά σε μία αναφορά. Για να το πετύχετε αυτό θα πρέπει να κάνετε τις εξής ενέργειες:

- **Πολλαπλασιάστε τις σχετικές συχνότητες με το 100**.
- **Στρογγυλοποιήστε το αποτέλεσμα σε ένα δεκαδικό ψηφίο.** Αυτό μπορείτε να το επιτύχετε χρησιμοποιώντας τη συνάρτηση round().

- **Επικολλήστε το σύμβολο του ποσοστού μετά τον αριθμό.** Η συνάρτηση paste() είναι στη διάθεσή σας για αυτή την εργασία
- **Εκτυπώστε το αποτέλεσμα.** Η συνάρτηση print() μπορεί να το κάνει αυτό.

Αναλυτικά, οι εντολές της R για να επιτύχετε το σκοπό σας είναι:

```
x <- c(0.4583, 0.2567, 0.1345, 0.15050)

percent1<- x * 100

percent2 <- round(percent1, digits = 1)

result <- paste(percent2, "%", sep = "")

print(result)
[1] "45.8%" "25.7%" "13.5%" "15%"
```

Το παραπάνω μπλοκ κώδικα από διαδοχικές εντολές μπορείτε να το αποθηκεύσετε ως αρχείο script- για παράδειγμα, *createPercent.R*, και μπορείτε να καλέσετε αυτό το σενάριο οποιαδήποτε στιγμή με την ακόλουθη εντολή:

source ("createPercent.R")

Το παραπάνω σενάριο θα λειτουργεί, εφόσον θέλετε να δείτε τις ίδιες τέσσερις σχετικές συχνότητες, κάθε φορά που καλείτε το σενάριο. Για οποιαδήποτε άλλα δεδομένα θα πρέπει να αλλάξετε το σενάριο και να το προσαρμόσετε στα νέα δεδομένα. Ένας τρόπος για να αποφύγουμε αυτή την αλλαγή είναι να μετατρέψουμε το σενάριο σε μία συνάρτηση της R.

7.4. Μετατρέποντάς το σενάριο σε συνάρτηση

Για να μπορείτε να χρησιμοποιείτε τον κώδικά σας με διαφορετικούς αριθμούς (σχετικές συχνότητες) κάθε φορά, χωρίς να χρειάζεται να αλλάζετε τον κώδικα θα πρέπει να μετατρέψετε αυτό το σενάριο σε συνάρτηση, δηλαδή

```
createPercent1<- function(x)
{
percent1<- x * 100
percent2 <- round(percent1, digits = 1)
result <- paste(percent2, "%", sep = "")
return(result)
}
```

Σημείωση: στο κεφάλαιο αυτό, θα αναφερόμαστε στη συνάντηση με το όνομα createPerecent. Ο αριθμός μετά το όνομα createPerecent αφορά μία διαφορετική έκδοση της συνάρτησης και χρησιμοποιείται προς διευκόλυνση των αναγνωστών.

7.5. Τα δομικά στοιχεία της συνάρτησης στην R

Η συνάρτηση στην R έχει τα ακόλουθα δομικά στοιχεία:

- Η λέξη-κλειδί **function** πρέπει πάντα να ακολουθείται από παρενθέσεις. Αυτό ενημερώνει την R ότι αυτό που ακολουθεί είναι μια συνάρτηση.
- Οι παρενθέσεις μετά τη λέξη **function** αποτελούν την είσοδο, ή τη λίστα ορισμάτων, της συνάρτησης. Στη συγκεκριμένη συνάρτηση υπάρχει μόνο ένα όρισμα, το διάνυσμα x.
- Τα άγκιστρα, **{}**, μπορούν να θεωρηθούν ως τα όρια (δηλαδή, η αρχή και το τέλος) της συνάρτησης. Όλες οι εντολές μεταξύ των δύο αγκίστρων είναι μέρος του κώδικά ή του σώματος της συνάρτησης
- Η εντολή **return()** είναι η έξοδος της συνάρτησης. Το αντικείμενο που βάζετε ανάμεσα στις παρενθέσεις της εντολής **return**, επιστρέφεται από το εσωτερικό της συνάρτησης στο χώρο εργασίας σας στην R.
  - Προσοχή: Μπορείτε να βάλετε μόνο ένα αντικείμενο ανάμεσα στις παρενθέσεις της εντολής **return,** άρα μόνο ένα αντικείμενο μπορεί να επιστρέψει η συνάντηση στον κύριο χώρο εργασίας σας στην R.

7.6. Καλώντας την συνάρτηση

Αρχικά θα πρέπει να «εκτελέσετε/τρέξετε» τον κώδικά της συνάρτησης, επιλέγοντας την πρώτη γραμμή της συνάρτησης (ή όλες τις γραμμές της συνάρτησης) και πατώντας το κουμπί Run του RStudio (θυμηθείτε ότι, με την εντολή *ls()* βλέπετε όλα τα αντικείμενα του χώρου εργασίας σας στην R καθώς και τις συναρτήσεις που έχετε ενεργές). Τώρα μπορείτε να χρησιμοποιήσετε την συνάρτηση με οποιοδήποτε διάνυσμα αριθμών

```
new.numbers <- c(0.4583, 0.2567, 0.1345, 0.15050)
createPercent1(new.numbers)
```

```
[1] "45.8%" "25.7%" "13.5%" "15%"
```

7.7. Αντίγραφα συναρτήσεων

Επειδή μια συνάρτηση που κατασκευάσατε είναι απλώς ένα άλλο αντικείμενο στην R, μπορείτε να την χειριστείτε με τον ίδιο τρόπο όπως χειρίζεστε τα άλλα αντικείμενα. Για παράδειγμα, μπορείτε να εκχωρήσετε τη συνάρτηση σε ένα νέο αντικείμενο και με αυτό τον τρόπο να την αντιγράψετε, ως εξής:

```
copyPercent <- createPercent1
```

Τώρα το *copyPercent()* είναι επίσης μια συνάρτηση που κάνει ακριβώς τα ίδια πράγματα με την *createPercent1()*. Σημειώστε ότι όταν αντιγράφετε μία συνάρτηση δεν βάζετε παρενθέσεις μετά το όνομα της συνάρτησης, εδώ το *createPercent1()*. Εάν προσθέσετε τις παρενθέσεις, τότε *καλείτε* τη συνάρτηση και στο αντικείμενο *copyPercent()* θα τοποθετηθεί το αποτέλεσμα της συνάρτησης createPercent1. Εάν δεν προσθέσετε τις παρενθέσεις, τότε αναφέρεστε στη συνάρτηση χωρίς να την καλέσετε.

Επίσης, μπορείτε να εκτυπώσετε το περιεχόμενο μιας συνάρτησης στην R, απλά πληκτρολογώντας το όνομά της, π.χ. :

```
copyPercent
```

```
function(x)
{
percent1<- x * 100
percent2 <- round(percent1, digits = 1)
result <- paste(percent2, "%", sep = "")
return(result)
}
```

7.8. Μειώνοντας τον αριθμό των γραμμών μιας συνάρτησης

Δεν είναι όλα τα δομικά στοιχεία μιας συνάρτησης υποχρεωτικά. Στην πραγματικότητα, η δήλωση/εντολή *return()* είναι προαιρετική, γιατί, από προεπιλογή, η R επιστρέφει πάντα την τιμή της τελευταίας γραμμής κώδικα της συνάρτησης στο χώρο εργασίας.

Αναλυτικά, στην περίπτωση που δεν βάλετε την εντολή return(), τότε η συνάρτηση θα επιστρέψει το αντικείμενο της τελευταίας γραμμής κώδικα αλλά μόνο με εκχώρηση σε άλλο αντικείμενο. Δοκιμάστε την παρακάτω έκδοση της συνάρτησης που δεν περιέχει την εντολή return() και χρησιμοποιήστε την εντολή εκτύπωσης print() για να καλέσετε την συνάρτηση:

```
createPercent2<- function(x)
{
  percent1<- x * 100
  percent2 <- round(percent1, digits = 1)
  paste(percent2, "%", sep = "")
}
print( createPercent2(new.numbers) )
```

```
[1] "45.8%" "25.7%" "13.5%" "15%"
```

Μπορεί να φαίνεται ότι η εντολή return() δεν είναι τελικά χρήσιμη, αλλά την χρειάζεστε εάν θέλετε να βγείτε από τη συνάρτηση πριν από το τέλος του κώδικα στο κυρίως σώμα του προγράμματος σας. Για παράδειγμα, μπορείτε να προσθέσετε μια γραμμή στην συνάρτηση που ελέγχει εάν το x είναι αριθμητικό και, αν όχι, να επιστρέφει NULL, ως εξής:

```
createPercent2a <- function(x){
  if(!is.numeric(x)) return(NULL)
  percent <- round(x * 100, digits = 1)
  paste(percent, "%", sep = "")
}
```

Κατασκεύασε τώρα ένα αλφαριθμητικό διάνυσμα c1 και καλέστε την συνάρτηση:

```
c1=c("low")

createPercent2a(c1)
```

```
NULL
```

7.9. Αφαιρώντας τα άγκιστρα

Τα άγκιστρα, {}, δηλώνουν τα όρια της συνάρτησης, δηλαδή είναι σαν τους τοίχους γύρω από το σώμα (μπλοκ) εντολών της συνάρτησης. Σε μερικές περιπτώσεις όμως δεν είναι υποχρεωτικά. Αυτό συμβαίνει, εάν μια συνάρτηση αποτελείται από μία μόνο γραμμή κώδικα. Τότε μπορείτε απλώς να προσθέσετε αυτήν τη γραμμή μετά τη λίστα ορισμάτων της συνάρτησης, χωρίς να την περικλείεται με άγκιστρα. Σε αυτή την περίπτωση, η R θα θεωρήσει την μοναδική γραμμή κώδικα μετά τη λίστα των ορισμάτων ως το σώμα της συνάρτησης. Θα μπορούσατε να κάνετε το ίδιο με τη συνάρτηση createPercent ως εξής:

```
createPercent3 <- function(x) paste(round(x * 100, digits = 1), "%",
sep = "")
```

7.10. Τα ορίσματα της συνάρτησης

Στην ενότητα αυτή θα παρουσιάσουμε κάποιες πληροφορίες σχετικά με τα ορίσματα μιας συνάρτησης.

- Τα ορίσματα έχουν πάντα όνομα όταν ορίζετε τη συνάρτηση (στις παρενθέσεις μετά την εντολή function). Αλλά όταν καλείτε τη συνάρτηση, δεν χρειάζεται να δώσετε το όνομα του ορίσματος με το οποίο εμφανίζεται στη λίστα ορισμάτων της συνάρτησης.
- Τα ορίσματα μπορεί να είναι προαιρετικά, οπότε θα μπορούσε μία συνάρτηση να μην έχει καθόλου ορίσματα.
- Τα ορίσματα μπορεί να έχουν μια προεπιλεγμένη τιμή, η οποία χρησιμοποιείται εάν δεν δώσετε τιμή όταν καλείτε την συνάρτηση.

7.11. Προσθήκη περισσότερων ορισμάτων

Η συνάρτηση createPercent3() που κατασκευάσαμε, είναι πρακτική αν θέλουμε να μετατρέψουμε σχετικές συχνότητες σε ποσοστά, αλλά ας υποθέσουμε ότι θα θέλαμε η συνάντηση μας να μπορεί να χειρίζεται και διανύσματα που περιέχουν αριθμούς που είναι ήδη ποσοστά. Για παράδειγμα, έστω ότι έχετε το παρακάτω διάνυσμα ποσοστών:

```
pososta1 <- c (25.23, 54.4, 20.37)
```

Αν θέλετε να χρησιμοποιήσετε την συνάντηση createPercent3() για να επικολλήσετε το σύμβολο του ποσοστού τότε θα πρέπει το διάνυσμα αυτό (pososta1) να το διαιρέσετε με το 100, για να λάβετε το σωστό αποτέλεσμα:

```
pososta1 <- c(25.23, 54.4, 20.37)

createPercent3 (pososta1/100)

[1] "25.2%" "54.4%" "20.4%"
```

Αυτό μπορείτε να το αποφύγετε προσθέτοντας ένα επιπλέον όρισμα στη συνάρτηση που θα ελέγχει τον παράγοντα του πολλαπλασιασμού, πχ το όρισμα mult, όπως παρακάτω:

```
createPercent4 <- function(x, mult){
  percent <- round(x * mult, digits = 1)
  paste(percent, "%", sep = "")
}
```

7.12. Προσθήκη προεπιλεγμένης τιμής

Η προσθήκη ενός πρόσθετου ορίσματος σάς δίνει περισσότερο έλεγχο στη συνάρτηση, αλλά επίσης εισάγει ένα νέο πρόβλημα. Εάν δεν καθορίσετε τιμή για το όρισμα mult όταν καλείται την συνάρτηση createPercent4(), τότε θα έχετε το ακόλουθο αποτέλεσμα:

```
createPercent4(pososta1)

Error in createPercent4(pososta1): argument "mult" is missing, with
no default
```

Αυτό συμβαίνει επειδή δεν καθορίσατε το όρισμα mult όταν καλέσατε την συνάρτηση και η R δεν έχει κανέναν τρόπο να γνωρίζει με ποιον αριθμό θέλετε να πολλαπλασιάσετε x, οπότε σταματά την εκτέλεση της συνάρτησης και σας αναφέρει ότι χρειάζεται περισσότερες πληροφορίες.

Για να αποφύγετε αυτό το πρόβλημα, δηλαδή να χρειάζεται να δώσετε τιμή για κάθε ένα από τα ορίσματα της συνάρτησης σας, η R σας δίνει τη δυνατότητα να καθορίσετε προεπιλεγμένες τιμές για τα ορίσματα που σας ενδιαφέρουν.

Στην R, μπορείτε να ορίσετε προεπιλεγμένες τιμές για οποιοδήποτε όρισμα στη λίστα ορισμάτων προσθέτοντας το σύμβολο = και την προεπιλεγμένη τιμή μετά το αντίστοιχο όρισμα.

Στη συνάντηση μας θα χρησιμοποιήσουμε προεπιλεγμένη τιμή το 100 για το όρισμα mult και ο κώδικας θα πάρει την εξής μορφή:

```
createPercent5 <- function(x, mult = 100){
  percent <- round(x * mult, digits = 1)
  paste(percent, "%", sep = "")
}
```

Δοκιμάστε τώρα τις παρακάτω εντολές για να δείτε πώς λειτουργεί η προεπιλεγμένη τιμή στα ορίσματα

```
createPercent5(new.numbers)

[1] "45.8%" "25.7%" "13.5%" "15%"

createPercent5(new.numbers, 1)

[1] "0.5%" "0.3%" "0.1%" "0.2%"

createPercent5(pososta1, 1)

[1] "25.2%" "54.4%" "20.4%"
```

Άρα, όταν κάνουμε την συνάρτηση και δεν δίνουμε τιμή για το όρισμα mult, η συνάντηση εκτελείται θεωρώντας ότι το όρισμα mult έχει τιμή 100 (δηλαδή την προεπιλεγμένη τιμή.

7.13. Το όρισμα dots

Η συνάρτηση createPercent5() στρογγυλοποιεί κάθε ποσοστό σε ένα δεκαδικό ψηφίο. Θα είχε ενδιαφέρον να προσθέσετε ένα άλλο όρισμα για να καθορίσετε τον αριθμό των ψηφίων που χρησιμοποιούνται στη στρογγυλοποίηση και με τον τρόπο αυτό να κάνετε πιο γενική τη συνάντηση σας και κατά συνέπεια χρήσιμη σε περισσότερες περιπτώσεις.

Στην παρακάτω συνάρτηση, προσθέτουμε ένα επιπλέον όρισμα το dekadika (με προεπιλεγμένη τιμή το 1) για να καθορίσουμε τον αριθμό των δεκαδικών στην στρογγυλοποίηση των ποσοστών.

```
createPercent6 <- function(x, mult = 100, dekadika = 1){
  percent <- round(x * mult, digits = dekadika)
  paste(percent, "%", sep = "")
}
```

Σημειώστε ότι το συγκεκριμένο όρισμα (dekadika) της συνάρτησης createPercent6() αφορά το όρισμα της συνάρτησης round που χρησιμοποιείται μέσα στην δική μας συνάντηση. Στην συνάρτηση round, όμως, το όρισμα των δεκαδικών έχει το όνομα digits. Για αυτές περιπτώσεις, η R έχει μια έξυπνη λύση, το όρισμα dots (...) .

Για να μεταβιβάσετε οποιοδήποτε όρισμα στη συνάρτηση round() μέσα στο σώμα της createPercent6(), προσαρμόστε τον κωδικό ως εξής:

```
createPercent7 <- function(x, mult = 100, ...){
  percent <- round(x * mult, ...)
  paste(percent, "%", sep = "")
}
```

Τώρα μπορείτε να καθορίσετε το όρισμα ψηφίων για το *round()* στην *createPercent7()* ως εξής:

```
createPercent7(new.numbers, digits = 2)

[1] "45.83%" "25.67%" "13.45%" "15.05%"

createPercent7(new.numbers)

[1] "46%" "26%" "13%" "15%"
```

Δηλαδή, όταν καλείται την συνάρτηση, ότι γράφετε στις τρεις τελείες (στα dots) θα μεταφερθεί ως όρισμα στην συνάρτηση *round()*. Με αυτό τον τρόπο αποφεύγετε την χρήση διαφορετικών ονομάτων για ορίσματα συναρτήσεων που περιλαμβάνονται στον κώδικα την συνάρτησή σας.

7.14. Χρήση συναρτήσεων ως ορίσματα

Η R σας δίνει, επίσης, την δυνατότητα να μπορείτε να μεταβιβάσετε μια συνάρτηση ως όρισμα σε μία άλλη και αυτό σας επιτρέπει να δημιουργείτε συναρτήσεις που να είναι γενικές και να μπορούν να χρησιμοποιηθούν σε περισσότερες περιπτώσεις.

Στην R, μπορείτε να χρησιμοποιήσετε δύο συναρτήσεις αναφορικά με την στρογγυλοποίηση αριθμών: α) την *round()* που στρογγυλοποιεί ως προς τον αριθμό των δεκαδικών και β) την *signif()* που στρογγυλοποιεί ως προς τον αριθμό των συνολικών ψηφίων του αριθμού.

Η συνάρτηση createPercent7() χρησιμοποιεί την συνάρτηση round() για στρογγυλοποίηση των ποσοστών, αλλά μπορεί να θέλετε κάποιες φορές να χρησιμοποιήσετε και την επιλογή

στρογγυλοποίησης ως προς τον αριθμό των ψηφίων του αριθμού, δηλαδή την συνάρτηση signif().

Σε αυτή την περίπτωση, θα μπορούσατε να γράψετε μια δεύτερη συνάρτηση για να στρογγυλοποιεί σε συγκεκριμένο αριθμό ψηφίων ή μπορείτε απλώς να προσαρμόσετε το createPercent7() με τέτοιο τρόπο ώστε να δέχεται ως όρισμα τη συνάρτηση (round ή signif) που θέλετε να χρησιμοποιήσετε.

Αναλυτικά, μπορείτε να προσθέσετε ένα επιπλέον όρισμα στη λίστα ορισμάτων της συνάρτησή σας - σε αυτήν την περίπτωση, το FUN - και στη συνέχεια μπορείτε να χρησιμοποιήσετε το όνομα αυτού του ορίσματος ως συνάρτηση. Ο καθορισμός προεπιλεγμένης τιμής σε αυτή την περίπτωση, λειτουργεί ακριβώς όπως και με τα άλλα ορίσματα. Δηλαδή, απλώς καθορίστε την προεπιλεγμένη τιμή μετά το σύμβολο =, σε αυτήν την περίπτωση τη συνάρτηση round().

```
createPercent8 <- function(x, mult = 100, FUN = round, ...){
  percent <- FUN(x * mult, ...)
  paste(percent, "%", sep = "")
}
```

Εάν θέλετε τώρα να χρησιμοποιήσετε την συνάρτηση signif() για στρογγυλοποίηση των αριθμών σε τρία ψηφία, μπορείτε εύκολα να το κάνετε χρησιμοποιώντας την ακόλουθη εντολή:

```
createPercent8(new.numbers, FUN = signif, digits = 3)

[1] "45.8%" "25.7%" "13.4%" "15%"
```

Με την παραπάνω εντολή εκτελούνται οι παρακάτω διαδικασίες:

- Όπως και πριν, η R παίρνει το διάνυσμα new.numbers και το πολλαπλασιάζει με 100, γιατί αυτή είναι η προεπιλεγμένη τιμή για το mult.
- Η R εκχωρεί τον κώδικα της συνάρτησης signif() στο FUN, οπότε τώρα το FUN() είναι αντίγραφο της signif() και λειτουργεί ακριβώς με τον ίδιο τρόπο.
- Η R παίρνει το όρισμα digits και το περνάει στο FUN().

Σημειώστε την απουσία παρενθέσεων στην εκχώρηση συνάρτησης ως όρισμα. Εάν προσθέτατε παρενθέσεις σε εκείνο το σημείο, τότε θα αναθέσετε στο FUN το αποτέλεσμα μιας κλήσης της signif(), αντί της ίδιας της συνάρτησης. Στην περίπτωση αυτή, η R θα ερμήνευε το signif() ως μία ένθετη λειτουργία. Επιπλέον, η R θα έβγαζε σφάλμα γιατί, σε αυτήν την περίπτωση, καλείτε την signif() χωρίς ορίσματα και η R δεν το δέχεται αυτό.

7.15. Καθολικές και τοπικές μεταβλητές

Όπως σε όλες τις γλώσσες προγραμματισμού, έτσι και στην R, έχουμε καθολικές και τοπικές μεταβλητές. Αναλυτικά, οι μεταβλητές που δημιουργούνται **εκτός** μιας συνάρτησης είναι γνωστές ως **καθολικές** (global) μεταβλητές, ενώ οι μεταβλητές που δημιουργούνται **εντός** μιας συνάρτησης ονομάζονται **τοπικές** (local) μεταβλητές.

Οι καθολικές μεταβλητές μπορούν να χρησιμοποιηθούν από όλους, τόσο εντός των συναρτήσεων όσο και εκτός. Για να μελετήσουμε τις καθολικές και τοπικές μεταβλητές θα δούμε μια σειρά από παραδείγματα.

Παράδειγμα 1: Δημιουργήστε μια μεταβλητή έξω από μια συνάρτηση και στην συνέχεια χρησιμοποιήστε την μέσα στη συνάρτηση.

```
txt <-"awesome"
my_function <- function()
{paste("R is",txt)}

my_function()
```

```
[1] "R is awesome"
```

Όπως βλέπετε, μπορούμε να χρησιμοποιήσουμε μία καθολική μεταβλητή εντός μιας συνάρτησης.

Παράδειγμα 2: Δημιουργήστε μια μεταβλητή μέσα σε μια συνάρτηση που να έχει το ίδιο όνομα με μία υπάρχουσα καθολική μεταβλητή

Εάν δημιουργήσετε μια μεταβλητή με το ίδιο όνομα μέσα σε μια συνάρτηση, αυτή η μεταβλητή θα είναι τοπική και μπορεί να χρησιμοποιηθεί μόνο μέσα στη συνάρτηση. Αν υπάρχει και καθολική μεταβλητή με το ίδιο όνομα, τότε αυτή θα παραμείνει ως είχε, καθολική και με την αρχική τιμή (την τιμή που είχε πριν καλεστεί η συνάρτηση), εκτός από τη στιγμή που εκτελείται η συνάντηση.

```
txt <- "global variable"

my_function <- function() {
  txt = "fantastic"
  paste("R is", txt)
}

my_function()
```

```
[1] "R is fantastic"
```

```
print(txt)
```

```
[1] "global variable"
```

Εάν προσπαθήσετε να εκτυπώσετε την txt έξω από την συνάρτηση, θα επιστρέψει ” global variable” επειδή η txt είναι καθολική μεταβλητή και η τιμή της δεν μπορεί να αλλάξει μέσα σε συνάντηση.

7.16. Ο συντελεστής καθολικής ανάθεσης

Κανονικά, όταν δημιουργείτε μια μεταβλητή μέσα σε μια συνάρτηση, αυτή η μεταβλητή είναι τοπική (local) και μπορεί να χρησιμοποιηθεί μόνο εντός αυτής της συνάρτησης. Μπορούμε όμως να δημιουργήσουμε καθολικές μεταβλητές μέσα σε μία συνάρτηση ή να αλλάξουμε τιμή σε μια καθολική μεταβλητή μέσα σε μια συνάρτηση χρησιμοποιώντας τον τελεστή καθολικής εκχώρησης **<<-** .

Άρα

- για να δημιουργήσετε μια καθολική μεταβλητή μέσα σε μια συνάρτηση, πρέπει να χρησιμοποιήσετε τον τελεστή καθολικής εκχώρησης **<<-**
- για να αλλάξετε την τιμή μιας καθολικής μεταβλητής μέσα σε μια συνάρτηση, αναθέστε στη μεταβλητή την τιμή που θέλετε χρησιμοποιώντας τον τελεστή καθολικής εκχώρησης **<<-** .

Παραδειγμα 1:

```
my_function <- function() {
  txt <<- "fantastic"
  paste("R is", txt)
}
my_function()
```

```
[1] "R is fantastic"
```

```
print(txt)
```

```
[1] "fantastic"
```

Στο παραπάνω παράδειγμα η μεταβλητή *txt* δημιουργείται εντός της συνάρτησης *my_function()* με τον τελεστή καθολικής εκχώρησης. Κατά συνέπεια είναι καθολική μεταβλητή και συνεχίζει να υπάρχει εφόσον έχει ολοκληρωθεί η συνάρτηση

Παράδειγμα 2:

```
txt <- "awesome"
my_function <- function() {
  txt <<- "fantastic"
  paste("R is", txt)
}
my_function()
```

```
[1] "R is fantastic"
```

```
paste("R is", txt)
```

```
[1] "R is fantastic"
```

Στο παραπάνω παράδειγμα η μεταβλητή *txt* είναι καθολική μεταβλητή που αρχικά παίρνει τιμή “awesome”. Στη συνέχεια, εντός της συνάρτησης *my_function()* παίρνει τιμή “fantastic” με την χρήση του τελεστή καθολικής εκχώρησης. Μετά το πέρας της συνάρτησης *my_function()* η μεταβλητή txt συνεχίζει έχει τιμή “*fantastic*” διότι εντός συνάρτησης χρησιμοποιήθηκε ο τελεστή καθολικής εκχώρησης.

Ερωτήσεις αυτοαξιολόγησης

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

21. Έστω οι παρακάτω εντολές:

```
x<-4
f1<-function(){
  x<-2
  y<-x^2
}
y<-f1()
y==x
```

Το αποτέλεσμα της τελευταίας γραμμής κώδικα θα είναι

α. FALSE

β. TRUE

γ. Error

22. Έστω η συνάρτηση

```
f2<-function(x=40,y=20){
  z<-x+y
}
```

Οι παρακάτω εντολές

```
y1<-f2()
y2<-f2(40,20)
y1==y2
```

Θα δώσουν αποτέλεσμα;

α. FALSE

β. TRUE

γ. Error

23. Σωστό - Λάθος: Μπορώ να έχω προεπιλεγμένες τιμές για μόνο ένα όρισμα μιας συνάρτησης;

24. Έστω η συνάρτηση

```
f2<-function(x,y=20){
  z<-x+y
}
```

Ποια ή ποιες από τις παρακάτω κλήσεις της συνάρτησης της συνάρτησης είναι σωστή/ες

α. f2(10)

β. f2(10,15)

γ. f2(10,20)

25. Έστω οι παρακάτω εντολές:

```
x<-4
f3<-function(){
  x<<-2
  y<-x^2
}
y<-f3()

print(x)
```

Το αποτέλεσμα της τελευταίας γραμμής κώδικα θα είναι

α. 2

β. 4

γ. 6

Κεφάλαιο 8 Επιλογές και βρόχοι στην R

8.1. Εισαγωγή

Στο προηγούμενο κεφάλαιο παρουσιάσαμε τις συναρτήσεις στα πλαίσια της R και όπως είδαμε μία συνάρτηση είναι μια απλή ακολουθία ενεργειών/εντολών. Συχνά όμως, θέλετε να κάνετε επιλογές και ενέργειες ανάλογα με μια συγκεκριμένη λογική συνθήκη.

Οι επιλογές (if) στον προγραμματισμό γενικά είναι της μορφής:

«Αν (if) ισχύει μία λογική σχέση τότε κάνε αυτό, αλλιώς (else) κάνε το άλλο»

Οι επιλογές δεν είναι τα μόνα πράγματα που μπορούν να είναι χρήσιμα στον προγραμματισμό συναρτήσεων και όχι μόνο. Οι βρόχοι (loops) μπορούν να χρησιμεύσουν ώστε να μην χρειαστεί να ξαναγράψετε τον ίδιο κώδικα ξανά και ξανά. Για παράδειγμα, αν θέλετε να υπολογίσετε τα περιγραφικά στατιστικά μέτρα για διαφορετικά σύνολα δεδομένων (αρχεία δεδομένων), μπορείτε να γράψετε τον κώδικα για να υπολογίσετε αυτά τα στατιστικά μέτρα και μετά να πείτε στην R να το επαναλάβει τις εντολές για όλα τα αρχεία που σας ενδιαφέρουν.

8.2. Κάνοντας επιλογές με δηλώσεις if

Ο ορισμός μιας επιλογής (**if**) στον κώδικα της R είναι πολύ απλός:

«Εάν ισχύει μια συνθήκη, τότε εκτέλεσε μια συγκεκριμένη εργασία».

Οι περισσότερες γλώσσες προγραμματισμού σας επιτρέπουν να το κάνετε τέτοιου είδους επιλογές, χρησιμοποιώντας ακριβώς αυτές οι λέξεις αλλά στα αγγλικά, δηλαδή:

if ... then ...

Στα πλαίσια της R, είναι ακόμα πιο εύκολο γιατί δεν χρειάζεται η λέξη **then.**

8.3. Η επιλογή If στην R

Η δήλωση if στην R αποτελείται από τρία στοιχεία:

1. τη λέξη-κλειδί ***if***
2. μια λογική τιμή μεταξύ παρενθέσεων (ή μία έκφραση που οδηγεί σε μία λογική τιμή)
3. ένα μπλοκ κώδικα μέσα σε άγκιστρα, που πρέπει να εκτελεστεί όταν η λογική τιμή στο 2 παίρνει τιμή TRUE (ΑΛΗΘΕΙΑ)

Στα πλαίσια αυτού του κεφαλαίου, θα μελετήσετε τις επιλογές μέσω ενός παραδείγματος.

Παράδειγμα

Έστω ότι είσαστε ένας ελεύθερος επαγγελματίας και θέλετε να φτιάξετε μία συνάρτηση που να υπολογίζει την τιμή που πρέπει να χρεώνετε στους πελάτες σας. Η τιμή ανά πελάτη υπολογίζεται βάσει των ωρών δουλειάς που κάνατε για αυτόν τον πελάτη. Άρα για να κατασκευάσετε την συνάντηση που θα υπολογίζει την αμοιβή σας, θα πρέπει να ξέρετε τον αριθμό των ωρών

εργασίας καθώς και την τιμή που χρεώνεται ανά ώρα. Κατά συνέπεια, η συνάρτηση αυτή θα πρέπει να παίρνει τον αριθμό των ωρών (hours) και την τιμή ανά ώρα (pph) ως στοιχεία εισόδου.

Η συνάρτηση για τον υπολογισμό της καθαρής αξίας των έργων σας, θα είναι της μορφής:

```
timologisi1 <- function(hours, pph=40){
  net.price <- hours * pph
  round(net.price)
}
```

Για να δούμε αναλυτικά τον παραπάνω κώδικα:

- Με τη λέξη-κλειδί **function**, ορίζετε τη συνάρτηση.
- Μεταξύ των παρενθέσεων, καθορίζετε τα ορίσματα τις συναρτήσεις που είναι οι ώρες (hours) και η τιμή ανά ώρα (pph).
- Το όρισμα hours δεν έχει προεπιλεγμένη τιμή, ενώ το όρισμα pph έχει προεπιλεγμένη τιμή 40 ευρώ ανά ώρα
- Όλες οι εντολές ανάμεσα στα άγκιστρα είναι το σώμα της συνάρτησης. Αναλυτικά,
  - Υπολογίζετε την καθαρή τιμή/αξία (net price) της εργασίας σας πολλαπλασιάζοντας τις hours με το pph.
  - Το αποτέλεσμα της τελευταίας δήλωσης στο σώμα της συνάρτησης είναι η επιστρεφόμενη τιμή της συνάρτησης. Σε αυτήν την περίπτωση, αυτή είναι η συνολική τιμή που στρογγυλοποιείται στο ευρώ.

Στην παραπάνω συνάρτηση, θα μπορούσατε να μην βάλετε το όρισμα pph και να πολλαπλασιάσετε τις ώρες με 40 μέσα στα άγκιστρα (στο σώμα της συνάρτησης). Αλλά αυτό θα σήμαινε ότι εάν, για παράδειγμα, ο συνάδελφός σας χρησιμοποιεί διαφορετική ωριαία χρέωση ή εσείς χρεώνεται ένα πελάτη με διαφορετική τιμή, τότε θα πρέπει να αλλάξει την τιμή στο σώμα της συνάρτησης για να μπορεί να την χρησιμοποιήσετε.

Όπως παρουσιάσαμε στο προηγούμενο κεφάλαιο είναι καλή πρακτική προγραμματισμού να χρησιμοποιείτε ορίσματα με προεπιλεγμένες τιμές για οποιαδήποτε τιμή που μπορεί να αλλάζει όταν εφαρμόζεται την συνάντηση σας διότι, κάτι τέτοιο κάνει μια συνάρτηση πολύ πιο ευέλικτη.

8.4. Εισάγοντας μία επιλογή

Υποθέστε τώρα, ότι έχετε μερικούς «μεγάλους» πελάτες που σας δίνουν πολλή δουλειά. Είναι συνήθης πρακτική στους πελάτες που προσφέρουν πολλή εργασία να κάνετε μία μείωση της ωριαίας τιμής προκειμένου να τους κρατήσετε ως πελάτες. Αποφασίζετε λοιπόν, να τους κάνετε μία μείωση 10 τοις εκατό στην τιμή ανά ώρα για δουλείες/έργα που απαιτούν περισσότερες από 100 ώρες εργασίας.

Αυτό σημαίνει ότι, εάν ο αριθμός των ωρών εργασίας είναι μεγαλύτερος από 100, η καθαρή αξία του έργου θα πρέπει να πολλαπλασιαστεί με 0.9, ενώ για έργα μέχρι 100 ώρες, η καθαρή αξία θα πρέπει να πιαστεί με το 1.

Για να μπορέσετε να εισάγεται αυτήν την διαφοροποίηση στην συνάντηση σας θα πρέπει να μετατρέψετε τον κώδικα της συνάρτησης ***timologisi1*** και να εισάγεται μία επιλογή (**if**), όπως παρακάτω:

```
timologisi2 <- function(hours, pph=40){
  net.price <- hours * pph
  if(hours > 100) {
    net.price <- net.price * 0.9
  }
  round(net.price)
}
```

Δοκιμάστε την συνάρτηση timologisi2() για ένα έργο 55 ωρών και ένα άλλο έργο 110 ωρών χρησιμοποιώντας τις παρακάτω εντολές

```
timologisi2(hours = 55)

[1] 2200

timologisi2(hours = 110)

[1] 3960
```

Όπως αναφέρθηκε, η δήλωση ***if*** στην R αποτελείται από τρία στοιχεία:

- τη λέξη-κλειδί if,
- μία λογική συνθήκη μεταξύ παρενθέσεων και
- ένα μπλοκ κώδικα μεταξύ αγκίστρων που θα εκτελεστεί όταν η λογική συνθήκη είναι **ΑΛΗΘΗΣ** (TRUE).

Αν κοιτάξετε το μπλοκ εντολών της if στην προηγούμενη συνάρτηση, θα βρείτε αυτά τα τρία στοιχεία.

Αυτή η δομή είναι ο γενικός τρόπος με τον οποίο μπορείτε να καθορίσετε μια δήλωση if στα πλαίσια της R. Σε περίπτωση όμως που έχετε μόνο μια γραμμή κώδικα στο μπλοκ κώδικα της if, τότε δεν χρειάζεται να βάλετε άγκιστρα. Στην περίπτωση αυτή ο κώδικας της συνάρτησης θα πάρει την παρακάτω μορφή

```
timologisi3 <- function(hours, pph=40){
  net.price <- hours * pph
  if(hours > 100) net.price <- net.price * 0.9
  round(net.price)
}
```

8.5. Βάζοντας και το else με το if

Σε ορισμένες περιπτώσεις, χρειάζεστε την συνάρτησή σας για να κάνετε κάτι (μία σειρά ενεργειών) εάν ισχύει μια συνθήκη και κάτι άλλο (μία άλλη σειρά ενεργειών) αν δεν ισχύει η συνθήκη.

Θα μπορούσατε να το επιτύχετε αυτό με δύο δηλώσεις ***if***, αλλά υπάρχει ένας ευκολότερος τρόπος στην R:

- μια δήλωση ***if*** ... ***else***.

Η δήλωση «***if*** .... ***else*** ...» περιέχει τα ίδια στοιχεία με μια δήλωση ***if***, και στη συνέχεια κάποια επιπλέον στοιχεία, δηλαδή:

- τη λέξη-κλειδί ***else***, τοποθετείται μετά το πρώτο μπλοκ κώδικα (το μπλοκ της δήλωσης if)
- ένα δεύτερο μπλοκ κώδικα, που περικλείεται με άγκιστρα, που πρέπει να εκτελεστεί αν δεν είναι αληθής η λογική συνθήκη του ***if***.

Για να δούμε πως μπορούμε να γενικεύσουμε την συνάρτηση που τιμολογεί την εργασία σας στα πλαίσια του παραδείγματος. Σε ορισμένες χώρες, ο φόρος προστιθέμενης αξίας (ΦΠΑ-VAT) που πρέπει να καταβληθεί για ορισμένες υπηρεσίες εξαρτάται από το αν ο πελάτης ανήκει στο δημόσιο τομέα ή στον ιδιωτικό τομέα. Στα πλαίσια του παραδείγματος, υποθέστε ότι οι δημόσιοι οργανισμοί πληρώνουν 6 τοις εκατό ΦΠΑ και οι ιδιωτικοί οργανισμοί πληρώνουν 12 τοις εκατό ΦΠΑ.

Πως πρέπει να μετατρέψετε την συνάρτηση **timologisi3()** για συμπεριλάβετε αυτή την διαφοροποίηση;

- Αρχικά πρέπει να προσθέσετε ένα επιπλέον όρισμα στην συνάρτηση που να αφορά στο είδος του πελάτη, π.χ. το αντικείμενο public που θα παίρνει τιμές TRUE για δημόσιο τομέα και FALSE για ιδιωτικό.
- Στην συνέχεια θα πρέπει να προσθέσετε μία επιπλέον δήλωση ***if...else*** για το προσδιορισμό του ΦΠΑ. Ο κώδικας της συνάρτησης θα πάρει την παρακάτω μορφή:

```
timologisi4 <- function(hours, pph=40, public=TRUE){
  net.price <- hours * pph
  if(hours > 100) net.price <- net.price * 0.9
  if(public) {
    tot.price <- net.price * 1.06
  } else {
    tot.price <- net.price * 1.12
  }
  round(tot.price)
}
```

Στην περίπτωση του δεύτερου if, όταν η λογική συνθήκη είναι αληθής τότε ο πελάτης είναι δημόσιος οργανισμός και η καθαρή αξία θα πολλαπλασιαστεί με το 1,06 για να μας δώσει την τελική αξία. Στην περίπτωση που η λογική συνθήκη του if είναι ψευδής τότε η καθαρή αξία θα πολλαπλασιαστεί με το 1,12 διότι ο πελάτης είναι ιδιωτικός οργανισμός.

Δοκιμάστε τώρα να τρέξετε την συνάρτησή σας για ένα έργο 25 ωρών που σας έχει αναθέσει ένας δημόσιος οργανισμός και ένα έργο 25 ωρών που σας έχει αναθέσει ένας ιδιωτικός οργανισμός. Για να επιτύχετε αυτό θα πρέπει να εκτελέσετε τις παρακάτω εντολές:

```
timologisi4(25,public=TRUE)

[1] 1060

timologisi4(25,public=FALSE)

[1] 1120
```

8.6. Απλοποιώντας το if...else

Στα πλαίσια αυτής της ενότητας, θα δούμε πως μπορούμε να απλοποιήσουμε την δήλωση ***if...else***.

Αρχικά, επειδή οι εντολές στο if και στο else είναι μόνο μία, μπορούμε να απλοποιήσουμε αφαιρώντας τα άγκιστρα. Άρα στα πλαίσια του παραδείγματος μας:

```
timologisi5 <- function(hours, pph=40, public=TRUE){
  net.price <- hours * pph
  if(hours > 100) net.price <- net.price * 0.9
  if(public) tot.price <- net.price * 1.06 else
    tot.price <- net.price * 1.12
  round(tot.price)
}
```

Επιπρόσθετα, η δήλωση if λειτουργεί σαν μια συνάρτηση και, ως εκ τούτου, επιστρέφει επίσης μια τιμή. Ως αποτέλεσμα, μπορείτε να αντιστοιχίσετε αυτήν την τιμή σε ένα αντικείμενο ή να το χρησιμοποιήσετε σε υπολογισμούς. Επομένως, αντί να υπολογίσουμε ξανά το net.price και εκχωρώντας το αποτέλεσμα σε tot.price στα μπλοκ κώδικα, μπορείτε να χρησιμοποιήσετε όπως παρακάτω:

```
timologisi6 <- function(hours, pph=40, public=TRUE){
  net.price <- hours * pph
  if(hours > 100) net.price <- net.price * 0.9
  tot.price <- net.price * if(public) 1.06 else 1.12
  round(tot.price)
}
```

8.7. Η εντολή if με τη χρήση διανυσμάτων

Η χρήση διανυσμάτων (vectors) είναι ένα από τα σημαντικά χαρακτηριστικά της γλώσσας R που την έχει κάνει δημοφιλής στα πλαίσια των ποσοτικών επιστημών.

Στο παράδειγμά μας, έστω ότι έχετε 2 έργα/πελάτες σε ένα διάνυσμα της R:

```
customers<-c(25, 110)
```

Αν τρέξετε την συνάρτηση timologisi6:

```
timologisi6(customers)

Error in if (hours > 100) net.price <- net.price * 0.9: the conditio
n has length > 1
```

Η R αναφέρει σφάλμα, και ο λόγος είναι ότι η εντολή if δεν δέχεται διανύσματα. Προκειμένου όμως να χρησιμοποιούμε διανύσματα, η R έχει μία έκδοση της if που παίρνει ως ορίσματα διανύσματα, και αυτή είναι η εντολή/συνάρτηση ***ifelse()***.

Η συνάρτηση ***ifelse()***, είναι ο τρόπος επιλογής τιμών από δύο διανύσματα, και αποτελείται από:

- ένα διάνυσμα ελέγχου με λογικές τιμές.
- ένα διάνυσμα με τιμές που πρέπει να επιστραφούν εάν η αντίστοιχη τιμή στο διάνυσμα ελέγχου είναι TRUE (ΑΛΗΘΗΣ).
- ένα διάνυσμα με τιμές που πρέπει να επιστραφούν εάν η αντίστοιχη τιμή στο διάνυσμα ελέγχου είναι FALSE.

Δοκιμάστε τις παρακάτω εκδόσεις της εντολής ***ifelse()***:

```
ifelse(c (1, 3) <2.5, 1: 2, 3: 4)
[1] 1 4
ifelse(c(3,1) < 2.5 , "small" , "big")
[1] "big"   "small"
ifelse(c(3,1,1) < 2.5 , "small" , "big")
[1] "big"   "small" "small"
```

Ας μελετήσουμε, αναλυτικά, πως δουλεύει η εντολή ifelse (c(1, 3) <2.5, 1: 2, 3: 4):

- Η έκφραση c (1, 3) <2.5 παράγει το διάνυσμα λογικών τιμών (TRUE, FALSE).
- Η πρώτη τιμή αυτού του διανύσματος είναι TRUE, επειδή το 1 είναι μικρότερο από 2.5. Έτσι, η πρώτη τιμή του αποτελέσματος είναι η πρώτη τιμή του δεύτερου ορίσματος/διανύσματος, δηλαδή του (1: 2), που είναι 1.
- Η επόμενη τιμή είναι FALSE, επειδή το 3 είναι μεγαλύτερο από 2.5. Ως εκ τούτου, η ifelse() παίρνει τη δεύτερη τιμή του τρίτου ορίσματος/διανύσματος, δηλαδή του (3:4), που είναι 4.
- Ως αποτέλεσμα της εντολής ***ifelse()*** επιστρέφεται ένα διάνυσμα με τις τιμές 1 και 4.

Μπορούμε λοιπόν να χρησιμοποιήσουμε την εντολή ***ifelse()*** για να μετατρέψουμε την συνάντηση τιμολόγησης προκειμένου να δέχεται πολλούς πελάτες. Ο κώδικας της συνάρτησης είναι

```
timologisi7 <- function(hours,pph=40,public){
  net.price <- hours * pph
  net.price <- net.price * ifelse(hours > 100 , 0.9, 1)
  tot.price <- net.price * ifelse(public, 1.06, 1.12)
  round(tot.price)
}
```

Σε αυτό το πλαίσιο, κατασκευάστε ένα data frame (pelatologio) με τους πελάτες σας που να περιέχει μία μεταβλητή με τις ώρες των έργων και μία μεταβλητή με το αν πελάτης είναι δημόσιος ή ιδιωτικός οργανισμός:

```
pelatologio <- data.frame(hours = c(25, 110, 125, 40), public = c(TR
UE,TRUE,FALSE,FALSE))
```

Μπορούμε τώρα να χρησιμοποιήσουμε αυτό το data frame με την συνάρτηση timologisi7():

```
timologisi7(pelatologio$hours, public = pelatologio$public)
```

```
[1] 1060 4198 5040 1792
```

Δοκιμάστε επίσης να τρέξετε την συνάρτηση timologisi7() χρησιμοποιώντας την εντολή with() (η συνάρτηση with μας επιτρέπει να εφαρμόσουμε μία συνάρτηση σε μεταβλητές ενός data frame χωρίς την χρήση του $ για να προσδιορίσουμε τις μεταβλητές):

```
with(pelatologio, timologisi7(hours, public = public))
```

```
[1] 1060 4198 5040 1792
```

8.8. Πολλές επιλογές

Οι εντολές **if** και **if ... else**, σας επιτρέπουν ακριβώς δύο επιλογές, αλλά πολύ συχνά είσαστε υποχρεωμένοι να επιλέξετε από πολλαπλές επιλογές.

Στο παράδειγμα για το οποίο έχουμε αναπτύξει την συνάρτηση τιμολόγησης, φανταστείτε ότι έχετε μερικούς πελάτες από το εξωτερικό. Ας υποθέσουμε ότι πελάτες με έδρα στο εξωτερικό δεν χρειάζεται να πληρώσουν ΦΠΑ. Άρα, τώρα έχετε τρεις διαφορετικούς συντελεστές ΦΠΑ: 12% για ιδιώτες πελάτες, 6 % για πελάτες στο δημόσιο και 0% για αλλοδαπούς πελάτες. Θα πρέπει να τροποποιήσετε την συνάρτηση τιμολόγησης για να περιλαμβάνει αυτές τις τρεις επιλογές ΦΠΑ.

Ο πιο διαισθητικός τρόπος για την επίλυση αυτού του προβλήματος είναι η αλυσίδα επιλογών, δηλαδή μία σειρά (μία αλυσίδα) εντολών if. Δηλαδή:

- Αν (if) ένας πελάτης ζει στο εξωτερικό, τότε να μην χρεώνετε ΦΠΑ.
- Διαφορετικά (else), ελέγξτε αν (if) ο πελάτης είναι δημόσιος ή ιδιωτικός και εφαρμόστε τον σχετικό συντελεστή ΦΠΑ.

Εάν ορίσετε ένα αντικείμενο pelates για τη συνάρτηση που μπορεί να λάβει τις τιμές ”abroad”, “public”, “private’, τότε μπορείτε να τροποποιήσετε τον κώδικα:

```
timologisi8 <- function(hours,pph=40,typepelatis){
  net.price <- hours * pph
  net.price <- net.price * ifelse(hours > 100 , 0.9, 1)
  if(typepelatis=='private'){
   tot.price <- net.price * 1.12      # 12% VAT
  } else {
   if(typepelatis=='public'){
     tot.price <- net.price * 1.06    # 6% VAT
   } else {
     tot.price <- net.price * 1    # 0% VAT
   }
  }
  round(tot.price)
}
```

Για να τιμολογήσουμε λοιπόν ένα έργο 25 ωρών από πελάτη με έδρα το εξωτερικό, θα πρέπει να εκτελέσουμε την εντολή:

```
timologisi8(245,typepelatis = "abroad")
```

```
[1] 8820
```

Με αντίστοιχο τρόπο μπορούμε να χρησιμοποιήσουμε και την εντολή ifelse που εφαρμόζεται σε διανύσματα, δηλαδή με αλυσίδα επιλογών

```
timologisi9 <- function(hours,pph=40,typepelatis){
  net.price <- hours * pph
  net.price <- net.price * ifelse(hours > 100 , 0.9, 1)
  VAT <- ifelse(typepelatis=='private', 1.12,
           ifelse(typepelatis == 'public', 1.06, 1)
         )
  tot.price <- net.price * VAT
  round(tot.price)
}
```

Τώρα στο data frame, pelatologio έχουμε μια επιπλέον μεταβλητή, την pelates που αφορά την έδρα της επιχείρησης.

```
pelatologio <- data.frame(
  hours = c(25, 110, 125, 40),
  public = c(TRUE,TRUE,FALSE,FALSE),
  pelates= c('public','abroad','private','abroad')
)
```

Για να τιμολογήσουμε τα έργα που αναφέρονται στο συγκεκριμένο data frame, αρκεί να εκτελέσουμε την εντολή

```
with(pelatologio, timologisi9(hours, typepelatis=pelates))
```

```
[1] 1060 3960 5040 1600
```

8.9. Εντολή Switch

Αντί για τα πολλαπλά (αλυσωτά) if, μπορείτε να χρησιμοποιήσετε την εντολή switch(), όπως παρακάτω:

```
pelates <- 'abroad'
VAT <- switch(pelates, private=1.12, public=1.06, abroad=1)
print(VAT)
```

```
[1] 1
```

Για την εφαρμογή της εντολής switch

- Δώστε μια τιμή στο πρώτο όρισμα (σε αυτή την περίπτωση στο pelates). Σημειώστε ότι η switch() δεν δέχεται διανύσματα.
- Μετά το πρώτο όρισμα, δίνετε μια λίστα επιλογών με τιμές. Σημειώστε ότι δεν χρειάζεται να βάλετε " " γύρω από τις επιλογές.

Να θυμάστε όμως ότι η switch() δεν λειτουργεί με διανύσματα. Άρα αν θέλουμε την εφαρμόσουμε σε πολλούς πελάτες θα πρέπει να χρησιμοποιήσουμε και ένα βρόχο (loop).

8.10. Βρόχοι - Loops

Μερικές από τις εντολές/συναρτήσεις της R δεν είναι διανυσματικές (όπως η switch), οπότε μπορείτε να χρησιμοποιήσετε μόνο μία τιμή. Θα μπορούσατε, φυσικά, να εφαρμόσετε τον κώδικα σε κάθε τιμή που έχετε, αλλά είναι πολύ πιο λογικό να θέλετε να αυτοματοποιήσετε αυτήν την εργασία. Αυτό μπορεί να γίνει με την χρήση βρόχων (loops).

Όπως σε πολλές άλλες γλώσσες προγραμματισμού, η R παρέχει αντίστοιχες εντολές για κατασκευή βρόχων, όπως θα δούμε παρακάτω.

8.11. Ο βρόχος for

Ο βρόχος ***for*** στην R έχει την ακόλουθη μορφή:

for(i in values)

{

... κάνε κάτι ...

}

Αναλυτικά, αποτελείται από τα ακόλουθα μέρη:

- Τη λέξη-κλειδί for, ακολουθούμενη από παρενθέσεις.
- Ένα δείκτη μεταξύ των παρενθέσεων. Σε αυτό το παράδειγμα, χρησιμοποιούμε το i, αλλά μπορεί να είναι οποιοδήποτε όνομα και αντικείμενο.

- Τη λέξη-κλειδί ***in***, που ακολουθεί το δείκτη.
- Ένα διάνυσμα με τιμές που θα πάρει ο βρόχος. Σε αυτό το παράδειγμα κώδικα, χρησιμοποιούμε το values.
- Ένα μπλοκ κώδικα μέσα σε άγκιστρα.

Μπορούμε να χρησιμοποιήσουμε ένα τέτοιο βρόχο και την εντολή switch() για να γενικεύσουμε τη συνάρτηση τιμολόγησης, όπως παρακάτω:

```
timologisi10 <- function(hours, pph=40, pelatestype){
  net.price <- hours * pph *
    ifelse(hours > 100, 0.9, 1)

  nclient <- length(pelatestype)
  VAT <- numeric(nclient)
  for(i in 1:nclient){
    VAT[i] <- switch(pelatestype[i], private=1.12, public=1.06, 1)
  }
  tot.price <- net.price * VAT
  round(tot.price)
}
```

Τώρα μπορούμε να χρησιμοποιήσουμε το πελατολόγιο μας (data frame) με αυτή την συνάρτηση

```
timologisi10(pelatologio$hours, pelatestype=pelatologio$pelates)
```

```
[1] 1060 3960 5040 1600
```

8.12. Οικογένεια εντολών Apply

Οι εντολές apply(), sapply() και lapply(), είναι συναρτήσεις που εφαρμόζουν μία άλλη συνάρτηση σε ένα αντικείμενο. Οι διαφορές μεταξύ τους αναφέρονται στο είδος του αντικειμένου έχουν για είσοδο και για έξοδο. Ο παρακάτω πίνακας παρουσιάζει εν συντομία τα διαφορετικά αντικείμενα εισόδου και εξόδου των εντολών apply, sapply και lapply.

Πίνακας 1: Εντολές apply, sapply, lapply

Συνάρτηση	Αντικείμενο στο οποίο εφαρμόζεται η συνάρτηση	Που εφαρμόζετε η συνάρτηση	Αντικείμενο εξόδου
apply	Matrix	Γραμμές ή στήλες	Το απλούστερο αντικείμενο (vector, matrix, array, list)
	Array	Γραμμές ή στήλες ή παραπάνω διάσταση	Το απλούστερο αντικείμενο (vector, matrix, array, list)
	Data Frame	Γραμμές ή στήλες	Το απλούστερο αντικείμενο (vector, matrix, array, list)

sapply	Vector	Στοιχεία διανύσματος	Το απλούστερο αντικείμενο (vector, matrix, array, list)
	Data Frame	Μεταβλητές	Το απλούστερο αντικείμενο (vector, matrix, array, list)
	List	Συστατικά Λίστας	Το απλούστερο αντικείμενο (vector, matrix, array, list)
lapply	Vector	Στοιχεία διανύσματος	List
	Data Frame	Μεταβλητές	List
	List	Συστατικά Λίστας	List

Ας δούμε ένα παράδειγμα: Έστω ότι μετρήσετε τους πόντους που σημείωσαν τα τρία σας παιδιά σε 6 αγώνες μπάσκετ του σχολικού πρωταθλήματος, και αποθηκεύσετε τα δεδομένα σε έναν πίνακα (matrix)

```
baskets.of.George <- c(12,4,5,6,9,3)
baskets.of.Gerasimos <- c(5,4,2,4,12,9)
baskets.of.Angelos <- c(6,3,2,8,1,7)

baskets.team1 <- cbind(baskets.of.George, baskets.of.Gerasimos, bask
ets.of.Angelos)
print(baskets.team1)
```

```
     baskets.of.George baskets.of.Gerasimos baskets.of.Angelos
[1,]                12                     5                   6
[2,]                 4                     4                   3
[3,]                 5                     2                   2
[4,]                 6                     4                   8
[5,]                 9                    12                   1
[6,]                 3                     9                   7
```

Στον πίνακα baskets.team1, κάθε στήλη αντιπροσωπεύει ένα διαφορετικό παιδί και κάθε σειρά αντιπροσωπεύει ένα διαφορετικό παιχνίδι.

Έστω ότι θέλετε να μάθετε τον μέσο αριθμό πόντων για κάθε παιδί. Για να το πετύχετε αυτό θα μπορούσατε να δημιουργήσετε ένα βρόχο και να εφαρμόσετε την συνάρτηση mean (συνάρτηση της R υπολογίζει το μέσο όρο) σε κάθε στήλη του πίνακα, ή να χρησιμοποιήσετε την συνάρτηση apply():

```
apply(baskets.team1, 2, mean)
```

```
   baskets.of.George baskets.of.Gerasimos   baskets.of.Angelos
                 6.5                   6.0                   4.5
```

Η συνάρτηση apply() έχει τα εξής ορίσματα:

- Το αντικείμενο στο οποίο πρέπει να εφαρμοστεί η συνάρτηση: Σε αυτήν την περίπτωση, είναι ο πίνακας baskets.team1.

- Η διάσταση στην οποία πρέπει να εφαρμοστεί η συνάρτηση: Ο αριθμός 1 δηλώνει τις γραμμές και ο αριθμός 2 δηλώνει τις στήλες. Στο παραπάνω παράδειγμα, η συνάρτηση θα εφαρμοστεί στις στήλες του πίνακα.
- Το όνομα της συνάρτησης που πρέπει να εφαρμοστεί: Μπορείτε να χρησιμοποιήσετε "" γύρω από το όνομα της συνάρτησης, αλλά δεν είναι υποχρεωτικό. Εδώ, εφαρμόζουμε την συνάρτηση ***mean()***.

Η συνάρτηση apply() λειτουργεί σε οποιοδήποτε αντικείμενο της R που έχει διαστάσεις, αλλά τι γίνεται αν το αντικείμενο δεν έχει διαστάσεις (για παράδειγμα, όταν έχετε μια λίστα ή ένα διάνυσμα); Για αυτές τις περιπτώσεις, έχετε δύο συναφείς συναρτήσεις : την sapply() και την lapply().

Οι δύο συναρτήσεις (sapply και lapply) λειτουργούν σχεδόν με τον ίδιο τρόπο. Η μόνη διαφορά είναι ότι το lapply() επιστρέφει ως αντικείμενο εξόδου, μια λίστα με το αποτέλεσμα, ενώ το sapply() επιστρέφει το πιο απλό αντικείμενο.

Όπως είδαμε στην προηγούμενη ενότητα η συνάρτηση switch(), δεν λειτουργεί με διανυσματικό τρόπο, αλλά μπορούμε να την χρησιμοποιήσουμε με την sapply() και να ξεπεράσουμε αυτό το εμπόδιο.

Για να δούμε ένα παράδειγμα: έστω ότι έχουμε ένα διάνυσμα με χαρακτήρες "a" και "b", και θέλουμε να βάλουμε "Hello" όπου υπάρχει το "a" και "Goodbye" στην θέση του κάθε "b". Αυτό μπορεί να επιτευχθεί με την παρακάτω εντολή:

```
sapply(c("a", "b"), switch, a = "Hello", b = "Goodbye")
```

```
        a         b
  "Hello" "Goodbye"
```

Η συνάρτηση sapply() λειτουργεί παρόμοια με την apply() εκτός από το γεγονός ότι δεν έχει όρισμα που να αναφέρεται στην διάσταση. Αναλυτικά,

- Το πρώτο όρισμα είναι το διάνυσμα στου οποίου τις τιμές θέλετε να εφαρμόσετε το συνάρτηση - σε αυτήν την περίπτωση, το διάνυσμα c ("a", "b").
- Το δεύτερο όρισμα είναι το όνομα της συνάρτησης - σε αυτήν την περίπτωση, switch.
- Όλα τα άλλα ορίσματα είναι απλά τα ορίσματα που μεταβιβάζετε στη συνάρτηση switch.

Σημειώστε ότι σε περίπτωση που αντί για sapply δηλώστε την lapply τότε ως έξοδο θα πάρετε μία λίστα.

Στην περίπτωση της συνάρτησης τιμολόγησης, θα μπορούσαμε να χρησιμοποιήσουμε την εντολή sapply() για να αποφύγουμε να χρησιμοποιήσουμε βρόχο

```
timologisi11 <- function(hours, pph=40, pelatestype){
  net.price <- hours * pph * ifelse(hours > 100, 0.9, 1)
  VAT <- sapply(pelatestype, switch, private=1.12, public=1.06, 1)
  tot.price <- net.price * VAT
```

```
  round(tot.price)
}
```

8.13. Ο βρόχος while και ο βρόχος repeat

Η γλώσσα ειδικού σκοπού R εκτός από το βρόχο ***for*** έχει και δύο ακόμη βρόχους, το **while** και το **repeat**. Οι βρόχοι αυτοί δεν είναι ιδιαίτερα δημοφιλείς στα πλαίσια των ποσοτικών μεθόδων για πολλούς και διάφορους λόγους, κυρίως γιατί συνήθως γνωρίζουμε το μέγεθος των δεδομένων μας και κατά συνέπεια εφαρμόζεται ο βρόχος **for** ή χρησιμοποιείται η οικογένεια εντολών apply/sapply/lapply.

Η γενική μορφή του βρόχου **while** είναι:

while (<condition>)

{

εντολές...

...

...

}

Όταν εκτελείται ο βρόχος while τότε ελέγχεται μία καθορισμένη συνθήκη. Εάν αξιολογηθεί ως TRUE, οι εντολές εντός του μπλοκ του while εκτελούνται μία προς μία. Η διαδικασία όπου το σύνολο των δηλώσεων εκτελείται ένα προς ένα ξεκινώντας από την πρώτη δήλωση μέχρι το τέλος του μπλοκ είναι γνωστή ως "επανάληψη του βρόχου".

Όταν το στοιχείο ελέγχου εκτέλεσης φτάσει στο τέλος του μπλοκ, η συνθήκη ελέγχεται ξανά και αν αξιολογηθεί ξανά ως TRUE, το μπλοκ του κώδικα εκτελείται ξανά. Αυτή η διαδικασία συνεχίζεται για όσο διάστημα η καθορισμένη συνθήκη αξιολογείται ως ΑΛΗΘΗΣ.

Εάν αξιολογηθεί FALSE, το μπλοκ κώδικα δεν θα εκτελεστεί. Σημειώστε ότι, εάν η συνθήκη δεν αξιολογηθεί ποτέ ως FALSE, ο βρόχος θα συνεχίσει να εκτελείται επ'άπειρον. Ένας τέτοιος βρόχος είναι γνωστός ως άπειρος βρόχος ή ατέρμων βρόχος.

Παράδειγμα: Έστω ότι θέλετε να εμφανίσετε/εκτυπώσετε τους αριθμούς από το ένα έως το 10 χρησιμοποιώντας ένα βρόχο while. Για να το πετύχετε αυτό θα πρέπει να εκτελέσετε τον παρακάτω κώδικα

```
Number <- 1
while(Number <= 10 ) {
 print (Number)
 Number = Number + 1
}
```

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

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

Ο παραπάνω ο κώδικας θα τυπώνει το Number όσο αυτό είναι μικρότερο ή ίσο του 10 και πρέπει να εκτελείτε όταν αριθμός γίνει μεγαλύτερος από 10.

Η γενική μορφή του βρόχου repeat είναι:

repeat {

εντολές…

…

…

}

Όταν συναντάται ο βρόχος repeat, οι δηλώσεις εντός του μπλοκ repeat εκτελούνται μία προς μία, μέχρι το τέλος του μπλοκ και στη συνέχεια αυτό το μπλοκ εντολών θα επαναλαμβάνεται άπειρες φορές. Ο τρόπος για να διακοπεί ένας βρόχος repeat (στα πλαίσια της R) είναι να υπάρχει μία λογική επιλογή (if) που όταν γίνεται αληθής να εκτελείτε η εντολή **break** (εντολή που διακόπτει την εκτέλεση).

Παράδειγμα: Έστω ότι θέλετε να εμφανίσετε/εκτυπώσετε τους αριθμούς από το 100 εως το 90 (φθίνουσα φορά) χρησιμοποιώντας ένα βρόχο repeat. Για να το πετύχετε αυτό θα πρέπει να εκτελέσετε τον παρακάτω κώδικα

```
Number <- 100
repeat{
  print (Number)
  Number = Number -1
  if (Number==89) break
}
```

```
[1] 100
[1] 99
[1] 98
[1] 97
[1] 96
[1] 95
[1] 94
[1] 93
[1] 92
[1] 91
[1] 90
```

Με το συγκεκριμένο κώδικα, θα εκτυπώνεται το Number και στη συνέχεια θα αφαιρείται μία μονάδα από το Number. Το επόμενο βήμα είναι να ελεγχθεί η επιλογή **if.** Όταν το Number γίνει 89 τότε η συνθήκη στην επιλογή θα είναι αληθής και θα εκτελεστεί η εντολή **break** που θα διακόψει το βρόχο.

Ερωτήσεις αυτοαξιολόγησης

(οι απαντήσεις βρίσκονται στο τέλος των σημειώσεων)

26. Έστω οι παρακάτω εντολές

```
m<-10:13
if(m>15) m<-m+1

Error in if (m > 15) m <- m + 1: the condition has le
ngth > 1
```

Το αποτέλεσμα τις τελευταίες γραμμές κώδικα είναι:

α. Το διάνυσμα m αλλά με στοιχεία από 11 έως το 14

β. Το διάνυσμα m αλλά με στοιχεία από 11 έως το 14

γ. σφάλμα

27. Ποιο είναι το αποτέλεσμα τις παρακάτω εντολής

```
z<-c(100,200,300)
ifelse(z >200, TRUE, FALSE)
```

α. FALSE, FALSE, FALSE

β. FALSE, FALSE, TRUE

γ. FALSE, TRUE, TRUE

28. Ποιο είναι το αποτέλεσμα των παρακάτω εντολών

```
n<-0
while(n<=100){
  print(n)
}
```

α. Θα εκτυπωθούν οι αριθμοί από το μηδέν μέχρι το 100.

β. Δεν θα εκτελεστεί ο βρόχος διότι υπάρχει λογικό σφάλμα

γ. Ο βρόχος είναι άπειρος

29. Έστω οι παρακάτω εντολές:

```
counts <- matrix(c(3,2,4,6,5,1,8,6,1), ncol=3)
colnames(counts) <- c('sparrow','dove','crow')
counts

     sparrow dove crow
[1,]       3    6    8
[2,]       2    5    6
[3,]       4    1    1
```

```
apply(counts, 2, max)

sparrow    dove    crow
      4       6       8
```

Η εντολή apply θα δώσει το αποτέλεσμα:

α. 4, 6, 8

β. 8, 6, 4

γ. Υπάρχει λάθος στην εντολή

30. Σωστό - Λάθος: Η εντολή switch μπορεί να χρησιμοποιηθεί με διανύσματα.

Απαντήσεις στις ερωτήσεις αυτοαξιολόγησης

Ερώτηση	Απάντηση
1	C
2	B
3	A
4	B
5	B
6	B
7	A
8	B
9	B
10	C
11	β
12	δ
13	β
14	α
15	α
16	δ
17	β
18	γ
19	α
20	α
21	β
22	β
23	Λάθος
24	αβγ
25	α
26	γ
27	β
28	γ
29	α
30	Λάθος

Βιβλιογραφία

- Καρλής Δ., Ντζούφρας Ι. (2015). **Εισαγωγή στον Προγραμματισμό και στη Στατιστική Ανάλυση με R.** Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών
- Wickham H., Grolemund G (2022). **Προγραμματισμός σε R για την επιστήμη των δεδομένων**. Εκδ. Κλειδάριθμος ISBN: 9789606452369
- Crawley M.J. (2007). **The R Book**. John Wiley & Sons, Ltd. ISBN: 978-0-470-51024-7
- Cotton R. (2013). **Learning R**. O'Reilly, ISBN: 978-1-449-35710-8
- Ismay C., Kim A.Y. (2020). **Statistical Inference via Data Science: A Modern Dive into R and the Tidyverse**. CRC Press, ISBN: 978-0-367-40982-1
- Zuur A.F., Ieno E.N., Meesters E. (2009). **A Beginner's Guide to R**. Springer, ISBN 978-0-387-93836-3