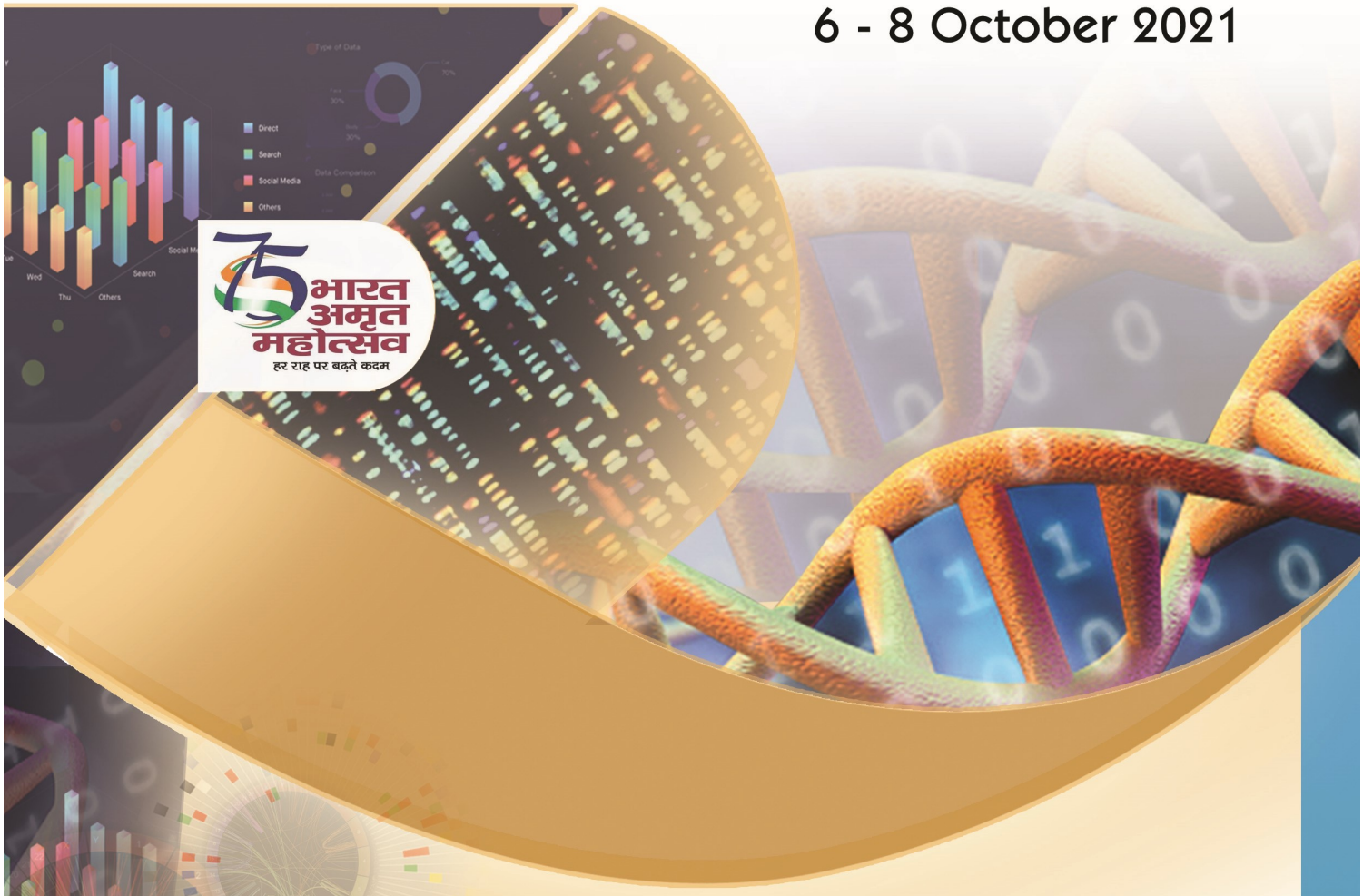
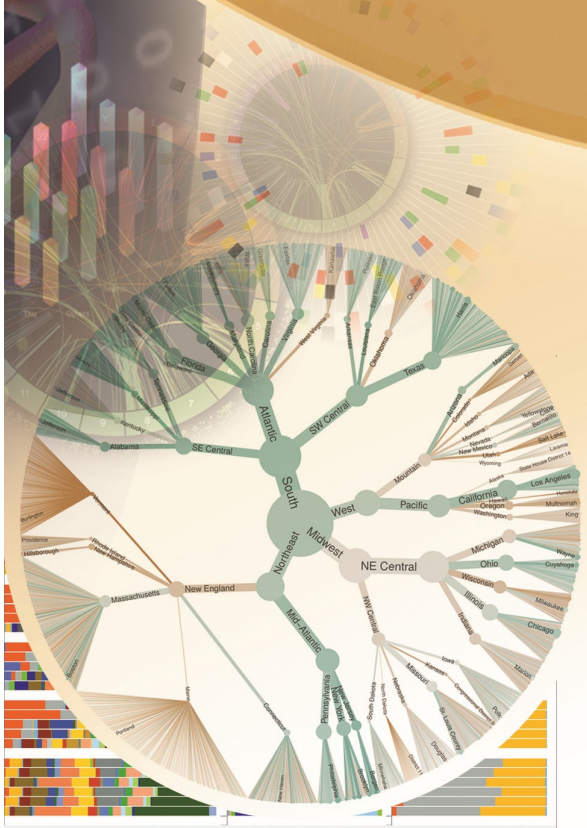
भाकृअनुप
ICAR

ONLINE WORKSHOP ON

R for BIOLOGISTS

6 - 8 October 2021





Workshop manual

Hands-on

Organized by

Bioinformatics Centre

ICAR-Indian Institute of Spices Research

Marikunnu, Kozhikode, Kerala

ONLINE WORKSHOP

ON

R FOR BIOLOGISTS

October 6-8, 2021

WORKSHOP MANUAL

(HANDS-ON)

Organized by

Bioinformatics Centre

ICAR- Indian Institute of Spices Research

Kozhikode, Kerala.

Published by

Dr. J Rema
Director, ICAR- Indian Institute of Spices Research

Organized by:

Bioinformatics Centre
ICAR- Indian Institute of Spices Research

Co-ordinators:

Ms. Sona Charles
Dr. Divya P Syamaladevi

Compiled & Edited by:

Ms. Sona Charles, Scientist, ICAR-IISR
Dr. Divya P Syamaladevi, Sr. Scientist, ICAR-IISR
Dr. Santhosh J Eapen, Principal Scientist & Head (Crop Protection) ICAR-IISR

This manual is an in-house publication intended for training purposes only and is not for public circulation.

Copyright © 2021 ICAR-IISR. All rights reserved. Reproduction and redistribution prohibited without approval.

CONTENTS

SCHEDULE

Day 1: 06-10-2021		
10:00 AM	Welcome Address	Dr. Divya P Syamaladevi, Senior Scientist (Agrl. Biotechnology), ICAR-Indian Institute of Spices Research
10:10 AM	Introductory Remarks and Release of Training Manual	Dr. J Rema, Director, ICAR-Indian Institute of Spices Research
10:20 AM	Felicitations	Dr. Santhosh J Eapen, Head, Crop Protection, ICAR-Indian Institute of Spices Research
10:25 AM	Introduction to the course and vote of thanks	Ms. Sona Charles, Scientist (Bioinformatics), ICAR-Indian Institute of Spices Research
Pre-evaluation and photo session		
10:45 AM	Biology: From Digital to Synthetic (Inaugural Lecture)	Dr. Santhosh J Eapen
11:45 AM	Introduction to R (Lecture)	Ms. Sona Charles
12:30 PM	Setting up the computer (Hands-on)	Dr. Divya P Syamaladevi
2:00 PM	Introduction to R (Hands- on)	Ms. Sona Charles
3:30 PM	Introduction to R coding: Control Statements (Lecture and Hands-on)	Dr. Divya P Syamaladevi
Day 2: 07-10-2021		
10:00 AM	Basics of Statistics	Dr Sreekumar J, Principal Scientist (Agricultural Statistics) ICAR-Central Tuber Crops Research Institute
2:00 PM	Statistics using R (Hands-on)	
Day 3: 08.10.2021		
10:00 AM	Introduction to data visualization	Ms. Sona Charles
11:00 AM	Basics of ggplot (Hands-on)	Ms. Sona Charles
2:00 PM	Case studies in Life Science (Hands-on)	Ms. Sona Charles
Post-evaluation		
Concluding Session		
4:00 PM	Welcome	Dr. Divya P Syamaladevi
4.05 PM	Feedback	Participants
4:10 PM	Concluding Remarks	Dr. Santhosh J Eapen
4:20 PM	Vote of Thanks	Ms. Sona Charles

1. INTRODUCTION TO R

Sona Charles
ICAR- Indian Institute of Spices Research
Email: sona.charles@icar.gov.in

Data types in R

```
> #Datatypes in R
> #Numeric
> num <- 1.2 #assigning the value 1.2 to the variable num
> print(num) #print the value of num
[1] 1.2
> class(num) #print the datatype of num
[1] "numeric"
> #Integer
> int <- as.integer(2.2) #Is 2.2 an integer?
> print(int)
[1] 2
> class(int)
[1] "integer"
> #character
> char <- "R is wonderful"
> print(char)
[1] "R is wonderful"
> class(char)
[1] "character"
> char1 <- "12345" #Any guess to what this will return??
> print(char1)
[1] "12345"
> class(char1)
[1] "character"
> #logical
> log_true <- TRUE
```

```
> print(log_true)
[1] TRUE
> class(log_true)
[1] "logical"
> log_false <- FALSE
> print(log_false)
[1] FALSE
> class(log_false)
[1] "logical"
> #factor
> fac <- factor(c("normal", "treatment1", "treatment2","treatment1", "normal", "treatment2"))
> print(fac)
[1] normal    treatment1 treatment2 treatment1 normal    treatment2
Levels: normal treatment1 treatment2
> class(fac)
[1] "factor"
> levels(fac)
[1] "normal"    "treatment1" "treatment2"
> nlevels(fac)
[1] 3
> class(levels(fac))
[1] "character"
```

Data structures in R

```
> #lists
> lis1 <- 1:5
> lis1
[1] 1 2 3 4 5
> lis2 <- factor(1:5)
```

```
> lis2
[1] 1 2 3 4 5
Levels: 1 2 3 4 5
> lis3 <- letters[1:5]
> lis3
[1] "a" "b" "c" "d" "e"
> combined_list <- list(lis1, lis2, lis3)
> combined_list
[[1]]
[1] 1 2 3 4 5

[[2]]
[1] 1 2 3 4 5
Levels: 1 2 3 4 5

[[3]]
[1] "a" "b" "c" "d" "e"

> combined_list[[3]]
[1] "a" "b" "c" "d" "e"
> combined_list[[3]][5]
[1] "e"
> flat_list <- unlist(combined_list)
> class(flat_list)
[1] "character"
> flat_list
 [1] "1" "2" "3" "4" "5" "1" "2" "3" "4" "5" "a" "b" "c" "d" "e"
> length(flat_list)
[1] 15
> #Vectors
> values <- c(88,65,90,40,65) #numeric vector
```

```
> class(values)
[1] "numeric"
> length(values)
[1] 5
> values[4]
[1] 40
> values[6] #Guess the result!!
[1] NA
> values[2:5] #slicing
[1] 65 90 40 65
> char_vector <- c("a", "b", "c") #character vector
> print(char_vector)
[1] "a" "b" "c"
> class(char_vector)
[1] "character"
> length(char_vector)
[1] 3
> char_vector[1:3]
[1] "a" "b" "c"
> char_num_vec <- c(1,2, "a") #what happens when a numeric and character values are mixed in a vector?
> class(char_num_vec)
[1] "character"
> vec <- c(1:1000) #vector using slicing
> vec[length(vec)]
[1] 1000
> vec[length(vec)/2]
[1] 500
> #Matrix
>#M <- matrix(vector, nrow=r, ncol=c, byrow=FALSE, dimnames=list(char_vector_rownames, char_vector_colnames))
```

```
> mat1 <- matrix(1:4, nrow = 2, ncol = 2) # create a matrix using a range of values
> mat1
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> mat1[1,2]
[1] 3
> mat1[2, ] #extract 2nd row
[1] 2 4
> mat1[,2 ] #extract 2nd column
[1] 3 4
> mat2 <- matrix(13:16, nrow = 2, ncol = 2)
> mat2
> mat1+mat2 #adding two matrices
     [,1] [,2]
[1,]   14   18
[2,]   16   20
> mat1 - mat2 #subtraction of two matrices
     [,1] [,2]
[1,]  -12  -12
[2,]  -12  -12
> 4 * mat1 #multiplication by a constant
     [,1] [,2]
[1,]    4   12
[2,]    8   16
> (mat1/mat2) #division
          [,1] [,2]
[1,] 0.07692308 0.20
[2,] 0.14285714 0.25
> M1 = matrix( c('AI','ML','DL','Tensorflow','Pytorch','Keras'), nrow = 2, ncol = 3, byrow = TRUE) # 2 rows and 3 columns
```

```
> print(M1)
     [,1]       [,2]     [,3]
[1,] "AI"       "ML"     "DL"
[2,] "Tensorflow" "Pytorch" "Keras"
> M2 = matrix( c('AI','ML','DL','Tensorflow','Pytorch','Keras'), nrow = 3, ncol = 2, byrow = TRUE) # 3 rows and 2 columns
> print(M2)
     [,1]     [,2]
[1,] "AI"     "ML"
[2,] "DL"     "Tensorflow"
[3,] "Pytorch" "Keras"
> M3 = matrix( c('AI','ML','DL','Tensorflow','Pytorch','Keras'), nrow = 2, ncol = 3, byrow = FALSE)# fill the matrix by column
> print(M3)
     [,1] [,2]       [,3]
[1,] "AI" "DL"       "Pytorch"
[2,] "ML" "Tensorflow" "Keras"
> M3[2,2]
[1] "Tensorflow"
> t(M3) #transpose a matrix
     [,1]     [,2]
[1,] "AI"     "ML"
[2,] "DL"     "Tensorflow"
[3,] "Pytorch" "Keras"
> #Array
> #my_array <- array(data, dim = (rows, colums, matrices, dimnames)
> v1 <- c(1,2,3)
> v2 <- c(4,5,6,7,8,9)
> result <- array(c(v1,v2),dim = c(3,3,2))
> result
, , 1
```

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

, , 2

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

> result <- array(c(v1,v2),dim = c(3,3,3)) #multidimentionality
> result
, , 1

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

, , 2

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

, , 3
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> #List
> num_list = c(3,4,5)
> char_list = c("a", "b", "c", "d", "e")
> logic_list = c(TRUE, TRUE, FALSE, TRUE)
> out_list = list(num_list, char_list, logic_list, 3)
> out_list
[[1]]
[1] 3 4 5

[[2]]
[1] "a" "b" "c" "d" "e"

[[3]]
[1]  TRUE  TRUE FALSE  TRUE

[[4]]
[1] 3
> #Data Frame
> gene_list = c("Gene 1", "Gene 2", "Gene 3")
> count_info = c(30,49,54)
> in_tissue = c(TRUE, FALSE, TRUE)
> data_frame = data.frame(gene_list, count_info, in_tissue)
> data_frame
  gene_list count_info in_tissue
1    Gene 1         30      TRUE
2    Gene 2         49     FALSE
3    Gene 3         54      TRUE
```

```
> gene_data <- data.frame(gene_list1 = c("ACE2", "BRCA2", "TP53"),
+                         snp = c(TRUE, FALSE, FALSE),
+                         count_data = c(13, 28, 31))
> gene_data
  gene_list1   snp count_data
1       ACE2  TRUE         13
2      BRCA2 FALSE         28
3       TP53 FALSE         31
> str(gene_data)
'data.frame':  3 obs. of  3 variables:
 $ gene_list1: chr  "ACE2" "BRCA2" "TP53"
 $ snp       : logi  TRUE FALSE FALSE
 $ count_data: num  13 28 31
```

Datasets

```
#experimenting with dataframes
> data(PlantGrowth)
> PlantGrowth
   weight group
1    4.17  ctrl
2    5.58  ctrl
3    5.18  ctrl
4    6.11  ctrl
5    4.50  ctrl
6    4.61  ctrl
7    5.17  ctrl
8    4.53  ctrl
9    5.33  ctrl
10   5.14  ctrl
11   4.81  trt1
```

```
12  4.17  trt1
13  4.41  trt1
14  3.59  trt1
15  5.87  trt1
16  3.83  trt1
17  6.03  trt1
18  4.89  trt1
19  4.32  trt1
20  4.69  trt1
21  6.31  trt2
22  5.12  trt2
23  5.54  trt2
24  5.50  trt2
25  5.37  trt2
26  5.29  trt2
27  4.92  trt2
28  6.15  trt2
29  5.80  trt2
30  5.26  trt2
```

File input

In this exercise we will be using the covid.txt dataset which is the data set related to SARS-CoV2 patients in different states collected as on 08.09.2021. The dataset has 8 columns with the first row as headings.

The dataset is as follows:

State/UTs	Total Cases	Active	Discharged	Deaths	Active Ratio (%)	Discharge Ratio (%)	Death Ratio (%)

Andaman and Nicobar	7584	8	7447	129	0.11	98.19	1.7
Andhra Pradesh	2030849	14652	2002187	14010	0.72	98.59	0.69
Arunachal Pradesh	53807	533	53004	270	0.99	98.51	0.5
Assam	595669	5615	584296	5758	0.94	98.09	0.97
Bihar	725833	77	716098	9658	0.01	98.66	1.33
Chandigarh	65160	32	64311	817	0.05	98.7	1.25
Chhattisgarh	1004902	376	990968	13558	0.04	98.61	1.35
Dadra and Nagar Haveli and Daman and Diu	10670	5	10661	4	0.05	99.92	0.04
Delhi	1438250	377	1412790	25083	0.03	98.23	1.74
Goa	174891	702	170972	3217	0.4	97.76	1.84
Gujarat	825629	161	815386	10082	0.02	98.76	1.22
Haryana	770659	351	760501	9807	0.05	98.68	1.27
Himachal Pradesh	215893	1521	210733	3639	0.7	97.61	1.69
Jammu and Kashmir	326990	1247	321329	4414	0.38	98.27	1.35
Jharkhand	348079	117	342829	5133	0.03	98.49	1.47
Karnataka	2962408	16269	2908622	37517	0.55	98.18	1.27
Kerala	4390489	209335	4158504	22650	4.77	94.72	0.52
Ladakh	20608	30	20371	207	0.15	98.85	1
Lakshadwee	10353	5	10297	51	0.05	99.46	0.49

p							
Madhya Pradesh	792353	131	781705	10517	0.02	98.66	1.33
Maharashtra	6500617	53427	6309021	138169	0.82	97.05	2.13
Manipur	117230	2617	112801	1812	2.23	96.22	1.55
Meghalaya	78359	1859	75141	1359	2.37	95.89	1.73
Mizoram	72883	13369	59273	241	18.34	81.33	0.33
Nagaland	30657	580	29432	645	1.89	96	2.1
Odisha	1016833	5919	1002810	8104	0.58	98.62	0.8
Puducherry	124836	860	122153	1823	0.69	97.85	1.46
Punjab	601072	309	584306	16457	0.05	97.21	2.74
Rajasthan	954198	90	945154	8954	0.01	99.05	0.94
Sikkim	30650	809	29464	377	2.64	96.13	1.23
Tamil Nadu	2635419	16522	2583707	35190	0.63	98.04	1.34
Telengana	661866	5253	652716	3897	0.79	98.62	0.59
Tripura	83672	485	82382	805	0.58	98.46	0.96
Uttar Pradesh	1709555	175	1686497	22883	0.01	98.65	1.34
Uttarakhand	343242	293	335560	7389	0.09	97.76	2.15
West Bengal	1557414	8096	1530731	18587	0.52	98.29	1.19

```
> covid <-read.delim("D:/workshop/covid.txt", header = TRUE)
> head(covid)
         State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andaman and Nicobar       7584      8      7447    129          0.11
2    Andhra Pradesh    2030849  14652   2002187  14010          0.72
3 Arunachal Pradesh      53807    533     53004    270          0.99
4             Assam     595669   5615    584296   5758          0.94
5             Bihar     725833     77    716098   9658          0.01
```

```
6       Chandigarh      65160    32     64311    817          0.05
  Discharge.Ratio.... Death.Ratio....
1            98.19         1.70
2            98.59         0.69
3            98.51         0.50
4            98.09         0.97
5            98.66         1.33
6            98.70         1.25
> tail(covid)
      State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
31    Tamil Nadu    2635419  16522    2583707  35190          0.63
32     Telengana     661866   5253     652716   3897          0.79
33       Tripura      83672    485      82382    805          0.58
34 Uttar Pradesh    1709555    175    1686497  22883          0.01
35   Uttarakhand     343242    293     335560   7389          0.09
36   West Bengal    1557414   8096    1530731  18587          0.52
  Discharge.Ratio.... Death.Ratio....
31            98.04         1.34
32            98.62         0.59
33            98.46         0.96
34            98.65         1.34
35            97.76         2.15
36            98.29         1.19
> covid <-read.delim("D:/workshop/covid.txt", header = FALSE)
> head(covid)
                V1       V2    V3       V4    V5           V6
1        State/UTs Total Cases Active Discharged Deaths Active Ratio (%)
2 Andaman and Nicobar       7584     8     7447    129          0.11
3     Andhra Pradesh    2030849  14652  2002187  14010          0.72
4   Arunachal Pradesh      53807   533    53004    270          0.99
5              Assam     595669  5615   584296   5758          0.94
```

```
6          Bihar     725833    77    716098   9658          0.01
             V7          V8
1 Discharge Ratio (%) Death Ratio (%)
2           98.19         1.7
3           98.59        0.69
4           98.51         0.5
5           98.09        0.97
6           98.66        1.33
> #covid <- read.delim("D:/workshop/covid.txt", header = TRUE, skip = 3) #skip first 3 lines
> head(covid)
         State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andaman and Nicobar       7584     8      7447    129          0.11
2    Andhra Pradesh    2030849  14652   2002187  14010          0.72
3  Arunachal Pradesh      53807    533     53004    270          0.99
4            Assam     595669   5615    584296   5758          0.94
5            Bihar     725833     77    716098   9658          0.01
6       Chandigarh      65160     32     64311    817          0.05
  Discharge.Ratio.... Death.Ratio....
1           98.19          1.70
2           98.59          0.69
3           98.51          0.50
4           98.09          0.97
5           98.66          1.33
6           98.70          1.25
> irisdata <- read.csv(url("http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"), header=FALSE)
> colnames(irisdata) <- c("sepal.length","sepal.width","petal.length","petal.width","species")
> head(irisdata)
  sepal.length sepal.width petal.length petal.width     species
```

```
1          5.1          3.5          1.4          0.2 Iris-setosa
2          4.9          3.0          1.4          0.2 Iris-setosa
3          4.7          3.2          1.3          0.2 Iris-setosa
4          4.6          3.1          1.5          0.2 Iris-setosa
5          5.0          3.6          1.4          0.2 Iris-setosa
6          5.4          3.9          1.7          0.4 Iris-setosa
> #dataframe indexing
> covid[2,3]   #value in second row, third column
[1] 14652
> covid[,1]     #first column, as a vector
 [1] "Andaman and Nicobar"
 [2] "Andhra Pradesh"
 [3] "Arunachal Pradesh"
 [4] "Assam"
 [5] "Bihar"
 [6] "Chandigarh"
 [7] "Chhattisgarh"
 [8] "Dadra and Nagar Haveli and Daman and Diu"
 [9] "Delhi"
[10] "Goa"
[11] "Gujarat"
[12] "Haryana"
[13] "Himachal Pradesh"
[14] "Jammu and Kashmir"
[15] "Jharkhand"
[16] "Karnataka"
[17] "Kerala"
[18] "Ladakh"
[19] "Lakshadweep"
[20] "Madhya Pradesh"
[21] "Maharashtra"
```

```
[22] "Manipur"
[23] "Meghalaya"
[24] "Mizoram"
[25] "Nagaland"
[26] "Odisha"
[27] "Puducherry"
[28] "Punjab"
[29] "Rajasthan"
[30] "Sikkim"
[31] "Tamil Nadu"
[32] "Telengana"
[33] "Tripura"
[34] "Uttar Pradesh"
[35] "Uttarakhand"
[36] "West Bengal"
> covid[2,]    #second row, as a data.frame
     State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
2 Andhra Pradesh    2030849  14652    2002187  14010          0.72
  Discharge.Ratio.... Death.Ratio....
2               98.59          0.69
> covid[,2:3] #second and third columns, as a data.frame
  Total.Cases Active
1        7584      8
2     2030849  14652
3       53807    533
4      595669   5615
5      725833     77
6       65160     32
7     1004902    376
8       10670      5
9     1438250    377
```

```
10     174891   702
11     825629   161
12     770659   351
13     215893  1521
14     326990  1247
15     348079   117
16    2962408 16269
17    4390489 209335
18      20608    30
19      10353     5
20     792353   131
21    6500617 53427
22     117230  2617
23      78359  1859
24      72883 13369
25      30657   580
26    1016833  5919
27     124836   860
28     601072   309
29     954198    90
30      30650   809
31    2635419 16522
32     661866  5253
33      83672   485
34    1709555   175
35     343242   293
36    1557414  8096
> covid[1]      #first column, as a data.frame
                    State.UTs
1              Andaman and Nicobar
2                  Andhra Pradesh
```

3	Arunachal Pradesh
4	Assam
5	Bihar
6	Chandigarh
7	Chhattisgarh
8	Dadra and Nagar Haveli and Daman and Diu
9	Delhi
10	Goa
11	Gujarat
12	Haryana
13	Himachal Pradesh
14	Jammu and Kashmir
15	Jharkhand
16	Karnataka
17	Kerala
18	Ladakh
19	Lakshadweep
20	Madhya Pradesh
21	Maharashtra
22	Manipur
23	Meghalaya
24	Mizoram
25	Nagaland
26	Odisha
27	Puducherry
28	Punjab
29	Rajasthan
30	Sikkim
31	Tamil Nadu
32	Telengana
33	Tripura

```
34                 Uttar Pradesh
35                  Uttarakhand
36                  West Bengal
> covid[1:5, c(3,5)]   #rows 1-5, columns 3 and 5
  Active Deaths
1      8    129
2  14652  14010
3    533    270
4   5615   5758
5     77   9658
> covid[,-1]    #everything but the first column
   Total.Cases Active Discharged Deaths Active.Ratio.... Discharge.Ratio....
1         7584      8      7447    129            0.11             98.19
2      2030849  14652   2002187  14010            0.72             98.59
3        53807    533     53004    270            0.99             98.51
4       595669   5615    584296   5758            0.94             98.09
5       725833     77    716098   9658            0.01             98.66
6        65160     32     64311    817            0.05             98.70
7      1004902    376    990968  13558            0.04             98.61
8        10670      5     10661      4            0.05             99.92
9      1438250    377   1412790  25083            0.03             98.23
10      174891    702    170972   3217            0.40             97.76
11      825629    161    815386  10082            0.02             98.76
12      770659    351    760501   9807            0.05             98.68
13      215893   1521    210733   3639            0.70             97.61
14      326990   1247    321329   4414            0.38             98.27
15      348079    117    342829   5133            0.03             98.49
16     2962408  16269   2908622  37517            0.55             98.18
17     4390489 209335   4158504  22650            4.77             94.72
18       20608     30     20371    207            0.15             98.85
19       10353      5     10297     51            0.05             99.46
```

20	792353	131	781705	10517	0.02	98.66
21	6500617	53427	6309021	138169	0.82	97.05
22	117230	2617	112801	1812	2.23	96.22
23	78359	1859	75141	1359	2.37	95.89
24	72883	13369	59273	241	18.34	81.33
25	30657	580	29432	645	1.89	96.00
26	1016833	5919	1002810	8104	0.58	98.62
27	124836	860	122153	1823	0.69	97.85
28	601072	309	584306	16457	0.05	97.21
29	954198	90	945154	8954	0.01	99.05
30	30650	809	29464	377	2.64	96.13
31	2635419	16522	2583707	35190	0.63	98.04
32	661866	5253	652716	3897	0.79	98.62
33	83672	485	82382	805	0.58	98.46
34	1709555	175	1686497	22883	0.01	98.65
35	343242	293	335560	7389	0.09	97.76
36	1557414	8096	1530731	18587	0.52	98.29

	Death.Ratio....
1	1.70
2	0.69
3	0.50
4	0.97
5	1.33
6	1.25
7	1.35
8	0.04
9	1.74
10	1.84
11	1.22
12	1.27
13	1.69

```
14          1.35
15          1.47
16          1.27
17          0.52
18          1.00
19          0.49
20          1.33
21          2.13
22          1.55
23          1.73
24          0.33
25          2.10
26          0.80
27          1.46
28          2.74
29          0.94
30          1.23
31          1.34
32          0.59
33          0.96
34          1.34
35          2.15
36          1.19
> covid[nrow(covid):1,]      #everything, with rows in reverse order
                State.UTs Total.Cases Active Discharged
36             West Bengal    1557414   8096    1530731
35             Uttarakhand     343242    293     335560
34           Uttar Pradesh    1709555    175    1686497
33                 Tripura      83672    485      82382
32               Telengana     661866   5253     652716
31              Tamil Nadu    2635419  16522    2583707
```

30	Sikkim	30650	809	29464
29	Rajasthan	954198	90	945154
28	Punjab	601072	309	584306
27	Puducherry	124836	860	122153
26	Odisha	1016833	5919	1002810
25	Nagaland	30657	580	29432
24	Mizoram	72883	13369	59273
23	Meghalaya	78359	1859	75141
22	Manipur	117230	2617	112801
21	Maharashtra	6500617	53427	6309021
20	Madhya Pradesh	792353	131	781705
19	Lakshadweep	10353	5	10297
18	Ladakh	20608	30	20371
17	Kerala	4390489	209335	4158504
16	Karnataka	2962408	16269	2908622
15	Jharkhand	348079	117	342829
14	Jammu and Kashmir	326990	1247	321329
13	Himachal Pradesh	215893	1521	210733
12	Haryana	770659	351	760501
11	Gujarat	825629	161	815386
10	Goa	174891	702	170972
9	Delhi	1438250	377	1412790
8	Dadra and Nagar Haveli and Daman and Diu	10670	5	10661
7	Chhattisgarh	1004902	376	990968
6	Chandigarh	65160	32	64311
5	Bihar	725833	77	716098
4	Assam	595669	5615	584296
3	Arunachal Pradesh	53807	533	53004
2	Andhra Pradesh	2030849	14652	2002187
1	Andaman and Nicobar	7584	8	7447

Deaths Active.Ratio.... Discharge.Ratio.... Death.Ratio....

36	18587	0.52	98.29	1.19
35	7389	0.09	97.76	2.15
34	22883	0.01	98.65	1.34
33	805	0.58	98.46	0.96
32	3897	0.79	98.62	0.59
31	35190	0.63	98.04	1.34
30	377	2.64	96.13	1.23
29	8954	0.01	99.05	0.94
28	16457	0.05	97.21	2.74
27	1823	0.69	97.85	1.46
26	8104	0.58	98.62	0.80
25	645	1.89	96.00	2.10
24	241	18.34	81.33	0.33
23	1359	2.37	95.89	1.73
22	1812	2.23	96.22	1.55
21	138169	0.82	97.05	2.13
20	10517	0.02	98.66	1.33
19	51	0.05	99.46	0.49
18	207	0.15	98.85	1.00
17	22650	4.77	94.72	0.52
16	37517	0.55	98.18	1.27
15	5133	0.03	98.49	1.47
14	4414	0.38	98.27	1.35
13	3639	0.70	97.61	1.69
12	9807	0.05	98.68	1.27
11	10082	0.02	98.76	1.22
10	3217	0.40	97.76	1.84
9	25083	0.03	98.23	1.74
8	4	0.05	99.92	0.04
7	13558	0.04	98.61	1.35
6	817	0.05	98.70	1.25

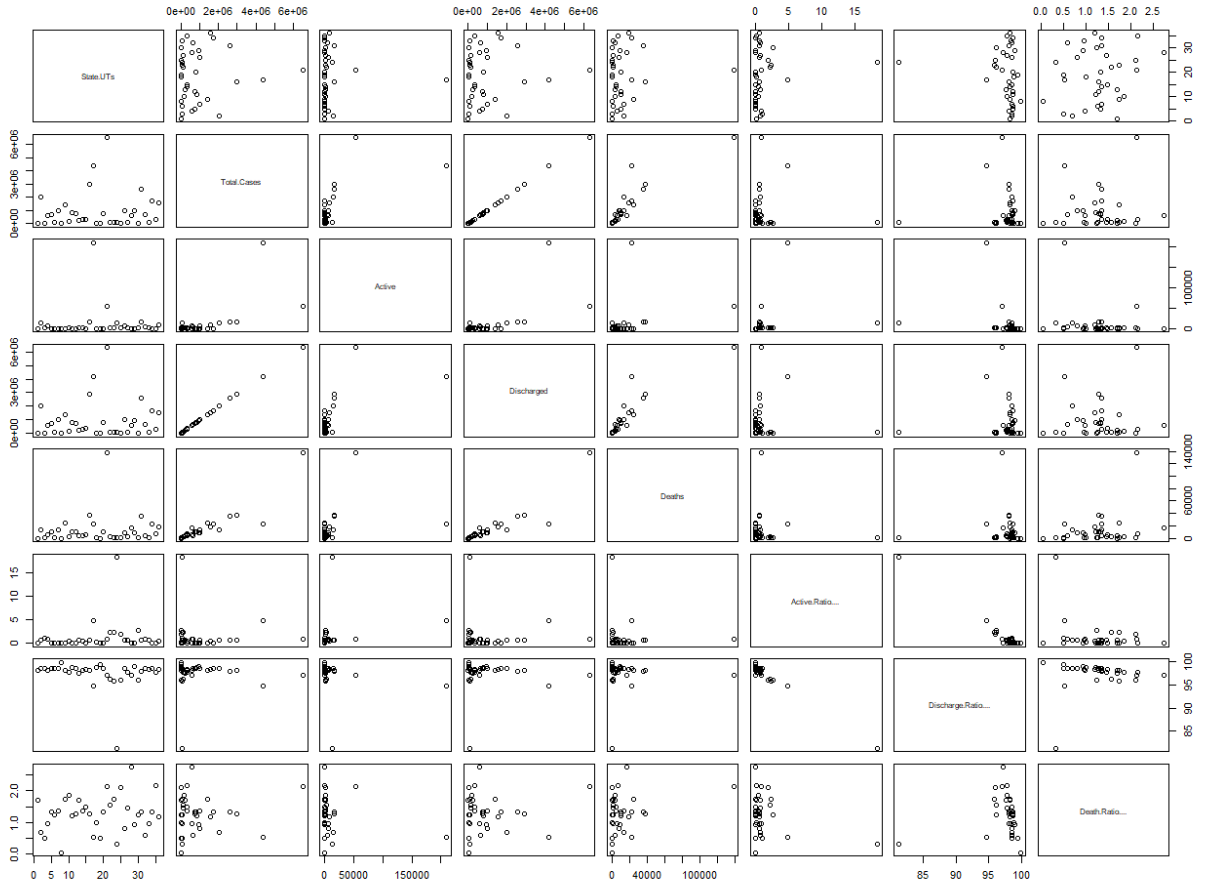
```
5   9658          0.01            98.66          1.33
4   5758          0.94            98.09          0.97
3    270          0.99            98.51          0.50
2  14010          0.72            98.59          0.69
1    129          0.11            98.19          1.70
> covid[covid[,2] < 10000,]          #rows of covid (with all columns) where the value in the second column is less than 10000
          State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andaman and Nicobar       7584     8      7447    129           0.11
  Discharge.Ratio.... Death.Ratio....
1               98.19           1.7
> covid$State.UTs    #State.UTs column, as a vector
 [1] "Andaman and Nicobar"
 [2] "Andhra Pradesh"
 [3] "Arunachal Pradesh"
 [4] "Assam"
 [5] "Bihar"
 [6] "Chandigarh"
 [7] "Chhattisgarh"
 [8] "Dadra and Nagar Haveli and Daman and Diu"
 [9] "Delhi"
[10] "Goa"
[11] "Gujarat"
[12] "Haryana"
[13] "Himachal Pradesh"
[14] "Jammu and Kashmir"
[15] "Jharkhand"
[16] "Karnataka"
[17] "Kerala"
[18] "Ladakh"
[19] "Lakshadweep"
```

```
[20] "Madhya Pradesh"
[21] "Maharashtra"
[22] "Manipur"
[23] "Meghalaya"
[24] "Mizoram"
[25] "Nagaland"
[26] "Odisha"
[27] "Puducherry"
[28] "Punjab"
[29] "Rajasthan"
[30] "Sikkim"
[31] "Tamil Nadu"
[32] "Telengana"
[33] "Tripura"
[34] "Uttar Pradesh"
[35] "Uttarakhand"
[36] "West Bengal"

> covid[,c("State.UTs", "Active")] #State.UTs and Active columns, as a data.frame
                       State.UTs Active
1             Andaman and Nicobar      8
2                  Andhra Pradesh  14652
3               Arunachal Pradesh    533
4                           Assam   5615
5                           Bihar     77
6                      Chandigarh     32
7                    Chhattisgarh    376
8  Dadra and Nagar Haveli and Daman and Diu      5
9                           Delhi    377
10                            Goa    702
11                        Gujarat    161
```

```
12                  Haryana    351
13         Himachal Pradesh   1521
14         Jammu and Kashmir   1247
15                Jharkhand    117
16                Karnataka  16269
17                   Kerala 209335
18                   Ladakh     30
19              Lakshadweep      5
20           Madhya Pradesh    131
21              Maharashtra  53427
22                  Manipur   2617
23                Meghalaya   1859
24                  Mizoram  13369
25                 Nagaland    580
26                   Odisha   5919
27               Puducherry    860
28                   Punjab    309
29                Rajasthan     90
30                   Sikkim    809
31               Tamil Nadu  16522
32                Telengana   5253
33                  Tripura    485
34            Uttar Pradesh    175
35              Uttarakhand    293
36              West Bengal   8096
> covid["10",]          #row named "10", as a data.frame
  State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
10      Goa     174891    702     170972   3217           0.4
  Discharge.Ratio.... Death.Ratio....
10               97.76          1.84
>    covid[order(covid$Active),    c("State.UTs",    "Total.Cases",    "Deaths",    "Active")]
```

```
#ordering according to active cases and displaying only 4 columns
                                    State.UTs Total.Cases Deaths Active
8  Dadra and Nagar Haveli and Daman and Diu      10670     4      5
19                                Lakshadweep     10353    51      5
1                        Andaman and Nicobar       7584   129      8
18                                   Ladakh      20608   207     30
6                                 Chandigarh     65160   817     32
5                                      Bihar    725833  9658     77
29                                 Rajasthan    954198  8954     90
15                                 Jharkhand    348079  5133    117
20                            Madhya Pradesh    792353 10517    131
11                                   Gujarat    825629 10082    161
34                             Uttar Pradesh   1709555 22883    175
35                               Uttarakhand    343242  7389    293
28                                    Punjab    601072 16457    309
12                                   Haryana    770659  9807    351
7                                Chhattisgarh   1004902 13558    376
9                                     Delhi    1438250 25083    377
33                                   Tripura     83672   805    485
3                           Arunachal Pradesh     53807   270    533
25                                 Nagaland      30657   645    580
10                                     Goa      174891  3217    702
30                                   Sikkim      30650   377    809
27                               Puducherry     124836  1823    860
14                          Jammu and Kashmir    326990  4414   1247
13                          Himachal Pradesh     215893  3639   1521
23                                 Meghalaya     78359  1359   1859
22                                   Manipur    117230  1812   2617
32                                 Telengana    661866  3897   5253
4                                     Assam     595669  5758   5615
26                                    Odisha   1016833  8104   5919
```

```
36                  West Bengal    1557414  18587   8096
24                      Mizoram      72883    241  13369
2                Andhra Pradesh    2030849  14010  14652
16                     Karnataka    2962408  37517  16269
31                     Tamil Nadu    2635419  35190  16522
21                    Maharashtra    6500617 138169  53427
17                      Kerala    4390489  22650 209335
> nrow(covid)
[1] 36
> ncol(covid)
[1] 8
> dim(covid)
[1] 36  8
> str(covid)
'data.frame':  36 obs. of  8 variables:
 $ State.UTs         : chr  "Andaman and Nicobar" "Andhra Pradesh" "Arunachal Pradesh"
"Assam" ...
 $ Total.Cases         : int  7584 2030849 53807 595669 725833 65160 1004902 10670
1438250 174891 ...
 $ Active          : int  8 14652 533 5615 77 32 376 5 377 702 ...
 $ Discharged          : int  7447 2002187 53004 584296 716098 64311 990968 10661
1412790 170972 ...
 $ Deaths          : int  129 14010 270 5758 9658 817 13558 4 25083 3217 ...
 $ Active.Ratio....   : num  0.11 0.72 0.99 0.94 0.01 0.05 0.04 0.05 0.03 0.4 ...
 $ Discharge.Ratio....: num  98.2 98.6 98.5 98.1 98.7 ...
 $ Death.Ratio....    : num  1.7 0.69 0.5 0.97 1.33 1.25 1.35 0.04 1.74 1.84 ...
> plot(covid)
```




```
> summary(covid)
 State.UTs         Total.Cases         Active          Discharged
Length:36          Min.   :  7584   Min.   :    5.0   Min.   :  7447
Class :character   1st Qu.: 76990   1st Qu.:  153.5   1st Qu.: 72434
Mode  :character   Median : 471874   Median :  556.5   Median : 463562
                   Mean   : 924711   Mean   : 10061.3   Mean   : 902338
                   3rd Qu.:1007885   3rd Qu.:  5343.5   3rd Qu.: 993928
                   Max.   :6500617   Max.   :209335.0   Max.   :6309021
    Deaths       Active.Ratio....  Discharge.Ratio....  Death.Ratio....
Min.   :    4   Min.   : 0.0100   Min.   :81.33        Min.   :0.040
1st Qu.:   814   1st Qu.: 0.0500   1st Qu.:97.72        1st Qu.:0.955
Median :  5446   Median : 0.4600   Median :98.28        Median :1.300
Mean   : 12311   Mean   : 1.1750   Mean   :97.56        Mean   :1.267
3rd Qu.: 13671   3rd Qu.: 0.7975   3rd Qu.:98.65        3rd Qu.:1.585
```

```
 Max.   :138169   Max.   :18.3400   Max.   :99.92      Max.   :2.740
> summary(covid$Total.Cases)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   7584   76990  471874  924711 1007885 6500617
> min(covid$Total.Cases)
[1] 7584
> max(covid$Total.Cases)
[1] 6500617
> sd(covid$Total.Cases)
[1] 1363114
> var(covid$Total.Cases)
[1] 1.85808e+12
> prod(covid$Total.Cases)
[1] 8.596361e+196
> sum(covid$Active)
[1] 362207
```

Data Wrangling

```
> #install.packages("dplyr")
> library(dplyr)
> #Selecting columns
> select_data <-select(covid, State.UTs, Total.Cases, Deaths)
> head(select_data)
         State.UTs Total.Cases Deaths
1 Andaman and Nicobar       7584    129
2     Andhra Pradesh    2030849  14010
3   Arunachal Pradesh      53807    270
4             Assam     595669   5758
5             Bihar     725833   9658
6        Chandigarh      65160    817
```

```
> head(select(covid, -Discharged)) #To select all the columns except a specific column
         State.UTs Total.Cases Active Deaths Active.Ratio....
1 Andaman and Nicobar       7584     8    129          0.11
2     Andhra Pradesh    2030849  14652  14010          0.72
3  Arunachal Pradesh      53807    533    270          0.99
4            Assam     595669   5615   5758          0.94
5            Bihar     725833     77   9658          0.01
6       Chandigarh      65160     32    817          0.05
  Discharge.Ratio.... Death.Ratio....
1           98.19          1.70
2           98.59          0.69
3           98.51          0.50
4           98.09          0.97
5           98.66          1.33
6           98.70          1.25
> head(select(covid, State.UTs:Deaths)) #To select a range of columns
         State.UTs Total.Cases Active Discharged Deaths
1 Andaman and Nicobar       7584     8      7447    129
2     Andhra Pradesh    2030849  14652    2002187  14010
3  Arunachal Pradesh      53807    533     53004    270
4            Assam     595669   5615    584296   5758
5            Bihar     725833     77    716098   9658
6       Chandigarh      65160     32     64311    817
> head(select(covid, starts_with("D")))
  Discharged Deaths Discharge.Ratio.... Death.Ratio....
1     7447    129           98.19          1.70
2    2002187  14010           98.59          0.69
3     53004    270           98.51          0.50
4    584296   5758           98.09          0.97
5    716098   9658           98.66          1.33
6     64311    817           98.70          1.25
```

```
> head(select(covid, ends_with("s")))
          State.UTs Total.Cases Deaths
1 Andaman and Nicobar       7584    129
2     Andhra Pradesh    2030849  14010
3   Arunachal Pradesh      53807    270
4             Assam     595669   5758
5             Bihar     725833   9658
6        Chandigarh      65160    817
> head(select(covid, contains("Ratio")))
  Active.Ratio.... Discharge.Ratio.... Death.Ratio....
1             0.11               98.19            1.70
2             0.72               98.59            0.69
3             0.99               98.51            0.50
4             0.94               98.09            0.97
5             0.01               98.66            1.33
6             0.05               98.70            1.25
> head(select(covid, contains("hs")))
  Deaths
1    129
2  14010
3    270
4   5758
5   9658
6    817
> #Filtering rows
> filter(covid, Deaths >= 16000)
     State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1       Delhi    1438250    377    1412790  25083            0.03
2   Karnataka    2962408  16269    2908622  37517            0.55
3      Kerala    4390489 209335    4158504  22650            4.77
4 Maharashtra    6500617  53427    6309021 138169            0.82
```

```
5      Punjab     601072   309    584306  16457          0.05
6   Tamil Nadu    2635419  16522   2583707  35190          0.63
7 Uttar Pradesh    1709555   175   1686497  22883          0.01
8  West Bengal    1557414   8096   1530731  18587          0.52
  Discharge.Ratio.... Death.Ratio....
1            98.23          1.74
2            98.18          1.27
3            94.72          0.52
4            97.05          2.13
5            97.21          2.74
6            98.04          1.34
7            98.65          1.34
8            98.29          1.19
> filter(covid, Active >= 10000, Deaths >= 10000)
      State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andhra Pradesh    2030849  14652    2002187  14010          0.72
2     Karnataka    2962408  16269    2908622  37517          0.55
3        Kerala    4390489 209335    4158504  22650          4.77
4   Maharashtra    6500617  53427    6309021 138169          0.82
5   Tamil Nadu    2635419  16522    2583707  35190          0.63
  Discharge.Ratio.... Death.Ratio....
1            98.59          0.69
2            98.18          1.27
3            94.72          0.52
4            97.05          2.13
5            98.04          1.34
#Pipe operator: %>%
> covid %>%
+   select(State.UTs, Total.Cases, Deaths) %>%
+   head
         State.UTs Total.Cases Deaths
```

```
1 Andaman and Nicobar       7584   129
2    Andhra Pradesh    2030849  14010
3  Arunachal Pradesh      53807   270
4           Assam     595669   5758
5           Bihar     725833   9658
6       Chandigarh      65160   817

> covid %>% arrange(Active) %>% head
                        State.UTs Total.Cases Active Discharged
1 Dadra and Nagar Haveli and Daman and Diu      10670      5      10661
2                      Lakshadweep      10353      5      10297
3                Andaman and Nicobar       7584      8       7447
4                           Ladakh      20608     30      20371
5                       Chandigarh      65160     32      64311
6                            Bihar     725833     77     716098
  Deaths Active.Ratio.... Discharge.Ratio.... Death.Ratio....
1      4            0.05             99.92            0.04
2     51            0.05             99.46            0.49
3    129            0.11             98.19            1.70
4    207            0.15             98.85            1.00
5    817            0.05             98.70            1.25
6   9658            0.01             98.66            1.33
> covid %>%
+   select(State.UTs, Total.Cases, Deaths) %>%
+   arrange(Deaths, Total.Cases) %>%
+   head
                        State.UTs Total.Cases Deaths
1 Dadra and Nagar Haveli and Daman and Diu      10670      4
2                      Lakshadweep      10353     51
3                Andaman and Nicobar       7584    129
4                           Ladakh      20608    207
```

```
5                    Mizoram      72883    241
6              Arunachal Pradesh      53807    270
>
> covid %>%
+   select(State.UTs, Total.Cases, Deaths) %>%
+   arrange(Total.Cases, Deaths) %>%
+   filter(Deaths <= 250)
                      State.UTs Total.Cases Deaths
1              Andaman and Nicobar       7584    129
2                   Lakshadweep      10353     51
3 Dadra and Nagar Haveli and Daman and Diu      10670      4
4                     Ladakh      20608    207
5                    Mizoram      72883    241
>
> covid %>%
+   mutate(Ratio = Active / Total.Cases) %>%
+   head
        State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andaman and Nicobar       7584      8      7447    129          0.11
2    Andhra Pradesh    2030849  14652   2002187  14010          0.72
3  Arunachal Pradesh      53807    533     53004    270          0.99
4             Assam     595669   5615    584296   5758          0.94
5             Bihar     725833     77    716098   9658          0.01
6        Chandigarh      65160     32     64311    817          0.05
  Discharge.Ratio.... Death.Ratio....       Ratio
1               98.19           1.70 0.0010548523
2               98.59           0.69 0.0072147166
3               98.51           0.50 0.0099057743
4               98.09           0.97 0.0094263761
5               98.66           1.33 0.0001060850
6               98.70           1.25 0.0004910988
```

```
> glimpse(covid)
Rows: 36
Columns: 8
$ State.UTs         <chr> "Andaman and Nicobar", "Andhra Pradesh", "Arunach~
$ Total.Cases       <int> 7584, 2030849, 53807, 595669, 725833, 65160, 1004~
$ Active            <int> 8, 14652, 533, 5615, 77, 32, 376, 5, 377, 702, 16~
$ Discharged        <int> 7447, 2002187, 53004, 584296, 716098, 64311, 9909~
$ Deaths            <int> 129, 14010, 270, 5758, 9658, 817, 13558, 4, 25083~
$ Active.Ratio....  <dbl> 0.11, 0.72, 0.99, 0.94, 0.01, 0.05, 0.04, 0.05, 0~
$ Discharge.Ratio.... <dbl> 98.19, 98.59, 98.51, 98.09, 98.66, 98.70, 98.61, ~
$ Death.Ratio....   <dbl> 1.70, 0.69, 0.50, 0.97, 1.33, 1.25, 1.35, 0.04, 1~
> summarise(covid, mean = mean(Deaths))
     mean
1 12311.47
> summarise(covid, mean = mean(Deaths))
     mean
1 12311.47
> summarise(covid, min = min(Deaths))
  min
1   4
> summarise(covid, max = max(Deaths))
    max
1 138169
> summarise(covid, med = median(Deaths))
    med
1 5445.5
#random sampling
> # Printing three rows
> sample_n(covid, 3) #3 random samples
       State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1      Karnataka    2962408  16269    2908622  37517          0.55
```

```
2     West Bengal    1557414   8096    1530731  18587          0.52
3 Himachal Pradesh     215893   1521     210733   3639          0.70
  Discharge.Ratio.... Death.Ratio....
1               98.18          1.27
2               98.29          1.19
3               97.61          1.69
> sample_n(covid, 3) #sample again
   State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1    Punjab     601072    309     584306  16457           0.05
2 Tamil Nadu    2635419  16522    2583707  35190           0.63
3 Puducherry     124836    860     122153   1823           0.69
  Discharge.Ratio.... Death.Ratio....
1               97.21          2.74
2               98.04          1.34
3               97.85          1.46
>
> # Printing 50 % of the rows
> sample_frac(covid, 0.10)
          State.UTs Total.Cases Active Discharged Deaths Active.Ratio....
1 Andaman and Nicobar       7584      8       7447    129           0.11
2         Telengana     661866   5253     652716   3897           0.79
3     Uttar Pradesh    1709555    175    1686497  22883           0.01
4       Lakshadweep      10353      5      10297     51           0.05
  Discharge.Ratio.... Death.Ratio....
1               98.19          1.70
2               98.62          0.59
3               98.65          1.34
4               99.46          0.49
```

2. CONTROL STATEMENTS IN R

Divya P Syamaladevi
ICAR-Indian Institute of Spices Research
Email: P.Syamaladevi@icar.gov.in

Participants may open R or go to google colab by clicking on the link below
https://colab.research.google.com/notebook#create=true&language=r

Lists are the R objects which contain elements of different types like – numbers, strings, vectors etc. A list can also contain a matrix or a function as its elements. List is created using list() function.

Creating a List

Following is an example to create a list

Example 1

```
list_data1<- list("Red", "Green", "Blue")
print (list_data1)
list_data2 <- list(1, 2)
print(list_data2)
```

Control Statements

In R programming, there are 8 types of control statements: for loop, if condition, if-else condition, repeat and break statement, next statement, nested loops, while loop and return statement

For loop

A for loop is used for iterating over a sequence. With the for loop we can execute a set of statements, once for each item in a vector, array, list, etc.

Syntax:

```
for (expression){
   statements
   ....
   ....
}
```

Example 1: Print every item in a list:

```
fruits <- list ("apple", "banana", "cherry")

for (x in fruits) {
  print(x)
}
```

Example 2

```
dice <- c(1, 2, 3, 4, 5, 6)

for (x in dice) {
  print(x)
}
```

If Condition:

This control structure checks the expression provided in parenthesis is true or not. If true, the execution of the statements in braces {} continues.

Syntax:

```
if(expression){
   statements}
```

```
fruits <- list("apple", "banana", "cherry", "Cherry")

for (x in fruits) {
  if (x == "cherry") {
    print (x)
  }
}
```

If-else condition

It is similar to **if** condition but when the test **expression** in if condition fails, then statements in **else** condition are executed.

Syntax:

```
if(expression){
   statements
   ....
   ....}
else{
   statements
   ....
   ....}
```

```
fruits <- list("apple", "banana", "tomato")
for (x in fruits) {
  if (x == "tomato") {
    print(paste("found a vegetable"))
    }
else{print(x)}}
```

Break

With the break statement, we can stop the loop before it has looped through all the items:

```
fruits <- list("apple", "banana", "cherry")

for (x in fruits) {
  if (x == "cherry") {
    break
  }
  print(x)
}
```

Next

With the next statement, we can skip an iteration without terminating the loop:

```
fruits <- list("apple", "banana", "cherry")

for (x in fruits) {
  if (x == "banana") {
    next
  }
```

```
  print(x)
}
```

Nested loop: You can also have a loop inside of a loop which is called a nested loop

```
adj <- list("red", "big", "tasty")

fruits <- list("apple", "banana", "cherry")
  for (x in adj) {
    for (y in fruits) {
      print(paste(x, y))
  }
}
```

3. BASICS STATISTICS USING R

Dr. Sreekumar J

ICAR-Central Tuber Crops Research Institute
Email: sreejyothi_in@yahoo.com

```
install.packages ("dplyr")
install.packages("ggpubr")
library(dplyr)
library(foreign)
library(readxl)
library(Hmisc)
nitya <- read_excel("/media/bsl/708E77378E76F546/nitya/nitya1control.xlsx")
View(nitya)
dataset1 <- read.table("~/Documents/dataset1.txt", header=TRUE, quote="\"")
ttestdata<-read.table("~/Documents/ttestdata.txt", header=TRUE, quote="\"")
names(ttestdata)

# Read dataset Pradeepika1 : Use import dataset

is.data.frame(pradeepika1)
dim(pradeepika1)
names(pradeepika1)

attach(pradeepika1)
names(pradeepika1)
#basic summary statistics
summary(pradeepika1)
mean1=mean(pradeepika1$BulkDensity)
mean1
sd1=sd(pradeepika1$BulkDensity)
sd1
cv1=sd1/mean1*100
cv1
```

```
meadian2=median(pradeepika1$Ragflour)
meadian2
v4<-var(pradeepika1$WAP)
v4
```

Tools for Descriptive Statistics

```
install.packages("DescTools")
install.packages("Hmisc")
library(DescTools)

mean(pradeepika1$BulkDensity)
median(pradeepika1$BulkDensity)
Mode(pradeepika1$BulkDensity)
sd(pradeepika1$BulkDensity)
var(pradeepika1$BulkDensity)
CoefVar(pradeepika1$BulkDensity)
Skew(pradeepika1$BulkDensity)
Kurt(pradeepika1$BulkDensity)
Desc(pradeepika1$BulkDensity)
```

Histogram

```
library(Hmisc)
hist(dataset1, col="red")
```

Density Plot

```
dens =density(dataset1$v1)
plot(dens)
```

Box Plot

```
boxplot(dataset1)
# boxplot more examples

boxplot(len ~ dose, data = ToothGrowth,
      boxwex = 0.25, at = 1:3 - 0.2,
      subset = supp == "VC", col = "yellow",
      main = "Guinea Pigs' Tooth Growth",
      xlab = "Vitamin C dose mg",
      ylab = "tooth length",
      xlim = c(0.5, 3.5), ylim = c(0, 35), yaxs = "i")
boxplot(len ~ dose, data = ToothGrowth, add = TRUE,
      boxwex = 0.25, at = 1:3 + 0.2,
      subset = supp == "OJ", col = "orange")
legend(2, 9, c("Ascorbic acid", "Orange juice"),
     fill = c("yellow", "orange"))

 ## With less effort (slightly different) using factor *interaction*:

boxplot(len ~ dose:supp, data = ToothGrowth,
      boxwex = 0.5, col = c("orange", "yellow"),
      main = "Guinea Pigs' Tooth Growth",
      xlab = "Vitamin C dose mg", ylab = "tooth length",
      sep = ":", lex.order = TRUE, ylim = c(0, 35), yaxs = "i")

## more examples in help(bxp)
```

Tidying your datasets

```
library(tidyverse)
```

```
SeasonalTemps <- data.frame(Year = c(2015, 2016, 2017, 2018),
                            Winter = c(40, 38, 42, 44),
                            Spring = c(46, 40, 50, 48),
                            Summer = c(70, 62, 81, 76),
                            Fall = c(52, 46, 54, 56))
SeasonalTemps

ggplot(SeasonalTemps, aes(x = Year)) +
  geom_line(aes(y = Winter), color = "purple") +
  geom_line(aes(y = Spring), color = "green") +
  geom_line(aes(y = Summer), color = "blue") +
  geom_line(aes(y = Fall), color = "red")

LongTemps <- gather(data = SeasonalTemps, key = Season, value = AvgTemp, -Year)
LongTemps

ggplot(LongTemps, aes(x = Year, y = AvgTemp, color = Season)) +
  geom_line()


WideTemps <- spread(LongTemps, Season, AvgTemp)
WideTemps

OrderWideTemps <- select(WideTemps, c(Year, Winter, Spring, Summer, Fall))
OrderWideTemps

select(WideTemps, c(Year, Winter, Spring, Fall))



hist(dataset1, col="red")
```

```
boxplot(count ~ spray, data = InsectSprays, col = "lightgray")

boxplot(decrease ~ treatment, data = OrchardSprays, col = "bisque",
      log = "y")
boxplot(decrease ~ treatment, data = OrchardSprays, col = "bisque",
      log = "x", horizontal=TRUE)



rb <- boxplot(decrease ~ treatment, data = OrchardSprays, col = "bisque")
title("Comparing boxplot()s and non-robust mean +/- SD")
mn.t <- tapply(OrchardSprays$decrease, OrchardSprays$treatment, mean)
sd.t <- tapply(OrchardSprays$decrease, OrchardSprays$treatment, sd)
xi <- 0.3 + seq(rb$n)
points(xi, mn.t, col = "orange", pch = 18)
arrows(xi, mn.t - sd.t, xi, mn.t + sd.t,
     code = 3, col = "pink", angle = 75, length = .1)
```

One Sample T Test

```
t.test (x, mu = 0, alternative = "two.sided")

# Two sample t-test for testing two sample means, samples independent
x = rnorm(10)
y = rnorm(10)
t.test(x,y)
ttest = t.test(x,y)
names(ttest)
ttest$statistic
```

Two Sample T-Test

```
# Two sample t-test for testing two sample means, samples independent, homogeneous variance

set.seed(0)
ClevelandSpending <- rnorm(50, mean = 250, sd = 75)
NYSpending <- rnorm(50, mean = 300, sd = 80)
t.test(ClevelandSpending, NYSpending, var.equal = TRUE)



# Two sample t-test for testing two sample means, samples independent, variance unequal.

var.test(ClevelandSpending, NYSpending)
t.test(ClevelandSpending, NYSpending, var.equal = FALSE)

# Paired  t-test for testing two sample means, samples dependent

t.test(x, y, paired=TRUE)
set.seed(2820)
preTreat <- c(rnorm(1000, mean = 145, sd = 9))
postTreat <- c(rnorm(1000, mean = 138, sd = 8))
t.test(preTreat, postTreat, paired = TRUE)
```

Correlation

```
correlation<-read.table("~/Documents/correlation.txt", header=TRUE, quote="\"")
attach(correlation)
names(correlation)

# Correlation among all possible pairs of variables
cor(correlation, method="pearson")
```

```
# Correlation between specific pairs of variables
corr=cor.test(pp,ph)
str(corr)

# Testing the significance of Correlation Coefficient

corr$statistic
corr$p.value
corr$conf.int
detach(correlation)
```

Regression

```
# R-Script for Regression Analysis : To test the average relationship, to predict the defendant variable for unit change in
# independent variables



attach(reg)
names(reg)
lm1 =lm(Y~X1+X2+X3, reg)
summary(lm1)
plot(lm1)



#Test for Non-Constant Variance - quick way from fitted values as in lm1
summary(lm(abs(residuals(lm1)) ~ fitted(lm1)))

# Testing the Normality Residuals Using Shapiro-Wilk Test
hist(residuals(lm1))
```

```
boxplot(residuals(lm1))
shapiro.test(residuals(lm1))

#test of independence of residuals using Durbin Watson test
library("lmtest")
dwtest(lm1)

#influential observations using Hat matrix

influence1 =influence(lm1)
summary(influence1)
summary(influence1$hat)

# Looking for multi-collinearity
library(car)
vif(lm1)
detach(reg)

# Regression
x = c(18,23,25,35,65,54,34,56,72,19,23,42,18,39,37)
y = c(202,186,187,180,156,169,174,172,153,199,193,174,198,183,178)

plot(x,y) # make a plot
abline(lm(y ~ x)) # plot the regression line
lm(y~x)
lm.result=simple.lm(x,y)
```

Analysis of Variance (ANOVA)

```
model <-aov (yield ~ fertilizer, data = field)
```

```
size <- c(3,4,5,6,4,5,6,7,7,8,9,10)
pop <- c("A","A","A","A","B","B","B","B","C","C","C","C")
lm.model <- lm(size ~ pop)
summary(lm.model)
anova(lm.model)

# Anova using aov

aov.model <- aov(size ~ pop)
summary(aov.model)
TukeyHSD(aov.model)
```

RBD, CRD, LSD

```
install.packages("agricolae")
library(agricolae)



# Creating CRD
str(design.crd)
trt <- c("A", "B", "C")
repeticion <- c(4, 3, 4)
outdesign <- design.crd(trt,r=repeticion,seed=777,serie=0)
book1 <- outdesign$book

# Writing the design to excel
write.csv(book1,"book1.csv",row.names=FALSE)

# Creating RBD
str(design.rcbd)
function (trt, r, serie = 2, seed = 0, kinds = "Super-Duper",
```

```
        first = TRUE, continue = FALSE, randomization = TRUE)
trt <- c("A", "B", "C","D","E")
repeticion <- 4
outdesign <- design.rcbd(trt,r=repeticion, seed=-513, serie=2)

# book2 <- outdesign$book

book2<- zigzag(outdesign) # zigzag numeration
print(outdesign$sketch)

# R-Script for CRD

attach(crd)
names(crd)
trt=factor(trt)
lm1=lm(yield~trt)
anova(lm1)

library(lsmeans)
lsm=lsmeans(lm1,"trt")
lsm
pairs(lsm)

#pairs statement is an optional statement
#to provide letters for groups, need to install multcompView
library(multcompView)
cld(lsm,Letters="abc")
detach(crd)

# R-Script for RCB Design (it is same for any block design)
```

```
attach(rbd)
names(rbd)
trt=factor(trt)
blk=factor(blk)
lm1=lm(yield~trt+blk)
anova(lm1)
library(lsmeans)
lsm=lsmeans(lm1,"trt")
lsm
pairs(lsm)

# Pairs statement is an optional statement
# to provide letters for groups, need to install multcompView

library(multcompView)
cld(lsm, Letters="abcdefghij")
detach(rbd)

# LSD Multiple comparison

LSD.test(y, trt, DFerror, MSerror, alpha = 0.05, p.adj=c("none","holm","hommel",
"hochberg", "bonferroni", "BH", "BY", "fdr"), group=TRUE, main =
NULL,console=FALSE)

# LSD Example

library(agricolae)
data(sweetpotato)
model<-aov(yield~virus, data=sweetpotato)
out <- LSD.test(model,"virus", p.adj="bonferroni")
```

```
#stargraph
# Variation range: max and min

plot(out)

#endgraph
# Old version LSD.test()
df<-df.residual(model)
MSerror<-deviance(model)/df
out <- with(sweetpotato,LSD.test(yield,virus,df,MSerror))

#stargraph
# Variation interquartil range: Q75 and Q25

plot(out,variation="IQR")

#endgraph

out<-LSD.test(model,"virus",p.adj="hommel",console=TRUE)
plot(out,variation="SD") # variation standard deviation
# }
```

Split-plot analysis

```
# split-plot analysis

library(agricolae)
data(plots)
str(plots)
plots[,1] <-as.factor(plots[,1])
```

```
model <- aov(yield ~ block + A + Error(plot)+ B + A:B, data=plots)
summary(model)
attach(plots)
b<-nlevels(B)
a<-nlevels(A)
r<-nlevels(block)
dfa <- df.residual(model$plot)
Ea <-deviance(model$plot)/dfa
dfb <- df.residual(model$Within)
Eb <-deviance(model$Within)/dfb
Eab <- (Ea +(b-1)*Eb)/(b*r)
#dfab<-(Ea +(b-1)*Eb)^2/(Ea^2/dfa +((b-1)*Eb)^2/dfb)
# Comparison A, A(b1), A(b2), A(b3)
title1= "CD at 5% for main plot means"
comparison1 <-LSD.test(yield,A,dfa,Ea)
title1
Comparison1
title2="CD at 5% for sub plot means"
comparison2 <-LSD.test(yield,B,dfb,Eb)
title2
comparison2
title3="CD at 5% for sub plot means at same level main plot"
sd3=sqrt(2*Eb/r)
cd3=qt(0.975,dfb)*sd3
title3
cd3
#comparison6 <-LSD.test(yield[A=="a1"],B[A=="a1"],dfb,Eb)
#comparison7 <-LSD.test(yield[A=="a2"],B[A=="a2"],dfb,Eb)
#comparison8 <-LSD.test(yield[B=="b1"],A[B=="b2"],dfab,Eab)
title4= "calculation Main Plot Means at same or different levels of subplot treatment"
```

```
ab <- sqrt(2*((b-1)*Eb+Ea)/(b*r))
tval <- ((b-1)*Eb*qt(0.975,dfb)+Ea*qt(0.975,dfa))/((b-1)*Eb + Ea)
title4=
CD4 <- ab*tval
title4
CD4
plots
detach(plots)
```

AMMI (Genotype Environment Interaction)

```
data(plrv)
model<- with(plrv,AMMI(Locality, Genotype, Rep, Yield, console=FALSE))
model$ANOVA
# see help(plot.AMMI)
# biplot
plot(model)

# triplot PC 1,2,3
plot(model, type=2, number=TRUE)

# biplot PC1 vs Yield
plot(model, first=0,second=1, number=TRUE)

# Example 2

data(CIC)
data1<-CIC$comas[,c(1,6,7,17,18)]
data2<-CIC$oxapampa[,c(1,6,7,19,20)]
cic <- rbind(data1,data2)
model<-with(cic,AMMI(Locality, Genotype, Rep, relative))
```

```
model$ANOVA
plot(model,0,1,angle=20,ecol="brown")

# Example 3
# Only means. Mean square error is well-known.

data(sinRepAmmi)
REP <- 3
MSerror <- 93.24224

#startgraph
model<-with(sinRepAmmi,AMMI(ENV, GEN, REP, YLD, MSerror,PC=TRUE))

# print anova
print(model$ANOVA,na.print = "")

# Biplot with the one restored observed.

plot(model,0,1,type=1)

# with principal components model$PC is class "princomp"

pc<- model$PC
pc$loadings
summary(pc)
biplot(pc)

# Principal components by means of the covariance similar AMMI
# It is to compare results with AMMI

cova<-cov(model$genXenv)
```

```
values<-eigen(cova)
total<-sum(values$values)
round(values$values*100/total,2)

# AMMI: 64.81 18.58 13.50  3.11  0.00
# }
```

Principal Component Analysis

```
library(readxl)
Nitya1stress <- read_excel("~/Documents/nitya1stress.xlsx")
View(Nitya1stress)

newdata<-data.frame(Nitya1stress,row.names=Nitya1stress$Genotypes)
rownames(newdata)

#   PCA Analysis
Nitya1stress.pca <- prcomp(Nitya1stress[2:15],center = TRUE,scale. = TRUE)

devtools::install_github("hadley/devtools")
library(devtools)

install_github("vqv/ggbiplot")
library(ggbiplot)

g<-
ggbiplot(Nitya1stress.pca,obs.scale=1,var.scale=1,labels=rownames(newdata),labels.size
=3)
g<-
g+theme(axis.title=element_text(face="bold",size=12),axis.text=element_text(face="bo
ld",size=10))

write.csv(Nitya1stress.pca$x,file="C:/Users/SREEWS/Desktop/nitya/nitya1stressloadi
```

```
ngs.csv")
summary(Nitya1stress.pca)


# biplot(prcomp(Nitya1stress[2:15]),center = TRUE,scale. = TRUE)
#library(ggfortify)
#df<-Nitya1stress[2:15]
#autoplot(prcomp(df),data=Nitya1stress, label=TRUE,label.size=4,shape=6)
```

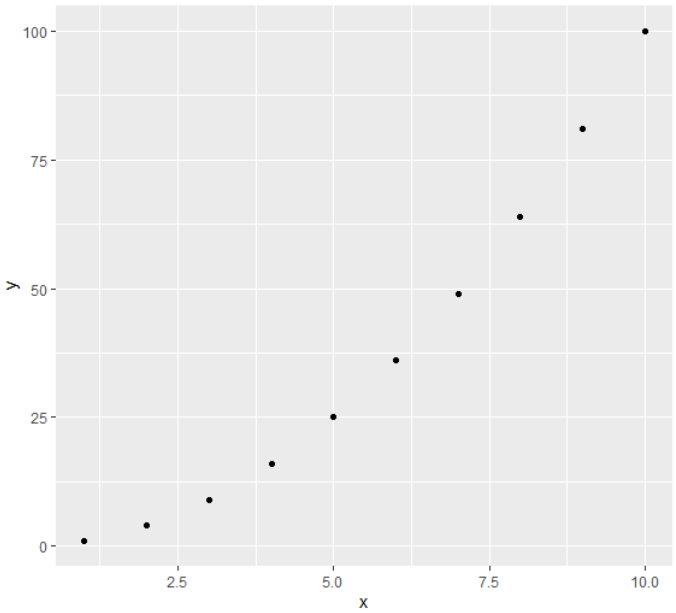
4. BASICS OF GGPLOT

Sona Charles,
ICAR-Indian Institute of Spices Research
Email: sona.charles@icar.gov.in

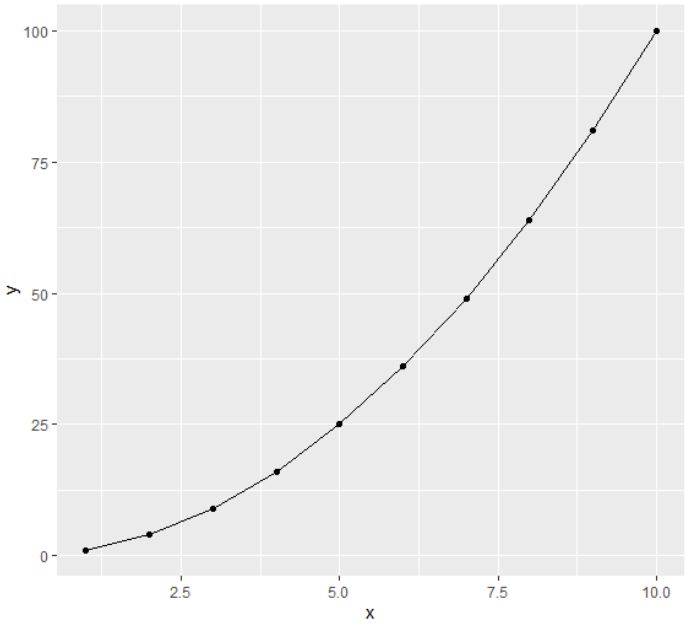
```
> install.packages("ggplot2")
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.0/ggplot2_3.3.5.zip'
Content type 'application/zip' length 4129871 bytes (3.9 MB)
downloaded 3.9 MB

package ‘ggplot2’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in <PATH>
> library(ggplot2)
Warning message:
package ‘ggplot2’ was built under R version 4.0.5
```

Your First quick ggplot!

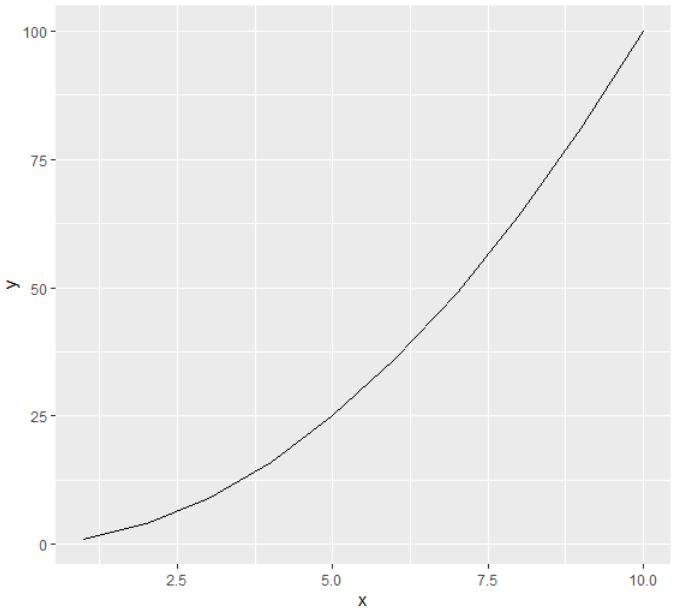
```
> x <- 1:10
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> y = x*x
> y
 [1]   1   4   9  16  25  36  49  64  81 100
> qplot(x,y)
```



> qplot(x, y, geom=c("line", "point"))



> qplot(x, y, geom=c("line"))



Scatterplots

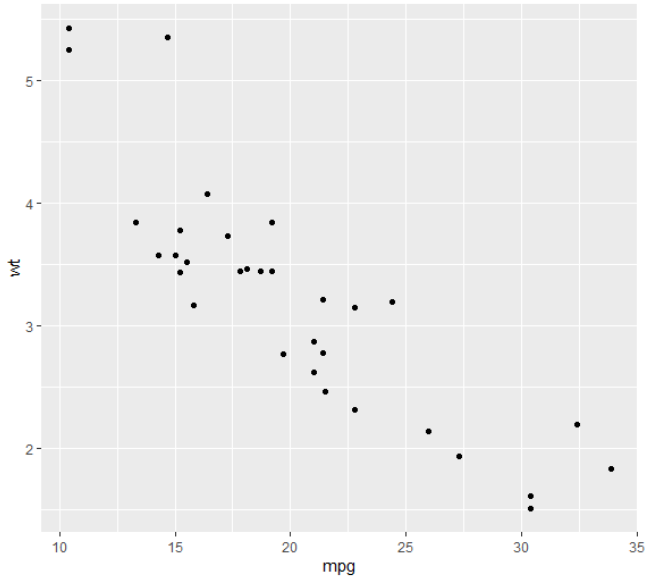
Dataset: mtcars **(**Motor Trend Car Road Tests)

Description: The data comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973 - 74 models).

Format: A data frame with 32 observations on 3 variables.

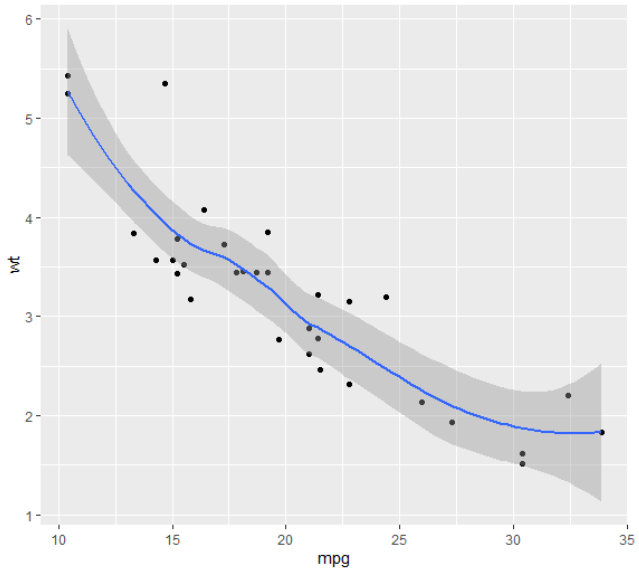
```
> data(mtcars)
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    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
> df <- mtcars[, c("mpg", "cyl", "wt")]
> head(df)
```

```
                  mpg cyl    wt
Mazda RX4         21.0   6 2.620
Mazda RX4 Wag     21.0   6 2.875
Datsun 710        22.8   4 2.320
Hornet 4 Drive    21.4   6 3.215
Hornet Sportabout 18.7   8 3.440
Valiant           18.1   6 3.460
> qplot(mpg, wt, data=mtcars)
```



The option “smooth” is used to add a smoothed line with its standard error.

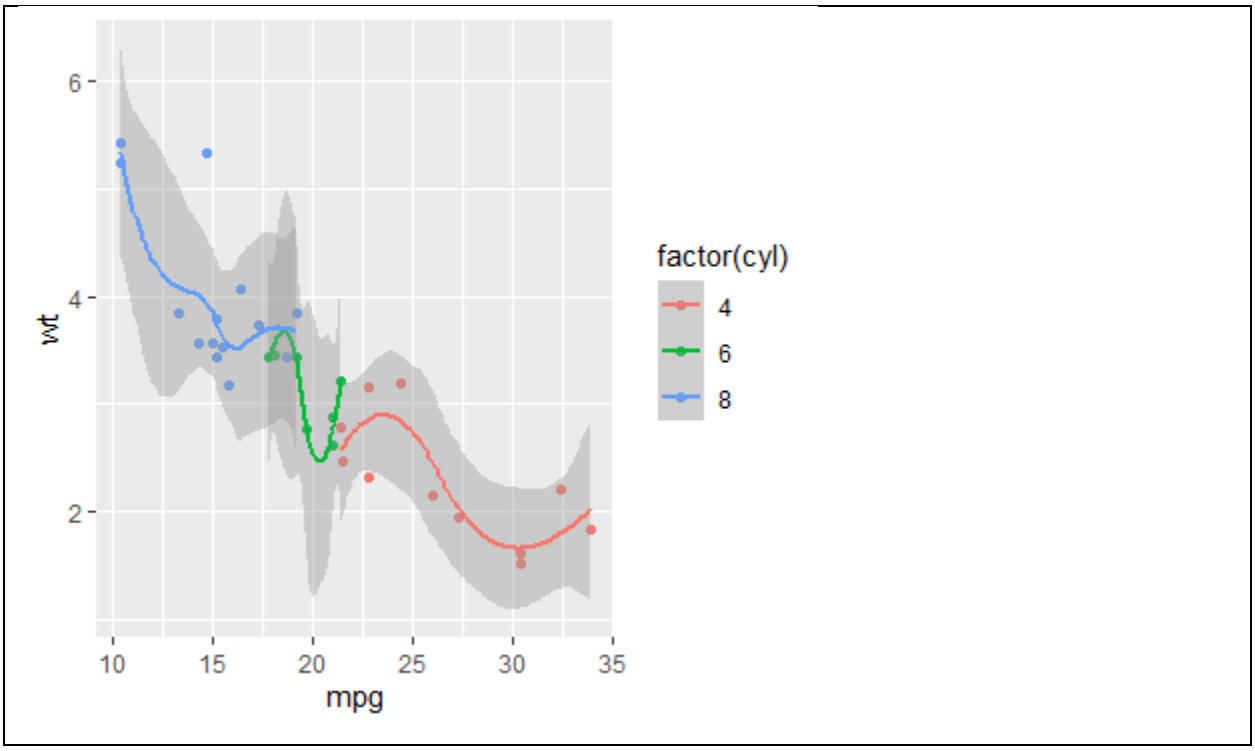
```
#Scatter plots with smoothed line
> qplot(mpg, wt, data = mtcars, geom = c("point", "smooth"))
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



*LOESS is a popular tool used in regression analysis that creates a smooth line through a timeplot or scatter plot to help you to see relationship between variables and foresee trends.

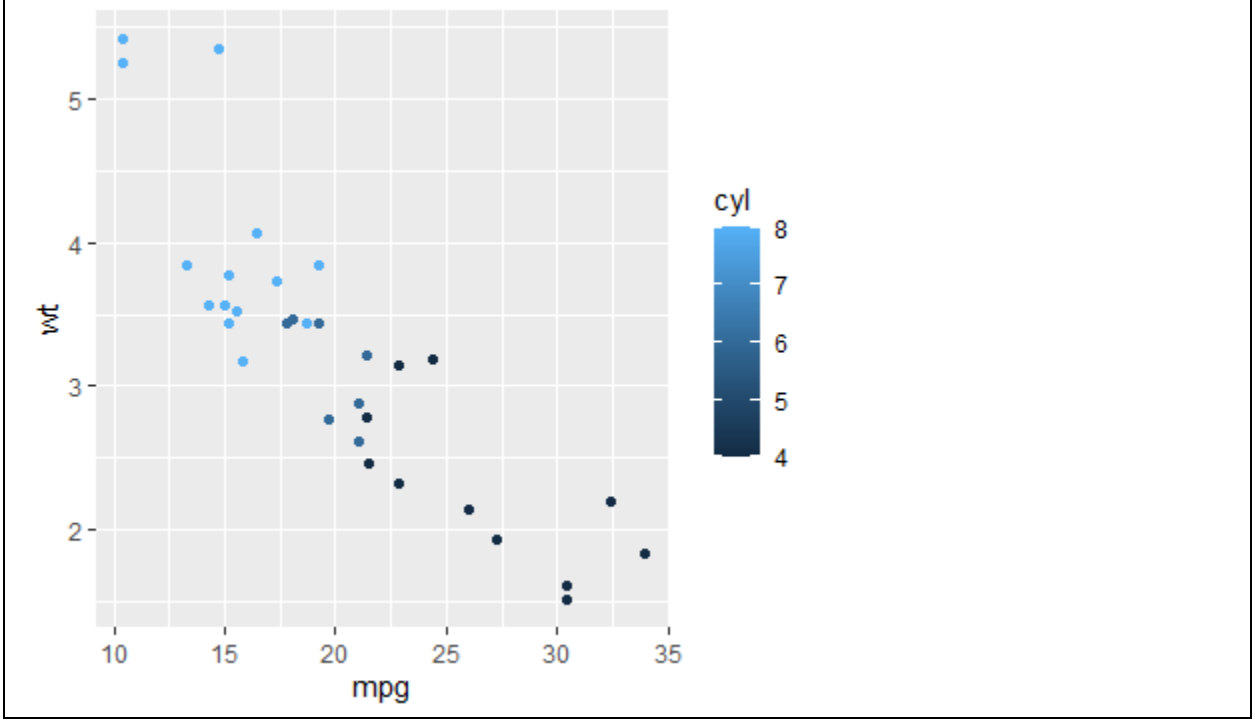
The argument “color” is used to tell R that we want to color the points by groups:

```
> qplot(mpg, wt, data = mtcars, color = factor(cyl),
+       geom=c("point", "smooth"))
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

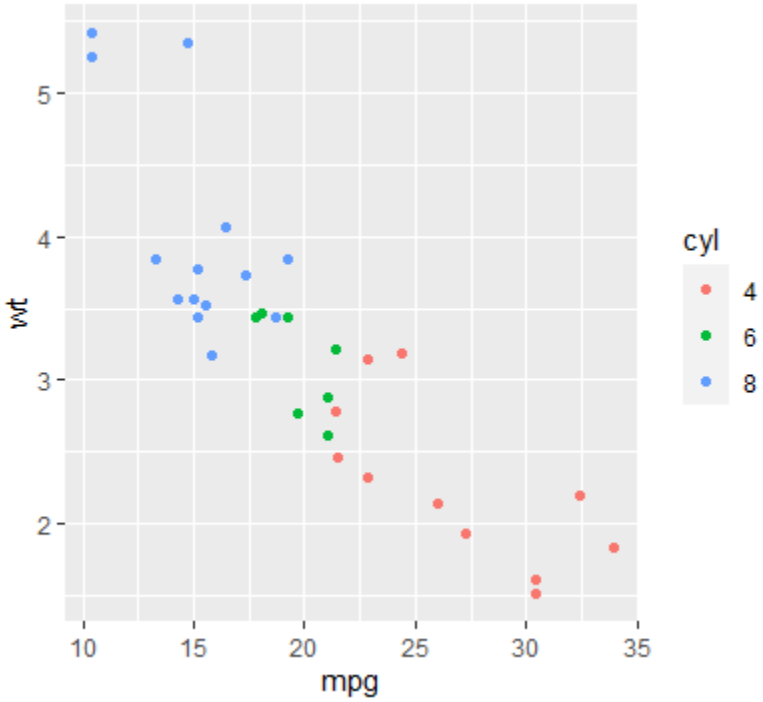


Points can be colored according to the values of a continuous or a discrete variable. The argument “colour” is used.

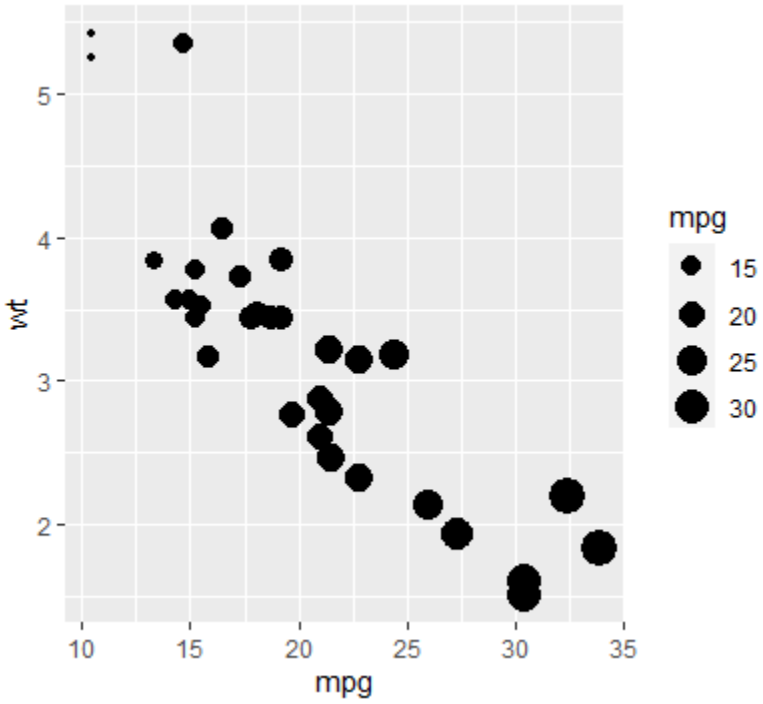
```
> qplot(mpg, wt, data = mtcars, colour = cyl)
```



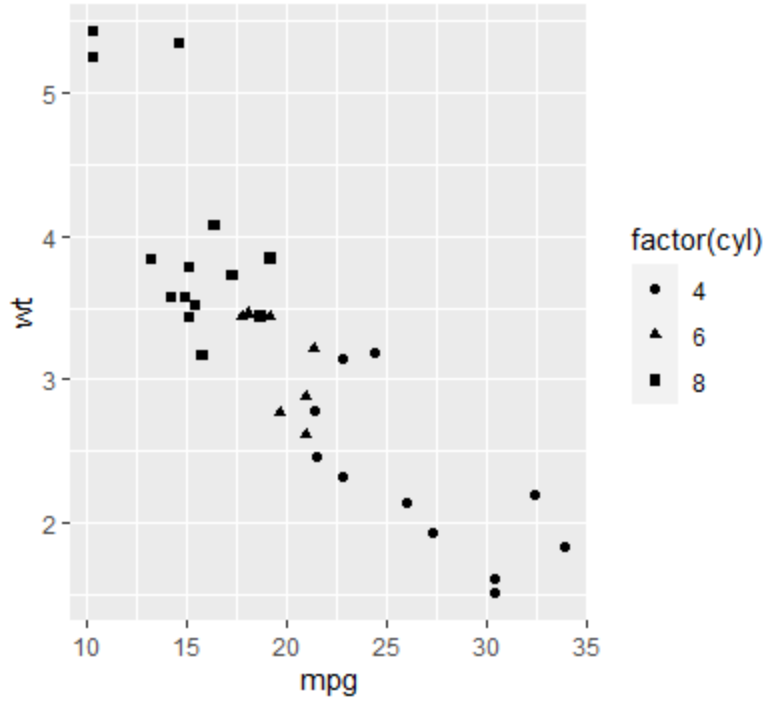
```
> # Change the color by groups (factor)
> df <- mtcars
> df[,'cyl'] <- as.factor(df[,'cyl']) #convert the cyl column to a factor
> qplot(mpg, wt, data = df, colour = cyl)
```




```
# Change the size of points according to  the values of a continuous variable
> qplot(mpg, wt, data = mtcars, size = mpg)
```

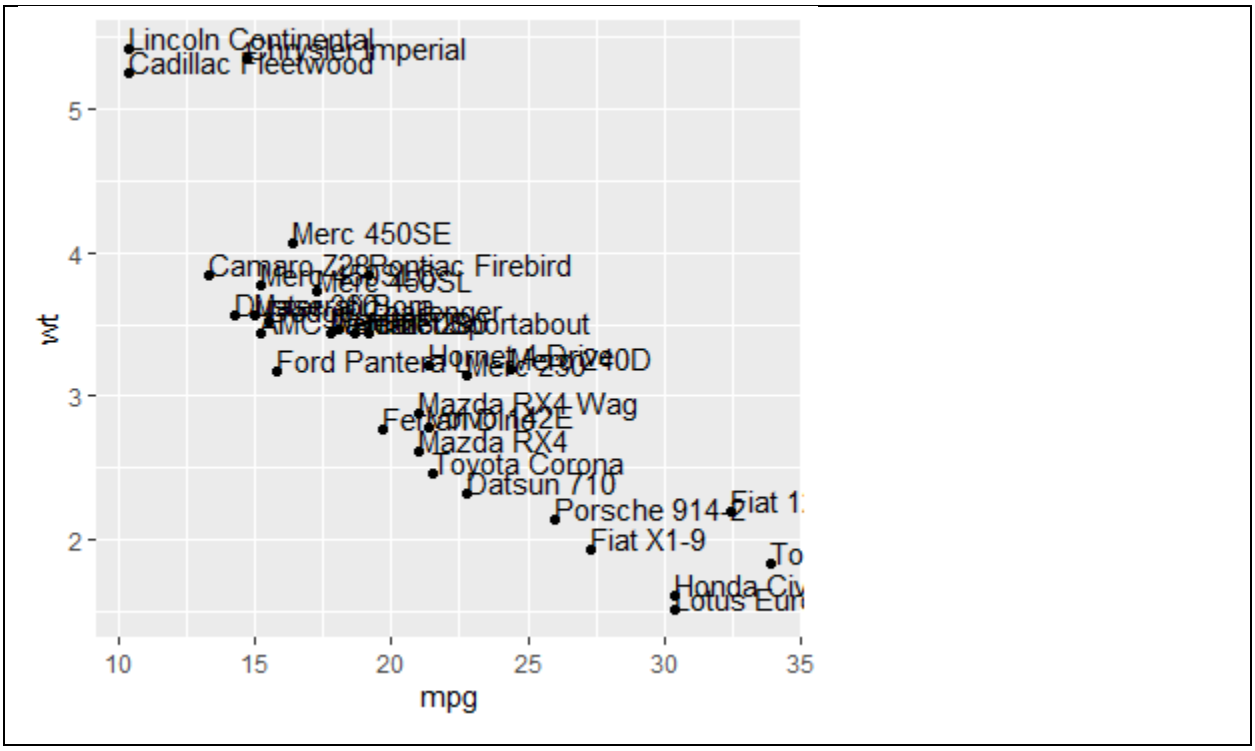


```
> # Change point shapes by groups
> qplot(mpg, wt, data = mtcars, shape = factor(cyl))
```



Scatter plot with texts

```
> qplot(mpg, wt, data = mtcars, label = rownames(mtcars),
+       geom=c("point", "text"),
+       hjust=0, vjust=0)
```



Box Plot

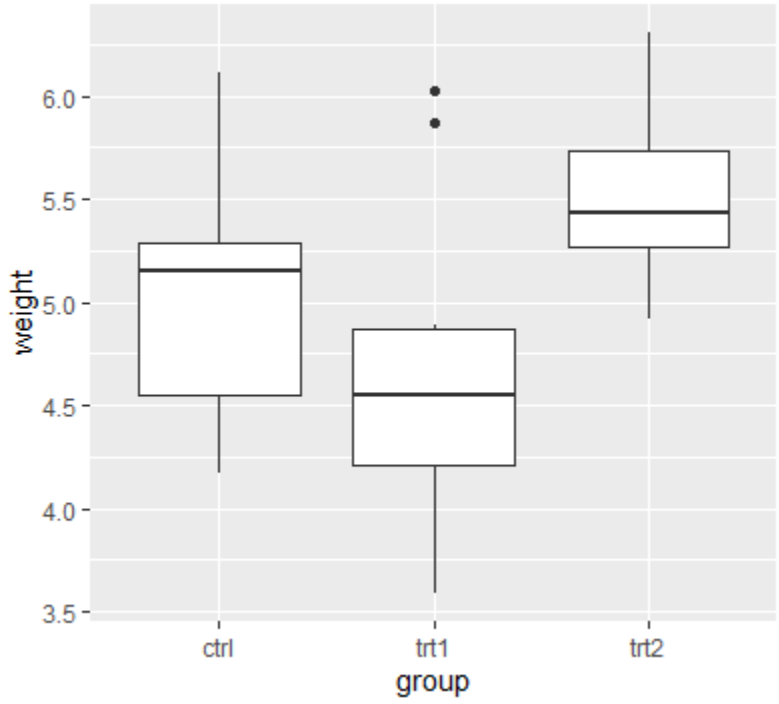
Dataset: PlantGrowth

Description: Results from an experiment to compare yields (as measured by dried weight of plants) obtained under a control and two different treatment conditions.

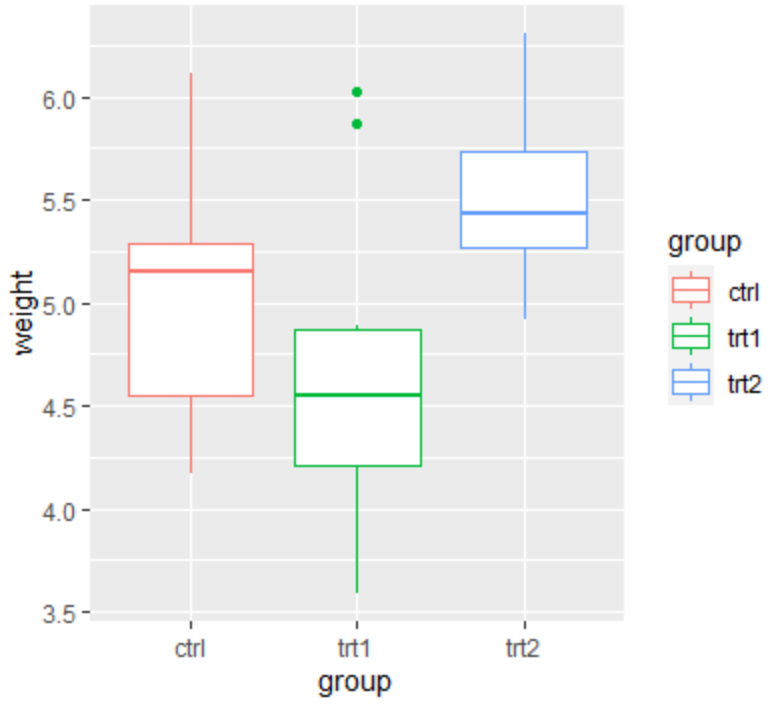
Format: A data frame of 30 cases on 2 variables.

```
> data("PlantGrowth")
> head(PlantGrowth)
  weight group
1   4.17  ctrl
2   5.58  ctrl
3   5.18  ctrl
4   6.11  ctrl
5   4.50  ctrl
6   4.61  ctrl
> qplot(group, weight, data = PlantGrowth,
```

```
+       geom=c("boxplot"))
```

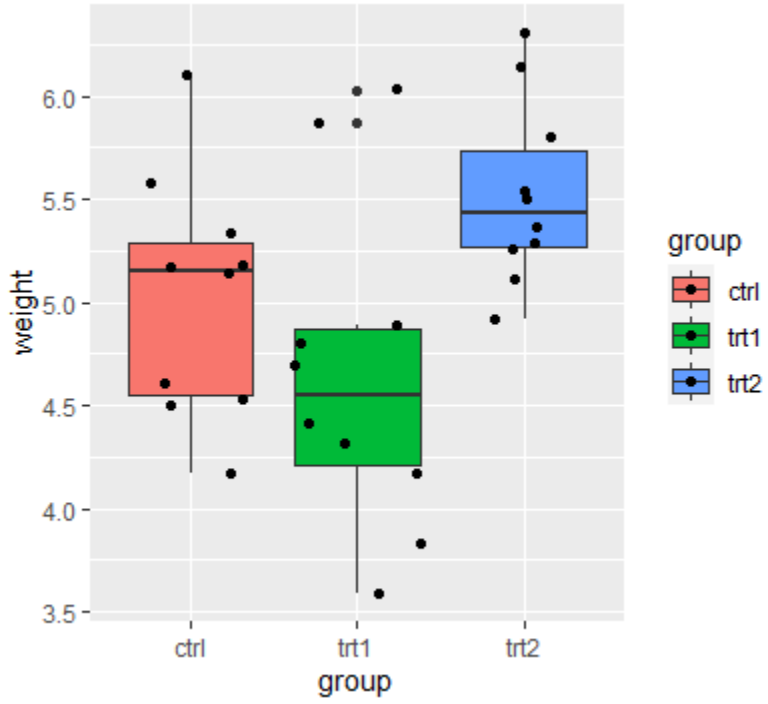



```
> qplot(group, weight, data = PlantGrowth, color = group,
+       geom=c("boxplot")
```




```
> qplot(group, weight, data = PlantGrowth,
```

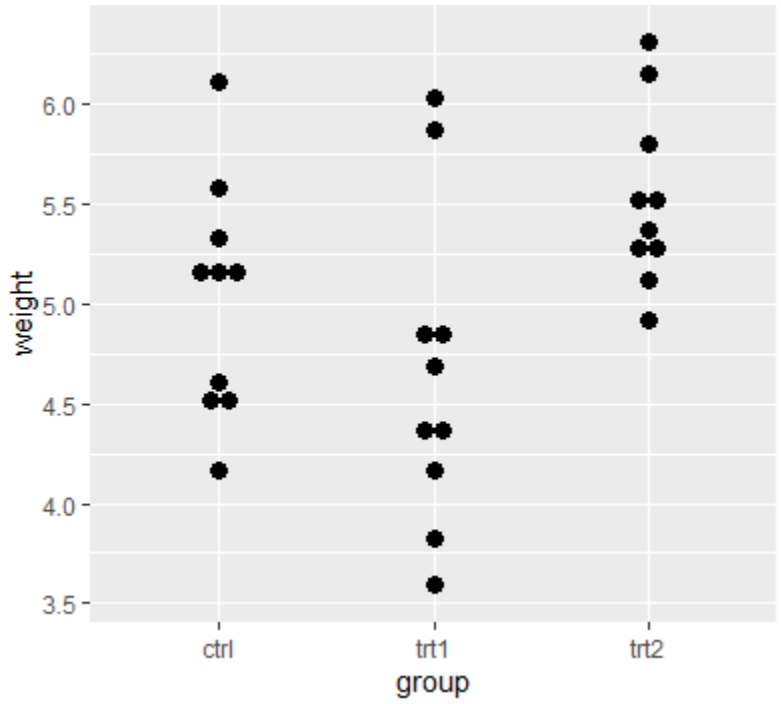
```
+      geom=c("boxplot", "jitter"), fill = group)
```



Dot Plot

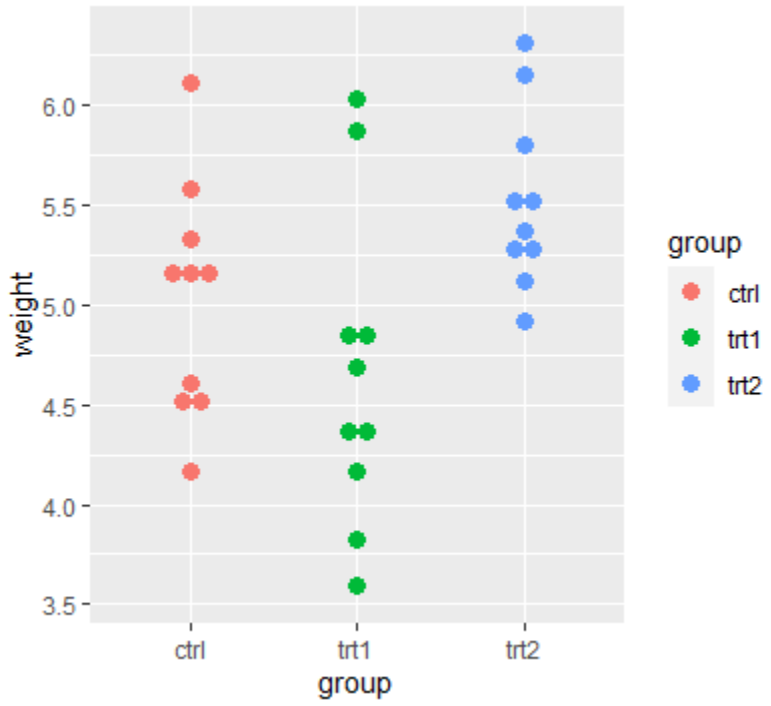
```
> qplot(group, weight, data = PlantGrowth,
    geom=c("dotplot"),
    stackdir = "center", binaxis = "y")
```

Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.



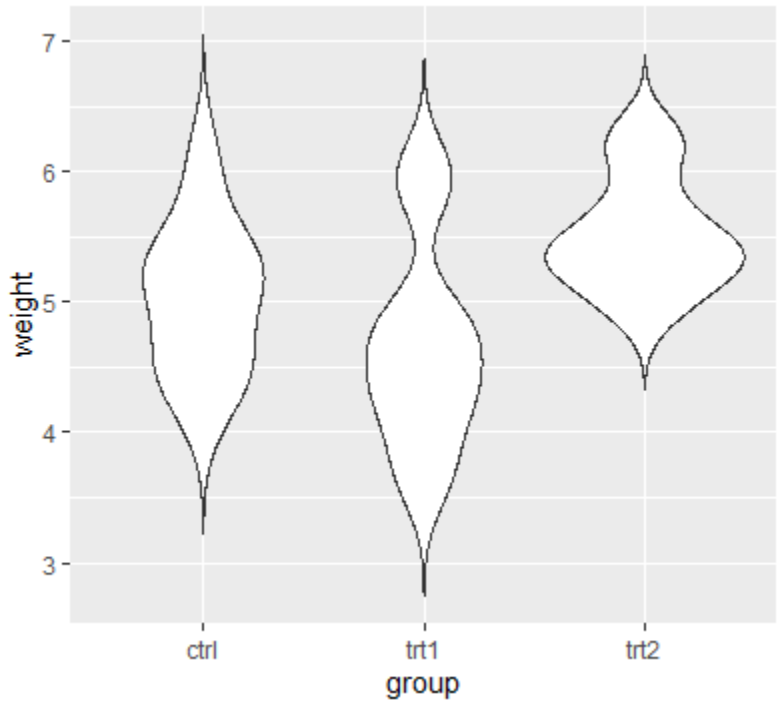

```
> qplot(group, weight, data = PlantGrowth,
+       geom = "dotplot", stackdir = "center", binaxis = "y",
+       color = group, fill = group)
```

Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.



Violin Plot

```
> qplot(group, weight, data = PlantGrowth,
+       geom=c("violin"), trim = FALSE)
```



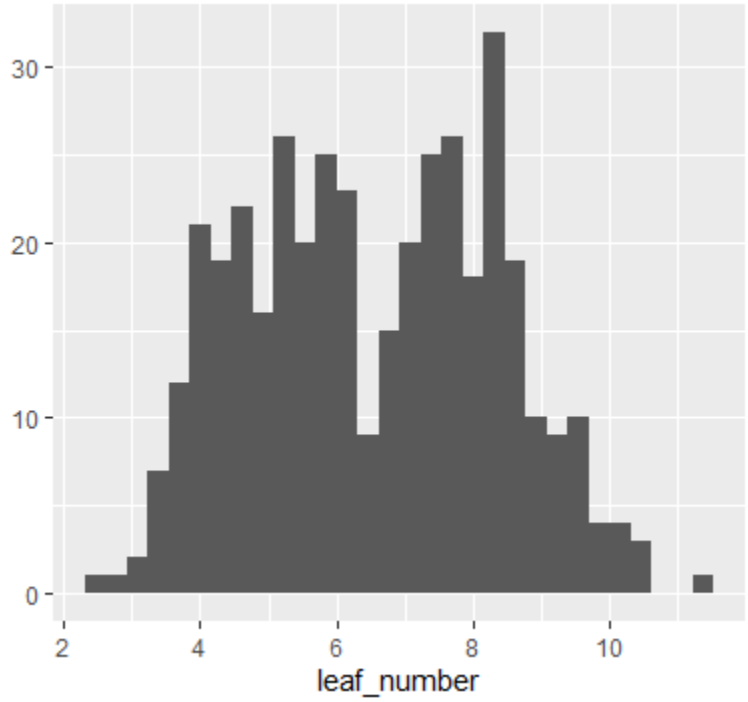
Histogram

Dataset: We will generate some data.

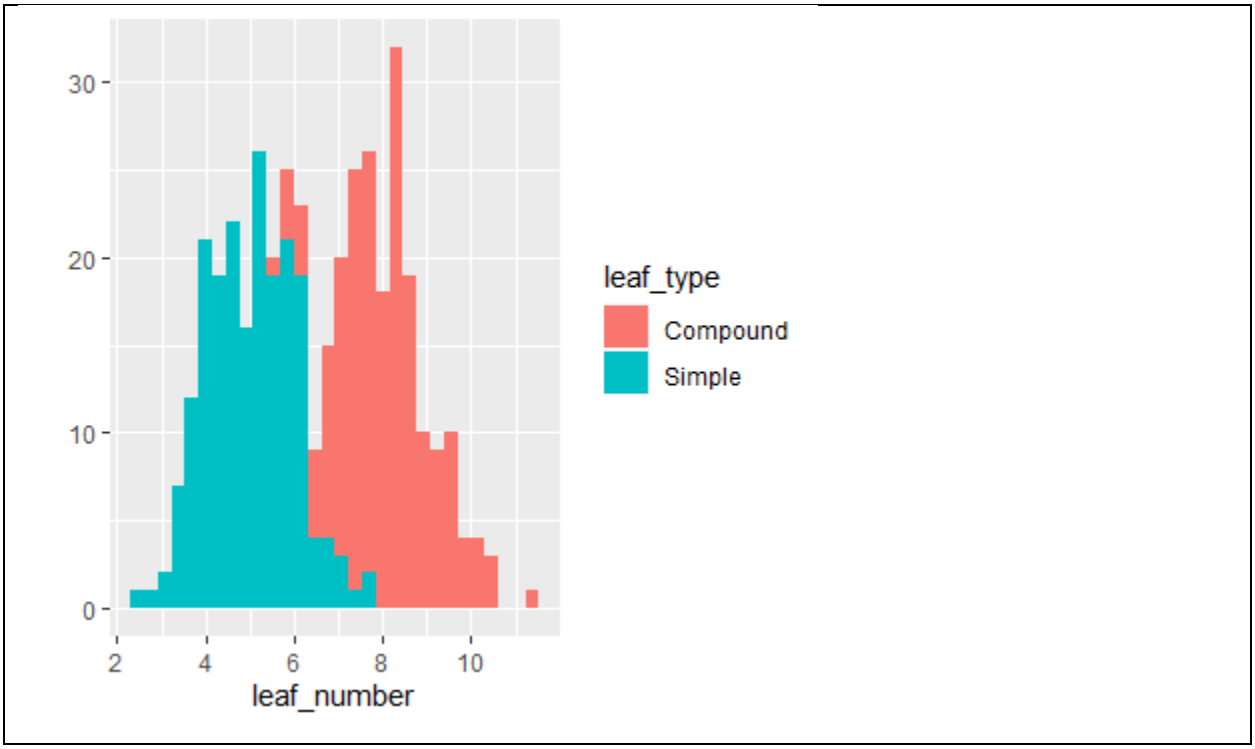
The set.seed() function sets the starting number used to generate a sequence of random numbers

```
> set.seed(3)
> created = data.frame(
+   leaf_type = factor(rep(c("Simple", "Compound"), each=200)),
+   leaf_number = c(rnorm(200, 5), rnorm(200, 8)))
> head(created)
  leaf_type leaf_number
1    Simple    4.038067
2    Simple    4.707474
```

```
3   Simple   5.258788
4   Simple   3.847868
5   Simple   5.195783
6   Simple   5.030124
> qplot(leaf_number, data = created, geom = "histogram")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
> # Change histogram fill color by group (leaf_type)
> qplot(leaf_number, data = created, geom = "histogram",
+       fill = leaf_type)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

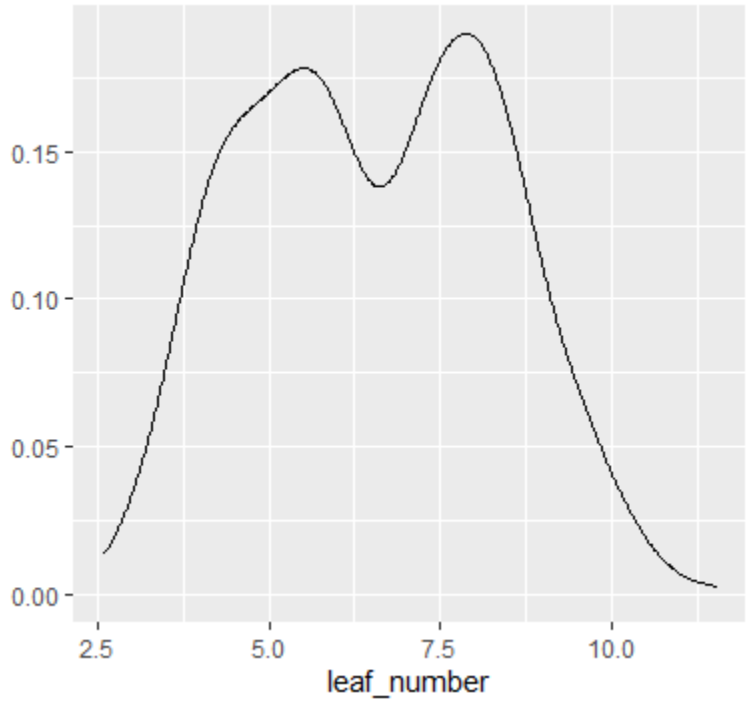


Density Plot

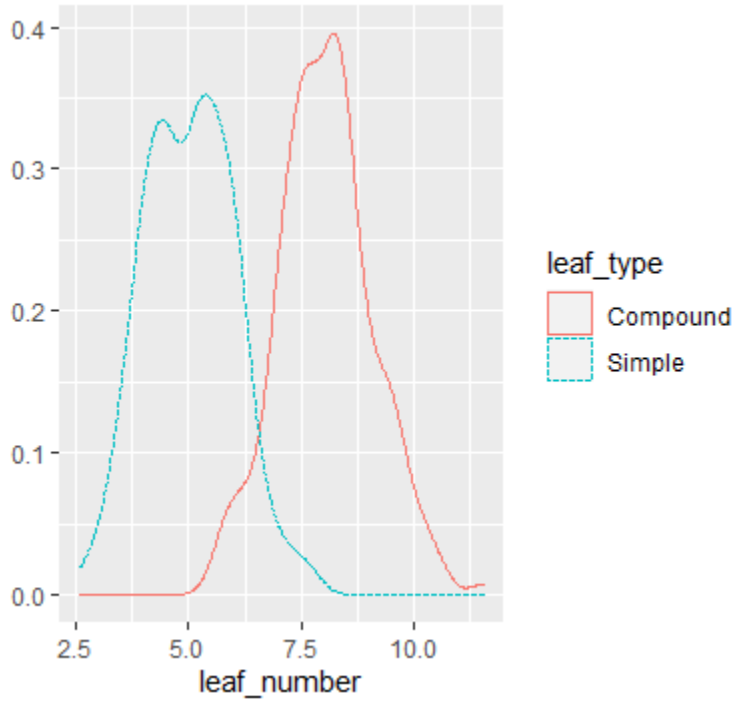
Dataset: Data generated for histogram.

A density plot is a representation of the distribution of a numeric variable. It is a smoothed version of the histogram and is used in the same concept.

```
> qplot(leaf_number, data = created, geom = "density")
```

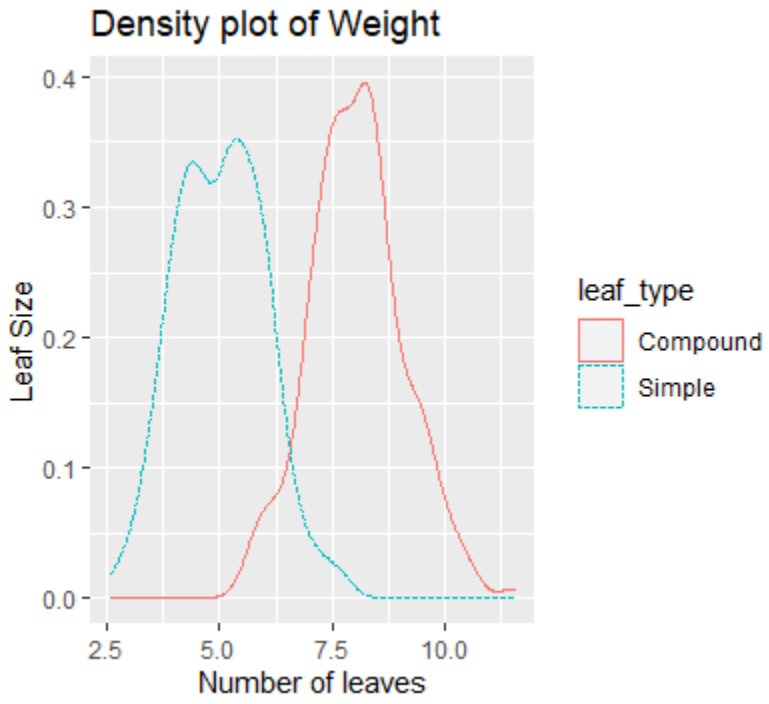



```
> qplot(leaf_number, data = created, geom = "density",
+       color = leaf_type, linetype = leaf_type)
```




```
> qplot(leaf_number, data = created, geom = "density",
+       color = leaf_type, linetype = leaf_type,
```

```
+      xlab = "Number of leaves", ylab = "Leaf Size",
+      main = "Density plot of Weight")
```



Strip Charts/ Jitter Plot

Dataset: ToothGrowth

Description: Length of the teeth in each of 10 guinea pigs at three Vitamin C dosage levels (0.5, 1, and 2 mg) with two delivery methods (orange juice or ascorbic acid).

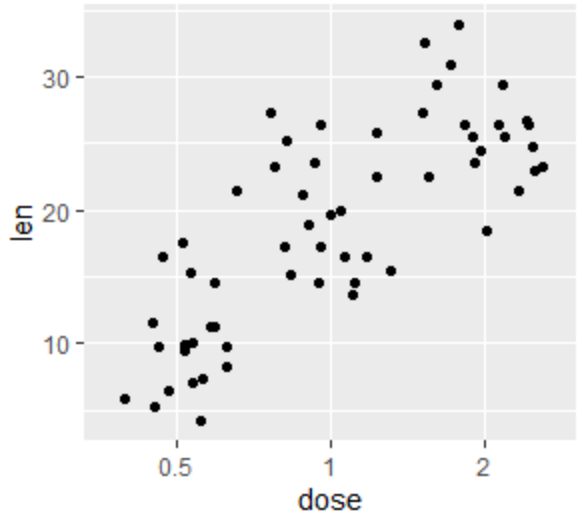
Format: The file contains 60 observations of 3 variables

```
#STRIP CHART/ JITTER PLOT
> 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
```

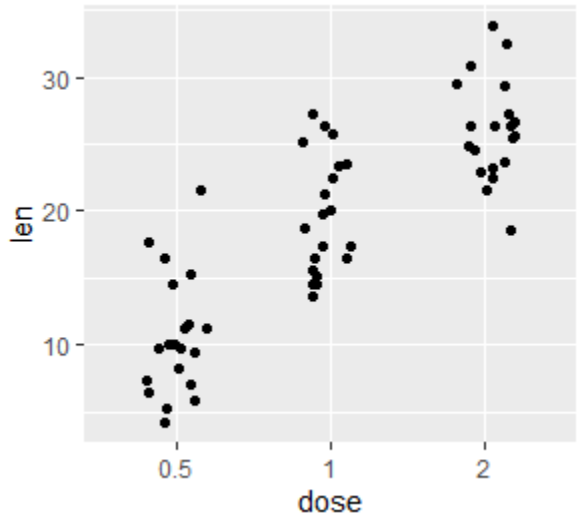
```
5   6.4   VC  0.5
6  10.0   VC  0.5
7  11.2   VC  0.5
8  11.2   VC  0.5
9   5.2   VC  0.5
10  7.0   VC  0.5
11 16.5   VC   1
12 16.5   VC   1
13 15.2   VC   1
14 17.3   VC   1
15 22.5   VC   1
16 17.3   VC   1
17 13.6   VC   1
18 14.5   VC   1
19 18.8   VC   1
20 15.5   VC   1
21 23.6   VC   2
22 18.5   VC   2
23 33.9   VC   2
24 25.5   VC   2
25 26.4   VC   2
26 32.5   VC   2
27 26.7   VC   2
28 21.5   VC   2
29 23.3   VC   2
30 29.5   VC   2
31 15.2   OJ  0.5
32 21.5   OJ  0.5
33 17.6   OJ  0.5
34  9.7   OJ  0.5
35 14.5   OJ  0.5
```

```
36 10.0   OJ  0.5
37  8.2   OJ  0.5
38  9.4   OJ  0.5
39 16.5   OJ  0.5
40  9.7   OJ  0.5
41 19.7   OJ    1
42 23.3   OJ    1
43 23.6   OJ    1
44 26.4   OJ    1
45 20.0   OJ    1
46 25.2   OJ    1
47 25.8   OJ    1
48 21.2   OJ    1
49 14.5   OJ    1
50 27.3   OJ    1
51 25.5   OJ    2
52 26.4   OJ    2
53 22.4   OJ    2
54 24.5   OJ    2
55 24.8   OJ    2
56 30.9   OJ    2
57 26.4   OJ    2
58 27.3   OJ    2
59 29.4   OJ    2
60 23.0   OJ    2
> str(ToothGrowth)
'data.frame':  60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: Factor w/ 3 levels "0.5","1","2": 1 1 1 1 1 1 1 1 1 1 ...
> ToothGrowth$dose <- as.factor(ToothGrowth$dose)
```

```
> str(ToothGrowth)
'data.frame':  60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: Factor w/ 3 levels "0.5","1","2": 1 1 1 1 1 1 1 1 1 1 ...
> ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter()
```

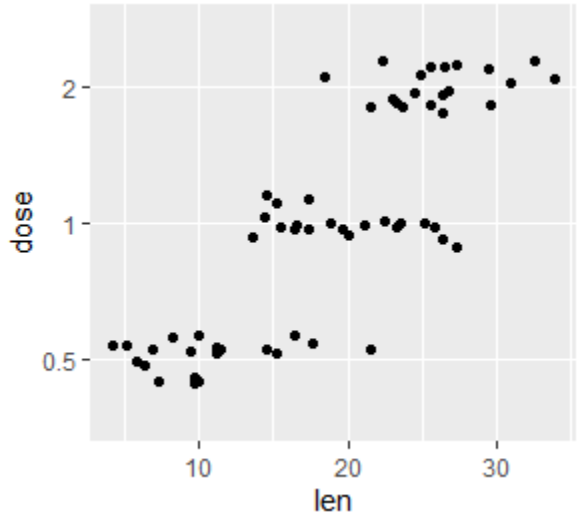



```
> p<-ggplot(ToothGrowth, aes(x=dose, y=len)) +
  geom_jitter(position=position_jitter(0.2))
> p
```

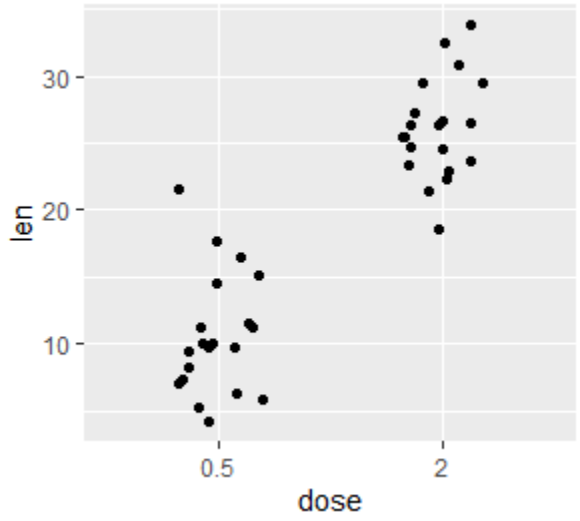



```
> p + coord_flip()
```

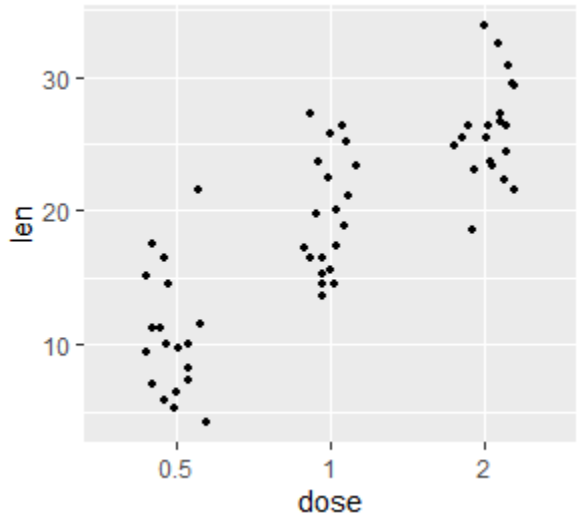
```
> p
```



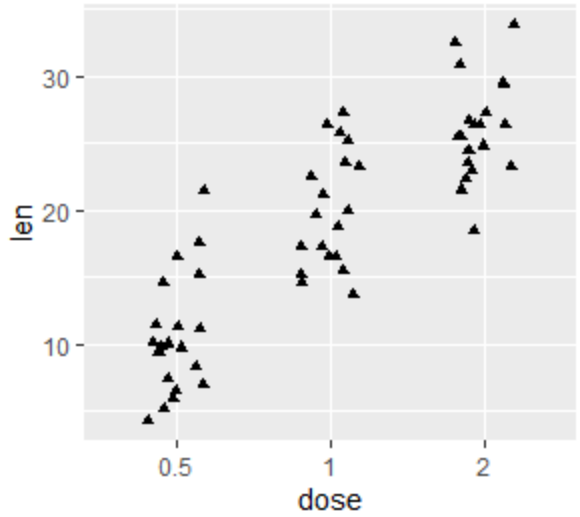

```
> p + scale_x_discrete(limits=c("0.5", "2"))
Warning message:
Removed 20 rows containing missing values (geom_point).
```



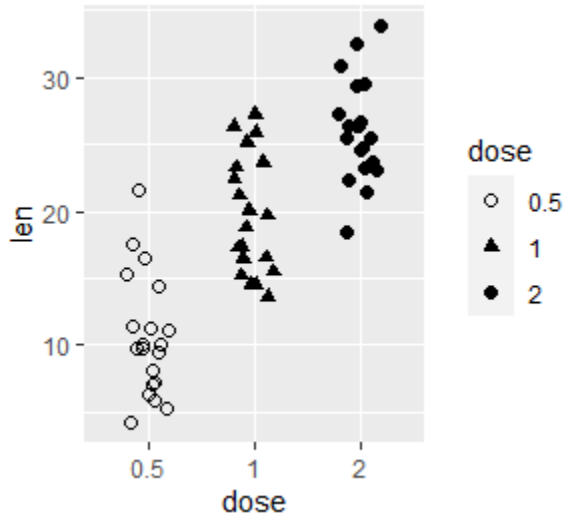

```
> #change the size of points
> ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter(position=position_jitter(0.2), cex=1.2)
```



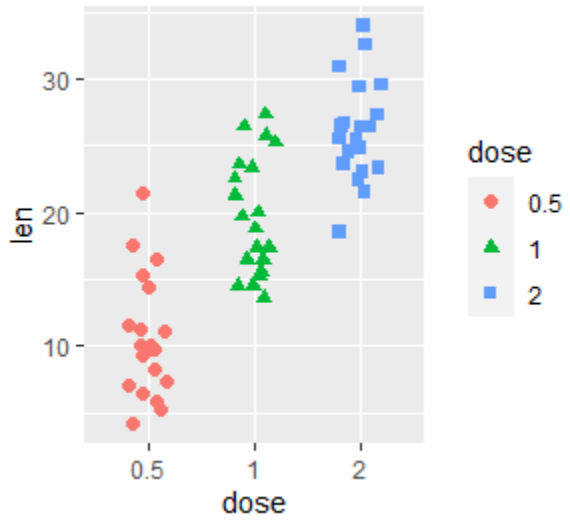

```
> #change shape of points
> ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter(position=position_jitter(0.2), shape=17)
```



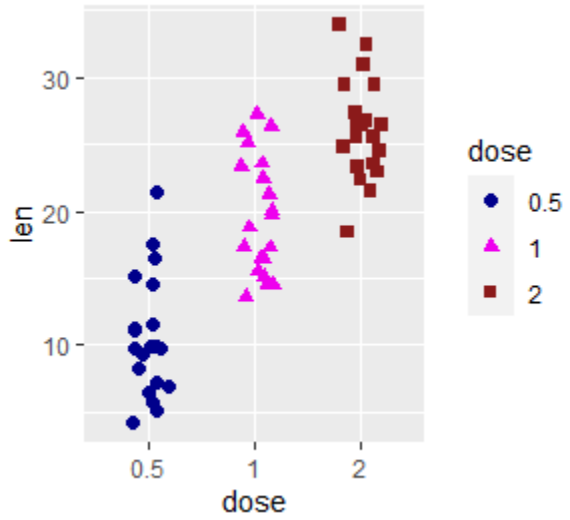

```
> #Change shape according to dose
> p <- ggplot(ToothGrowth, aes(x=dose, y=len, shape=dose)) +
+   geom_jitter(position=position_jitter(0.2), cex=2)
> p + scale_shape_manual(values=c(1,17,19))
```




```
> # Change shape and color according to dose
> p<-ggplot(ToothGrowth, aes(x=dose, y=len, shape=dose, color=dose)) +
+   geom_jitter(position=position_jitter(0.2), cex=2)
> p
```

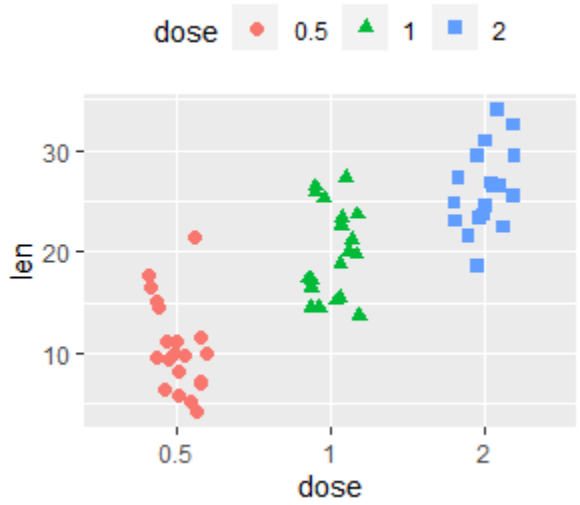



```
> #choose your colors
> p+scale_color_manual(values=c("blue4", "magenta2", "firebrick4"))
```



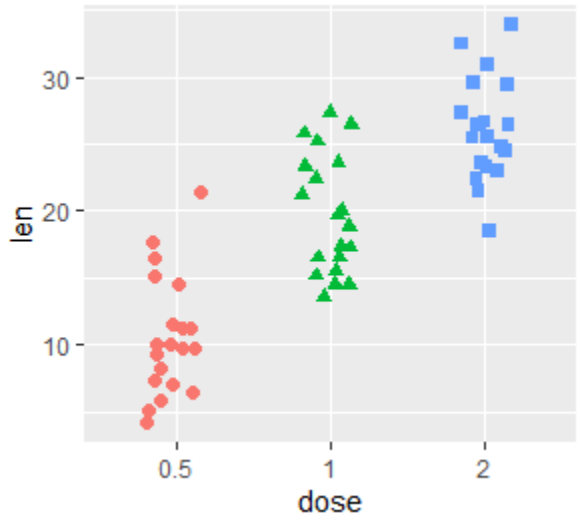

```
> # Choose your palette
> p+scale_color_brewer(palette="Dark2")
```


```
> #Change the legend position
> p + theme(legend.position="top")
```

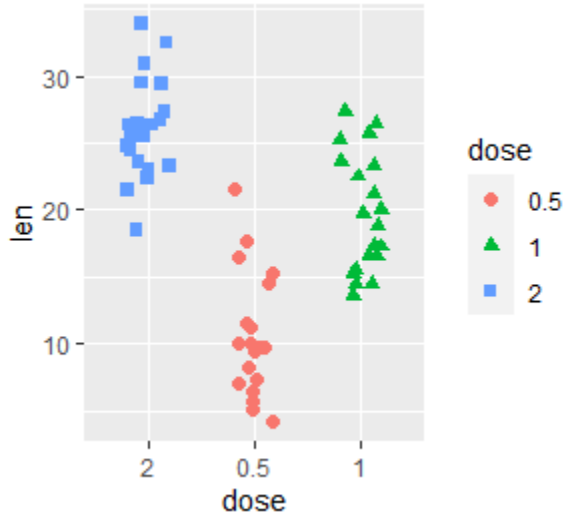


> p + theme(legend.position="bottom")

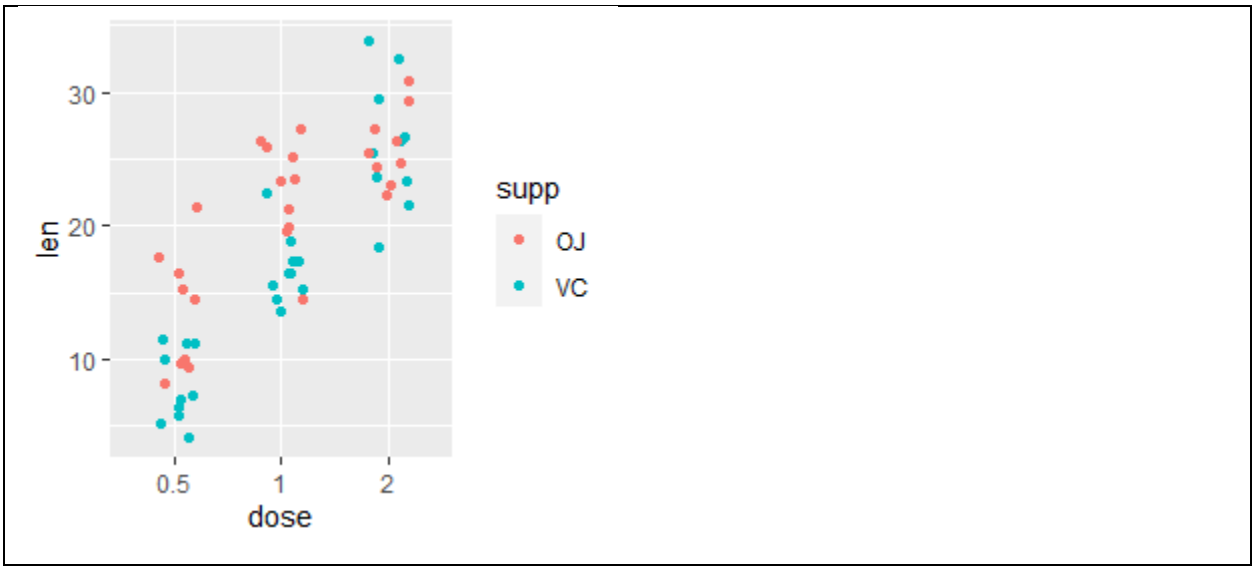
> p + theme(legend.position="none")



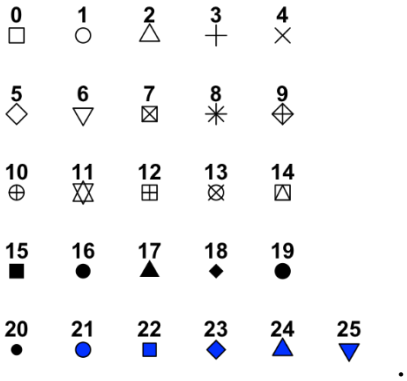

```
> #Change the order of items in the legend
> p + scale_x_discrete(limits=c("2", "0.5", "1"))
```




```
# Change stripchart colors by groups
ggplot(ToothGrowth, aes(x=dose, y=len, color=supp)) +
  geom_jitter(position=position_jitter(0.2))
```



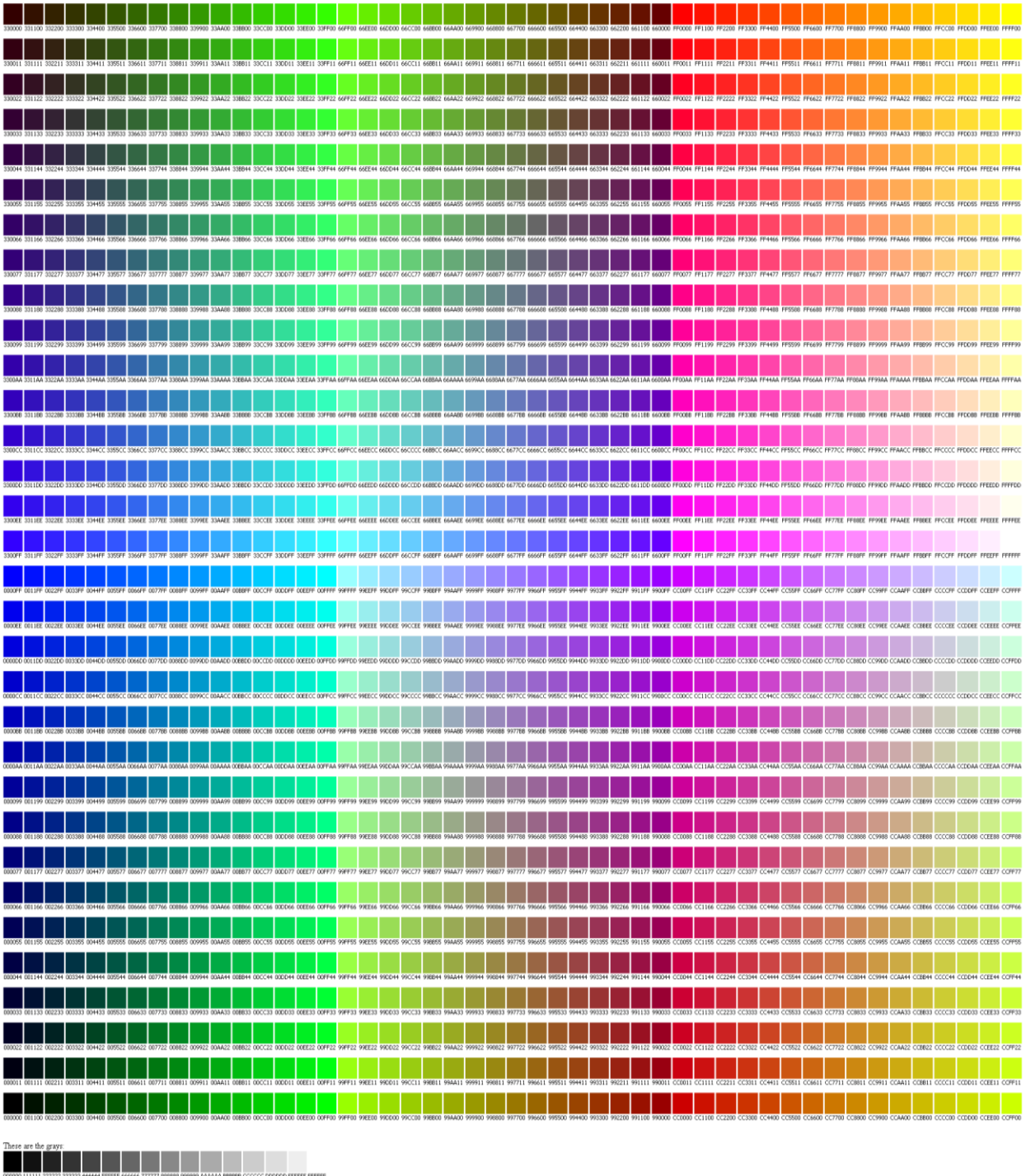
It adds a small amount of random variation to the location of each point, and is a useful way of handling overplotting caused by discreteness in smaller datasets.



You can choose one of the 657 named colors in R

brown4	darkorange4	gray	gray57	hotpink3	lightsalmon4	navajowhite1	plum3	slategray3	antiquewhite
brown3	darkorange3	goldenrod4	gray56	hotpink2	lightsalmon3	navajowhite	plum2	slategray2	aliceblue
brown2	darkorange2	goldenrod3	gray55	hotpink1	lightsalmon2	moccasin	plum1	slategray1	white
brown1	darkorange1	goldenrod2	gray54	hotpink	lightsalmon1	mistyrose4	plum	slategray	yellowgreen
brown	darkorange	goldenrod1	gray53	honeydew4	lightsalmon	mistyrose3	pink4	slateblue4	yellow4
blueviolet	darkolivegreen4	goldenrod	gray52	honeydew3	lightpink4	mistyrose2	pink3	slateblue3	yellow3
blue4	darkolivegreen3	gold4	gray51	honeydew2	lightpink3	mistyrose1	pink2	slateblue2	yellow2
blue3	darkolivegreen2	gold3	gray50	honeydew1	lightpink2	mistyrose	pink1	slateblue1	yellow1
blue2	darkolivegreen1	gold2	gray49	honeydew	lightpink1	mintcream	pink	slateblue	yellow
blue1	darkolivegreen	gold1	gray48	greenyellow	lightpink	midnightblue	peru	skyblue4	whitesmoke
blue	darkmagenta	gold	gray47	green4	lightgrey	mediumvioletred	peachpuff4	skyblue3	wheat4
blanchedalmond	darkkhaki	ghostwhite	gray46	green3	lightgreen	mediumturquoise	peachpuff3	skyblue2	wheat3
black	darkgrey	gainsboro	gray45	green2	lightgray	mediumspringgreen	peachpuff2	skyblue1	wheat2
bisque4	darkgreen	forestgreen	gray44	green1	lightgoldenrodyellow	mediumslateblue	peachpuff1	skyblue	wheat1
bisque3	darkgray	floralwhite	gray43	green	lightgoldenrod4	mediumseagreen	peachpuff	sienna4	wheat
bisque2	darkgoldenrod4	firebrick4	gray42	gray100	lightgoldenrod3	mediumpurple4	papayawhip	sienna3	violetred4
bisque1	darkgoldenrod3	firebrick3	gray41	gray99	lightgoldenrod2	mediumpurple3	palevioletred4	sienna2	violetred3
bisque	darkgoldenrod2	firebrick2	gray40	gray98	lightgoldenrod1	mediumpurple2	palevioletred3	sienna1	violetred2
beige	darkgoldenrod1	firebrick1	gray39	gray97	lightgoldenrod	mediumpurple1	palevioletred2	sienna	violetred1
azure4	darkgoldenrod	firebrick	gray38	gray96	lightcyan4	mediumpurple	palevioletred1	seashell4	violetred
azure3	darkcyan	dodgerblue4	gray37	gray95	lightcyan3	mediumorchid4	palevioletred	seashell3	violet
azure2	darkblue	dodgerblue3	gray36	gray94	lightcyan2	mediumorchid3	paleturquoise4	seashell2	turquoise4
azure1	cyan4	dodgerblue2	gray35	gray93	lightcyan1	mediumorchid2	paleturquoise3	seashell1	turquoise3
azure	cyan3	dodgerblue1	gray34	gray92	lightcyan	mediumorchid1	paleturquoise2	seashell	turquoise2
aquamarine4	cyan2	dodgerblue	gray33	gray91	lightcoral	mediumorchid	paleturquoise1	seagreen4	turquoise1
aquamarine3	cyan1	dimgrey	gray32	gray90	lightblue4	mediumblue	paleturquoise	seagreen3	turquoise
aquamarine2	cyan	dimgray	gray31	gray89	lightblue3	mediumaquamarine	palegreen4	seagreen2	tomato4
aquamarine1	cornsilk4	deepskyblue4	gray30	gray88	lightblue2	maroon4	palegreen3	seagreen1	tomato3
aquamarine	cornsilk3	deepskyblue3	gray29	gray87	lightblue1	maroon3	palegreen2	seagreen	tomato2
antiquewhite4	cornsilk2	deepskyblue2	gray28	gray86	lightblue	maroon2	palegreen1	sandybrown	tomato1
antiquewhite3	cornsilk1	deepskyblue1	gray27	gray85	lemonchiffon4	maroon1	palegreen	salmon4	tomato
antiquewhite2	cornsilk	deepskyblue	gray26	gray84	lemonchiffon3	maroon	palegoldenrod	salmon3	thistle4
antiquewhite1	cornflowerblue	deeppink4	gray25	gray83	lemonchiffon2	magenta4	orchid4	salmon2	thistle3
antiquewhite	coral4	deeppink3	gray24	gray82	lemonchiffon1	magenta3	orchid3	salmon1	thistle2
aliceblue	coral3	deeppink2	gray23	gray81	lemonchiffon	magenta2	orchid2	salmon	thistle1
white	coral2	deeppink1	gray22	gray80	lawngreen	magenta1	orchid1	saddlebrown	thistle
bisque3	coral1	deeppink	gray21	gray79	lavenderblush4	magenta	orchid	royalblue4	tan4
bisque2	coral	darkviolet	gray20	gray78	lavenderblush3	linen	orangered4	royalblue3	tan3
bisque1	chocolate4	darkturquoise	gray19	gray77	lavenderblush2	limegreen	orangered3	royalblue2	tan2
bisque	chocolate3	darkslategrey	gray18	gray76	lavenderblush1	lightyellow4	orangered2	royalblue1	tan1
beige	chocolate2	darkslategray4	gray17	gray75	lavenderblush	lightyellow3	orangered1	royalblue	tan
azure4	chocolate1	darkslategray3	gray16	gray74	lavender	lightyellow2	orangered	rosybrown4	steelblue4
azure3	chocolate	darkslategray2	gray15	gray73	khaki4	lightyellow1	orange4	rosybrown3	steelblue3
azure2	chartreuse4	darkslategray1	gray14	gray72	khaki3	lightyellow	orange3	rosybrown2	steelblue2
azure1	chartreuse3	darkslategray	gray13	gray71	khaki2	lightsteelblue4	orange2	rosybrown1	steelblue1
azure	chartreuse2	darkslateblue	gray12	gray70	khaki1	lightsteelblue3	orange1	rosybrown	steelblue
aquamarine4	chartreuse1	darkseagreen4	gray11	gray69	khaki	lightsteelblue2	orange	red4	springgreen4
aquamarine3	chartreuse	darkseagreen3	gray10	gray68	ivory4	lightsteelblue1	olivedrab4	red3	springgreen3
aquamarine2	cadetblue4	darkseagreen2	gray9	gray67	ivory3	lightsteelblue	olivedrab3	red2	springgreen2
aquamarine1	cadetblue3	darkseagreen1	gray8	gray66	ivory2	lightslategrey	olivedrab2	red1	springgreen1
aquamarine	cadetblue2	darkseagreen	gray7	gray65	ivory1	lightslategray	olivedrab1	red	springgreen
antiquewhite4	cadetblue1	darksalmon	gray6	gray64	ivory	lightslateblue	olivedrab	purple4	snow4
antiquewhite3	cadetblue	darkred	gray5	gray63	indianred4	lightskyblue4	oldlace	purple3	snow3
antiquewhite2	burlywood4	darkorchid4	gray4	gray62	indianred3	lightskyblue3	navyblue	purple2	snow2
antiquewhite1	burlywood3	darkorchid3	gray3	gray61	indianred2	lightskyblue2	navy	purple1	snow1
antiquewhite	burlywood2	darkorchid2	gray2	gray60	indianred1	lightskyblue1	navajowhite4	purple	snow
aliceblue	burlywood1	darkorchid1	gray1	gray59	indianred	lightskyblue	navajowhite3	powderblue	slategrey
white	burlywood	darkorchid	gray0	gray58	hotpink4	lightseagreen	navajowhite2	plum4	slategray4

And if you are still in search of colors, you can use the hexadecimal codes.

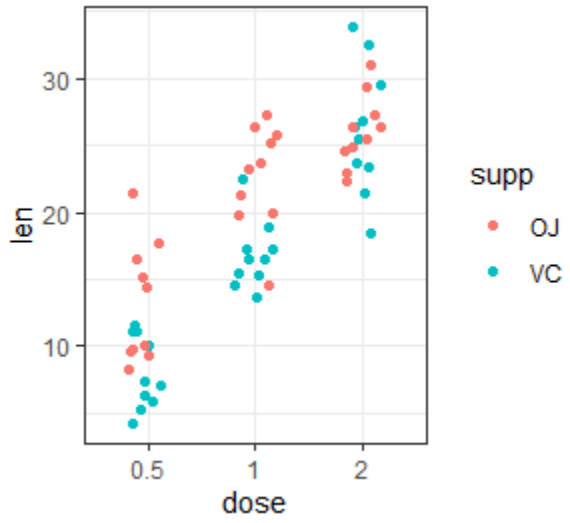


Themes in ggplot

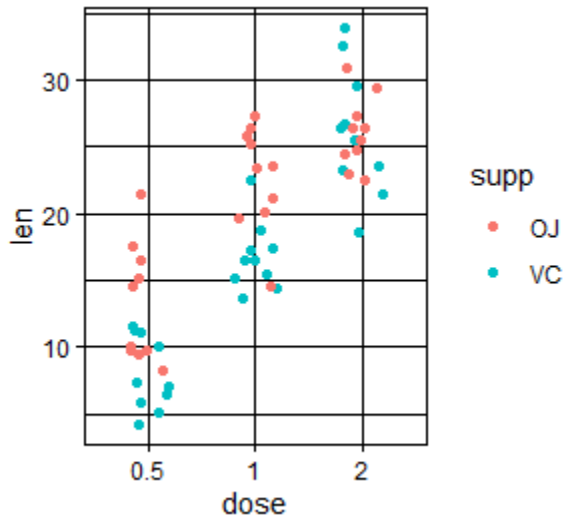
Themes are a powerful way to customize the non-data components of your plots: i.e. titles, labels, fonts, background, gridlines, and legends. Themes can be used to give plots a consistent customized look.

> p + theme_gray()

> p + theme_bw()

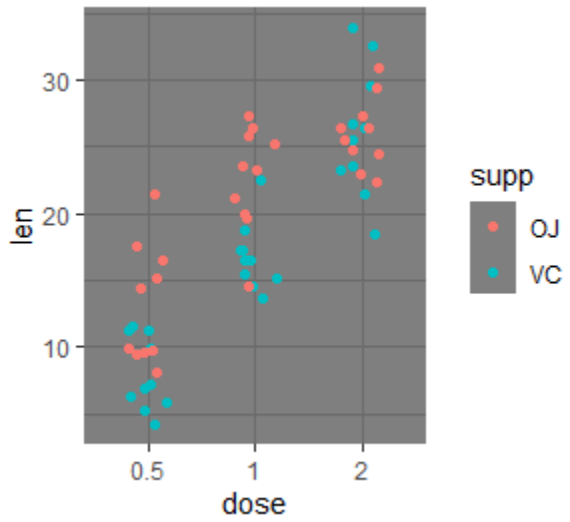


> p + theme_linedraw()

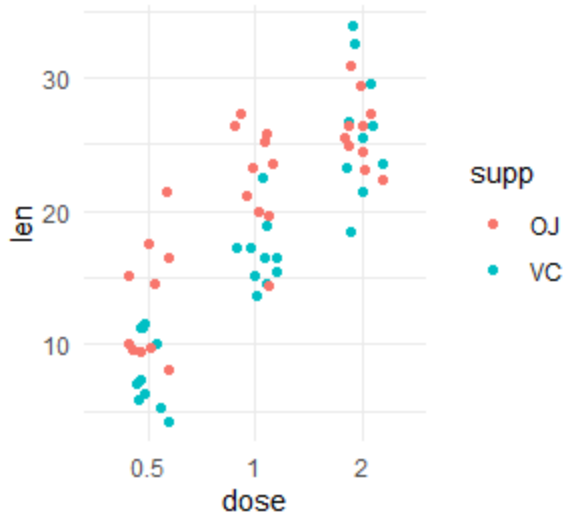


> p + theme_light()

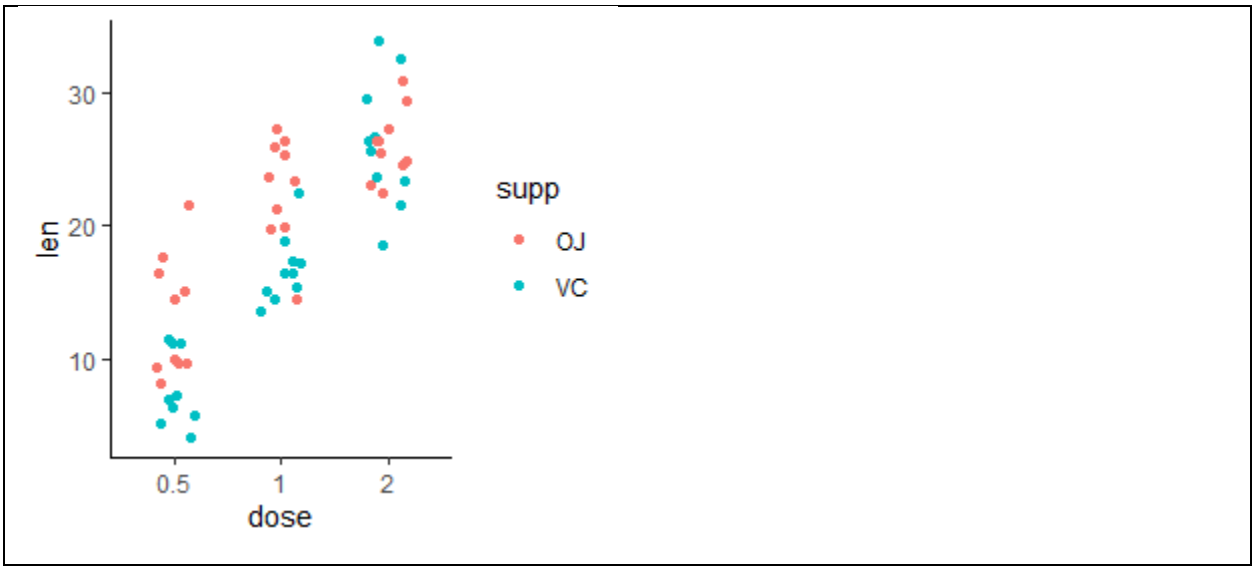
> p + theme_dark()



> p + theme_minimal()



> p + theme_classic()

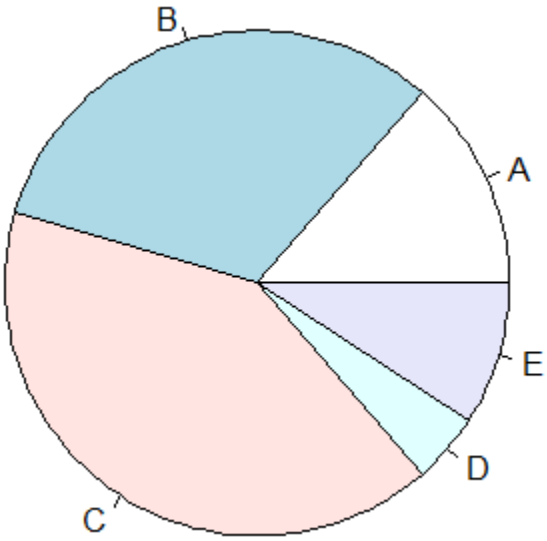


To modify an individual theme component you can use code like `plot + theme (element.name = element_function())`.

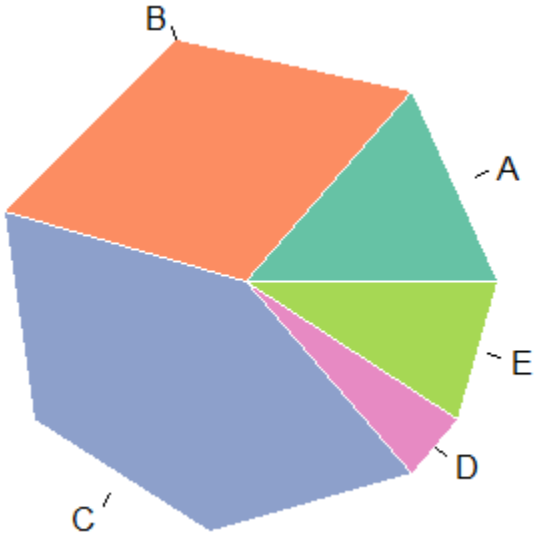
Pie Chart

A pie chart is a circle divided into sectors that each represent a proportion of the whole.

```
> # PIE CHART
> countdata <- c(3,7,9,1,2)
> species = c("A","B","C","D","E")
> pie(countdata , species)
```




```
> #Non-circular piechart
> install.packages("RColorBrewer")
> library(RColorBrewer)
> myPalette <- brewer.pal(5, "Set2")
> pie(countdata , labels = species, border="white", col=myPalette, edges = 10 )
```



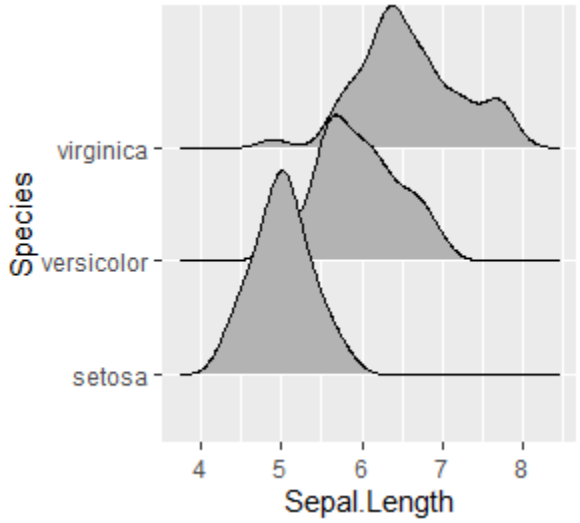
Ridgeline Plot

Dataset: Iris

Description: 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters.

Format: 150 observations of 4 features

```
library(ggridges)
ggplot(iris, aes(x = Sepal.Length, y = Species)) + geom_density_ridges()
> ggplot(iris, aes(x = Sepal.Length, y = Species)) + geom_density_ridges()
Picking joint bandwidth of 0.181
```



Circular Bar Plot

Dataset: We will create a new dataset for this exercise

Format: 60 count values of genes with T1, T2, T3 and C levels of treatment.

```
#CIRCULAR BAR PLOT
#install.packages("tidyverse")
>library(tidyverse)

# Create dataset
> data <- data.frame(
 individual=paste( "Gene ", seq(1,60), sep=""),
```

```
  group=c( rep('T1', 10), rep('T2', 30), rep('T3', 14), rep('C', 6)) ,
  value=sample( seq(10,100), 60, replace=T)
)

# Set a number of 'empty bar' to add at the end of each group
>empty_bar <- 3
>to_add <- data.frame( matrix(NA, empty_bar*nlevels(data$group), ncol(data)) )
>colnames(to_add) <- colnames(data)
>to_add$group <- rep(levels(data$group), each=empty_bar)
>data <- rbind(data, to_add)
>data <- data %>% arrange(group)
>data$id <- seq(1, nrow(data))

# Get the name and the y position of each label
>label_data <- data
>number_of_bar <- nrow(label_data)
>angle <- 90 - 360 * (label_data$id-0.5) /number_of_bar      # I substract 0.5 because the letter must have the angle of the center of the bars. Not extreme right(1) or extreme left (0)
>label_data$hjust <- ifelse( angle < -90, 1, 0)
>label_data$angle <- ifelse(angle < -90, angle+180, angle)

# prepare a data frame for base lines
>base_data <- data %>%
  group_by(group) %>%
  summarize(start=min(id), end=max(id) - empty_bar) %>%
  rowwise() %>%
  mutate(title=mean(c(start, end)))

# prepare a data frame for grid (scales)
>grid_data <- base_data
```

```
>grid_data$end <- grid_data$end[ c( nrow(grid_data), 1:nrow(grid_data)-1)] + 1
>grid_data$start <- grid_data$start - 1
>grid_data <- grid_data[-1,]

# Make the plot
>p <- ggplot(data, aes(x=as.factor(id), y=value, fill=group)) +        # Note that id is a factor. If x is numeric, there is some space between the first bar

  geom_bar(aes(x=as.factor(id), y=value, fill=group), stat="identity", alpha=0.5) +

  # Add a val=100/75/50/25 lines. I do it at the beginning to make sur barplots are OVER it.
  geom_segment(data=grid_data, aes(x = end, y = 80, xend = start, yend = 80), colour = "grey", alpha=1, size=0.3 , inherit.aes = FALSE ) +
  geom_segment(data=grid_data, aes(x = end, y = 60, xend = start, yend = 60), colour = "grey", alpha=1, size=0.3 , inherit.aes = FALSE ) +
  geom_segment(data=grid_data, aes(x = end, y = 40, xend = start, yend = 40), colour = "grey", alpha=1, size=0.3 , inherit.aes = FALSE ) +
  geom_segment(data=grid_data, aes(x = end, y = 20, xend = start, yend = 20), colour = "grey", alpha=1, size=0.3 , inherit.aes = FALSE ) +

  # Add text showing the value of each 100/75/50/25 lines
  annotate("text", x = rep(max(data$id),4), y = c(20, 40, 60, 80), label = c("20", "40", "60", "80") , color="grey", size=3 , angle=0, fontface="bold", hjust=1) +

  geom_bar(aes(x=as.factor(id), y=value, fill=group), stat="identity", alpha=0.5) +
  ylim(-100,120) +
  theme_minimal() +
  theme(
    legend.position = "none",
    axis.text = element_blank(),
```

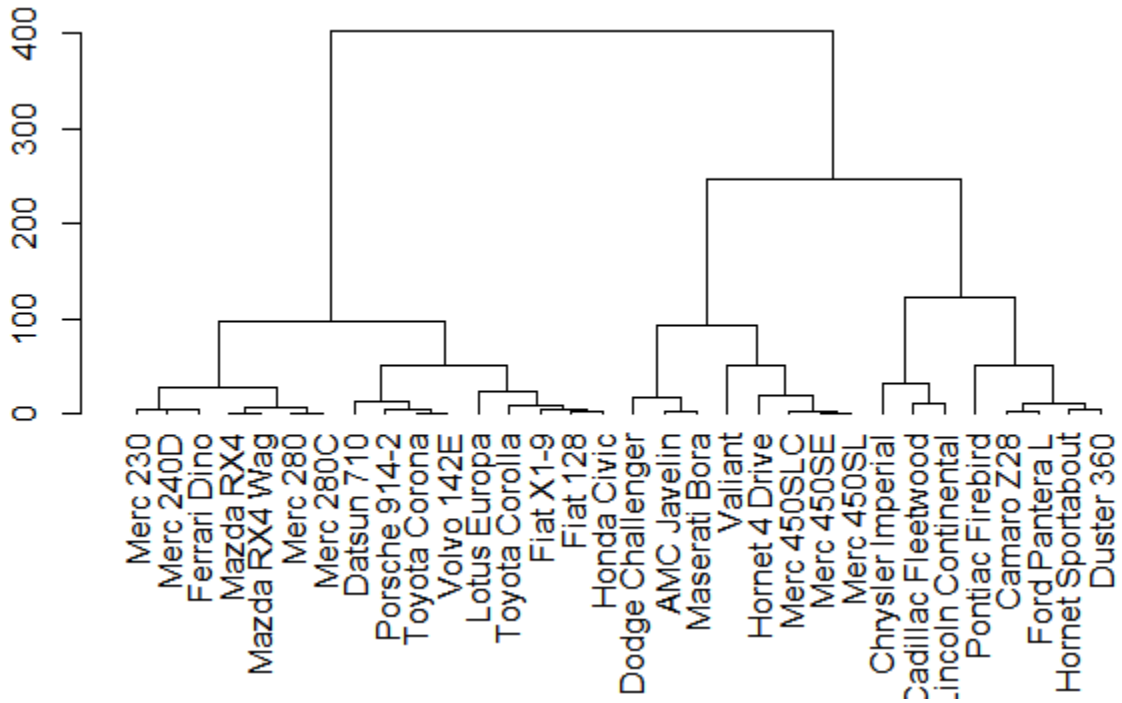
```
    axis.title = element_blank(),
    panel.grid = element_blank(),
    plot.margin = unit(rep(-1,4), "cm")
  ) +
  coord_polar() +
  geom_text(data=label_data, aes(x=id, y=value+10, label=individual, hjust=hjust), color="black", fontface="bold",alpha=0.6, size=2.5, angle= label_data$angle, inherit.aes = FALSE ) +

  # Add base line information
  geom_segment(data=base_data, aes(x = start, y = -5, xend = end, yend = -5), colour = "black", alpha=0.8, size=0.6 , inherit.aes = FALSE )  +
  geom_text(data=base_data, aes(x = title, y = -18, label=group), hjust=c(1,1,0,0), colour = "black", alpha=0.8, size=4, fontface="bold", inherit.aes = FALSE)


> p
```

Dendrogram

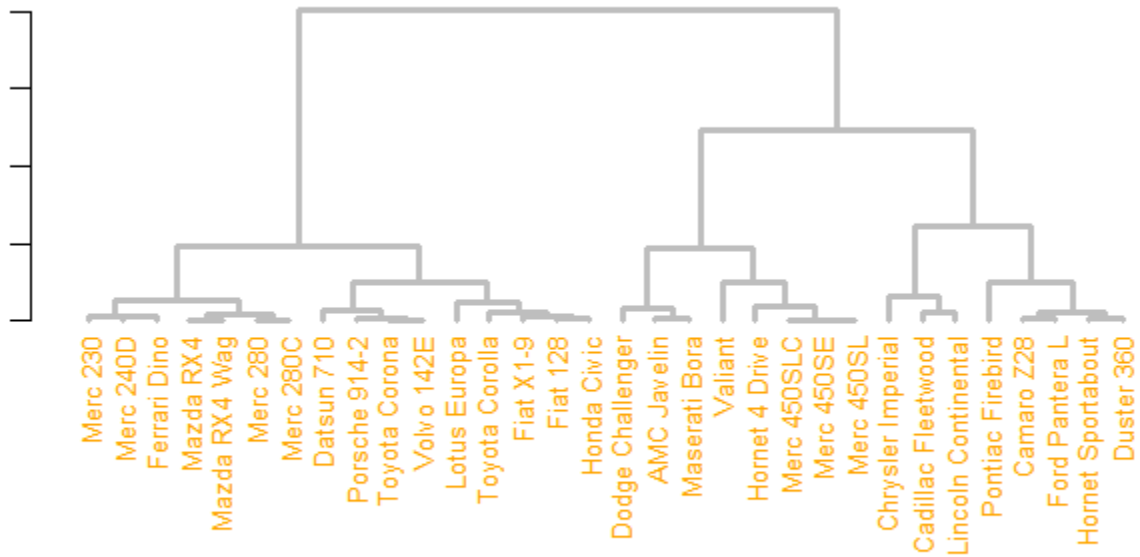
Dataset: mtcars

```
>library(tidyverse)

# Data
head(mtcars)

# Clusterisation using 3 variables
>mtcars %>%
  select(mpg, cyl, disp) %>%
  dist() %>%
  hclust() %>%
  as.dendrogram() -> dend
```

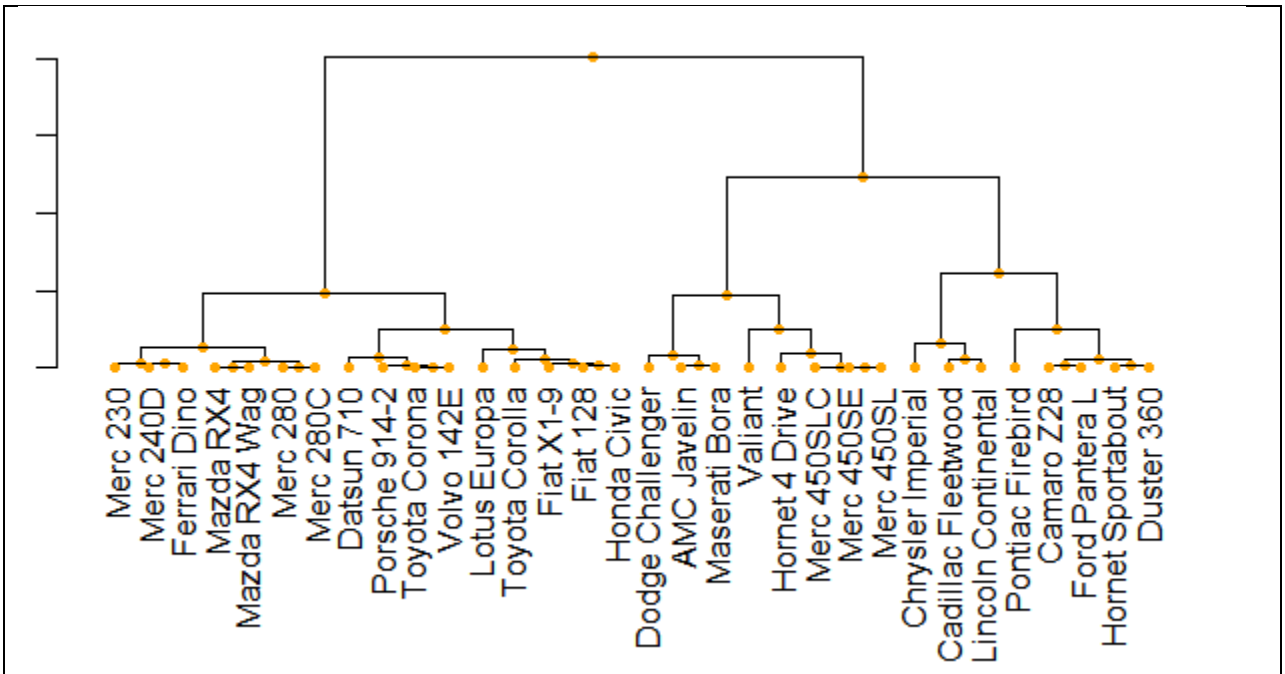
```
# Plot
>par(mar=c(7,3,1,1))  # Increase bottom margin to have the complete label
>plot(dend)
```



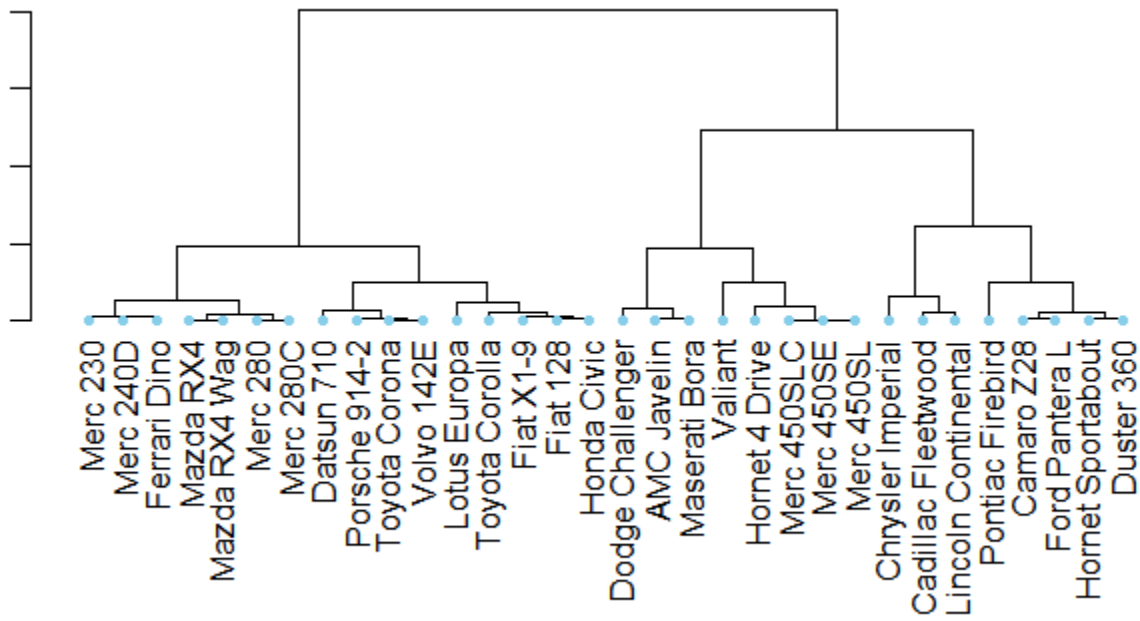

```
#install the package dendextend
>library(dendextend)

# Chart (left)
>dend %>%
 # Custom branches
 > set("branches_col", "grey") %>% set("branches_lwd", 3) %>%
 # Custom labels
 >set("labels_col", "orange") %>% set("labels_cex", 0.8) %>%
 >plot()
```



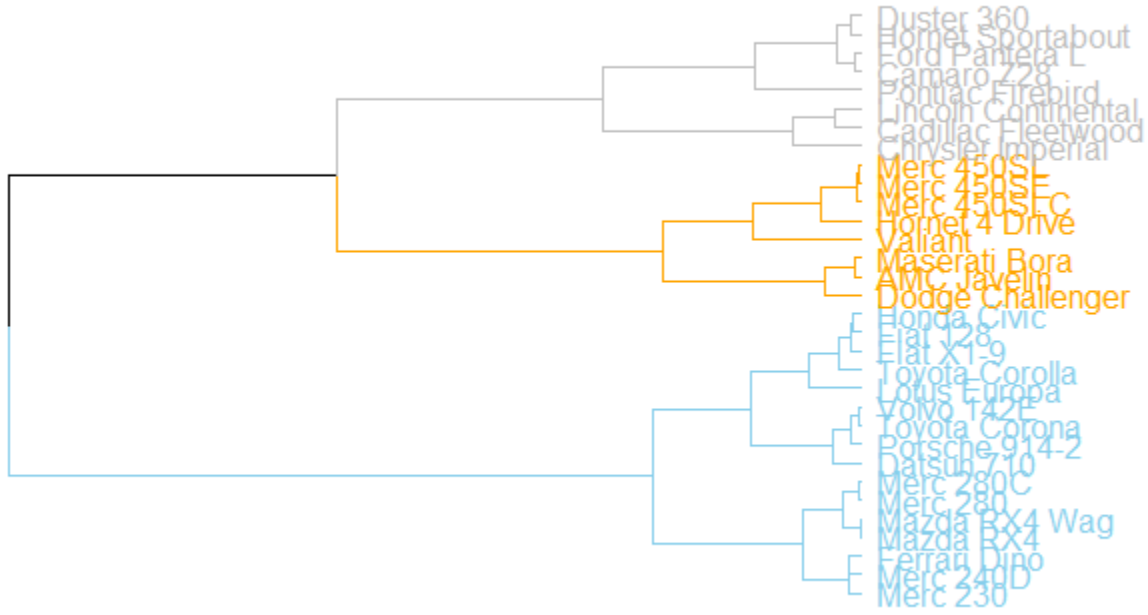

```
# Middle
>dend %>%
 set("nodes_pch", 19)  %>%
 set("nodes_cex", 0.7) %>%
 set("nodes_col", "orange") %>%
 plot()
```



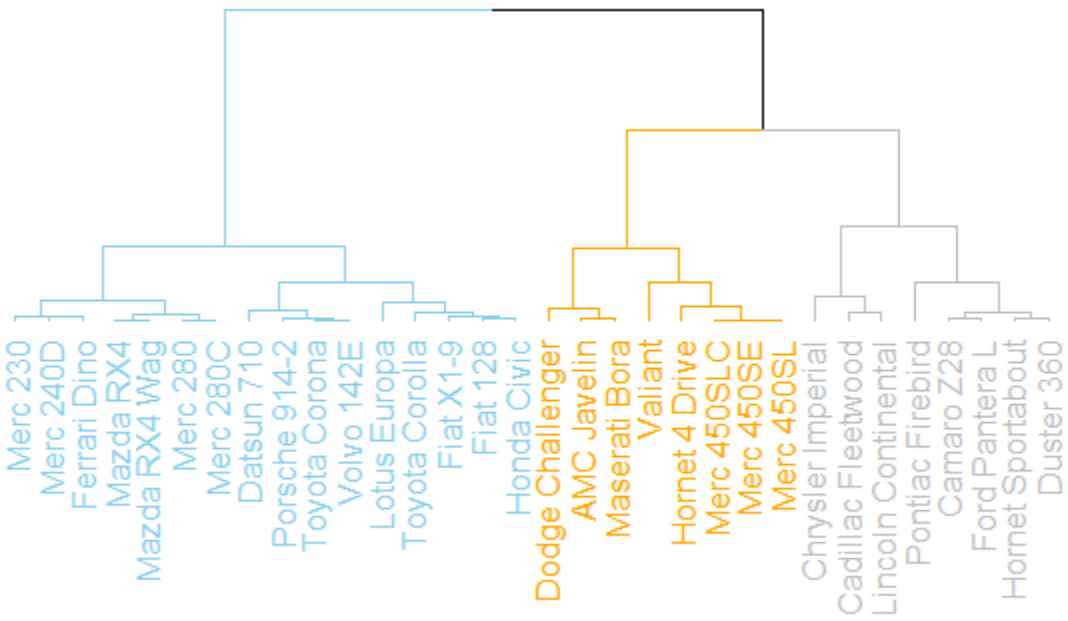

```
# right
>dend %>%
 set("leaves_pch", 19)  %>%
 set("leaves_cex", 0.7) %>%
 set("leaves_col", "skyblue") %>%
 plot()
```



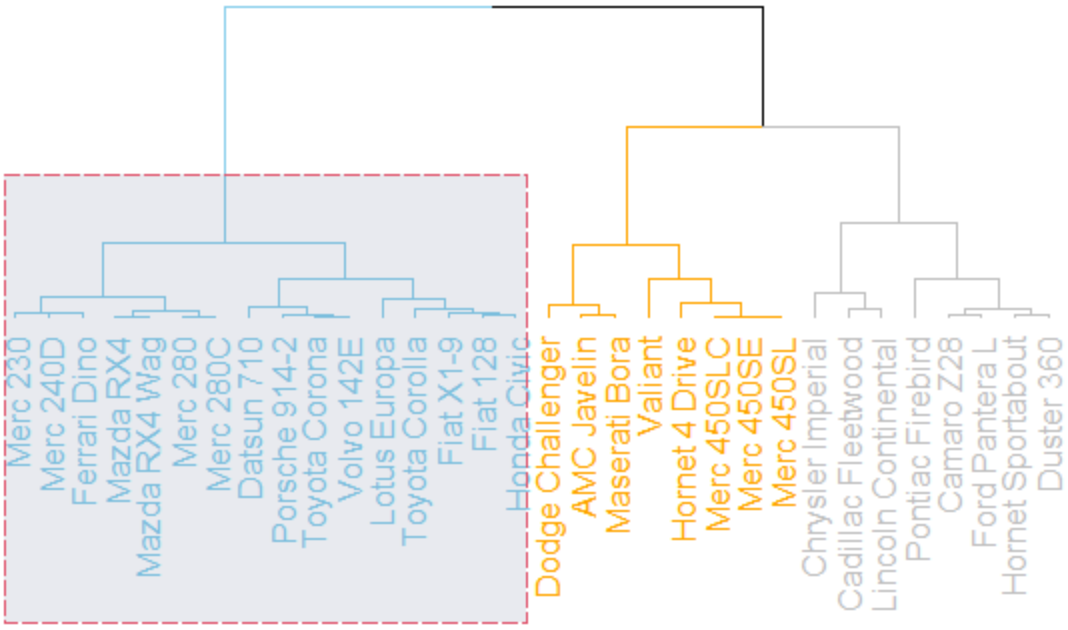

```
#customizing the dendrogram
par(mar=c(1,1,1,7))
dend %>%
  set("labels_col", value = c("skyblue", "orange", "grey"), k=3) %>%
  set("branches_k_color", value = c("skyblue", "orange", "grey"), k = 3) %>%
  plot(horiz=TRUE, axes=FALSE)
```




```
abline(v = 350, lty = 2)
# Highlight a cluster with rectangle
par(mar=c(9,1,1,1))
dend %>%
  set("labels_col", value = c("skyblue", "orange", "grey"), k=3) %>%
  set("branches_k_color", value = c("skyblue", "orange", "grey"), k = 3) %>%
  plot(axes=FALSE)
```



rect.dendrogram(dend, k=3, lty = 5, lwd = 0, x=1, col=rgb(0.1, 0.2, 0.4, 0.1))



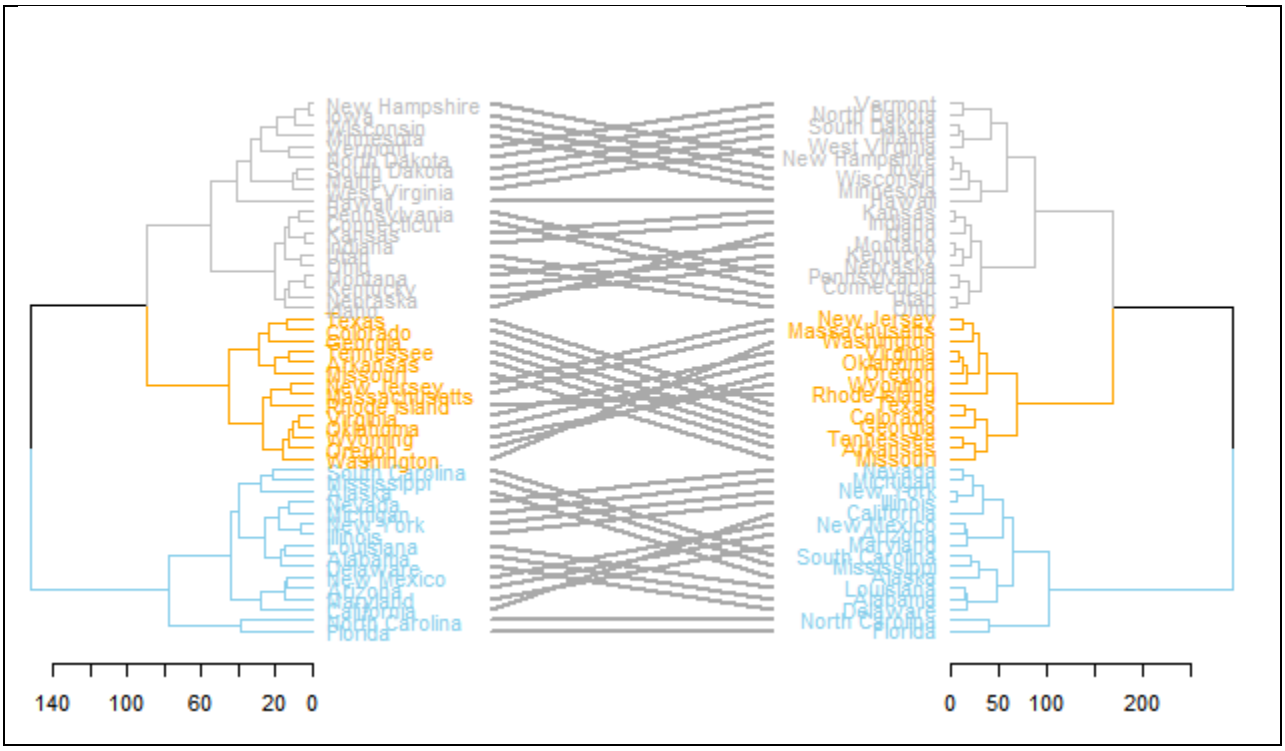
Tanglegram

Dataset: USArrests

Description: Details regarding arrests in US for various crimes

Format: A data frame with 50 observations on 4 variables

```
d1 <- USArrests %>% dist() %>% hclust( method="average" ) %>% as.dendrogram()
d2 <- USArrests %>% dist() %>% hclust( method="complete" ) %>% as.dendrogram()
# Custom these kendo, and place them in a list
dl <- dendlist(
  d1 %>%
    set("labels_col", value = c("skyblue", "orange", "grey"), k=3) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("skyblue", "orange", "grey"), k = 3),
  d2 %>%
    set("labels_col", value = c("skyblue", "orange", "grey"), k=3) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("skyblue", "orange", "grey"), k = 3)
)
# Plot them together
tanglegram(dl,
        common_subtrees_color_lines = FALSE, highlight_distinct_edges = TRUE,
highlight_branches_lwd=FALSE,
        margin_inner=7,
        lwd=2
)
```

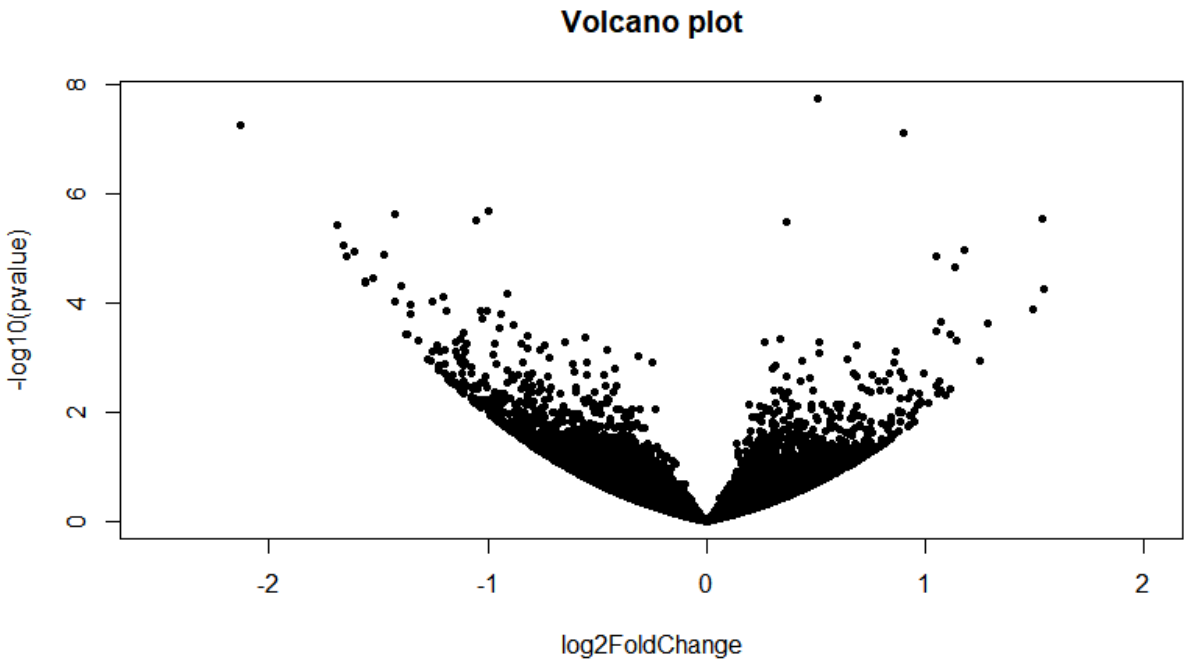

New Hampshire
Texas
Florida
140
100
60
20
0
Vermont
0
50
100
200

6. CASE STUDIES IN LIFE SCIENCE

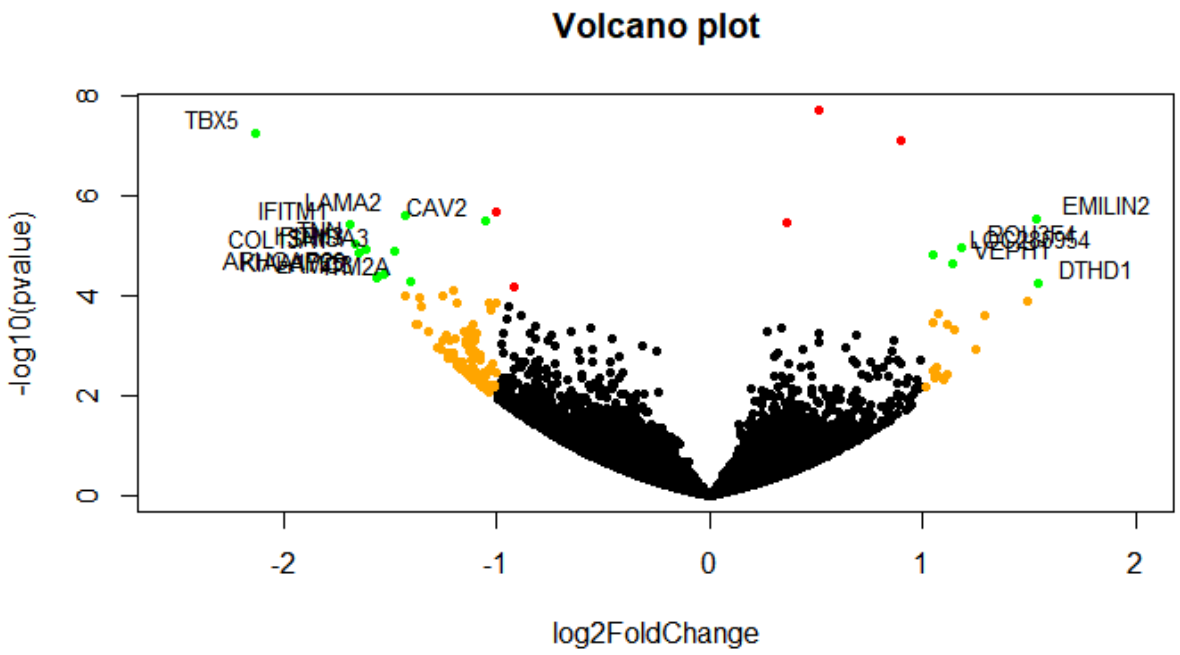
Sona Charles,
ICAR- Indian Institute of Spices Research

Volcano Plot

```
>res <- read.csv("C:/Users/user/Desktop/Workshop/material/volcano.txt", sep="", stringsAsFactors=TRUE)
> head(res)
    Gene log2FoldChange    pvalue      padj
1    DOK6        0.5100 1.861e-08 0.0003053
2    TBX5       -2.1290 5.655e-08 0.0004191
3 SLC32A1        0.9003 7.664e-08 0.0004191
4  IFITM1       -1.6870 3.735e-06 0.0068090
5   NUP93        0.3659 3.373e-06 0.0068090
6 EMILIN2        1.5340 2.976e-06 0.0068090
# Make a basic volcano plot
with(res, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot", xlim=c(-2.5,2)))
```



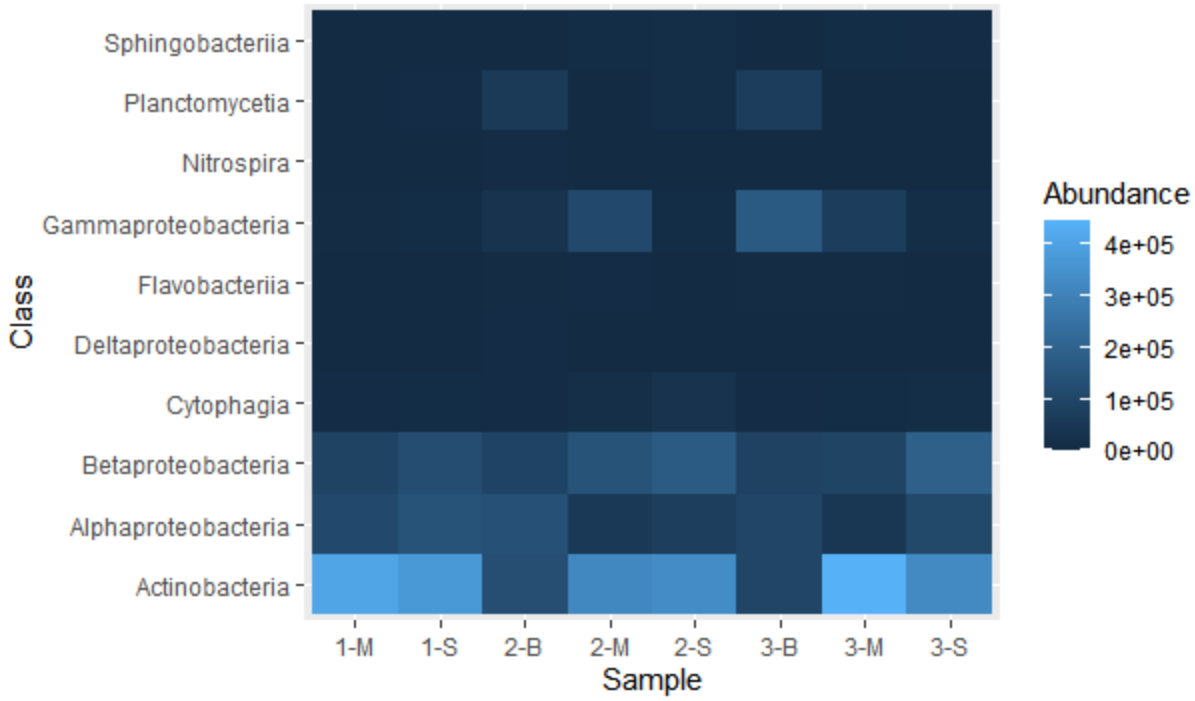
```
# Add colored points: red if padj<0.05, orange of log2FC>1, green if both)
with(subset(res, padj<.05 ), points(log2FoldChange, -log10(pvalue), pch=20,
col="red"))
with(subset(res, abs(log2FoldChange)>1), points(log2FoldChange, -log10(pvalue),
pch=20, col="orange"))
with(subset(res, padj<.05 & abs(log2FoldChange)>1), points(log2FoldChange, -
log10(pvalue), pch=20, col="green"))

# Label points with the textxy function from the calibrate plot
install.packages("calibrate")
library(calibrate)
with(subset(res, padj<.05 & abs(log2FoldChange)>1), textxy(log2FoldChange, -
log10(pvalue), labs=Gene, cex=.8))
```



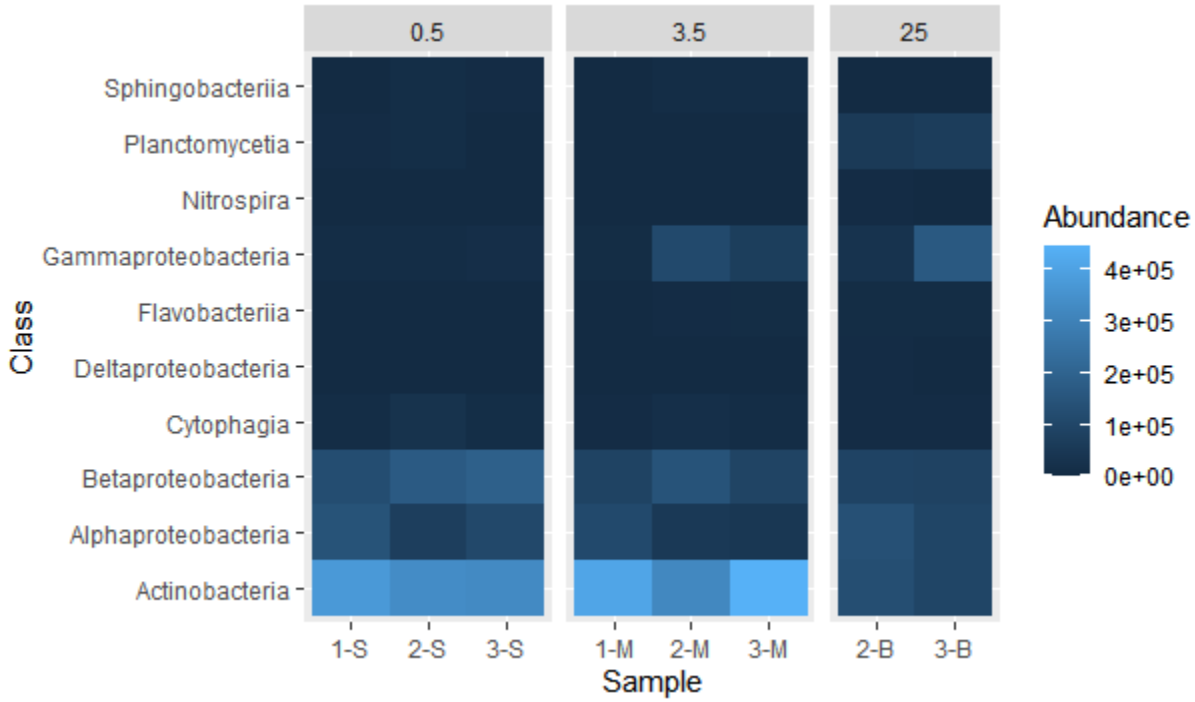
Heatmap

```
library("ggplot2")
```

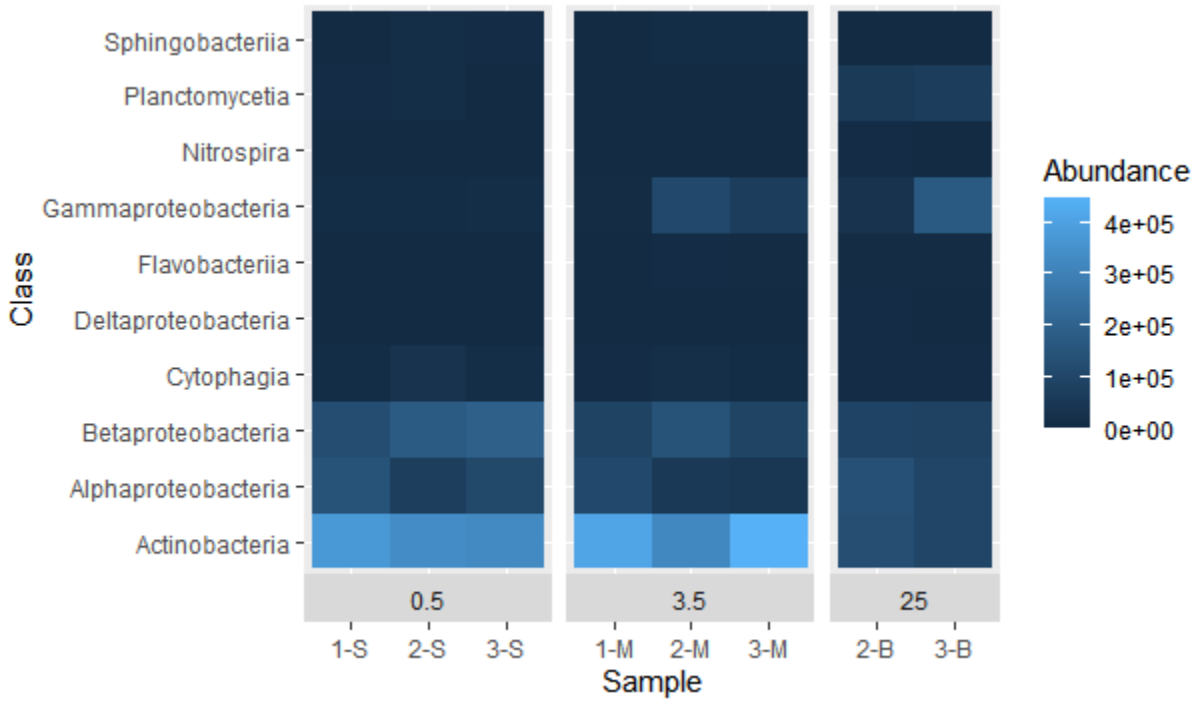
```
> heatdata <- read.csv(file = "C:/Users/user/Desktop/Workshop/material/heat1.csv")
> heatmap <- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                                  y = Class,
+                                                  fill = Abundance)) +
+   geom_tile() +
+   xlab(label = "Sample")
> heatmap
```



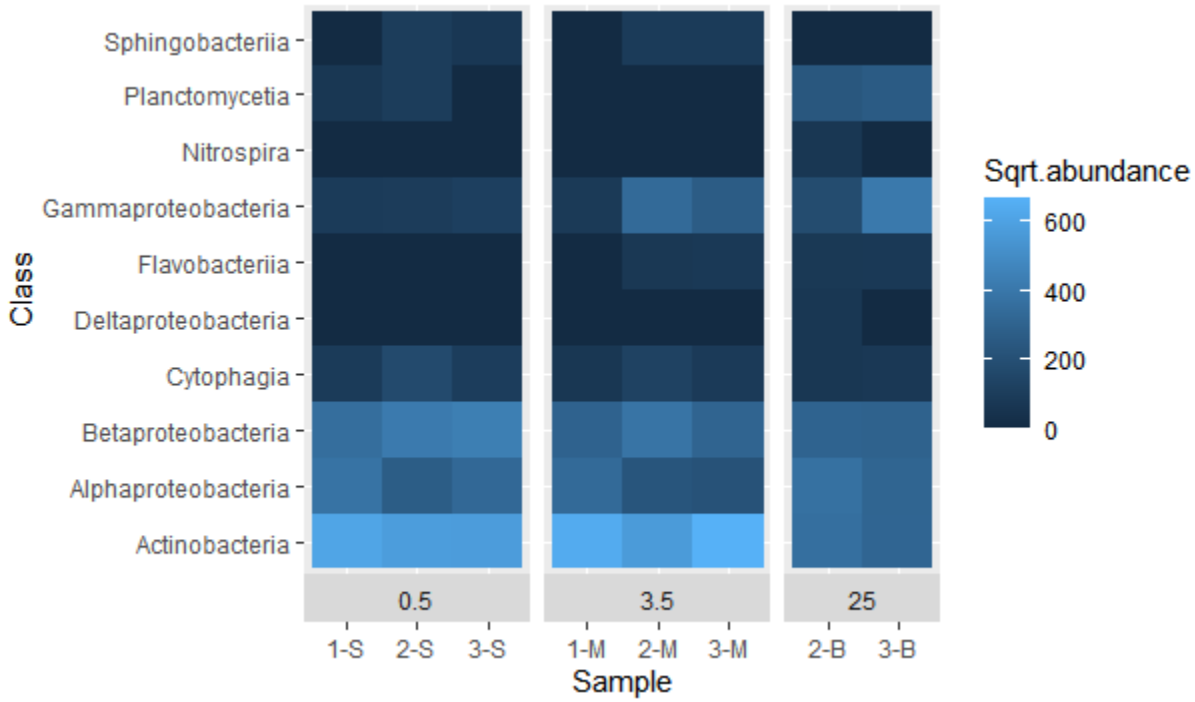

```
> #faceting the heatmap
> facetheatmap <- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                                       y = Class,
+                                                       fill = Abundance)) +
+   geom_tile() +
+   xlab(label = "Sample") +
+   facet_grid(~ Depth, scales = "free_x", space = "free_x")
> facetheatmap
```



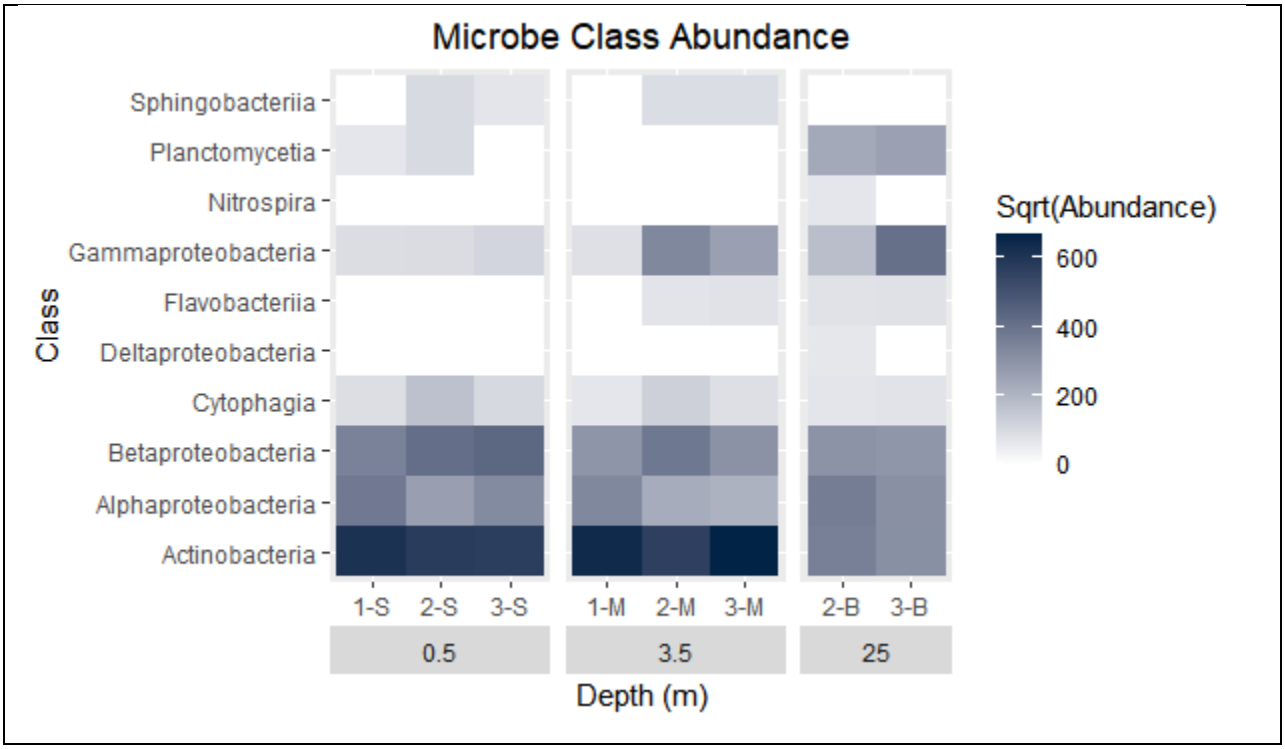

```
> #switching the labels
> labelchange_heatmap <- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                                              y = Class,
+                                                              fill = Abundance)) +
+   geom_tile() +
+   xlab(label = "Sample") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x")
> labelchange_heatmap
```




```
#scaling the data
> heatdata$Sqrt.abundance <- sqrt(heatdata$Abundance)
> scaleheatmap <- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                                       y = Class,
+                                                       fill = Sqrt.abundance)) +
+   geom_tile() +
+   xlab(label = "Sample") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x")
> scaleheatmap
```




```
#add title
> heatdata$Sqrt.abundance <- sqrt(heatdata$Abundance)
> titleheatmap <- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                                       y = Class,
+                                                       fill = Sqrt.abundance)) +
+   geom_tile() +
+   xlab(label = "Depth (m)") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x") +
+   scale_fill_gradient(name = "Sqrt(Abundance)",
+                       low = "#FFFFFF",
+                       high = "#012345") +
+   theme(strip.placement = "outside",
+         plot.title = element_text(hjust = 0.5)) +
+   ggtitle(label = "Microbe Class Abundance")
> titleheatmap
```



Circos Plot/ Idiogram

Dataset: Two files with names location_c.txt and location_nc.txt in BED format

Description: Location of coding and non-coding regions in the genome

Format: BED

```
> library(circlize)
> circos.initializeWithIdeogram(plotType = c("labels", "axis"))
> location_nc <-
read.delim("C:/Users/user/Desktop/Workshop/material/location_nc.txt",
stringsAsFactors=TRUE)
> location_c <-
read.delim("C:/Users/user/Desktop/Workshop/material/location_c.txt",
stringsAsFactors=TRUE)
> circos.genomicDensity(location_nc, col = c("#0000FF80"), track.height = 0.1)
Warning message:
Some of the regions have end position values larger than the end of the
```

chromosomes.

> circos.genomicDensity(location_nc, col = c("#FF000080"), track.height = 0.1)

Warning message:

Some of the regions have end position values larger than the end of the chromosomes.