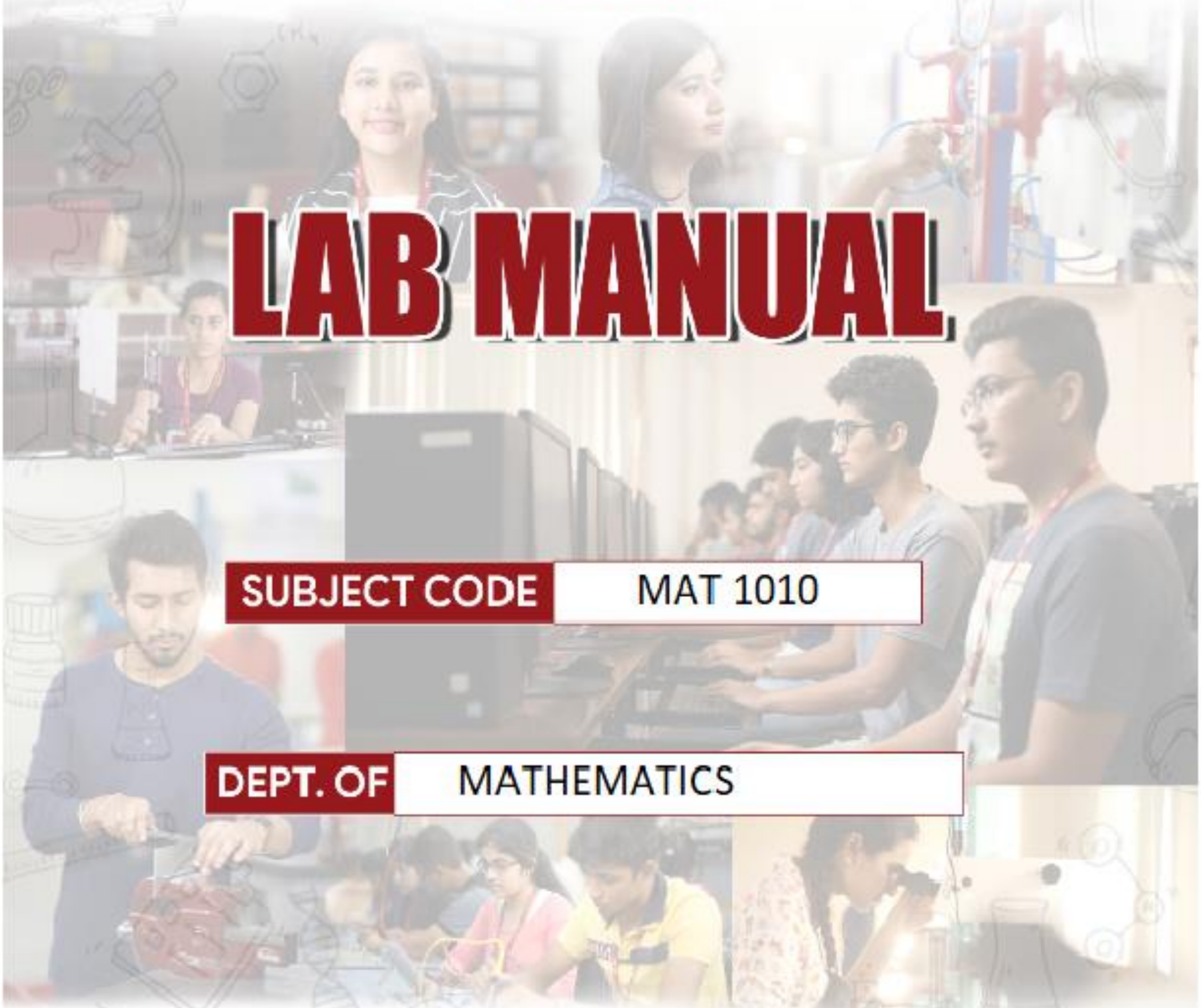
VIT
AP
AMARAVATI



LAB MANUAL

SUBJECT CODE MAT 1010

DEPT. OF MATHEMATICS

VIT-AP UNIVERSITY

Beside AP Secretariat, Amaravati - 522 237, Andhra Pradesh
Phone : 0863-2370444, Email : info@vitap.ac.in

LAB MANUAL

STATISTICS AND PROBABILITY (MAT1010)

SOFTWARE: R

Preface:

Experimental design is a crucial part of data analysis in any field, whether you work in business, health or tech. If you want to use data to answer a question, you need to design an experiment! In this course you will learn about statistical techniques using R, basic experimental design and commonly used statistical tests, such as z-test and t-tests. You will use built-in R data and real world datasets.

Compiled by:

Dr. M. Phani Kumar

Dr. Santanu Mandal

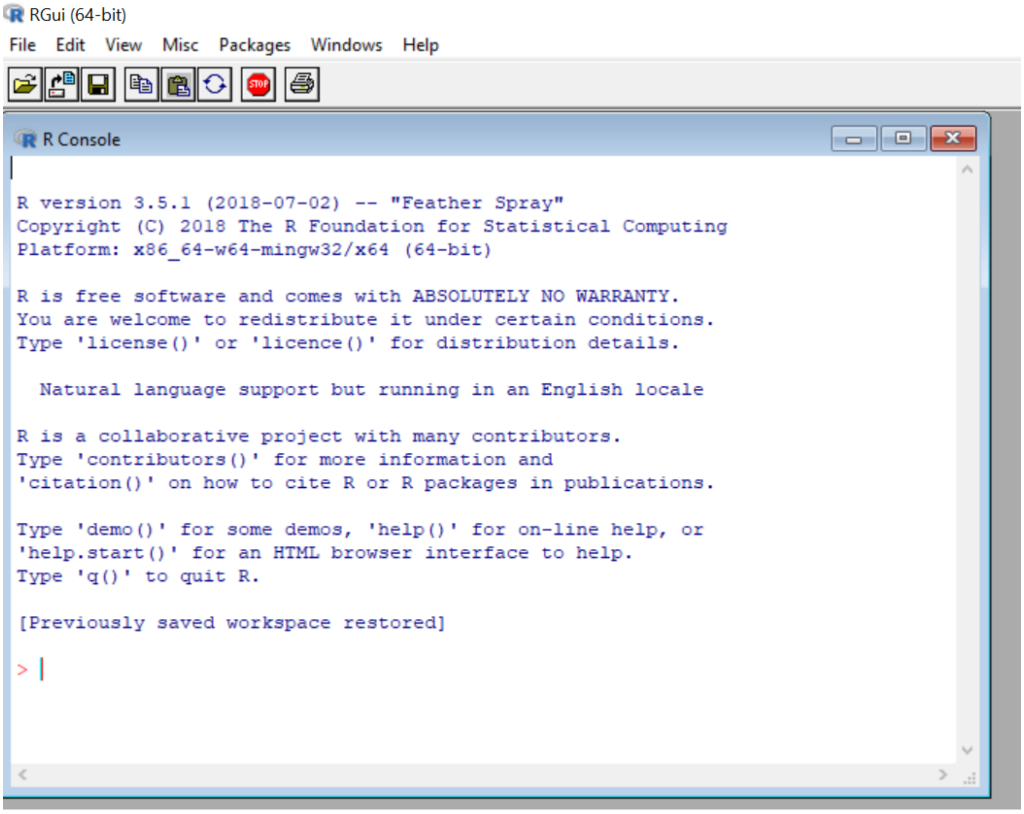
Dr. M. Sudhakar

Department of Mathematics, VIT-AP, India

Contents:

1. Introduction to R

R console



```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> 
```

Viewing Data

Click on the **R** icon to start a session and then load the Rlab package by clicking on packages at the top of the window or by typing

library(Rlab)

(If Rlab cannot be found, that means you haven't installed it. Go back to Start and Set-up R and see how to install Rlab.Installation on a personal machine is done just once, but loading is required for every session.) After loading the Rlab package, you can see a listing of the class data sets, by entering

ls.rlab()

You can see an annotated list by looking of this manual.

You can get a list of the class functions with **ls.rlab("functions")** and a list of everything in the class directory with **ls.rlab("all")**.

You can see an annotated list of the most important functions used in R-Lab by looking at Appendix B of this manual. These include the R-Lab functions and important "native" R functions. In R, **help(functionname)**, for example, **help(plot)**, will usually bring up a nice help window. Also, an R-Lab feature is **ex(functionname)**, for example, **ex(plot)**, gives example usage.

Basic operations:

```
•  > 9                                            output:      [1] 9

•  > pi                                           output:      [1] 3.141593

> 9%%3   # gives reminder                         output:      [1] 0

> 9%/%5 # gives coefficient                       output:      [1] 1

•  n1:n2   #gives list of integers between n1 and n2

•  > 5:12                                          output:      [1]  5  6  7  8  9
   10 11 12

   > 10:-10

   output:  [1]  10   9   8   7   6   5   4   3   2   1   0  -1  -2  -3  -4  -5  -6  -7  -8  -9 -10
   > 1:10%%2                                       output:      [1] 1 0 1 0 1 0 1
   0 1 0

•  > 1:100

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

  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36

  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54

 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72

 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90

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

•  > 1/0              output:       [1] Inf

•  > -1/0             output:        [1] –Inf

•  > 0/0 output:      output:       [1] NaN  (Not a Number)

•  #creates a vector x. Here “c” stands for concatenate

•  > x=c(1,2,3,4,5,6,7,8,9,10)                  output:>x     [1]  1  2  3  4  5  6  7  8
   9 10

.  > x=c(1,3,5,7,10,13,15)                      output:  > x   [1]  1  3  5  7 10 13 15

   > x[4]    #displays 4th number                output:        [1] 7

•  > class(x)                                   output:        [1] “numeric”

   > length(x)          .>min(x)          >max(x)
```

```
[1] 7                    [1] 1                [1] 15
```

- Reading our own data

```
> z<-c(1,2,3,4)        # creates a vector                    output:   > z    [1] 1 2 3 4
> y<-c(“a”,”b”,”c”,”d”)  # creates a vector                output:  > y    [1] „a” „b” „c” „d“
> class(y)                                                 output:     [1] „character”
```

Note: a vector can contain Strings, Numbers or Logical but not a mixture

- `> x=3,    > y=5,  > z=x>y    #is x larger than y           output:  > z    [1] FALSE`
- `> class(z)                                  output:                         [1] logical`
- `> z=2+3i                                    output:   > z                   [1] 2+3i`
- `> class(z)                                  output:                         [1] complex`
- **Logical operations & AND, | OR, ! NEGATION**

```
> x=c(1,2,3,4)
> x<2 | x>3                                  output:     [1]  TRUE FALSE FALSE TRUE
```

- `> all (x>0)                                 output:     [1] TRUE`
- `> any(x>2)                                  output:     [1] TRUE`
- `> sqrt(25)                                  output:     [1] 5`
- `> sqrt(-1)                                  output:     [1] NaN` (Not a Number)
- Warning message: In sqrt(-1) : NaNs produced
- `> sqrt(-1+0i)                               output:     [1] 0+1i`

  or

- `> sqrt(as.complex(-1))         output:    [1] 0+1i`
- `> x<-12                         output:     > x        [1] 12`
- `> x=x+13                      output:      > x        [1] 25`
- `> a=c(1,3,5,7)`
- `> b=c(1,2,4,8)`
- `> a+b,                         output:            [1]  2  5  9 15`
- `> a*b                          output:             [1]  1  6 20 56`

```
• > a/b                         output:         [1] 1.000 1.500 1.250 0.875
• > a^2                         output:          [1]  1  9 25 49
• > z<-list(1,2,3,4)            output:  > z    [[1]]        [1] 1
                                                 [[2]]       [1] 2
                                                 [[3]]        [1] 3
                                                 [[4]]        [1] 4
```

- **> help(list)**
- > z<-list(c(1,2,3),C(4,5,6),C(7,8,9))
- Error in C(4, 5, 6) : object not interpretable as a factor
- > z<-list(c(1,2,3),c(4,5,6),c(7,8,9))

```
  Output:  > z      [[1]]    #single element of the list    [1] 1 2 3
                    [[2]]    [1] 4 5 6
                    [[3]]     [1] 7 8 9
```

- **Vector index**

```
>    p=c(“aa”,”bb”,”cc”,”dd”,”ee”)
        > p[3]         output:   [1] “cc“
       > p{-2]        output:   > p[6]  [1] N
```

Matrices:

Matrix(data, nrow, ncol, byrow, dimnames)

- Enter a matrix of order 2x3

```
  A=matrix(c(2,4,3,1,5,7), nrow=2,ncol=3,byrow=TRUE)
> A
      [,1] [,2] [,3]
 [1,]    2    4    3
 [2,]    1    5    7
```

A[m,] display m^{th} row

A[,n]   display n^{th} column

- > A[2,3]   #Display 2nd row 3rd column element   output: [1] 7
- > A[2,]   #Display 2nd row elements   output: [1] 1 5 7

-
  ```
  > A[,c(1,3)]  # displays 1th and 3th colums only
  Output:
       [,1] [,2]
  [1,]    2    3
  [2,]    1    7
  ```

-
  ```
  > t(A)                    #transpose of A
           [,1] [,2]
    [1,]    2    1
    [2,]    4    5
    [3,]    3    7
  ```

-
  ```
  > C=matrix(c(6,2),nrow=1,ncol=2)
  > C
       [,1] [,2]
   [1,]    6    2
  ```

-
  ```
  > B=matrix(c(2,4,3,1,5,7),nrow=3,ncol=2,byrow=FALSE)
     > B
      [1,]    2    1
      [2,]    4    5
      [3,]    3    7
  ```

-
  ```
  > C=matrix(c(7,4,2),nrow=3,ncol=1)
      [1,]    7
      [2,]    4
      [3,]    2
  ```

-
  ```
  > cbind(B,C)          #column bind
   [1,]    2    1    7
   [2,]    4    5    4
   [3,]    3    7    2
  ```

-
  ```
  > D=matrix(c(6,20),nrow=1,ncol=2)
     [1,]    6   20
  ```

- > rbind(B,D)   #row bind

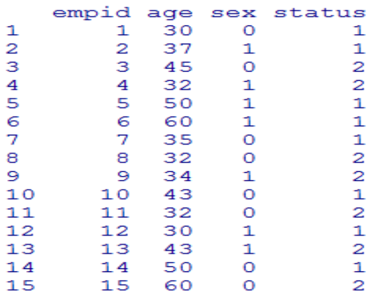
```
[1,]   2   1
[2,]   4   5
[3,]   3   7
[4,]   6  20
```

- > a=c(1,2,3)
- > b=c(“a”,”b”,”c”)
- > d=c(TRUE, FALSE, TRUE)
- > df=data.frame(a,b,d)   #df is a data frame

output:

```
> df
      a b     d
    1 a TRUE
    2 b FALSE
    3 c TRUE
```

Example: to create a data



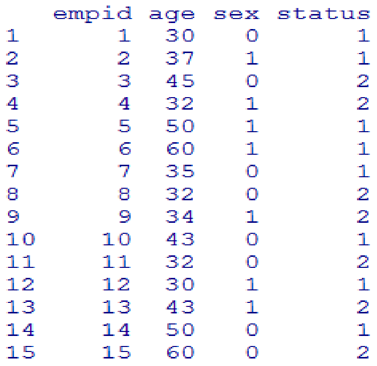
```
      empid  age  sex  status
1         1   30    0       1
2         2   37    1       1
3         3   45    0       2
4         4   32    1       2
5         5   50    1       1
6         6   60    1       1
7         7   35    0       1
8         8   32    0       2
9         9   34    1       2
10       10   43    0       1
11       11   32    0       2
12       12   30    1       1
13       13   43    1       2
14       14   50    0       1
15       15   60    0       2
```

- > empid=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)   #creating a vector empid
- > age=c(30,37,45,32,50,60,35,32,34,43,32,30,43,50,60)
- > sex=c(0,1,0,1,1,1,0,0,1,0,0,1,1,0,0)
- > empinfo=data.frame(empid,age,sex,status)   #creates a combining vectors

  > empinfo

  gives a matrix form data

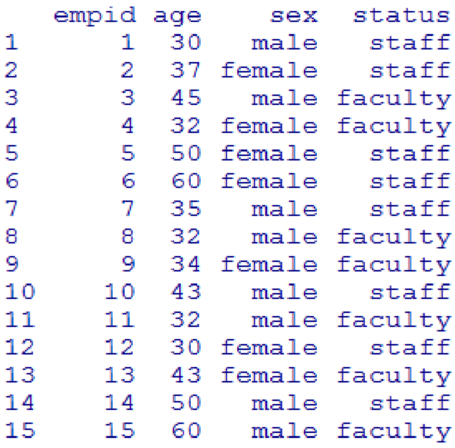
  output:



```
   empid age sex status
1      1  30   0      1
2      2  37   1      1
3      3  45   0      2
4      4  32   1      2
5      5  50   1      1
6      6  60   1      1
7      7  35   0      1
8      8  32   0      2
9      9  34   1      2
10    10  43   0      1
11    11  32   0      2
12    12  30   1      1
13    13  43   1      2
14    14  50   0      1
15    15  60   0      2
```

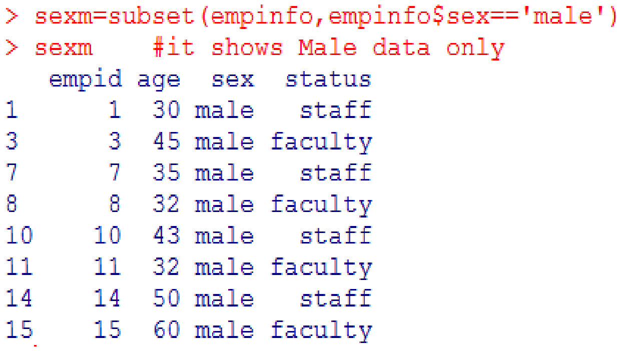
- > empinfo[7,2]   output: [1] 35
- > empinfo[3,]   Output: empid age sex status

  3   3 45 0 2
- > empinfo[3:8,1]   output: [1] 3 4 5 6 7 8
- > empinfo$age[empinfo$sex>0]   output: [1] 37 32 50 60 34 30 43
- > empinfo$sex=factor(empinfo$sex, labels=c(“male”,”female”))
- > empinfo$status=factor(empinfo$status, labels=c(“staff”,”faculty”))
- >empinfo



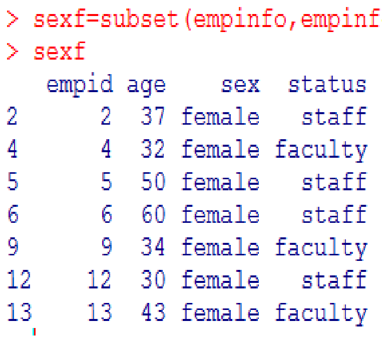
```
   empid age    sex  status
1      1  30   male   staff
2      2  37 female   staff
3      3  45   male faculty
4      4  32 female faculty
5      5  50 female   staff
6      6  60 female   staff
7      7  35   male   staff
8      8  32   male faculty
9      9  34 female faculty
10    10  43   male   staff
11    11  32   male faculty
12    12  30 female   staff
13    13  43 female faculty
14    14  50   male   staff
15    15  60   male faculty
```

it show male date only



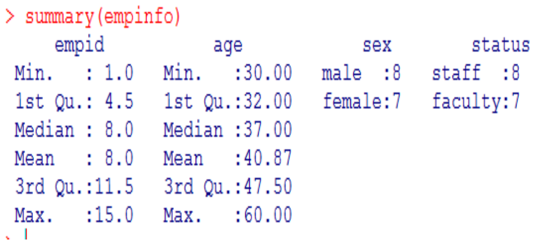
```
> sexm=subset(empinfo,empinfo$sex=='male')
> sexm     #it shows Male data only
   empid age  sex  status
1      1  30 male   staff
3      3  45 male faculty
7      7  35 male   staff
8      8  32 male faculty
10    10  43 male   staff
11    11  32 male faculty
14    14  50 male   staff
15    15  60 male faculty
```

- #The following command shows female data only



```
> sexf=subset(empinfo,empinfo$sex=='female')
> sexf
   empid age    sex  status
2      2  37 female   staff
4      4  32 female faculty
5      5  50 female   staff
6      6  60 female   staff
9      9  34 female faculty
12    12  30 female   staff
13    13  43 female faculty
```

- Summary statistics for empinfo data



```
> summary(empinfo)
     empid            age            sex         status 
 Min.   : 1.0   Min.   :30.00   male  :8   staff  :8  
 1st Qu.: 4.5   1st Qu.:32.00   female:7   faculty:7  
 Median : 8.0   Median :37.00                         
 Mean   : 8.0   Mean   :40.87                         
 3rd Qu.:11.5   3rd Qu.:47.50                         
 Max.   :15.0   Max.   :60.00                         
```

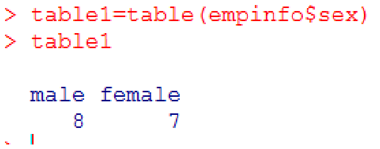
- > summary(empinfo$age)

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

30.00 32.00 37.00 40.87 47.50 60.00

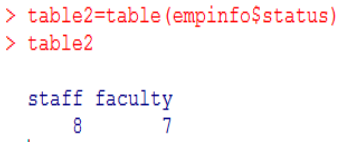
- Creating one-way table

1. For sex



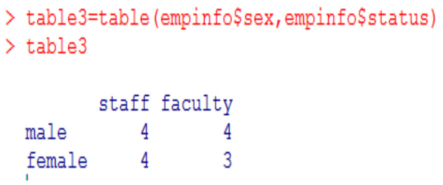
```
> table1=table(empinfo$sex)
> table1

  male female 
     8      7 
```

2. For status



```
> table2=table(empinfo$status)
> table2

  staff faculty 
      8       7 
```

3. Creating two-way table



```
> table3=table(empinfo$sex,empinfo$status)
> table3
        
         staff faculty
  male       4       4
  female     4       3
```

1.1 Experiment:

1. Enter the elements {5,12,7,2,6,45,11,3,63} and store in a variable x
   - Display x values
   - Find the number of elements of x

- Find the 5th, 8th elements
- Find the minimum element of x.
- Find the maximum element of x.

Enter the data {1, 2, …. ,19,20} in a variable x.

- Find the 3rd element in the data list.
- Find 3rd to 5th element in the data list.
- Find 2nd, 5th, 6th, and 12th element in the list.
- Print the data as {20, 19, …, 2, 1} without again entering the data.

2. Reading a data file and working with it:

a) Read the file first and store it in a.

b) How many rows are there in this table? How many columns are there?

c) How to find the number of rows and number of columns by a single command?

d) What are the variables in the data file?

e) If the file is very large, naturally we cannot simply type `a', because it will cover the entire screen and we won't be able to understand anything. So how to see the top or bottom few lines in this file?

f) If the number of columns is too large, again we may face the same problem. So how to see the first 5 rows and first 3 columns?

g) How to get 1st, 3rd, 6th, and 10th row and 2nd, 4th, and 5th column?

h) How to get values in a specific row or a column?

2. Function

- **Seq()** # Generate regular sequences

Ex: > seq(1,21, by=2) output: [1] 1 3 5 7 9 11 13 15 17 19 21

Typical usages are

seq(from, to)

seq(from, to, by=)

seq(from, to, length.out=)

seq(along.with=)

seq(from)

seq(length.out=) More details: *? seq* *help(seq)*

- **Rep( )**

Ex: to repeat 4, 20 times

> rep(4,20) output: [1] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4

more about repeat command can be known by *help(rep) or ?rep*

To repeat each value twice up to

> rep(1:10, rep(2,10)) output: [1] 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10

- **sort( )** # read the elements

v <- c(3,8,4,5,0,11, -9, 304) output: > v [1] 3 8 4 5 0 11 -9 304

- **Sort the elements of the vector.**

> sort.result <- sort(v)

> print(sort.result) output: [1] -9 0 3 4 5 8 11 304

# Sort the elements in the reverse order.

v <- c(3,8,4,5,0,11, -9, 304)

- > revsort.result <- sort(v, **decreasing** = TRUE)

> print(revsort.result) output: [1] 304 11 8 5 4 3 0 -9

- **list()** it is one way of organizing multiple pieces of output from functions.

NOTE: In contrast to a vector, in which all elements must be of the same mode, R's list

structure can combine objects of different types.

```
> z<-list(a="VIT-AP",b=25)

Output :   > z           $`a`          [1] "VIT-AP"          $b          [1] 25
```

- **length()** gives the number of components in a vector or list

Example:

```
> z<-list(a="VIT-AP",b=25)

    Output: > z        $`a`         [1] "VIT-AP"          $b          [1] 25

> length(z)    output:                    [1] 2
```

- **Function:**

A function is a total number of statements prepared collectively to carry out a precise

task.

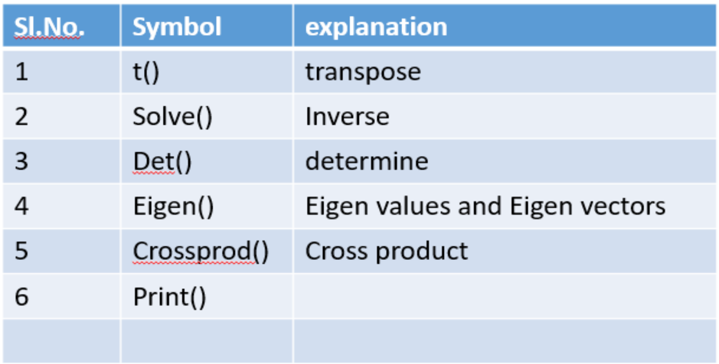
Syntax:

```
function_name<-function (arg1, arg2, ….)

 {                    Function Body

      }
```

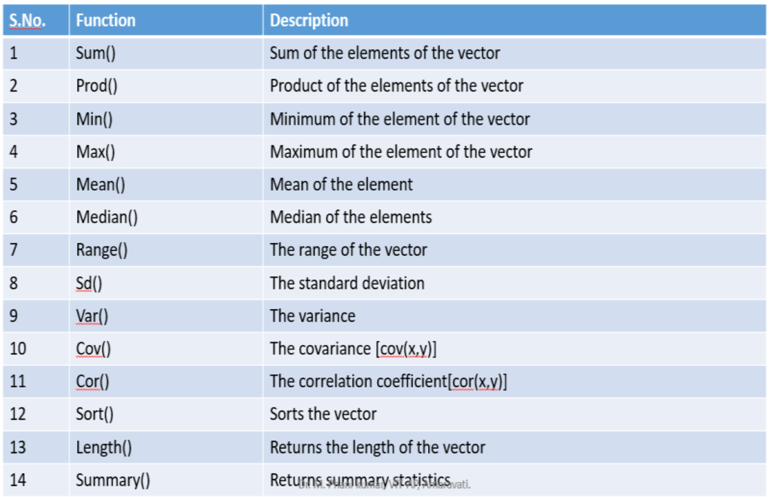
- Functions are of two types

  1. Built in function

  2. Custom functions



Sl.No.	Symbol	explanation
1	t()	transpose
2	Solve()	Inverse
3	Det()	determine
4	Eigen()	Eigen values and Eigen vectors
5	Crossprod()	Cross product
6	Print()	



S.No.	Function	Description
1	Sum()	Sum of the elements of the vector
2	Prod()	Product of the elements of the vector
3	Min()	Minimum of the element of the vector
4	Max()	Maximum of the element of the vector
5	Mean()	Mean of the element
6	Median()	Median of the elements
7	Range()	The range of the vector
8	Sd()	The standard deviation
9	Var()	The variance
10	Cov()	The covariance [cov(x,y)]
11	Cor()	The correlation coefficient[cor(x,y)]
12	Sort()	Sorts the vector
13	Length()	Returns the length of the vector
14	Summary()	Returns summary statistics

Create a function:

- > # Create a function to print squares of numbers in sequence.
- new.function <- function(a) {   for(i in 1:a) {   b <- i^2

  print(b) }   }
- > new.function(6)

```
[1]  1
 [1]  4
 [1]  9
[1]  16
[1]  25
[1] 36
```

About ‘Packages’

- In R, a package is a module containing functions, data and documents
- To see which packages are loaded, run
- search()

```
[1] ".GlobalEnv"        "package:stats"     "package:graphics"
[4] "package:grDevices" "package:utils"     "package:datasets"
[7] "package:methods"   "Autoloads"         "package:base"
```

- To install any package, run

Install.packages(“knitr”)   # here an example ‘knitr’ is taken

2.1 Experiment:

1. Few simple statistical measures:

a) Enter data as 1,2, … ,10.

b) Find sum of the numbers.

c) Find mean, median.

d) Find sum of squares of these values.

e) Find the value of $\frac{1}{n}\Sigma|x_i - \bar{x}| = 1$, This is known as mean deviation about mean ($MD\bar{x}$).

f) Check whether $MD\bar{x}$ is less than or equal to standard deviation.

g) Find standard deviation using formula.

2. Random sampling

a) In R, you can simulate these situations with the sample function. Pick five numbers at random from the set 1:40.

b) Notice that the default behaviour of sample is sampling without replacement. That is, the samples will not contain the same number twice, and size obviously cannot be bigger than the length of the vector to be sampled. If you want sampling with replacement, then you need to add the argument replace=TRUE.

Sampling with replacement is suitable for modelling coin tosses or throws of a die. So, for instance, simulate 10 coin tosses.

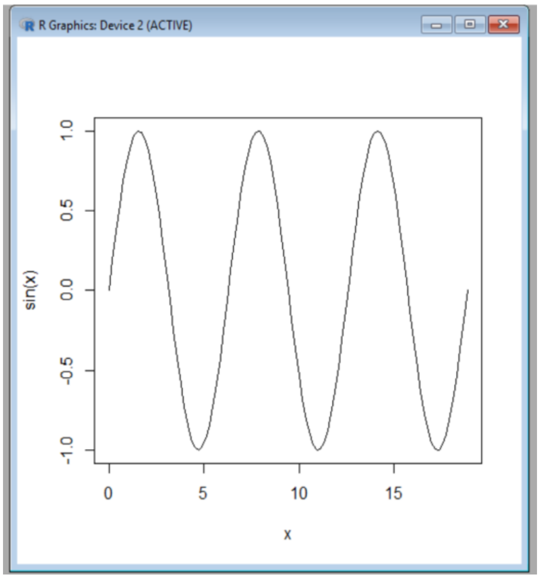
c) In fair coin-tossing, the probability of heads should equal the probability of tails, but the idea of a random event is not restricted to symmetric cases. It could be equally well applied to other cases, such as the successful outcome of a surgical procedure. Hopefully, there would be a better than 50% chance of this. Simulate data with nonequal probabilities for the outcomes (say, a 90% chance of success) by using the prob argument to sample.

3. Plot

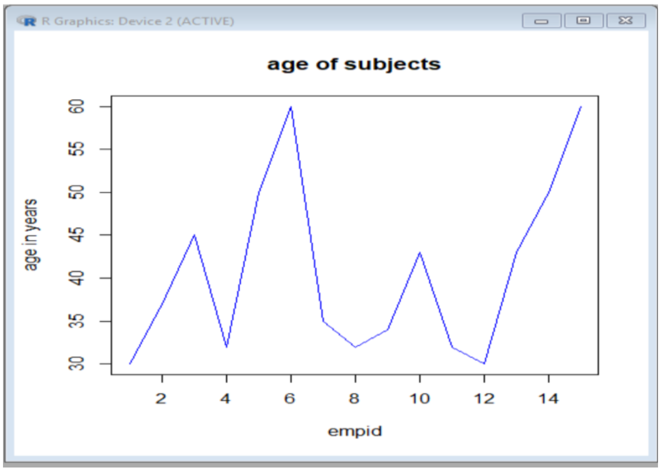
Graphs in R:

- **Curve( )**

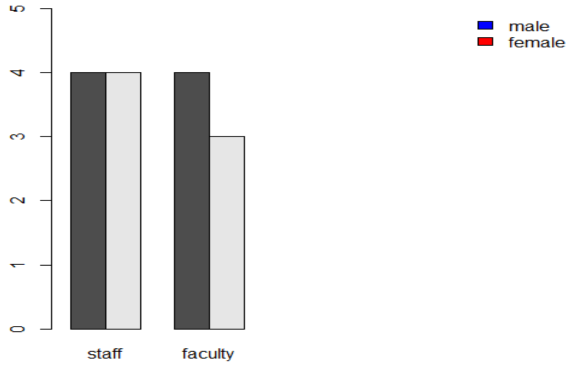
```
> curve(expr = sin, from =0, to = 6 * pi)
```



- **Plot( )**
- > plot(empinfo$age,type="l", main="age of subjects", xlab="empid", ylab="age in years", col="blue")

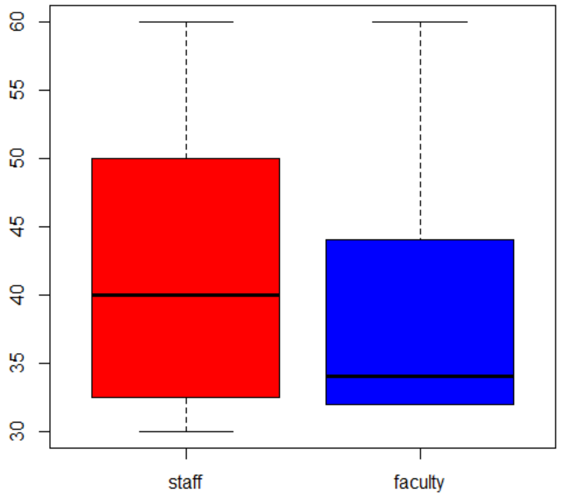


- **Barplot()**
- table5=table(empinfo$sex,empinfo$status)
- > barplot(table5,beside=T, xlim=c(1,15), ylim=c(0,5))
- >legend("topright",legend=rownames(table5),fill=c('blue','red'),bty="n")



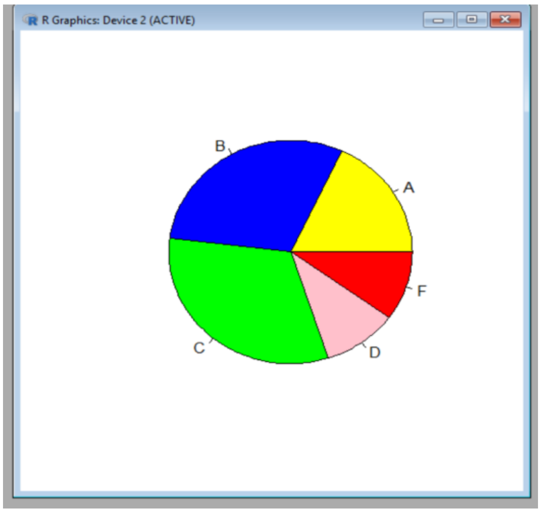
- **Boxplot()**

```
boxplot(empinfo$age ~ empinfo$status, col=c("red", "blue"))
```



- **Pie()**

```
 x=c(18,30,32,10,10)
> labels=c("A","B","C","D","F")

> pie(x,labels, col=c("yellow","blue","green","pink","red"))
```



3.1.Experiment:

Calculate simple statistical measures using the values in the data file.

a) Find means, medians, standard deviations of Price, Floor Area, Rooms, and Age.

b) How many houses have central heating and how many don't have?

c) Plot Price vs. Floor, Price vs. Age, and Price vs. rooms, in separate graphs.

d) Draw histograms of Prices, FloorArea, and Age.

e) Draw box plots of Price, FloorArea, and Age.

f) Draw all the graphs in (c), (d), and (e) in the same graph paper.

4. Discrete Random Variables

4.1 Binomial Distribution

Bernoulli trails

let n represents the number of trails and $\boldsymbol{x}$ the number of success.

The following conditions should be satisfied

1. There are only two possible outcomes for each trail, arbitrarily called success and failure.
2. The probability of the success is the same for each trail.
3. There are n trails where n is a constant.
4. The n trails are independent.

Trails satisfying these conditions are called Bernouli trails.

If this conditions are not satisfied the theory which we studied is not applied.

- **Probability function of a binomial distribution:**

to find, the probability of getting $\boldsymbol{x}$ success in n trails.

the probability of success be p

the probability of failure is $(1-p)$ i.e., q

In n trails, the total number of possible ways of obtaining $\boldsymbol{x}$ success and $(n-x)$ failures is n_{c_x} all of which are mutually disjoint.

Hence the required probability is

$$P(X=\boldsymbol{x}) = n_{c_x}\, p^x q^{n-x}\,,\ \ (x=0,1,2,3,\dots\dots)$$

- For a B.D.

$$P(X=x) = n_{c_x}\, p^x q^{n-x}\,,\ \ (x=0,1,2,3,\dots\dots)$$

Here

1. $\sum P(x) = 1.$

Sol: $\sum_{x=0}^{n} P(x) = \sum_{x=0}^{n} n_{c_x}\, p^x q^{n-x}$

$$= n_{c_0}\, q^n + n_{c_1}\, p^1 q^{n-1} + n_{c_2}\, p^2 q^{n-2} + \dots + n_{c_n}\, p^n$$

$$= (q+p)^n = 1^n = 1$$

- Mean$(\mu) = \sum \boldsymbol{x}\, P(X=x) = \sum \boldsymbol{x}\ n_{c_x}\, p^x q^{n-x}$

$$= np$$

- variance$(\sigma) = \sum (x-\mu)^2\, P(X=x)$

$$= \sum (x-\mu)^2\ n_{c_x}\, p^x q^{n-x}$$

$$= npq$$

Standard deviation(S.D.)=$\sqrt{npq}$

Binomial distribution(B.D.) with R

Syntax:

- Is x is the number of Binomial events
- $P(X=x)$

dbinom(x, size, prob)

- $P(X \le x)$ can be computed by

pbinom(x, size, prob, lower.tail=TRUE)

- $P(X > X)$

pbinom(x, size, prob, lower.tail=FALSE)

- qbinom() gives the quantiles for the binomial distribution

- rbinom() is used to generate binomial pseudorandom numbers

	Syntax	**Description**
1	dbinom	Gives the density
2	pbinom	Gives the distribution function
3	qbinom	Gives the quantile function
4	rbinom	Generates random generates

Worked out Example:

If 20% of the bolts produced by a machine are found by defective. Determine the probability that out of 4 bolts chosen at random
(a) one (b) zero (c) at most two will be defective.
Solution: success = a bolt being defective

$$p = \frac{20}{100} = \frac{1}{5}, \; q = \frac{80}{100} = \frac{4}{5}$$, the trails are n=4

The probability $P(X = x) = n_{c_x}\, p^x q^{n-x}$

$$= 4_{c_x} \left(\frac{1}{5}\right)^x \left(\frac{4}{5}\right)^{4-x}$$

(a) $P(X = 1) = 0.4096$
(b) $P(X = 0) = 0.4096$
At most two
$P(X \leq 2) = 0.9728$

4.1.1. Experiment:

1. The incidence of an occupational disease in an industry is such that the workers have a 20% chance of suffering from in it, what is the probability that out of 6 workers at random, four or more will suffer from the disease?

(ans: 0.0175)

4.2. Poisson Distribution

The Poisson distribution can be obtained as a limiting case of binomial distribution under the following conditions:
(a). n, the number of trails is very large i.e., $n \to \infty$.
(b). P, the probability of success for each trail is very small i.e., $p \to 0$.
(c). $np = \lambda$(say) is finite.
The probability distribution function is

$$P(X = x) = \frac{e^{-\lambda}\,\lambda^{r}}{r!}$$

The probability distribution function is $P(X = x) = \frac{e^{-\lambda}\,\lambda^{r}}{r!}$
Sum of probabilities

$$\sum P(x) = \sum_{r=0}^{\infty} \frac{e^{-\lambda}\,\lambda^{r}}{r!} = e^{-\lambda} + \frac{e^{-\lambda}\,\lambda}{1!} + \frac{e^{-\lambda}\,\lambda^{2}}{2!} + \frac{e^{-\lambda}\lambda^{3}}{3!} + \cdots$$

$$= e^{-\lambda}\left[1 + \frac{\lambda}{1!} + \frac{\lambda^{2}}{2!} + \frac{\lambda^{3}}{3!} + \cdots\right] = e^{-\lambda}\left[e^{\lambda}\right] = 1$$

$\therefore$ sum of probabilities is one.

- Mean of the poison distribution
  Mean$(\mu) = E(X) = \sum_{x=0}^{\infty} x\,P(x)$

$$= \sum_{x=0}^{\infty} x\,\frac{e^{-\lambda}\,\lambda^{x}}{x!} = e^{-\lambda}\sum_{x=1}^{\infty}\frac{\lambda^{x}}{(x-1)!}$$

$$= e^{-\lambda}\sum_{x=1}^{\infty}\lambda\,\frac{\lambda^{x-1}}{(x-1)!} = e^{-\lambda}\,\lambda\sum_{x=1}^{\infty}\frac{\lambda^{x-1}}{(x-1)!}$$

$$= e^{-\lambda}\,\lambda\left(e^{\lambda}\right) = \lambda$$

- Variance of the Poission distribution
  variance= $\sum(x-\mu)^{2}\,P(x) = \lambda$

Syntax:

Is x is the number of Poisson events

- $P(X = x)$ dpois(x, lambda)
- $P(X \le x)$ can be computed by
  ppois (x, lambda, lower.tail=TRUE)
- $P(X \ge X)$ or 1-$P(X \le x)$
  ppois (x, lambda, lower.tail=FALSE)
- qpois () gives the quantiles for the binomial distribution
- rpois () is used to generate binomial pseudorandom numbers

Worked out examples:

1. A car-hire firm has two cars which it hires out day by day. The number of demands for a car on each day is distributed as a Poisson distribution with mean 1.5. Calculate the proportion of days (i) on which there is no demand (ii) on which demand is refused.

Solution: Given the mean as $\lambda = 1.5$

$P(X = x) = \frac{e^{-\lambda}\,\lambda^{r}}{r!}$

P(no demand)=P(x=0)=0.2231

Some demand is refused if the number of demands is more than two

P(demand refused)=P$(r > 2) = 0.1913$

2. A manufacture of cotter clips knows that 5% of his product is defective. If he sells clips inboxes of 100 and guarantees that not more than 10 clips will be defective. What is the probability that a box will fail to meet the guarantee quality?

Solution:

Given $n = 100$, $p = \frac{5}{100} = 0.05,$ $\lambda = np = 100X0.05 = 5$

$$P(X = x) = \frac{e^{-5}\, 5^{x}}{x!}$$

If the defects are more than 10 it is not guaranteed

$$P(X > 10) = 0.0137$$

4.2.1. Experiments:

1. The probability that an individual suffers a bad reaction from an injection is 0.001. Determine the probability that out of 2000 individuals (a) three (b) more than two will suffer the reaction. (answers: (a)0.18044 (b) 0.3233)

2. If 2% of light bulbs are defective. Find (i). at least one is defective. (ii). Exactly 7 are defective. (iii). $P(1 < x < 8)$ in a sample of 100.

(answers: (i) 0.8647 (ii) 0.0034 (iii) 0.593)

3. If 10% of the tools produced in a certain manufacturing process turns out to be defective.

Find the probability that a sample of 10 tools chosen at random, exactly two will be defective by using

(a) Binomial distribution (b) Poisson Distribution.

(answers: (a) 0.1937 (b) 0.1839)

5. Continuous Random Variables

5.1. Normal Distribution

A continuous random variable X is said to have a normal distribution with parameter mean (μ) and

standard deviation (σ), if the probability density function is

$$P(X = x) = \frac{1}{\sqrt{2\pi}\ \sigma}\ e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where $-\infty < x < \infty,\ -\infty < \mu < \infty,\ \sigma > 0$

$X \sim N(\mu, \sigma)$ means: X follows the Normal distribution with mean μ and S.D. σ.

Properties: here (i) $P(x) \geq 0$ (ii) $\int_{-\infty}^{\infty} P(x)dx = 1$

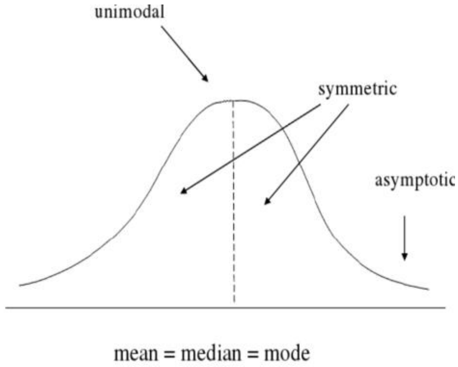
- Mean of a normal distribution:

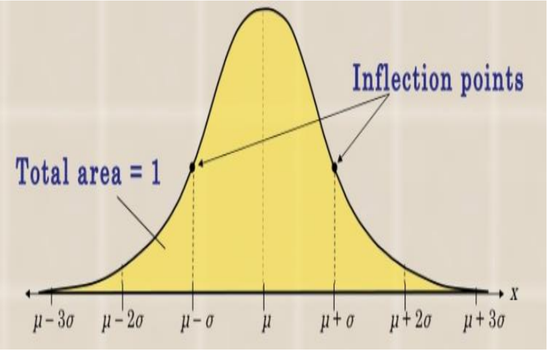
$$\text{mean } (\mu) = \int_{-\infty}^{\infty} x\, P(x)\, dx = \int_{-\infty}^{\infty} x \frac{1}{\sqrt{2\pi}\ \sigma}\ e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \mu$$

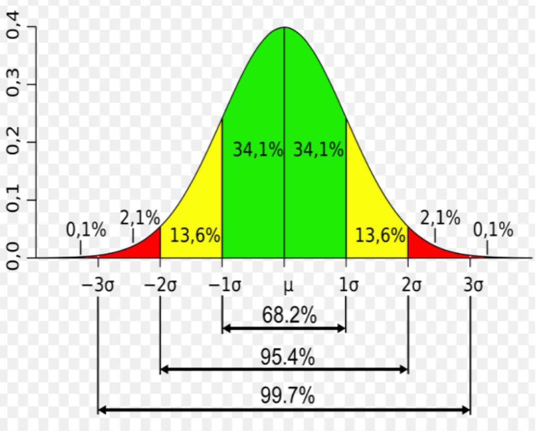
Z-value (standard value)

it is the number of standard deviations that a particular x value is away from the mean. The formula for finding the z value is

$$z = \frac{\boldsymbol{value}\ - mean}{standard\ deviation} = \frac{\boldsymbol{x} - \mu}{\sigma}$$







Calculation part:

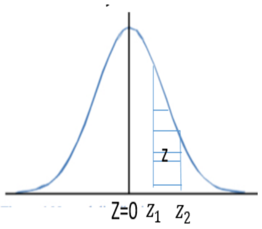
- When μ and σ are given to find $P(x_1 \leq x \leq x_2)$ with $x_1 \leq x_2$

since $z = \frac{x-\mu}{\sigma}$

corresponding to x_1, x_2 we get z_1, z_2 respectively

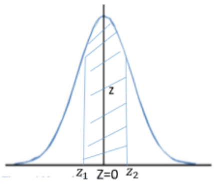
1. To find $P(x_1 \leq x \leq x_2)$=$P(z_1 \leq z \leq z_2)$

(a) if both z_1, z_2 are positive (or both negative) then,



$P(x_1 \leq x \leq x_2)$
=$A(z_2) - A(z_1)$
=(Area under the normal curve from 0 to z_2)-(Area under the normal curve from 0 to z_1)

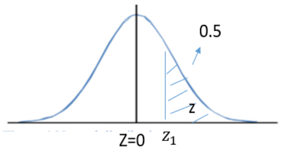
- If $z_1 < 0$, $z_2 > 0$
$P(x_1 \leq x \leq x_2)$=$A(z_2) + A(z_1)$



2. To find $P(z > z_1)$

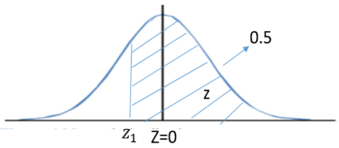
(a) if $z_1 > 0$

then $P(z > z_1) = 0.5 - A(z_1)$



(b) if $z_1 < 0$

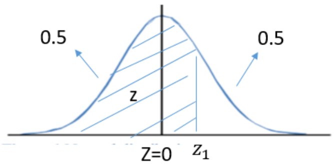
then $P(z > z_1) = 0.5 + A(z_1)$



3. To find $P(z < z_1)$=1 $- P(z > z_1)$

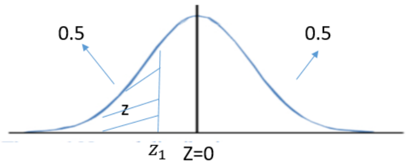
(a) if $z_1 > 0$

then $P(z < z_1) = 0.5 + A(z_1)$



(b) if $z_1 < 0$

then $P(z < z_1) = 0.5 - A(z_1)$



Using R

R has four in built functions to generate normal distribution. They are described below.

dnorm (x, mean, sd)
pnorm(x, mean, sd)
qnorm(p, mean, sd)
rnorm(n, mean, sd)

Following is the description of the parameters used in above functions:

x is a vector of numbers.

p is a vector of probabilities.

n is number of observations (sample size).

mean is the mean value of the sample data. It's default value is zero.

sd is the standard deviation. It's default value is 1

dnorm() :

This function gives height of the probability distribution at each point for a given mean and standard deviation.

pnorm():

This function gives the probability of a normally distributed random number to be less that the value of a given number. It is also called "Cumulative Distribution Function".

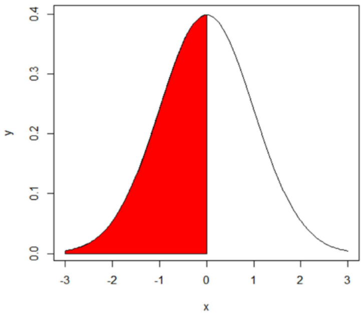
rnorm()

This function is used to generate random numbers whose distribution is normal. It takes the sample size as input and generates that many random numbers. We draw a histogram to show the distribution of the generated numbers.

CODE 1:

```
# Draw another normal curve, use a mean=50 and a standard deviation=10.

>x=seq(20,80,length=200)
>y=dnorm(x,mean=50,sd=10)
>plot(x,y,type="l")
# Find the area under the curve to left of the mean
>x=seq(-3,3,length=200)
>y=dnorm(x,mean=0,sd=1)
>plot(x,y,type="l")
>x=seq(-3,0,length=100)
>y=dnorm(x,mean=0,sd=1)
>polygon(c(-3,x,0),c(0,y,0),col="red")
# Find the area to the left of mean=0 (it should be 0.5)
>pnorm(0,mean=0,sd=1)
```

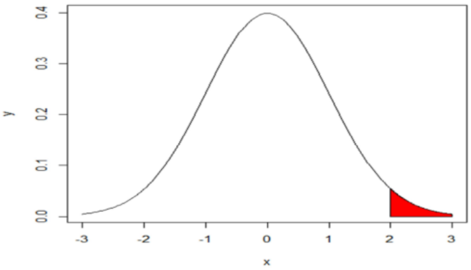
Output:



CODE 2:-

```
# Get the area to the right of 2. First, draw an image, then compute
>x=seq(-3,3,length=200)
>y=dnorm(x,mean=0,sd=1)
>plot(x,y,type="l")
>x=seq(2,3,length=100)
>y=dnorm(x,mean=0,sd=1)
>polygon(c(2,x,3),c(0,y,0),col="red")
>1-pnorm(2,mean=0,sd=1)
```

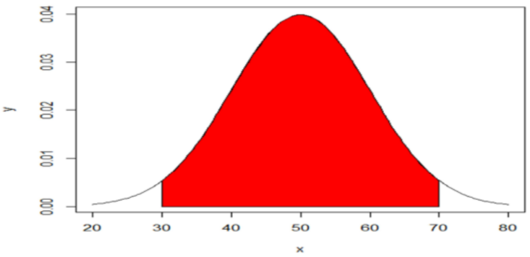
Output:



CODE 3:

```
# Use the pnorm command to find areas under the normal density curve,
# regardless of the mean and standard deviation values.

# Example, mean=50 and standard deviation=10.

> x=seq(20,80, length=200)
>y=dnorm(x, mean=50,sd=10)
>plo t(x, y, type="l")
>x=seq(30,70, length=100)
>y=dnorm (x, mean=50, sd=10)
>polygon (c(30, x,70),c(0,y,0),col="red")
>pnorm(70, mean=50,sd=10)-pnorm(30,mean=50,sd=10)
```

Output:



Code 4:

Find $P(0 < z < 1.24)$

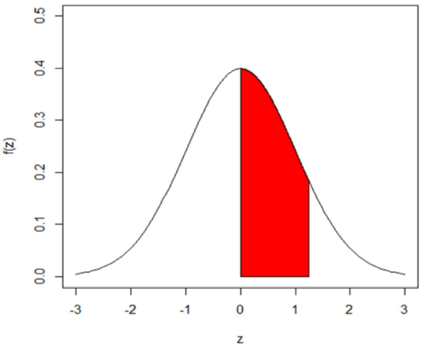
Solution:

```
        > pnorm(1.24) - pnorm(0)
         [1] 0.3925123
```

Normal Probability Shape :

```
> plot.new()
> curve(dnorm, xlim = c(-3, 3), ylim = c(0, 0.5), xlab = "z", ylab="f(z)")
> zleft = 0
> zright = 1.24
> x = c(zleft, seq(zleft, zright, by=.001), zright)
> y = c(0, dnorm(seq(zleft, zright, by=.001)), 0)
> polygon(x, y, col="red")
```

Output:



Worked out example:

1. The speed of a file transfer from a server on campus to a personal computer at a student's home on a weekday evening is normally distributed with a mean of 60 kilobits per second and a standard deviation of four kilobits per second.
   (a). What is the probability that the file will transfer at a speed of 68 kilobits per second or more?
   (b). What is the probability that the file will transfer at a speed of less than 55 kilobits per second?

Solution:

given mean $(\mu) = 60$, standard deviation $(\sigma) = 4$

here we have to find (a) $P(x \geq 68)$ (b) $P(x < 55)$

$$z = \frac{x-\mu}{\sigma} = \frac{x-60}{4}$$

- . $P(x \geq 68) = P(z \geq 2) = 0.5 - A(2) = 0.5 - 0.477 = 0.0228$
- . $P(x < 55) = P(z < -1.25) = 0.5 - A(1.25) = 0.5 - 0.3944 = 0.106$

5.1.1. Experiments:

1. The serum cholesterol level X in 14-year old boys has approximately a normal distribution with mean 170 and standard deviation 30.
   (a). Find the probability that the serum cholesterol level of a randomly chosen 14-Year old boy exceeds 230.
   (b). In a middle school there are 300, 14-year-old boys. Find the probability that at Least 8 boys have serum cholesterol level that exceeds 230.
2. Lifetimes of batteries in a certain application are normally distributed with mean 50 hours and standard deviation 5 hours. Find the probability that a randomly chosen battery lasts between 42 and 52 hours.

6. Regression and Correlation

6.1. Regression

Principle of Least Squares

The vertical deviation of the point (x_i, y_i) from the line $y = b_0 + b_1 x$ is

$$\text{height of point} - \text{height of line} = y_i - (b_0 + b_1 x_i)$$

The sum of squared vertical deviations from the points $(x_1, y_1), \ldots, (x_n, y_n)$ to the line is then

$$f(b_0, b_1) = \sum_{i=1}^{n} [y_i - (b_0 + b_1 x_i)]^2$$

The point estimates of β_0 and β_1, denoted by $\hat{\beta}_0$ and $\hat{\beta}_1$ and called the **least squares estimates,** are those values that minimize $f(b_0, b_1)$. That is, $\hat{\beta}_0$ and $\hat{\beta}_1$ are such that $f(\hat{\beta}_0, \hat{\beta}_1) \le f(b_0, b_1)$ for any b_0 and b_1. The **estimated regression line** or **least squares line** is then the line whose equation is $y = \hat{\beta}_0 + \hat{\beta}_1 x$.

The minimizing values of b_0 and b_1 are found by taking partial derivatives of $f(b_0, b_1)$ with respect to both b_0 and b_1, equating them both to zero [analogously to $f'(b) = 0$ in univariate calculus], and solving the equations

$$\frac{\partial f(b_0, b_1)}{\partial b_0} = \sum 2(y_i - b_0 - b_1 x_i)\,(-1) = 0$$
$$\frac{\partial f(b_0, b_1)}{\partial b_1} = \sum 2(y_i - b_0 - b_1 x_i)\,(-x_i) = 0$$

Cancellation of the -2 factor and rearrangement gives the following system of equations, called the **normal equations:**

$$nb_0 + (\sum x_i) b_1 = \sum y_i$$
$$(\sum x_i) b_0 + (\sum x_i^2) b_1 = \sum x_i y_i$$

These equations are linear in the two unknowns b_0 and b_1. Provided that not all x_i's are identical, the least squares estimates are the unique solution to this system.

The least squares estimate of the slope coefficient β_1 of the true regression line is

$$b_1 = \hat{\beta}_1 = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_i - \overline{x})^2} = \frac{S_{xy}}{S_{xx}}$$

Computing formulas for the numerator and denominator of $\hat{\beta}_1$ are

$$S_{xy} = \sum x_i y_i - (\sum x_i)(\sum y_i)/n \qquad S_{xx} = \sum x_i^2 - (\sum x_i)^2/n$$

The least squares estimate of the intercept β_0 of the true regression line is

$$b_0 = \hat{\beta}_0 = \frac{\sum y_i - \hat{\beta}_1 \sum x_i}{n} = \overline{y} - \hat{\beta}_1 \overline{x}$$

The cetane number is a critical property in specifying the ignition quality of a fuel used in a diesel engine. Determination of this number for a biodiesel fuel is expensive and time-consuming. The article "Relating the Cetane Number of Biodiesel Fuels to Their Fatty Acid Composition: A Critical Study" (*J. of Automobile Engr.*, 2009: 565–583) included the following data on x = iodine value (g) and y = cetane number for a sample of 14 biofuels. The iodine value is the amount of iodine necessary to saturate a sample of 100 g of oil. The article's authors fit the simple linear regression model to this data, so let's follow their lead.

x	132.0	129.0	120.0	113.2	105.0	92.0	84.0	83.2	88.4	59.0	80.0	81.5	71.0	69.2
y	46.0	48.0	51.0	52.1	54.0	52.0	59.0	58.7	61.6	64.0	61.4	54.6	58.8	58.0

The necessary summary quantities for hand calculation can be obtained by placing the x values in a column and the y values in another column and then creating columns for x^2, xy, and y^2 (these latter values are not needed at the moment but will be used shortly). Calculating the column sums gives $\sum x_i = 1307.5$, $\sum y_i = 779.2$, $\sum x_i^2 = 128{,}913.93$, $\sum x_i y_i = 71{,}347.30$, $\sum y_i^2 = 43{,}745.22$, from which

$$S_{xx} = 128{,}913.93 - (1307.5)^2/14 = 6802.7693$$

$$S_{xy} = 71{,}347.30 - (1307.5)(779.2)/14 = -1424.41429$$

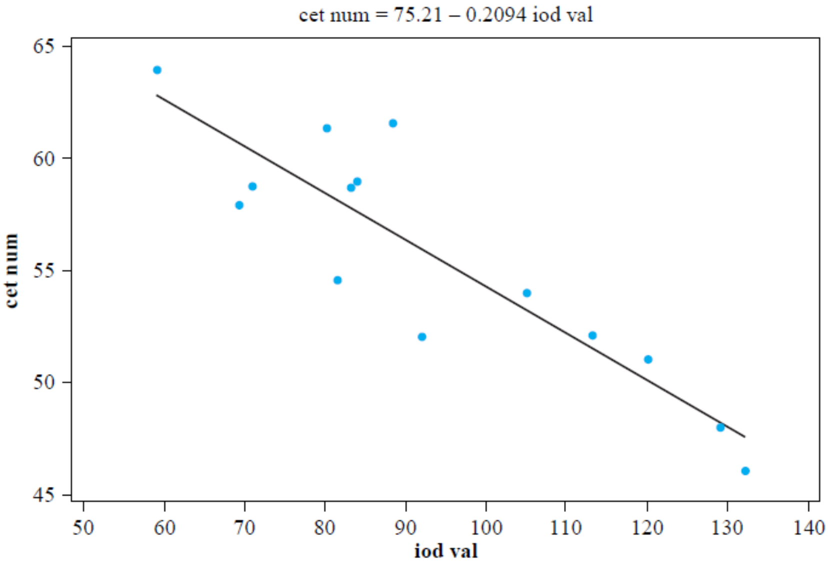
The estimated slope of the true regression line (i.e., the slope of the least squares line) is

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}} = \frac{-1424.41429}{6802.7693} = -.20938742$$

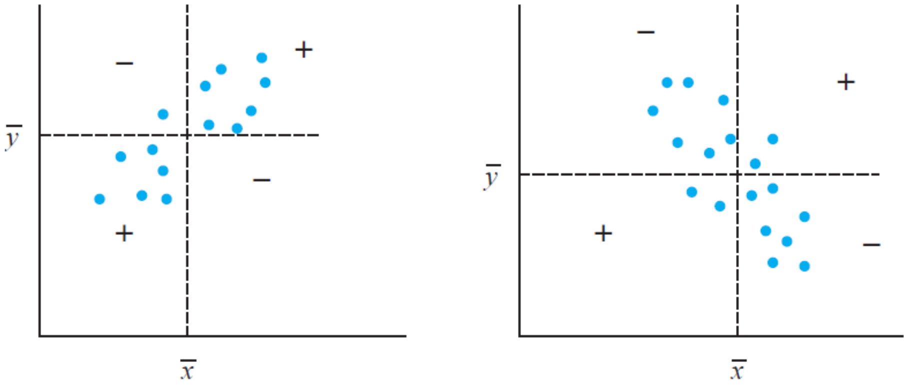
We estimate that the expected change in true average cetane number associated with a 1 g increase in iodine value is −.209—i.e., a decrease of .209. Since $\overline{x} = 93.392857$ and $\overline{y} = 55.657143$, the estimated intercept of the true regression line (i.e., the intercept of the least squares line) is

$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x} = 55.657143 - (-.20938742)(93.392857) = 75.212432$$

The equation of the estimated regression line (least squares line) is $y = 75.212 - .2094x$, exactly that reported in the cited article. Figure displays a scatter plot of the data with the least squares line superimposed. This line provides a very good summary of the relationship between the two variables.

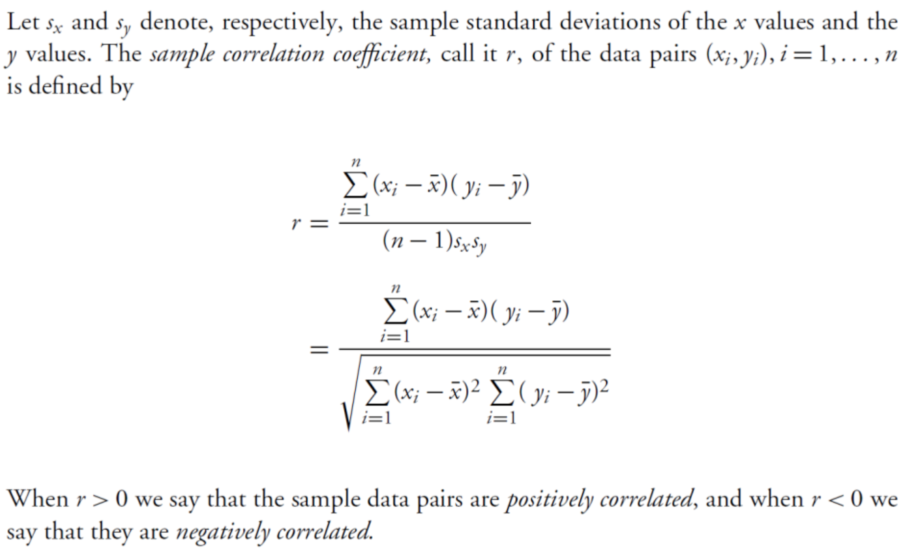


6.2 Correlation



(a)

(b)

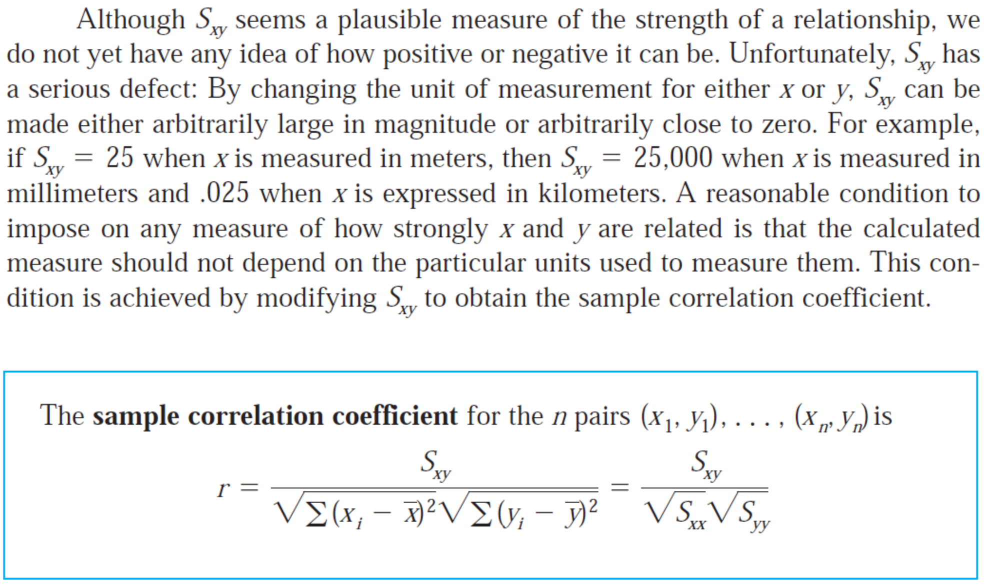


Let s_x and s_y denote, respectively, the sample standard deviations of the x values and the y values. The *sample correlation coefficient,* call it r, of the data pairs $(x_i, y_i), i = 1, \ldots, n$ is defined by

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

$$= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

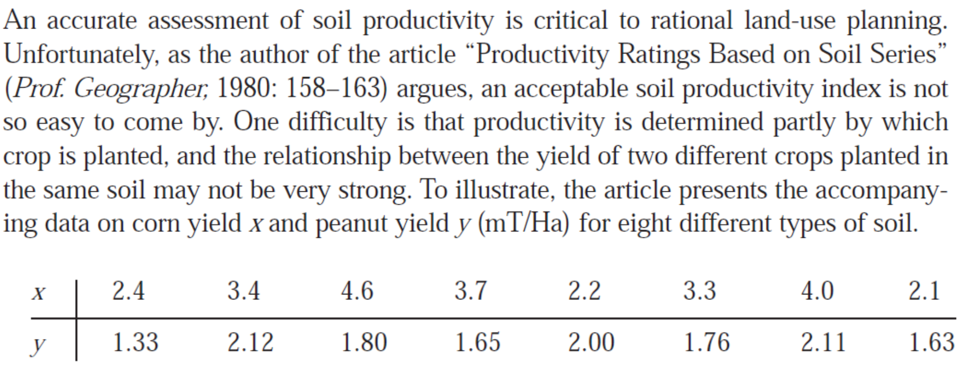
When $r > 0$ we say that the sample data pairs are *positively correlated*, and when $r < 0$ we say that they are *negatively correlated.*



Although S_{xy} seems a plausible measure of the strength of a relationship, we do not yet have any idea of how positive or negative it can be. Unfortunately, S_{xy} has a serious defect: By changing the unit of measurement for either x or y, S_{xy} can be made either arbitrarily large in magnitude or arbitrarily close to zero. For example, if $S_{xy} = 25$ when x is measured in meters, then $S_{xy} = 25{,}000$ when x is measured in millimeters and .025 when x is expressed in kilometers. A reasonable condition to impose on any measure of how strongly x and y are related is that the calculated measure should not depend on the particular units used to measure them. This condition is achieved by modifying S_{xy} to obtain the sample correlation coefficient.

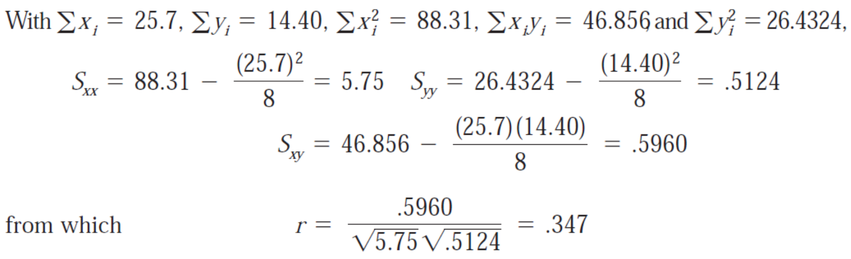
The **sample correlation coefficient** for the n pairs $(x_1, y_1), \ldots, (x_n, y_n)$ is

$$r = \frac{S_{xy}}{\sqrt{\Sigma(x_i - \bar{x})^2}\sqrt{\Sigma(y_i - \bar{y})^2}} = \frac{S_{xy}}{\sqrt{S_{xx}}\sqrt{S_{yy}}}$$



An accurate assessment of soil productivity is critical to rational land-use planning. Unfortunately, as the author of the article “Productivity Ratings Based on Soil Series” (*Prof. Geographer;* 1980: 158–163) argues, an acceptable soil productivity index is not so easy to come by. One difficulty is that productivity is determined partly by which crop is planted, and the relationship between the yield of two different crops planted in the same soil may not be very strong. To illustrate, the article presents the accompanying data on corn yield x and peanut yield y (mT/Ha) for eight different types of soil.

x	2.4	3.4	4.6	3.7	2.2	3.3	4.0	2.1
y	1.33	2.12	1.80	1.65	2.00	1.76	2.11	1.63



With $\sum x_i = 25.7$, $\sum y_i = 14.40$, $\sum x_i^2 = 88.31$, $\sum x_i y_i = 46.856$ and $\sum y_i^2 = 26.4324$,

$$S_{xx} = 88.31 - \frac{(25.7)^2}{8} = 5.75 \quad S_{yy} = 26.4324 - \frac{(14.40)^2}{8} = .5124$$

$$S_{xy} = 46.856 - \frac{(25.7)(14.40)}{8} = .5960$$

from which $$r = \frac{.5960}{\sqrt{5.75}\sqrt{.5124}} = .347$$

Add Straight Lines to a Plot

abline

Description:

This function adds one or more straight lines through the current plot.

Usage:

```
abline(a = NULL, b = NULL, h = NULL, v = NULL, reg = NULL,
    coef = NULL, untf = FALSE, ...)
```

Arguments:

a, b: the intercept and slope, single values.

untf: logical asking whether to _untransform. See ‘Details’.

h: the y-value(s) for horizontal line(s).

v: the x-value(s) for vertical line(s).

coef: a vector of length two giving the intercept and slope.

reg: an object with a ‘coef’ method. See ‘Details’.

...: graphical parameters such as ‘col’, ‘lty’ and ‘lwd’ (possibly as vectors: see ‘Details’) and ‘xpd’ and the line characteristics ‘lend’, ‘ljoin’ and ‘lmitre’.

Details:

Typical usages are

```
abline(a, b, untf = FALSE, ...)
abline(h =, untf = FALSE, ...)
abline(v =, untf = FALSE, ...)
abline(coef =, untf = FALSE, ...)
abline(reg =, untf = FALSE, ...)
```

The first form specifies the line in intercept/slope form (alternatively ‘a’ can be specified on its own and is taken to contain the slope and intercept in vector form).

The ‘h=’ and ‘v=’ forms draw horizontal and vertical lines at the specified coordinates.

The ‘coef’ form specifies the line by a vector containing the slope and intercept.

‘reg’ is a regression object with a ‘coef’ method. If this returns a vector of length 1 then the value is taken to be the slope of a line through the origin, otherwise, the first 2 values are taken to be the intercept and slope.

If ‘untf’ is true, and one or both axes are log-transformed, then a curve is drawn corresponding to a line in original coordinates, otherwise a line is drawn in the transformed coordinate system. The ‘h’ and ‘v’ parameters always refer to original coordinates.

The graphical parameters ‘col’, ‘lty’ and ‘lwd’ can be specified; see ‘par’ for details. For the ‘h=’ and ‘v=’ usages they can be vectors of length greater than one, recycled as necessary.

Specifying an ‘xpd’ argument for clipping overrides the global ‘par("xpd")’ setting used otherwise.

References:

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) _The New S Language_. Wadsworth & Brooks/Cole.

Murrell, P. (2005) _R Graphics_. Chapman & Hall/CRC Press.

See Also:

‘lines’ and ‘segments’ for connected and arbitrary lines given by their _endpoints_. ‘par’.

Examples:

```
## Setup up coordinate system (with x == y aspect ratio):
plot(c(-2,3), c(-1,5), type = "n", xlab = "x", ylab = "y", asp = 1)
## the x- and y-axis, and an integer grid
abline(h = 0, v = 0, col = "gray60")
text(1,0, "abline( h = 0 )", col = "gray60", adj = c(0, -.1))
abline(h = -1:5, v = -2:3, col = "lightgray", lty = 3)
abline(a = 1, b = 2, col = 2)
text(1,3, "abline( 1, 2 )", col = 2, adj = c(-.1, -.1))
```

```
## Simple Regression Lines:
require(stats)
sale5 <- c(6, 4, 9, 7, 6, 12, 8, 10, 9, 13)
plot(sale5)
abline(lsfit(1:10, sale5))
abline(lsfit(1:10, sale5, intercept = FALSE), col = 4) # less fitting

z <- lm(dist ~ speed, data = cars)
plot(cars)
abline(z) # equivalent to abline(reg = z) or
abline(coef = coef(z))

## trivial intercept model
abline(mC <- lm(dist ~ 1, data = cars)) ## the same as
abline(a = coef(mC), b = 0, col = "blue")
```

Fitting Linear Models

Description:

‘lm’ is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance (although ‘aov’ may provide a more convenient interface for these).

Usage:

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

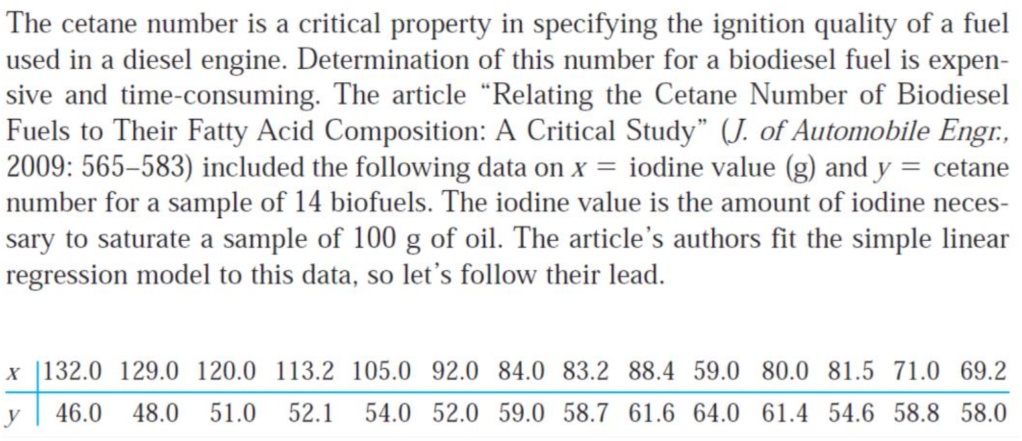
6.3. Experiment

Find

1.Scatter plot,

2. Finding True regression Line

3. Plot regression line and Predict the value of y for given x=90.
   for the following problem



The cetane number is a critical property in specifying the ignition quality of a fuel used in a diesel engine. Determination of this number for a biodiesel fuel is expensive and time-consuming. The article "Relating the Cetane Number of Biodiesel Fuels to Their Fatty Acid Composition: A Critical Study" (*J. of Automobile Engr.*, 2009: 565–583) included the following data on x = iodine value (g) and y = cetane number for a sample of 14 biofuels. The iodine value is the amount of iodine necessary to saturate a sample of 100 g of oil. The article's authors fit the simple linear regression model to this data, so let's follow their lead.

x	132.0	129.0	120.0	113.2	105.0	92.0	84.0	83.2	88.4	59.0	80.0	81.5	71.0	69.2
y	46.0	48.0	51.0	52.1	54.0	52.0	59.0	58.7	61.6	64.0	61.4	54.6	58.8	58.0

7. Test of Hypothesis

7.1. Large Sample – z – Test

prop.test

Test of Equal or Given Proportions

Description:

‘prop.test’ can be used for testing the null that the proportions (probabilities of success) in several groups are the same, or that they equal certain given values.

Usage:

```
prop.test(x, n, p = NULL,
          alternative = c("two.sided", "less", "greater"),
          conf.level = 0.95, correct = TRUE)
```

Arguments:

x: a vector of counts of successes, a one-dimensional table with two entries, or a two-dimensional table (or matrix) with 2 columns, giving the counts of successes and failures, respectively.

n: a vector of counts of trials; ignored if ‘x’ is a matrix or a table.

p: a vector of probabilities of success. The length of ‘p’ must be the same as the number of groups specified by ‘x’, and its elements must be greater than 0 and less than 1. t.test package:stats R Documentation

- *To test the hypothesisfor Large Samples by using Two–sample Z-test*

Tests about a proportion using x and n using the prop.test function:

Usage: prop.test(c(x1,x2), c(n1,n2), correct=, alternate =).

1. x1 and x2 are the number of successes in sample 1 and 2 respectively.
2. n1 and n2 are the sample sizes or number of trials.
3. correct = TRUE (use a continuity correction factor) or FALSE (do not).
4. alternate = "two.sided" (default), "less", or "greater".

7.1.1. Experiment:

1. A popular cold-remedy was tested for it's efficacy. In a sample of 150 people who took the remedy upon getting a cold, 117 (78%) had no symptoms one week later. In a sample of 125 people who took the placebo upon getting a cold, 90 (75%) had no symptoms one week later. The table summarizes this information.

Group	*#who are symptom Free after one*	*Total # in group (n)*	*Proportion* $\hat{p} = x/n$
Remedy	*117*	*150*	*0.78*
Placebo	*90*	*120*	*0.75*

The Test: Test the claim that the proportion of all remedy users who are symptom-Free a fter one week is greater than the proportion for placebo users. Test this claim at the 0.05 significance level.

2. The Trial Urban District Assessment (TUDA) is a study sponsored by the government of student achievement in large urban school district. In 2009, 1311 of a random sample of 1900 eighth-graders from Houston performed at or above the basic level in mathematics. In 2011, 1440 of a random sample of 2000 eighth-graders from Houstonperformed at or above the basic level. (The study reports the proportions).

   (A) Is there an increase in the proportion of eighth-graders who performed at or above the basic level in mathematics from 2009 to 2011 at the 5% significance level?

   (B) Compute the 95% confidence interval for the difference in proportion of eighth- graders who performed at or above the basic level in mathematics from 2009 to 2011.

3. The use of helmet among recreational alpine skiers and snowboarders are generally low. A study from Norway wanted to examine if helmet use reduces the risk of head injury. In the study, they compared the helmet use among skiers and snowboarders that was injured with a control group. The control group consisted of skiers and snowboarders that was uninjured. 96 of 578 people with head injuries used a helmet and 656 of 2992 people in the uninjured group used a helmet. Is helmet use lower among skiers and snowboarders who had head injuries?

7.2. Small Samples – t - Test

Student's t-Test

Description:

Performs one and two sample t-tests on vectors of data.

Usage:

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Arguments:

x: a (non-empty) numeric vector of data values.

y: an optional (non-empty) numeric vector of data values.

alternative: a character string specifying the alternative hypothesis, must be one of ‘"two.sided"’ (default), ‘"greater"’ or ‘"less"’. You can specify just the initial letter.

mu: a number indicating the true value of the mean (or difference in means if you are performing a two sample test).

t-test for single mean and t-testfor difference of means

The t.test( ) function produces a variety of t-tests. Unlike most statistical packages, the default assumes unequal variance.

```
# independent 2-group t-test

> t.test(y~x)                # where y is numeric and x is a binary factor

# independent 2-group t-
test
                                # where y1 and y2 are numeric
> t.test(y1,y2)

# paired t-test
```

```
> t.test(y1,y2,paired=TRUE)          # where y1 & y2 are numeric

# one sample t-test

> t.test(y,mu=3)                     # Ho: mu=3
```

We can use the var.equal = TRUE option to specify equal and a pooled variance estimate, use the alternative="less" or alternative="greater" option to specify a one tailed test.

```
> t.test(len ~ supp, data = ToothGrowth, alt = "greater", var.equal =
TRUE) > x <- rnorm(13, mean = 2, sd = 3)

> t.test(x, mu = 0, conf.level = 0.9, alternative = "greater")
```

One Sample t-test:-

Comparing the sample mean with a known value, when population variance is not known.

7.2.1. Experiments:

1. An outbreak of salmonella-related illness was attributed to ice produced at a certain factory. Scientists measured the level of Salmonella in 9 randomly sampled batches ice crean.The levels(in MPN/g) were:

0.593	0.142	0.329	0.691	0.231	0.793	0.519	0.392	0.418

   Is there evidence that the mean level pf Salmonella in ice cream greater than 0.3 MPN/g?

2. Suppose that 10 volunteers have taken an intelligence test; here are the results obtained. The average score of the entire population is 75 in the same test. Is there any significant difference (with a significance level of 95%) between the sample and population means, assuming that the variance of the population is not known.

   Scores: 65, 78, 88, 55, 48, 95, 66, 57, 79, 81

3. Comparing two independent sample means, taken from two populations with unknown variance.The following data shows the heights of individuals of two different countries with unknown population variances. Is there any significant difference b/n the average heights of two groups.

A:	175	168	168	190	156	181	182	175	174	179
B:	185	169	173	173	188	186	175	174	179	180