Statistical Learning

ISLR- Done by Prateek Minhas

2025-10-18

Ch-2: Statistical Learning

Lab

Basic commands

```
x=c(1,3,4,5)
x
```

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

```
y=c(4,5,80,-3)
y
```

```
## [1]  4  5 80 -3
```

```
length(x)
```

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

```
length(x)+length(y)
```

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

```
x-y
```

```
## [1]  -3  -2 -76   8
```

```
z=x+y
```

```
ls()
```

ls() and Rm()

```
## [1] "x" "y" "z"
```

```
rm(z)
ls()
```

```
## [1] "x" "y"
```

```
rm(list=ls()) #empty out the list
ls()
```

```
## character(0)
```

```
x = matrix(data=c(1,2,3,4,5,6), nrow=2 , ncol = 3, byrow=TRUE)
x
```

matrix() fucntion

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

```
y = matrix(data=c(1,2,3,4,5,6), nrow=2 , ncol = 3, byrow=FALSE)
y
```

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

removing the data=, nrow=, ncol= we can also write directly , by default we get byrows = false meaning the columns get filled first

```
z= matrix(c(1,2,3,4,5,6), 3,2)
z
```

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

```
sqrt(x)
```

```
##      [,1]     [,2]     [,3]
## [1,]    1 1.414214 1.732051
## [2,]    2 2.236068 2.449490
```

```
y^2
```

```
##      [,1] [,2] [,3]
## [1,]    1    9   25
## [2,]    4   16   36
```

rnorm() gives random values every time we use it

```
x=rnorm(50)

y=x+rnorm(50,mean=50, sd=.1)
cor(x,y)
```

```
## [1] 0.9951687
```

we use set seed() so that we get same random numbers every time for a particular seed value

```
set.seed(1303)
rnorm(50)
```

```
##  [1] -1.1439763145  1.3421293656  2.1853904757  0.5363925179  0.0631929665
##  [6]  0.5022344825 -0.0004167247  0.5658198405 -0.5725226890 -1.1102250073
## [11] -0.0486871234 -0.6956562176  0.8289174803  0.2066528551 -0.2356745091
## [16] -0.5563104914 -0.3647543571  0.8623550343 -0.6307715354  0.3136021252
## [21] -0.9314953177  0.8238676185  0.5233707021  0.7069214120  0.4202043256
## [26] -0.2690521547 -1.5103172999 -0.6902124766 -0.1434719524 -1.0135274099
## [31]  1.5732737361  0.0127465055  0.8726470499  0.4220661905 -0.0188157917
## [36]  2.6157489689 -0.6931401748 -0.2663217810 -0.7206364412  1.3677342065
```

```
## [41]  0.2640073322  0.6321868074 -1.3306509858  0.0268888182  1.0406363208
## [46]  1.3120237985 -0.0300020767 -0.2500257125  0.0234144857  1.6598706557
```

```
set.seed(3)
y=rnorm(100)
mean(y)
```

```
## [1] 0.01103557
```

```
var(y)
```

```
## [1] 0.7328675
```

```
sqrt(var(y))
```

```
## [1] 0.8560768
```

```
sd(y)
```

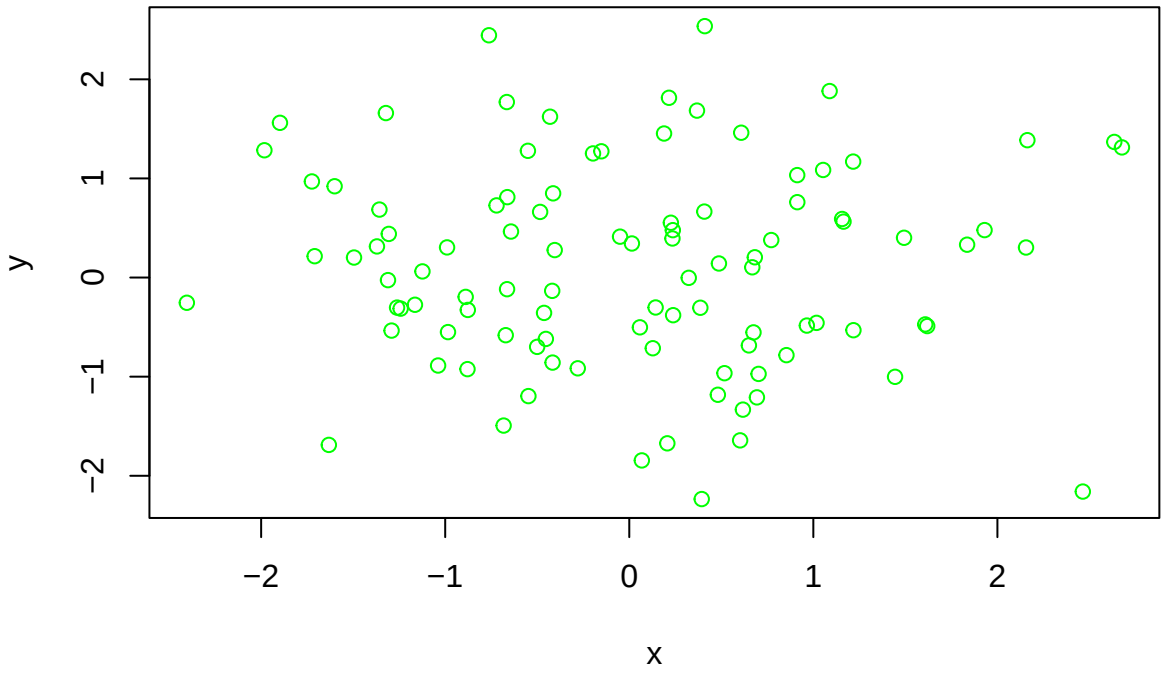
```
## [1] 0.8560768
```

```
#as we see variance = standard deviation.sq
```

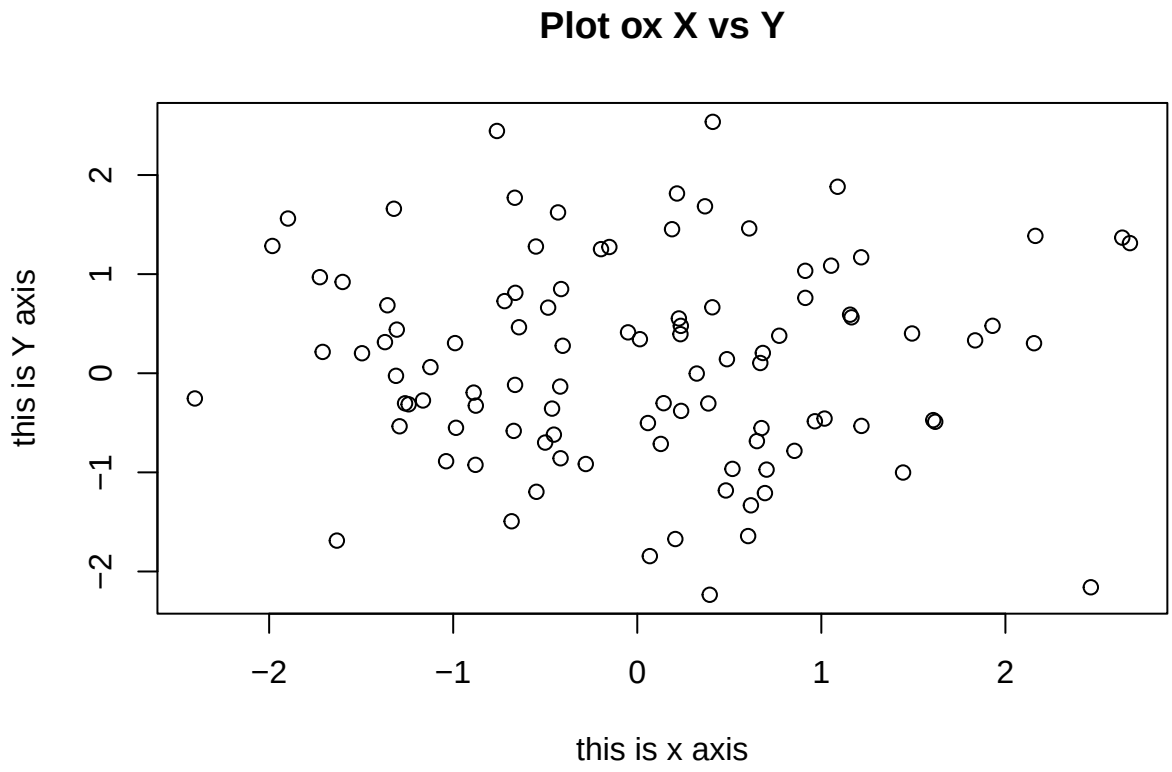
Graphics

plot()

```
x=rnorm(100)
y=rnorm(100)
plot(x,y, col="green")
```



```r
plot(x,y,xlab = "this is x axis", ylab = "this is Y axis", main="Plot ox X vs Y")
```



Seq()

```r
x=seq(1,10)
x
```

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

```r
y=1:10
y
```

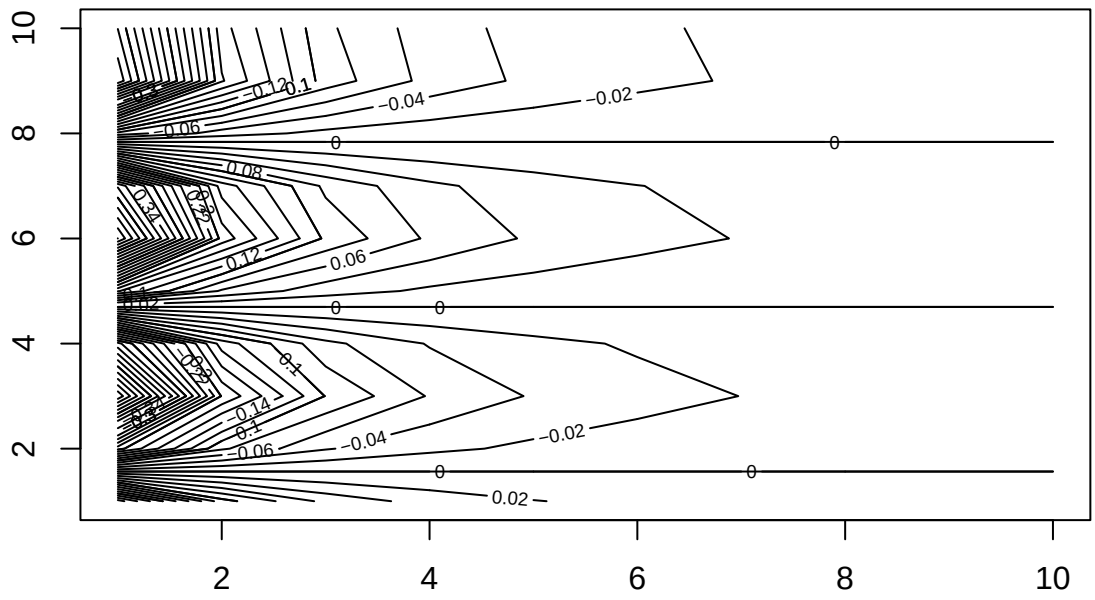
```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
z=seq(-pi,pi,length=50)
z
```

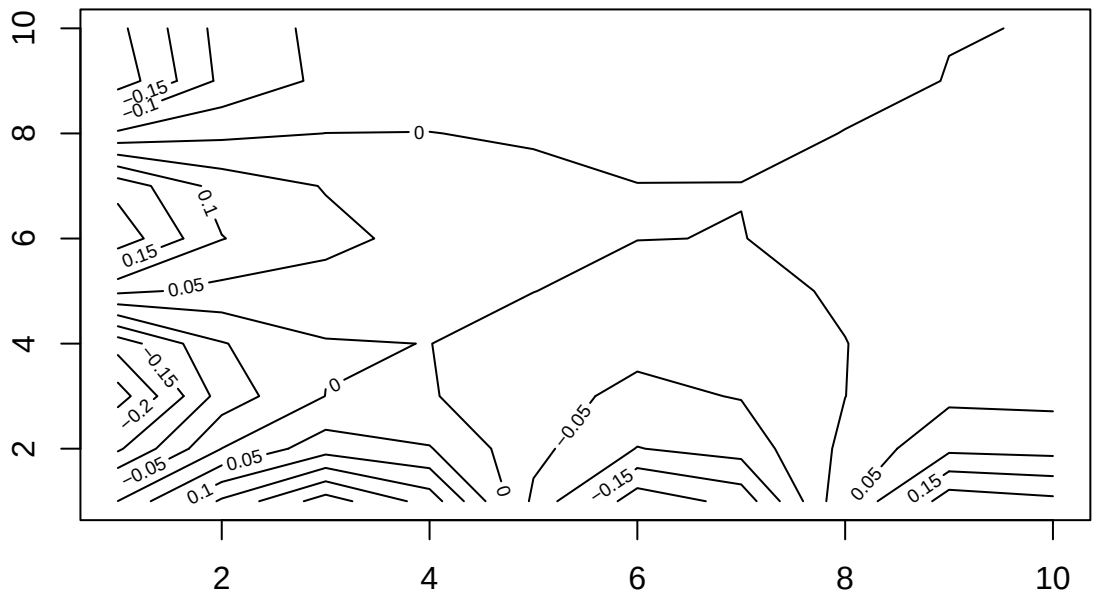
```
##  [1] -3.14159265 -3.01336438 -2.88513611 -2.75690784 -2.62867957 -2.50045130
##  [7] -2.37222302 -2.24399475 -2.11576648 -1.98753821 -1.85930994 -1.73108167
## [13] -1.60285339 -1.47462512 -1.34639685 -1.21816858 -1.08994031 -0.96171204
## [19] -0.83348377 -0.70525549 -0.57702722 -0.44879895 -0.32057068 -0.19234241
## [25] -0.06411414  0.06411414  0.19234241  0.32057068  0.44879895  0.57702722
## [31]  0.70525549  0.83348377  0.96171204  1.08994031  1.21816858  1.34639685
## [37]  1.47462512  1.60285339  1.73108167  1.85930994  1.98753821  2.11576648
## [43]  2.24399475  2.37222302  2.50045130  2.62867957  2.75690784  2.88513611
## [49]  3.01336438  3.14159265
```

contour() for 3d graphs

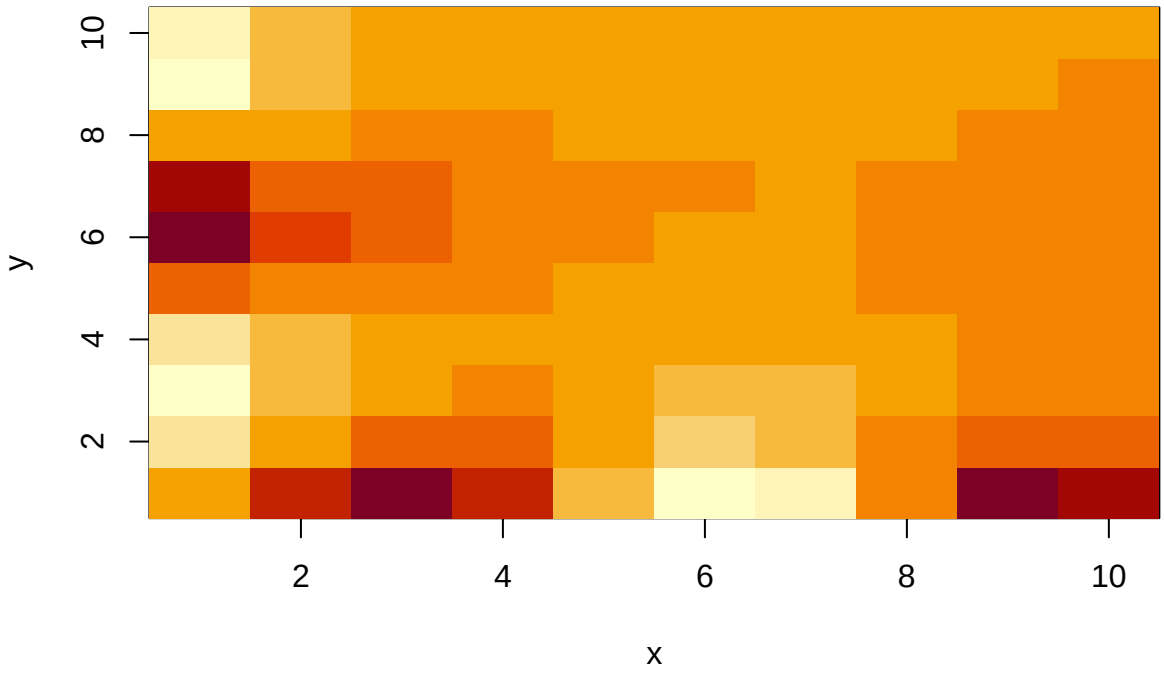
```r
y=x
f= outer(x,y, function(x,y) cos(y)/(1+x^2))
contour(x,y,f)
contour(x,y,f , nlevels = 45 , add = T)
```



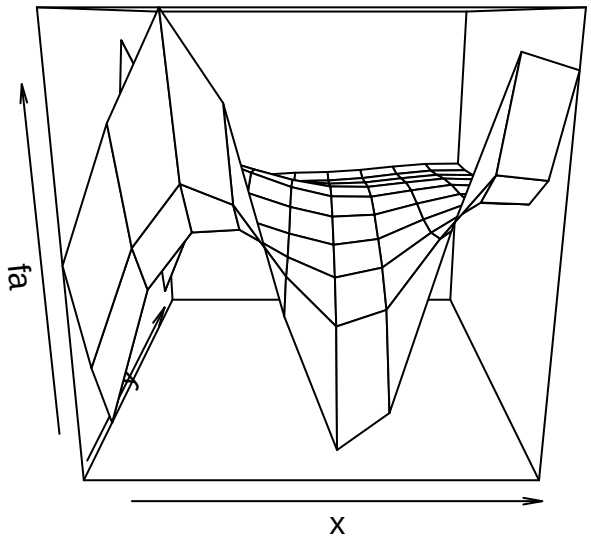

```r
fa=(f-t(f))/2
contour(x,y,fa,nlevels = 15)
```


10
8
6
4
2
2
4
6
8
10
−0.15
−0.1
0
0.1
0.15
0.05
−0.15
−0.2
−0.05
0
0.05
0.1
0
−0.05
−0.15
0.05
0.15

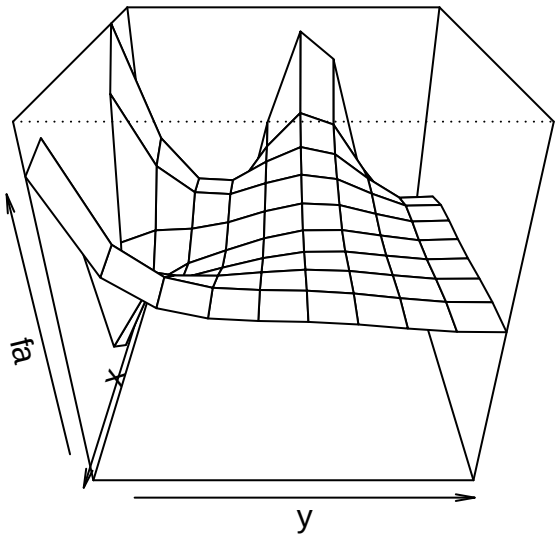
```
image(x,y,fa)
```



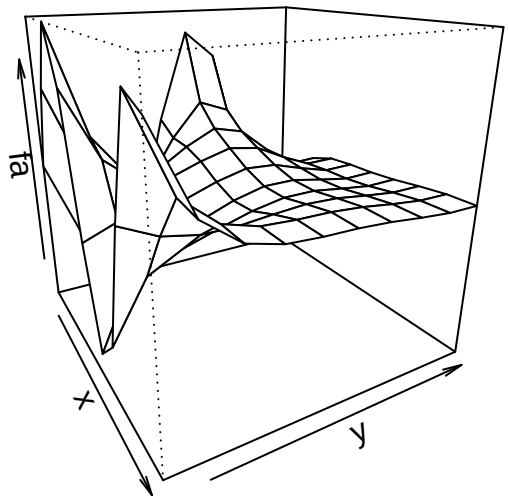

```
persp(x,y,fa)
```



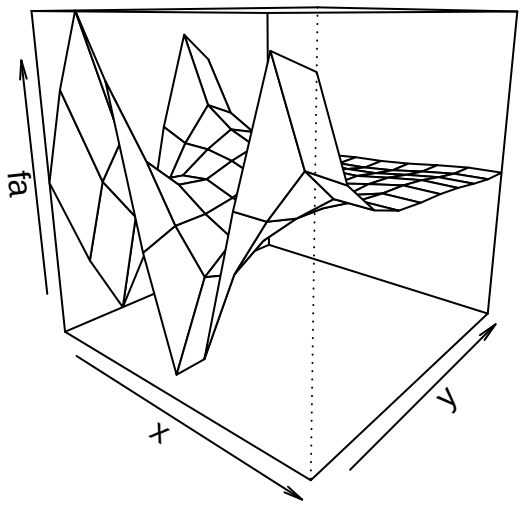

```
persp(x,y,fa, theta = 90, phi=30)
```




```
persp(x,y,fa, theta = 60, phi=20)
```




```
persp(x,y,fa, theta = 40, phi=15)
```



Indexing data

```
A= matrix(1:16, 4,4)
A
```

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

```
A[2,3]
```

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

```
A[c(1,3),c(2,4)]
```

```
##      [,1] [,2]
## [1,]    5   13
## [2,]    7   15
```

```
A[1:3,2:4]
```

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

```
A[1:2,]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
```

```
A[,1:2]
```

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

```
A[1,]
```

```
## [1]  1  5  9 13
```

-ve sign indicates that select all the elements Except those mentioned , it is like a negation

```
A
```

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

```
A[c(1,3),]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    3    7   11   15
```

```
A[-c(1,3),]#second and last row
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    6   10   14
## [2,]    4    8   12   16
```

```
A[-c(1,3),-c(1,3,4)]
```

```
## [1] 6 8
```

```
dim(A)
```

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

Loading Data