Knowledge Mining EPPS 6323 Dr. Ho Assignment 1

Glen Cooper

2/14/2022

```
# R Programming (base)
# Adapted from ISLR Chapter 2 Lab: Introduction to R
# Objectives: use basic commands, create object, use function, simple
statistics plotting object
# Basic Commands

# Create object using the assignment operator (<-, =)
x <- c(1,3,2,5)
x # show object
```

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

```
x = c(1,6,2)
x
```

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

```
y = c(1,4,3)

# Using function

length(x)  # What does length() do?
```

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

```
length(y)
```

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

```
# Using +, -, *, /,^ operators
x+y
```

```
## [1]  2 10  5
```

```
ls() # List objects in the environment
```

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

```
rm(x,y) # Remove objects
ls()
```

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

```r
rm(list=ls()) # Danger! What does this do?

# Matrix operations

?matrix

## starting httpd help server ... done

x=matrix(data=c(1,2,3,4), nrow=2, ncol=2) # Create a 2x2 matrix object
x

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

x=matrix(c(1,2,3,4),2,2)
matrix(c(1,2,3,4),2,2,byrow=TRUE) # What about byrow=F?

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

sqrt(x) # What does x look like?

##          [,1]     [,2]
## [1,] 1.000000 1.732051
## [2,] 1.414214 2.000000

x^2

##      [,1] [,2]
## [1,]    1    9
## [2,]    4   16

x=rnorm(50) # Generate a vector of 50 numbers using the rnorm() function

y=x+rnorm(50,mean=50,sd=.1) # What does rnorm(50,mean=50,sd=.1) generate?

cor(x,y) # Correlation of x and y

## [1] 0.99446

set.seed(1303) # Set the seed for Random Number Generator (RNG) to generate
values that are reproducible.
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) # Try different seeds?
y=rnorm(100)

# Simple descriptive statistics

mean(y)
```

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

```
var(y)
```

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

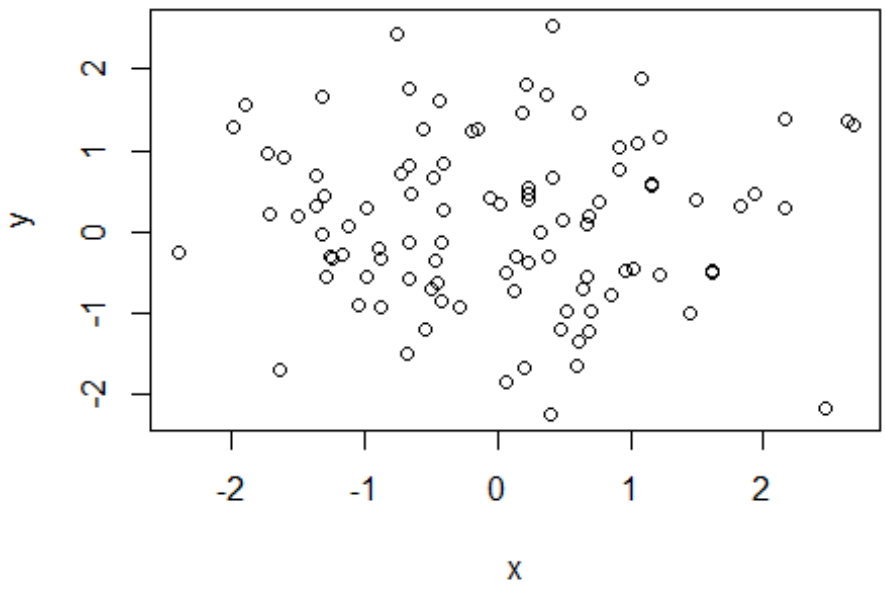
```
sqrt(var(y))
```

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

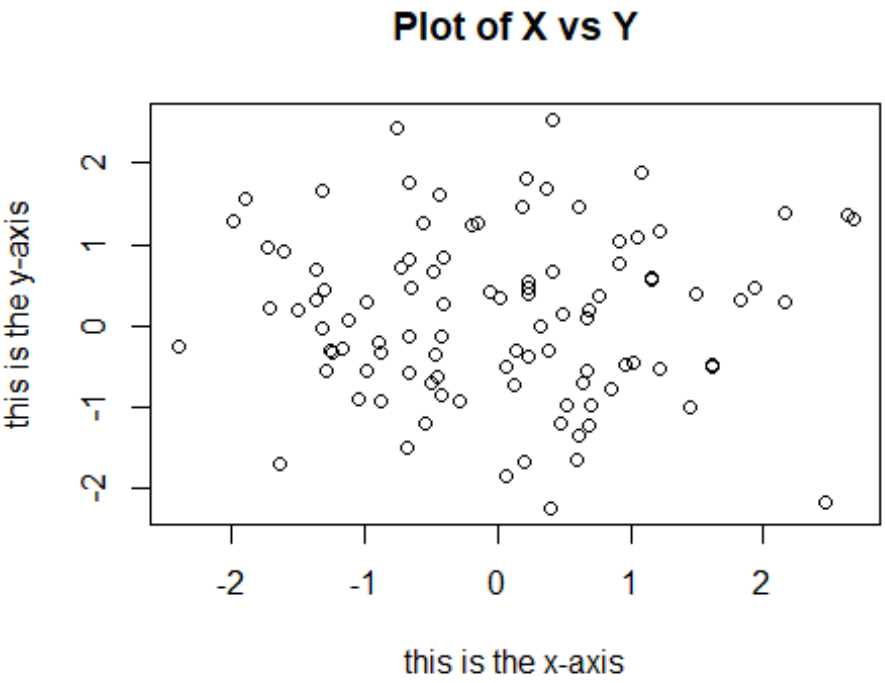
```
sd(y)
```

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

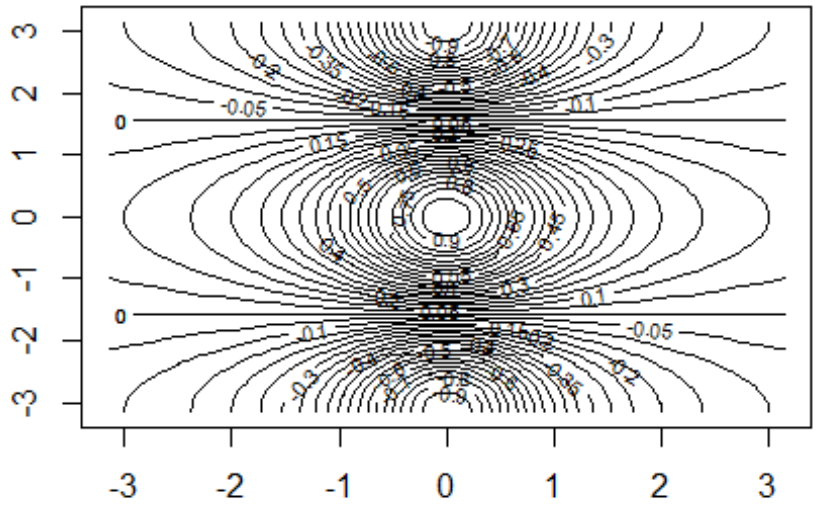
```
# Graphics using R Graphics (without packages)

x=rnorm(100)
y=rnorm(100)
plot(x,y) # Scatterplot for two numeric variables by default
```



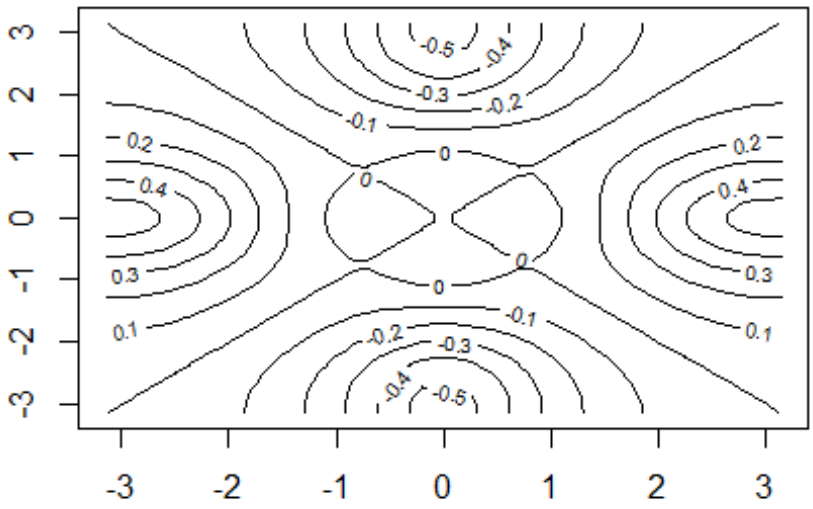

```r
plot(x,y,xlab="this is the x-axis",ylab="this is the y-axis",main="Plot of X vs Y") # Add labels
```



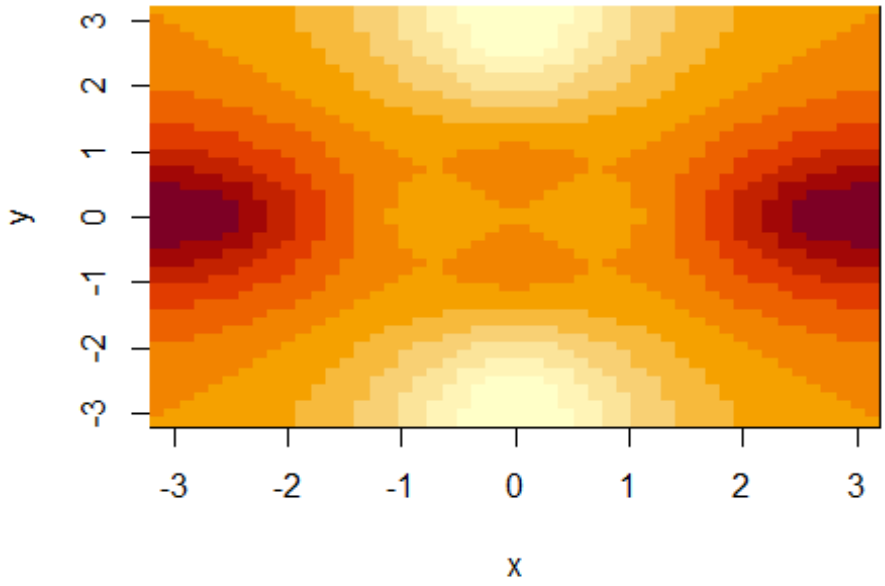
```r
pdf("Figure.pdf") # Save as pdf, add a path or it will be stored on the
project directory
plot(x,y,col="green") # Try different colors?
dev.off() # Close the file using the dev.off function

## png
##   2

x=seq(1,10) # Same as x=c(1:10)
x

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

x=1:10
x

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

x=seq(-pi,pi,length=50)
y=x

## Fancy graphs: contour, image, persp (3D)
f=outer(x,y,function(x,y)cos(y)/(1+x^2))
contour(x,y,f)
contour(x,y,f,nlevels=45,add=T)
```



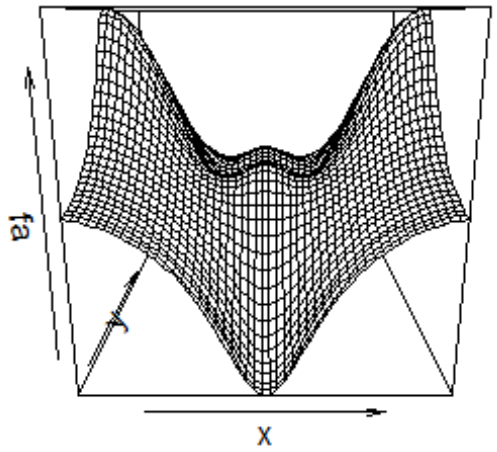

```r
fa=(f-t(f))/2 # matrix transpose
contour(x,y,fa,nlevels=15)
```



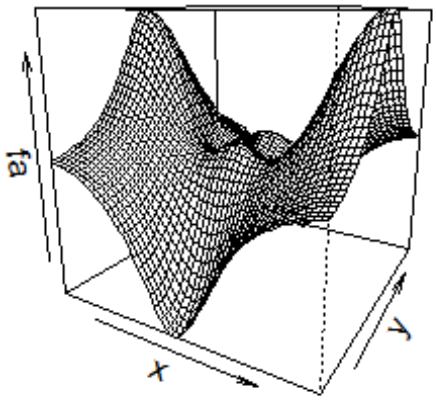

```
image(x,y,fa)
```



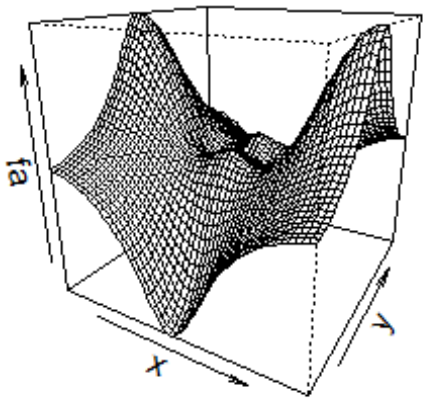

```
persp(x,y,fa)
```



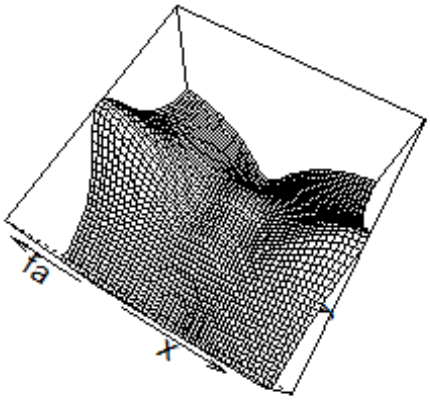

```
persp(x,y,fa,theta=30)
```



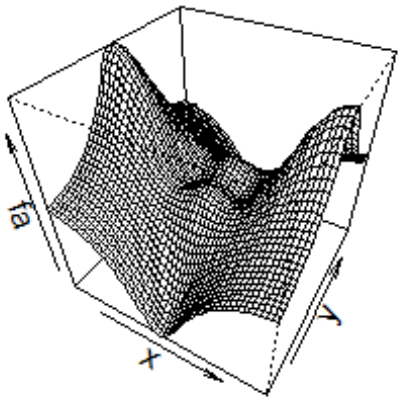

```
persp(x,y,fa,theta=30,phi=20)
```




```
persp(x,y,fa,theta=30,phi=70)
```




```
persp(x,y,fa,theta=30,phi=40) # Gradient descent?
```


fa
x
y