MATH560 R Lab 1

Jared Andreatta

2025-03-12

2.3.1 Basic commands

Vectors

```r
# Defining vector
x <- c(1, 7, 5, 8)
y <- c(8, 9, 3, 2)
x
```

```
## [1] 1 7 5 8
```

```r
y
```

```
## [1] 8 9 3 2
```

```r
# Length of vector
length(x)
```

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

```r
length(y)
```

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

```r
# Vector addition
x+y
```

```
## [1]  9 16  8 10
```

```r
# ls and rm
ls() # List of all objects in environment
```

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

```
rm(x,y) # Delete vectors
rm(list = ls()) # Delete entire list
```

Matrices

```
# Help
?matrix
```

```
## starting httpd help server ... done
```

```
# Defining matrix
X <- matrix(c(1,2,3,4),2,2)
X
```

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

```
X_byrow <- matrix(c(1,2,3,4),2,2,byrow = TRUE) # Entries go by row now, not columns
X_byrow
```

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

```
# Some math operations
X_sqrt <- sqrt(X)
X_sqrt
```

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

```
X_sq <- X^2
X_sq
```

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

Some statistics

```
# rnorm: generate normal random vector

set.seed(150) # Reproduce same random vectors

x <- rnorm(50) # n=50, mu=0, std=1
y <- x + rnorm(50, mean=50, sd=.1)
cor(x,y) # Correlation between random vectors
```

```
## [1] 0.9960615
```

```
# Statistics
mean(y) # Sample mean
```

```
## [1] 50.07481
```

```
var(y) # Sample variance
```

```
## [1] 1.21916
```

```
sqrt(var(y)) # Sample std deviation
```

```
## [1] 1.104156
```

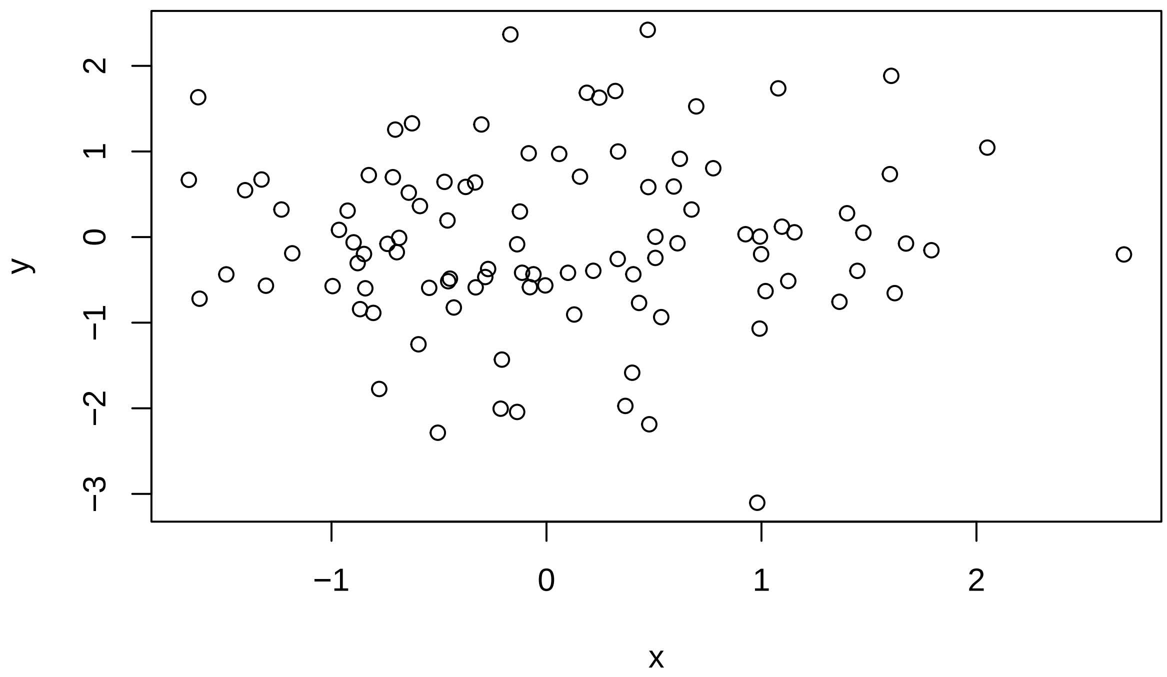
```
sd(y) #Also sample std deviation
```

```
## [1] 1.104156
```

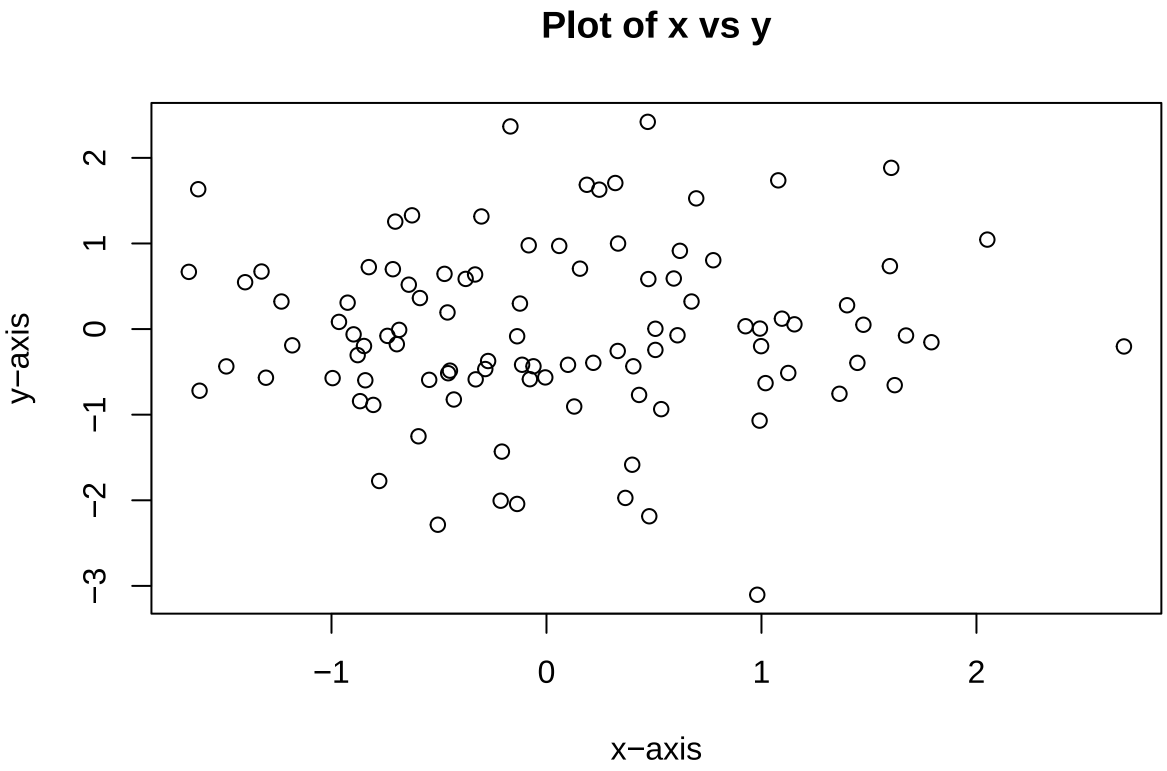
2.3.2 Graphics

Basic plotting

```
x <- rnorm(100)
y <- rnorm(100)

plot(x,y)# Basic plot of x vs y
```



```r
plot(x,y,
     xlab="x-axis",
     ylab="y-axis",
     main="Plot of x vs y") # Plot with labels
```



```r
# Creating pdf
pdf("example.pdf")
plot(x,y, col="orange")
dev.off() # Done with plotting
```

```
## pdf
##   2
```

```r
# Sequences
x <- seq(1,10) #seq function
x <- 1:10 # Easier way to create sequence
x
```

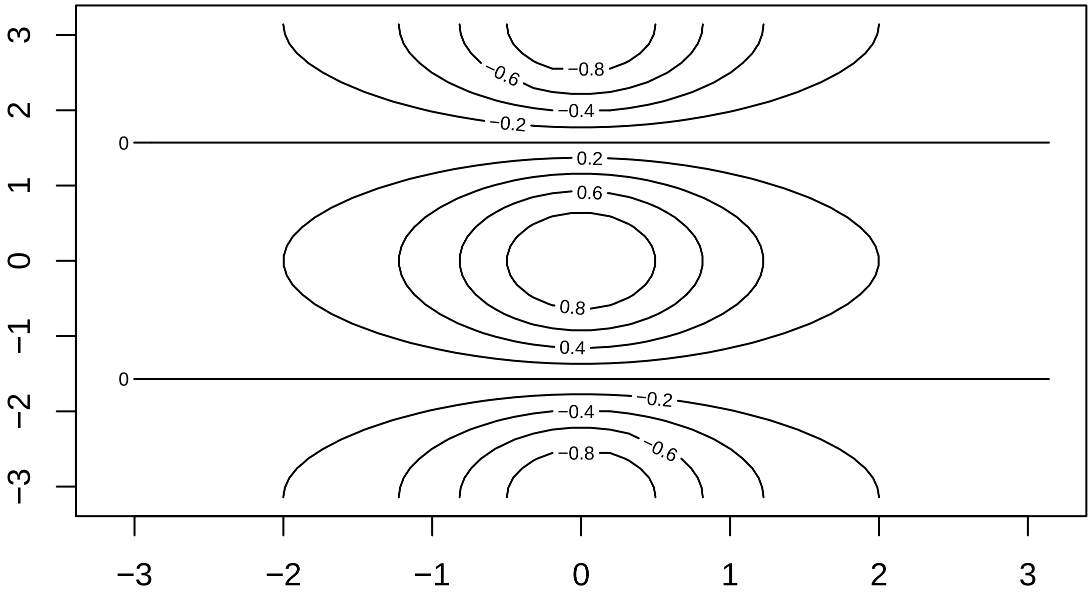
```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
x <- seq(-pi, pi, length=50) # Evenly spaced sequence of numbers between (-pi,pi)
x
```

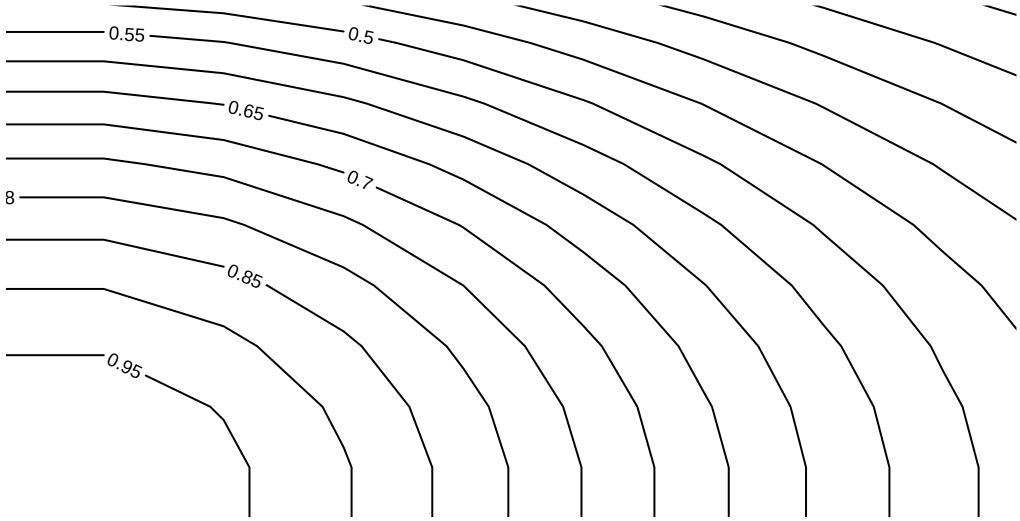
```
##  [1] -3.14159265 -3.01336438 -2.88513611 -2.75690784 -2.62867957 -2.50045130
##  [7] -2.37222302 -2.24399475 -2.11576648 -1.98753821 -1.85930994 -1.73108167
## [13] -1.60285339 -1.47462512 -1.34639685 -1.21816858 -1.08994031 -0.96171204
## [19] -0.83348377 -0.70525549 -0.57702722 -0.44879895 -0.32057068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

```
y <- x
f <- outer(x, y, function(x, y) cos(y) / (1 + x^2))
contour(x, y, f)
```



```
# Note: 3 args for contour:
# 1. x vals
# 2. y vals
# 3. matrix of corresponding function values of the coordinate pair (x,y)

plot.new()

contour(x, y, f, nlevels = 45, add = T)
```



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

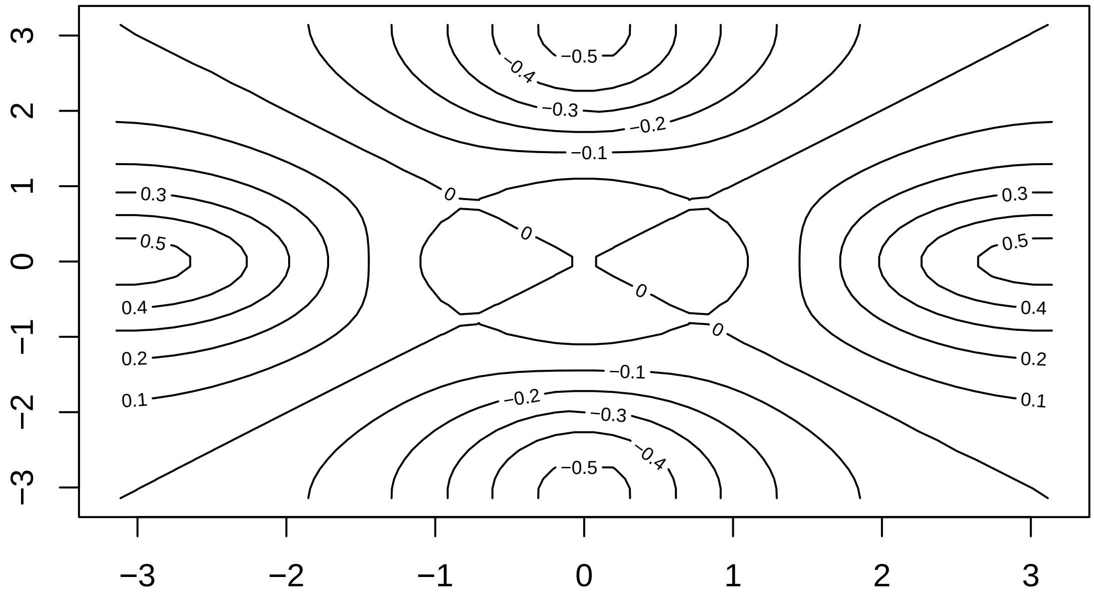
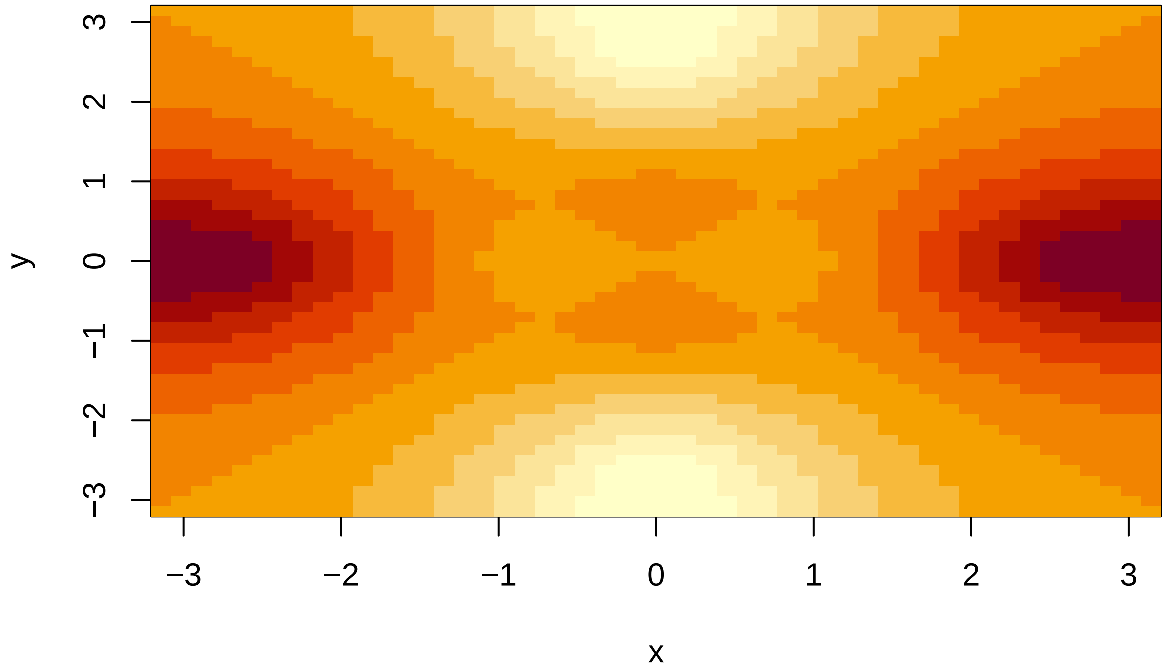
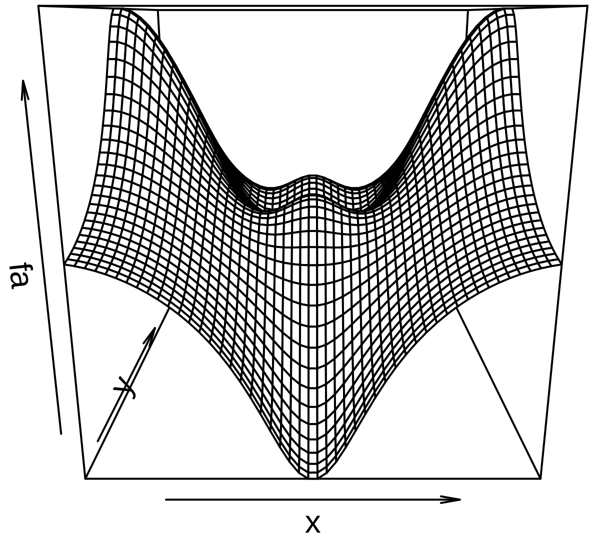


Image function

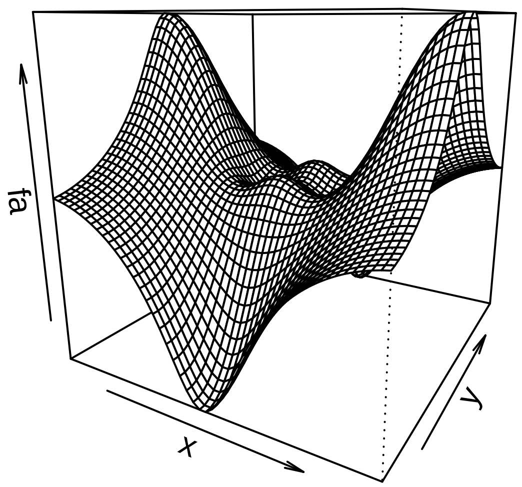
```
image(x, y, fa) # Creates a heatmap where color depends on z vals
```



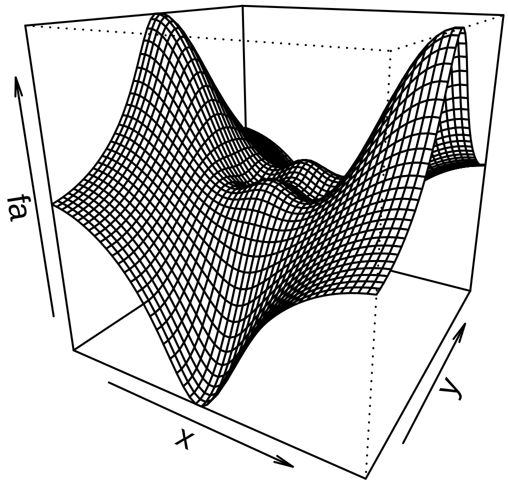
```
persp(x, y, fa) # Creates 3d plots
```



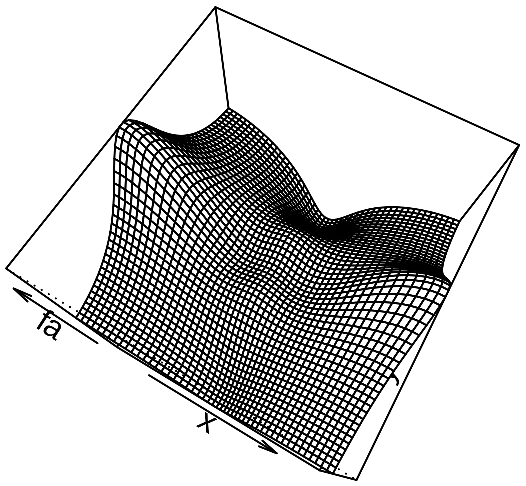
```r
persp(x, y, fa, theta = 30)
```



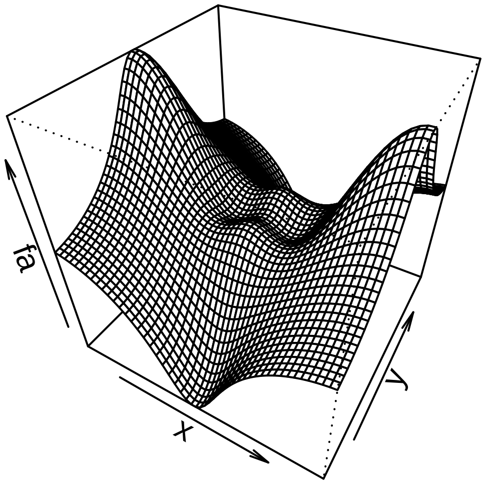
```
persp(x, y, fa, theta = 30, phi = 20) # theta and phi params control angle at which we view the 3d plot
```



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



```r
persp(x, y, fa, theta = 30, phi = 40)
```



2.3.3 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] #i=2, j=3
```

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

```
A[c(1, 3), c(2, 4)] # A_12, A_14, A_32, A_34
```

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

```
A[1:3, 2:4] # Indexing rows 1-3 and cols 2-4
```

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

```
A[1:2, ] # First two rows, all cols
```

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

```
A[, 1:2] # All rows, first two cols
```

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

```
# Note: R treats any column or row as a vector
A[1,] # First row vector of A
```

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

```
A[,1] # First col vector of A
```

```
## [1] 1 2 3 4
```

```
A[-c(1,3), -c(2,3)] # Negative indexing removes rows 1 and 3, cols 2 and 3
```

```
##      [,1] [,2]
## [1,]    2   14
## [2,]    4   16
```

```
# Dimension of matrix
dim(A)
```

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

2.3.4 Loading Data

```
# Loading Auto data
auto <- read.table("Auto.data")
View(auto) # View in separate window
head(auto)
```

```r
auto <- read.table("Auto.data", header = TRUE, na.strings = "?", stringsAsFactors = TRUE)
# Notes:
# header = TRUE : Indicate to use first row of dataset as variable names
# na.strings = "?" : Indicates that na vals == "?" in the data
# stringsAsfactors = TRUE : Indicate that any string vars are qualitative
head(auto)
```

```r
# CSV data
auto = read.csv("Auto.csv", na.strings = TRUE, stringsAsFactors = TRUE)
View(auto)
dim(auto)
```

```
## [1] 397   9
```

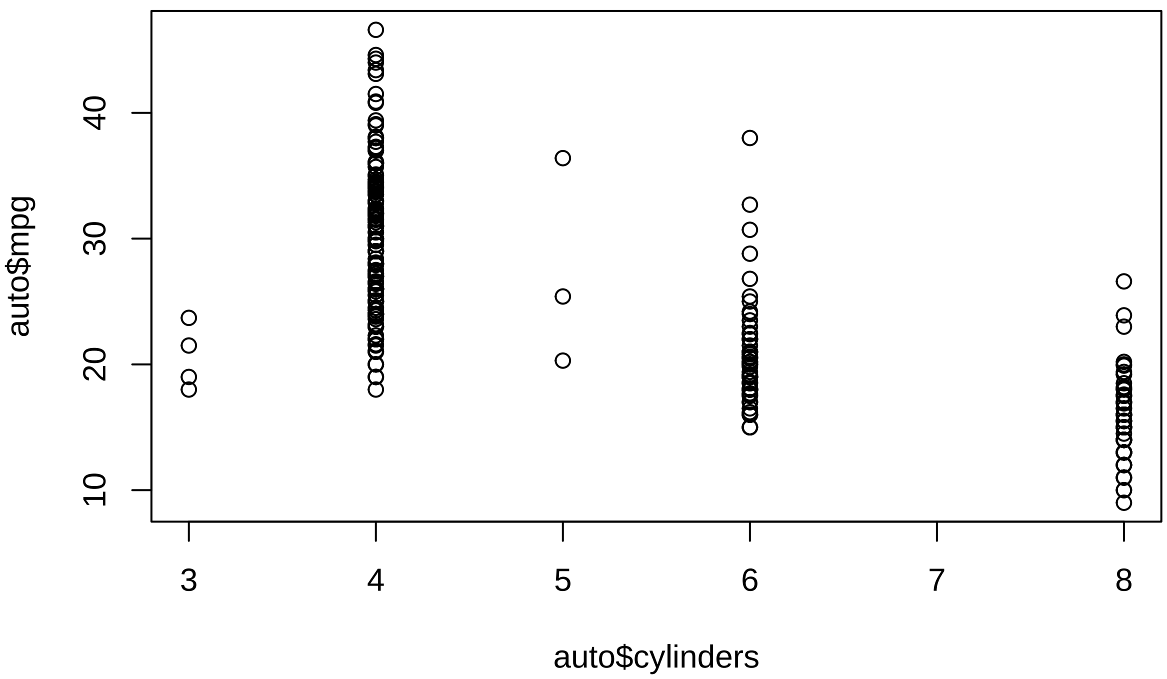
```r
names(auto) # Var names
```

```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"      
## [6] "acceleration" "year"         "origin"       "name"
```

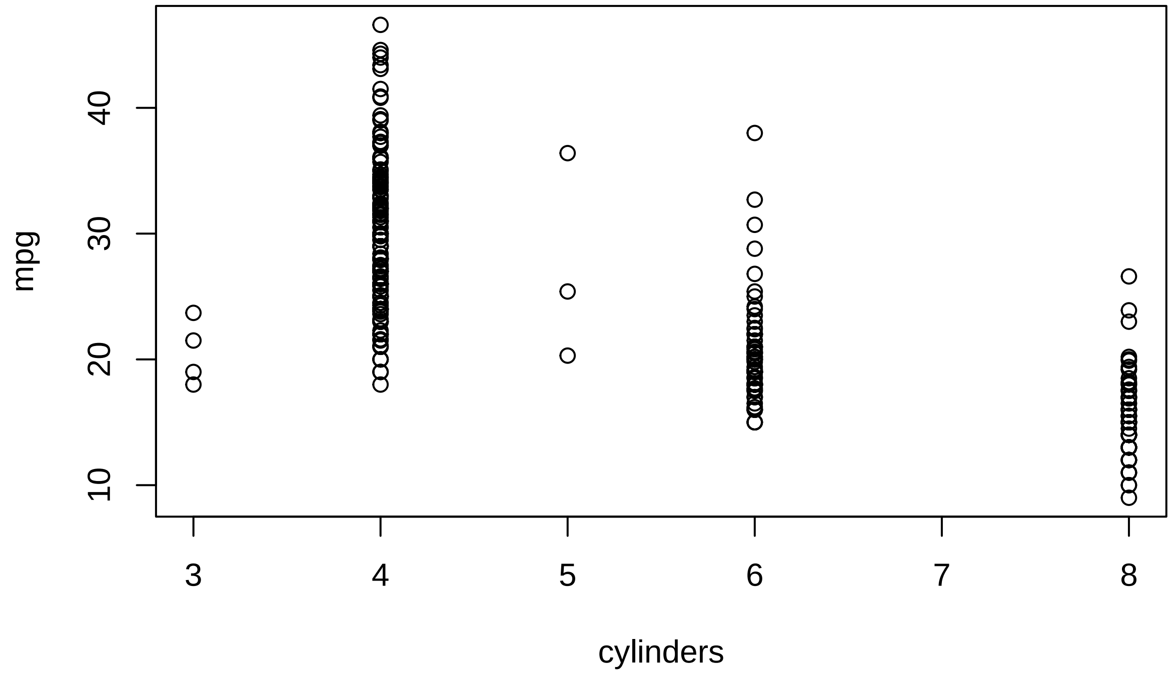
2.3.5 Additional numerical and graphical methods

Plotting from datasets

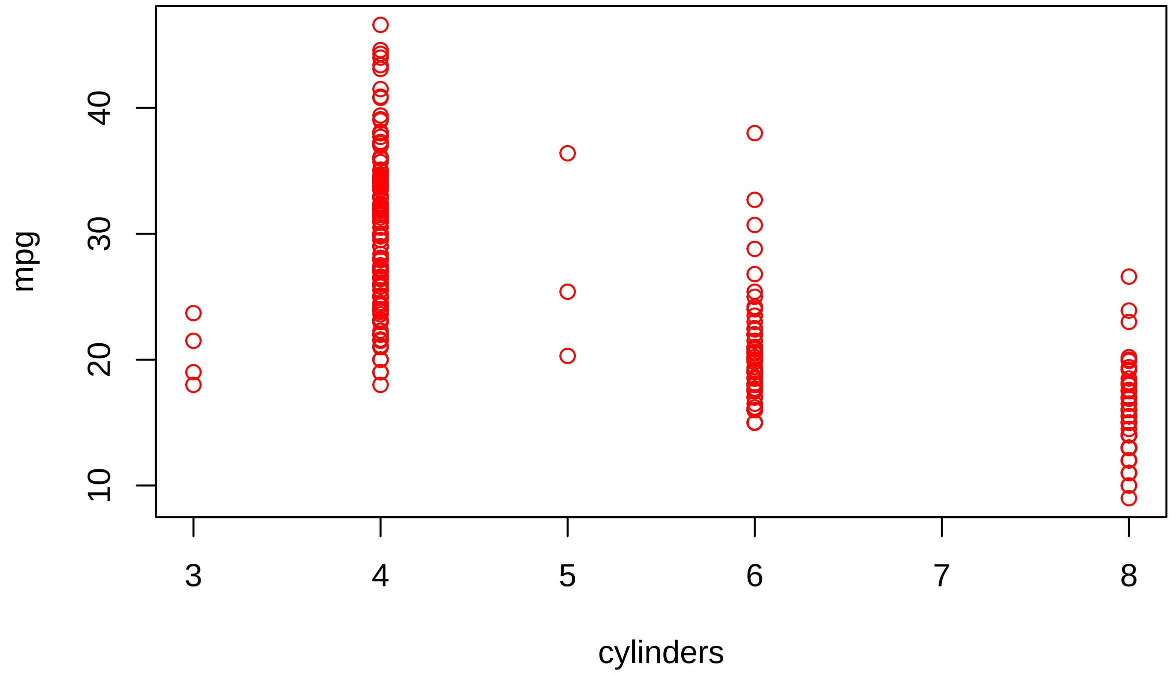
```r
# We can reference variables from the dataset with $
plot(auto$cylinders, auto$mpg)
```



```
# Alternatively, we can use attach() to access the variables of auto
attach(auto)
plot(cylinders,mpg)
```

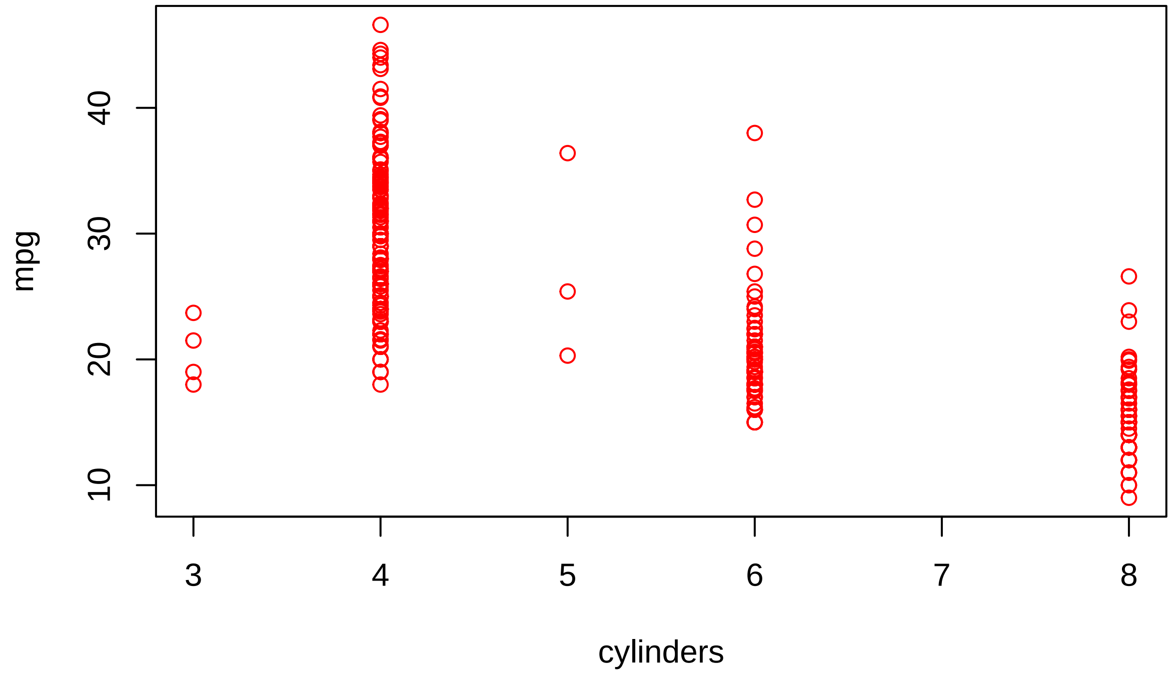


```
# Cylinders is a categorical variable, so we can change it to a qualitative variable
auto$cylinders <- as.factor(cylinders)

# Modifying plots
plot(cylinders, mpg)
plot(cylinders, mpg, col = "red") # Changes points to red color
```



```
plot(cylinders, mpg, col = "red", varwidth = TRUE)
```

```
## Warning in plot.window(...): "varwidth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "varwidth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter
## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter

## Warning in box(...): "varwidth" is not a graphical parameter

## Warning in title(...): "varwidth" is not a graphical parameter
```

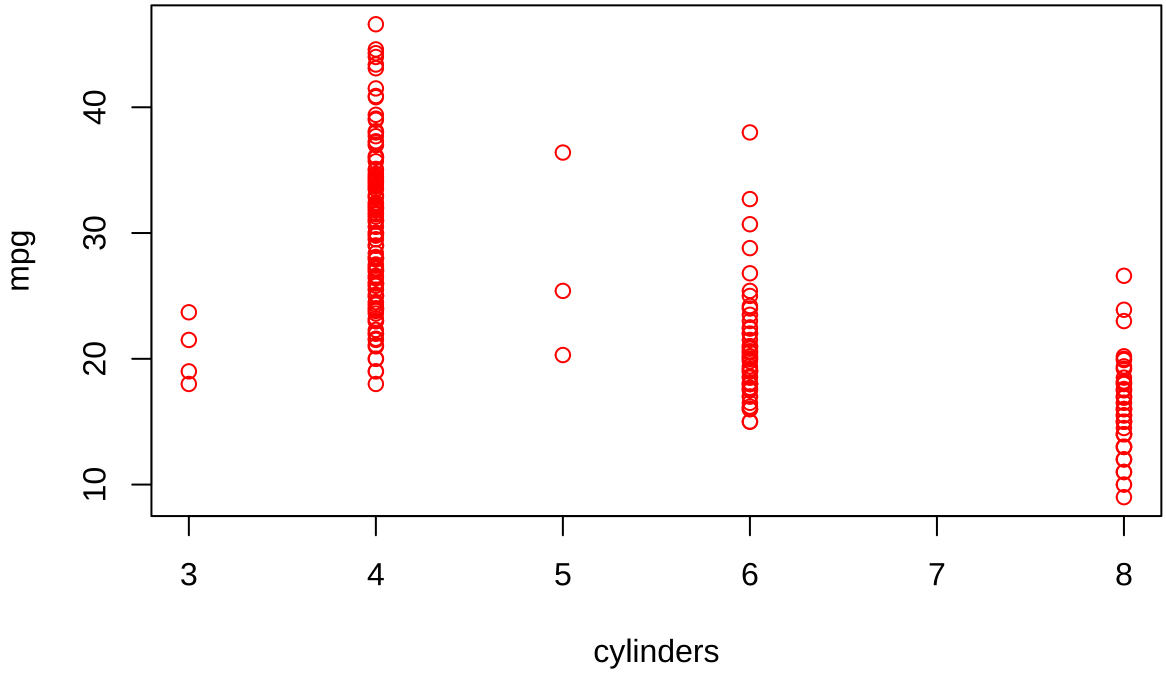


```
plot(cylinders, mpg, col = "red", varwidth = T,
 horizontal = T)
```

```
## Warning in plot.window(...): "varwidth" is not a graphical parameter

## Warning in plot.window(...): "horizontal" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "varwidth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "horizontal" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "horizontal" is
## not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "horizontal" is
## not a graphical parameter

## Warning in box(...): "varwidth" is not a graphical parameter
## Warning in box(...): "horizontal" is not a graphical parameter
```

```
## Warning in title(...): "varwidth" is not a graphical parameter
```

```
## Warning in title(...): "horizontal" is not a graphical parameter
```



```r
plot(cylinders, mpg, col = "red", varwidth = T,
 xlab = "cylinders", ylab = "MPG")
```

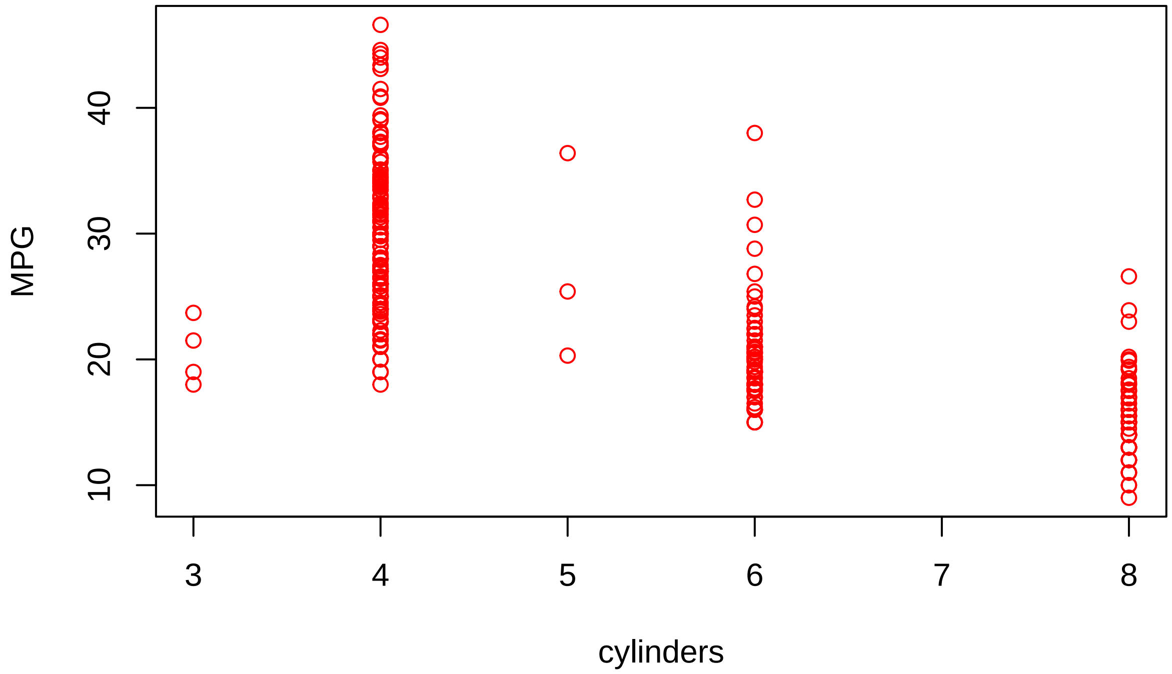
```
## Warning in plot.window(...): "varwidth" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "varwidth" is not a graphical parameter
```

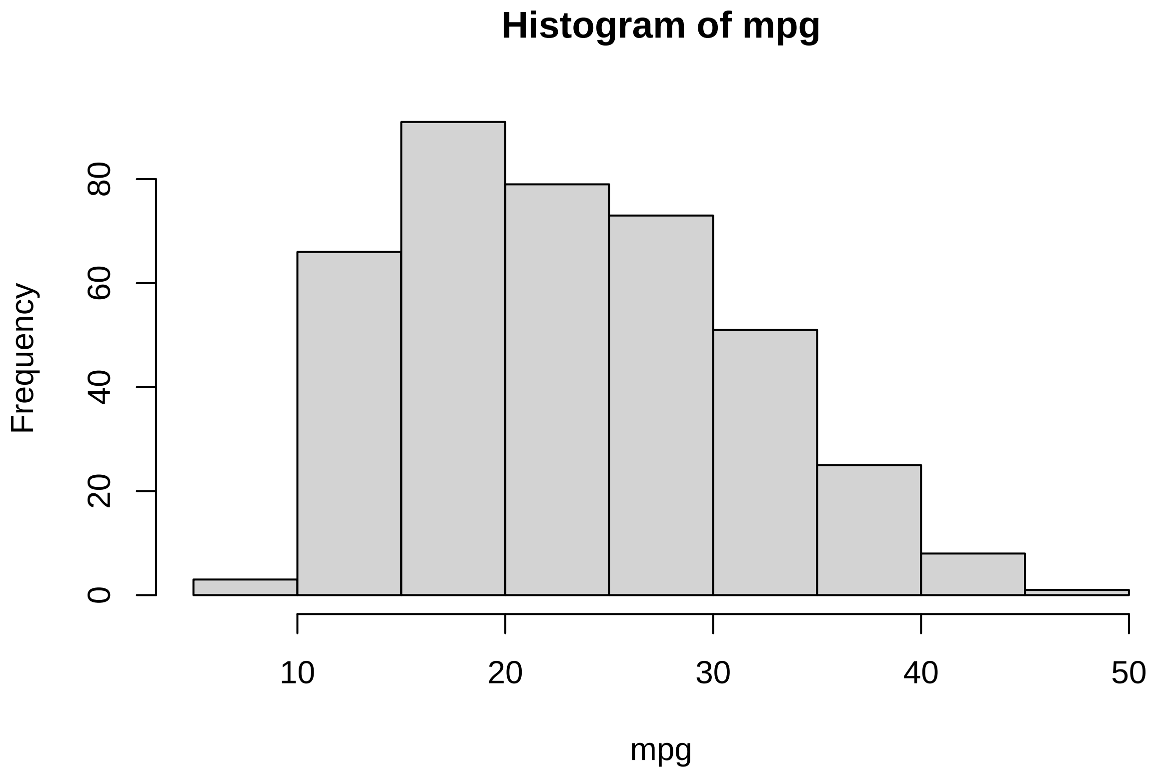
```
## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter
## Warning in axis(side = side, at = at, labels = labels, ...): "varwidth" is not
## a graphical parameter
```

```
## Warning in box(...): "varwidth" is not a graphical parameter
```

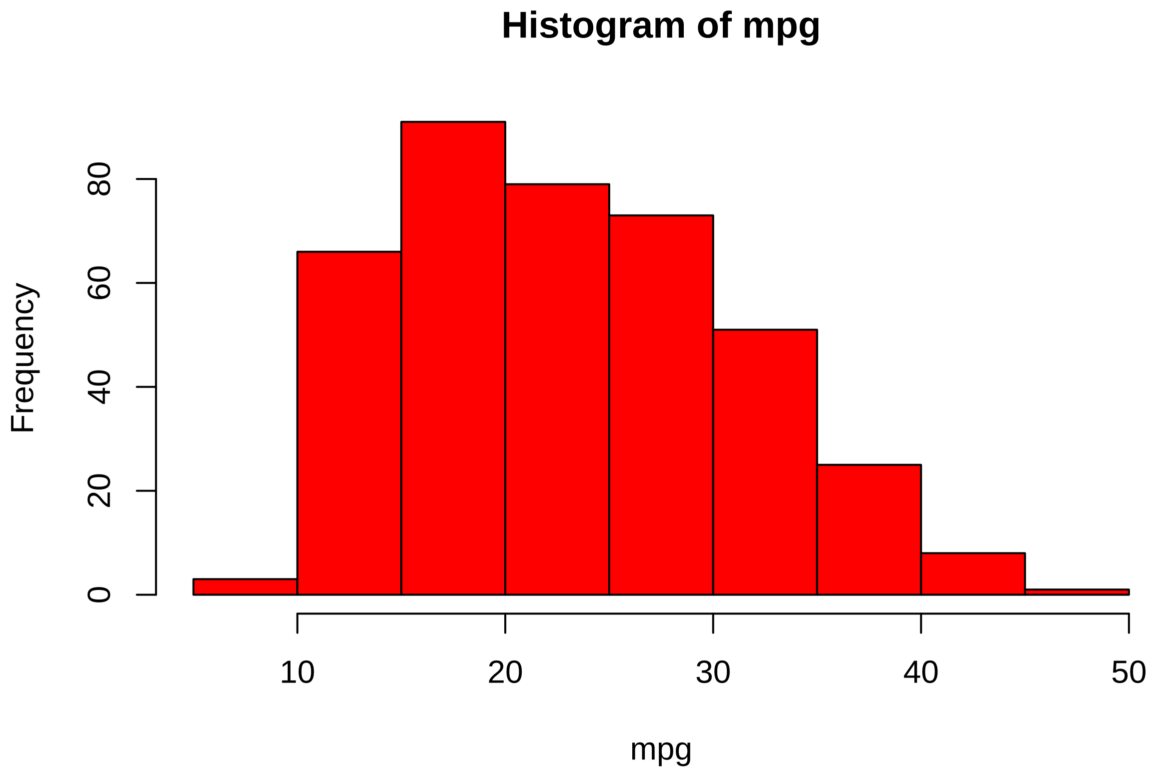
```
## Warning in title(...): "varwidth" is not a graphical parameter
```



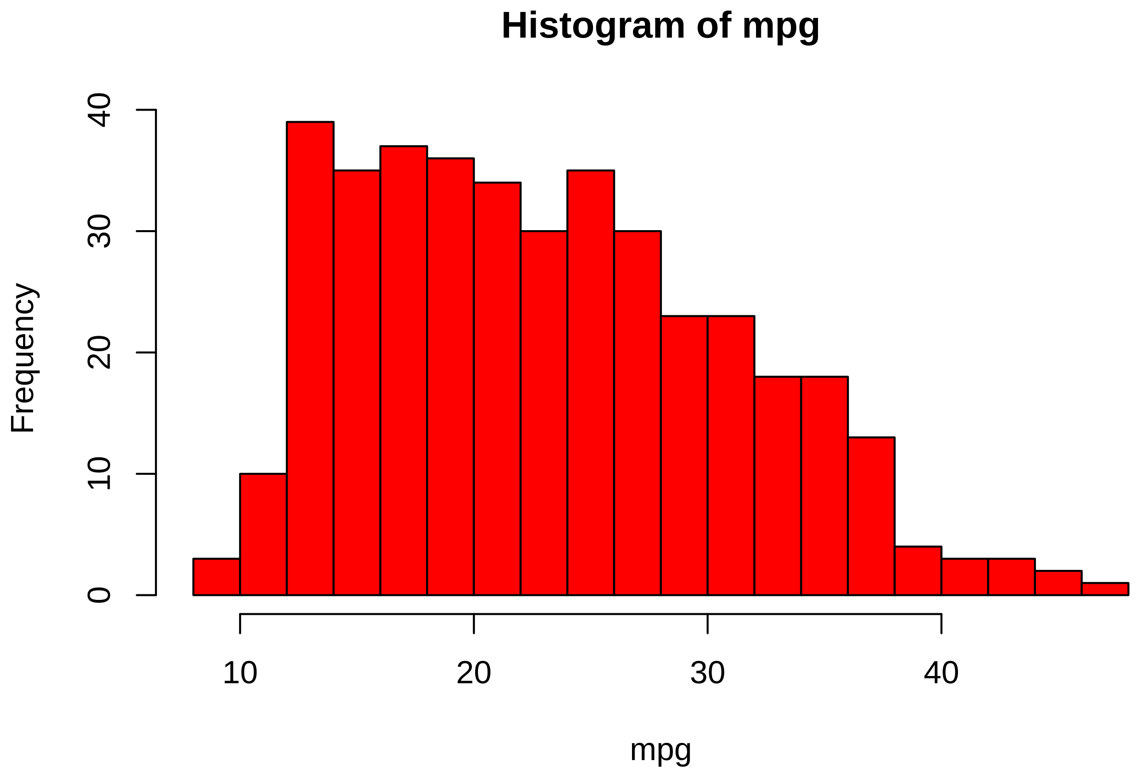
```
# Note: Says varwidth and horizontal are not params? What to do about this?

# Histograms
hist(mpg)
```



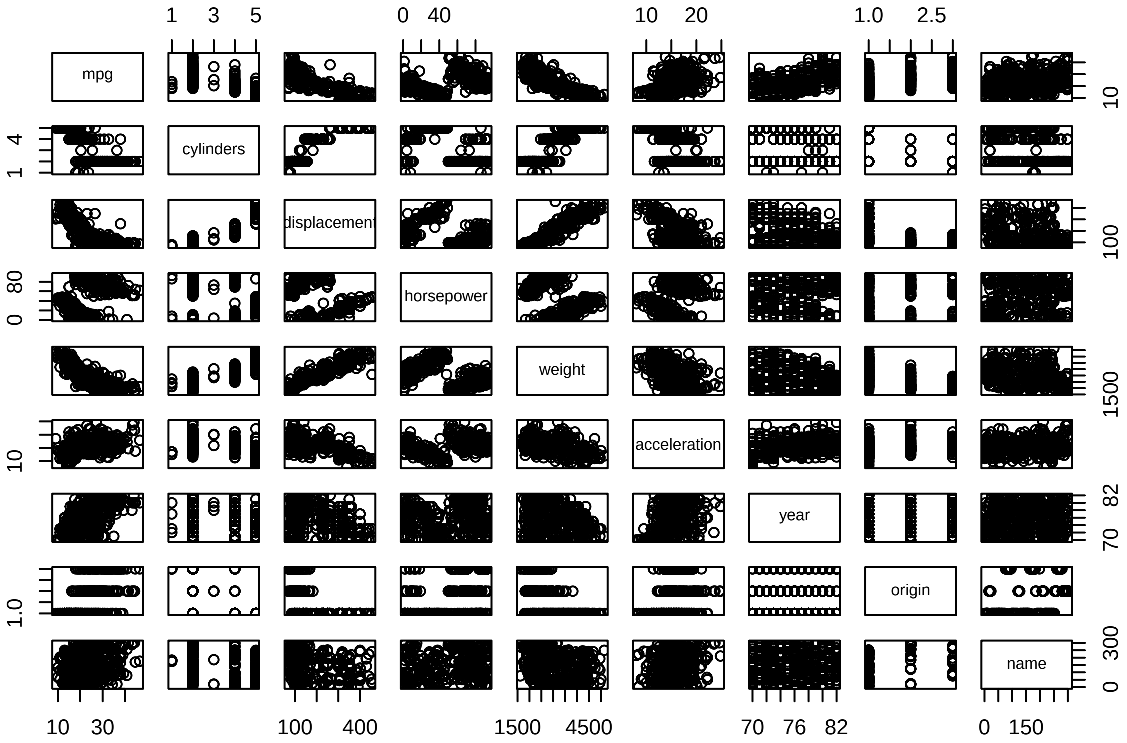
```
hist(mpg, col="red") # Change color of bars to red
```



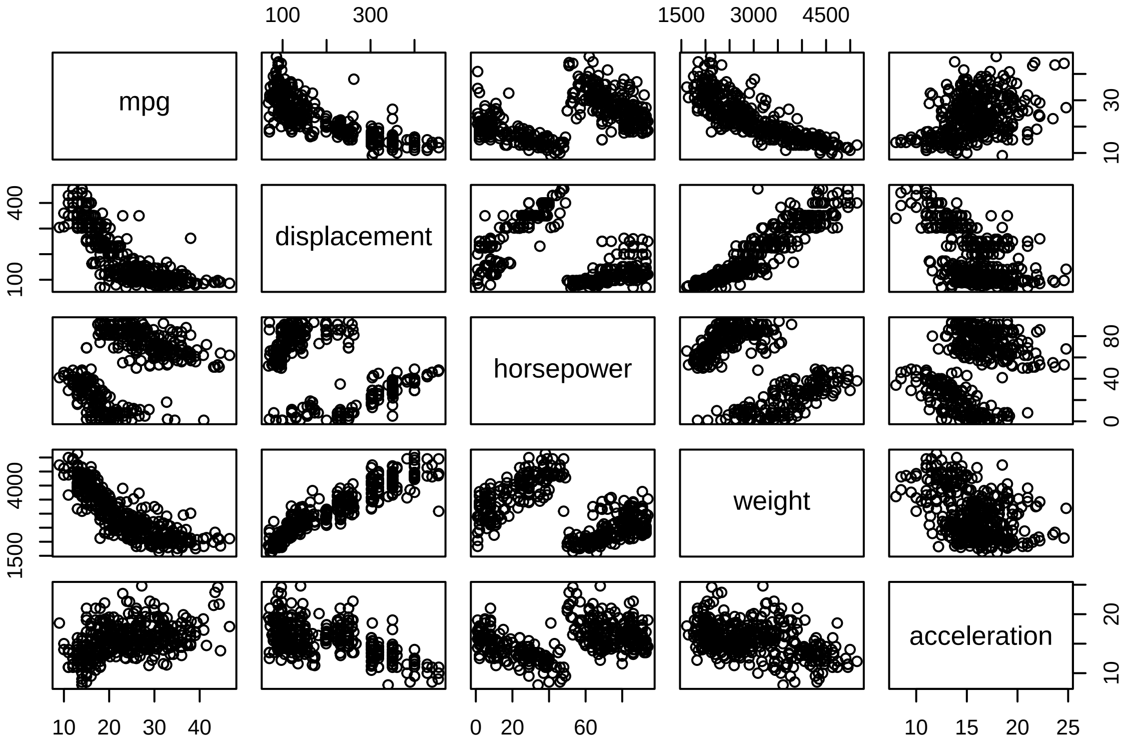
```r
hist(mpg, col="red", breaks=15) # Change amount of bars
```



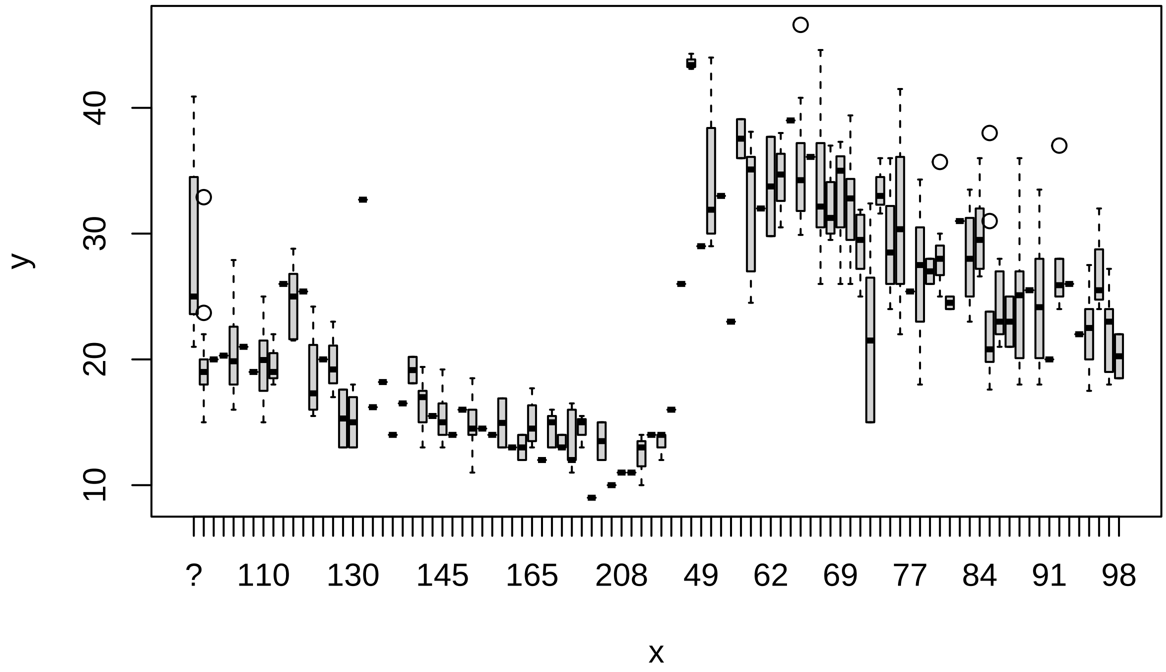
```
# Scatterplot matrices using pairs()
pairs(auto)
```



```r
pairs(
 ~mpg + displacement + horsepower + weight + acceleration,
 data = auto
 ) # Specify a subset of variables for the plot
```



```
# identify() function
plot(horsepower, mpg)
# 3 args: x var, y var, and var we want to see printed for each point
identify(horsepower, mpg, name)
```



```
## integer(0)
```

Numerical functions

```
# Summary table for each variable in the dataset
summary(auto)
```

```
##       mpg          cylinders  displacement     horsepower      weight    
##  Min.   : 9.00   3:  4     Min.   : 68.0   150    : 22   Min.   :1613  
##  1st Qu.:17.50   4:203     1st Qu.:104.0   90     : 20   1st Qu.:2223  
##  Median :23.00   5:  3     Median :146.0   88     : 19   Median :2800  
##  Mean   :23.52   6: 84     Mean   :193.5   110    : 18   Mean   :2970  
##  3rd Qu.:29.00   8:103     3rd Qu.:262.0   100    : 17   3rd Qu.:3609  
##  Max.   :46.60             Max.   :455.0   75     : 14   Max.   :5140  
##                                            (Other):287                 
##   acceleration        year           origin                   name    
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   ford pinto    :  6  
##  1st Qu.:13.80   1st Qu.:73.00   1st Qu.:1.000   amc matador   :  5  
##  Median :15.50   Median :76.00   Median :1.000   ford maverick :  5  
##  Mean   :15.56   Mean   :75.99   Mean   :1.574   toyota corolla:  5  
##  3rd Qu.:17.10   3rd Qu.:79.00   3rd Qu.:2.000   amc gremlin   :  4  
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   amc hornet    :  4  
##                                                  (Other)       :368
```

```
# We can also do a summary of a single var
summary(mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    9.00   17.50   23.00   23.52   29.00   46.60
```

```
# Saving and loading history
#savehistory()
#loadhistory()
```