Graphics

Canhong Wen

Agenda

- Basic graphics
- Custom graphics

Functions for graphics

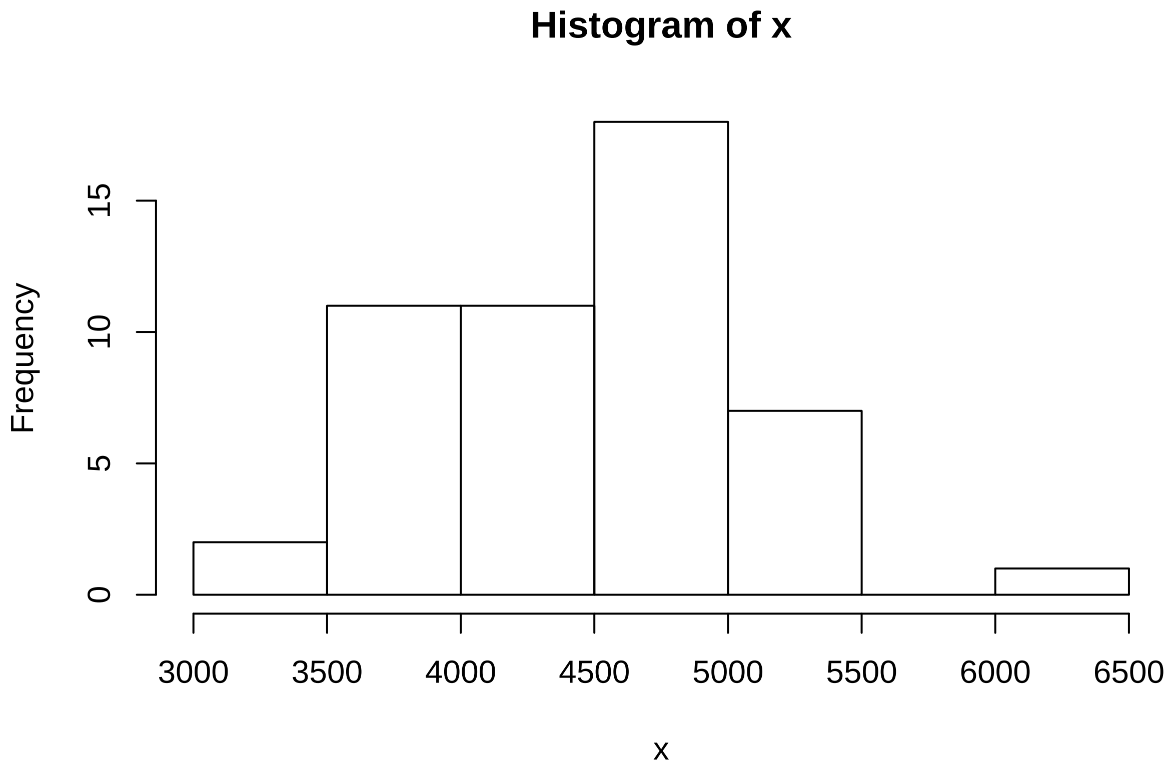
- The functions `hist()`, `boxplot()`, `plot()`, `points()`, `lines()`, `text()`, `mtext()`, `axis()`, etc. form a suite that plot graphs and add features to the graph
- Each of these functions have various options, to learn more about them, use the help
- `par()` can be used to set or query graphical parameters

Basic graphics

- Univariate data:
  - continuous: density plots(histogram and kernel density plots), ecdf
  - Categorical: pie charts, bar charts
- Bivariate data:
  - Two continuous data: Sscatterplots, qqplots
  - One contunuous and one categorical data: boxplots
  - Two categorical data: stacked/grouped bar charts
- Trivariate data:
  - pairwise plots
  - 3D plots(image plots, coutour plots)

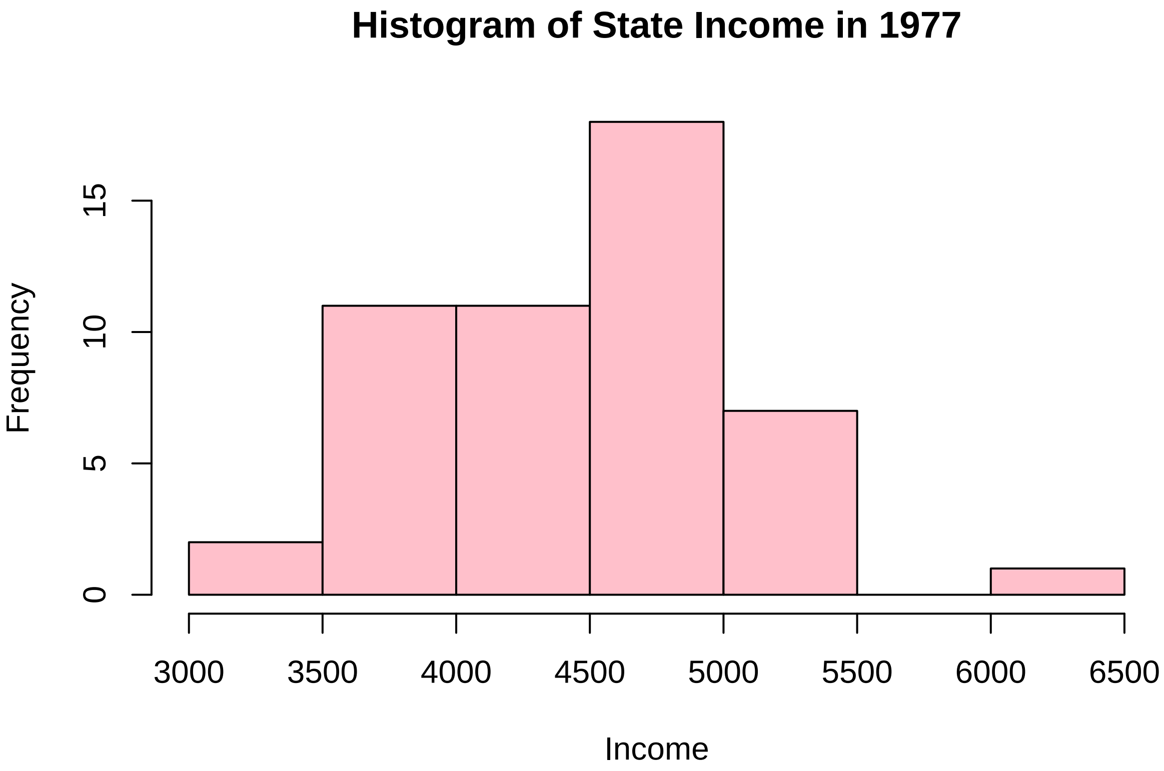
Univariate data: Histogram

```
x <- state.x77[, 2]            # 50 average state incomes in 1977
hist(x)
```



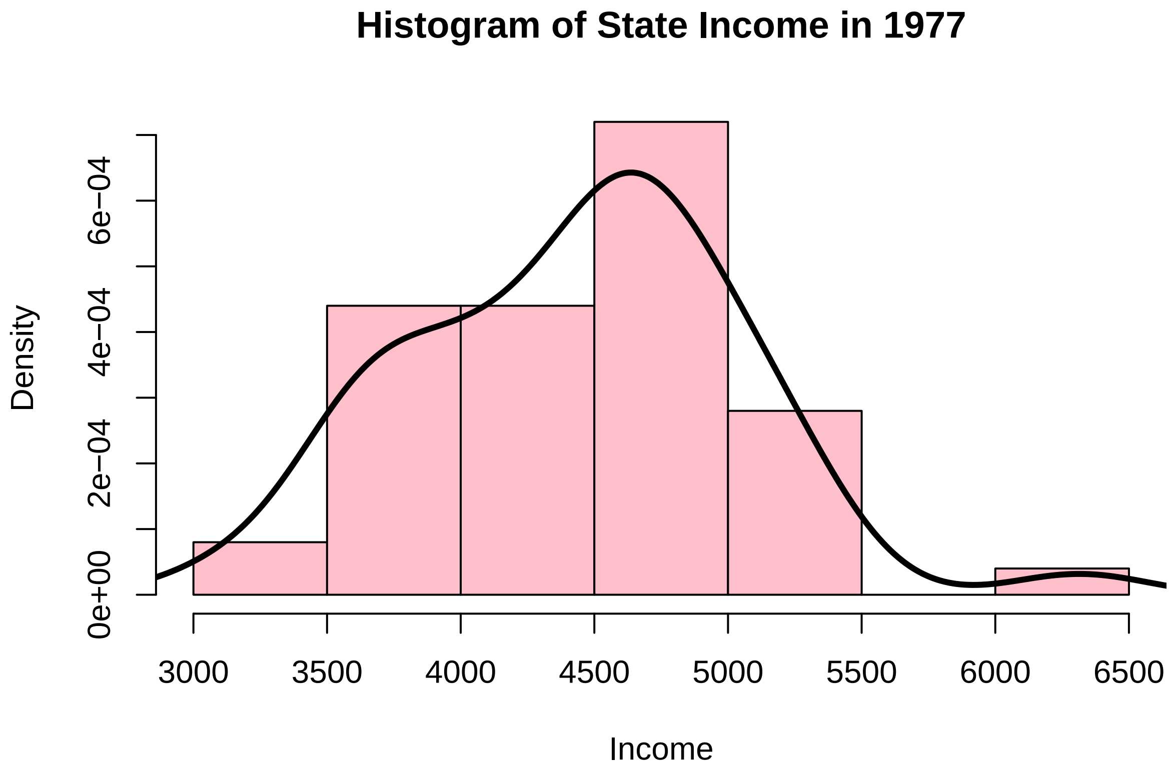
Univariate data: Histogram

```
hist(x, breaks = 8, col="pink", xlab="Income", main="Histogram of State Income in 1977")
```



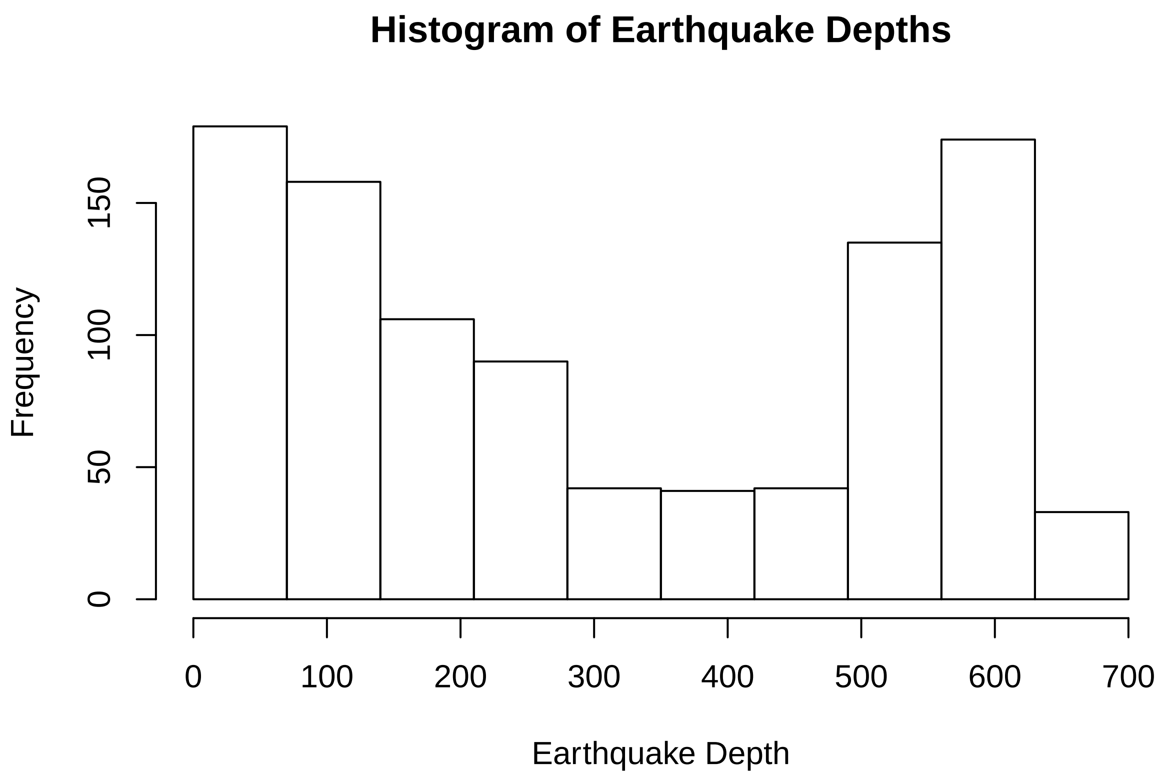
Univariate data: Histogram with a density plot

```r
hist(x, breaks = 8, col="pink", freq = FALSE, xlab="Income", main="Histogram of State Income in 1977")
lines(density(x), lwd=3)
```



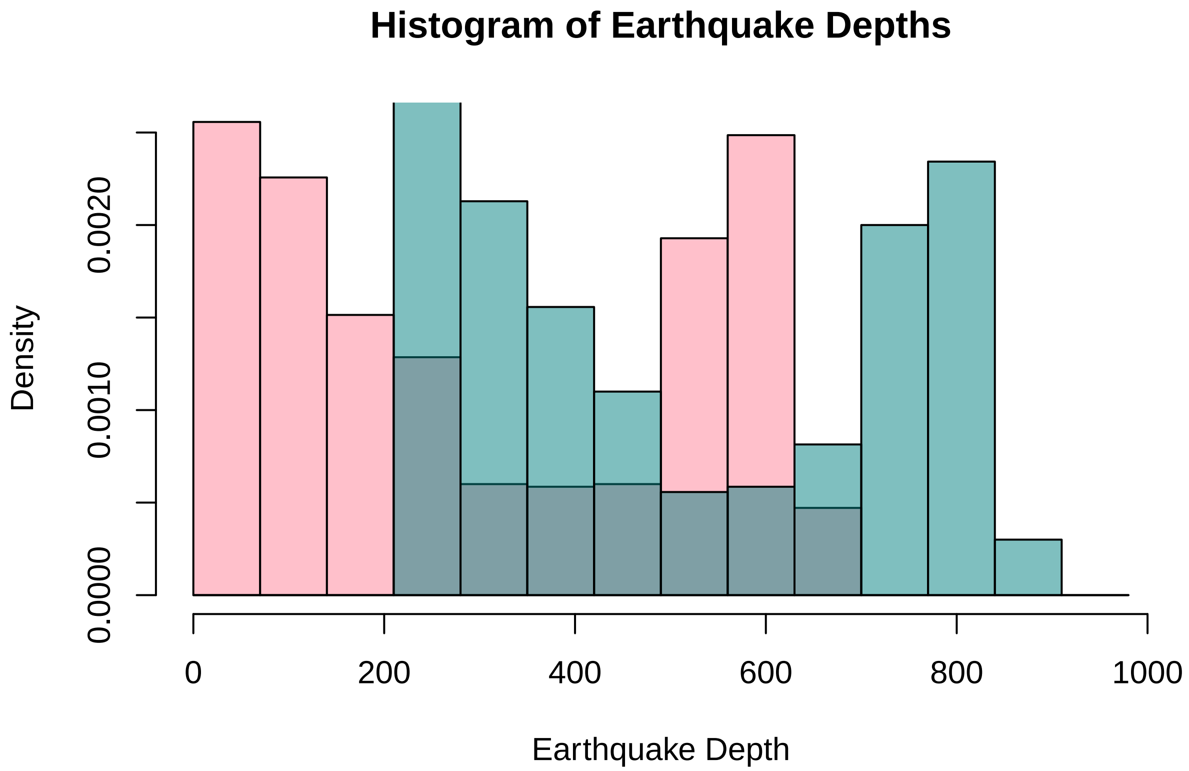
Univariate data: Histogram

```
y <- quakes$depth                            # 1000 earthquake depths
hist(y, seq(0, 700, by = 70), xlab="Earthquake Depth", main="Histogram of Earthquake Depths")
```



Univariate data: Histogram

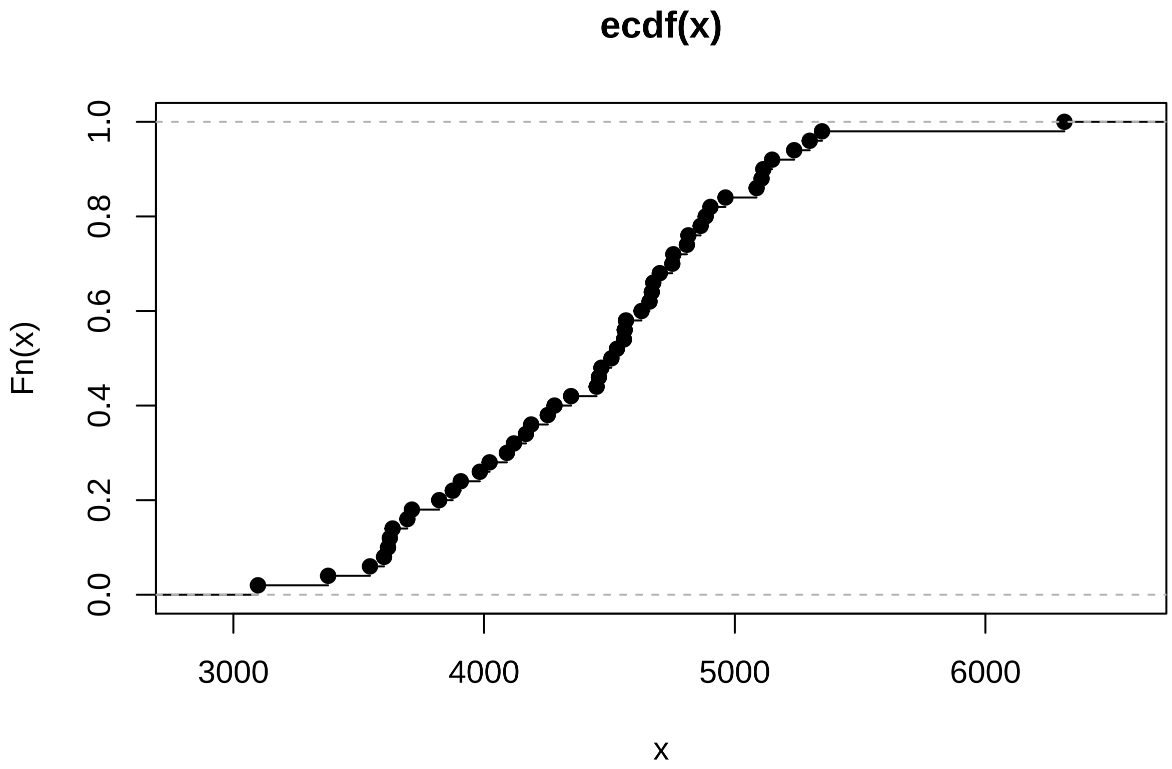
```
y <- quakes$depth                            # 1000 earthquake depths
hist(y, seq(0, 1000, by = 70), freq = FALSE, col = "pink",
     xlab="Earthquake Depth", main="Histogram of Earthquake Depths")
hist(y+ 200, seq(0, 1000, by = 70), freq = FALSE, col = rgb(0,0.5, 0.5, 0.5), add = TRUE)
```



Univariate data: Empirical CDF

Function `plot.ecdf()` provides data for empirical cdf

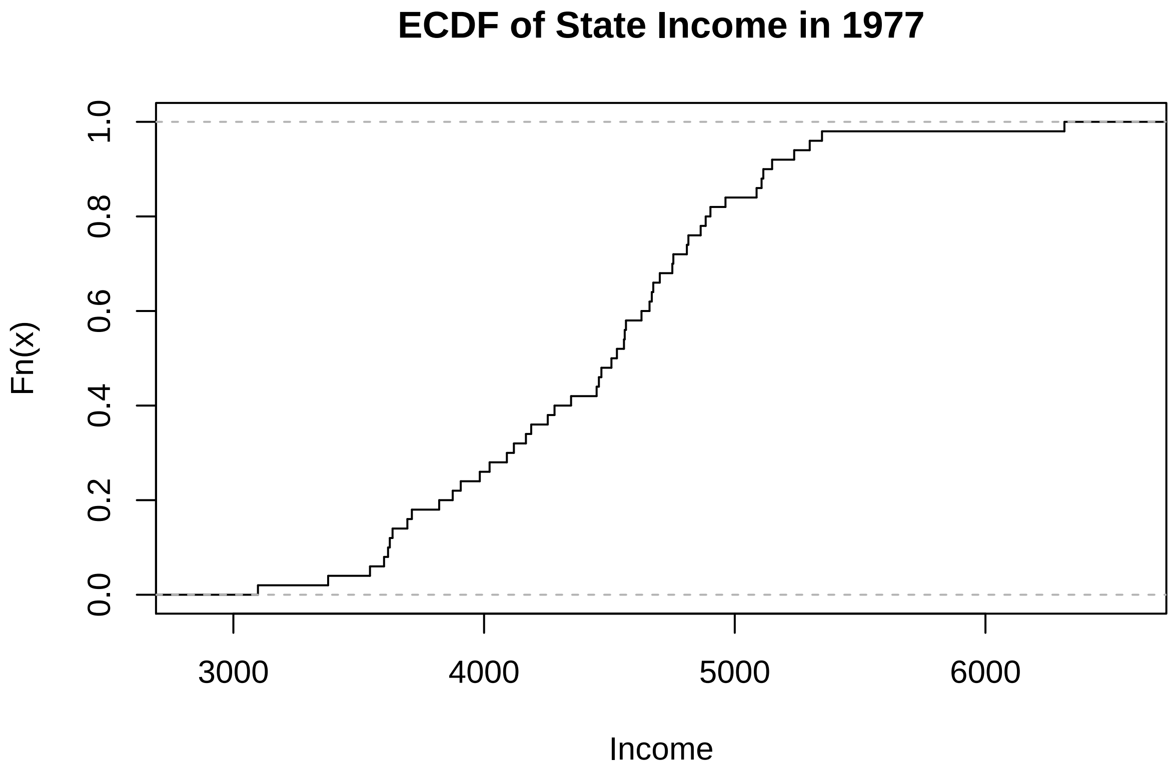
```
plot.ecdf(x)
```



Univariate data: Empirical CDF

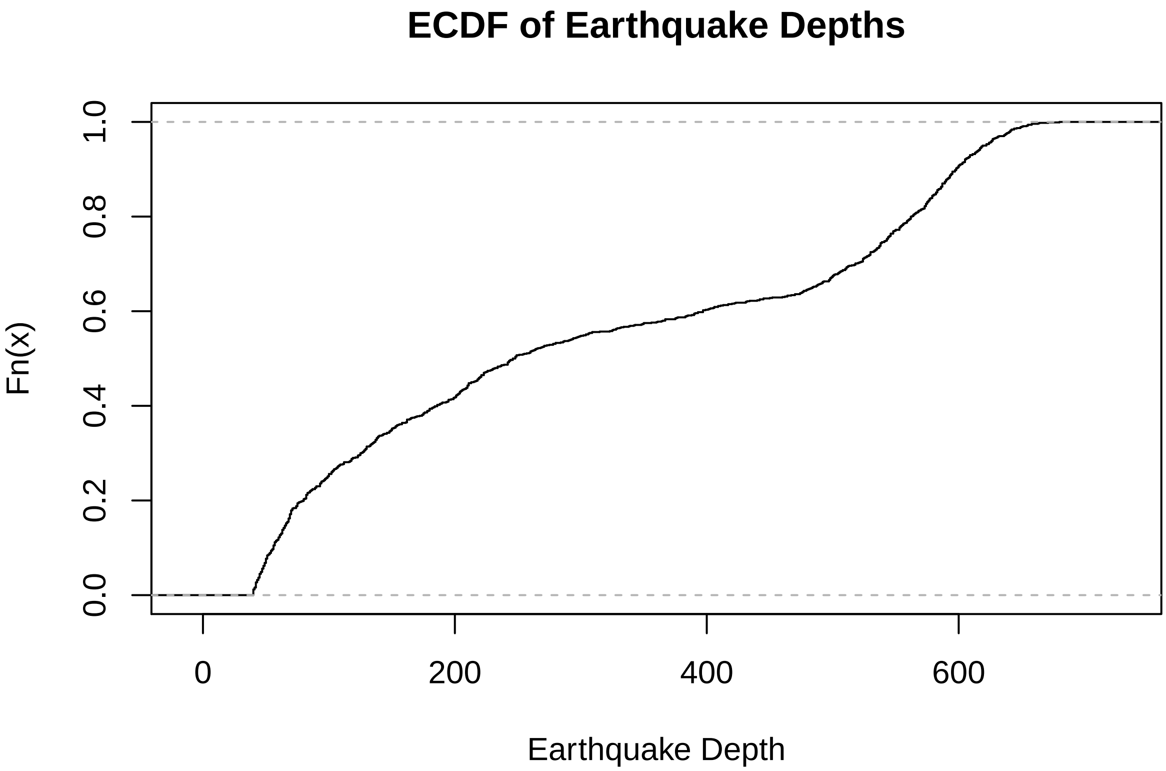
Can add vertical lines and remove dots

```
plot.ecdf(x, verticals = T, pch = "", xlab = "Income",
          main = "ECDF of State Income in 1977")
```



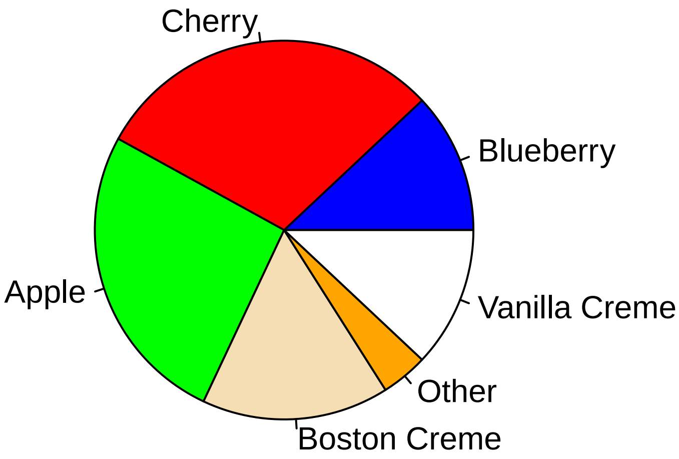
Univariate data: Empirical CDF

```
plot.ecdf(y, verticals = T, pch = "", xlab = "Earthquake Depth",
          main = "ECDF of Earthquake Depths")
```



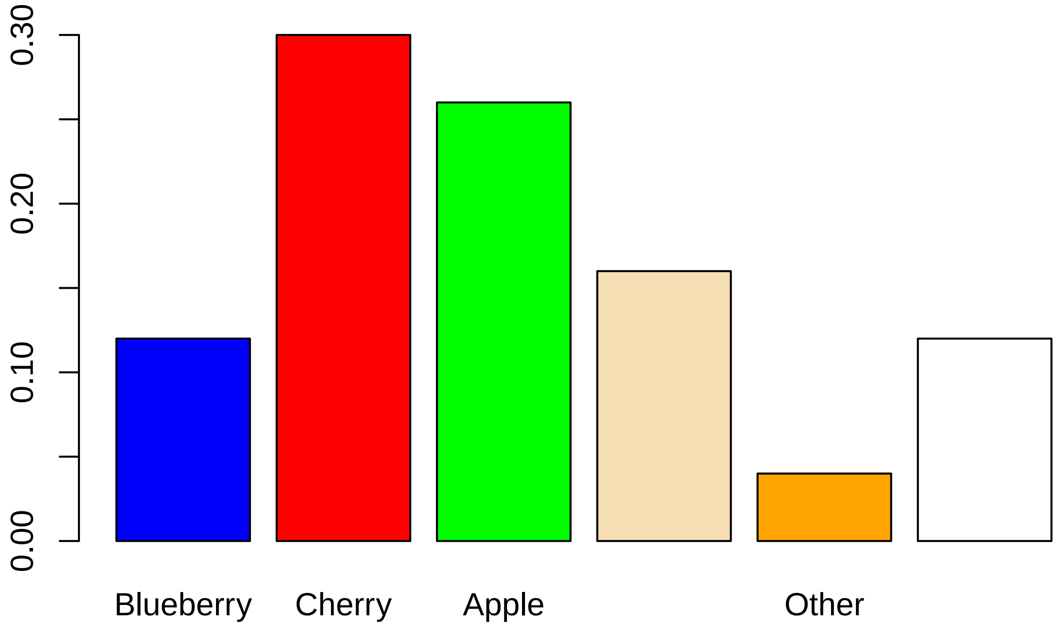
Univariate data: Pie charts

```
pie.sales = c(0.12, 0.30, 0.26, 0.16, 0.04, 0.12)
names(pie.sales) = c("Blueberry", "Cherry", "Apple", "Boston Creme",
                     "Other", "Vanilla Creme")
pie(pie.sales, col = c("blue", "red", "green", "wheat", "orange", "white"))
```

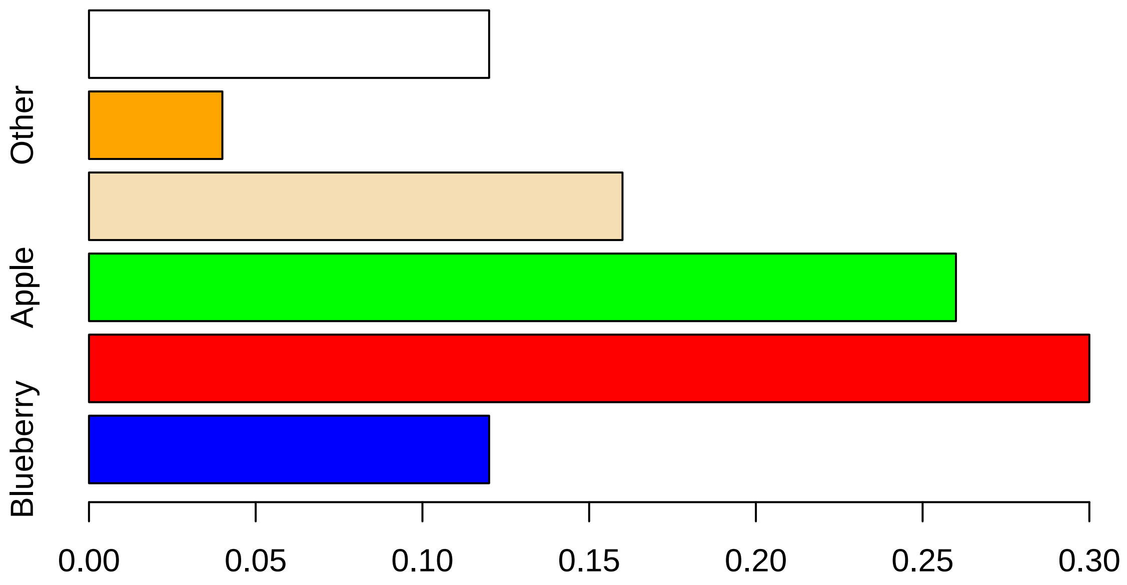


Univariate data: bar charts

```
barplot(pie.sales, col = c("blue", "red", "green", "wheat", "orange", "white"))
```



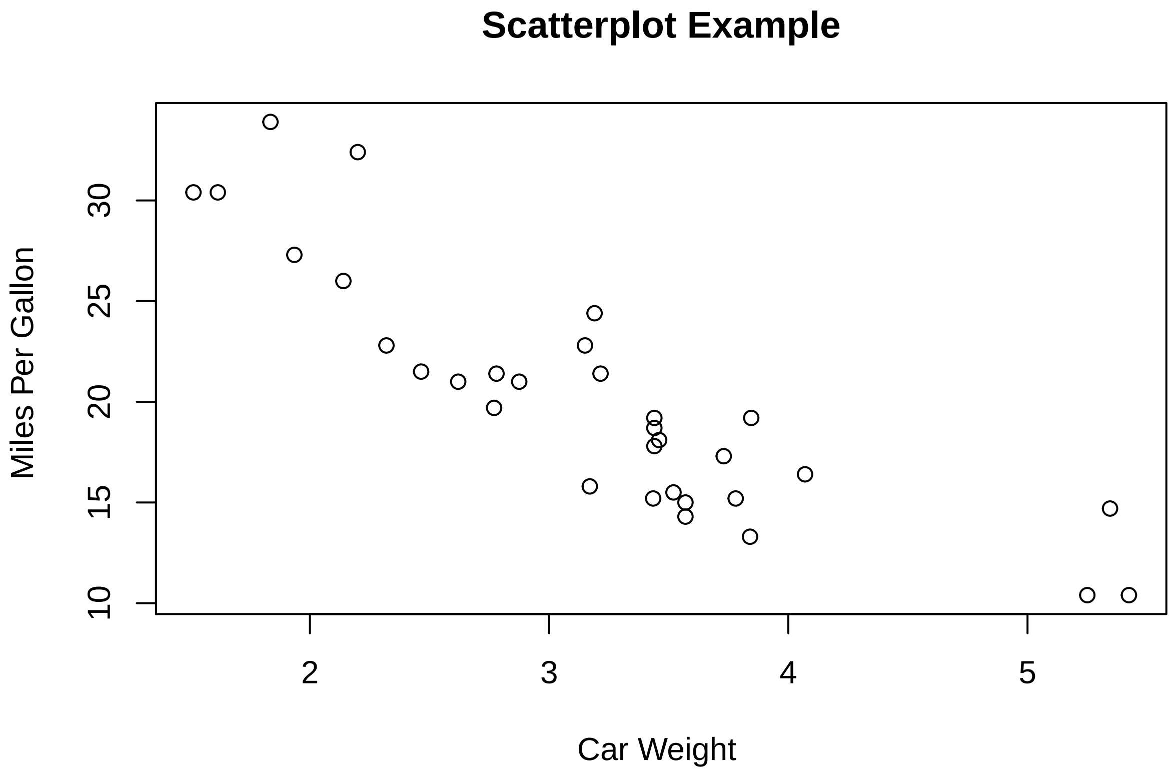
```r
barplot(pie.sales, col = c("blue", "red", "green", "wheat", "orange", "white"), horiz = TRUE)
```



Bivariate data

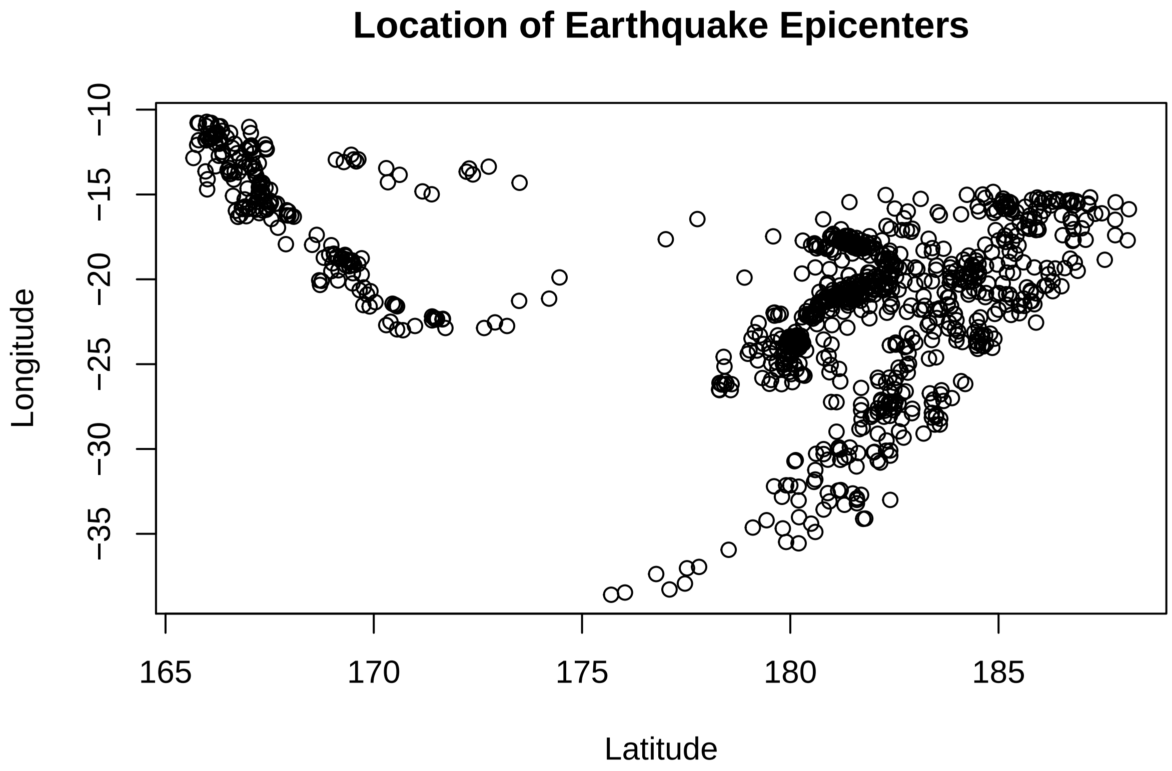
Scatterplots: `plot(x, y)`

```r
# Simple Scatterplot
attach(mtcars)
plot(wt, mpg, main="Scatterplot Example",
   xlab="Car Weight ", ylab="Miles Per Gallon ")
```



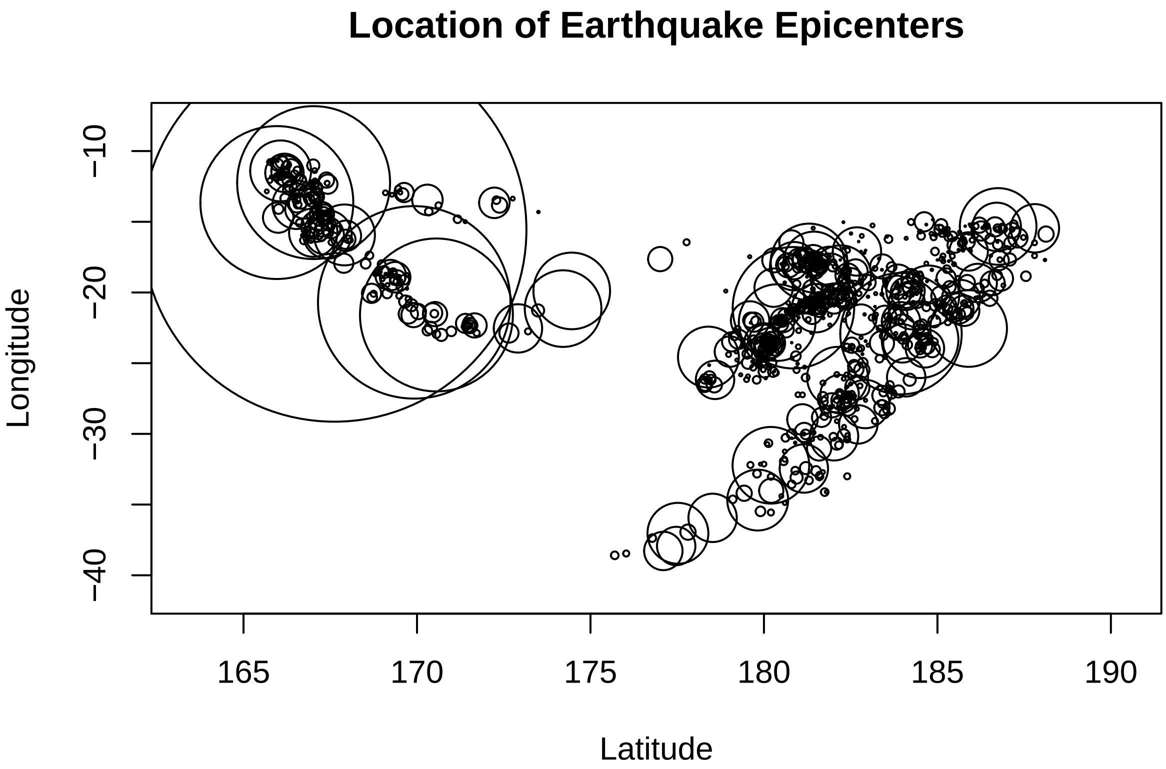
```
detach(mtcars)
```

```
plot(quakes$long, quakes$lat, xlab="Latitude", ylab="Longitude",
     main="Location of Earthquake Epicenters")
```



Scatterplots: `plot(x, y)`

```r
symbols(quakes$long, quakes$lat, circles = 10 ^ quakes$mag,
        xlab="Latitude", ylab="Longitude",
        main="Location of Earthquake Epicenters")
```

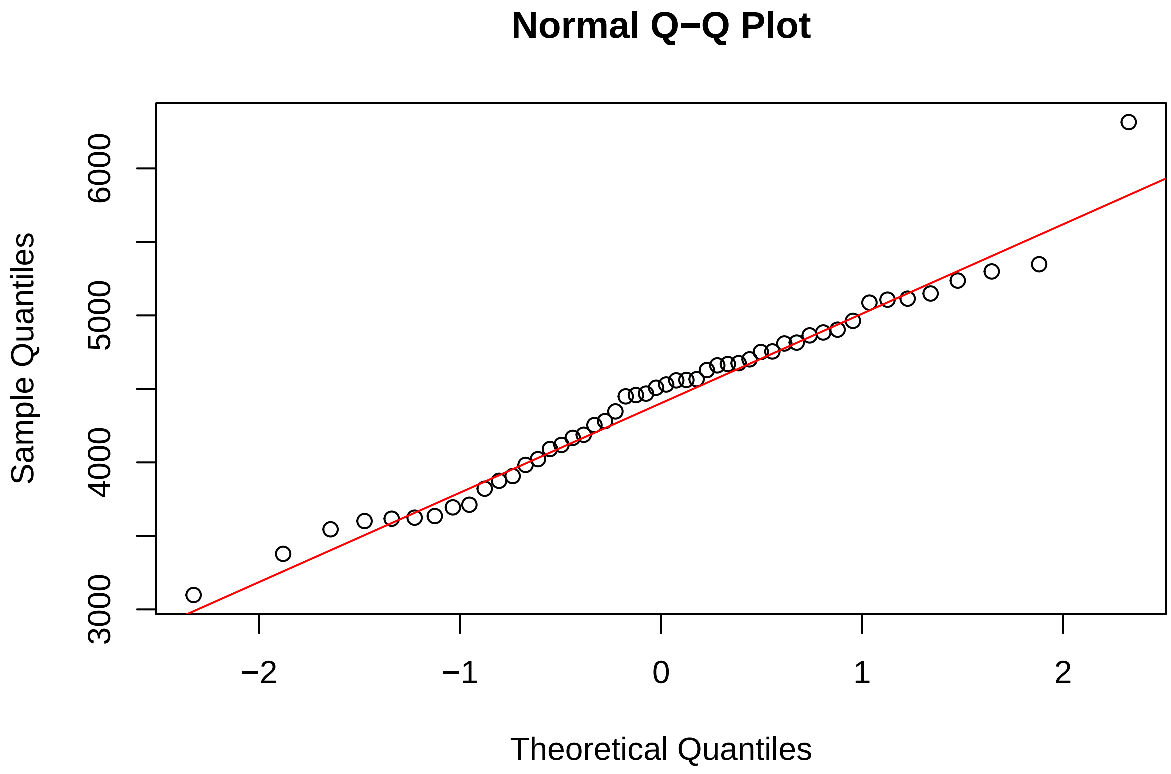


`qqnorm()` and `qqplot()`

- `qqnorm()` plots the quantiles of a data set against the quantiles of a Normal distribution
- `qqplot()` plots the quantiles of a first data set against the quantiles of a second data set

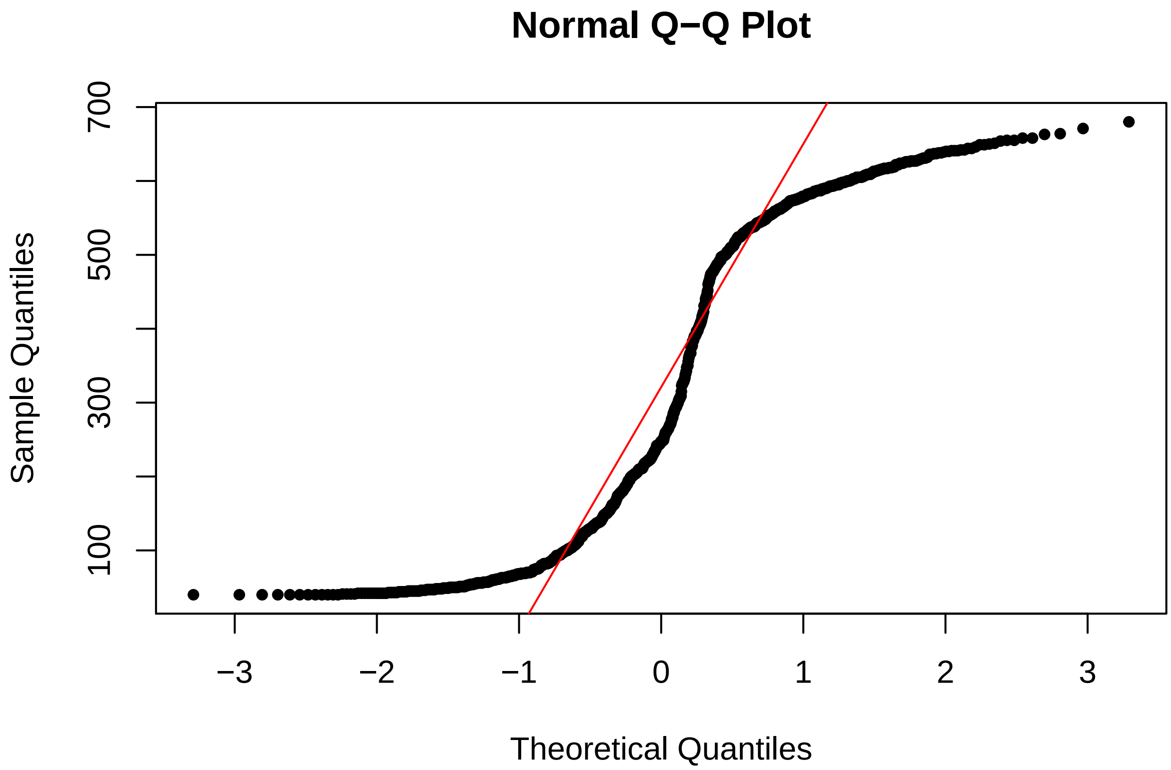
`qqnorm()` and `qqplot()`

```
qqnorm(x)                         # qq plot for the earthquake depths
qqline(x, col = "red")             # red reference line
```



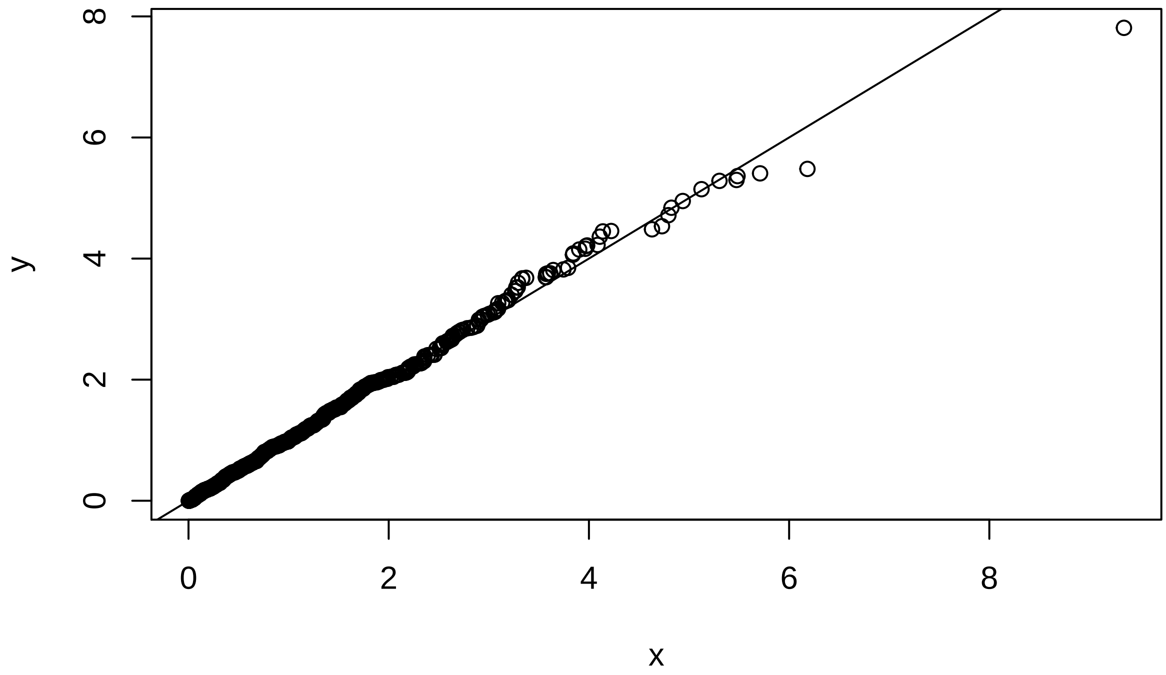
`qqnorm()` and `qqplot()`

```
qqnorm(y, pch=20)                        # qq plot for the earthquake depths
qqline(y, col = "red")         # red reference line
```



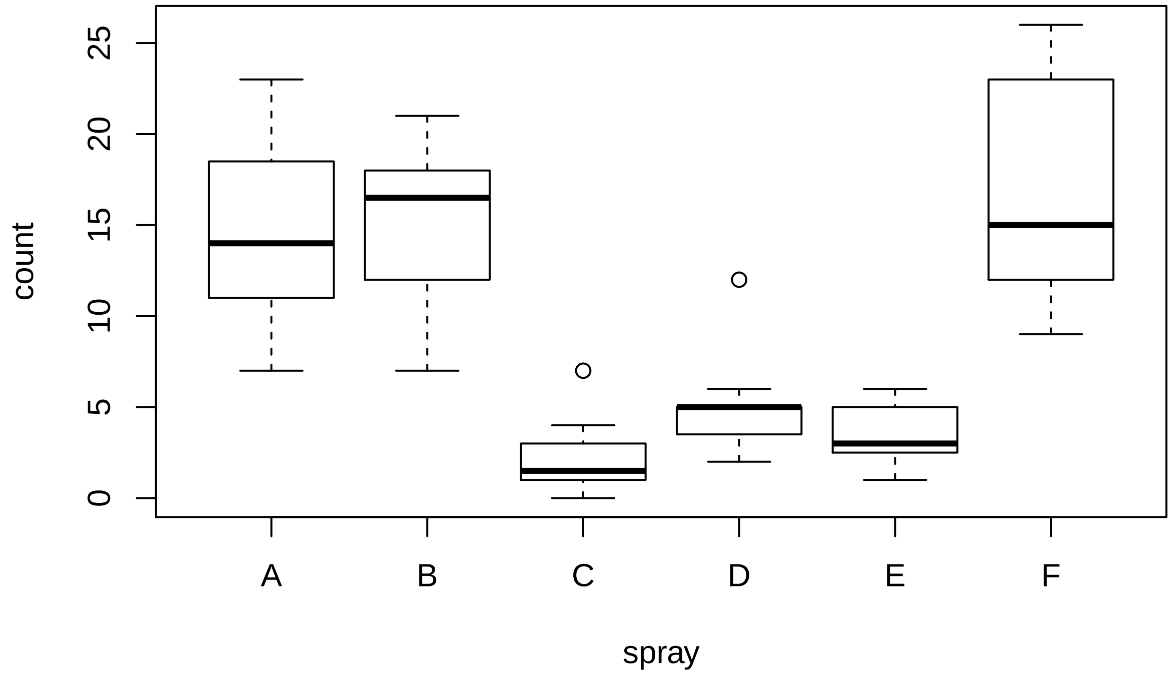
`qqnorm()` and `qqplot()`

```
x <- rexp(1000)
y <- rexp(1000)
qqplot(x,y)
abline(a=0,b=1)
```



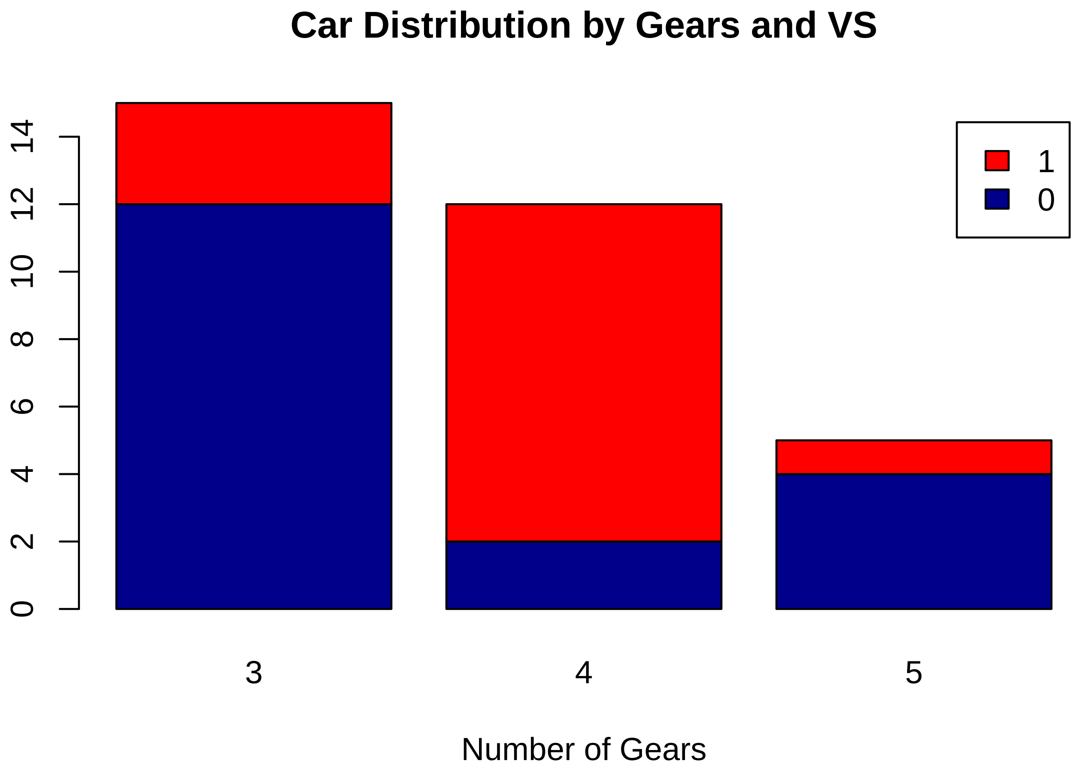
Box plots

```r
boxplot(count ~ spray, data = InsectSprays)
```



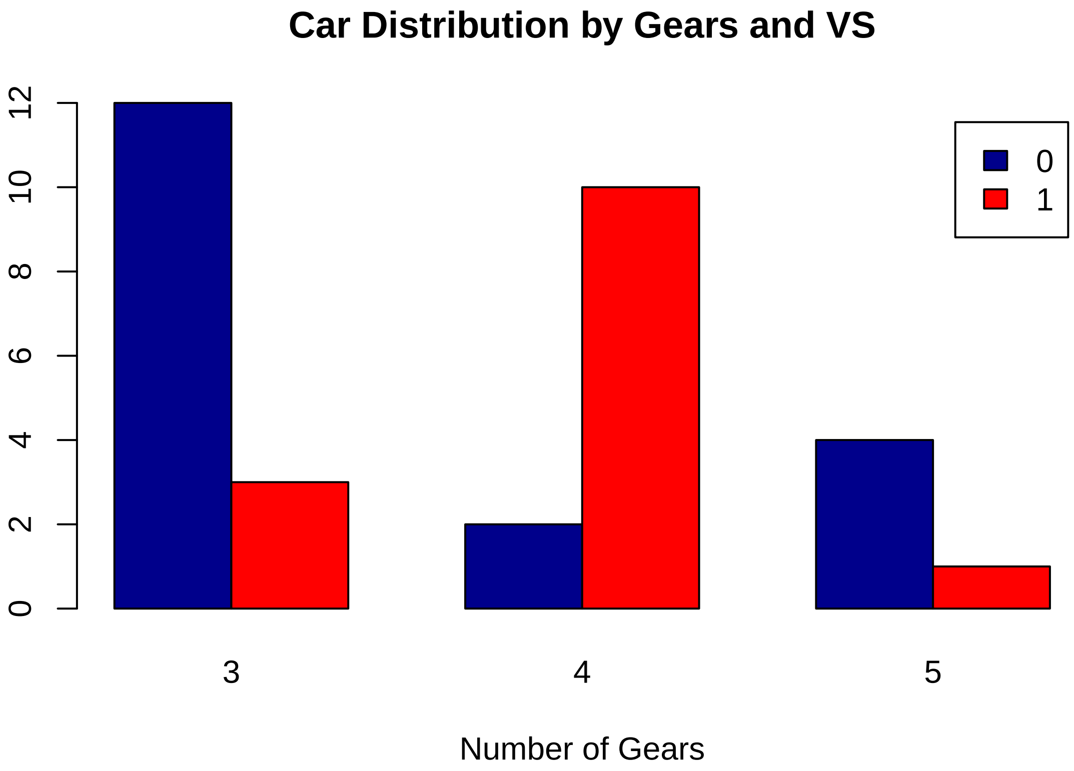
Stacked Bar plot

```
counts <- table(mtcars$vs, mtcars$gear)
barplot(counts, main="Car Distribution by Gears and VS",
  xlab="Number of Gears", col=c("darkblue","red"),
  legend = rownames(counts))
```



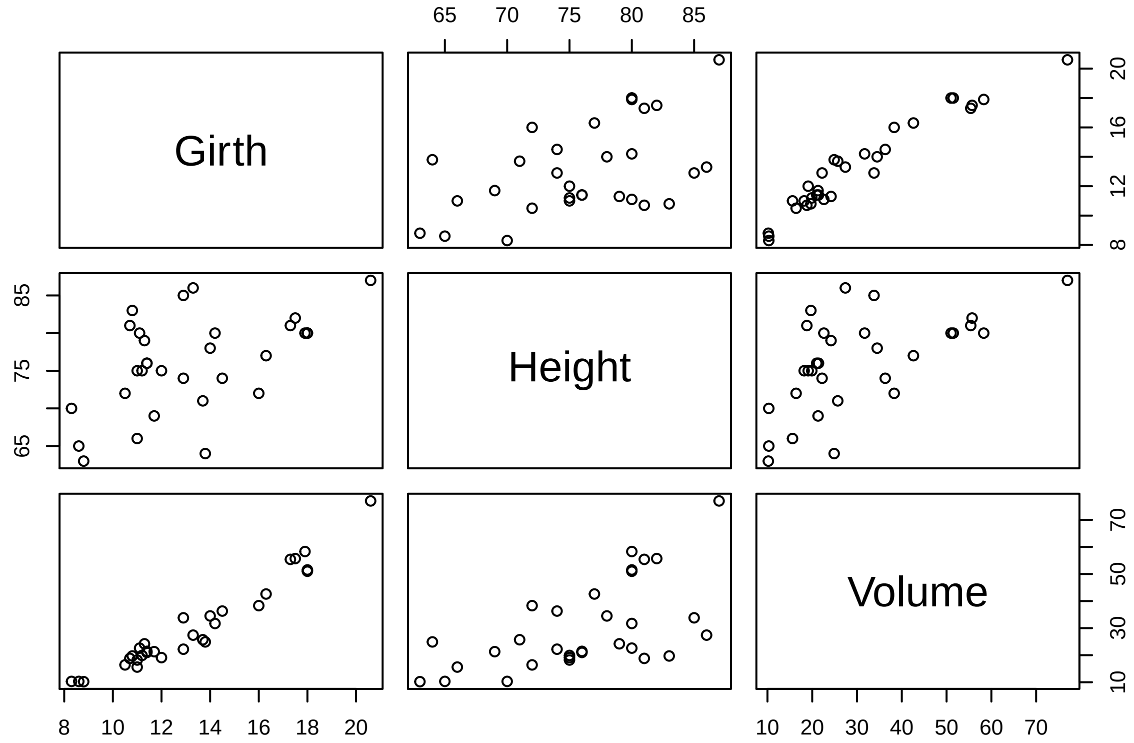
Grouped Bar Plot

```
counts <- table(mtcars$vs, mtcars$gear)
barplot(counts, main="Car Distribution by Gears and VS",
  xlab="Number of Gears", col=c("darkblue","red"),
  legend = rownames(counts), beside=TRUE)
```



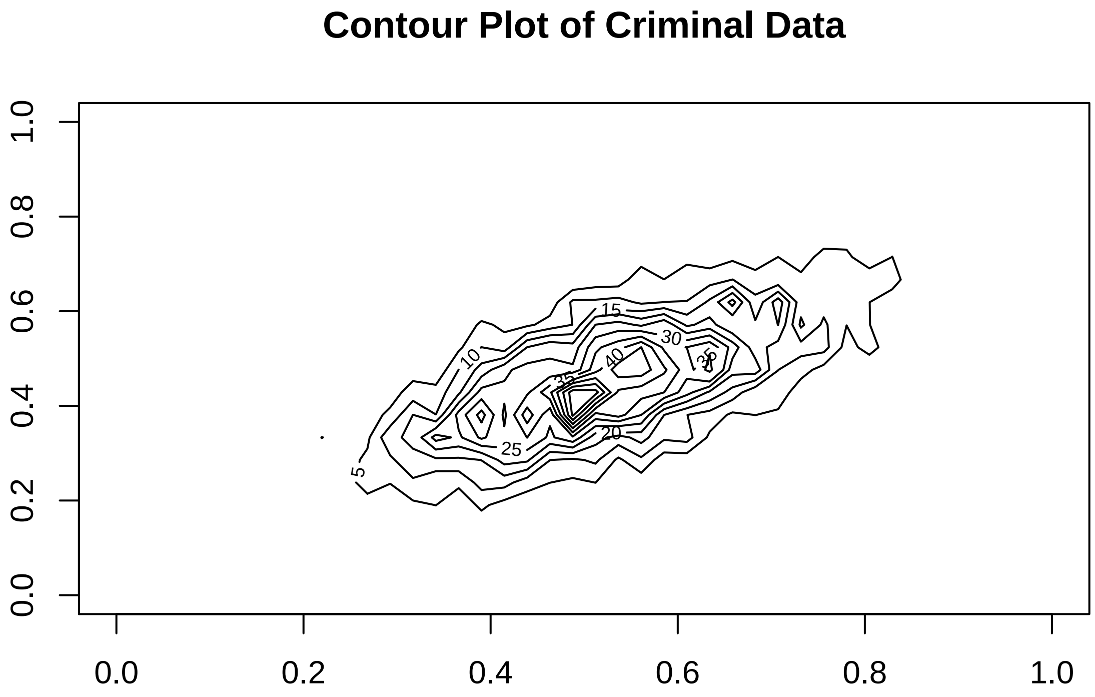
Three-dimensional data: `pairs(x)`

```
pairs(trees)
```



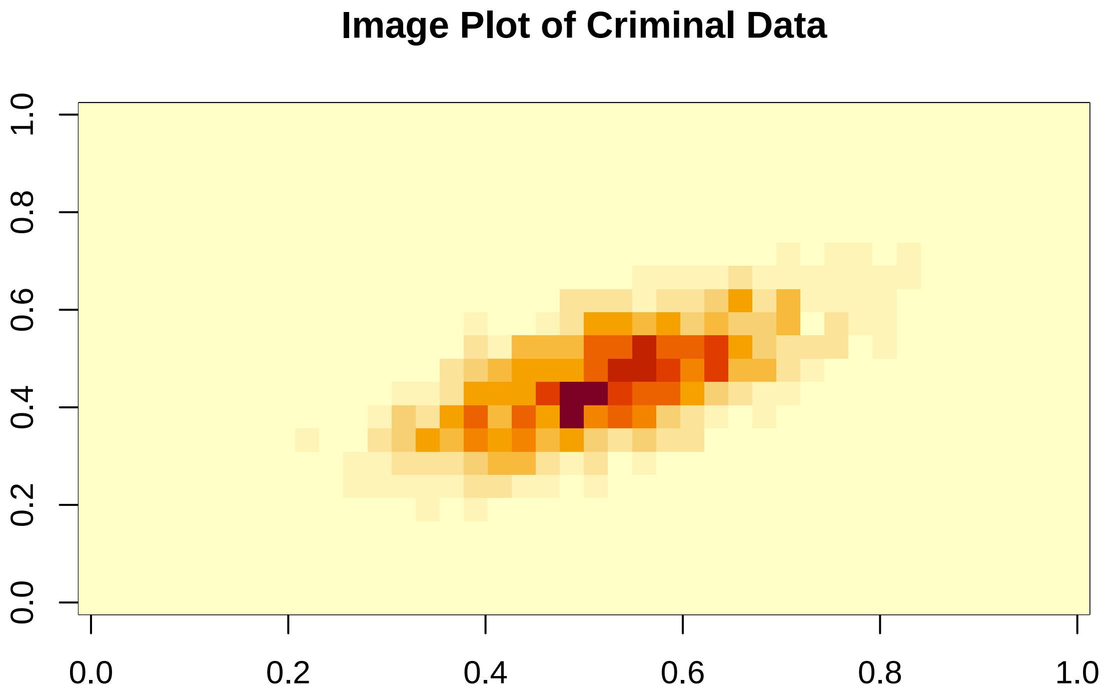
Three dimensional plots: `contour()`

```
contour(crimtab, main="Contour Plot of Criminal Data")
```

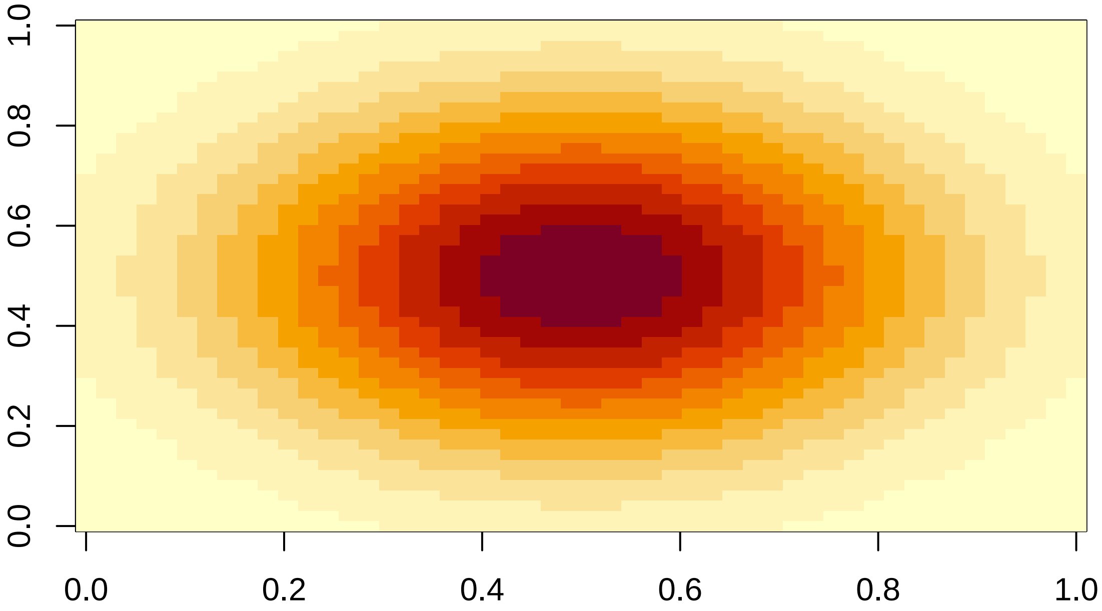


Three dimensional plots: `image()`

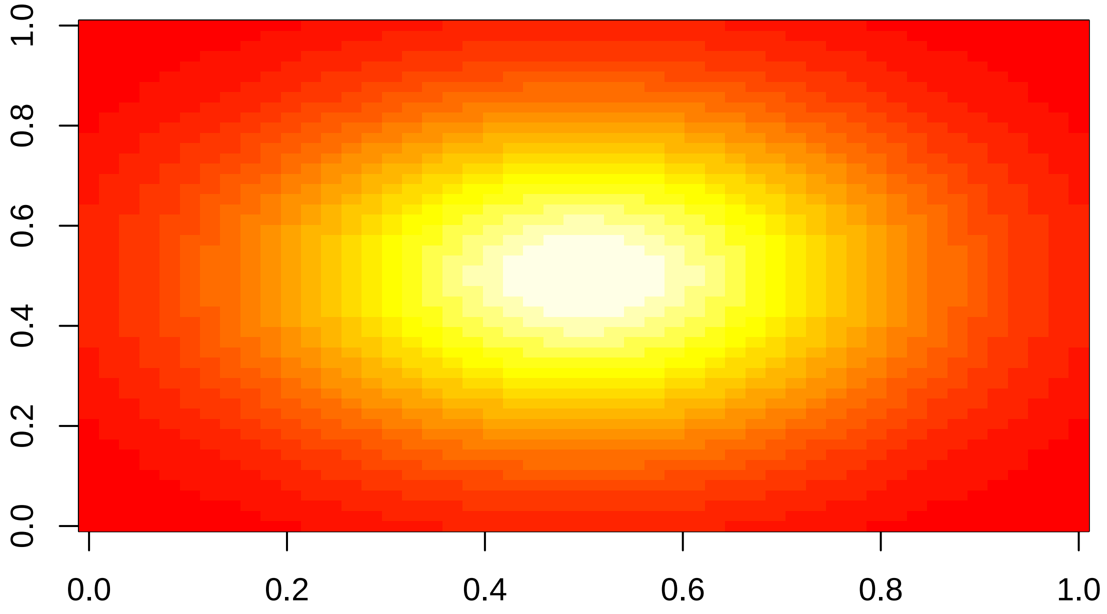
```r
image(crimtab, main="Image Plot of Criminal Data")
```



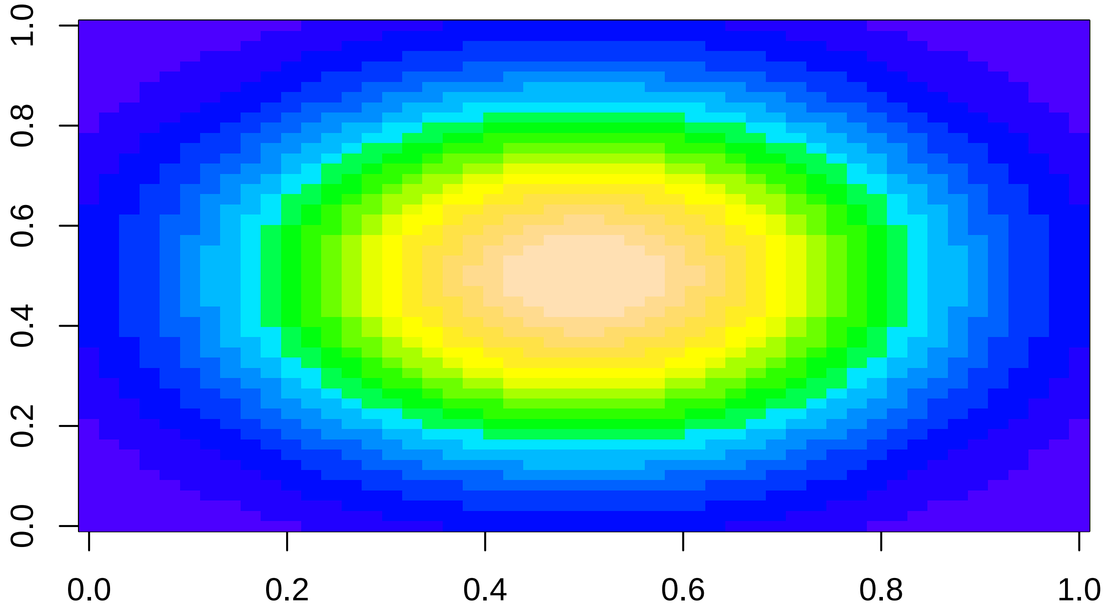
```r
phi <- dnorm(seq(-2,2,length=50))
normal.mat <- phi %o% phi
image(normal.mat)
```



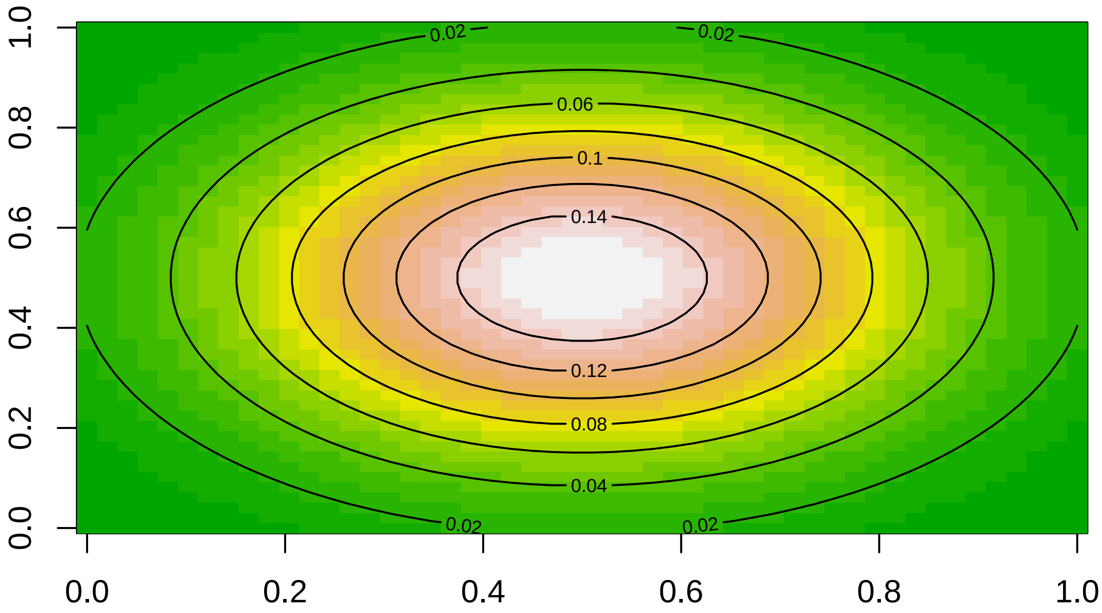
```
image(normal.mat, col=heat.colors(20))
```



```r
image(normal.mat, col=topo.colors(20))
```

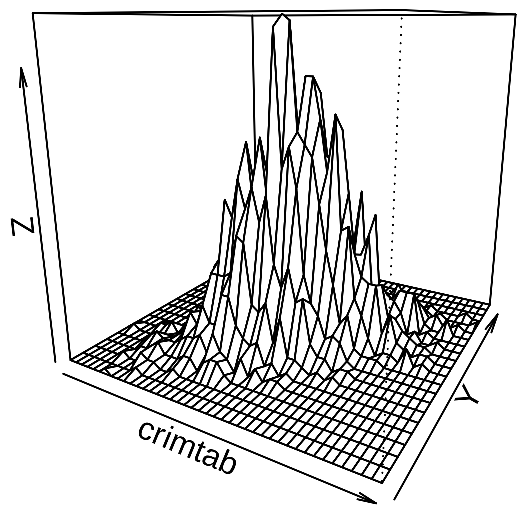


```
image(normal.mat, col=terrain.colors(20))
contour(normal.mat, add = TRUE)
```



Three dimensional plots

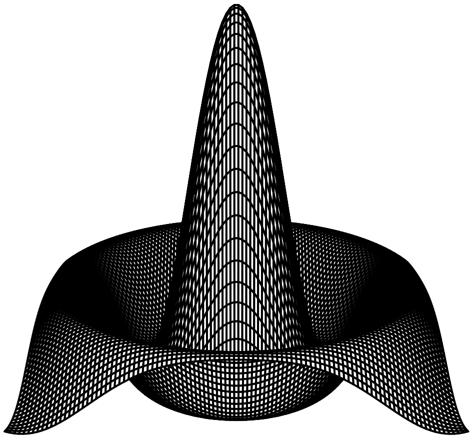
```
persp(crimtab, theta=30, main="Perspective Plot of Criminal Data")
```



Perspective Plot of Criminal Data

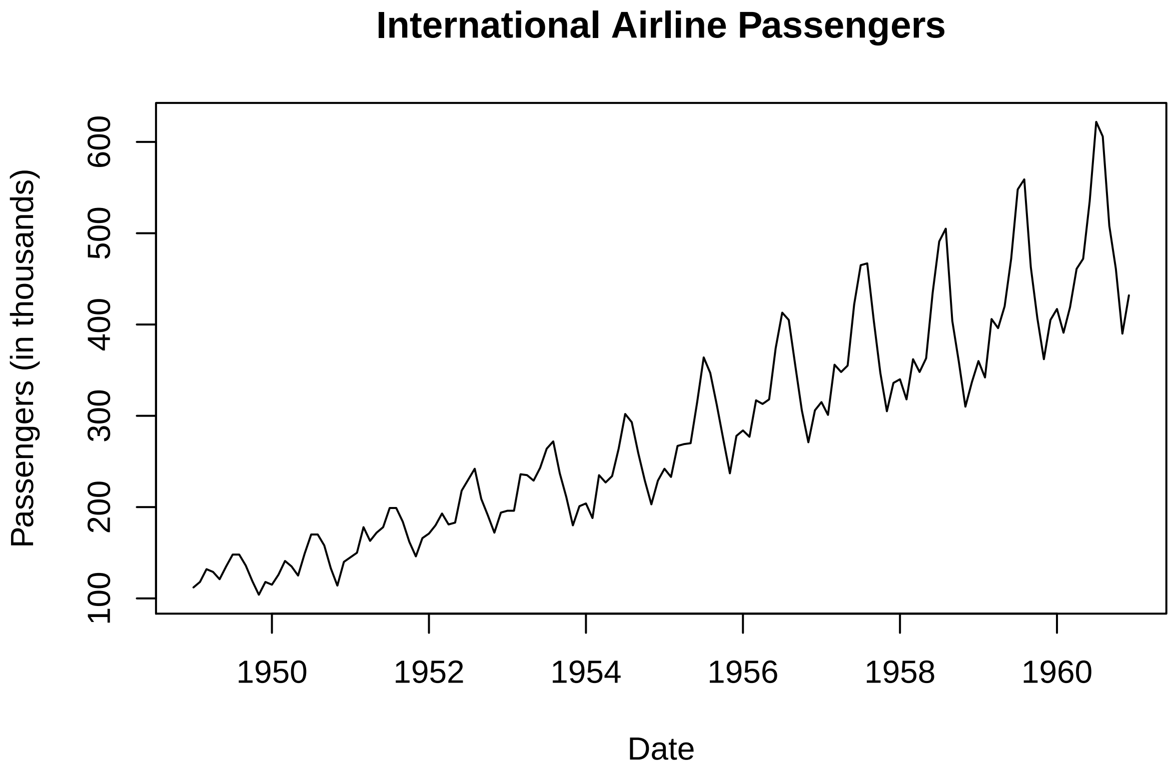
Plot using `persp()` for wire mesh

```
x <- seq(-8, 8, length = 100)
y <- x
f <- function(x, y) sin(sqrt(x ^ 2 + y ^ 2)) / (sqrt (x ^ 2 + y ^ 2))
z <- outer(x, y, f)
persp(x, y, z, xlab = "", ylab = "", zlab = "", axes = F, box = F)
```



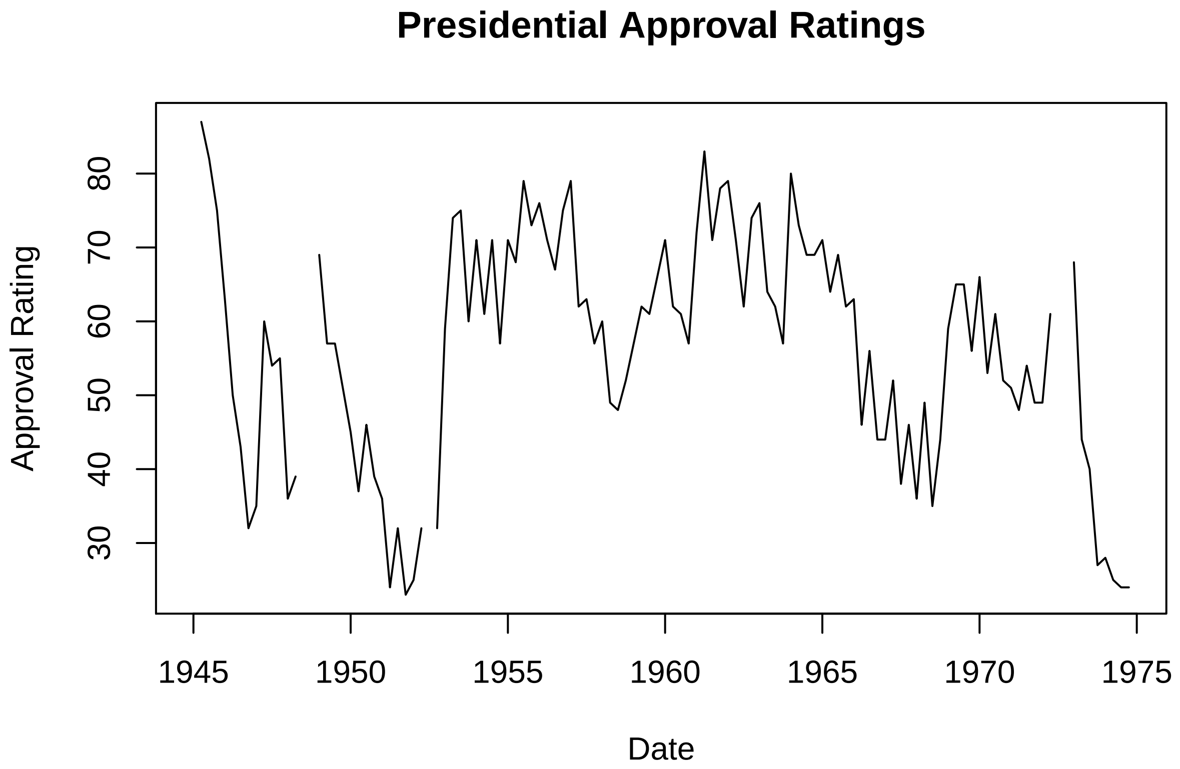
Time series plots

```r
ts.plot(AirPassengers, xlab="Date", ylab="Passengers (in thousands)",
        main="International Airline Passengers")
```



Time series plots

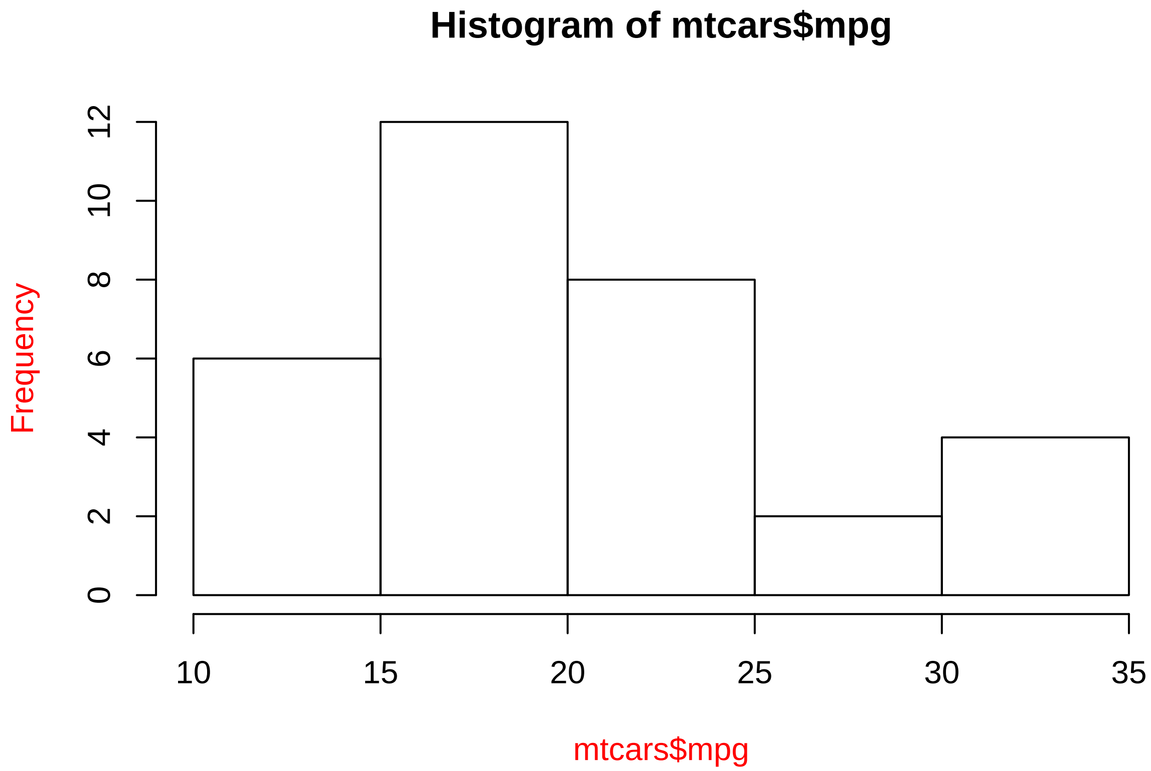
```
ts.plot(presidents, xlab="Date", ylab="Approval Rating",
        main="Presidential Approval Ratings")
```



Custom graphics

- `par()` can be used to set or query graphical parameters (fonts, colors, axes, titles)
- We've already used some of these
- `col`, `col.axis`, `col.lab`, . . . : color specification
- `lty`: line type, e.g. dashed, dotted, solid (default), longdash, . . .
- `lwd`: line width (helpful to increase for presentation plots)
- `pch`: point types
- . . .

```
opar <- par()
par(col.lab="red")
hist(mtcars$mpg)
```

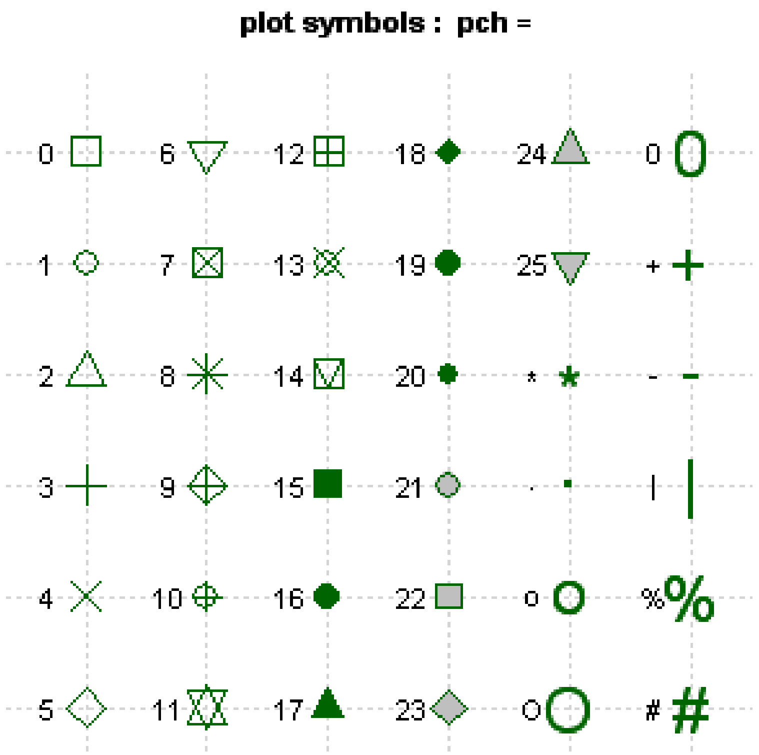


Text and symbol size

- `cex`: number indicating the amount by which plotting text and symbols should be scaled relative to the default.
- 1=default,
- 1.5 is 50% larger,
- 0.5 is 60% smaller,
- ...
- `cex.axis`
- `cex.lab`
- `cex.main`

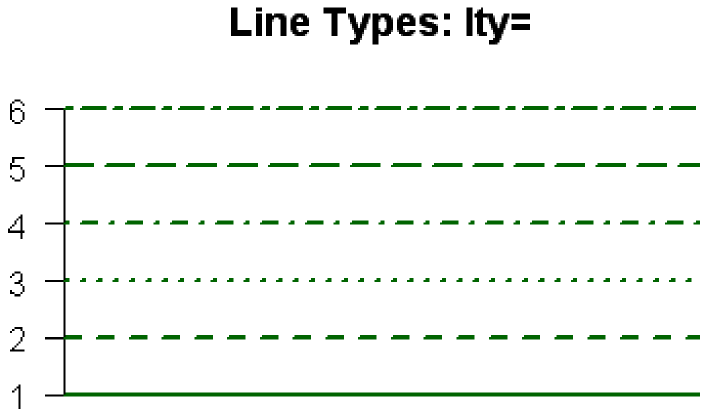
Plotting symbols

- `pch`



Line type

- `lty`

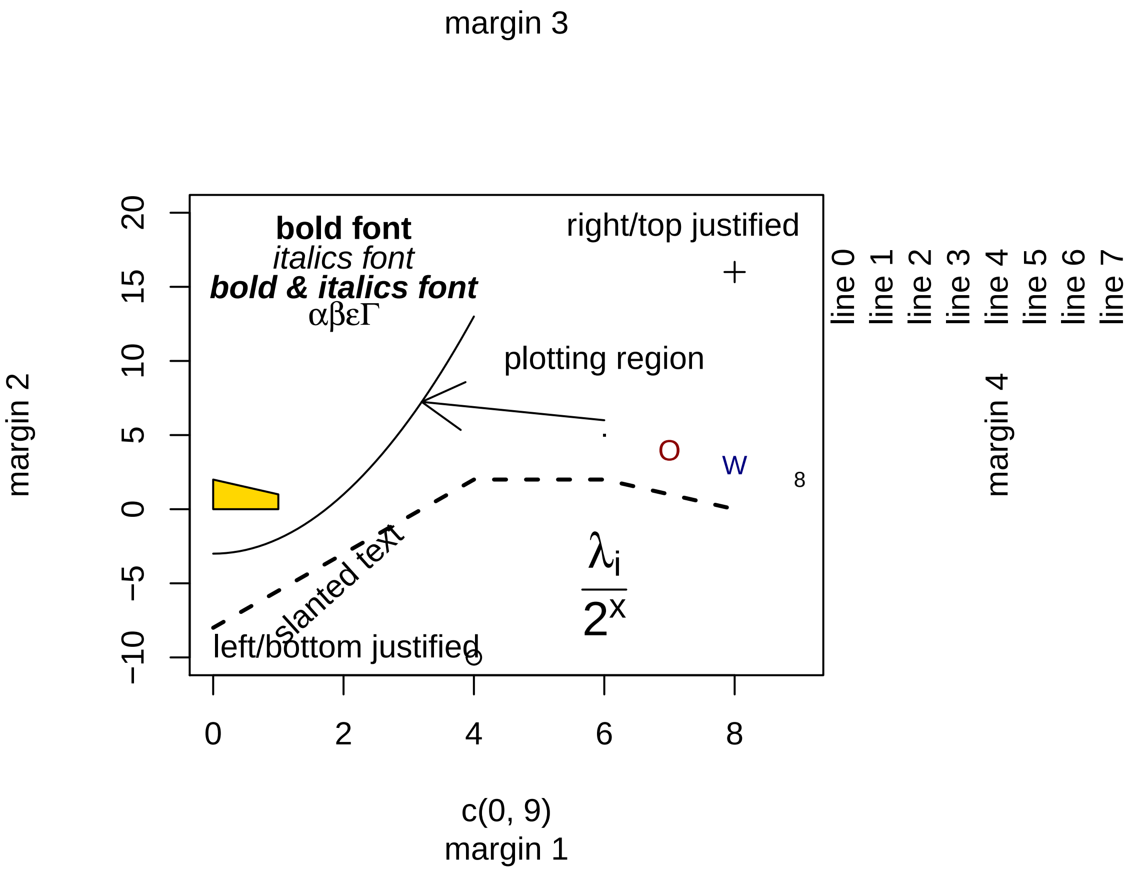


Fonts

- `font`, `font.axis`, `font.lab`, `font.main`
- 1 = plain
- 2 = bold
- 3 = italic
- 4 = bold italic
- 5 = symbol
- `ps`: font point size

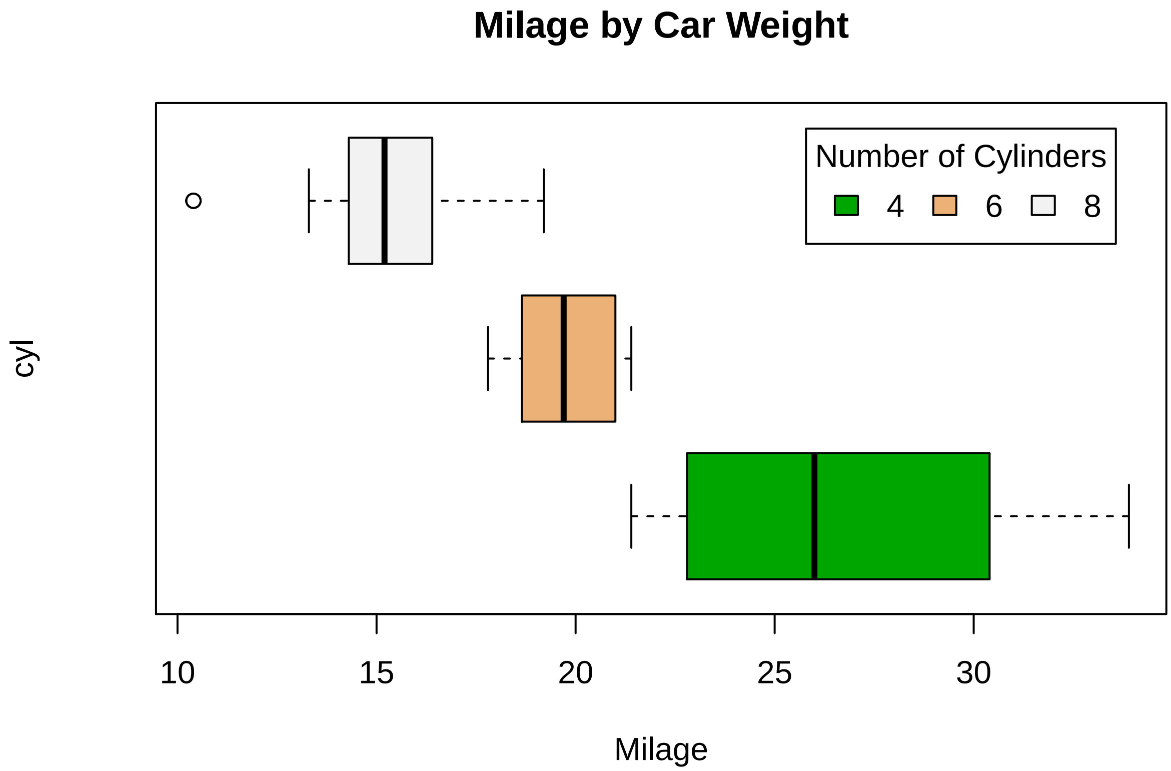
- `family`: font family such as “serif”, “sans”.

Custom plot:



Add legend

```
boxplot(mpg~cyl, data = mtcars, main="Milage by Car Weight",
   yaxt="n", xlab="Milage", horizontal=TRUE,
   col=terrain.colors(3))
legend("topright", inset=.05, title="Number of Cylinders",
   c("4","6","8"), fill=terrain.colors(3), horiz=TRUE)
```

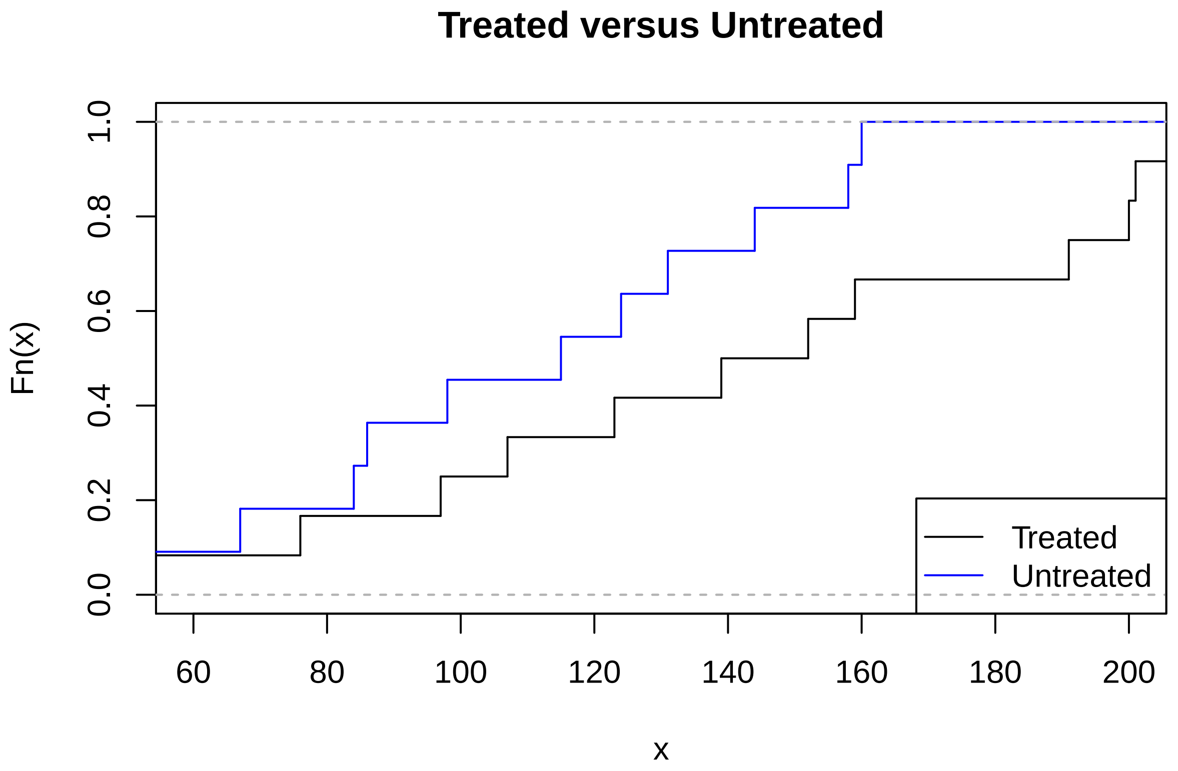


Multiple plots: `Puromycin` dataset

```
x <- Puromycin$rate[Puromycin$state == "treated"]
y <- Puromycin$rate[Puromycin$state == "untreated"]
```

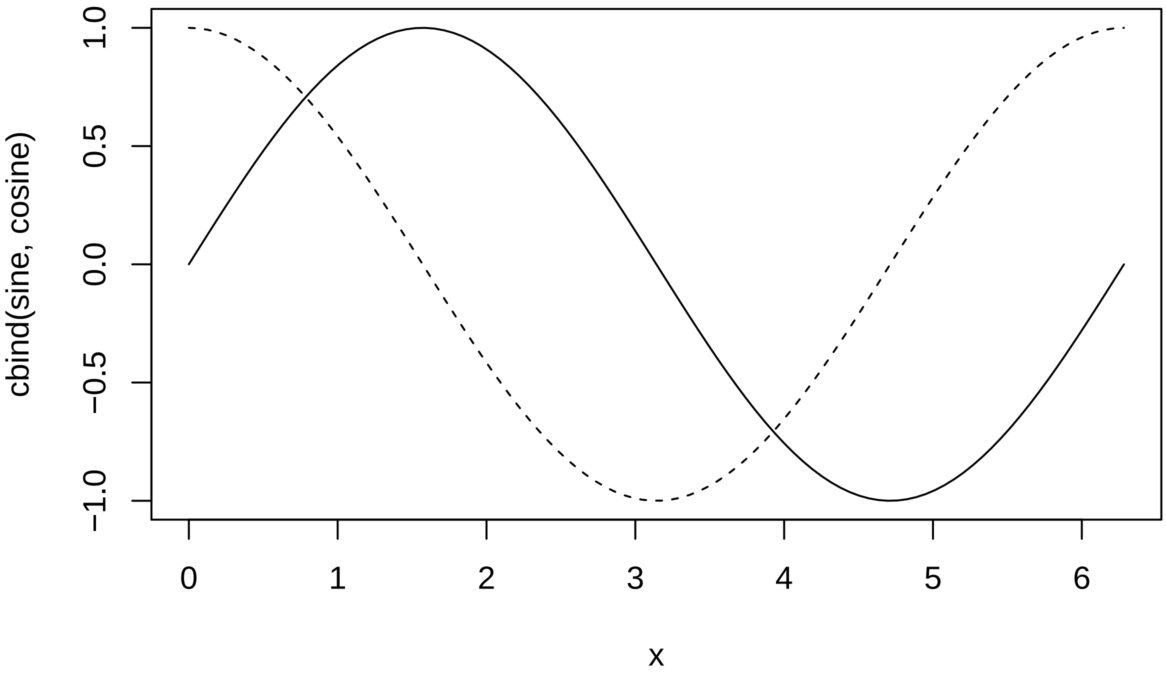
Multiple plots: `Puromycin` dataset

```
plot.ecdf(x, verticals = TRUE, pch = "", xlim = c(60, 200), main="Treated versus Untreated")
lines(ecdf(y), verticals = TRUE, pch = "", xlim = c(60, 200), col="blue")
legend("bottomright", c("Treated", "Untreated"), pch = "", col=c("black", "blue"), lwd = 1)
```



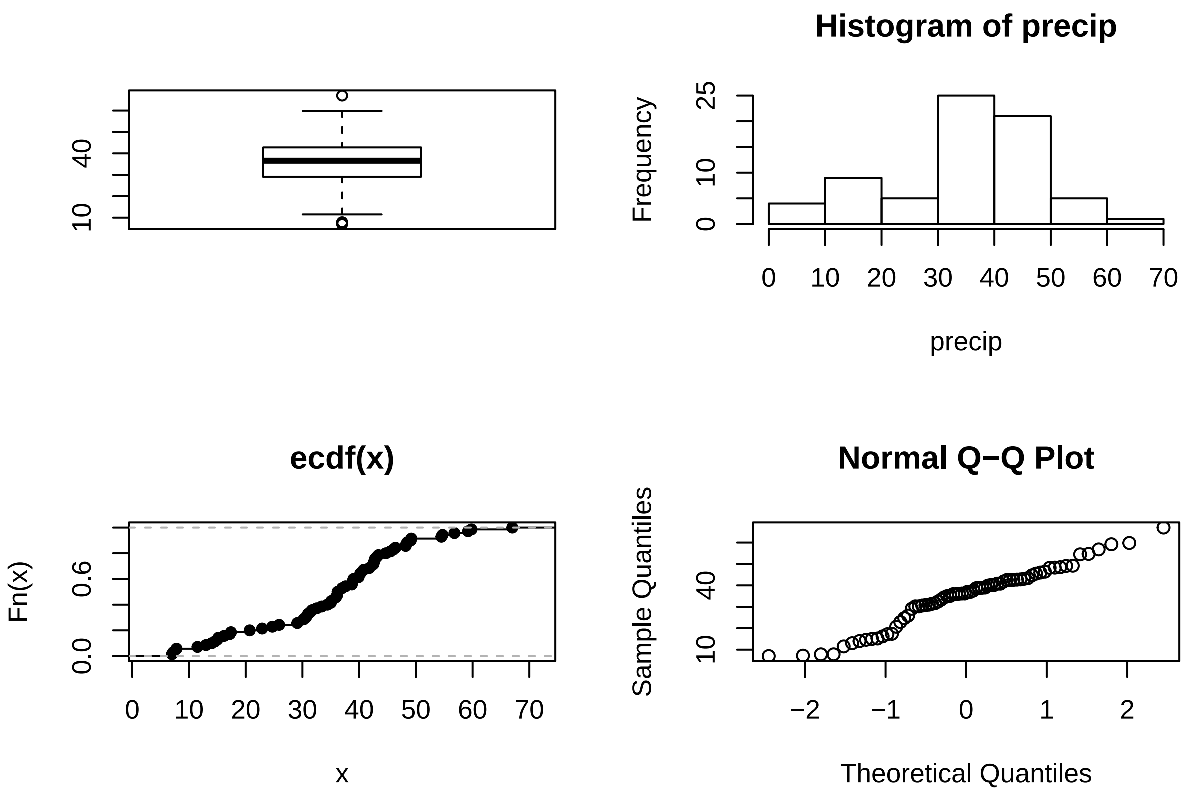
Multiple plots on one set of axes

```
x <- seq(0, 2 * pi, length = 100)
sine <- sin(x)
cosine <- cos(x)
matplot(x, cbind(sine, cosine), col = c(1, 1), type = "l")
```



Multiple frame plots: `mfrow`

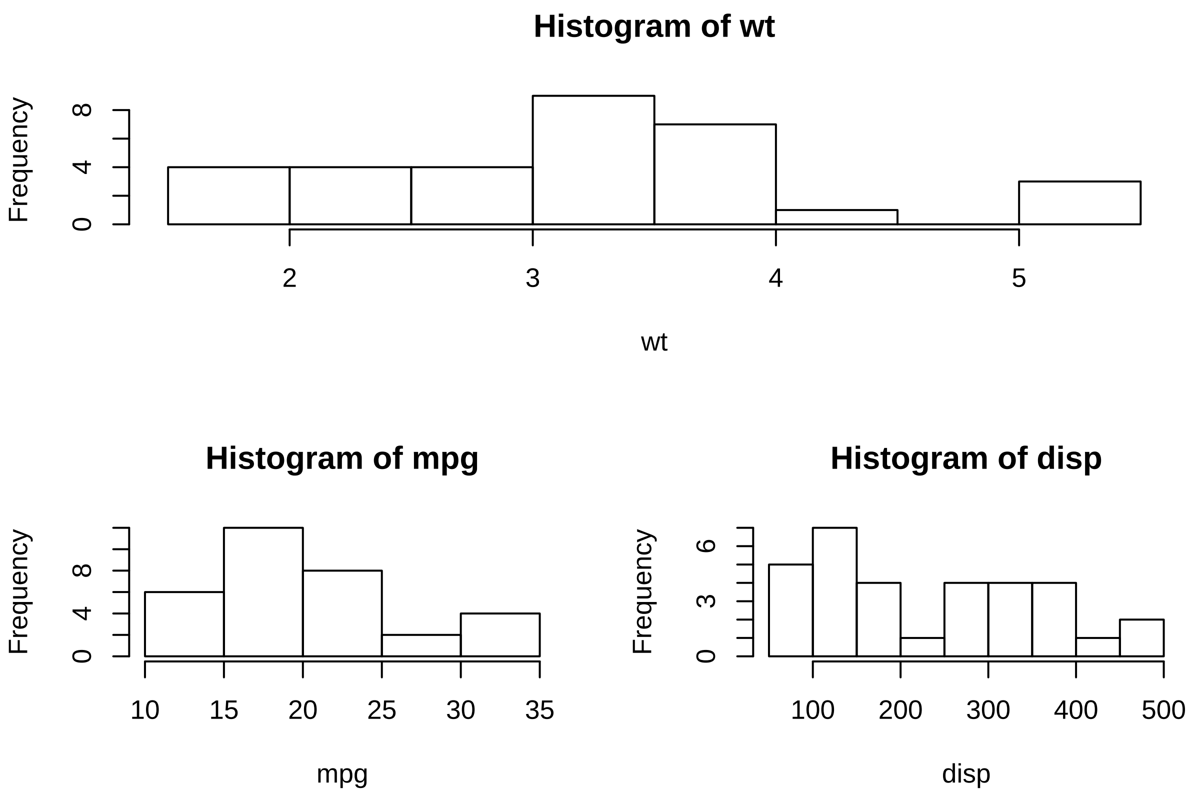
```
par(mfrow = c(2, 2))
boxplot(precip)
hist(precip)
plot.ecdf(precip)
qqnorm(precip)
```



```r
par(mfrow = c(1, 1))
```

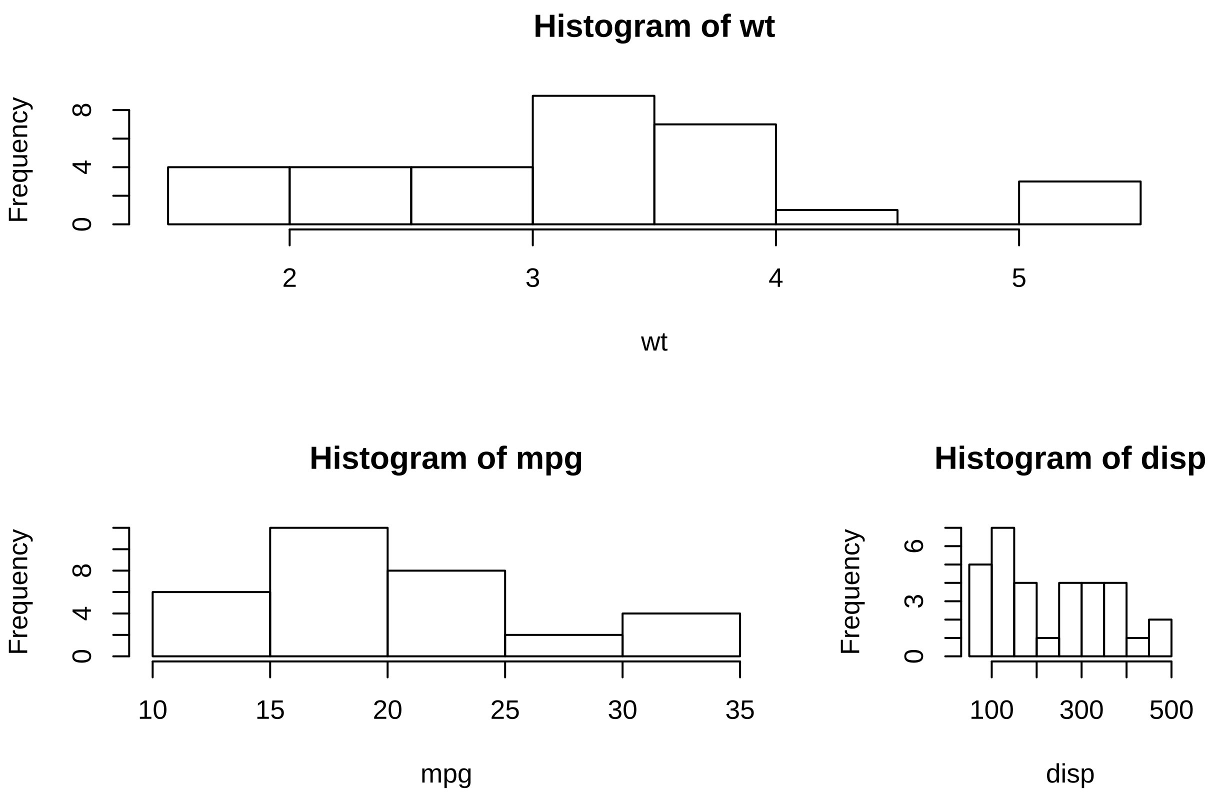
Multiple frame plots: `layout()`

```r
attach(mtcars)
layout(matrix(c(1,1,2,3), 2,2, byrow = TRUE))
hist(wt)
hist(mpg)
hist(disp)
```



```
detach(mtcars)
```

```
attach(mtcars)
layout(matrix(c(1,1,2,3), 2,2, byrow = TRUE), widths = c(2,1), heights = c(1,1))
hist(wt)
hist(mpg)
hist(disp)
```



```
detach(mtcars)
```

Saving a plot to a file

- Begin with functions `postscript()`, `pdf()`, `tiff()`, `jpeg()`, …
- … put all your plotting commands here …
- Finish with `dev.off()`

```
pdf("2cdfs.pdf", width=6, height=4)
plot.ecdf(x, verticals = TRUE, pch = "", xlim = c(60, 200), main="Treated versus Untreated")
lines(ecdf(y), verticals = TRUE, pch = "", xlim = c(60, 200), col="blue")
legend("bottomright", c("Treated", "Untreated"), pch = "", col=c("black", "blue"), lwd = 1)
dev.off()
```

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