NYC DATA SCIENCE
ACADEMY

Data Science with R (Data Analytics)

Data Visualization

Outline

This lecture will explore the application of primary drawing functions and advanced drawing functions in R as well as a general understanding of the methods of data exploration by visualization.

Case study: Analyzing NBA data with graphics

Outline:

❖ Introduction to data visualization

❖ Various visualization functions in R

Data visualization

We'll start by looking at a dataset with four paired variables to explore the most basic of visualizations: the scatterplot.

```
data = read.table('data/anscombe.txt', header=TRUE)
data = data[ ,-1]
str(data)
```

```
'data.frame':   11 obs. of  8 variables:
 $ x1: int  10 8 13 9 11 14 6 4 12 7 ...
 $ x2: int  10 8 13 9 11 14 6 4 12 7 ...
 $ x3: int  10 8 13 9 11 14 6 4 12 7 ...
 $ x4: int  8 8 8 8 8 8 8 19 8 8 ...
 $ y1: num  8.04 6.95 7.58 8.81 8.33 ...
 $ y2: num  9.14 8.14 8.74 8.77 9.26 8.1 6.13 3.1 9.13 7.26 ...
 $ y3: num  7.46 6.77 12.74 7.11 7.81 ...
 $ y4: num  6.58 5.76 7.71 8.84 8.47 7.04 5.25 12.5 5.56 7.91 ...
```

Data visualization

Let's first calculate some statistical indicators. First calculate the mean of each variable, and then calculate the correlation coefficient between each pairing.

```
colMeans(data)
```

```
x1  x2  x3  x4  y1  y2  y3  y4
9.0 9.0 9.0 9.0 7.5 7.5 7.5 7.5
```

```
sapply(1:4, function(x) cor(data[,x], data[,x+4]))
```

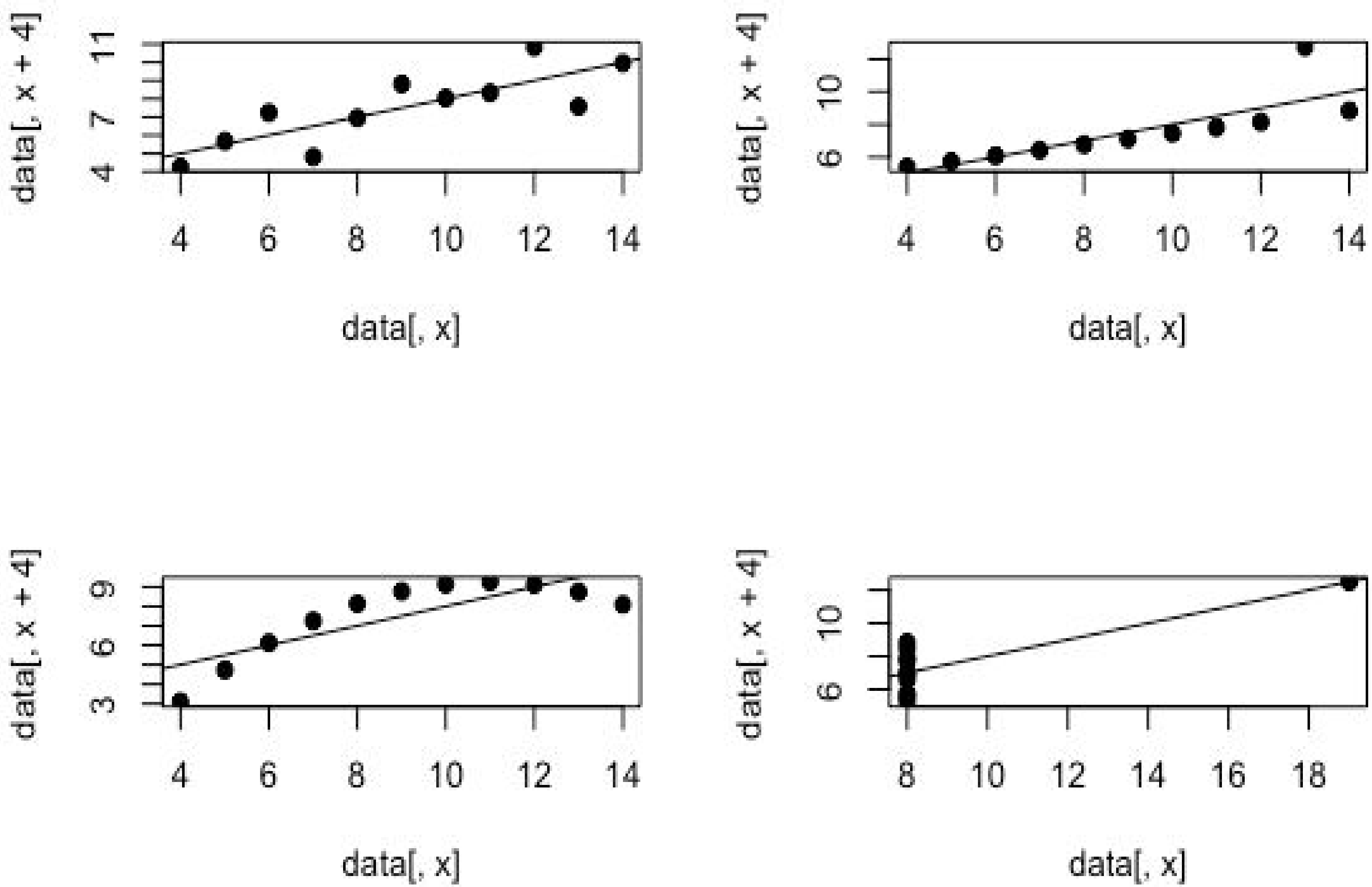
```
[1] 0.8164205 0.8162365 0.8162867 0.8165214
```

Data visualization

Let's now plot the data to see how well our mean and correlation indicators summarize the relationships between and .

```r
oldpar <- par()
par(mfcol=c(2,2)) #prepare frame for 4 plots
for (x in 1:4) {
  plot(data[,x], data[,x+4], pch=19)
  abline(lm(data[,x+4] ~ data[,x]))
}
par(oldpar)
```

Data visualization



Some basic principles

- First, determine the target/goal of your visualizations
  - Exploratory visualization
  - Explanatory visualization
- Understand the characteristics of the data and your audience
  - Which variables are important and interesting?
  - Consider the background and goals of the audience
- Be concise while still conveying enough information

Visualization functions in R

Mapping graphic elements

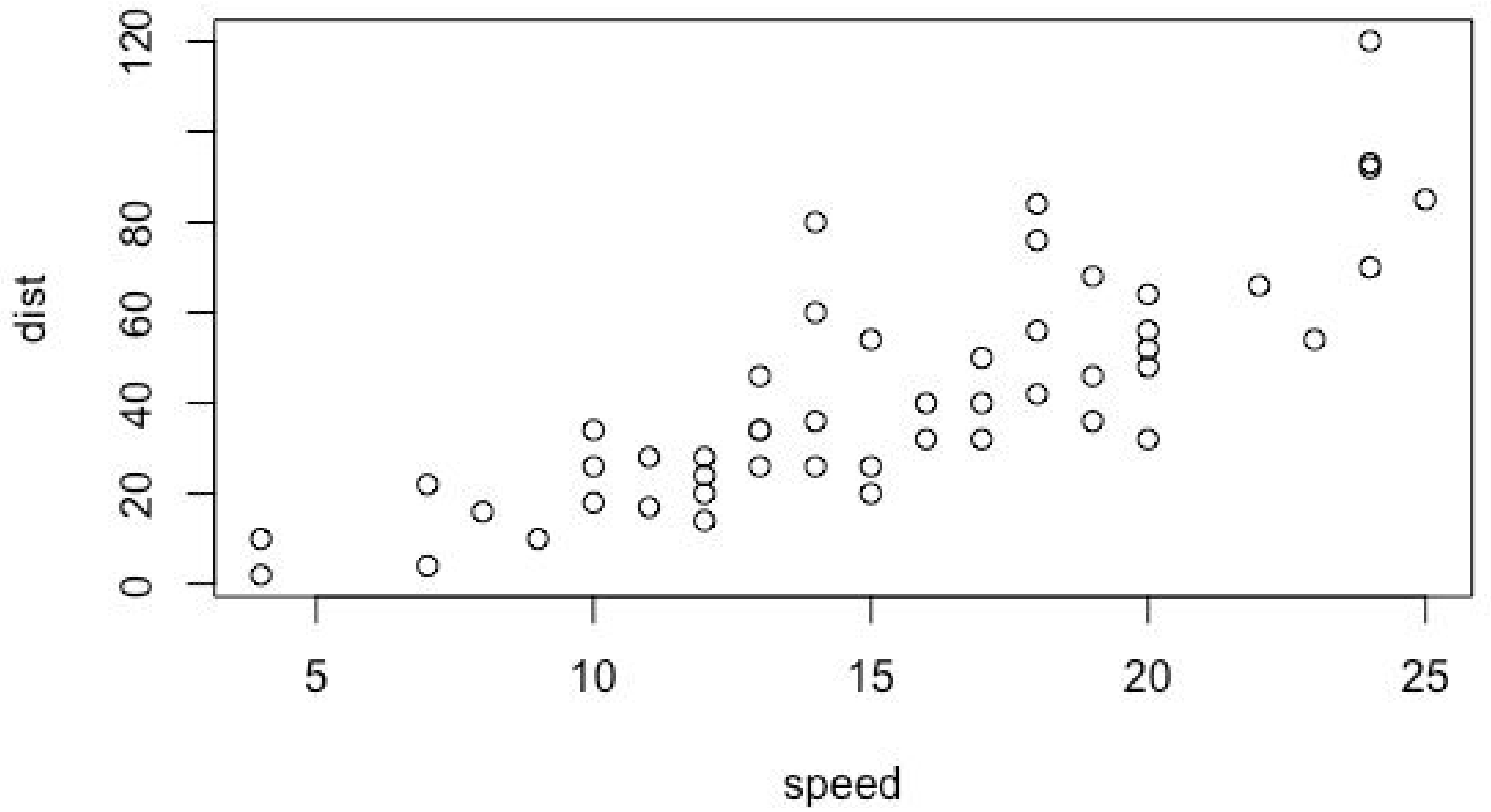
- Coordinate position
- Line
- Size
- Color
- Shape
- Text

Visualization packages in R

- graphics
- lattice
- ggplot2

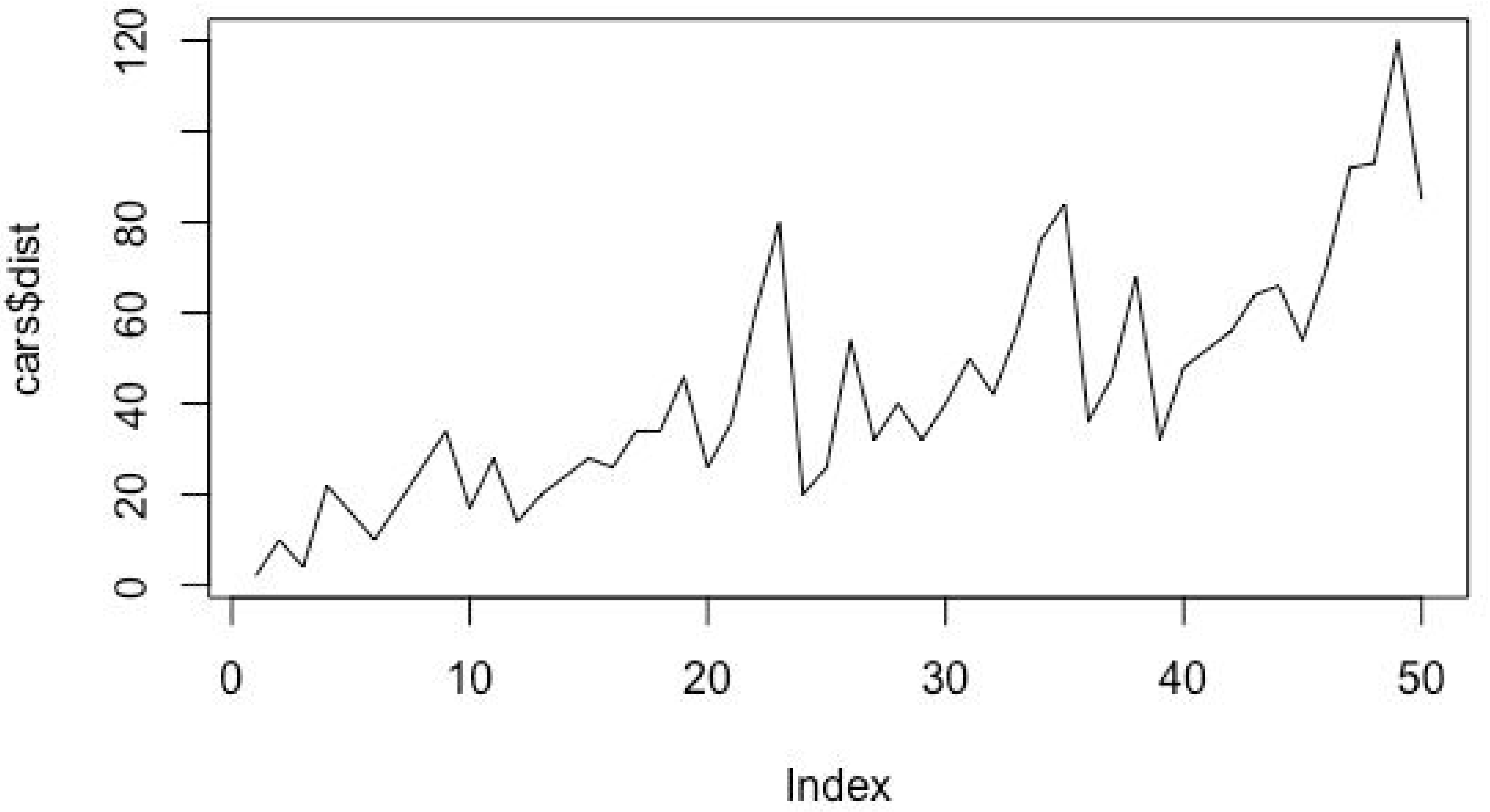
Elementary plotting: Scatterplot

```
plot(dist ~ speed, data=cars)
```



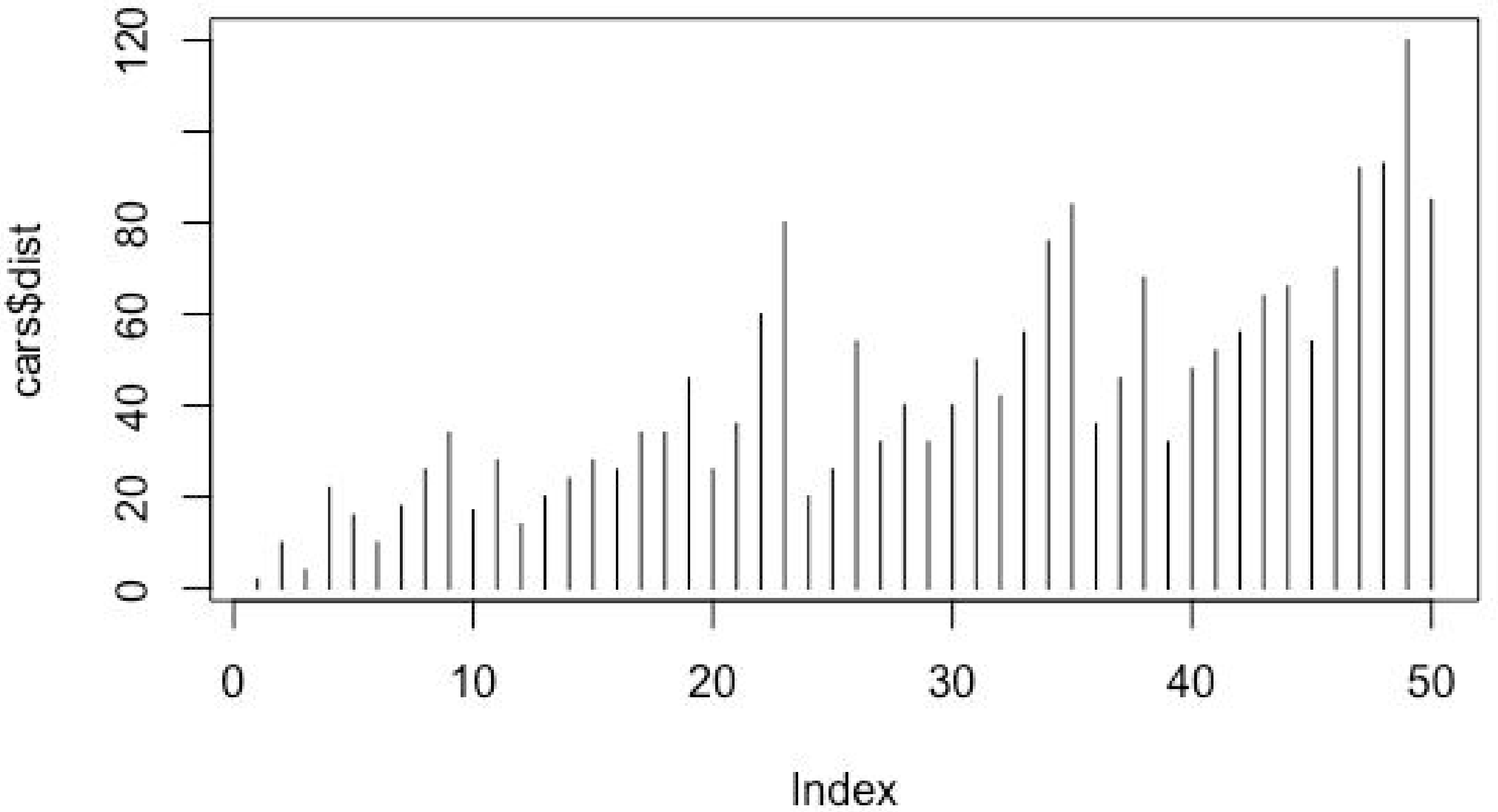
Elementary plotting: Line Graph

```
plot(cars$dist, type='l')
```



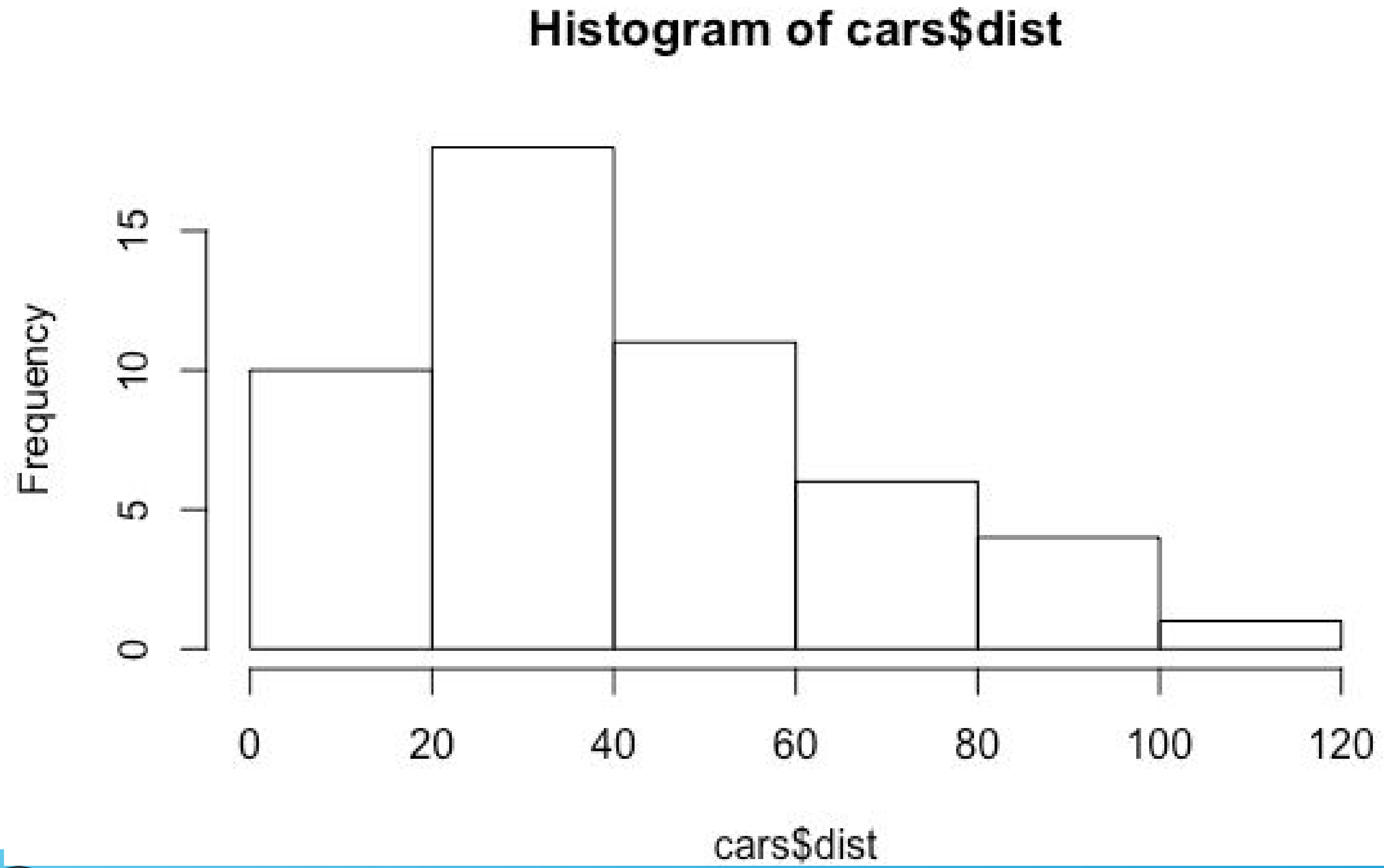
Elementary plotting: Vertical Density Lines

```
plot(cars$dist, type='h') #h for 'histogram-like'
```



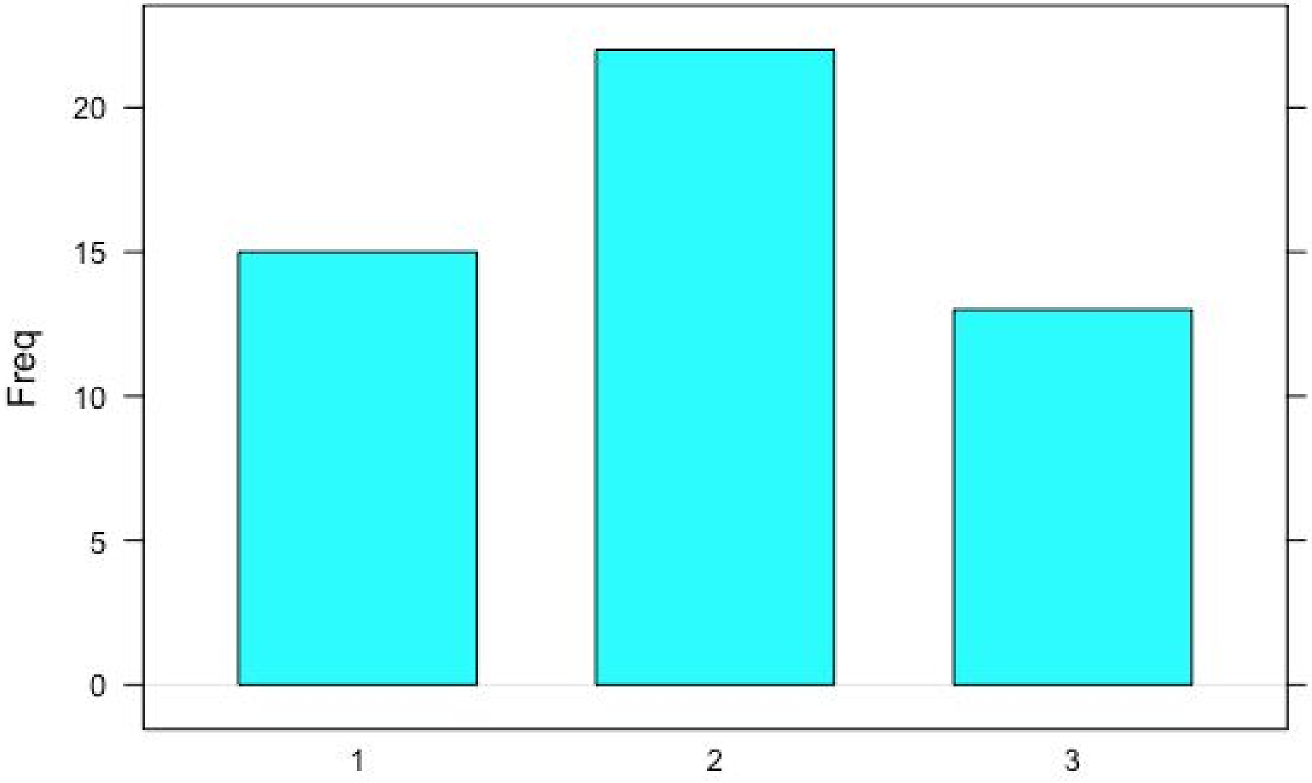
Elementary plotting: Histogram

```
hist(cars$dist)
```



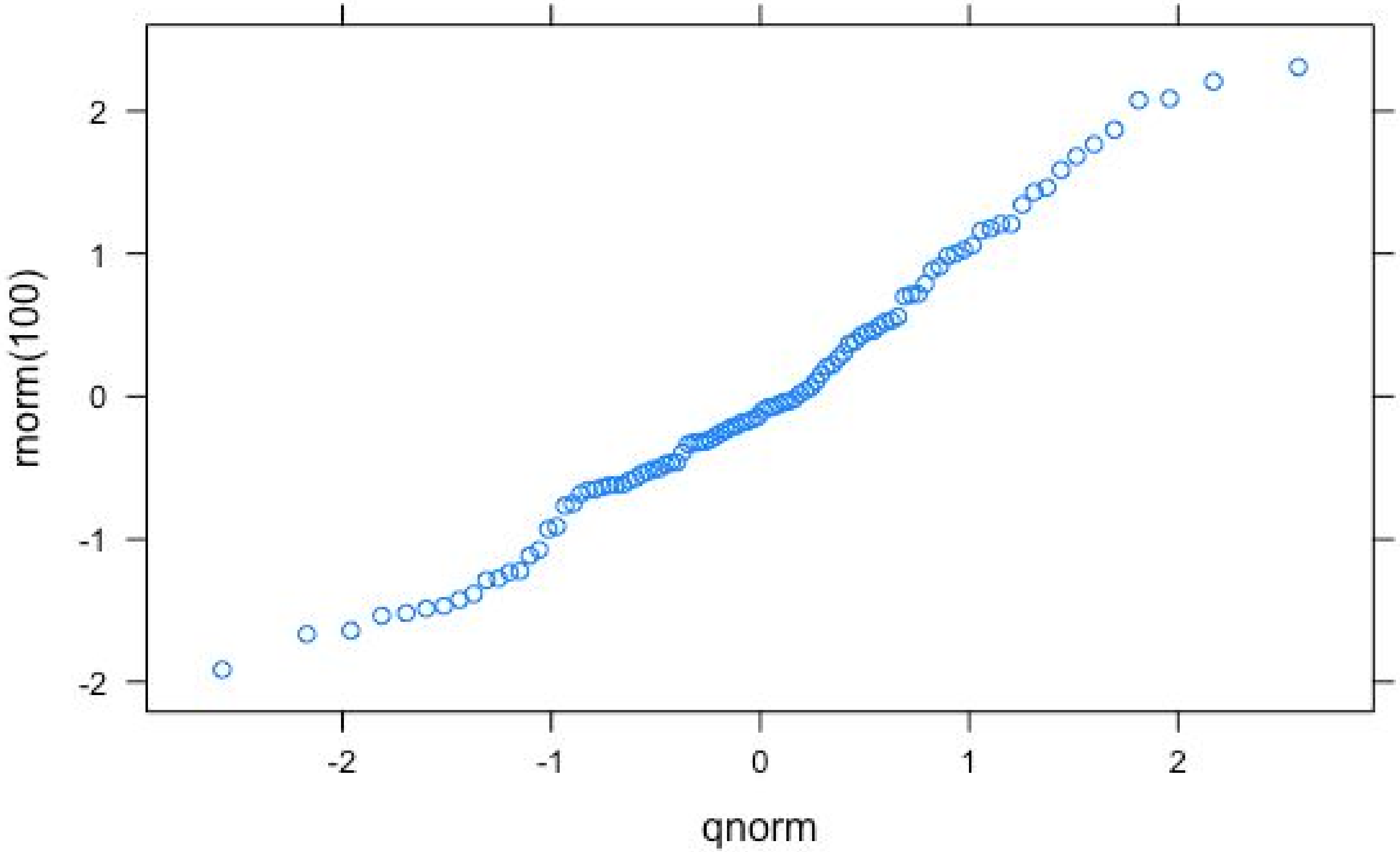
Elementary plotting with *lattice*

```
#install.packages("lattice")
library(lattice)
num <- sample(1:3, size=50, replace=TRUE)
barchart(table(num), horizontal=FALSE)
```



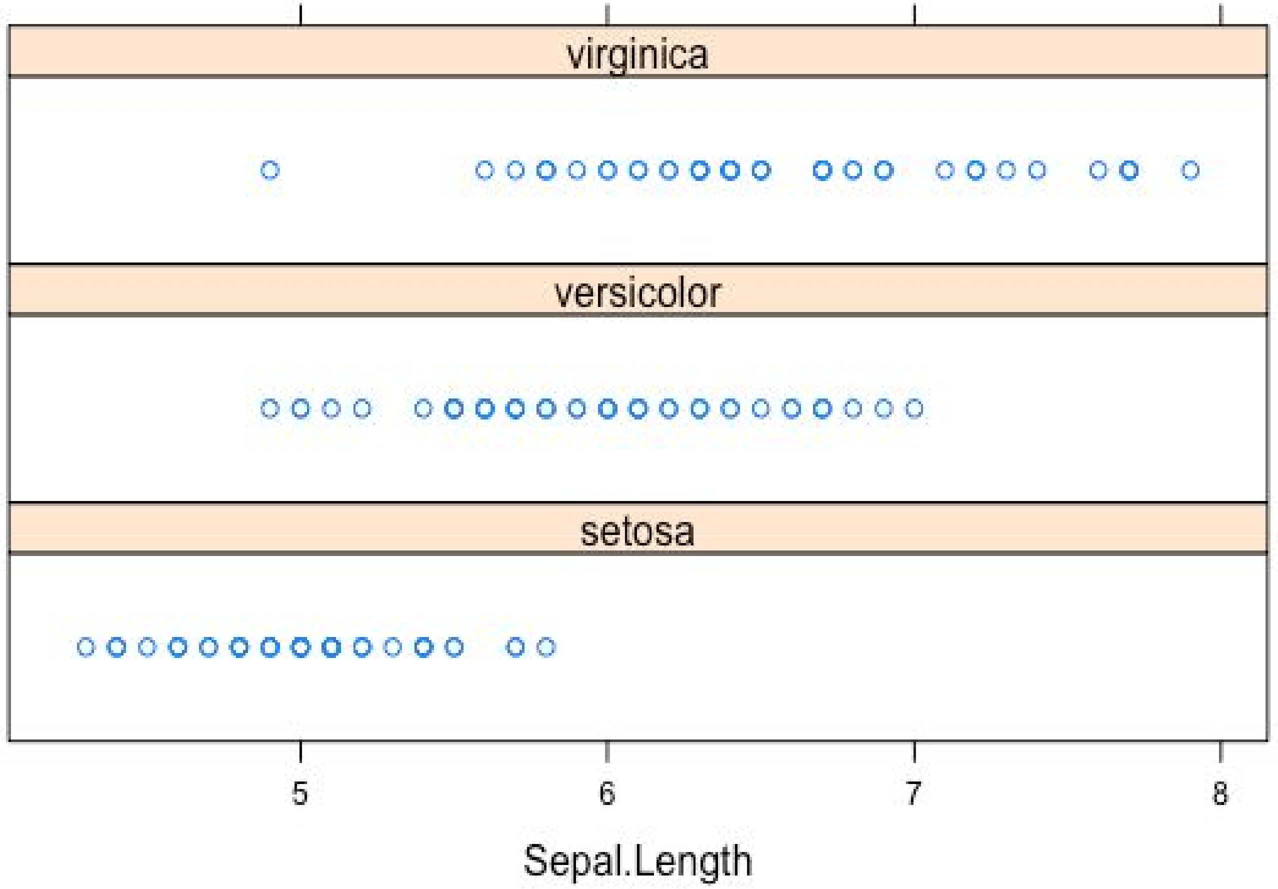
Elementary plotting with *lattice*

```
#test the normality assumption (QQ-plot)
qqmath(rnorm(100))
```



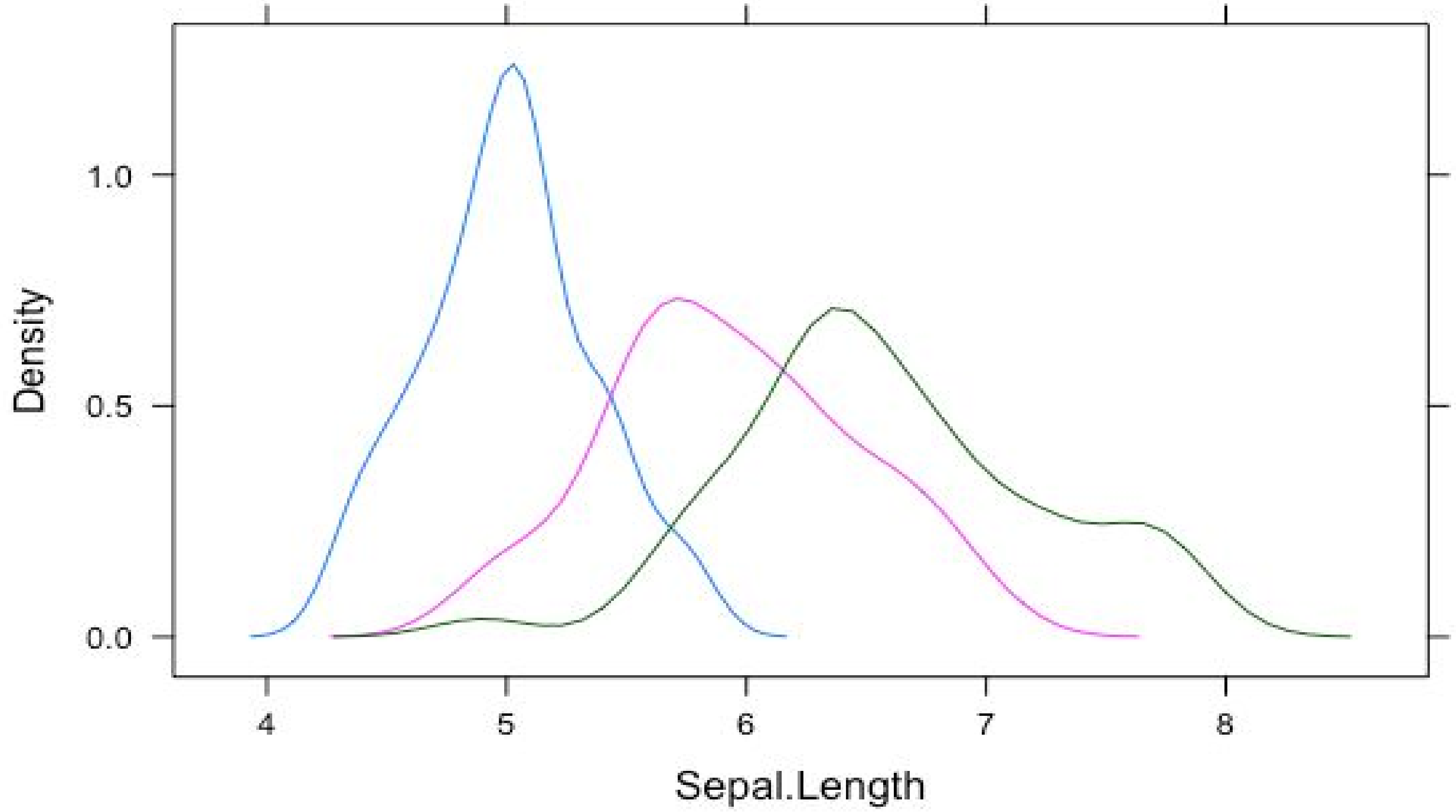
Elementary plotting with *lattice*

```
#note the pipe (|) used to group plot results by a categorical variable
stripplot(~ Sepal.Length | Species, data = iris, layout=c(1,3))
```



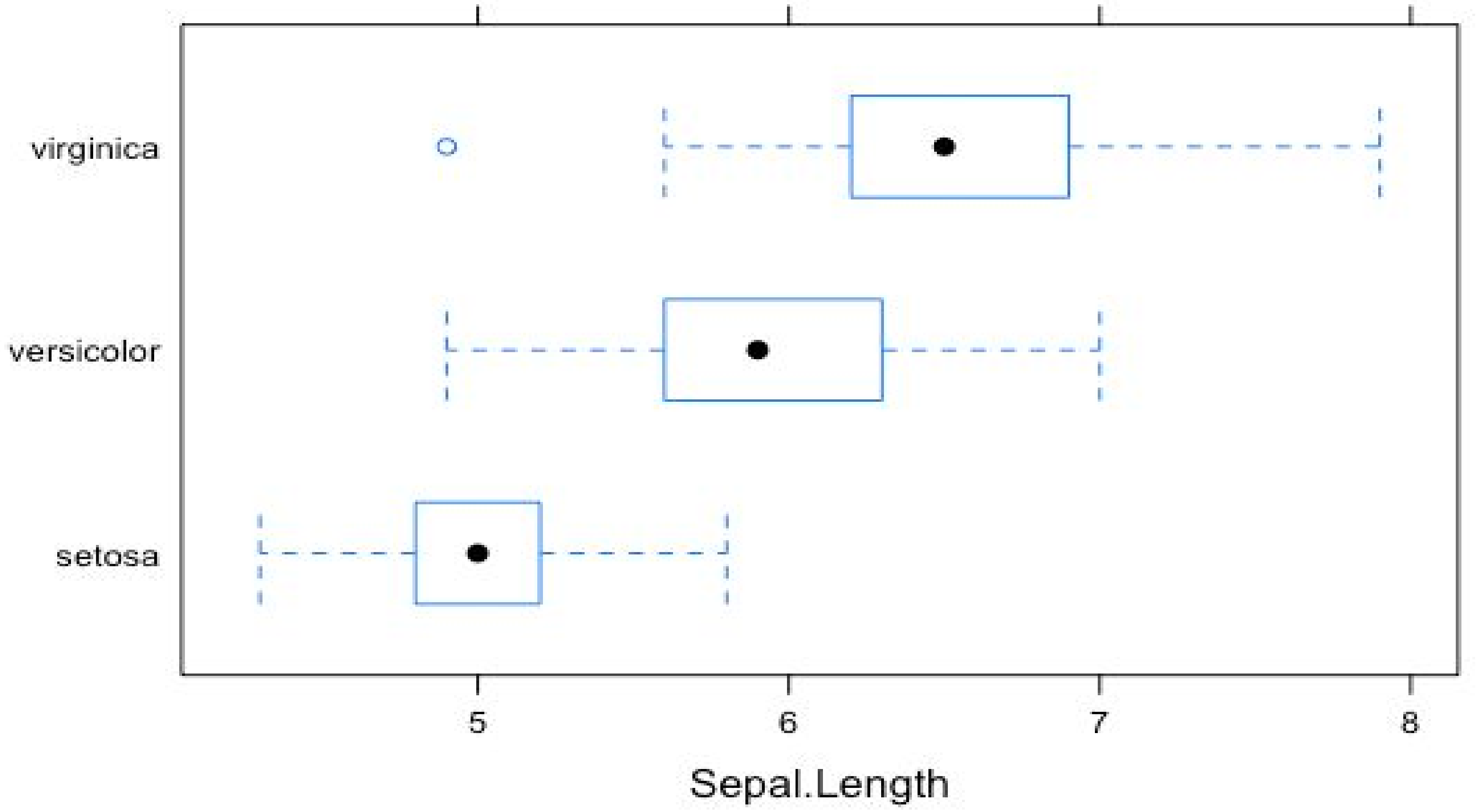
Elementary plotting with *lattice*

```
#specifying groups as an argument differentiates categories by color
densityplot(~ Sepal.Length, groups=Species, data=iris, plot.points=FALSE)
```



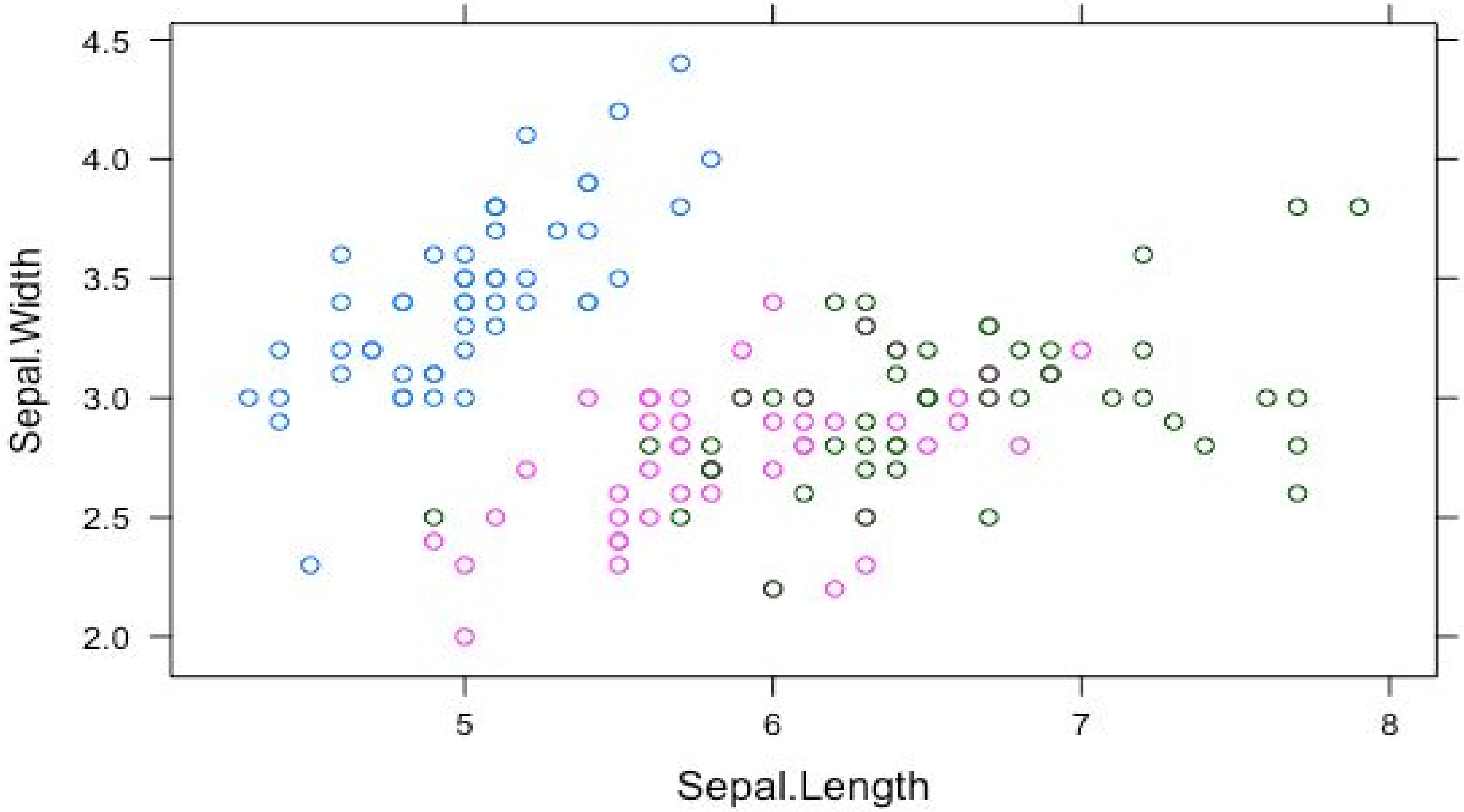
Elementary plotting with *lattice*

```
#boxplots are useful for summarizing data
bwplot(Species ~ Sepal.Length, data=iris)
```



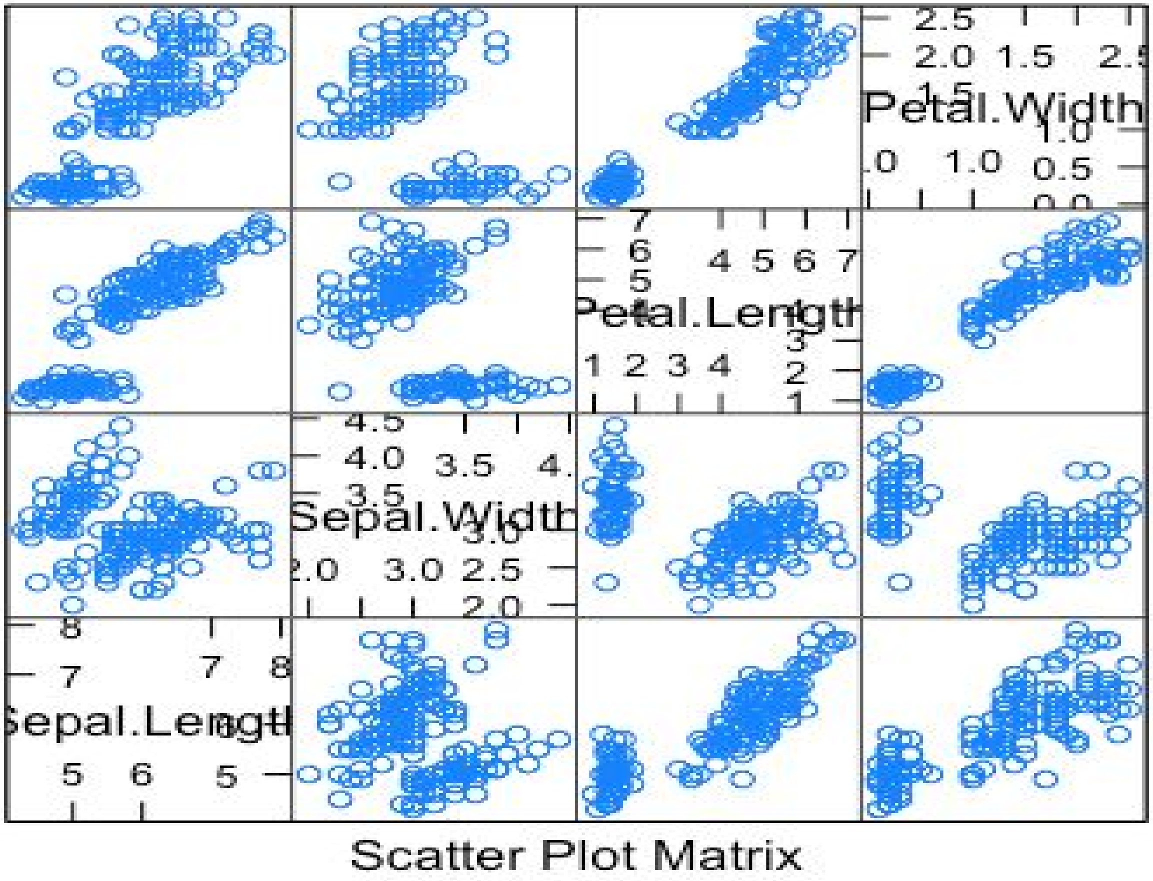
Elementary plotting with *lattice*

```
#scatterplots can also be color coded by category
xyplot(Sepal.Width ~ Sepal.Length, groups=Species, data=iris)
```



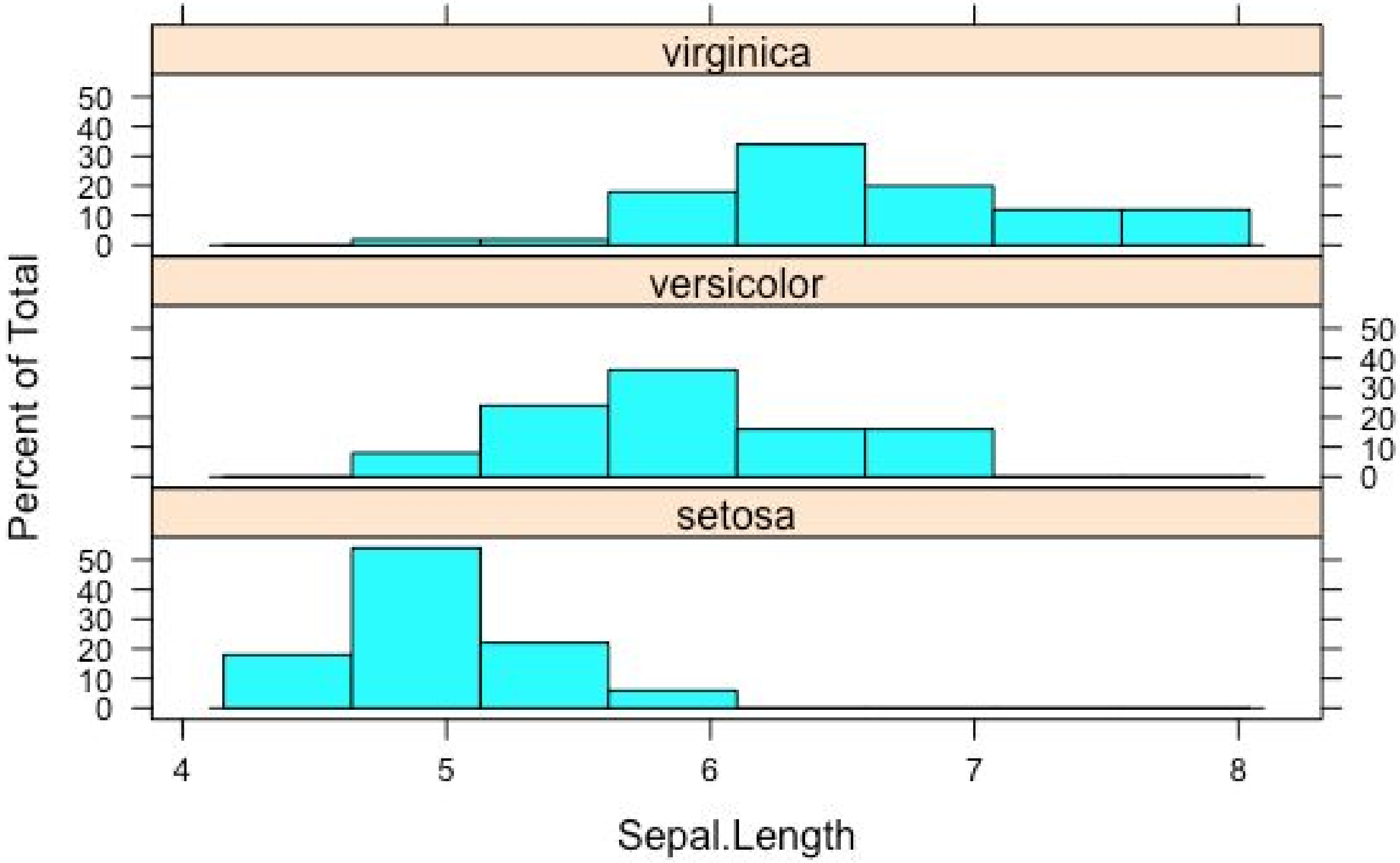
Elementary plotting with *lattice*

```
#examining pairwise relationships:
splom(iris[1:4])
```



Elementary plotting with *lattice*

```
histogram(~ Sepal.Length | Species, data=iris, layout=c(1,3))
```

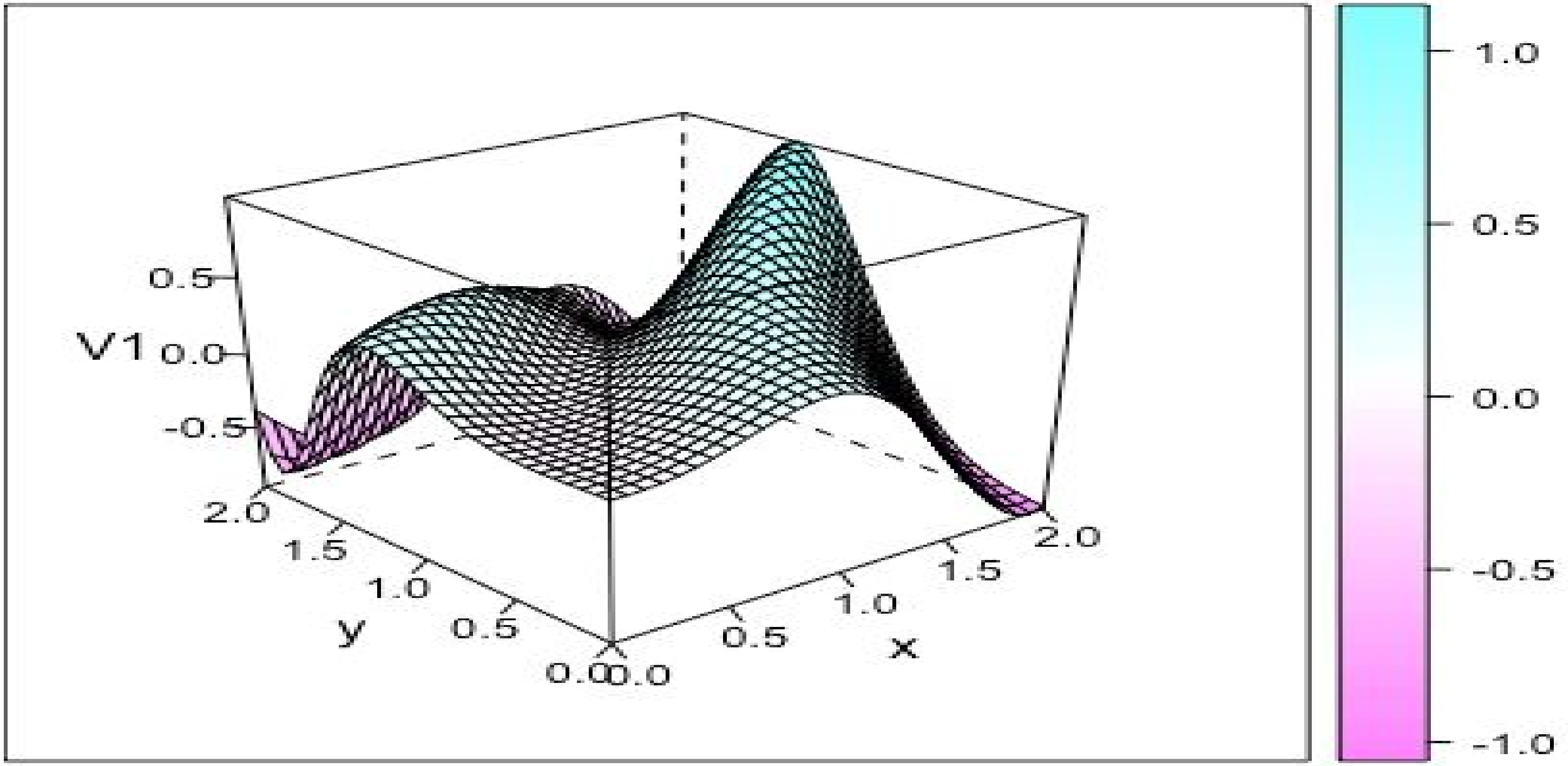


Three-dimensional graphs in *lattice*

```
#install.packages("plyr")
library(plyr)
func3d <- function(x,y) {
          sin(x^2/2 - y^2/4) * cos(2*x - exp(y))
          }
vec1 <- vec2 <- seq(0, 2, length=30)
para <- expand.grid(x=vec1, y=vec2)
result6 <- mdply(.data=para, .fun=func3d)
```

Three-dimensional graphs in *lattice*

```
library(lattice)
wireframe(V1 ~ x * y, data=result6, scales=list(arrows=FALSE),
          drape=TRUE, colorkey=TRUE)
```


V1
0.5
0.0
-0.5
y
2.0
1.5
1.0
0.5
0.0
x
0.0
0.5
1.0
1.5
2.0
1.0
0.5
0.0
-0.5
-1.0