CS 660: Mathematical Foundations for Analytics

Dr. Francis Parisi

Pace University

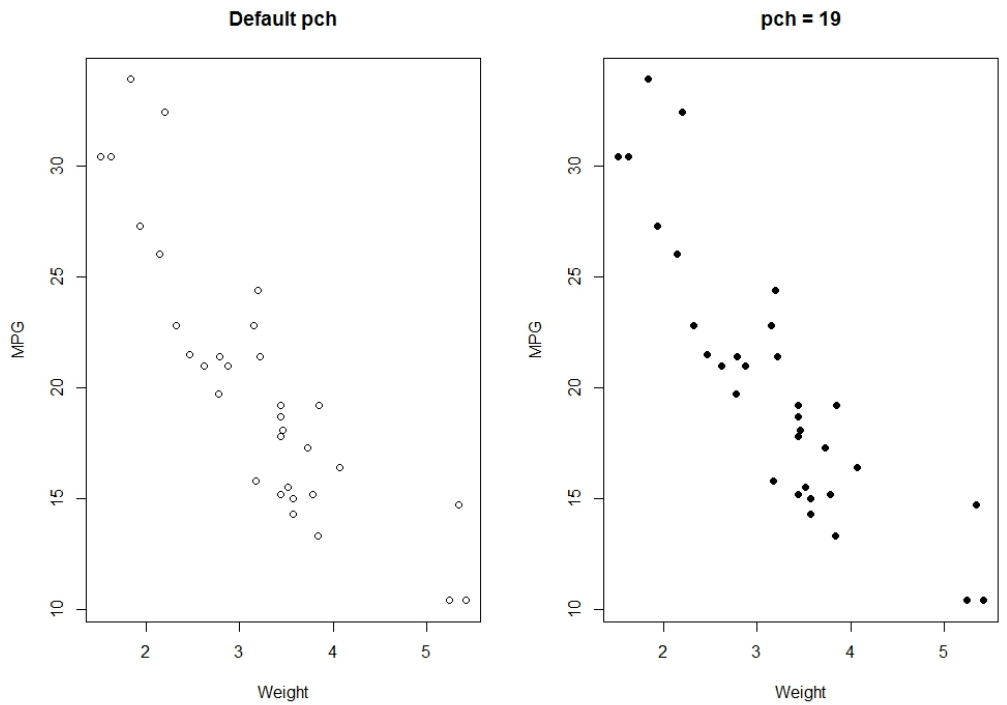
Spring 2017

Basic Graphics

- **Scatterplots** Simple plotting of two variables

```
plot(mtcars$mpg ~ mtcars$wt,
     xlab = "Weight", ylab = "MPG")
plot(mtcars$mpg ~ mtcars$wt,
     xlab = "Weight", ylab = "MPG", pch = 19)
```

- `pch` allows you to specify the `p`lotting `ch`aracter

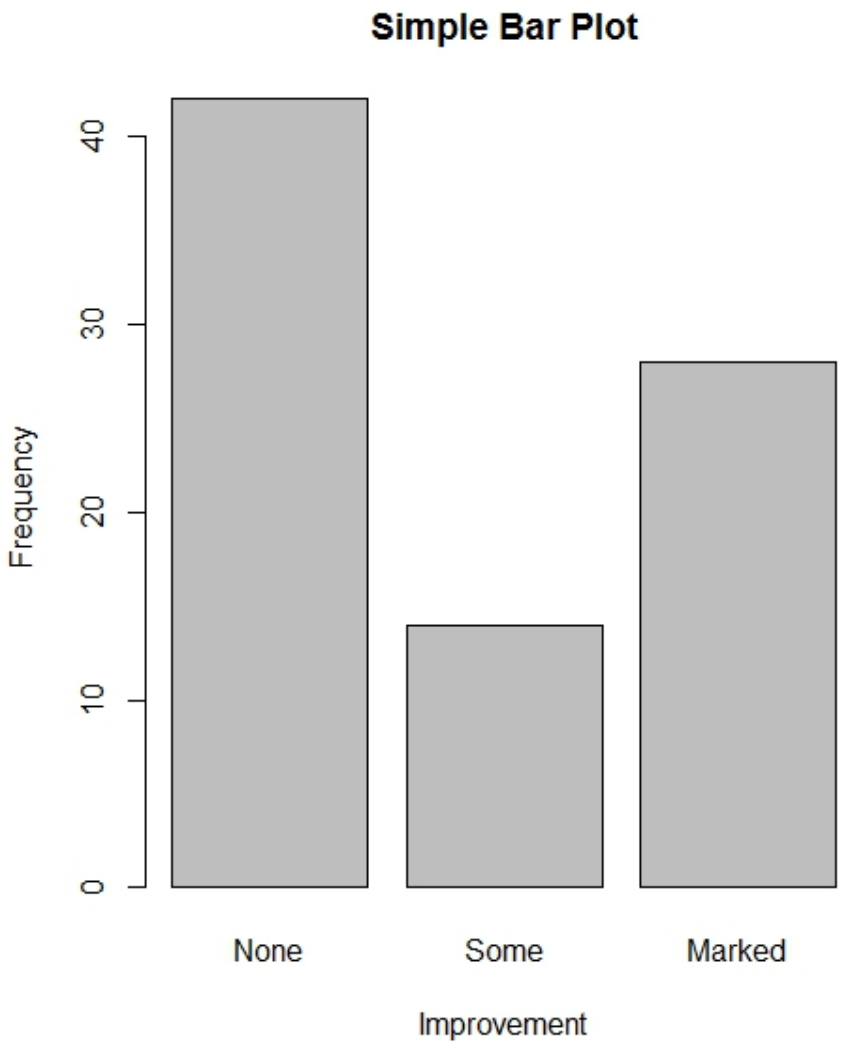


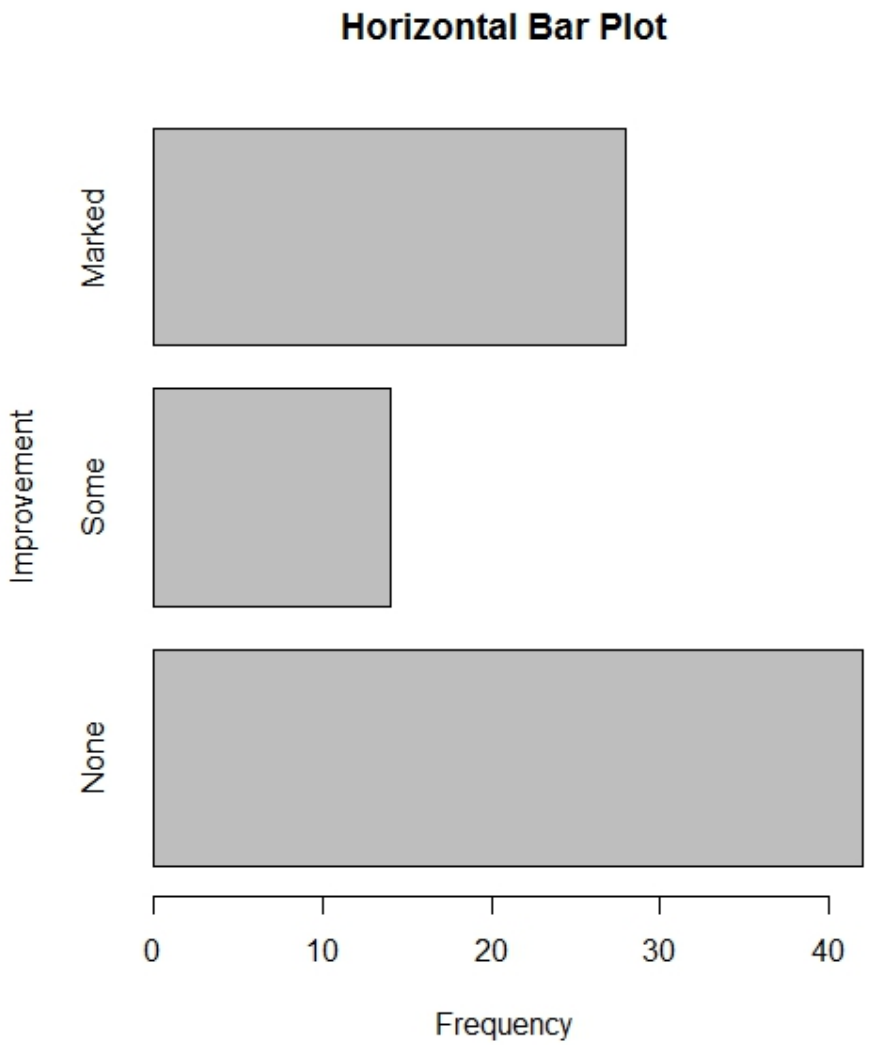
Basic Graphics

- **Bar plots** Display the distribution (frequency) of a categorical variable

```
install.packages("vcd")
library(vcd)
counts <- table(Arthritis$Improved)
barplot(counts,
    main="Simple Bar Plot",
    xlab="Improvement", ylab="Frequency")
barplot(counts,
    main="Horizontal Bar Plot",
    xlab="Frequency", ylab="Improvement",
    horiz=TRUE)
```

Basic Graphics





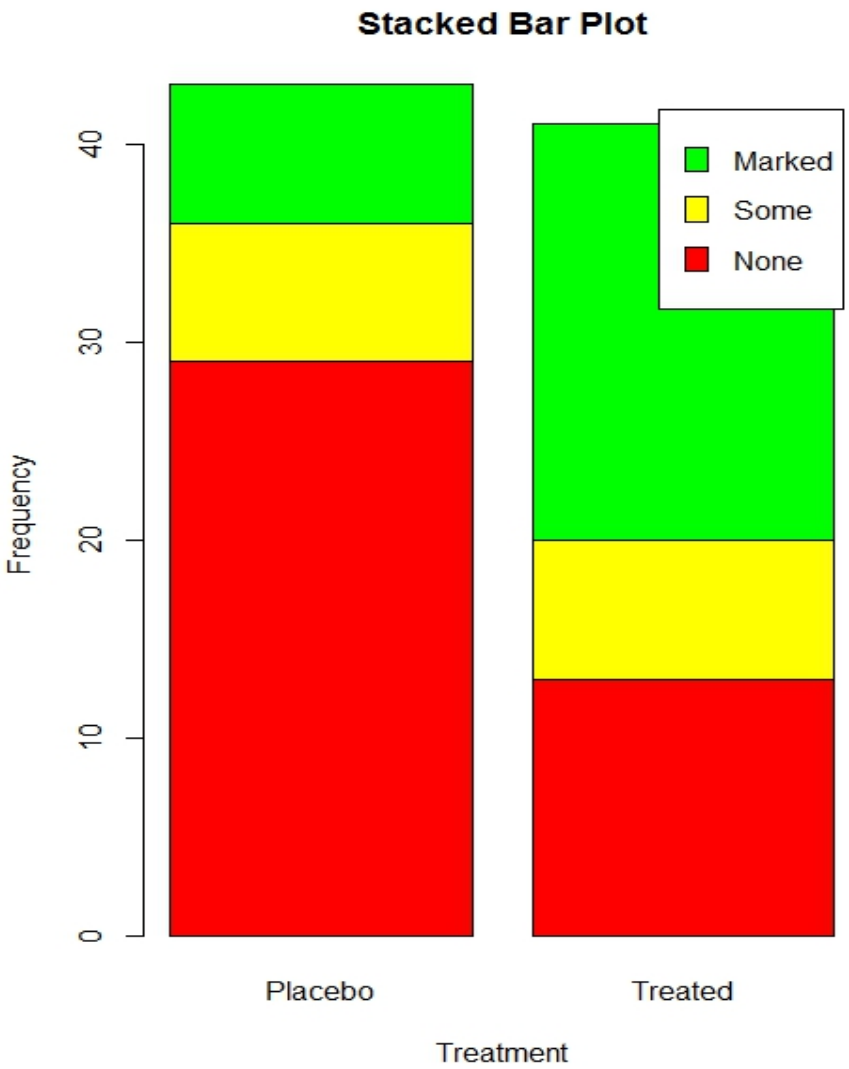
Basic Graphics

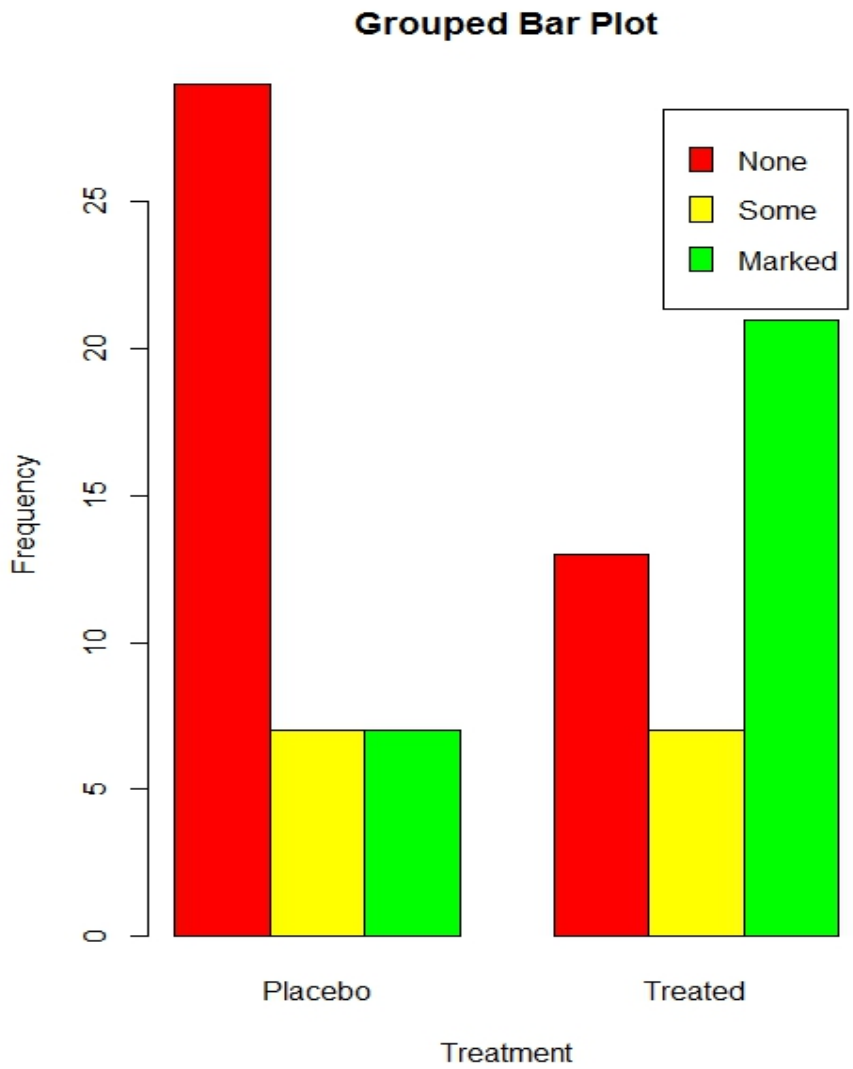
If your data is a matrix rather than a vector, the resulting graph will be a stacked or grouped bar plot

```
barplot(counts2, main="Stacked Bar Plot",
  xlab="Treatment", ylab="Frequency",
  col=c("red", "yellow","green"),
  legend=rownames(counts))

barplot(counts2, main="Grouped Bar Plot",
  xlab="Treatment", ylab="Frequency",
  col=c("red", "yellow", "green"),
  legend=rownames(counts), beside=TRUE)
```

By setting **`beside = TRUE`** we get a grouped bar plot

Basic Graphics



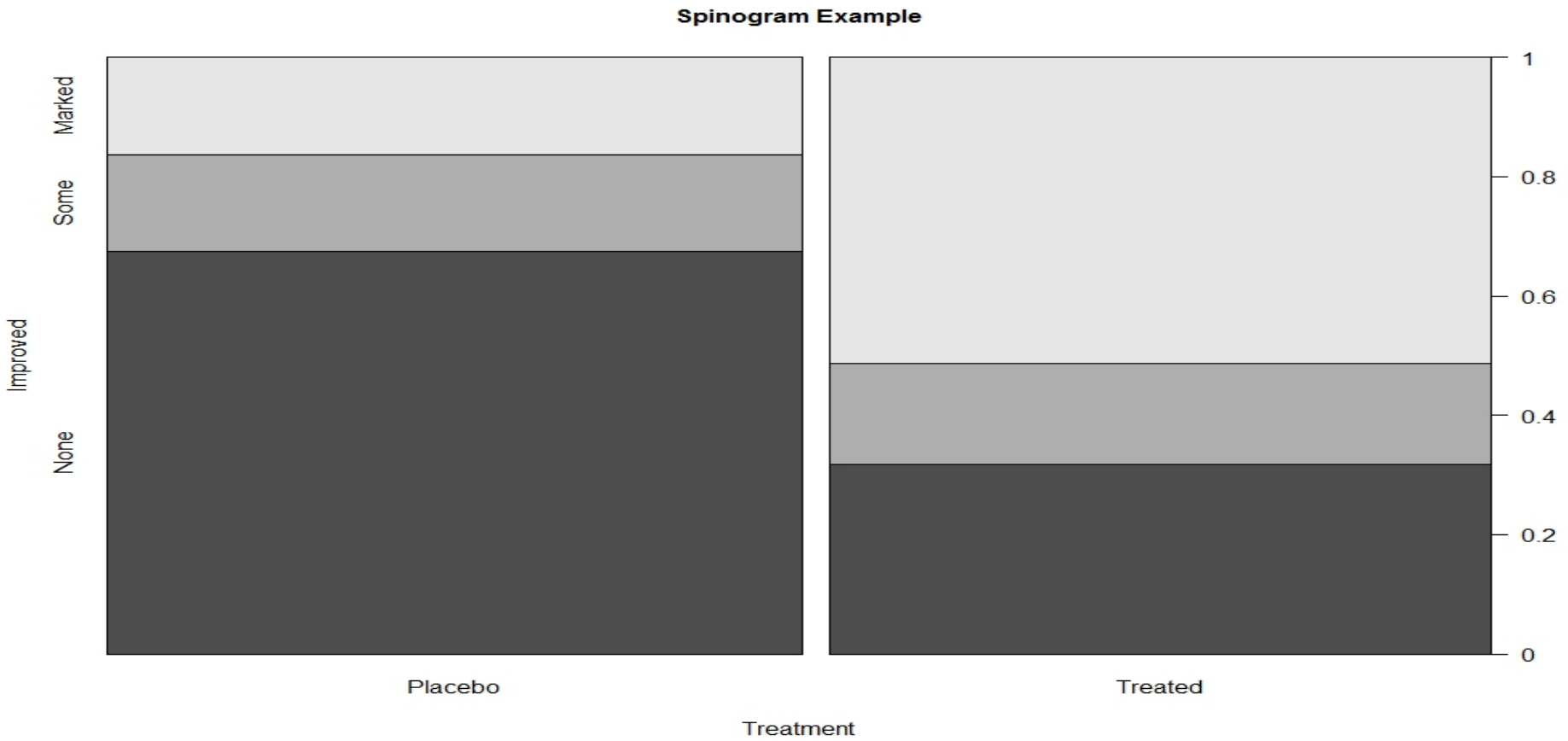


Basic Graphics

- As we finish our discussion of bar plots, lets take a look at a specialized version called a *spinogram*
- A spinogram is a stacked bar plot is rescaled so that the height of each bar is 1 and the segment heights represent proportions
- Spinograms are created with the `spine()` function of the `vcd` package

Basic Graphics

```
attach(Arthritis)
counts3 <- table(Treatment, Improved)
spine(counts, main="Spinogram Example")
detach(Arthritis)
```



Basic Graphics

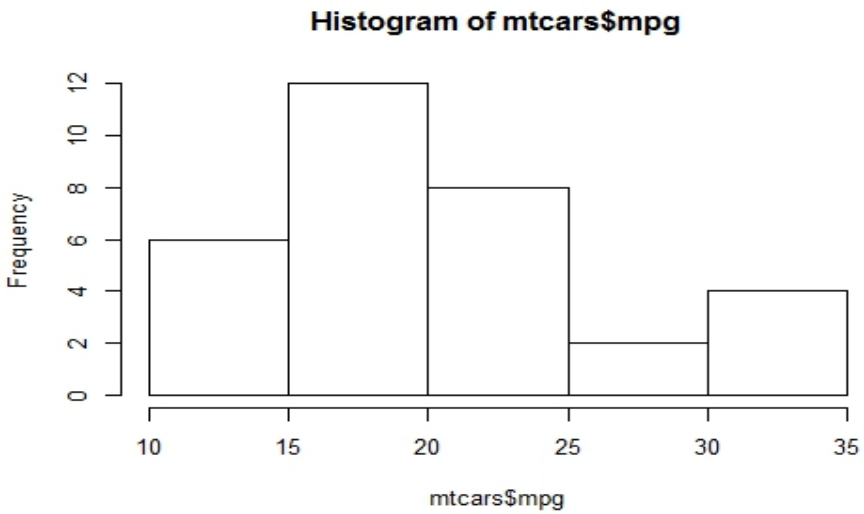
- **Histograms** Display the distribution of a continuous variable by dividing the range a number of bins on the x-axis and displaying the frequency in each bin on the y-axis
- The following code creates four histograms – from basic to more complex

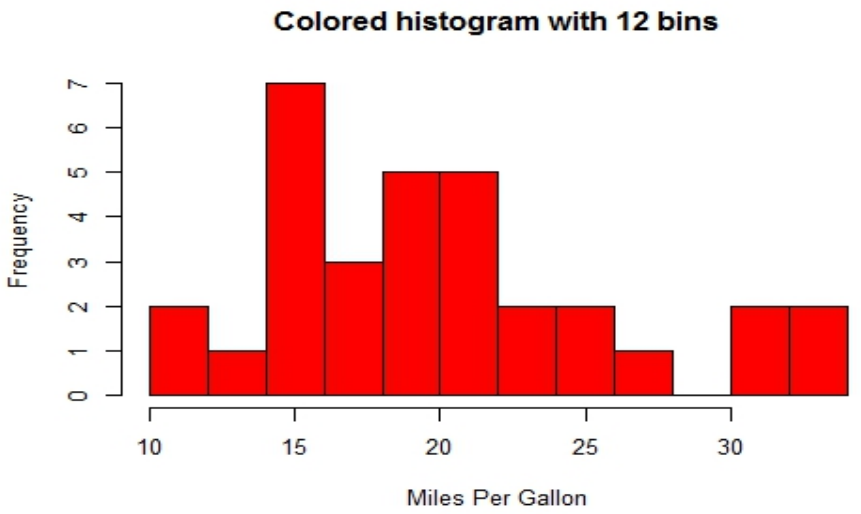
```
par(mfrow=c(2,2))
hist(mtcars$mpg)

hist(mtcars$mpg, breaks=12, col="red",
  xlab="Miles Per Gallon",
  main="Colored histogram with 12 bins")
```

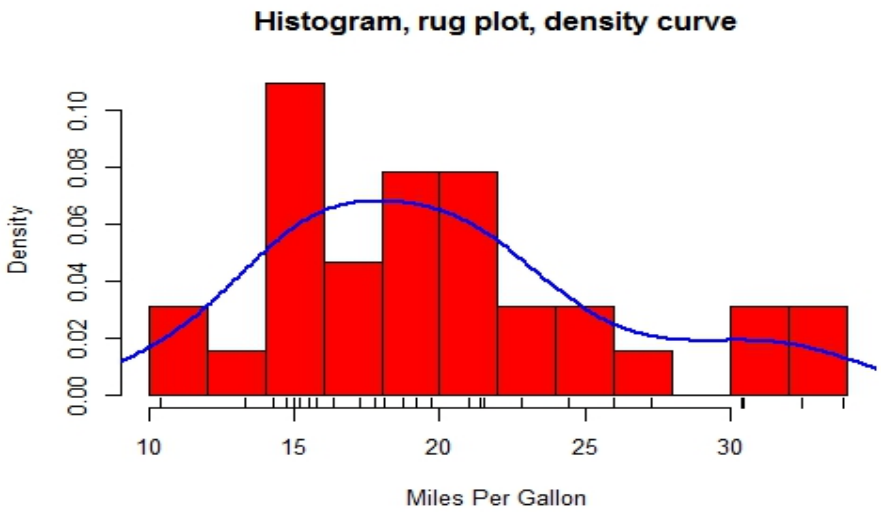
Basic Graphics

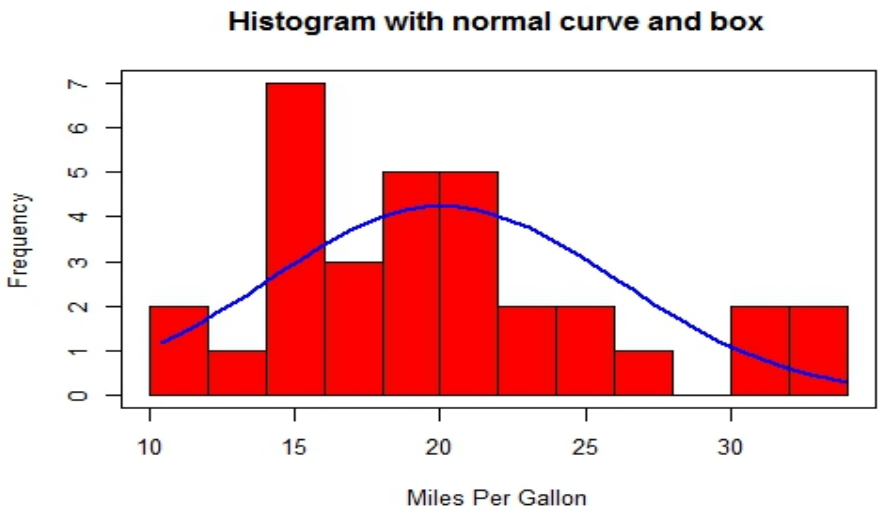
```
hist(mtcars$mpg, freq=FALSE, breaks=12,
  col="red", xlab="Miles Per Gallon",
  main="Histogram, rug plot, density curve")
  rug(jitter(mtcars$mpg))
  lines(density(mtcars$mpg), col="blue", lwd=2)

x <- mtcars$mpg
h<-hist(x, breaks=12,  col="red",
  xlab="Miles Per Gallon",
  main="Histogram with normal curve and box")
  xfit<-seq(min(x), max(x), length=40)
  yfit<-dnorm(xfit, mean=mean(x), sd=sd(x))
  yfit <- yfit*diff(h$mids[1:2])*length(x)
  lines(xfit, yfit, col="blue", lwd=2)
  box()
```

Basic Graphics







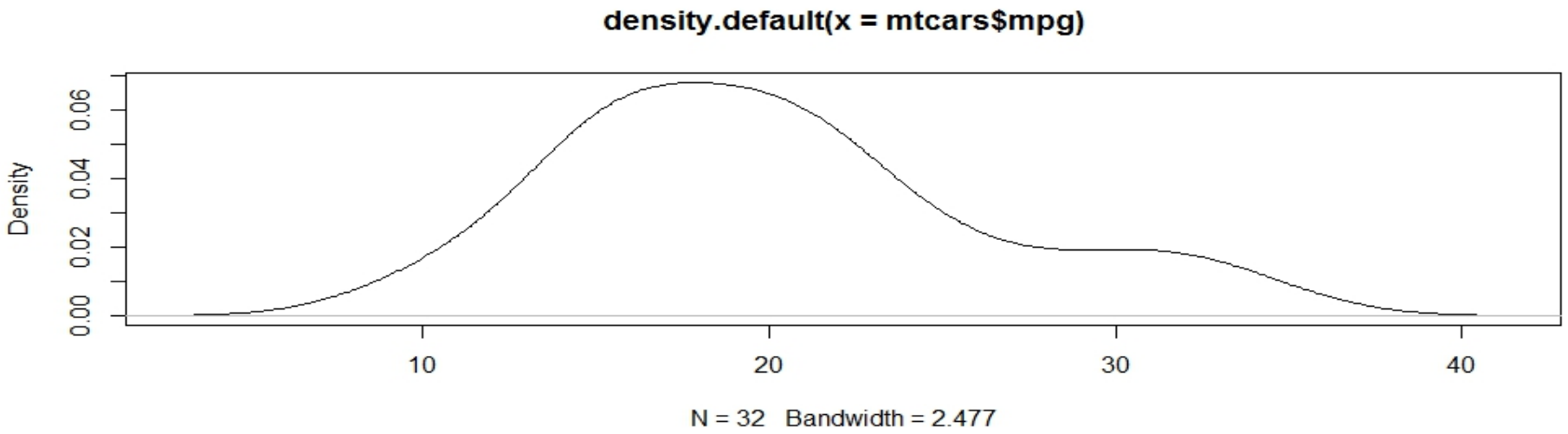


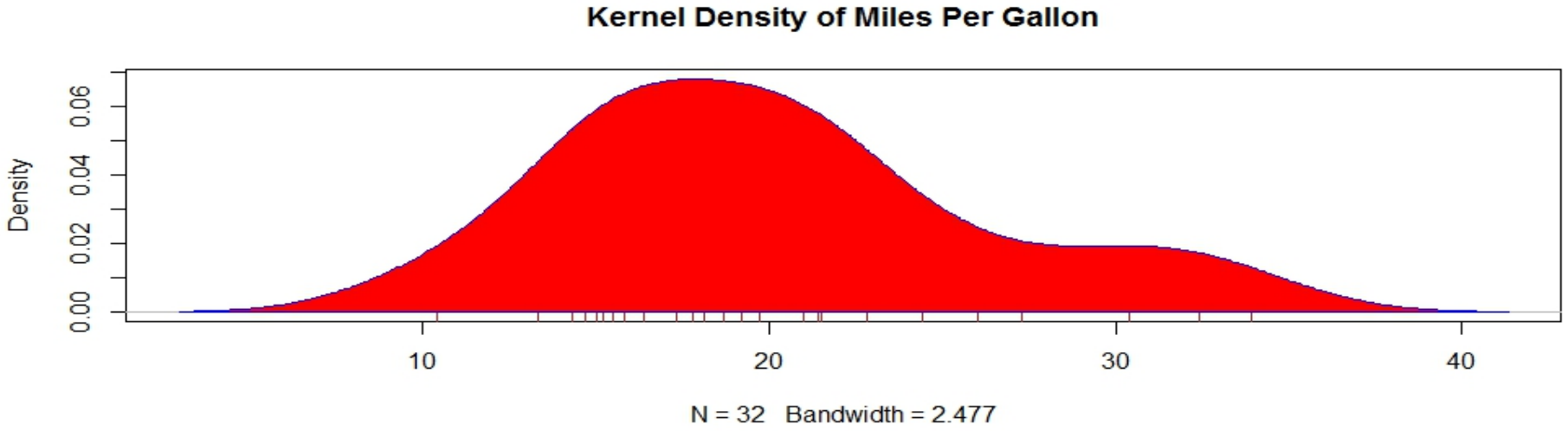
Basic Graphics

- **Density plots** Kernel density estimation is a nonparametric method for estimating the probability density function of a random variable
- Generally, density plots can be an effective way to view the distribution of a continuous variable

```
par(mfrow=c(2,1))
d <- density(mtcars$mpg)
plot(d)
d <- density(mtcars$mpg)
plot(d, main="Kernel Density of Miles Per Gallon")
polygon(d, col="red", border="blue")
rug(mtcars$mpg, col="brown")
```

Basic Graphics

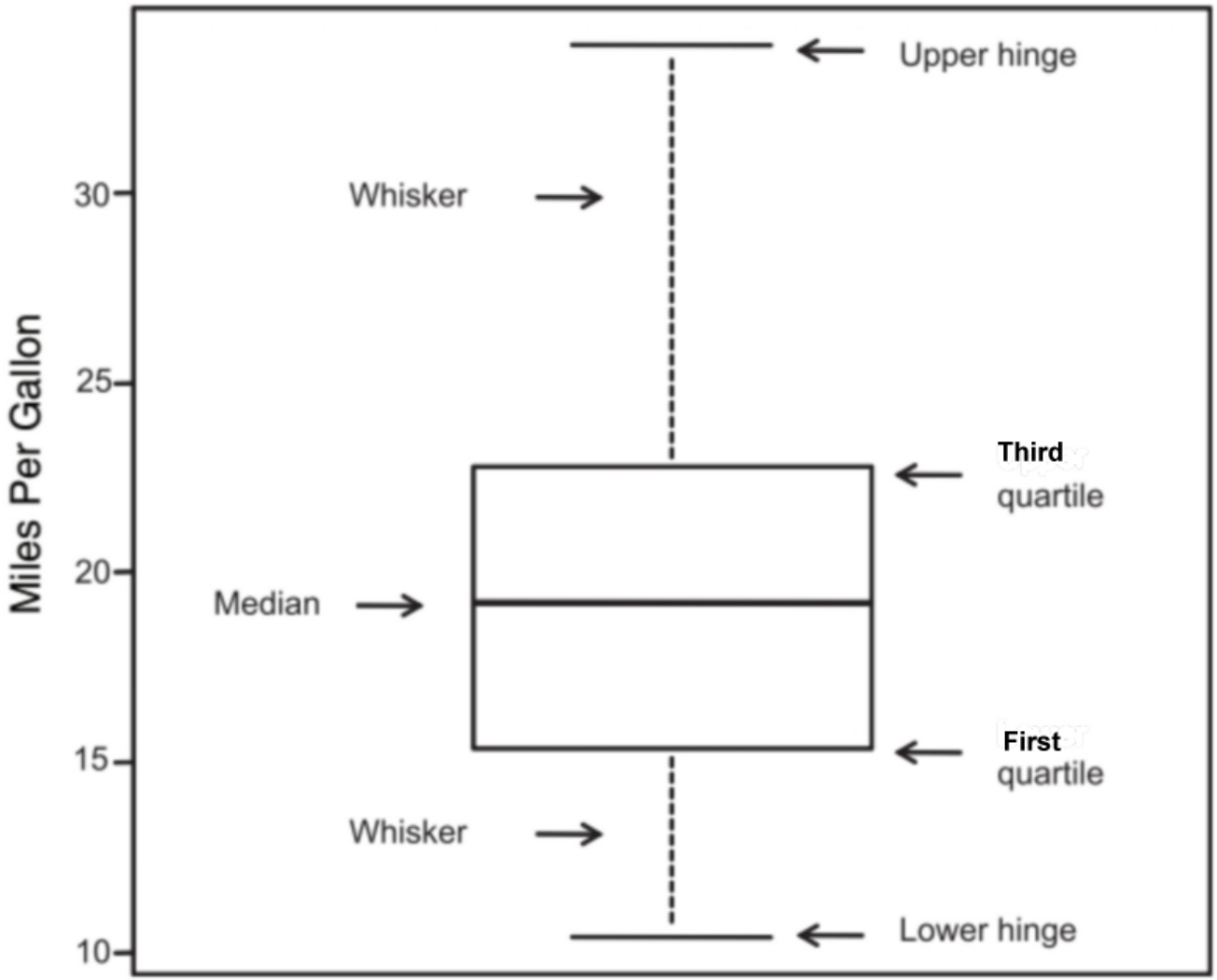




Basic Graphics

- **Boxplots** A *box-and-whiskers* plot describes the distribution of a continuous variable by plotting its five-number summary: the minimum, first quartile (25th percentile), median (50th percentile), third quartile (75th percentile), and maximum
- Includes observations that may be outliers (values outside the range of $\pm 1.5 * IQR$, where IQR is the interquartile range defined as the third quartile minus the first quartile)
- By default, each whisker extends to the most extreme data point, which is no more than 1.5 times the interquartile range
- Values outside this range are depicted as dots and may be outliers

Basic Graphics



Source: Kabacoff (2015)

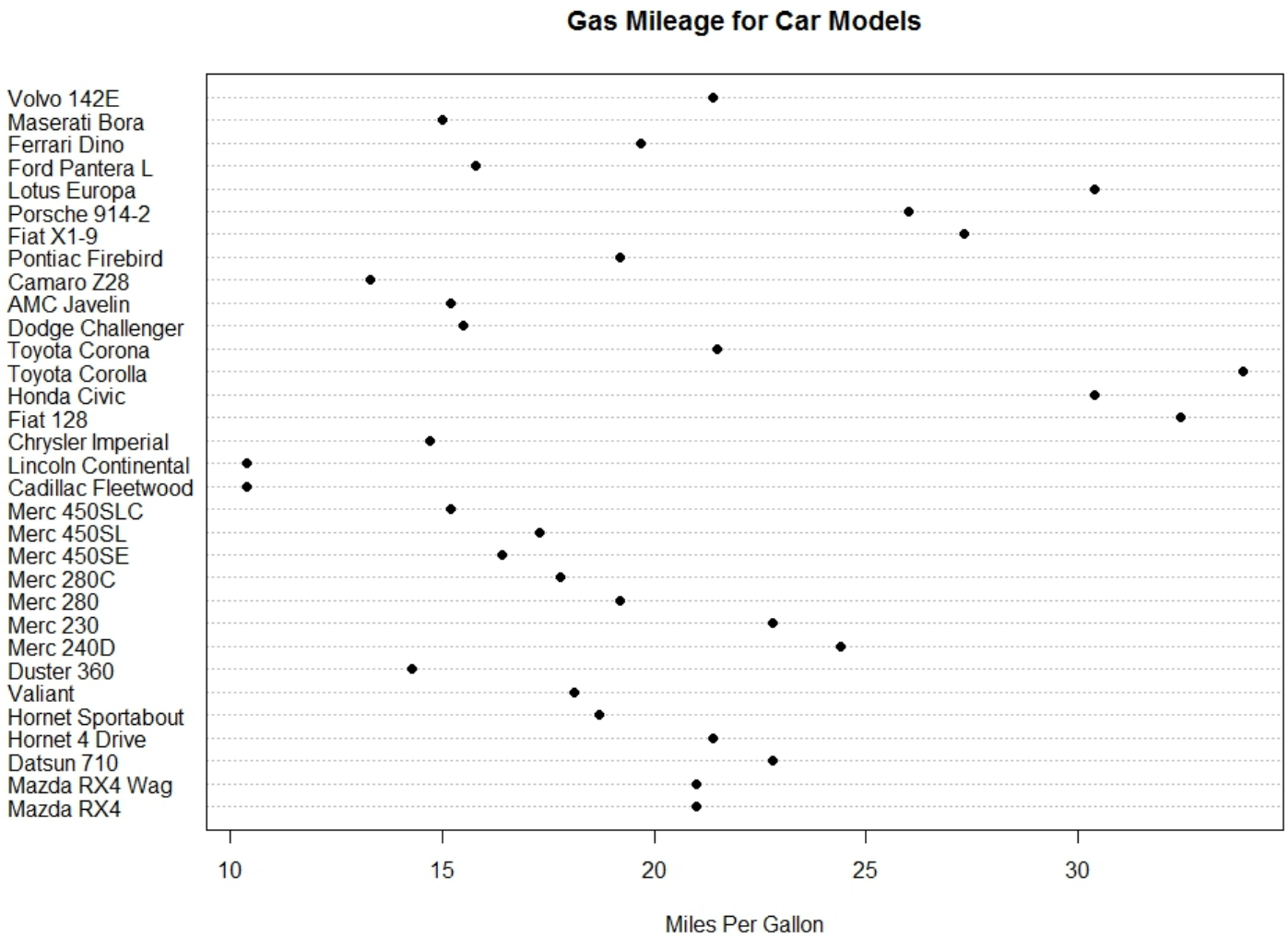
Basic Graphics

We can create the previous boxplot without the annotaitons with the following code

```
boxplot(mtcars\$mpg, main="Box plot",
  ylab="Miles per Gallon")
```

Basic Graphics

- **Dot plots** Provide a method of plotting a large number of labeled values on a simple horizontal scale



Advanced Graphics

There are four graphics packages in R:

System	Included in base installation?	Must be explicitly loaded?
`base`	Yes	No
`grid`	Yes	Yes
`lattice`	Yes	Yes
`ggplot2`	No	Yes

`grid` provides drawing primitives, but no tools for producing statistical graphics; We will not cover `grid`

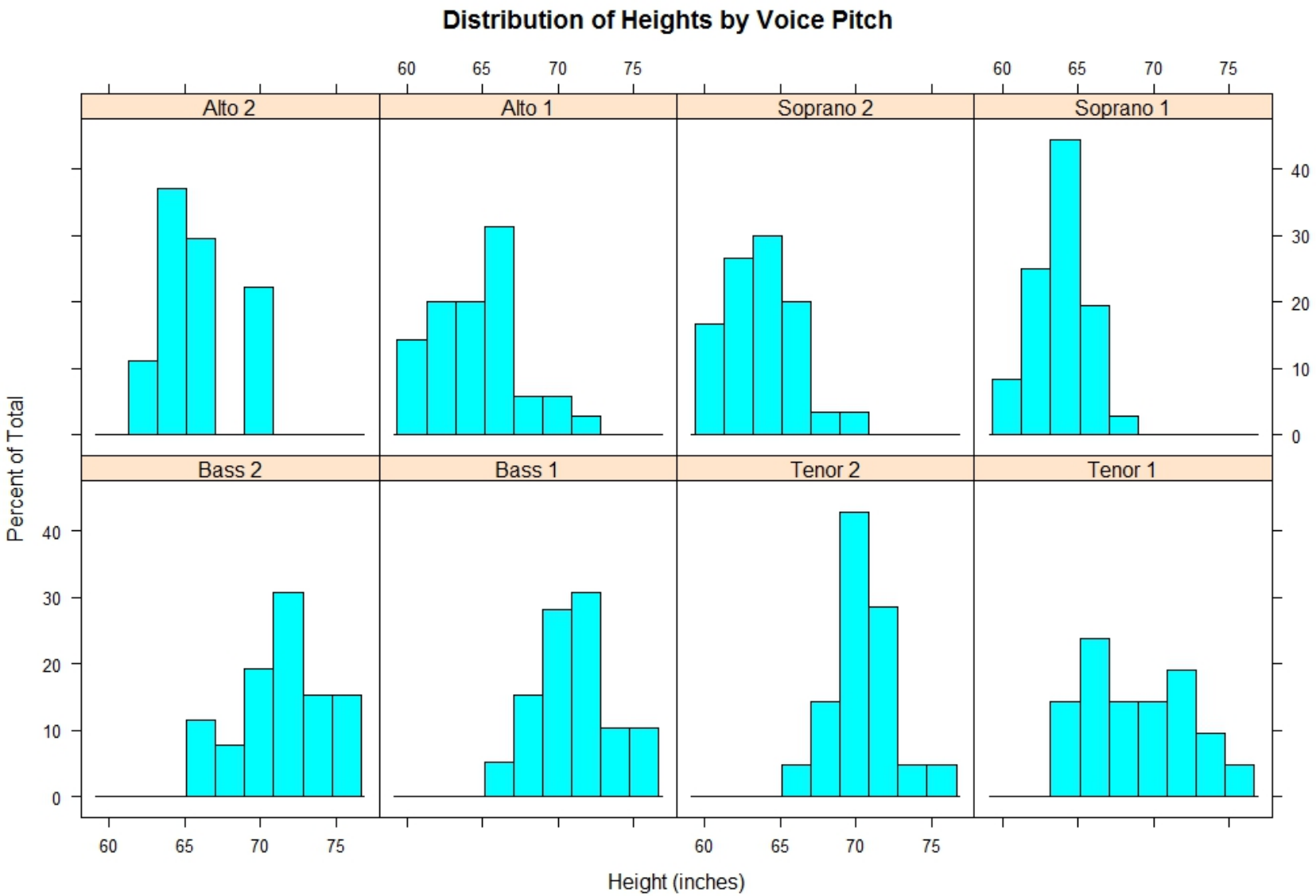
Base graphics are fine for assignments/projects–We will look more advanced packages so you are aware that they exist

Advanced Graphics

- `lattice` provides a comprehensive graphical system for visualizing univariate and multivariate data
- `lattice` has the ability to easily generate trellis graphs
- A trellis graph displays the distribution of a variable, or the relationship between variables, separately for each level of one or more other variables
- Let's look at an example: *How do the heights of singers in the New York Choral Society vary by their vocal parts?*

```
library(lattice)
histogram(~height | voice.part, data = singer,
   main="Distribution of Heights by Voice Pitch",
   xlab="Height (inches)")
```

Advanced Graphics



Advanced Graphics

Graph types and functions from the `lattice` package

Graph type	Function	Formula examples
3D contour plot	`contourplot()`	*z~x*y*
3D level plot	`levelplot()`	*z~y*x*
3D scatter plot	`cloud()`	*z~x*y|A*
3D wireframe graph	`wireframe()`	*z~y*x*
Bar chart	`barchart()`	*x~A* or *A~x*
Box plot	`bwplot()`	*x~A* or *A~x*
Dot plot	`dotplot()`	*~x|A*
Histogram	`histogram()`	*~x*
Kernel-density plot	`densityplot()`	*~x|A*B*
Parallel-coordinates plot	`parallelplot()`	`dataframe`
Scatter plot	`xyplot()`	*y~x|A*
Scatter-plot matrix	`splom()`	`dataframe`
Strip plots	`stripplot()`	*A~x* or *x~A*

Advanced Graphics – `lattice` Examples

```
library(lattice)
attach(mtcars)

gear <- factor(gear, levels=c(3, 4, 5),
  labels=c("3 gears", "4 gears", "5 gears"))
cyl <- factor(cyl, levels=c(4, 6, 8),
  labels=c("4 cylinders", "6 cylinders",
           "8 cylinders"))

densityplot(~mpg, main="Density Plot",
    xlab="Miles per Gallon")

densityplot(~mpg | cyl,
     main="Density Plot by Number of Cylinders",
     xlab="Miles per Gallon")
```

Advanced Graphics – `lattice` Examples

```
bwplot(cyl ~ mpg | gear,
  main="Box Plots by Cylinders and Gears",
  xlab="Miles per Gallon", ylab="Cylinders")

xyplot(mpg ~ wt | cyl * gear,
  main="Scatter Plots by Cylinders and Gears",
  xlab="Car Weight", ylab="Miles per Gallon")

cloud(mpg ~ wt * qsec | cyl,
  main="3D Scatter Plots by Cylinders")

dotplot(cyl ~ mpg | gear,
  main="Dot Plots by Number of Gears and Cylinders",
  xlab="Miles Per Gallon")
```

Advanced Graphics – `lattice` Examples

```
splom(mtcars[c(1, 3, 4, 5, 6)],
  main="Scatter Plot Matrix for mtcars Data")

detach(mtcars)
```

Advanced Graphics

- `ggplot2` developed by Hadley Wickham
- R package for producing graphics
- `ggplot2` is designed to work by adding layers of annotations and statistical summaries
- Takes a little practice learning the "grammar" of `ggplot2`
- Read the `ggplot2` book posted on Blackboard

Advanced Graphics – ggplot2 Examples

```
library(ggplot2)
set.seed(1410) # Make the sample reproducible
dsmall <- diamonds[sample(nrow(diamonds), 100), ]

qplot(carat, price, data = diamonds)
qplot(log(carat), log(price), data = diamonds)
qplot(carat, x * y * z, data = diamonds)
qplot(carat, price, data = dsmall, colour = color)
qplot(carat, price, data = dsmall, shape = cut)

qplot(carat, price, data = dsmall,
  geom = c("point", "smooth"))
```

Advanced Graphics – ggplot2 Examples

```
## Use GAM as a smoother

library(mgcv)
qplot(carat, price, data = dsmall,
  geom = c("point", "smooth"),
  method = "gam", formula = y ~ s(x))

## qplot does a lot for us but we can build
## layer by layer

qplot(sleep_rem / sleep_total, awake, data = msleep,
  geom = c("point", "smooth"))

ggplot(msleep, aes(sleep_rem / sleep_total, awake)) +
  geom_point() + geom_smooth()
```

References

Kabacoff, R. I. (2015).
R in Action.
Manning, Shelter Island, NY, second edition.

Lander, J. P. (2014).
R for Everyone.
Addison-Wesley, Upper Saddle River.

Zumel, N. and Mount, J. (2014).
Practical Data Science with R.
Manning, Shelter Island, NY, second edition.