

Topic 6: Basics of Plotting in R


1. Managing Plots in RStudio window

2. Functions for plotting

- `plot(x, y)`
- `hist()`
- `boxplot()`
- More with `plot(x, y)`

3. Plotting a Linear Model Object

1. Managing Plots with RStudio

- RStudio plot Window vs. R Markdown plots
 - R scripts will send plots to the Plots window
 - R Markdown will plot in editor Window
 - R Markdown can print plot in document with code when Knitting to file (more later)
- RStudio Plots window for convenient quick viewing of plots
 - R scripts in Plots window allow scrolling through previous plots 
 - Will need to Export plots and Copy and paste to include plots in documents
 - Can export as file
 - Easiest to Copy to clipboard and then paste in document
 - Can change size of Plots window to better fit plots
 - Can get error if Plots window is too small for plot

2. Basic Plots

- `plot(x,y)` cartesian plot (scatter plot)

```
#Create sequence of values for x (aka vector)
x<- seq(-3,3, length = 20)
#Create values of y that are normally distributed
function of x
y<- dnorm(x)
plot(x,y)
```

- Look up help for `plot()`

- Either `help(plot)` or use Help tab in Rstudio
- Note the different types
- R defaults to points when type not specified, try type as line

```
plot(x,y, type = "l")
```

Now try changing length to something larger than 20, say 50...

Note: you'll need to rerun code for redefining y as well

Basic Plots continued with labeling plots

- If sharing plots with others, important to have at least minimum of labels
- Some plot functions will default to labeling with column names
- Need to use quotes when using text labels
- Set of common label arguments for base R plotting functions
- `xlab = "Text label for x-axis"`
- `ylab = "Text label for y-axis"`
- `main = "text label for the overall plot"`

Basic Plots continued with other options

- Once a plot is created...
 - Other features can be added in separate lines and functions
 - Keep in mind scale and coordinate location
 - Lines as example
 - `abline()` draws a straight line
 - `lines()` adds a curve (or straight line) according to specified coordinates
- Color options can be applied to most any feature of plot
 - `col = "colorname"` (example of misleading code, mistake for "column")
- Syntax different for labeling with other plotting packages
 - ggplot2 package has similarities to some base R syntax on plots
 - more on ggplot2 (it's a popular and versatile plotting package)

Basic Plots continued with `hist()`

- Base R histogram function to visualize distribution of quantitative values
- Change number of bins (number of groups for frequency numeric)

```
data(mtcars)
hist(mtcars$mpg, breaks=20, main="Breaks=20")
hist(mtcars$mpg, breaks=5, main="Breaks=5")
```

- Include normal curve with histogram

```
x <- mtcars$mpg
h <- hist(x, breaks=10, col="red", xlab="Miles Per Gallon",
          main = "Histogram with Normal Curve")
xfit <- seq(min(x), max(x), length = 40)
yfit <- dnorm(xfit, mean = mean(x), sd = sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```

Basic Plots continued with `boxplot()`

- Base R function to visualize quantitative value for multiple categories
- Order of variables in functions matters
- ‘~’ is a versatile and useful character in R
- R expects quantitative variable ‘by’ a factor
- `cyl` is numeric, but R will still plot
- Order of `cyl` is intuitive here, but may re-order levels of factor (see managing data)

```
boxplot(mpg ~ cyl, mtcars, main="mpg by cylinder")
```

```
#the following throws an error
```

```
boxplot(cyl ~ mpg, mtcars)
```

```
#the following plots cyl as one box and mpg as another on same y-axis
```

```
boxplot(mtcars$cyl, mtcars$mpg) #this plots but is not correct
```

Basic Plots continued with `plot()`

- Base R function to visualize quantitative values by another set of quantitative values, i.e. scatter plot
- Order matters, y-axis variable is first
- Typically, y-axis is dependent variable (the response variable)
- Typically, x-axis is independent variable (the predictor variable)

```
plot(mpg~wt, data= mtcars)
```

- `lm()` function fits a linear least-squares model
- Similarly, can add line fitted line to scatter to plot above

```
abline(lm(mpg~wt, data= mtcars))
```

```
plot(mtcars$mpg, mtcars$wt) #also creates scatter  
plot, but note axes
```


3. Plotting a linear model object

- Same function , different argument
 - `plot(linearmodel)`
- simple and excellent tool for validating linear models
- But can be cumbersome manage
- Note instructions in console... Must click in console window and hit return to see plots
- If unfamiliar, will learn more about plots as part of coursework

```
mpgbywt <- lm(mpg~wt, data = mtcars)  
plot(mpgbywt)
```

Plotting a linear model object continued

- Will need to hit return 4 times to see all plots
- Can scroll through plots with arrows
- Or, can see all 4 plots in same Plots window
- Create 4 panels to put in plots, but may need to make window larger!

```
par(mfrow = c(2, 2))
```

```
plot(mpgbywt)
```

- Plot window remains divided until reset session, or hit  or use code like:

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