Chapter 4: Presentation Graphics

Alex Chi

Exercises

4.1 (Speed and stopping distance).

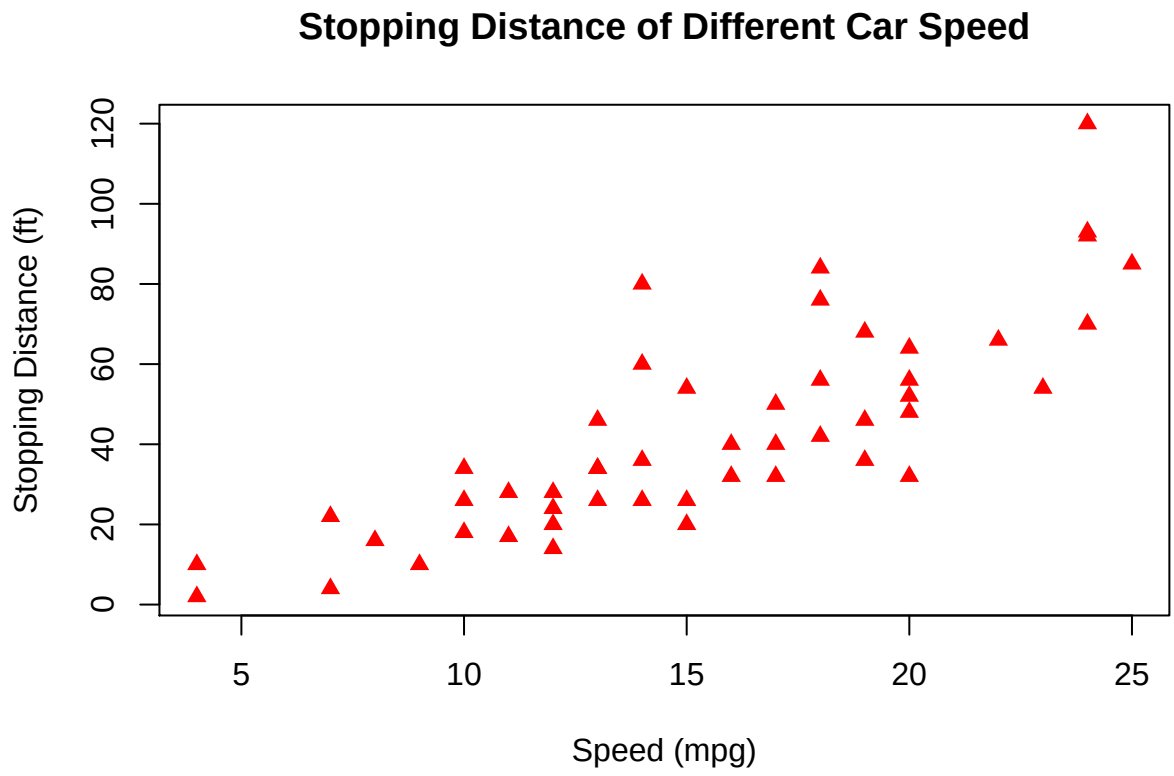
The data frame cars in the datasets package gives the speed (in mph) and stopping distance (in ft) for 50 cars.

a. Use the plot function to construct a scatterplot of speed (horizontal) against dist (vertical).

b. Revise the basic plot by labeling the horizontal axis with “Speed (mpg)”and the vertical axis with “Stopping Distance (ft),” Add a meaningful title to the plot.

c. Revise the plot by changing the plot symbol from the default open circles to red filled triangles (col=“red”, pch=17).

```
plot(cars$speed, cars$dist,
     xlab="Speed (mpg)",
     ylab="Stopping Distance (ft)",
     "main"="Stopping Distance of Different Car Speed",
     col="red", pch=17)
```



4.2 (Speed and stopping distance (continued)).

Suppose that one wishes to compare linear and quadratic fits to the (speed,dist) observations.

One can construct these two fits in R using the code

fit.linear = lm(dist ~ speed, data=cars)

fit.quadratic = lm(dist ~ speed + I(speed^2), data=cars)

a. Construct a scatterplot of speed and stopping distance.

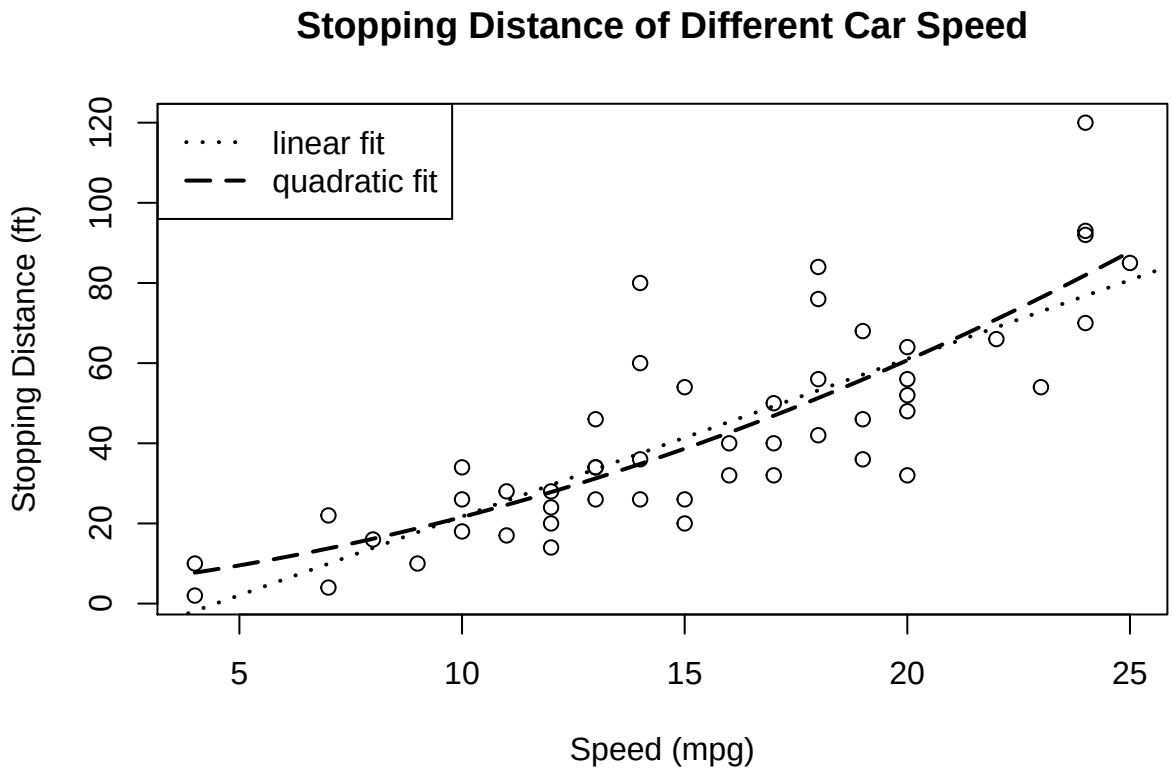
b. Using the abline function with argument fit.linear, overlay the best line fit using line type “dotted” and using a line width of 2.

c. Using the lines function, overlay the quadratic fit using line type “long- dash” and a line width of 2.

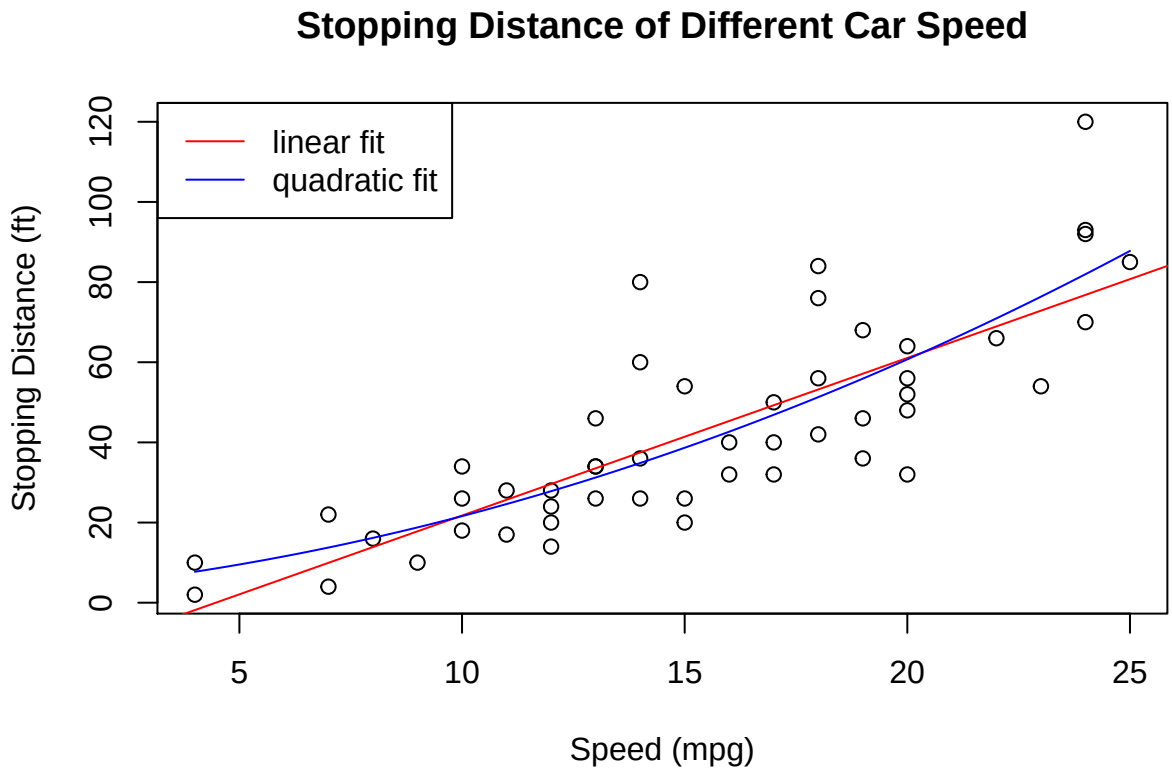
d. Use a legend to show the line types of the linear and quadratic fits.

e. Redo parts (a) - (d) using two contrasting colors (say red and blue) for the two different fits.

```
plot(dist~speed, cars,
     xlab="Speed (mpg)",
     ylab="Stopping Distance (ft)",
     "main"="Stopping Distance of Different Car Speed")
abline(lm(dist~speed, cars), lty="dotted", lwd=2)
fit ← lm(dist~poly(speed,2,raw=TRUE), cars)
curve(predict(fit,newdata=data.frame(speed=x)),
      lty="longdash", lwd=2,
      add=TRUE)
legend("topleft",
       legend=c("linear fit", "quadratic fit"),
       lty=c("dotted", "longdash"),
       lwd=2)
```



```r
plot(dist~speed, cars,
     xlab="Speed (mpg)",
     ylab="Stopping Distance (ft)",
     "main"="Stopping Distance of Different Car Speed")
abline(lm(dist~speed, cars), col="red")
fit ← lm(dist~poly(speed,2,raw=TRUE), cars)
curve(predict(fit,newdata=data.frame(speed=x)),
      col="blue", add=TRUE)
legend("topleft",
       legend=c("linear fit", "quadratic fit"),
       lty=c(1, 1),
       col=c("red", "blue"))
```



4.3 (Speed and stopping distance (continued)).

a. Construct a residual plot for the linear fit by typing plot(cars$speed, fit.linear$residual)

b. Add a blue, thick (lwd=3) horizontal line to the residual plot using the abline function.

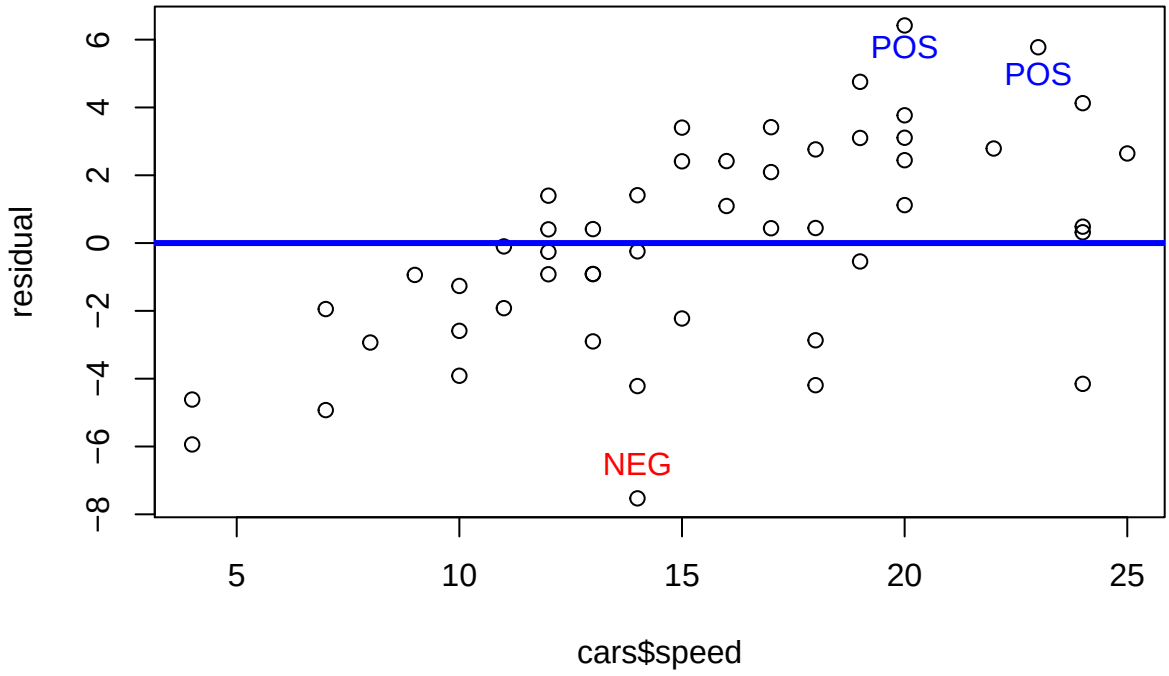
c. There are two large positive residuals in this graph. By two applications of the text function, label each residual using the label "POS" in blue.

d. Label the one large negative residual in the graph with the label "NEG"in red.

e. Use the identify function to find the row numbers of two observations that have residuals close to zero.

Note: 似乎现在的 R 语言没有 fit.linear，因此使用 lm 生成线性模型。

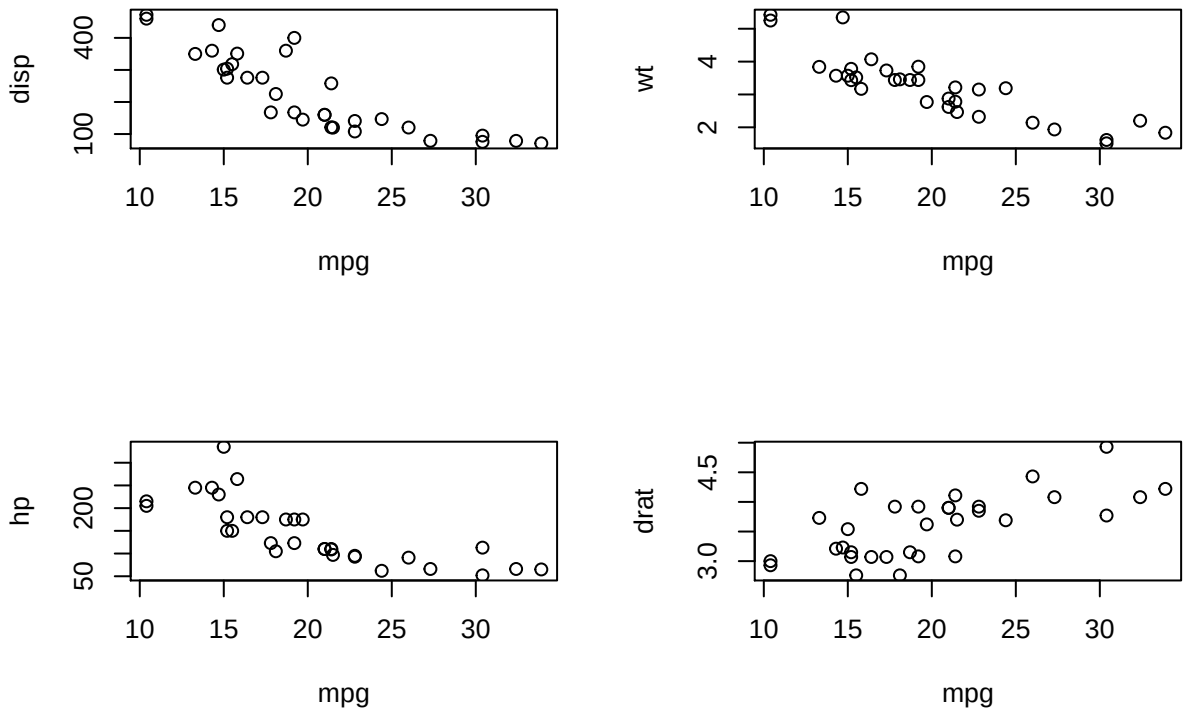
```r
fit = lm(speed ~ dist, cars)
residual = resid(fit)
plot(residual ~ cars$speed)
abline(0, 0, lwd=3, col="blue")
text(20, 5.8, "POS", col="blue")
text(23, 5, "POS", col="blue")
text(14, -6.5, "NEG", col="red")
```



4.4 (Multiple graphs).

The dataset mtcars contains measurements of fuel consumption (variable mpg) and other design and performance characteristics for a group of 32 automobiles. Using the mfrow argument to par, construct scatterplots of each of the four variables disp (displacement), wt (weight), hp (horsepower), drat (rear axle ratio) with mileage (mpg) on a two by two ar- ray. Comparing the four graphs, which variable among displacement, weight, horsepower, and rear axle ratio has the strongest relationship with mileage?

```r
par(mfrow = c(2, 2))
plot(disp ~ mpg, mtcars)
plot(wt ~ mpg, mtcars)
plot(hp ~ mpg, mtcars)
plot(drat ~ mpg, mtcars)
```



4.5 (Drawing houses).

The following function house plots an outline of a house centered about the point (x, y)

a. Read the function house into R.

b. Use the plot.new function to open a new plot window. Using the plot.window function, set up a coordinate system where the horizontal and vertical scales both range from 0 to 10.

c. Using three applications of the function house, draw three houses on the current plot window centered at the locations (1, 1), (4, 2), and (7, 6).

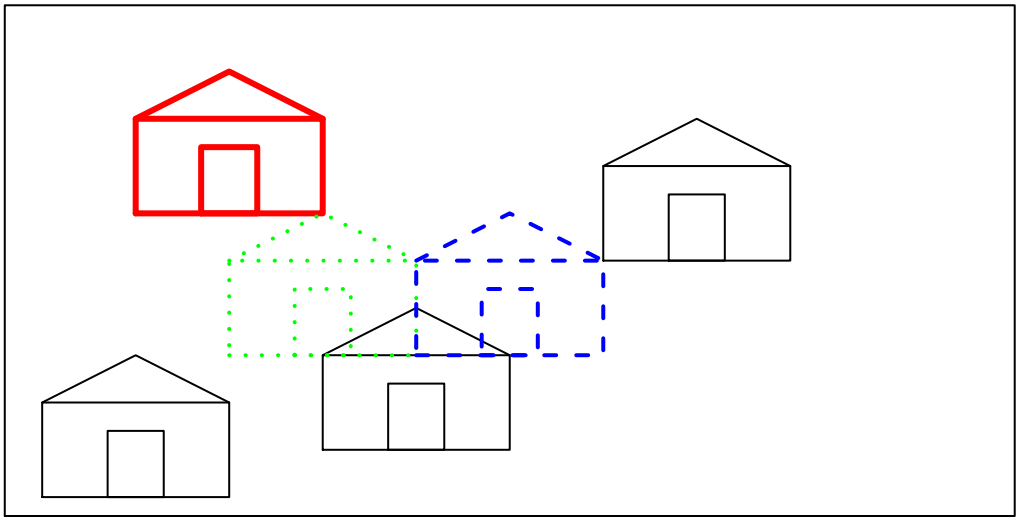
d. Using the ... argument, one is able to pass along parameters that mod- ify attributes of the line function. For example, if one was interested in drawing a red house using thick lines at the location (2, 7), one can type house(2, 7, col="red", lwd=3) Using the col and lty arguments, draw three additional houses on the current plot window at different locations, colors, and line types.

e. Draw a boundary box about the current plot window using the box func- tion.

```
house=function(x, y, ...) {
  lines(c(x - 1, x + 1, x + 1, x - 1, x - 1), c(y - 1, y - 1, y + 1, y + 1, y - 1), ...)
  lines(c(x - 1, x, x + 1), c(y + 1, y + 2, y + 1), ...)
  lines(c(x - 0.3, x + 0.3, x + 0.3, x - 0.3, x - 0.3), c(y - 1, y - 1, y + 0.4, y + 0.4, y - 1), ...
}

plot.new()
plot.window(xlim=c(0, 10), ylim=c(0, 10), pty="s")
house(1, 1)
house(4, 2)
house(7, 6)
```

```
house(2, 7, col="red", lwd=3)
house(3, 4, col="green", lwd=2, lty="dotted")
house(5, 4, col="blue", lwd=2, lty="dashed")
box()
```



4.6 (Drawing beta density curves).

a. Use three applications of the curve function to display the Beta(2, 6), Beta(4, 4), and Beta(6, 2) densities on the same plot. (The curve function with the add=TRUE argument will add the curve to the current plot.)

b. Use the following R command to title the plot with the equation of the beta density.

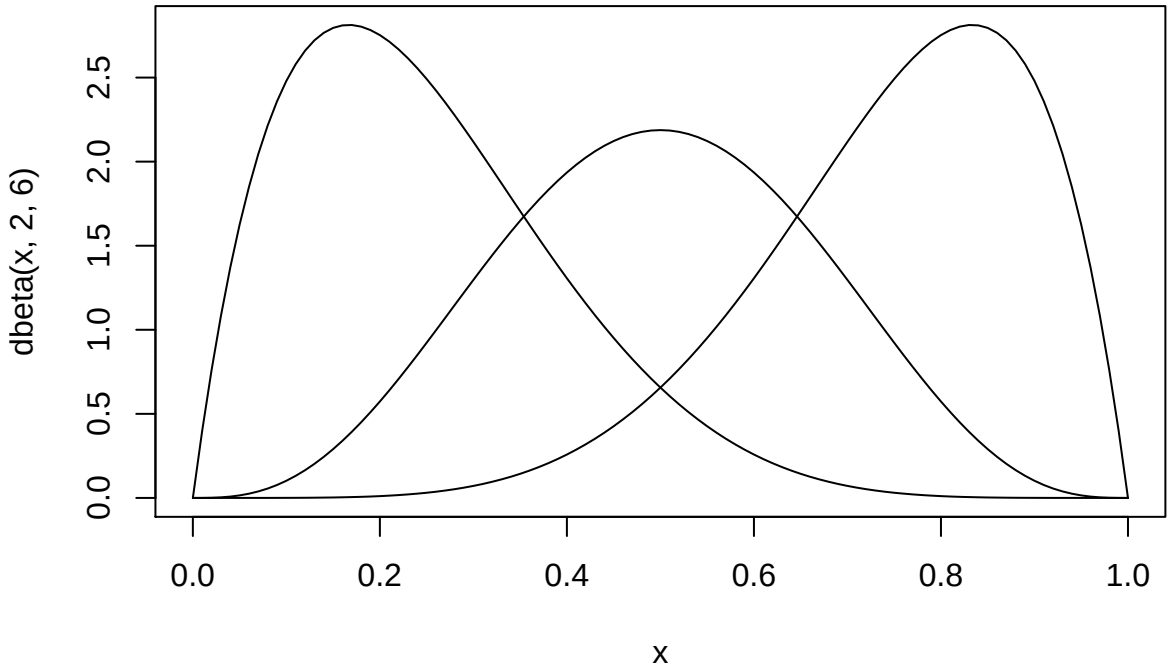
c. Using the text function, label each of the beta curves with the correspond- ing values of the shape parameters a and b.

d. Redraw the graph using different colors or line types for the three beta density curves.

e. Instead of using the text function, add a legend to the graph that shows the color or line type for each of the beta density curves.

```
curve(dbeta(x, 2, 6), from=0, to=1)
curve(dbeta(x, 4, 4), from=0, to=1, add=TRUE)
curve(dbeta(x, 6, 2), from=0, to=1, add=TRUE)
title(expression(f(y)==frac(1,B(a,b))*y^{a-1}*(1-y)^{b-1}))
```

$$f(y) = \frac{1}{B(a, b)} y^{a-1}(1-y)^{b-1}$$



4.7 (lattice graphics)

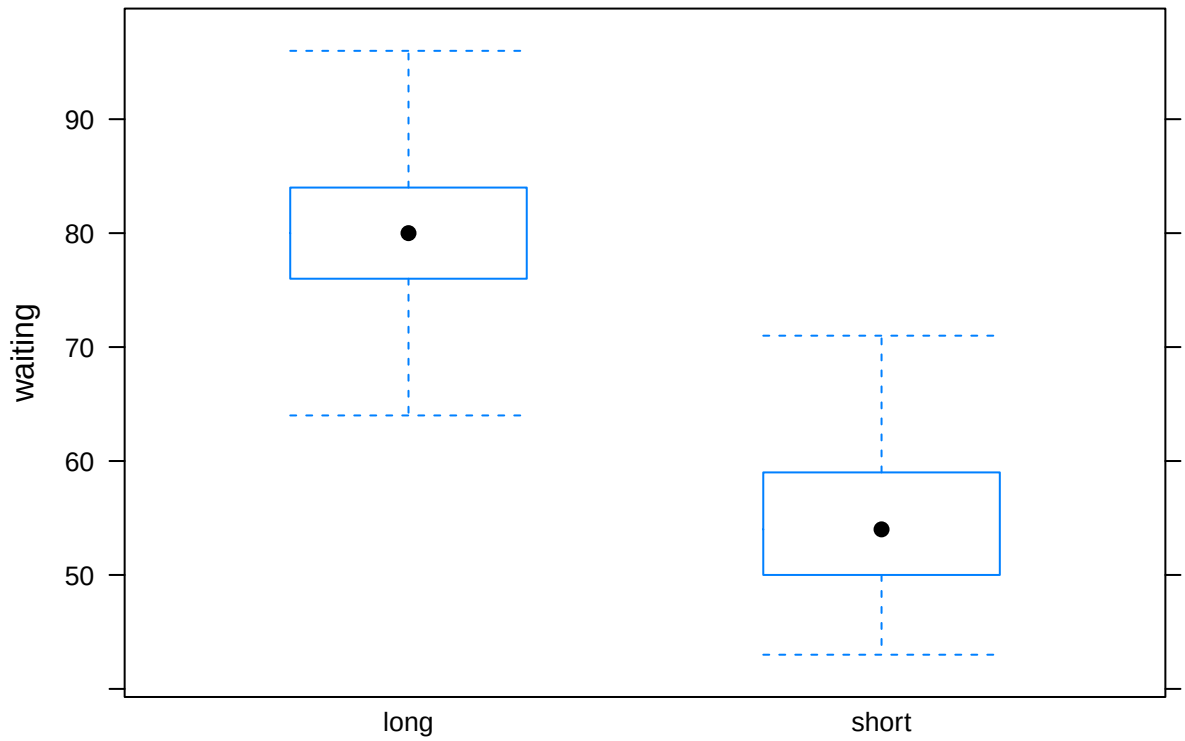
The dataset faithful contains the duration of the eruption (in minutes) eruptions and the waiting time until the next eruption waiting (in minutes) for the Old Faithful geyser. One is interested in exploring the relationship between the two variables.

a. Create a factor variable length that is "short" if the eruption is smaller than 3.2 minutes, and "long" otherwise.

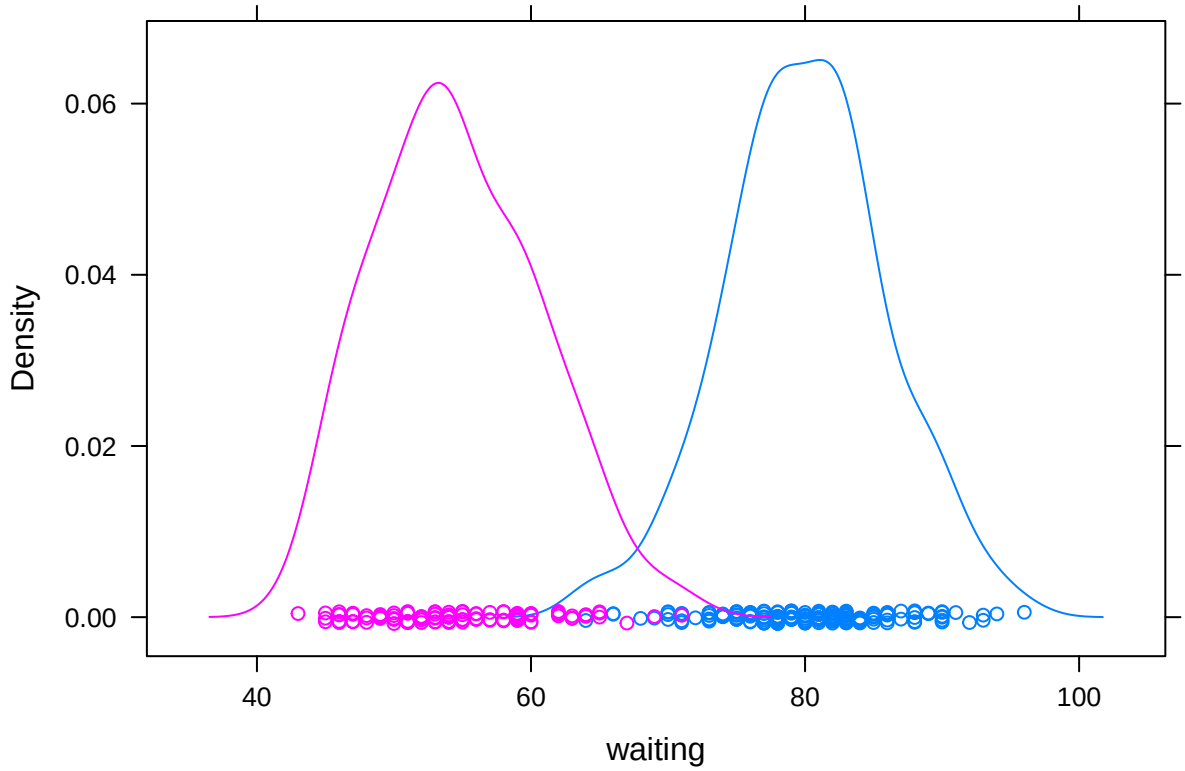
b. Using the bwplot function in the lattice package, construct parallel box- plots of the waiting times for the "short" and "long" eruptions.

c. Using the densityplot function, construct overlapping density plots of the waiting times of the "short" and "long" eruptions.

```
library(lattice)
faithful$length = ifelse(faithful$eruptions < 3.2, "short", "long")
bwplot(waiting ~ length, faithful)
```




```
densityplot(~ waiting, groups = length, faithful)
```



4.8 (ggplot2 graphics).

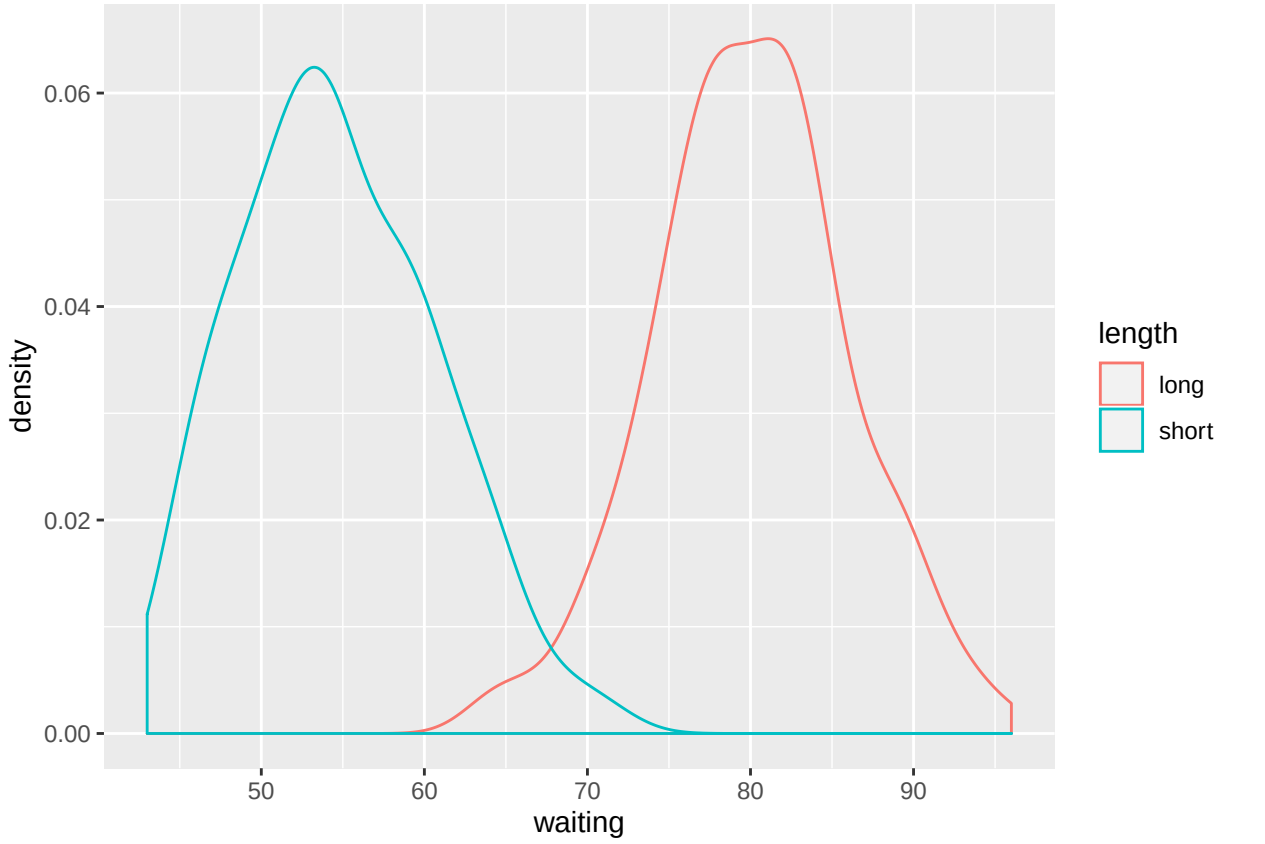
In Exercise 4.7, the waiting times for the Old Faithful geysers were compared for the short and long eruptions where the variable length in the faithful data frame defines the duration of the erup- tion.

a. Suppose a data frame dframe contains a numeric variable num.var and a factor factor.var. After the ggplot2 package has been loaded, then the R commands ggplot(dframe, aes(x = num.var, color = factor.var)) + geom_density() will construct overlapping density estimates of the variable num.var for each value of the factor factor.var. Use these commands to construct overlapping density estimates of the waiting times of the geysers with short and long eruptions.

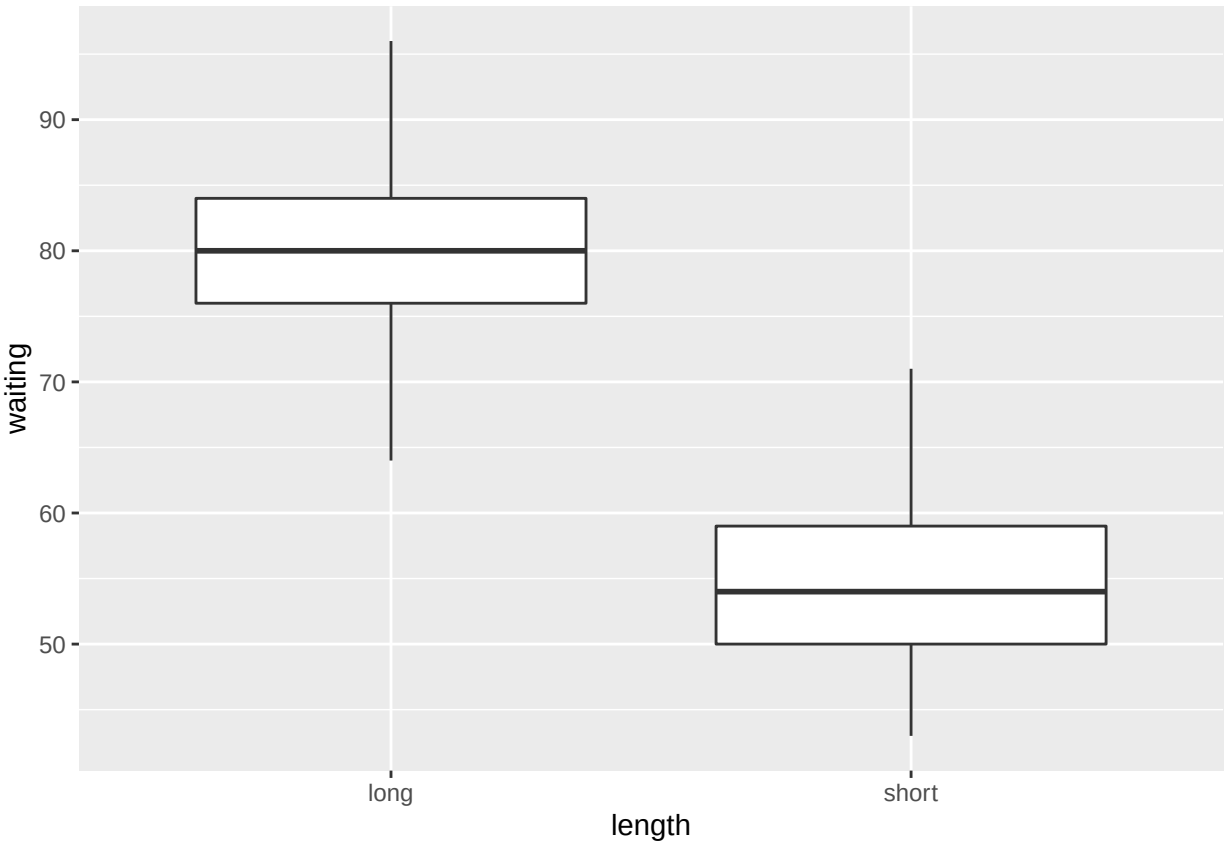
b. With a data frame dframe containing a numeric variable num.var and a factor factor.var, the ggplot2 syntax will construct parallel boxplots of the variable num.var for each value of the factor factor.var. Use these commands to construct parallel boxplots of the waiting times of the geysers with short and long eruptions.

```
library(ggplot2)

ggplot(faithful, aes(x = waiting, color = length)) + geom_density()
```




```
ggplot(faithful, aes(y = waiting, x = length)) + geom_boxplot()
```


90
80
70
60
50
waiting
long
short
length