IS5 in R: Scatterplots, Association, and Correlation (Chapter 6)

Nicholas Horton (nhorton@amherst.edu)

December 19, 2020

Introduction and background

This document is intended to help describe how to undertake analyses introduced as examples in the Fifth Edition of *Intro Stats* (2018) by De Veaux, Velleman, and Bock. This file as well as the associated R Markdown reproducible analysis source file used to create it can be found at http://nhorton.people.amherst.edu/is5.

This work leverages initiatives undertaken by Project MOSAIC (http://www.mosaic-web.org), an NSF-funded effort to improve the teaching of statistics, calculus, science and computing in the undergraduate curriculum. In particular, we utilize the `mosaic` package, which was written to simplify the use of R for introductory statistics courses. A short summary of the R needed to teach introductory statistics can be found in the mosaic package vignettes (https://cran.r-project.org/web/packages/mosaic). A paper describing the mosaic approach was published in the *R Journal*: https://journal.r-project.org/archive/2017/RJ-2017-024.

Chapter 6: Scatterplots, Association, and Correlation

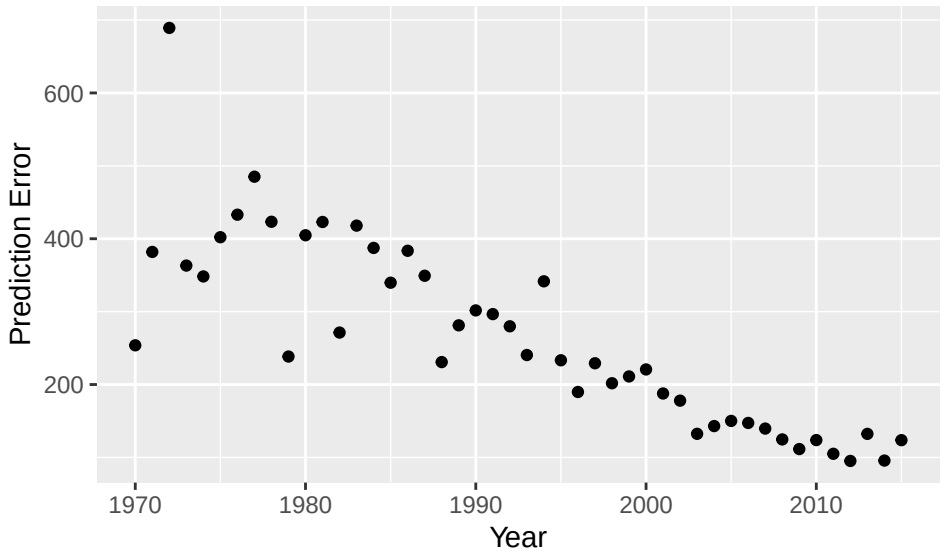
We begin by reading in the data.

```
library(mosaic)
library(readr)
library(janitor)
Hurricanes <- read_csv("http://nhorton.people.amherst.edu/is5/data/Tracking_hurricanes_2015.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Error_24h = col_double(),
##   Error_48h = col_double(),
##   Error_72h = col_double()
## )
```

By default, `read_csv()` prints the variable names. These messages can (and should!) be suppressed using the `message=FALSE` code chunk option to save space and improve readability.

```
# Figure 6.1, page 164
gf_point(Error_72h ~ Year, data = Hurricanes, ylab = "Prediction Error")
```



Section 6.1: Scatterplots

See dots on pages 164-165.

Example 6.1: Comparing Prices Worldwide We begin by reading in the data.

```
Prices <- read_csv("http://nhorton.people.amherst.edu/is5/data/Prices_and_Earnings.csv") %>%
  janitor::clean_names()
names(Prices)
```

```
##  [1] "city"                  "food_costs"
##  [3] "womens_clothing"       "mens_clothing"
##  [5] "i_phone_4s_hr"         "clothing_index"
##  [7] "hours_worked"          "wage_gross"
##  [9] "wage_net"              "vacation_days"
## [11] "col_excl_rent"         "col_incl_rent"
## [13] "pur_power_gross"       "pur_power_net"
## [15] "pur_power_annual"      "big_mac_min"
## [17] "bread_kg_in_min"       "rice_kg_in_min"
## [19] "goods_and_services"    "good_and_services_index"
## [21] "food_index"
```

```
glimpse(Prices)
```

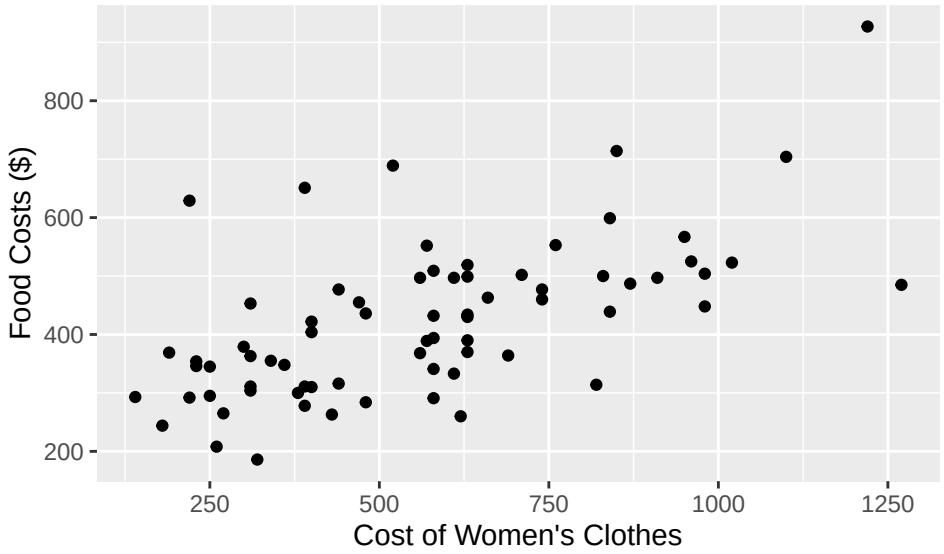
```
## Rows: 72
## Columns: 21
## $ city                  <chr> "Amsterdam", "Athens", "Auckland", "Bangkok...
## $ food_costs            <dbl> 364, 390, 497, 422, 394, 463, 389, 363, 345...
## $ womens_clothing       <dbl> 690, 630, 560, 400, 580, 660, 570, 310, 250...
## $ mens_clothing         <dbl> 1040, 1110, 670, 600, 1110, 700, 710, 440, ...
## $ i_phone_4s_hr         <dbl> 44.5, 86.0, 51.0, 165.0, 52.5, 184.0, 55.5,...
## $ clothing_index        <dbl> 110.8, 112.5, 79.2, 64.2, 109.2, 87.5, 82.5...
## $ hours_worked          <dbl> 1755, 1822, 1852, 2312, 1761, 1979, 1742, 1...
## $ wage_gross            <dbl> 78.3, 41.4, 59.8, 14.6, 59.6, 17.0, 79.2, 2...
## $ wage_net              <dbl> 69.4, 40.0, 63.5, 17.4, 58.7, 18.0, 70.1, 2...
## $ vacation_days         <dbl> 24, 23, 20, 7, 29, 9, 29, 15, 24, 20, 26, 2...
## $ col_excl_rent         <dbl> 77.0, 66.1, 76.7, 55.3, 74.7, 60.3, 72.3, 5...
```

```
## $ col_incl_rent           <dbl> 69.0, 58.1, 67.7, 48.1, 65.6, 51.8, 64.1, 4...
## $ pur_power_gross         <dbl> 101.6, 62.6, 78.0, 26.5, 79.7, 28.3, 109.6,...
## $ pur_power_net           <dbl> 90.1, 60.5, 82.9, 31.4, 78.6, 29.9, 97.0, 4...
## $ pur_power_annual        <dbl> 75.7, 52.1, 74.8, 33.7, 66.8, 28.2, 82.1, 3...
## $ big_mac_min             <dbl> 16, 30, 16, 36, 19, 34, 16, 52, 32, 20, 57,...
## $ bread_kg_in_min         <dbl> 7, 13, 17, 26, 12, 28, 11, 34, 21, 11, 21, ...
## $ rice_kg_in_min          <dbl> 9, 26, 8, 20, 6, 16, 9, 17, 20, 12, 27, 27,...
## $ goods_and_services      <dbl> 3034, 2605, 3019, 2178, 2941, 2375, 2847, 2...
## $ good_and_services_index <dbl> 77.0, 66.1, 76.7, 55.3, 74.7, 60.3, 72.3, 5...
## $ food_index              <dbl> 66.0, 70.7, 90.0, 76.5, 71.3, 83.9, 70.5, 6...
```

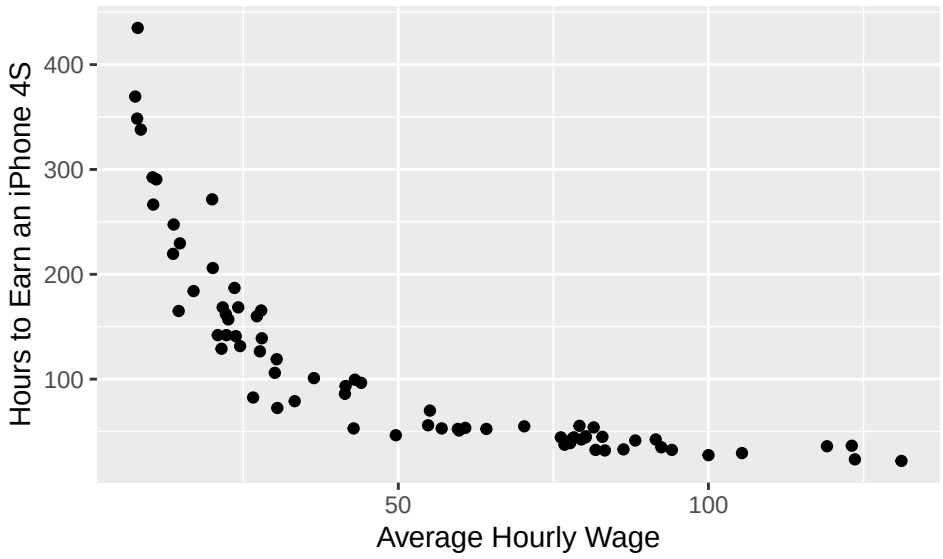
Here we use the `clean_names()` function from the `janitor` package to sanitize the names of the columns (which would otherwise contain special characters or whitespace).

The `names()` function displays the names while the `glimpse()` function provides more detail about the dataset.

```
gf_point(food_costs ~ womens_clothing, data = Prices) %>%
  gf_labs(x = "Cost of Women's Clothes", y = "Food Costs ($)")
```

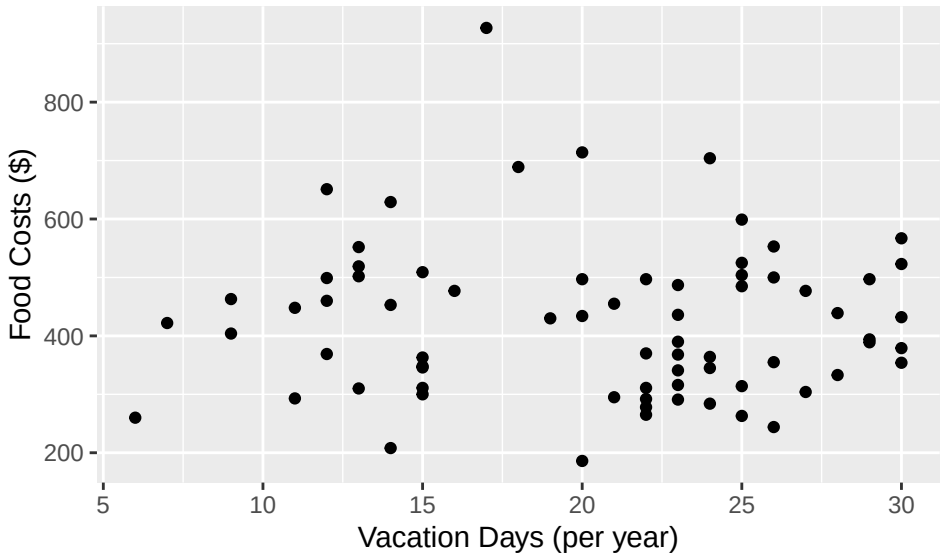



```
gf_point(i_phone_4s_hr ~ wage_gross, data = Prices) %>%
  gf_labs(x = "Average Hourly Wage", y = "Hours to Earn an iPhone 4S")
```




```
gf_point(clothing_index ~ hours_worked, data = Prices) %>%
  gf_labs(x = "Working Hours", y = "Clothes Index")
```

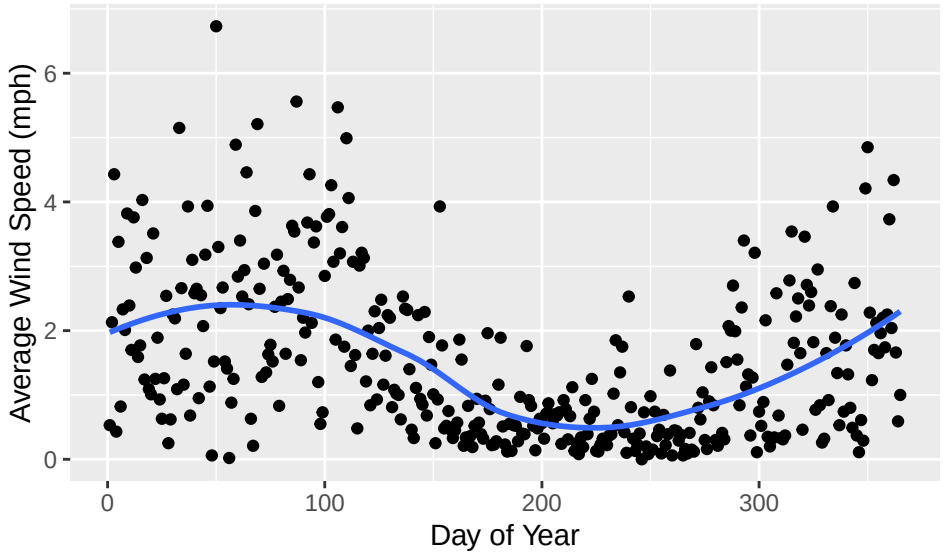

```
gf_point(food_costs ~ vacation_days, data = Prices) %>%
  gf_labs(x = "Vacation Days (per year)", y = "Food Costs ($)")
```



Roles for Variables

Smoothing Scatterplots We begin by reading in the data.

```
HopkinsForest <- read_csv("http://nhorton.people.amherst.edu/is5/data/Hopkins_Forest.csv") %>%
  janitor::clean_names()
# Figure 6.2, page 168
gf_point(avg_wind_mph ~ day_of_year, data = HopkinsForest) %>%
  gf_smooth(se = FALSE) %>%
  gf_labs(x = "Day of Year", y = "Average Wind Speed (mph)")
```

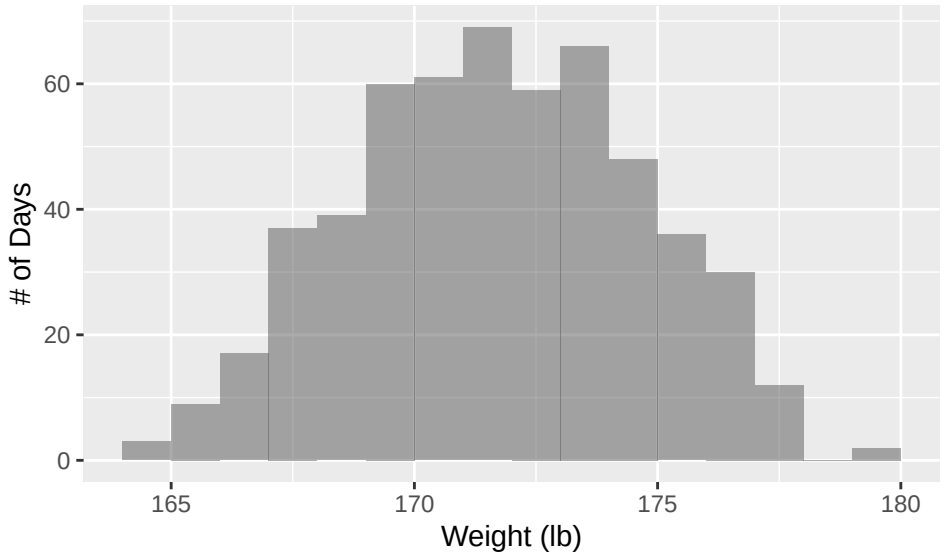


The smoother warning messages provided by `gf_smooth()` have been removed from this output using the `warning = FALSE` code chunk option.

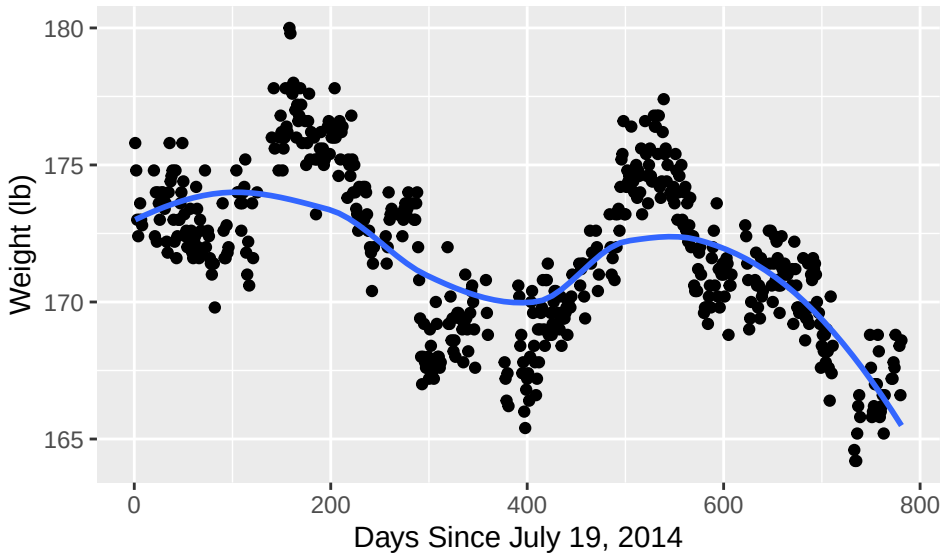
Example 6.2: Smoothing Timeplots We will explore smoothing using the fitness data.

```
Fitness <- read_csv("http://nhorton.people.amherst.edu/is5/data/Fitness_data.csv") %>%
  janitor::clean_names()
```

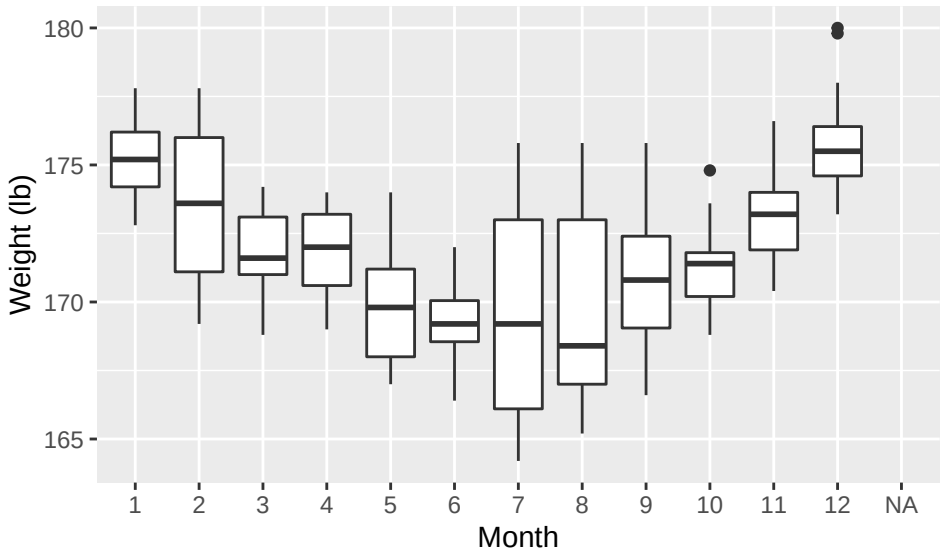
```
gf_histogram(~weight, data = Fitness, binwidth = 1, center = .5) %>%
  gf_labs(x = "Weight (lb)", y = "# of Days")
```




```
gf_point(weight ~ days_since_july_19_2014, data = Fitness) %>%
  gf_smooth(se = FALSE) %>%
  gf_labs(x = "Days Since July 19, 2014", y = "Weight (lb)")
```




```
gf_boxplot(weight ~ as.factor(month), data = Fitness) %>%
  gf_labs(x = "Month", y = "Weight (lb)")
```



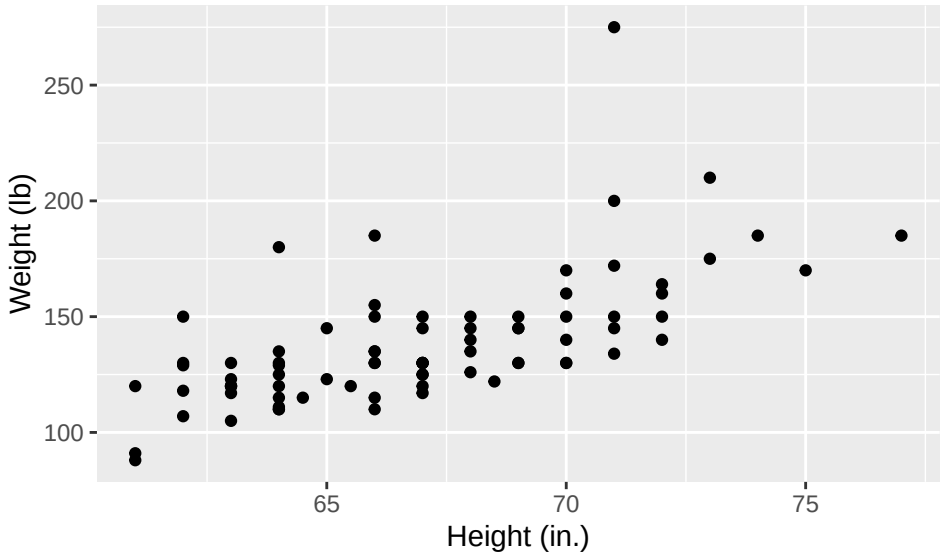
Warnings should be suppressed for your work once you have determined that they are innocuous.

Section 6.2: Correlation

We begin by reading in the data.

```
HeightsWeights <- read_csv("http://nhorton.people.amherst.edu/is5/data/Heights_and_weights.csv")

# Figure 6.3, page 170
gf_point(Weight ~ Height, data = HeightsWeights) %>%
  gf_labs(x = "Height (in.)", y = "Weight (lb)")
```



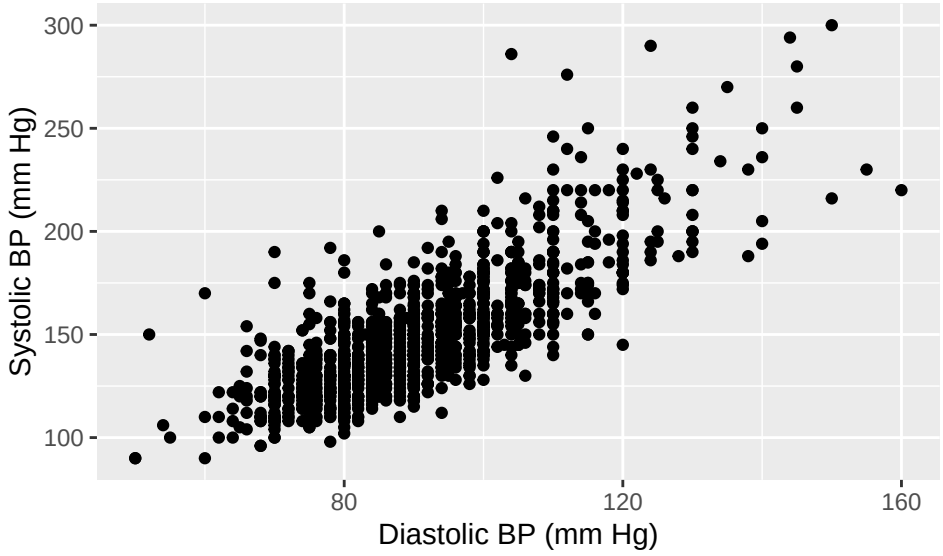

```
cor(Weight ~ Height, data = HeightsWeights)
```

```
## [1] 0.6440311
```

See displays on pages 170 - 171.

Step-by-Step Example: Looking at Association We begin by loading the Framingham data.

```
Framingham <- read_csv("http://nhorton.people.amherst.edu/is5/data/Framingham.csv")
gf_point(SBP ~ DBP, data = Framingham) %>%
  gf_labs(x = "Diastolic BP (mm Hg)", y = "Systolic BP (mm Hg)")
```

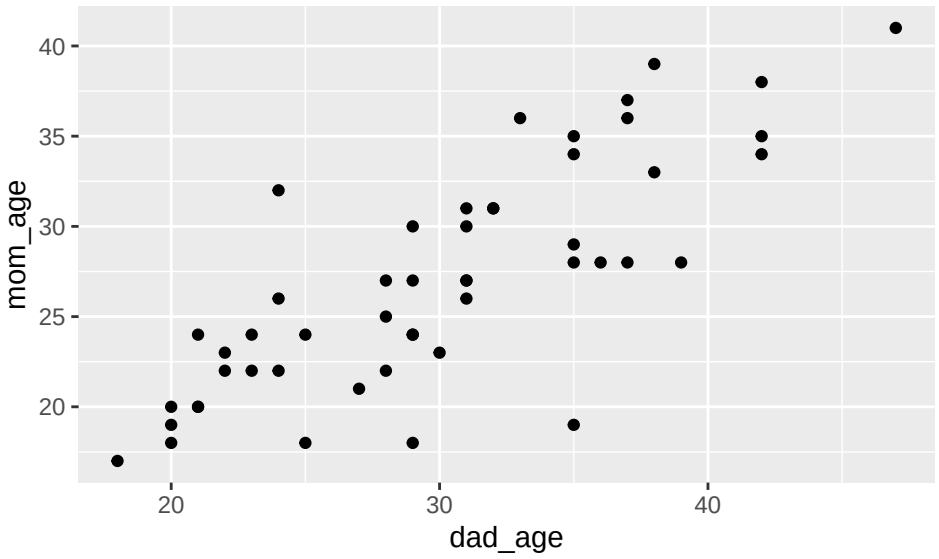



```
cor(SBP ~ DBP, data = Framingham)
```

```
## [1] 0.7924792
```

Random Matters: Correlations Vary A recurring theme in the course involves random sampling of various sorts.
Here we explore a sample of babies born in 1998.

```
LiveBirths <- read_csv("http://nhorton.people.amherst.edu/is5/data/Babysamp_98.csv") %>%
  janitor::clean_names()
LiveBirths <- LiveBirths %>%
  filter(dad_age != "NA")
set.seed(14513) # To ensure we get the same values when we run it multiple times
numsim <- 10000 # Number of samples
gf_point(mom_age ~ dad_age, data = sample(LiveBirths, size = 50))
```




```
# Graph will look different for different samples
cor(mom_age ~ dad_age, data = LiveBirths)
```

```
## [1] 0.7516507
```

```
# What does mosaic::do() do?
cor(mom_age ~ dad_age, data = sample(LiveBirths, size = 50)) # Correlation of one random sample
```

```
## [1] 0.7596176
```

```
cor(mom_age ~ dad_age, data = sample(LiveBirths, size = 50)) # Correlation of another random sample
```

```
## [1] 0.8087199
```

```
do(2) * cor(mom_age ~ dad_age, data = sample(LiveBirths, size = 50)) # Finds the correlation twice
```

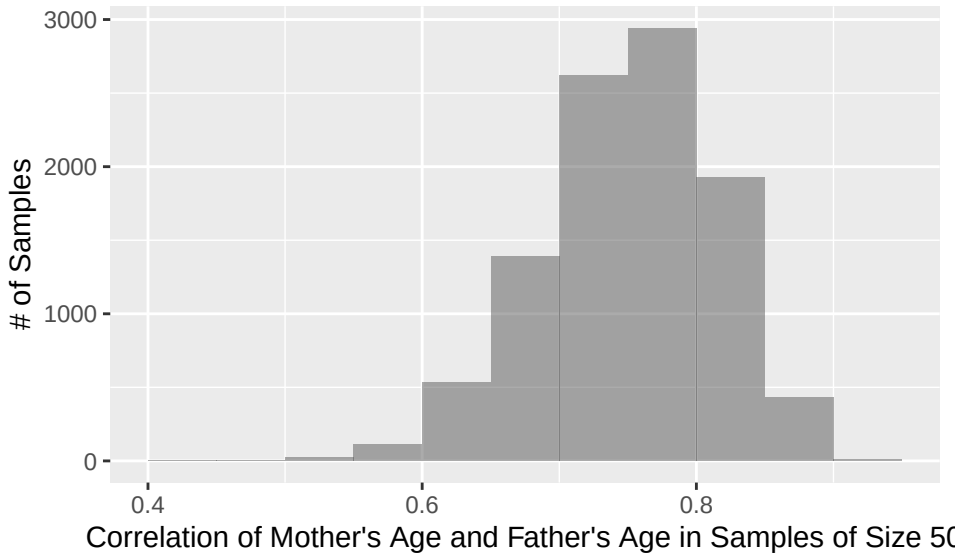
```
##         cor
## 1 0.7447461
## 2 0.7583890
```

```
# For the visualization, we need 10,000 correlations
LiveCorr <- do(numsim) * cor(mom_age ~ dad_age, data = sample(LiveBirths, size = 50))
```

The `do()` function runs, 10,000 times, the correlation and sampling functions on a random sample of 50.

(We can use the chunk option `cache=TRUE` to enable caching to save results for next time.)

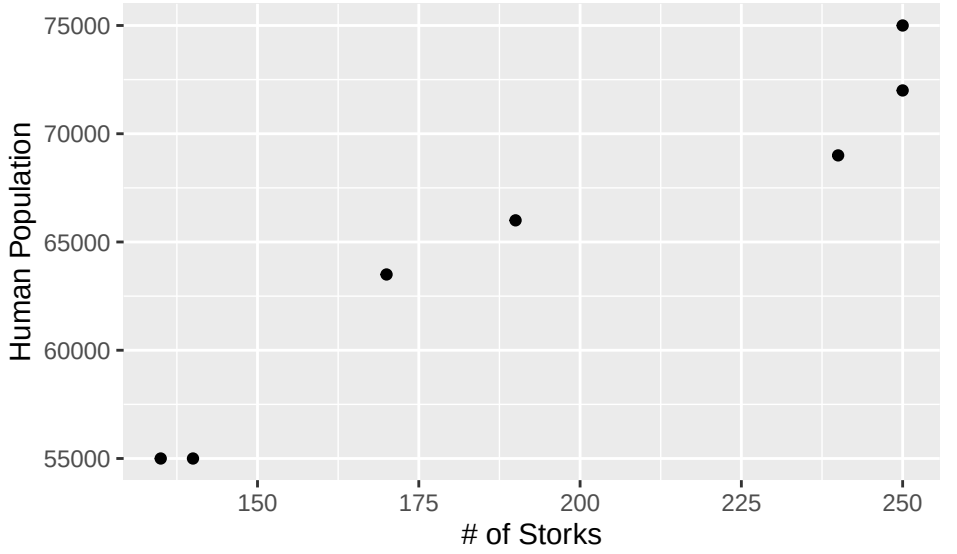
```
# Figure 6.8, page 176
gf_histogram(~cor, data = LiveCorr, binwidth = .05, center = .025) %>%
  gf_labs(
    x = "Correlation of Mother's Age and Father's Age in Samples of Size 50",
    y = "# of Samples"
  )
```



Section 6.3: Warning: Correlation $\neq$ Causation

The storks data is a classic example of how correlation does not always imply causation.

```
Storks <- read_csv("http://nhorton.people.amherst.edu/is5/data/Storks.csv")
# Figure 6.9
gf_point(Population ~ Storks, data = Storks) %>%
  gf_labs(x = "# of Storks", y = "Human Population")
```



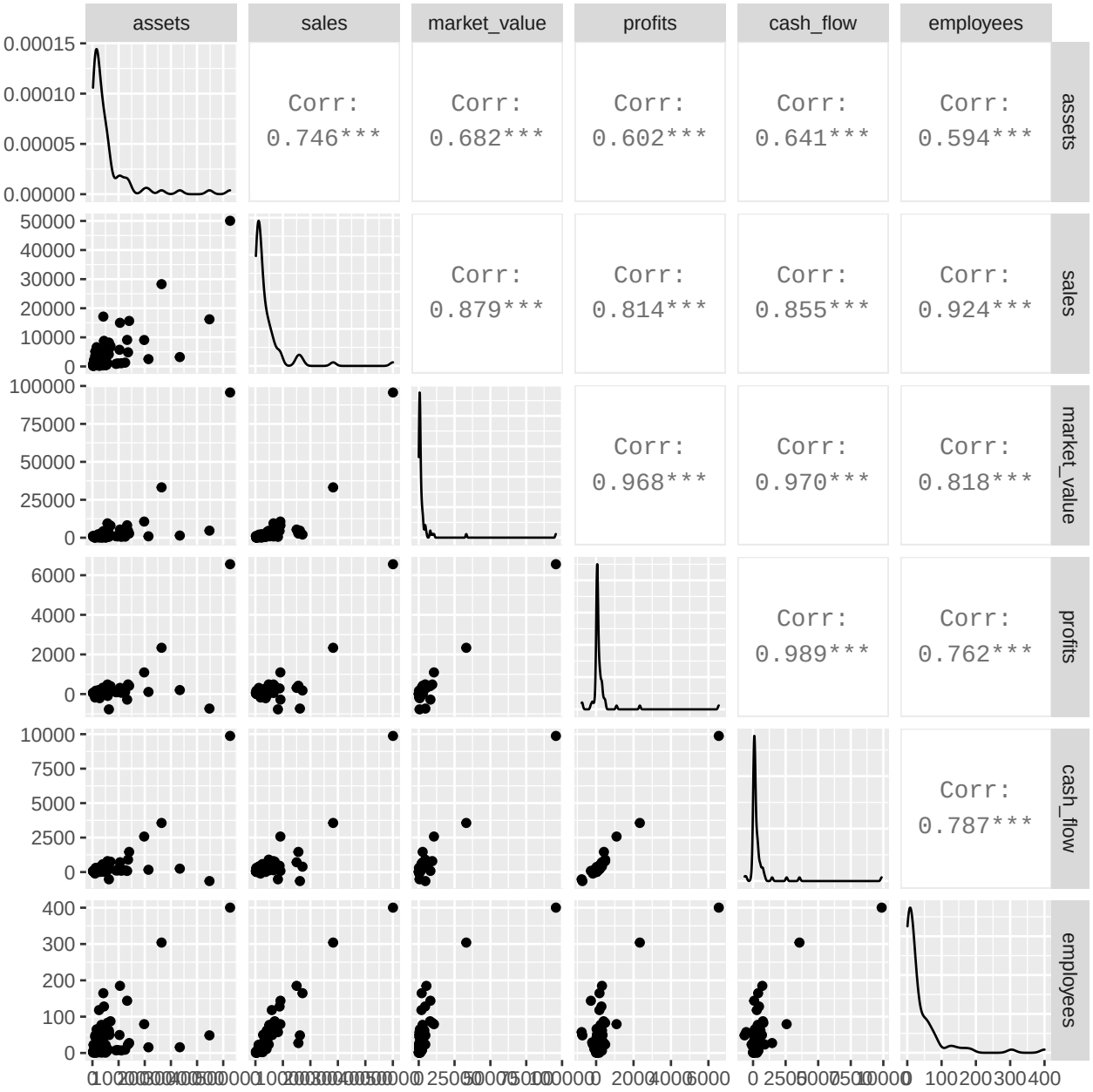
Correlation Tables We can display correlation tables as seen in Table 6.1 on page 178.

```
Companies <- read_csv("http://nhorton.people.amherst.edu/is5/data/Companies.csv") %>%
  janitor::clean_names()
# Table 6.1, page 178
Companies %>%
  select(assets, sales, market_value, profits, cash_flow, employees) %>%
  cor()
```

```
##                assets     sales market_value   profits cash_flow employees
```

```
## assets       1.0000000 0.7464649    0.6822122 0.6016986 0.6409018 0.5943581
## sales        0.7464649 1.0000000    0.8788920 0.8137758 0.8549172 0.9240429
## market_value 0.6822122 0.8788920    1.0000000 0.9681987 0.9702851 0.8182161
## profits      0.6016986 0.8137758    0.9681987 1.0000000 0.9887795 0.7621057
## cash_flow    0.6409018 0.8549172    0.9702851 0.9887795 1.0000000 0.7866148
## employees    0.5943581 0.9240429    0.8182161 0.7621057 0.7866148 1.0000000
```

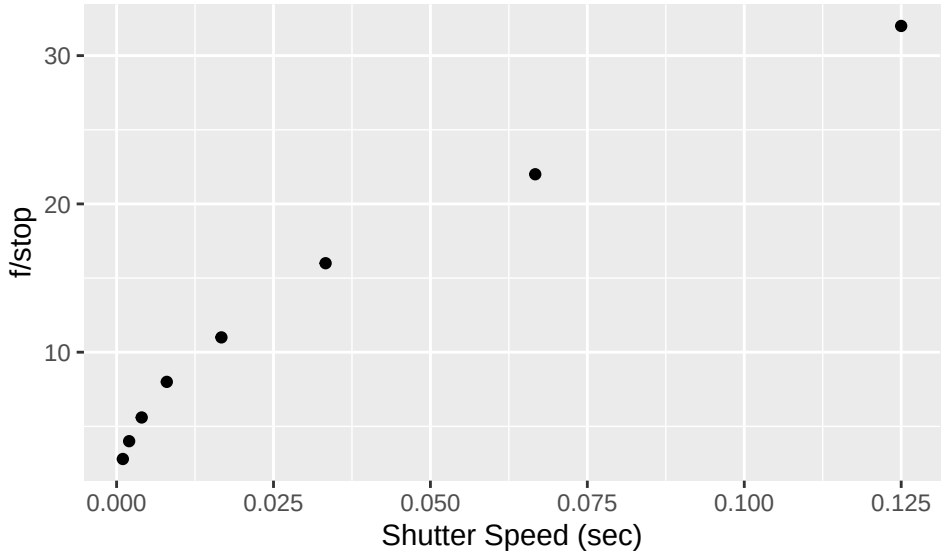
```
Companies %>%
  select(assets, sales, market_value, profits, cash_flow, employees) %>%
  GGally::ggpairs()
```



Section 6.4: Straightening Scatterplots

It's often possible to straightend scatterplots through use of a transformation.

```
FStops <- read_csv("http://nhorton.people.amherst.edu/is5/data/F-stops.csv") %>%
  janitor::clean_names()
# Figure 6.10, page 179
gf_point(f_stop ~ shutter_speed, data = FStops) %>%
  gf_labs(x = "Shutter Speed (sec)", y = "f/stop")
```



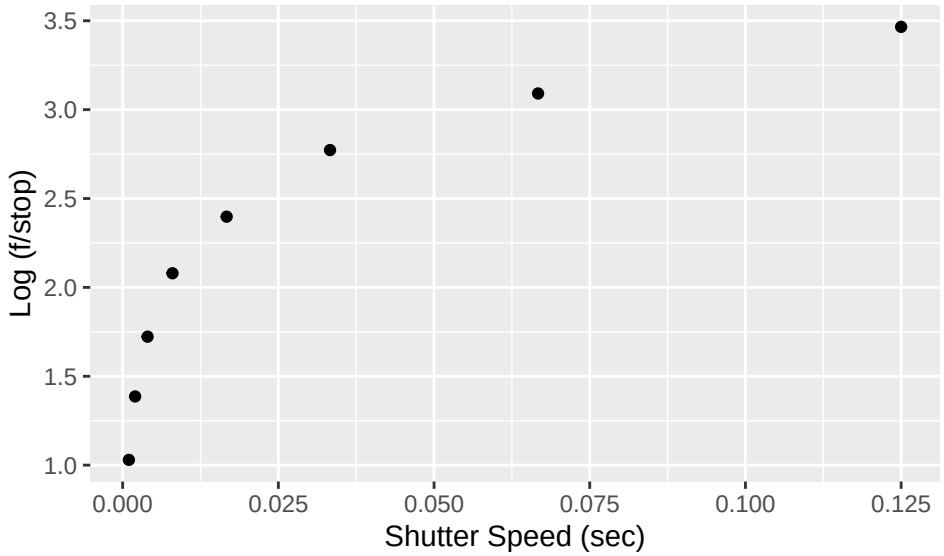

```
cor(f_stop ~ shutter_speed, data = FStops)
```

```
## [1] 0.9786716
```

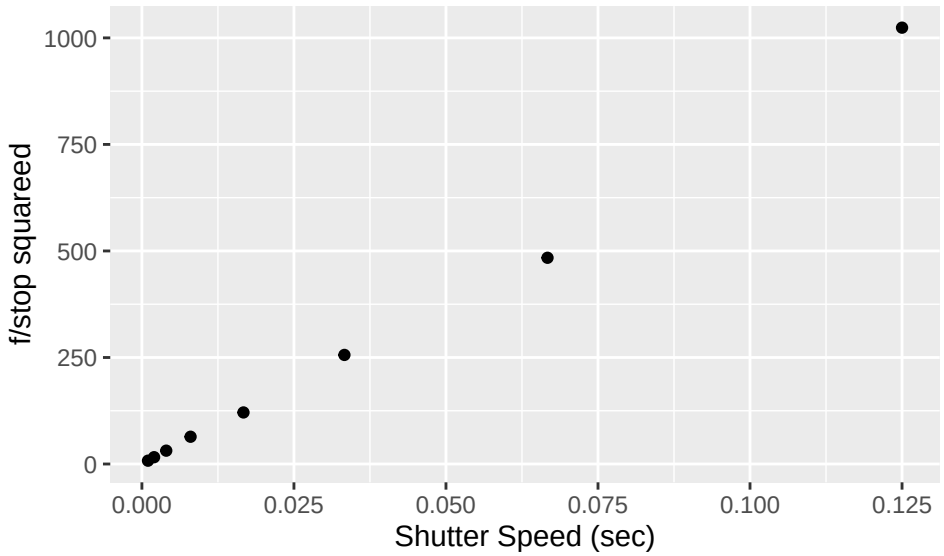
The Ladder of Powers

f/Stops Again The f/Stops example is reviewed on page 181 (Figure 6.11)

```
gf_point(log(f_stop) ~ shutter_speed, data = FStops) %>%
  gf_labs(x = "Shutter Speed (sec)", y = "Log (f/stop)")
```



```
# Figure 6.12
gf_point((f_stop)^2 ~ shutter_speed, data = FStops) %>%
  gf_labs(x = "Shutter Speed (sec)", y = "f/stop squareed")
```



See the displays in "What Can Go Wrong?" on pages 181-183.