Julia Vineyard

Worked with: Andrew Gordon

Lab 4: Uncertainty and Error

1.
   ```
   norm_mean=10.4
   norm_sd=2.4

   norm_17=rnorm(17, mean=norm_mean, sd=norm_sd)
   norm_30=rnorm(30, mean=norm_mean, sd=norm_sd)
   norm_300=rnorm(300, mean=norm_mean, sd=norm_sd)
   norm_3000=rnorm(3000, mean=norm_mean, sd=norm_sd)
   ```

2.
   ```
   require(here)
   png(
     filename=here("images", "lab_04_hist_01.png"),
       width = 1500, height = 1600,
       res=180)

   par(mfrow=c(2,2))
   hist(norm_17, main="17 Random Points")
   hist(norm_30, main="30 Random Points")
   hist(norm_300, main="300 Random Points")
   hist(norm_3000, main="3000 Random Points")

   dev.off()
   ```

3. Uploaded file to Moodle
4. The histogram of 17 random points is the most straight across, and the 3000 random points histogram shows the most normal distribution. While each of these are representative of a normal distribution the 17 random points looks least like a normal distribution, while the 3000 random points is the closest to a normal distribution.
5. The histograms are different because as you add more random points to the sample the sampling error is reduced and the histograms more closely resemble a normal distribution.
6. A standard normal distribution has 2 parameters, mean which is equal to 0 and standard deviation which is equal to 1.
7.
   ```
    require(here)
   svg(
     filename=here("images", "norm_1.svg"),
     width=7, height=7, pointsize=12)
   x = seq(0, 20, length.out = 1000)
   y = dnorm(x, mean=10.4, sd=2.4)
   ```

```
plot(x, y, main = "Standard Normal PDF \nMean=10.4 SD=2.4", type = "l", xlim = c(0, 20))
abline(h = 0)

dev.off()
```

8. Uploaded file to Moodle
9.
   ```
   set.seed(2)
   n_pts = 8
   x_min = 0
   x_max = 24
   x = runif(n = n_pts, min = x_min, max = x_max)
   dat_1 = data.frame(x = x, y_observed = rnorm(n_pts))
   ```

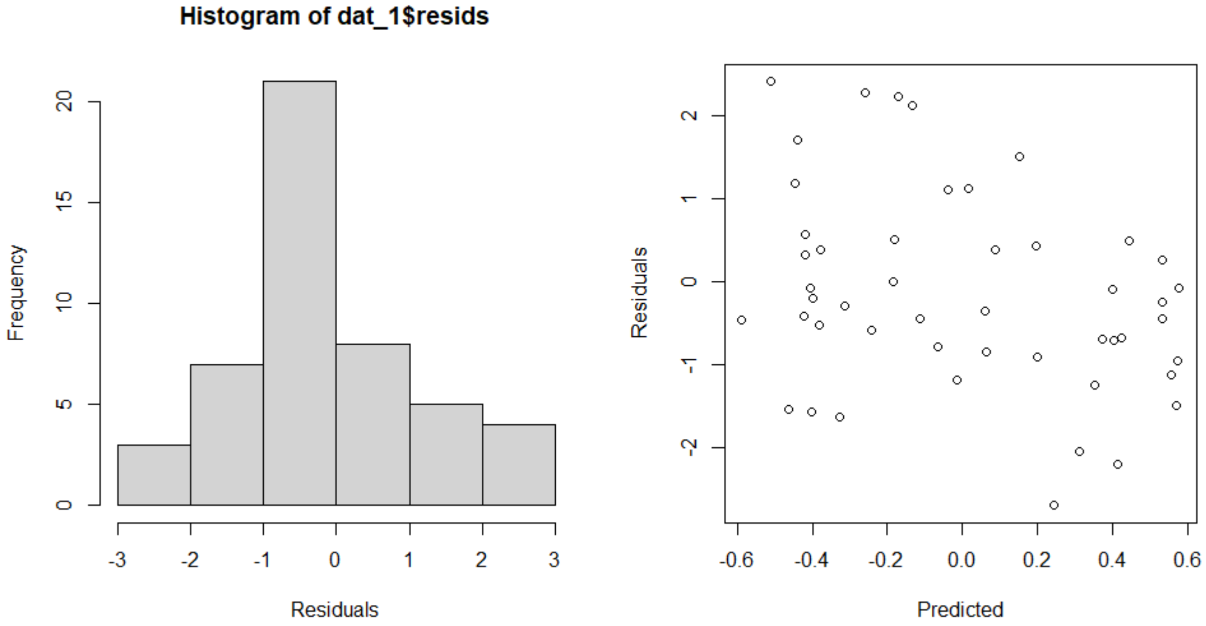
10. Uploaded file to Moodle
11.
    ```
    set.seed(2)
    n_pts = 48
    x_min = 0
    x_max = 24
    x = runif(n = n_pts, min = x_min, max = x_max)
    dat_1 = data.frame(x = x, y_observed = rnorm(n_pts))
    line_point_slope = function(x, x1, y1, slope)
    {
      get_y_intercept =
        function(x1, y1, slope)
          return(-(x1 * slope) + y1)

      linear =
        function(x, yint, slope)
          return(yint + x * slope)

      return(linear(x, get_y_intercept(x1, y1, slope), slope))
    }
    plot(dat_1)
    curve(line_point_slope(x, 12, 0, 0.05), add=TRUE)
    ```
12. Uploaded file to Moodle
13.
    ```
    set.seed(2)
    n_pts = 48
    x_min = 0
    x_max = 24
    x = runif(n = n_pts, min = x_min, max = x_max)
    dat_1 = data.frame(x = x, y_observed = rnorm(n_pts))
    y_predicted_vec=line_point_slope(dat_1$x, 12, 0, 0.05)
    dat_1$y_predicted=y_predicted_vec
    ```

```
resid_vec=dat_1$y_observed - dat_1$y_predicted
dat_1$resids=resid_vec
dat_1
```

14.


Histogram of dat_1$resids
Frequency
Residuals
Residuals
Predicted