Keegan Moynahan

Lab 4

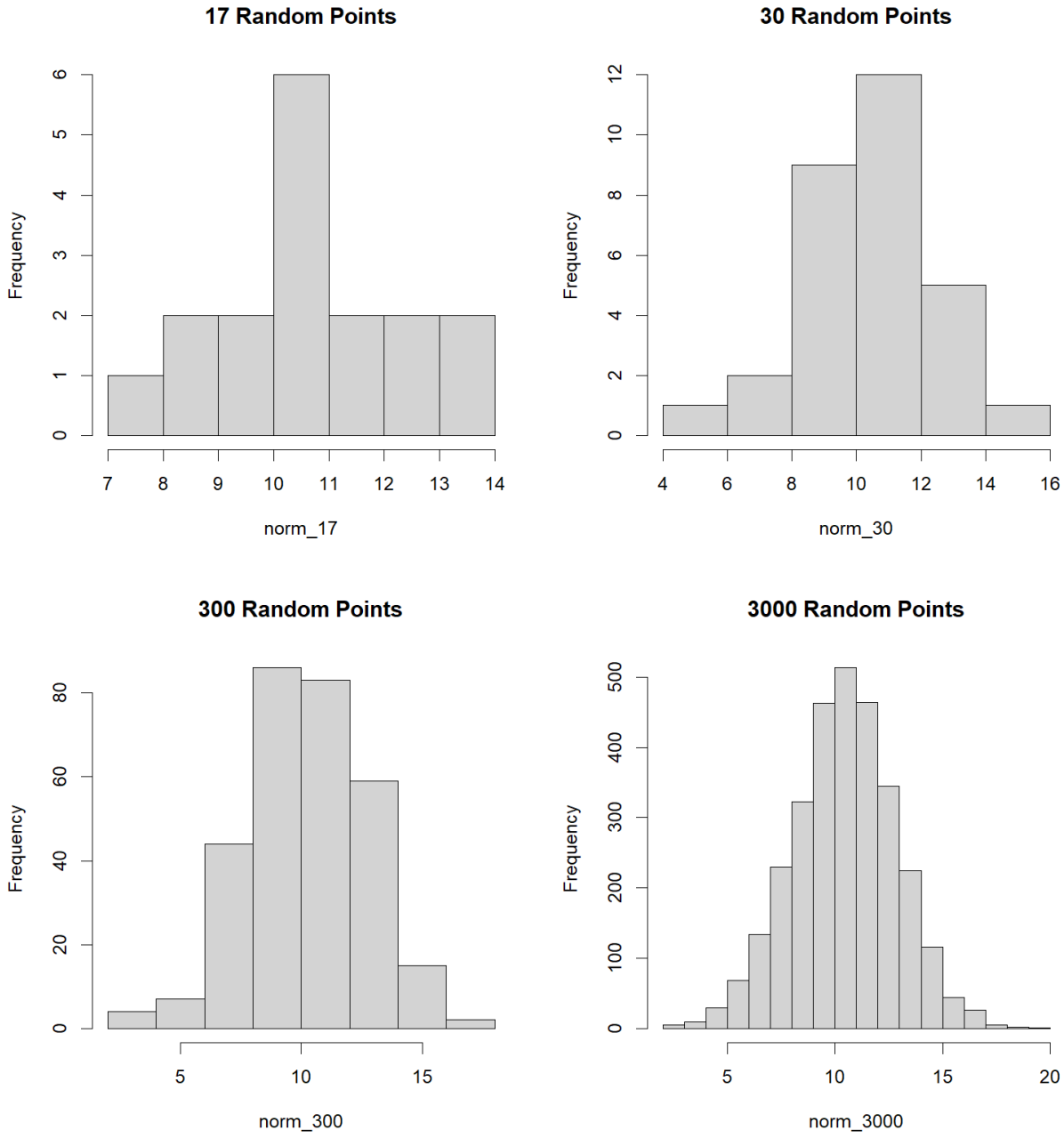
1)

```
norm_17 = rnorm(n = 17, mean = 10.4, sd = 2.4)
norm_30 = rnorm(n = 30, mean = 10.4, sd = 2.4)
norm_300 = rnorm(n = 300, mean = 10.4, sd = 2.4)
norm_3000 = rnorm(n = 3000, mean = 10.4, sd = 2.4)
```

2)

```
require(here)
png(
 filename = here("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)



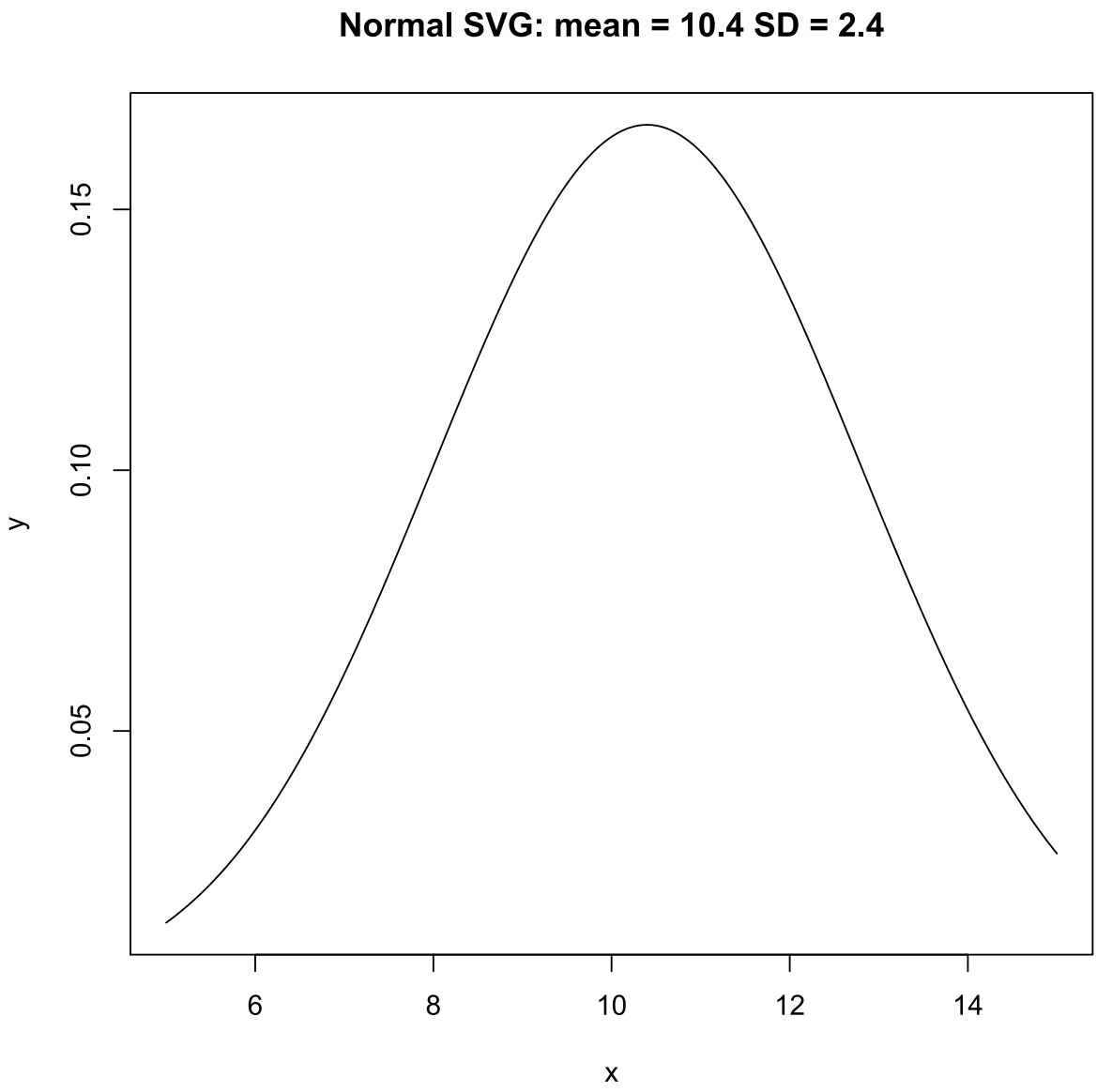
4) The first histogram (17 random points) looks semi uniform except for the 4th bar. This histogram is a mix of uniform and random. The numbers all seem to be somewhat evenly distributed, with the extreme being the 4th bar. The second histogram (30 random points) is unimodal, most of the random numbers generated in this histogram where between 8 and 12. The third histogram (300 random points) most of the random points are centered around the middle of the histogram. This graph has extreme low frequencies (tails) and extreme highs

(middle). The last histogram in unimodal and very close to a bell shape. This is the most evenly distributed histogram.

5) All the histograms are different because the sample sizes are different. The random numbers are going to have different values in each case because of the two criteria each histogram had to be met.
6) The parameters of the standard normal are the mean and the standard deviation. The mean equals 0 and the standard deviation equals 1.
7)

```
svg("norm_1.svg")
x = seq(5, 15, length.out = 1000)
y = dnorm(x, mean = 10.4, sd = 2.4)
plot(x, y, main = "Normal SVG: mean = 10.4 SD = 2.4", type = "l", xlim = c(5, 15))
abline(h = 0)
dev.off()
```

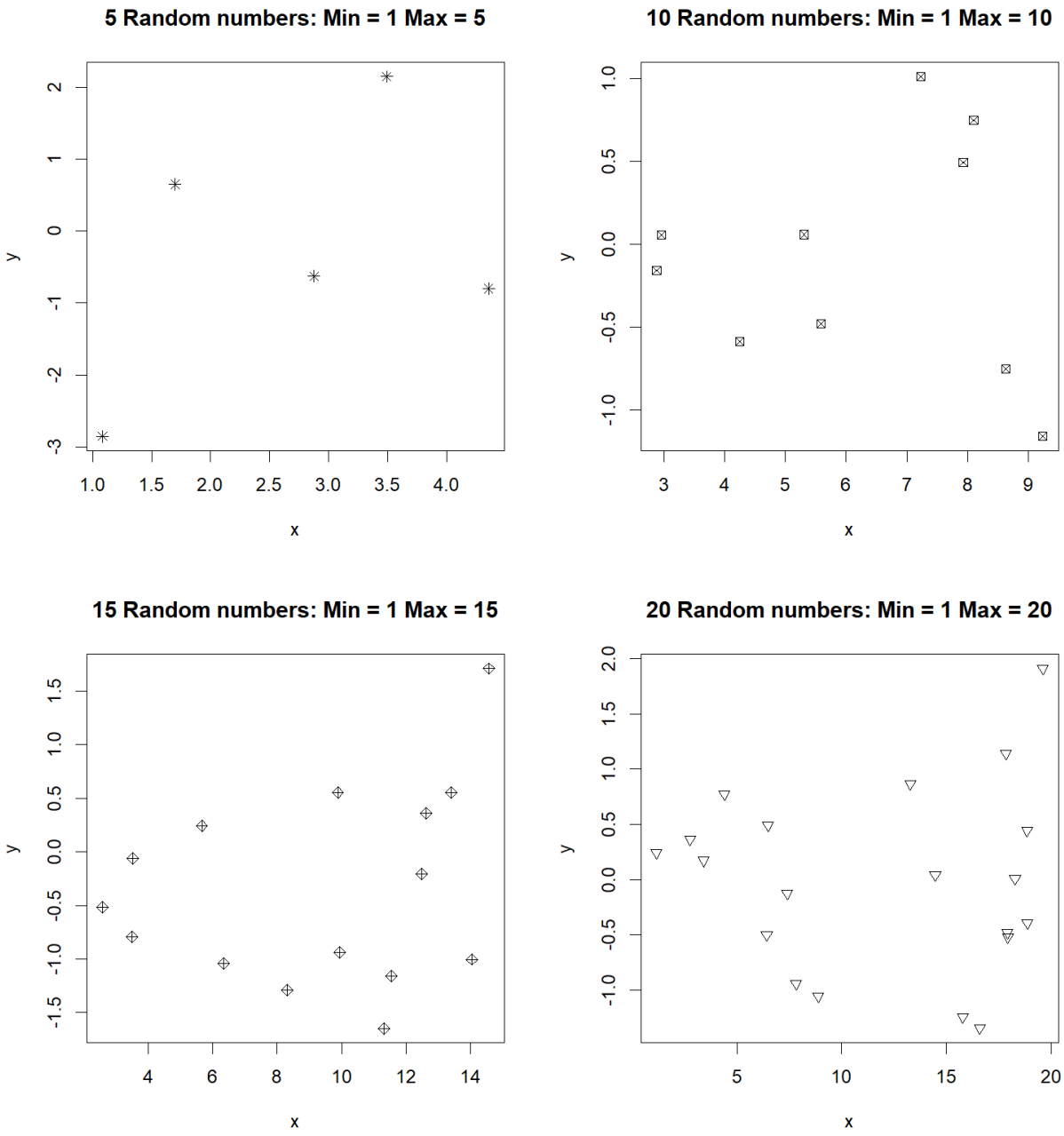
8)



9) This created the first dataset (shown in plot 1)

```
x_random = runif(n = 5, min = 1, max = 5)
y_random = rnorm(n = 5)
dat_random1 = data.frame(x = x_random, y = y_random)
```

10)



11)

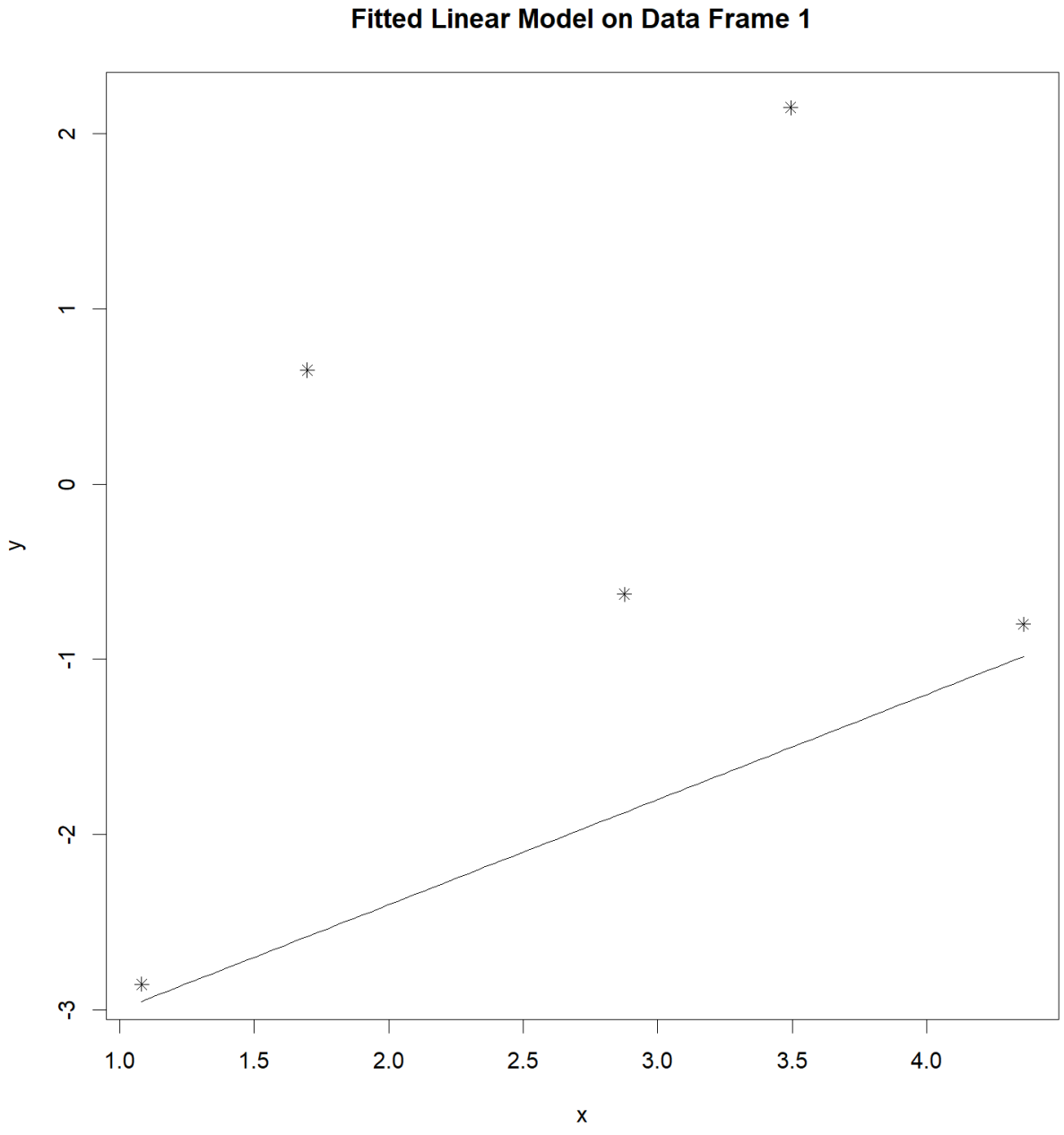
```
require(here)
png(
  filename = here("Fitted_Linear_Model.png"),
  width = 1500, height = 1600,
  res = 180)
```

plot(y ~ x, data = dat_random1, pch = 8)

curve(line_point_slope(x, 6, 0, 0.6), add = T)

dev.off()

12)



13)

dat_random1$predicted = line_point_slope(dat_random1$x, 6, 0, 0.6)

dat_random1$residuals = dat_random1$y - dat_random1$predicted

14)