M1L2 Homework Assignment

Joshua Conte

September 17, 2017

1 M1L2 Homework Assignment

R studio was configured with the following parameters before beginning the project:

```
# clears the console in RStudio
cat("\014")
```

```
# clears environment
rm(list = ls())

# Set working directory
setwd("C:/R/DA5030/module_01")

# Load required packages
require(ggplot2)
require(reshape2)
library(knitr) # Used for making tables in this report
```

1.1 Part 1: Cauchy distributions

From Wikipedia, the Cauchy distribution, named after Augustin Cauchy, is a continuous probability distribution. It is also known, especially among physicists, as the Lorentz distribution (after Hendrik Lorentz), Cauchy-Lorentz distribution, Lorentz(ian) function, or Breit-Wigner distribution. The Cauchy distribution is the distribution of the x-intercept of a ray issuing from with a uniformly distributed angle. It is also the distribution of the ratio of two independent normally distributed random variables if the denominator distribution has mean zero.

1.1.1 Cauchy Probability Density Function

I used the Cauchy distribution feature in R. dcauchy is used for density location is x0 and scale is gamma. I used the same location and scale parameters in the Wikipedia example.

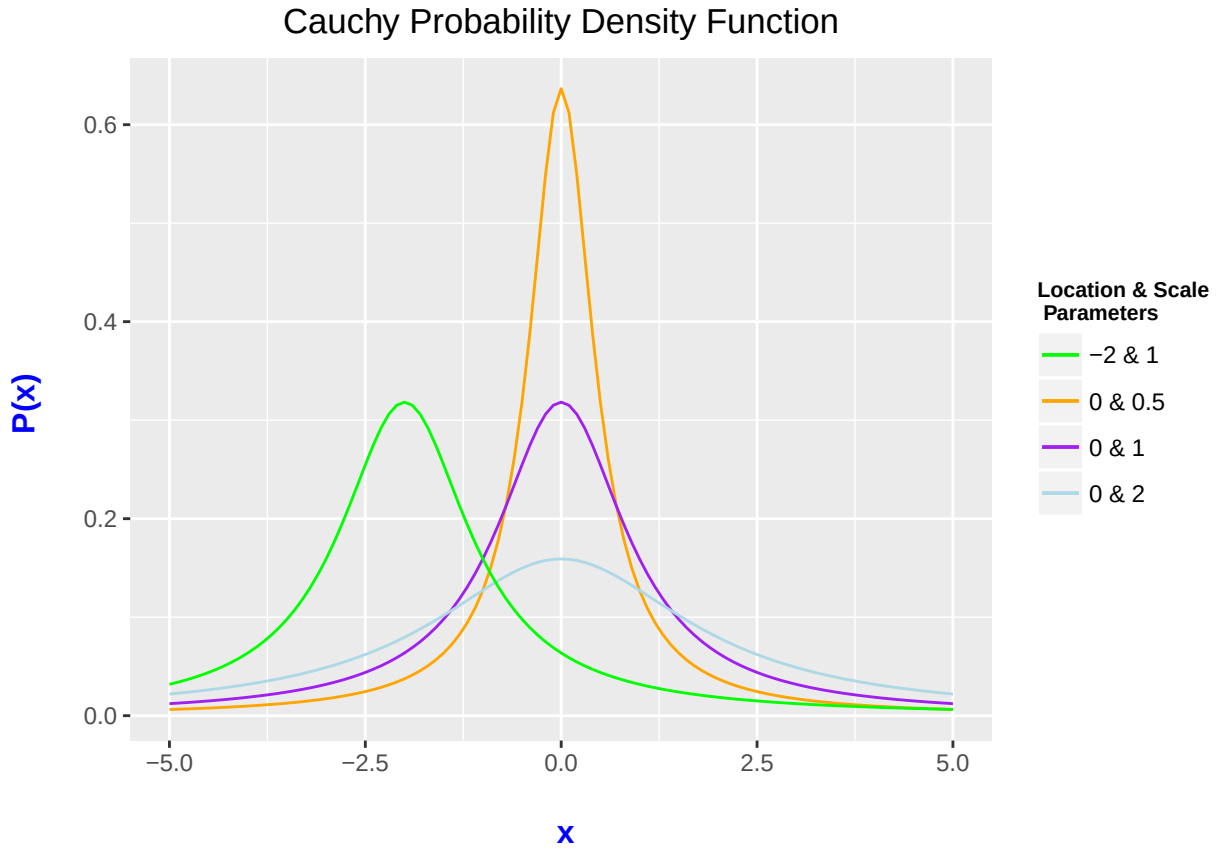
For the ggplot info, I used lower limit of -5 and upper of 5. I used fun and args for the function and aes for aesthetic mappings. I added theme to make the plot more aesthetic too.

```
# Cauchy Probability Density Function:

x_lower <- -5
x_upper <- 5

ggplot(data.frame(x = c(x_lower , x_upper)), aes(x = x)) +
  xlim(c(x_lower , x_upper)) +
  stat_function(fun = dcauchy,
                args = list(location = 0, scale = 0.5),
                aes(colour = "0 & 0.5")) +
  stat_function(fun = dcauchy,
                args = list(location = 0, scale = 1),
                aes(colour = "0 & 1")) +
  stat_function(fun = dcauchy,
                args = list(location = 0, scale = 2),
                aes(colour = "0 & 2")) +
  stat_function(fun = dcauchy,
                args = list(location = -2, scale = 1),
                aes(colour = "-2 & 1")) +
```

```
scale_color_manual("Location & Scale \n Parameters",
                   values = c("green", "orange", "purple", "light blue")) +
labs(x = "\n x", y = "P(x) \n",
     title = "Cauchy Probability Density Function") +
theme(
  plot.title = element_text(hjust = 0.5),
  axis.title.x = element_text(
    face = "bold",
    colour = "blue",
    size = 12
  ),
  axis.title.y = element_text(
    face = "bold",
    colour = "blue",
    size = 12
  ),
  legend.title = element_text(face = "bold", size = 8),
  legend.position = "right"
)
```



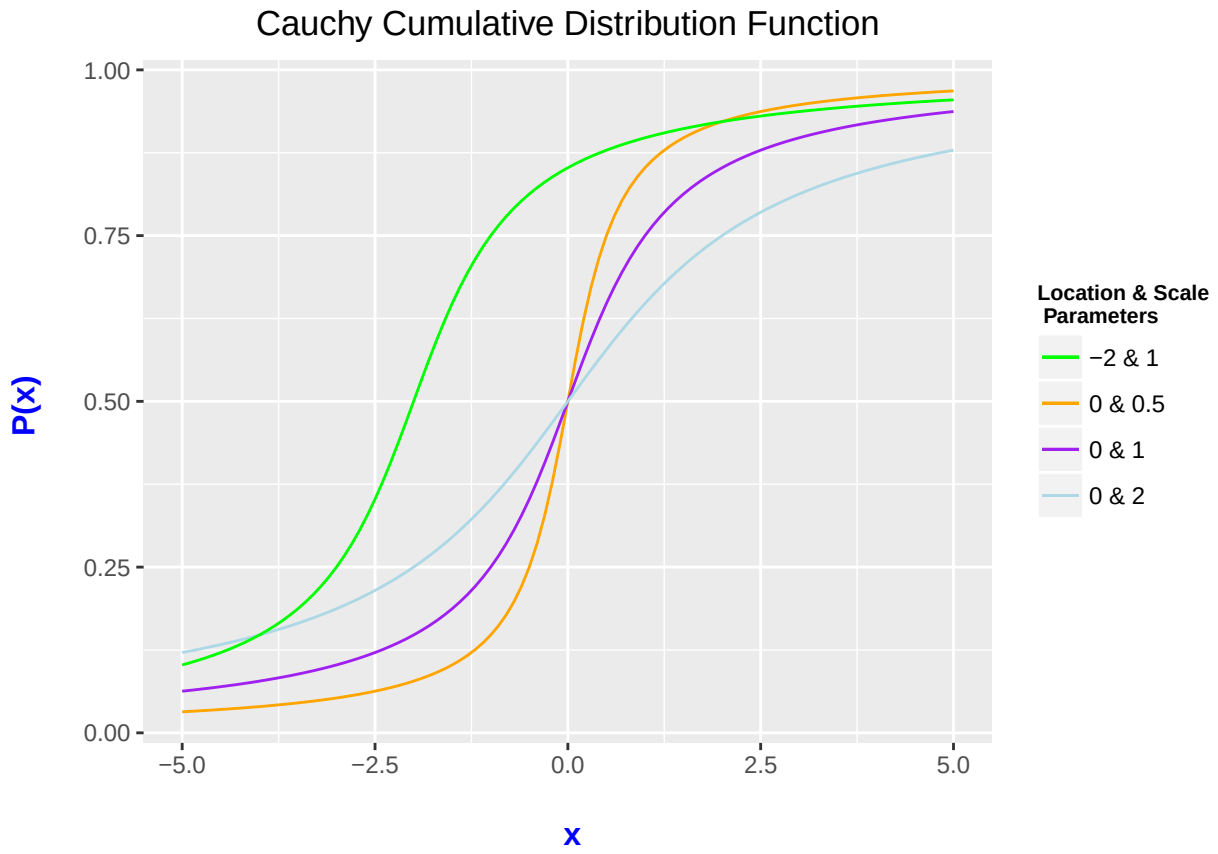
1.1.2 Cauchy Cumulative Distribution Function

This is the same as above but pcauchy is used for distribution function.

```r
# Cauchy Cumulative Distribution Function:

x_lower <- -5
x_upper <- 5

ggplot(data.frame(x = c(x_lower , x_upper)), aes(x = x)) +
  xlim(c(x_lower , x_upper)) +
  stat_function(fun = pcauchy,
                args = list(location = 0, scale = 0.5),
                aes(colour = "0 & 0.5")) +
  stat_function(fun = pcauchy,
                args = list(location = 0, scale = 1),
                aes(colour = "0 & 1")) +
  stat_function(fun = pcauchy,
                args = list(location = 0, scale = 2),
                aes(colour = "0 & 2")) +
  stat_function(fun = pcauchy,
                args = list(location = -2, scale = 1),
                aes(colour = "-2 & 1")) +
  scale_color_manual("Location & Scale \n Parameters",
                     values = c("green", "orange", "purple", "light blue")) +
  labs(x = "\n x", y = "P(x) \n",
       title = "Cauchy Cumulative Distribution Function") +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(
      face = "bold",
      colour = "blue",
      size = 12
    ),
    axis.title.y = element_text(
      face = "bold",
      colour = "blue",
      size = 12
    ),
    legend.title = element_text(face = "bold", size = 8),
    legend.position = "right"
  )
```



1.2 Part 2: M01_Lesson_02_Q1

1.2.1 load the file M01_Lesson_02_Q1.csv.

I opened the data using read.csv2 and removed the first column since it appears to be row numbers. I also changed the data to numeric so I could analyze it with R.

Below is my R code:

```
# Some VCF files are really big and take a while to open.  This command checks to
# see if it is already opened, if it is, it does not open it again.
# I also omitted the first column
if (!exists("csv.df")) {
  csv.df <-
    read.csv2(
      'M01_Lesson_02_Q1.csv',
      sep = ",",
      stringsAsFactors = FALSE,
      row.names = NULL,
      header = TRUE,
      colClasses = c("NULL", NA, NA, NA, NA, NA)
    )
}

# Check to make sure the data is in numeric form for analysis:
sapply(csv.df, class)
```

```
##           A           B           C           D           E
## "character" "character"   "integer"   "integer" "character"
```

```
# It looks like the data is in character and integer form and I will change it all
# to numeric for easy analysis:
df1 <-
  data.frame(sapply(csv.df, function(x)
    as.numeric(as.character(x))))

# Check to make sure it is in numeric form:
sapply(df1, class)
```

```
##         A         B         C         D         E
## "numeric" "numeric" "numeric" "numeric" "numeric"
```

1.2.2 How is the data distributed?

I concluded that:

- column A is Normal
- column B is Normal
- column C is Poisson
- column D is Poisson
- column E is Uniform

My analysis is listed below:

To understand how the data is distributed, I needed to review the data and plot it too. I began with a Shapiro test to see if the P value was greater than 0.05 to confirm that we cannot reject the hypothesis that the sample comes from a population which has a normal distribution. Then use QQ plots to verify the results:

```
# I began by running a Shapiro-Wilk test:
shapiro.test(df1$A)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df1$A
## W = 0.99646, p-value = 0.6722
```

```
shapiro.test(df1$B)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df1$B
## W = 0.99749, p-value = 0.8964
```

```
shapiro.test(df1$C)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df1$C
## W = 0.94753, p-value = 1.673e-09
```

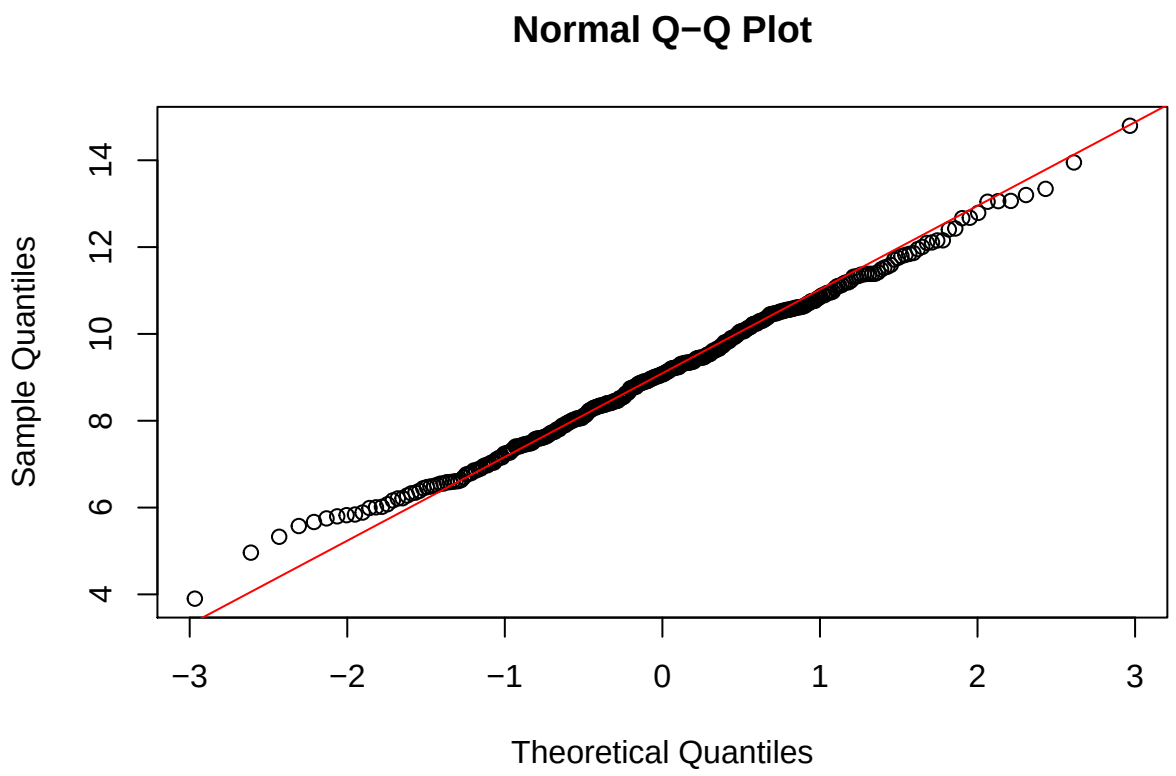
```
shapiro.test(df1$D)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df1$D
## W = 0.97474, p-value = 1.402e-05
```

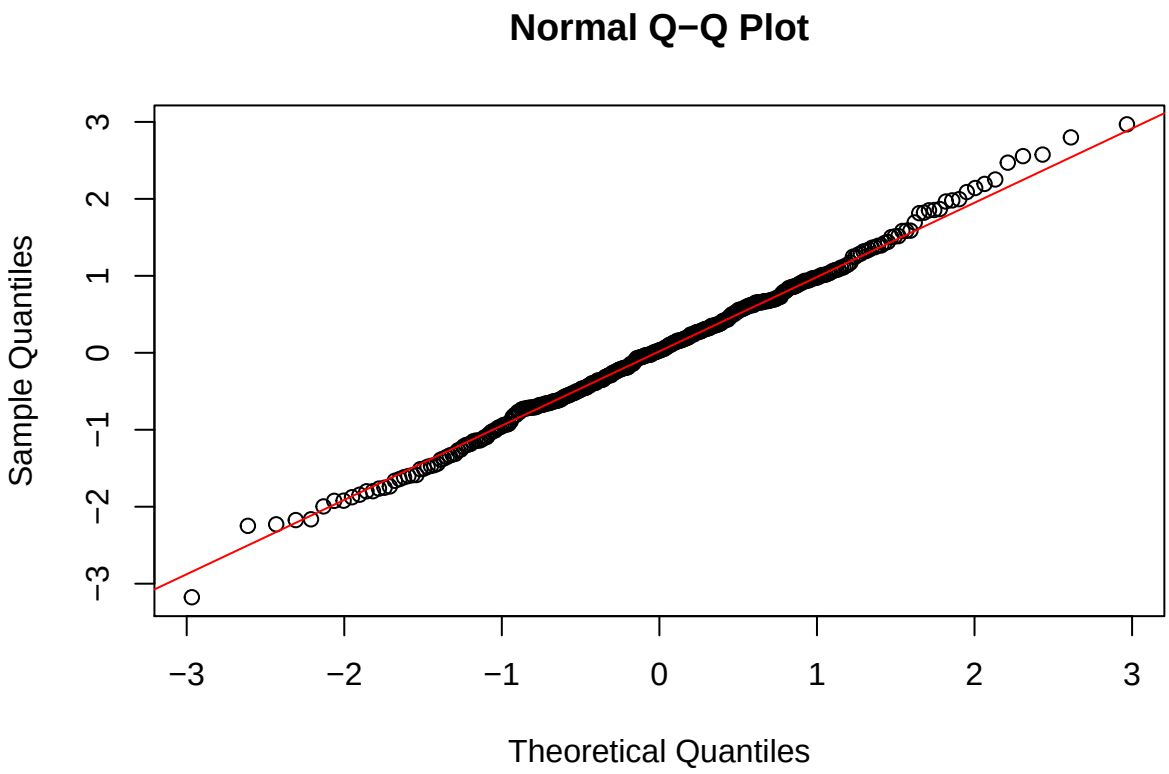
```
shapiro.test(df1$E)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df1$E
## W = 0.95563, p-value = 1.718e-08
```

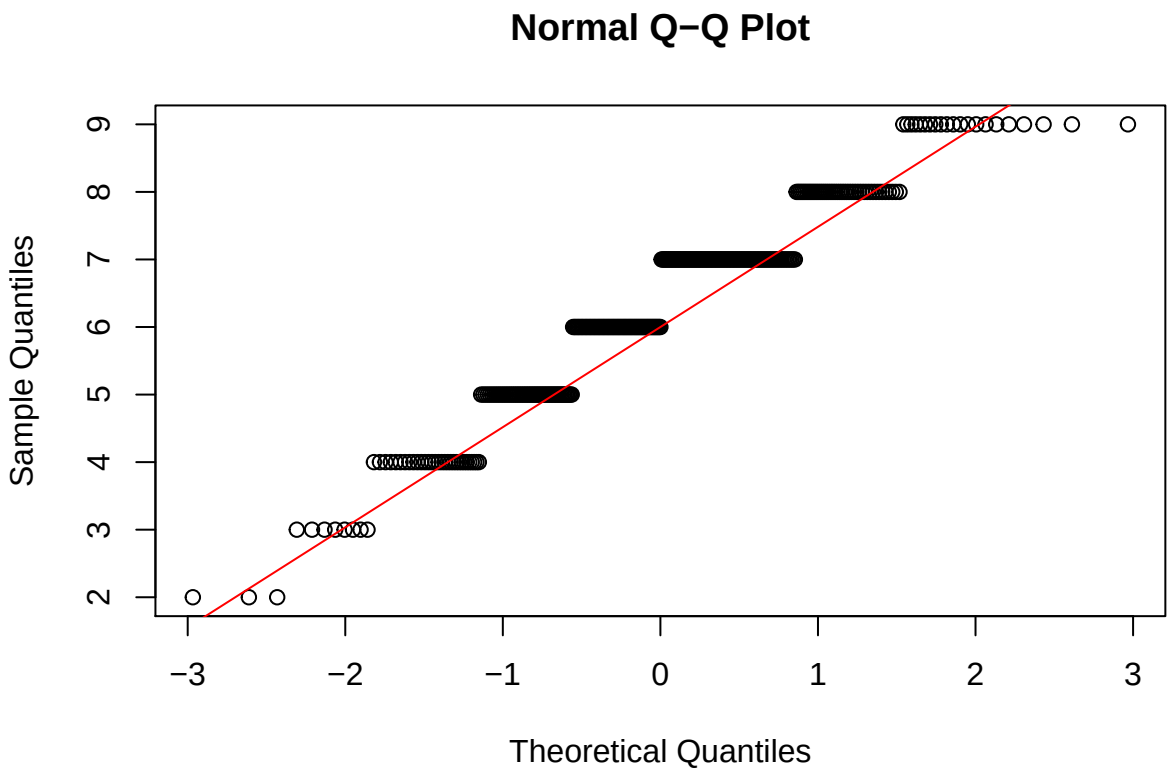
```
# Then I plotted the data:
qqnorm(df1$A)
qqline (df1$A, col=2)
```



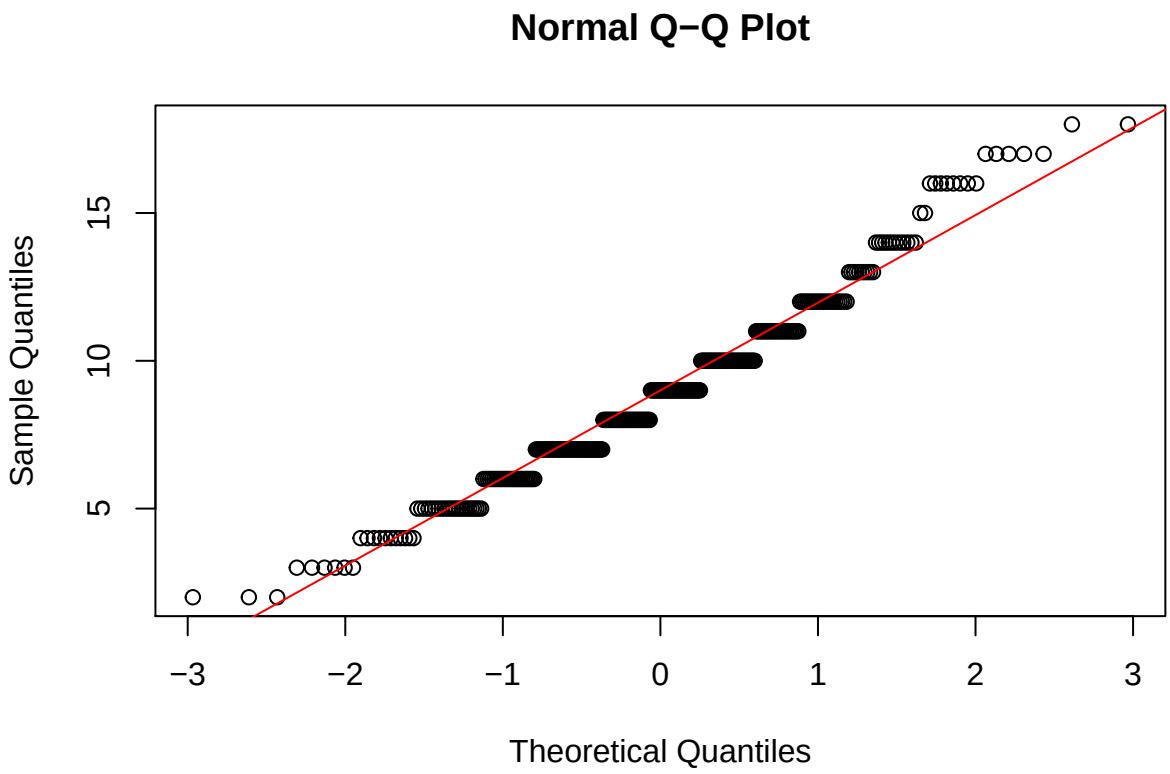

```
qqnorm(df1$B)
qqline (df1$B, col=2)
```



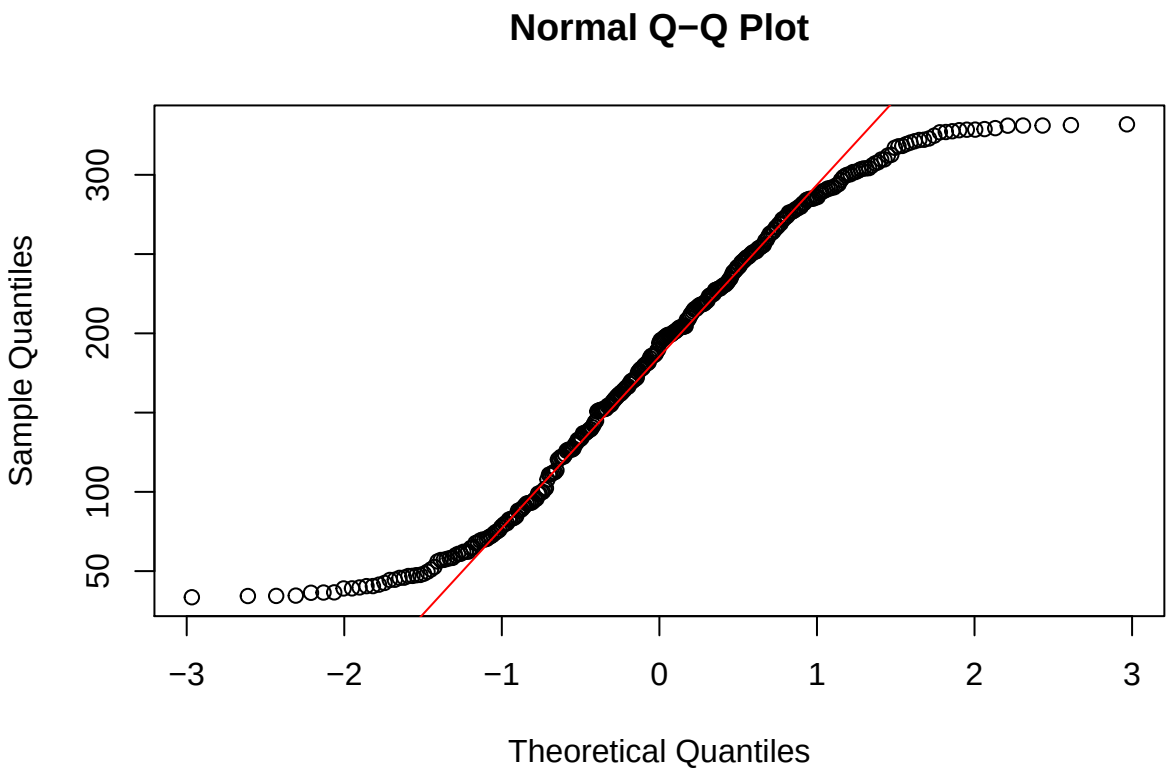

```
qqnorm(df1$C)
qqline (df1$C, col=2)
```




```
qqnorm(df1$D)
qqline (df1$D, col=2)
```




```
qqnorm(df1$E)
qqline (df1$E, col=2)
```



Looking at the results of the Shapiro-Wilk test, regarding A and B, we cannot reject the hypothesis that the sample comes from a population which has a normal distribution. For C, D, and E we can reject the hypothesis that the sample comes from a population which has a normal distribution. Verifying the plots, A and B are normal and the others are not.

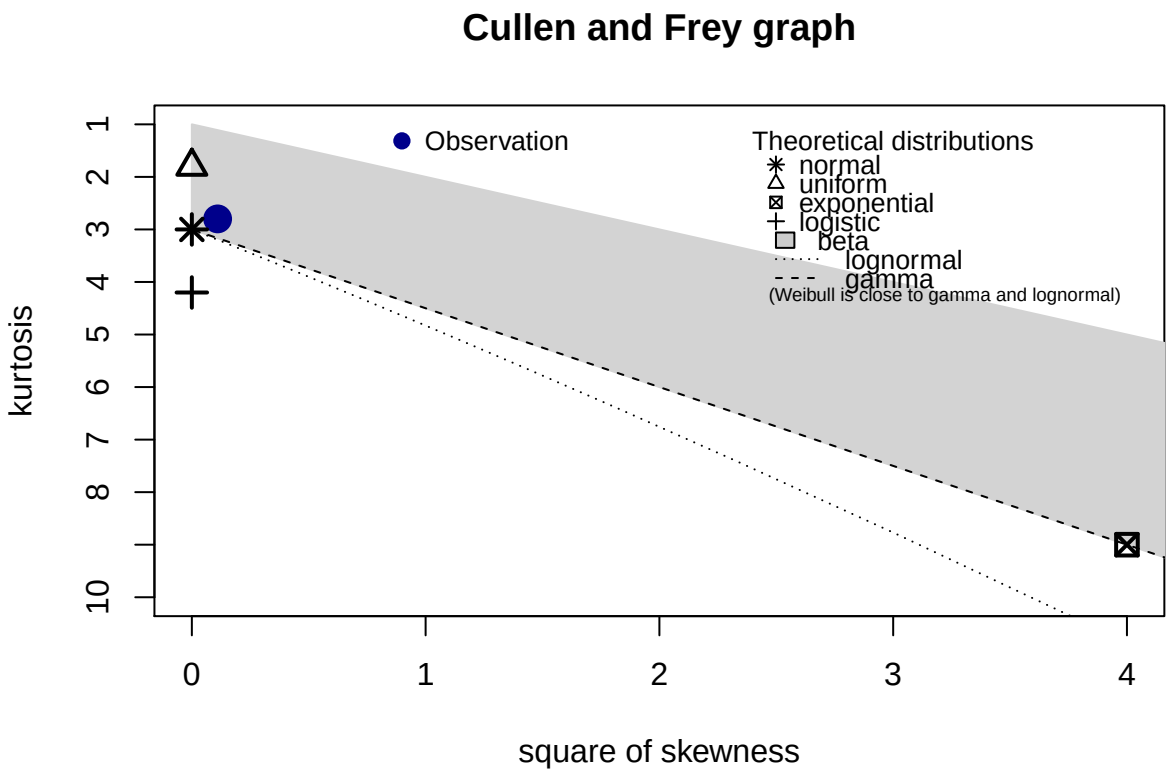
Regarding C, D , and E, I needed to run some more tests using fitdistrplus:

```
library(fitdistrplus)
```

```
## Loading required package: MASS
```

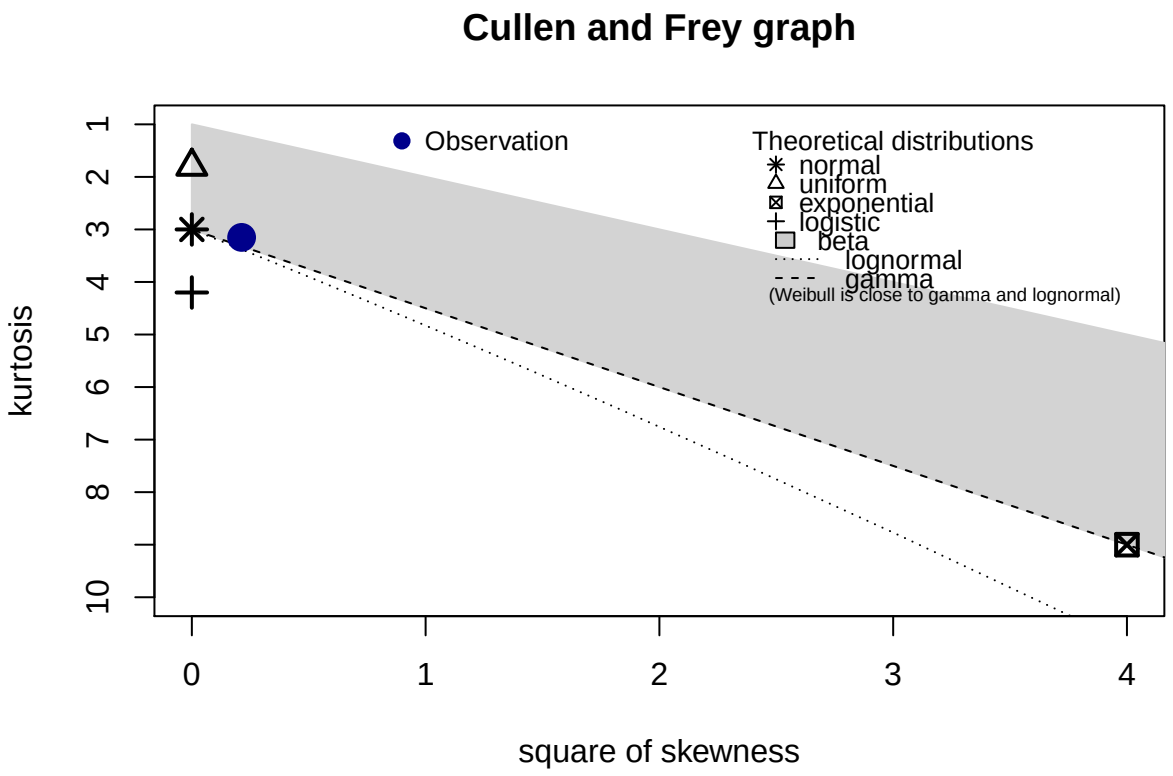
```
## Loading required package: survival
```

```
# This plots the data
descdist(df1$C, discrete = FALSE)
```



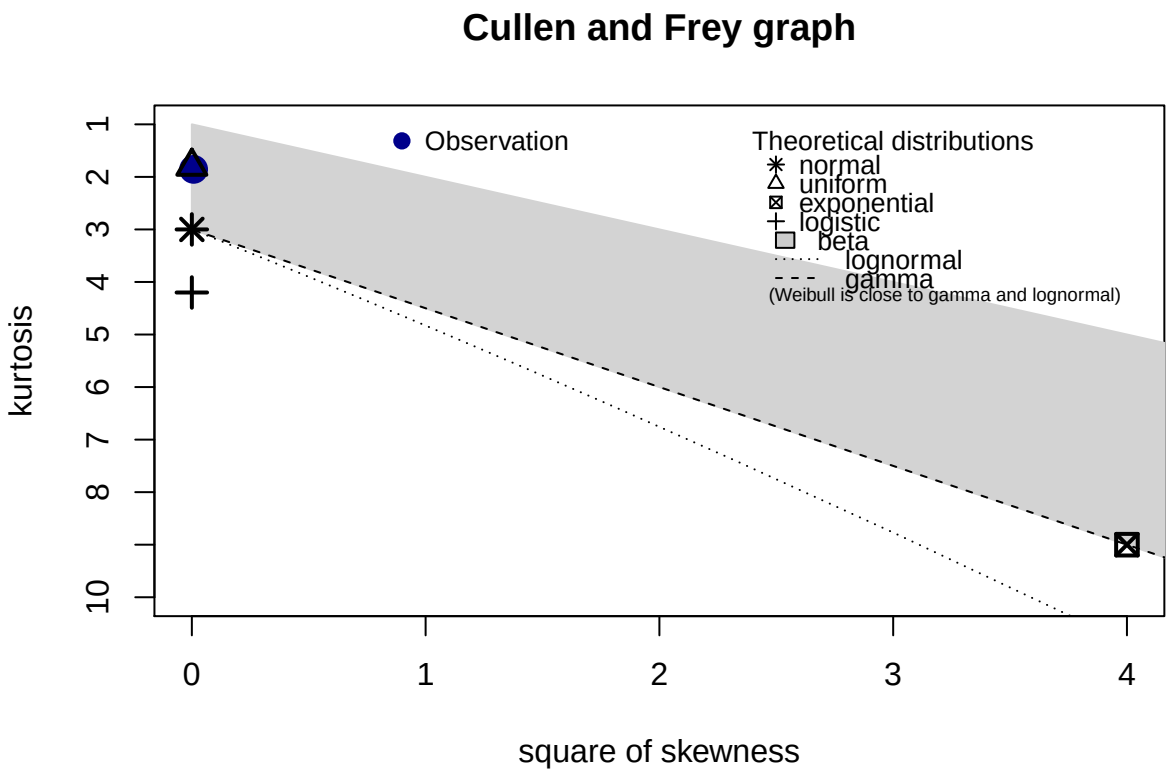

```
## summary statistics
## ------
## min:  2   max:  9
## median:  6
## mean:  6.3003
## estimated sd:  1.48686
## estimated skewness:  -0.3321043
## estimated kurtosis:  2.799626
```

```
descdist(df1$D, discrete = FALSE)
```




```
## summary statistics
## ------
## min:  2   max:  18
## median:  9
## mean:  8.918919
## estimated sd:  3.134923
## estimated skewness:  0.4613681
## estimated kurtosis:  3.151488
```

```
descdist(df1$E, discrete = FALSE)
```

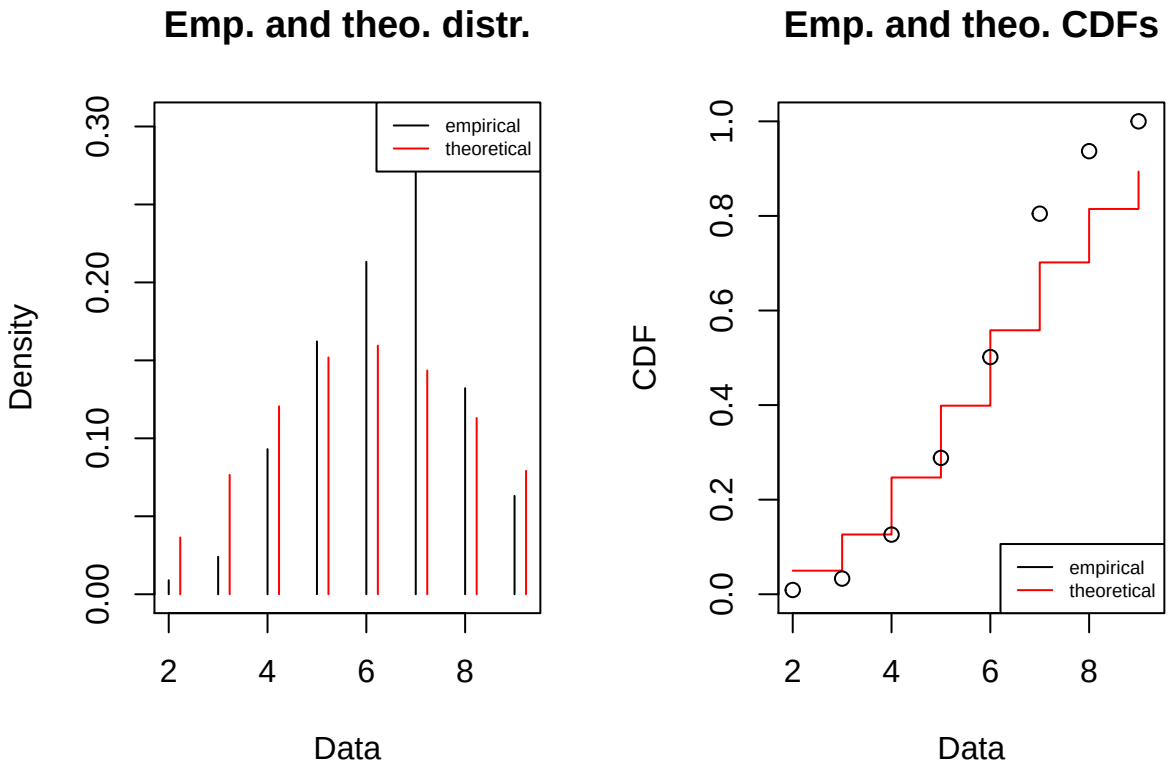



```
## summary statistics
## ------
## min:  33.52111   max:  331.8417
## median:  194.1186
## mean:  185.9274
## estimated sd:  87.42228
## estimated skewness:  -0.08673404
## estimated kurtosis:  1.856751
```

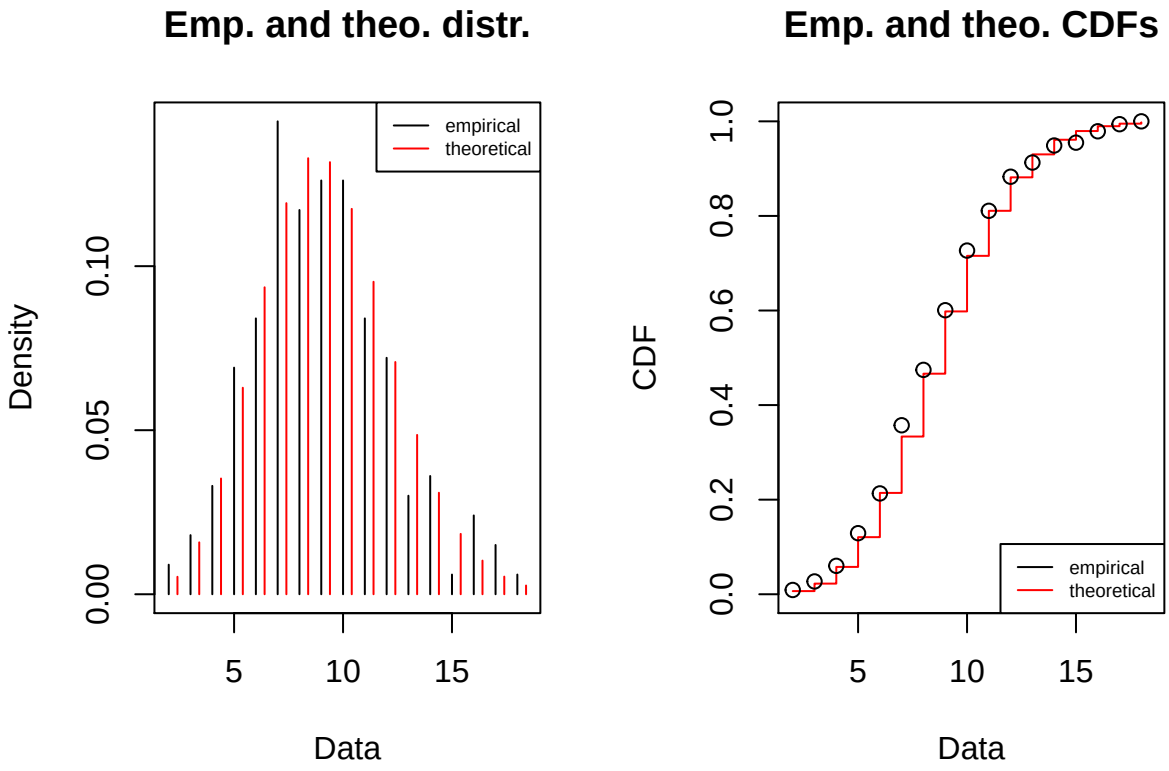
E is uniform (see below too), but C and D require additional tests but will be binomial or Poisson.

Based off of the fitdist plots, C is Poisson and D is binomial:

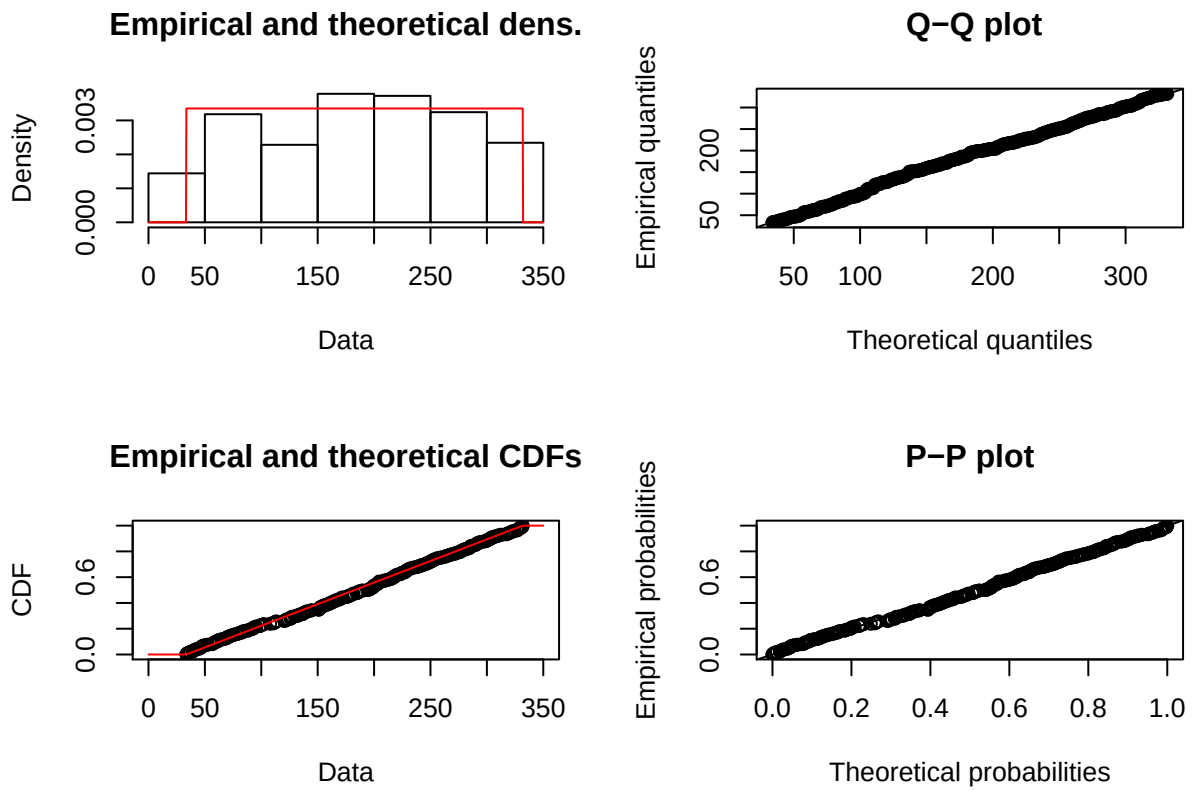
```
# For C
fit.pois2<-fitdist(df1$C, "pois")
plot(fit.pois2)
```




```
# For D
fit.pois<-fitdist(df1$D, "pois")
plot(fit.pois)
```




```
# For E
fit.unif<-fitdist(df1$E, "unif")
plot(fit.unif)
```



1.2.3 What are the summary statistics?

Below are the summary statistics with the graphs:

```
# internal structure
str(df1)
```

```
## 'data.frame':    333 obs. of  5 variables:
##  $ A: num  8.26 10.56 8.74 6.56 9.36 ...
##  $ B: num  -0.656 -0.716 0.8 1.583 1.027 ...
##  $ C: num  6 7 7 6 7 7 2 7 8 4 ...
##  $ D: num  8 8 5 10 8 12 10 10 9 5 ...
##  $ E: num  310 302 159 293 261 ...
```

```
# This is a summary of the data:
summary(df1)
```

```
##        A                B                  C             D
##  Min.   : 3.902   Min.   :-3.17616   Min.   :2.0   Min.   : 2.000
##  1st Qu.: 7.793   1st Qu.:-0.63195   1st Qu.:5.0   1st Qu.: 7.000
##  Median : 9.072   Median : 0.03412   Median :6.0   Median : 9.000
##  Mean   : 9.079   Mean   : 0.03063   Mean   :6.3   Mean   : 8.919
##  3rd Qu.:10.395   3rd Qu.: 0.67029   3rd Qu.:7.0   3rd Qu.:11.000
##  Max.   :14.794   Max.   : 2.96851   Max.   :9.0   Max.   :18.000
##        E
##  Min.   : 33.52
##  1st Qu.:112.28
```

```
##  Median :194.12
##  Mean   :185.93
##  3rd Qu.:258.43
##  Max.   :331.84
```

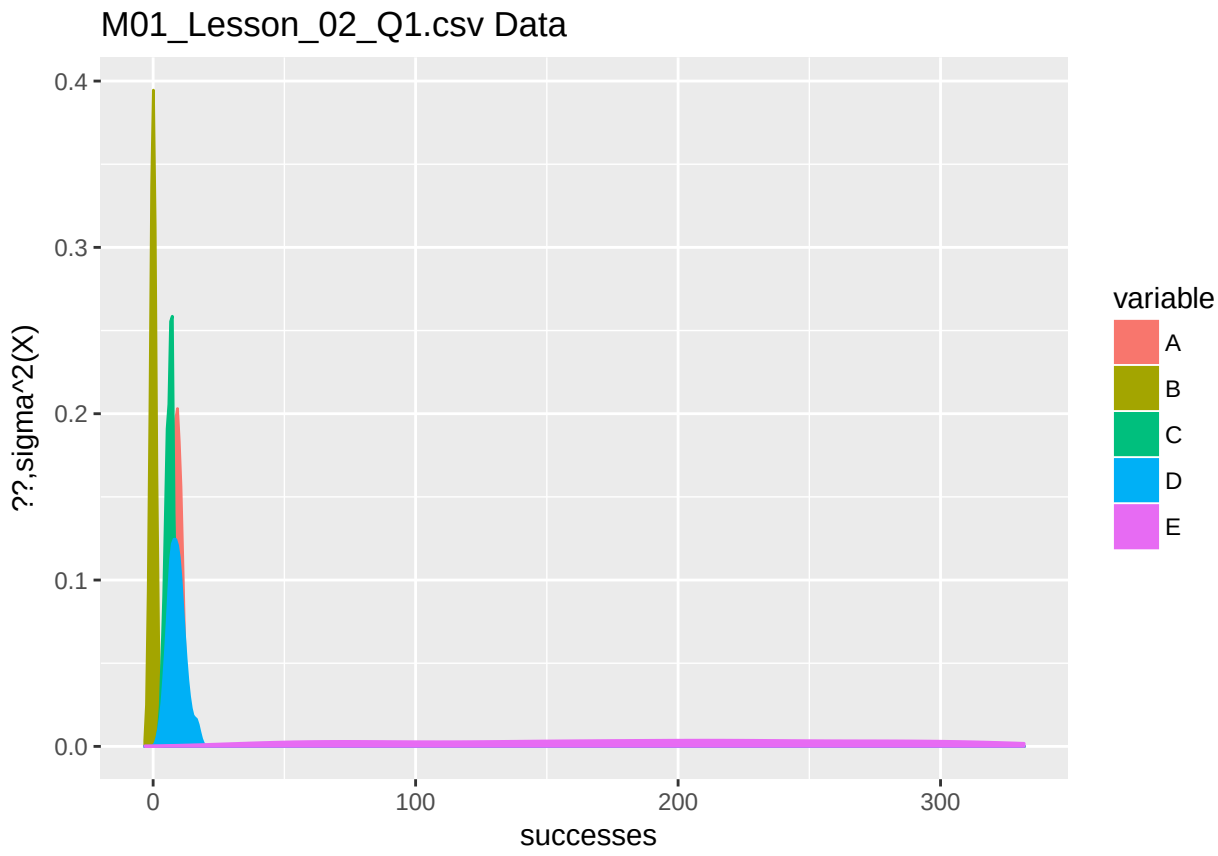
```
# Melt the data requires reshape2
rnd <- melt(data = df1)
```

```
## No id variables; using all as measure variables
```

```
# Review the data to make sure it looks ok
summary(rnd)
```

```
##  variable     value
##  A:333    Min.   : -3.176
##  B:333    1st Qu.:  5.000
##  C:333    Median :  8.000
##  D:333    Mean   : 42.051
##  E:333    3rd Qu.: 11.819
##           Max.   :331.842
```

```
# Plot the data:
ggplot(rnd, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable,
  fill = variable
)) + labs(title = "M01_Lesson_02_Q1.csv Data", y = "??,sigma^2(X)", x =
            " successes")
```



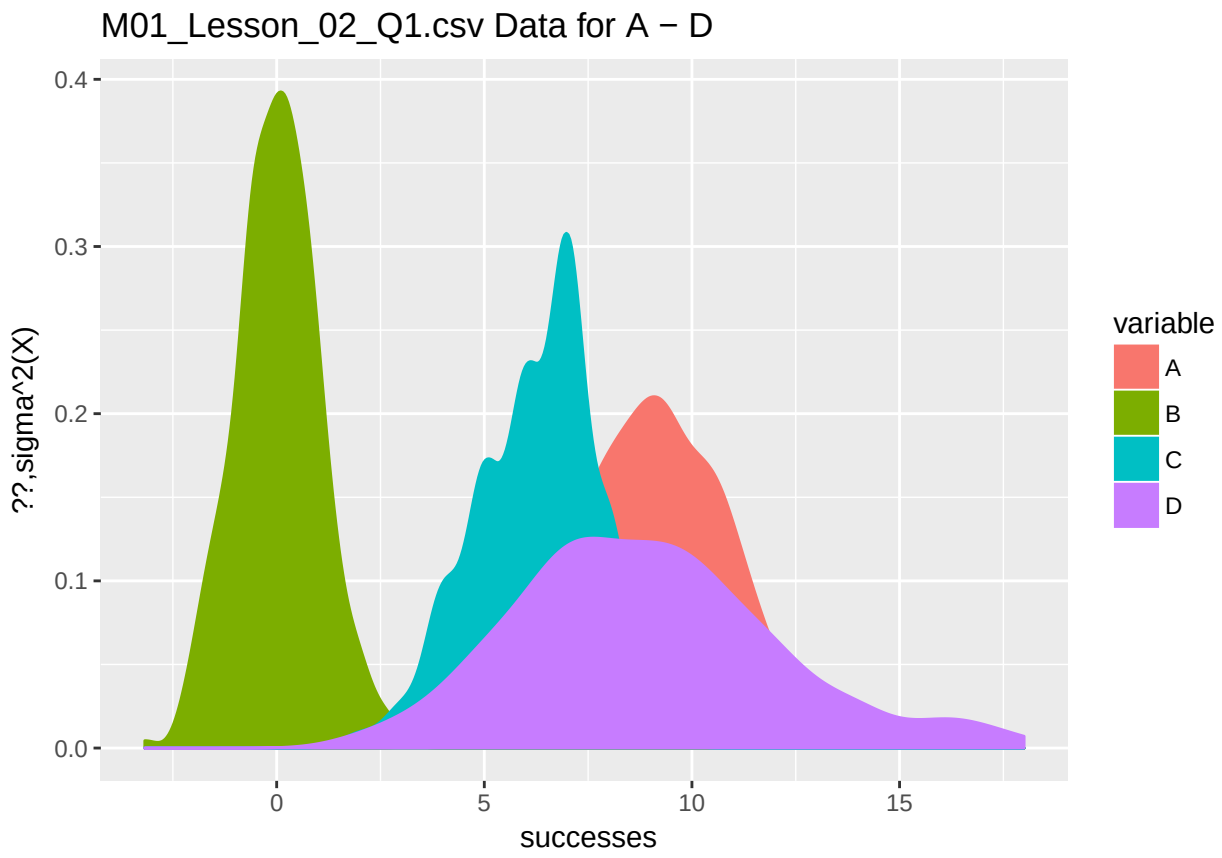
```r
# The data looks hard to read, so I am going to separate A-D and E.
# This is A-D:
df2 <- data.frame(
  A = df1$A,
  B = df1$B,
  C = df1$C,
  D = df1$D
)
rnd2 <- melt(data = df2)
```

```
## No id variables; using all as measure variables
```

```r
summary(rnd2)
```

```
##  variable     value
##  A:333    Min.   :-3.176
##  B:333    1st Qu.: 2.238
##  C:333    Median : 7.000
##  D:333    Mean   : 6.082
##           3rd Qu.: 9.000
##           Max.   :18.000
```

```r
ggplot(rnd2, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable,
  fill = variable
)) + labs(title = "M01_Lesson_02_Q1.csv Data for A - D", y = "??,sigma^2(X)", x =
            " successes")
```



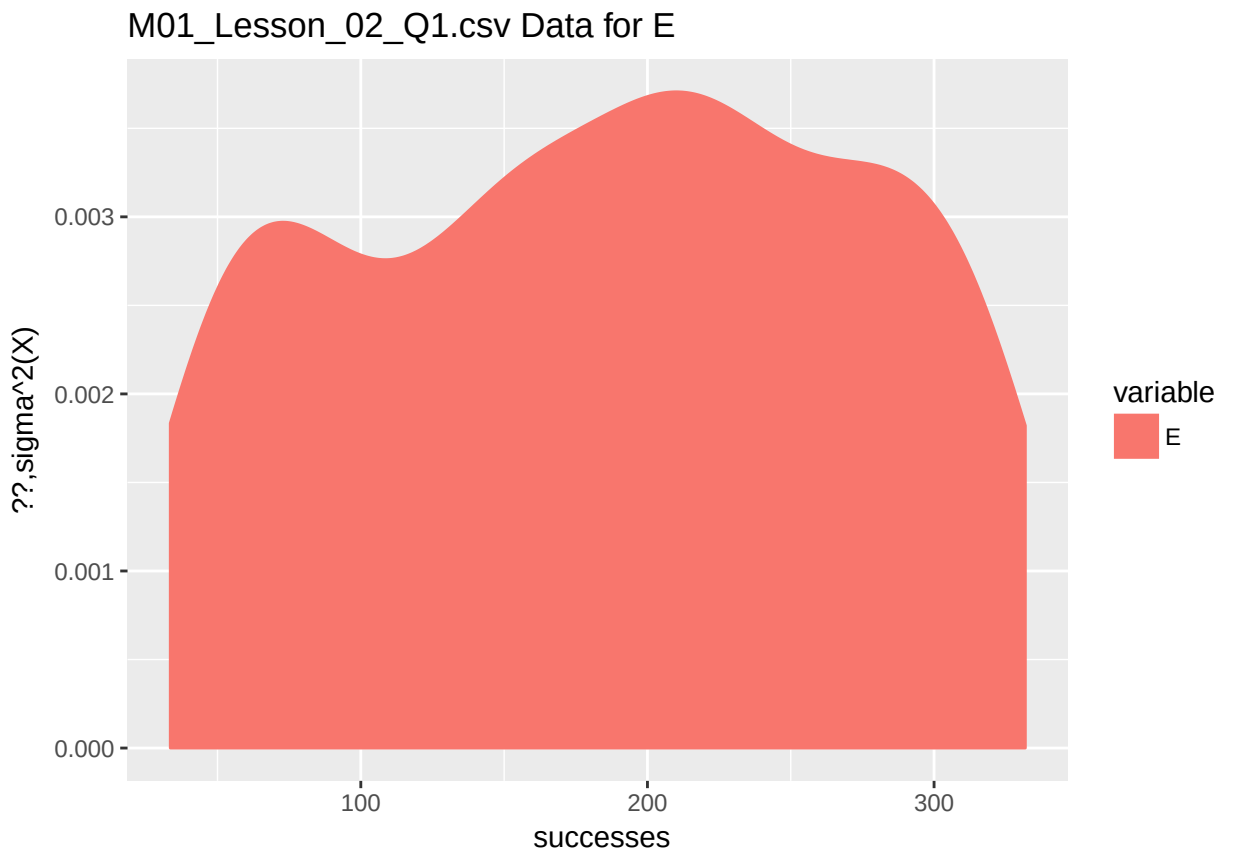

```
# This is E:
df3 <- data.frame(E = df1$E)
rnd3 <- melt(data = df3)
```

```
## No id variables; using all as measure variables
```

```
summary(rnd3)
```

```
##  variable     value
##  E:333    Min.   : 33.52
##           1st Qu.:112.28
##           Median :194.12
##           Mean   :185.93
##           3rd Qu.:258.43
##           Max.   :331.84
```

```
ggplot(rnd3, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable,
  fill = variable
)) + labs(title = "M01_Lesson_02_Q1.csv Data for E", y = "??,sigma^2(X)", x =
            " successes")
```



1.2.4 Are there anomalies/outliers?

I had a hard time trying to understand if C and D were possion or binomial distributed, other than that I do not think there are any anomalies and the data looks good.

1.2.5 Try to regenerate the data in each column and plot your regenerated data versus the original data using a faceted graph. How does it compare?

Below is regeneration of the data using R:

```
n = 333
norm_dist <- data.frame(
  regenerateA = rnorm(n = n, mean = mean(df1$A), sd = sd(df1$A)),
  regenerateB = rnorm(n = n, mean = mean(df1$B), sd = sd(df1$B)),
  regenerateC = rpois(n = n, lambda = mean(df1$C)),
  regenerateD = rpois(n = n, lambda = mean(df1$D)),
  regenerateE = runif(n, min(df1$E), max(df1$E)+1)
)

summary(norm_dist)
```

```
##   regenerateA      regenerateB       regenerateC      regenerateD
##  Min.   : 2.818   Min.   :-2.5057   Min.   : 0.000   Min.   : 2.000
##  1st Qu.: 7.955   1st Qu.:-0.5648   1st Qu.: 5.000   1st Qu.: 7.000
##  Median : 9.478   Median : 0.1438   Median : 6.000   Median : 9.000
```

```
##  Mean   : 9.319   Mean   : 0.1732   Mean   : 6.237   Mean   : 8.877
##  3rd Qu.:10.600   3rd Qu.: 0.8430   3rd Qu.: 8.000   3rd Qu.:11.000
##  Max.   :14.650   Max.   : 3.1313   Max.   :15.000   Max.   :21.000
##   regenerateE
##  Min.   : 33.73
##  1st Qu.:100.97
##  Median :175.70
##  Mean   :181.44
##  3rd Qu.:263.92
##  Max.   :332.28
```

```
head(norm_dist)
```

```
##   regenerateA regenerateB regenerateC regenerateD regenerateE
## 1    7.531290   0.9054337           4           5   275.78901
## 2    9.799171   0.5600626           6           7    63.47168
## 3    9.562771   0.7981800           5           8   178.20487
## 4    9.683227  -0.1019229           6          11   278.18844
## 5    8.383622   0.1916915           8           9   172.50221
## 6   10.508829  -1.3117657           2          10   233.08041
```

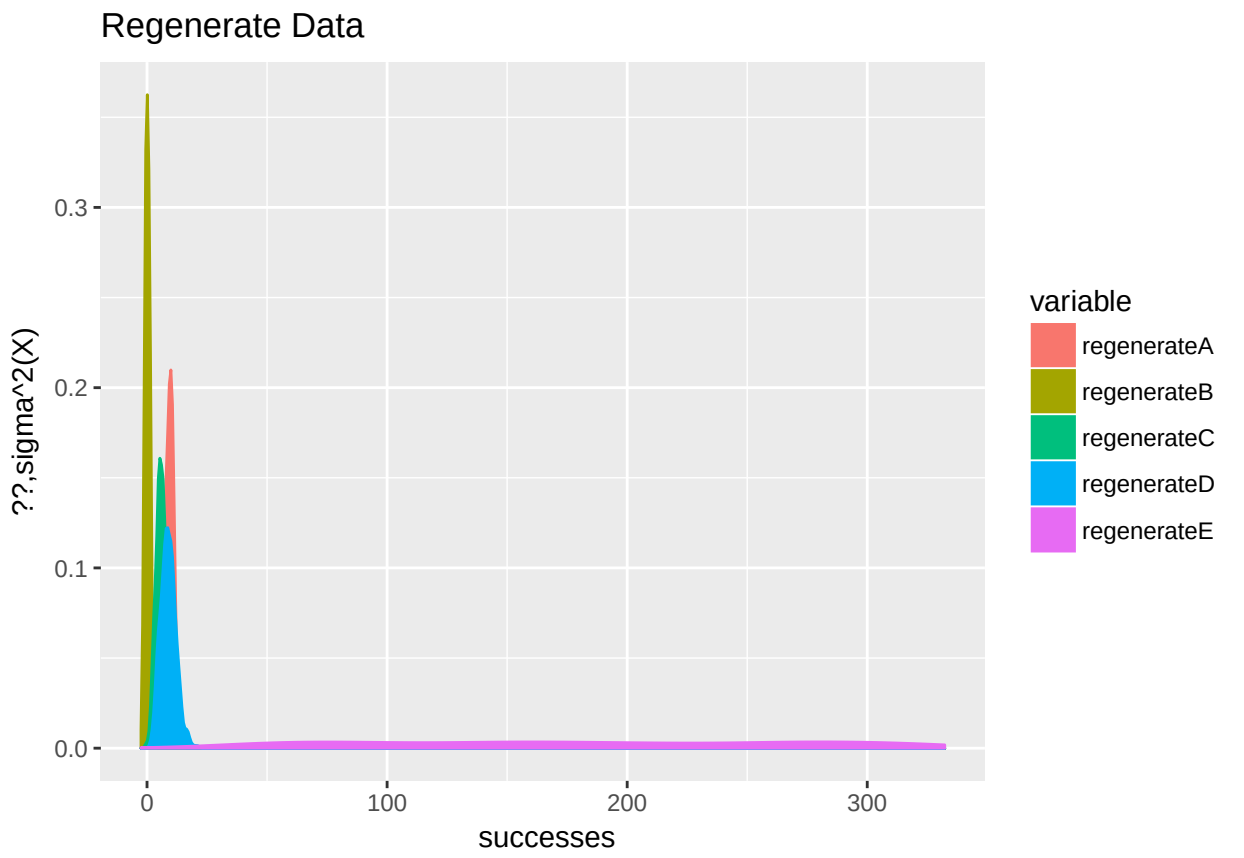
```
# Melt the data
rndReg <- melt(data = norm_dist)
```

```
## No id variables; using all as measure variables
```

```
# Review the data to make sure it looks ok
summary(rndReg)
```

```
##         variable       value
##  regenerateA:333   Min.   : -2.506
##  regenerateB:333   1st Qu.:  4.000
##  regenerateC:333   Median :  8.000
##  regenerateD:333   Mean   : 41.210
##  regenerateE:333   3rd Qu.: 12.000
##                    Max.   :332.279
```

```
# Plot the data:
ggplot(rndReg, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable,
  fill = variable
)) + labs(title = "Regenerate Data", y = "??,sigma^2(X)", x =
            " successes")
```

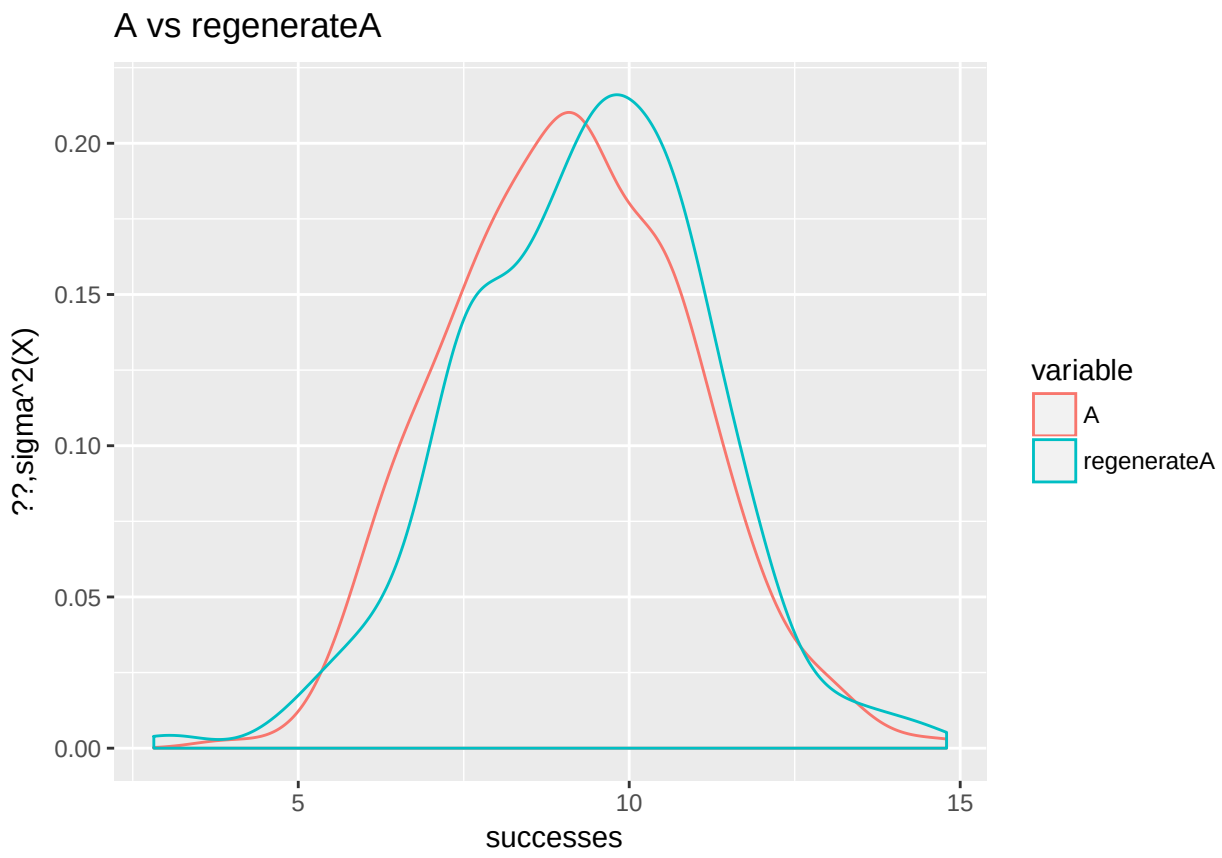


Comparing the data one to one, it looks like it matches well. However, I had a hard time getting C to match up, it looks like it could be in the ballpark but it doesn't fit as well as the others:

```
distA <- data.frame(A = df1$A, regenerateA = norm_dist$regenerateA)
distA <- melt(data = distA)
```

```
## No id variables; using all as measure variables
```

```
ggplot(distA, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable
)) + labs(title = "A vs regenerateA", y = "??,sigma^2(X)", x =
            " successes")
```




```
distB <- data.frame(B = df1$B, regenerateB = norm_dist$regenerateB)
distB <- melt(data = distB)
```

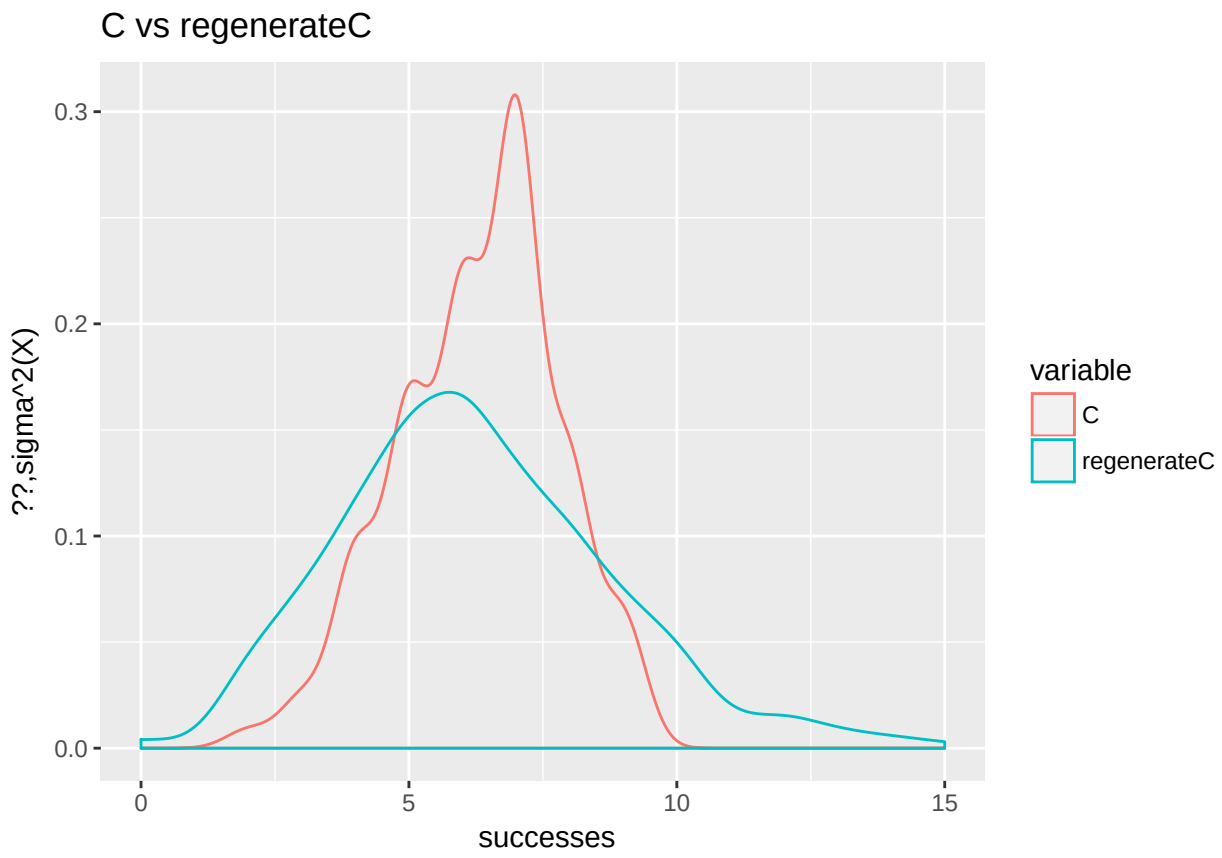
```
## No id variables; using all as measure variables
```

```
ggplot(distB, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable
)) + labs(title = "B vs regenerateB", y = "??,sigma^2(X)", x =
            " successes")
```


```
distC <- data.frame(C = df1$C, regenerateC = norm_dist$regenerateC)
distC <- melt(data = distC)
```

```
## No id variables; using all as measure variables
```

```
ggplot(distC, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable
)) + labs(title = "C vs regenerateC", y = "??,sigma^2(X)", x =
            " successes")
```




```
distD <- data.frame(D = df1$D, regenerateD = norm_dist$regenerateD)
distD <- melt(data = distD)
```

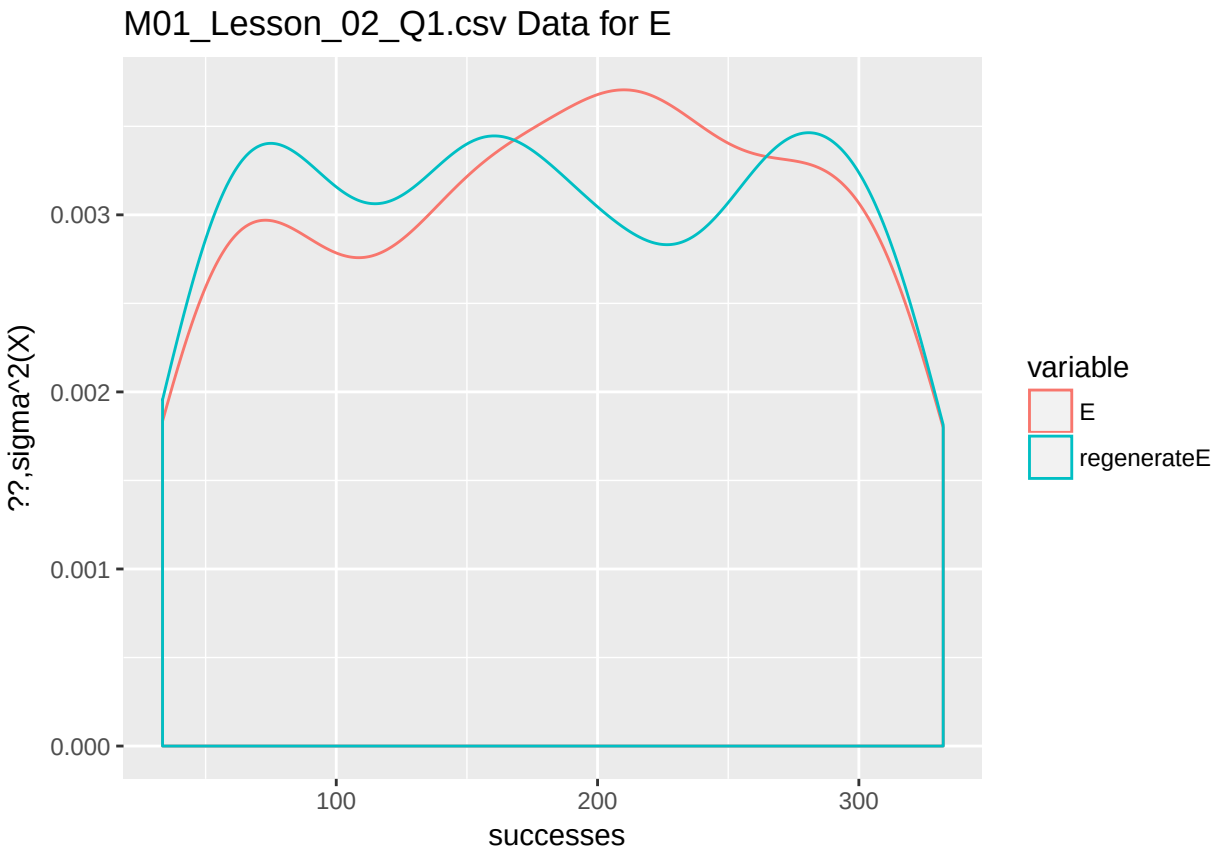
```
## No id variables; using all as measure variables
```

```
ggplot(distD, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable
)) + labs(title = "D vs regenerateD", y = "??,sigma^2(X)", x =
            " successes")
```


```
distE <- data.frame(E = df1$E, regenerateE = norm_dist$regenerateE)
distE <- melt(data = distE)
```

```
## No id variables; using all as measure variables
```

```
ggplot(distE, aes(x = value)) + geom_density(aes(
  group = variable,
  color = variable
)) + labs(title = "M01_Lesson_02_Q1.csv Data for E", y = "??,sigma^2(X)", x =
            " successes")
```


M01_Lesson_02_Q1.csv Data for E
??,sigma^2(X)
0.003
0.002
0.001
0.000
100
200
300
successes
variable
E
regenerateE