MA 354: Data Analysis – Fall 2021 – Due 10/8 at 5p
Homework 2: Emily Clark, Isabel Gephart, and Emma Hart

1. Select a continuous distribution (Not the uniform or exponential). It does not have to be one that we cover in the notes! To explore the PDF of your distribution, specify two sets of parameter(s) for your distribution.
Solution: The Gaussian Distribution: Parameter sets: (mean=0, SD=1) and (mean=1, SD=2)

   (a) **History** Discuss what types of random variables are modeled with your distribution. Be sure to include a discussion about the support and ensure to provide the density function, and CDF. This requires some internet research – what's the history of the distribution, why was it created and named? What are some exciting applications of this distribution?
   Cite all of your sources in LaTeX by adding a BibTeX citation to the .bib file. To help, I've cited R (R Core Team, 2021) in parentheses here. R Core Team (2021) provides helpful tools for the rest of the questions below. BibTeX citations are available through Google Scholar by clicking the cite button below the article of interest and selecting the BibTeX option.
   Solution: Here I cite (Wallis, 2014) and (Stahl, 2006). We chose to look at the Gaussian distribution for our continuous distribution. The Gaussian distribution is also known as the normal distribution and is very common in terms of independent, randomly generated variables. It is well known for the bell-shaped curve and is a very important probability graph in statistics. Its characterized by two parameters, the mean and standard deviation. Abraham DeMoivre derived formulas related to the curve early on in the 1600s; however, the name "Gaussian distribution" comes from Carl Friedrich Gauss, a German mathematician. He independently created a two-parameter exponential function in 1809 related to astronomical observation errors. James Clerk Maxwell, a British physicist, also had an early application of the normal distribution in 1859 with his law of distribution of molecular velocities. The "normal curve" formally appeared in 1870.

   (b) Show that you have a valid PDF. You will find the `integrate()` function in `R` helpful.

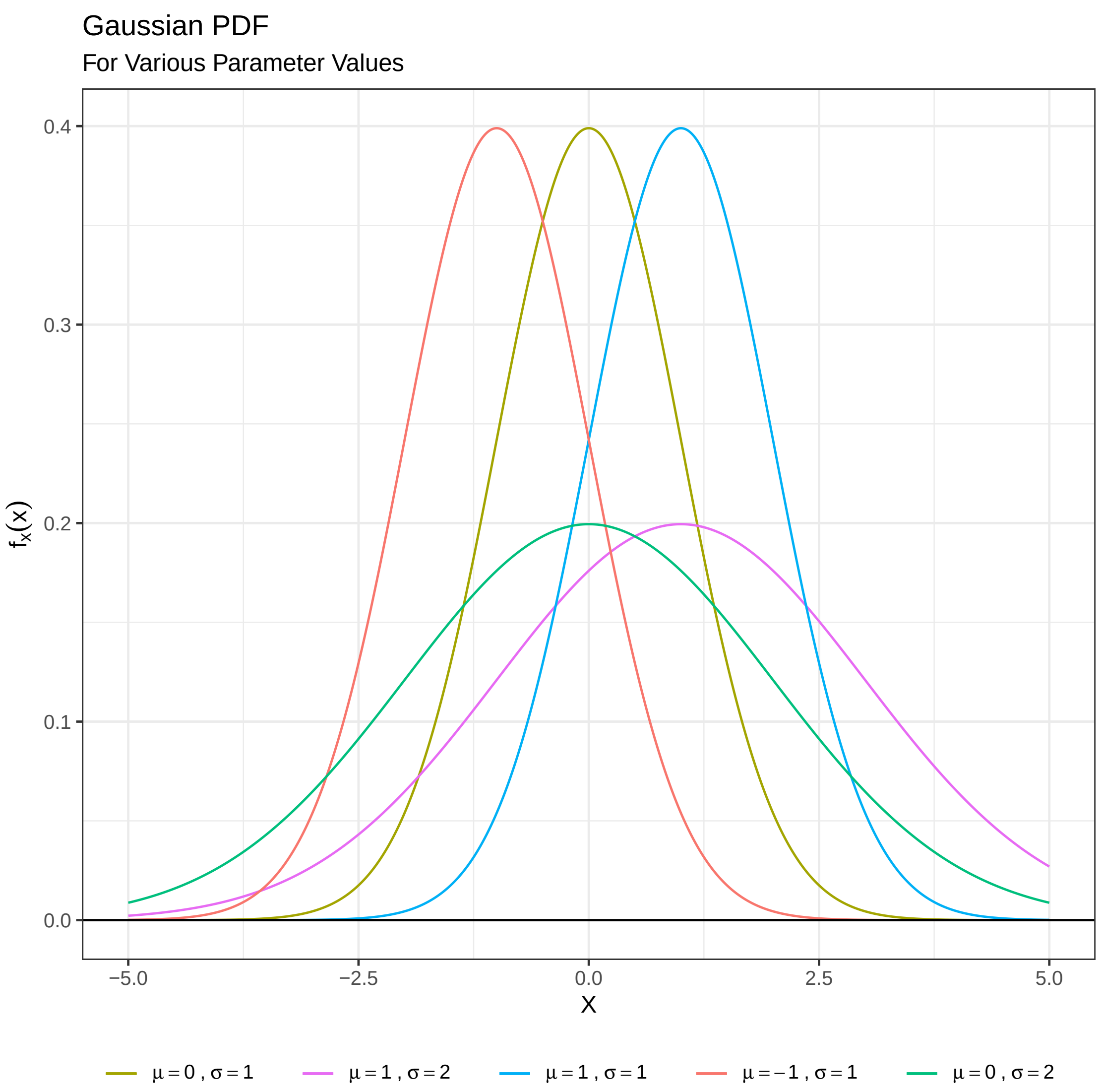
```
x <- seq(-5,5,0.001)
all(dnorm(x) >= 0)   #all positive

## [1] TRUE

integrate(dnorm, lower=-5, upper=5, subdivisions=10,000) #integral ~= 1

## 0.9999994 with absolute error < 8.7e-10
```

   (c) Find the median for your two sets of parameter(s). Conduct some research to find the median based on our PDF to confirm that your numerical approach is correct.
   Solution:

```
points<-data.frame(x=seq(-5,5,0.001),
                   f1=dnorm(x=seq(-5,5,0.001),mean=0,sd=1),
                   f2=dnorm(x=seq(-5,5,0.001),mean=1,sd=2),
                   f3=dnorm(x=seq(-5,5,0.001),mean=1,sd=1),
                   f4=dnorm(x=seq(-5,5,0.001),mean=-1,sd=1),
                   f5=dnorm(x=seq(-5,5,0.001),mean=0,sd=2))
median(points$f1)

## [1] 0.0175283

median(points$f2)

## [1] 0.09132454

#since this is a normal distribution the median and mean should be approximately equal
```

(d) Graph the PDF for several values of the parameter(s) including the two sets you specified. What does changing the parameter(s) do to the shape of the PDF?

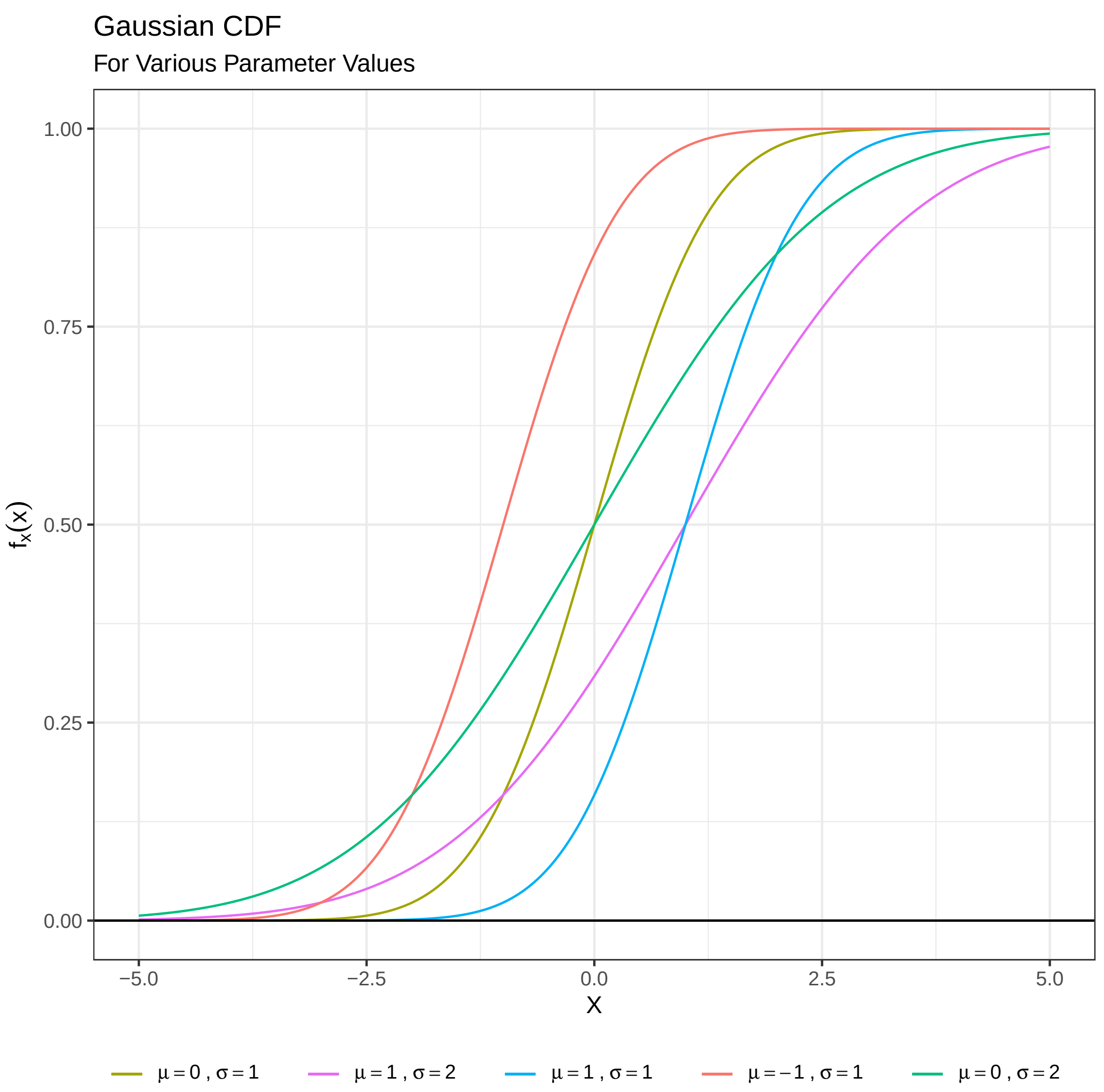
```r
library(ggplot2)
g1<-ggplot(data=points,aes(x=x))+
  geom_line(aes(y=f1,color="m=0 sd=1"))+
  geom_line(aes(y=f2,color="m=1 sd=2"))+
  geom_line(aes(y=f3,color="m=1 sd=1"))+
  geom_line(aes(y=f4,color="m=-1 sd=1"))+
  geom_line(aes(y=f5,color="m=0 sd=2"))+
  geom_hline(yintercept=0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("Gaussian PDF",subtitle="For Various Parameter Values")+
  scale_color_discrete("",breaks=c("m=0 sd=1","m=1 sd=2","m=1 sd=1",
                                   "m=-1 sd=1","m=0 sd=2"),
                     labels=c(bquote(mu==0~","~sigma==1),
                              bquote(mu==1~","~sigma==2),bquote(mu==1~","~sigma==1),
                              bquote(mu==-1~","~sigma==1),bquote(mu==0~","~sigma==2)))+
  theme(legend.position="bottom",
        legend.text = element_text(margin = margin(r = 15)))
g1
```



Solution: Changing the mean of these curves affects where the peak (max) is along the x-axis. The standard deviation determines the actual height (max). The curves with standard deviations of 2 are smaller than those with standard deviation 1.

(e) Graph the CDF for the same values of the parameter(s) as you did in Question 1d. What does changing the parameter(s) do to the shape of the CDF? Comment on the aspects of the CDFs that show that the CDF is valid.

```
points2<-data.frame(x=seq(-5,5,0.001),
                    f1=pnorm(q=seq(-5,5,0.001),mean=0,sd=1),
                    f2=pnorm(q=seq(-5,5,0.001),mean=1,sd=2),
                    f3=pnorm(q=seq(-5,5,0.001),mean=1,sd=1),
                    f4=pnorm(q=seq(-5,5,0.001),mean=-1,sd=1),
                    f5=pnorm(q=seq(-5,5,0.001),mean=0,sd=2))
g2<-ggplot(data=points2,aes(x=x))+
  geom_line(aes(y=f1,color="m=0 sd=1"))+
  geom_line(aes(y=f2,color="m=1 sd=2"))+
  geom_line(aes(y=f3,color="m=1 sd=1"))+
  geom_line(aes(y=f4,color="m=-1 sd=1"))+
  geom_line(aes(y=f5,color="m=0 sd=2"))+
  geom_hline(yintercept=0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("Gaussian CDF",subtitle="For Various Parameter Values")+
  scale_color_discrete("",breaks=c("m=0 sd=1","m=1 sd=2","m=1 sd=1",
                                   "m=-1 sd=1","m=0 sd=2"),
                       labels=c(bquote(mu==0~","~sigma==1),
                                bquote(mu==1~","~sigma==2),bquote(mu==1~","~sigma==1),
                                bquote(mu==-1~","~sigma==1),bquote(mu==0~","~sigma==2)))+
  theme(legend.position="bottom",
        legend.text = element_text(margin = margin(r = 15)))
g2
```

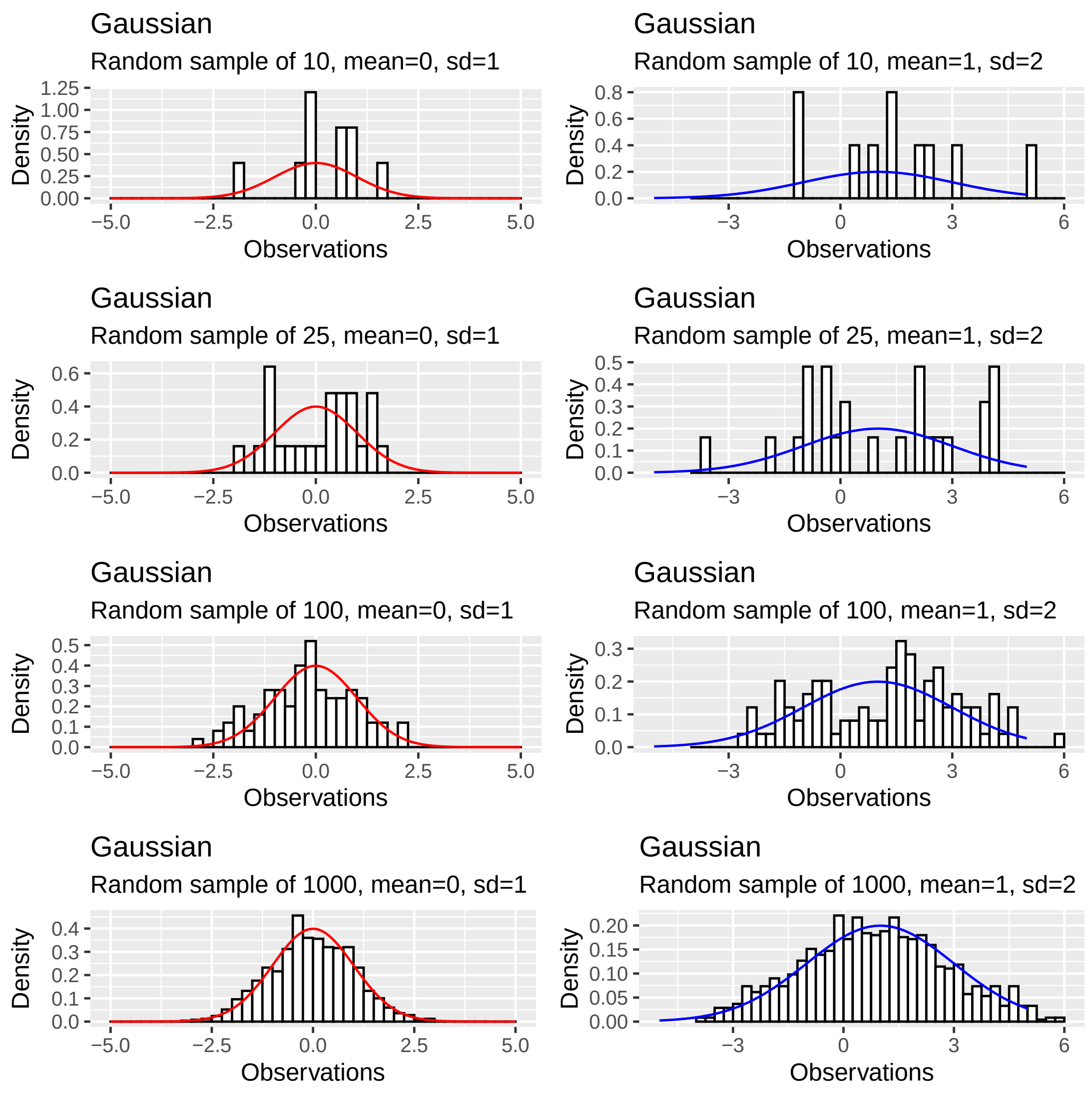


Solution: Increasing the mean shifts the curves on the x-axis and where they end up when f(x) is 0.5. The standard deviation effects their path in terms of steepness. The curves with standard deviation 1 are steeper than the curves with standard deviation. Further, the validity of this CDF is shown as all values are positive and less than or equal to 1.

(f) Generate a random sample of size $n = 10, 25, 100$, and 1000 for your two sets of parameter(s). In a 4×2 grid, plot a histogram of each set of data and superimpose the true density function at the specified parameter values. Interpret the results.
Solution:

```r
library(patchwork)

df <- data.frame(a = rnorm(10,mean=0, sd=1))
p1a <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-5, 5, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f1), color="red")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 10, mean=0, sd=1")


df <- data.frame(a = rnorm(25,mean=0, sd=1))
p1b <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-5, 5, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f1), color="red")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 25, mean=0, sd=1")


df <- data.frame(a = rnorm(100,mean=0, sd=1))
p1c <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-5, 5, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f1), color="red")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 100, mean=0, sd=1")

df <- data.frame(a = rnorm(1000,mean=0, sd=1))
p1d <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-5, 5, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f1), color="red")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 1000, mean=0, sd=1")
```

```r
################ second parameter set ###
df <- data.frame(a = rnorm(10, mean=1, sd=2))
p2a <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-4, 6, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f2), color="blue")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 10, mean=1, sd=2")


df <- data.frame(a = rnorm(25, mean=1, sd=2))
p2b <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-4, 6, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f2), color="blue")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 25, mean=1, sd=2")


df <- data.frame(a = rnorm(100, mean=1, sd=2))
p2c <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-4, 6, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f2), color="blue")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 100, mean=1, sd=2")

df <- data.frame(a = rnorm(1000, mean=1, sd=2))
p2d <- ggplot(df, aes(x = a)) +
  geom_histogram(aes(y =..density..),
                 breaks = seq(-4, 6, by = 0.25),
                 colour = "black",
                 fill = "white") +
  geom_line(data = points, aes(x=x, y=f2), color="blue")+
  xlab("Observations")+
  ylab("Density") +
  ggtitle("Gaussian",subtitle="Random sample of 1000, mean=1, sd=2")
(p1a + p2a) / (p1b + p2b) / (p1c + p2c) / (p1d + p2d)
```



```
# (p1a + p1b + p1c + p1d) / (p2a + p2b + p2c + p2d)
```

2. Continue with the continuous distribution you selected for Question 1.

   (a) Provide the mean, standard deviation, skewness, and kurtosis of the PDF. Ensure to interpret each.
   Solution:

```
#function that returns mean, SD, skew, and kurtosis for Gaussian PDF
norm.summary <- function(mu,sigma){
  norm.mean <- mu          #expected value E(x)
  norm.sd   <- sigma     #var(x)
  norm.skew <- 0           #skew(x)
  norm.kurt <- 0           #kurt(x)
  norm.stats <- data.frame(mean=norm.mean, SD=norm.sd, skewness=norm.skew, kurtosis=norm.kurt)
  return(norm.stats)
}

#input the parameters we are using
norm.summary(0,1)

##   mean SD skewness kurtosis
## 1    0  1        0        0

norm.summary(1,2)

##   mean SD skewness kurtosis
## 1    1  2        0        0
```

   The parameters for the Gaussian distribution are out inputs, so it is expected to get the parameters back for mean and standard deviation. The skewness and kurtosis for the Guassian distribution are always 0.

   (b) Generate a random sample of size $n = 10, 25, 100$, and 1000 for your two sets of parameter(s). Calculate the sample mean, standard deviation, skewness, and kurtosis. Interpret the results.

```
#generate random samples with mu = 0, sigma = 1
n10.m0.sd1 <- rnorm(10, 0, 1) #n = 10
n25.m0.sd1 <- rnorm(25, 0, 1) #n = 25
n100.m0.sd1 <- rnorm(100, 0, 1) #n = 100
n1000.m0.sd1 <- rnorm(1000, 0, 1) #n = 1000

#generate random samples with mu = 0, sigma = 1
n10.m1.sd2 <- rnorm(10, 1, 2) #n = 10
n25.m1.sd2 <- rnorm(25, 1, 2) #n = 25
n100.m1.sd2 <- rnorm(100, 1, 2) #n = 100
n1000.m1.sd2 <- rnorm(1000, 1, 2) #n = 1000

library(tidyverse)
library(e1071)

#summarize first distribution with n = 10
data.frame(mean = mean(n10.m0.sd1),
sd = sd(n10.m0.sd1),
skew = skewness(n10.m0.sd1),
kurt = kurtosis(n10.m0.sd1))

##          mean        sd       skew      kurt
## 1 -0.06558696 0.8619429 -0.3131221 -1.471219
```

These numbers are pretty far off the expected values indicated above. A sample size of 10 is not enough to generate an accurate distribution.

```r
#summarize first distribution with n = 25
data.frame(mean = mean(n25.m0.sd1),
sd = sd(n25.m0.sd1),
skew = skewness(n25.m0.sd1),
kurt = kurtosis(n25.m0.sd1))
```

```
##        mean        sd      skew       kurt
## 1 0.1789662 0.6327803 0.1892573 -0.6989673
```

These results are still not equal to the true values, but are closer than the sample with n = 10 which makes sense because the sample is more than twice as large.

```r
#summarize first distribution with n = 100
data.frame(mean = mean(n100.m0.sd1),
sd = sd(n100.m0.sd1),
skew = skewness(n100.m0.sd1),
kurt = kurtosis(n100.m0.sd1))
```

```
##          mean        sd     skew       kurt
## 1 -0.06679043 0.9741799 0.244731 -0.6943778
```

These results are still not equal to the true values, and the mean appears to be farther away than the estimates mean for n = 25.

```r
#summarize first distribution with n = 1000
data.frame(mean = mean(n1000.m0.sd1),
sd = sd(n1000.m0.sd1),
skew = skewness(n1000.m0.sd1),
kurt = kurtosis(n1000.m0.sd1))
```

```
##          mean        sd       skew      kurt
## 1 -0.02139329 0.9791003 -0.1152584 0.2862117
```

Although these are still not equal to the true values, the larger sample size of n = 1000 is the closest estimation presented here.

```r
#summarize second distribution with n = 10
data.frame(mean = mean(n10.m1.sd2),
sd = sd(n10.m1.sd2),
skew = skewness(n10.m1.sd2),
kurt = kurtosis(n10.m1.sd2))
```

```
##        mean       sd      skew       kurt
## 1 0.5824659 2.525313 0.6874124 -0.6570221
```

The mean in this sample is very close to the true mean of 1, but the standard devaition is very far off. This is due to small sample size.

```r
#summarize second distribution with n = 25
data.frame(mean = mean(n25.m1.sd2),
sd = sd(n25.m1.sd2),
skew = skewness(n25.m1.sd2),
kurt = kurtosis(n25.m1.sd2))
```

```
##        mean       sd       skew       kurt
## 1 0.3969054 2.208929 0.05712029 -0.8633324
```

In this sample the standard deviation is closer to the true value, but the mean is underestimated. Again, due to small sample size.

```r
#summarize second distribution with n = 100
data.frame(mean = mean(n100.m1.sd2),
sd = sd(n100.m1.sd2),
skew = skewness(n100.m1.sd2),
kurt = kurtosis(n100.m1.sd2))

##       mean       sd      skew      kurt
## 1 1.249404 2.087717 0.1193891 -0.143424
```

With n = 100 the estimated mean and standard deviation are much closer to their true values.

```r
#summarize second distribution with n = 1000
data.frame(mean = mean(n1000.m1.sd2),
sd = sd(n1000.m1.sd2),
skew = skewness(n1000.m1.sd2),
skurt = kurtosis(n1000.m1.sd2))

##       mean       sd        skew      skurt
## 1 1.074001 1.974181 0.001018816 0.03103829
```

These are the closest to the estimated values because of the larger sample size.

Overall, as sample size increases the sample statistics approach the distribution parameters.

(c) Generate a random sample of size $n = 10$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution:
Plots are created using Auguie (2015).

```r
#################################### FUNCTIONS #################################
#functions adapted from class
#MOM function
norm.mom <- function(par,data){
  mu <- par[1]
  sigma <- par[2]

  EX1 <- mu #expected values
  EX2 <- mu^2 + sigma^2

  xbar1 <- mean(data) #sample values
  xbar2 <- mean(data^2)

  c(EX1-xbar1, EX2-xbar2) #finds difference
}
#MLE function
norm.ll<-function(par, data, neg=T){
  mu <- par[1]
  sigma <- par[2]
  ll <- sum(dnorm(x=data, mean=mu, sd=sigma, log = TRUE))
  ifelse(neg, -ll, ll)
}
```

```
#load libraries
library(ggplot2)
library(gridExtra)
library(lemon)
library(nleqslv)
```

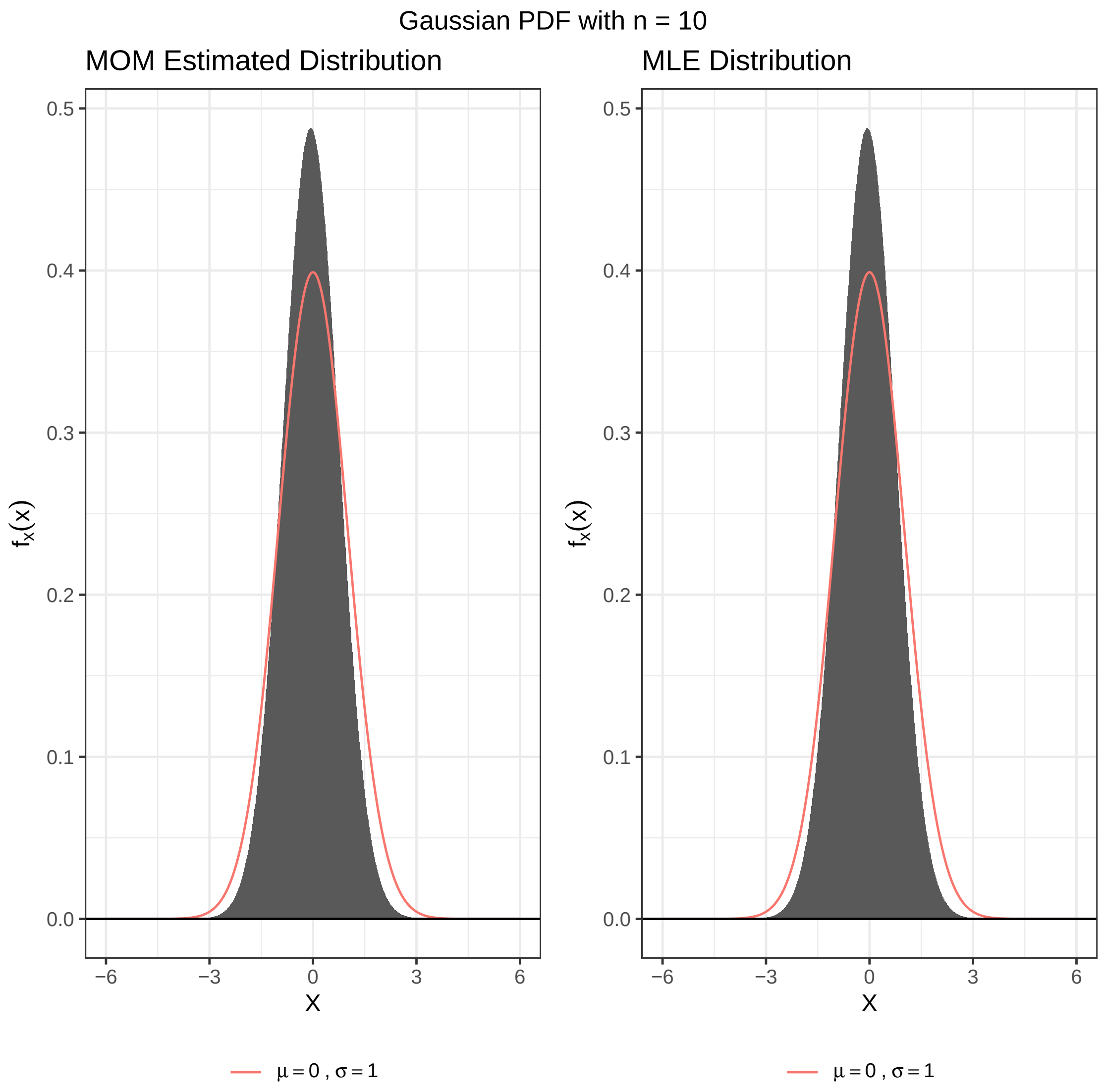
The following solution uses Wickham (2016), Auguie (2015), Edwards (2020), Hasselman (2018).

```
#################################### n = 10, par 1 #############################
#MOM: n = 10
mom.m0.sd1 <- nleqslv(x = c(0,1),
                      fn = norm.mom,
                      data = n10.m0.sd1)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #sets x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom distribution
                                   mean = mom.m0.sd1$x[1],
                                   sd = mom.m0.sd1$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

## Warning: Ignoring unknown parameters: binwidth, bins, pad

#MLE: n = 10
mle.m0.sd1 <- optim(par = c(0,1),
                    fn = norm.ll,
                    data=n10.m0.sd1)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle distribution
                                   mean = mle.m0.sd1$par[1],
                                   sd = mle.m0.sd1$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
```

```r
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())
```

```
## Warning:  Ignoring unknown parameters:  binwidth, bins, pad
```

```r
#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 10"))
```



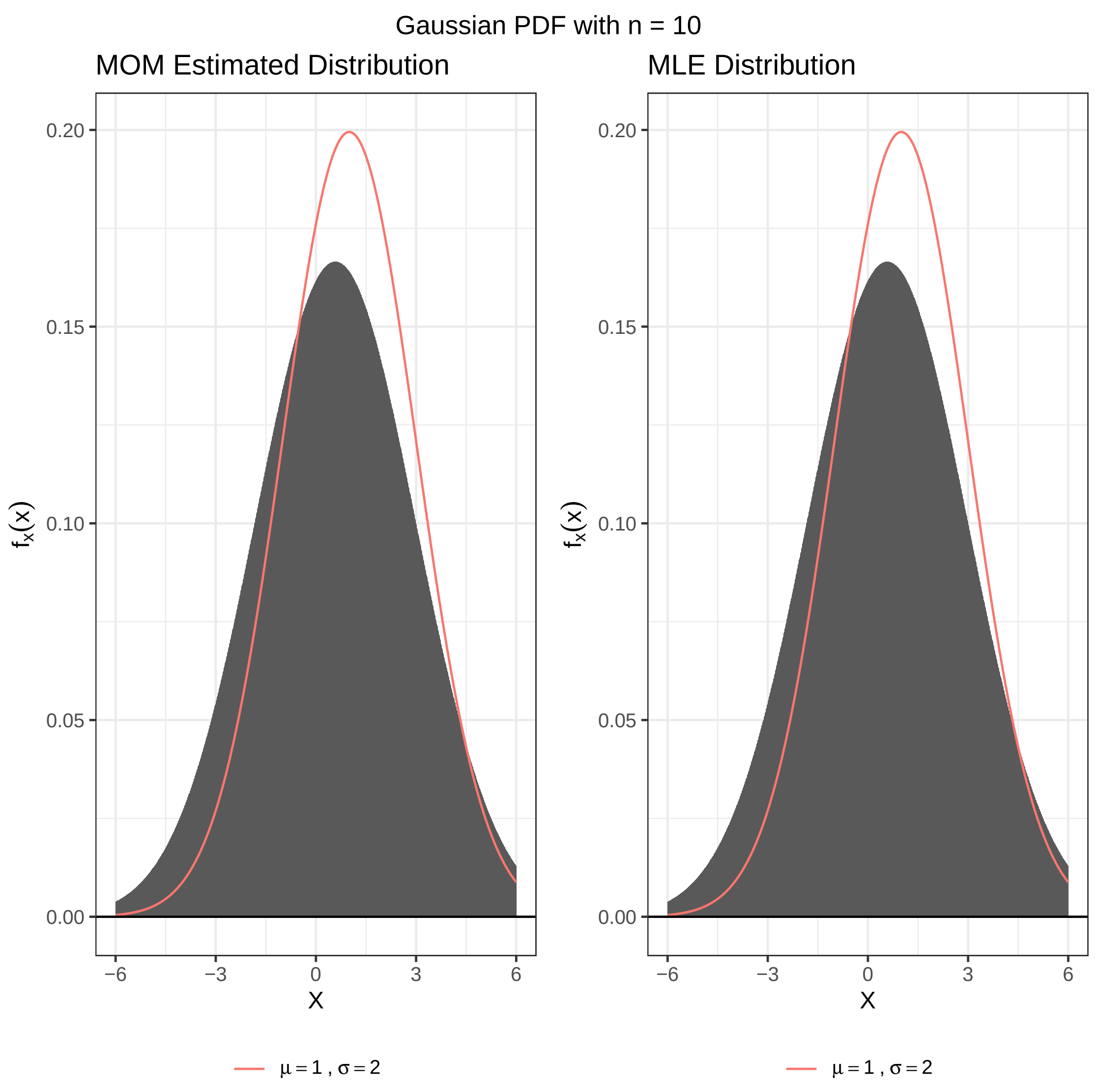
```r
#################################### n = 10, par 2 ############################
#MOM: n = 10
mom.m1.sd2 <- nleqslv(x = c(1,2),
                      fn = norm.mom,
                      data = n10.m1.sd2)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m1.sd2$x[1],
                                   sd = mom.m1.sd2$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

```
#MLE: n = 10
mle.m1.sd2 <- optim(par = c(1,2),
                    fn = norm.ll,
                    data=n10.m1.sd2)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01),  #mle dist
                                   mean = mle.m1.sd2$par[1],
                                   sd = mle.m1.sd2$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

```
#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 10"))
```

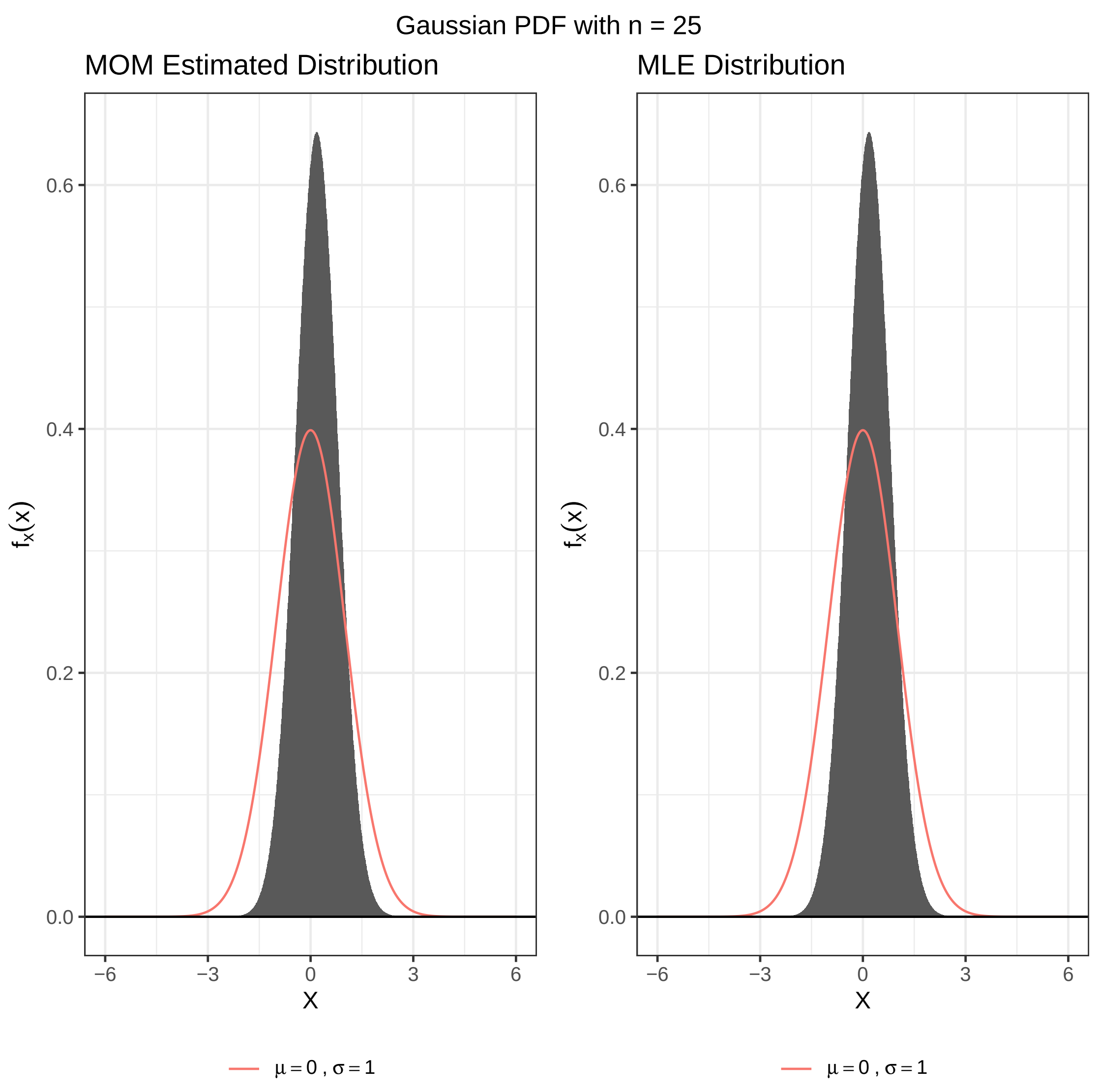


(d) Generate a random sample of size $n = 25$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.

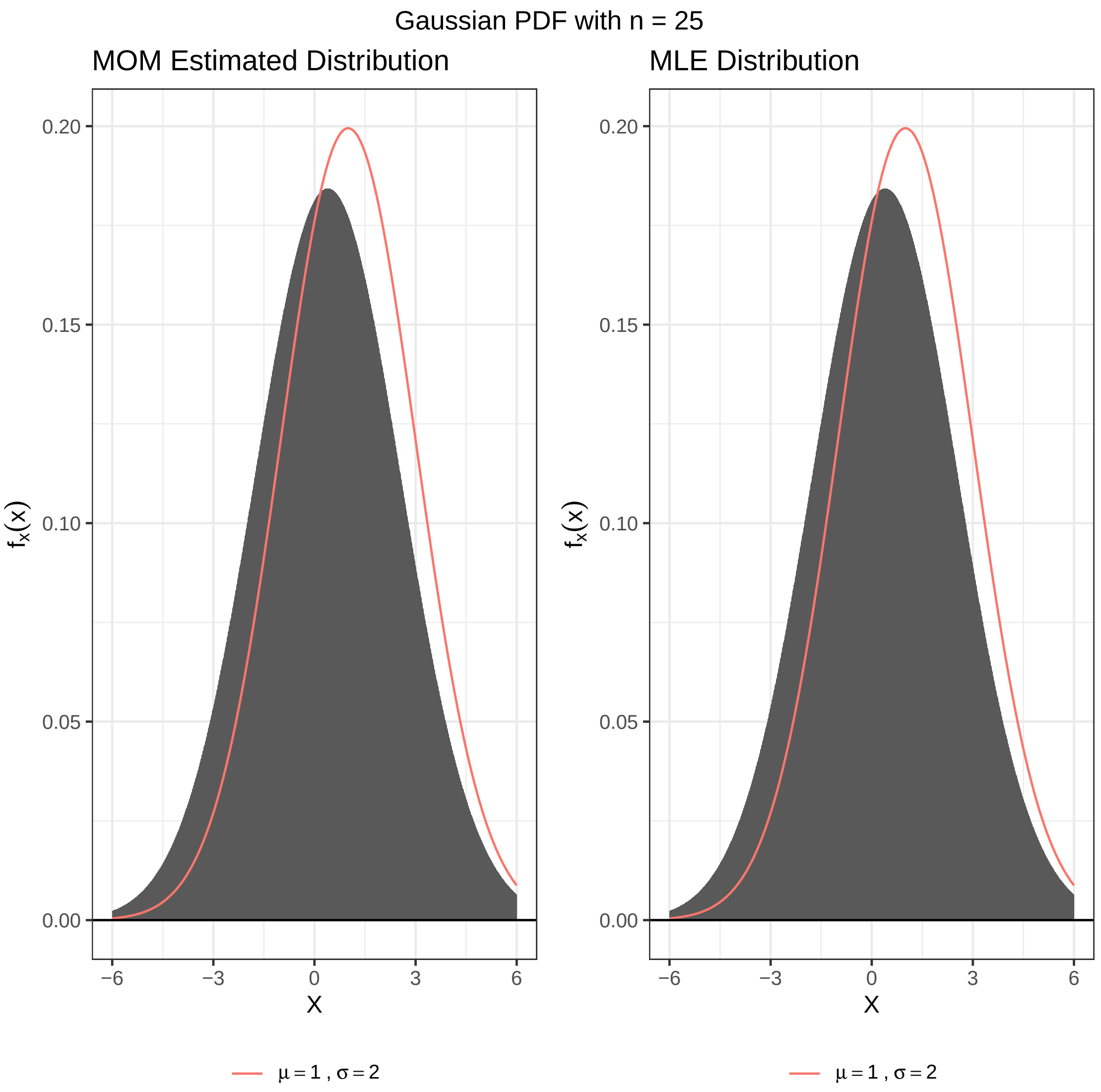
Solution:

```
#################################### n = 25, par 1 ############################
#MOM: n = 10
mom.m0.sd1 <- nleqslv(x = c(0,1),
                      fn = norm.mom,
                      data = n25.m0.sd1)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m0.sd1$x[1],
```

```r
                                   sd = mom.m0.sd1$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#MLE: n = 10
mle.m0.sd1 <- optim(par = c(0,1),
                    fn = norm.ll,
                    data=n25.m0.sd1)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m0.sd1$par[1],
                                   sd = mle.m0.sd1$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 25"))
```

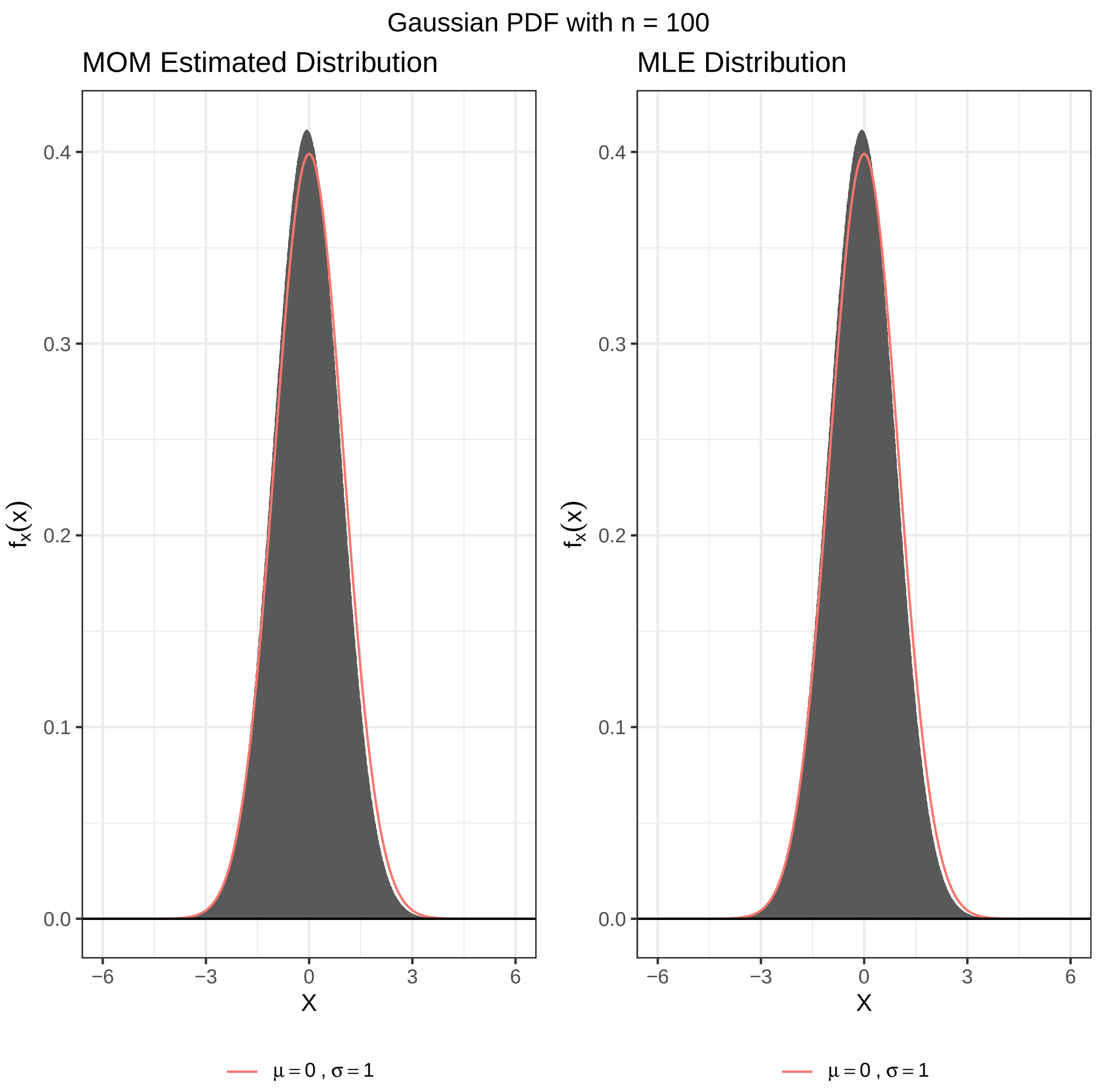


```
################################### n = 25, par 2 ############################
#MOM: n = 10
mom.m1.sd2 <- nleqslv(x = c(1,2),
                      fn = norm.mom,
                      data = n25.m1.sd2)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m1.sd2$x[1],
                                   sd = mom.m1.sd2$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#MLE: n = 10
mle.m1.sd2 <- optim(par = c(1,2),
                    fn = norm.ll,
                    data=n25.m1.sd2)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m1.sd2$par[1],
                                   sd = mle.m1.sd2$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 25"))
```

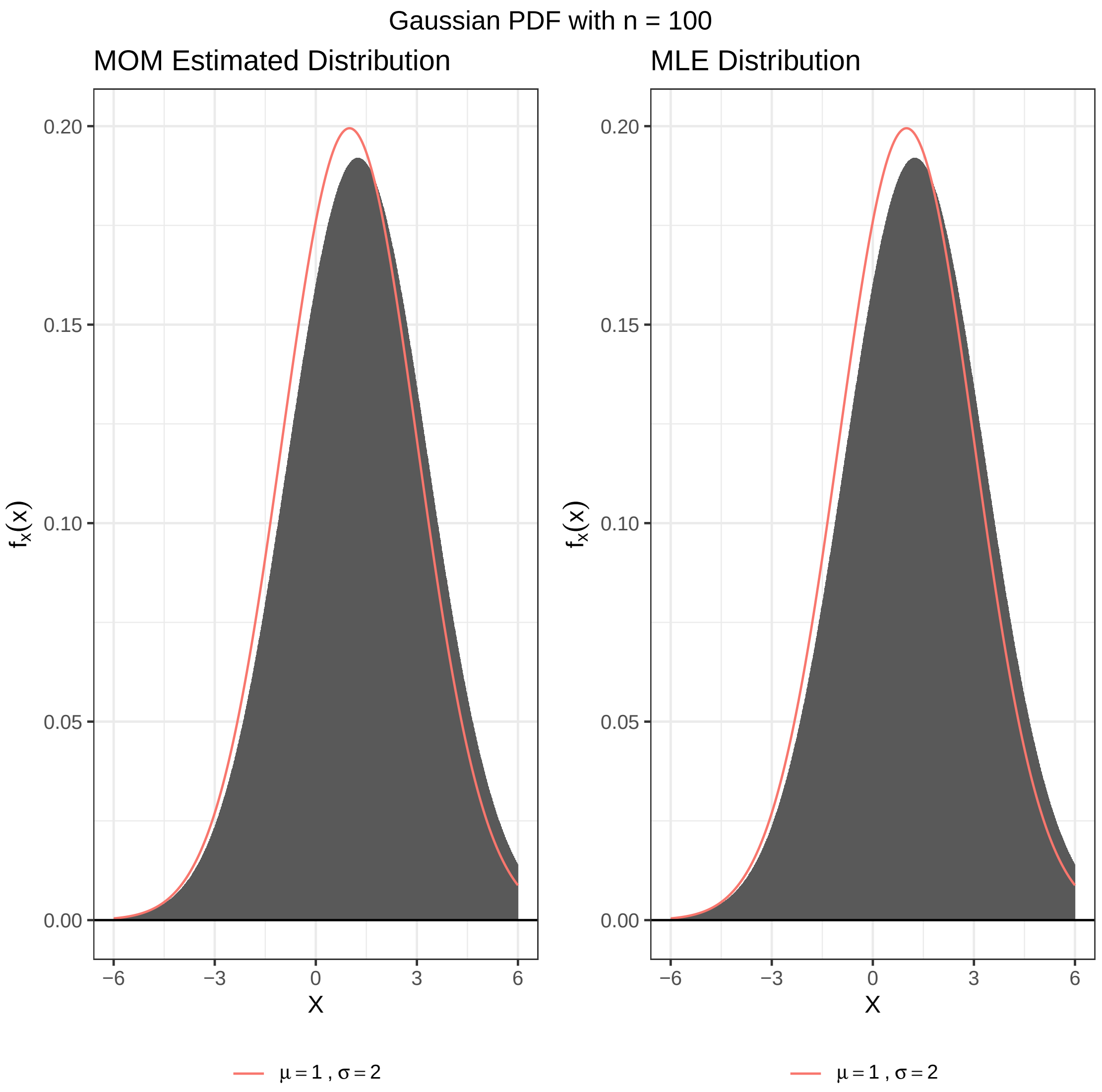


(e) Generate a random sample of size $n = 100$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution:

```
#################################### n = 100, par 1 ###########################
#MOM: n = 10
mom.m0.sd1 <- nleqslv(x = c(0,1),
                      fn = norm.mom,
                      data = n100.m0.sd1)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m0.sd1$x[1],
                                   sd = mom.m0.sd1$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())


#MLE: n = 10
mle.m0.sd1 <- optim(par = c(0,1),
                    fn = norm.ll,
                    data=n100.m0.sd1)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m0.sd1$par[1],
                                   sd = mle.m0.sd1$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())


#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 100"))
```



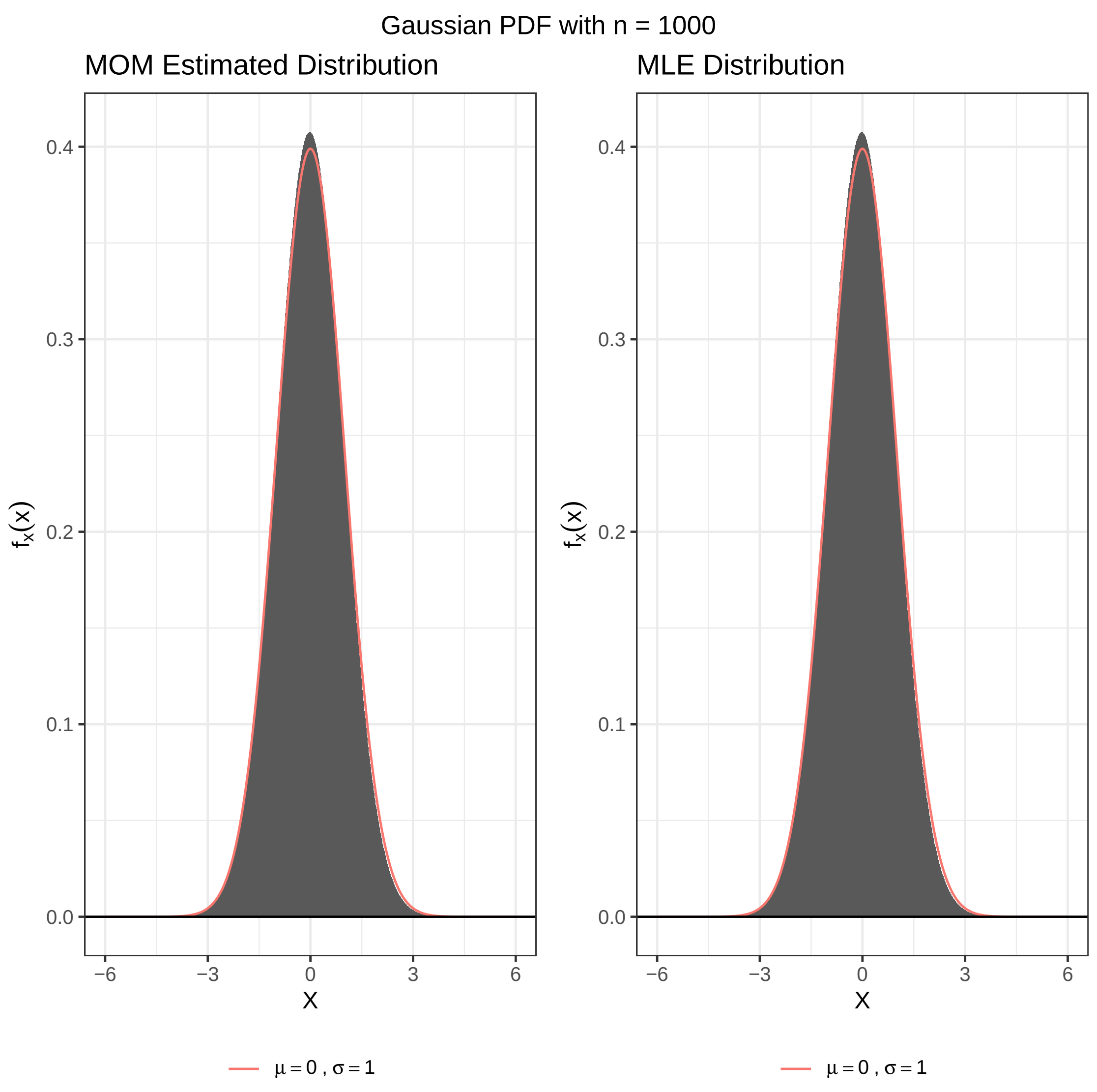
```
################################## n = 100, par 2 ###########################
#MOM: n = 10
mom.m1.sd2 <- nleqslv(x = c(1,2),
                      fn = norm.mom,
                      data = n100.m1.sd2)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m1.sd2$x[1],
                                   sd = mom.m1.sd2$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#MLE: n = 10
mle.m1.sd2 <- optim(par = c(1,2),
                    fn = norm.ll,
                    data=n100.m1.sd2)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m1.sd2$par[1],
                                   sd = mle.m1.sd2$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 100"))
```



(f) Generate a random sample of size $n = 1000$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution:

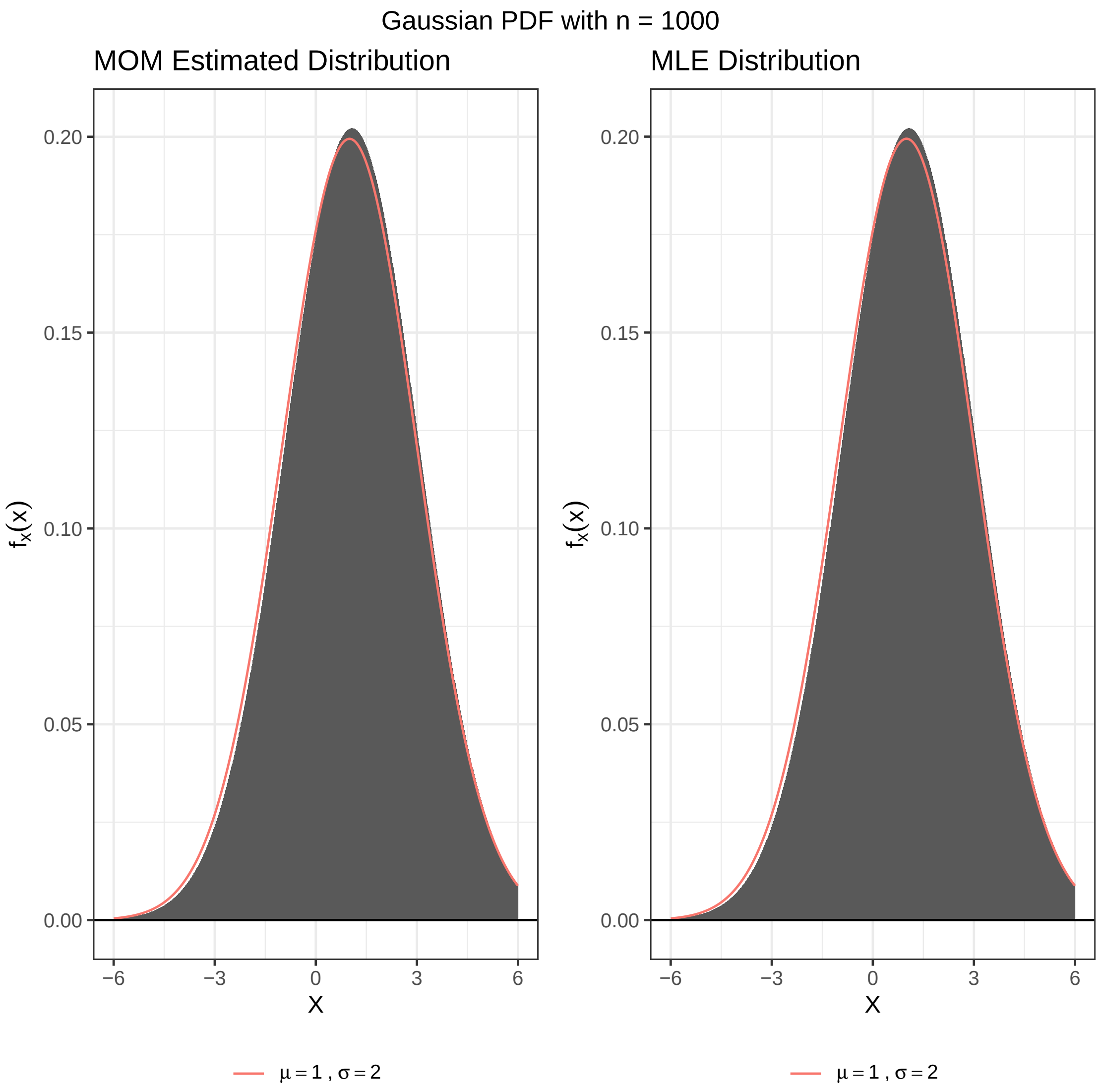
```
#################################### n = 1000, par 1 ##########################
#MOM: n = 10
mom.m0.sd1 <- nleqslv(x = c(0,1),
                      fn = norm.mom,
                      data = n1000.m0.sd1)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m0.sd1$x[1],
```

```r
                                   sd = mom.m0.sd1$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#MLE: n = 10
mle.m0.sd1 <- optim(par = c(0,1),
                    fn = norm.ll,
                    data=n1000.m0.sd1)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 0,sd=1), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m0.sd1$par[1],
                                   sd = mle.m0.sd1$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=0, sd=1"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==0~","~sigma==1)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 1000"))
```



```
################################## n = 1000, par 2 #########################
#MOM: n = 10
mom.m1.sd2 <- nleqslv(x = c(1,2),
                      fn = norm.mom,
                      data = n1000.m1.sd2)

#create MOM plot: n = 10
ggdat.mom <- data.frame(x = seq(-6,6,0.01), #x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mom dist
                                   mean = mom.m1.sd2$x[1],
                                   sd = mom.m1.sd2$x[2]))
#plot
p1.mom <- ggplot(data = ggdat.mom, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
```

```r
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MOM Estimated Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#MLE: n = 10
mle.m1.sd2 <- optim(par = c(1,2),
                    fn = norm.ll,
                    data=n1000.m1.sd2)

#create MLE plot: n = 10
ggdat.mle <- data.frame(x = seq(-6,6,0.01), # x axis
                        F1 = dnorm(seq(-6,6,0.01),mean = 1,sd=2), #true dist
                        F2 = dnorm(seq(-6,6,0.01), #mle dist
                                   mean = mle.m1.sd2$par[1],
                                   sd = mle.m1.sd2$par[2]))
#plot
p1.mle <- ggplot(data = ggdat.mle, aes(x=x, y = F2))+
  geom_histogram(stat='identity')+
  geom_line(aes(y = F1,color = "m=1, sd=2"))+
  geom_hline(yintercept = 0)+
  theme_bw()+
  xlab("X")+
  ylab(bquote(f[x](x)))+
  ggtitle("MLE Distribution")+
  scale_color_discrete(labels=c(bquote(mu==1~","~sigma==2)))+
  theme(legend.position = "bottom",
        legend.text = element_text(margin = margin(r = 15))) +
  theme(legend.title = element_blank())

#combine plots
grid.arrange(arrangeGrob(p1.mom,
                         p1.mle,
                         nrow = 1,
                         top = "Gaussian PDF with n = 1000"))
```



(g) Comment on the results of parts (c)-(f).

Solution:

As expected, as n increases the estimated distributions become more similar to the true distributions. The MOM and MLE estimators seem to be comparable estimators, and usually come up with very similar results depending on the random sample. An interesting observation is that it seems the estimators do a better job with the first set of parameters than the second.

3. Select a discrete distribution (not the Poisson). It does not have to be one that we cover in the notes! To explore the PMF of your distribution, specify two sets of parameter(s) for your distribution.

   (a) **History** Discuss what types of random variables are modeled with your distribution. Be sure to include a discussion about the support and ensure to provide the mass function, and CDF. This requires some internet research – what's the history of the distribution, why was it created and named? What are some exciting applications of this distribution? Cite all of your sources.

   Solution: The Bernoulli distribution, discussed in both Cipolli (2021) and Sinharay (2010), is a special case of the binomial distribution, in which each observation can take one of two outcomes. We define these as a "success," where the observation of interest occurs, or a "failure," where it does not. Thus, the support for this distribution (or "the set of all possible values of our random variable") is $\chi = \{x \in \{0,1\}\}$ (Cipolli, 163). We use p to denote the probability that a "success" occurs and $q = 1 - p$ to denote the probability that a "failure" occurs. Thus, the probability mass function for the Bernoulli distribution is given by $f_X(x|p) = p^x(1-p)^{1-x}$, where x is either 0 or 1 (Sinharay, 132). We can write this to be robust to other inputs by including an indicator function, I, such that the PMF is given by $f_X(x|p) = p^x(1-p)^{1-x}I(x \in \{0,1\})$ (Cipolli, 173). The cumulative distribution function of the Bernoulli distribution is given by:

$$F_X(x|p) = \begin{cases} 0 & x < 0 \\ 1-p & 0 \leq x < 1 \\ 1 & 1 \leq x \end{cases}$$

   (Sinharay, 132). We could similarly write this using the floor operator and indicator function as:

$$F_X(x|p) = [(1-p)I(\lfloor x \rfloor = 0)] + I(\lfloor x \rfloor \geq 1)$$

   (Cipolli, 173). The Bernoulli distribution, created by Jakob Bernoulli, is often used to in the context of flipping coins because it takes one of two values (Sinharay, 132). Another area of application is in dealing with the probabilities of wins and losses (or any binary outcome kind of system). A fair coin would be represented by a Bernoulli distribution with both p and q equal to 0.5 (while any other parameter values would represent an unfair coin flip). I will investigate when the parameters p=0.5, q=0.5 and p=0.7,q=0.3.

   (b) Show that you have a valid PMF. You can show this approximately by calculating the series in a repeat loop until probability mass evaluations are infinitesimally small.

   Solution: For a PMF to be valid, all values of the PMF must remain within 0 and 1, and the sum of all PMF values over the support must be equal to one:

$$0 \leq f_X(x) \leq 1 \quad \text{for all } x \in \mathbb{R}$$

$$\sum_{-\infty}^{\infty} f_X(x) = \sum_{\chi} f_X(x) = 1$$

   (Cipolli, 167). For a distribution like the Binomial distribution, for example, proving a PMF to be valid could require calculating it for increasing x until the probability is sufficiently small. However, for the Bernoulli distribution, this is a little more direct because the support includes only two possible observations. Thus, for each $p \in (0,1)$ for this distribution, the PMF will take only one of two forms: $f_X(0|p) = p^0(1-p)^{1-0} = 1-p$ or $f_X(1|p) = p^1(1-p)^{1-1} = p$. Because $p \in (0,1)$, each of these possible values for the PMF will remain between 0 and 1, satisfying the first criteria. Further, because $(1-p)+p = 1$ the sum of these is always 1, satisfying the second criteria.

   (c) Find the median for your two sets of parameter(s). Conduct some research to find the median based on our PMF to confirm that your numerical approach is correct.

Solution: When p=0.5 and q=0.5, the median value depends in some ways on how we define some of the specifics of a median. Because this situation represents a distribution where an equal number of "successes" and "failures" occur, we could say that there is no median. We could also define the median as the in-between value $x = 0.5$. When p=0.7 and q=0.3, the median value is more straightforward. This distribution represents the case when 70% of observations are "successes" and 30% are "failures," and therefore the median value would be a "success" with x=1. In general for the Bernoulli distribution, the median would be equal to the x that maximizes the PMF. Said another way, when $p > q$, the median is 1, and when $p < q$, the median is 0. The border case when $p = q$ and $f_X(0|p) = f_X(1|p)$, as discussed above, depends on how we want to define the nuances of the median in our given context (Sinharay, 132).

(d) Graph the PMF for several values of the parameter(s) including the two sets you specified. What does changing the parameter(s) do to the shape of the PMF?
Solution:

```
################################################################
##    ADAPTED FROM CHAPTER 6 NOTES CODE: ch6-Bernoulli.R
################################################################

library("tidyverse")
library("ggplot2")
library("patchwork")

###############################################################
###Bernoulli Distribution #####################################
###############################################################
dbern<-function(x,prob){                #DEFINING THE PMF
  if(prob<0 | prob>1){
    errormsg <- "This function is only valid for success probabilities between 0 and 1."
    stop(errormsg)
  }
  indicator <- rep(0, length(x))
  indicator[x==0] <- 1 # indicator should be one if x=0
  indicator[x==1] <- 1 # indicator should be one if x=1
  fx <- (prob^x * (1-prob)^(1-x)) * indicator # PMF formula
  return(fx)
}

pbern<-function(q, prob){
  if(prob<0 | prob>1){
    errormsg<-"This function is only valid for success probabilities between 0 and 1."
    stop(errormsg)
  }
  indicator1 <- rep(1, length(q))
  indicator1[q != 0] <- 0 #indicator should be zero if x!=0
  indicator2 <- rep(1, length(q))
  indicator2[q < 1] <- 0 #indicator should be zero if x<1
  Fx <- (1-prob) * indicator1 + indicator2
  return(Fx)
}
qbern <- function(p, prob){
  if(prob<0 | prob>1){
    errormsg<-"This function is only valid for success probabilities between 0 and 1."
    stop(errormsg)
  }
```

```r
  if(any(p<0) | any(p>1)){
    errormsg<-"This function is only valid for percentiles between 0 and 1."
    stop(errormsg)
  }
  q <- rep(1, length(p))
  q[p <= 1-prob] <- 0 # should return zero if p<=(1-prob)
  q[p > 1-prob] <- 1  # should return one if p>(1-prob)
  return(q)
}

## Plots for the PMF
plotbern <- function(prob){ # Pass in the success probability
  ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = prob), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = prob))
  ## Plot PMF
  PMF <- ggplot(data = ggdat, aes(x = x)) +
    geom_linerange(aes(ymax = f), ymin = 0) +
    geom_hline(yintercept = 0) +
    theme_bw() +
    ylim(0, 1) +
    xlab("X") +
    ylab(bquote(f[x](x))) +
    ggtitle("Bernoulli PMF", subtitle = paste("p =", prob))

  return(PMF)
}
```

```
(plotbern(prob=0.25))/
  (plotbern(prob=0.50))/(plotbern(prob=0.75))
```

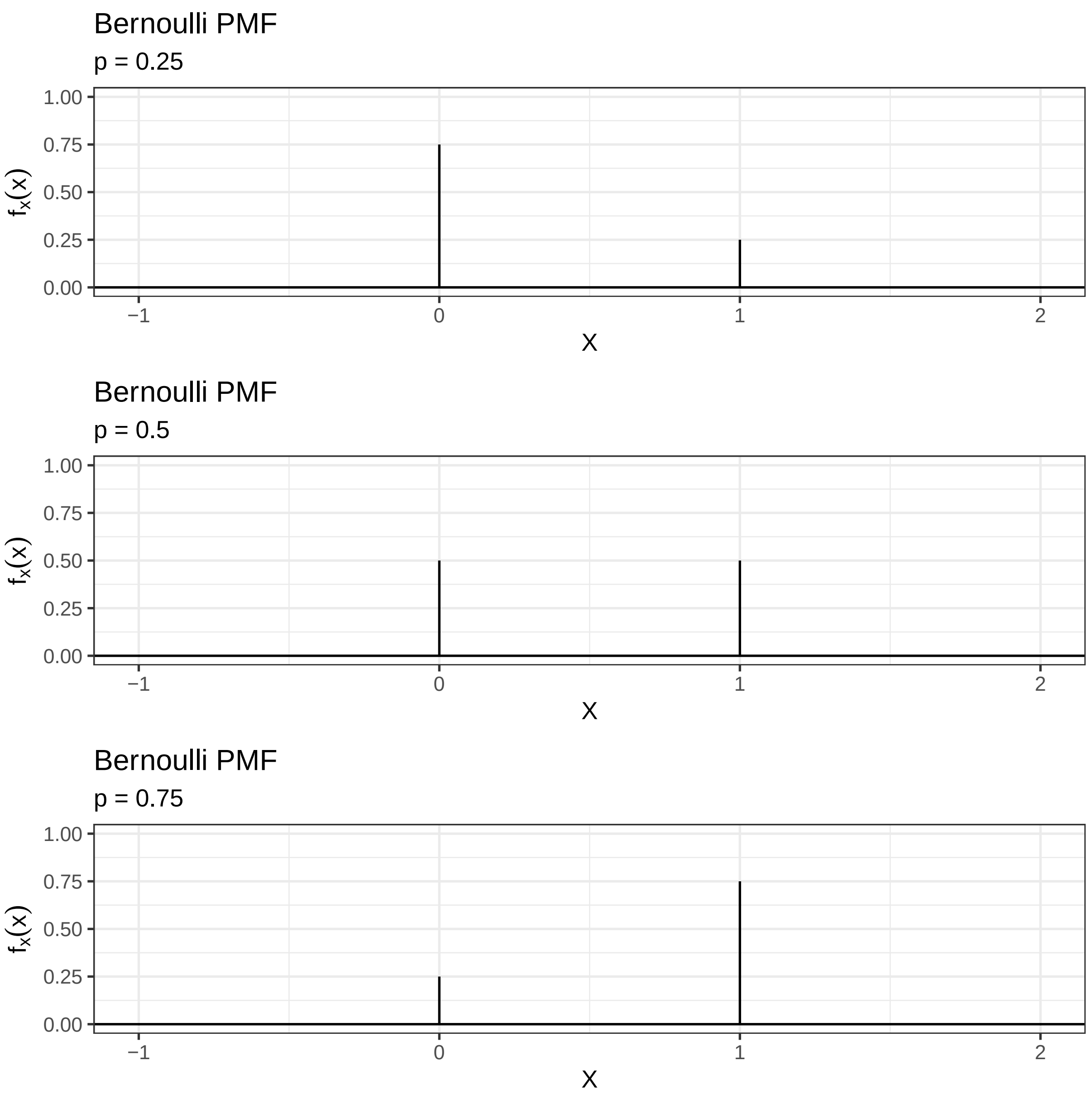


Figure 1: Bernoulli Distribution PMFs for p=0.25, 0.5, and 0.75.

For all valid parameter values, the graph of the Bernoulli PMF shows two lines at the values 0 and 1, corresponding to the probabilities of those observations. Thereby, the heights of these bars are controlled by the parameter p. Lower parameter values correspond to a higher bar at X=0 and a lower bar at X=1, because the distribution represents a scenario in which it is more likely for a "failure" to occur (X=0) than a "success" (X=1). Higher parameter values correspond to a lower bar at X=0 and a higher bar at X=1, because the distribution represents a scenario in which it is more likely for a "success" to occur (X=1) than a "failure" (X=0). Three example PMFs of the Bernoulli distribution with varied parameter values are shown in figure 1. I create these figures using Wickham (2016), Wickham et al. (2019), and Pedersen (2020), and these are used throughout the homework to create plots.

(e) Graph the CDF for the same values of the parameter(s) as you did in Question 3d. What does changing the parameter(s) do to the shape of the CDF? Comment on the aspects of the CDFs that show that the CDF is valid.
Solution:

```
        ##############################################################
        ##    ADAPTED FROM CHAPTER 6 NOTES CODE: ch6-Bernoulli.R
        ##############################################################

## Plots for the CDF
plotbern <- function(prob){ # Pass in the success probability
  ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = prob), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = prob))

  ggdat.openpoints <- data.frame(x = ggdat$x,
                                 y = pbern(ggdat$x-1, prob = prob))
  ggdat.closedpoints <- data.frame(x = ggdat$x,
                                   y = pbern(ggdat$x,prob=prob))
  CDF<-ggplot(data = ggdat, aes(x = x, y = F)) +
    geom_step()+
    geom_point(data = ggdat.openpoints, aes(x = x, y = y), shape = 1) +
    geom_point(data = ggdat.closedpoints, aes(x = x, y = y)) +
    theme_bw()+
    xlab("X")+
    ylab(bquote(F[x](x)))+
    ggtitle("Bernoulli CDF",subtitle=(paste("p =", prob)))
    return(CDF)
}
```

```
(plotbern(prob=0.25))/(plotbern(prob=0.50))/(plotbern(prob=0.75))
```

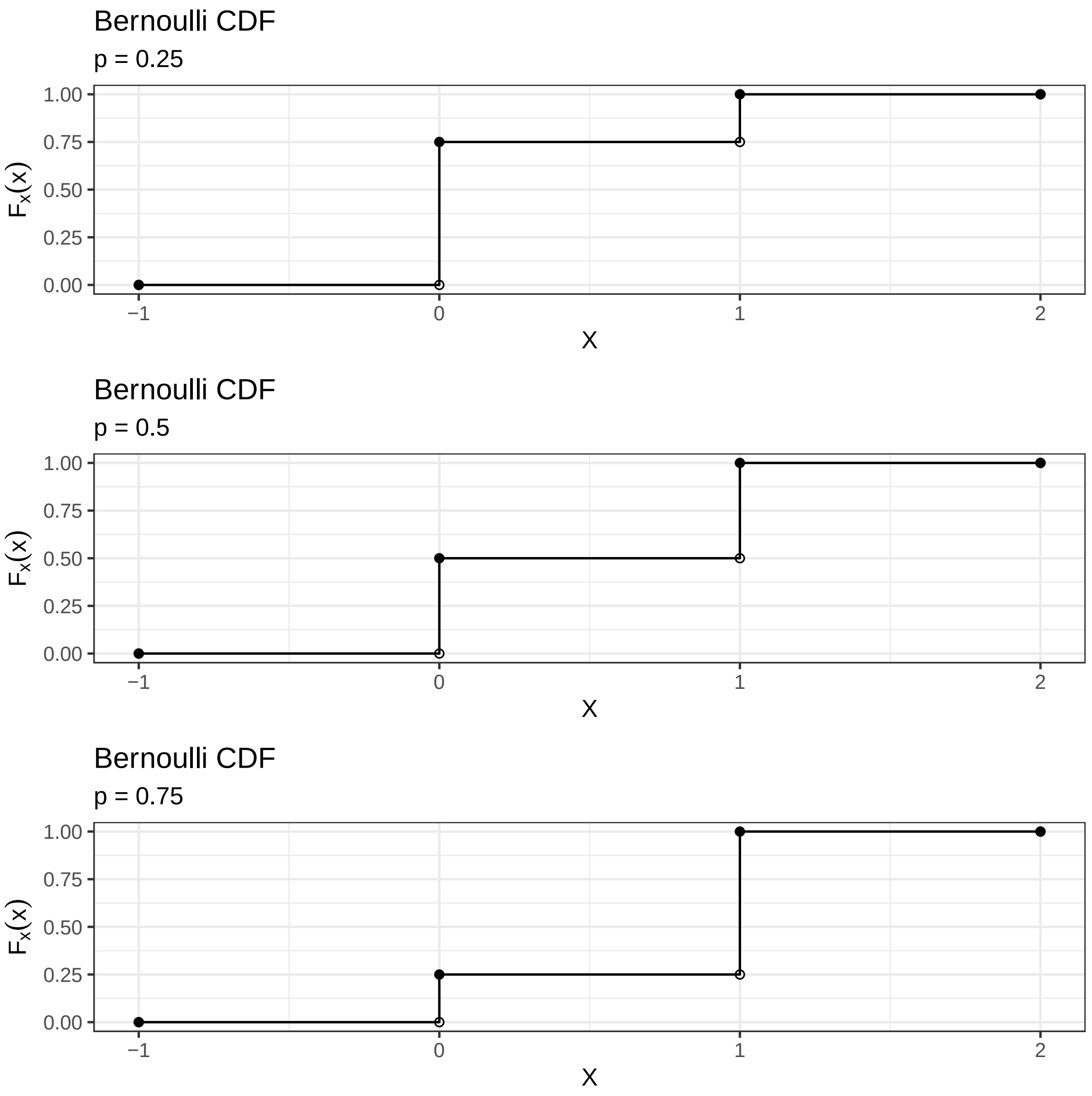


Figure 2: Bernoulli Distribution CDFs for p=0.25, 0.5, and 0.75.

The changes in the CDF with varied parameter values are very similar to those of the PMF; because the CDF represents the cumulative sum of the PMF bars, before X=0, the CDF is 0. The probability of observing an X lower than 0 is zero. We can see that we have a valid CDF because at and after X=1, the CDF is 1. We are guaranteed to observe values only between 0 and 1 inclusive. As with the PMF bars, lower parameter values correspond to a higher bar at X=0 (and therefore a lower bar at X=1 that takes the CDF to 1), because the distribution represents a scenario in which it is more likely for a "failure" to occur (X=0) than a "success" (X=1). Similarly again, higher parameter values correspond to a lower bar at X=0 (and therefore a higher bar at X=1 that takes the CDF to 1), because the distribution represents a scenario in which it is more likely for a “success” to occur (X=1) than a “failure” (X=0). This results in a CDF that, visually, looks more “concave up” for parameter values greater than 0.5, “concave down” for values less than 0.5, and linear for values equal to 0.5. Three example CDFs of the Bernoulli distribution with varied parameter values are shown in figure 2.

(f) Generate a random sample of size $n = 10, 25, 100$, and 1000 for your two sets of parameter(s). In a 4×2 grid, plot a histogram (with bin size 1) of each set of data and superimpose the true mass function at the specified parameter values. Interpret the results.
Solution:

```
###############################################################
        ##    ADAPTED FROM CHAPTER 6 NOTES CODE (pg176)
        ###############################################################

rbern<-function(n,p){
  support<-c(0,1)
  x<-sample(x=support, size=n, replace=TRUE, prob=dbern(support, p=p))
  return(x)
}

x.10.50   <-data.frame(rbern(10, .5))
x.25.50   <-data.frame(rbern(25, .5))
x.100.50  <-data.frame(rbern(100, .5))
x.1000.50 <-data.frame(rbern(1000, .5))

x.10.75   <-data.frame(rbern(10, .75))
x.25.75   <-data.frame(rbern(25, .75))
x.100.75  <-data.frame(rbern(100, .75))
x.1000.75 <-data.frame(rbern(1000, .75))
## PMF for p=0.5
ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = 0.5),
                    F = pbern(q = (-1:2), prob = 0.5))

g1<-ggplot(NULL) +
  geom_histogram(data=x.10.50,                          #uses random sample
                 aes(x=rbern.10..0.5.),                 #makes histogram
                 breaks=seq(-0.5,1.5,1),                #Set up bins to use
                 fill = "lightgrey", color="black") +   #color the histogram
  xlab("X") +                                           #x axis label
  ylab("Frequency")+                                    #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.5 and n=10") +           #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=10*f, ymin=0),           #Adds PMF
                 color="red")                           #Makes PMF red

g2<-ggplot(NULL) +
  geom_histogram(data=x.25.50,                          #uses random sample
                 aes(x=rbern.25..0.5.),                 #makes histogram
                 breaks=seq(-0.5,1.5,1),                #Set up bins to use
                 fill = "lightgrey", color="black") +   #color the histogram
  xlab("X") +                                           #x axis label
  ylab("Frequency")+                                    #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.5 and n=25") +          #add title to plot
  geom_hline(yintercept=0) +                           #adds a line for the x-axis
  geom_linerange(data=ggdat,                           #uses earlier PMF data
                 aes(x=x, ymax=25*f, ymin=0),          #Adds PMF
                 color="red")                          #Makes PMF red

g3<-ggplot(NULL) +
  geom_histogram(data=x.100.50,                        #uses random sample
                 aes(x=rbern.100..0.5.),               #makes histogram
                 breaks=seq(-0.5,1.5,1),               #Set up bins to use
                 fill = "lightgrey", color="black") +  #color the histogram
  xlab("X") +                                          #x axis label
  ylab("Frequency")+                                   #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.5 and n=100") +         #add title to plot
  geom_hline(yintercept=0) +                           #adds a line for the x-axis
  geom_linerange(data=ggdat,                           #uses earlier PMF data
                 aes(x=x, ymax=100*f, ymin=0),         #Adds PMF
                 color="red")                          #Makes PMF red


g4<-ggplot(NULL) +
  geom_histogram(data=x.1000.50,                       #uses random sample
                 aes(x=rbern.1000..0.5.),              #makes histogram
                 breaks=seq(-0.5,1.5,1),               #Set up bins to use
                 fill = "lightgrey", color="black") +  #color the histogram
  xlab("X") +                                          #x axis label
  ylab("Frequency")+                                   #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.5 and n=1000") +        #add title to plot
  geom_hline(yintercept=0) +                           #adds a line for the x-axis
  geom_linerange(data=ggdat,                           #uses earlier PMF data
                 aes(x=x, ymax=1000*f, ymin=0),        #Adds PMF
                 color="red")                          #Makes PMF red

## PMF for p=0.5
ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = 0.75),
                    F = pbern(q = (-1:2), prob = 0.75))

g5<-ggplot(NULL) +
  geom_histogram(data=x.10.75,                         #uses random sample
                 aes(x=rbern.10..0.75.),               #makes histogram
                 breaks=seq(-0.5,1.5,1),               #Set up bins to use
                 fill = "lightgrey", color="black") +  #color the histogram
  xlab("X") +                                          #x axis label
  ylab("Frequency")+                                   #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.75 and n=10") +         #add title to plot
  geom_hline(yintercept=0) +                           #adds a line for the x-axis
  geom_linerange(data=ggdat,                           #uses earlier PMF data
                 aes(x=x, ymax=10*f, ymin=0),          #Adds PMF
                 color="red")                             #Makes PMF red

g6<-ggplot(NULL) +
  geom_histogram(data=x.25.75,                            #uses random sample
                 aes(x=rbern.25..0.75.),                  #makes histogram
                 breaks=seq(-0.5,1.5,1),                  #Set up bins to use
                 fill = "lightgrey", color="black") +     #color the histogram
  xlab("X") +                                             #x axis label
  ylab("Frequency")+                                      #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.75 and n=25") +            #add title to plot
  geom_hline(yintercept=0) +                              #adds a line for the x-axis
  geom_linerange(data=ggdat,                              #uses earlier PMF data
                 aes(x=x, ymax=25*f, ymin=0),             #Adds PMF
                 color="red")                             #Makes PMF red

g7<-ggplot(NULL) +                                        #left NULL so that we can superimpose
  geom_histogram(data=x.100.75,                           #uses random sample
                 aes(x=rbern.100..0.75.),                 #makes histogram
                 breaks=seq(-0.5,1.5,1),                  #Set up bins to use
                 fill = "lightgrey", color="black") +     #color the histogram
  xlab("X") +                                             #x axis label
  ylab("Frequency")+                                      #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.75 and n=100") +           #add title to plot
  geom_hline(yintercept=0) +                              #adds a line for the x-axis
  geom_linerange(data=ggdat,                              #uses earlier PMF data
                 aes(x=x, ymax=100*f, ymin=0),            #Adds PMF
                 color="red")                             #Makes PMF red


g8<-ggplot(NULL) +                                        #left NULL so that we can superimpose
  geom_histogram(data=x.1000.75,                          #uses random sample
                 aes(x=rbern.1000..0.75.),                 #makes histogram
                 breaks=seq(-0.5,1.5,1),                  #Set up bins to use
                 fill = "lightgrey", color="black") +     #color the histogram
  xlab("X") +                                             #x axis label
  ylab("Frequency")+                                      #y axis label
  ggtitle("Random Bernoulli Distribution",
          subtitle = "With p=0.75 and n=1000") +          #add title to plot
  geom_hline(yintercept=0) +                              #adds a line for the x-axis
  geom_linerange(data=ggdat,                              #uses earlier PMF data
                 aes(x=x, ymax=1000*f, ymin=0),           #Adds PMF
                 color="red")                             #Makes PMF red
```

```
(g1 + g5) / (g2 + g6) / (g3 + g7) / (g4 + g8)
```

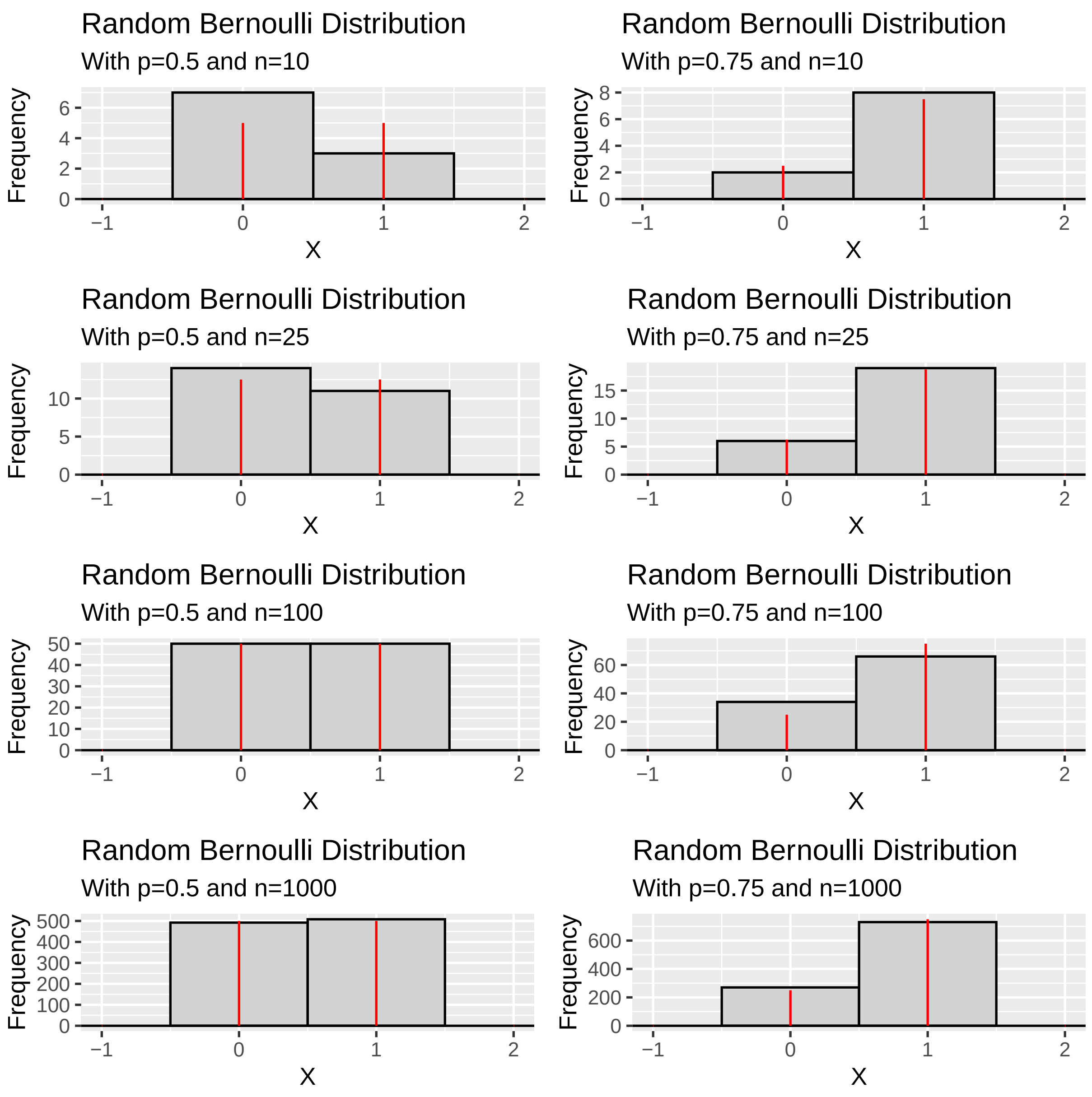


Figure 3: Bernoulli Distributions for randomly generated samples, with varied n and p. The red lines prepresent the PMF, while the histograms show the observed counts.

The greater the size of our random samples, the closer, generally, the spread of the observations we make are to the number we would predict from the PMF. We can see this in that the heights of the grey histogram bars seem to get closer to the red lines that represent the PMF's prediction.

4. Continue with the discrete distribution you selected for Question 3.

   (a) Provide the mean, standard deviation, skewness, and kurtosis of the PMF. Ensure to interpret each.
   Solution:

```r
bern.summ <- function(p){
bern.mean <- p                      #mean = expected value = p
bern.sd   <- sqrt(p*(1-p))          #sqrt of variance
bern.skew <- (1-2*p)/bern.sd
bern.kurt <- (1-6*p*(1-p))/(p*(1-p))
datf <- data.frame(mean=bern.mean, sd=bern.sd, skew=bern.skew, kurt=bern.kurt)
return(datf)
}

bern.summ(0.5)

##   mean  sd skew kurt
## 1  0.5 0.5    0   -2

bern.summ(0.75)

##   mean        sd      skew       kurt
## 1 0.75 0.4330127 -1.154701 -0.6666667
```

For the parameter 0.5, both the mean and standard deviation are 0.5. This makes sense, as with this probability, there are an equal number of observations of 0 and 1, making the mean 0.5 and standard deviation 0.5 (as both 0 and 1 are 0.5 away from the mean 0.5). Because it is a symmetric distribution, the skewness is zero. The kurtosis is -2, representing this flat, platykurtic distribution.

For the parameter 0.75, both the mean is 0.75 and with standard deviation 0.433. Because it is not symmetric distribution, with higher values occuring more frequently, the skewness is -1.15. The kurtosis is -0.67, again representing a platykurtic distribution.

(b) Generate a random sample of size $n = 10, 25, 100$, and 1000 for your two sets of parameter(s). Calculate the sample mean, standard deviation, skewness, and kurtosis. Interpret the results.
Solution:

```r
library(e1071)
library("tidyverse")
library("ggplot2")
library("patchwork")

#############################################################
###Bernoulli Distribution ###################################
#############################################################
rbern<-function(n,p){
  support<-c(0,1)
  x<-sample(x=support, size=n, replace=TRUE, prob=dbern(support, p=p))
  return(x)
}
dbern<-function(x,prob){                #DEFINING THE PMF
  if(prob<0 | prob>1){
errormsg <- "This function is only valid for success probabilities between 0 and 1."
stop(errormsg)
  }
  indicator <- rep(0, length(x))
  indicator[x==0] <- 1 # indicator should be one if x=0
  indicator[x==1] <- 1 # indicator should be one if x=1
  fx <- (prob^x * (1-prob)^(1-x)) * indicator # PMF formula
  return(fx)
}
```

```
pbern<-function(q, prob){
  if(prob<0 | prob>1){
errormsg<-"This function is only valid for success probabilities between 0 and 1."
stop(errormsg)
  }
  indicator1 <- rep(1, length(q))
  indicator1[q != 0] <- 0 #indicator should be zero if x!=0
  indicator2 <- rep(1, length(q))
  indicator2[q < 1] <- 0 #indicator should be zero if x<1
  Fx <- (1-prob) * indicator1 + indicator2
  return(Fx)
}

bern.rand.summ <- function(n,p){
  obs<-rbern(n, p)
  bern.mean <- mean(obs)
  bern.sd   <- sd(obs)
  bern.skew <- skewness(obs)
  bern.kurt <- kurtosis(obs)
  datf <- data.frame(mean=bern.mean, sd=bern.sd, skew=bern.skew, kurt=bern.kurt)
  return(datf)
}
## n=10, p=0.5
bern.rand.summ(10,.5)

##   mean        sd       skew      kurt
## 1  0.7 0.4830459 -0.7452708 -1.572857

## n=25, p=0.5
bern.rand.summ(25,.5)

##   mean  sd  skew    kurt
## 1  0.4 0.5 0.384 -1.9248

## n=100, p=0.5
bern.rand.summ(100,.5)

##   mean        sd        skew      kurt
## 1 0.52 0.5021167 -0.07886612 -2.013617

## n=1000, p=0.5
bern.rand.summ(1000,.5)

##    mean        sd       skew      kurt
## 1 0.494 0.5002142 0.02396573 -2.001424

## n=10, p=0.75
bern.rand.summ(10, .75)

##   mean       sd      skew    kurt
## 1  0.8 0.421637 -1.280722 -0.3675

## n=25, p=0.75
bern.rand.summ(25, .75)

##   mean        sd      skew    kurt
## 1  0.8 0.4082483 -1.410906 -0.0048
## n=100, p=0.75
bern.rand.summ(100, .75)

##   mean        sd      skew     kurt
## 1  0.8 0.4020151 -1.477556 0.185325

## n=1000, p=0.75
bern.rand.summ(1000, .75)

##    mean        sd      skew       kurt
## 1 0.732 0.4431392 -1.046028 -0.9067293
```

The values seen here are close to the ones predicted above. We can see that in the cases that p=0.5, the mean, standard deviation, skewness, and kurtosis remain relatively close to the predicted 0.5, 0.5, 0, and -2, respectively. Similarly, in the cases that p=0.75, the mean, standard deviation, skewness, and kurtosis remain relatively close to the predicted 0.75, 0.433, -1.15, and -0.67. Further, it seems that generally, as n increases, the summary statistics derived from observation approach those predicted. The package e1071, from Meyer et al. (2021), is used for the calculation of skewness and kurtosis.

(c) Generate a random sample of size $n = 10$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram (with bin size 1) of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution: These plots are shown in figure 4. In this and following parts, I create these figures using Wickham (2016), Wickham et al. (2019), and Pedersen (2020).

```
x.10.50   <- data.frame(xs=rbern(10, .5))

###############################################
## MOM
###############################################

## AS EXPLAINED ABOVE
# bern.mom <- function(par,data){
#   p <- par[1]
#   EX1 <- p
#   xbar <- mean(data)
#   return(EX1-xbar)
#   # minimizing this is identical to making the p = mean of the data
# }

mom.p     <- mean(x.10.50$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g1<-ggplot(NULL) +
  geom_histogram(data=x.10.50,                        #uses random sample
                 aes(x=xs),                           #makes histogram
                 breaks=seq(-0.5,1.5,1),              #Set up bins to use
                 fill = "lightgrey", color="black") + #color the histogram
  xlab("X") +                                         #x axis label
  ylab("Frequency")+                                  #y axis label
  ggtitle("True and MOM Estimated",
          subtitle = "With p=0.5 and n=10") +          #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=10*f, ymin=0),           #Adds PMF
                 color="red")                           #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
  p <- par[1]
  ll <- sum(log(dbern(x=data, p)))
  ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
      par = 0.5,
      fn = lbern.ll,
      data = x.10.50$xs,
      lower = 0,
      upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g2<-ggplot(NULL) +
  geom_histogram(data=x.10.50,                          #uses random sample
                 aes(x=xs),                             #makes histogram
                 breaks=seq(-0.5,1.5,1),                #Set up bins to use
                 fill = "lightgrey", color="black") +   #color the histogram
  xlab("X") +                                           #x axis label
  ylab("Frequency")+                                    #y axis label
  ggtitle("True and ML Estimated"
          ,subtitle = "With p=0.5 and n=10") +          #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=10*f, ymin=0),           #Adds PMF
                 color="red")                           #Makes PMF red
```

```
x.10.75   <-data.frame(xs=rbern(10, .75))
###############################################
## MOM
###############################################
mom.p     <- mean(x.10.75$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g3<-ggplot(NULL) +
  geom_histogram(data=x.10.75,                          #uses random sample
                  aes(x=xs),                                       #makes histogram
                  breaks=seq(-0.5,1.5,1),                          #Set up bins to use
                  fill = "lightgrey", color="black") +             #color the histogram
  xlab("X") +                                                      #x axis label
  ylab("Frequency")+                                               #y axis label
  ggtitle("True and MOM Estimated",
          subtitle = "With p=0.75 and n=10") +                      #add title to plot
  geom_hline(yintercept=0) +                                       #adds a line for the x-axis
  geom_linerange(data=ggdat,                                       #uses earlier PMF data
                  aes(x=x, ymax=10*f, ymin=0),                     #Adds PMF
                  color="red")                                     #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
  p <- par[1]
  ll <- sum(log(dbern(x=data, p)))
  ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
      par = 0.5,
      fn = lbern.ll,
      data = x.10.75$xs,
      lower = 0,
      upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g4<-ggplot(NULL) +
  geom_histogram(data=x.10.75,                                     #uses random sample
                  aes(x=xs),                                       #makes histogram
                  breaks=seq(-0.5,1.5,1),                          #Set up bins to use
                  fill = "lightgrey", color="black") +             #color the histogram
  xlab("X") +                                                      #x axis label
  ylab("Frequency")+                                               #y axis label
  ggtitle("True and ML Estimated"
          ,subtitle = "With p=0.75 and n=10") +                     #add title to plot
  geom_hline(yintercept=0) +                                       #adds a line for the x-axis
  geom_linerange(data=ggdat,                                       #uses earlier PMF data
                  aes(x=x, ymax=10*f, ymin=0),                     #Adds PMF
                  color="red")                                     #Makes PMF red
```

(d) Generate a random sample of size $n = 25$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram (with bin size 1) of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution: These plots are shown in figure 5.

```
(g1 + g2) / (g3 + g4)
```

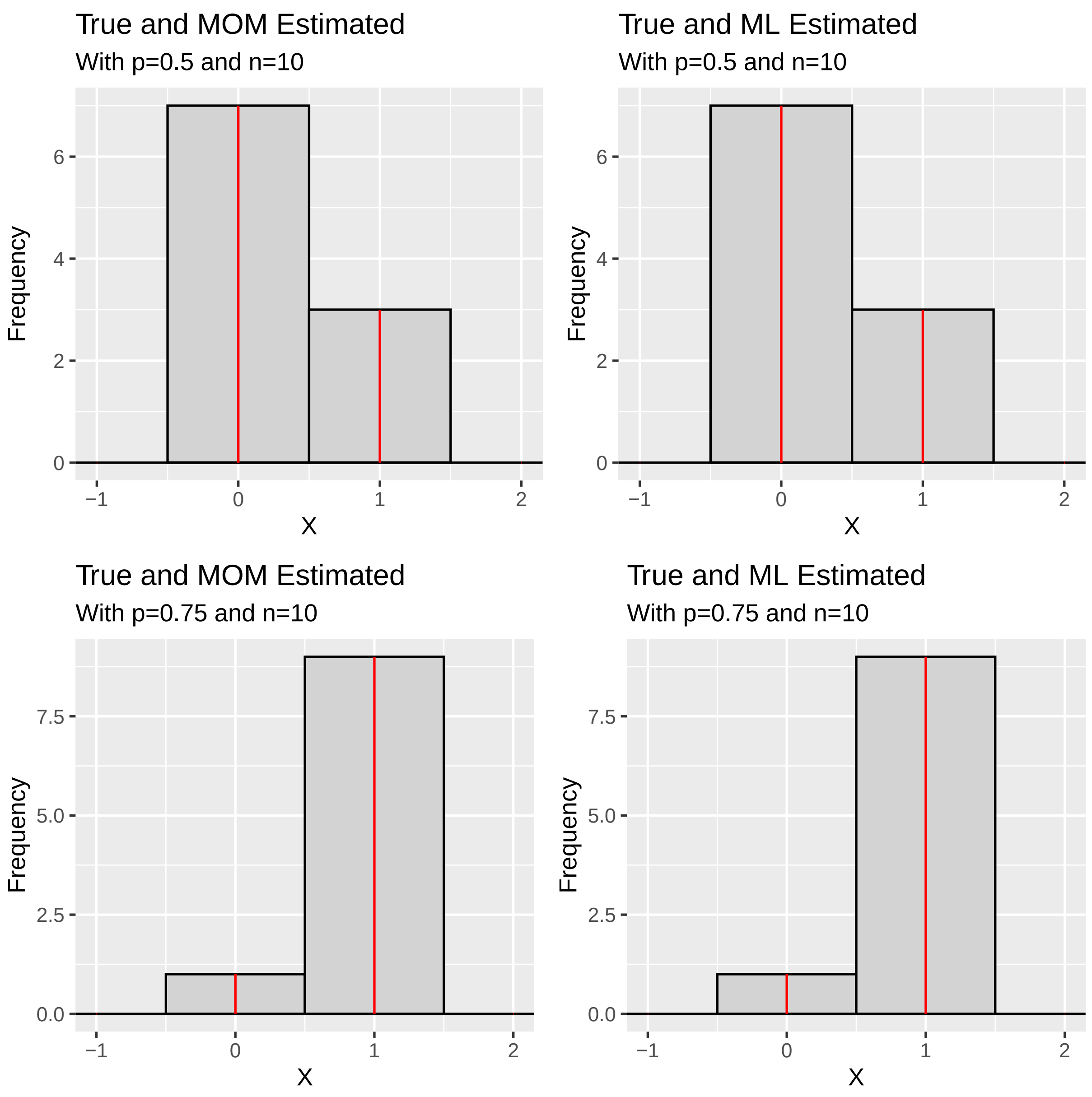


Figure 4: MOM and ML estimators for n=10, p=0.5 (top) and n=10, p=0.75 (bottom). The grey histogram bars show the randomly created data, while the red lines represent the distributions estimated by the MOM and ML methods.

```
x.25.50   <- data.frame(xs=rbern(25, .5))

###############################################
## MOM
###############################################

## AS EXPLAINED ABOVE
# bern.mom <- function(par,data){
#   p <- par[1]
#   EX1 <- p
#   xbar <- mean(data)
#   return(EX1-xbar)
#   # minimizing this is identical to making the p = mean of the data
# }

mom.p     <- mean(x.25.50$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g1<-ggplot(NULL) +
geom_histogram(data=x.25.50,                          #uses random sample
               aes(x=xs),                             #makes histogram
               breaks=seq(-0.5,1.5,1),                #Set up bins to use
               fill = "lightgrey", color="black") +   #color the histogram
xlab("X") +                                           #x axis label
ylab("Frequency")+                                    #y axis label
ggtitle("True and MOM Estimated",
        subtitle = "With p=0.5 and n=25") +           #add title to plot
geom_hline(yintercept=0) +                            #adds a line for the x-axis
geom_linerange(data=ggdat,                            #uses earlier PMF data
               aes(x=x, ymax=25*f, ymin=0),           #Adds PMF
               color="red")                           #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
p <- par[1]
ll <- sum(log(dbern(x=data, p)))
ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
    par = 0.5,
    fn = lbern.ll,
    data = x.25.50$xs,
    lower = 0,
    upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g2<-ggplot(NULL) +
geom_histogram(data=x.25.50,                          #uses random sample
               aes(x=xs),                             #makes histogram
               breaks=seq(-0.5,1.5,1),                #Set up bins to use
               fill = "lightgrey", color="black") +   #color the histogram
xlab("X") +                                           #x axis label
ylab("Frequency")+                                    #y axis label
ggtitle("True and ML Estimated"
        ,subtitle = "With p=0.5 and n=25") +          #add title to plot
geom_hline(yintercept=0) +                            #adds a line for the x-axis
geom_linerange(data=ggdat,                            #uses earlier PMF data
               aes(x=x, ymax=25*f, ymin=0),           #Adds PMF
               color="red")                           #Makes PMF red
```

```
x.25.75   <-data.frame(xs=rbern(25, .75))
###############################################
## MOM
###############################################
mom.p     <- mean(x.25.75$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g3<-ggplot(NULL) +
  geom_histogram(data=x.25.75,                          #uses random sample
                 aes(x=xs),                             #makes histogram
                 breaks=seq(-0.5,1.5,1),                #Set up bins to use
                 fill = "lightgrey", color="black") +   #color the histogram
  xlab("X") +                                           #x axis label
  ylab("Frequency")+                                    #y axis label
  ggtitle("True and MOM Estimated",
          subtitle = "With p=0.75 and n=25") +           #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=25*f, ymin=0),           #Adds PMF
                 color="red")                           #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
  p <- par[1]
  ll <- sum(log(dbern(x=data, p)))
  ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
      par = 0.5,
      fn = lbern.ll,
      data = x.25.75$xs,
      lower = 0,
      upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = mle.p))

g4<-ggplot(NULL) +
  geom_histogram(data=x.25.75,                           #uses random sample
                 aes(x=xs),                              #makes histogram
                 breaks=seq(-0.5,1.5,1),                 #Set up bins to use
                 fill = "lightgrey", color="black") +    #color the histogram
  xlab("X") +                                            #x axis label
  ylab("Frequency")+                                     #y axis label
  ggtitle("True and ML Estimated"
          ,subtitle = "With p=0.75 and n=25") +          #add title to plot
  geom_hline(yintercept=0) +                             #adds a line for the x-axis
  geom_linerange(data=ggdat,                             #uses earlier PMF data
                 aes(x=x, ymax=25*f, ymin=0),            #Adds PMF
                 color="red")                            #Makes PMF red
```

(e) Generate a random sample of size $n = 100$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram (with bin size 1) of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution: These plots are shown in figure 6.

```
x.100.50   <- data.frame(xs=rbern(100, .5))

###############################################
## MOM
###############################################

mom.p     <- mean(x.100.50$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g1<-ggplot(NULL) +
geom_histogram(data=x.100.50,                             #uses random sample
               aes(x=xs),                                #makes histogram
               breaks=seq(-0.5,1.5,1),                   #Set up bins to use
               fill = "lightgrey", color="black") +      #color the histogram
xlab("X") +                                              #x axis label
ylab("Frequency")+                                       #y axis label
ggtitle("True and MOM Estimated",
        subtitle = "With p=0.5 and n=100") +              #add title to plot
geom_hline(yintercept=0) +                               #adds a line for the x-axis
geom_linerange(data=ggdat,                               #uses earlier PMF data
               aes(x=x, ymax=100*f, ymin=0),              #Adds PMF
               color="red")                              #Makes PMF red
```

```
(g1 + g2) / (g3 + g4)
```

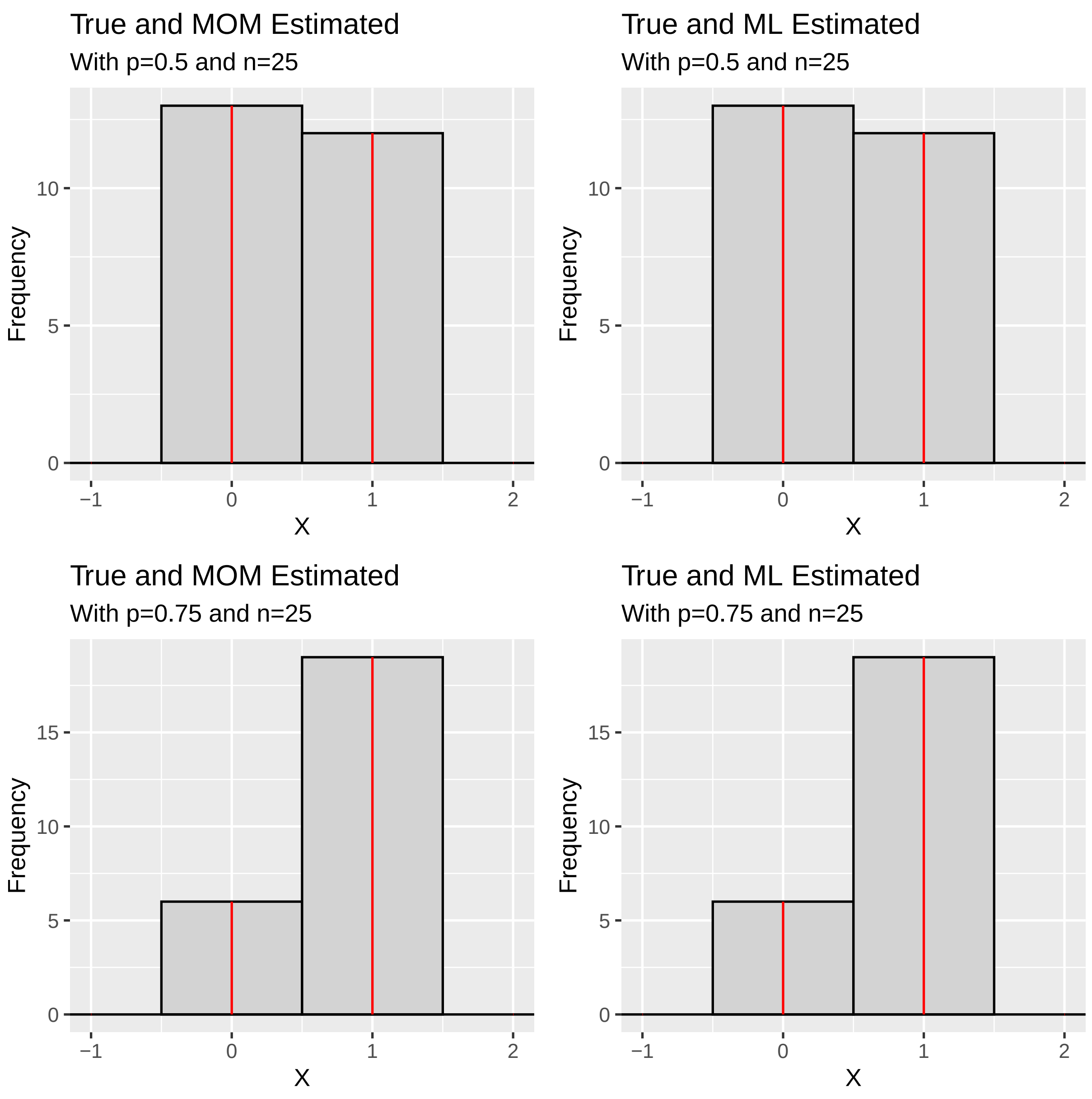


Figure 5: MOM and ML estimators for n=25, p=0.5 (top) and n=10, p=0.75 (bottom). The grey histogram bars show the randomly created data, while the red lines represent the distributions estimated by the MOM and ML methods.

```
#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
p <- par[1]
ll <- sum(log(dbern(x=data, p)))
ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
    par = 0.5,
    fn = lbern.ll,
    data = x.100.50$xs,
    lower = 0,
    upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g2<-ggplot(NULL) +
geom_histogram(data=x.100.50,                            #uses random sample
               aes(x=xs),                               #makes histogram
               breaks=seq(-0.5,1.5,1),                  #Set up bins to use
               fill = "lightgrey", color="black") +     #color the histogram
xlab("X") +                                             #x axis label
ylab("Frequency")+                                      #y axis label
ggtitle("True and ML Estimated"
        ,subtitle = "With p=0.5 and n=100") +            #add title to plot
geom_hline(yintercept=0) +                              #adds a line for the x-axis
geom_linerange(data=ggdat,                              #uses earlier PMF data
               aes(x=x, ymax=100*f, ymin=0),             #Adds PMF
               color="red")                             #Makes PMF red
```

```
x.100.75   <-data.frame(xs=rbern(100, .75))
###############################################
## MOM
###############################################
mom.p     <- mean(x.100.75$xs)

ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = mom.p))

g3<-ggplot(NULL) +
  geom_histogram(data=x.100.75,                            #uses random sample
                 aes(x=xs),                               #makes histogram
                 breaks=seq(-0.5,1.5,1),                  #Set up bins to use
                 fill = "lightgrey", color="black") +     #color the histogram
  xlab("X") +                                             #x axis label
  ylab("Frequency")+                                      #y axis label
  ggtitle("True and MOM Estimated",
          subtitle = "With p=0.75 and n=100") +         #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=100*f, ymin=0),          #Adds PMF
                 color="red")                           #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
  p <- par[1]
  ll <- sum(log(dbern(x=data, p)))
  ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
      par = 0.5,
      fn = lbern.ll,
      data = x.100.75$xs,
      lower = 0,
      upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g4<-ggplot(NULL) +
  geom_histogram(data=x.100.75,                          #uses random sample
                 aes(x=xs),                             #makes histogram
                 breaks=seq(-0.5,1.5,1),                #Set up bins to use
                 fill = "lightgrey", color="black") +   #color the histogram
  xlab("X") +                                           #x axis label
  ylab("Frequency")+                                    #y axis label
  ggtitle("True and ML Estimated"
          ,subtitle = "With p=0.75 and n=100") +          #add title to plot
  geom_hline(yintercept=0) +                            #adds a line for the x-axis
  geom_linerange(data=ggdat,                            #uses earlier PMF data
                 aes(x=x, ymax=100*f, ymin=0),           #Adds PMF
                 color="red")                           #Makes PMF red
```

(f) Generate a random sample of size $n = 1000$ for your two sets of parameter(s). Calculate the method of moments estimator(s) and maximum likelihood estimator(s). In a 1×2 grid, plot a histogram (with bin size 1) of each set of data with (1) the method of moments estimated distribution, (2) the maximum likelihood estimated distribution, and superimpose the true distribution in both.
Solution: These plots are shown in figure 7.

```
x.1000.50   <- data.frame(xs=rbern(1000, .5))

###############################################
## MOM
###############################################
```

```
(g1 + g2) / (g3 + g4)
```

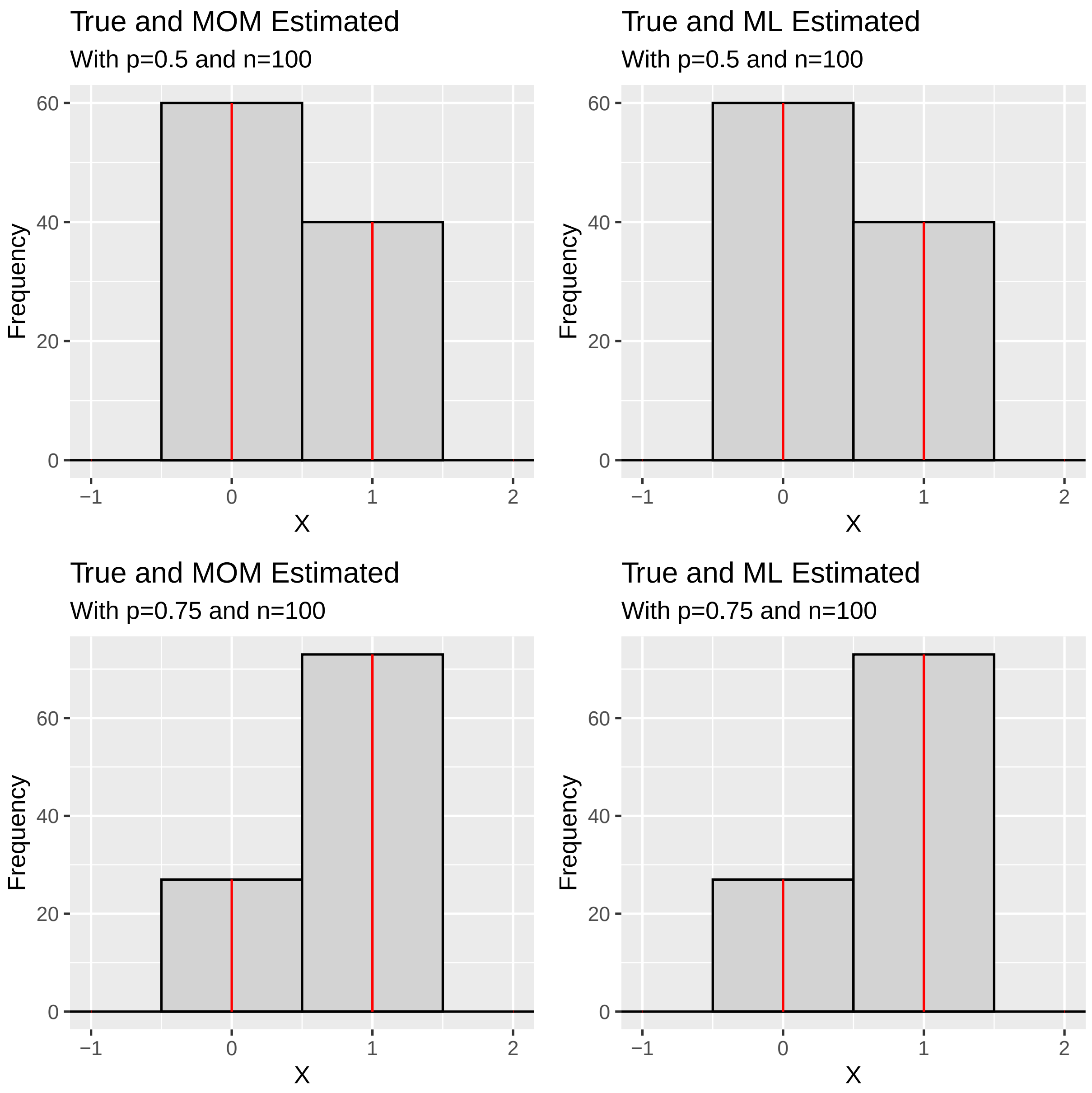


Figure 6: MOM and ML estimators for n=100, p=0.5 (top) and n=10, p=0.75 (bottom). The grey histogram bars show the randomly created data, while the red lines represent the distributions estimated by the MOM and ML methods.

```
## AS EXPLAINED ABOVE
# bern.mom <- function(par,data){
#   p <- par[1]
#   EX1 <- p
#   xbar <- mean(data)
#   return(EX1-xbar)
#   # minimizing this is identical to making the p = mean of the data
# }

mom.p     <- mean(x.1000.50$xs)

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mom.p))

g1<-ggplot(NULL) +
geom_histogram(data=x.1000.50,                              #uses random sample
               aes(x=xs),                                 #makes histogram
               breaks=seq(-0.5,1.5,1),                    #Set up bins to use
               fill = "lightgrey", color="black") +       #color the histogram
xlab("X") +                                               #x axis label
ylab("Frequency")+                                        #y axis label
ggtitle("True and MOM Estimated",
        subtitle = "With p=0.5 and n=1000") +               #add title to plot
geom_hline(yintercept=0) +                                #adds a line for the x-axis
geom_linerange(data=ggdat,                                #uses earlier PMF data
               aes(x=x, ymax=1000*f, ymin=0),               #Adds PMF
               color="red")                               #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
p <- par[1]
ll <- sum(log(dbern(x=data, p)))
ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
    par = 0.5,
    fn = lbern.ll,
    data = x.1000.50$xs,
    lower = 0,
    upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                    f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                    F = pbern(q = (-1:2), prob = mle.p))

g2<-ggplot(NULL) +
geom_histogram(data=x.1000.50,                            #uses random sample
               aes(x=xs),                                 #makes histogram
```

```
                breaks=seq(-0.5,1.5,1),                #Set up bins to use
                fill = "lightgrey", color="black") +   #color the histogram
xlab("X") +                                            #x axis label
ylab("Frequency")+                                     #y axis label
ggtitle("True and ML Estimated"
        ,subtitle = "With p=0.5 and n=1000") +         #add title to plot
geom_hline(yintercept=0) +                             #adds a line for the x-axis
geom_linerange(data=ggdat,                             #uses earlier PMF data
                aes(x=x, ymax=1000*f, ymin=0),         #Adds PMF
                color="red")                           #Makes PMF red
```

```
x.1000.75   <-data.frame(xs=rbern(1000, .75))
###############################################
## MOM
###############################################
mom.p     <- mean(x.1000.75$xs)

ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = mom.p), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = mom.p))

g3<-ggplot(NULL) +
  geom_histogram(data=x.1000.75,                          #uses random sample
                 aes(x=xs),                               #makes histogram
                 breaks=seq(-0.5,1.5,1),                  #Set up bins to use
                 fill = "lightgrey", color="black") +     #color the histogram
  xlab("X") +                                             #x axis label
  ylab("Frequency")+                                      #y axis label
  ggtitle("True and MOM Estimated",
          subtitle = "With p=0.75 and n=1000") +          #add title to plot
  geom_hline(yintercept=0) +                              #adds a line for the x-axis
  geom_linerange(data=ggdat,                              #uses earlier PMF data
                 aes(x=x, ymax=1000*f, ymin=0),           #Adds PMF
                 color="red")                             #Makes PMF red

#######################################################
# MLE
# ADAPTED FROM Lecture 10 Code: Continuous Distribution and Point Estimation
#######################################################
lbern.ll<-function(par, data, neg=T){
  p <- par[1]
  ll <- sum(log(dbern(x=data, p)))
  ifelse(neg, -ll, ll)
}
mle <- optim(method = 'Brent',
      par = 0.5,
      fn = lbern.ll,
      data = x.1000.75$xs,
      lower = 0,
      upper = 1)
mle.p <- mle$par

ggdat <- data.frame(x = (-1:2),
                      f = dbern(x = (-1:2), prob = mle.p), #SETS X AXES NICELY
                      F = pbern(q = (-1:2), prob = mle.p))

g4<-ggplot(NULL) +
  geom_histogram(data=x.1000.75,                               #uses random sample
                 aes(x=xs),                                    #makes histogram
                 breaks=seq(-0.5,1.5,1),                       #Set up bins to use
                 fill = "lightgrey", color="black") +          #color the histogram
  xlab("X") +                                                  #x axis label
  ylab("Frequency")+                                           #y axis label
  ggtitle("True and ML Estimated"
          ,subtitle = "With p=0.75 and n=1000") +              #add title to plot
  geom_hline(yintercept=0) +                                   #adds a line for the x-axis
  geom_linerange(data=ggdat,                                   #uses earlier PMF data
                 aes(x=x, ymax=1000*f, ymin=0),                #Adds PMF
                 color="red")                                  #Makes PMF red
```

@

(g) Comment on the results of parts (c)-(f).
Solution: For the Bernoulli Distribution, the methods of moments estimation is quite simplified. Because only the values of 0 and 1 are included in the support, and because each power of both 1 and 0 are equal to 1 and 0 ($1^k = 1$ and $0^k = 0$ for all k), each moment is equal to the original mean of the data. Therefore, the function we would minimize for our estimation would be the parameter p minus the mean of the data, which we can solve easily by setting p equal to the mean of the data. In each case, the method of moments estimator will be able to perfectly represent the randomly generated set of data. With the maximum likelihood estimator, by a similar logic, the method can always exactly find a parameter to match the distribution, because the support can take only two values. We can see this in figures 4-7. In each case, the MOM and ML estimated distributions match the true distributions exactly. Increasing n makes the random distributions more closely match what we would expect (p as the percentage of "success" observations), but increasing n has no effect on how well the MOM and ML methods map the true distributions. For all n values, they are exact (because our support includes only two values and unique solutions exist are guaranteed to exist).

```
(g1 + g2) / (g3 + g4)
```

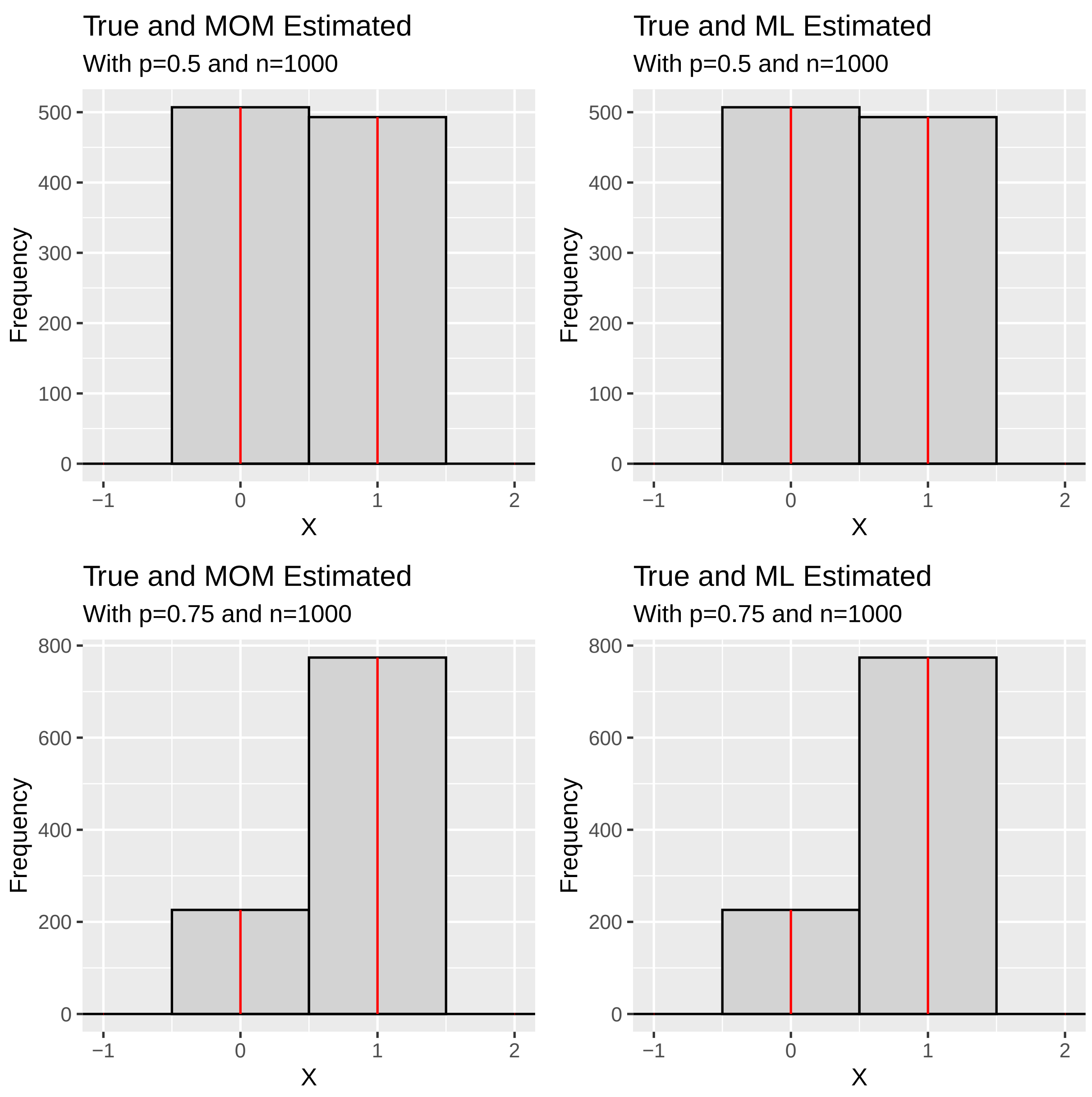


Figure 7: MOM and ML estimators for n=1000, p=0.5 (top) and n=10, p=0.75 (bottom). The grey histogram bars show the randomly created data, while the red lines represent the distributions estimated by the MOM and ML methods.

References

Auguie, B. (2015). *gridExtra: Miscellaneous Functions for "Grid" Graphics.* R package version 2.0.0.

Cipolli, W. (2021). Chapter 6: Probability distributions.

Edwards, S. M. (2020). *lemon: Freshing Up your 'ggplot2' Plots.* R package version 0.4.5.

Hasselman, B. (2018). *nleqslv: Solve Systems of Nonlinear Equations.* R package version 3.3.2.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., and Leisch, F. (2021). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien.* R package version 1.7-6.

Pedersen, T. L. (2020). *patchwork: The Composer of Plots.* R package version 1.1.1.

R Core Team (2021). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

Sinharay, S. (2010). *International Encylopedia of Education (Third Edition).*

Stahl, S. (2006). The evolution of the normal distribution. *Mathematics Magazine*, 79(2):96–113.

Wallis, K. F. (2014). The two-piece normal, binormal, or double gaussian distribution: Its origin and rediscoveries. *Statistical Science*, 29(1):106–112.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., Francois, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Muller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.