A3

Yuqi Gu

7/1/2024

```
setwd('/Users/yuki/Desktop/STAT 341/Assignments/A3')
EDM <- read.csv("EDM.csv")
gameswatched <- read.csv("gamesWatched.csv")
```

QUESTION 1: Parameter Estimation in the Poisson Distribution

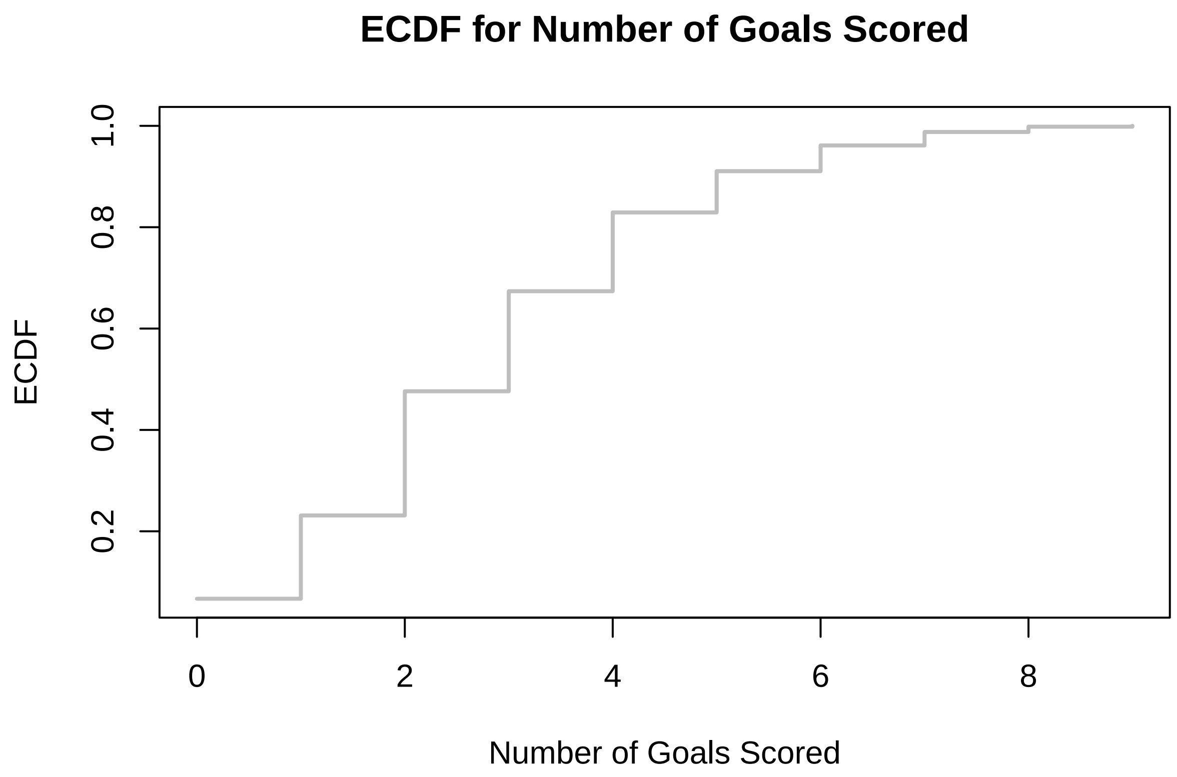
(a)

```
EDM.pop <- EDM[EDM$situation == "all",]
summary(EDM.pop$goalsFor)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   2.000   3.000   2.865   4.000   9.000
```

(b)

```
ECDF <- function(y_u){
  sapply(y_u, function(y){mean(y_u<=y)})
}
y = EDM.pop$goalsFor
plot(sort(y), ECDF(sort(y)), xlab="Number of Goals Scored", ylab="ECDF",
     type = "s", col = "grey", lwd = 2, main="ECDF for Number of Goals Scored")
```



(c)

$$
\begin{aligned}
l(\theta;\mathcal{P}) &= \log\left(L(\theta;\mathcal{P})\right) \\
&= N\bar{y}(log(\theta)) - N\theta - \sum_{u\in\mathcal{P}} \log(y_u!)
\end{aligned}
$$

$$l'(\theta;\mathcal{P}) = \frac{N\bar{y}}{\theta} - N$$

To get $\hat{\theta}_{MLE}$, we solve the following equation.

$$l'(\theta;\mathcal{P}) = \frac{N\bar{y}}{\theta} - N = 0$$

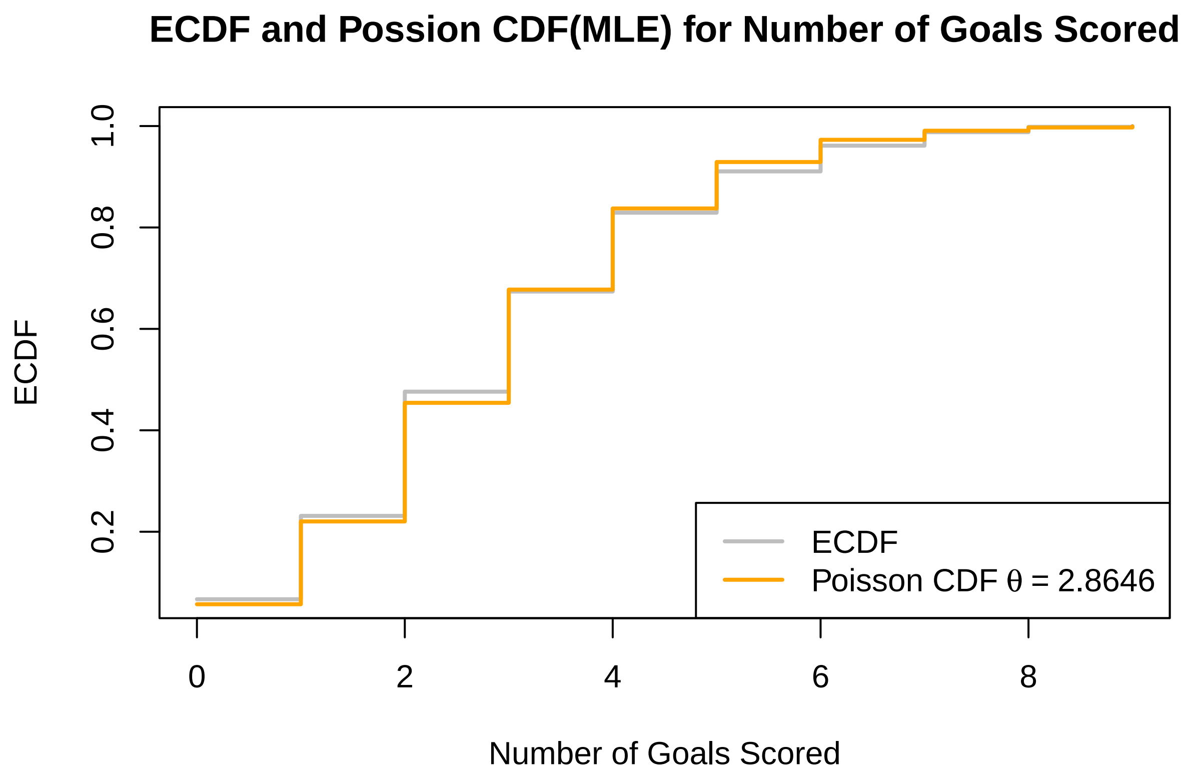
Thus, the maximum likelihood estimate $\hat{\theta}_{MLE} = \bar{y}$.

```
theta_MLE <- mean(y)
theta_MLE
```

```
## [1] 2.864625
```

(d)

```r
plot(sort(y), ECDF(sort(y)), xlab="Number of Goals Scored", ylab="ECDF",
     type = "s", col = "grey", lwd = 2, main="ECDF and Possion CDF(MLE) for Number of Goals Scored")
lines(x = seq(0,9,0.1), y = ppois(q = seq(0,9,0.1), lambda = theta_MLE),
      type = "s", col = "orange", lwd = 2)
legend("bottomright", legend = c("ECDF", expression("Poisson CDF"~theta~"="~2.8646)),
       lwd = 2, col=c("grey", "orange"))
```



This Poisson distribution appears to fit this data well since we can observe that the orange and grey lines are close.

(e)

####(i) We know

$$F(y;\theta) = \sum_{k \le y} \frac{\theta^k e^{-\theta}}{k!}$$

for $y, k \in \mathbb{Z}^{\geq 0}$ and $\theta > 0$, so we get

$$\begin{aligned}
\frac{dF(y;\theta)}{d\theta} &= \frac{d\left(\sum_{k\le y}\frac{\theta^k e^{-\theta}}{k!}\right)}{d\theta} \\
&= \sum_{k=0}^{y}\frac{k\theta^{k-1}e^{-\theta}+\theta^k(-e^{-\theta})}{k!} \\
&= \sum_{k=0}^{y}\frac{k\theta^{k-1}e^{-\theta}}{k!} - \sum_{k=0}^{y}\frac{\theta^k e^{-\theta}}{k!} \\
&= \sum_{k=0}^{y}\frac{k\theta^{k-1}e^{-\theta}}{k!} - F(y;\theta) \\
&= \sum_{k=1}^{y}\frac{\theta^{k-1}e^{-\theta}}{(k-1)!} - F(y;\theta) \\
&= \sum_{m=0}^{y-1}\frac{\theta^{m}e^{-\theta}}{m!} - F(y;\theta) \textit{ by letting } m = k-1 \\
&= F(y-1;\theta) - F(y;\theta)
\end{aligned}$$

(ii)

$$\begin{aligned}
\psi(\theta;\mathcal{P}) &= \frac{d\left(\frac{1}{N}\sum_{u\in\mathcal{P}}\left[F(y_u;\theta)-F(y_u;\mathcal{P})\right]^2\right)}{d\theta} \\
&= \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\left[F(y_u;\theta)-F(y_u;\mathcal{P})\right]\frac{dF(y_u;\theta)}{d\theta}\right) \\
&= \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\left[F(y_u;\theta)-F(y_u;\mathcal{P})\right]\left[F(y_u-1;\theta)-F(y_u;\theta)\right]\right)
\end{aligned}$$

$$\begin{aligned}
\psi'(\theta;\mathcal{P}) &= \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\frac{dF(y_u;\theta)}{d\theta}\left[F(y_u-1;\theta)-F(y_u;\theta)\right]\right) + \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\left[F(y_u;\theta)-F(y_u;\mathcal{P})\right]\left[\frac{dF(y_u-1;\theta)}{d\theta}-\frac{dF(y_u;\theta)}{d\theta}\right]\right) \\
&= \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\left[F(y_u-1;\theta)-F(y_u;\theta)\right]^2\right) + \frac{2}{N}\sum_{u\in\mathcal{P}}\left(\left[F(y_u;\theta)-F(y_u;\mathcal{P})\right]\left[F(y_u-2;\theta)-2F(y_u-1:\theta)+F(y_u;\theta)\right]\right)
\end{aligned}$$

```r
createMDEPsiFns <- function(y) {
  psiFn <- function(theta) {
    2 * mean((ppois(y, theta) - ECDF(y)) * (ppois(y-1, theta) - ppois(y, theta)))
  }
  psiPrimeFn <- function(theta) {
    2 * mean((ppois(y-1, theta) - ppois(y, theta))^2) +
    2 * mean((ppois(y, theta) - ECDF(y)) *
               (ppois(y-2, theta) - 2*ppois(y-1, theta) + ppois(y, theta)))
  }
  list(psiFn = psiFn, psiPrimeFn = psiPrimeFn)
}
```

(iii)

```r
testConvergence <- function(thetaNew, thetaOld, tolerance = 1e-10, relative = FALSE) {
    sum(abs(thetaNew - thetaOld)) < if (relative)
        tolerance * sum(abs(thetaOld)) else tolerance
}

Newton <- function(theta = 0,
                   psiFn, psiPrimeFn,
                   testConvergenceFn = testConvergence,
                   maxIterations = 100,   # maximum number of iterations
                   tolerance = 1E-6,      # parameters for the test
                   relative = FALSE       # for convergence function
) {
  ## Initialize
  converged <- FALSE
  i <- 0
  ## LOOP
  while (!converged & i <= maxIterations) {
    ## Update theta
    thetaNew <- theta - psiFn(theta)/psiPrimeFn(theta)
    ##
    ## Check convergence
    converged <- testConvergenceFn(thetaNew, theta,
                                   tolerance = tolerance,
                                   relative = relative)
    ## Update iteration
    theta <- thetaNew
    i <- i + 1
  }
  ## Return last value and whether converged or not
  list(theta = theta,
       converged = converged,
       iteration = i,
       fnValue = psiFn(theta)
       )
}
```

```r
fn <- createMDEPsiFns(y)
output <- Newton(theta = theta_MLE, psiFn = fn$psiFn, psiPrimeFn = fn$psiPrimeFn)
output
```

(iv)

```
## $theta
## [1] 2.831762
##
## $converged
## [1] TRUE
##
## $iteration
## [1] 3
```

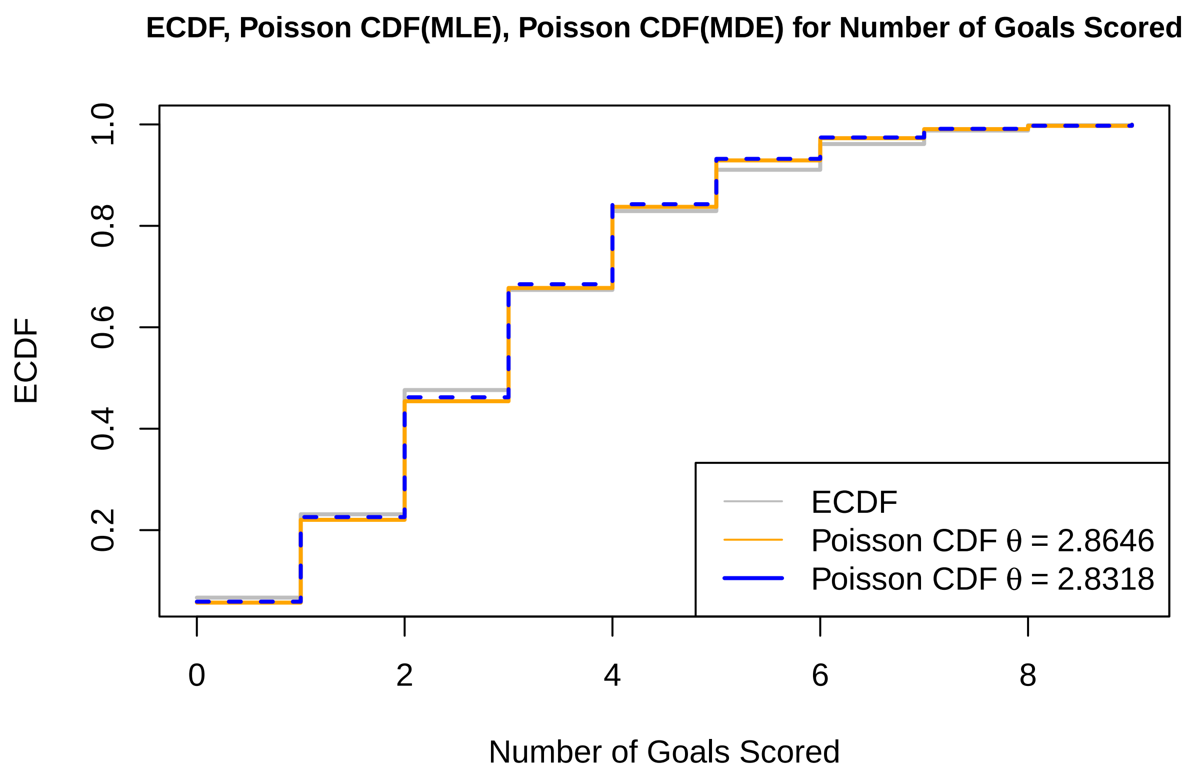
```
##
## $fnValue
## [1] -7.166389e-18
```

```
theta_MDE <- output$theta
theta_MDE
```

```
## [1] 2.831762
```

(f)

```
plot(sort(y), ECDF(sort(y)), xlab="Number of Goals Scored", ylab="ECDF", lty = 1,
     type = "s", col = "grey", lwd = 2, cex.main = 0.9,
     main = "ECDF, Poisson CDF(MLE), Poisson CDF(MDE) for Number of Goals Scored")
lines(x = seq(0,9,0.1), y = ppois(q = seq(0,9,0.1), lambda = theta_MLE),
      type = "s", col = "orange", lwd = 2, lty = 1)
lines(x = seq(0,9,0.1), y = ppois(q = seq(0,9,0.1), lambda = output$theta),
      type = "s", lty = 2, col = "blue", lwd = 2)
legend("bottomright",
       legend = c("ECDF",
                  expression("Poisson CDF"~theta~"="~2.8646),
                  expression("Poisson CDF"~theta~"="~2.8318)),
       lwd = c(1, 1, 2),
       col = c("grey", "orange", "blue"))
```



The fitted Poisson model with the two estimation methods are close. They both fit this data well.

QUESTION 2: Exploration of MLE and MDE Sampling Distribution

(a)

```
set.seed(341)
s <- sample(y, size=50)
theta_MLE_sample <- mean(s)
theta_MLE_sample
```

```
## [1] 2.3
```

```
fn2 <- createMDEPsiFns(s)
output2 <- Newton(theta = theta_MLE_sample, psiFn = fn2$psiFn,
                  psiPrimeFn = fn2$psiPrimeFn)
output2
```

```
## $theta
## [1] 2.360147
##
## $converged
## [1] TRUE
##
## $iteration
## [1] 3
##
## $fnValue
## [1] -5.197171e-17
```

```
theta_MDE_sample <- output2$theta
theta_MDE_sample
```

```
## [1] 2.360147
```

The ML estimate of θ is 2.3 and the MD estimate of θ is 2.360147.

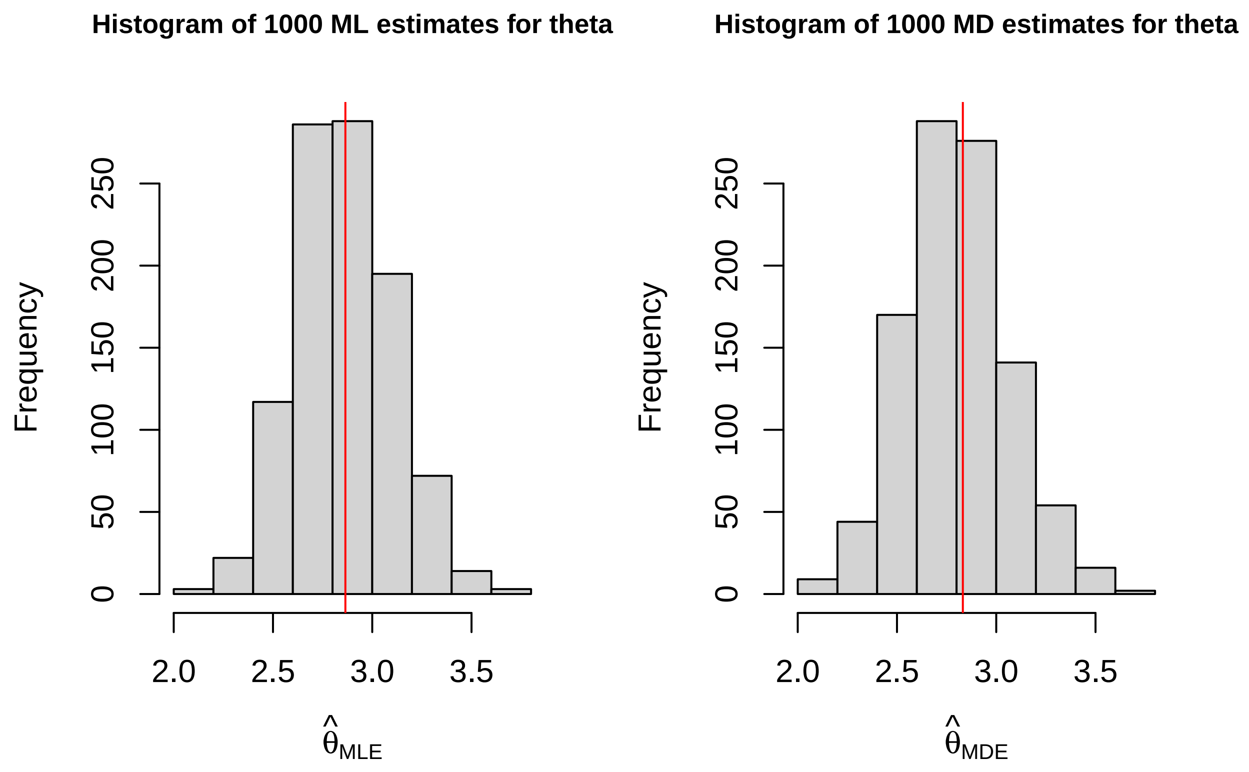
(b)

```
set.seed(341)
thetas <- matrix(0, nrow=1000, ncol = 2)
for (i in 1:1000) {
  s <- sample(y, size=50)
  thetas[i,1] = mean(s)
  fns <- createMDEPsiFns(s)
  thetamle <- mean(s)
  outputs <- Newton(theta = thetamle, psiFn = fns$psiFn,
                    psiPrimeFn = fns$psiPrimeFn)
  thetas[i,2] = outputs$theta
}
```

```
par(mfrow = c(1,2))

hist(thetas[,1], xlab = bquote(hat(theta)[MLE]),
     main = "Histogram of 1000 ML estimates for theta", cex.main = 0.8)
abline(v=theta_MLE, col = "red")

hist(thetas[,2], xlab = bquote(hat(theta)[MDE]),
     main = "Histogram of 1000 MD estimates for theta", cex.main = 0.8)
abline(v=theta_MDE, col = "red")
```



(c)

```
mean(thetas[,1]) - theta_MLE
```

```
## [1] 0.0009746978
```

```
var(thetas[,1])
```

```
## [1] 0.06112416
```

```
mean((thetas[,1] - theta_MLE)^2)
```

```
## [1] 0.06106399
```

```
mean(thetas[,2]) - theta_MDE
```

```
## [1] -0.03136231
```

```
var(thetas[,2])
```

```
## [1] 0.06919736
```

```
mean((thetas[,2] - theta_MDE)^2)
```

```
## [1] 0.07011176
```

For ML estimation method, bias is 0.0009746978. Variance is 0.06112416. Mean squared error is 0.06106399. For MD estimation method, bias is -0.03136231. Variance is 0.06919736. Mean squared error is 0.07011176.

(d)

ML estimation method is preferable. By referencing the plots constructed in part (b), the range of the xlabs for both plots are from 2 to 3.5. Moreover, the vertical line signifying the population ML estimate is nearly at the center of the left plot, i.e. Histogram of 1000 ML estimates for theta. The vertical line signifying the population MD estimate is nearly at the center of the right plot, i.e. Histogram of 1000 MD estimates for theta. However, by the values calculated in part (c), we observe that the mean squared error of MD estimation is larger than the ML estimation. Thus, ML estimation method is preferable than MD estimation.

QUESTION 3: Horvitz Thompson Estimation

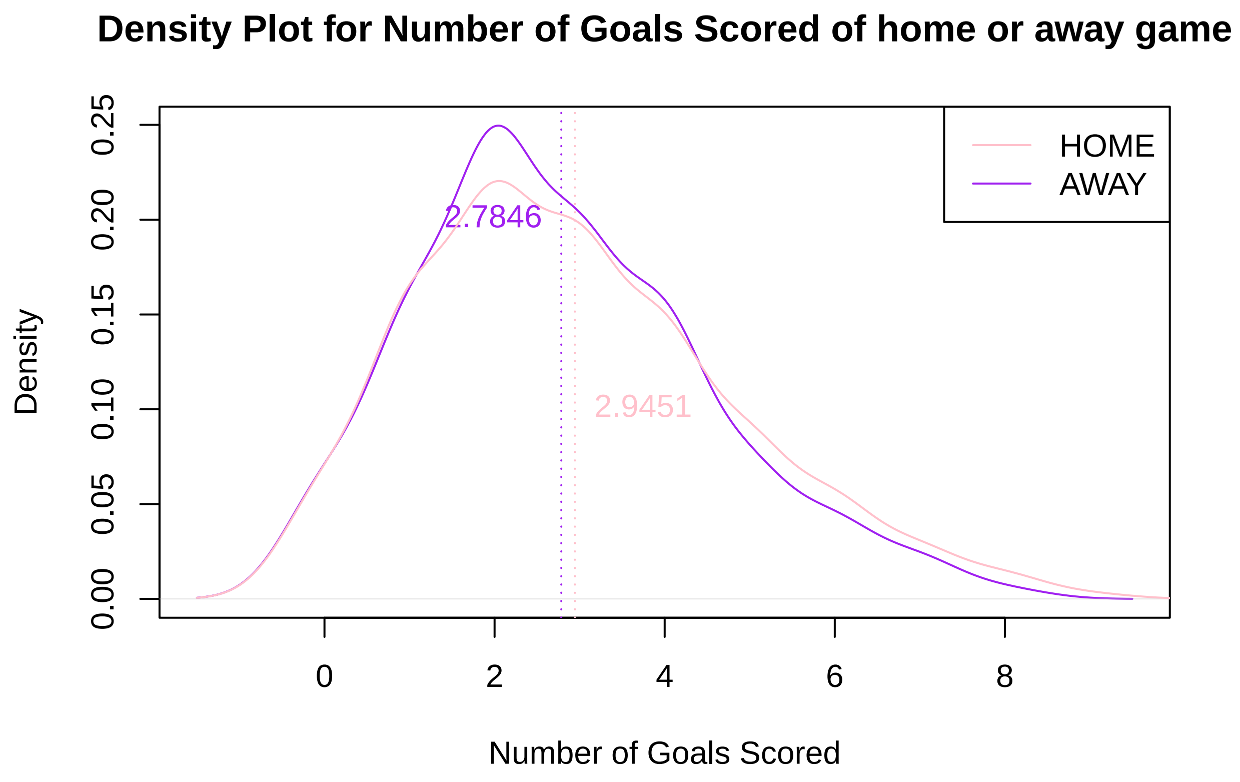
(a)

```
EDM_away <- EDM.pop[EDM.pop$home_or_away == "AWAY",]
EDM_home <- EDM.pop[EDM.pop$home_or_away == "HOME",]


plot(density(EDM_away$goalsFor, bw =0.5), col = "purple",
     xlab = "Number of Goals Scored",
     main = "Density Plot for Number of Goals Scored of home or away game")
abline(v=mean(EDM_away$goalsFor), col = "purple", lty = 3)
text(x = 2.7846, y = 0.2, labels = 2.7846, pos = 2, col = "purple")

lines(density(EDM_home$goalsFor, bw =0.5), col = "pink")
abline(v=mean(EDM_home$goalsFor), col = "pink", lty = 3)
text(x = 2.9451, y = 0.1, labels = 2.9451, pos = 4, col= "pink")

legend ("topright", legend = c("HOME", "AWAY"),
        col = c("pink", "purple"), lty = 1)
```



(b)

```r
hEDM2021 <- EDM_home[EDM_home$season >= 2021,]
aEDM2021 <- EDM_away[EDM_away$season >= 2021,]
hs <- hEDM2021[hEDM2021$gameDate %in% unique(gameswatched)$gameDate, ]
as <- aEDM2021[aEDM2021$gameDate %in% unique(gameswatched)$gameDate, ]


nh <- sum(gameswatched$gameDate %in% hEDM2021$gameDate)
na <- sum(gameswatched$gameDate %in% aEDM2021$gameDate)

Nh <- nrow(hEDM2021)
Na <- nrow(aEDM2021)

pi_uh <- rep(1-(1-1/Nh)^nh, nrow(hs))
t_uh <- hs$goalsFor/Nh
HTavgh <- sum(t_uh/pi_uh)
HTavgh
```

(i)

```
## [1] 3.834544
```

```r
pi_ua <- rep(1-(1-1/Na)^na, nrow(as))
t_ua <- as$goalsFor/Nh
HTavga <- sum(t_ua/pi_ua)
HTavga
```

```
## [1] 2.989311
```

The Horvitz-Thompson estimate of $a(\mathcal{P}_H)$ is 3.834544, and the Horvitz-Thompson estiamte of $a(\mathcal{P}_A)$ is 2.989311.

```r
estVarHT <- function(t_u, pi_u, pi_uv){
  ## t_u = an n element array containing the variate values for the sample
  ## pi_u = an n element array containing the (marginal) inclusion probabilities for the sample
  ## pi_uv = an nxn matrix containing the joint inclusion probabilities for the sample
  delta <- pi_uv - outer(pi_u, pi_u)
  estimateVar <-  sum( (delta/pi_uv) * outer(t_u/pi_u,t_u/pi_u) )
  return(abs(estimateVar))
}

pi_uvh <- matrix(1 - 2*((Nh-1)/Nh)^nh + ((Nh-2)/Nh)^nh, nrow=nrow(hs), ncol=nrow(hs))
diag(pi_uvh) <- pi_uh
pi_uva <- matrix(1 - 2*((Na-1)/Na)^na + ((Na-2)/Na)^na, nrow=nrow(as), ncol=nrow(as))
diag(pi_uva) <- pi_ua

hvar <- estVarHT(t_uh, pi_uh, pi_uvh)
sqrt(hvar)
```

(ii)

```
## [1] 0.4036672
```

```
avar <- estVarHT(t_ua, pi_ua, pi_uva)
sqrt(avar)
```

```
## [1] 0.277629
```

The standard error for the estimate of $a(\mathcal{P}_H)$ is 0.4036672. The standard error for the estimate of $a(\mathcal{P}_A)$ is 0.277629.

```
var <- hvar+avar
sd <- sqrt(var)

HTavg <- HTavgh-HTavga
HTavg + 2*c(-1,1) * sd
```

(iii)

```
## [1] -0.1346139  1.8250801
```

The approximate 95% confidence interval for $a(\mathcal{P}_H) - a(\mathcal{P}_A)$ is [-0.1346139, 1.8250801].

(iv) Since the confidence interval in part iii. contains 0, we conclude that there is absence of the so-called "home ice advantage".

(c)

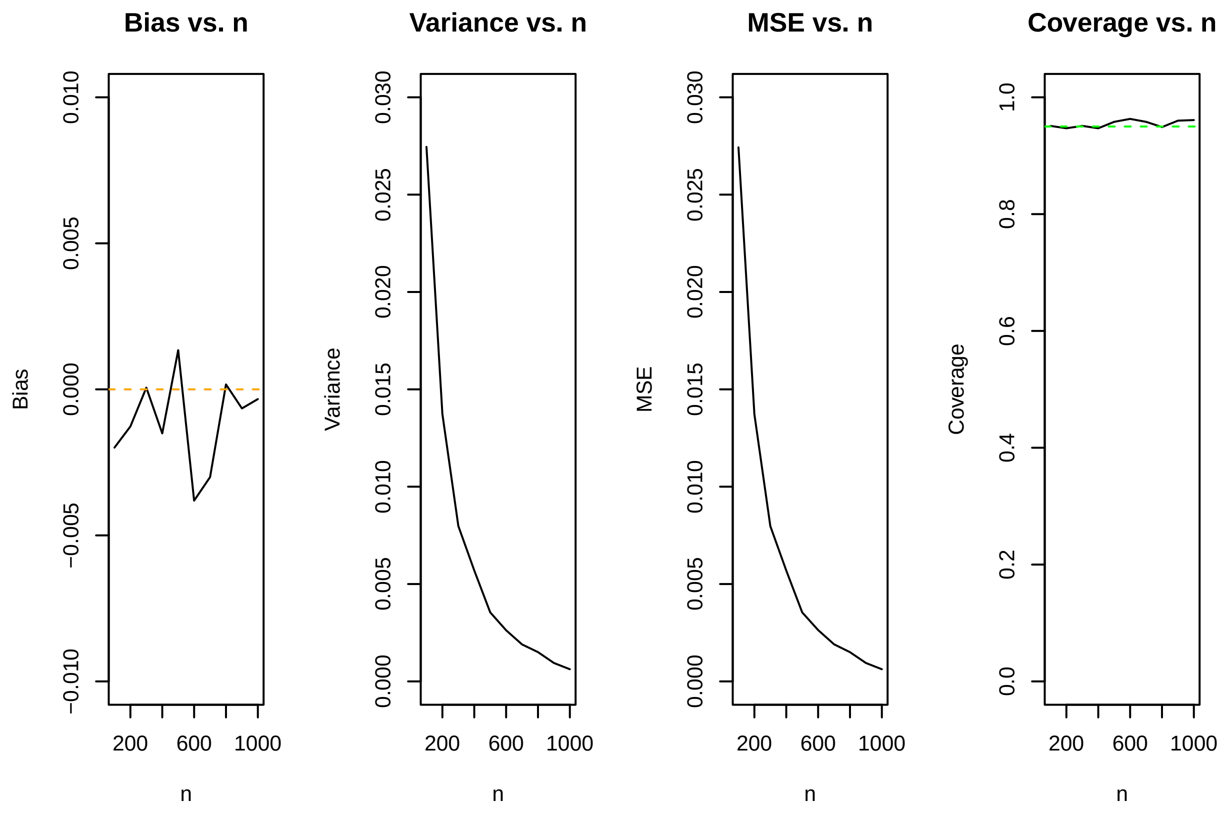
```
avepop <- mean(EDM.pop$goalsFor)
n = c(100,200,300,400,500,600,700,800,900,1000)
N <- nrow(EDM.pop)
sampbias <- rep(0,length(n))
sampvars <- rep(0,length(n))
sampmses <- rep(0,length(n))
sampcovs <- rep(0,length(n))

for(i in 1:length(n)){
  est <- rep(0, 1000)
  ci <- matrix(0, nrow = 1000, ncol = 2)
  pi_u <- rep(n[i]/N, n[i])
  pi_uv <- matrix((n[i]*(n[i]-1))/(N*(N-1)),nrow=n[i],ncol=n[i])
  diag(pi_uv) <- pi_u
  for(m in 1:1000){
    samp <- sample(EDM.pop$goalsFor, size = n[i], replace = FALSE)
    t_u <- samp/N
    est[m] <- sum(t_u/pi_u)
    se <- sqrt(estVarHT(t_u, pi_u, pi_uv))
    ci[m,] <- sum(t_u / pi_u) + 2*c(-1,1)*se
  }
  sampbias[i] <- mean(est-avepop)
  sampvars[i] <- var(est)
  sampmses[i] <- mean((est-avepop)^2)
  sampcovs[i] <- mean(apply(X = ci, MARGIN = 1, FUN = function(u){avepop >= u[1] & avepop <= u[2]}))
}
```

```
par(mfrow=c(1,4))
plot(n, sampbias, ylim = c(-0.01,0.01),type="l", main= "Bias vs. n", ylab = "Bias")
abline(h=0, col="orange", lty = 2)

plot(n, sampvars, ylim = c(0,0.03),type = "l", main= "Variance vs. n", ylab = "Variance")
plot(n,sampmses, ylim = c(0,0.03),type = "l",   main= "MSE vs. n", ylab = "MSE")

plot(n,sampcovs, ylim = c(0, 1),type = "l",main= "Coverage vs. n", ylab = "Coverage")
abline(h = 0.95, col="green", lty = 2)
```



(i)

(ii) For this Horvitz-Thompson Estimator, we find that bias is generally around 0. No matter what n, sample size, is, Bias is around 0. As n, sample size, increases, the variance decreases and the MSE decreases. Coverage is around and close to 0.95 regardelss of n as shown in the plot.