Assignment_3

Abbas Zal- 2072054

#adding library

```
library(ggplot2)
library(reshape2)
```

Exercise 1

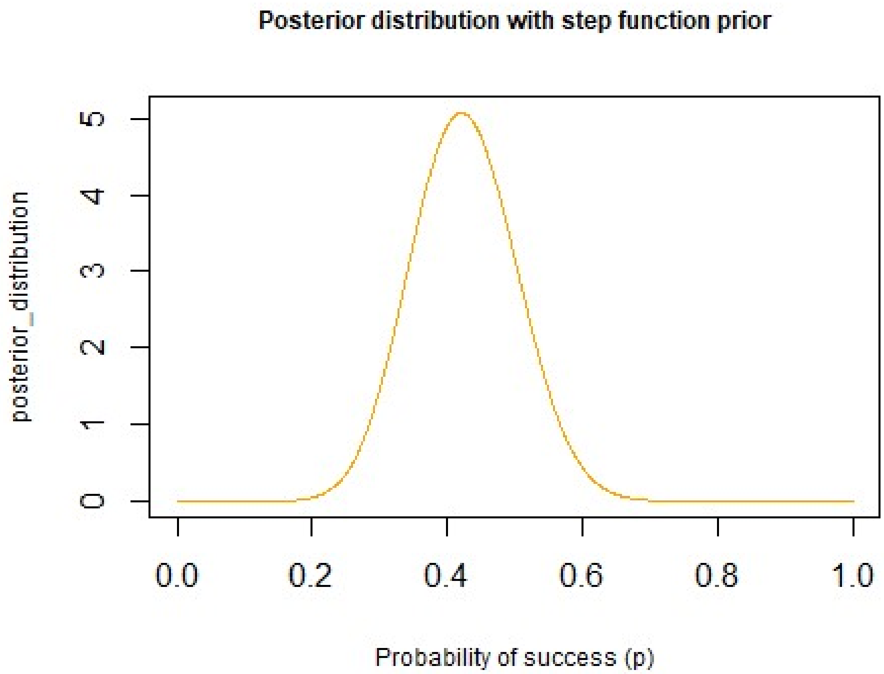
study the binomial inference for a study that reports y = 7 successes in n = 20 independent trial. Assume the following priors:

- a uniform distribution
- a Jeffrey's prior
- a step function:

```
g = function(x){
  ifelse(x <= 0.2 , x ,
         ifelse(x>0.2& x<=0.3 , 0.2 ,
                ifelse(x>0.3 & x<=0.5 , 0.5 - p , 0)))
  }
```

• plot the posterior distribution and summerize the results computing the first two moments

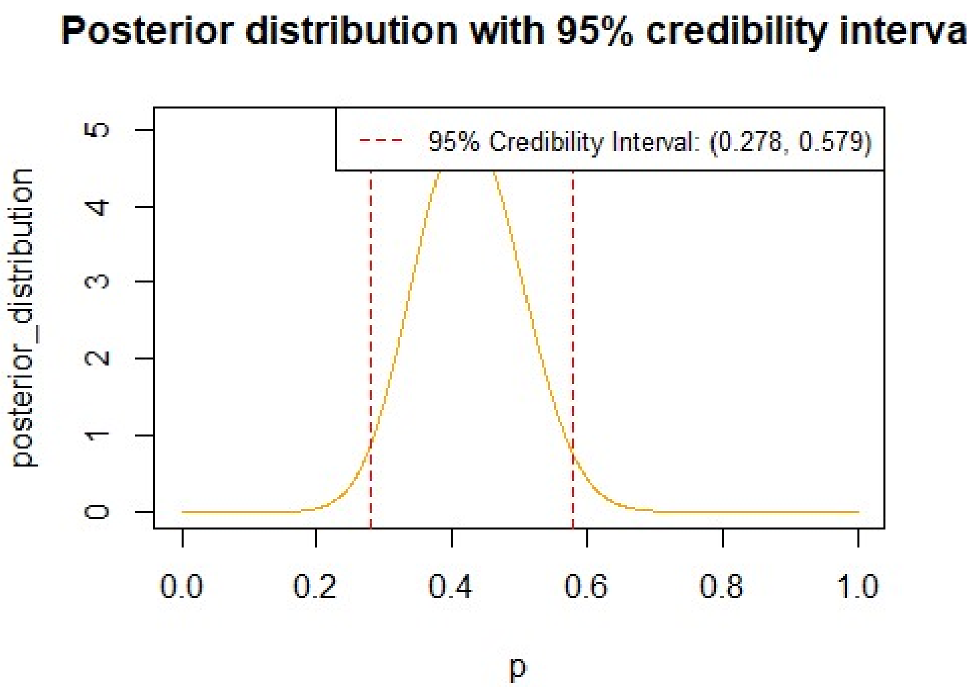
```
# Data
y <- 7
n <- 20
#g(0.5) == 0.5 so:
# Posterior distribution
posterior_alpha_step <- y + 0.5 * n
posterior_beta_step <- n - y + 0.5 * n

posterior_distribution_step <- dbeta(seq(0, 1, length.out = 1000),
posterior_alpha_step, posterior_beta_step)

# Plot posterior distribution
plot(seq(0, 1, length.out = 1000), posterior_distribution_step, type = "l",
     xlab = "Probability of success (p)", ylab = "posterior_distribution",
cex.lab = 0.75,
     main = "Posterior distribution with step function prior", cex.main =
0.75, col = "orange")
```



• compute a 95% credibility interval and give the results in a summary table

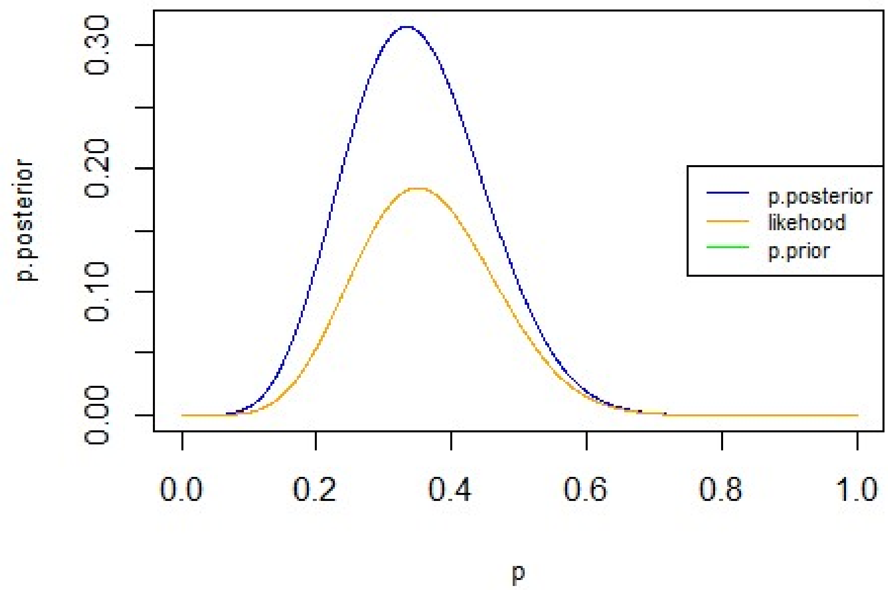
```
# Compute first two moments
mean_step <- posterior_alpha_step / (posterior_alpha_step +
posterior_beta_step)
variance_step <- posterior_alpha_step * posterior_beta_step /
  ((posterior_alpha_step + posterior_beta_step)^2 * (posterior_alpha_step +
posterior_beta_step + 1))


# Compute credibility interval
lower_ci_step <- qbeta(0.025, posterior_alpha_step, posterior_beta_step)
upper_ci_step <- qbeta(0.975, posterior_alpha_step, posterior_beta_step)



# Plot posterior distribution with credibility interval
plot(seq(0, 1, length.out = 1000), posterior_distribution_step, type = "l",
xlab = "p", ylab = "posterior_distribution",
     main = "Posterior distribution with 95% credibility interval", col =
"orange")

# Add vertical lines for credibility interval
abline(v = lower_ci_step, col = "red", lty = 2)
abline(v = upper_ci_step, col = "red", lty = 2)
```

```r
# Add legend for credibility interval
legend("topright", legend = paste0("95% Credibility Interval: (",
round(lower_ci_step, 3), ", ", round(upper_ci_step, 3), ")"), col = "red",
lty = 2, cex = 0.8)
```

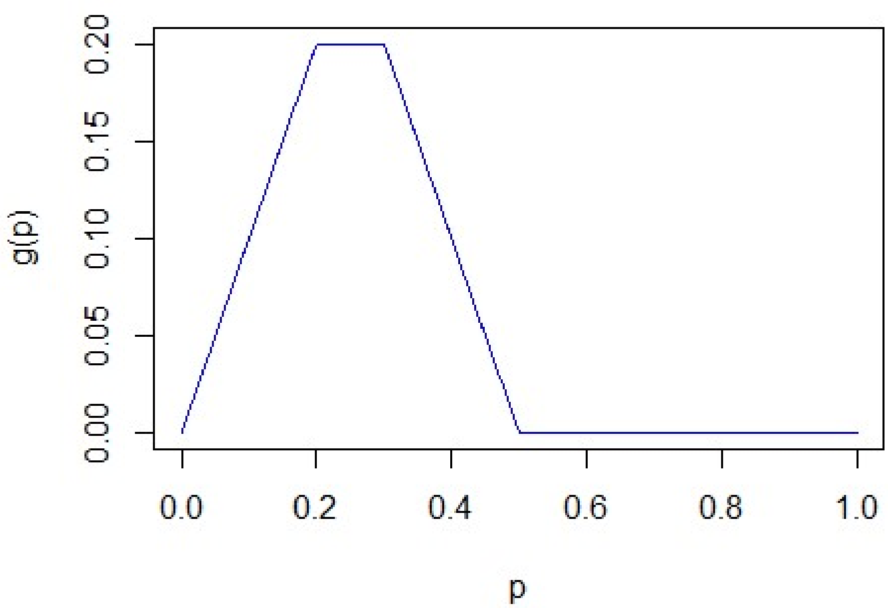


- draw the limits on the plot of the posterior distribution

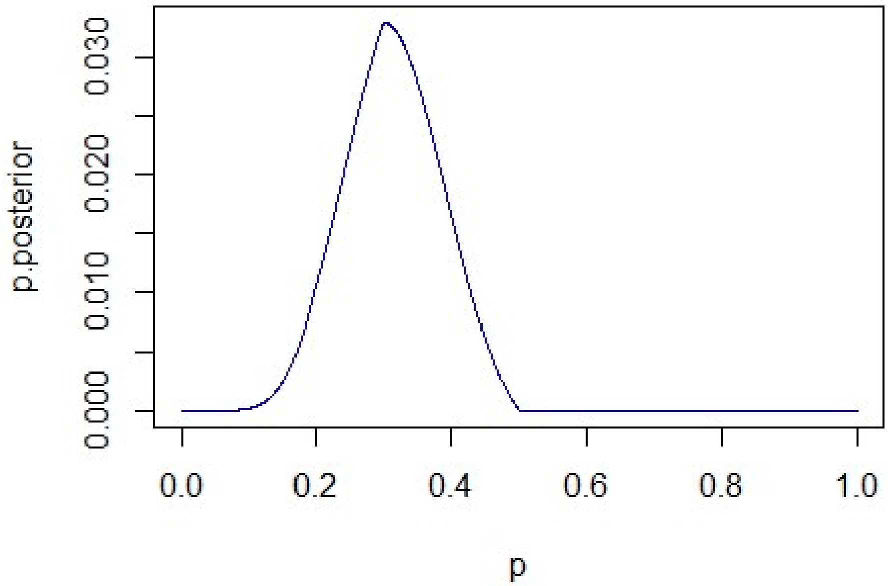
```r
#for Jeffery's prior

p <- seq(0, 1, length.out = 1000)

p.prior <- 1/sqrt(p)
likehood <-  dbinom(x = y , size = n , prob = p)
p.posterior <- p.prior * likehood

plot(p, p.posterior , type = "l", col = "blue",lwd=0.5, xlab ="p", ylab =
"p.posterior", cex.main=0.8 , cex.lab=0.8)
options(scipen = 10)
# add  line to the plot
lines(p, likehood, col = "orange",lwd=0.5)
lines(p, p.prior, col = "green",lwd=0.5)

legend("right", legend=c("p.posterior", "likehood", "p.prior"),
       col= c("blue", "orange", "green"), lty=c(1,1),cex = 0.7)
```



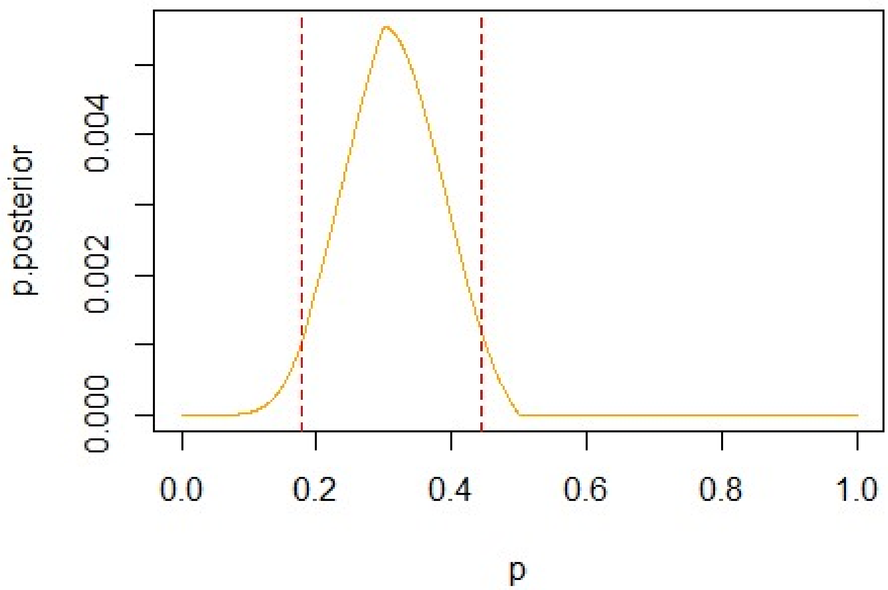

```
#For prior

plot(p , g(p) , type = 'l' , lwd = 1.5 , col = 'blue')
```



```r
p.prior = g(p)
likehood =  dbinom(x = y , size = n , prob = p)
p.posterior = p.prior * likehood
plot(p , p.posterior , type = 'l' , lwd = 1.5 , col = 'blue')
```

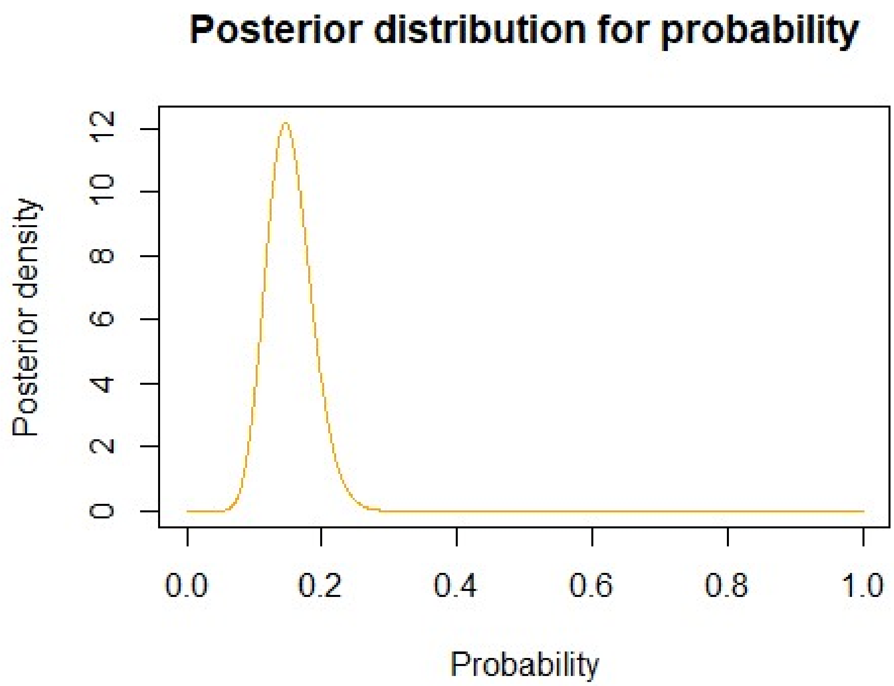



```r
p.posterior <- p.posterior/sum(p.posterior)

p1 <- sum(p * p.posterior)
p2 <- sum(p^2 * p.posterior)
#rand number
samples = sample(p , size = 1000 , replace = TRUE , prob = p.posterior)
ci_step = quantile(samples , c(0.025 , 0.975))

plot(p , p.posterior , type = 'l' , lwd = 1.5 , col = 'orange')

abline(v = ci_step[1], col = "red", lty = "dashed")
abline(v = ci_step[2], col = "red", lty = "dashed")
```

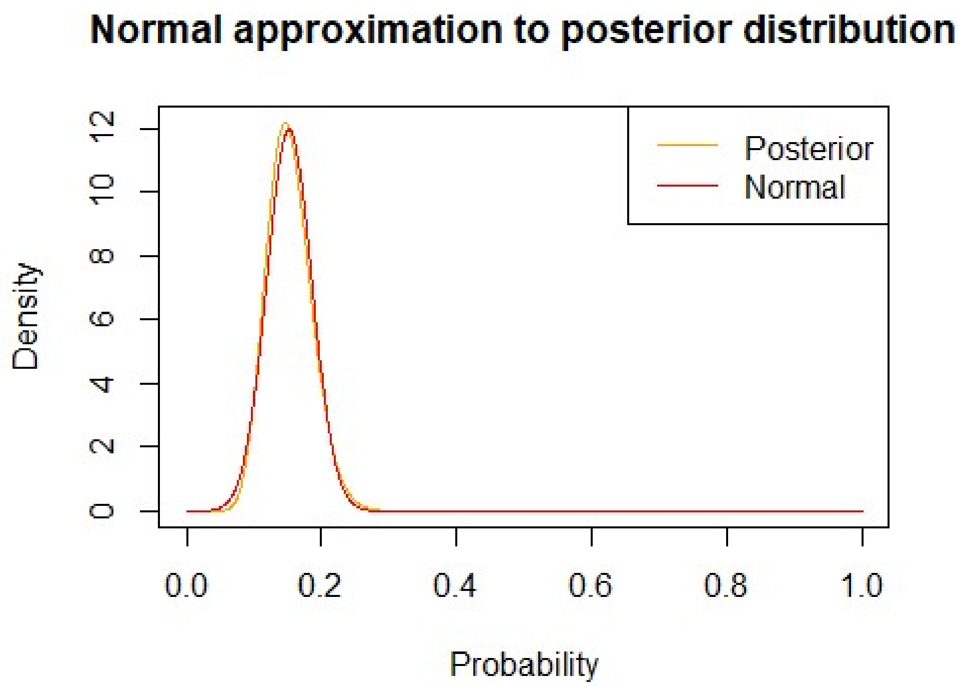


Exercise 2

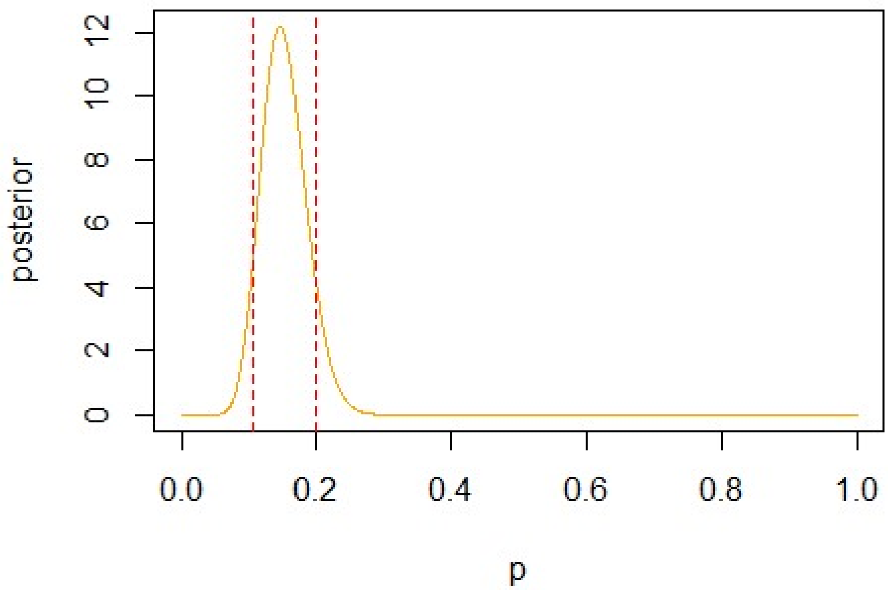
```
# Set values for n and y
n <- 116
y <- 17

# Compute posterior parameters
alpha <- y + 1
beta <- n - y + 1


p <- seq(0, 1, length.out = 1000)

# Compute posterior distribution
posterior <- dbeta(p, alpha, beta)

# Plot posterior distribution
plot(p, posterior, type = "l", xlab = "Probability", ylab = "Posterior 
density",
     main = "Posterior distribution for probability",col="orange")
```




```r
p1 = 1/1000 * sum(p * posterior)
p2 = 1/1000 * sum(p^2 * posterior)

m = p1
s2 = p2 - p1 ** 2

p.normal = dnorm(p , m , sqrt(s2))

# Plot posterior and normal densities
plot(p, posterior, type = "l", xlab = "Probability", ylab = "Density",
     main = "Normal approximation to posterior distribution",col="orange")
lines(p, p.normal, col = "red")
legend("topright", legend = c("Posterior", "Normal"),
       col = c("orange", "red"), lty = 1)
```




```
ci = qbeta(c(0.025 , 0.975), alpha + y , beta + n - y)

plot(p , posterior, type = 'l' , lwd = 1.5 , col = 'orange')

abline(v = ci[1], lty = "dashed", col = "red")
abline(v = ci[2], lty = "dashed", col = "red")
```



Exercise 3

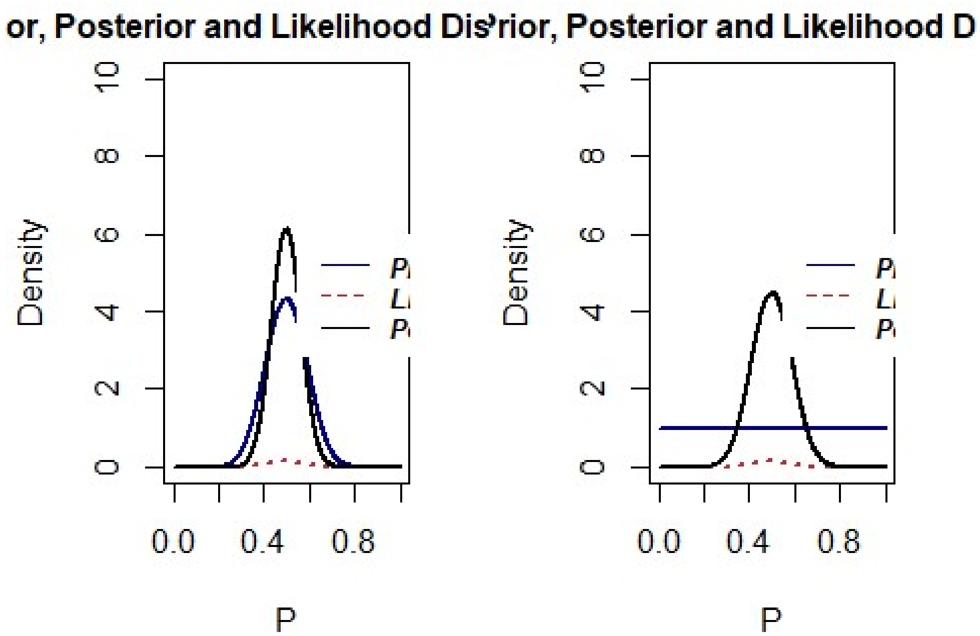
A coin is flipped n = 30 times with the following outcomes:

T, T, T, T, T, H, T, T, H, H, T, T, H, H, H, T, H, T, H, T, H, H, T, H, T, H, T, H, H, H

Assuming a flat prior, and a beta prior, plot the likelihood, prior and posterior distributions for the data set. Evaluate the most probable value for the coin probability p and, integrating the posterior probability distribution, give an estimate for a 95% credibility interval

```
#two functions that calculate alpha and beta given E(x) and STD(x)
beta.func  <- function(exp.x, std.x){return((((1-exp.x)*exp.x)/((std.x)**2) -
1) / (1+ exp.x/(1-exp.x)))}
alpha.func <- function(exp.x, std.x){return((exp.x/(1-exp.x))*(((1-
exp.x)*exp.x)/((std.x)**2) - 1) / (1+ exp.x/(1-exp.x)))}

# a function that plots prior, posterior, likelihood at the same time
all.plotter <- function(n,r,prior, exp.x, std.x, ylim ){
    if (prior == 'Beta'){
        alpha.prior <- alpha.func(exp.x, std.x ); beta.prior <- beta.func
(exp.x, std.x )}
    else if(prior == 'Uniform'){alpha.prior<- 1; beta.prior <-1}
    else {cat("unrecognized prior function!")}

    p.prior <- dbeta(x=p,alpha.prior ,beta.prior) #can be eigther flat or
beta
    p.like  <- dbinom(x=r, size=n, prob=p)
```

```r
    p.like  <- p.like /( delta.p*sum(p.like ))
    p.post  <- dbeta(x=p, shape1=alpha.prior+r, shape2=beta.prior+n-r)
    plot(p, p.prior , col="navyblue",lwd = 2.6,type="l", xlim=c(0,1),
         ylim = ylim,xlab="P", ylab="Density")
    lines(p, p.like , col="firebrick3",lwd=2.6, lty=3,xlab="P",
ylab="Density")
    lines(p, p.post , col="black",lwd=2.6,xlab="P", ylab="Density")
    legend(0.55,6,legend=c("Prior", "Likelihood", "Posterior"), lty=1:2,
cex=0.8,
           col=c("navyblue", "firebrick3", "black"), box.lty=0, text.font=4)
    title(main=paste(prior, "Prior, Posterior and Likelihood Dist.","r/n
=",r,"/",n), line=0.7, cex.main=1)}


#posterior function, for integrating purpose
post.func <- function(x){
    if (prior == 'Beta'){alpha.prior <- alpha.func(exp.x, std.x); beta.prior
<- beta.func (exp.x, std.x )}
    else if(prior == 'Uniform'){alpha.prior<- 1; beta.prior <-1}
    else {cat("unrecognized prior function!")}
    return(dbeta(x, shape1=alpha.prior+r, shape2=beta.prior+n-r))}

#a function that finds 95 credibility interval
cred.finder <- function(post.function){
    k <-0; integral <-1; interval <- 0.95
    while (integral > 0.95){k <- k + 0.01; integral <-
integrate(post.function, k,1-k)$value}
    return (c(k, 1-k))}

#a function which find the most probable value
most.prob <- function(n,r, prior, exp.x,std.x){
    a  <- alpha.func(exp.x, std.x)
    b  <- beta.func (exp.x, std.x)
    if (prior=='Beta'){return((r+a-1)/(n+a+b-2))}
    else if (prior =='Uniform'){return(r/n)}}

n <- 30
r <- 15
exp.x <- 0.5
std.x <- 0.09
delta.p<- 1/30
par(mfrow=c(1,2))
ylim <- c(0,10)  #setting the y axis range
all.plotter(n, r,'Beta', exp.x,    std.x, ylim)
all.plotter(n, r,'Uniform', exp.x, std.x, ylim)
```

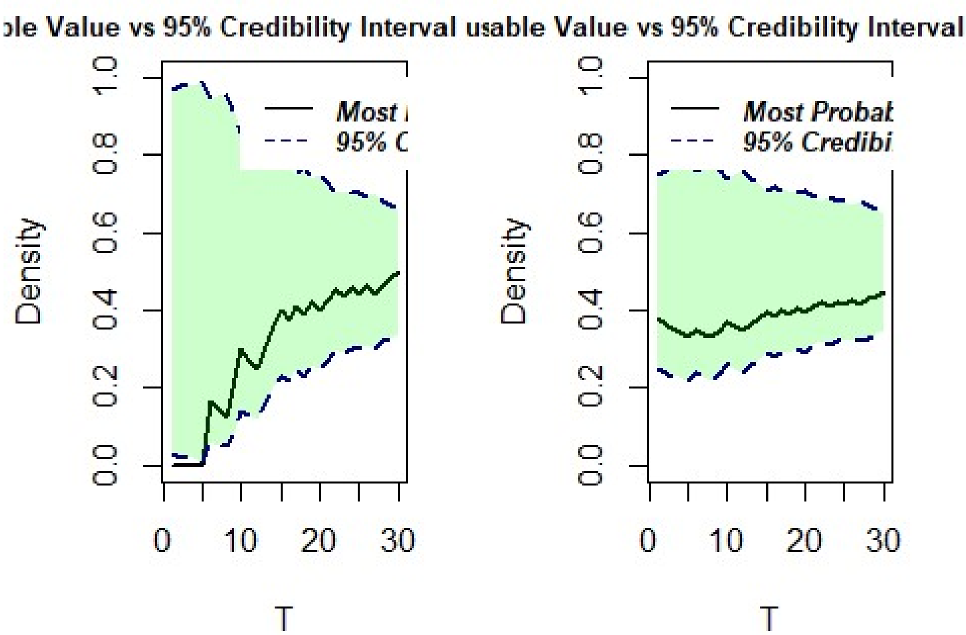



```
cat("The Most Probable Value for the Uniform dist. is",   most.prob(n,r,
"Uniform", exp.x, std.x))

## The Most Probable Value for the Uniform dist. is 0.5

cat("\nThe Most Probable Value for the Beta dist. is"   , most.prob(n,r,
"Beta", exp.x,std.x))

##
## The Most Probable Value for the Beta dist. is 0.5

prior="Beta"
cat("The 0.95 Credibility Interval for Beta Prior is",
cred.finder(post.func))

## The 0.95 Credibility Interval for Beta Prior is 0.38 0.62

prior="Uniform"
cat("\nThe 0.95 Credibility Interval for Uniform Prior is",
cred.finder(post.func))

##
## The 0.95 Credibility Interval for Uniform Prior is 0.34 0.66
```

Repeat the same analysis assuming a sequential analysis of the data.Show how the most probable value and the credibility interval change as a function of the number of coin tosses (i.e. from 1 to 30).

```
data.seq <-c('T', 'T', 'T', 'T', 'T', 'H', 'T', 'T', 'H', 'H', 'T', 'T', 'H',
'H','H',
             'T', 'H', 'T', 'H', 'T', 'H', 'H', 'T', 'H', 'T', 'H', 'T', 'H',
'H','H')
uni.most.prob      <- vector(); beta.most.prob      <- vector()
uni.cred.interval1 <- vector(); beta.cred.interval1 <- vector();
number.coin.toss<- vector()
uni.cred.interval2 <- vector(); beta.cred.interval2 <- vector()

r <- 0; n <- 0
exp.x<- 0.4; std.x<- 0.09; interval <- 0.95 #these values remain constatns
for (t in seq(1,length(data.seq))){
    n <- n+1
    number.coin.toss  <- append(number.coin.toss, t)

    if (data.seq[t] =='H'){r <- r+1}

    uni.most.prob     <- append(uni.most.prob     , most.prob(n, r,
'Uniform', 0.4, 0.09))
    beta.most.prob    <- append(beta.most.prob    , most.prob(n, r, 'Beta',
0.4, 0.09))

    prior="Uniform"
    uni.cred.interval1 <- append(uni.cred.interval1 ,
cred.finder(post.func)[1])
    uni.cred.interval2 <- append(uni.cred.interval2 ,
cred.finder(post.func)[2])

    prior="Beta"
    beta.cred.interval1<- append(beta.cred.interval1,
cred.finder(post.func)[1])
    beta.cred.interval2<- append(beta.cred.interval2,
cred.finder(post.func)[2])}


options(repr.plot.width=11, repr.plot.height =6)  #changing size of plots
par(mfrow=c(1,2))
plot(number.coin.toss ,col="gray0",  uni.most.prob ,ylim = c(0,1)  ,lwd = 2
,type="l" ,xlab="T", ylab="Density"    )
polygon(c(1:30, 30:1), c(uni.cred.interval1, rev(uni.cred.interval2)),col =
rgb(0, 1, 0, alpha=0.2), border = NA)
lines(number.coin.toss,  uni.cred.interval1   ,lwd = 2, lty='dashed', col
='navyblue' )
lines(number.coin.toss,  uni.cred.interval2  ,lwd = 2,  lty='dashed', col =
'navyblue')
title(main=paste("Most Probable Value vs 95% Credibility Interval using
Uniform Prior"), line=0.7, cex.main=0.8)
legend(10,1,legend=c("Most Probable Value", "95% Credibility Interval"),
lty=1:2, cex=0.8,
```

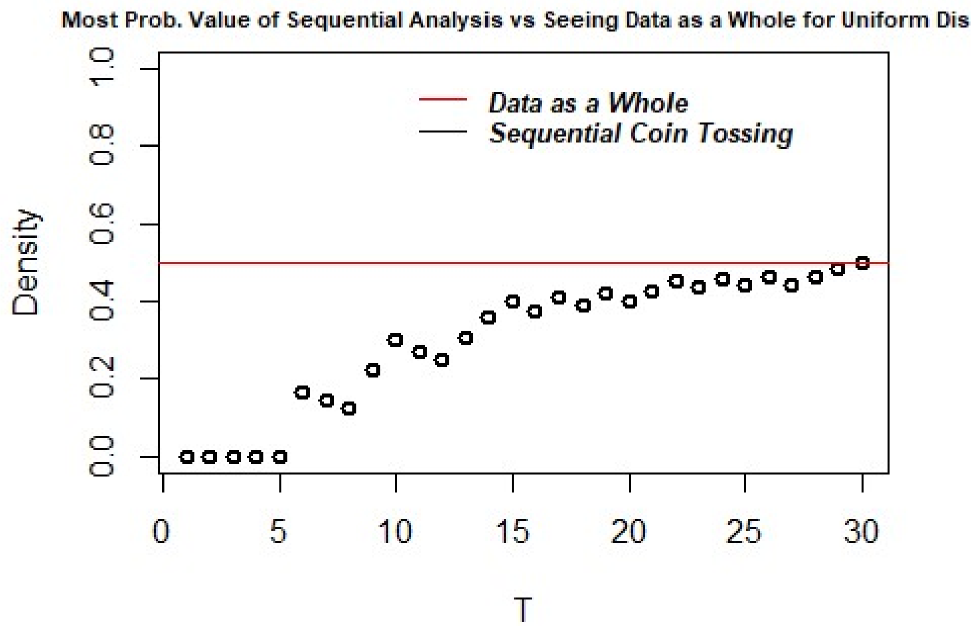
```
             col=c("gray0", "navyblue"), box.lty=0, text.font=4)

plot(number.coin.toss  ,col="gray0", beta.most.prob, ylim = c(0,1),
type="l",lwd = 2,xlab="T", ylab="Density")
polygon(c(1:30, 30:1), c(beta.cred.interval1, rev(beta.cred.interval2)),col =
rgb(0, 1, 0, alpha=0.2), border = NA)
lines(number.coin.toss , beta.cred.interval1,lwd = 2,  lty='dashed' ,
col='navyblue')
lines(number.coin.toss , beta.cred.interval2,lwd = 2,  lty='dashed' ,col
='navyblue')
title(main=paste("Most Probable Value vs 95% Credibility Interval using Beta
Prior"), line=0.7, cex.main=0.8)
legend(0,1,legend=c("Most Probable Value", "95% Credibility Interval"),
lty=1:2, cex=0.8,
             col=c("gray0", "navyblue"), box.lty=0, text.font=4)
```



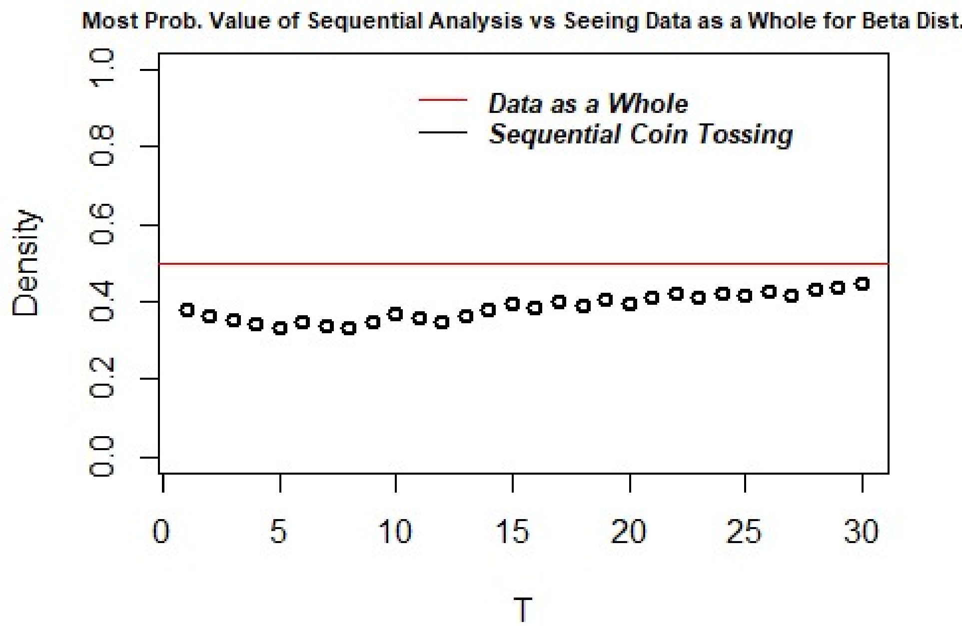
Do you get a different result, by analyzing the data sequentially with respect to a one-step analysis (i.e. considering all the data as a whole) ?

```
#ar(mfrow=c(1,2))
plot(number.coin.toss ,col="gray0",  uni.most.prob ,ylim = c(0,1)  ,lwd = 2
,xlab="T", ylab="Density"    )
abline(h=0.5, col="red")
title(main=paste("Most Prob. Value of Sequential Analysis vs Seeing Data as a
Whole for Uniform Dist."), line=0.7, cex.main=0.7)
legend(10,1,legend=c("Data as a Whole", "Sequential Coin Tossing"), lty=1:1,
```

```
cex=0.8,
           col=c("red", "gray0"), box.lty=0, text.font=4)
```




```
plot(number.coin.toss  ,col="gray0", beta.most.prob, ylim = c(0,1),lwd =
2,xlab="T", ylab="Density")
abline(h=0.5, col="red")
title(main=paste("Most Prob. Value of Sequential Analysis vs Seeing Data as a
Whole for Beta Dist."), line=0.7, cex.main=0.7)
legend(10,1,legend=c("Data as a Whole", "Sequential Coin Tossing"), lty=1:1,
cex=0.8,
           col=c("red", "gray0"), box.lty=0, text.font=4)
```



As we can see, in the case of sequential coin tossing, the most probable value converges to the one obtained from seeing the data as a whole.

Exercise 4 - Six Boxes Toy Model : inference

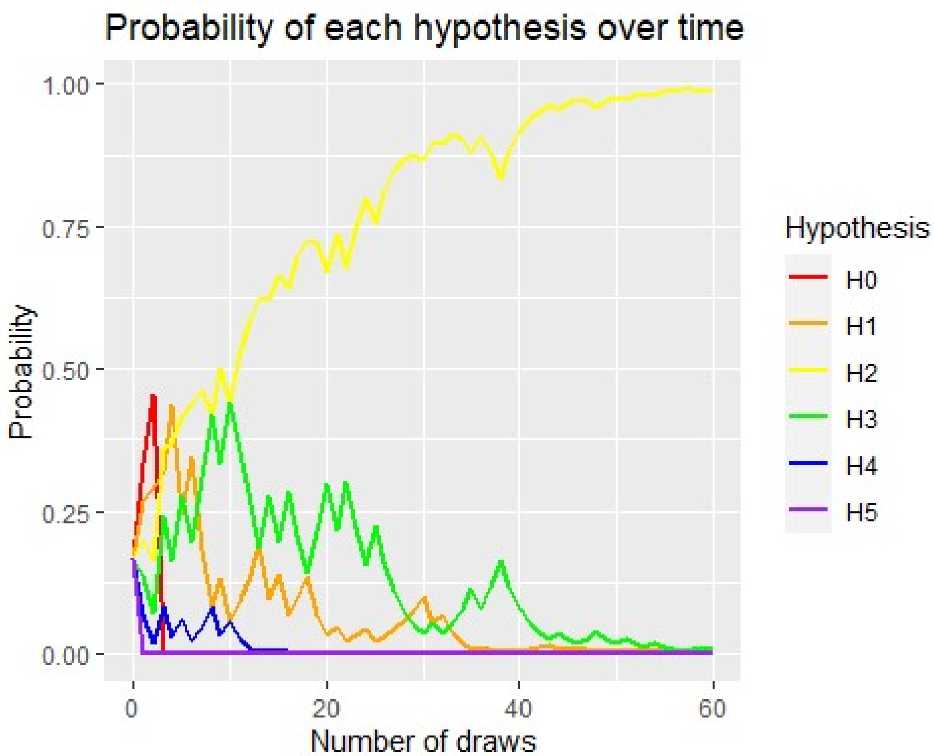
```
# Set the initial values
N <- 0
prob <- rep(1/6, 6)
results <- data.frame(N = 0, H0 = 1/6, H1 = 1/6, H2 = 1/6, H3 = 1/6, H4 =
1/6, H5 = 1/6)

# Create a function to update the probabilities
update_prob <- function(color, j, prob){
  if (color == 'white') {
    prob <- (j/5 * prob) / sum(j/5 * prob)
  } else {
    prob <- ((5-j)/5 * prob) / sum((5-j)/5 * prob)
  }
  prob
}

# Simulate the process 20 times
for (i in 1:60){

  # Draw a ball randomly from the box
  box <- c(rep('white', sample(6, 1)), rep('black', 5))
  color <- sample(box, 1)
```

```
  # Update the probabilities
  j <- 0:5
  prob <- update_prob(color, j, prob)

  # Increment N and save the results
  N <- N + 1
  results[nrow(results) + 1,] <- c(N, prob)
}

# Print the final results
head(results)
```

```
##   N        H0        H1        H2         H3         H4        H5
## 1 0 0.1666667 0.1666667 0.1666667 0.16666667 0.16666667 0.1666667
## 2 1 0.3333333 0.2666667 0.2000000 0.13333333 0.06666667 0.0000000
## 3 2 0.4545455 0.2909091 0.1636364 0.07272727 0.01818182 0.0000000
## 4 3 0.0000000 0.3200000 0.3600000 0.24000000 0.08000000 0.0000000
## 5 4 0.0000000 0.4383562 0.3698630 0.16438356 0.02739726 0.0000000
## 6 5 0.0000000 0.2461538 0.4153846 0.27692308 0.06153846 0.0000000
```

```
# Plot the results
ggplot(data = results, aes(x = N)) +
  geom_line(aes(y = H0, color = "H0"), size = 1, show.legend = TRUE) +
  geom_line(aes(y = H1, color = "H1"), size = 1, show.legend = TRUE) +
  geom_line(aes(y = H2, color = "H2"), size = 1, show.legend = TRUE) +
  geom_line(aes(y = H3, color = "H3"), size = 1, show.legend = TRUE) +
  geom_line(aes(y = H4, color = "H4"), size = 1, show.legend = TRUE) +
  geom_line(aes(y = H5, color = "H5"), size = 1, show.legend = TRUE) +
  labs(title = "Probability of each hypothesis over time",
       x = "Number of draws",
       y = "Probability") +
  scale_color_manual(values = c("H0" = "red", "H1" = "orange", "H2" =
"yellow",
                                "H3" = "green", "H4" = "blue", "H5" =
"purple"),
                     name = "Hypothesis",
                     labels = c("H0", "H1", "H2", "H3", "H4", "H5"))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```


Probability of each hypothesis over time
Probability
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Number of draws
Hypothesis
H0
H1
H2
H3
H4
H5