Nicolai_ex5

Andrea Nicolai

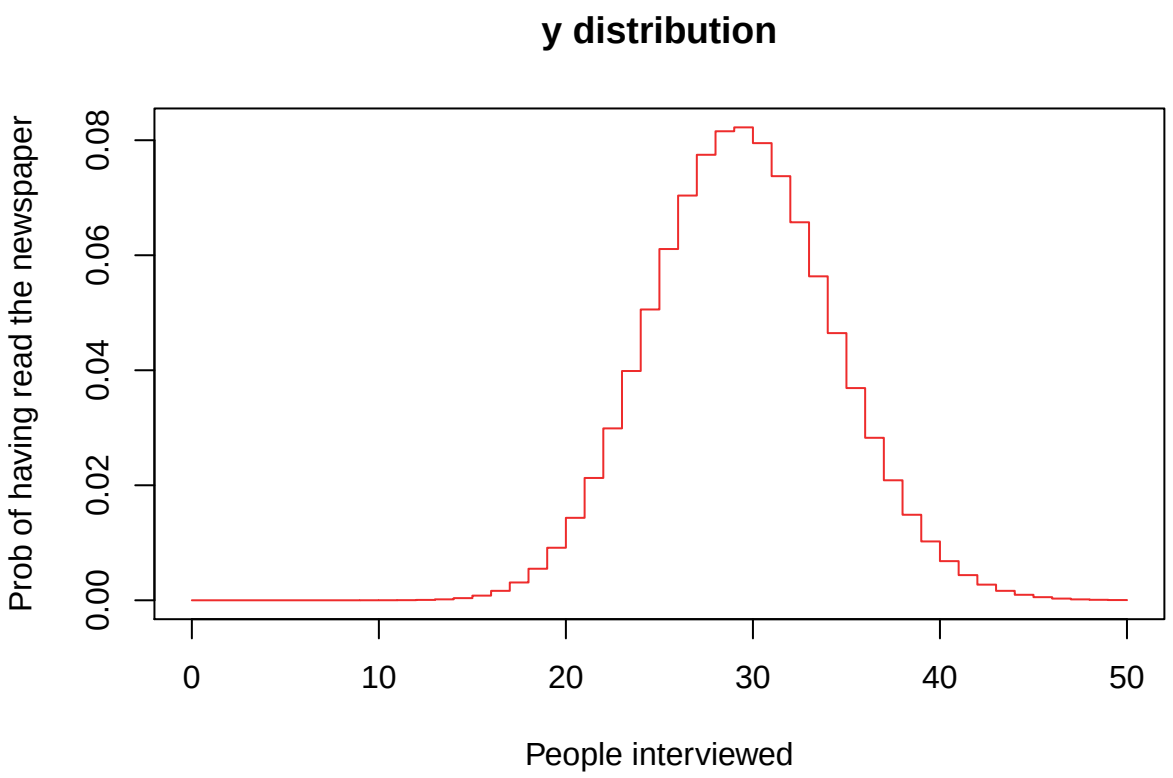
5/5/2020

Exercise 1

A publishing company has recently launched a new journal. In order to determine how effective it is in reaching its possible audience, a market survey company selects a random sample of people from a possible target audience and interviews them. Out of 150 interviewed people, 29 have read the last issue of the journal.

What kind of distribution would you assume for y, the number of people that have seen the last issue of the journal? A Binomial distribution with parameters ($k = 29,\ n = 150,\ p = 29/150$): a person might have read the last issue of the journal as well as he might not have, so each interview is indeed a Bernoulli trial.

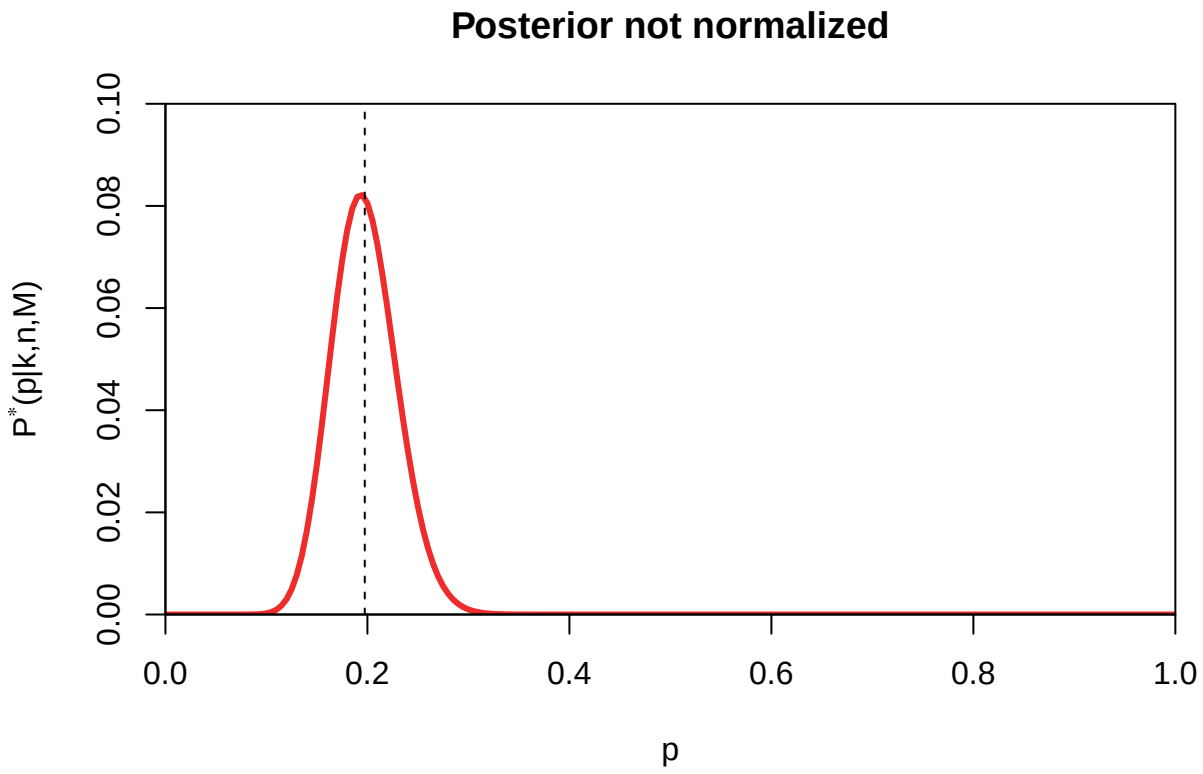
```
plot(0:50, dbinom(0:50, size = 150, prob = 29/150), main = "y distribution", xlab = "People interviewed
```



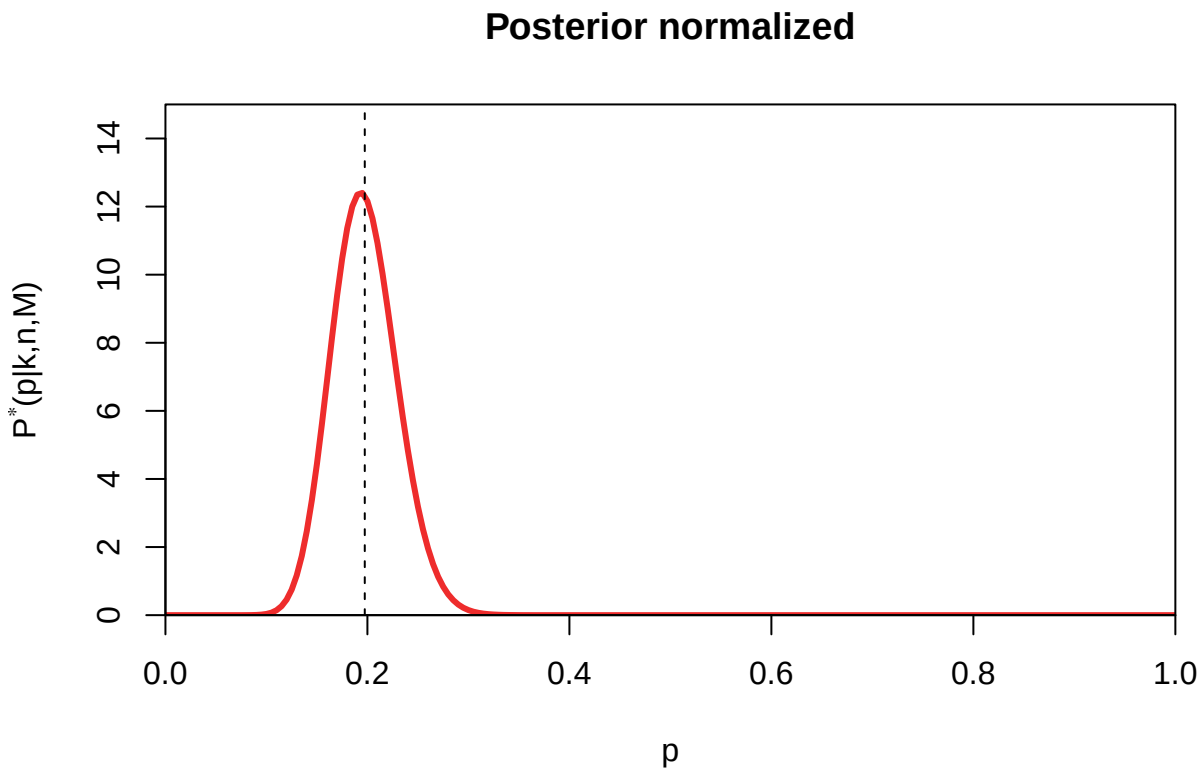
Assuming a uniform prior, what is the posterior distribution for y ? If we adopt a uniform prior then the Posterior pdf is simply proportional to the Likelihood, which is itself binomial.

```
n <- 150
k <- 29
delta.p <- 0.005
p <- seq(0, 1, length.out = 1/delta.p + 1)
p.post <- dbinom (x=k, size=n, prob=p)
```

```
p.norm <- p.post/( delta.p*sum(p.post ))
p.mean <- delta.p*sum(p*p.norm)


plot(p, p.post , main = "Posterior not normalized" , xaxs='i', yaxs='i', col='firebrick2', type='l', lty
abline (v=p.mean , lty=2)
```

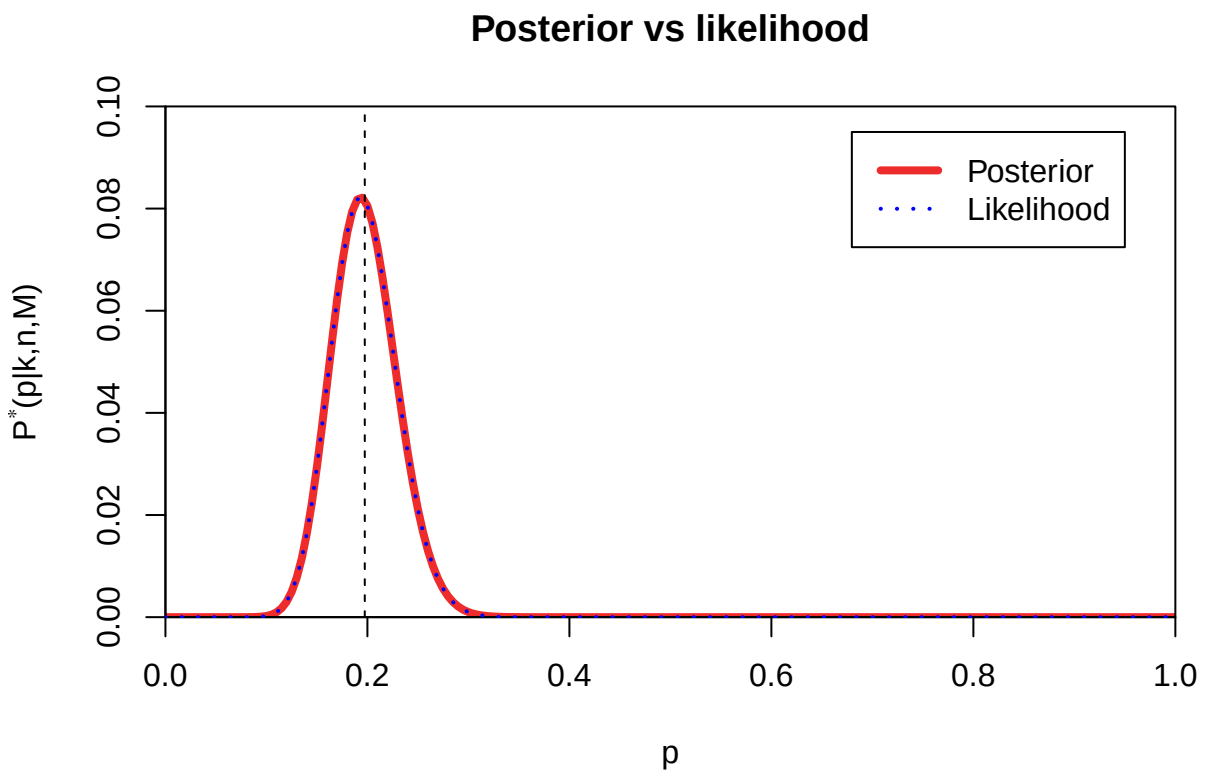



```
plot(p, p.norm , main = "Posterior normalized" , xaxs='i', yaxs='i', col='firebrick2', type='l', lty=1,
abline (v=p.mean , lty=2)
```



Plot both posterior and likelihood ditributions functions. They are indeed the same, being the uniform prior 1 in the whole interval [0,1].

```
n <- 150
k <- 29

delta.p <- 0.005
p <- seq(0, 1, length.out = 1/delta.p + 1)

likelihood <- dbinom(k , n , prob = p)
p.post <- dbinom (x = k, size=n, prob=p)
p.mean <- delta.p*sum(p*p.norm)

plot(p, p.post , main = "Posterior vs likelihood" , xaxs='i', yaxs='i', col='firebrick2', type='l', lty=
lines(p, dbinom(k , n , prob = p) , lwd = 2, col = 'blue', lty = 3)
legend("topright", inset = 0.05 ,c("Posterior", "Likelihood"), col = c('firebrick2','blue'), lty = c(1,3
abline (v=p.mean , lty=2)
```



Exercise 2

Three students want to construct their prior probability about the proportion of residents that support the building of a new concert hall in their small town. + Anna thinks that her prior is a beta distribution with mean 0.2 and a standard deviation of 0.08. + Benny moved only recently to this new town and therefore he does non have the slightest idea about it. Therefore he decides to use a uniform prior. + Chris believes that his prior should have a trapezoidal shape

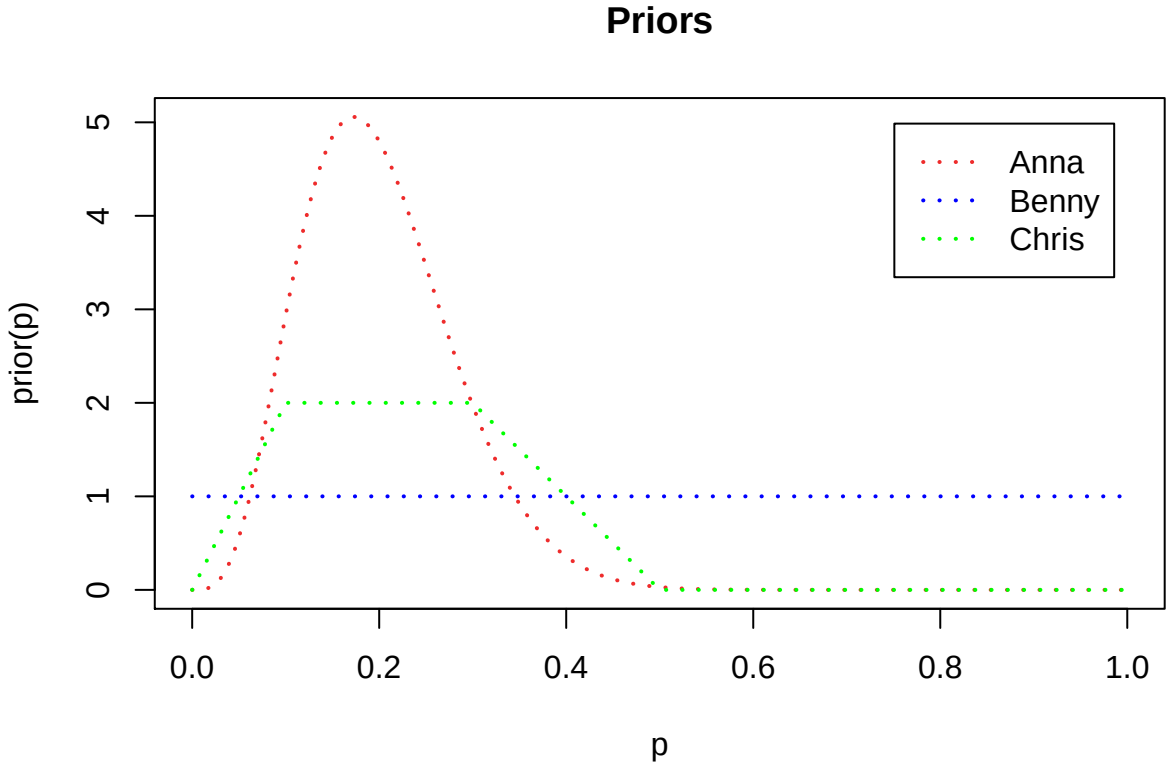
$$f(X) = \begin{cases} 20x & 0 \leqslant x < 0.1 \\ 2 & 0.1 \leqslant x < 0.3 \\ 5 - 10x & 0.3 \leqslant x < 0.5 \\ 0 & x \geqslant 0.5 \end{cases}$$

Draw and compare the three prior distributions.

```
delta.p <- 0.005
p <- seq(0, 1, length.out = 1/delta.p + 1)

prior.Anna <- function(x, mu = 0.2, std = 0.08 ) {
  var <- std**2
  alpha <- ((1 - mu)/var - 1/mu)*mu^2
  beta <- alpha*(1/mu - 1)
  return (dbeta(x, shape1 = alpha, shape2 = beta))
}

prior.Benny <- function (x, b = 1, a = 0){
  return (dunif(x,a,b))
}

prior.Chris <- function (y) {
  prob <- c(NA)
```

```
  for (x in y) {
    if (  0 <= x & x <  0.1)   {prob <- c(prob, 20*x)}
    else if (0.1 <= x & x <  0.3)   {prob <- c(prob, 2)}
    else if (0.3 <= x & x <  0.5)   {prob <- c(prob, 5-10*x)}
    else if (  0 >  x | x >= 0.5)   {prob <- c(prob, 0*x)}
  }
  return (prob[!is.na(prob)])
}
```

```
x <- seq(0, 1, length.out = 1/delta.p + 1)

plot(x, prior.Anna(x) , main = "Priors" , col='firebrick2', type='l', lty=3, lwd = 2, xlab="p", ylab= "
lines(x, prior.Benny(x) , lwd = 2, col = 'blue', lty = 3)
lines(x, prior.Chris(x) , lwd = 2, col = 'green', lty = 3)
legend("topright", inset = 0.05 ,c("Anna", "Benny", "Chris"), col = c('firebrick2','blue', "green"), lty
```

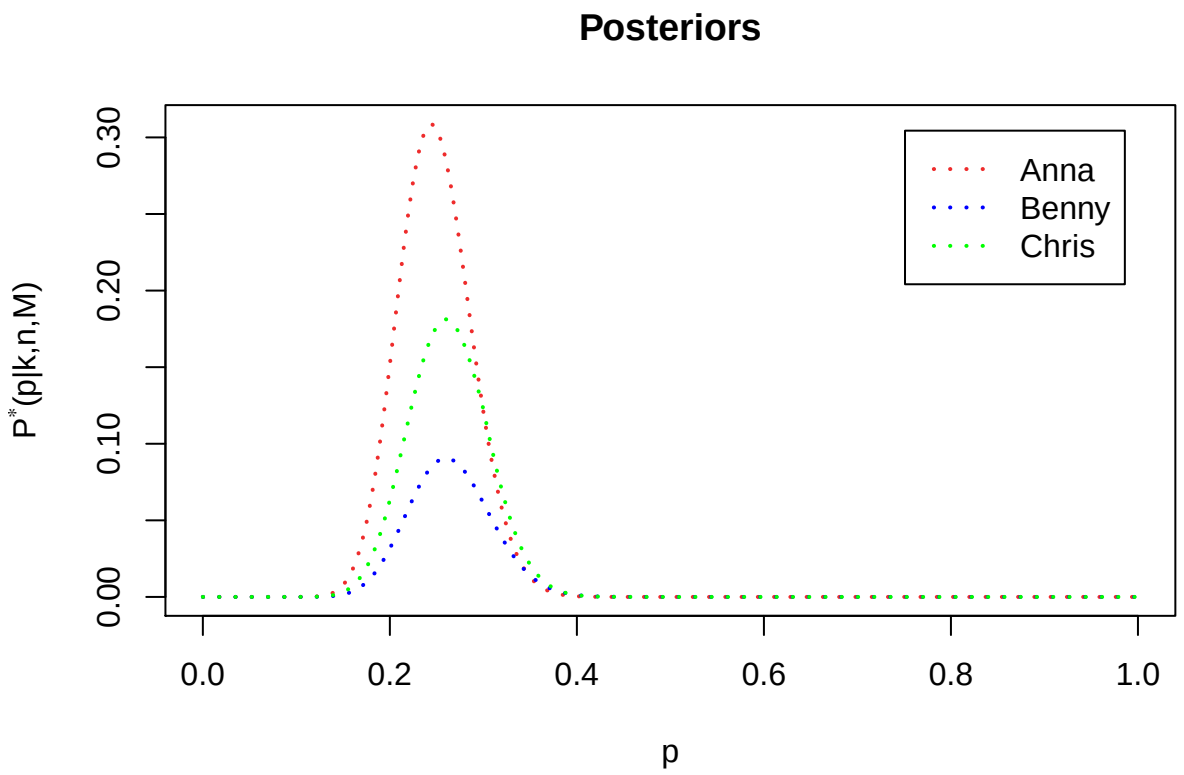


The next day the three students decide to interview a sample of 100 citizens of the small town, asking for their opinion. Out of the interviewed sample, 26 support the building of the new concert hall.

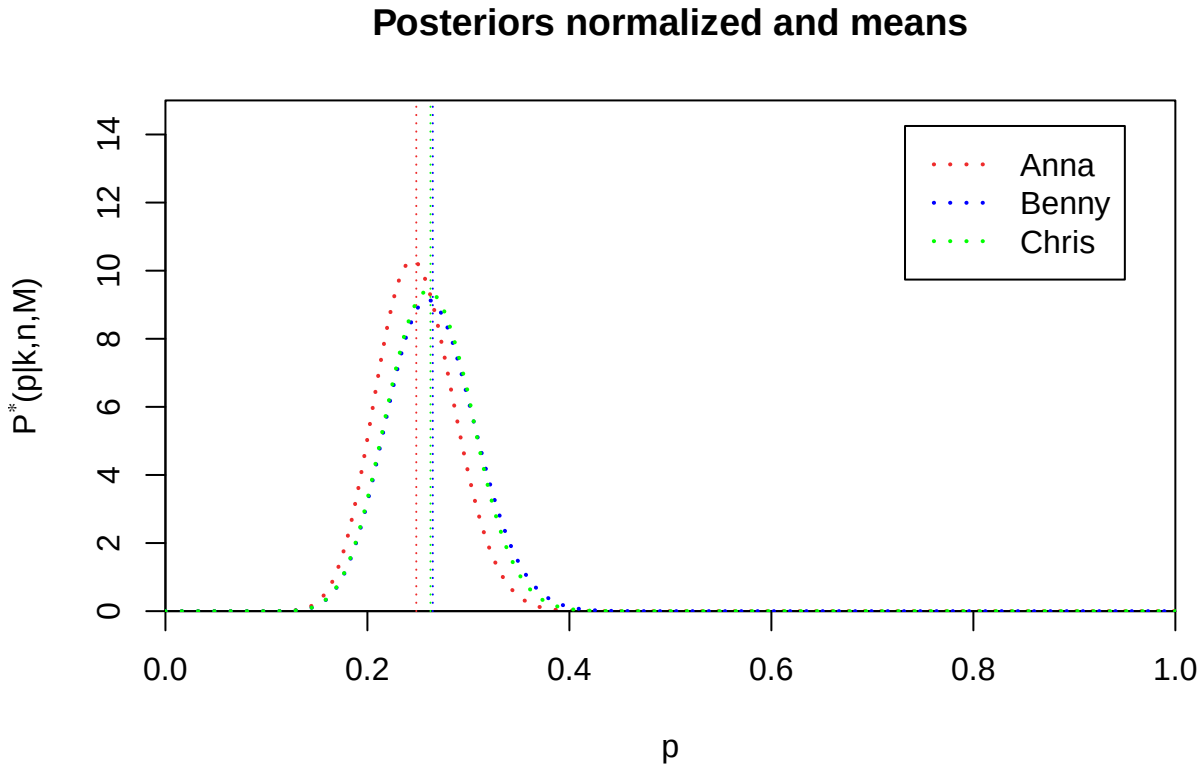
Evaluate and draw the three prior distributions. The likelihood distribution we use is the binomial with parameter k = 26, n = 100. We expect to see that Benny's posterior will be proportional to the likelihood distribution because her prior is uniform, whereas for Anna's posterior we expect to see an other Beta distribution.

```
k <- 26
n <- 100

p <- seq(0, 1, length.out = 1/delta.p + 1)
likelihood <- dbinom (x=k, size=n, prob=p)
```

```
p.post.Anna  <- likelihood*prior.Anna(p)
p.post.Benny <- likelihood*prior.Benny(p)
p.post.Chris <- likelihood*prior.Chris(p)

plot(p, p.post.Anna , main = "Posteriors" , col='firebrick2', type='l', lty=3, lwd = 2, xlab="p", ylab=
lines(p, p.post.Benny , lwd = 2, col = 'blue', lty = 3)
lines(p, p.post.Chris , lwd = 2, col = 'green', lty = 3)
legend("topright", inset = 0.05 ,c("Anna", "Benny", "Chris"), col = c('firebrick2','blue', "green"), lty
```

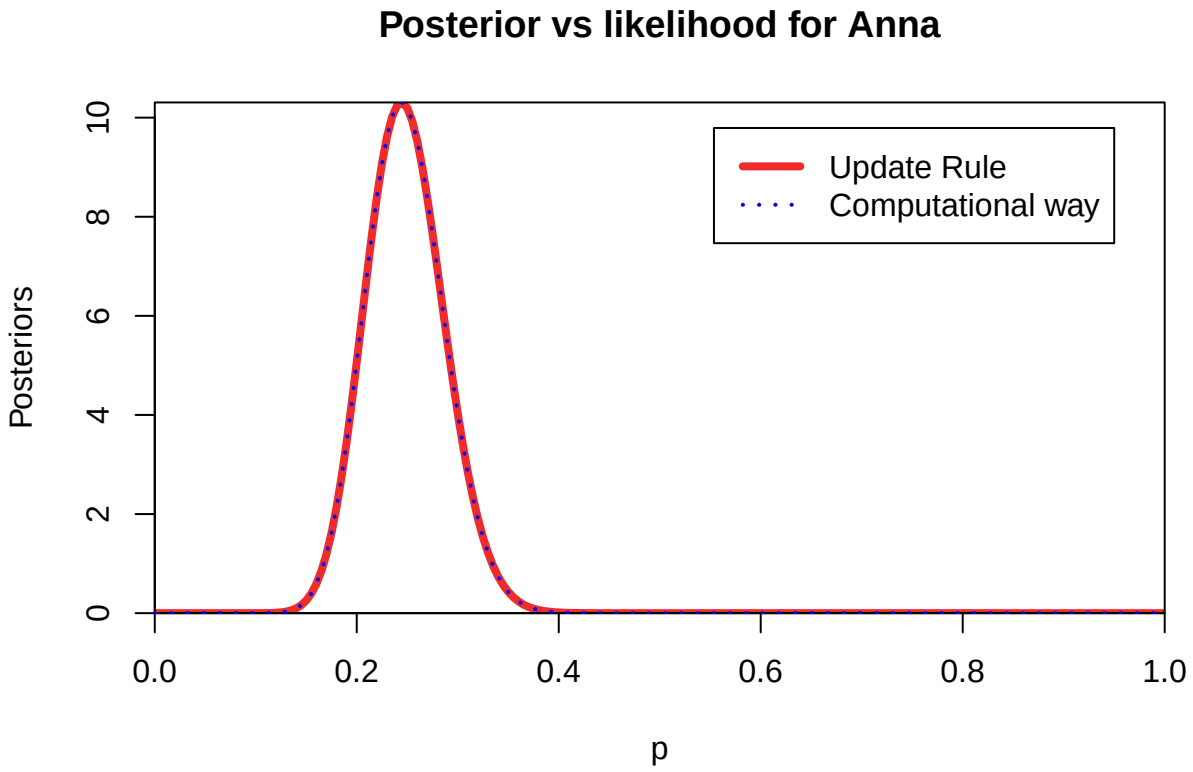



```
p.norm.Anna  <- p.post.Anna /( delta.p*sum(p.post.Anna))
p.norm.Benny <- p.post.Benny/( delta.p*sum(p.post.Benny))
p.norm.Chris <- p.post.Chris/( delta.p*sum(p.post.Chris))

plot(p, p.norm.Anna , main = "Posteriors normalized and means" , xaxs='i', yaxs='i', col='firebrick2',
p.mean.Anna <- delta.p*sum(p*p.norm.Anna)
abline (v=p.mean.Anna , lty=3, col='firebrick2')


lines(p, p.norm.Benny , lwd = 2, col = 'blue', lty = 3)
p.mean.Benny <- delta.p*sum(p*p.norm.Benny)
abline (v=p.mean.Benny , lty=3, col='blue')


lines(p, p.norm.Chris , lwd = 2, col = 'green', lty = 3)
p.mean.Chris <- delta.p*sum(p*p.norm.Chris)
abline (v=p.mean.Chris , lty=3, col='green')
legend("topright", inset = 0.05 ,c("Anna", "Benny", "Chris"), col = c('firebrick2','blue', "green"), lty
```



Note that we can rewrite the beta distribution found by computation by using the update rule we have learnt during lessons.

```
mu  <- 0.2
std <- 0.08
var <- std**2

#update rule
alpha.prior <- ((1 - mu)/var - 1/mu)*mu^2
beta.prior <- alpha.prior*(1/mu - 1)
alpha.post_param <- alpha.prior+k
beta.post_param <- beta.prior+n-k

post.beta <- dbeta(x=p, alpha.post_param , beta.post_param )
plot(p, post.beta, main = "Posterior vs likelihood for Anna" , xaxs='i', yaxs='i', col='firebrick2', ty
lines(p, p.norm.Anna , lwd = 2, col = 'blue', lty = 3)
legend("topright", inset = 0.05 ,c("Update Rule", "Computational way"), col = c('firebrick2','blue'), l
```



The normalized posterior found by computation and the one with the update rule indeed coincide!

Give an estimate of the most probable value and the 95% credibility interval.

```
message(sprintf("Most probable value for Anna is: %.3f", p.mean.Anna))
```

```
## Most probable value for Anna is: 0.248
```

```
f <- function(x) dbeta(x, shape1 = alpha.post_param , shape2 = beta.post_param)
lowerbound1 <- 0.185
upperbound1 <- 0.39
conf1 <- integrate(f , lowerbound1, upperbound1)
message(sprintf("95percent CL interval for Anna is: [%.3f, %.3f]", lowerbound1, upperbound1))
```

```
## 95percent CL interval for Anna is: [0.185, 0.390]
```

```
message(sprintf("\nMost probable value for Benny is: %.3f", p.mean.Benny))
```

```
##
## Most probable value for Benny is: 0.265
```

```
#first we need to normalize
f <- function(p) dbinom(x = k, size = n, prob = p)
norm <- integrate(f, 0 , 1)$value

#redefine the function normalized
f.norm <- function(p) dbinom(x = k, size = n, prob = p)/norm

#compute the CL interval
lowerbound2 <- 0.195
upperbound2 <- 0.40
conf2 <- integrate(f.norm , lowerbound2, upperbound2 )
message(sprintf("95percent CL interval for Benny is: [%.3f, %.3f]", lowerbound2, upperbound2))
```

```
## 95percent CL interval for Benny is: [0.195, 0.400]
```

```
message(sprintf("\nMost probable value for Chris is: %.3f", p.mean.Chris))
```

```
##
## Most probable value for Chris is: 0.262
```

```
f <- function(p) dbinom(x=k, size=n, prob=p)*prior.Chris(p)
norm <- integrate(f, 0 , 1)$value

#redefine the function normalized
f.norm <- function(p) dbinom(x = k, size = n, prob = p)*prior.Chris(p)/norm

#compute the CL interval
lowerbound3 <- 0.195
upperbound3 <- 0.40
conf3 <- integrate(f.norm , lowerbound3, upperbound3 )

message(sprintf("95percent CL interval for Chris is: [%.3f, %.3f]", lowerbound3, upperbound3))
```

```
## 95percent CL interval for Chris is: [0.195, 0.400]
```

Exercise 3

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

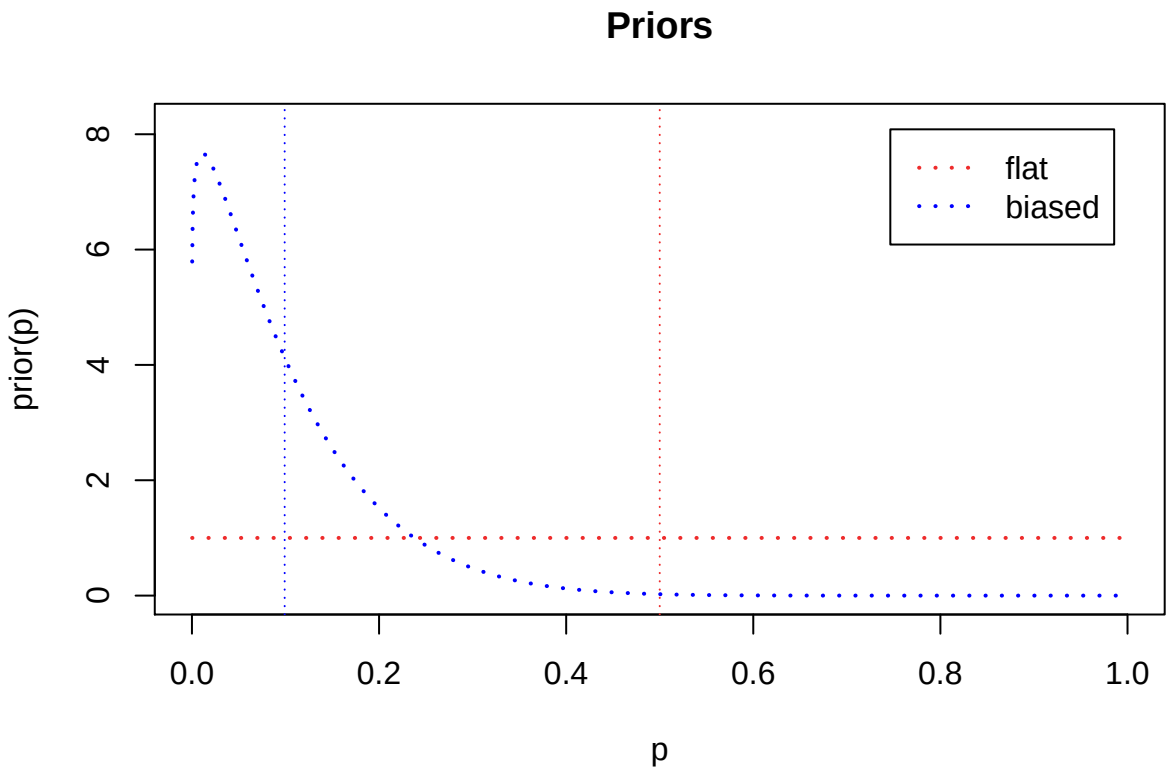
T, T, T, T, T, H, T, T, H, H, T, T, H, H, H, T, H, T, H, T, H, H, T, H, T, H, T, H, H, H

Assuming a flat prior, and a beta prior, plot the likelihood, prior and posterior distributions for the data set.

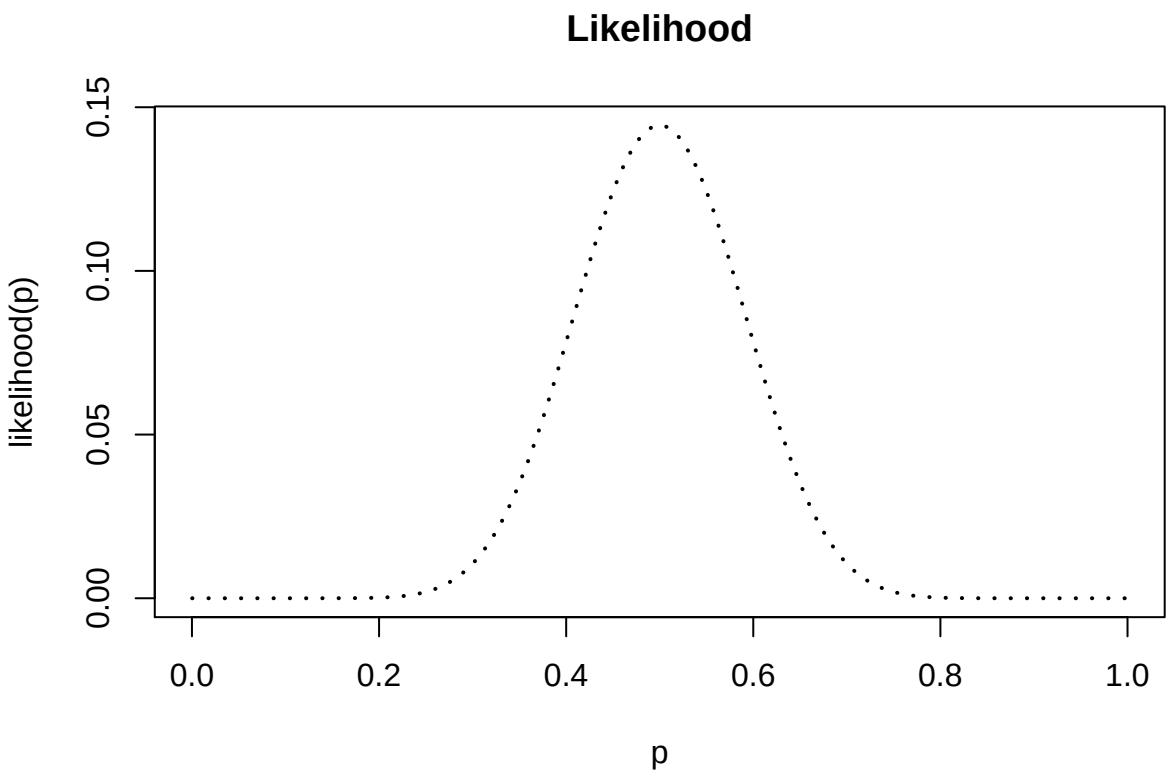
Let us assume the coin is not fair at all, so the beta prior will be peaked at one side.

```
#r is number of heads
n = 30
r = 15

unif.alpha <- 1
unif.beta  <- 1
prior.beta.flat <- function (x, b = 1, a = 0){
  return (dbeta(x, shape1 = 1, shape2 = 1))
}

bias.alpha <- 1.1
bias.beta  <- 10
prior.beta.biased <- function(x) {
  return (dbeta(x, shape1 = 1.1, shape2 = 10))
}

likelihood <- function(p) dbinom(x = r, size = n, prob = p)

n.sample <- 2000
delta.p <- 1/n.sample
p <- seq(from=1/(2*n.sample ), by=1/n.sample , length.out = n.sample )

plot(p, prior.beta.flat(p), main = "Priors" , col='firebrick2', type='l', lty=3, lwd = 2, xlab="p", ylab
lines(p, prior.beta.biased(p) , lwd = 2, col = 'blue', lty = 3)
legend("topright", inset = 0.05 ,c("flat", "biased"), col = c('firebrick2','blue'), lty = c(3,3), lwd =
```

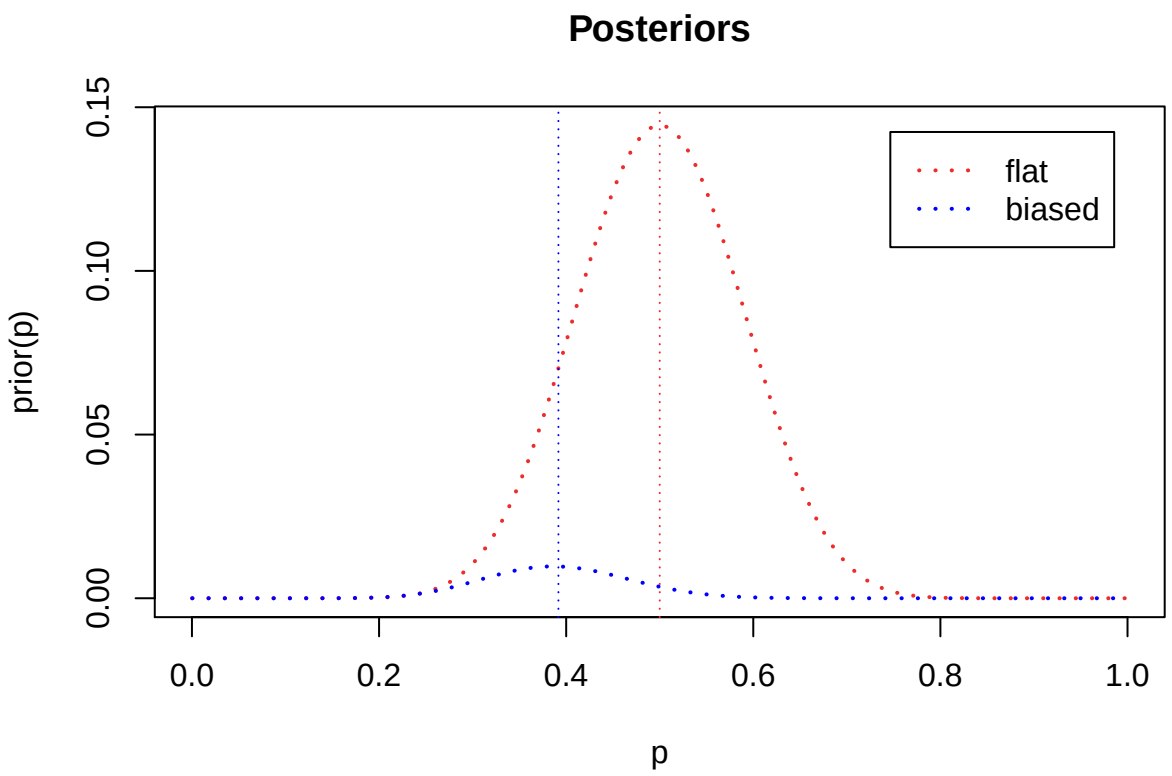
```r
mean.beta <- function(alpha, beta, r = 0, n = 0)  (alpha +r)/(alpha+beta+n)
mode.beta <- function(alpha, beta, r = 0, n = 0)  (alpha +r-1)/(alpha+beta+n-2)

#expected values for priors
abline (v = mean.beta(unif.alpha, unif.beta) , lty=3, col='firebrick2')
abline (v = mean.beta(bias.alpha, bias.beta) , lty=3, col='blue')
```



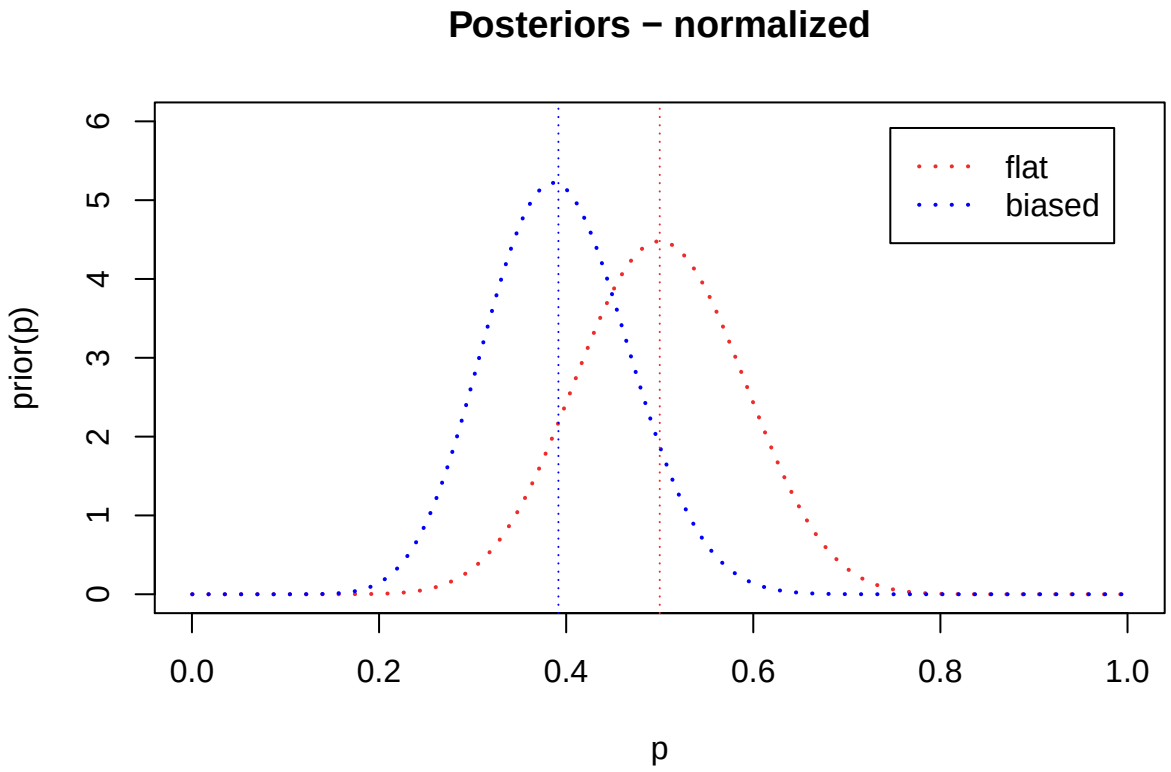

```r
plot(p, likelihood(p), main = "Likelihood" , col='black', type='l', lty=3, lwd = 2, xlab="p", ylab= "li
```




```
#now plot the posteriors

posterior.flat <- function(p) prior.beta.flat(p)*likelihood(p)
posterior.biased <- function(p) prior.beta.biased(p)*likelihood(p)

plot(p, posterior.flat(p), main = "Posteriors" , col='firebrick2', type='l', lty=3, lwd = 2, xlab="p", y
lines(p, posterior.biased(p) , lwd = 2, col = 'blue', lty = 3)
legend("topright", inset = 0.05 ,c("flat", "biased"), col = c('firebrick2','blue'), lty = c(3,3), lwd =

#expected values for posteriors
abline (v = mean.beta(unif.alpha, unif.beta, r = 15, n = 30) , lty=3, col='firebrick2')
abline (v = mean.beta(bias.alpha, bias.beta, r = 15, n = 30) , lty=3, col='blue')
```




```
#now normalize
norm1 <- integrate(posterior.flat, 0, 1 )$value
posterior.flat.norm <- function(p) posterior.flat(p)/norm1

norm2 <- integrate(posterior.biased, 0, 1 )$value
posterior.biased.norm <- function(p) posterior.biased(p)/norm2

plot(p, posterior.flat.norm(p), main = "Posteriors - normalized" , col='firebrick2', type='l', lty=3, lw
lines(p, posterior.biased.norm(p) , lwd = 2, col = 'blue', lty = 3)
legend("topright", inset = 0.05 ,c("flat", "biased"), col = c('firebrick2','blue'), lty = c(3,3), lwd =

#expected values for posteriors normalized
abline (v = mean.beta(unif.alpha, unif.beta, r = 15, n = 30) , lty=3, col='firebrick2')
abline (v = mean.beta(bias.alpha, bias.beta, r = 15, n = 30) , lty=3, col='blue')
```



We can see as if we choose a biased prior, we would need some more extractions in order to say something about the coin. It is probable that it is fair, but in order to be pretty sure of that we should extract more. Note in addition that when the posterior is not normalized, the probability for the different p's are quite small compaired to the flat prior. We can always compute the statistics through normalization, but still it should tells us that maybe the biased prior is not the best one that one can pick because it is way less probable.

Evaluate the most probable value for the coin probability p and, integrating the posterior probability distribution, give an estimate for a 95% credibility interval.

```
message(sprintf("Most probable value for p and the uniform prior is the mode: %.3f", mode.beta(unif.alp
```

```
## Most probable value for p and the uniform prior is the mode: 0.500
```

```
lowerbound1 <- 0.24
upperbound1 <- 0.65
conf1 <- integrate(posterior.flat.norm , lowerbound1, upperbound1)$value
message(sprintf("95percent CL interval for uniform prior is: [%.3f, %.3f]", lowerbound1, upperbound1))
```

```
## 95percent CL interval for uniform prior is: [0.240, 0.650]
```

```
message(sprintf("Most probable value for p and the biased prior is the mode: %.3f", mode.beta(bias.alpha
```

```
## Most probable value for p and the biased prior is the mode: 0.386
```

```
lowerbound2 <- 0.24
upperbound2 <- 0.54
conf2 <- integrate(posterior.biased.norm , lowerbound2, upperbound2)$value
message(sprintf("95percent CL interval for the biased prior is: [%.3f, %.3f]", lowerbound2, upperbound2
```

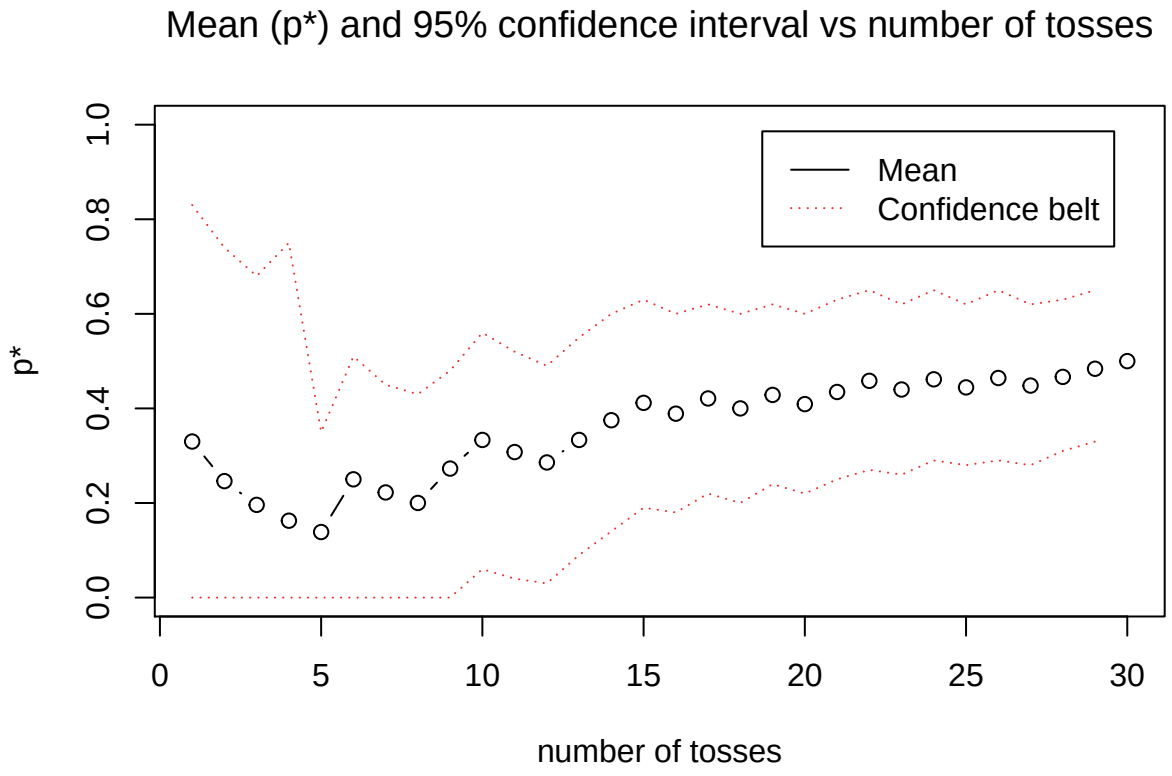
```
## 95percent CL interval for the biased prior is: [0.240, 0.540]
```

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). I assumed the posterior to be symmetric, and so it follows that the integral limits used for computed the Confidence

intervals are symmetric wrt to the mode.

```
n <- 30

extractions <- c(0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1,

n.sample <- 100
delta.p <- 1/n.sample
p <- seq( from=0, by=1/n.sample, length.out = n.sample+1 )

alpha.prior <- 1
beta.prior  <- 1
p.prior <- function(p) {
  return (dbeta(x = p , alpha.prior , beta.prior) )
}

heads <- 0
index <- 1
prior.val <- p.prior(p)
p.mean <- c()
p.mode <- c()
confidence <- c()

for (r in extractions){

  #likelihood
  p.like.fun <- function(p) {
    return(dbinom(x = r, size = 1 , prob = p))
  }

  #normalize the posterior
  p.post <- p.like.fun(p)*prior.val
  const <- delta.p*sum(p.post)
  p.post.norm <- p.post/const

  #find the mean
  p.mean <- c(p.mean, delta.p*sum(p*p.post.norm))
  mode_val <- delta.p*match(max(p.post.norm) , p.post.norm)
  p.mode <- c(p.mode, mode_val)

  #find the confidence level
  for (single_value in p){

    pos.number.bins <- min((mode_val + single_value) %/% delta.p, n.sample)
    neg.number.bins <- max(1, (mode_val - single_value) %/% delta.p)

    cum.left <- delta.p*(sum(p.post.norm[1:neg.number.bins-1]))
    cum.right <- delta.p*(sum(p.post.norm[1:pos.number.bins+1]))

    integral <- cum.right - cum.left

    if (integral > 0.95 ) {
      confidence <- c(confidence, single_value)
      break
```

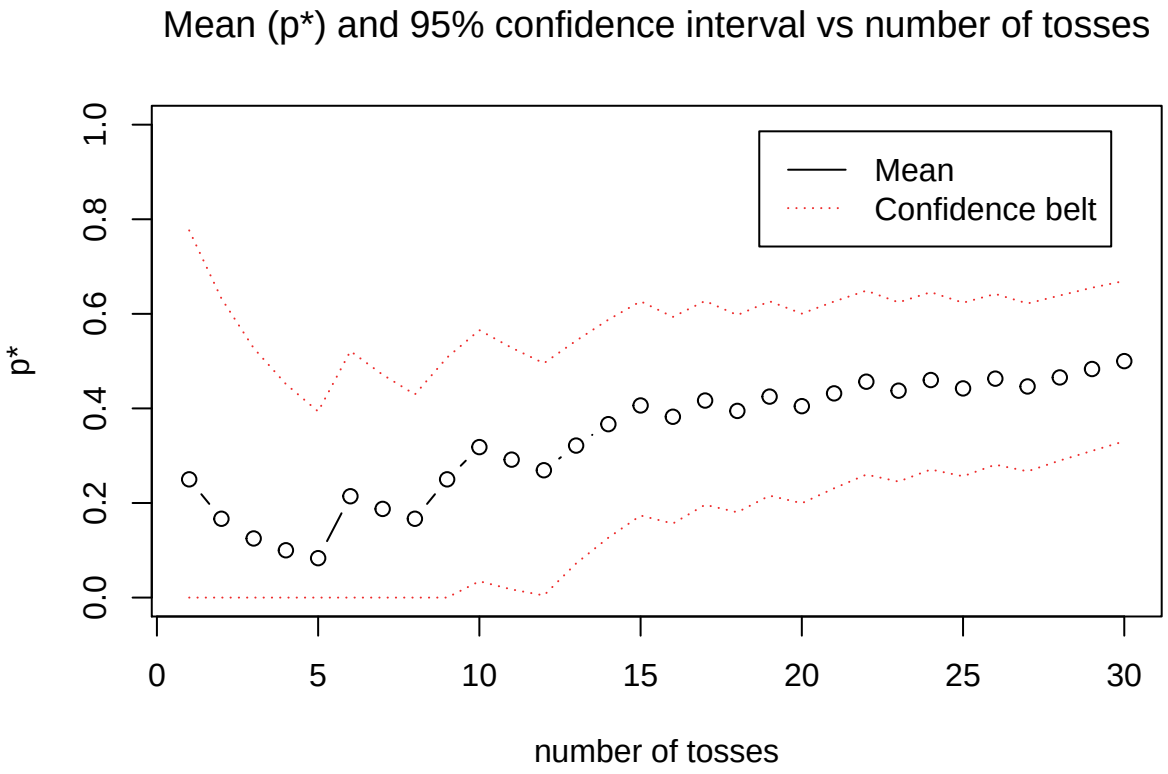
```
    }
  }
  index <- index + 1

  #use the posterior as new prior
  prior.val <- p.post.norm
  index <- index + 1

  integral <- cumsum(p.post.norm)

}

lowerbound.fun <- function(vec) {
  confid <- c()
  index  <- 1
  for (i in vec) {
    ifelse ((p.mode[index] - i ) < 0, confid <- c(confid, 0), confid <- c(confid, p.mode[index] - i ))
    index <- index + 1
  }
  return (confid)
}

upperbound.fun <- function(vec) {
  confid <- c()
  index  <- 1
  for (i in vec) {
    ifelse ((p.mode[index] + i ) > 1, confid <- c(confid, 1), confid <- c(confid, p.mode[index] + i))
    index <- index + 1
  }
  return (confid)
}

lowerbound <- lowerbound.fun(confidence)
upperbound <- upperbound.fun(confidence)

plot(1:30, p.mean, main = expression("Mean (p*) and 95% confidence interval vs number of tosses"), ylim
lines(lowerbound, col='firebrick2', lty = 3)
lines(upperbound, col='firebrick2', lty = 3)
legend("topright", inset = 0.05 ,c("Mean", "Confidence belt"), col = c('black','firebrick2'), lty = c(1
```



Doing it analytically:

```
n <- 30

extractions <- c(0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1,
#extractions <- c(0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0)
n.sample <- 1000
delta.p <- 1/n.sample
p <- seq(from=1/(2*n.sample ), by=1/n.sample , length.out = n.sample )

alpha.prior <- 1
beta.prior  <- 1
p.prior <- function(p) {
  return (dbeta(x = p , alpha.prior , beta.prior) )
}

heads <- 0
index <- 1
prior.val <- p.prior(p)
p.mean <- c()
confidence <- c()
p.mode <- c()

for (r in extractions){
  #tells how many extractions and heads we have done up to now
  heads <- heads + r

  alpha.post <- alpha.prior + heads
  beta.post <- beta.prior + index - heads
  p.post <- dbeta(x, alpha.post, alpha.post )
```

```
  mean <- (alpha.post + heads)/(alpha.post + beta.post + index)
  mode <- (heads/index)

  #find the mean
  p.mean <- c(p.mean, mean)
  p.mode <- c(p.mode, mode)

  #find the confidence level
  for (single_value in p){
    integral <- (pbeta(mode + single_value, alpha.post, beta.post ) - pbeta(mode - single_value, alpha.p
    if (integral > 0.95) {
      confidence <- c(confidence, single_value)
      break
    }
  }
  index <- index + 1
}

lowerbound.fun <- function(vec) {
  confid <- c()
  index  <- 1
  for (i in vec) {
    ifelse ((p.mode[index] - i ) < 0, confid <- c(confid, 0), confid <- c(confid, p.mode[index] - i ))
    index <- index + 1
  }
  return (confid)
}

upperbound.fun <- function(vec) {
  confid <- c()
  index  <- 1
  for (i in vec) {
    ifelse ((p.mode[index] + i ) > 1, confid <- c(confid, 1), confid <- c(confid, p.mode[index] + i))
    index <- index + 1
  }
  return (confid)
}

lowerbound <- lowerbound.fun(confidence)
upperbound <- upperbound.fun(confidence)

plot(1:30, p.mean, main = expression("Mean (p*) and 95% confidence interval vs number of tosses"), ylim
lines(lowerbound, col='firebrick2', lty = 3)
lines(upperbound, col='firebrick2', lty = 3)
legend("topright", inset = 0.05 ,c("Mean", "Confidence belt"), col = c('black','firebrick2'), lty = c(1
```



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) ? No they return actually the same results, but of course analyzing it step by step involves more computational resources. Moreover the 95% confidence belt changes slightly between the computational formulas implemented and the analytical results, but it is still quite accurate.