Viterbo_Giuseppe_rlab03

April 28, 2023

[1]:
```
install.packages('tidyverse')
install.packages('gridExtra')
install.packages('emdbook')


library(tidyverse)
library(gridExtra)
library(emdbook)
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done

Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done

also installing the dependencies ‘bdsmatrix’, ‘mvtnorm’, ‘coda’, ‘bbmle’


Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done
```

Attaching core tidyverse packages

```
                 tidyverse 2.0.0
 dplyr     1.1.1        readr     2.1.4
 forcats   1.0.0        stringr   1.5.0
 ggplot2   3.4.2        tibble    3.2.1
 lubridate 1.9.2        tidyr     1.3.0
 purrr     1.0.1
```

Conflicts

```
              tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()
 Use the conflicted package
(<http://conflicted.r-lib.org/>) to force all conflicts to
become errors

Attaching package: ‘gridExtra’


The following object is masked from ‘package:dplyr’:

    combine
```

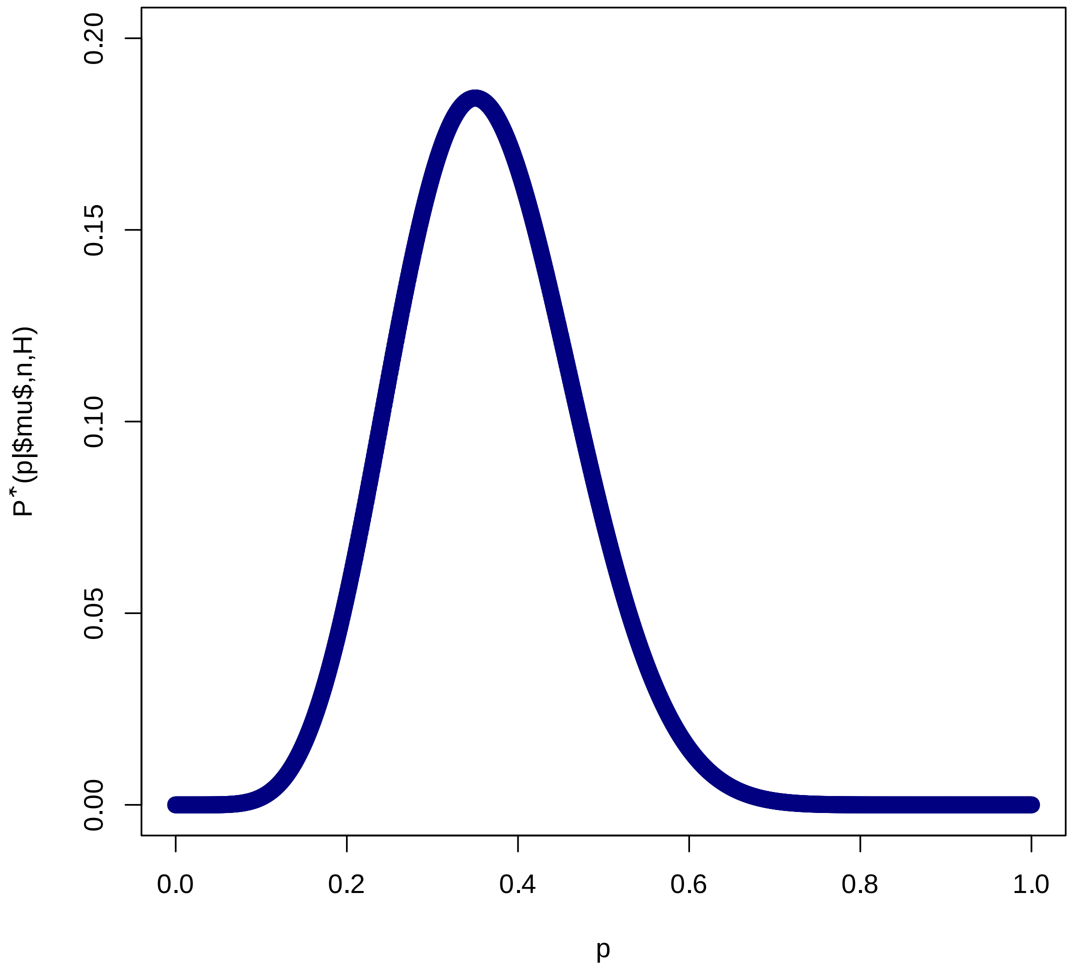
1 Ex.1

- 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:
- plot the posterior distribution and summerize the results computing the first two moments
- compute a 95% credibility interval and give the results in a summary table
- draw the limits on the plot of the posterio distribution

1.0.1 Uniform distribution prior

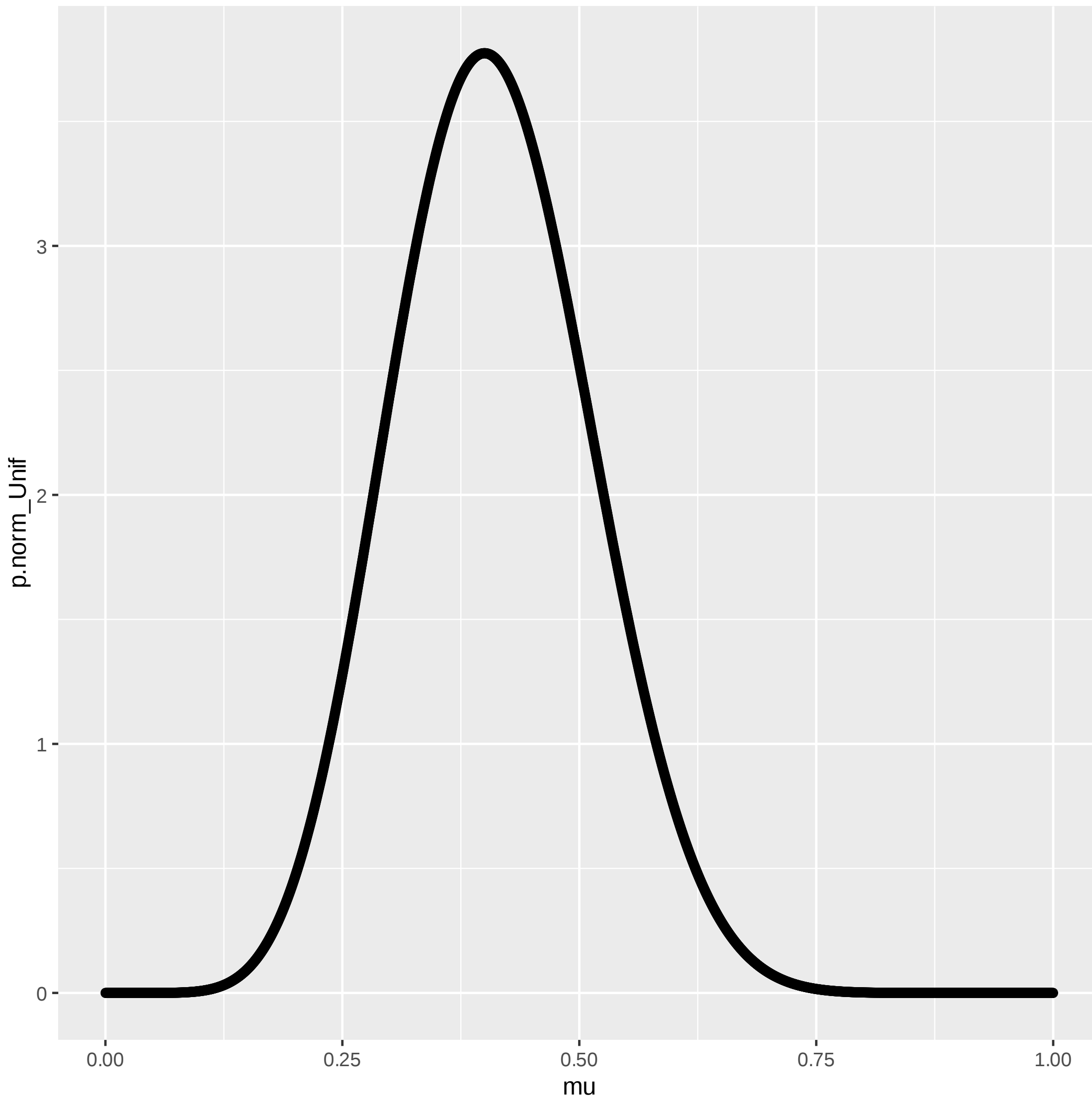
the normalization $P(\pi|H)$ dosen't depend on π

[5]:
```
y <- 7   #success
n <- 20  #number of trial
mu <- seq(0,1, length.out=1000)
p.post_notnorm <- dbinom(x=y, size=n, prob=mu)
plot(mu, p.post_notnorm,
     col='navy', lty=1, lwd=3,
     ylim=c(0,0.2),
     xlab='p',
     ylab=expression(paste(P^symbol('*'), '(p|$mu$,n,H)')))
```



[7]:
```
y <- 8
n <- 20;
n.sample <- 2000 #number of interval in which calculate the approximate␣
 ↪normalization function
delta.p <- 1/n.sample #length of the interval
mu <- seq(from = 1/(2*n.sample), by=1/n.sample, length.out = n.sample)␣
 ↪#possible chose of the p in the binomial distribution, called mu

p.star_Unif <- dbinom(x=y, size=n, prob=mu)
p.norm_Unif <- p.star_Unif/(delta.p * sum(p.star_Unif))
# plot(p, p.norm, col='red')
# points(p, p.star, col='navy')
ggplot() +
```

```
geom_point(aes(mu, p.norm_Unif))
```

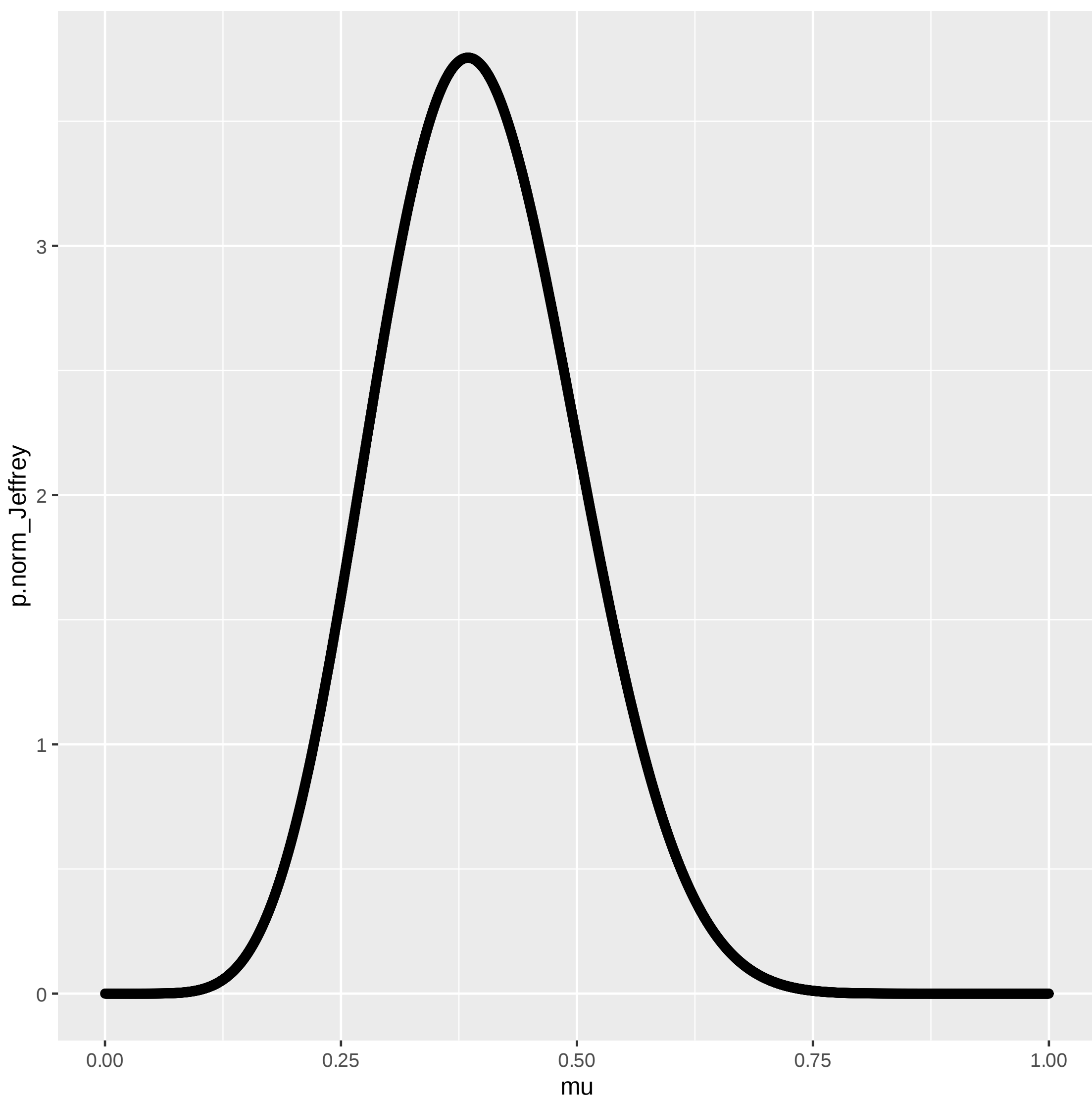


1.0.2 Jeffrey's Prior

it is a prior invariant under any continous transformation of the parameter: g(μ) $\propto \frac{1}{\sqrt{\mu}}$ for $\mu > 0$

[8]:
```
y <- 8
n <- 20
n.sample <- 2000
delta.p <- 1/n.sample
mu <- seq(from = 1/(2*n.sample), by=1/n.sample, length.out = n.sample)

#p.star <- dgamma(mu, shape=(y+1/2), scale=n)
```

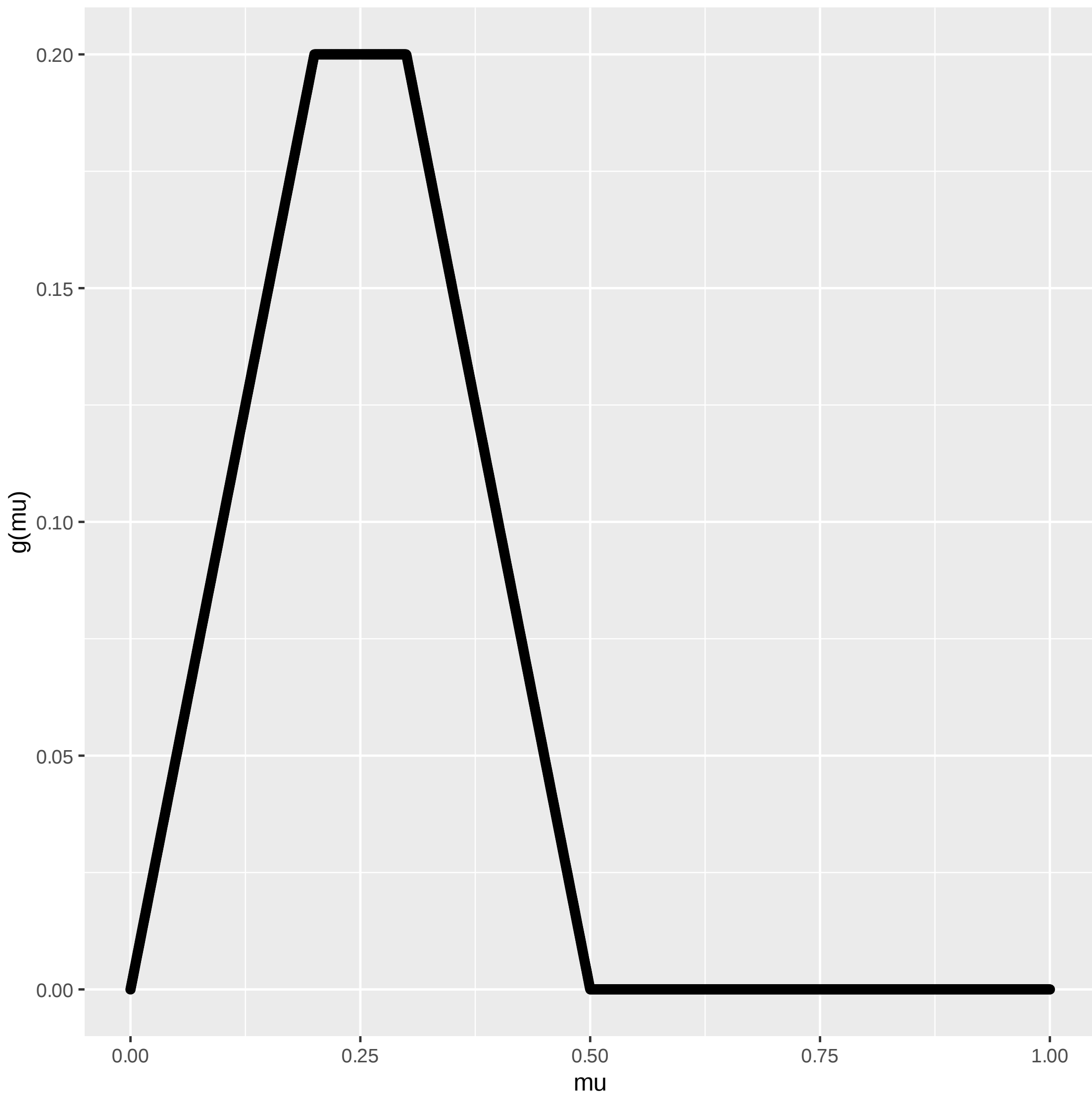
```r
# p.norm <- p.star / (delta.p * sum(p.star))
#p.norm_Jeffrey <- p.star / (integrate(function(k) {dgamma(k, shape=(y+1/2),
 ↪scale=n)}, lower = 0, upper = 1))$value #return the same as the aproximated
 ↪version

p.star_Jeffrey <- dbinom(x=y, size=n, prob=mu) * 1/((mu)**(1/2))
p.norm_Jeffrey <- p.star_Jeffrey / integrate(function(k) {dbinom(x=y, size=n,
 ↪prob=k) * 1/((k)**(1/2)) }, lower = 0, upper = 1)$value


ggplot() +
geom_point(aes(mu, p.norm_Jeffrey))
```



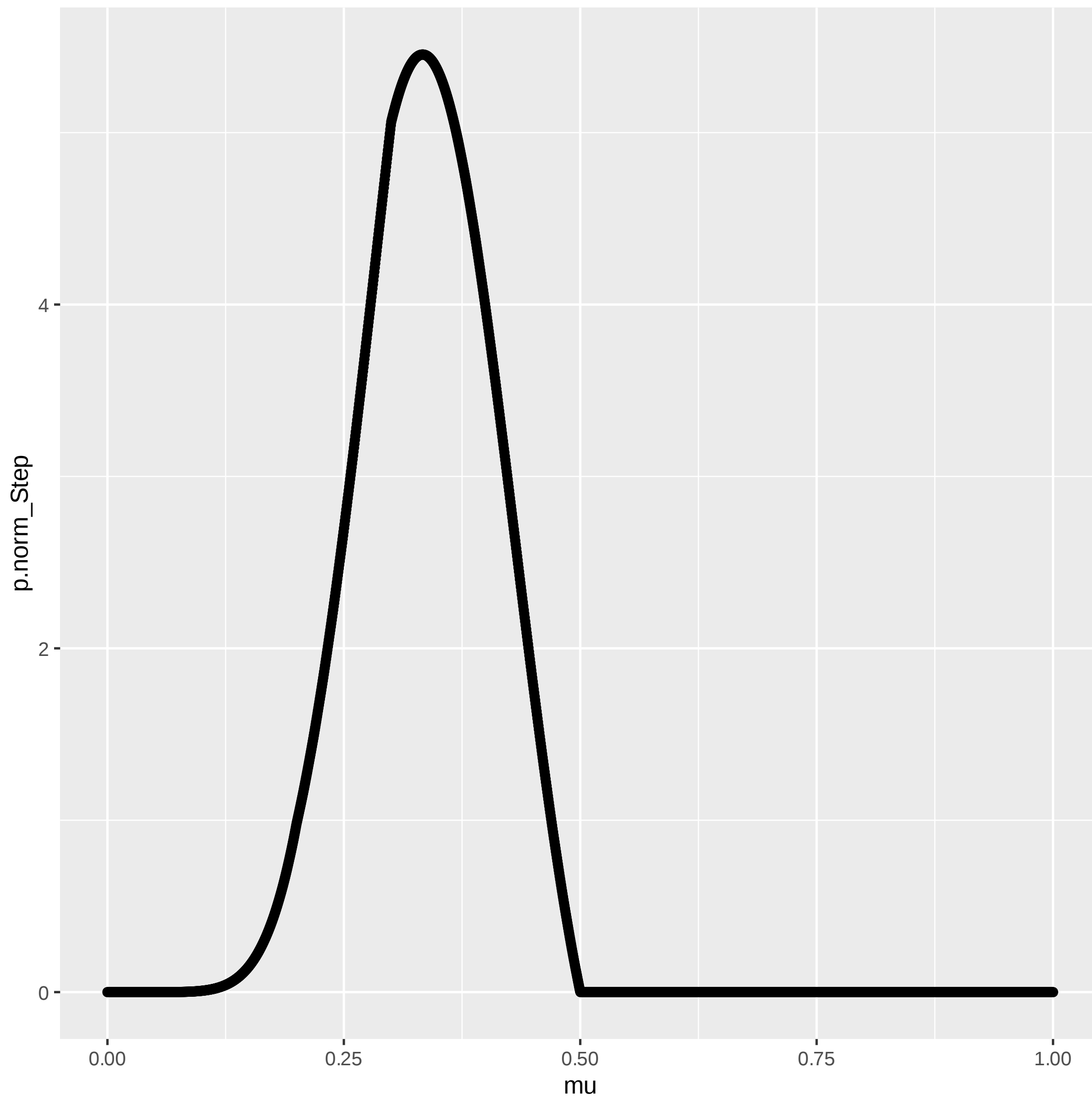
1.0.3 Step function

[9]:
```
y <- 8
n <- 20
mu <- seq(0,1,length.out = 2000)

g  <- function(k) {
        g.val <- ifelse(k <= 0.2,
                        k,
                        ifelse(k > 0.2 & k <= 0.3,
                               0.2,
                               ifelse(k > 0.3 & k <= 0.5,
                                      0.5 - k,
                                      ifelse(k>0.5,
                                             0,
                                             0)
                                     )
                              )
                        )
        return(g.val)
}
```

[10]:
```
ggplot()+
geom_point(aes(mu, g(mu)))
```

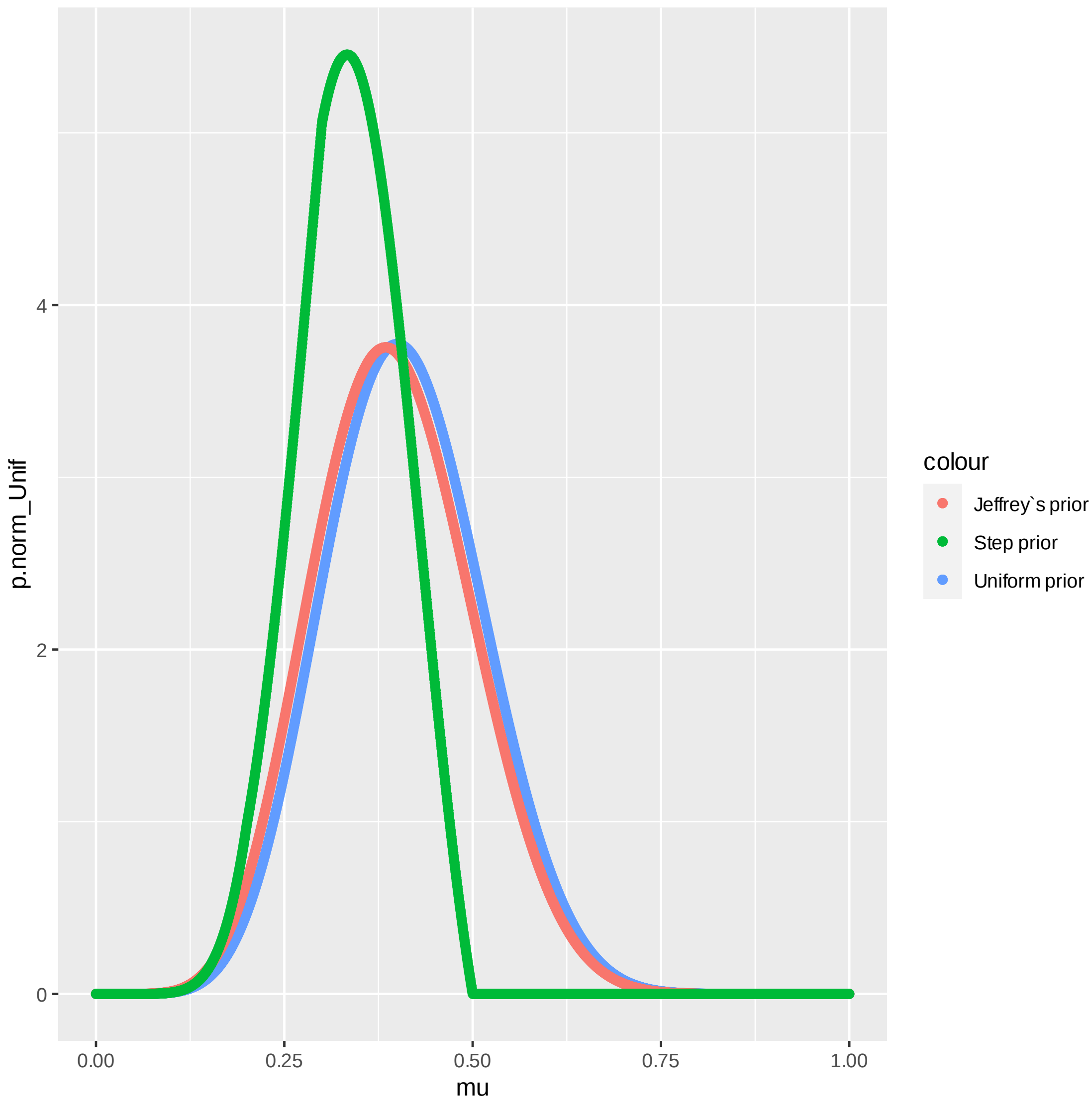


[11]:
```r
p.star_Step <- dbinom(x=y, size=n, prob=mu) * g(mu)
p.norm_Step <- p.star_Step/ (integrate(function(k) {dbinom(x=y, size=n, prob=k)
 ↪* g(k)}, lower = 0, upper = 1))$value

ggplot() +
geom_point(aes(mu, p.norm_Step))
```



2 Plot Posterior distribution

[12]:
```
ggplot() +
geom_point(aes(mu, p.norm_Unif, color='Uniform prior')) +
geom_point(aes(mu, p.norm_Jeffrey, color='Jeffrey`s prior')) +
geom_point(aes(mu, p.norm_Step, color='Step prior')) -> posterior_plot
posterior_plot
```



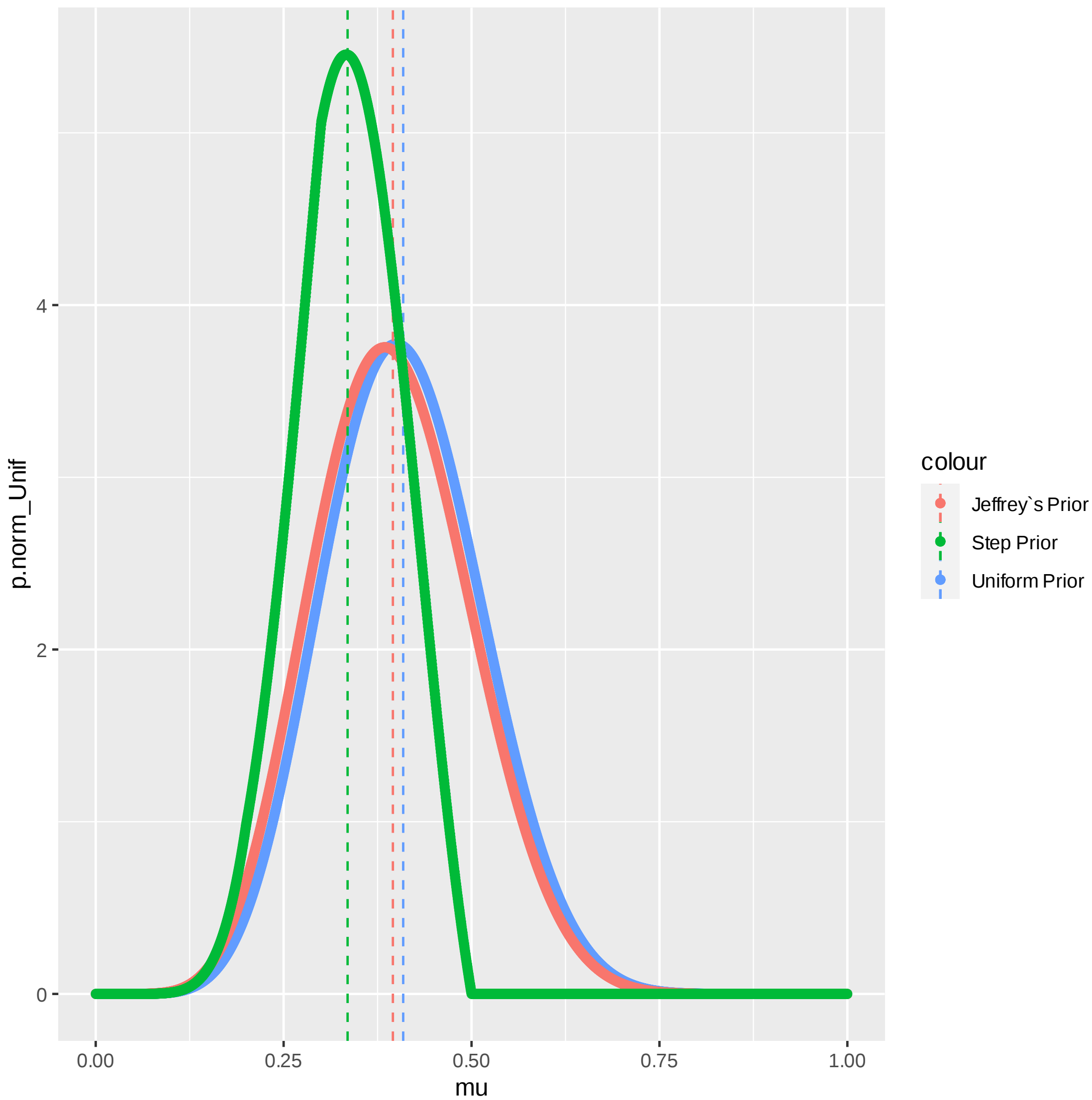
2.1 First and Second Momenta

[172]:
```
y <- 8; n <- 20; n.sample <- 2000; delta.mu <- 1/n.sample;
mu <- seq(from=1/(2*n.sample), by=1/n.sample, length.out = n.sample)

#Uniform prior
p.star_Unif<- dbinom(x=y, size=n, prob=mu) * dunif(mu, min=0, max=1)
p.norm_Unif <- p.star_Unif/(delta.mu * sum(p.star_Unif))
first_moment_Unif <- delta.mu * sum(mu * p.norm_Unif)
second_moment_Unif <- delta.mu * sum(((first_moment_Unif-mu)**2)*p.norm_Unif)


#Jeffrey prior
```

```r
p.star_Jeffrey <- dbinom(x=y, size=n, prob=mu) * 1/(mu ** (1/2))
p.norm_Jeffrey <- p.star_Jeffrey / (delta.mu*sum(p.star_Jeffrey))
first_moment_Jeffrey <- delta.mu * sum(mu * p.norm_Jeffrey)
second_moment_Jeffrey <- delta.mu * sum(((first_moment_Jeffrey-mu)**2) * p.norm_Jeffrey)


#Step prior
p.star_Step <- dbinom(x=y, size=n, prob=mu) * g(mu)
p.norm_Step <- p.star_Step / (delta.mu * sum(p.star_Step))
first_moment_Step <- delta.mu * sum(mu * p.norm_Step)
second_moment_Step <- delta.mu * sum(((first_moment_Step-mu)**2) * p.norm_Step)

#PLot
ggplot()+
geom_point(aes(mu, p.norm_Unif, color='Uniform Prior')) +
geom_vline(aes(xintercept=first_moment_Unif, color='Uniform Prior'), linetype='dashed')+

geom_point(aes(mu, p.norm_Jeffrey, color='Jeffrey`s Prior'))+
geom_vline(aes(xintercept=first_moment_Jeffrey, color='Jeffrey`s Prior'), linetype='dashed') +

geom_point(aes(mu, p.norm_Step, color='Step Prior'))+
geom_vline(aes(xintercept=first_moment_Step, color='Step Prior'), linetype='dashed')



Momenta <- tibble(
            Prior = c('Uniform', 'Jeffrey`s', 'Step'),
            First_momenta  = c(first_moment_Unif,  first_moment_Jeffrey, first_moment_Step),
            Second_momenta = c(second_moment_Unif, second_moment_Jeffrey, second_moment_Step),
            )
Momenta
```

A tibble: 3 × 3

Prior <chr>	First_momenta <dbl>	Second_momenta <dbl>
Uniform	0.4090909	0.010510241
Jeffrey‘s	0.3953488	0.010624362
Step	0.3350930	0.004673032



3 Compute 95% credibility interval and Plot the limits

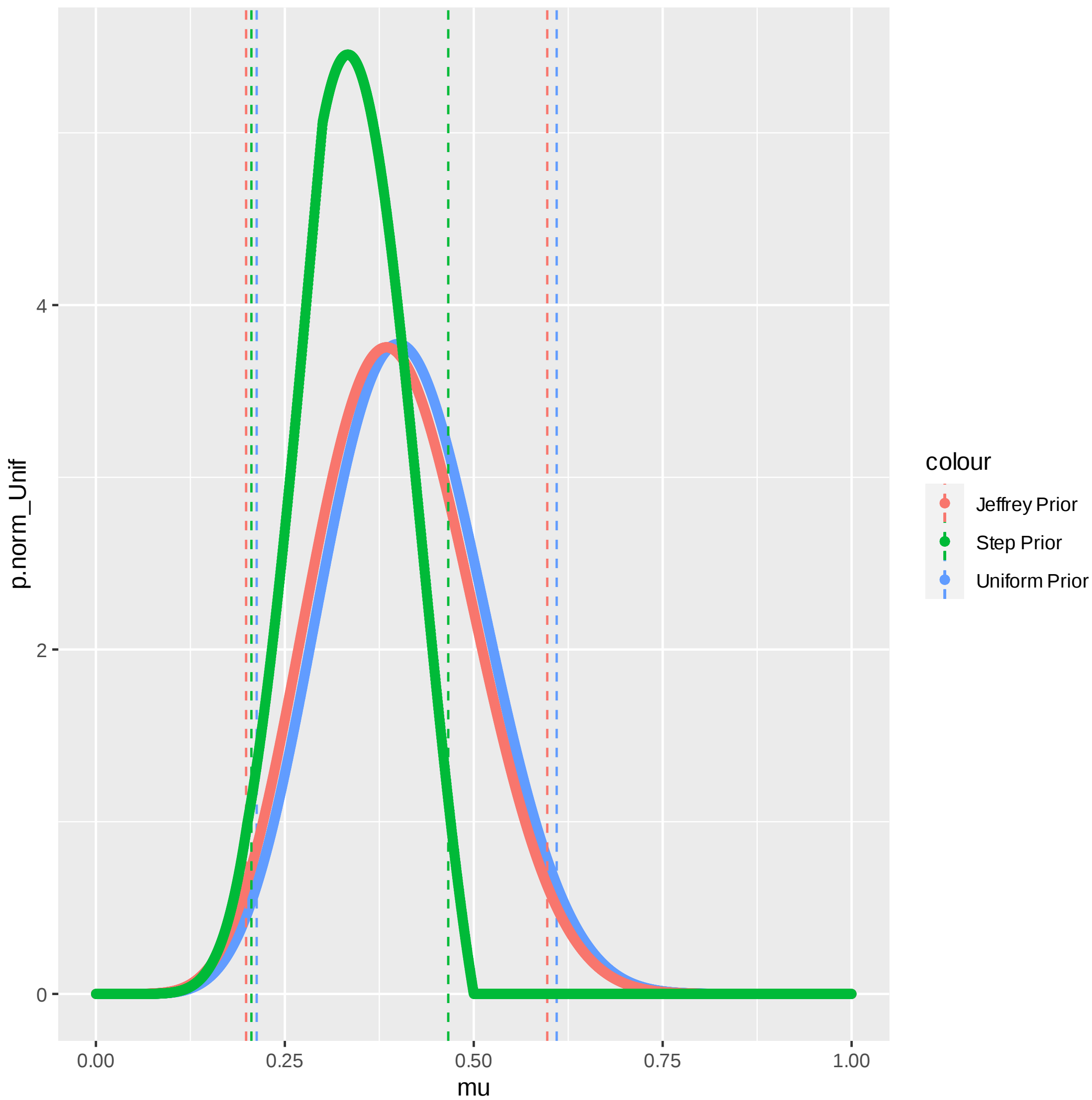
The function `ncredit` is used in order to compute the credibility interval

[147]:
```
# Uniform prior
lower_bound_Unif <- ncredint(mu, p.norm_Unif, level=0.95)[['lower']]
upper_bound_Unif <- ncredint(mu, p.norm_Unif, level=0.95)[['upper']]

# Jeffrey prior
lower_bound_Jeffrey <- ncredint(mu, p.norm_Jeffrey, level=0.95)[['lower']]
upper_bound_Jeffrey <- ncredint(mu, p.norm_Jeffrey, level=0.95)[['upper']]
```

```r
# Step prior
lower_bound_Step <- ncredint(mu, p.norm_Step, level=0.95)[['lower']]
upper_bound_Step <- ncredint(mu, p.norm_Step, level=0.95)[['upper']]

#Summary table
Credibility_Interval <- tibble(
                        Prior = c('Uniform', 'Jeffrey', 'Step'),
                        Lower_bound = c(lower_bound_Unif, lower_bound_Jeffrey,
 ↪lower_bound_Step),
                        Upper_bound = c(upper_bound_Unif, upper_bound_Jeffrey,
 ↪upper_bound_Step)
                        )
Credibility_Interval

#Plot the limits
ggplot()+
geom_point(aes(mu, p.norm_Unif, color='Uniform Prior')) +
geom_vline(aes(xintercept=lower_bound_Unif,color='Uniform Prior'),
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Unif, color='Uniform Prior'),
 ↪linetype='dashed')+

geom_point(aes(mu, p.norm_Jeffrey, color='Jeffrey Prior')) +
geom_vline(aes(xintercept=lower_bound_Jeffrey,color='Jeffrey Prior'),
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Jeffrey, color='Jeffrey Prior'),
 ↪linetype='dashed') +

geom_point(aes(mu, p.norm_Step, color='Step Prior')) +
geom_vline(aes(xintercept=lower_bound_Step,color='Step Prior'),
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Step, color='Step Prior'),
 ↪linetype='dashed')
```

A tibble: 3×3

Prior <chr>	Lower_bound <dbl>	Upper_bound <dbl>
Uniform	0.21275	0.60975
Jeffrey	0.19875	0.59725
Step	0.20575	0.46625



4 Ex.2

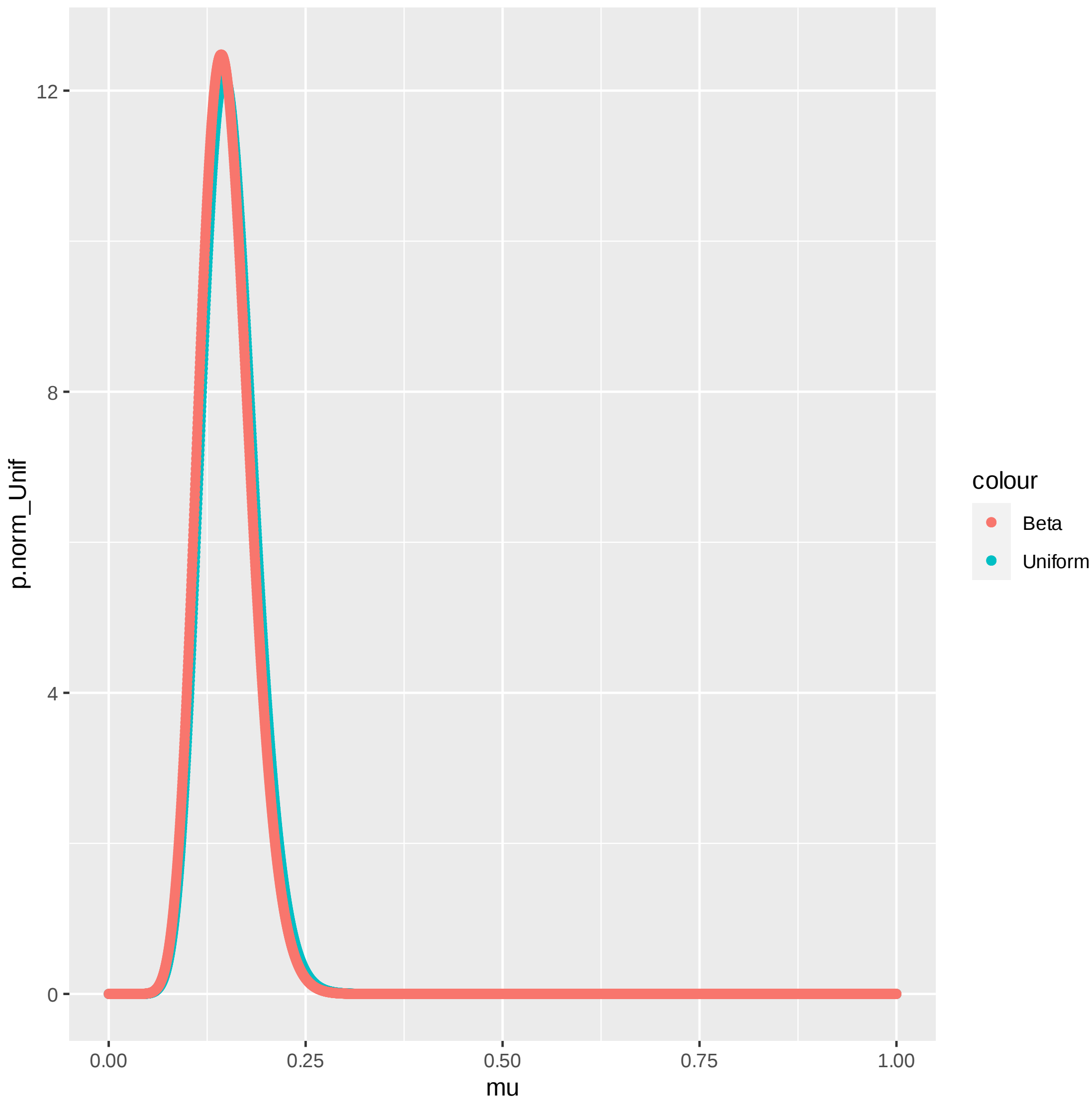
4.1 Plot the posterior distribution and compute first and second moments

[178]:
```
n <- 116
y <- 17
n.sample <- 4000; delta.mu <- 1/n.sample
mu <- seq(from=1/(2*n.sample), by = delta.mu, length.out = n.sample)

# Uniform prior
p.star_Unif <- dbinom(x=y, size=n, prob=mu) * dunif(mu, min = 0, max=1)
p.norm_Unif <- p.star_Unif/(delta.mu * sum(p.star_Unif))
```

```r
first_moment_Unif <- delta.mu * sum(mu * p.norm_Unif)
second_moment_Unif <- delta.mu * sum(((first_moment_Unif-mu)**2) * p.norm_Unif)

#Beta prior
p.star_Beta <- dbinom(x=y, size=n, prob=mu) * dbeta(mu,1,4)
p.norm_Beta <- p.star_Beta / (delta.mu * sum(p.star_Beta))
first_moment_Beta <- delta.mu * sum(mu * p.norm_Beta)
second_moment_Beta <- delta.mu * sum(((first_moment_Beta - mu)**2) * p.
  ↪norm_Beta)

Momenta <- tibble(
            Prior = c('Unif', 'Beta'),
            First_moment = c(first_moment_Unif, first_moment_Beta),
            Second_moment = c(second_moment_Unif, second_moment_Beta)
            )
Momenta

ggplot() +
geom_point(aes(mu, p.norm_Unif, color='Uniform')) +
geom_point(aes(mu, p.norm_Beta, color='Beta'))
```

A tibble: 2 × 3

Prior <chr>	First_moment <dbl>	Second_moment <dbl>
Unif	0.1525424	0.001086329
Beta	0.1487603	0.001037957

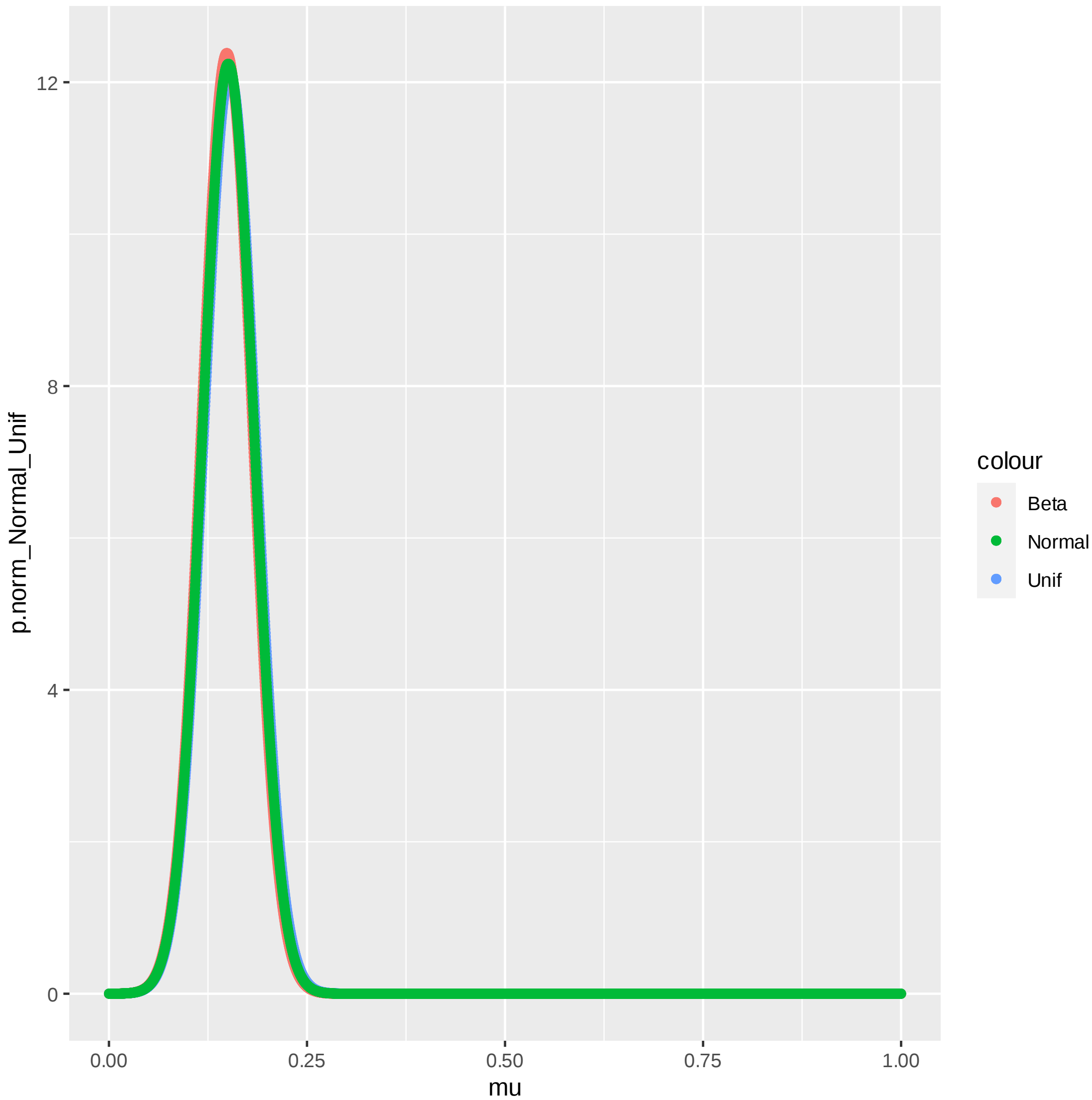


4.2 Find a normal approximation for the posterior

For the normal posterior distribution I'm gonna use a normal distribution with mean eqaul to the first moments of the posterior distribution find in the last ex, same for the std

[180]:
```
p.norm_Normal_Unif <- dnorm(mu, mean=first_moment_Unif,␣
↪sd=sqrt(second_moment_Unif))
p.norm_Normal_Beta <- dnorm(mu, mean=first_moment_Beta,␣
↪sd=sqrt(second_moment_Beta))

p.norm_Normal <- dnorm(mu, mean=(first_moment_Unif + first_moment_Beta)/2,
                         sd=sqrt((second_moment_Unif + second_moment_Beta)/2))
```

```
ggplot() +
geom_point(aes(mu, p.norm_Normal_Unif, color='Unif'))+
geom_point(aes(mu, p.norm_Normal_Beta, color='Beta')) +
geom_point(aes(mu, p.norm_Normal, color='Normal'))
```



4.3 Compute a 95% credibility interval both for the original posterior and for the normal approximation, giving the results in a summary table

[186]:
```
lower_bound_Unif <- ncredint(mu, p.norm_Unif, level = 0.95)[['lower']]
upper_bound_Unif <- ncredint(mu, p.norm_Unif, level = 0.95)[['upper']]

lower_bound_Beta <- ncredint(mu, p.norm_Beta, level = 0.95)[['lower']]
```

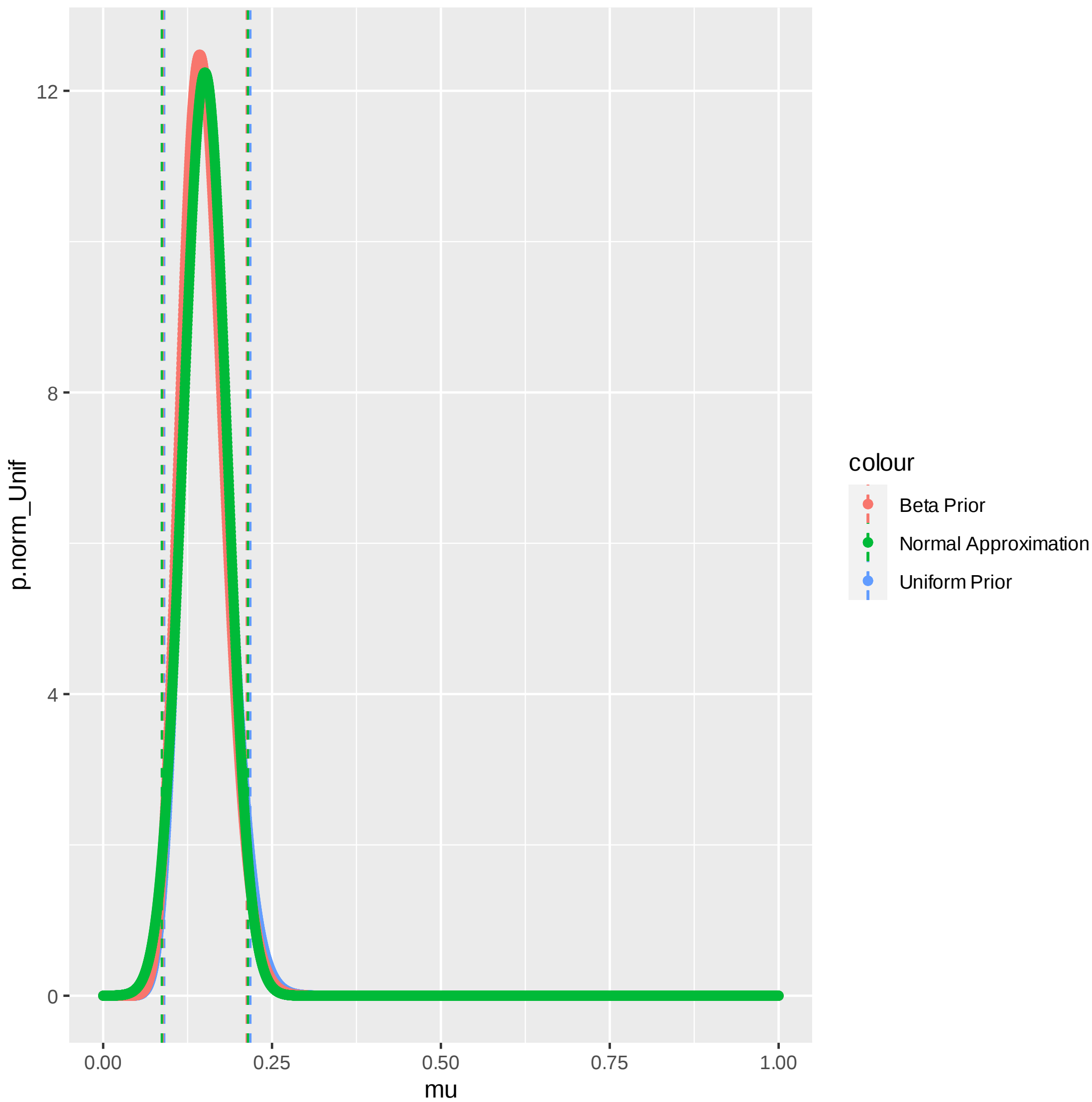
```
upper_bound_Beta <- ncredint(mu, p.norm_Beta, level = 0.95)[['upper']]

lower_bound_Normal <- ncredint(mu, p.norm_Normal, level = 0.95)[['lower']]
upper_bound_Normal <- ncredint(mu, p.norm_Normal, level = 0.95)[['upper']]

Credibility_Interval <- tibble(
                        Prior = c('Unif', 'Beta', 'Normal approximation'),
                        lower_bound = c(lower_bound_Unif, lower_bound_Beta,␣
 ↪lower_bound_Normal),
                        upper_bound = c(upper_bound_Unif, upper_bound_Beta,␣
 ↪upper_bound_Normal)
                        )
Credibility_Interval
```

A tibble: 3 × 3

Prior <chr>	lower_bound <dbl>	upper_bound <dbl>
Unif	0.090375	0.217875
Beta	0.088125	0.212625
Normal approximation	0.086875	0.214375

4.4 Add the limits on the plot of the posterior distributions

[188]:
```
ggplot() +
geom_point(aes(mu, p.norm_Unif, color='Uniform Prior')) +
geom_vline(aes(xintercept=lower_bound_Unif,color='Uniform Prior'),␣
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Unif, color='Uniform Prior'),␣
 ↪linetype='dashed')+

geom_point(aes(mu, p.norm_Beta, color='Beta Prior')) +
geom_vline(aes(xintercept=lower_bound_Beta,color='Beta Prior'),␣
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Beta, color='Beta Prior'),␣
 ↪linetype='dashed') +

geom_point(aes(mu, p.norm_Normal, color='Normal Approximation')) +
geom_vline(aes(xintercept=lower_bound_Normal,color='Normal Approximation'),␣
 ↪linetype='dashed')+
geom_vline(aes(xintercept=upper_bound_Normal, color='Normal Approximation'),␣
 ↪linetype='dashed')
```

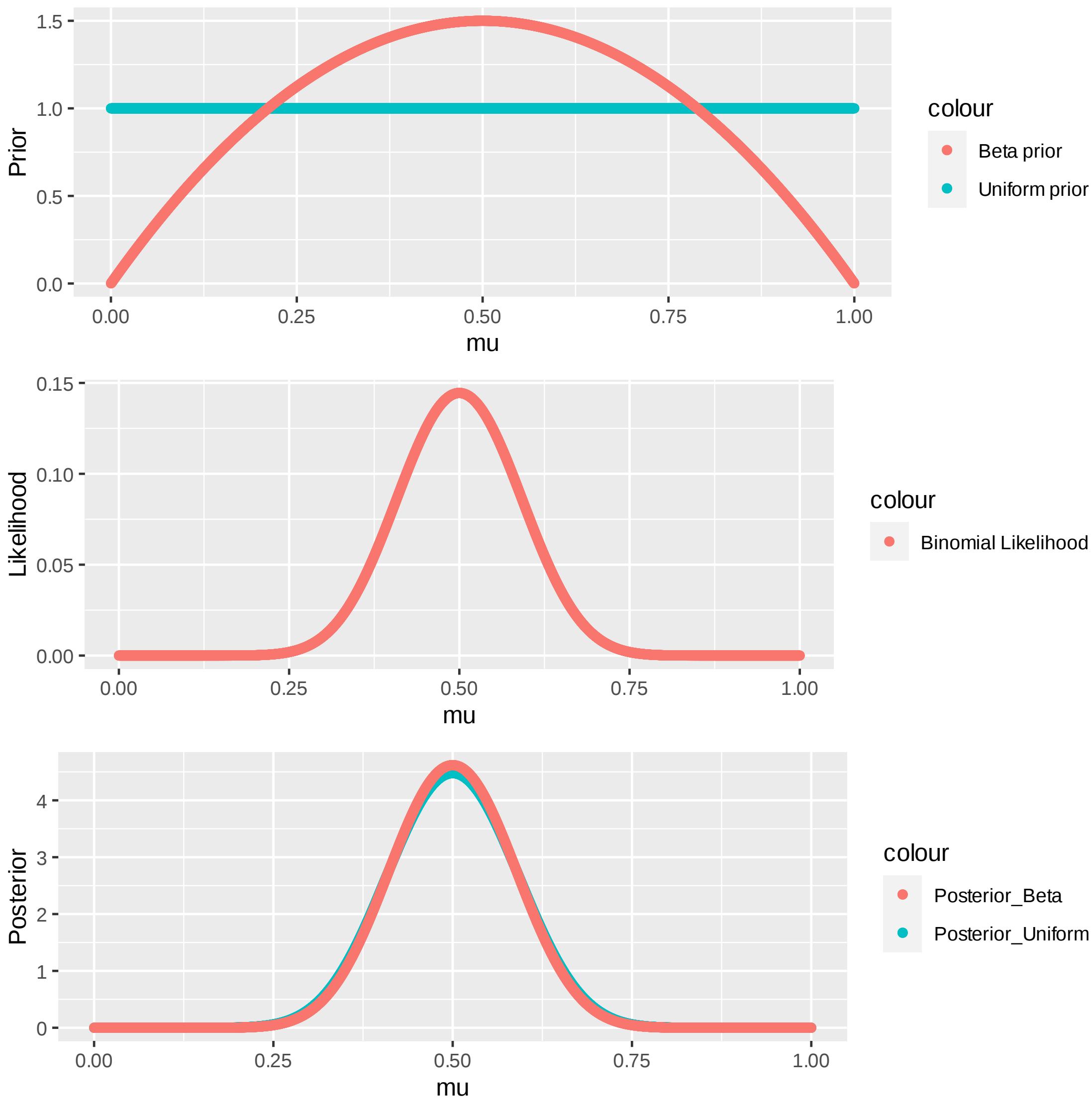


5 Ex. 3

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

[214]:
```
n <- 30
y <- 15

n.sample <- 4001; delta.mu = 1/n.sample
mu <- seq(from=1/(2*n.sample), by = delta.mu, length.out = n.sample)

# Uniform prior
```

```r
p.star_Unif <- dbinom(x=y, size=n, prob=mu) * dunif(mu, min = 0, max=1)
p.norm_Unif <- p.star_Unif/(delta.mu * sum(p.star_Unif))
first_moment_Unif <- delta.mu * sum(mu * p.norm_Unif)
second_moment_Unif <- delta.mu * sum(((first_moment_Unif-mu)**2) * p.norm_Unif)

#Beta prior
p.star_Beta <- dbinom(x=y, size=n, prob=mu) * dbeta(mu,2, 2)
p.norm_Beta <- p.star_Beta / (delta.mu * sum(p.star_Beta))
first_moment_Beta <- delta.mu * sum(mu * p.norm_Beta)
second_moment_Beta <- delta.mu * sum(((first_moment_Beta - mu)**2) * p.
 ↪norm_Beta)

#Prior plot
ggplot() +
geom_point(aes(mu, dunif(mu, min=0, max=1), color='Uniform prior')) +
geom_point(aes(mu, dbeta(mu, 2,2), colour='Beta prior')) +
labs(y='Prior')-> prior_plot

#Likelihood plot
ggplot() +
geom_point(aes(mu, dbinom(x=y, size=n, prob=mu), color='Binomial Likelihood')) +
labs(y='Likelihood')-> Likelihood_plot

#Posterior plot
ggplot() +
geom_point(aes(mu, p.norm_Unif, color='Posterior_Uniform')) +
geom_point(aes(mu, p.norm_Beta, color='Posterior_Beta'))+
labs(y='Posterior')-> posterior_plot


grid.arrange(prior_plot, Likelihood_plot, posterior_plot, nrow=3)
```



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

[215]:
```
cat('The most probable value is:', mu[which.max(p.norm_Beta)])
```

```
The most probable value is: 0.5
```

[216]:
```
lower_bound <- ncredint(mu, p.norm_Beta, level=0.95)[['lower']]
upper_bound <- ncredint(mu, p.norm_Beta, level=0.95)[['upper']]
cat('The 95% credibility interval is:','[', lower_bound, ',', upper_bound, ']')
```

```
The 95% credibility interval is: [ 0.3355411 , 0.6644589 ]
```

5.3 Repeat the same analysis assuming a sequential analysis of the data 1. 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)

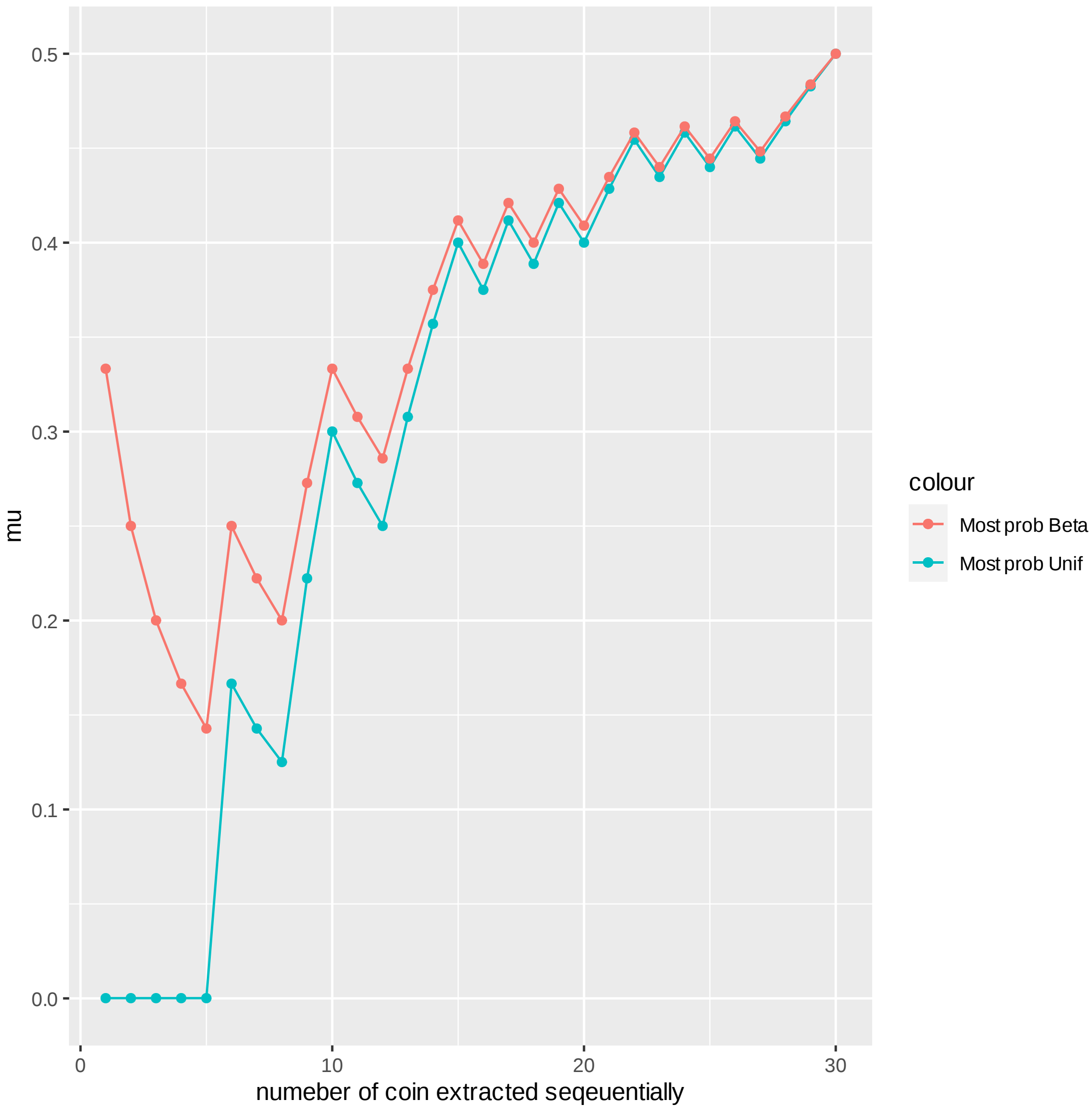
Head H is 1, Tail is 0

[221]:
```
trial = c(0,0,0,0,0,1,0,0,1,1,0,0,1,1,1,0,1,0,1,0,1,1,0,1,0,1,0,1,1,1)
```

[241]:
```
t <- c()

most_prob_Unif <- c()
most_prob_Beta <- c()

t_lower_bound_Unif <- c()
t_upper_bound_Unif <- c()
t_lower_bound_Beta <- c()
t_upper_bound_Beta <- c()

for(i in trial){
    t <- c(t,i)
    n <- length(t)
    y <- sum(t)

    n.sample <- 4001; delta.mu = 1/n.sample
    mu <- seq(from=1/(2*n.sample), by = delta.mu, length.out = n.sample)

    # Uniform prior
    p.star_Unif <- dbinom(x=y, size=n, prob=mu) * dunif(mu, min = 0, max=1)
    p.norm_Unif <- p.star_Unif/(delta.mu * sum(p.star_Unif))
    most_prob_Unif <- c(most_prob_Unif, mu[which.max(p.norm_Unif)])
    t_lower_bound_Unif <- c(t_lower_bound_Unif, ncredint(mu, p.norm_Unif,␣
↪level=0.95)[['lower']] )
    t_upper_bound_Unif <- c(t_upper_bound_Unif, ncredint(mu, p.norm_Unif,␣
↪level=0.95)[['upper']] )

    #Beta prior
    p.star_Beta <- dbinom(x=y, size=n, prob=mu) * dbeta(mu,2, 2)
    p.norm_Beta <- p.star_Beta / (delta.mu * sum(p.star_Beta))
    most_prob_Beta <- c(most_prob_Beta, mu[which.max(p.norm_Beta)])
    t_lower_bound_Beta <- c(t_lower_bound_Beta, ncredint(mu, p.norm_Beta,␣
↪level=0.95)[['lower']] )
    t_upper_bound_Beta <- c(t_upper_bound_Beta, ncredint(mu, p.norm_Beta,␣
↪level=0.95)[['upper']] )
    }
```

[258]:
```
ggplot() +
geom_point(aes(seq(1,30), most_prob_Unif, color='Most prob Unif')) +
geom_line(aes(seq(1,30), most_prob_Unif, color='Most prob Unif')) +
```

```
geom_point(aes(seq(1,30), most_prob_Beta, color='Most prob Beta')) +
geom_line(aes(seq(1,30), most_prob_Beta, color='Most prob Beta')) +
labs(x ='numeber of coin extracted seqeuentially', y='mu')
```



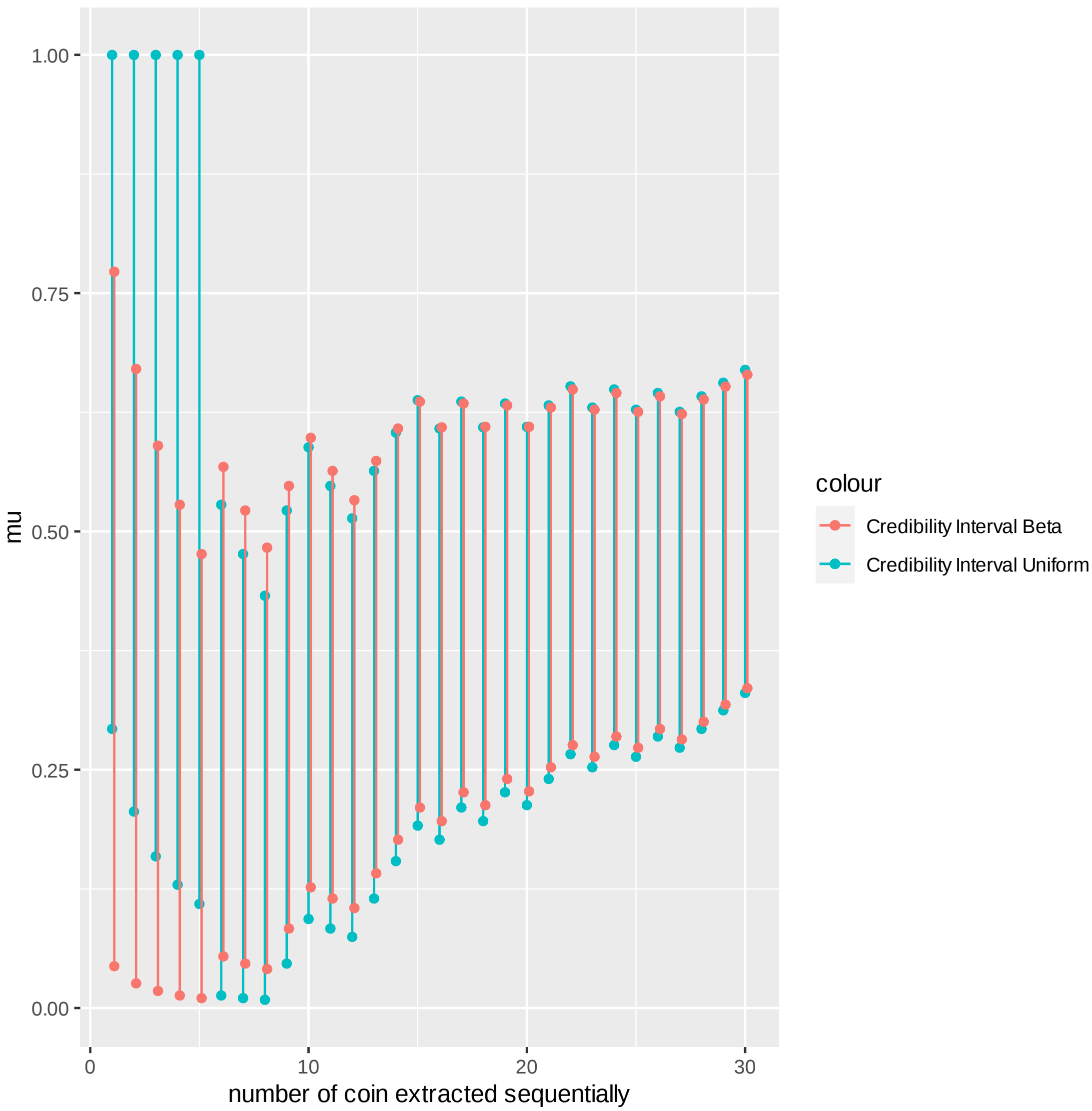
[252]:
```
t_upper_bound_Unif
```

1. 0.99987503124219 2. 0.99987503124219 3. 0.99987503124219 4. 0.99987503124219
5. 0.99987503124219 6. 0.527993001749563 7. 0.476255936015996 8. 0.432516870782304
9. 0.521994501374656 10. 0.588227943014247 11. 0.547738065483629 12. 0.51374656335916
13. 0.563484128967758 14. 0.603724068982754 15. 0.637715571107223 16. 0.607973006748313
17. 0.636215946013497 18. 0.609222694326418 19. 0.634216445888528 20. 0.609722569357661
21. 0.632216945763559 22. 0.652211947013247 23. 0.629967508122969 24. 0.648962759310173
25. 0.627718070482379 26. 0.645213696575856 27. 0.62546863284179 28. 0.641714571357161

29. 0.655961009747563 30. 0.669457635591102

[ ]:
```
t
```

[259]:
```
ggplot() +
geom_segment(aes(x = seq(1,30), y= t_lower_bound_Unif, xend=seq(1,30),␣
↪yend=t_upper_bound_Unif, colour='Credibility Interval Uniform')) +
geom_point(aes(x=seq(1,30), y = t_lower_bound_Unif, colour='Credibility␣
↪Interval Uniform'))+
geom_point(aes(x=seq(1,30), y = t_upper_bound_Unif, colour='Credibility␣
↪Interval Uniform')) +
geom_segment(aes(x = seq(from=1.1, to=30.1, by=1), y= t_lower_bound_Beta,␣
↪xend=seq(from=1.1, to=30.1, by=1), yend=t_upper_bound_Beta,␣
↪colour='Credibility Interval Beta')) +
geom_point(aes(x = seq(from=1.1, to=30.1), y = t_lower_bound_Beta,␣
↪colour='Credibility Interval Beta'))+
geom_point(aes(x = seq(from=1.1, to=30.1), y = t_upper_bound_Beta,␣
↪colour='Credibility Interval Beta'))+
labs(y='mu', x='number of coin extracted sequentially')
```



5.4 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)?

No, in the end the most probable value and the credibility interval are evaluated in the same way when the last coin result is considered

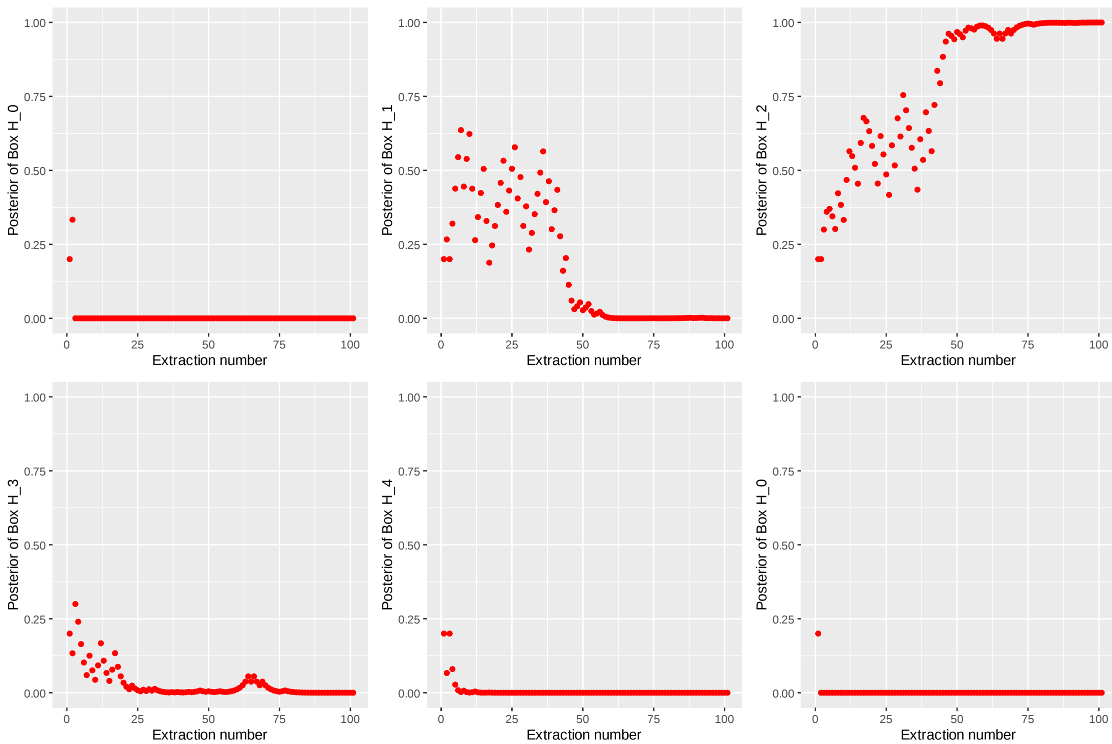
6 Ex.4

[109]:
```
#1 is white
#0 is black
```

```r
#box k
k = sample(seq(0,5,1), 1)
cat('the Box is the :', k,'\n')

#n is number of extraction
n=100
extraction <- rbinom(n, size=1, prob=k/5)
cat('Sequence of balls:', extraction)

#function to obtain the P(E_w|H_j) and p(E_b|H_j)
p.star <- function(i,j){
    if(i==1){
        return(j/5)
    }else{
    return((5-j)/5)
    }
}

#Posterior before extracting the first ball is the flat distribution
P_0 <- c(1/5)
P_1 <- c(1/5)
P_2 <- c(1/5)
P_3 <- c(1/5)
P_4 <- c(1/5)
P_5 <- c(1/5)

P_posterior <- c(rep(1/5, 6))

#Function that for each extraction update the value of the Posterior␣
 ↪distribution
for(i in extraction){
    P_posterior_star <- c()
    for(j in seq(1,6,1)){
        P_posterior_star <- c(P_posterior_star, p.star(i,(j-1))*P_posterior[j])
        }
    P_posterior <- P_posterior_star/(sum(P_posterior_star))

    P_0 <- c(P_0, P_posterior[1])
    P_1 <- c(P_1, P_posterior[2])
    P_2 <- c(P_2, P_posterior[3])
    P_3 <- c(P_3, P_posterior[4])
    P_4 <- c(P_4, P_posterior[5])
    P_5 <- c(P_5, P_posterior[6])
    }

#Plot
```

```r
plot_P_0 <- ggplot() + geom_point(aes(seq(1, n+1), P_0), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_0') + xlim(0, n+1) +
  ylim(0,1)
plot_P_1 <- ggplot() + geom_point(aes(seq(1, n+1), P_1), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_1') + xlim(0, n+1) +
  ylim(0,1)
plot_P_2 <- ggplot() + geom_point(aes(seq(1, n+1), P_2), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_2') + xlim(0, n+1) +
  ylim(0,1)
plot_P_3 <- ggplot() + geom_point(aes(seq(1, n+1), P_3), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_3') + xlim(0, n+1) +
  ylim(0,1)
plot_P_4 <- ggplot() + geom_point(aes(seq(1, n+1), P_4), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_4') + xlim(0, n+1) +
  ylim(0,1)
plot_P_5 <- ggplot() + geom_point(aes(seq(1, n+1), P_5), color='red') +
  labs(x='Extraction number', y='Posterior of Box H_0') + xlim(0, n+1) +
  ylim(0,1)

options(repr.plot.width=12, repr.plot.height=8)
grid.arrange(plot_P_0, plot_P_1, plot_P_2, plot_P_3, plot_P_4, plot_P_5,
  nrow=2, ncol=3)
```

```
the Box is the : 2
Sequence of balls: 0 1 0 0 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0
0 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 0 1 0 0 1 0 0 0
0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1 1 0 1 0 0
```


Posterior of Box H_0
Posterior of Box H_1
Posterior of Box H_2
Posterior of Box H_3
Posterior of Box H_4
Posterior of Box H_0
Extraction number
1.00
0.75
0.50
0.25
0.00
0
25
50
75
100

[ ]: