Assignment 11: Bayes Prior Distributions

Tianzhi, Abhishek, Harshini

2020-11-18

```r
suppressWarnings({

    library(actuar)
    library(tidyverse)

    wholeest<-function(vec0,funcname,prior=c()){
      n <-length(vec0)
      meanv<<-mean(vec0)
      snv=sum((vec0-meanv)^2)
      sumv<<-sum(vec0)
      sqrv=sum(vec0^2)
      ########################################
      # Bernoulli Distribution
      ########################################
      if (funcname=="Bernoulli"){
        cat(sprintf("\n ---- Bernoulli ----\n"))
        funct<-function(x)(x^sumv)*((1-x)^(n-sumv))
        phat<-sumv/n
        print(paste ("phat (MOM)=",phat))
        pval=optimize(funct,c(0,1),tol=0.0001,maximum = TRUE)
        print(paste ("phat (MLE):", pval$maximum))
        p = seq(0,1, length=n)
        hat=ks.test(vec0,"pbinom",1,pval$maximum)$statistic
        print(paste("kstest result:",hat))
        plot(p, dbeta(p, prior[1]+sumv, n-sumv+prior[2]), ylab="density",
             type ="l", main="For Bernoulli Distribution")
      }
      ########################################
      # Geometric Distribution
      ########################################
      if (funcname=="Geometric"){
        cat(sprintf("\n ---- Geometric ----\n"))
        funct<-function(x)(n*log2(x))+(sumv*log2(1-x))
        pval=optimize(funct,c(0,1),tol=0.0001,maximum = TRUE)
        print(paste ("phat (MLE):", pval$maximum))
        phat=1/((sumv/n)+1)
        print (paste("phat(MOM)=",phat))
        hat=ks.test(vec0,"pgeom",pval$maximum)$statistic
        print(paste("kstest result:",hat))
        p = seq(0,1, length=n)
        plot(p, dbeta(p, prior[1]+n, sumv+prior[2]-1), ylab="density",
             type ="l", main="For Geometric Distribution")
```

```
}
##########################################
# Normal Distribution
##########################################
if (funcname=="Normal"){
  cat(sprintf("\n ---- Normal ----\n"))
  mu=sumv/n
  print (paste("mu (MLE):",mu))
  samu=sum((vec0-mu)^2)
  funct<-function(x)-(n/2)*(log(2*pi*x^2)) + (-1/(2*x^2)) *samu
  pval=optimize(funct,c(0,1000),tol=0.001,maximum = TRUE)
  print(paste ("sigma (MLE):", pval$maximum))
  mu=sumv/n
  sig2=snv/n
  print(paste("mu(MOM)=",mu,"sigmasquare(MOM)=",sig2))
  hat=ks.test(vec0,"pnorm",mu,pval$maximum)$statistic
  print(paste("kstest result:",hat))

  # x = seq(-5,10, length=n)
  # odsesq=1/(pval$maximum)^2
  # w=odsesq/(odsesq+(1/prior[2]^2))
  # plot(x, dnorm(x,w*sumv/n+(1-w)*prior[1],1/(odsesq+(1/prior[2]^2))),
  # ylab="density", type ="l")

  # Consider a normal sample of 1000 data points
  # sample <- rnorm(n = 1000, mean = 14, sd = 20)
  sample <- vec0

  r <- 1
  tau <- 6
  mu <- 4
  mean_sample <- mean(sample)

  # Let's assume alpha and beta for the prior distribution to be 1 and 3
  prior_alpha <- 1
  prior_beta <- 3

  # Now compute the posterior distribution parameters
  conditional_distribution_mean <- (tau*mu + n*mean_sample) / (tau + n)
  conditional_distribution_precision <- (tau + n) * r
  print(paste("The posterior normal distribution parameters (mean, precision): ",
      c(conditional_distribution_mean, conditional_distribution_precision)))

  marginal_distribution_alpha <- prior_alpha + n/2
  marginal_distribution_beta <- 1 / ((1/prior_beta)
                                  + 1/2*(sum((sample - mean_sample)**2)))
                             + tau*n*((mean_sample - mu)**2)/2*(tau + n)

  print(paste("The marginal posterior gamma distribution parameters are (alpha, beta): ",
            c(marginal_distribution_alpha, marginal_distribution_beta)))

  # Now using the posterior parameters, let's plot the density of 10K data points
  conditional_joint_distribution <- rnorm(n = 10000,
```

```
                                        mean = conditional_distribution_mean,
                                        1/sqrt(conditional_distribution_precision))
  marginal_joint_distribution <- rgamma(n = 10000,
                                      marginal_distribution_alpha,
                                      marginal_distribution_beta)

  plot(density(conditional_joint_distribution),
     main = "Conditional Joint Probability Distribution
     for Mean M for Normal Distribution")

  plot(density(marginal_joint_distribution),
     main = "Marginal Joint Probability Distribution
     for Precision R for Normal Distribution")
}
########################################
# Binomial Distribution
########################################
if (funcname=="Binomial"){
  cat(sprintf("\n ---- Binomial ----\n"))
  phat=1-(snv/sumv)
  nhat=sumv/(n*phat)
  print(paste("p(MOM)=",phat,"n(MOM)=",nhat))

  nval = (1.0/p_true)*mean(vec0)
  pval=sumv / (length(vec0)*nval)
  print(paste ("p (MLE):", pval))

  theta_hat_func <- function(data) {
    n1 <- length(data)
    estimated_p <- (1 / n1) * (sum(data)/n1)
    return(estimated_p)
  }

  theta_hat <- theta_hat_func(vec0)
  print(paste("kstest result:",theta_hat))

  # Consider a binomial sample of 1000 data points
  sample <- vec0

  # Let's assume alpha and beta for the prior distribution to be 1
  prior_alpha <- 1
  prior_beta <- 1
  r <- 1

  # Now compute the posterior distribution parameters
  posterior_alpha <- prior_alpha + sum(sample)
  posterior_beta <- prior_beta + r * n - sum(sample)

  print(paste("The posterior beta distribution parameters are: ",
            c(posterior_alpha, posterior_beta)))

  # Now using this posterior_alpha and posterior_beta plot the density
  # of 10K data points
```

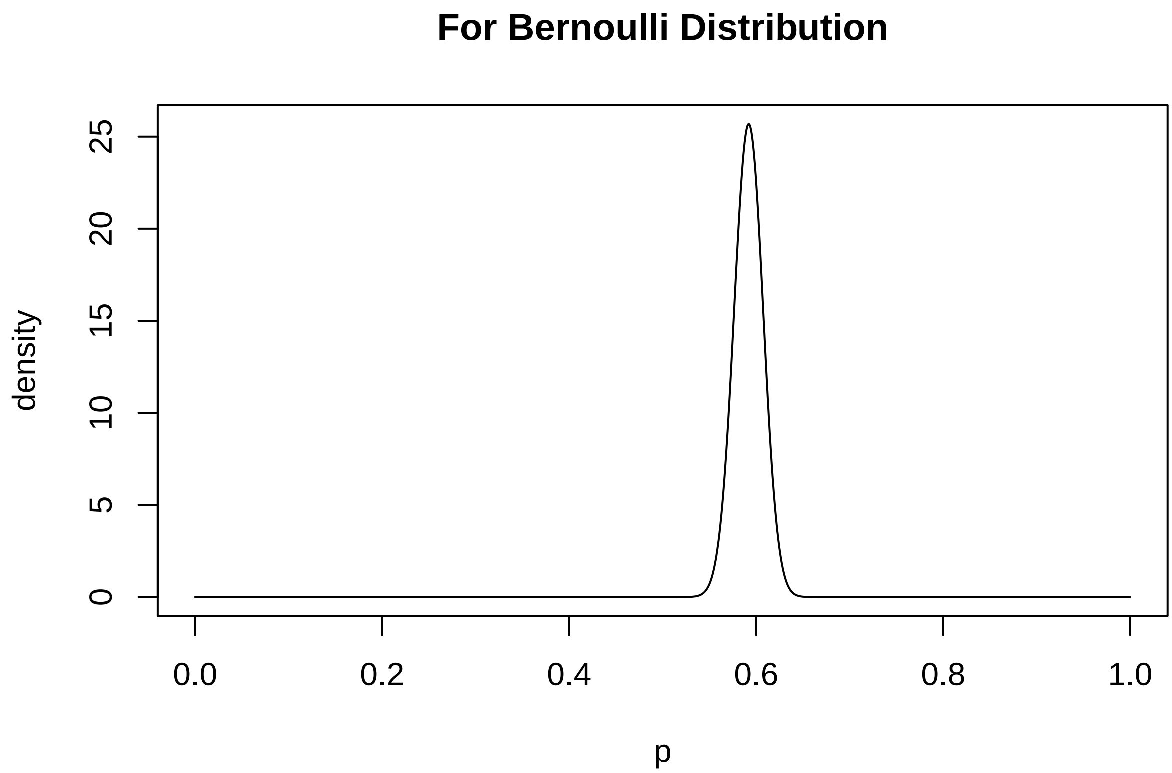
```r
  posterior_distribution_sample <- rbeta(n = 10000,
                                          shape1 = posterior_alpha,
                                          shape2 = posterior_beta)

  plot(density(posterior_distribution_sample),
       main = "Posterior Beta Distribution for Binomial Distribution")
}
########################################
# Uniform Distribution
########################################
if (funcname=="Uniform"){
  cat(sprintf("\n ---- Uniform ----\n"))
  a=sumv/n-(sqrt(3*snv/n))
  b=sumv/n+(sqrt(3*snv/n))
  print(paste("a (MOM)=",a,"b (MOM)=",b))


  theta_hat_func <- function(data) {
    estimated_a <- min(data)
    estimated_b <- max(data)
    return (c(estimated_a, estimated_b))
  }

  theta_hat <- theta_hat_func(vec0)
  print(paste("a (MLE)=",theta_hat[1],"b (MLE)=",theta_hat[2]))

  nboot <- 1000

  q_hat <- qunif(c(1:n)/(n+1), theta_hat[1], theta_hat[2])

  D0 <- ks.test(vec0, q_hat)$statistic
  D_vec<-NULL

  for(i in 1:nboot){
    x_star <- runif(n, theta_hat[1], theta_hat[2])
    theta_hat_star <- theta_hat_func(x_star)

    q_hat_star <- qunif(c(1:n)/(n+1), theta_hat_star[1], theta_hat_star[2])
    D_star <- ks.test(x_star, q_hat_star)$statistic
    D_vec <- c(D_vec, D_star)
  }
  p_value <- sum(D_vec > D0)/nboot
  print(paste("kstest result: The p-value is",p_value))

  # Consider a uniform sample of 1000 data points
  sample <- vec0

  # Let's assume alpha and W0 for the prior distribution to be 1
  prior_w0 <- 1
  prior_alpha <- 1

  # Now compute the posterior distribution parameters
  posterior_w0 <- max(c(prior_w0, sample))
```

```r
    posterior_alpha <- prior_alpha + n

    print(paste("The posterior pareto distribution parameters are: ",
                c(posterior_w0, posterior_alpha)))

    # Now using the posterior_w0 and posterior_alpha, plot the density
    # of 10K data points
    posterior_distribution_sample <- rpareto(n = 10000,
                                            posterior_w0,
                                            posterior_alpha)

    plot(density(posterior_distribution_sample),
         main = "Posterior Pareto Distribution for Uniform Distribution")
  }
  ########################################
  # Poisson Distribution
  ########################################
  if (funcname=="Poisson"){
    cat(sprintf("\n ---- Poisson ----\n"))
    lambdahat=sumv/n
    print(paste("lambda (MOM)=",lambdahat))

    theta_hat = meanv
    print(paste("lambda (MLE)=",theta_hat))

    q_hat <- qpois(c(1:n)/(n+1), theta_hat)
    D0 <- ks.test(vec0, q_hat)$statistic

    D_vec<-NULL

    nboot = 1000
    for(i in 1 : nboot){
      x_star <- rpois(n, theta_hat)
      theta_hat_star <- mean(x_star)

      q_hat_star <- qpois(c(1:n)/(n+1), theta_hat_star)
      D_star <- ks.test(x_star, q_hat_star)$statistic
      D_vec <- c(D_vec, D_star)
    }

    p_value <- sum(D_vec > D0)/nboot
    print(paste("kstest result: The p-value is",p_value))

    sample <- vec0

    # Let's assume alpha and beta for the prior distribution to be 1
    prior_alpha <- 1
    prior_beta <- 1

    # Now compute the posterior distribution parameters
    posterior_alpha <- prior_alpha + sum(sample)
    posterior_beta <- 1/(1/prior_beta + n)
```

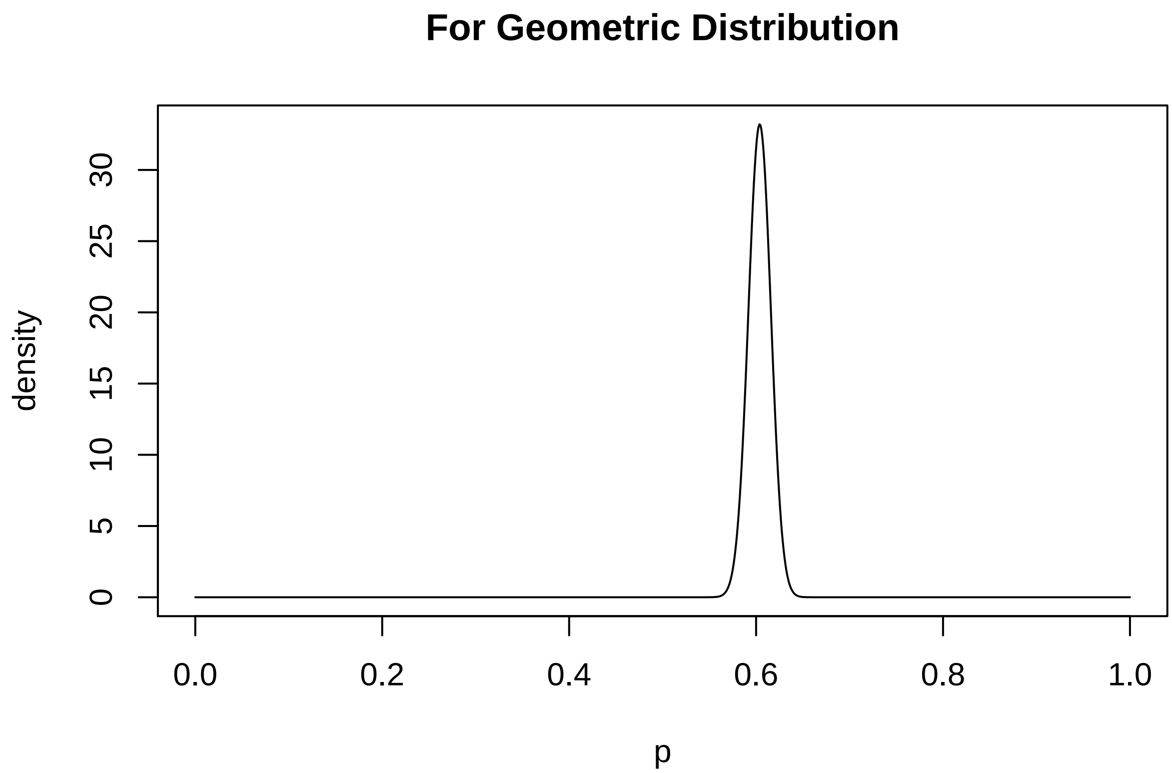
```r
    print(paste("The posterior gamma distribution parameters are: ",
                c(posterior_alpha, posterior_beta)))

    # Now using the posterior_alpha and posterior_beta, plot the density
    # of 10K data points
    posterior_distribution_sample <- rgamma(n = 10000, posterior_alpha, posterior_beta)

    plot(density(posterior_distribution_sample),
         main = "Posterior Gamma Distribution for Poisson Distribution")
  }
  ########################################
  # Exponential Distribution
  ########################################
  if (funcname=="Exponential"){
    cat(sprintf("\n ---- Exponential ----\n"))

    beta=sumv/n
    print(paste("beta (MOM)=",beta))

    theta_hat_func <- function(data) {
      estimated_theta <- mean(data)
      return(estimated_theta)
    }
    theta_hat <- theta_hat_func(vec0)
    print(paste("beta (MLE)=", theta_hat))

    nboot <- 1000
    q_hat <- qexp(c(1:n) / (n+1), beta)

    D_0 <- ks.test(vec0, q_hat)$statistic
    D_vec<-NULL

    for(i in 1:nboot){
      x_star <- rexp(n, beta)
      theta_hat_star <- theta_hat_func(x_star)

      q_hat_star <- qexp(c(1:n)/(n+1), theta_hat_star)
      D_star <- ks.test(x_star, q_hat_star)$statistic

      D_vec <- c(D_vec, D_star)
    }
    p_value <- sum(D_vec > D_0)/nboot
    print(paste("kstest result: The p-value is",p_value))

    # Consider a normal sample of 1000 data points
    sample <- vec0

    # Let's assume alpha and beta for the prior distribution to be 1
    prior_alpha <- 1
    prior_beta <- 1

    # Now compute the posterior distribution parameters
    posterior_alpha <- prior_alpha + n
```

```r
        posterior_beta <- 1/(1/prior_beta + sum(sample))

        print(paste("The posterior gamma distribution parameters are:",
                    c(posterior_alpha, posterior_beta)))

        # Now using the posterior parameters, let's plot the density of 10K data points
        posterior_distribution_sample <- rgamma(n = 10000, posterior_alpha, posterior_beta)
        plot(density(posterior_distribution_sample),
             main = "Posterior Gamma Distribution for Exponential Distribution")
      }
    }

    wholeest(rbinom(1000,1,0.6),"Bernoulli",c(1,1))
    wholeest(rgeom(1000,0.6),"Geometric",c(1,1))
    wholeest(rnorm(1000,3,5),"Normal",c(5,15))

    n_true<-1000
    p_true<-0.76
    wholeest(rbinom(n_true, 1, p_true),"Binomial")

    wholeest(runif(1000, 0, 15), "Uniform")
    wholeest(rpois(1000,4),"Poisson")
    wholeest(rexp(1000,8),"Exponential")

})
```

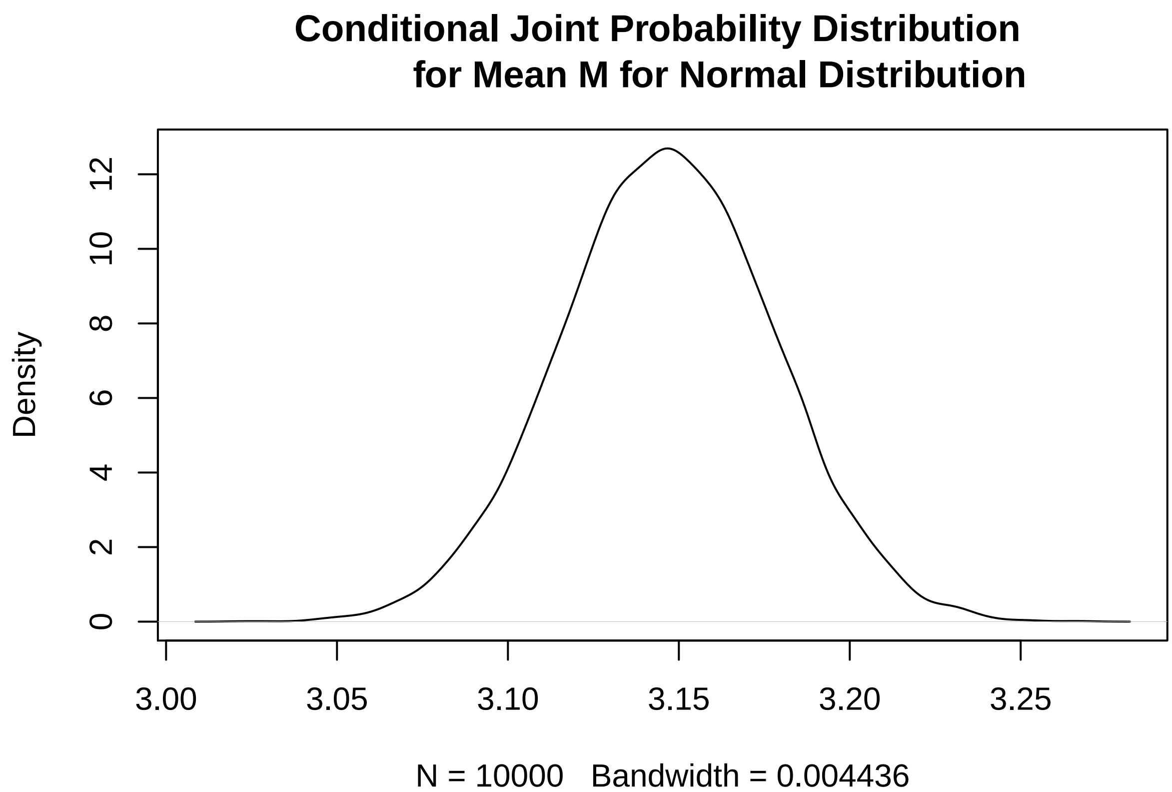
```
##
## Attaching package: 'actuar'

## The following object is masked from 'package:grDevices':
##
##     cm

## -- Attaching packages --------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

##
##  ---- Bernoulli ----
## [1] "phat (MOM)= 0.592"
## [1] "phat (MLE): 0.591994411491554"
## [1] "kstest result: 0.592"
```

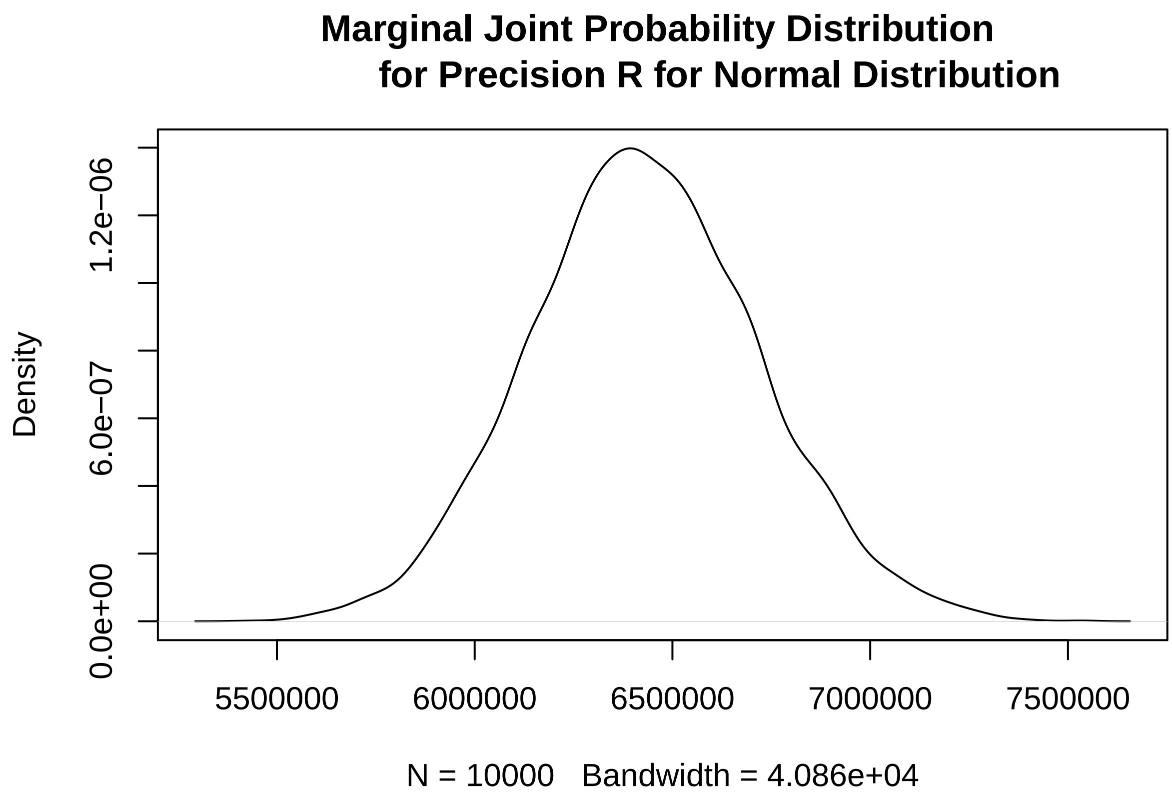



```
##
##  ---- Geometric ----
## [1] "phat (MLE): 0.603499416969091"
## [1] "phat(MOM)= 0.603500301750151"
## [1] "kstest result: 0.603499416969091"
```

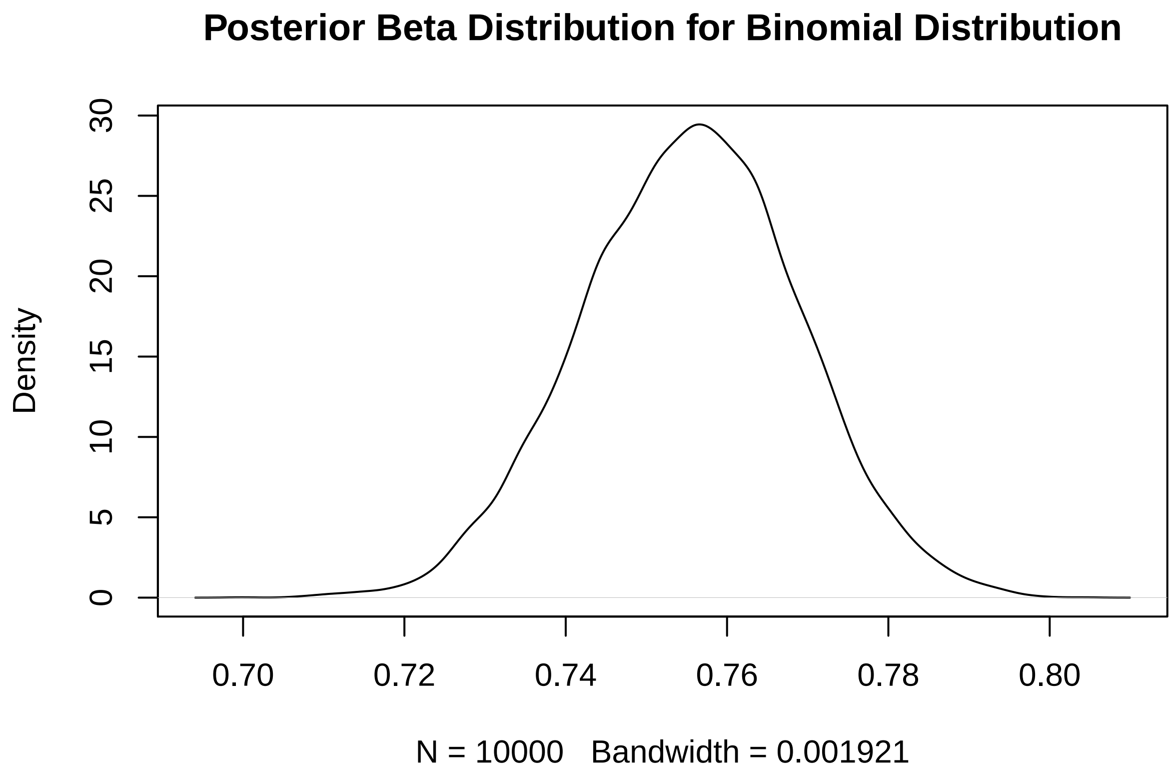



```
##
##  ---- Normal ----
## [1] "mu (MLE): 3.14207941950593"
## [1] "sigma (MLE): 5.06722078024272"
## [1] "mu(MOM)= 3.14207941950593 sigmasquare(MOM)= 25.6763684876148"
## [1] "kstest result: 0.0225458129442648"
## [1] "The posterior normal distribution parameters (mean, precision):  3.1471962420536"
## [2] "The posterior normal distribution parameters (mean, precision):  1006"
## [1] "The marginal posterior gamma distribution parameters are (alpha, beta):  501"
## [2] "The marginal posterior gamma distribution parameters are (alpha, beta):  7.78906126810561e-05"
```

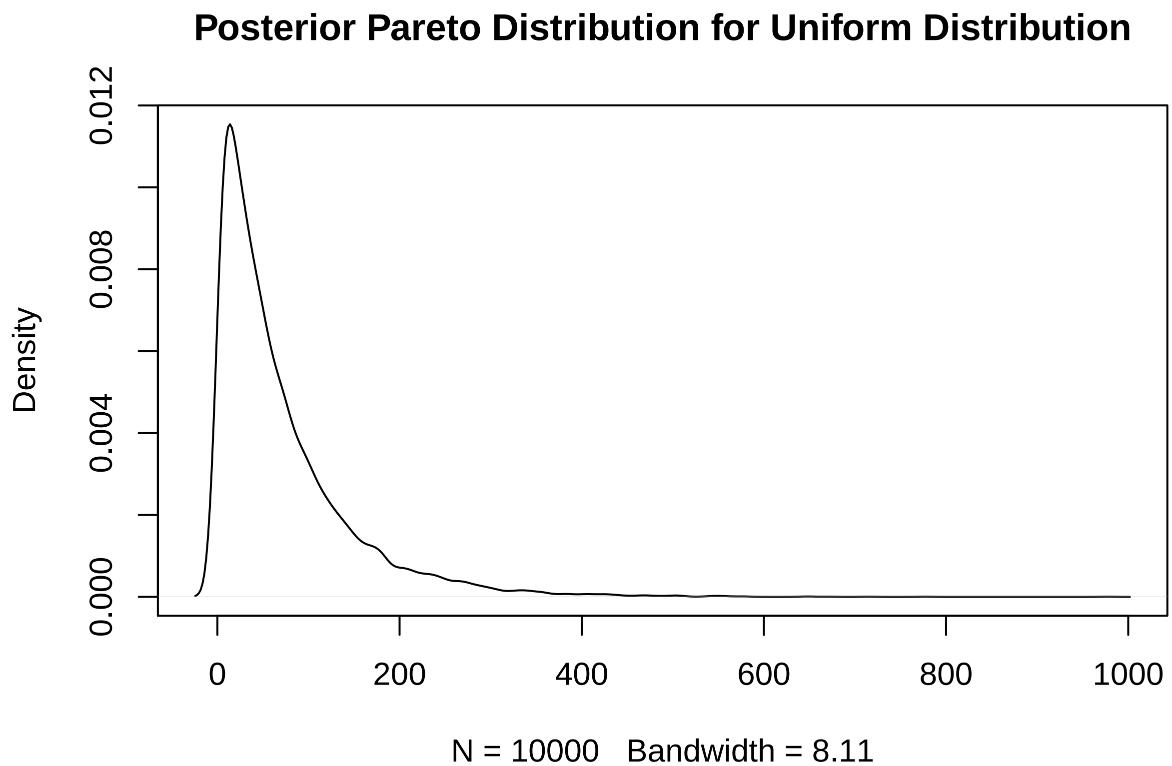

Conditional Joint Probability Distribution
for Mean M for Normal Distribution
Density
0
2
4
6
8
10
12
3.00
3.05
3.10
3.15
3.20
3.25
N = 10000   Bandwidth = 0.004436



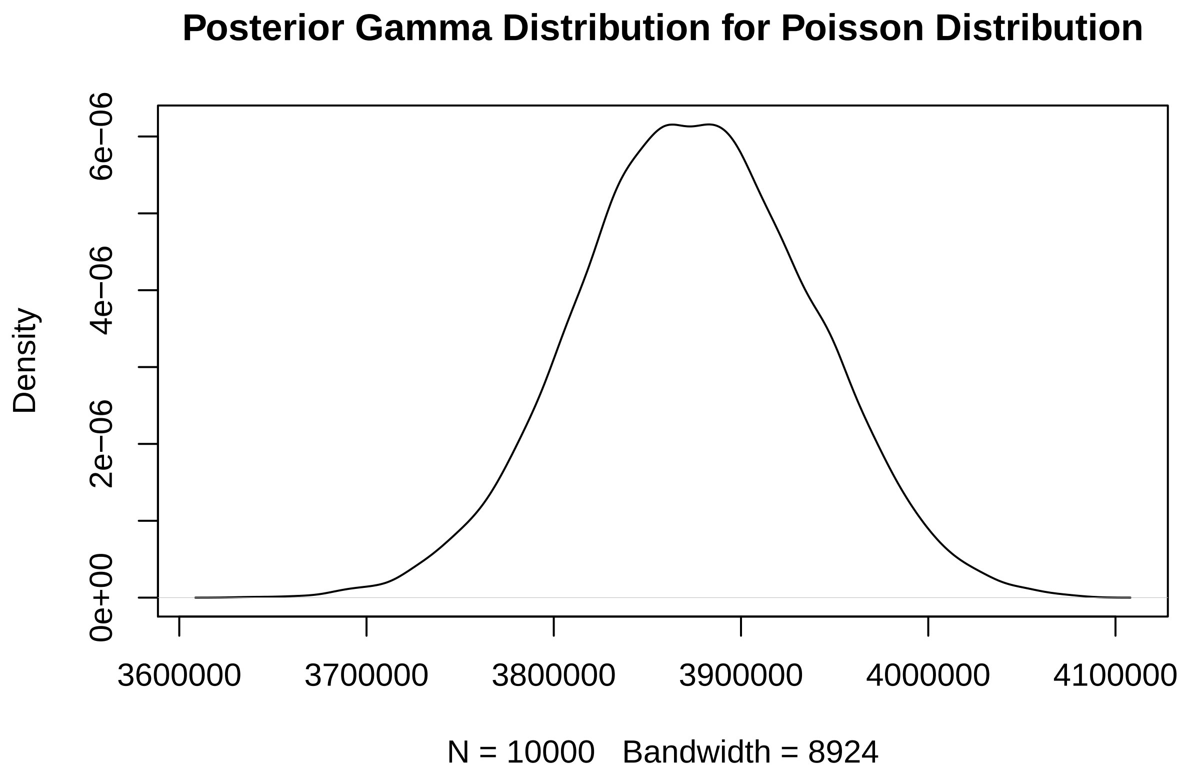

```
##
##  ---- Binomial ----
## [1] "p(MOM)= 0.756 n(MOM)= 1"
## [1] "p (MLE): 0.76"
## [1] "kstest result: 0.000756"
## [1] "The posterior beta distribution parameters are:  757"
## [2] "The posterior beta distribution parameters are:  245"
```



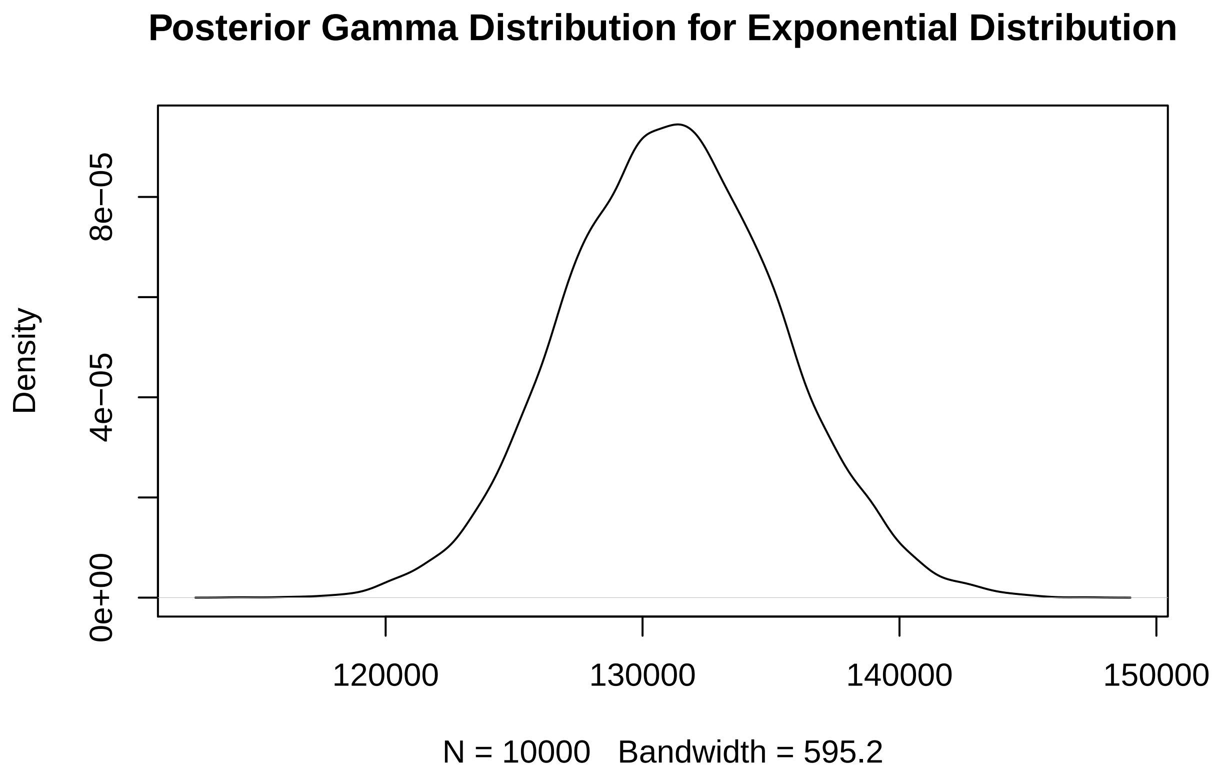

```
##
##  ---- Uniform ----
## [1] "a (MOM)= 0.362370456091329 b (MOM)= 15.0829414918765"
## [1] "a (MLE)= 0.0477080466225743 b (MLE)= 14.9909275991376"
## [1] "kstest result: The p-value is 0.087"
## [1] "The posterior pareto distribution parameters are:  14.9909275991376"
## [2] "The posterior pareto distribution parameters are:  1001"
```




```
##
##  ---- Poisson ----
## [1] "lambda (MOM)= 3.871"
## [1] "lambda (MLE)= 3.871"
## [1] "kstest result: The p-value is 0.649"
## [1] "The posterior gamma distribution parameters are:  3872"
## [2] "The posterior gamma distribution parameters are:  0.000999000999000999"
```




```
##
##  ---- Exponential ----
## [1] "beta (MOM)= 0.130028004109076"
## [1] "beta (MLE)= 0.130028004109076"
## [1] "kstest result: The p-value is 0.399"
## [1] "The posterior gamma distribution parameters are: 1001"
## [2] "The posterior gamma distribution parameters are: 0.00763195628903526"
```


Posterior Gamma Distribution for Exponential Distribution
Density
0e+00
4e−05
8e−05
120000
130000
140000
150000
N = 10000 Bandwidth = 595.2