Assignment 1

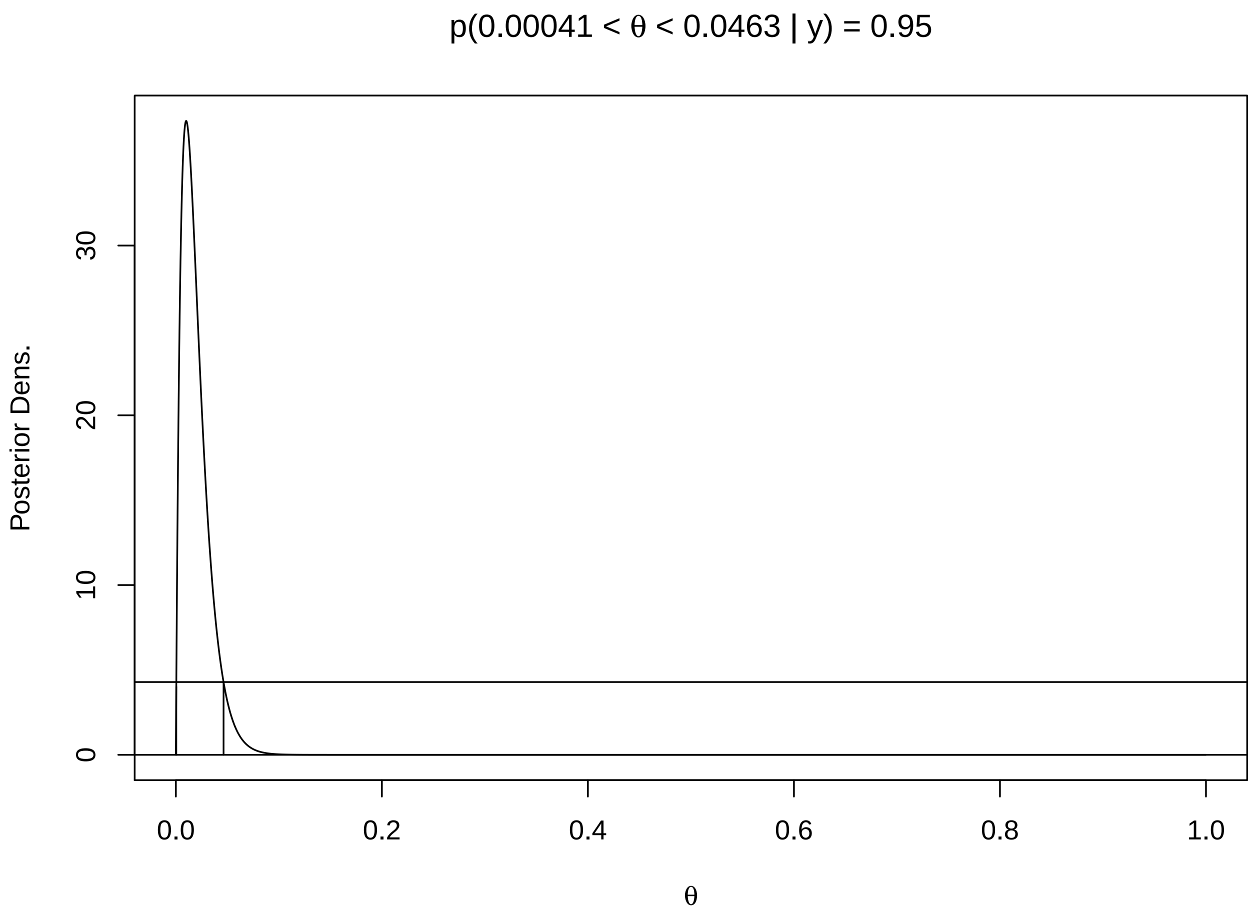
2017-04-09

Question 1: Inference for the binomial parameter:

(a) Develop an R function to calculate HPD intervals for data (x, n), given a *beta(a,b)* prior. I actually developed two functions. My initial attempts were based around the idea that you could optimise for some height 'h'. However, when it came to actually implementing this method in future questions I ran into trouble. So, I instead opted to try and optimise the length of the interval between the upper and lower limits, which produced results that are useable in parts (b)-(e). Here is my first attempt.

```r
solve.HPD.beta = function(shape1, shape2, credint = 0.95, plot=FALSE, ...){

  hpdfunc <- function(h, shape1, shape2){
    mode = (shape1 - 1)/(shape1 + shape2 - 2)
    lt = uniroot(f=function(x){ dbeta(x,shape1, shape2) - h},
                 lower=0, upper=mode)$root
    ut = uniroot(f=function(x){ dbeta(x,shape1, shape2) - h},
                 lower=mode, upper=1)$root
    coverage = pbeta(ut, shape1, shape2) - pbeta(lt, shape1, shape2)

    hpdval = abs(credint-coverage)

    results <<- data_frame("Lower"    = lt,
                           "Upper"    = ut,
                           "Coverage" = coverage,
                           "Height"   =h)
    return(hpdval)
  }
  upper = max(dbeta(seq(0,1, by = 0.001), shape1, shape2))
  h = optimize(hpdfunc,
               interval = seq(0,upper,by = 0.001),
               lower = 0,
               tol = .Machine$double.eps,
               shape1,
               shape2)

  if (plot) {
    th = seq(0, 1, length=10000)
    plot(th, dbeta(th, shape1, shape2),
         t="l",xlab=expression(theta),
         ylab="Posterior Dens.")
    abline(h=h)
    segments(results[[2]],0,results[[2]],dbeta(results[[2]],shape1,shape2))
    segments(results[[1]],0,results[[1]],dbeta(results[[1]],shape1,shape2))
    title(bquote(paste("p(", .(round(results[[1]], 5))," < ", theta, " < ",
                       .(round(results[[2]],5)), " | " , y, ") = ",
                       .(round(results[[3]], 5)))))
  }
  return(results)}

y=1; n=100; a=1; b=1; p=0.95
```

```r
solve.HPD.beta(shape1 = y + a,
               shape2 = n - y + b,
               plot = TRUE)
```



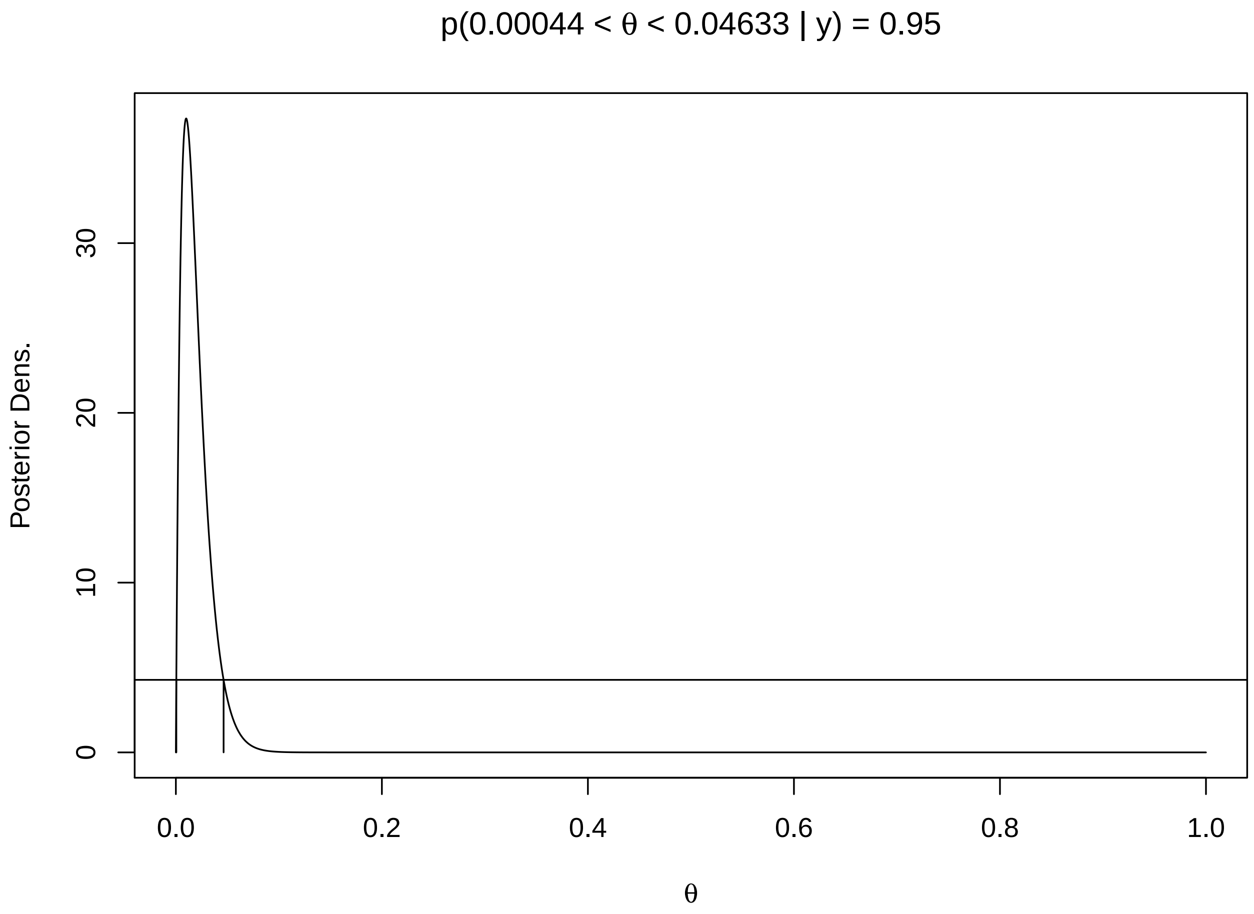
```
## # A tibble: 1 x 4
##          Lower       Upper Coverage    Height
##          <dbl>       <dbl>    <dbl>     <dbl>
## 1 0.00041443751 0.046302914     0.95 4.2861667
```

But - as I said - this didn't work well for the method I applied to the rest of the questions. So I came up with this instead.

```r
HDIofqbeta = function(shape1,
                      shape2,
                      credint = 0.95,
                      tol = 1e-8,
                      plot = FALSE,
                      one.sided = FALSE) {

  if(one.sided){  # returns the one-sided interval if required.
    return(qbeta(credint,
                 shape1,
                 shape2))
  } else {
```

```r
# a function to calculate the interval for specific values of the upper and
# lower limits

interval = function(lowint,  # for 95%: 0 <= lowint <=0.05
                    credint, # e.g. 95%
                    shape1,  # i.e: y - a
                    shape2   # i.e: n - y + b
                    ){

  hig = qbeta(credint + lowint, # the upper limit
              shape1,
              shape2)

  low = qbeta(lowint, # the lower limit.
              shape1,
              shape2)

  hig - low # returns the interval.
}

alpha    = 1 - credint # an upper limit for the interval optimiser.
length  = optimize(interval,
                   c(0, alpha),
                   shape1 = shape1,
                   shape2 = shape2,
                   credint = 0.95,
                   tol = 1e-8)

HDIinterval = length$minimum # the actual optimised value from 0 to
                             # the lower limit.

lt = qbeta(HDIinterval, shape1, shape2)
ut = qbeta(HDIinterval + credint, shape1, shape2)
coverage = credint
l = HDIinterval

results <<- data_frame("Lower"    = lt,
                       "Upper"    = ut,
                       "Coverage" = coverage,
                       "Length"   = l)

if (plot) {
  th = seq(0, 1, length=10000)
  plot(th, dbeta(th, shape1, shape2),
       t="l",xlab=expression(theta),
       ylab="Posterior Dens.")
  abline(h = dbeta(results[[1]],shape1,shape2))
  segments(results[[2]],0,results[[2]],dbeta(results[[2]],shape1,shape2))
  segments(results[[1]],0,results[[1]],dbeta(results[[1]],shape1,shape2))
  title(bquote(paste("p(", .(round(results[[1]], 5))," < ", theta, " < ",
                     .(round(results[[2]],5)), " | " , y, ") = ",
                     .(round(results[[3]], 5)))))
}
```

```
    return(results)
  }
}

y <- 1; n <- 100; a <- 1; b <- 1; p <- 0.95
HDIofqbeta(shape1 = y + a, shape2 = n - y + b, plot = TRUE)
```

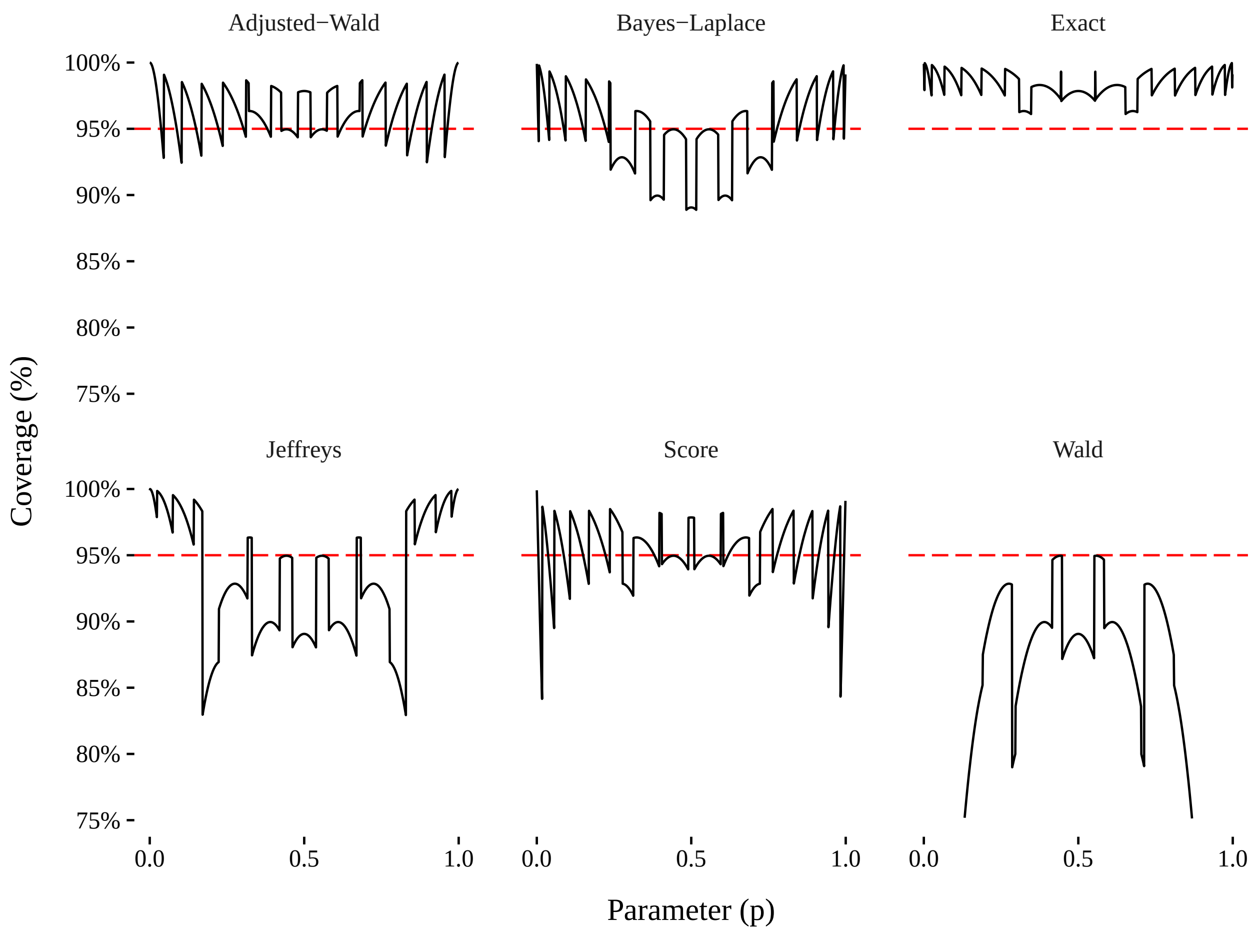


```
## # A tibble: 1 x 4
##           Lower       Upper Coverage        Length
##           <dbl>       <dbl>    <dbl>         <dbl>
## 1 0.00044187206 0.046329504     0.95 0.00095772305
```

Both of these return approximately the same results

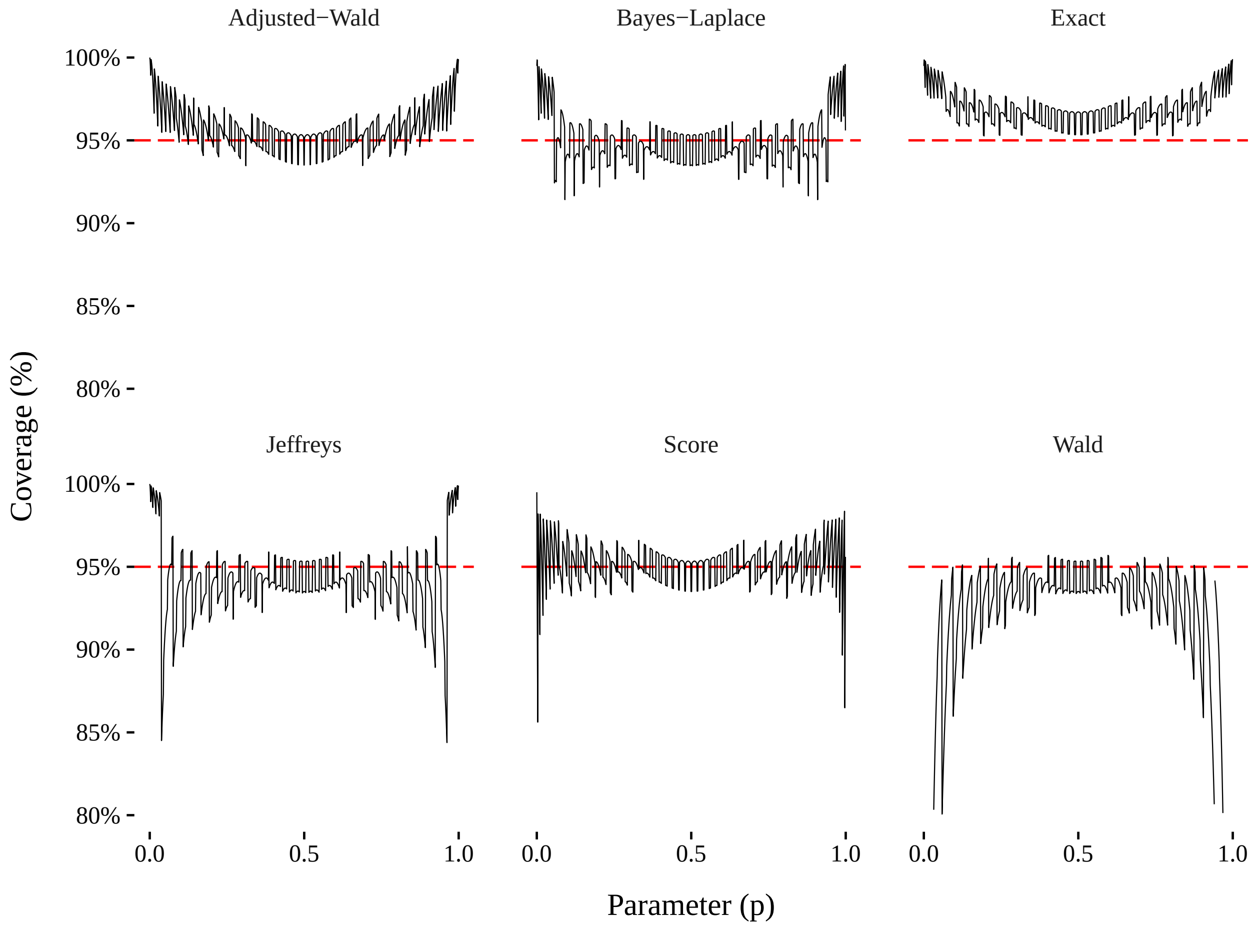
(b) Reproduce Agresti & Coull's (1998) Figure 4 ($n = 10$), and replicate for the Score and Bayes-Laplace & Jeffreys HPD intervals The code for generating the intervals in questions (b) - (e) is given below.

```
n <- 10
a <- 0.05
p <- seq(0.0001,0.9999,1/1000)
z <- abs(qnorm(.5*a,0,1))
source("Question1/1bData.R")
```



(c) Repeat (b) for $n = 50$.

```r
n <- 50
a <- 0.05
p <- seq(0.0001,0.9999,1/1000)
z <- abs(qnorm(.5*a,0,1))
source("Question1/1cData.R")
```



(d) Compare the minimum coverage of the six graphs at (c)

covinterval	mean(dens)	median(dens)	min(dens)
Adjusted-Wald	0.95799768	0.95565926	0.93468072
Bayes-Laplace	0.95003874	0.94690300	0.91417115
Exact	0.96932051	0.96776141	0.95266397
Jeffreys	0.94189934	0.94089149	0.84383600
Score	0.95188054	0.95301352	0.85620902
Wald	0.90051289	0.93485093	0.00498777

(e) The adjusted Wald interval appears to perform well with respect to frequentist coverage, if close to nominal combined with reasonable minimum coverage is aimed for. From a Bayesian point of view, performance of individual intervals is just as, if not more, important. Given $x = 0$, compare the adjusted Wald interval with the exact & Score intervals (all two-sided), and with the Bayes-Laplace & Jeffreys HPD intervals, for a range of values of n and α, comment on its limitations, and give an appropriate graphical illustration.

Question 2: Inference for the Cauchy parameter:

(a) Develop an R function to find percentiles of a (general) Cauchy posterior as discussed by Jaynes (1976, Example 6) and Box & Tiao (1973, p.64), to be used for the examples below.

```r
source("Question2/2a.R")
CauchyPercentage <- function(x, # a vector of samples
                             p, # a vector of possible parameters
                             y = NULL # a value to test Pr[p < y]
) {
  cauchydens <- function(x,p){

    cauchydist <- function(x,p) {
      H = vector(mode = "numeric",length = length(x))
      for(i in 1:length(x)){
        H[i] = (1+(x[i] - p)^2)^(-1)
      }
      return(prod(H))
    }
    dens = vapply(p,cauchydist,min(p), x = x)
    c = (integrate(dens, -Inf, Inf)$value)^(-1)
    data.frame(vparams = p, postdens = c*dens(p))
  }

  df = cauchydens(x,p)

  yind = ifelse(length(which(df$vparams == y))==1,
                which(df$vparams == y),
                max(which(df$vparams < y)))

  plot(df$vparams, df$postdens, type = 'l')
  abline(v = df$vparams[yind])
  abline(h = df[yind,"postdens"])

  cumdist = c*integrate(dens,-Inf,y)$value

  return((c(yind,df[yind,"postdens"],cumdist)))
}

CauchyHPD <- function(x, # vector of samples
                      p, # vector of possible parameter values
                      alpha = 0.95, # HPD interval value
                      tol = 0.0001) { # level of tolerance for exact HPD interval

  cauchydens <- function(x,p){

    cauchydist <- function(x,p) {
      H = vector(mode = "numeric",length = length(x))
      for(i in 1:length(x)){
        H[i] = (1+(x[i] - p)^2)^(-1)
      }
      return(prod(H))
    }
    dens = function(p) vapply(p,cauchydist,min(p), x = x)
    c = (integrate(dens, -Inf, Inf)$value)^(-1)
    data.frame(vparams = p, postdens = c*dens(p))
  }

  df = cauchydens(x,p)

  cumdist = cumsum(df$postdens)*diff(df$vparams)[1]
  post_median = which.min(abs(cumdist-0.5))

  HPDlimits <- function(post_dens) { ## find lower and upper values for which
    ## prob dens is closest to target value
    lower = which.min(abs(df$postdens[1:post_median]-post_dens))
    upper = which.min(abs(df$postdens[(post_median+1):length(df$postdens)]-post_dens))+post_median
    limits = c(lower,upper)
  }

  HPDlimitarea <- function(post_dens) {
    limitints = HPDlimits(post_dens)
    limitarea = sum(df$postdens[limitints[1]:limitints[2]])*diff(df$vparams)[1]
  }
  ## find credible interval
  v2 = seq(0,max(df$postdens),by=tol)
  vals = sapply(v2,HPDlimitarea)
  w = which.min(abs(vals-alpha))
  r = c(df$vparams[HPDlimits(v2[w])])
  names(r) = c("lower","upper")
  par(mfrow = c(1,2))
  plot(df$vparams, cumdist, type = 'l')
  abline(h = 0.5)
  abline(v = df$vparams[post_median], col = 'red')
  abline(v = r["upper"], col = 'blue')
  abline(v = r["lower"], col = 'blue')
  plot(df$vparams, df$postdens, type = 'l')
  abline(v = df$vparams[post_median], col = 'red')
  abline(h = df[HPDlimits(v2[w])[1],"postdens"])
  abline(v = r["upper"], col = 'blue')
  abline(v = r["lower"], col = 'blue')
  return(r)
}
```

(b) Consider Jaynes' example of $n = 2$ observations $(3, 5)$: plot the posterior and calculate the 90% central credible interval. Explain why it is quite different from the confidence interval derived by Jaynes (p.202).

```
source("Question2/2b.R")
```

(c) Consider Box & Tiao's example of $n = 5$ observations $(11.4, 7.3, 9.8, 13.7, 10.6)$: plot the posterior and calculate 95% central and HPD credible intervals and check $Pr[\theta < 11.5]$ given by Box & Tiao.

```
source("Question2/2c.R")
```

(d) Consider Berger's (1985, p.141) example of $n = 5$ observations $(4.0, 5.5, 7.5, 4.5, 3.0)$: calculate 95% central and HPD credible intervals, with and without Berger's restriction $(\theta > 0)$.

```
source("Question2/2d.R")
```

(e) Clearly, Berger's restriction $(\theta > 0)$ will sometimes lead to a posterior quite different from the unrestricted posterior. Plot this restricted posterior for the hypothetical negative version of Berger's example: i.e. $(-4.0, -5.5, -7.5, -4.5, -3.0)$, and calculate the 95% HPD interval.

```r
source("Question2/2e.R")
```