Assignment 1

2017-04-11

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. This function will optimise the height 'h' on the y-axis in the manner that I described in class.

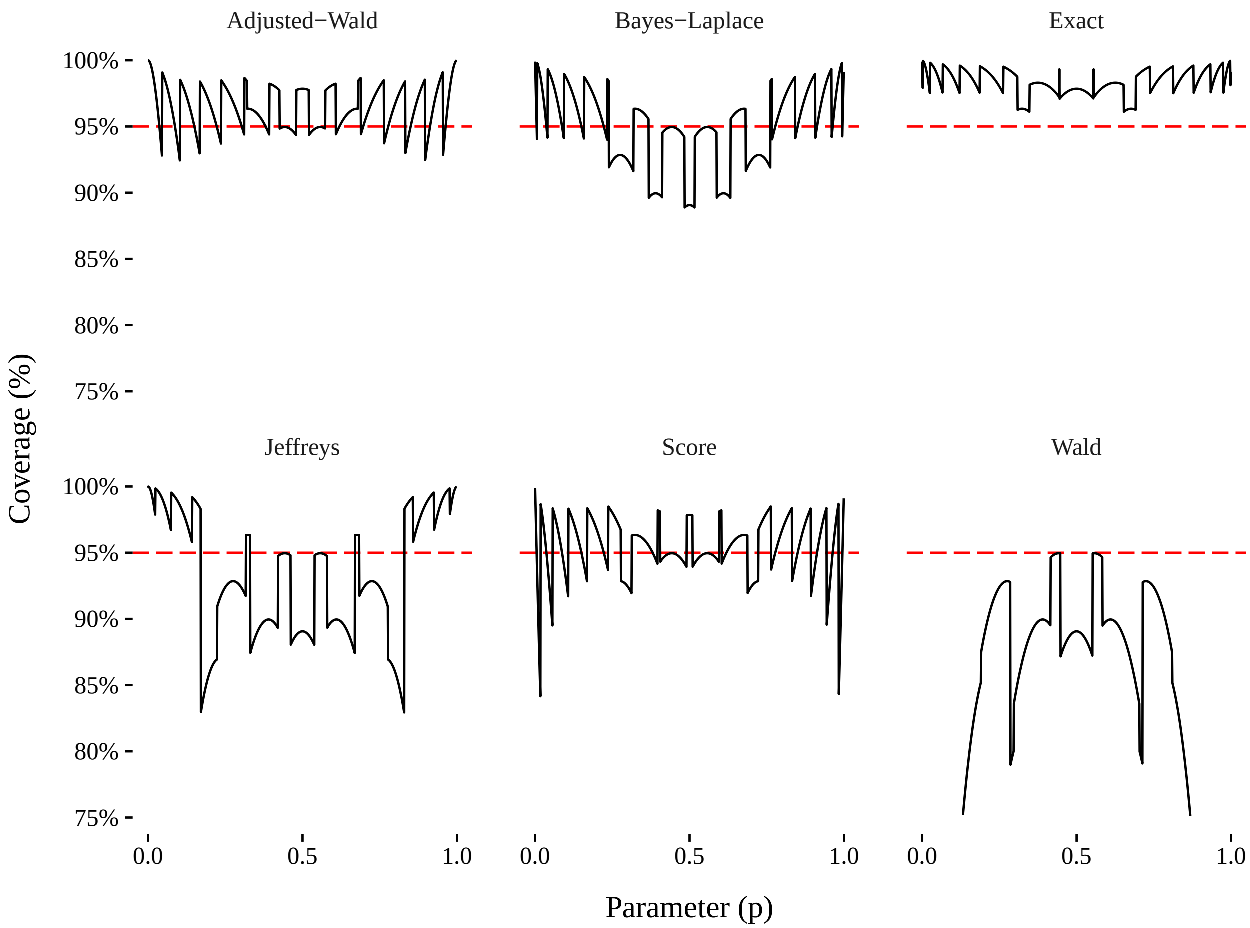
```
solve.HPD.beta = function(shape1, shape2, # shape1 and shape2 from the beta.
                          credint = 0.95, one.sided = FALSE,...){

  if(shape1 <= 1 | one.sided == TRUE){ # for x = 0 and one-sided intervals.
    lt = 0
    ut = qbeta(credint, shape1, shape2)
    coverage = credint
    results = data_frame("Lower"    = lt,
                         "Upper"    = ut,
                         "Coverage" = coverage,
                         "Height"   = ut)
    return(results)
  }
  if(shape1 > n){ # for x = n
    lt = qbeta(1-credint, shape1, shape2)
    ut = 1
    coverage = credint
    results = data_frame("Lower"    = lt,
                         "Upper"    = ut,
                         "Coverage" = coverage,
                         "Height"   = lt)
    return(results)
  } else { # otherwise
    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)
      abs(credint-coverage)
    }
    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$double.eps,
                 shape1,
                 shape2)
    # This will return the actual values after the optimiser is done.
    h <- h$minimum
    mode = (shape1 - 1)/(shape1 + shape2 - 2)
    lt = uniroot(f=function(x){ dbeta(x,shape1, shape2) - h},
                 lower=0, upper=mode)$root
```

```r
    ut = uniroot(f=function(x){ dbeta(x,shape1, shape2) - h},
                 lower=mode, upper=1)$root
    coverage = pbeta(ut, shape1, shape2) - pbeta(lt, shape1, shape2)
    results = data_frame("Lower"    = lt,
                         "Upper"    = ut,
                         "Coverage" = coverage,
                         "Height"   = h)
    return(results)
  }}

# Call to see the results.
y=1; n=100; a=1; b=1; p=0.95
solve.HPD.beta(shape1 = y + a, shape2 = n - y + b)
```

```
## # A tibble: 1 x 4
##          Lower      Upper Coverage   Height
##          <dbl>      <dbl>    <dbl>    <dbl>
## 1 0.0004144375 0.04630291     0.95 4.286167
```

(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. The values for the sawtooth graphs are obtained by using vapply and a vector of values for the parameter between 0 and 1: e.g for the Wald we use "vapply(p,waldcover,0)".

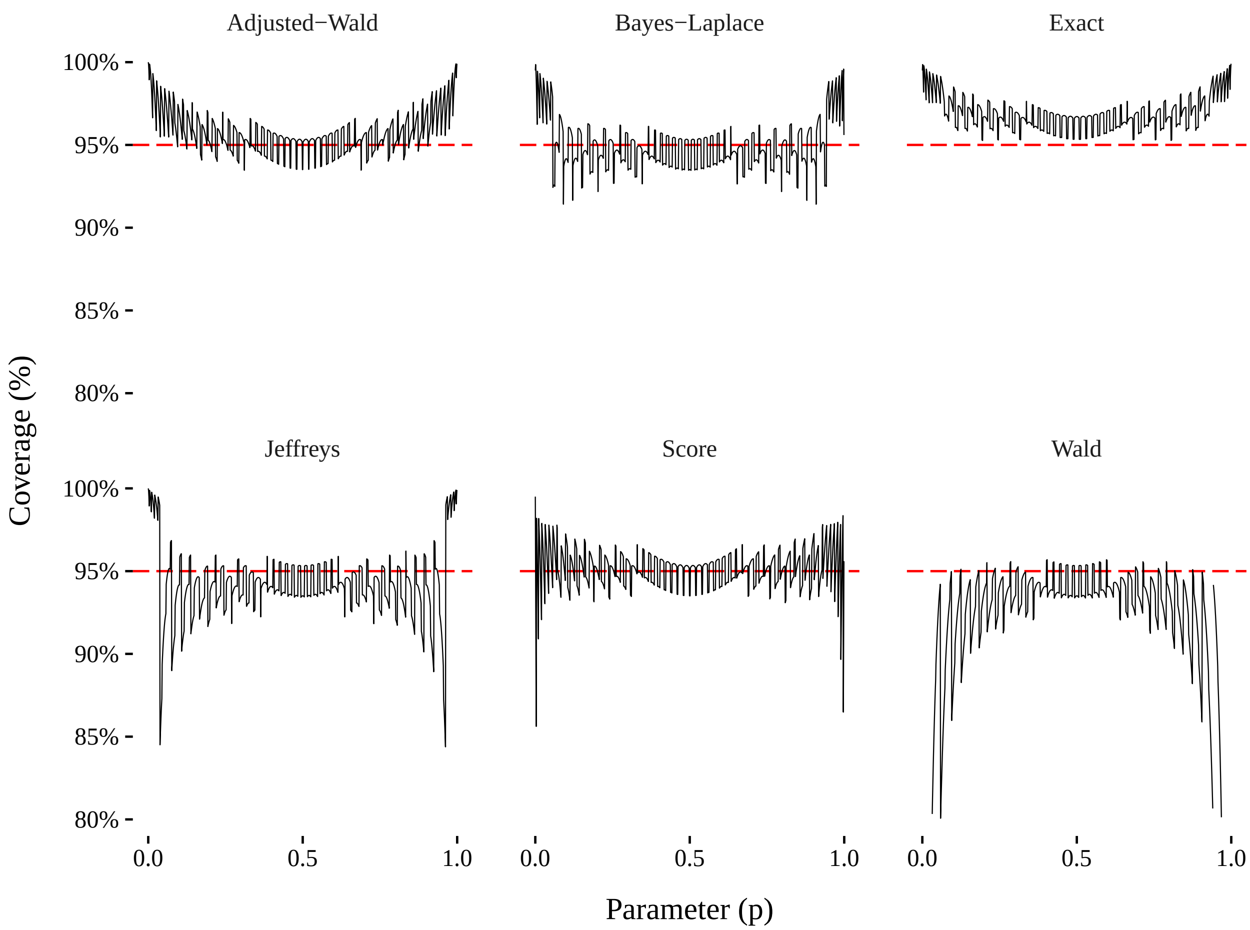
```r
n <- 10; a <- 0.05; p <- seq(0.0001,0.9999,1/1000); z <- abs(qnorm(.5*a,0,1))

Q1bdata <- tbl_df(p) %>%
  rename(p = value) %>%
  mutate(Wald = vapply(p,waldcover,0)) %>%
  mutate("Adjusted-Wald" = vapply(p,adjwaldcover, 0)) %>%
  mutate(Exact = vapply(p,exactcover,0)) %>%
  mutate(Score = vapply(p,scorecover,0)) %>%
  mutate("Bayes-Laplace" = vapply(p, blcover,0)) %>%
  mutate(Jeffreys = vapply(p,jeffreyscover,0)) %>%
  gather(key = covinterval, value = dens, -p)

q1bchart <- ggplot(data = Q1bdata, aes(x = p, y = dens)) +
  geom_hline(yintercept = 1-a, linetype = 5, colour = "red") +
  geom_line() +
  scale_y_continuous(labels = scales::percent,limits = c(0.75,1)) +
  facet_wrap(~covinterval, nrow = 2) +
  theme_tufte(base_size = 14) +
  scale_x_continuous(breaks = c(0,0.5,1)) +
  ylab("Coverage (%)") +
  xlab("Parameter (p)") +
  theme(panel.margin.x=unit(1.5, "lines")) +
  theme(axis.title.y=element_text(margin = margin(0,15,0,0))) +
  theme(axis.title.x = element_text(margin = margin(15,0,0,0)))

q1bchart # calls the chart.
```



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

```
n <- 50; a <- 0.05; p <- seq(0.0001,0.9999,1/1000); z <- abs(qnorm(.5*a,0,1))

Q1cdata <- tbl_df(p) %>%
  rename(p = value) %>%
  mutate(Wald = vapply(p,waldcover,0)) %>%
  mutate("Adjusted-Wald" = vapply(p,adjwaldcover, 0)) %>%
  mutate(Exact = vapply(p,exactcover,0)) %>%
  mutate(Score = vapply(p,scorecover,0)) %>%
  mutate("Bayes-Laplace" = vapply(p, blcover,0)) %>%
  mutate(Jeffreys = vapply(p,jeffreyscover,0)) %>%
  gather(key = covinterval, value = dens, -p)

q1cchart <- ggplot(data = Q1cdata, aes(x = p, y = dens)) +
  geom_hline(yintercept = 1-a, linetype = 5, colour = "red") +
  geom_line(size = 0.25) +
  scale_y_continuous(labels = scales::percent, limits = c(0.8,1)) +
  facet_wrap(~covinterval, nrow = 2) +
  theme_tufte(base_size = 14) +
  scale_x_continuous(breaks = c(0,.5,1)) +
  ylab("Coverage (%)") +
  xlab("Parameter (p)") +
  theme(panel.margin.x=unit(1.5, "lines")) +
  theme(axis.title.y=element_text(margin = margin(0,15,0,0))) +
```

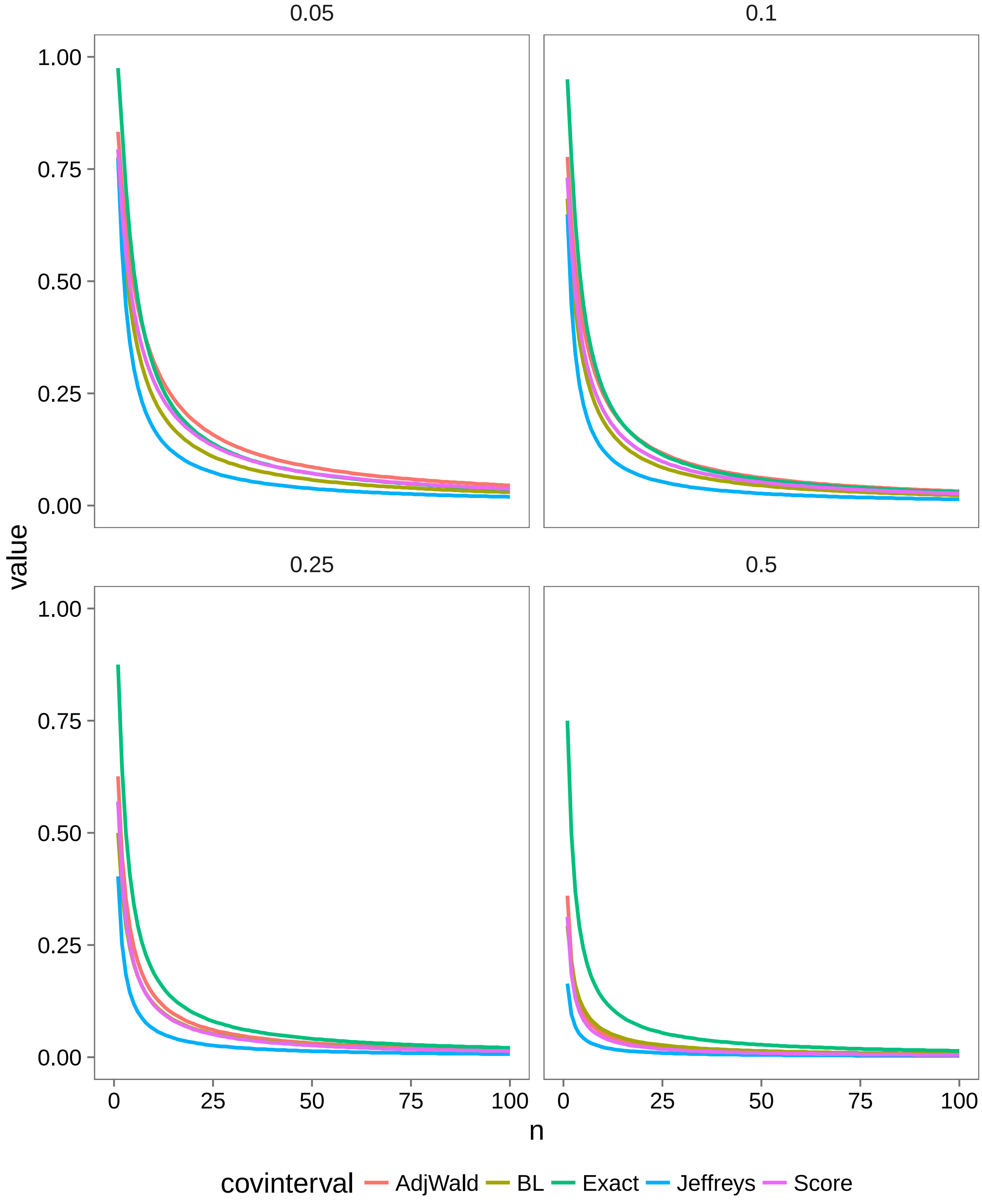
```
  theme(axis.title.x = element_text(margin = margin(15,0,0,0)))

q1cchart
```



(d) Compare the minimum coverage of the six graphs at (c) Firslty, the minimum coverage of the Wald interval is terrible.

covinterval	mean(dens)	median(dens)	min(dens)
Adjusted-Wald	0.9579977	0.9556593	0.9346807
Bayes-Laplace	0.9500028	0.9468787	0.9141711
Exact	0.9693205	0.9677614	0.9526640
Jeffreys	0.9418993	0.9408915	0.8438360
Score	0.9518805	0.9530135	0.8562090
Wald	0.9005129	0.9348509	0.0049878

(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. Below, I've reproduced the upper limits for each interval - for n between 1 and 100 - for different values of alpha.


0.05
0.1
0.25
0.5
value
1.00
0.75
0.50
0.25
0.00
0
25
50
75
100
n
covinterval
AdjWald
BL
Exact
Jeffreys
Score

Question 2: Inference for the Cauchy parameter:

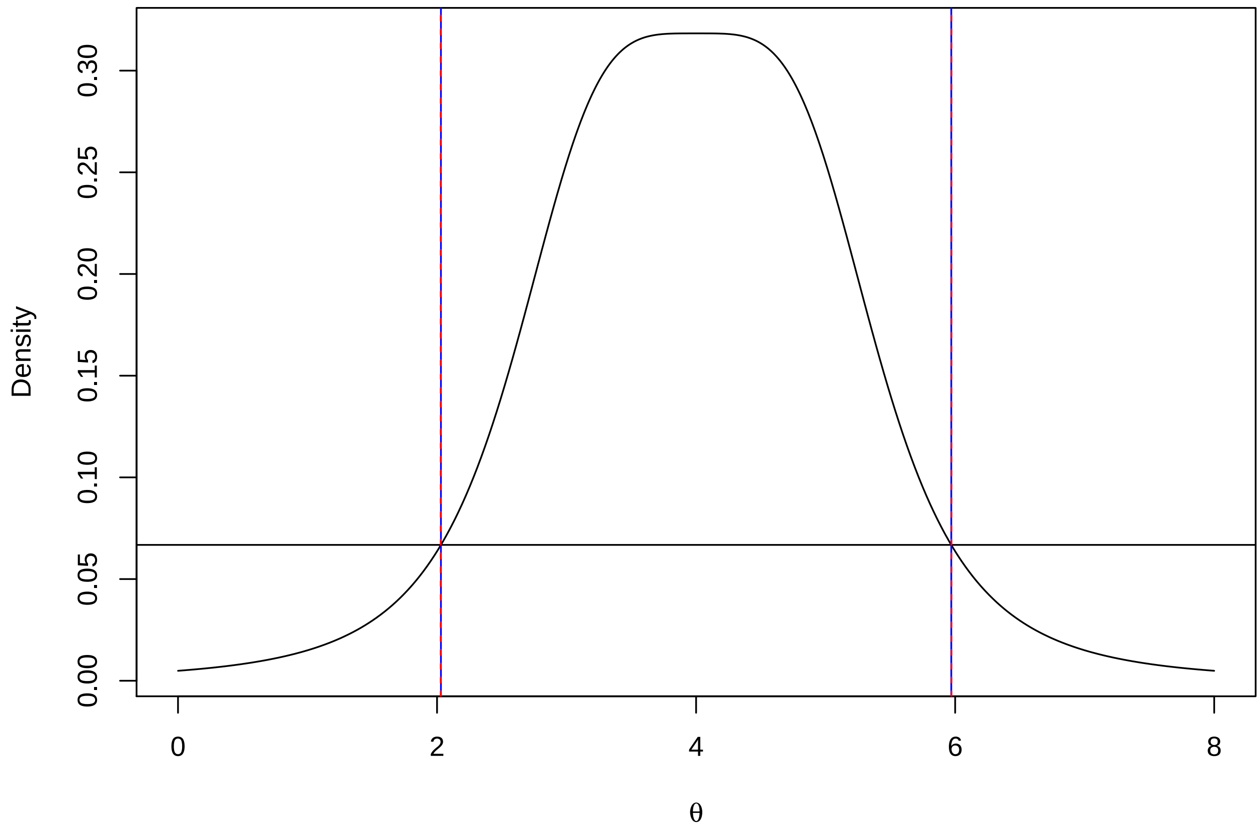
(a) Develop an R function to find percentiles of a (general) Cauchy posterior as discussed by Jaynes (1976, Example 6) and Box & Tiao (1973, p.64), to be used for the examples below.

```
CauchyPercentage <- function(y = NULL,# a value to test Pr[p < y]
                             x, # a vector of samples
                             p, # a vector of possible parameters
                             climlow = -Inf,
                             climhigh = Inf) {

  cauchydens = function(x,p){ # returns a df containing dens and param values.
    cauchydist = function(x,p) { # Cauchy density described in notes.
      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) # Call the above function.

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

  c = (integrate(dens, climlow, climhigh)$value)^(-1)
  c*integrate(dens,climlow,y)$value
}

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
                      plot = TRUE) {

  cauchydens = function(x,p){ # returns a df containing dens and param values.
    cauchydist = function(x,p) { # Cauchy density described in notes.
      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) # Call the above function.

  cumdist = cumsum(df$postdens)*diff(df$vparams)[1] # Cumulative distribution
  post_median = which.min(abs(cumdist-0.5)) # The posterior median.
```

```
  # find lower and upper values for which prob dens is closest to target value
  HPDlimits = function(post_dens) {
    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) { # find the area corresponding to that interval
    limitints = HPDlimits(post_dens)
    limitarea = sum(df$postdens[limitints[1]:limitints[2]])*diff(df$vparams)[1]
  }

  # Now calculate the HPD using the above functins.
  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 (HPD)","Upper (HPD)")
  if(plot) {
    plot(df$vparams, df$postdens, type = 'l',
         xlab = expression(theta), ylab = "Density")
    abline(h = df[HPDlimits(v2[w])[1],"postdens"])
    abline(v = r["Upper (HPD)"], col = 'blue')
    abline(v = r["Lower (HPD)"], 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).

```
x <- c(3,5); p <- seq(0,8,1/100)
interval <- seq(0,3, by = 1/1000)
h <- vector("numeric",2L)
names(h) <- c("Lower (Central)", "Upper (Central)")
h[1] <- optimize(CauchyPercentage,
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.05)$minimum

interval <- seq(5,8, by = 1/1000)
h[2] <- optimize(CauchyPercentage,
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.95)$minimum
r <- CauchyHPD(x = x, p = p, alpha = 0.9, tol = 0.0001)
abline(v = h[1], col = "red", lty = 2)
abline(v = h[2], col = "red", lty = 2)
```



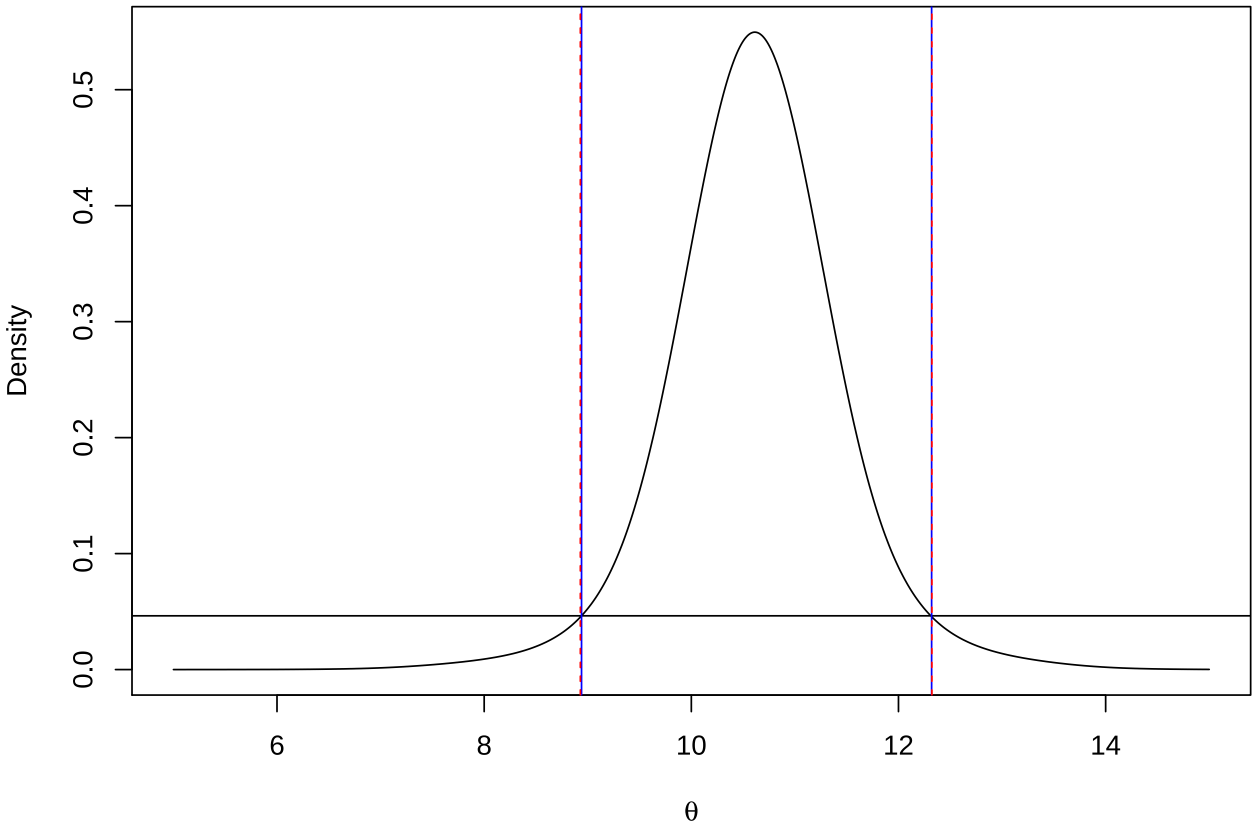

```r
c(r, h)
```

```
##           lower           upper Lower (Central) Upper (Central)
##        2.030000        5.970000        2.028646        5.971354
```

(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. Here's the code and the graph.

```r
x <- c(11.4, 7.3, 9.8, 13.7, 10.6); p <- seq(5,15,by = 1/100)
h <- vector("numeric",2L); names(h) <- c("Lower (Central)", "Upper (Central)")
interval <- seq(5,10, by = 1/1000)
h[1] <- optimize(CauchyPercentage, # Lower Central limit
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.025)$minimum

interval <- seq(10,15, by = 1/1000)

h[2] <- optimize(CauchyPercentage, # Upper Central limit
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.975)$minimum
```

```r
r <- CauchyHPD(x=x,p=p,alpha = 0.95,tol = 0.00001)
abline(v = h[1], col = "red", lty = 2) # Lower Central limit
abline(v = h[2], col = "red", lty = 2) # Upper Central limit
```




```r
c(r, h)
```

```
##     Lower (HPD)     Upper (HPD) Lower (Central) Upper (Central)
##        8.940000       12.320000        8.928723       12.322678
```

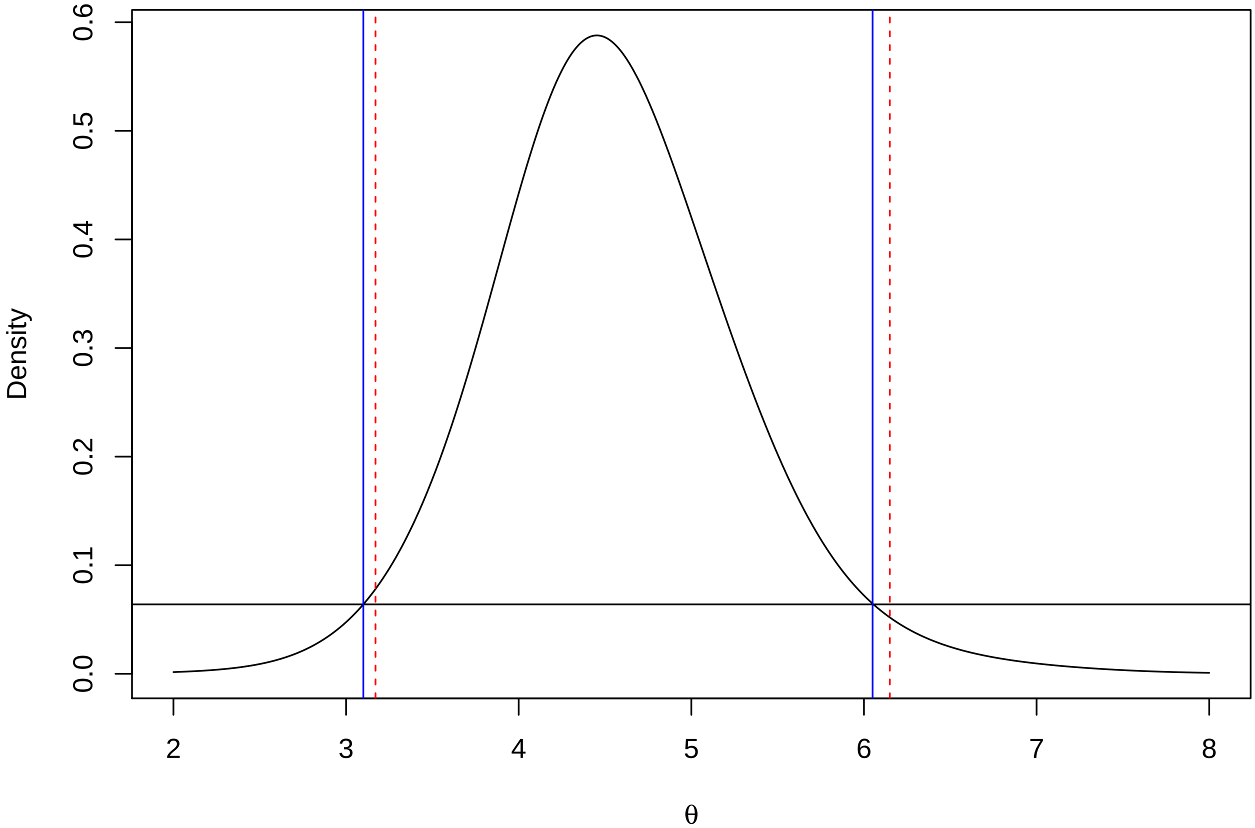
```r
CauchyPercentageNO(11.5, x=x, p=p)
```

```
## [1] 0.8772614
```

(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 ($\theta > 0$). Firstly, without the restriction.

```r
x <- c(4.0, 5.5, 7.5, 4.5, 3.0); p <- seq(2,8,1/100)
h <- vector("numeric",2L); names(h) <- c("Lower (Central)", "Upper (Central)")
interval <- seq(2,5, by = 1/1000)
h[1] <- optimize(CauchyPercentage, # Lower Central limit
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.025)$minimum

interval <- seq(6,8, by = 1/1000)
```

```r
h[2] <- optimize(CauchyPercentage, # Upper Central limit
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.975)$minimum
r <- CauchyHPD(x=x,p=p,alpha = 0.95,tol = 0.00001)
abline(v = h[1], col = "red", lty = 2) # Lower Central limit
abline(v = h[2], col = "red", lty = 2) # Upper Central limit
```



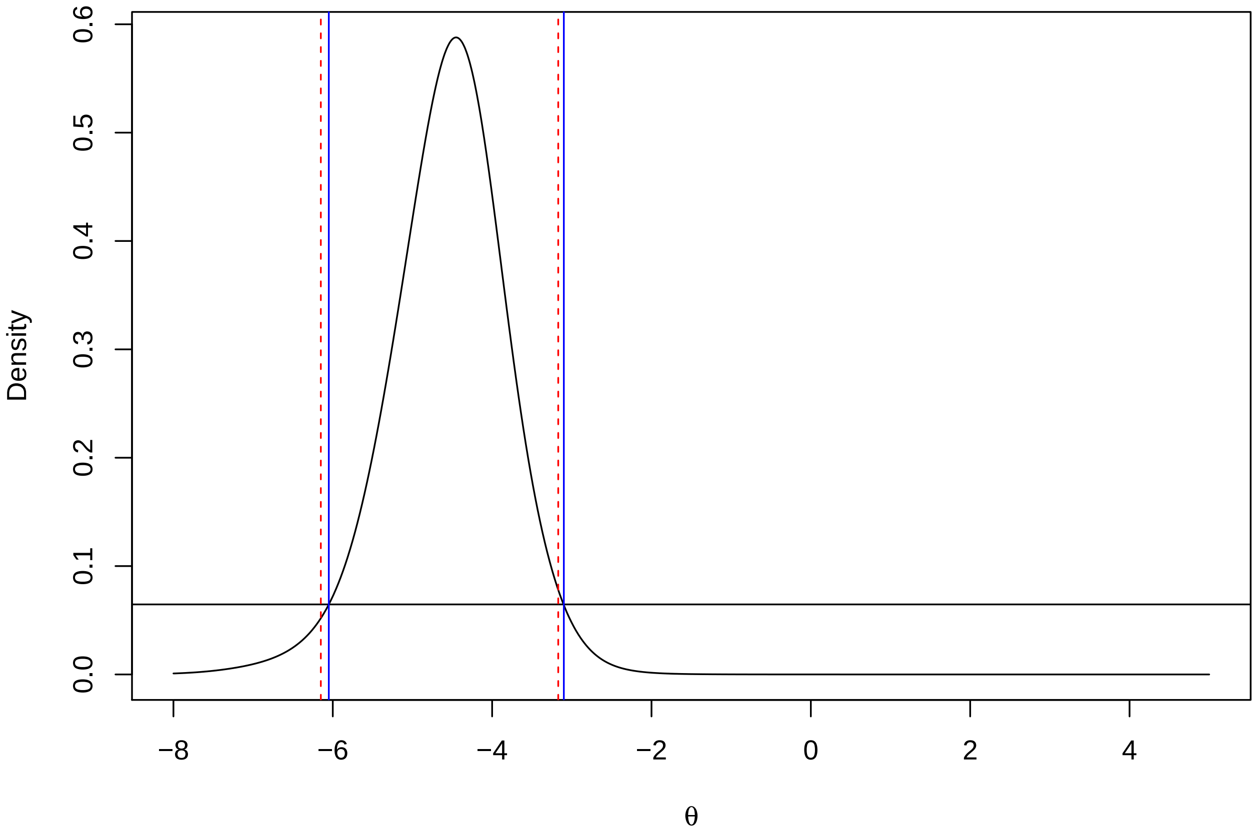

```r
c(r, h)
```

```
##     Lower (HPD)     Upper (HPD) Lower (Central) Upper (Central)
##        3.100000        6.050000        3.170594        6.149956
```

From this we can see that Berger's restriction won't affect this particular posterior. However, if we plot the negative version for the next question, we will see where this restriction may impact on our analysis.

```r
x <- c(-4.0, -5.5, -7.5, -4.5, -3.0); p <- seq(-8,5,1/100)
h <- vector("numeric",2L); names(h) <- c("Lower (Central)", "Upper (Central)")
interval <- seq(-8,-5, by = 1/1000)
h[1] <- optimize(CauchyPercentage,
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
```

```r
                 alpha = 0.025)$minimum

interval <- seq(-5,8, by = 1/1000)

h[2] <- optimize(CauchyPercentage,
                 interval = interval,
                 lower = min(interval),
                 tol = .Machine$double.eps^0.25,
                 x=x,
                 p=p,
                 alpha = 0.975)$minimum
r <- CauchyHPD(x=x,p=p,alpha = 0.95,tol = 0.00001)
abline(v = h[1], col = "red", lty = 2) # upper Central limit
abline(v = h[2], col = "red", lty = 2) # lower Central limit
```

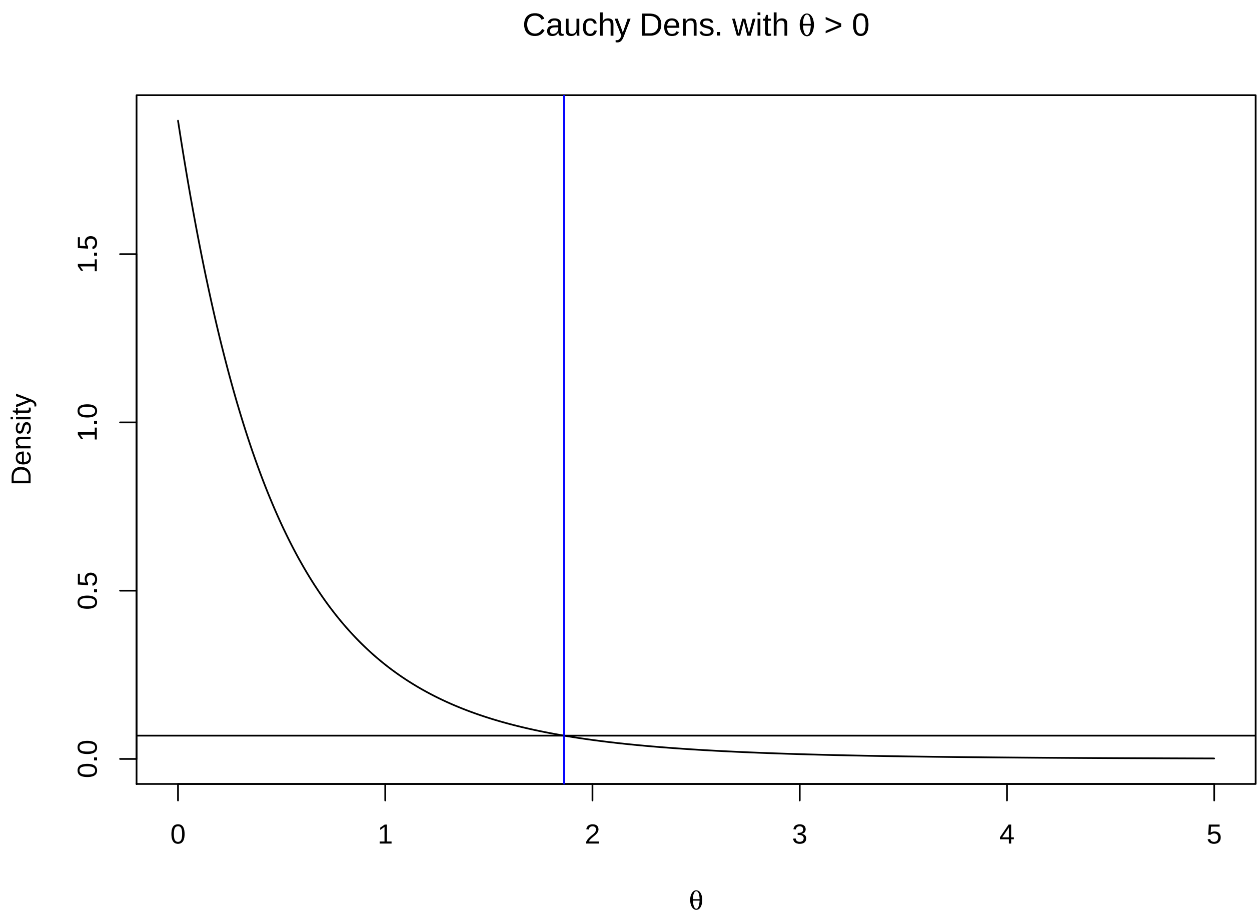



```r
c(r, h)
```

```
##     Lower (HPD)     Upper (HPD) Lower (Central) Upper (Central)
##       -6.050000       -3.100000       -6.149995       -3.170589
```

(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
x <- c(-4.0, -5.5, -7.5, -4.5, -3.0); p <- seq(0,5,1/1000)
CauchyHPD(x=x,p=p, climlow = 0, climhigh = Inf)
```




```
##     lower     upper      dens
## 0.0000000 1.8630000 0.9490335
```