R codes for technical report

Debodeep Banerjee, Trina Sahoo, Subbiah M., M.R. Srinivasan

14 January 2018

This is Web Appendix for Technical Report

Comparative Analysis of two $1 \times$ J Contingency Tables using Bayesian Perspective

1. In order to calculate the required value of the upper limit θ_u , (say), we transformed it into F distribution.

We use R to calculate the required upper limit of the parameters.

For calculating the upper limit, the code is,

qf (0.975, m , n)

In order to calculate the lower limit, the R code is,

qf (0.025, m, n)

[In both the cases m and n are the respective degrees of freedom. The user has to put the value of m and n while using the software.]

2. R code for generating the probability of ratio of θ_i and λ_i

a. Comparative analysis based on 'Female security'

```
knitr::opts_chunk$set(echo = TRUE)
#For P($$\theta_7 \ge$$\lambda_7) (i.e.,P(($$\theta_7/$$\lambda_7) \ge 1$$)) and P($$\theta_7\le$$\lamb
#Packages required#
library(hypergeo)
require(elliptic)
#parameters for theta1
a=3.5
b=59.5
#parameters for theta2
al=18.5
be=48.5
A=beta(a,b)*beta(al,be)
B1=beta(a+al,b)
B2=beta(a+al,be)

#w > 1
fff <- function(w) {(B1/A)*w^(-(al+1))*hypergeo(a+al,1-be,a+al+b,1/w)}
#0 < w < 1
fff1 <- function(w) {(B2/A)*w^(a-1)*hypergeo(a+al,1-b,a+al+be,w)}

m1=myintegrate(fff,1,Inf)
m2=myintegrate(fff1,0,1)
m1
```

```
## [1] 0.0001744699+0i
```

```
m2
```

```
## [1] 0.9998255-0i
```

b. Comparative analysis based on 'Religious extremism'

```
knitr::opts_chunk$set(echo = TRUE)
#For P($$\theta_3 \ge$$\lambda_3) (i.e.,P(($$\theta_3/$$\lambda_3) \ge 1$$)) and P($$\theta_3\le$$\lamb
library(hypergeo)
require(elliptic)
#parameters for theta1
a=6.5
b=37.5
#parameters for theta2
al=7.5
be=36.5
A=beta(a,b)*beta(al,be)
B1=beta(a+al,b)
B2=beta(a+al,be)

#w > 1
fff <- function(w) {(B1/A)*w^(-(al+1))*hypergeo(a+al,1-be,a+al+b,1/w)}
#0 < w < 1
fff1 <- function(w) {(B2/A)*w^(a-1)*hypergeo(a+al,1-b,a+al+be,w)}

m1=myintegrate(fff,1,Inf)
m2=myintegrate(fff1,0,1)
m1
```

```
## [1] 0.3822509+0i
```

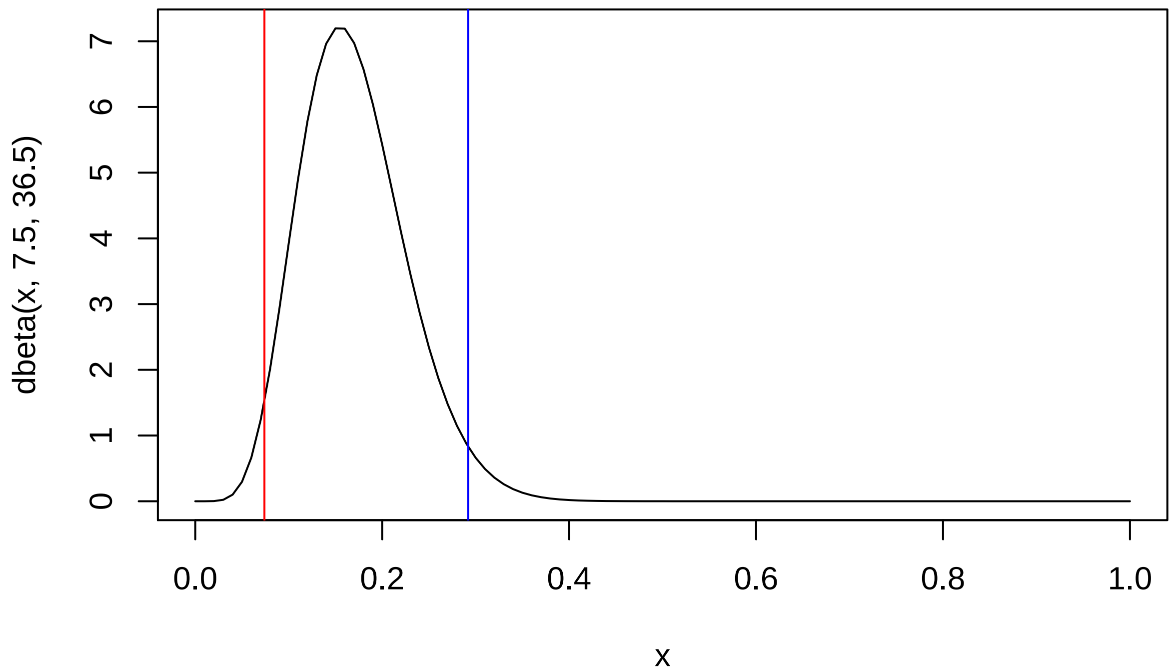
```
m2
```

```
## [1] 0.6177491-0i
```

3. Code to plot the credibility intervals. Generalised form. Curve(dbeta(x,shape1,shape2),0,1) # to draw the curve ll<-value (enter the lower limit) ul<-value (enter the upper limit)

abline(v=ll) abline(v=ul) example. Let us construct the credible interval for θ_6 The shape parameters are 7.5 and 36.5, respectively. Lower limit of the credibility interval is 0.074 Upper limit of the credibility interval is 0.292 Then the R code will be as the following.

```
knitr::opts_chunk$set(echo = TRUE)
curve(dbeta(x,7.5,36.5),0,1)
ll<-0.074
ul<-0.292
abline(v=ll,col="red")
abline(v=ul,col="blue")
```



4. Code fot the Monte Carlo integration

```
knitr::opts_chunk$set(echo = TRUE)
library(MCMCpack)
x<-c(2,10,12,5,7,18,3,2) #Given Multinomial data for males(Data1)
y<-c(4,10,8,2,5,14,18,2) #Given Multinomial data for females(Data2)
l1<-length(x)
l2<-length(y)
ransamp<-100000 #NO OF RUNS FOR SIMULATION OF POSTERIOR
po_jfx<-x+0.5 #POSTERIOR BASED ON JEFFREYS PRIOR ON DIRIC PARAMETERS for Data1
po_jfy<-y+0.5 #POSTERIOR BASED ON JEFFREYS PRIOR ON DIRIC PARAMETERS for Data2

              #==============#Summary for Data1 and Data2#==================#
drx<-rdirichlet(ransamp,po_jfx) #POSTERIOR SIMULATION BASED ON JEFFREYS PRIOR for Data1
dry<-rdirichlet(ransamp,po_jfy) #POSTERIOR SIMULATION BASED ON JEFFREYS PRIOR for Data2
#===##Data1##===#
m1=0
v1=0
low1=0
up1=0
dif1=0
for(j in 1:l1)
{
  m1[j]=round(mean(drx[,j]),4)                  #POINT ESTIMATE
  v1[j]=round(var(drx[,j]),4)
  low1[j]=round(quantile(drx[,j],0.025),4)        #INTERVAL ESTIMATES
  up1[j]=round(quantile(drx[,j],0.975),4)
```

```r
  dif1[j]=up1[j]-low1[j]                                  #LENGTH OF INTERVAL
}

#====##Data2##====#
m2=0
v2=0
low2=0
up2=0
dif2=0
for(k in 1:l2)
{
  m2[k]=round(mean(dry[,k]),4)                       #POINT ESTIMATE
  v2[k]=round(var(dry[,k]),4)
  low2[k]=round(quantile(dry[,k],0.025),4)       #INTERVAL ESTIMATES
  up1[k]=round(quantile(dry[,k],0.975),4)
  dif1[k]=up2[k]-low2[k]                                  #LENGTH OF INTERVAL
}
                #--------------------------------------------------------------------------------
#RESULTS#

#====##Data1##====#
ans_PEx=cbind(m1)
ans_PVx=cbind(v1)
colnames(ans_PEx)=c("Jeff")
ans_CIx=cbind(cbind(low1,up1))
colnames(ans_CIx)=c("Je_LL","Je_UL")
ans_PRx=cbind(prod(dif1))
colnames(ans_PRx)=c("Jeff")

ans_PEx
```

```
##        Jeff
## [1,] 0.0397
## [2,] 0.1666
## [3,] 0.1988
## [4,] 0.0871
## [5,] 0.1189
## [6,] 0.2937
## [7,] 0.0556
## [8,] 0.0396
```

```r
ans_PVx
```

```
##          v1
## [1,] 0.0006
## [2,] 0.0022
## [3,] 0.0025
## [4,] 0.0012
## [5,] 0.0016
## [6,] 0.0033
## [7,] 0.0008
## [8,] 0.0006
```

```
ans_CIx
```

```
##        Je_LL  Je_UL
## [1,] 0.0068 0.1377
## [2,] 0.0860 0.2528
## [3,] 0.1102 0.2159
## [4,] 0.0315 0.0934
## [5,] 0.0516 0.1570
## [6,] 0.1894 0.3222
## [7,] 0.0140 0.3877
## [8,] 0.0067 0.0939
```

```
ans_PRx
```

```
##       Jeff
## [1,]   NA
```

```
#===##Data2##===#
ans_PEy=cbind(m1)
ans_PVy=cbind(v1)
colnames(ans_PEy)=c("Jeff")
ans_CIy=cbind(cbind(low1,up1))
colnames(ans_CIy)=c("Je_LL","Je_UL")
ans_PRy=cbind(prod(dif1))
colnames(ans_PRy)=c("Jeff")

ans_PEy
```

```
##         Jeff
## [1,] 0.0397
## [2,] 0.1666
## [3,] 0.1988
## [4,] 0.0871
## [5,] 0.1189
## [6,] 0.2937
## [7,] 0.0556
## [8,] 0.0396
```

```
ans_PVy
```

```
##           v1
## [1,] 0.0006
## [2,] 0.0022
## [3,] 0.0025
## [4,] 0.0012
## [5,] 0.0016
## [6,] 0.0033
## [7,] 0.0008
## [8,] 0.0006
```

```
ans_CIy
```

```
##        Je_LL  Je_UL
## [1,] 0.0068 0.1377
## [2,] 0.0860 0.2528
## [3,] 0.1102 0.2159
## [4,] 0.0315 0.0934
```

```
## [5,] 0.0516 0.1570
## [6,] 0.1894 0.3222
## [7,] 0.0140 0.3877
## [8,] 0.0067 0.0939
```

```r
ans_PRy
```

```
##      Jeff
## [1,]   NA
```

```r
#___________________________________________________________-
#This section illustrates how probabilities can be computed from the posterior simulation

#1. p(theta1>theta1) based on three priors
measure<-vector()
for(i in 1:l1)
{
 for(j in 1:l2)
  {
   if(i==j)
    {
      measure[i]<-length(which(drx[,i]>dry[,i]))/ransamp
    }
  }
}
measure
```

```
## [1] 0.22800 0.55945 0.87056 0.89463 0.76366 0.84591 0.00009 0.52525
```