JAGS Homework

Tim Hogan

4/13/2020

Introduction

Hello! I went through this on a Mac, so all of the JAGS outputs ended up coming out through XTerm (XQuartz) and I have NO idea how this is gonna work with RMarkdown. For this, I'm going to mostly include some text description around some of the output from the coding. As mentioned, I primarily referred to and ajdusted his example code, "Jags-Ydich-XnomSsubj-MbernBeta-Example.R" with examples from his book! Also, as a change of pace, I will be including R output in this report, because I feel like the exercises want to see them. Sorry for when it looks awful!

8.1)

I didn't want to make a new file for this, so I just directly input it into R. I had three groups, a big one with values determined with rbinom, and unequal smaller groups of only 1's and 0's.

```
myData <- data.frame(y = c(rbinom(20,1,0.5), rep(1,3),rep(0,8)),
                     s = c(rep("A",20),rep("B",3),rep("C",8)))
```

After this, I simply run his code! I commented out the file-saving portion of it, but it's included in GitHub as JAGS81.eps

```
source("Jags-Ydich-XnomSsubj-MbernBeta.R")
```

```
##
## ************************************************************************
## Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition:
## A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
## ************************************************************************
```

```
mcmcCoda = genMCMC( data=myData , numSavedSteps=10000 )
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 31
##    Unobserved stochastic nodes: 3
##    Total graph size: 68
##
## Initializing model
##
## Burning in the MCMC chain...
## Sampling final MCMC chain...
```

```
plotMCMC( mcmcCoda , data=myData , compVal=NULL , compValDiff=0.0 ,
#         saveName=fileNameRoot , saveType=graphFileType
)
```

Overall, the sampler seems to be estimating relatively close to the estimates, with the distributions still pulled closer to 0.5-0.6. The group (theta 2) with the least parameters seem sto be pulledd closest to the center, which most likely reflects the prior information outweighing the samples. Theta 3, which is just as extreme

but with more samples, seems to have a lower weighting on the prior as its distribution leans closer to the true value, 1.

8.2

First I'll go back to using the original data.

```
myData <- data.frame(y = c(1,0,1,1,1,1,1,0,0,0,1,0,0,1,0),
                     s = c(rep("Reginald",8),rep("Tony",7)))

mcmcCoda = genMCMC( data=myData , numSavedSteps=10000 ,)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 15
##    Unobserved stochastic nodes: 2
##    Total graph size: 35
##
## Initializing model
##
## Burning in the MCMC chain...
## Sampling final MCMC chain...
```

```
smryMCMC( mcmcCoda , compVal=NULL , compValDiff=0.0,rope=c(0.45,0.55) )
```

```
##                       Mean    Median      Mode   ESS HDImass     HDIlow
## theta[1]          0.6667007 0.6762220 0.7084132 10000    0.95  0.4155877
## theta[2]          0.3622952 0.3544864 0.3232001 10000    0.95  0.1046568
## theta[1]-theta[2] 0.3044055 0.3122903 0.3387454 10000    0.95 -0.0633084
##                     HDIhigh CompVal PcntGtCompVal ROPElow ROPEhigh
## theta[1]          0.9072403      NA            NA    0.45     0.55
## theta[2]          0.6301145      NA            NA    0.45     0.55
## theta[1]-theta[2] 0.6723525       0         93.88      NA       NA
##                   PcntLtROPE PcntInROPE PcntGtROPE
## theta[1]                5.94      13.38      80.68
## theta[2]               73.87      16.15       9.98
## theta[1]-theta[2]         NA         NA         NA
```

```
##                       Mean    Median      Mode   ESS HDImass     HDIlow
## theta[1]          0.6667007 0.6762220 0.7084132 10000    0.95  0.4155877
## theta[2]          0.3622952 0.3544864 0.3232001 10000    0.95  0.1046568
## theta[1]-theta[2] 0.3044055 0.3122903 0.3387454 10000    0.95 -0.0633084
##                     HDIhigh CompVal PcntGtCompVal ROPElow ROPEhigh
## theta[1]          0.9072403      NA            NA    0.45     0.55
## theta[2]          0.6301145      NA            NA    0.45     0.55
## theta[1]-theta[2] 0.6723525       0         93.88      NA       NA
##                   PcntLtROPE PcntInROPE PcntGtROPE
## theta[1]                5.94      13.38      80.68
## theta[2]               73.87      16.15       9.98
## theta[1]-theta[2]         NA         NA         NA
```

In terms of the summary, the first rows show summary statistics of the distribution sampled by the Gibbs Sampler. The HDI specifies the range where theta is going to be given the HDImass specified; it's effectively a Gibbs version of a confidence interval. HDIlow and HDI high are the upper and lower bounds of the HDImass (0.95) interval.

The "rope" argument specifies a range of theta values that it can take. It shows the lower and upper ranges of the theta being looked at (ROPElow and ROPEhigh), and then shows the percentage of the data less than (PcntLtROPE), within (PcntInRope), and greater than (PcntGtROPE) the range specified.

8.3

I'll include the code but won't run it and will just describe what's being saved!

```
# myData <- data.frame(y = c(1,0,1,1,1,1,1,0,0,0,1,0,0,1,0),
#                      s = c(rep("Reginald",8),rep("Tony",7)))

# source("Jags-Ydich-XnomSsubj-MbernBeta.R")

# fileNameRoot = "Jags-Ydich-XnomSsubj-MbernBeta-"
# graphFileType = "eps"

# mcmcCoda = genMCMC( data=myData , numSavedSteps=50000 , saveName=fileNameRoot )

# parameterNames = varnames(mcmcCoda) # get all parameter names
# for ( parName in parameterNames ) {
#   diagMCMC( codaObject=mcmcCoda , parName=parName ,
#             saveName=fileNameRoot , saveType=graphFileType )
# }


# summaryInfo = smryMCMC( mcmcCoda , compVal=NULL , #rope=c(0.45,0.55) ,
#                         compValDiff=0.0 , #ropeDiff = c(-0.05,0.05) ,
#                         saveName=fileNameRoot )

# plotMCMC( mcmcCoda , data=myData , compVal=NULL , #rope=c(0.45,0.55) ,
#           compValDiff=0.0 , #ropeDiff = c(-0.05,0.05) ,
#           saveName=fileNameRoot , saveType=graphFileType )
```

From when I ran it, the following was saved:

First, an RData file including the simulated theta's for each gibbs sampler was saved. The second and third files includes the plots produced for each theta parameter made by diagMCMC(). Next, summaryInfo saves a CSV including the output data frame for the summary information. Lastly, the gridded plot produced by plotMCMC was saved as a file. There is also a file created called "TEMPmodel" which shows the prior being used.

8.4

8.4.1

There's a line of code in Jags-Ydich-XnomSsubj-MbernBeta.R that converts the information from the data into a structure. Commenting out y like below removes the data and allows us to work solely with prior information.

```
#This is all commented out because otherwise the RMD won't run
# and I'm feeling too lazy to find a creative solution to this.
#dataList = list(
# #   y = y ,
#    s = s ,
#    Ntotal = Ntotal ,
#    Nsubj = Nsubj
```

```
#  )
```

To replicate the chart, I primarily used the above change and an adjusted source code, named JAGS841.R. This is instructed in the textbook, so it's pretty straightforward.

```
source("JAGS841.R")
```

```
##
## ************************************************************************
## Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition:
## A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
## ************************************************************************
```

```
mcmcCoda = genMCMC( data=myData , numSavedSteps=10000 )
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 0
##    Unobserved stochastic nodes: 17
##    Total graph size: 35
##
## Initializing model
##
## Burning in the MCMC chain...
## Sampling final MCMC chain...
```

```
plotMCMC( mcmcCoda , data=myData , compVal=NULL , compValDiff=0.0 ,
#         saveName=fileNameRoot , saveType=graphFileType
)
```

The graphic (JAGS841.eps) produced by this shows the shape of the beta prior, a dome-like shape with a rounded peak in the middle and out at the end. The two distributions seem to be spread around a similar pattern.

8.4.2

For this one, we simply need to change the prior beta's parameters in the original code to (1,1). There's a line in the code that's around line 35:

```
#for ( sIdx in 1:Nsubj ) {
#       theta[sIdx] ~ dbeta( 2 , 2 ) # N.B.: 2,2 prior; change as appropriate.
#    }
```

Changing the coefficients will change the coefficients in the prior. This file is active under the name JAGS842.R.

```
source("JAGS842.R")
```

```
##
## ************************************************************************
## Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition:
## A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
## ************************************************************************
```

```
mcmcCoda = genMCMC( data=myData , numSavedSteps=10000 )
```

```
## Compiling model graph
##    Resolving undeclared variables
```

```
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 0
##    Unobserved stochastic nodes: 17
##    Total graph size: 35
##
## Initializing model
##
## Burning in the MCMC chain...
## Sampling final MCMC chain...
```

```
plotMCMC( mcmcCoda , data=myData , compVal=NULL , compValDiff=0.0 ,
#         saveName=fileNameRoot , saveType=graphFileType
)
```

Once again, this is looking at the prior beta distribution created by the sampler. A beta with parameters 1,1 is equivalent to a uniform distribution. We're seeing this in this image (JAGS842.eps), with both theta's exhibiting flat line behaviors.

8.4.3

The method for the last task is the same as it is for the above, but instead of changing the parameters to 1, it's to 0.5. The code will be included in JAGS843.R

```
source("JAGS843.R")
```

```
##
## ********************************************************************
## Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition:
## A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
## ********************************************************************
```

```
mcmcCoda = genMCMC( data=myData , numSavedSteps=10000 )
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 0
##    Unobserved stochastic nodes: 17
##    Total graph size: 35
##
## Initializing model
##
## Burning in the MCMC chain...
## Sampling final MCMC chain...
```

```
plotMCMC( mcmcCoda , data=myData , compVal=NULL , compValDiff=0.0 ,
#         saveName=fileNameRoot , saveType=graphFileType
)
```

The distribution shown (JAGS843.eps) seems to be the inverse of the first one, with a low, rounded dip in the middle and higher values around the end. This would seem to weigh extremes and not the middle.