Gabriela Niemcewicz

17746555

2nd year BAFS

Bayesian Analysis Assignment 2

I use Jaggs by adding the library(rjags). I’ve made a general model using symbols, model is applicable throughout the analysis, rather than hardcoding values into the repeated algorithm:

*model{*

*# the likelihood*

*for(i in 1:N){*

*x[i] ~ dpois(lambda)*

*}*

*# the prior*

*lambda ~ dgamma(r, lambdap)*

*}*

I believe code is cleaner, not repeated, reusable, cohesive and understandable if variables needed for each posterior calculation are passed by assigning values of inits. Inits are just gamma prior’s assumed values (r and lambdap).

For 1(a),

*inits = list(lambdap = 9, r=6)*

*And length of data is passed within the list: misprints = list(N=6,x=c(3,4,2,1,2,3))*

For 2(a), it’s enough to change this line of code.

*Prior is assumed and informative, but updating increasing data amount had greater impact on posterior (by weighting down impact of prior) when data size was small at N= 6.*

*Mean of first observations is 2.5. Mean of added ones is 3.3. Mean of 16 observations of raw data is 3. Prior Gamma(9,6) put highest probability close to Gamma mean 1.3, with most density between 1 and 2 expected misprints, up to small probability of 4 of them. Such prior weighted down first 1(a) posterior mean than what’s inferred from data likelihood, as if first observations were overshooting what performance we expected from author. Posterior’s mean 1(a) at this small datasets estimates around 2- value in the middle between prior mean and 6-observations mean. But the author keeps making mistakes, and misprints of data added next by extra data’s likelihood to even next Posterior from 2(a/b) makes next posteriori misprints’ mean even higher, so we stop believing in author, but in observed mistakes he makes. First, posterior is the same, from mathematics, in both cases in 2(a), we get Gamma with mean close to 2 and standard deviation ~= 0.4. Then, after adding 10 observations, posterior’s mean is higher, but still less than 3.*

*jmodel=jags.model(file="misprints.model", data=misprints, inits) samps=jags.samples(jmodel,"lambda",n.iter=1000) # create 1000 samples from misprints posteriori model, in order to plot them:*

xmin= 0

xmax= 20 #higher than I believe to be safe at first run

*plot(density(samps$lambda),col=3,lwd=2,xlab="per page",main="number of misprints", xlim=c(xmin,xmax))*

# Now compare to the theoretical results of Po, that is Posterior*, either in this way or with many barplots:*

lambdaPo = 24

rPo= 12

lambdap = 6

curve(dgamma(lambdaPo, rPo),add=T,col=5,lwd=2,lty=2) #plot shape of theoretical posteriori d.

**Shapes are extremely similar- they look like the very same plot. I plotted many smaller samples (not depicted in assignment) from Gamma(24,12). Picture applicable to:**

theoreticalGamma <- round(rgamma(100000,shape = 24,rate = 12),1)

hist(theoreticalGamma, breaks= 1000)

legend("topleft", c("JAGS posterior","theoretical posterior"), lty=c(2,2,1,2), col=c(3,5), lwd=c(2,2))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. (a)

Using correct init and list of data, we can reuse our model.

For 2(a), it’s enough to change already used line of code into:

*Inits2 = list(lambdap = 24, r=12)*

*Misprints list is analogous:*

*misprints2 = list(N=10,x=c(2, 2, 3, 5, 2, 5, 6, 4, 3, 1))*

*Similarly, in point 2, when conducting analysis with older prior on whole dataset of length=16, we’re reusing prior from inits from 1(a), list has N=10+6, vector consists of two datasets merged.*

*Both analyses yelded exactly the same result.*

*jmodel2=jags.model(file="misprints.model", data=misprints2, inits2)*

*samps2=jags.samples(jmodel2,"lambda",n.iter=100000)*

*plot(density(samps2$lambda),col=3,lwd=2,xlab="per page",main="numer of misprints", xlim=c(xmin,xmax))*

*#posterior gamma distribution from algorithm learning from previous analysis in 1(a)*

*Prior is assumed and informative, but updating real data first time changed our inferenced posterior the most, in comparison to updating data another time, when prior’s weight on result decreased but still made expected mean go down below observed by 1 stdev.*

*# 2(a) continued now compare to using old 1(a) prior and full dataset*

*misprints2a = list(N=16,x=c(3, 4, 2, 1, 2, 3, 2, 2, 3, 5, 2, 5, 6, 4, 3, 1))*

*inits2a = list(labdap = 9, r=6)*

*jmodel2a=jags.model(file="misprints.model", data=misprints2a, inits2a)*

*samps2a=jags.samples(jmodel2a,"lambda",n.iter=100000) # create 100000 samples from misprints model posterior*

*xmin= 0*

*xmax= 15 #lower since we’re becoming more sure about the spread now*

*plot(density(samps2a$lambda),col=3,lwd=2,xlab="per page",main="numer of misprints", xlim=c(xmin,xmax))*

*We obtained gamma posterior of the same shape as Gamma(57,22). It’s mean according to R is 2,59…, so 2,6 for such a small dataset, and standard deviation of perfect representation of used Ga(57,22)- 0.34… . Full raw dataset’s sd is 1.46 because of law of large numbers, and mean is 3, so we believe our observations overshoot what we expected from the author. Oberved mean of author’s mistakes is within 1.176 standard deviations of currently believed true Gamma model. This sample from assumed model is not unbelievable nor unexpected, just higher bounded of expected.*

*In both cases we get Ga(57,22).*

*2(b) I understand we have to do what I’ve already done in first part of 2(a), and by:*

*samps2=jags.samples(jmodel2,"lambda",n.iter=1000)*

*So we can reuse it and make it solution of 2(b), and beginning of 3(a) file.*

#same as previous inferences from this reusable model#

- - - -

#if we use older prior and full dataset, we obtain the same result. It mathematically makes sense, and I obtained the same distributions in R/Jaggs.

misprints = list(y=c(3,4,2,1,2,3, 2, 2, 3, 5, 2, 5, 6, 4, 3, 1),lambdap = 9, r=6)

#same algorithm as first#

#either plot both or get lambda/r, or other parameters of both

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

samps2=jags.samples(jmodel,"lambda",n.iter=1000) # create 1000 samples from misprints model poster

3.

n\_vectors=1000

sampslambda=jags.samples(jmodel2,"lambda",n.iter=n\_vectors) #previously randomly chosen lambdas from Gamma

simulations = as.data.frame(); #first created empty twodimensional data frame

# because can behave similar to ArrayList<int Arrays[16]> of final max length 1000

for (lambdas\_index in 1:n\_vectors){

inits3 = list(labdap = sampslambda[lambdas\_index], r=12) #use each lambda as first guess

jmodel3each=jags.model(file="misprints.model", data=misprints2a, inits3)

samps3each=jags.samples(jmodel3each,"lambda",n.iter=16)

#vector Simulation, append it to dataframe simulation

simulations <- rbind(simulations,samps3each)

}

#we generated 1000 simulations of vectors of 16 examples of how observed data with given 'assumed true' lambda could look like

#probably it would be cleaner to loop inside model file, by utilizing 2 for loops, outside loop looping through vector/datasets/models, and inside loop generating posterior values of Gamma nature, but I’ve opted to do this this way for convenience, and it does it’s intended functionality.

#now we calculate 1000 quadruple summary statistics- for each vector of simulated 16 observations

summary\_stats = as.data.frame(); #or pass everything as argument here, once it’s created

s\_s\_column\_names = c("maximum", "minimum", "mean", "std deviation")

colnames(summary\_stats) <- s\_s\_column\_names

for (each\_vector\_i in 1:n\_vectors){

#for each vector under next index, using full length 16, calculate 'local', 'temp' summary statistics

maximum <- max(simulations[each\_vector\_i,])

min <- min(simulations[each\_vector\_i,])

mean <- mean(simulations[each\_vector\_i,])

std\_dev <- sd(simulations[each\_vector\_i,])

#creating vectors out of single temporary ss that will be appended as rows to summary\_stats

s\_s\_each <- c(maximum,minimum,mean,std\_dev)

summary\_stats <- rbind(summary\_stats, s\_s\_each)

}

#creating tables to aid creation of barplots

counts\_max <- table(summary\_stats$maximum)

counts\_min <- table(summary\_stats$minimum)

#plotting 1 of 4 summary statistics datasets at a time

barplot(counts\_max)

barplot(counts\_min)

hist(summary\_stats$mean, breaks= 10)

hist(summary\_stats$std\_dev, breaks= 10)

3(b)

Mean has almost regular shape- the values of the each of the means with maximal densities (2;3.5) are skewed towards zero. Thus, more often there are more pages with very little errors, than pages with more errors than in standard case. The observed lambda is to the right of the ‘peak’ of the bellshape, so it’s a little higher than the most common cases of mean of lambdas generated by computer simulation. We might be observing a slightly overestimated lambda, having taken under consideration we checked 1000 possibilities and usually lambda is lower, but there is a minute possibility difference might be not high enough to not come from random nature of those checks.

Maximal values are most often between 4 and 7. There is little difference between two maximal numbers of misprints observed- 5 and 6. Since maximum, unlike mean, is always a |Natural number, it is correct to assume our observed mean is in accordance with the true mean, even if it doesn’t come from counts of the very first max.There are single outliers- high max like 11, while the minimal maximas are more clumped towards the center values, but that is natural, taking into consideration that Poisson distribution’s nature is to generate high unexpected values, in rare circumstances with P(rare)->0. Small maxima happen rarely naturally, because each of as many as 16 random trials in vector have a potential to override previous small maximum, bringing it up, and we expect to get 3 errors. With so many trials, it is unlikely to never reach expectation or at least just above that.

Minimal values are mostly 1, sometimes 2, rather not often 0- my simulated Gamma rarely reached below 1.0. Exceptions are very rare, since even the Poisson mean is low, and density function of vector values is skewed towards the minimum, so we’re expecting a small value, so the minimum is even smaller. Again, there are many trials with minimums having potential to override each other, but more often repeating each other than in previous case.

My whole simulation’s sigma (all 16000 values together) was around 1/4th, for sqrt of that around half. As expected with law of large numbers, sigmas of much smaller individual datasets (16) would be higher, and precision- lower. Our observation’s standard deviation was around 1.5, continuous Gamma’s(24,12) was 0.4. It was scattered. Mean, that is not a large numer at all, compared to 1-2-3 standard deviations values away from it, so stdev influences the real observed values to be ‘multiplities of mean’ observed in dataset a lot, rather than small percentage changes, if mean was larger. That’s why without running inference it ‘feels’ scattered, but just like other individual 16-float datasets.

