Gibbs Sample for Zero-inflated Poisson model

Patrick Graham

4 October 2021

Backround

We use an example from the Institute for Digital Research and Education at UCLA https://stats.idre.ucla.edu (https://stats.idre.ucla.edu)

The description given there is ``The state wildlife biologists want to model how many fish are being caught by fishers at a state park. Visitors are asked how long they stayed, how many people were in the group, were there children in the group and how many fish were caught. Some visitors do not fish, but there is no data on whether a person fished or not. Some visitors who did fish did not catch any fish so there are excess zeros in the data because of the people that did not fish."

The data contain the response of 250 groups of park visitors who participated in the survey.

Preliminaries

Set-up packages needed

```
## Loading required package: MASS
```

```
## Loading required package: Matrix
```

```
## Loading required package: lme4
```

```
##
## arm (Version 1.11-1, built: 2020-4-27)
```

```
## Working directory is C:/Users/Patrick Graham/Documents/Patrick/Stat314-2021/code
```

```
## Warning: package 'mvtnorm' was built under R version 4.0.3
```

```
##
## Attaching package: 'coda'
```

```
## The following object is masked from 'package:arm':
##
##     traceplot
```

```
## [1] "C:/Users/Patrick Graham/Documents/Patrick/Stat314-2021/code"
```

Set up a function for the Gelman-Rubin diagnostic and effective Monte Carlo sample size.

```
## implement the chunked  version  of the Gelman-Rubin convergence diagnostic
## only  works  for  a scalar parameter and assumes the posterior sample
## is organised  as as npost by nchains matrix
##where  npost  is the simulation sample size after the burn-in has been
## discarded.

GRchunk <-  function(post){
  require(coda)

  possize <- nrow(post)

  n1 <- round(possize/2)  ##size of first chunk

  chunk1 <- post[1:n1,]
  chunk2 <- post[(round(possize/2)+1):possize,]

  post_chunked <- cbind(chunk1,chunk2)

    ##obtain the chain means
  chain_mean <- colMeans(post)
  ##obtain the within chain variance
  chain_sd <- apply(post,MARGIN=2,FUN=sd)
  chain_var <- chain_sd^2
  W = mean(chain_var)
  ##get between chain variance

  possize_chunked <- nrow(post_chunked)
  B <- possize_chunked*(sd(chain_mean))^2

  ##Now build the components of the Gelman-Rubin statistic

  Vplus <- ((possize_chunked-1)/possize_chunked) * W + (1/possize_chunked)*B
  Rhat <- sqrt(Vplus/W)
  Rhat

  ##compute  estimated effective sample size, using Coda

  post.mcmc <- coda::as.mcmc(post_chunked )

  codaNeff_chains  <- coda::effectiveSize(post.mcmc)

  codaNeff <- sum(codaNeff_chains)

  outlist <- list(Rhat=Rhat,codaNeff=codaNeff,codaNeff_chains=codaNeff_chains)
  return(outlist)

}
```

Read in the data and check the basic structure. We assume the data is in a sub-folder of the working director called data

```
fishdata <- read.csv("data/fish.csv")
```

If the data are in the working directory unhash and run the following commands

```
#fishdata <- read.csv("data/fish.csv") #assuming the data  is in the current wd
```

```
str(fishdata)
```

```
## 'data.frame':    250 obs. of  8 variables:
##  $ nofish  : int  1 0 0 0 0 0 0 0 1 0 ...
##  $ livebait: int  0 1 1 1 1 1 1 1 0 1 ...
##  $ camper  : int  0 1 0 1 0 1 0 0 1 1 ...
##  $ persons : int  1 1 1 2 1 4 3 4 3 1 ...
##  $ child   : int  0 0 0 1 0 2 1 3 2 0 ...
##  $ xb      : num  -0.896 -0.558 -0.402 -0.956 0.437 ...
##  $ zg      : num  3.05 1.746 0.28 -0.602 0.528 ...
##  $ count   : int  0 0 0 0 1 0 0 0 0 1 ...
```

set-up some variables as factors - this code fragment is direct from IDRE

```
fishdata <- within(fishdata, {
  nofish <- factor(nofish)
  livebait <- factor(livebait)
  camper <- factor(camper)
})
```

Take a look at some basic summaries

```
summary(fishdata)
```

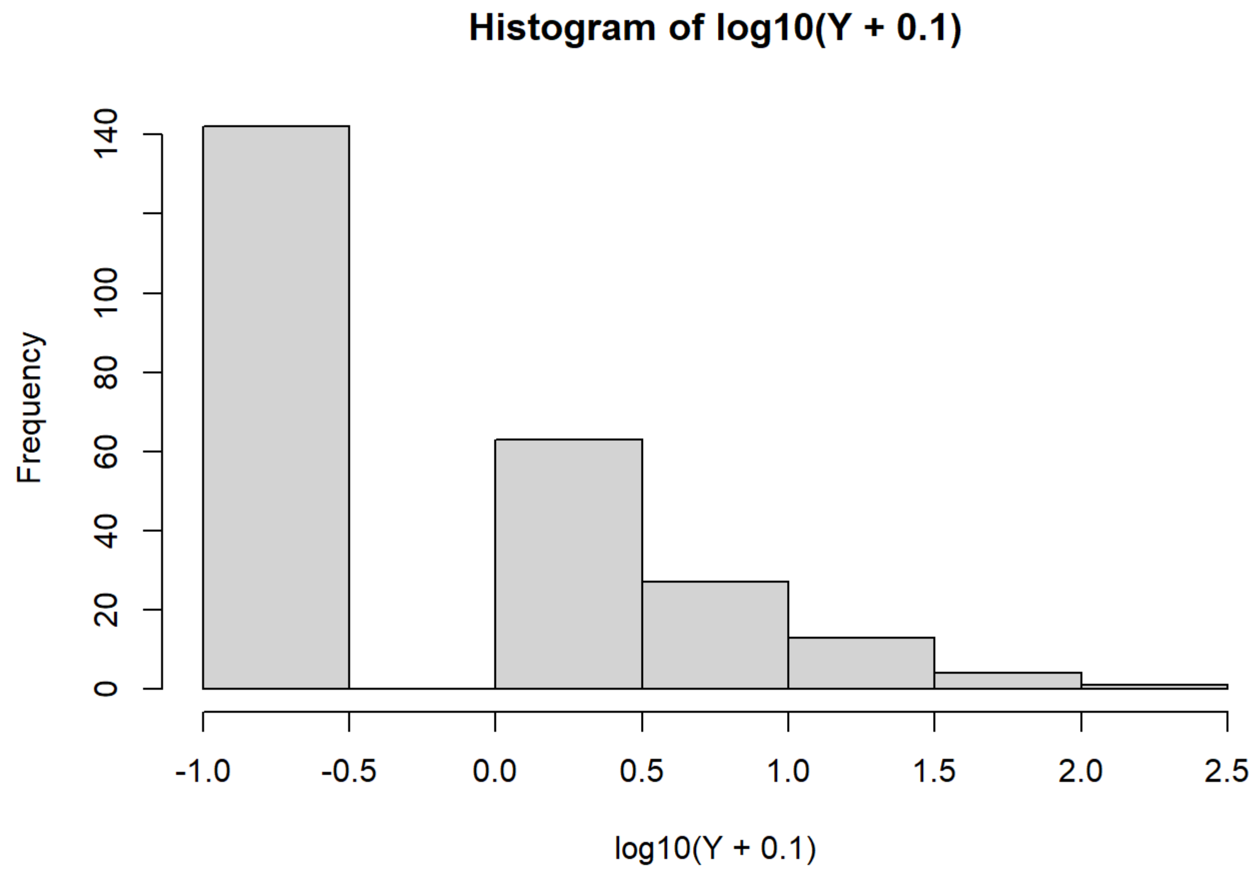
```
##  nofish  livebait camper     persons          child              xb
##  0:176   0: 34    0:103   Min.   :1.000   Min.   :0.000   Min.   :-3.275050
##  1: 74   1:216    1:147   1st Qu.:2.000   1st Qu.:0.000   1st Qu.: 0.008267
##                           Median :2.000   Median :0.000   Median : 0.954550
##                           Mean   :2.528   Mean   :0.684   Mean   : 0.973796
##                           3rd Qu.:4.000   3rd Qu.:1.000   3rd Qu.: 1.963855
##                           Max.   :4.000   Max.   :3.000   Max.   : 5.352674
##        zg              count
##  Min.   :-5.6259   Min.   :  0.000
##  1st Qu.:-1.2527   1st Qu.:  0.000
##  Median : 0.6051   Median :  0.000
##  Mean   : 0.2523   Mean   :  3.296
##  3rd Qu.: 1.9932   3rd Qu.:  2.000
##  Max.   : 4.2632   Max.   :149.000
```

For this example we will work with just the count variable. It is unclear what the meaning of the “nofish variable is:”

```
Y <- fishdata$count
N <- length(Y)   ###Number of observations

# histogram with x axis in log10 scale - add 0.1 to avoid log(0)

hist(log10(Y+0.1))
```



A zero-inflated Poisson model for the counts

We let Z denote the unobserved variable “fished” coded 1 if a visiting group fished and coded 0 if they did not. We adopt the following model

$$\begin{aligned} Z|\phi &\sim \text{Bernoulli}(\phi), \text{ independently } \text{ for } i \text{ in } 1,\ldots,n \\ [Y|Z=1,\theta] &\sim \text{Poisson}(\theta); \\ \Pr(Y=0|Z=0) &= 1 \\ \phi &\sim \text{Beta}(a,b) \\ \theta &\sim \text{Gamma}(c,d) \end{aligned}$$

Note that given a group does not fish, they will not catch any fish, hence $\Pr(Y=0|Z=0)=1$.

And we make the usual conditional independence assumptions. So, letting $\mathbf{Y}=(Y_1,\ldots,Y_N)$, $\mathbf{Z}=(Z_1,\ldots,Z_N)$

$$\begin{aligned} p(\mathbf{Z}|\phi) &= \prod_i \text{Binomial}(Z_i|\phi) \\ p(\mathbf{Y}|\mathbf{Z},\theta) &= \prod_{i:Z_i=1} \text{Poisson}(Y_i|\theta) \end{aligned}$$

For illustrative purposes we will use uninformative priors, setting $a=1, b=1$ and $c=0.01, d=0.01$, giving

$$\begin{aligned} \phi &\sim \text{Beta}(1,1) \\ \theta &\sim \text{Gamma}(0.01,0.01) \end{aligned}$$

To help keep these prior parameters clear in the code below we rename the parameters of the prior for θ as follows

$$
\begin{aligned}
c &\rightarrow theta\_prior1 = 0.01\\
d &\rightarrow theta\_prior2 = 0.01
\end{aligned}
$$

and the parameters of the prior for ϕ to

$$
\begin{aligned}
a &\rightarrow phi\_prior1 = 1\\
b &\rightarrow phi\_prior2 = 1
\end{aligned}
$$

##Gibbs-sampler

The unknowns in this problem are $(\mathbf{Z}, \theta, \phi)$. We use the Gibbs sampler to compute $p(\mathbf{Z}, \theta, \phi|\mathbf{Y})$. Primary interest is on θ -expected catch given that you fish; but ϕ - probability of fishing is also of some interest. Consequently our real aim is to compute

$$
p(\theta, \phi|\mathbf{Y}) = \int p(\mathbf{Z}, \theta, \phi|\mathbf{Y})\,\mathrm{d}\mathbf{Z}.
$$

Since $\mathbf{Z}$ is a vector of binary varibles, the integral is really a sum over all $\mathbf{Z}$ values.

Having obtained obtained a sample from the joint posterior $p(\mathbf{Z}, \theta, \phi|\mathbf{Y})$ via Gibbs sampling, the Monte Carlo equivalent to integrating over the latent $\mathbf{Z}$ values is simply to ignore them and use the sample of θ, ϕ values for inference for θ and ϕ.

The full conditional (conditional) posterior distributions for this problem are
(i)

$$
\begin{aligned}
p(\theta|\mathbf{Z}, \phi, \mathbf{Y}) &\propto p(\mathbf{Y}|\mathbf{Z}\theta)p(\theta)\\
&\propto \left[\prod_{i:Z_i=1} \text{Poisson}(Y_i|\theta)\right] \text{Gamma}(\theta|0.01, 0.01)\\
&= \text{Gamma}\left(\theta| \left(0.01 + \sum_{i:Z_i=1} Y_i\right), \left(0.01 + \sum_{i:Z_i=1} 1\right)\right)
\end{aligned}
$$

(ii)

$$
\begin{aligned}
p(\phi|\mathbf{Z}, \theta, \mathbf{Y}) &\propto p(\mathbf{Y}, \mathbf{Z}|\theta, \phi)p(\theta)p(\phi)\\
&= p(\mathbf{Y}|\mathbf{Z}, \theta)p(\mathbf{Z}|\phi)p(\theta)p(\phi)\\
&\propto p(\mathbf{Z}|\phi)p(\phi)\\
&= \left[\prod_{i} \text{Bernoulli}(Z_i|\phi)\right] \text{Beta}(\theta|1, 1)\\
&= \text{Beta}\left(\theta| \left(1 + \sum_i Z_i\right), \left(1 + N - \sum_i Z_i\right)\right)
\end{aligned}
$$

(iii) $p(\mathbf{Z}|\theta, \phi, \mathbf{Y})$
Under our conditional independence modelling assumptions (given model parameters)

$$
p(\mathbf{Z}|\theta, \phi, \mathbf{Y}) = \prod_i p(Z_i|\theta, \phi, Y_i)).
$$

That is, given $(\theta, \phi, \mathbf{Y})$ we can can generate the individual unobserved Z values independently. There are really only two cases to consider, depending on whether the observed count was greater than zero or equal to zero.

Recall $\Pr(Y=0|Z=0)=1$ since no fish can be caught if no fishing is done. Logically, if $Y>1$ then it must be the case that $Z=1$, (If fish were caught some fishing must have been done; that is

$$\Pr(Z=1|\theta,\phi,Y=y)=1;\ y>0$$

We can show this mathematically

$$\begin{aligned}\Pr(Z=1|Y=y,\theta,\phi) &= 1-\Pr(Z=0|Y=y,\theta,\phi)\\ &= 1-\frac{\Pr(Y=y|Z=0,\theta,\phi)\Pr(Z=0|\phi)}{\Pr(Y=y|Z=0,\theta,\phi)\Pr(Z=0|\phi)+\Pr(Y=y|Z=1,\theta)\Pr(Z=1|\phi)}\\ &= 1-0;\ y>0\end{aligned}$$

since $\Pr(Y=y|Z=0,\theta,\phi)=0;\ y>0$.

So if $Y_i>0$ we should assign $Z_i=1$. The only interesting case is therefore:

$$\begin{aligned}\Pr(Z=1|Y=0,\theta,\phi) &= \frac{\Pr(Y=0|Z=1|\theta)\Pr(Z=1|\phi)}{\Pr(Y=0|Z=1|\theta)\Pr(Z=1|\phi)+\Pr(Y=0|Z=0)\Pr(Z=0|\phi)}\\ &= \frac{\text{Poisson}(0|\theta)\phi}{\text{Poisson}(0|\theta)\phi+1\times(1-\phi)}\\ &= \frac{\exp(-\theta)\phi}{\exp(-\theta)\phi+(1-\phi)}\end{aligned}$$

Consequently to simulate $p(\mathbf{Z}|\theta,\phi,\mathbf{Y})$ we

1. set $Z_i=1$ if $Y_i=1$
2. draw Z_i as $\text{Bernoulli}\left(\frac{\exp(-\theta)\phi}{\exp(-\theta)\phi+(1-\phi)}\right)$ if $Y_i=0$.

Obviously the details of the derivations of the full conditional (conditional posterior distributions) are specific to this problem. However in problems of ``the missing data type" it does usually work out that the full conditionals for the parameters given the simulated missing data are straightforward to derive, whereas a little more work is required to derive the full conditional for the missing data.

```
## set priors
theta_prior1 <- 0.01
theta_prior2 <- 0.01

phi_prior1 <- 1
phi_prior2 <- 1

nchains <- 5
nsim <- 2000
burnin <- 1000

store_phi <- matrix(NA,nrow=nsim,ncol=nchains)
store_theta <- matrix(NA, nrow=nsim,ncol=nchains)
```

Note we are only storing the generated values for ϕ and θ. We could also store $\mathbf{Z}$ but since the Z values are not of intrinsic interest for this particular analysis we do not store them.

```
#starting values
##simple approach for theta - assume ordinary Poisson gamma model;
##unlikely to be that accurate

for (j in 1:nchains) {

  ##starting value for theta
theta <- rgamma(n=1,shape=sum(Y)+theta_prior1,rate=(N+theta_prior2))

##Could / should use a smaller N above to get an
#overdispersed approximation

##No obvious approximation for beta so we just draw from the prior

##starting value for phi
phi <- rbeta(n=1,shape1=1,shape2=1)

 for (i in 1:nsim) {

  ##Simulate the Z values

   ###everyone with Y > 0 must have Z=1 so we only need to calculate
   ##obtain Pr(Z=1|Y=0)

   pZ_1 <- (dpois(0,theta)*phi) / (dpois(0,theta)*phi + 1-phi)

   pZ <- (Y>0) + (Y==0)*pZ_1  ##this is now a vector of length N
                              ###if (Y > 0) pZ=1; else pZ = pZ_1

   Z <- rbinom(n=N,prob=pZ,size=1)  ##vector of 0 or 1 indicating fishing
                                    ##statu

   ##update theta

   ##only learn about theta from the Z=1 group (fishers) so subset out this
   ##group
    Y1 <- Y[Z==1]
    sumY1 <- sum(Y1)

    theta <- rgamma(n=1,shape=(sumY1+theta_prior1),
                        rate=(length(Y1)+theta_prior2) )

   ###Update phi

   phi <- rbeta(n=1,shape1=(sum(Z)+phi_prior1),
                    shape2=(N-sum(Z)+phi_prior2) )


  ###store phi and theta
  store_phi[i,j ] <- phi
  store_theta[i,j] <- theta
 }
}
```

Now check for convergence

Discard burn-in samples:

```
post_theta <- store_theta[(burnin+1):nsim,]
post_phi <-  store_phi[(burnin+1):nsim,]
```

Gelman-Rubin diagnostics for theta

```
GRtheta <- GRchunk(post_theta)

GRtheta$Rhat
```

```
## [1] 0.9993479
```

```
GRtheta$codaNeff
```

```
## [1] 5142.301
```

```
GRtheta$codaNeff_chains
```

```
##     var1     var2     var3     var4     var5     var6     var7     var8 
## 500.0000 422.9462 500.0000 573.3479 500.0000 500.0000 500.0000 567.7893 
##     var9    var10 
## 578.2173 500.0000
```

Gelman-Rubin diagnostics for ϕ

```
GRphi <- GRchunk(post_phi)

GRphi$Rhat
```

```
## [1] 0.9990513
```

```
GRphi$codaNeff
```

```
## [1] 5081.363
```

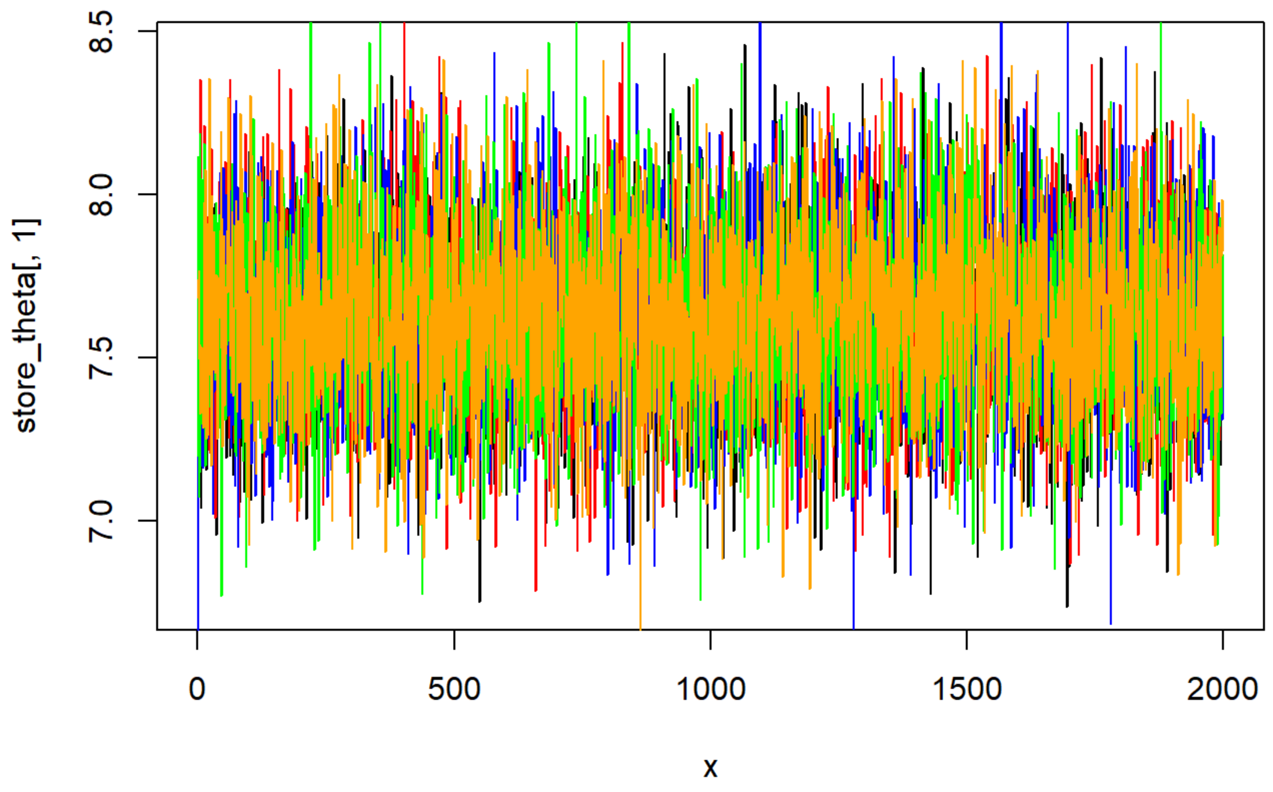
```
GRphi$codaNeff_chains
```

```
##     var1     var2     var3     var4     var5     var6     var7     var8 
## 500.0000 651.7193 500.0000 500.0000 500.0000 358.1221 500.0000 571.5219 
##     var9    var10 
## 500.0000 500.0000
```

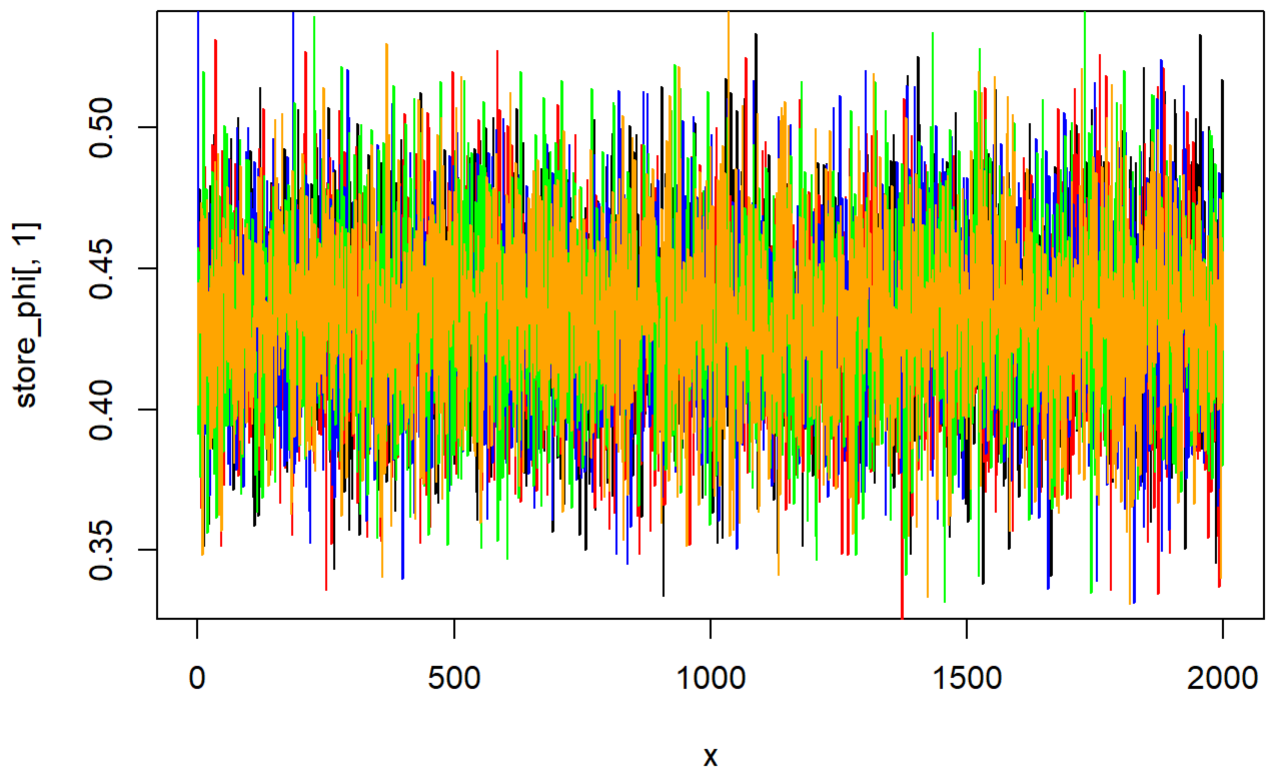
Convergence and effective Monte Carlo sample size look very good.

Take a look at the trace plots

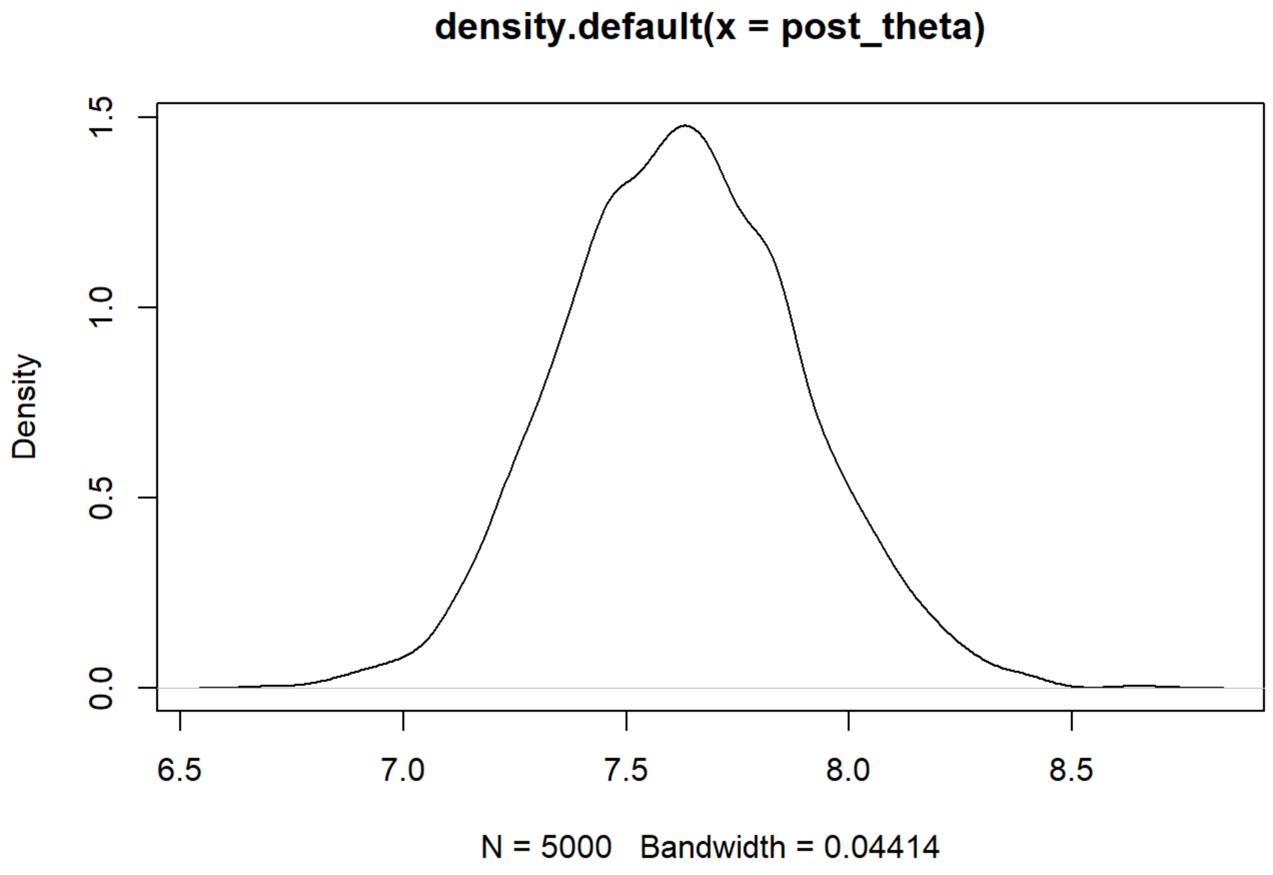
```
#\theta
x <- seq(from=1,to=nsim,by=1)
plot(x,store_theta[,1],type="l")
lines(x,store_theta[,2],col="red")
lines(x,store_theta[,3],col="blue")
lines(x,store_theta[,4],col="green")
lines(x,store_theta[,5],col="orange")
```



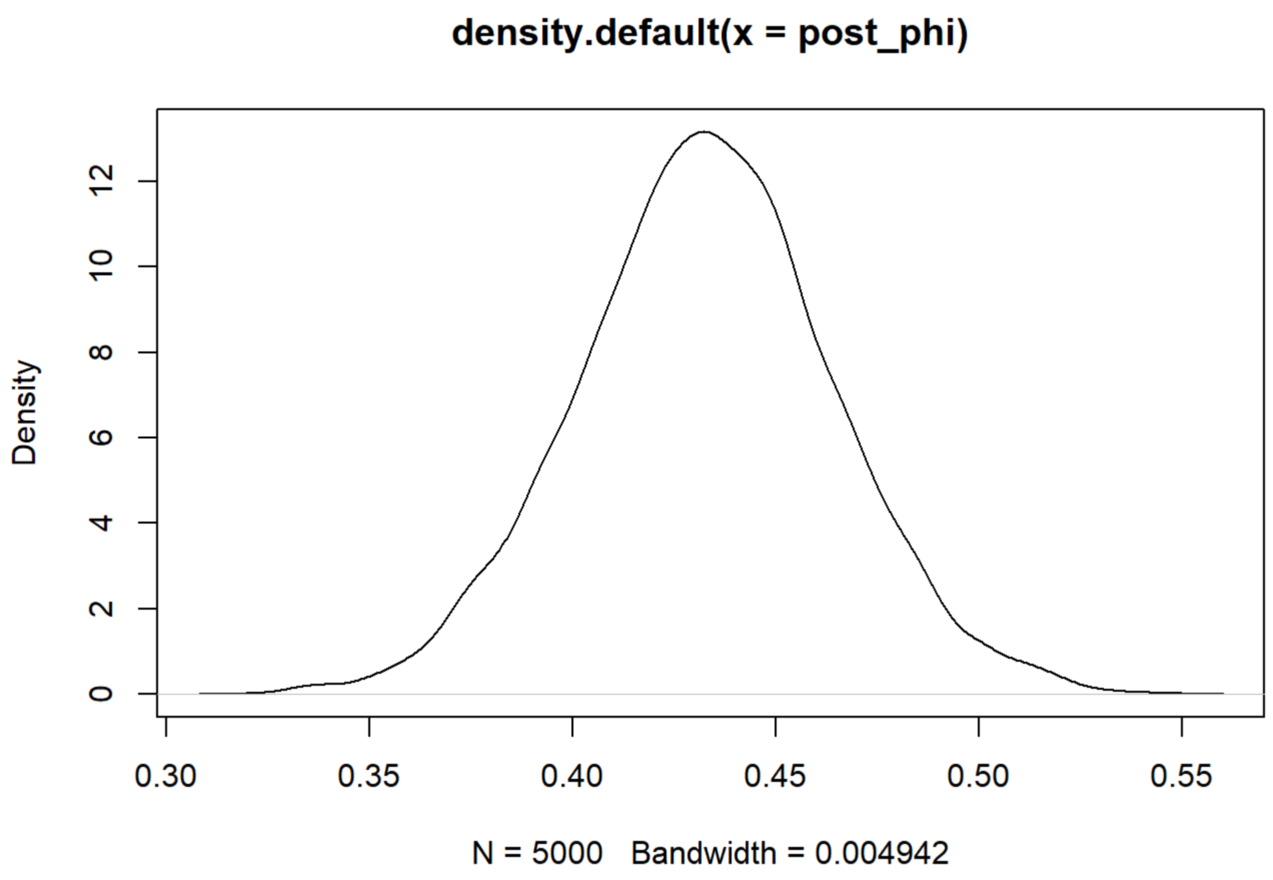
```
# \phi
x <- seq(from=1,to=nsim,by=1)
plot(x,store_phi[,1],type="l")
lines(x,store_phi[,2],col="red")
lines(x,store_phi[,3],col="blue")
lines(x,store_phi[,4],col="green")
lines(x,store_phi[,5],col="orange")
```



```
###posterior density plots
plot(density(post_theta))
```



```
plot(density(post_phi))
```



```
##posterior summary statistics
probs <- c(0.025,0.5,0.975)

quantile(post_theta,probs)
```

```
##     2.5%      50%    97.5% 
## 7.116817 7.620266 8.163439
```

```
##compare this with the naive MLE from a model that does not
#take  account of the fact that some people do not fish (Z=0)
cat("naive MLE", (sum(Y)/N) )
```

```
## naive MLE 3.296
```

Finally look at posterior inference for the proportion of visiting groups that fish

```
quantile(post_phi,probs)
```

```
##      2.5%       50%     97.5% 
## 0.3713149 0.4329025 0.4949896
```

The data is reasonably informative about ϕ even though Z is not fully observed