Bayesian Final Project

Harry Bendekgey
Math 153: Bayesian Statistics

May 11, 2018

1 Hierarchical Beta Models

This summer I will be doing research on using Bayesian methods for election forecasting. This is being done already using various methods and in different countries, and the amount of literature I would need to get through to make progress directly towards that felt outside the scope of a final project. Thus I decided to pick a specific topic to dive deeper into, which knowing more about might prove fruitful over the summer: how to aggregate data across studies, or in the case of elections, across districts.

Hierarchical models are considered the gold standard for combining information across studies. Here, we will consider the hierarchical beta model. We consider:

$$x_i \sim \text{Binom}(p_i, n_i)$$

$$p_i \sim \text{Beta}(\alpha, \beta)$$

$$f(\alpha, \beta) \propto (\alpha + \beta)^{-5/2}$$

The hyperprior on alpha and beta is meant to be non-informative, and was given to us earlier this semester.[1]

Using this model results in shrinkage: we push our predicted values of p_i towards the overall mean, providing us with an estimator so good it makes the maximum likelihood estimator inadmissible. A logical follow-up question is therefore: how much do we push it towards the mean?

One way to measure it is to compare it to the power prior. The power prior works as follows: consider we have data from a similar experiment, call it D_0. The prior on our parameter θ is:

$$f(\theta) \propto L(\theta|D_0)^{a_0} f_0(\theta)$$

Logic dictates that $0 \leq a_0 \leq 1$. In the Beta case, this gives a convenient posterior, namely:

$$f(p|D, D_0, a_0, \alpha, \beta) \sim \text{Beta}(\alpha + x + a_0 x_0, \beta + n - x + a_0(n_0 - x_0))$$

The disadvantage of this is that we need to decide a value of a_0 which encodes how much we "trust" the past data. But we can use this model to our advantage: consider that we run hierarchical modeling on two samples, and compare the result to a power posterior so we can get a measure of how much one study "pulled" the other one. Chen and Ibrahim found that in the case of normal data, there exists a mapping between hierarchical models with two samples and specific values of a_0.[2] I could not derive any such mapping for the Beta case because the marginal distributions of α and β are not of recognizable form. To continue with lab 5, I will use Baseball data.

[1] http://www.stat.cmu.edu/ brian/724/week06/lec15-mcmc2.pdf
[2] https://projecteuclid.org/download/pdf_1/euclid.ba/1340371052

1.1 Baseball Data Simulation

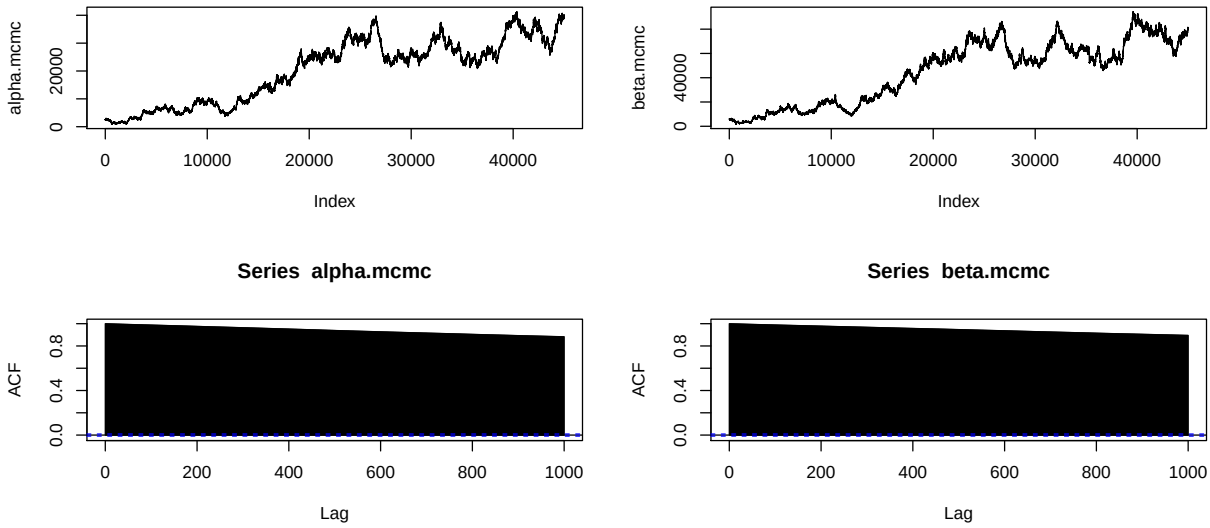
In 2017 Buster Posey got 158 hits out of 494 at-bats. So far in the 2018 season, he's 24 for 88. I want to estimate his true batting average this season, using a hierarchical model with these two samples. I'm ultimately interested in the posterior marginal of this season's batting average, which will look like the likelihood function pulled towards his previous batting average (he's batting 0.273 now and was 0.320 last season).

However, I wasn't particularly happy with the way the algorithm behaved in Lab 5. I decided to limit α and β to be integers. Then we recognize the kernel of the posterior conditional:

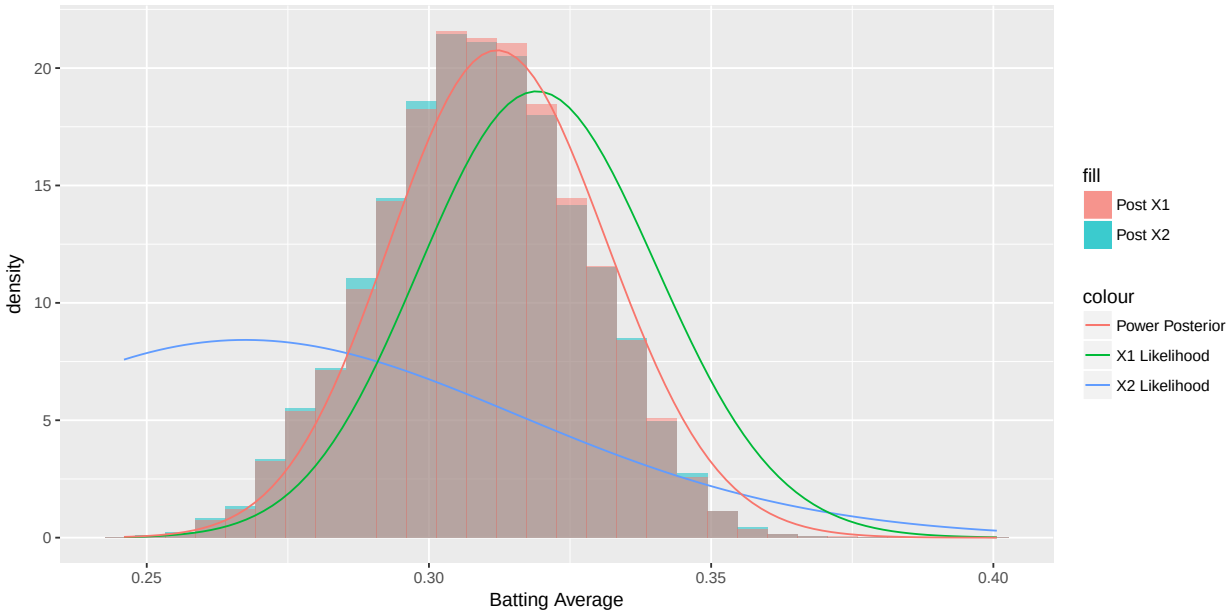
$$f(\alpha, \beta, p_0, p_1 | D_0, D_1) \propto L(p_0|x_0)L(p_1|x_1)L(\alpha, \beta|p_0)L(\alpha, \beta|p_1)f(\alpha, \beta)$$

$$f(\alpha|\beta, p_0, p_1) \propto L(\alpha, \beta|p_0, p_1)f(\alpha, \beta) \propto \left(\frac{(\alpha+\beta-1)!}{(\alpha-1)!}\right)^2 p_0^\alpha p_1^\alpha$$

...if we let $f(\alpha, \beta) \propto 1$ (a weakly informative prior, just to make the math more convenient). Thus $(\alpha - 1) \sim \text{NBinom}(\beta + 1, (1 - p_0)) \cdot \text{NBinom}(\beta + 1, (1 - p_1))$ if we use the R intepretation of the negative binomial, where the first argument is the target for the number of **successful** trials and the second argument is the probability of **success** in each trial). Symmetrically, $(\beta - 1) \sim \text{NBinom}(\alpha + 1, p_0) \cdot \text{NBinom}(\beta + 1, p_1)$. We use the Metropolis-Hastings Algorithm to simulate on Buster Posey's data:



This doesn't look like a converging chain. Let's examine the posterior marginal of p_0 and p_1:



I have overlayed the power posterior where $a_0 = 1$ to show that these results are consistent with the claim that $p_0 = p_1$.

1.2 Non-integrable Likelihoods

I conjecture that the values of α and β here are going to infinity. In order to conceptualize this, imagine the normal hierarchical model:

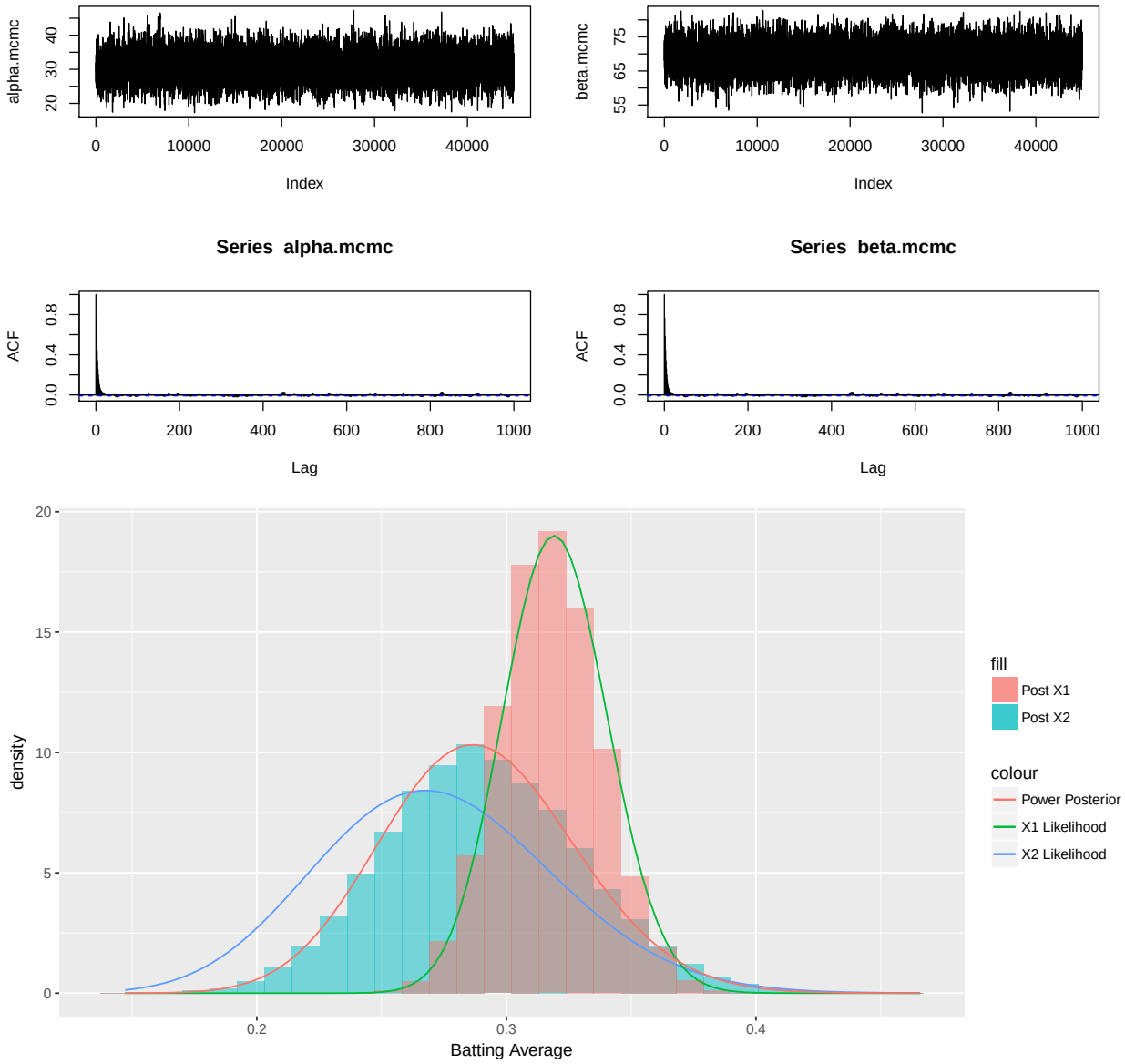
$$f(\mu, \sigma^2, \theta|\mathbf{x}) \propto L(\theta|\mathbf{x})L(\mu, \sigma^2|\theta)f(\mu, \sigma^2)$$

There is a problem here. As $\theta_i \to \mu$ and $\sigma^2 \to 0$, the second term on the right increases to infinity while the first and third terms converge to finite non-zero values. Therefore this posterior contains no maximum, and is therefore non-integrable, rendering our analysis of it useless.

The same analysis works for our Beta hierarchical model. If we are looking at the subregion of our posterior where $\frac{\alpha}{\alpha+\beta} = p_0 = p_1$, we realize that this conditional is a monotonically increasing function of $\alpha + \beta$. It is impossible for a function that is monotonically increasing on the positive reals to be a density.

Because the Gibbs sampler doesn't care about the normalizing constant, we might not realize that this problem exists if we don't get "trapped" in this region. For example, if we had data that didn't look at all like $p_0 = p_1$, it would still be true that if we came up with an α, β large enough, and a $\mathbf{p}$ conforming enough, $L(\alpha, \beta|\mathbf{p})$ would get large enough to overpower the other terms of the posterior. Thus our MCMC results look valid, but are not coming from a valid density.

How do we overcome this? One solution would be to fix $\alpha + \beta$. In some ways, this is akin to fixing σ^2 in the normal data case. To simulate this, I return to the random walk used in Lab 5 but set $\psi = \alpha + \beta$ and therefore generate β deterministically at every step from α:



I have overlayed the power posterior with $a_0 = 0.1$ for reference.

This method is certainly mixing better! But it comes with advantages and disadvantages. For example, if we set ψ too small, we can end up in a situation where the model expects a more dispersed set of p values than it actually has, which leads to the opposite of shrinkage.

Really, the main disadvantage is this: we're back to picking an arbitrary parameter. Picking ψ is theoretically no different than picking a_0. We're still encoding subjective information in a model that is supposed to be the one place Bayesians avoid subjectivity, by letting the prior be learned from the data.

I don't have a big problem with this in the election forecasting case. I don't expect partisanship to suddenly dissipate (or suddenly expand) over the course of two years. I'm happy fitting a Beta (or rather Dirichlet) distribution to the the way different districts voted in 2016, and use that to calculate ψ for the 2018 election cycle. This is akin to setting σ^2 based on previous data and expecting that only μ has changed. I'm primarily interested in the way the country swings as a whole based on the national political environment.

Finally, I would like to defend the immense importance of subjectivity when modelling something so rife with bias as polls and election forecasts. The most common argument against Bayesian statistics is that priors can be arbitrary. I will argue in the following section that sometimes subjectivity is important, and in some situations is the only thing that allows Bayesian methods to work.

2 College Sexual Assault

Consider the following model:

$$x_i \sim \text{Binom}(n_i, p_i)$$
$$p_i \sim \text{Beta}(\alpha, \beta)$$
$$n_i \sim \text{Pois}(\lambda)$$
$$\lambda \sim \text{Gamma}(\gamma, \delta)$$
$$f(\alpha, \beta) \propto 1$$

This is a more complex hierarchical model. If we examine the posterior conditionals, we see:

$$p_i|x_i, n_i, \alpha, \beta \sim \text{Beta}(\alpha + x_i, \beta + n_i - x_i)$$
$$(n_i - x_i)|p_i, \lambda \sim \text{Pois}(\lambda(1-p))$$
$$\lambda|x_i, n_i \sim \text{Gamma}(\sum(n_i - x_i) + \gamma, J + \delta)$$

Where J is the total number of Binomial trials. Again, the posterior distribution on α and β are not of recognizable form.

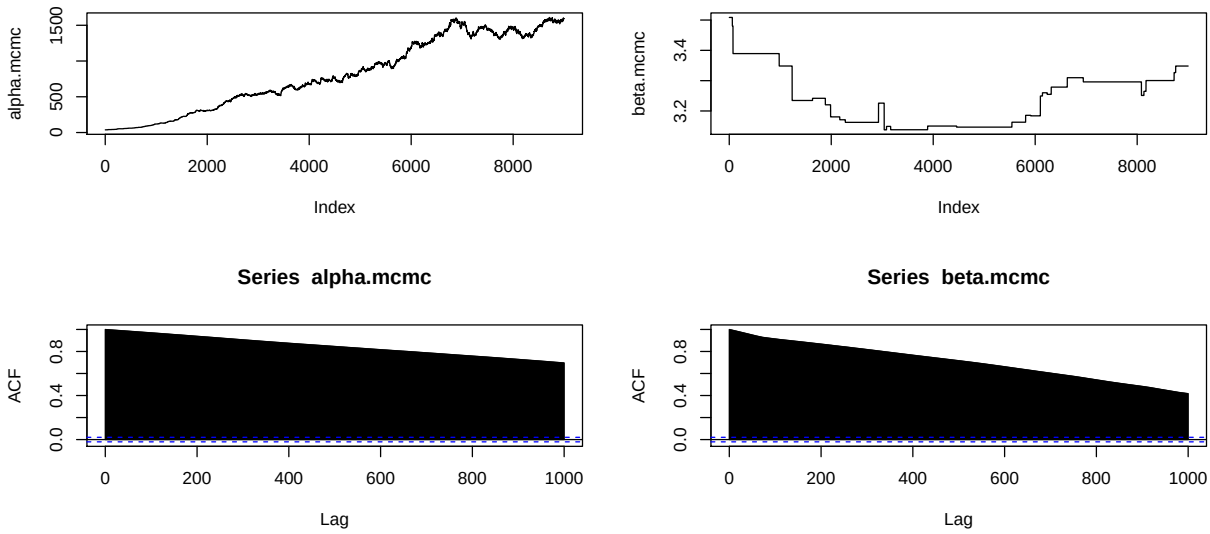
I will use this model on data about sexual assaults on college campuses. We assume that the actual number of sexual assaults on campuses follow a Poisson distribution, and the proportion of these that get reported function like a binomial trial with report probability p. Our primary interest is with $n_i - x_i$, namely the number of sexual assaults that go unreported.

2.1 Simulation

The US Department of Education has a Campus Safety and Security Database[3]. I limited my search to on-campus rapes. This database has the reported values for 6397 colleges in 2016. It might be concerning to us that we are assuming the number of sexual assaults are Poisson distributed across campuses with no regard for the number of students enrolled, but it will become evident that accounting for that will not change any of the results below.

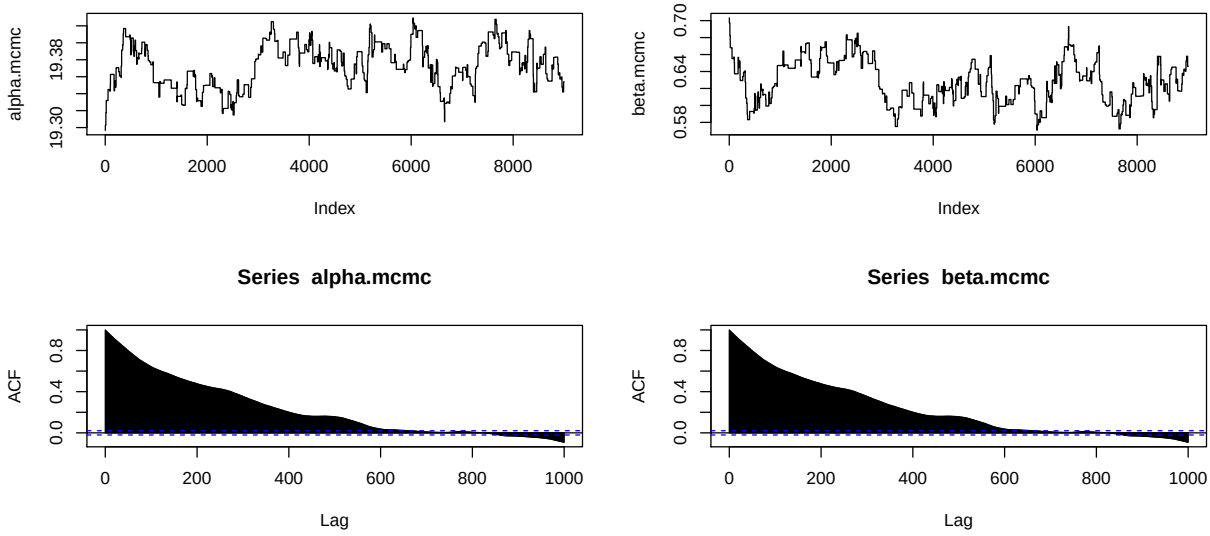
First, I will simulate using the random walk algorithm from Lab 5. I had to use a trick to get this to work; because we're dealing with a lot of data points, I've been using the natural log of each term in the Metropolis Hastings Ratio. In order to keep $\ln(p)$ and $\ln(1-p)$ finite, I limited values of p to be between 0.0001 and 0.9999. I also set $\gamma = \delta = 0$ which gives us an uninformative prior on λ, albeit an improper prior.

[3] https://ope.ed.gov/campussafety/#/customdata/datafiltered



We seem to be experiencing the same problems as in section 1, namely α running off. Interestingly, the mean seems to be converging to 1, meaning that all sexual assaults are reported, which is highly unexpected.

Let's run it again using my algorithm, and a somewhat generous spread of $\psi = 20$.



Fascinatingly, the p-values are all converging to 1. Let's look at our point estimate for the total number of unreported sexual assaults in 2016 across all college campuses.

```
sum(colMeans(n.mcmc) - y)

## [1] 0.03033

mean(lambda)

## [1] 0.003303
```

Unreported sexual assaults don't happen, according to this.

2.2 Garbage in, Garbage out

```
nrow(sa)

## [1] 6397

sa %>% filter(rape == 0) %>% nrow()

## [1] 5224
```

Of the more than six thousand colleges surveyed, over five thousand didn't report a single rape in 2016. This is baffling. Giving this data to the Gibbs sampler resulted in it concluding that sexual assaults just don't happen, and in the very very rare case that they somehow do, they will be reported. The absolute lack of variation in the number of sexual assaults reported is consistent with $p \approx 0$ or $p \approx 1$, and the sampler prefers the latter because it can also set $\lambda \approx 0$ and get really good likelihoods.

This is the principle of "Garbage-in, Garbage-out". It doesn't really matter how sophisticated our model is, or how well the machine learns the prior. If our data is garbage, no amount of mathematical trickery can conquer that.

This is where it is important to have subjective priors. In fact, this is the exact kind of problem that Bayesians should be able to solve which frequentists find insurmountable. I argue that election forecasting is like sexual assault reporting: there's no way to get truly good data free of the myriad of biases that shroud politics. A convincing Datart blog post argues that even a sophisticated model does poorly when it doesn't take into account the biases of each polling company; if you assume that the biases will all even out in the end, you're unlikely to be right.[4]

Thus I launch into my summer research, keen on developing the mathematical tools to forecast, but weary of the idea that some amount of calculus or algorithmic trickery is capable of solving this problem on its own.

3 Code

3.1 Graphing

```
graph.alpha.beta <- function() {
  par(mfrow=c(2,2))
  plot(alpha.mcmc,type="l")
  plot(beta.mcmc,type="l")
  acf(alpha.mcmc,1000)
  acf(beta.mcmc,1000)
}

graph.post <- function(a0) {
  power.alpha <-  a0 * y[1] + y[2]
  power.beta <- n[2] + n[1] * a0 - power.alpha
  theta.df <- data.frame(theta.mcmc)
  ggplot(theta.df) + geom_histogram(alpha=0.5, aes(X2, fill="Post X2", y =..density..)) +
    xlab("Batting Average") +
```

[4] http://datart.com.br/blog/us_2016_elections_en.html

```
    geom_histogram(alpha=0.5, aes(X1, fill="Post X1", y =..density..)) +
    stat_function(fun=function(x) dbeta(x,y[2], n[2]-y[2]), aes(col="X2 Likelihood")) +
    stat_function(fun=function(x) dbeta(x,y[1], n[1]-y[1]), aes(col="X1 Likelihood")) +
    stat_function(fun=function(x) dbeta(x,power.alpha, power.beta), aes(col="Power Posterior"))
}
```

3.2 Negative Binomial Gibbs Sampler

```
draw.thetas <- function(alpha,beta) {
  return(rbeta(J,alpha+y,beta+n-y))
}

draw.alpha.nbinom <- function(alpha,beta,theta) {
  alpha.star <- rnbinom(1,size=beta+1,prob=1-theta[1]) + 1
  ratio <- dnbinom(alpha.star-1,size=beta+1,prob=1-theta[2]) /
    dnbinom(alpha-1,size=beta+1,prob=1-theta[2])
  acc <- ifelse(runif(1) <= ratio,1,0)
  return(ifelse(acc,alpha.star,alpha))
}

draw.beta.nbinom <- function(alpha,beta,theta) {
  beta.star <- rnbinom(1,size=alpha+1,prob=theta[1]) + 1
  ratio <- dnbinom(beta.star-1,size=alpha+1,prob=theta[2]) /
    dnbinom(beta-1,size=alpha+1,prob=theta[2])
  acc <- ifelse(runif(1) <= ratio,1,0)
  return(ifelse(acc,beta.star,beta))
}
```

```
set.seed(4747)
y <- c(158, 24)
n <- c(494, 88)
J <- 2
B <- 15000 #burnout
M <- 45000 #main
MM <- B + M
```

```
alpha <- matrix(NA,MM)
beta <- alpha
theta <- matrix(NA,nrow=MM,ncol=J)

# Initial values for the chain
alpha[1] <- 10
beta[1] <- 10
theta[1,] <- draw.thetas(alpha[1],beta[1])

# MCMC simulation
for (m in 2:MM) {
```

```
  alpha[m] <- draw.alpha.nbinom(alpha[m-1],beta[m-1],theta[m-1,])
  beta[m] <- draw.beta.nbinom(alpha[m],beta[m-1],theta[m-1,])
  theta[m,] <- draw.thetas(alpha[m],beta[m])
}
```

```
good <- (B+1):MM
alpha.mcmc <- alpha[good]
beta.mcmc <- beta[good]
theta.mcmc <- theta[good,]
graph.alpha.beta()
graph.post(a0 = 1)
```

3.3 Fixed Sample Size Gibbs Sampler

```
set.seed(4747)
psi <- 100
sd <- psi/10
draw.alpha <- function(alpha,theta,prop.sd) {
  alpha.star <- rnorm(1,alpha,prop.sd)
  num <- alpha.star*sum(log(theta/(1-theta))) -
    J*(lgamma(psi - alpha.star) + lgamma(alpha.star))
  den <- alpha*sum(log(theta/(1-theta))) -
    J*(lgamma(psi - alpha) + lgamma(alpha))
  acc <- ifelse((log(runif(1))<=num - den)&&(alpha.star>0),1,0)
  return(ifelse(acc,alpha.star,alpha))
}

alpha <- matrix(NA,MM)
beta <- alpha
theta <- matrix(NA,nrow=MM,ncol=J)

# Initial values for the chain
alpha[1] <- psi/2
beta[1] <- psi/2
theta[1,] <- draw.thetas(alpha[1],beta[1])

# MCMC simulation
for (m in 2:MM) {
  alpha[m] <- draw.alpha(alpha[m-1],theta[m-1,],sd)
  beta[m] <- psi - alpha[m]
  theta[m,] <- draw.thetas(alpha[m],beta[m])
}
```

```
good <- (B+1):MM
alpha.mcmc <- alpha[good]
beta.mcmc <- beta[good]
theta.mcmc <- theta[good,]
```

```
graph.alpha.beta()
graph.post(a0 = 0.1)
```

3.4 Sexual Assault

```
set.seed(4747)
sa <- read_csv("~/rstudio/Criminal_Offenses_On_campus.csv") %>%
  select(`Institution name`, Rape) %>%
  group_by(`Institution name`) %>%
  summarise(rape = sum(Rape)) %>%
  data.frame()
y <- sa$rape
J <- length(y)
B <- 1000 #burnout
M <- 9000 #main
MM <- B + M
```

```
log.prior <- function(alpha,beta) {
  {-2.5}*log(alpha + beta)
}

draw.alpha <- function(alpha,beta,theta,prop.sd) {
  alpha.star <- rnorm(1,alpha,prop.sd)
  if (alpha.star <= 0) {
    return(alpha)
  }
  num <- J*(lgamma(alpha.star+beta) - lgamma(alpha.star)) +
    alpha.star*sum(log(theta)) + log.prior(alpha.star,beta)
  den <- J*(lgamma(alpha+beta)      - lgamma(alpha)) +
    alpha     *sum(log(theta)) + log.prior(alpha,beta)
  acc <- ifelse(log(runif(1))<=num - den,1,0)
  return(ifelse(acc,alpha.star,alpha))
}

draw.beta <- function(alpha,beta,theta,prop.sd) {
  beta.star <- rnorm(1,beta,prop.sd)
  if (beta.star <= 0) {
    return(beta)
  }
  num <- J*(lgamma(alpha+beta.star) - lgamma(beta.star)) +
    beta.star*sum(log(1-theta)) + log.prior(alpha,beta.star)
  den <- J*(lgamma(alpha+beta)      - lgamma(beta)) +
    beta     *sum(log(1-theta)) + log.prior(alpha,beta)
  acc <- ifelse(log(runif(1))<=num - den,1,0)
  return(ifelse(acc,beta.star,beta))
}

draw.thetas <- function(alpha,beta,n) {
```

```
  thetas <- rbeta(J,alpha+y,beta+n-y) %>%
    sapply(min, 0.9999) %>%
    sapply(max, 0.0001) %>%
    return()
}

draw.ns <- function(lambda, thetas) {
  return(y + rpois(J, lambda * (1-thetas)))
}

draw.lambda <- function(n) {
  a <- sum(n) - sum(y) + 1
  b <- J
  return(rgamma(1, shape=a, rate=b))
}
```

```
B <- 1000 #burnout
M <- 9000 #main

MM <- B + M

alpha <- matrix(NA,MM)
beta <- alpha
lambda <- beta
theta <- matrix(NA,nrow=MM,ncol=J)
n <- theta

# Initial values for the chain
alpha[1] <- 10
beta[1] <- 10
lambda[1] <- 10
theta[1,] <- draw.thetas(alpha[1],beta[1],max(y))
n[1,] <- draw.ns(lambda[1], theta[1,])

# tuning parameters
alpha.sd <-  9
beta.sd <-   9

# MCMC simulation
for (m in 2:MM) {
  alpha[m] <- draw.alpha(alpha[m-1],beta[m-1],theta[m-1,],alpha.sd)
  beta[m] <- draw.beta(alpha[m],beta[m-1],theta[m-1,],beta.sd)
  lambda[m] <- draw.lambda(n[m-1,])
  theta[m,] <- draw.thetas(alpha[m],beta[m],n[m-1,])
  n[m,] <- draw.ns(lambda[m],theta[m,])
}
```

```
good <- (B+1):MM
alpha.mcmc <- alpha[good]
```

```
beta.mcmc <- beta[good]
theta.mcmc <- theta[good,]
graph.alpha.beta()
```

3.5 Fixed Sample Size Sexual Assault

```
psi <- 20
sd <- psi/100
set.seed(4747)
draw.alpha <- function(alpha,theta,prop.sd) {
  alpha.star <- rnorm(1,alpha,prop.sd)
  if (alpha.star<=0 || alpha.star > psi) {
    return(alpha)
  }
  num <- alpha.star*sum(log(theta) - log(1-theta)) -
    J*(lgamma(psi - alpha.star) + lgamma(alpha.star))
  den <- alpha*sum(log(theta) - log(1-theta)) -
    J*(lgamma(psi - alpha) + lgamma(alpha))
  acc <- ifelse((log(runif(1))<=num - den),1,0)
  return(ifelse(acc,alpha.star,alpha))
}
```

```
alpha <- matrix(NA,MM)
beta <- alpha
lambda <- beta
theta <- matrix(NA,nrow=MM,ncol=J)
n <- theta

# Initial values for the chain
alpha[1] <- 10
beta[1] <- 10
lambda[1] <- 10
theta[1,] <- draw.thetas(alpha[1],beta[1],max(y))
n[1,] <- draw.ns(lambda[1], theta[1,])

# MCMC simulation
for (m in 2:MM) {
  alpha[m] <- draw.alpha(alpha[m-1],theta[m-1,],sd)
  beta[m] <- psi - alpha[m]
  lambda[m] <- draw.lambda(n[m-1,])
  theta[m,] <- draw.thetas(alpha[m],beta[m],n[m-1,])
  n[m,] <- draw.ns(lambda[m],theta[m,])
}
```

```
good <- (B+1):MM
alpha.mcmc <- alpha[good]
beta.mcmc <- beta[good]
```

```
theta.mcmc <- theta[good,]
n.mcmc <- n[good,]
graph.alpha.beta()
```