Introduction to Bayesian Statistics and Markov Chain Monte Carlo Estimation

PSQF 7375: Missing Data Methods

Lecture 06

March 12, 2025

Today's Class

- An introduction to Bayesian statistics:
  - What it is
  - What it does
  - Why people use it

- An introduction to Markov Chain Monte Carlo (MCMC estimation)
  - How it works
  - Features to look for when using MCMC
  - Why people use it

AN INTRODUCTION TO BAYESIAN STATISTICS

Bayesian Statistics: The Basics

- Bayesian statistical analysis refers to the use of models where some or all of the parameters are treated as **random components**
  - Each parameter comes from some type of distribution

- The likelihood function of the data is then augmented with an additional term that represents the likelihood of the **prior distribution** for each parameter
  - Think of this as saying each parameter has a certain likelihood – the height of the prior distribution

- The final estimates are then considered summaries of the **posterior distribution** of the parameter, conditional on the data
  - In practice, we use these estimates to make inferences, just as we have when using the non-Bayesian approaches we have used throughout this class (e.g., maximum likelihood/least squares)

Bayesian Statistics: Why It Is Used

- Bayesian methods get used because the ***relative*** accessibility of one method of estimation (MCMC – to be discussed shortly)

- There are four main reasons why people use MCMC:

1. Missing data
   - Multiple imputation: MCMC is used to estimate model parameters then "impute" data
   - More complicated models for certain types of missing data

2. Lack of software capable of handling large sized analyses
   - Have a zero-inflated negative binomial with 21 multivariate outcomes per 18 time points?

3. New models/generalizations of models not available in software
   - Have a new model?
   - Need a certain link function not in software?

4. Membership in the cult of Bayesians
   - They believe philosophical differences exist between numbers from Bayesian analysis and other types of estimators

Bayesian Statistics: Perceptions and Issues

- The use of Bayesian statistics has been controversial
  - The use of certain prior distributions can produce results that are biased or reflect subjective judgment rather than objective science
- Most MCMC estimation methods are **computationally intensive**
  - Until recently, very few methods available for those who aren't into programming in Fortran, C, or C++
- Understanding of what Bayesian methods are and how they work is limited outside the field of mathematical statistics
  - Especially the case in the educational and social sciences
- Over the past 20 years, Bayesian methods have become widespread – making new models estimable and becoming standard in some social science fields (quantitative psychology and educational measurement)

HOW BAYESIAN METHODS WORK

How Bayesian Statistics Work

- The term Bayesian refers to Thomas Bayes (1701-1761)
  - Formulated Bayes' Theorem

- Bayesian methods rely on Bayes' Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

  - $P(A)$ is the **prior distribution (pdf) of A → WHY THINGS ARE BAYESIAN**
  - $P(B)$ is the **marginal distribution (pdf) of B**
  - $P(B|A)$ is the **conditional distribution (pdf) of B, given A**
  - $P(A|B)$ is the **posterior distribution (pdf) of A, given B**

- Bayes' Theorem Example…

Imagine a patient takes a test for a rare disease (present 1% of the population) that has a 95% accuracy rate…what is the probability the patient actually has the disease?

Bayes' Theorem Example

Imagine a patient takes a test for a rare disease (present 1% of the population) that has a 95% accuracy rate…what is the probability the patient actually has the disease?

- D = the case where the person actually has the disease
- ND = the case where the person does not have the disease
- + = the test for the disease is positive

The question is asking for: P(D|+)

From Bayes' Theorem:

$$P(D|+) = \frac{P(+|D)P(D)}{P(+)}$$

What we know:

$$P(D) = .01$$
$$P(+|D) = .95$$

Back to Distributions

- We don't know $P(+)$ directly from the problem, but we can figure it out if we recall how distributions work:

- $P(+)$ is a marginal distribution
- $P(+|D)$ is a conditional distribution

- We can get to the marginal by summing across the conditional:

$$P(+) = P(+|D)P(D) + P(+|ND)P(ND)$$
$$= .95 * .01 + .05 * .99 = .059$$

- So, to figure out the answer, if a person tests positive for the disease, the **posterior probability** they actually have the disease is:

$$P(D|+) = \frac{P(+|D)P(D)}{P(+)} = \frac{.01 * .99}{.059} = .17$$

A (Perhaps) More Relevant Example

- The old-fashioned Bayes' Theorem example I've found to be difficult to generalize to your actual data, so…

- Imagine you administer an IQ test to a sample of 50 people
  - y_p = person p's IQ test score

- To put this into a linear-models context, the empty model for Y:

$$y_p = \beta_0 + e_p$$

Where $e_p \sim N(0, \sigma_e^2)$

- From this empty model, we know that:
  - β_0 is the mean of the Y (the mean IQ)
  - σ_e^2 is the sample variance of Y
  - The conditional distribution of Y is then: $f(y_p|\beta_0, \sigma_e^2) \sim N(\beta_0, \sigma_e^2)$

Non-Bayesian Analysis

- Up to this point in the class, we have analyzed these data using ML and REML

- For ML, we maximized the joint likelihood of the sample with respect to the two unknown parameters β_0 and σ_e^2

$$L(\beta_0, \sigma_e^2) = \prod_{p=1}^{N} f(y_p|\beta_0, \sigma_e^2) = \prod_{p=1}^{N} \frac{1}{\sqrt{2\pi\sigma_e^2}} \exp\left(-\frac{(y_p - \beta_0)^2}{2\sigma_e^2}\right)$$

- Here, using gls(), I found:

$$\beta_0 = 102.769$$
$$\sigma_e^2 = 239.490$$

- Also, I found:

$$LogL = -207.91$$

Setting up a Bayesian Approach

- The (fully) Bayesian approach would treat each parameter as a random instance from some **prior distribution**

- Let's say you know that this version of the IQ test is supposed to have a mean of 100 and a standard deviation of 15
  - So β_0 should be 100 and σ_e^2 should be 225

- Going a step further, let's say you have seen results for administrations of this test that led you to believe that the mean came from a normal distribution with a SD of 2.13
  - This indicates the prior distribution for the **mean**…or

$$f(\beta_0) \sim N(100, 2.13^2)$$

- Let's also say that you don't really have an idea as for the distribution of the variance, but you have seen it range from 200 to 400, so we can come up with a prior distribution for the **variance** of:

$$f(\sigma_e^2) \sim U(200, 400)$$

- Here the prior is a uniform distribution meaning all values from 200 to 400 are equally likely

More on the Bayesian Approach

- The Bayesian approach is now to seek to find the **posterior distribution** of the parameters given the data:

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right)$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- We can again use Bayes' Theorem (but for continuous parameters):

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right) = \frac{f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0, \sigma_e^2)}{f\left(\boldsymbol{y}_p\right)} = \frac{f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)}{f\left(\boldsymbol{y}_p\right)}$$

- Because $f\left(\boldsymbol{y}_p\right)$ essentially is a constant (which involves integrating across β_0 and σ_e^2 to find its value), this term is often referred to as:

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right) \propto f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)$$

- The symbol $\propto$ is read as "is proportional to" – meaning it is the same as when multiplied by a constant
  - So it is the same for all values of β_0 and σ_e^2

Unpacking the Posterior Distribution

- $f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right)$ is the **conditional distribution** of the data given the parameters – we know this already from our linear model (slide 12)

$$f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) = \prod_{p=1}^{N} f\left(y_p \middle| \beta_0, \sigma_e^2\right) = \prod_{p=1}^{N} \frac{1}{\sqrt{2\pi\sigma_e^2}} \exp\left(-\frac{\left(y_p - \beta_0\right)^2}{2\sigma_e^2}\right)$$

- $f(\beta_0)$ is the **prior distribution** of β_0, which we decided would be $N(100, 2.13^2)$, giving the height of any β_0:

$$f(\beta_0) = \frac{1}{\sqrt{2\pi\sigma_{\beta_0}^2}} \exp\left(-\frac{\left(\beta_0 - \mu_{\beta_0}\right)^2}{2\sigma_{\beta_0}^2}\right)$$

$$= \frac{1}{\sqrt{2\pi * 2.13^2}} \exp\left(-\frac{(\beta_0 - 100)^2}{2 * 2.13^2}\right)$$

Unpacking the Posterior Distribution

- $f(\sigma_e^2)$ is the **prior distribution** of σ_e^2, which we decided would be $\mathrm{U}(200,400)$, giving the height of any value of σ_e^2 as:

$$f(\sigma_e^2) = \frac{1}{b_{\sigma_e^2} - a_{\sigma_e^2}} = \frac{1}{400 - 200} = \frac{1}{200} = .005$$

- Some useful terminology:
  - The parameters of the model (for the data) get prior distributions
  - The prior distributions each have parameters – these parameters are called **hyper-parameters**
  - The hyper-parameters are not estimated in our example, but could be – giving us a case where we would call our priors **empirical priors**
    - **AKA random intercept variance**

Up Next: Estimation (first using non-MCMC)

- Although MCMC is commonly thought of as the only method for Bayesian estimation, there are several other forms

- The form analogous to ML (where the value of the parameters that maximize the likelihood or log-likelihood) is called **Maximum (or Modal) a Posteriori estimation (MAP)**
  - The term modal comes from the maximum point coming at the peak (the mode) of the posterior distribution

- In practice, this functions similar to ML, only instead of maximizing the joint likelihood of the data, we now have to worry about the prior:

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right) = \frac{f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)}{f\left(\boldsymbol{y}_p\right)} \propto f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)$$

- Because it is often more easy to work with, the log of this is often used:

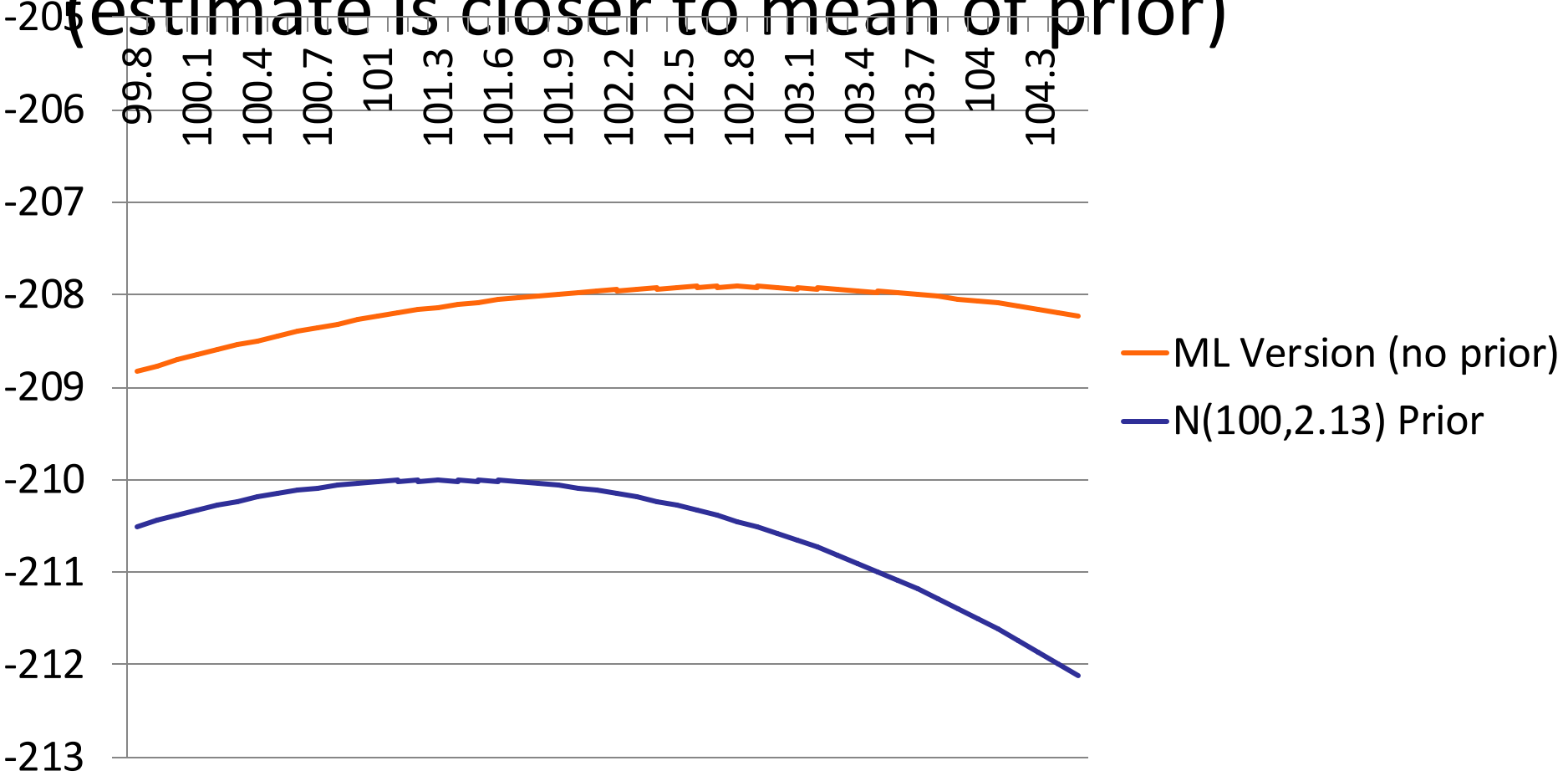
$$\log\left(f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right)\right) \propto \log f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) + \log f(\beta_0) + \log f(\sigma_e^2)$$

Grid Searching for the MAP Estimate of $\boldsymbol{\beta_0}$

- To demonstrate, let's imagine we know $\sigma_e^2 = 239.490$
  - Later we won't know this…when we use MCMC
- We will use Excel to search over a grid of possible values for β_0
- In each, we will use $\log f(\boldsymbol{y}_p|\beta_0) + \log f(\beta_0)$
- As a comparison, we will also search over the ML log likelihood function $\log f(\boldsymbol{y}_p|\beta_0)$

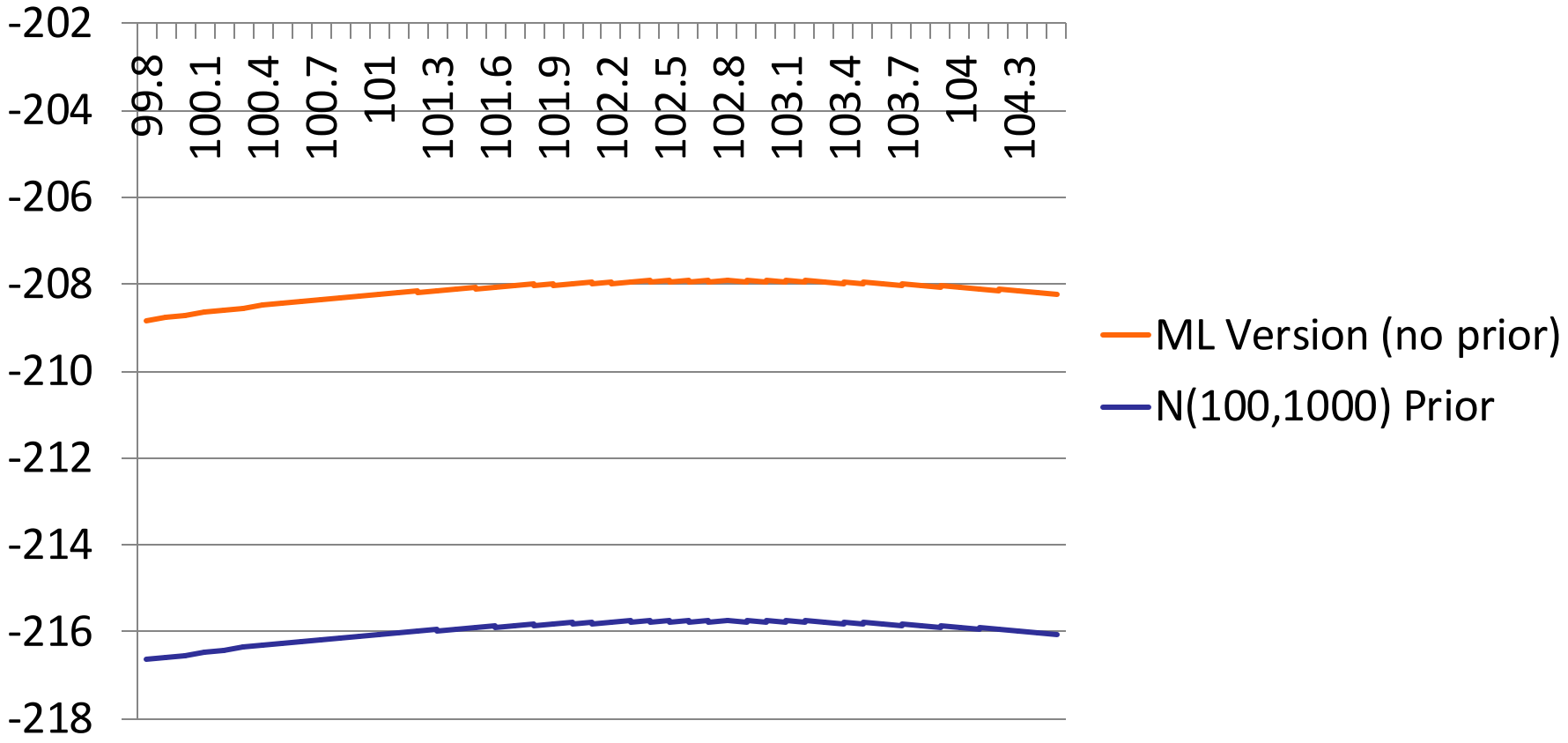
ML v. Prior for β_0 of N(100, 2.13^2)

- Maximum for ML: 102.8
- Maximum for Bayes: 101.4 (estimate is closer to mean of prior)



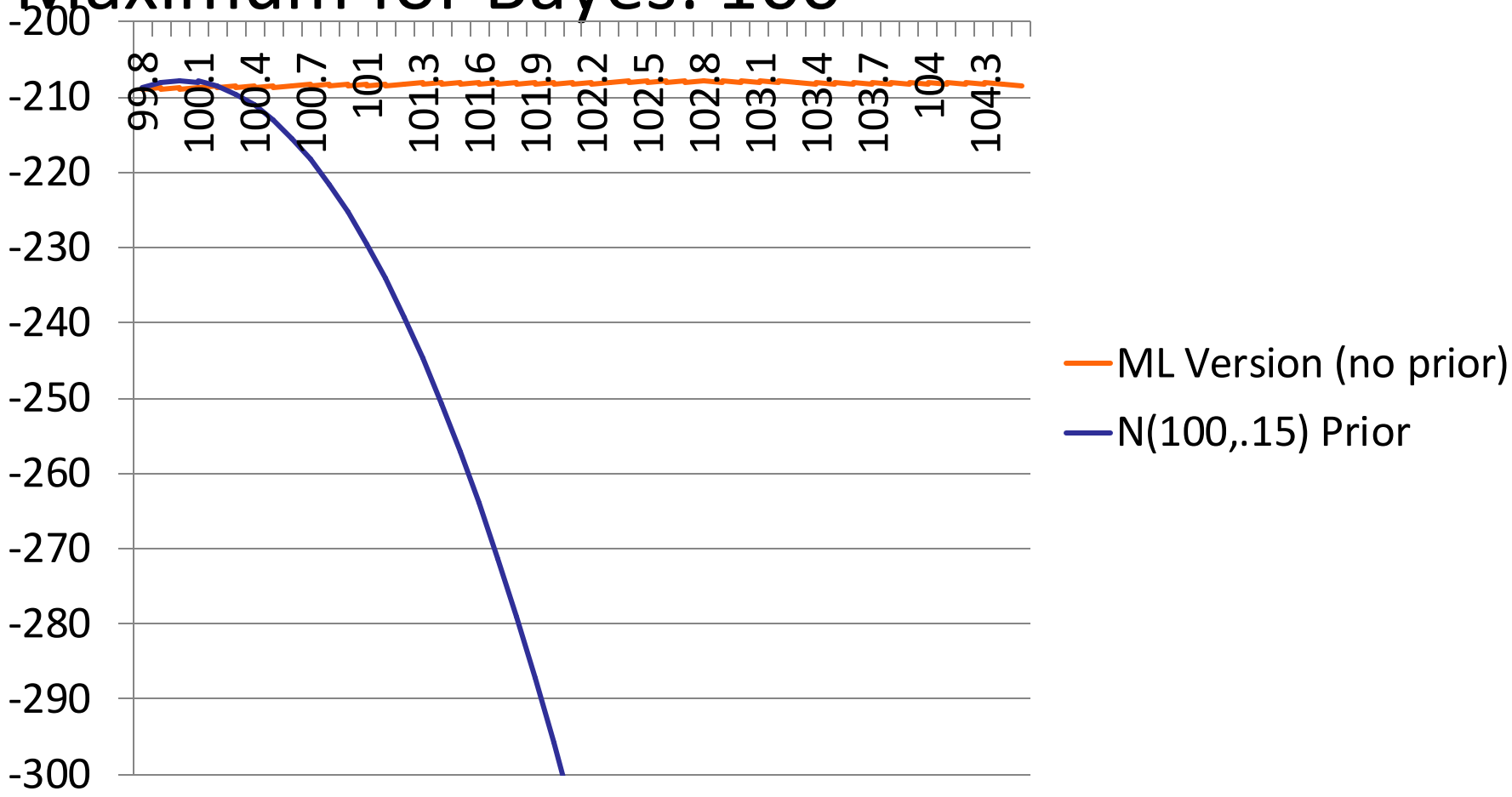
ML vs. Prior for β_0 of N(100, 1000^2)

- Maximum for ML: 102.8
- Maximum for Bayes: 102.8



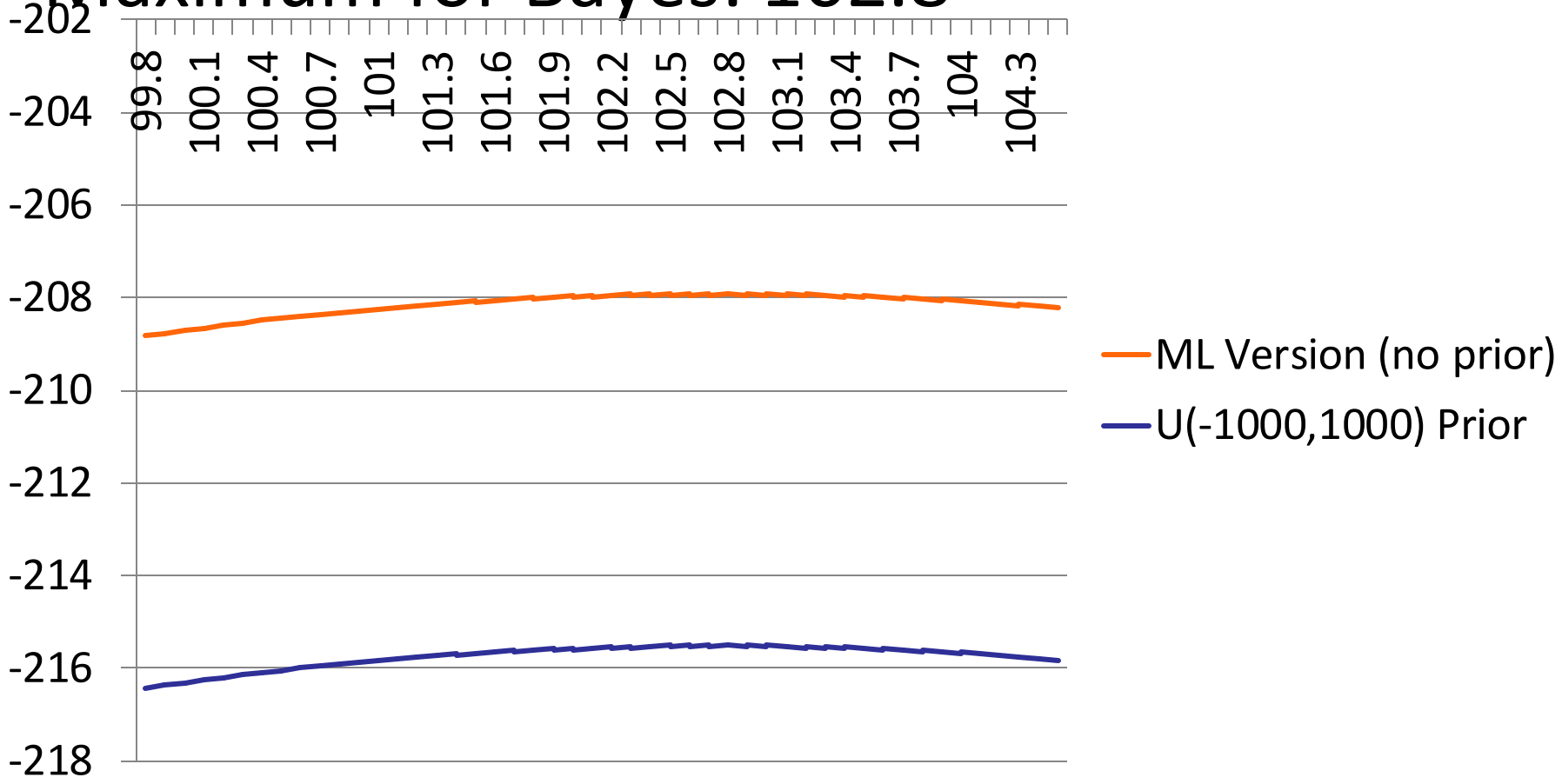
ML vs. Prior for β_0 of N(100, 0.15^2)

- Maximum for ML: 102.8
- Maximum for Bayes: 100



ML vs. Prior for β_0 of U(-1000,1000)

- Maximum for ML: 102.8
- Maximum for Bayes: 102.8



Summarizing Bayesian So Far

- Bayesian → parameters have prior distributions

- Estimation in Bayesian → MAP estimation is much like estimation in ML, only instead of likelihood of data, now have to add in likelihood for prior of all parameters
  - But…MAP estimation may be difficult as figuring out derivatives for gradient function (for Newton-Raphson) are not always easy
  - Where they are easy: **Conjugate** priors → prior distributions that are the same as the posterior distribution (think multilevel with normal outcomes)

- Priors can be **informative** (highly peaked) or **uninformative** (not peaked)
  - Some uninformative priors will give MAP estimates that are equal to ML

- Up next: estimation by brute force: Markov Chain Monte Carlo

MARKOV CHAIN MONTE CARLO ESTIMATION: THE BASICS

How Estimation Works (More or Less)

- Most estimation routines do one of three things:

1. **Minimize Something:** Typically found with names that have "least" in the title. Forms of least squares include "Generalized", "Ordinary", "Weighted", "Diagonally Weighted", "WLSMV", and "Iteratively Reweighted." Typically the estimator of last resort…

2. **Maximize Something:** Typically found with names that have "maximum" in the title. Forms include "Maximum likelihood", "ML", "Residual Maximum Likelihood" (REML), "Robust ML". Typically the gold standard of estimators (and we now know why).

3. **Use Simulation to Sample from Something:** more recent advances in simulation use resampling techniques. Names include "Bayesian Markov Chain Monte Carlo", "Gibbs Sampling", "Metropolis Hastings", "Metropolis Algorithm", and "Monte Carlo". Used for complex models where ML is not available or for methods where prior values are needed.

How MCMC Estimation Works

- MCMC estimation works by taking samples from the posterior distribution of the data given the parameters:

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right) = \frac{f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)}{f\left(\boldsymbol{y}_p\right)}$$

  - How is that possible? We don't know $f(\boldsymbol{y}_p)$…but…we'll see…

- After enough values are drawn, a rough shape of the distribution can be formed
  - From that shape we can take summaries and make them our parameters (i.e., mean)

- How the sampling mechanism happens comes from several different algorithms that you will hear about, the most popular being:
  - **Gibbs Sampling**: used when $f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right)$ is known
    - Parameter values are drawn and kept throughout the chain
  - **Metropolis-Hastings (within Gibbs):** used when $f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right)$ is unknown
    - Parameter values are proposed, then either kept or rejected
    - SAS PROC MCMC uses the latter
    - TRIVIA NOTE: The Metropolis algorithm comes from Chemistry (in 1950)
  - **Hybrid MC:** Newer versions (1980s; implemented in Stan)

- In some fields (Physics in particular), MCMC estimation is referred to as Monte Carlo estimation

MCMC Estimation with MHG

- The Metropolis-Hastings algorithm works a bit differently than Gibbs sampling:

1. Each parameter (here β_0 and σ_e^2) is given an initial value
2. In order, a new value is proposed for each model parameter from some distribution:

$$\beta_0^* \sim Q(\beta_0^*|\beta_0); \sigma_e^{2^*} \sim Q\left(\sigma_e^{2^*}\middle|\sigma_e^2\right)$$

3. The proposed value is then accepted as the current value with probability $\max(r_{MHG}, 1)$:

$$r_{MHG} = \frac{f\left(\boldsymbol{y}_p\middle|\beta_0^*, \sigma_e^{2^*}\right) f(\beta_0^*) f\left(\sigma_e^{2^*}\right) Q\left(\beta_0\middle|\beta_0^*\right) Q\left(\sigma_e^2\middle|\sigma_e^{2^*}\right)}{f\left(\boldsymbol{y}_p\middle|\beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2) Q(\beta_0^*|\beta_0) Q\left(\sigma_e^{2^*}\middle|\sigma_e^2\right)}$$

4. The process continues for a pre-specified number of iterations (more is better)

Notes About MHG

- The constant in the denominator of the posterior distribution:

$$f\left(\beta_0, \sigma_e^2 \middle| \boldsymbol{y}_p\right) = \frac{f\left(\boldsymbol{y}_p \middle| \beta_0, \sigma_e^2\right) f(\beta_0) f(\sigma_e^2)}{f\left(\boldsymbol{y}_p\right)}$$

…cancels when the ratio is formed

- The proposal distributions $Q(\beta_0^* | \beta_0)$ and $Q\left(\sigma_e^{2^*} \middle| \sigma_e^2\right)$ can literally be any statistical distribution
  - The trick is picking ones that make the chain "converge" quickly
  - Want to find values that lead to moderate number of accepted parameters
  - SAS PROC MCMC/WINBUGS don't make you pick these

- Given a long enough chain, the final values of the chain will come from the posterior distribution
  - From that you can get your parameter estimates

Introducing Jags…

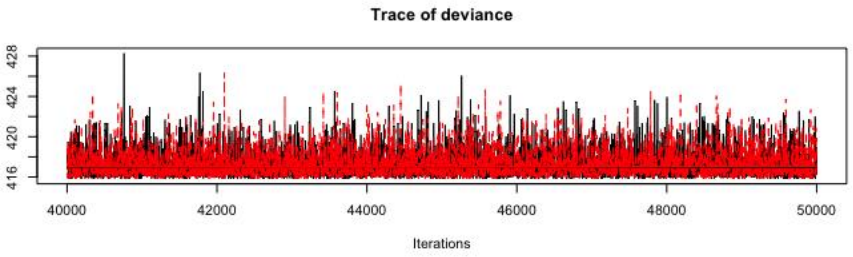
```
# estimation with Bayesian

model01Bayes = function(){

  # likelihood
  for (i in 1:n){
    y[i] ~ dnorm(mu, tau)
  }

  #priors
  mu ~ dnorm(100, 1/2.13^2)
  tau ~ dunif(1/400, 1/200)
  sigma2 = 1/tau
}

data = list(y = dataIQ$y, n = nrow(dataIQ))

jags.param = c("mu", "tau", "sigma2")

fit <- jags.parallel(data=data,
                     parameters.to.save=jags.param,
                     n.iter=50000, n.chains=2,n.thin=2,n.burnin=40000,
                     model.file=model01Bayes)
```

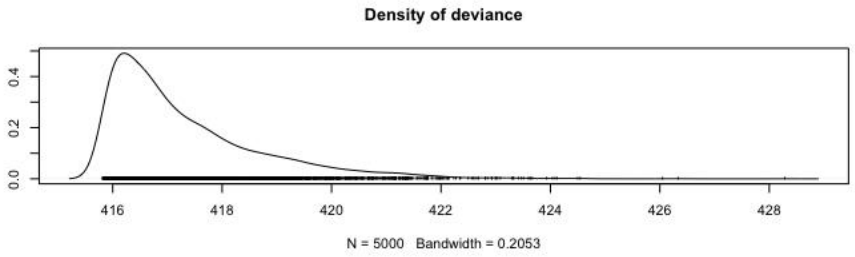
Iteration History from JAGS

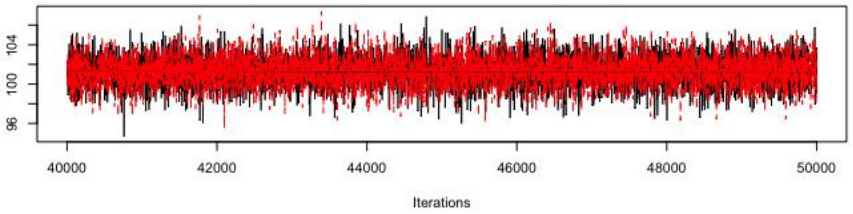
Filter

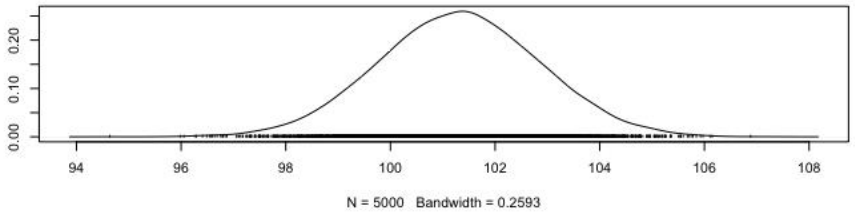
	deviance	mu	sigma2	tau
1	416.7964	100.85794	267.4554	0.003738941
2	418.1109	102.33473	328.7272	0.003042036
3	416.7720	100.62472	242.2023	0.004128781
4	416.7956	100.59997	246.9883	0.004048774
5	416.7415	100.74722	257.4887	0.003883666
6	417.8188	99.76745	230.0463	0.004346951
7	419.4729	98.87860	298.0748	0.003354863
8	415.9143	102.97544	254.1412	0.003934821
9	416.2015	101.86554	219.4816	0.004556191
10	417.2012	102.11906	303.9016	0.003290539
11	415.8802	103.19065	247.1723	0.004045762
12	417.0017	101.14455	285.1704	0.003506675
13	417.2518	100.24214	229.8218	0.004351197
14	415.8366	103.05208	240.4516	0.004158841
15	416.1867	104.09353	243.5673	0.004105641
16	416.5808	100.85335	244.6015	0.004088282
17	416.4756	101.11430	228.4360	0.004377594
18	419.4313	99.29725	314.4079	0.003180582
19	416.1548	101.89660	221.1416	0.004521989
20	416.5363	101.11042	224.6534	0.004451302
21	416.2943	101.30912	250.4290	0.003993148
22	415.8509	102.77879	248.1755	0.004029407
23	419.3382	98.95027	296.3523	0.003374363
24	415.9175	103.40843	245.3931	0.004075094
25	417.6465	102.22143	316.6267	0.003158293
26	420.4722	103.00402	381.0396	0.002624399
27	416.8376	101.28403	208.8647	0.004787788
28	417.6739	100.28041	287.1524	0.003482472
29	417.9725	104.73157	310.0394	0.003225396

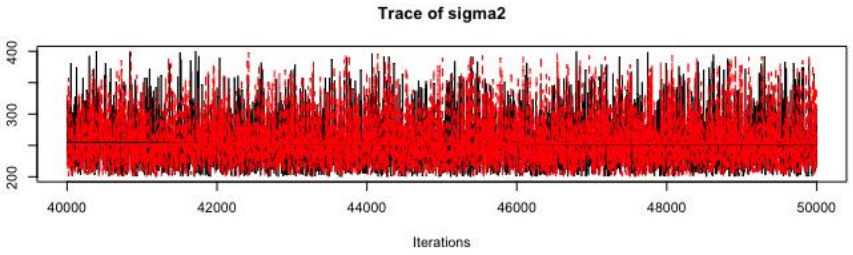
Examining the Chain and Posteriors

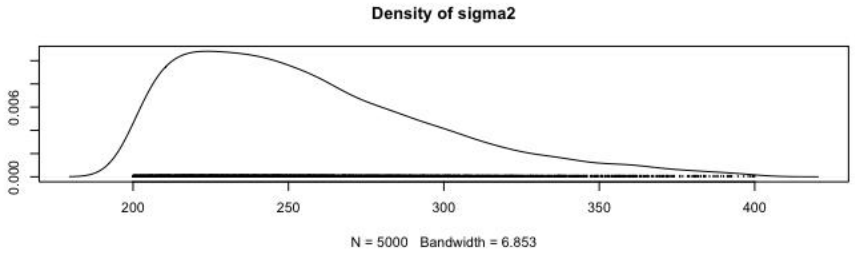


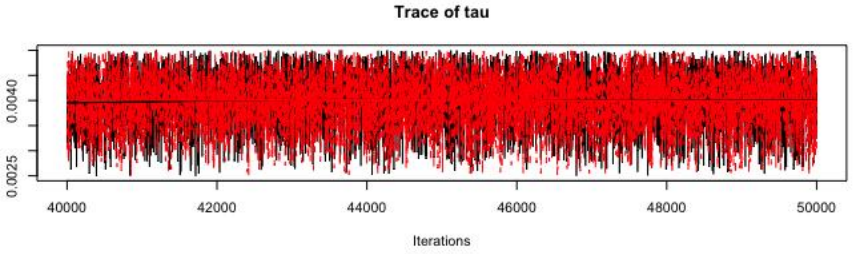


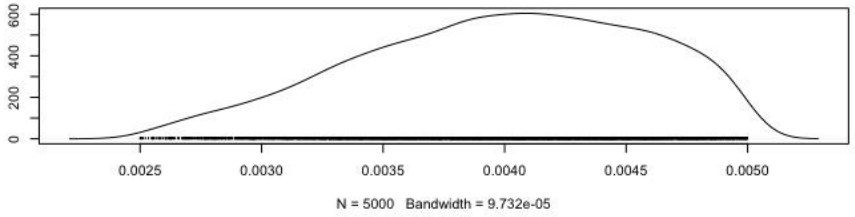












Practical Specifics in MCMC Estimation

- A **burn-in** period is used where a chain is run for a set number of iterations before the sampled parameter values are used in the posterior distribution

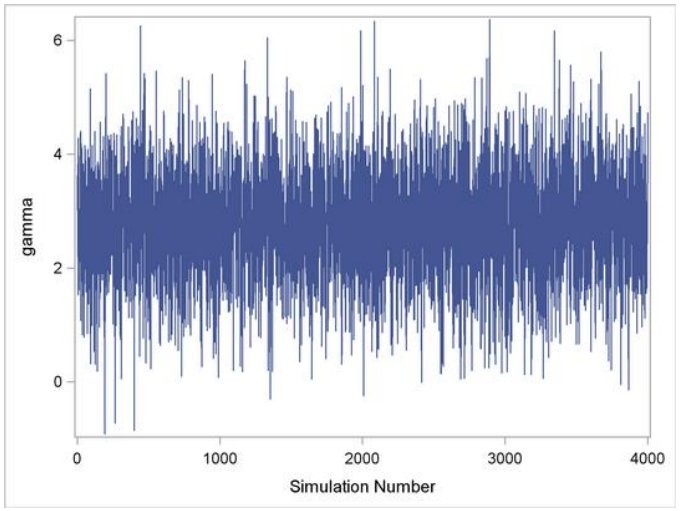
- Because of the rejection/acceptance process, any two iterations are likely to have a high correlation (called **autocorrelation**) → posterior chains use a **thinning interval** to take every Xth sample to reduce the autocorrelation
  - A high autocorrelation may indicate the standard error of the posterior distribution will be smaller than it should be

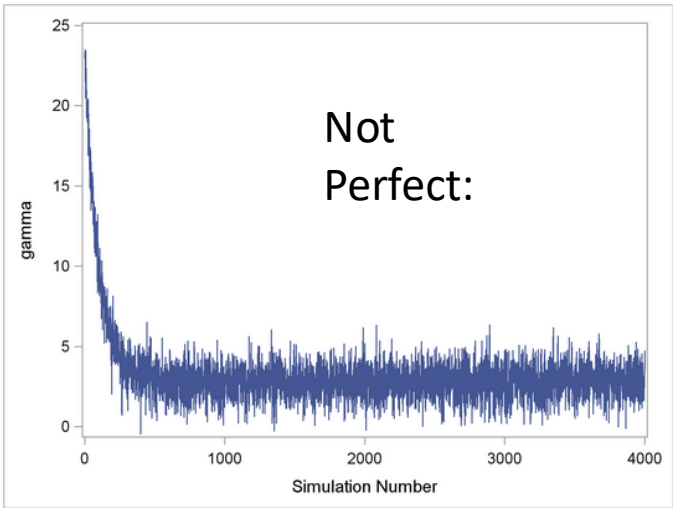
- The **chain length** (and sometimes number of chains) must also be long enough so the rejection/acceptance process can reasonably approximate the posterior distribution

- How does one what values to pick for these? Output diagnostics
  - Trial. And. Error.

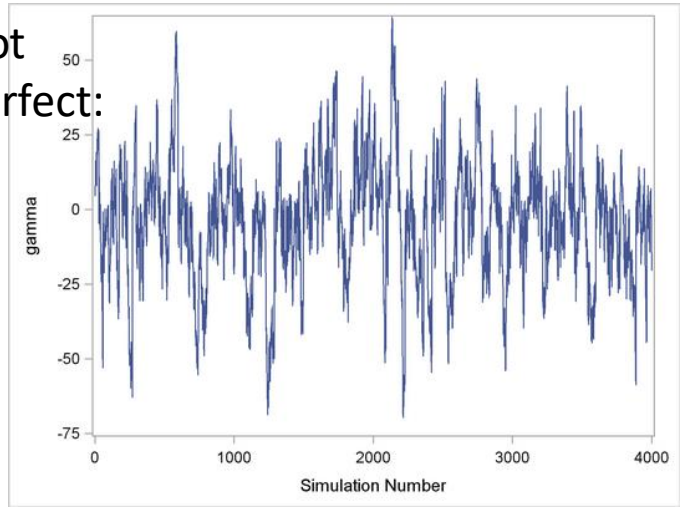
Best Output Diagnostics: the Eye Ball Test

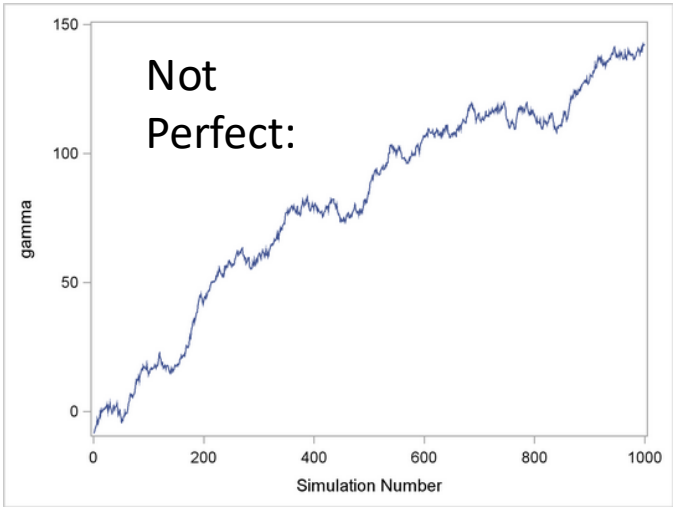
Perfect:





Not
Perfect:





Output Statistics and Diagnostics

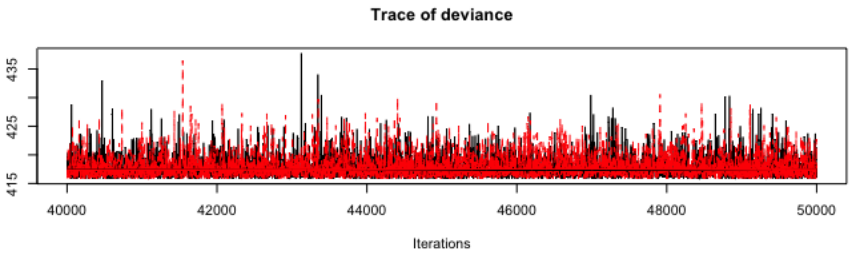
```
> fit
Inference for Bugs model at "model01Bayes", fit using jags,
 2 chains, each with 50000 iterations (first 40000 discarded), n.thin = 2
 n.sims = 10000 iterations saved
          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu        101.312   1.546  98.280 100.279 101.316 102.347 104.349 1.001 10000
sigma2    256.627  40.791 202.918 224.724 248.240 280.247 358.019 1.001 10000
tau         0.004   0.001   0.003   0.004   0.004   0.004   0.005 1.001 10000
deviance  417.317   1.404 415.862 416.284 416.869 417.922 421.059 1.001 10000

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 1.0 and DIC = 418.3
DIC is an estimate of expected predictive error (lower deviance is better).
```

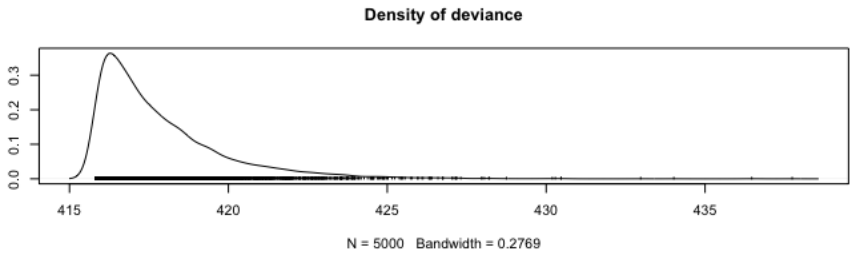
Changing Up the Prior

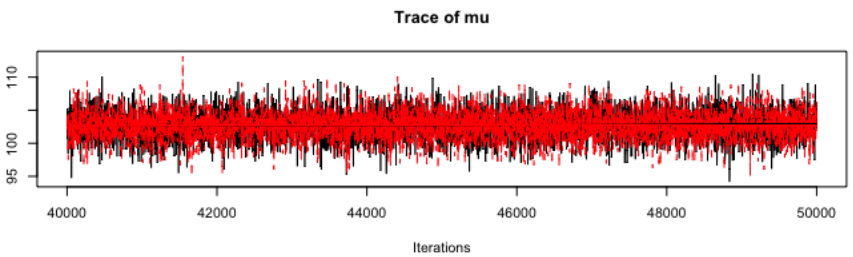
- To demonstrate how changing the prior affects the analysis, we will now try a few prior distributions for our parameters

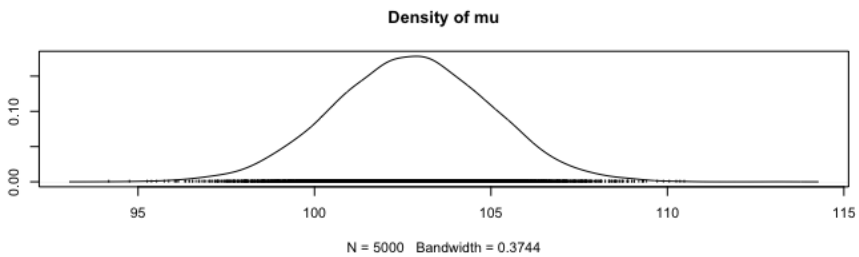
```
> fit2
Inference for Bugs model at "model02Bayes", fit using jags,
 2 chains, each with 50000 iterations (first 40000 discarded), n.thin = 2
 n.sims = 10000 iterations saved
          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu        102.750   2.229  98.385 101.239 102.758 104.251 107.100 1.001 10000
sigma2    244.899  51.326 164.789 208.259 238.353 273.458 362.843 1.001 10000
tau         0.004   0.001   0.003   0.004   0.004   0.005   0.006 1.001 10000
deviance  417.856   2.028 415.869 416.415 417.241 418.624 423.285 1.001  3200

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 2.1 and DIC = 419.9
DIC is an estimate of expected predictive error (lower deviance is better).
```

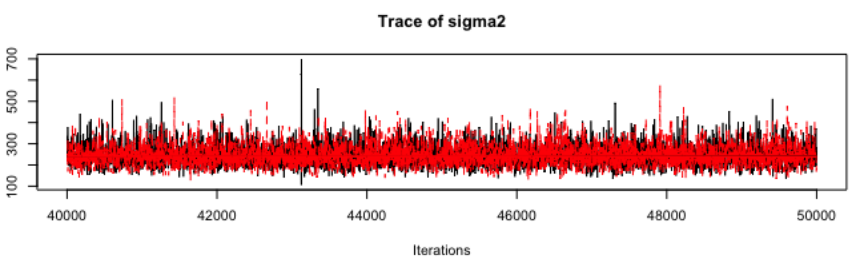
Chain Plots

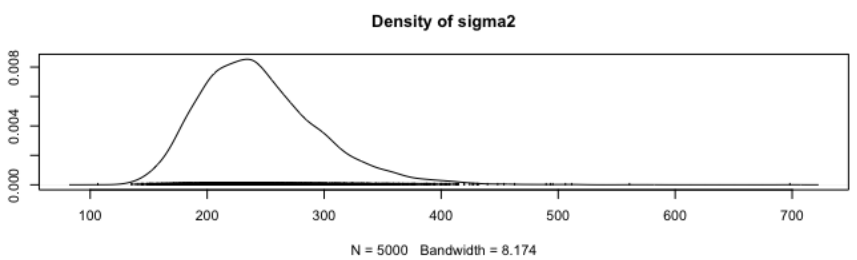

Trace of deviance
415
425
435
40000
42000
44000
46000
48000
50000
Iterations

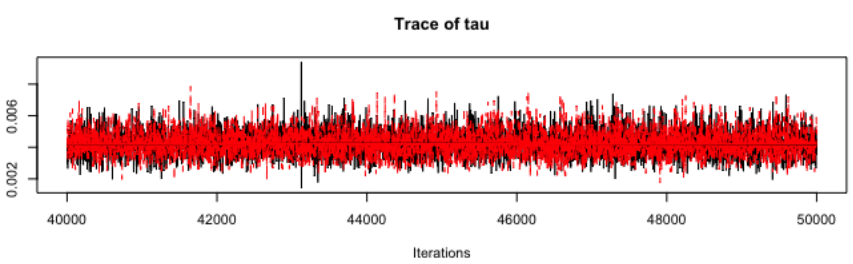

Density of deviance
0.0
0.1
0.2
0.3
415
420
425
430
435
N = 5000 Bandwidth = 0.2769

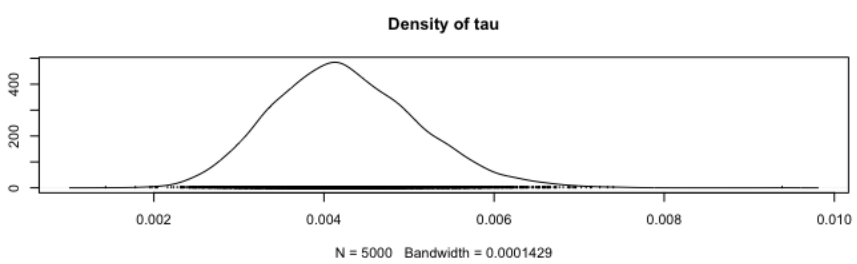

Trace of mu
95
100
110
40000
42000
44000
46000
48000
50000
Iterations


Density of mu
0.00
0.10
95
100
105
110
115
N = 5000 Bandwidth = 0.3744


Trace of sigma2
100
300
500
700
40000
42000
44000
46000
48000
50000
Iterations


Density of sigma2
0.000
0.004
0.008
100
200
300
400
500
600
700
N = 5000 Bandwidth = 8.174


Trace of tau
0.002
0.006
40000
42000
44000
46000
48000
50000
Iterations

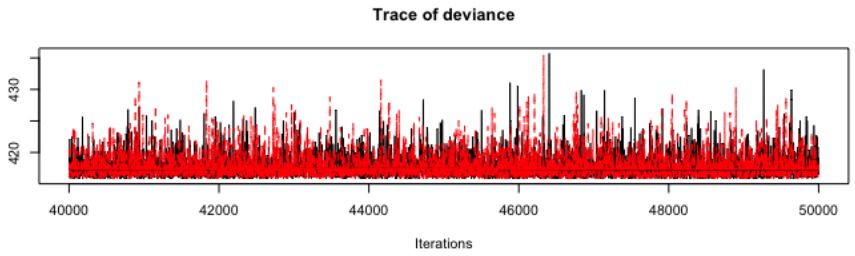

Density of tau
0
200
400
0.002
0.004
0.006
0.008
0.010
N = 5000 Bandwidth = 0.0001429

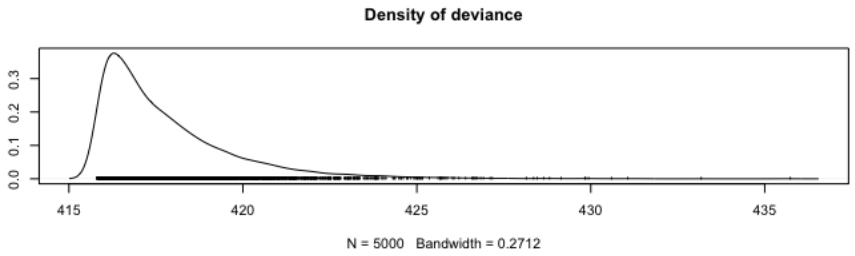
Changing Up the Prior

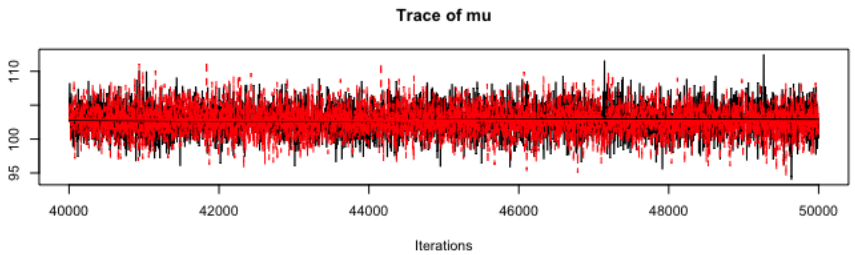
- Prior: $\beta_0 \sim N(0,100{,}000)$;
- $\sigma_e^{-2} \sim gamma(r = .01, \lambda = .01)$

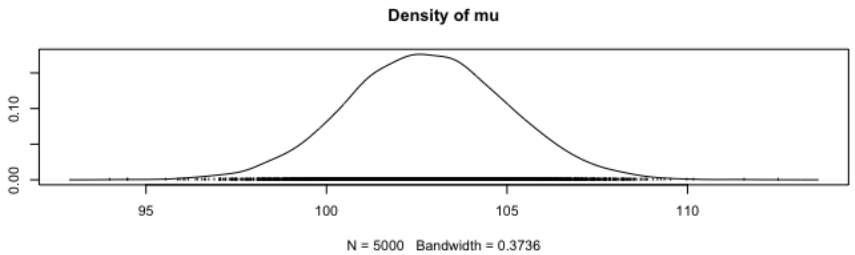
```
> fit3
Inference for Bugs model at "model03Bayes", fit using jags,
 2 chains, each with 50000 iterations (first 40000 discarded), n.thin = 2
 n.sims = 10000 iterations saved
          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu        102.784   2.224  98.426 101.274 102.758 104.254 107.175 1.001  7300
sigma2    253.996  52.970 171.522 216.053 247.279 284.606 375.192 1.001  9500
tau         0.004   0.001   0.003   0.004   0.004   0.005   0.006 1.001  9500
deviance  417.812   1.994 415.872 416.409 417.203 418.572 423.105 1.003  1700

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 2.0 and DIC = 419.8
DIC is an estimate of expected predictive error (lower deviance is better).
```

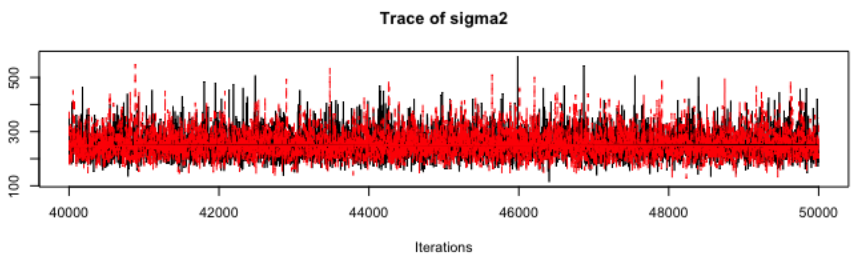
Chain Plots

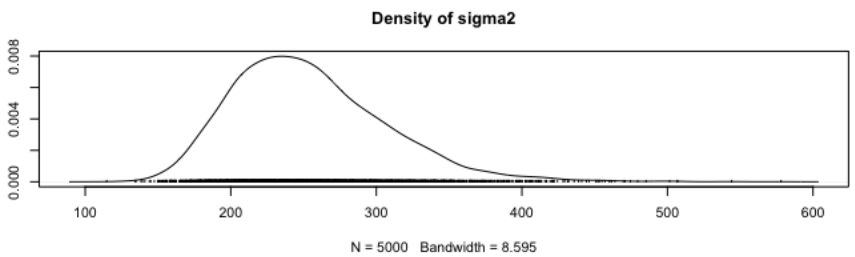

Trace of deviance
420
430
40000
42000
44000
46000
48000
50000
Iterations

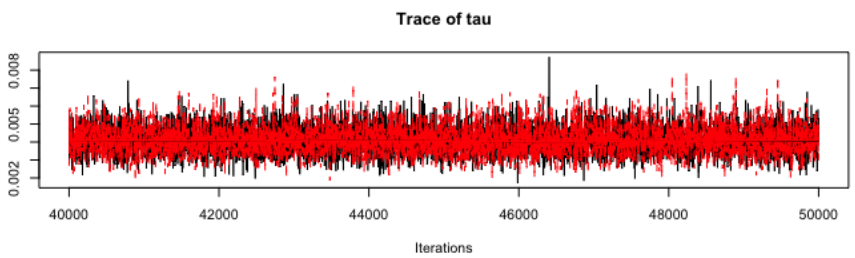

Density of deviance
0.0
0.1
0.2
0.3
415
420
425
430
435
N = 5000 Bandwidth = 0.2712

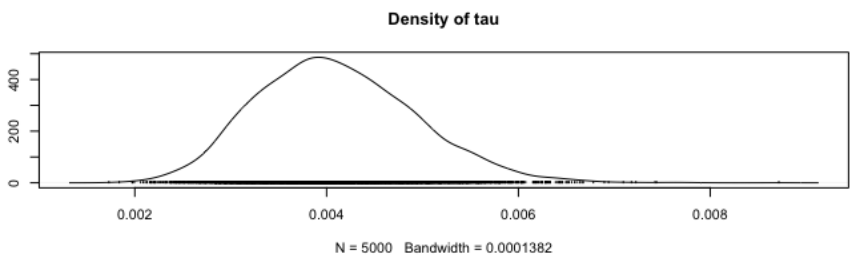

Trace of mu
95
100
110
40000
42000
44000
46000
48000
50000
Iterations


Density of mu
0.00
0.10
95
100
105
110
N = 5000 Bandwidth = 0.3736


Trace of sigma2
100
300
500
40000
42000
44000
46000
48000
50000
Iterations


Density of sigma2
0.000
0.004
0.008
100
200
300
400
500
600
N = 5000 Bandwidth = 8.595


Trace of tau
0.002
0.005
0.008
40000
42000
44000
46000
48000
50000
Iterations

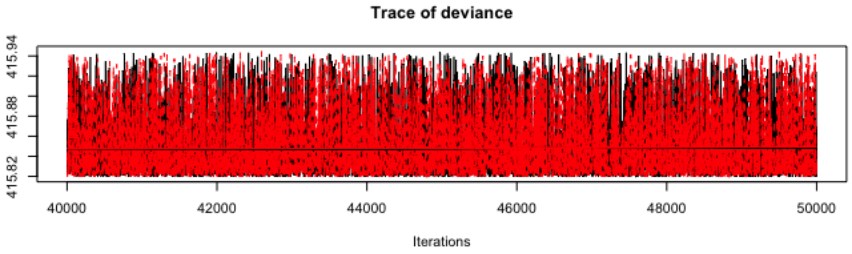

Density of tau
0
200
400
0.002
0.004
0.006
0.008
N = 5000 Bandwidth = 0.0001382

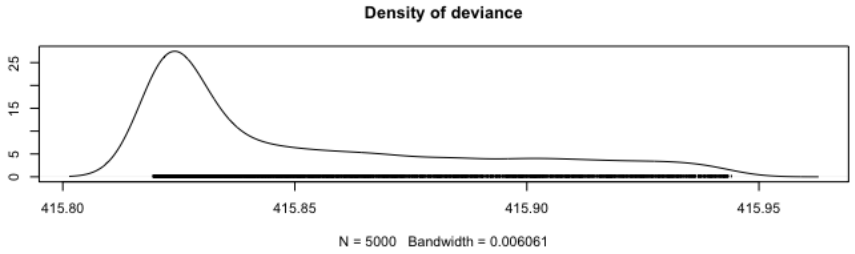
What About an Informative Prior?

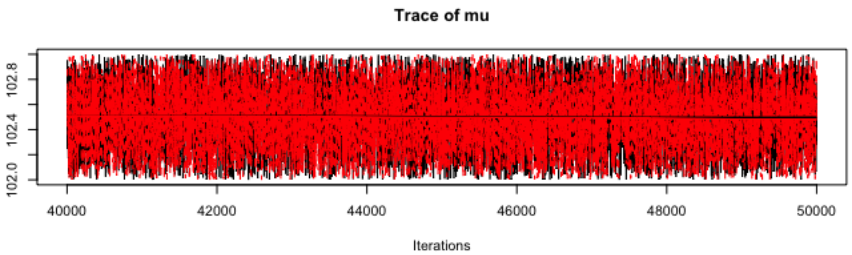
- Prior: $\beta_0 \sim U(102,103); \sigma_e^2 \sim U(238,242)$

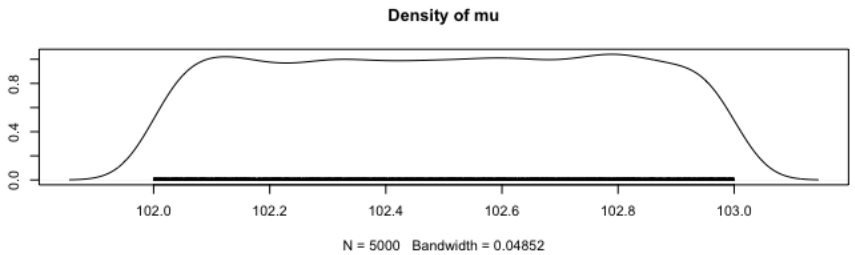
```
> fit4
Inference for Bugs model at "model04Bayes", fit using jags,
 2 chains, each with 50000 iterations (first 40000 discarded), n.thin = 2
 n.sims = 10000 iterations saved
         mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu       102.500   0.289 102.026 102.250 102.502 102.752 102.975 1.001  8200
sigma2   239.992   1.155 238.104 238.979 240.011 240.993 241.890 1.001 10000
tau        0.004   0.000   0.004   0.004   0.004   0.004   0.004 1.001 10000
deviance 415.853   0.036 415.820 415.823 415.835 415.876 415.935 1.000     1

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 0.0 and DIC = 415.9
DIC is an estimate of expected predictive error (lower deviance is better).
```

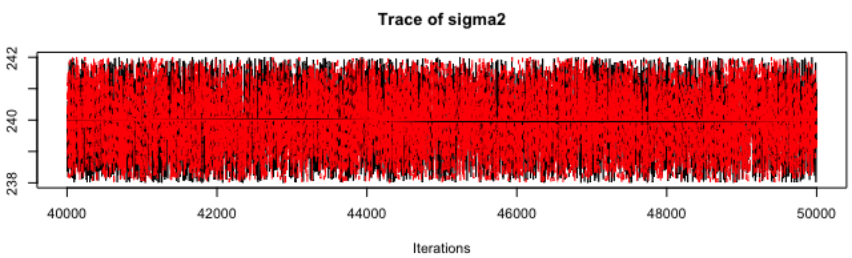
Chain Plots

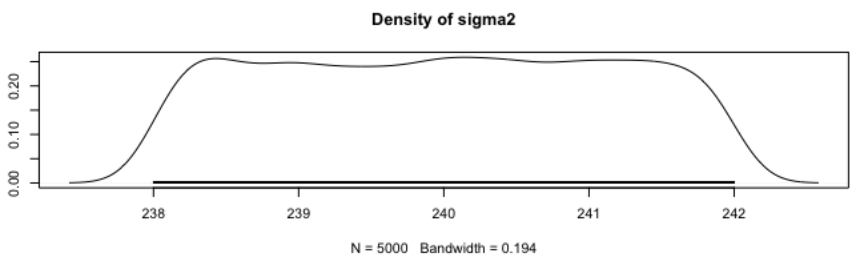

Trace of deviance
415.82
415.88
415.94
40000
42000
44000
46000
48000
50000
Iterations

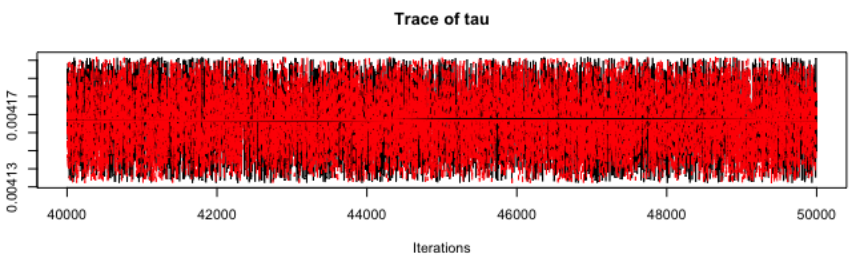

Density of deviance
0
5
15
25
415.80
415.85
415.90
415.95
N = 5000 Bandwidth = 0.006061

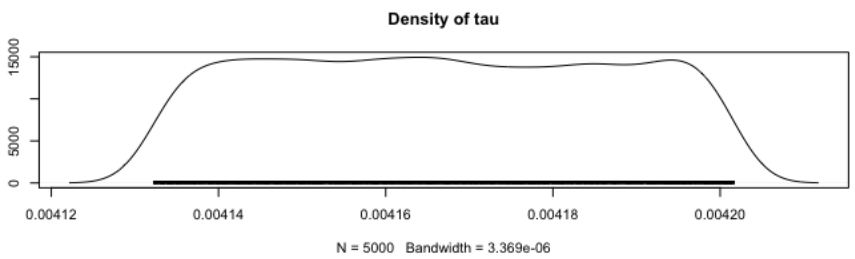

Trace of mu
102.0
102.4
102.8
40000
42000
44000
46000
48000
50000
Iterations


Density of mu
0.0
0.4
0.8
102.0
102.2
102.4
102.6
102.8
103.0
N = 5000 Bandwidth = 0.04852


Trace of sigma2
238
240
242
40000
42000
44000
46000
48000
50000
Iterations


Density of sigma2
0.00
0.10
0.20
238
239
240
241
242
N = 5000 Bandwidth = 0.194


Trace of tau
0.00413
0.00417
40000
42000
44000
46000
48000
50000
Iterations


Density of tau
0
5000
15000
0.00412
0.00414
0.00416
0.00418
0.00420
N = 5000 Bandwidth = 3.369e-06

MCMC in R

- R itself does not have an MCMC engine native to the language – but there are many free versions available outside of R

- For instance, if you wanted to estimate a path model with MCMC you can:
  - Install the blavaan package (Bayesian lavaan)
  - Run the path analysis with MCMC

- I am not showing you these because I they all end up being really frustrating
  - Very buggy
  - Took me about an hour to just install all code

WRAPPING UP

Wrapping Up

- Today was an introduction to Bayesian statistics
  - Bayes = use of prior distributions on parameters
- We used two methods for estimation:
  - MAP estimation – far less common
  - MCMC estimation
    - Commonly, people will say Bayesian and mean MCMC – but Bayesian is just the addition of priors. MCMC is one way of estimating Bayesian models!
- MCMC is effective for most Bayesian models:
  - Model likelihood and prior likelihood are all that are needed
- MCMC is estimation by brute force:
  - Can be very slow, computationally intensive, and disk-space intensive