Lab 8

PSTAT 115, Fall 2019

Dec. 4, 2019

Objectives

- MCMC Review
- Hierarchical Modeling

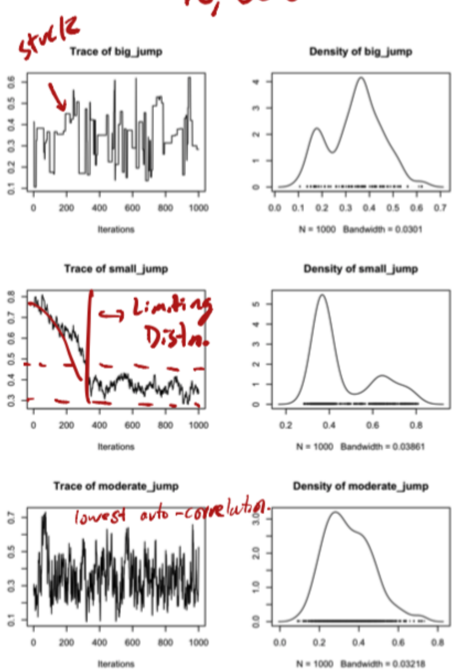
MCMC Review

Metropolis Algorithm

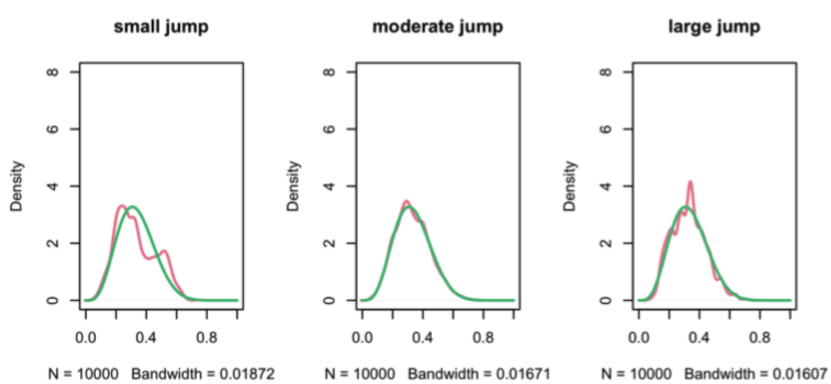
- Let $P(\theta \mid y)$ be a Beta(5, 10) posterior distribution
- 1-d sampling: lets try sampling from the Beta using the Metropolis algorithm
- Initialize θ_0 to 0.9
  - Note that the probability of drawing a value larger than 0.9 from a Beta(5, 10) is smaller than 1e-8
  - Our initial value is far from the high posterior density
  - In the long run this won't matter
- Define transition kernel $J(\theta_{t+1} \mid \theta_t)$ as $\theta^* \sim N(\theta_t, \tau^2)$
  - How does choice of τ^2 effect performance of MC sampler?

Small, Moderate and Large Proposal variance

10, 000



10,000 Samples




```
## [1] "Effective sample size (small proposal variance): 14.00"
## [1] "Effective sample size (medium proposal variance): 1007.50"
## [1] "Effective sample size (large proposal variance): 459.49"
```

For more demonstration, please check MCMC Interaction.

- **Effective sample size:** correlated chain of samples is equivalent to this number of independent samples = correlated samples.
  - High rejection rate implies a lot of duplicate samples so effective size is smaller than number of iterations
  - High autocorrelation means neighboring samples are very similar (even if not exactly the same)

Monte Carlo Error for indep.

$$\frac{Var(\theta|Y)}{\boxed{ESS}} = \frac{\sum_i \sum_j Cov(\theta_i, \theta_j | Y)}{S^2}$$

Hierarchical Modeling (Eight Schools Example)

Eight schools example

- A study was performed for the Educational Testing Service (ETS) to evaluate the effects of coaching programs on SAT preparation
- Each of eight different schools used a short-term SAT prep coaching program
- Compute the average SAT score in those who did take the program minus those that did not participate in the program
- We observe the average difference varies by school. What accounts for these differences?

- Two extremes:
  - Estimate the effect of the program in every school independently
  - A separate prior distribution for each school effect
  - Or assume the effect is the same in every school
  - Combine all the data
  - A compromise between the above 2 options?

Model Specification:

$$\theta_j \sim N(\mu, \tau^2)$$
$$y_j \sim N(\theta_j, \sigma_j^2)$$

- θ_j are the true unknown effects of the program in school j
- y_j is the observed effects of the program in school j
- Add a shared normal prior distribution to θ_j
- Assume the global mean, μ is also unknown
- τ^2 determines how much weight weight we put on the independent estimate vs the pooled estimate

Eight schools example

- If τ^2 is large, the prior for θ_j is not very strong
  - If $\tau^2 \to \infty$ equivalent to the no pooling model
- If τ^2 is small, we assume a priori that θ_j are very close
  - if $\tau^2 \to 0$ equivalent to the complete pooling model, $\theta_j = \mu$

```
## data ##
schools_dat <- list(J = 8,
                    y = c(28,  8, -3,  7, -1,  1, 18, 12),
                    sigma = c(15, 10, 16, 11,  9, 11, 10, 18))
```

Estimate hyperparameter μ and τ

```
data {
  int<lower=0> J;          // # of schools
  real y[J];               // estimated treatment
  real<lower=0> sigma[J];  // std err of effect
}

parameters {
 real theta[J];        // school effect
```

```
 real mu;              // mean for schools
 real<lower=0> tau;    // variance between schools
}

model {
 theta ~ normal(mu, tau);
 y ~ normal(theta, sigma);
}
```

Further, introduce η, the unscaled deviation from μ by school and let:

$$\theta_j = \mu + \tau * \eta_j,$$

where $\eta_j \sim N(0, 1)$

Noting that

$$\theta_j \mid \mu, \tau \sim N(\mu, \tau^2)$$

```
//saved as 8schools.stan
data {
 int<lower=0> J;         // number of schools
 real y[J];              // estimated treatment effects
 real<lower=0> sigma[J]; // standard error of effect estimates
}
parameters {
 real mu;                // population treatment effect
 real<lower=0> tau;      // standard deviation in treatment effects
 vector[J] eta;          // unscaled deviation from mu by school
}
transformed parameters {
 vector[J] theta = mu + tau * eta;        // school treatment effects
}
model {
 target += normal_lpdf(eta | 0, 1);       // prior log-density
 target += normal_lpdf(y | theta, sigma); // log-likelihood
}
```

```
stan_model <- rstan::stan_model("8schools.stan")
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.0/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.0/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.0/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
```

```
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

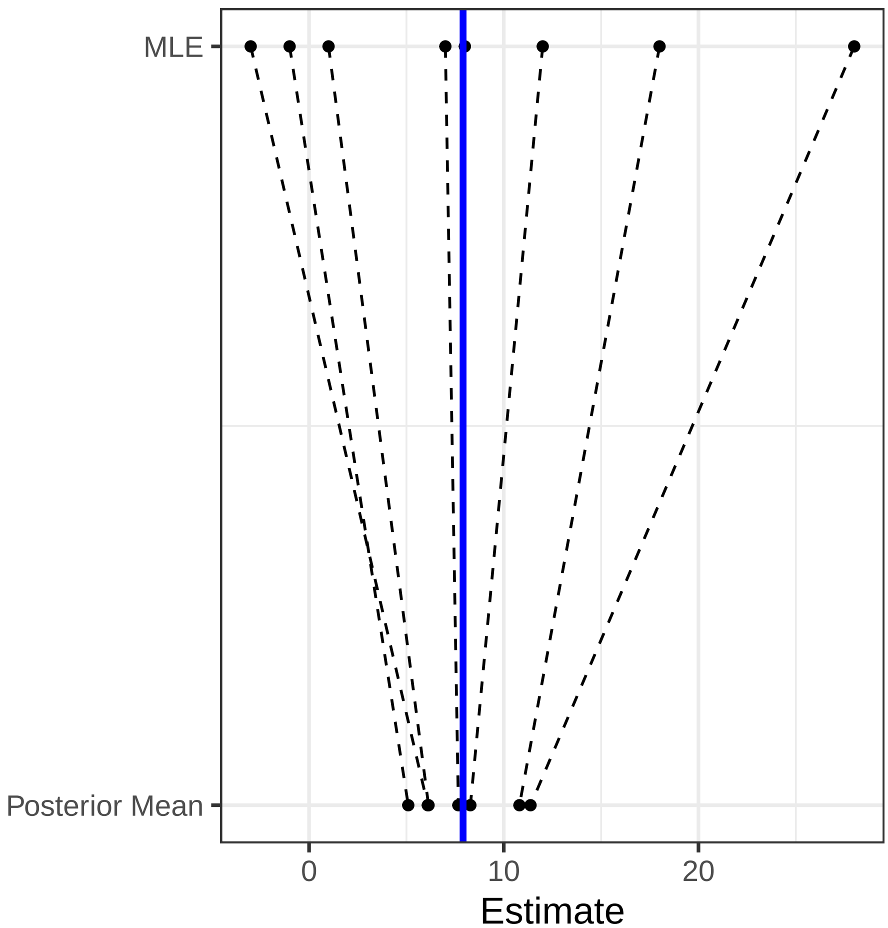
```
stan_fit <- sampling(stan_model,data = schools_dat, refresh = 0)
samples <- rstan::extract(stan_fit)
```

```
dim(samples$theta)
```

```
## [1] 4000    8
```

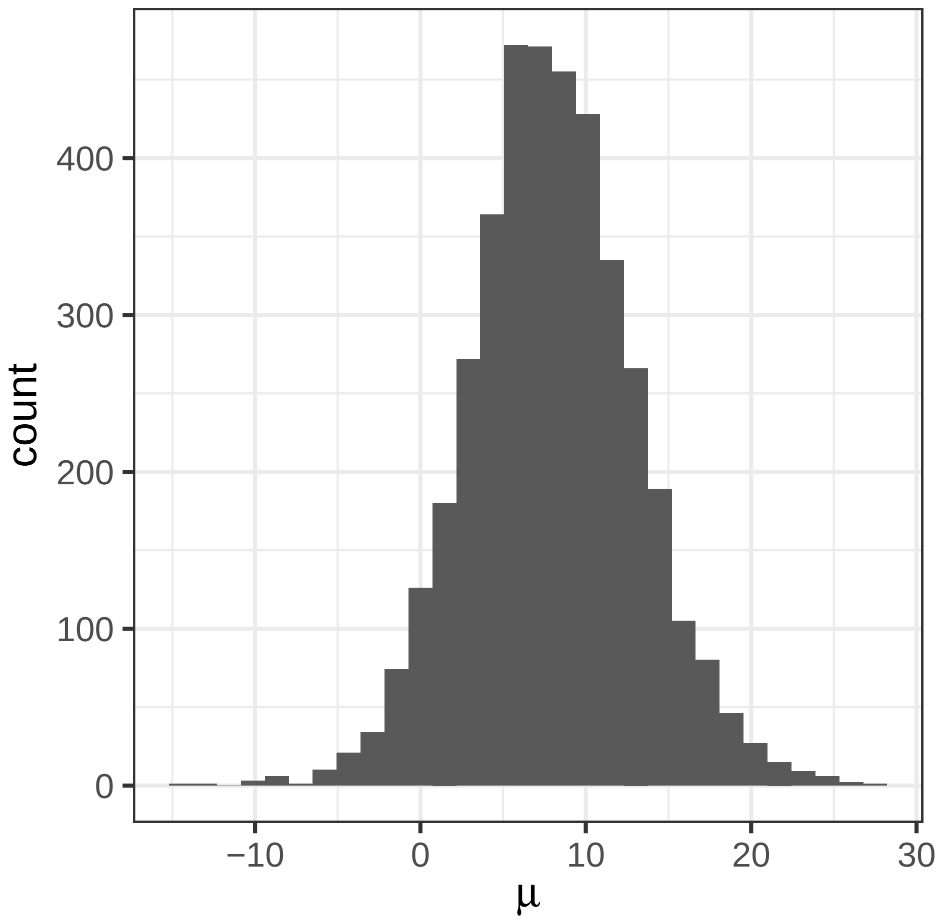
```
theta_post <- colMeans(samples$theta)

# shrinkage plot #
shrinkage_plot(schools_dat$y, theta_post)
```



The global effect, μ, is random:

```
# histogram for mu #
mu_post = tibble(mu = samples$mu)
ggplot(mu_post, aes(mu)) +
  geom_histogram() +
  theme_bw(base_size = 16) +
  labs(x = expression(mu))
```


count
400
300
200
100
0
−10
0
10
20
30
μ