STAT 578: Advanced Bayesian Modeling

Week 4 – Lesson 2

Normal Hierarchical Model in R/JAGS

Fall 2019

JAGS Analysis for 2016 Polls

As before, we use JAGS within R through package `rjags`.

We must also create the `sigma` variable (half the margin of error).

```
> d <- read.table("polls2016.txt", header=TRUE)

> d$sigma <- d$ME/2  # standard dev = half margin of error

> library(rjags)  # automatically loads coda package
Loading required package: coda
Linked to JAGS 4.3.0
Loaded modules: basemod,bugs
```

Start with the model similar to BDA3, Sec. 5.5 (approximately flat hyperprior):

```
> m1 <- jags.model("polls20161.bug", d)
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 7
   Unobserved stochastic nodes: 9
   Total graph size: 42

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
Warning messages:
1: In jags.model("polls20161.bug", d) : Unused variable "poll" in data
2: In jags.model("polls20161.bug", d) : Unused variable "ME" in data
```

After 2500 “burn-in” iterations, take 10000 samples:

```
> update(m1, 2500)  # burn-in
  |**************************************************| 100%

> x1 <- coda.samples(m1, c("mu","tau","theta"), n.iter=10000)
  |**************************************************| 100%
```

Note: Must explicitly specify which nodes to save.

```
> summary(x1)

Iterations = 3501:13500
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000
```

1. Empirical mean and standard deviation for each variable, plus standard error of the mean:

	Mean	SD	Naive SE	Time-series SE
mu	3.7506	0.6693	0.006693	0.01920
tau	0.9729	0.8160	0.008160	0.03778
theta[1]	3.8646	0.6541	0.006541	0.01500
theta[2]	3.5767	0.9560	0.009560	0.01895
theta[3]	3.5169	0.8436	0.008436	0.01962
theta[4]	3.8452	0.8211	0.008211	0.01695
theta[5]	3.0275	1.1220	0.011220	0.03810
theta[6]	4.2608	1.0751	0.010751	0.02860
theta[7]	4.1450	0.8937	0.008937	0.02177

(Similar example in BDA3, Sec. 5.5)

```
2. Quantiles for each variable:

            2.5%    25%    50%   75% 97.5%
mu       2.36976 3.3473 3.7619 4.143 5.102
tau      0.05094 0.3664 0.7835 1.354 3.052
theta[1] 2.58710 3.4379 3.8537 4.278 5.204
theta[2] 1.40257 3.0802 3.6439 4.137 5.445
theta[3] 1.63013 3.0486 3.5930 4.051 5.044
theta[4] 2.18579 3.3425 3.8326 4.325 5.611
theta[5] 0.26272 2.4273 3.2526 3.794 4.724
theta[6] 2.50408 3.5655 4.0890 4.810 6.822
theta[7] 2.59234 3.5721 4.0507 4.616 6.176
```

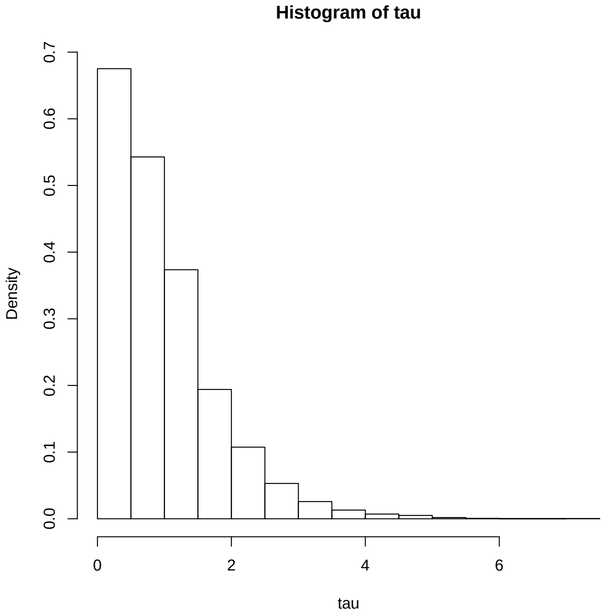
(Similar example in BDA3, Table 5.3)

Extract samples of τ and form a histogram:

```
> tau <- as.matrix(x1)[,"tau"]

> hist(tau, freq=FALSE)
```

(Similar example in BDA3, Figure 5.5)

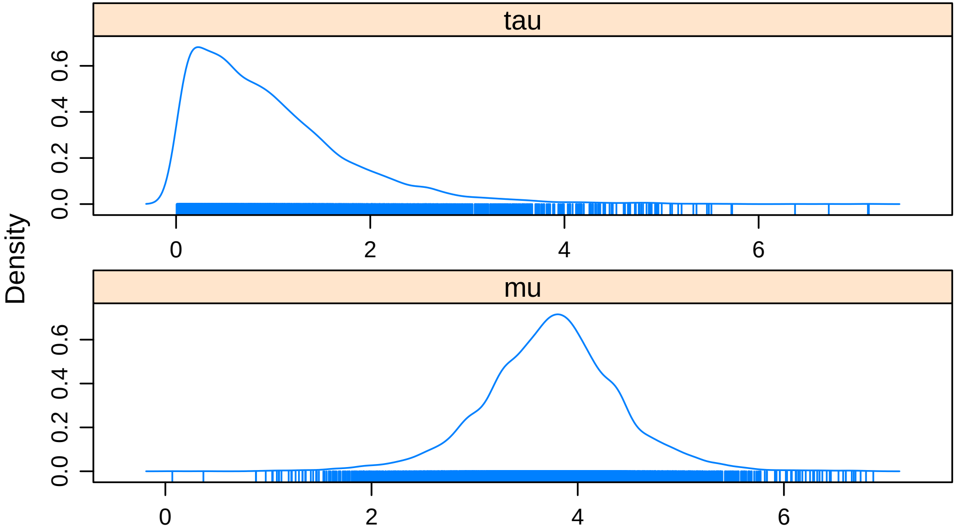

Histogram of tau
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
2
4
6
tau

Estimate densities of μ and τ:

```
> require(lattice)
...
> densityplot(x1[,c("mu","tau")])
```

(For τ, compare BDA3, Figure 5.5.)

For comparison, Clinton won the popular vote by a margin of 2.1%.



tau
mu
Density
0.0
0.2
0.4
0.6
0
2
4
6

Were the uniform priors too restrictive (compared with improper flat priors)?

Apparently not:

- Samples of μ are not even close to the limits of -1000 and 1000.
- Samples of τ are not even close to the upper limit of 1000.

Now try the alternative prior specification (diffuse normal for μ, scaled inverse chi-square for τ^2):

```
> m2 <- jags.model("polls20162.bug", d)

...

> update(m2, 2500)  # burn-in
  |**************************************************| 100%

> x2 <- coda.samples(m2, c("mu","tau","theta"), n.iter=10000)
  |**************************************************| 100%
```

```
> summary(x2)

...

           2.5%    25%    50%   75% 97.5%
mu       2.4405 3.2928 3.7175 4.125 5.007
tau      0.3906 0.6348 0.8543 1.183 2.251
theta[1] 2.5280 3.4038 3.8615 4.316 5.197
theta[2] 1.5170 2.9556 3.5811 4.179 5.456
theta[3] 1.6332 2.9172 3.4849 4.028 5.089
theta[4] 2.1409 3.2791 3.8108 4.349 5.499
theta[5] 0.6845 2.3634 3.0833 3.662 4.770
theta[6] 2.3534 3.5369 4.1588 4.825 6.546
theta[7] 2.4505 3.5225 4.0838 4.690 6.034
```

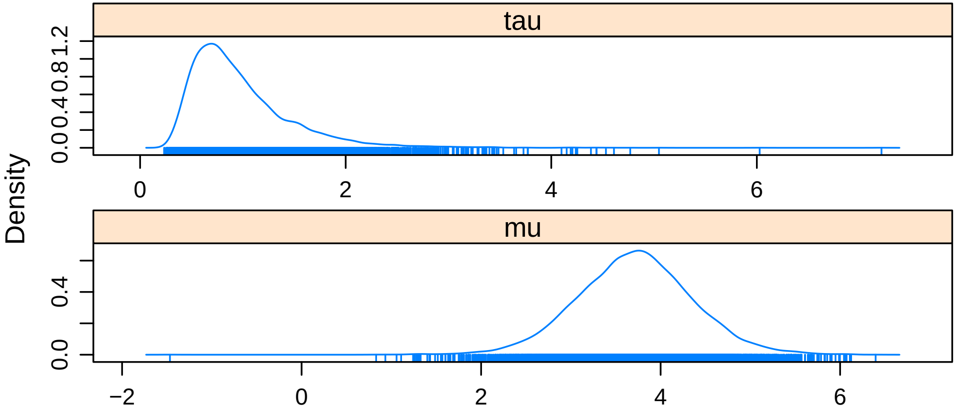
Interval for τ is narrower. Medians of θ_js are similar.

Check density estimates of μ and τ:

```
> require(lattice)
...

> densityplot(x2[,c("mu","tau")])
```

Not much different for μ, but narrower for τ ...


tau
0.0
0.4
0.8
1.2
0
2
4
6
Density
mu
0.0
0.4
−2
0
2
4
6

Compare the 95% posterior intervals for τ

under the flat hyperprior:	$(0.05,\ 3.05)$
under the hyperprior with $\tau^2 \sim \mathrm{Inv}\text{-}\chi^2(1,1)$:	$(0.39,\ 2.25)$

Inverse chi-square prior seems overly informative here.

Can we solve this problem by setting degrees of freedom and scale close to zero?

(The inverse chi-square is supposed to become less informative as its degrees of freedom and scale approach zero.)

Unfortunately, no: The posterior does not converge as those tend to zero.

Compare BDA3, Figure 5.9.