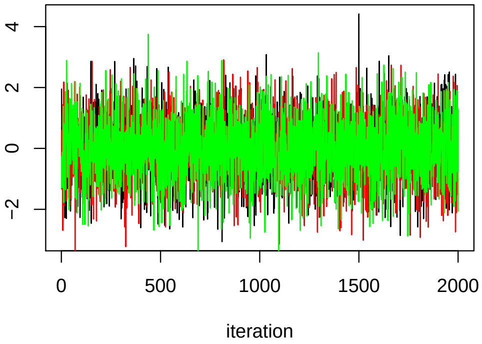
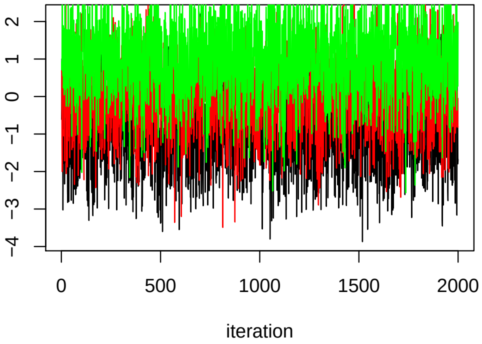
Bayes: Homework 9
Leslie Gains-Germain

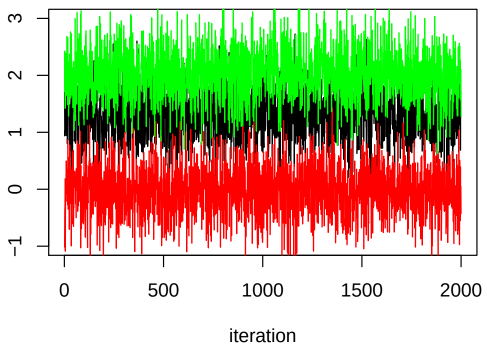
1. (a) i. Scenario 1: three chains from $N(0,1)$ white noise. I actually ran 2000 iterations in each chain so that I could use Raftery and Lewis's diagnostic in part (e).



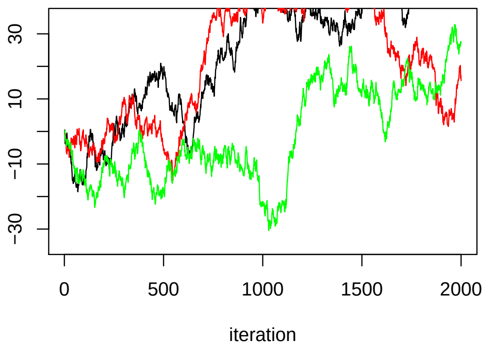
ii. Scenario 2: one chain from $N(-1,1)$, one chain from $N(0,1)$, and one chain from $N(1,1)$.



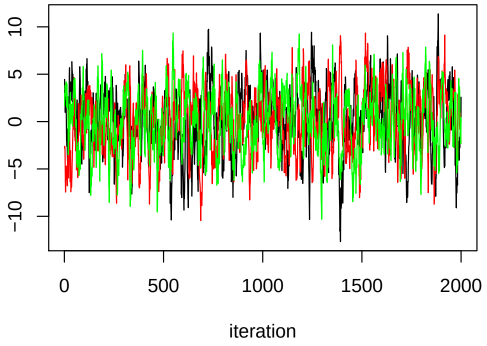
iii. Scenario 3: three from $MVN(0,\Sigma)$ with common $\sigma^2=1$ and $\rho=0.8$.



iv. Scenario 4: three from non-stationary correlated chains using `diffinv(rnorm(999))`.



v. Scenario 5: three from stationary chains with correlation using `filter(rnorm(1000), filter=rep(1, 10), circular=TRUE)`.



(b) The effective sample size is calculated differently in the newest version of the `coda` package than described in the textbook. In the textbook, the effective sample size is found by dividing the number of iterations (over all chains) by a formula that measures autocorrelation, so that chains with higher autocorrelation have lower effective sample sizes. In the newest version of the `coda` package, however, the number of iterations is divided by the spectral density. Details are on page 286 of the BDA3 text and the Effective Sample size section of the convergence diagnostic document.

In the coda package, $\hat{R}$ is calculated in the same way as the described in Chapter 11 in the BDA3 textbook.

(c) The below table shows the $\hat{R}$ and effective sample sizes for each scenario.

scenario	neff	rhat
1	5674.27	1.00
2	6291.69	1.74
3	5526.62	3.61
4	7.34	2.22
5	540.03	1.00

(d) The table below shows the z-statistics for all three chains from Geweke's diagnostic. The last three columns show the results from Raftery and Lewis's diagnostic (burn in and iterations required as well as the dependence factor).

scenario	zstats1	zstats2	zstats3	burn.in	n.iter	dependence.factor
1	0.00	-1.87	-0.60	2.00	3660.00	0.98
2	-0.17	-1.17	0.12	3.00	4255.00	1.14
3	-2.01	0.16	-0.74	3.00	4197.00	1.12
4	-11.17	-3.51	-2.35	198.00	215594.00	57.60
5	0.15	-2.19	-0.09	20.00	21264.00	5.68

(e) Convergence in Scenario 1 looks good - there is little autocorrelation within chains, and the between chain variability is low. Scenario 2 has little autocorrelation within chains, but large between chain variability. Scenario 3 looks similar to Scenario 2, but I think something might have gone wrong (see discussion below). Scenario 4 has both high correlation within chains and large between chain variability. Scenario 5 has small between chain variability but high correlation within chains.

I am confused about Scenario 3 and worried that I coded the simulations wrong. This scenario is supposed to be three chains with high correlation among the draws within each chain. But, for some reason, the chains get stuck in the part of the parameter space where they start. I would have expected to see a traceplot more similar to Scenario 4. Below is the code I used for Scenario 3:

```r
set.seed(321)
nsim <- 2000
Sigma <- diag(nsim)
require(gdata)
upperTriangle(Sigma) <- 0.8
lowerTriangle(Sigma) <- 0.8

require(LearnBayes)
one3 <- c(rmnorm(1, rep(0, nsim), Sigma))
two3 <- c(rmnorm(1, rep(0, nsim), Sigma))
three3 <- c(rmnorm(1, rep(0, nsim), Sigma))
```

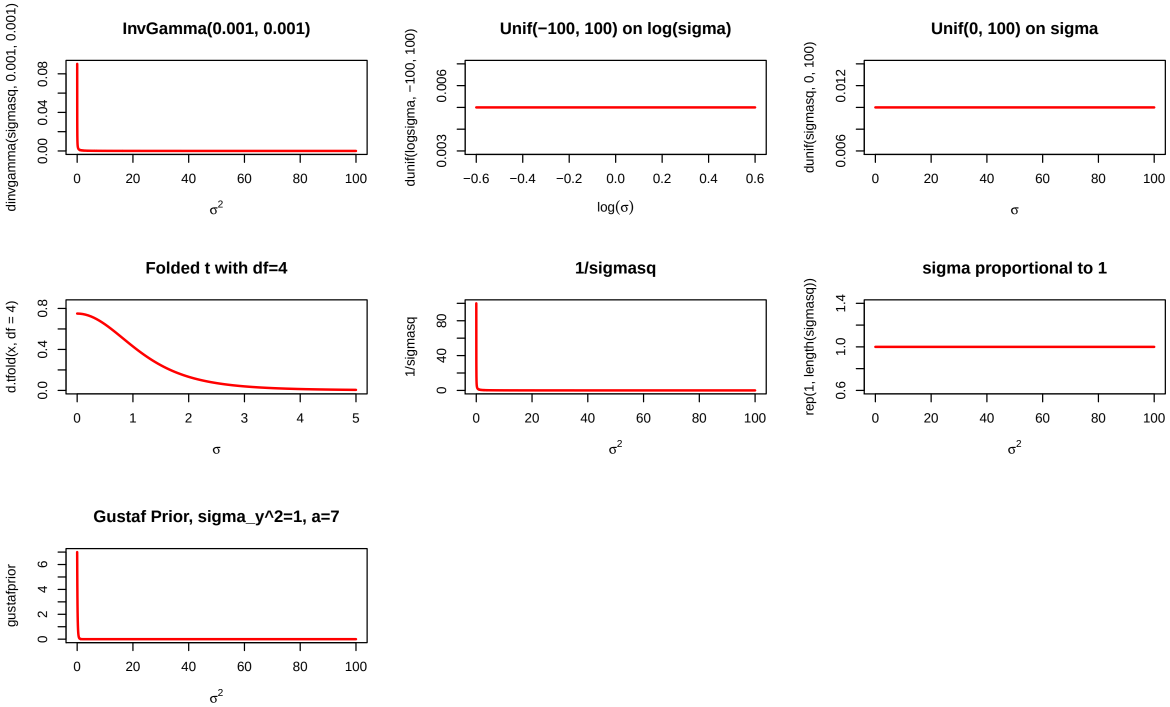
Geweke's Diagnostic picks up on a problem with Scenario 4, with large z-statistics for all three chains. Geweke's diagnostic does not, however, does not pick up on the large between chain variability in Scenario 2. I definitely would not want to use Geweke's diagnostic blindly without looking at a traceplot. Raftery and Lewis's diagnostic picks up on the high within chain correlation for Scenarios 4 and 5, but it also does not pick up on large variability between chains in Scenarios 2 and 3.

I think a main point of this exercise is to notice that it is not enough to look at one convergence diagnostic only. It's important to look at the traceplots and several convergence diagnostics. A good example of this is Scenario 2. In this scenario, the effective sample size indicates little autocorrelation within the chains, but the $\hat{R}$ value reveals the large amount of between chain variability. If I had only looked at the effective sample size without looking at the traceplots and $\hat{R}$, I wouldn't have known there was a problem with convergence. Another example of this is Scenario 5. In Scenario 5, the $\hat{R}$ value is close to 1, but the effective sample size is low (540 out of 2000 trials), indicating correlation among the chains. Lesson: Always consider several diagnostics and the traceplots when assessing convergence!

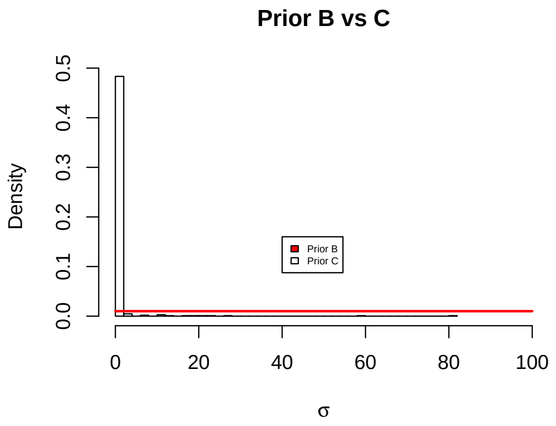
Question: What is the difference between saying a chain is stationary and saying that convergence has been reached?

2. (a) σ_α^2 is $Inv - Gamma(0.001, 0.001)$ which is the same as a scaled inverse chi squared distribution with parameters $\nu = 0.002$ and $s^2 = 1$.

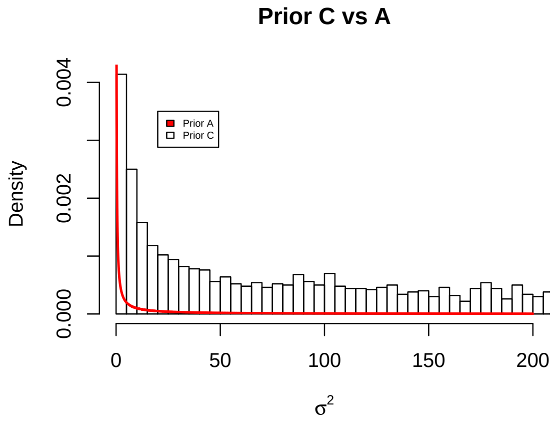
   (b) The plots of all the priors are shown below.



   (c) I took random draws from prior B and I exponentiated these draws to compare to prior C. Clearly, a prior of $log(\sigma)$ is very different from a uniform prior on σ. The plot is hard to see, but you can tell that after drawing from a $Unif(-100, 100)$ prior on $log(\sigma)$, you get many very large values for σ (not shown), and you get many very very small values for σ. This is why the $log(\sigma)$ prior blows up at zero. Priors B and C generate very different values for σ. It makes sense why Gelman recommends a uniform prior on the standard deviation rather than a uniform prior on the log standard deviation.



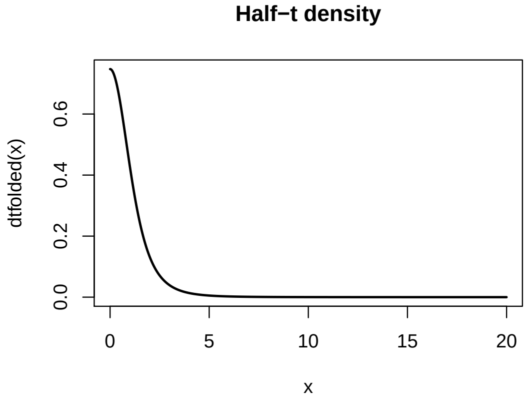
(d) I took 10000 random draws from a $Unif(0, 100)$ distribution of σ_θ, and then I exponentiated these draws to compare to the Inverse Gamma prior on σ_θ^2. It actually looks like the $Uniform(0, 100)$ prior on σ_θ is similar in shape to a $Inv-Gamma(0.001, 0.001)$ prior on σ_θ^2, but the uniform prior on the standard deviation looks better because although there are many σ_θ^2 values near 0, the prior does not blow up at zero like the Inverse Gamma prior does.



(e) A half-Cauchy distribution is a half t-distribution with 1 df.

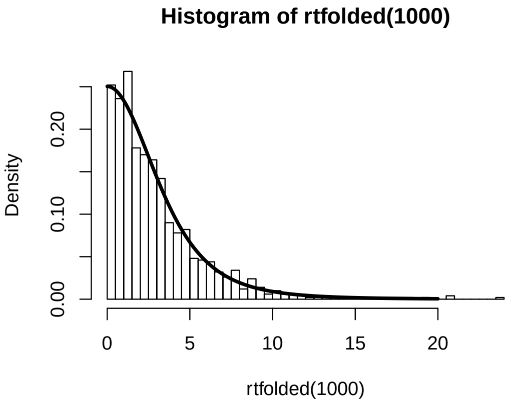
3. (a) i. The `dtfolded` function below calculates the density of a folded half t distribution with `df` degrees of freedom and a scale of A.

```r
# function to calculate density of a folded non-central t-distribution
dtfolded <- function (x, df=4, A=1) {
  p <- (1+1/df*(x/A)^2)^(-(df+1)/2)
  num.int <- sum(p*(x[2]-x[1]))
  dt <- p/num.int
  return(dt)
}

x <- seq(0, 20, by=0.01)

plot(x, dtfolded(x), type="l", main="Half-t density", lwd=2)
```



ii. The `rtfolded` function below generates a random draw from the folded t distribution with `df` degrees of freedom and scale parameter of A.

```r
rtfolded <- function(nsim, df=4, mu=0, sigma=1, A=3){
  z <- abs(rnorm(nsim, mu, sigma))
  x <- rgamma(nsim, df/2, (1/2*A^2))
  theta <- mu+sigma*z*sqrt(df/x)
  return(theta)
}

set.seed(31)
hist(rtfolded(1000), freq=FALSE, nclass=40)
lines(x, dtfolded(x, A=3), lwd=3)
```

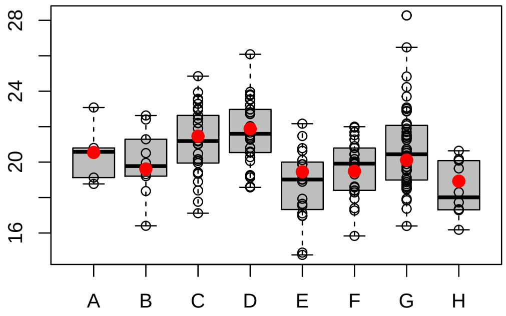


(b) The basic hierarchical model is:

$$y_{ij} \sim N(\mu + \alpha_j, \sigma_y^2)$$
$$\alpha_j \sim N(0, \sigma_\alpha^2)$$

(c) I used the code below to simulate y_{ij}. I first simulated the group effects, and then I simulated individual effects within each group. The code is adapted from your code in `SchoolsFakeData_CompareMethods_F13.R`. The boxplots showing the simulated data are shown below.

```
set.seed(10923)
 n.j <- c(5, 10, 30, 30, 20, 25, 50, 10)
 n.tot <- sum(n.j)

 J <- 8
 sigma.y <- 2

 mu <- 20
 sigma.alpha <- sqrt(2)

alpha <- numeric(J)
for(j in 1:J){
  alpha[j] <- rnorm(1, 0, sd=sigma.alpha)
}

 set.seed(2567)
 y.ij <- numeric(n.tot)
 school <- rep(NA,n.tot)
 n.index <- c(0, cumsum(n.j)) +1
 for (j in 1:J) {
  y.ij[n.index[j]:(n.index[j+1]-1)] <- rnorm(n.j[j], mean=mu+alpha[j], sd=sigma.y)
  school[n.index[j]:(n.index[j+1]-1)] <- rep(LETTERS[j], n.j[j])
}
school <- factor(school)
```

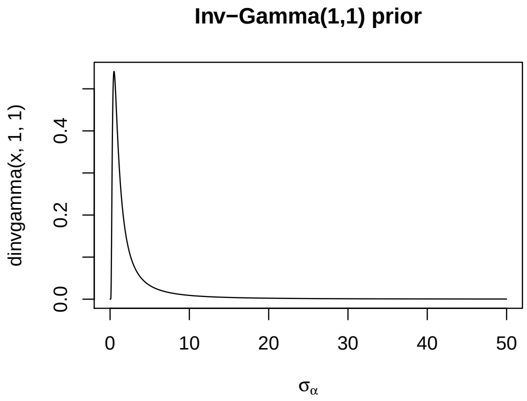


(d) I chose an improper uniform prior for μ, and improper uniform priors for σ_y and σ_α under the constraint that these parameters are larger than 0.

$$p(\mu) \propto 1$$
$$p(\sigma_y) \propto 1I_{\sigma_y>0}$$
$$p(\sigma_\alpha) \propto 1I_{\sigma_\alpha>0}$$

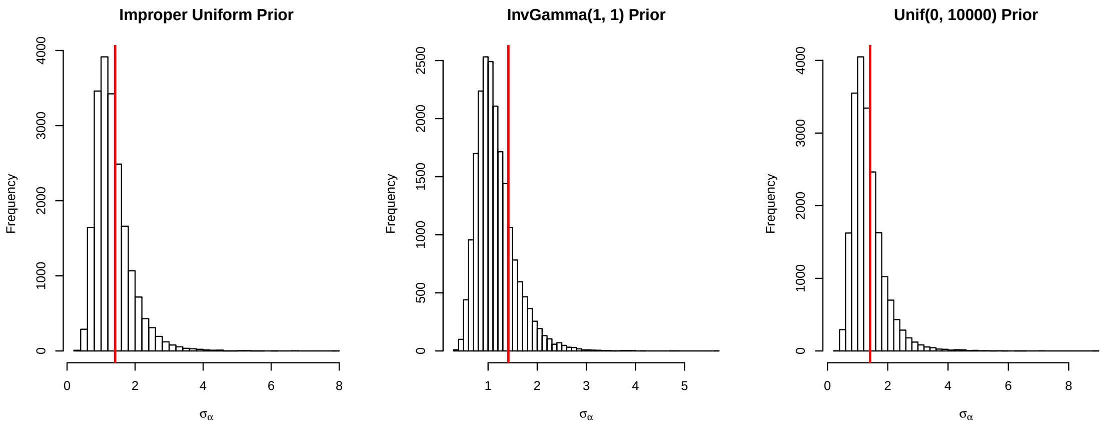
(e) I chose to use STAN to fit the models. I included model code in the appendix

i. I chose $\epsilon = 1$ so that an $Inv - Gamma(1,1)$ prior is used on σ^2_α. Gelman uses this prior in the 2006 prior, and although it appears to inform the posterior more than the improper uniform prior, it does not influence the posterior nearly as much as the $Gam(0.001, 0.001)$ prior. Additionally, I can't find a symmetric gamma prior that is more diffuse than the $Gam(1,1)$ that does not blow up at 0. Gelman says he does not recommend using the Inv-Gam(ϵ, ϵ) prior when the goal is to inform the posterior as little as possible.

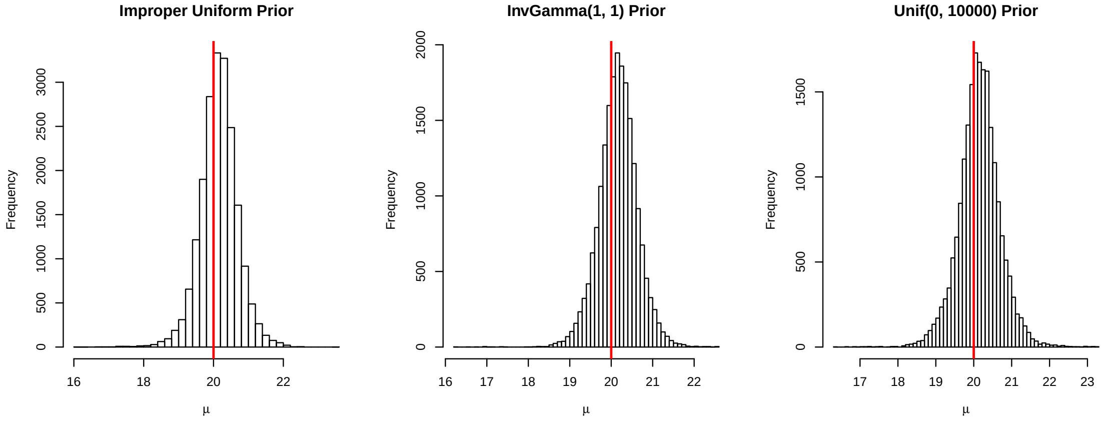


I just chose a very large number for A, $A = 10000$, with the goal of applying a very vague prior on σ_α^2. Normally, I would choose an A that represents a mathematical constraint on the upper bound of σ_α. For example, if I was using the schools example, I would say that a standard deviation of 100 is a reasonable maximum in the context of SAT scores. But, because I am choosing A for a general hierarchical model with no context, it's hard to choose an upper bound to reflect a natural constraint.

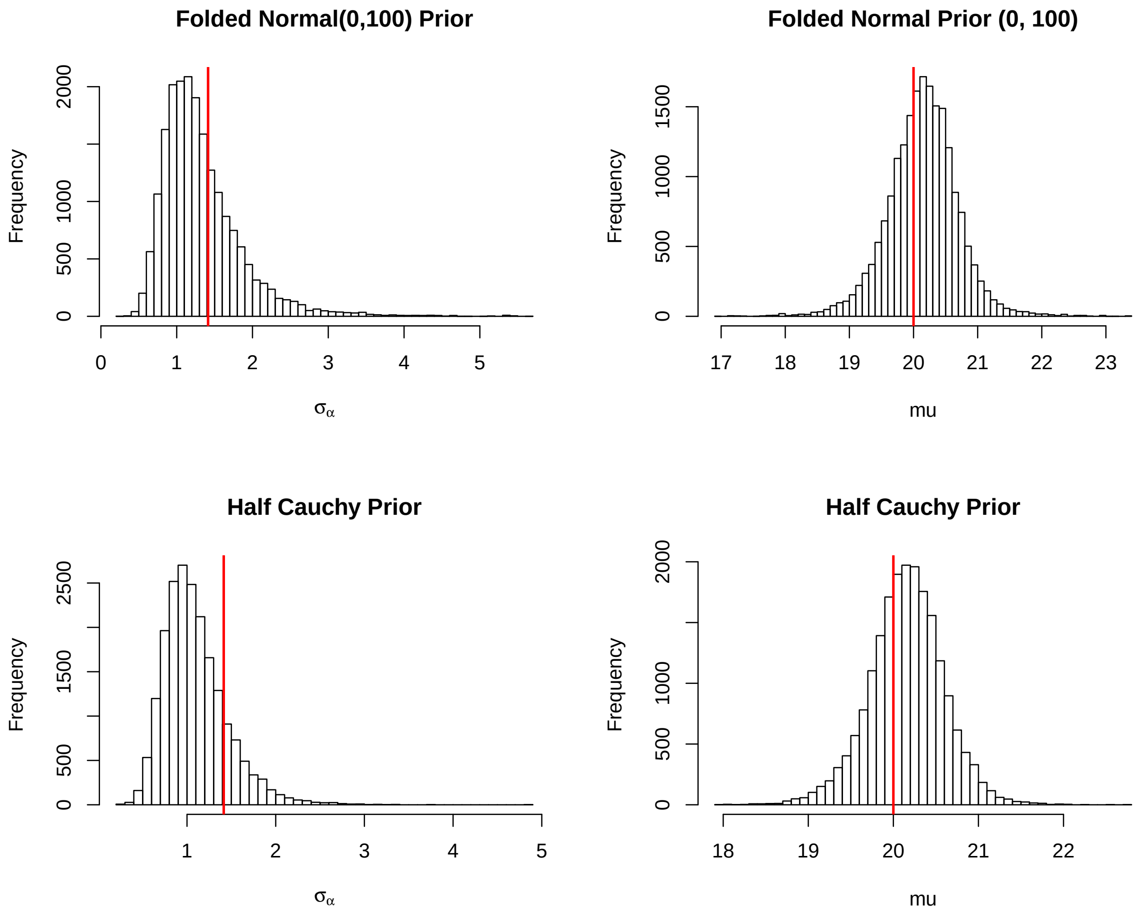
ii. Here I compare the posterior distributions of σ_α obtained from the three priors. The improper uniform prior has the largest range of posterior draws. I would expect this, because it is the most diffuse prior. The $Inv - Gamma(1, 1)$ prior does appear to inform the posterior because the range of posterior draws is the narrowest for this prior. It turns out the $Inv - Gamma(1, 1)$ prior is closer to being centered on the true σ_α, $\sqrt{2}$, than any of the other priors. I think this occured by coincidence; the inverse gamma prior happened to put more on values of σ_α closer to the truth. This was not intentional, however, and I don't think the $Inv - Gam(1, 1)$ prior should be thought of as non-informative.



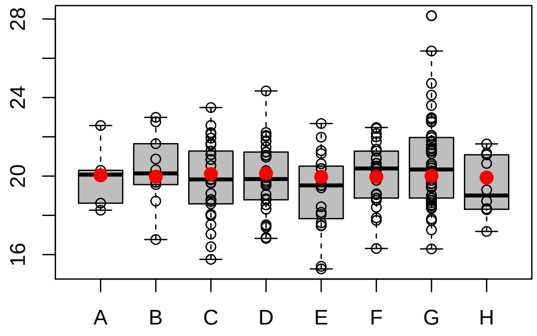
iii. When looking at the posterior distributions for μ obtained from the three priors, I noticed they are all centered around 20.5, just to the right of the true mean of 20. Results were fairly similar across all three prior distributions, but again we see that the spread of posterior draws for μ was widest when the uniform priors were used.



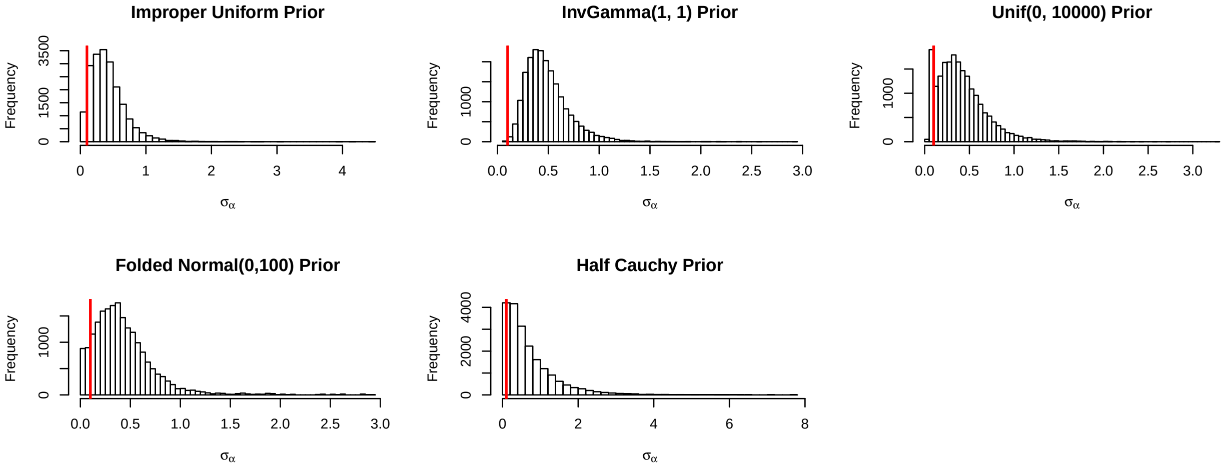
(f) The folded $N(0, 100)$ prior has a wider range of posterior draws for σ_α than the folded Cauchy. The range of posterior μ values is also slightly larger for the folded $N(0, 100)$. Both posterior distributions for μ look very similar to the posteriors from all priors in part (e). The posterior for σ_α from the half Cauchy looks similar to the posterior in part (e) from the inverse Gamma prior. The posterior for σ_α from the folded normal prior looks similar to the posterior from the uniform priors in (e).

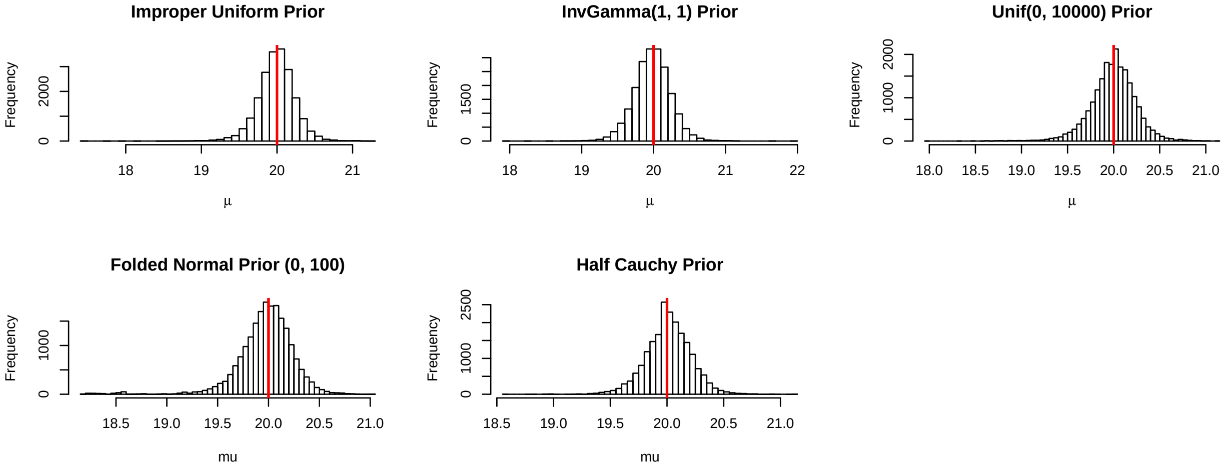


4. The new simulated data are shown below. I fit the same models as I did in problem 3 parts (e) and (f).



Below I plot the posterior distributions for μ and σ_α for the different priors.





(a) For all priors, the spreads of the posterior distributions for μ in problem 4 are smaller than problem 3. The posteriors for μ are still similar across priors, although the folded normal and improper uniform priors still have a wider range of values.

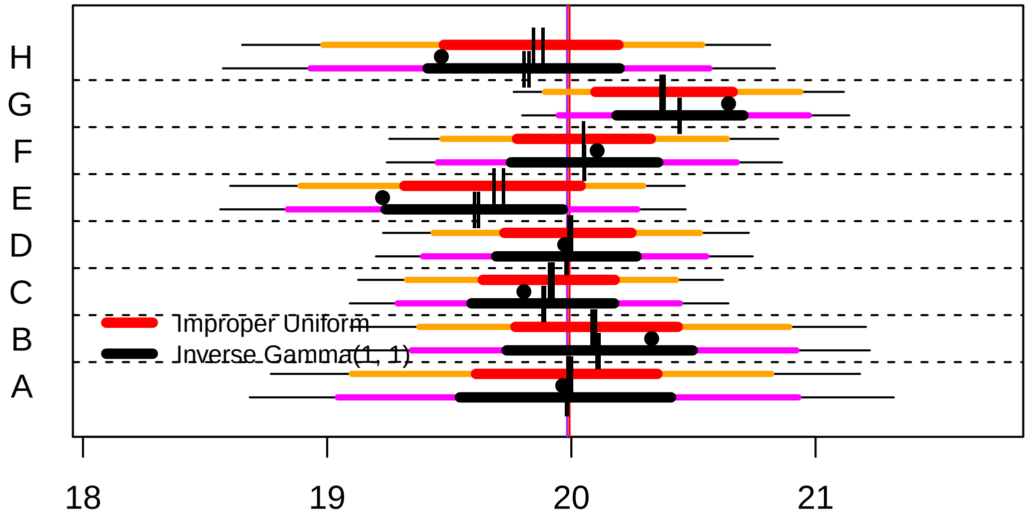
The posteriors for σ_α look very different than they did in problem 3. Most of them are bumped up against 0 and centered around 0.5, quite far from the true value of 0.1. All of the posteriors have a long right tail, and if the variability were any lower the posterior distributions may turn out to be improper. This is similar to what Gelman points out in section 5 of his 2006 paper when the number of groups is small (J=3 rather than

J=8), that if the variability among groups is small, or if the number of groups is small, the posterior distributions could turn out to be improper or very sensitive to the prior. In this case, the posteriors for σ_α end up overestimating the group to group variability because the vague priors used give more density to higher values of σ_α. Gelman says that this is unavoidable due to the asymmetry of the parameter space, but he recommends using the half Cauchy when the variability among groups is low. I do think the results from the half Cauchy look the best, but the posterior draws are still large compared to the true magnitude of σ_α.

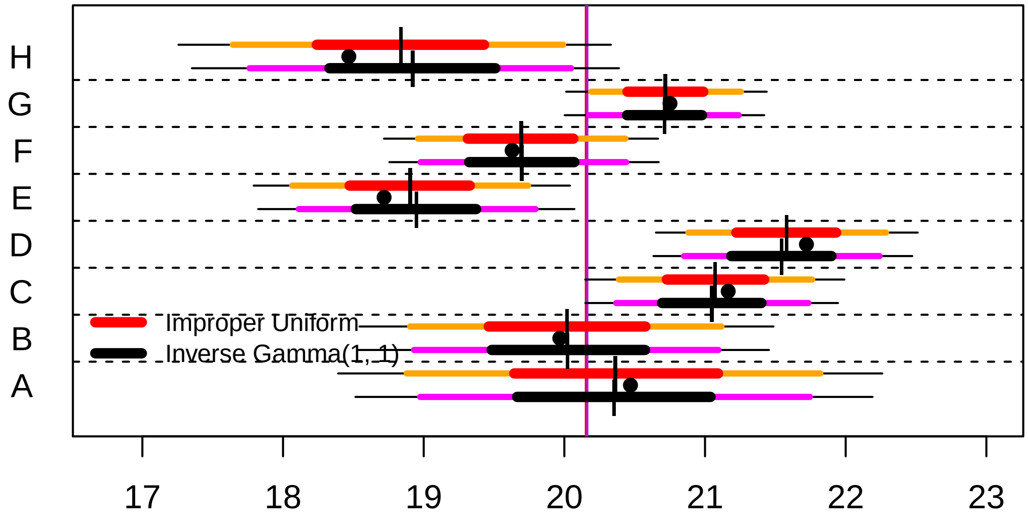
(b) See caterpillar plots on the next page. In problem 3, the degree is shrinkage is largest for the inverse gamma prior. I would expect this because the inverse gamma prior puts more density on lower values of σ_α. In the case of problem 4, however, the degree of shrinkage is largest for the improper uniform prior. I think this occurs because the density of the inverse gamma prior is concentrated above 0.5. So, in this case, the improper uniform actually has more density on lower values of σ_α than the inverse gamma prior, resulting in more shrinkage for the improper uniform prior.

The other thing I notice is that there is much more shrinkage overall in the results from problem 4, but the shrinkage should be more in problem 4 considering the true standard deviation among groups is only 0.1.

If we were testing for variability among the groups using classical methods, we would find no evidence of variability in problem 4, but in problem 3, we would find evidence of variability among the groups. We talked in class about how, with a Bayesian analysis, we don't have to choose between a separate means or a single mean model. But, in problem 4, the model greatly overestimates the variability among the groups and this seems to be unavoidable. When the variability among groups is small and we know there will be difficulties in estimating the group to group variability, would it be better to use the Bayesian model even though it overestimates variability among groups, or would it be better just to use a single mean model?



(c) Below is the caterpillar plot from problem 3. See part (b) for my discussion.



R Code Appendix - STAN model code

```
###improper uniform prior
data {
  int J;
  real y_1[5];
  real y_2[10];
  real y_3[30];
  real y_4[30];
  real y_5[20];
  real y_6[25];
  real y_7[50];
  real y_8[10];
}

parameters {
  real mu;
  real<lower=0> sigma_alpha;
  real<lower=0> sigma_y;
  real alpha[J];
}

transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] <- mu + alpha[j];
}

model {
  alpha ~ normal(0, sigma_alpha);

  y_1 ~ normal(theta[1], sigma_y);
  y_2 ~ normal(theta[2], sigma_y);
  y_3 ~ normal(theta[3], sigma_y);
  y_4 ~ normal(theta[4], sigma_y);
  y_5 ~ normal(theta[5], sigma_y);
  y_6 ~ normal(theta[6], sigma_y);
  y_7 ~ normal(theta[7], sigma_y);
  y_8 ~ normal(theta[8], sigma_y);
}
```

```
#inverse gamma prior
data {
  int J;
  real y_1[5];
  real y_2[10];
  real y_3[30];
  real y_4[30];
  real y_5[20];
  real y_6[25];
  real y_7[50];
  real y_8[10];
}
```

```
parameters {
  real mu;
  real<lower=0> sigma_alpha;
  real<lower=0> sigma_y;
  real alpha[J];
}

transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] <- mu + alpha[j];
}

model {
  sigma_alpha ~ inv_gamma(1, 1);
  alpha ~ normal(0, sigma_alpha);

  y_1 ~ normal(theta[1], sigma_y);
  y_2 ~ normal(theta[2], sigma_y);
  y_3 ~ normal(theta[3], sigma_y);
  y_4 ~ normal(theta[4], sigma_y);
  y_5 ~ normal(theta[5], sigma_y);
  y_6 ~ normal(theta[6], sigma_y);
  y_7 ~ normal(theta[7], sigma_y);
  y_8 ~ normal(theta[8], sigma_y);
}
```

```
#proper uniform prior
data {
  int J;
  real y_1[5];
  real y_2[10];
  real y_3[30];
  real y_4[30];
  real y_5[20];
  real y_6[25];
  real y_7[50];
  real y_8[10];
}

parameters {
  real mu;
  real<lower=0> sigma_alpha;
  real<lower=0> sigma_y;
  real alpha[J];
}

transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] <- mu + alpha[j];
}
```

```
model {
  sigma_alpha ~ uniform(0, 10000);
  alpha ~ normal(0, sigma_alpha);

  y_1 ~ normal(theta[1], sigma_y);
  y_2 ~ normal(theta[2], sigma_y);
  y_3 ~ normal(theta[3], sigma_y);
  y_4 ~ normal(theta[4], sigma_y);
  y_5 ~ normal(theta[5], sigma_y);
  y_6 ~ normal(theta[6], sigma_y);
  y_7 ~ normal(theta[7], sigma_y);
  y_8 ~ normal(theta[8], sigma_y);
}
```

```
#reparameterized with folded normal prior
data {
  real sigma_eta;
  int J;
  real y_1[5];
  real y_2[10];
  real y_3[30];
  real y_4[30];
  real y_5[20];
  real y_6[25];
  real y_7[50];
  real y_8[10];
}

parameters {
  real mu;
  real<lower=0> psi;
  real<lower=0> sigma_y;
  real eta[J];
}

transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] <- mu + psi * eta[j];
}

model {
  eta ~ normal(0, sigma_eta);
  psi ~ normal(0, 100);
  y_1 ~ normal(theta[1], sigma_y);
  y_2 ~ normal(theta[2], sigma_y);
  y_3 ~ normal(theta[3], sigma_y);
  y_4 ~ normal(theta[4], sigma_y);
  y_5 ~ normal(theta[5], sigma_y);
  y_6 ~ normal(theta[6], sigma_y);
  y_7 ~ normal(theta[7], sigma_y);
  y_8 ~ normal(theta[8], sigma_y);
}
```

```
#reparameterized folded Cauchy
data {
  int J;
  real y_1[5];
  real y_2[10];
  real y_3[30];
  real y_4[30];
  real y_5[20];
  real y_6[25];
  real y_7[50];
  real y_8[10];
}

parameters {
  real mu;
  real<lower=0> psi;
  real<lower=0> sigma_y;
  real eta[J];
  real<lower=0> sigma_eta;
}

transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] <- mu + psi * eta[j];
}

model {
  eta ~ normal(0, sigma_eta);
  psi ~ normal(0, 1);
  sigma_eta ~ normal(0, 1);
  y_1 ~ normal(theta[1], sigma_y);
  y_2 ~ normal(theta[2], sigma_y);
  y_3 ~ normal(theta[3], sigma_y);
  y_4 ~ normal(theta[4], sigma_y);
  y_5 ~ normal(theta[5], sigma_y);
  y_6 ~ normal(theta[6], sigma_y);
  y_7 ~ normal(theta[7], sigma_y);
  y_8 ~ normal(theta[8], sigma_y);
}
```