S626

HW5

John Koo

```r
library(ggplot2)
import::from(magrittr, `%>%`)

theme_set(theme_bw())
set.seed(626)
doMC::registerDoMC(8)
```

8.3

```r
# load the data
url.string <-
  'http://www.stat.washington.edu/people/pdhoff/Book/Data/hwdata/school'
filetype <- '.dat'
school.list <- lapply(seq(8), function(i) {
  # get the values for this school
  readLines(paste0(url.string, i, filetype)) %>%
    as.numeric()
})

# precompute summary statistics
ybar <- sapply(school.list, mean)
n <- sapply(school.list, length)
N <- sum(n)
```

```r
# set priors
mu0 <- 7
gamma0 <- sqrt(5)
tau0 <- sqrt(10)
eta0 <- sqrt(2)
sigma0 <- sqrt(15)
nu0 <- 2

# iterations
iter <- 1e4
```

a

```r
# preallocate
theta.out <- matrix(nrow = iter, ncol = 8)
sigma.out <- mu.out <- tau.out <- rep(NA, iter)

# initial guesses
# based on the data
theta <- ybar
```

```r
sigma <- sd(unlist(school.list))
mu <- mean(theta)
tau <- sd(theta)

# mcmc
for (i in seq(iter)) {
  # draw mu
  mu <- rnorm(
    1,
    (8 * mean(theta) / tau ** 2 + mu0 / gamma0 ** 2) /
      (8 / tau ** 2 + gamma0 ** -2),
    (8 / tau ** 2 + gamma0 ** -2) ** -.5
  )

  # draw tau
  tau <- rgamma(
    1,
    (eta0 + 8) / 2,
    (eta0 * tau0 ** 2 + sum((theta - mu) ** 2)) / 2
  ) ** -.5

  # draw thetas
  theta <- rnorm(
    8,
    (n * ybar / sigma ** 2 + mu / tau ** 2) / (n / sigma ** 2 + tau ** -2),
    (n / sigma ** 2 + tau ** -2) ** -.5
  )

  # draw sigma
  sigma <- rgamma(
    1,
    (nu0 + N) / 2,
    (nu0 * sigma0 ** 2 + sum(sapply(seq(8), function(j) {
      sum((school.list[[j]] - theta[j]) ** 2)
    }))) / 2
  ) ** -.5

  # store
  theta.out[i, ] <- theta
  sigma.out[i] <- sigma
  mu.out[i] <- mu
  tau.out[i] <- tau
}
```

Make sure effective sample sizes are large enough first since that's easier.

```r
# check effective sample sizes
coda::effectiveSize(mu.out)
```

```
    var1
8083.431
```

```r
coda::effectiveSize(tau.out)
```

```
    var1
6721.682
```

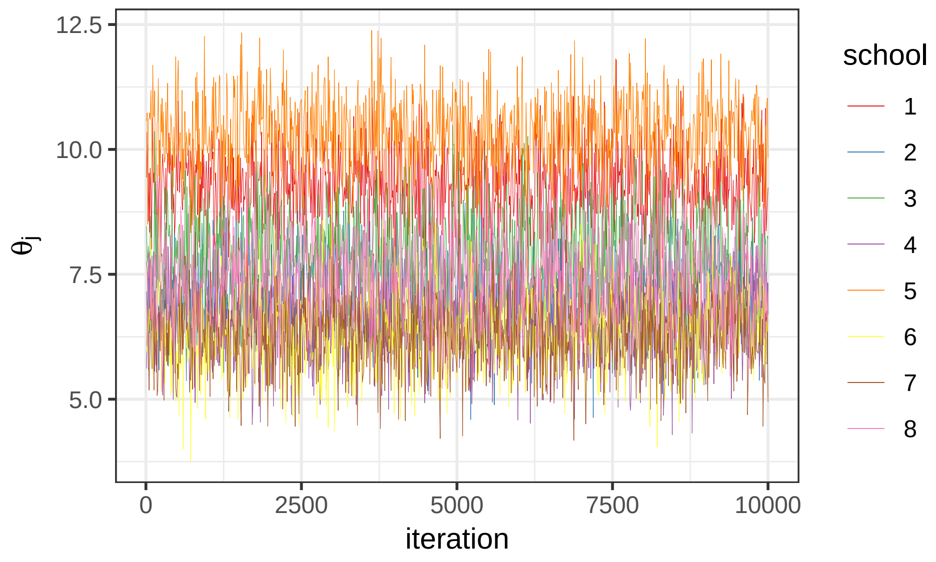
```
coda::effectiveSize(sigma.out)
```

```
    var1
9210.689
```

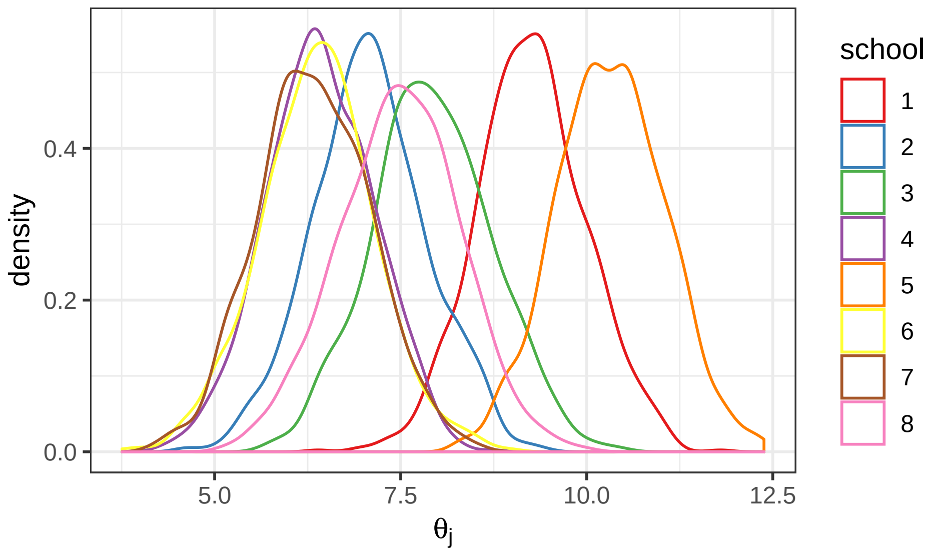
Check for convergence.

We can do a few other things, but for now, let's just do the simpliest thing, which is making sure all the samples are concentrated around a mode without any overall trend.

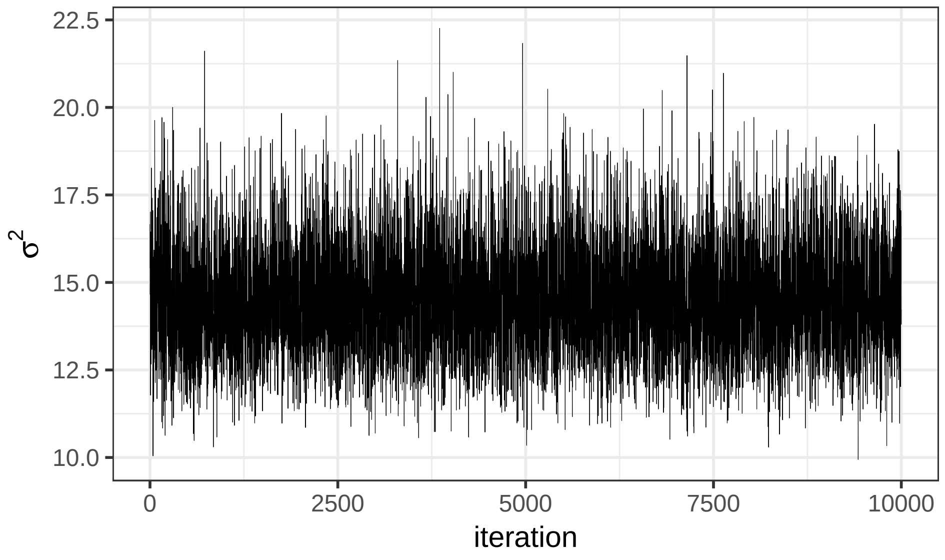
```
# plot for thetas
theta.df <- as.data.frame(theta.out) %>%
  dplyr::mutate(ind = seq(n())) %>%
  # subsample so plotting is faster
  dplyr::filter(ind %% 10 == 0)
theta.plot <- ggplot(theta.df) +
  labs(x = 'iteration', y = expression(theta[j]), colour = 'school') +
  scale_colour_brewer(palette = 'Set1')
for (j in seq(8)) {
  theta.plot <- theta.plot +
    geom_line(aes_string(x = 'ind', y = paste0('V', j),
                         colour = paste0('"', j, '"')),
              size = .1)
}
theta.plot
```



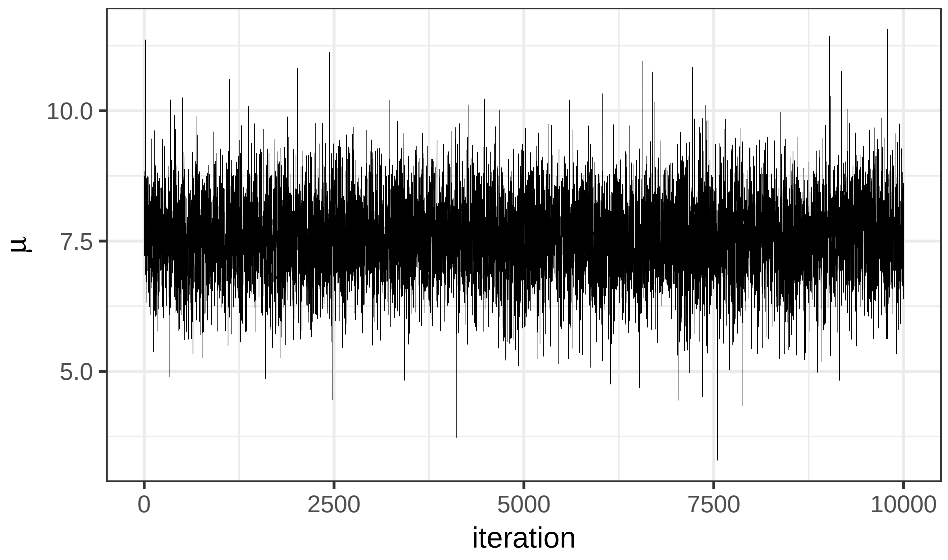
```
# and a density plot since the trace plot is hard to see
theta.df %>%
  tidyr::gather(school, theta, 1:8) %>%
  dplyr::mutate(school = gsub('V', '', school)) %>%
  ggplot() +
  geom_density(aes(x = theta, colour = school), fill = NA) +
  scale_colour_brewer(palette = 'Set1') +
  labs(x = expression(theta[j]))
```



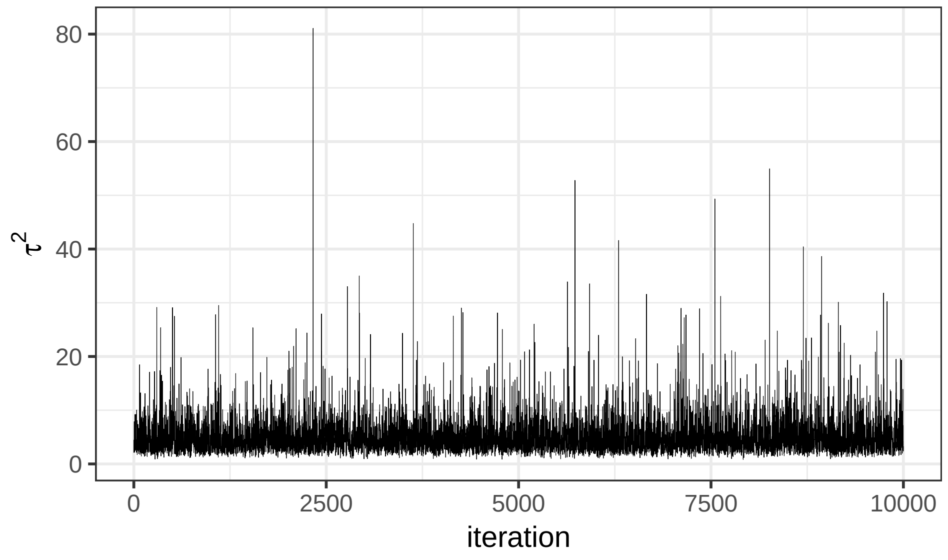
```
ggplot() +
  geom_line(aes(x = seq(iter), y = sigma.out ** 2), size = .1) +
  labs(x = 'iteration', y = expression(sigma^2))
```



```
ggplot() +
  geom_line(aes(x = seq(iter), y = mu.out), size = .1) +
  labs(x = 'iteration', y = expression(mu))
```



```
ggplot() +
  geom_line(aes(x = seq(iter), y = tau.out ** 2), size = .1) +
  labs(x = 'iteration', y = expression(tau^2))
```



All of the trace plots appear to have found some mode to oscillate around.

b

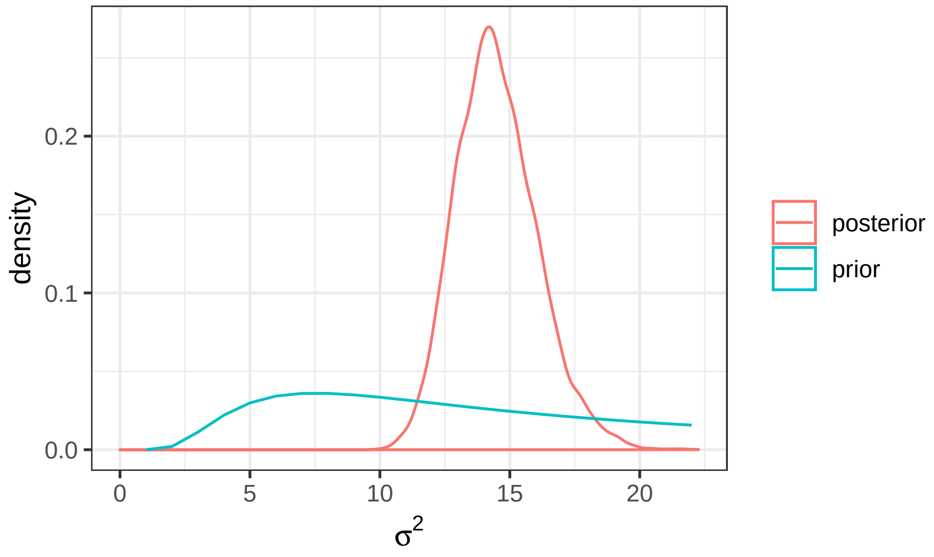
```
ci <- function(x, alpha = .05) quantile(x, c(alpha / 2, 1 - alpha / 2))
```

σ^2

```
# CI
ci(sigma.out ** 2)
```

```
    2.5%    97.5%
11.74695 17.87764
```

```
# prior and posterior
X <- seq(0, max(sigma.out ** 2))
prior <- invgamma::dinvgamma(X, nu0 / 2, nu0 * sigma0 ** 2 / 2)
ggplot() +
  geom_density(aes(x = sigma.out ** 2, colour = 'posterior')) +
  geom_line(aes(x = X, y = prior, colour = 'prior')) +
  labs(x = expression(sigma^2), colour = NULL)
```



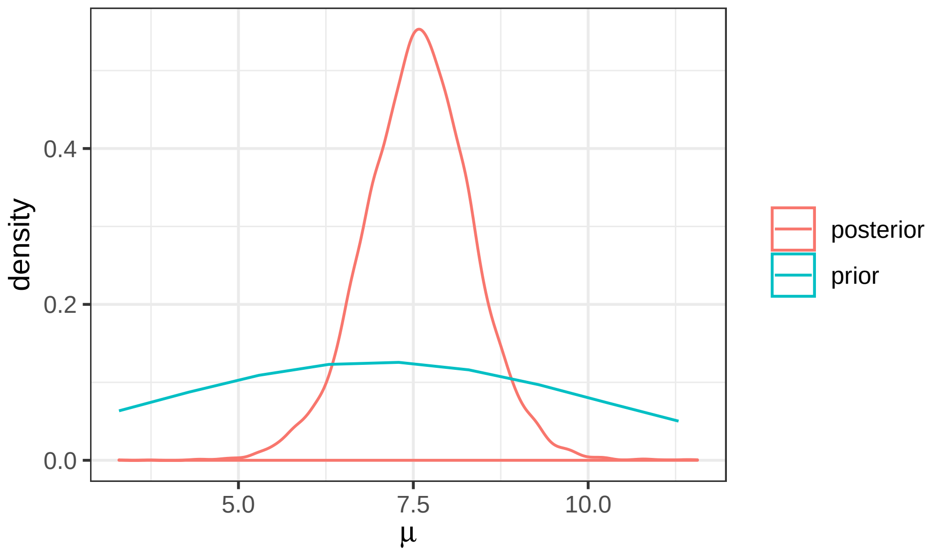
Our prior and posterior means do line up quite closely, but the plot tells a completely different story. This is because the prior is *very* heavily skewed to the right. The prior parameters were chose to reflect our uncertainty about the problem, but it resulted in a strange looking prior. But since the prior was relatively noninformative, the posterior quickly tightened up around a mode.

μ

```
# CI
ci(mu.out)
```

```
    2.5%    97.5%
6.002936 9.100119
```

```
# prior and posterior
X <- seq(min(mu.out), max(mu.out))
prior <- dnorm(X, mu0, tau0)
ggplot() +
  geom_density(aes(x = mu.out, colour = 'posterior')) +
  geom_line(aes(x = X, y = prior, colour = 'prior')) +
  labs(x = expression(mu), colour = NULL)
```



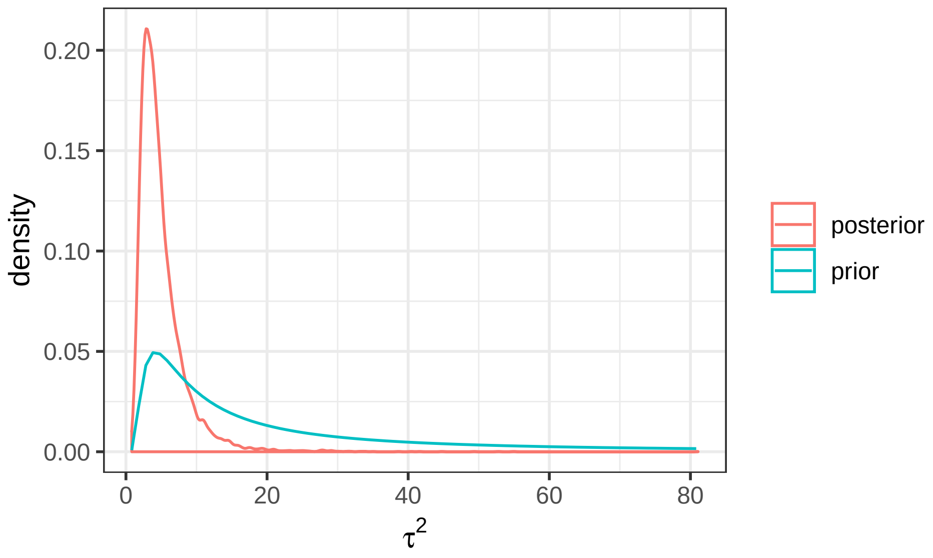
The prior and posterior means and modes line up relatively closely, but the posterior variance is much smaller than the prior variance.

τ^2

```r
# CI
ci(tau.out ** 2)
```

```
     2.5%     97.5%
 1.590965 13.661530
```

```r
# prior and posterior
X <- seq(min(tau.out ** 2), max(tau.out ** 2))
prior <- invgamma::dinvgamma(X, eta0 / 2, eta0 * tau0 ** 2 / 2)
ggplot() +
  geom_density(aes(x = tau.out ** 2, colour = 'posterior')) +
  geom_line(aes(x = X, y = prior, colour = 'prior')) +
  labs(x = expression(tau^2), colour = NULL)
```

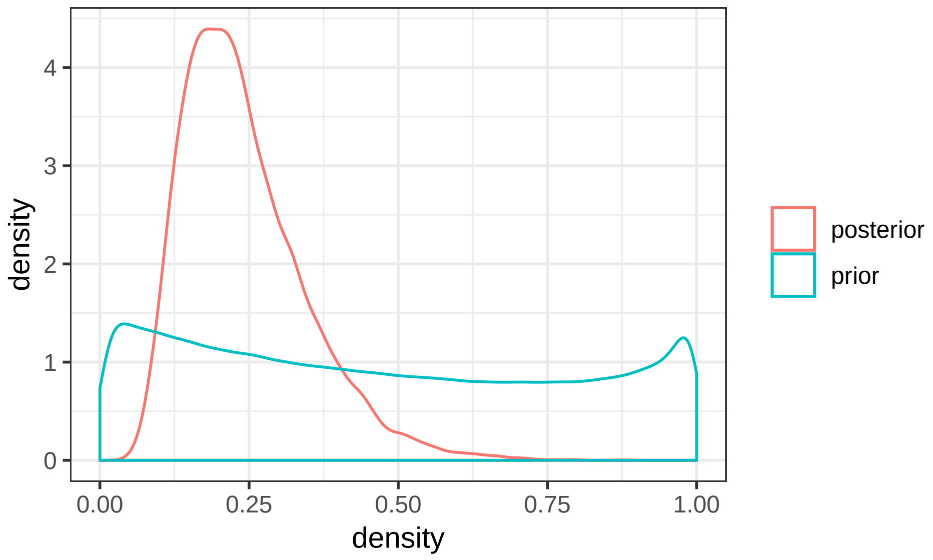


We see a similar behavior as we saw for σ^2 but to a much lesser degree.

c

I will use MC for the prior.

```r
tau.sq.prior <- rgamma(1e6, eta0 / 2, eta0 * tau0 ** 2 / 2) ** -1
sigma.sq.prior <- rgamma(1e6, nu0 / 2, nu0 * sigma0 ** 2 / 2) ** -1
R.prior <- tau.sq.prior / (tau.sq.prior + sigma.sq.prior)

R.posterior <- tau.out ** 2 / (tau.out ** 2 + sigma.out ** 2)

ggplot() +
  geom_density(aes(x = R.posterior, colour = 'posterior')) +
  geom_density(aes(x = R.prior, colour = 'prior')) +
  labs(x = 'density', colour = NULL)
```



The prior shows little to no knowledge of what R should be while the posterior is fairly confident that it is around 0.2, i.e., ~20% of the variation is between groups.

d

```r
mean(theta.out[, 7] < theta.out[, 6])
```

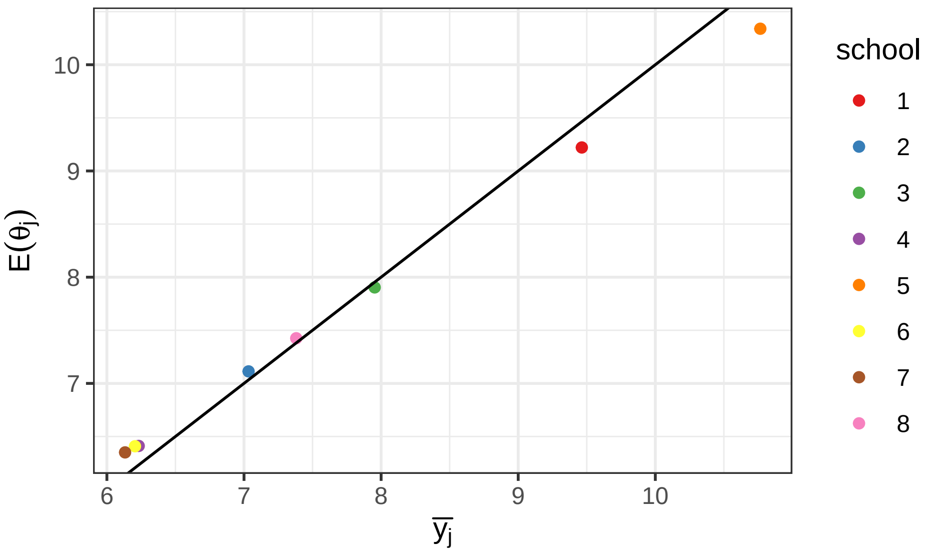
```
[1] 0.525
```

```r
mean(theta.out[, 7] <= apply(theta.out, 1, min))
```

```
[1] 0.3222
```

e

```r
ggplot() +
  geom_point(aes(x = ybar, y = apply(theta.out, 2, mean),
                 colour = as.character(seq(8)))) +
  geom_abline() +
  scale_colour_brewer(palette = 'Set1') +
  labs(x = expression(bar(y[j])), y = expression(E(theta[j])),
       colour = 'school')
```



There is some regression toward the mean, i.e., the slope between the two is less than 1, bringing up underperforming schools and bringing down high performing schools.

```r
# sample mean of data
sum(ybar * n) / N
```

```
[1] 7.691278
```

```r
# posterior mean of mu
mean(mu.out)
```

```
[1] 7.575149
```

```
# prior mean of mu
mu0
```

```
[1] 7
```

The posterior mean is between the sample and prior means, as expected. Since our confidence in the prior mean is low, the posterior mean is much closer to the sample mean.

9.1

```
dat.url <-
  'http://www.stat.washington.edu/people/pdhoff/Book/Data/hwdata/swim.dat'
swim.df <- read.table(dat.url) %>%
  t() %>%
  as.data.frame() %>%
  dplyr::mutate(week = seq(n())) %>%
  tidyr::gather(swimmer, time, seq(4))
```

a

I will use Zellner's g-prior.

Based on the information given, we might set the prior to:

$\beta_0 = (23, 0)^\top$ (based on information given about typical times)
$g = n$
$\nu_0 = 1$
$\sigma_0 = 1/2$ (assume 95% are between 22 and 24)

As a sanity check, we can try generating data from this prior and seeing if they match the information we are given.

```
beta0 <- c(23, 0)
g <- 6
nu0 <- 1
sigma0 <- .5

# draw x from 1 to 20 based on the fact that we have data from weeks 1-6
x <- sample(seq(20), iter, replace = TRUE)
x.model <- cbind(1, x)
xtx.inv <- solve(t(x.model) %*% x.model)

# draw sigma from prior
sigma <- rgamma(iter, nu0 / 2, nu0 + sigma0 ** 2 / 2) ** -.5

# draw beta from prior
beta <- sapply(seq(iter), function(i) {
  mvtnorm::rmvnorm(1,
                   beta0,
                   g * sigma[i] ** 2 * xtx.inv)
}) %>%
  t()
```

```
# draw error terms
e <- rnorm(iter, 0, sigma)

# draw responses
y <- sapply(seq(iter), function(i) {
  sum(x.model[i, ] * beta[i, ]) + e[i]
})

# check if the generated data make sense
summary(y)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-9442.04    21.48    23.00    20.81    24.48  3410.71 
```

This does not match our expectations. In particular, the minimum and maximum are too large. We can try increasing ν_0 to see if we can fix this.

```
nu0 <- 20

# draw sigma from prior
sigma <- rgamma(iter, nu0 / 2, nu0 + sigma0 ** 2 / 2) ** -.5

# draw beta from prior
beta <- sapply(seq(iter), function(i) {
  mvtnorm::rmvnorm(1,
                   beta0,
                   g * sigma[i] ** 2 * xtx.inv)
}) %>%
  t()

# draw error terms
e <- rnorm(iter, 0, sigma)

# draw responses
y <- sapply(seq(iter), function(i) {
  sum(x.model[i, ] * beta[i, ]) + e[i]
})

# check if these make sense
summary(y)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  15.44   22.00   22.98   22.98   23.96   29.36 
```

```
quantile(y, c(.025, .975))
```

```
    2.5%    97.5% 
20.05058 25.92064 
```

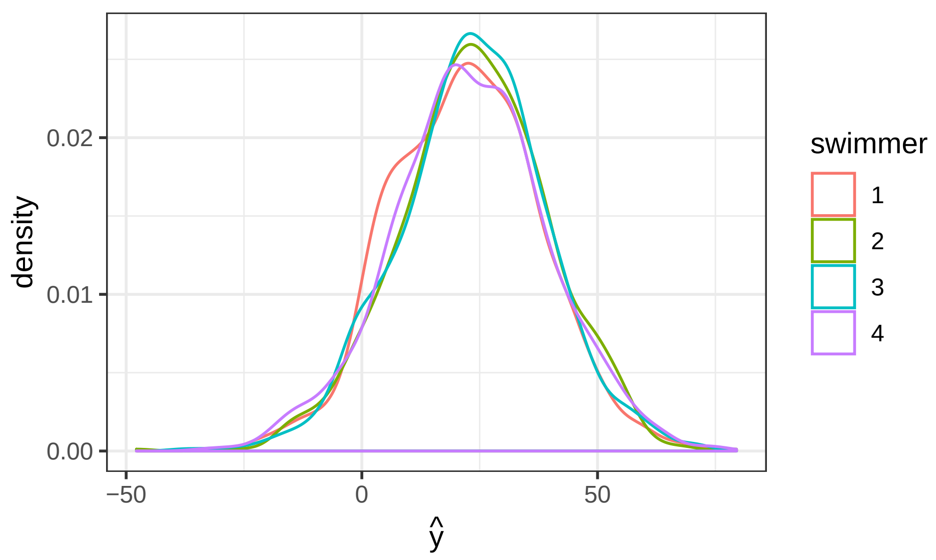
This seems more reasonable, so we'll use these for our prior.

```
# priors
beta0 <- c(23, 0)
g <- 6
nu0 <- 20
sigma0 <- 0.5
```

```r
# iterations
iter <- 1e3

# separate model for each swimmer
swimmers <- unique(swim.df$swimmer)
mcmc.out <- plyr::llply(swimmers, function(s) {
  # filter to just this swimmer
  temp.df <- dplyr::filter(swim.df, swimmer == s)

  # precompute
  X <- model.matrix(time ~ week, data = temp.df)
  XtX <- t(X) %*% X
  XtX.inv <- solve(XtX)
  H <- X %*% XtX.inv %*% t(X)
  beta.ols <- XtX.inv %*% t(X) %*% temp.df$time
  ssreg <- t(temp.df$time) %*% (diag(6) - g / (g + 1) * H) %*% temp.df$time

  # draw sigma
  sigma <- rgamma(
    iter,
    (nu0 + 6) / 2,
    (nu0 * sigma0 ** 2 + ssreg) / 2
  ) ** -.5

  # draw beta
  beta <- plyr::laply(sigma ** 2, function(s2) {
    beta <- mvtnorm::rmvnorm(
      1,
      g / (g + 1) * beta.ols + 1 / (g + 1) * beta0,
      s2 * g / (g + 1) * XtX.inv
    )
  })

  # draw y.pred
  y.pred <-  as.vector(beta %*% c(1, 20))

  list(beta = beta, sigma = sigma, y.pred = y.pred)

}, .parallel = TRUE)
```

b

```r
y.hat <- plyr::laply(mcmc.out, function(x) x$y.pred) %>%
  t()

y.hat %>%
  as.data.frame() %>%
  magrittr::set_colnames(seq(4)) %>%
  tidyr::gather('swimmer', 'y.hat', 1:4) %>%
  ggplot() +
  geom_density(aes(x = y.hat, colour = swimmer))  +
  labs(x = expression(hat(y)))
```



```r
sapply(seq(4), function(j) {
  sapply(seq(iter), function(i) {
    y.hat[i, j] <= min(y.hat[i, ])
  }) %>%
    mean()
})
```

```
[1] 0.272 0.224 0.239 0.265
```

There is too much uncertainty 2 weeks out to make a good prediction, with the first swimmer being slightly better. However, we can also note that these values do not make sense and are too spread out. So perhaps our prior is mis-specified.

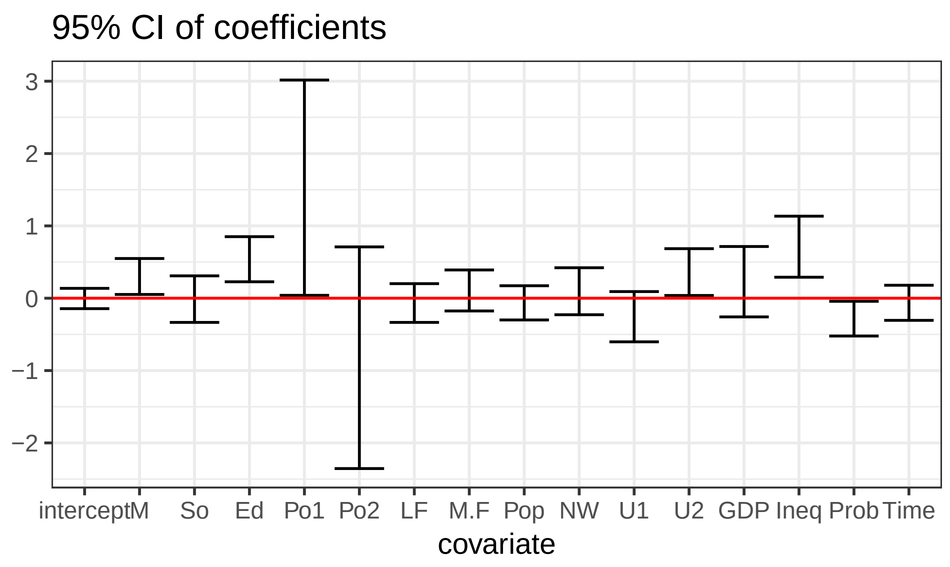
9.3

```r
# load the data
dat.url <-
  'http://www.stat.washington.edu/people/pdhoff/Book/Data/hwdata/crime.dat'
crime.df <- readr::read_table2(dat.url)

# summary statistics
n <- nrow(crime.df)
X <- model.matrix(y ~ ., crime.df)
XtX.inv <- solve(t(X) %*% X)
H <- X %*% XtX.inv %*% t(X)
beta.ols <- XtX.inv %*% t(X) %*% crime.df$y
```

a

```r
# priors
g <- n
nu0 <- 2
```

```r
sigma0 <- 1

# precompute
ssreg <- t(crime.df$y) %*% (diag(n) - g / (g + 1) * H ) %*% crime.df$y

# draw sigma
sigma <- rgamma(iter, (nu0 + n) / 2, (nu0 * sigma0 ** 2 + ssreg) / 2) ** -.5

# draw beta
beta <- plyr::laply(sigma ** 2, function(s2) {
  beta <- mvtnorm::rmvnorm(
    1,
    g / (g + 1) * beta.ols,
    s2 * g / (g + 1) * XtX.inv
  )
})

signif.df <- plyr::aaply(beta, 2, function(b) {
  quantile(b, c(.025, .975))
}) %>%
  as.data.frame() %>%
  dplyr::mutate(covariate = factor(c('intercept', colnames(crime.df[-1])),
                                   levels = c('intercept',
                                              colnames(crime.df[-1]))))

ggplot(signif.df) +
  geom_errorbar(aes(x = covariate, ymin = `2.5%`, ymax = `97.5%`)) +
  geom_abline(slope = 0, colour = 'red') +
  labs(title = '95% CI of coefficients')
```



Based on our results, it appears that `M`, `Ed`, `Po1`, `U2`, `Ineq`, and `Prob` are the most strongly predictive covariates.

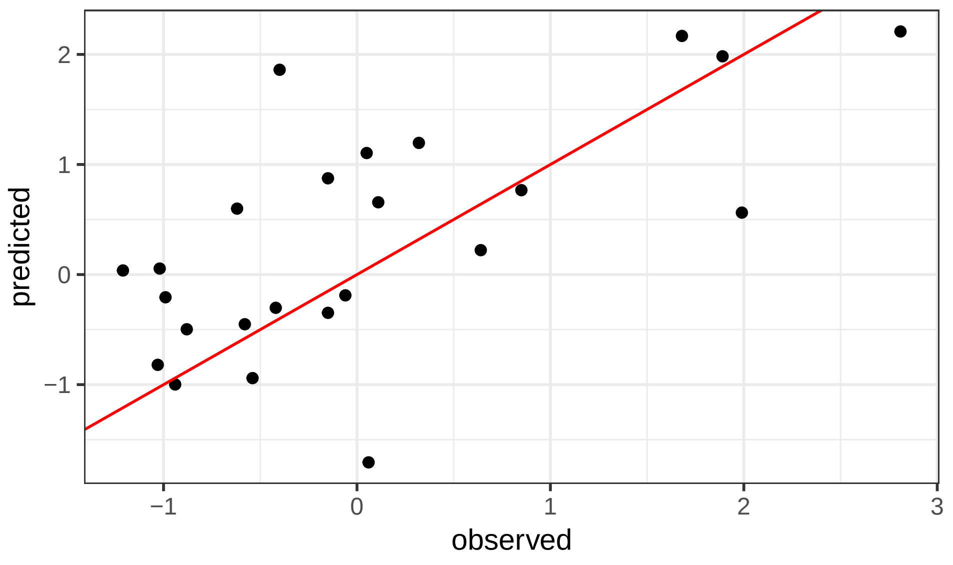
b

OLS

```r
# split the data
train.ind <- sample(n, n / 2)
train.df <- crime.df[train.ind, ]
test.df <- crime.df[-train.ind, ]
test.model.matrix <- model.matrix(y ~ ., test.df)

# fit OLS model on train data
beta.ols <- train.df %>%
  lm(y ~ ., data = .) %>%
  coef()

# fit to test data
yhat.test <- test.model.matrix %*% beta.ols

# training error on test data
mean((test.df$y - yhat.test) ** 2)
```

```
[1] 0.8140186
```

```r
ggplot() +
  geom_point(aes(x = test.df$y, y = yhat.test)) +
  geom_abline(colour = 'red') +
  labs(x = 'observed', y = 'predicted')
```



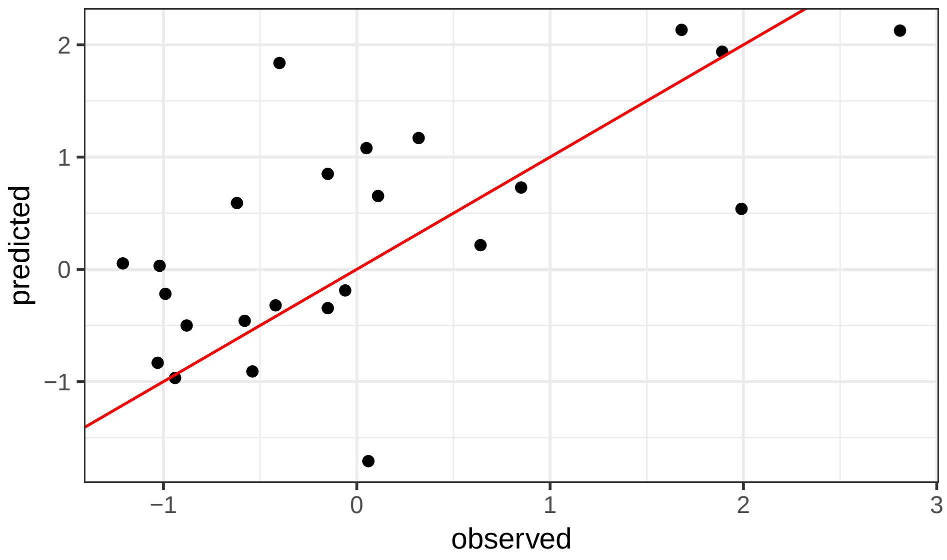
Bayes with g-prior

```r
# summary statistics for training data
n <- nrow(train.df)
X <- model.matrix(y ~ ., train.df)
XtX.inv <- solve(t(X) %*% X)
```

```r
H <- X %*% XtX.inv %*% t(X)
beta.ols <- XtX.inv %*% t(X) %*% train.df$y
ssreg <- t(train.df$y) %*% (diag(n) - g / (g + 1) * H ) %*% train.df$y

# draw sigma
sigma <- rgamma(iter, (nu0 + n) / 2, (nu0 * sigma0 ** 2 + ssreg) / 2) ** -.5

# draw beta
beta.bayes <- plyr::laply(sigma ** 2, function(s2) {
  beta <- mvtnorm::rmvnorm(
    1,
    g / (g + 1) * beta.ols,
    s2 * g / (g + 1) * XtX.inv
  )
}) %>%
  apply(2, mean)

# fit to test data
yhat.test <- test.model.matrix %*% beta.bayes

# training error on test data
mean((test.df$y - yhat.test) ** 2)
```

```
[1] 0.8061834
```

```r
ggplot() +
  geom_point(aes(x = test.df$y, y = yhat.test)) +
  geom_abline(colour = 'red') +
  labs(x = 'observed', y = 'predicted')
```



The results are very similar. We might expect different results if there were large outliers, but it appears that this is not the case.