1. *(5 points) How do Monte Carlo and Markov Chain Monte Carlo (MCMC) procedures differ?*

   MC procedures involve sampling from the marginal posterior distribution $p(\sigma^2|y_\sim)$ and using that value to sample from the full conditional $p(\theta|\sigma^2, y_\sim)$. All σ^2 are independently drawn, making each set $\phi = \theta_i$, σ_i^2 independent.

   MCMC differs in that apart from the starting value of σ^2, θ_i and σ_i^2 are iteratively sampled from their respective full conditional posterior distributions. In MCMC, each iteration is dependent on the previous one, and only the previous one. Conditional on the previous set of outcomes, the current set of outcomes is independent of all other previous sets of outcomes - "a signifying property of a Markov Chain".

   MC procedures have perfect mixing, because each sample is independent of the previous one. In MCMC procedures, the larger the number of simulations, the more likely that mixing has occured.

   Side note: It seems that estimates from iterations more than "one lag" apart would be correlated. – From class: the algorithm only uses the input from the previous iteration, and doesn't account for the fact that the previous iteration depended on say the (n-2)th iteration.

2. *(5 points) How are the full conditional distributions used in a Gibbs Sampler?*

   The full conditional distributions are used iteratively such that the random outcome from one full conditional becomes the input for the full conditional on the other parameter.

   The idea is to approximate the joint distribution $p(\theta, \sigma^2|y_\sim)$ with $p(\theta_{i+1}|\sigma_i^2, y_\sim)$*$p(\sigma_i^2|\theta_i, y_\sim)$ from many pairs of θ_i and σ_i^2's.

3. *(5 points) How do the marginal and conditional posterior distributions differ?*

   A marginal posterior distribution is a pdf of the parameter of interest conditional on the data, for example, $p(\theta|y_\sim)$. A conditional posterior is a pdf of the parameter of interst conditional on other parameters and conditional on the data, for example, $p(\theta|\sigma^2, y_i)$. The difference is that the marginal posterior is not conditioned on other parameters, while the conditional is conditioned on other parameters. MC uses a marginal posterior distribution on the precison and MCMC uses a conditional posterior distribution on the precision.

4. *(5 points) How do you visually assess convergence of an MCMC algorithm?*

   To visually assess convergence of an MCMC algorithm, a trace plot of the outcomes should show movement from lower to higher density regions, so that outcomes are not "stuck" in one region .

   With a trace plot, convergence would be met if the distribution of the chains is consistent as the number of iterations increase.

5. *(5 points) Explain the idea of the effective sample size of an MCMC algorithm.* Let ϕ = the variance of the parameter set

   $Var_{MCMC}[\phi] = Var_{MC}[\phi]$ + f(autocorrelation)

where f(autocorrelation) is a function of the autocorrelation and usually positive, the variance of MCMC approximations are generally larger than the variance of MC estimates.

If a researcher is interested in a minimum $Var_{MCMC}[\phi]$, we can find S=number of samples such that the $Var_{MC}[\phi] = \frac{\sigma^2}{S}$ = minimum $Var_{MCMC}[\phi]$. The value of S that permits the equality is the effective sample size.

Having a minimum variance of $Var_{MCMC}[\phi]$ is like estimating the $Var_{MC}[\phi]$ from a sample size of the effective sample size.

6. *(15 points) Sketch out the steps for a Gibbs sampler algorithm.*

0. Choose an initial σ_o^2 to begin with.

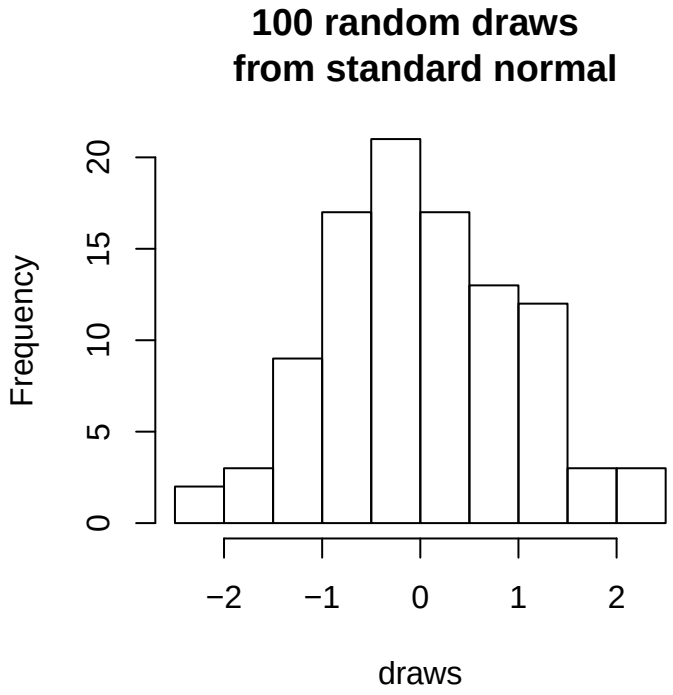
1. Randomly draw a θ from the full conditional, $\theta|\sigma_i^2, y_{\sim}$ N(μ, $\frac{\sigma_i^2}{n}$), denoted θ_i and note the first iteration, $\sigma_i^2 = \sigma_o^2$

2. Randomly draw a σ^2 from the full conditional, $\sigma^2|\theta_i, y_{\sim}$ INVGAM, denoted σ_i^2

3. Iteratively repeat steps (1) and (2), using σ_i^2 in place of σ_o until approximations achieve convergence and proper mixing

7. *Simulating data is a key step in verifying your algorithms are working correctly. This will be more apparent as we start studying sophisticated hierarchical models.*

   (a) *(10 points) Simulate 100 observations from a standard normal distribution and plot a his- togram of your data.*



(b) *(10 points) Select and state prior distributions for θ the mean of the normal distribution and σ^2 the variance (or alternatively you may parameterize your model using the precision term).*

Below is what you would set up for an MC sampler. In problem (c) I wrote out what I would start with for the Gibbs sampler.

PRIOR DISTRIBUTIONS

$\theta|\sigma^2 \sim \text{N}(\mu, \frac{\sigma^2}{\kappa_o})$

$\frac{1}{\sigma^2} \sim \text{GAM}(\frac{\nu_o}{2}, \frac{\nu_o \sigma_o^2}{2})$

Let $\mu = 0$ $\sigma_o = \text{var(prior samples)} = 1$, I'm assuming this cannot be taken from the var($y_\sim$) where y was generated in part (a)

κ_o = "hypothetical" sample size = 1, bigger = more informative prior on θ

ν_o = number prior samples = 1, bigger = more informative prior on σ^s

(c) *(20 points) Implement a Gibbs sampler to simulate from the joint posterior distribution $p(\theta, \sigma^2|y_1, \cdot, y_{100})$. Create a plot of the joint posterior distribution.*

To set up full conditionals on the parameters of a normal distribution, the distributions are more complicated than in part (b). We generally have to use the Gibbs sampler and the product of the full conditionals to estimate the joint posterior when the variance of the prior on θ is not directly

proportional to σ^2, making closed form solutions to the posteriors more difficult to find.

$\theta|\sigma^2 \sim \text{N}(\mu_n, \tau_n^2)$

$\sigma^2|\theta \sim \text{INVGAM}(\frac{\nu_n}{2}, \frac{\nu_n \sigma_n^2}{2})$

$\mu_n = \frac{\frac{\mu_o}{\tau_o^2} + \frac{n*ybar}{\sigma_n^2}}{\frac{1}{\tau_o^2} + n*\sigma^2}$

$\tau_n^2 = (\frac{1}{\tau_o^2} + \frac{n}{\sigma^2})^{-1}$

$\nu_n = \nu_o + n$

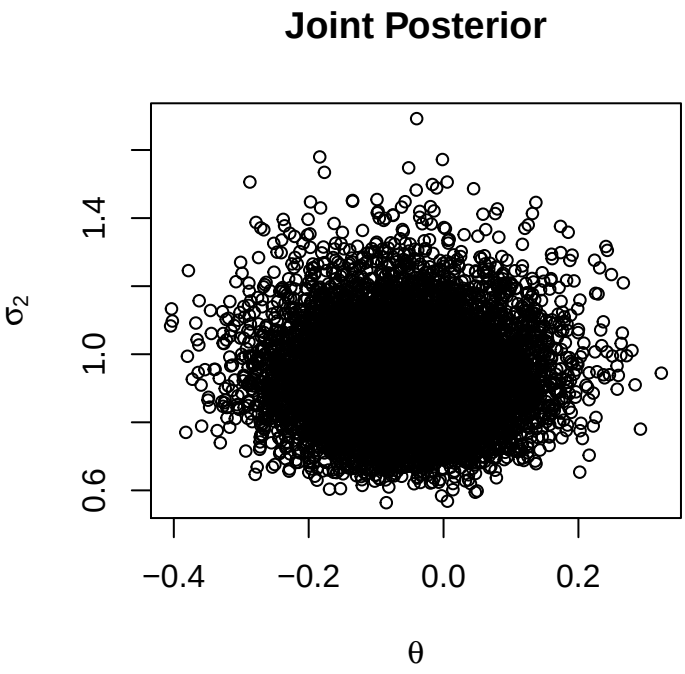
$\sigma_n^2(\theta) = \frac{1}{\nu_n} * [\nu_o \sigma_o^2 + n s_n^2(\theta)]$

$s_n^2(\theta) = \frac{\Sigma(y_i - \theta)^2}{n}$

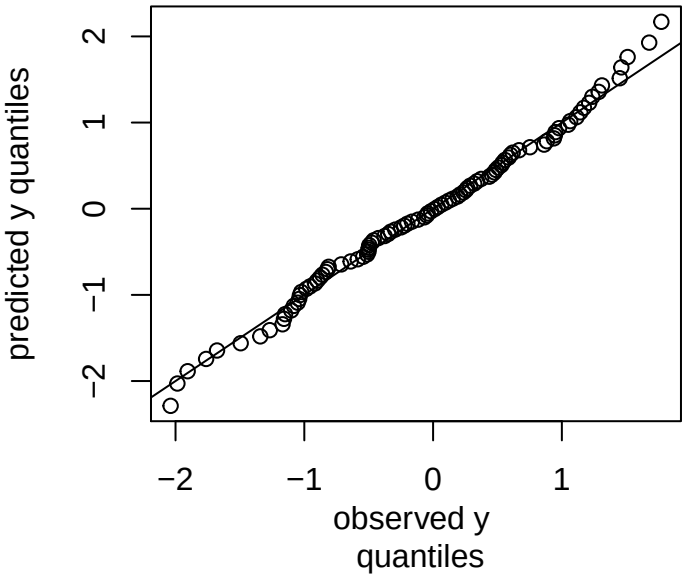
$\tau_o = \frac{\sigma_o}{n}$

Let n = 100, $\nu_o = 1$, $\sigma_o^2 = 1$,$\mu_o = 0$, and $\tau_o^2 = 1$ which would suggest an uninformative prior, as I don't have prior knowledge or information.

Generate y_i from a standard normal distribution.



(d) *(10 points) Use your MCMC samples to create a posterior predictive distribution. Compare the data and your posterior predictive distribution using a QQ plot qqnorm(.). Comment on the figure.*



	Var(y)	Vary(ystar)
1	0.91	0.93

The QQplot shows that the quantiles of the posterior predicted values are almost exactly the same as the quantiles from the intially generated data, with slight deviations in the left tail. If I had used a more informative prior, there should be less of a relationship between the quantiles. The table shows there was more variation in the predictions than in the originally generated y's, which is what was expected. The posterior predictive distribution is doing a good job of approximating the distribution of the original y's.

R Code

7. (a)
```
set.seed(53204)
y <- rnorm(100,0,1)

hist(y, main = "100 random draws \n from standard normal", xlab = "draws")
```

(c)
```
# this is my code, and I could not get it to work
set.seed(53204)
n <- 100
y <- rnorm(n,0,1)
```

```
nuo <- 1
nun <- nuo + n
so <- 1
mo <- 1
tau2o <- so/n


precision <- c(rep(0,n+1))
t <- c(rep(0,n))

tau2n <- c(rep(0,n+1))
mun <- c(rep(0,n+1))

s2ntheta <- c(rep(0,n))
sigma2ntheta <- c(rep(0,n))

mun[1] <- ((mo/tau2o) + n*mean(y)/so)/((1/tau2o) + n*so)
tau2n[1] <- ((1/tau2o) + n*so)^(-1)
precision[1] <- 1


for(i in 1:n){

  t[i] <- rnorm(1,mun[i], sqrt(1/tau2n[i]))

  s2ntheta[i] <- sum((y-t[i])^2)/n

  sigma2ntheta[i] <- (1/nun)*(nuo*so +
                                n*sigma2ntheta[i])

  precision[i+1] <- rgamma(1,nun/2,nun*sigma2ntheta[i]/2)

  mun[i+1] <- ((mo/tau2o) +
  n*mean(y)/1/precision[i+1])/((1/tau2o) + n*1/precision[i+1])

  tau2n[i+1] <- ((1/tau2o) + n*1/precision[i+1])^(-1)



}
```

```
# this is the code I actually used for 7c
num.obs <- 100
mu.true <- 0
sigmasq.true <- 1
y <- rnorm(num.obs,mu.true,sigmasq.true)
mean.y <- mean(y)
var.y <- var(y)
```

```
### initialize vectors and set starting values and priors
num.sims <- 10000
Phi <- matrix(0,nrow=num.sims,ncol=2)
Phi[1,1] <- 0 # initialize theta
Phi[1,2] <- 1 # initialize (1/sigmasq)
mu.0 <- 0
tausq.0 <- 1
nu.0 <- 1
sigmasq.0 <- 1

for (i in 2:num.sims){
  # sample theta from full conditional
  mu.n <- (mu.0 / tausq.0 + num.obs * mean.y *Phi[(i-1),2]) /
    (1 / tausq.0 + num.obs * Phi[(i-1),2] )
  tausq.n <- 1 / (1/tausq.0 + num.obs * Phi[(i-1),2])
  Phi[i,1] <- rnorm(1,mu.n,sqrt(tausq.n))

  # sample (1/sigma.sq) from full conditional
  nu.n <- nu.0 + num.obs
  sigmasq.n.theta <- 1/nu.n*(nu.0*sigmasq.0 + sum((y - Phi[i,1])^2))
  Phi[i,2] <- rgamma(1,nu.n/2,nu.n*sigmasq.n.theta/2)
}

# plot joint posterior
plot(Phi[,1],1/Phi[,2],xlim=range(Phi[,1]),ylim=range(1/Phi[,2]),cex=.8,
     ylab=expression(sigma[2]), xlab = expression(theta),
     main='Joint Posterior')
```

(d)