Tutorial on Bayesian Statistics: Replicating Sections of BDA3

Fernando Hoces de la Guardia

1 - Replicating example from pg 66-67

In Ch3 of BDA they present a first exercise of simulation (pg 66-67) where from a sample of 66 obs with mean 26.2 and standard deviation 10.8, they simulate the posterior for the mean assuming a joint normal likelihood and uninformative prior (where $\sigma^2|y \sim Inv-\chi^2(n-1,s^2)$ and $\mu|\sigma^2,y \sim N(\bar{y},\sigma^2/n)$).

```
# Draw 1000 random numbers from a Inverse chi-squared
f.sim.sigma   = function(draws) (n-1)*(se^2)/rchisq(draws,n-1)
f.sim.norm    = function(draws) rnorm(draws,ybar,(f.sim.sigma(draws)/(n-1))^.5)
sim.norm      = f.sim.norm(1000)
ci.mu         = sim.norm[order(sim.norm)][c(25,975)]
print(ci.mu)
```

```
## [1] 23.45 28.83
```

```
# And now repeating the similation excercise R = 100 times:
R             = 100
R.sim.norm    = sapply(1:R, function(x) f.sim.norm(1000))
ci.mu         = t(apply(R.sim.norm,2,function(x) x[order(x)][c(25,975)]))
print(c(median_low=median(ci.mu[,1]), median_high=median(ci.mu[,2]), p10_low =quantile(ci.mu[,1],c(.1),
```

```
##  median_low median_high p10_low.10%  p90_up.90%
##       23.53       28.87       23.37       29.04
```

---

2 - Simulating values from a postetior normal with σ^2 and μ unknown and conjugate prior.

The following is not an exercise in the book (that I'm aware of), but I though that it would be interesting to work on this. After going over the model derived in pg 67-69 in BDA3, I will simulate values from the joint posterior (and hopefully leave the code flexible enough to play later on with the hyper-parameters).

With the prior density:

$$p(\mu,\sigma^2) \propto \sigma^{-1}(\sigma^2)^{(\nu_0/2+1}exp\left(-\frac{1}{2\sigma^2}[\nu_0\sigma_0^2+\kappa_0(\mu_0-\mu)^2]\right)$$

And a likelihood:

$$p(y|\mu,\sigma^2) \propto \sigma^{-n}exp\left(-\frac{1}{2\sigma^2}[(n-1)s^2+n(\bar{y}-\mu)^2]\right)$$

We obtain the following joint posterior:

$$
\begin{aligned}
p(\mu, \sigma^2|y) &\propto \sigma^{-1}(\sigma^2)^{(\nu_0/2+1}exp\left(-\frac{1}{2\sigma^2}[\nu_0\sigma_0^2 + \kappa_0(\mu_0-\mu)^2]\right) \times \\
&\times (\sigma^2)^{-n/2}exp\left(-\frac{1}{2\sigma^2}[(n-1)s^2 + n(\bar{y}-\mu)^2]\right) \\
&= N - Inv - \chi^2(\mu_n, \sigma_n^2; \nu_n, \sigma_n^2) \\
\mu_n &= \frac{\kappa_0}{\kappa_0+n}\mu_0 + \frac{n}{\kappa_0+n}\bar{y} \\
\kappa_n &= \kappa_0 + n \\
\nu_n &= \nu_0 + n \\
\nu_n\sigma_n^2 &= \nu_0\sigma_0^2 + (n-1)s^2 + \frac{\kappa_0 n}{\kappa_0+n}(\bar{y}-\mu_0)^2.
\end{aligned}
$$

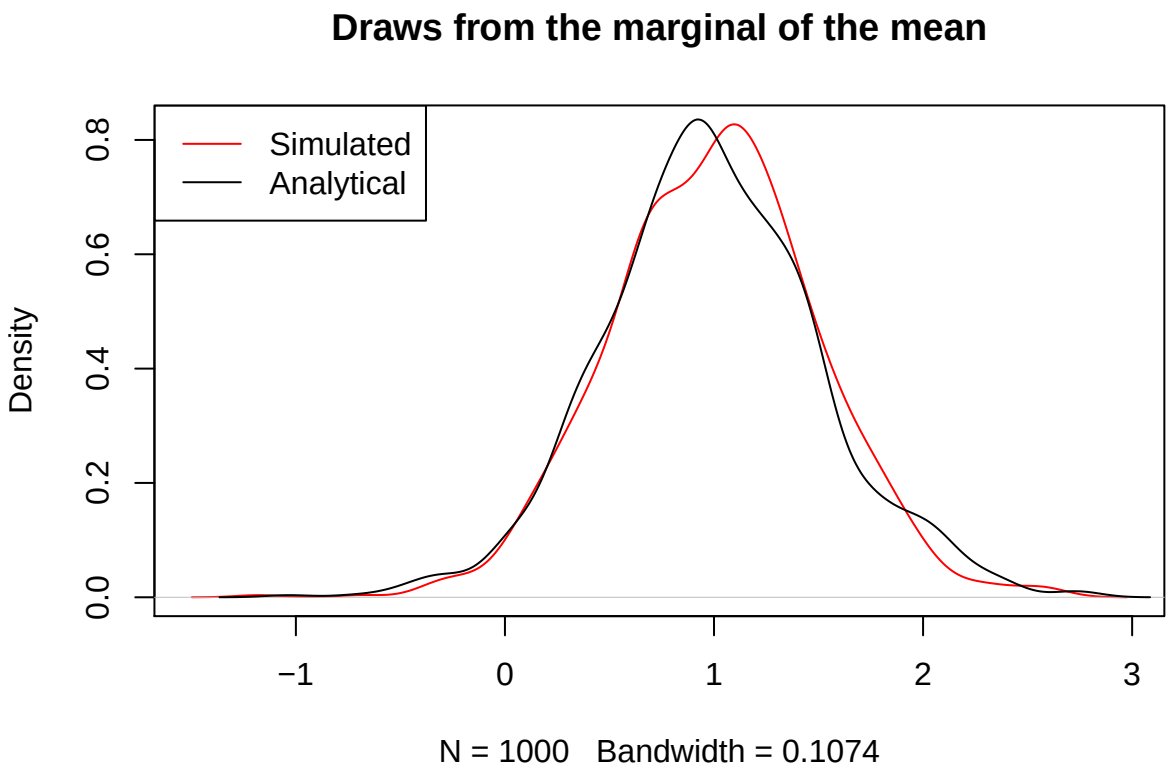
Using the fact that $p(\mu, \sigma^2|y) = p(\mu|\sigma^2, y)p(\sigma^2|y)$ with:

$$
\begin{aligned}
\mu|\sigma^2, y &\sim N(\mu_n, \sigma_n^2/\kappa_n) \\
\sigma^2|y &\sim Inv - \chi^2(\nu_n, \sigma_n^2) \\
labelpost.sig
\end{aligned}
$$

The simulation algorithm is as follows:
Draw a random number from the conditional posterior of sigma (equation \ref{post.sig})
Using that value of σ^2, draw a random number for the conditional posterior of μ (equation \ref{post.mu})

The following code implements this algorithm.

```
# Hyperparameters
p.mu_0      = 0
p.sigma2_0  = 1
p.nu_0      = 4
p.kappa_0   = 3

# Data
n           = 10
y_bar       = 1.3
s           = 2

# Simulation
mu_n        = ((p.kappa_0)/(p.kappa_0 + n)) * p.mu_0 + ((n)/(p.kappa_0 + n)) * y_bar
kappa_n     = p.kappa_0 + n
nu_n        = p.nu_0 + n
sigma2_n    =  (p.nu_0*p.sigma2_0 + (n-1)*s^{2} + ((p.kappa_0*n)/(p.kappa_0 + n)) *  (y_bar - p.mu_0)^{

set.seed(142857)
# Draw 1000 random numbers from a Inverse chi-squared
f.mu.post  = function(draws) {
  sigma2.cond = (nu_n)*(sigma2_n)/rchisq(draws,nu_n)
  mu.cond     = rnorm(draws,mu_n,(sigma2.cond/(kappa_n))^.5)
  return(mu.cond)
  }
sim.mu.post= f.mu.post(1000)
plot(density(sim.mu.post), main = "Draws from the marginal of the mean", col="red")
lines(density(mu_n +((sigma2_n/kappa_n)^.5)*rt(1000,nu_n)))
legend("topleft",col = c("red","black") ,legend=c("Simulated","Analytical"),lty = c(1,1))
```



The plot above should be in 3D $(\sigma^2, \mu, p(\sigma^2, \mu|y))$ but still don't know how to do that in R.

It also happens that for this particular problem there is a close form solution for the marginal of μ. (remember that $p(\mu|y) = \int p(\mu|\sigma^2, y)p(\sigma^2|y)d\sigma^2$):

$$\mu|\sigma^2, y \sim t_{\nu_n}(\mu|\mu_n, \sigma_n^2/\kappa_n)$$

I'm still confused with the following why is that to simulate the marginal of μ $(\mu|\sigma^2, y)$ I just need to draw from the conditional posterior of the mean (equation \ref{post.mu}). More generally, my confusion is the following: why is that to simulate $h(x) = \int g(x)dF(x)$ I just need to draw x^s from $dF(x)$ and evaluate $g(x^s)$. I know from MC simulation that by the LLN $E_{dF}[g(x)] = \int g(x)dF(x) \approx \sum_s g(x^s)/S$, but I'm having trouble seen why is that this applies to all points in the distribution (and not just the mean).

My own explanation so far: as MC can be used to approximate the mean using the LLN, the same technique can be used to approximate the whole distribution using CLT.

Simulating numbers from a Dirichlet distribution

```
draws       = 1000
#This are the parameters of the gamma used in generating the final Dirichlet.
gamma.scale = c(728,538,138)
aux1        = sapply(gamma.scale,function(x) rgamma(100,x))
aux1        = aux1 / apply(aux1,1,sum)
```

---

3 - Replicating Simulation Example 3.7 from BDA3 (pg 74+)

We are interested in modeling the dose-response of a certain drug (x_i) over the number of dead's (y_i) in a group of trial animals (n_i).We have 4 observations. Defining θ_i as the true death rate for each dosage, we can model this phenomena using a binomial distribution ($y_i \sim Bin(n_i, \theta_i)$). To model the dose-response relationship we start by looking at $\theta_i = \alpha + \beta x_i$ but we realize that this model predicts values out of range (θ, as a probability has to be between 0 and 1). Hence we apply the logit ($log(\theta_i/(1-\theta_i))$) transformation. This implies the following likelihood.

$$p(y_i|\alpha, \beta, n_i, x_i) \propto [logit^{-1}(\alpha + \beta x_i)]^{y_i}[1 - logit^{-1}(\alpha + \beta x_i)]^{n_i - y_i}$$

Question for Susan: why is that in the likelihood above we do not multiply by the Jacobian of the transformation? Using a non-informative prior ($p(\alpha, \beta) \propto 1$) we get that the conditional posterior has the form:

$$p(\alpha, \beta|y, n, x) \propto \prod_{i=1}^{k} p(y_i|\alpha, \beta, n_i, x_i)$$

First we compute, as a rough approximation of the parameters, the ML estimates.

```
# Data
x       = c(-0.86, -0.30, -0.05,  0.73)
y       = c(0,1,3,5)
n       = rep(5,4)

# ML estimation
mlfit <- glm( cbind(y, n-y) ~ x, family = binomial)
mlfit$coefficients = round(mlfit$coefficients*1000)/1000
res1 = paste(paste("(",mlfit$coefficients[1], sep=""), paste(mlfit$coefficients[2],")", sep=""), sep=",
```

With the estimates $(\hat{\alpha}, \hat{\beta}) = (0.847, 7.749)$.

Now we will simulated values of α and β from the posterior distribution. The approach is as follows.

- (i) Build a grid for all possible values of α and β. Although this can be the whole real line, we use the ML estimates and some trial and error to restrict the space to $\alpha \in [-5, 10]$ and $\beta \in [-10, 40]$.

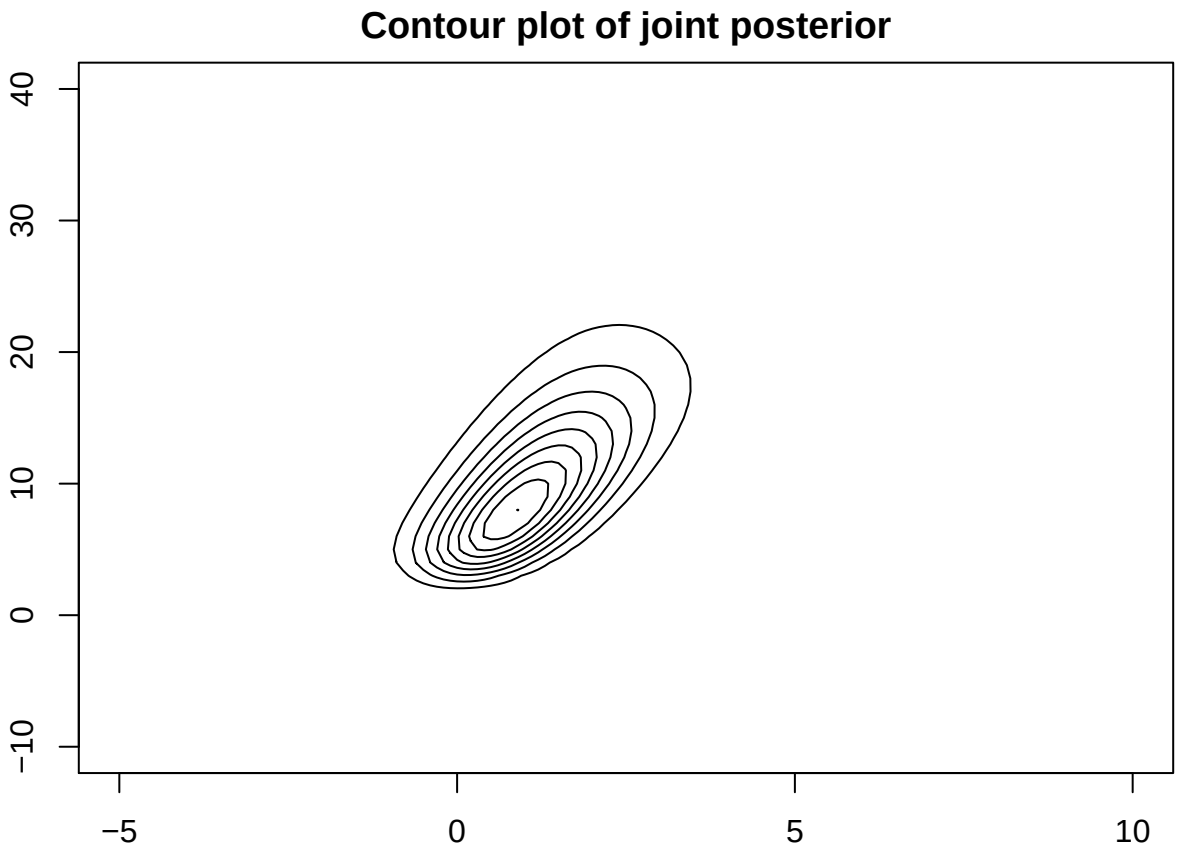
```
alpha     = seq(-5,10,.1)
beta      = seq(-10,40,1)
```

- (ii) Compute the posterior density (equation \ref{post.1}) over the whole grid and normalize it to 1.

```
post.a    = function(a,b,x,y,n) {
  post    = 1
  for (i in 1:length(y)) {
    post  = post * ( ((inv.logit(a+b*x[i]))^y[i])*((1-inv.logit(a+b*x[i]))^(n[i]-y[i])) )
  }
  post
}

post.dens = outer(alpha,beta,function(x1,x2) post.a(x1,x2,x,y,n))
post.dens = post.dens/sum(post.dens)
```

- (iii) Inspect the density using a contour plot (here we are looking for some indication that we are covering all the possible domain)

```
contours <- seq(min(post.dens), max(post.dens) , length=10)
par(mar = rep(2, 4))
contour(alpha, beta, post.dens,levels=contours, xlab=expression(alpha), ylab=expression(beta), xlim=c(m
```



- (iv) Compute the marginal posterior of α by summing over all β for each value of α. Notice so far that we had only compute the probability distribution of α and β. Is not until this point where we would be able to do things like compute the mean, median and CI's of these parameters.

```
samps       = 1000
alpha.dens  = apply(post.dens ,1, sum)
```

- (v) For $s = 1 \ldots 1000$:

  a. Draw α^s from its marginal posterior.

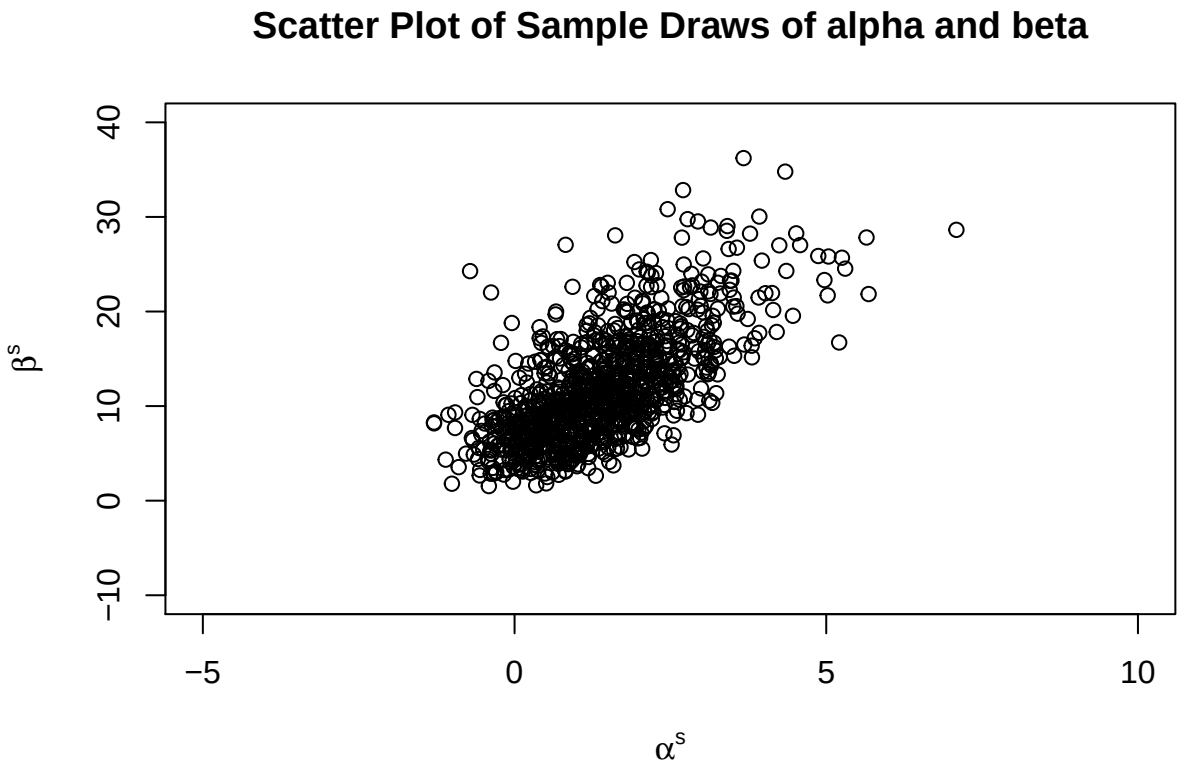
```
s.alpha     = sample(alpha,samps, replace=TRUE, prob = alpha.dens)
```

  b. Draw β^s from $p(\beta|\alpha, y)$ for each value of α^s

```
#Select the colum of the joint density corresponding to a specific value of alpha (p(beta|alpha))
cond.beta   = function(x) {
  post.dens[which(alpha == s.alpha[x]),]
}
#Sample a value of beta according the the conditional probatility above
s.beta      = sapply(1:samps,function(x) sample(beta,1,replace=TRUE,prob=cond.beta(x)))
```

c. For each sample values of α and β add a uniform random jitter centered at zero with with equal to the grid spacing. This will make the simulation draws more continuous. Plot the sampled values.

```
s.beta      = s.beta + runif(length(s.beta),-.5,.5)
s.alpha     = s.alpha + runif(length(s.alpha),-.05,.05)
plot(s.alpha, s.beta, xlab=expression(alpha^s), ylab=expression(beta^s), xlim=c(min(alpha),max(alpha))
```



The final result are two vectors $(\{\alpha^s\}), (\{\beta^s\})$, that represent probability distribution of each parameter.

4 - Description with my own words, of the fully Bayesian analysis of conjugate hierarchical models described in section 5.3 of BDA (pg 108 - 113)

The overall goal of hierarchical models is **to describe a statistical (uncertain) phenomena where are multiple parameters involved and exists a dependence structure**.

The analysis described here builds on what we just did in the example above. For this purpose we enumerate the steps followed, now in a more general notation.

1. Compute the joint posterior probability distribution of all the parameters conditional on the data and a prior distribution of those parameters (in the example above we use a non-informative prior).

2. Compute the marginal posterior of one parameter ($p(\theta_2|y)$) by summing the joint posterior over all the possible values of the other parameter (θ_1).
3. Compute the marginal of the other parameter ($p(\theta_1|y)$) using the following identity:

$$p(\theta_1|y) = \int p(\theta_1|\theta_2, y)p(\theta_2|y)d\theta_2 \approx p(\theta_1|\theta_2^s, y)$$

Where θ_2^s is a random draw using $p(\theta_2|y)$ (the justification for this last step is in section 2 of this document).

Now we will distinguish between two sets of parameters. The original parameters of interest, still represented by θ. And the parameters of the prior distribution, or *hyperparameters* represented by ϕ. This two sets of parameters fully describe the probabilistic process and have a joint distribution function:

$$p(\phi, \theta) = p(\phi)p(\theta|\phi)$$

And a joint posterior distribution:

$$\begin{aligned} p(\phi, \theta|y) &\propto p(\theta, \phi)p(y|\theta, \phi) \\ &= p(\phi)p(\theta|\phi)p(y|\theta) \end{aligned}$$

Where in the last line we used the assumption that $p(y|\theta, \phi)$ depends on ϕ only through θ. We call $p(\phi)$ the *hyperprior distribution*, $p(\theta|\phi)$ the population distribution, and the usual suspect $p(y|\theta)$ is the likelihood, or sampling, distribution.

As always our goal is to make inference about θ (and maybe ϕ also?). As in the exercise before, our end result will be a set of matrices $(\{\phi^s\}), (\{\theta^s\})$ with values for the parameters that follow the posterior joint distribution. To get this result we take the following steps:

1. Compute the conditional joint posterior $p(\phi, \theta|y)$. Choosing the right hyper-prior distribution is a whole topic on itself and will be addressed later.
2. Compute the marginal posterior of θ, conditional on ϕ, $p(\theta|\phi, y)$. This conditional posterior of θ can be obtain analytically in conjugate models for a given value of ϕ, it is simply the posterior of the non-hierarchical Bayesian case.

3. Compute the posterior marginal of ϕ. When step 2 has a closed form solution we can compute the marginal of ϕ as:

$$p(\phi|y) = \frac{p(\theta, \phi|y)}{p(\theta|\phi, y)}$$

In the absence of a closed form solution for step 2. We can compute the marginal by integrating the joint over all the possible values of θ.
4. Draw samples ϕ^s from $p(\phi|y)$. Notice that ϕ can have multiple components. If it has more than one element we follow the procedure described at the beginning of this section.
5. For each draw of ϕ^s, draw a sample of θ from $p(\theta|\phi^s, y)$. **This allow us to fully characterize the parameter of interest (our goal, remember?)**

Application

We have data from $j = 1 \dots J, J = 71$ experiments where, in each experiment n_j rats were exposed to a drug and y_j of them, presented tumors afterwards (think also something similar to unemployment rate in different states with n_j people in the active labor force and y_j unemployed). The results from the experiments are assumed to follow a binomial distribution ($y_j|\theta_j \sim Bin(n_j, \theta_j)$), and the parameters θ_j are assume to follow a beta prior distribution ($\theta_j|\alpha, \beta \sim Beta(\alpha, \beta)$), this last assumption is made to take advantage of conjugacy. With all this elements now we can follow the steps described above.

1. Compute the conditional joint posterior $p(\phi = (\alpha, \beta), \theta = \{\theta_j\}_{j=1}^J | y)$.

$$p(\theta, \alpha, \beta | y) \propto p(\alpha, \beta) p(\theta | \alpha, \beta) p(y | \theta, \alpha, \beta)$$
$$p(\theta, \alpha, \beta | y) \propto p(\alpha, \beta) \prod_{j=1}^{J} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_j^{\alpha-1} (1 - \theta_j)^{\beta-1} \prod_{j=1}^{J} \theta_j^{y_j} (1 - \theta_j)^{n_j - y_j}$$

2. Compute the marginal posterior of θ, conditional on α, β. For the beta-binomial case we have that given the hyper-parameters, each θ_j has a posterior distribution $Beta(\alpha + y_j, \beta + n_j - y_j)$. Assuming exchangeability:

$$p(\theta | \alpha, \beta, y) = \prod_{j=1}^{J} \frac{\Gamma(\alpha + \beta + n_j)}{\Gamma(\alpha + y_j)\Gamma(\beta + n_j - y_j)} \theta_j^{\alpha + y_j - 1} (1 - \theta_j)^{\beta + n_j - y_j - 1}$$

3. Compute the posterior marginal of (α, β). Given that we do have a closed form solution in step 2, we compute the ratio of (\ref{rat.joint.post}) and (\ref{rat.cond.post.theta}).

$$p(\alpha, \beta | y) \propto p(\alpha, \beta) \prod_{j=1}^{J} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha + y_j)\Gamma(\beta + n_j - y_j)}{\Gamma(\alpha + \beta + n_j)}$$

And here is the code for this function:

```
rat.marg.post.phi =  function(alpha, beta) {
  post    = 1
  for (i in 1:length(y)) {
    post  = post * ( ( ( gamma(alpha + beta) )/( gamma(alpha)*gamma(beta) ) ) * ( ( gamma(alpha + y[i] )
  }
  # The hyper prior is defined below
  rat.hyper.prior(alpha,beta) * post
}
```

Note: we assume a hyper-prior of the form $p(\alpha, \beta) \propto (\alpha + \beta)^{-5/2}$. So far I do not understand very well the whole discussion about proper and improper priors. **Need to review Ch10 of DeGroot's book**

4. Draw samples (α^s, β^s) from $p(\alpha, \beta | y)$. Before drawing the samples, the authors applied the following transformation to the parameter space: $(\alpha, \beta) \rightarrow (log(\frac{\alpha}{\beta}), log(\alpha + \beta))$. **I still don't know why they do this and why is that the transformation only applies to the hyper-prior only**. This step requires a important number of sub-steps:
   4.1. Compute the (prior) probability distribution of the transformed variables. Here is where we multiply by the Jacobian of the inverse transformation. The transformation method states that if $u \sim p_u(u)$ and there is a 1:1 function over u, $v = f(u)$, then $v \sim p_u(u)|J|$ where the J is the Jacobian of the function $f^{-1}(v)$ with respect to v. For this case we have $v_1 = f_1(\alpha, \beta) = log(\alpha/\beta)$ and $v_2 = f_2(\alpha, \beta) = log(\alpha + \beta)$, with inverse $f_1^{-1}(v_1, v_2) = \exp(v_1 + v_2)/(1 + exp(v_1))$ and $f_2^{-1}(v_1, v_2) = exp(v_2)/(1 + exp(v_1))$. Hence the Jacobian is:

$$J = \begin{bmatrix} \frac{\partial f_1^{-1}}{\partial v_1} & \frac{\partial f_1^{-1}}{\partial v_2} \\ \frac{\partial f_2^{-1}}{\partial v_1} & \frac{\partial f_2^{-1}}{\partial v_2} \end{bmatrix} = -\alpha\beta$$

Hence, using the transformation method we have that $p(log(\frac{\alpha}{\beta}), log(\alpha + \beta)) \propto \alpha\beta(\alpha + \beta)^{-5/2}$. And here is the code for this function:

```
y         =  c(0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
               0,  0,  0,  1,  1,  1,  1,  1,  1,  1,  1,
               3,  2,  2,  2,  2,  2,  2,  2,  2,  2,  1,
               5,  2,  5,  2,  7,  7,  3,  3,  2,  9, 10,
               4,  4,  4,  4,  4,  4,  4, 10,  4,  4,  4,
               5, 11, 12,  5,  5,  6,  5,  6,  6,  6,  6,
              16, 15, 15,  9,  4)
n         = c(20, 20, 20, 20, 20, 20, 20, 19, 19, 19, 19,
              18, 18, 17, 20, 20, 20, 20, 19, 19, 18, 18,
              27, 25, 24, 23, 20, 20, 20, 20, 20, 20, 10,
              49, 19, 46, 17, 49, 47, 20, 20, 13, 48, 50,
              20, 20, 20, 20, 20, 20, 20, 48, 19, 19, 19,
              22, 46, 49, 20, 20, 23, 19, 22, 20, 20, 20,
              52, 46, 47, 24, 14)

rat.hyper.prior   = function(alpha,beta) {
  alpha*beta*(alpha + beta)^(-5/2)
}
```

4.2. Identify the relevant domain for the transformed variables and its counterpart with the original variables. To do so we start computing the parameters α and β that would match the sample mean and standard deviation of all 70 experiments with a beta prior distribution:

$$\hat{\mu} = 0.1381 = \frac{\hat{\alpha_0}}{\hat{\alpha_0} + \hat{\beta_0}}$$

$$\hat{\sigma^2} = 0.0109 = \frac{\hat{\alpha_0}\hat{\beta_0}}{(\hat{\alpha_0} + \hat{\beta_0})^2(\hat{\alpha_0} + \hat{\beta_0} + 1)}$$

Solving for $(\hat{\alpha_0}, \hat{\beta_0})$:

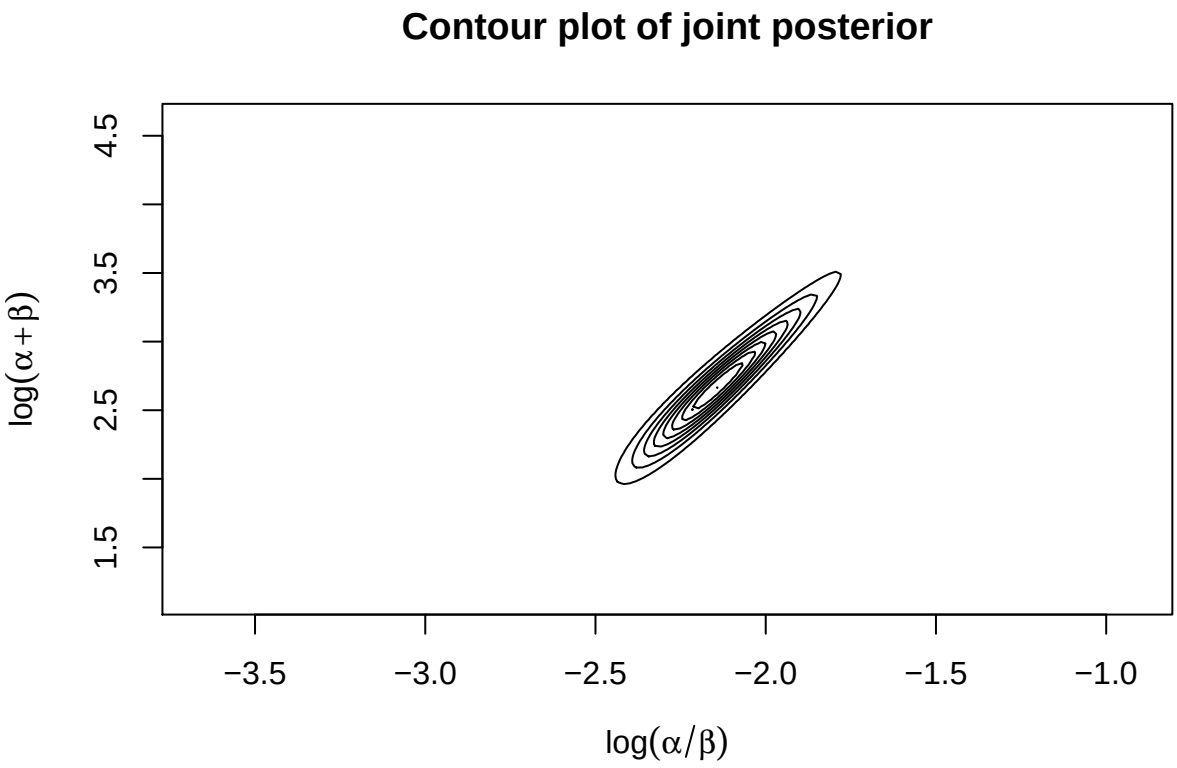
```
dslnex          <- function(x) {
    z           <- numeric(2)
    z[1]        <- x[1]/(x[1]+x[2]) - mean(y/n)
    z[2]        <- x[1]*x[2]/(((x[1]+x[2])^2)*(x[1]+x[2]+1)) - sd(y/n)^2
    z
}

sol1            <- nleqslv(c(1,1), dslnex)
res1            <- paste("(",round(sol1$x[1],1), ",", round(sol1$x[2],1), ")",sep="")
```

We get: $(\hat{\alpha_0}, \hat{\beta_0}) = (1.4, 8.6)$.

We center the grid (approximately) around that initial estimate and expand the grid to cover up to a factor of 4 of each parameter. The result is plotted in the following figure:

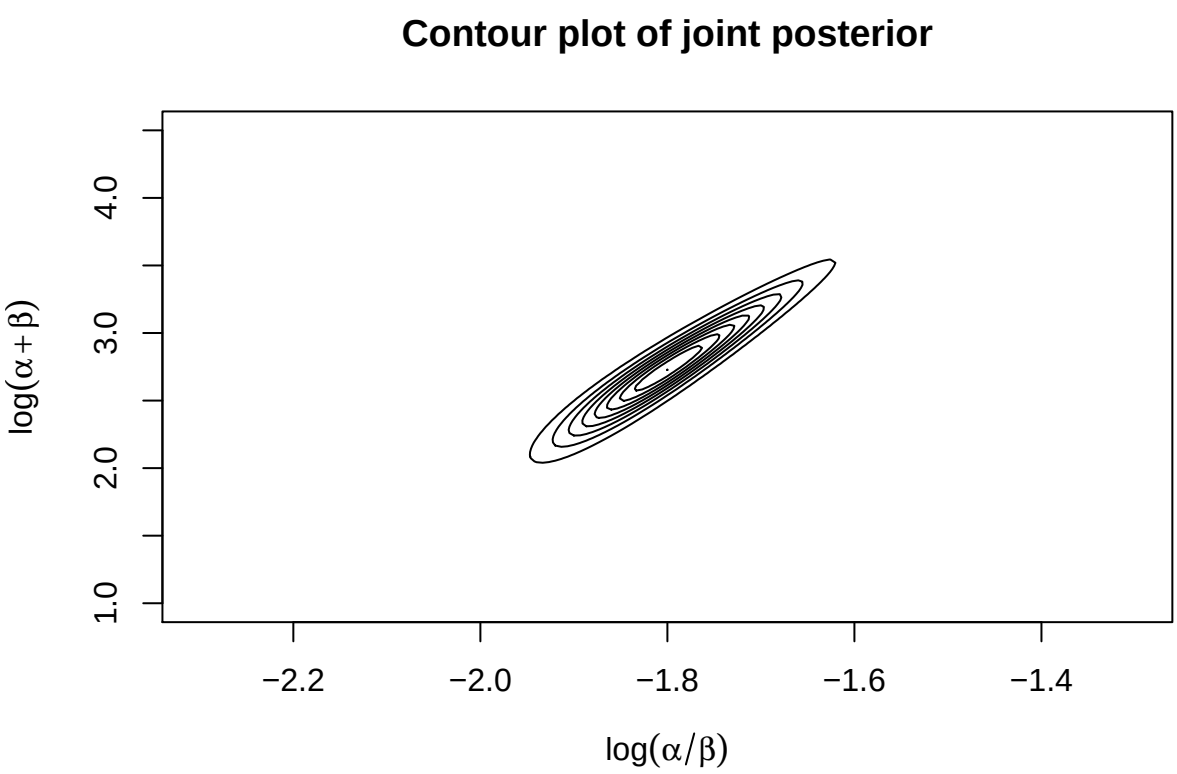
```
v1        = seq(log(sol1$x[1]/sol1$x[2])*2,log(sol1$x[1]/sol1$x[2])/2,length.out =151)
v2        = seq(log(sol1$x[1]+sol1$x[2])/2,log(sol1$x[1]+sol1$x[2])*2,length.out =151)
beta      = exp(v2)/(exp(v1)+1)
alpha     = exp(v2+v1)/(exp(v1)+1)

post.dens = outer(alpha,beta,function(x1,x2) log(rat.marg.post.phi(x1, x2)) )
post.dens = exp(post.dens - max(post.dens))
```

```
post.dens = post.dens/sum(post.dens)

contours <- seq(min(post.dens), max(post.dens) , length=10)
contour(v1, v2, post.dens,levels=contours, xlab=expression(log(alpha/beta)), ylab=expression(log(alpha+b
```



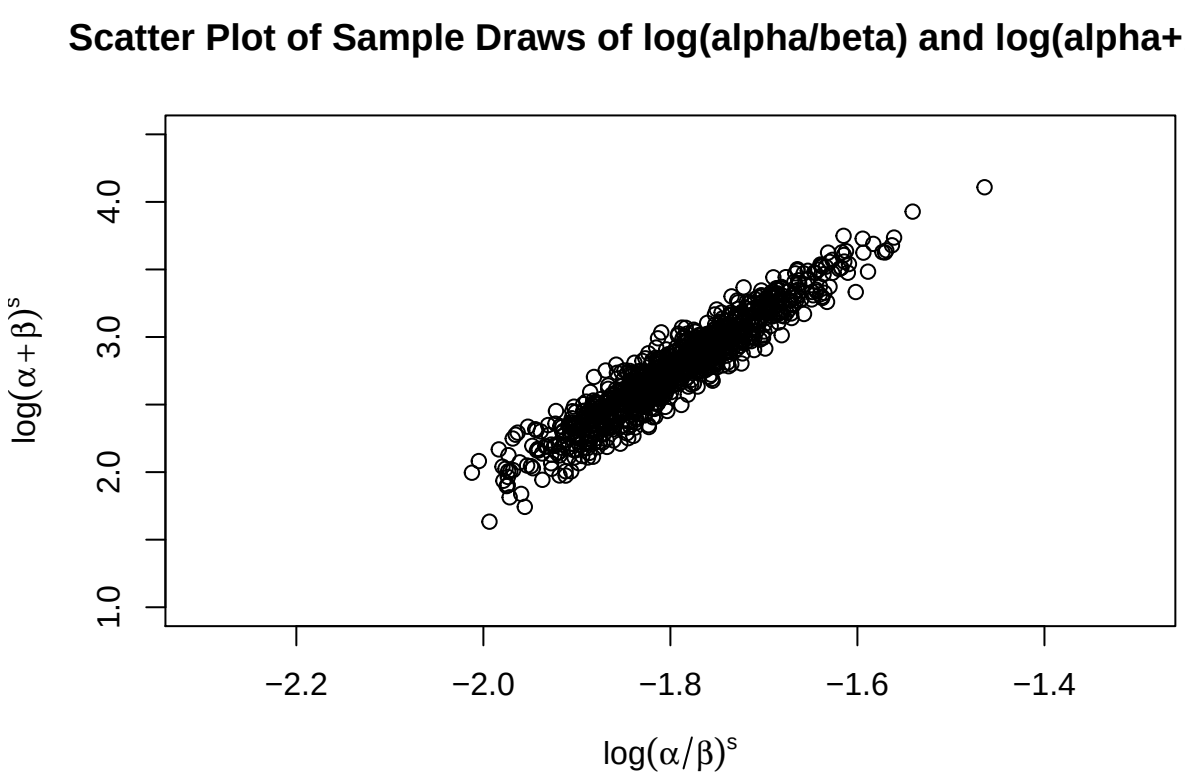
4.3. Recalculate the range of the grid such that includes all the density

```
v1        = seq(-2.3,-1.3,length.out =151)
# The booksets the range of "v2" up to 5, but my gamma function calculator gives me trouble after 4.8
v2        = seq(1 , 4.5,length.out =151)
beta      = exp(v2)/(exp(v1)+1)
alpha     = exp(v2+v1)/(exp(v1)+1)

post.dens = outer(alpha,beta,function(x1,x2) log(rat.marg.post.phi(x1, x2)) )
post.dens = exp(post.dens - max(post.dens))
post.dens = post.dens/sum(post.dens)

contours <- seq(min(post.dens), max(post.dens) , length=10)
contour(v1, v2, post.dens,levels=contours, xlab=expression(log(alpha/beta)), ylab=expression(log(alpha+b
```



4.4. Draw samples (α^s, β^s) from $p(\alpha, \beta|y)$ (finally!). Here we repeat the procedure used in section 3.(v) of this document.

```
samps       = 1000
v1.dens     = apply(post.dens ,1, sum)
s.v1        = sample(v1,samps, replace=TRUE, prob = v1.dens)

#Select the colum of the joint density corresponding to a specific value of v1 (p(v2|v1))
cond.v2     = function(x) {
  post.dens[which(v1 == s.v1[x]),]
}
#Sample a value of v2 according the the conditional probatility above
s.v2        = sapply(1:samps,function(x) sample(v2,1,replace=TRUE,prob=cond.v2(x)))

#Add a uniform random jitter centered at zero with with equal to the grid spacing. This will make the s
grid.v1     = v1[2]-v1[1]
grid.v2     = v2[2]-v2[1]
s.v2        = s.v2 + runif(length(s.v2),-grid.v2/2,grid.v2/2)
s.v1        = s.v1 + runif(length(s.v1),-grid.v1/2,grid.v1/2)
plot(s.v1, s.v2, xlab=expression(log(alpha/beta)^s), ylab=expression(log(alpha+beta)^s), xlim=c(min(v1)
```



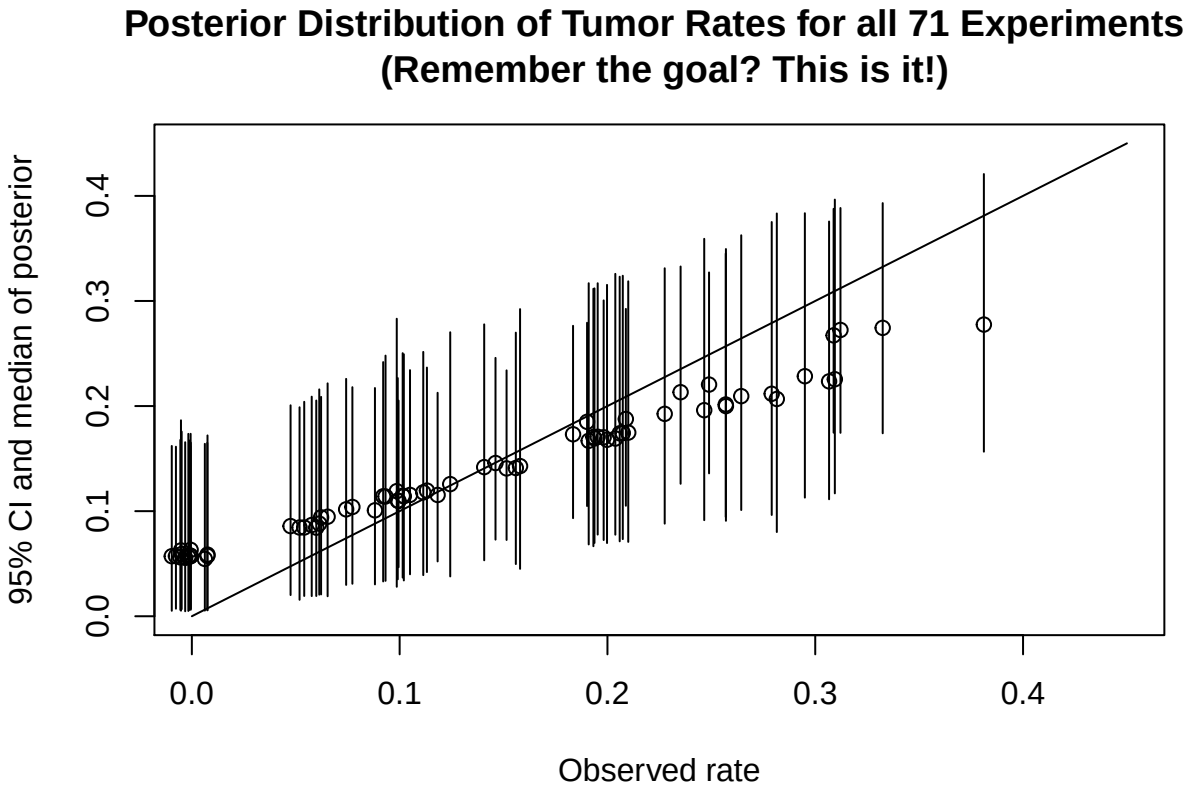
Note:this two figures do not match exactly their counterparts in the textbook (5.3a and 5.3b in BDA3), but so far I have not been able to detect my mistake.

4.5 By applying the inverse of the transformation we recover the marginal distribution of the original hyper-parameters.

```
s.beta      = exp(s.v2)/(exp(s.v1)+1)
s.alpha     = exp(s.v2+s.v1)/(exp(s.v1)+1)
```

5. For each draw of ϕ^s, draw a sample of θ from $p(\theta|\phi^s, y)$

```
s.beta      = exp(s.v2)/(exp(s.v1)+1)
s.alpha     = exp(s.v2+s.v1)/(exp(s.v1)+1)
theta.dist  = sapply(1:71, function(x) rbeta(1000,s.alpha+y[x], s.beta + n[x] - y[x]))
theta.dist  = apply(theta.dist,2,sort)
plot(0:450/1000,0:450/1000, type="l", xlab="Observed rate",ylab="95% CI and median of posterior")
jitter.x    = y/n + runif(length(y),-0.01,0.01)
points(jitter.x,theta.dist[500,])
segments(jitter.x,theta.dist[25,], jitter.x,theta.dist[975,] )
title(main="Posterior Distribution of Tumor Rates for all 71 Experiments \n (Remember the goal? This is
```



5 - Replicating section 5.5: Experiments in eight schools (normal model) [Without Stan]

```
#Data:
school.id       <- LETTERS[1:8]
effect          <- c(28,8,-3,7,-1,1,18,12)
se.effect       <- c(15,10,16,11,9,11,10,18)

pool.est        <- sum(effect*se.effect^-2)/sum(se.effect^-2)
pool.var        <- sum(se.effect^-2)^-1
pool.ci         <- c(-1.96,1.96)*pool.var^.5 + pool.est
```

The pooled estimated effect and variance are 7.69 and 16.58, with a 95% CI of [-0.3, 15.67].

Posterior simulation under ther hierarchical model
Using the identity:

$$p(\theta, \mu, \tau|y) = p(\tau|y)p(\mu|\tau, y)p(\theta|\mu, \tau, y)$$

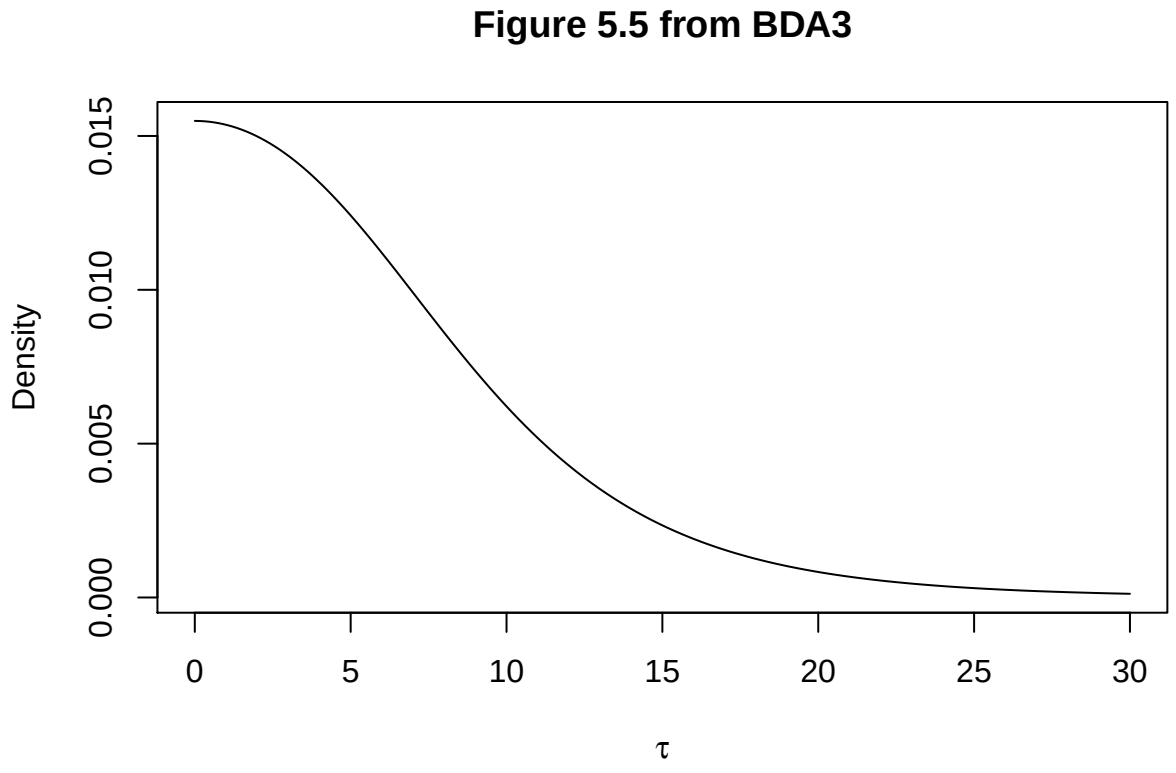
And the results from BDA in equation 5.17, 5.20 and 5.21 we code the joint posterior:

```
# Eqn 5.17 of BDA3
post.theta.j    <- function(mu,tau,j) (effect[j]/(se.effect[j]^2) + mu/(tau^2)) / (1/(se.effect[j]^2) +
post.v.theta.j  <- function(tau,j) 1/(1/(se.effect[j]^2) + 1/(tau^2))
# Eqn 5.20 of BDA3
post.mu.hat     <- function(tau) sum(effect*1/(se.effect^2 +tau^2))/sum(1/(se.effect^2 +tau^2))
```

```
post.v.mu       <- function(tau) (sum(1/(se.effect^2 +tau^2)))^-1

# Eqn 5.21 of BDA3
marginal.tau     <- function(tau) {
  hyper.prior(tau)*(post.v.mu(tau)^.5)*prod(((se.effect^2 + tau^2)^(-.5)) *
                  exp(-((effect - post.mu.hat(tau))^2)/(2*(se.effect^2 + tau^2))))
}
```

Define a hyper-prior and draw 200 samples from each distribution (for all 8 schools).

```
samps           <- 200

hyper.prior     <-  function(tau) 1
tau.grid        <-  seq(0.001,30, length=samps)
pdf.tau         <-  sapply(tau.grid,function(x) marginal.tau(x))
pdf.tau         <-  pdf.tau/sum(pdf.tau)

plot(tau.grid,pdf.tau, type="l", main="Figure 5.5 from BDA3", xlab=expression(tau), ylab="Density")
```



The sampling method in BDA3 suggest to apply the inverse method from the posterior of τ. I don't do this for two reasons: (i) I'm not sure the posterior has a closed for solution for its inverse, and (ii) given that I already have the density, I can directly draw from that distribution sampling using the `sample` command (which leads me to think that this command applies the inverse method, but **need to check with Susan**).

```
# Sampling
s.tau           <- sample(tau.grid,samps,prob=pdf.tau, replace=TRUE)
s.mu            <- sapply(s.tau,function(x) rnorm(1,post.mu.hat(x),(post.v.mu(x))^0.5))
s.theta         <- NULL
for (j in 1:length(school.id)) {
  s.theta[[j]]         <- sapply(1:samps,
                           function(x)
                           rnorm(1,
                                 post.theta.j(s.mu[x],s.tau[x],j),
                                 (post.v.theta.j(s.tau[x],j))^0.5
                                 ) )
  }
```

The following figures replicate the figures in pg 122 in BDA. Before doing the plots we need to 'average over μ'

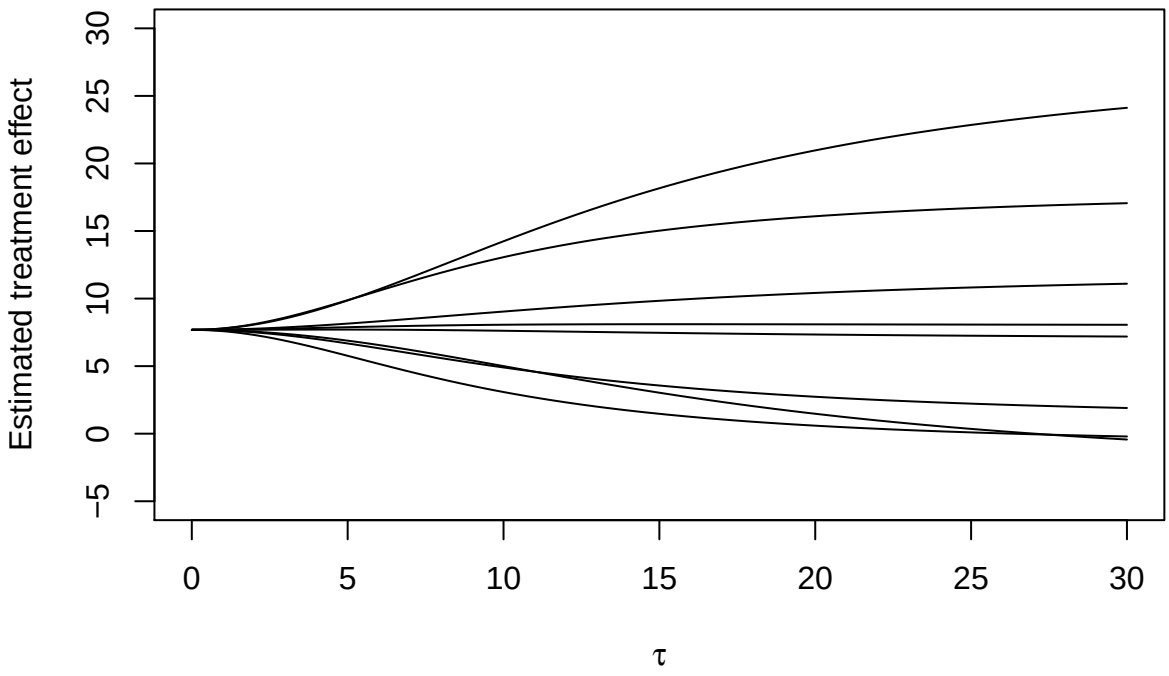
$$\begin{aligned} E(\theta_j|\tau,y) &= E_\mu\left[E(\theta_j|\tau,y,\mu)|\tau,y\right] \\ &= E_\mu\left[\frac{\frac{1}{\sigma_j^2}y_j+\frac{1}{\tau^2}\mu}{\frac{1}{\sigma_j^2}+\frac{1}{\tau^2}}|\tau,y\right] = \frac{\frac{1}{\sigma_j^2}y_j+\frac{1}{\tau^2}\hat{\mu}}{\frac{1}{\sigma_j^2}+\frac{1}{\tau^2}} \end{aligned}$$

$$\begin{aligned} Var(\theta_j|\tau,y) &= E_\mu\left[Var(\theta_j|\tau,y,\mu)|\tau,y\right] + Var_\mu\left[E(\theta_j|\tau,y,\mu)|\tau,y\right] \\ &= \frac{1}{\frac{1}{\sigma_j^2}+\frac{1}{\tau^2}} + V_\mu\left(\frac{\frac{1}{\tau^2}}{\frac{1}{\sigma_j^2}+\frac{1}{\tau^2}}\right) \end{aligned}$$

Where V_μ and $\hat{\mu}$ correspond to the expressions defined in Eq 5.20 of BDA3. Below is the code and plot of both equations.

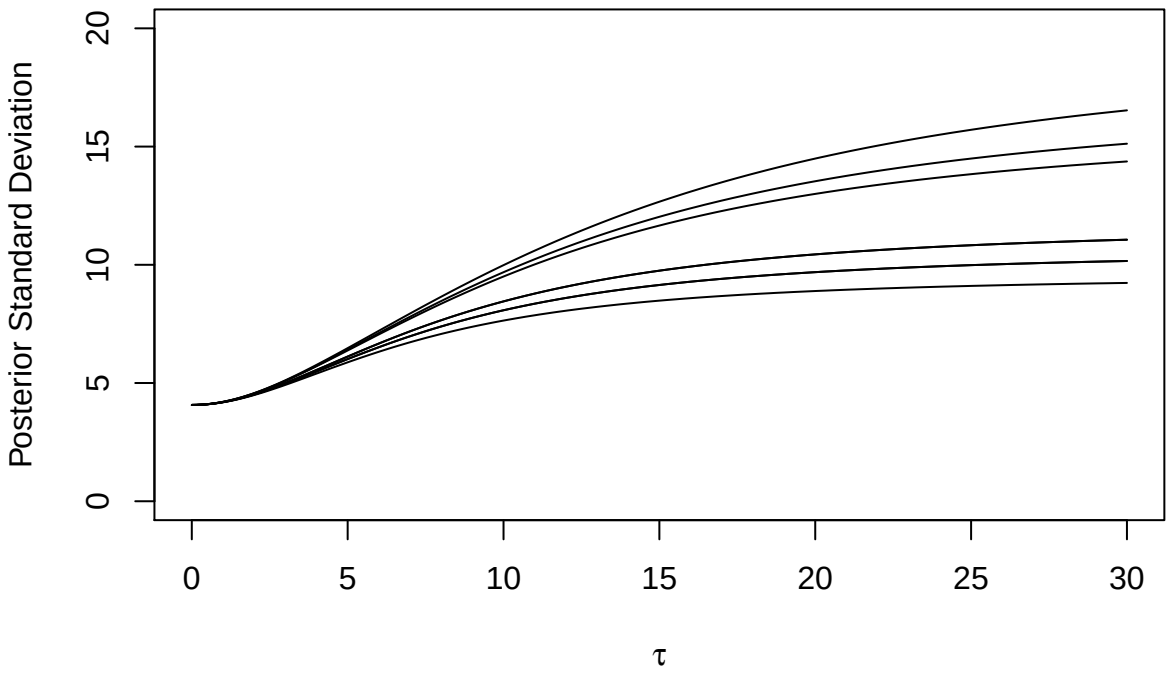
```
post.theta.j.no.mu     <- function(tau,j) post.theta.j(post.mu.hat(tau),tau,j)
post.se.theta.j.no.mu  <- function(tau,j) sqrt( (post.v.theta.j(tau,j)) * (1+post.v.mu(tau)*tau^(-2)) )

plot( tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,1)), type="l", ylim=c(-5,30), xlab="",
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,2)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,3)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,4)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,5)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,6)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,7)))
lines(tau.grid,sapply(tau.grid, function(x) post.theta.j.no.mu(x,8)))
title(main="Figure 5.6 from BDA3", xlab=expression(tau), ylab="Estimated treatment effect")
```

Figure 5.6 from BDA3




```
plot(tau.grid,sapply(tau.grid, function(x)  post.se.theta.j.no.mu(x,1)), type="l", ylim=c(0,20), xlab="
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,2)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,3)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,4)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,5)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,6)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,7)))
lines(tau.grid,sapply(tau.grid, function(x) post.se.theta.j.no.mu(x,8)))
title(main="Figure 5.7 from BDA3", xlab=expression(tau), ylab="Posterior Standard Deviation")
```

Figure 5.7 from BDA3




```
s.theta          <- matrix(unlist(s.theta), ncol = 8, byrow = FALSE)
s.theta.sort     <- apply(s.theta, 2, sort)
p                <- t( apply(s.theta.sort, 2, function(x) quantile(x,c(.025,.25,.5, .75, .975),type=1)) )
p                <- round(p,3)
```

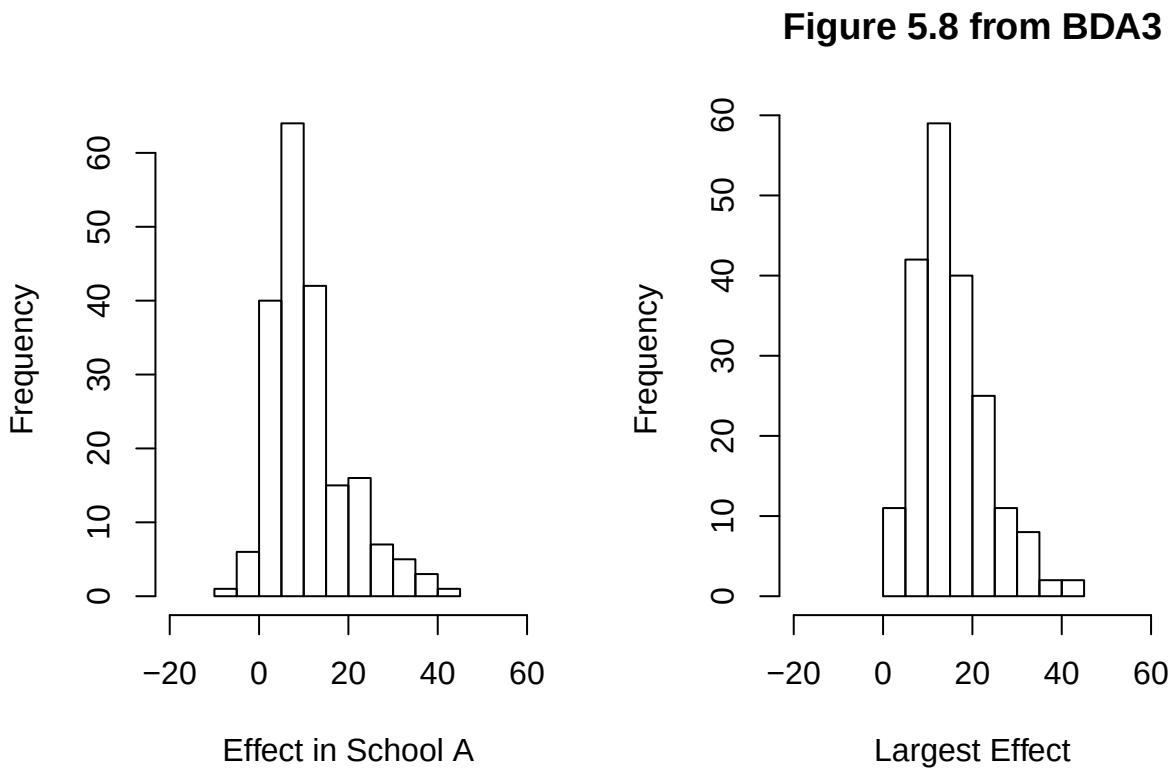
Table 5.3 from BDA3:

School	2.5%	25%	median	75%	97.5%
A	-0.959	5.126	8.799	14.701	31.639
B	-4.707	3.321	7.599	11.384	20.985
C	-15.041	2.191	6.602	10.44	17.38
D	-9.037	2.942	6.87	11.321	20.173
E	-8.086	2.217	6.21	9.562	18.036
F	-6.893	2.292	6.263	10.17	19.596
G	-1.023	5.646	9.295	14.107	29.198
H	-7.684	4.477	7.525	12.355	24.877

Here we reproduce figure 5.8 (with the same problems as above)

```
par(mfrow=c(1,2))
domain           <- c(-20,60)
hist(s.theta[,1], breaks=10, xlab="Effect in School A", main="", xlim=domain)
hist(apply(s.theta,1,max), breaks=10, xlab="Largest Effect", main="", xlim=domain)
```

```
title(main="Figure 5.8 from BDA3")
```



This last figure ("largest effect") is a good example of one the main advantage of a fully Bayesian hierarchical model: once we have correctly simulated the posterior, we can test all kinds of complicated hypothesis.

6 - Replicating section 5.5: Experiments in eight schools (normal model) [With Stan]

Appendix C in BDA3

7 - Replicating example of section 11.2: Metropolis sampling from bivariate normal [Witout Stan]

Here we follow the steps in p278 to simulate draws from a bivariate normal $N(0, I_2)$.

1. Define a starting point θ^0, for which $p(\theta^0|y) > 0$:

```
set.seed(142857)
library(mvtnorm)
theta.0         <- c(-2.5, 2.5)
```

Our starting point is $(-2.5, 2.5)$, the upper left most square dot from figure 11.1a.

2. For $t = 1, 2, \ldots$:

(a) Sample a proposal θ^* at time t from $J_t(\theta^*|\theta^{t-1})$. For this example the *jumping distribution* is $N(\theta^*|\theta^{t-1}, 0.2^2 I_2)$:

```
theta.star      <- function(theta.t_1) {
  n.p         <- length(theta.t_1)
  # I would like to learn a way to draw from a multivariate normal without a black-box function.
  rmvnorm(1, mean = theta.t_1, sigma = 0.2^2 * diag(n.p))
}
```

(b) Calculate the ratio of densities,

$$r = \frac{p(\theta^*|y)}{p(\theta^{t-1}|y)}$$

```
r.dens          <- function(theta.t_1, theta.star) {
  n.p         <- length(theta.t_1)
  dmvnorm(theta.star, mean = rep(0, n.p), sigma = diag(n.p)) /
  dmvnorm(theta.t_1 , mean = rep(0, n.p), sigma = diag(n.p))
  }
```

(c) Set

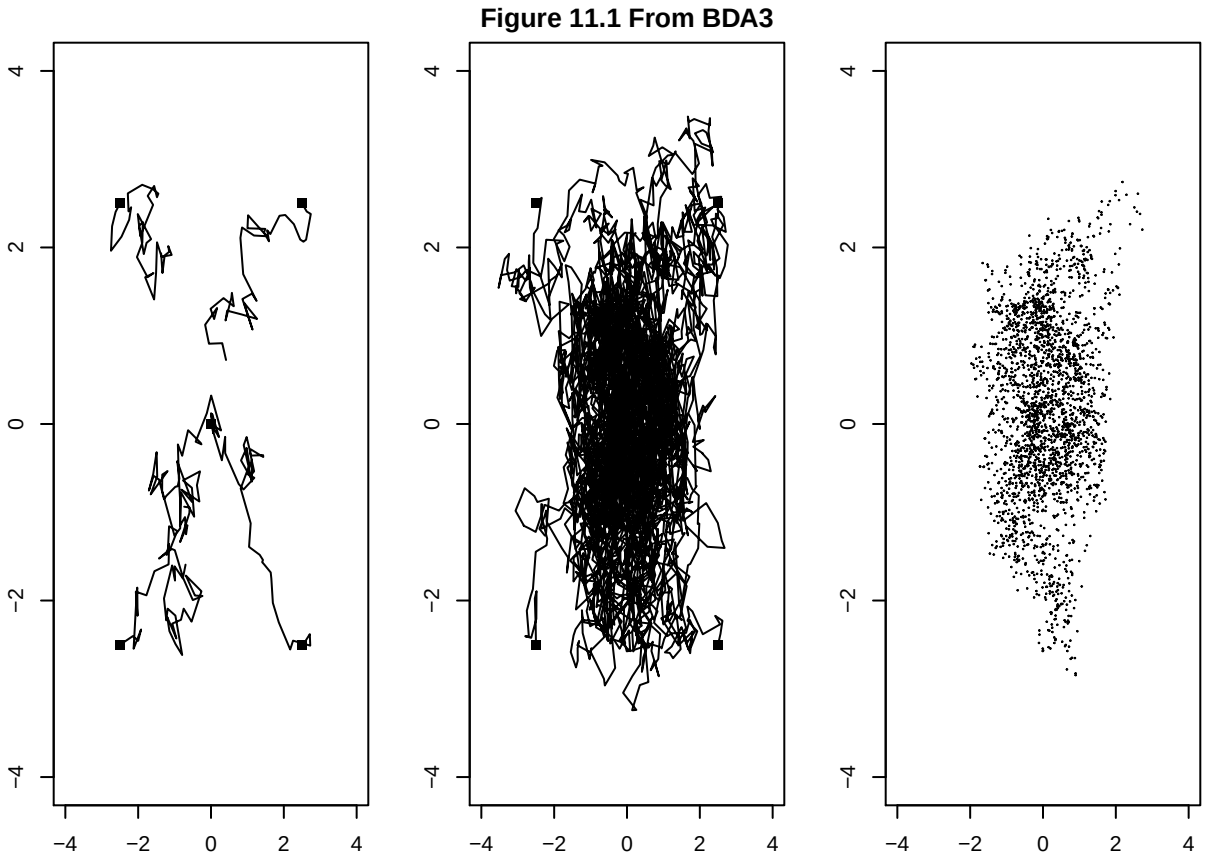
$$\theta^t = \begin{cases} \theta^* & \text{with probability } min(r, 1) \\ \theta^{t-1} & \text{otherwise.} \end{cases}$$

```
theta.t         <- function(theta.t_1, theta.star, r) {
  prob.aux      <- runif(1)
  if (prob.aux <= min(c(r,1)))
    theta.star
  else
    theta.t_1
}
```

(d) Execute:

```
theta           <- function(sims,theta.0) {
  s.theta       <- matrix(NA,sims,2)
  s.theta[1,]   <- t(as.matrix(theta.0))

  for (t in 2:sims) {
    s.theta.star<- theta.star(s.theta[t-1,])
    r           <- r.dens(s.theta[t-1,], s.theta.star)
    s.theta[t,] <- theta.t(s.theta[t-1,], s.theta.star, r)
  }
return(s.theta)
}
par(mfrow=c(1,3))
par(mar = rep(2,4))

# 11.1a
s.theta.1       <- theta(50,theta.0)
s.theta.2       <- theta(50,c(2.5,2.5))
s.theta.3       <- theta(50,c(2.5,-2.5))
```

```r
s.theta.4       <- theta(50,c(-2.5,-2.5))
s.theta.5       <- theta(50,c(0,0))
plot(s.theta.1, xlim=c(-4,4), ylim=c(-4,4), type="l", sub = "50 simulations")
points(theta.0[1], theta.0[2], pch=15)
lines(s.theta.2)
points(2.5, 2.5, pch=15)
lines(s.theta.3)
points(2.5, -2.5, pch=15)
lines(s.theta.4)
points(-2.5, -2.5, pch=15)
lines(s.theta.5)
points(0, 0, pch=15)

# 11.1b
s.theta.1       <- theta(1000, theta.0)
s.theta.2       <- theta(1000, c(2.5,2.5))
s.theta.3       <- theta(1000, c(2.5,-2.5))
s.theta.4       <- theta(1000, c(-2.5,-2.5))
s.theta.5       <- theta(1000, c(0,0))
plot(s.theta.1, xlim=c(-4,4), ylim=c(-4,4), type="l",  sub = "1000 simulations", main="Figure 11.1 From
points(theta.0[1], theta.0[2], pch=15)
lines(s.theta.2)
points(2.5, 2.5, pch=15)
lines(s.theta.3)
points(2.5, -2.5, pch=15)
lines(s.theta.4)
points(-2.5, -2.5, pch=15)
lines(s.theta.5)
points(0, 0, pch=15)

s.theta.1       <- s.theta.1[501:1000,] + matrix(runif(1000)/20, nrow = 500, ncol = 2)
s.theta.2       <- s.theta.2[501:1000,] + matrix(runif(1000)/20, nrow = 500, ncol = 2)
s.theta.3       <- s.theta.3[501:1000,] + matrix(runif(1000)/20, nrow = 500, ncol = 2)
s.theta.4       <- s.theta.4[501:1000,] + matrix(runif(1000)/20, nrow = 500, ncol = 2)
s.theta.5       <- s.theta.5[501:1000,] + matrix(runif(1000)/20, nrow = 500, ncol = 2)

# 11.1c
plot(s.theta.1, xlim=c(-4,4), ylim=c(-4,4), type="p", pch = 20, cex =.1, sub = "Second half of 1000 simu
points(s.theta.2, pch = 20, cex =.1)
points(s.theta.3, pch = 20, cex =.1)
points(s.theta.4, pch = 20, cex =.1)
points(s.theta.5, pch = 20, cex =.1)
```



8 - Replicating example of section 11.6: Gibbs sampling for a hierarchical normal model (Coagulations experiment data) [Witout Stan]

```
#Data
id              <- rep(LETTERS[1:4],c(4,6,6,8))
y               <- c(62,60,63,59,63,67,71,64,65,66,68,66,71,67,68,68,56,62,60,61,63,64,63,59)
J               <- length(unique(id))
n               <- length(y)
```

The Model
Data $y_{ij}, i = 1, ..., n_j, j = 1, .., J$ are iid within each of J groups with $y_{ij} \sim N(\theta_j, \sigma^2)$. The prior distribution of θ_j is assume to be normal with hyperparameters μ, τ ($\theta_j \sim N(\mu, \tau^2)$. σ is assumed to be unkonwn (confusing!) and the hyperprior is assumed to be uniform over $(\mu, log(\sigma), \tau)$, which implies $p(\mu, log(\sigma), log(\tau)) \propto \tau$. The joint posterior denstity for all the parameters is:

$$p(\theta, \mu, log(\sigma), log(\tau)|y) \propto \tau \prod_{j=1}^{J} N(\theta_j|\mu, \tau^2) \prod_{j=1}^{J} \prod_{i=1}^{n_j} N(y_{ij}|\theta_j, \sigma^2)$$

Starting points
Select 10 θ_j randomly from the y_{ij} sample. And take μ to be the average starting values of θ_j.

```
set.seed(142857)
theta.0          <- sapply(1:4,function(x) sample(y[id==LETTERS[x]],10, replace=TRUE))
mu.0             <- apply(theta.0, 1,mean)
```

[Here I follow the exact reverse order that is proposed in the book, going from step 4 to 1 instead of 1 to 4. It is not clear to me how to do it otherwise].

4. Draw from conditional posterior of τ^2 using:

$$\hat{\tau}^2 = \frac{1}{J-1}\sum_{j=1}^{J}(\theta_j - \mu)^2$$

```
tau.hat.2        <- function(theta) {
  mu             <- mean(theta)
  tau.2          <- ( 1/(J-1) ) * sum((theta - mu)^2)
  return(tau.2)
}
```

And the fact that:

$$\tau^2|\theta, \mu, \sigma, y \sim Inv - \chi^2(J-1, \hat{\tau}^2)$$

We can draws samples for τ^2

```
s.tau.post       <- function(theta) {
  tau.2          <- tau.hat.2(theta)
  tau.cond       <- (J - 1) * (tau.2)/rchisq(1,J-1)
  return(tau.cond)
  }
```

3. Draw from conditional posterior of σ^2 Using:

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{j=1}^{J}\sum_{i=1}^{n_j}(y_{ij} - \theta_j)^2$$

```
f.sigma.hat.2    <- function(theta) {
  sigma.hat.2    <-  sapply(1:4, function(x) (y[id==LETTERS[x]] - theta[x])^2)
  sigma.hat.2    <-  (1/n) * sum(unlist(sigma.hat.2))
  return(sigma.hat.2)
}
```

And the fact that:

$$\sigma^2|\theta, \mu, \tau, y \sim Inv - \chi^2(n, \hat{\sigma}^2)$$

We can draws samples for σ^2

```
s.sigma.post       <-  function(theta) {
  sigma2.hat       <- f.sigma.hat.2(theta)
  sigma2.post       <-  (n) * (sigma2.hat)/rchisq(1,n)
  return(sigma2.post)
  }
```

2. Draw from conditional posterior of μ Using:

$$\hat{\mu} = \frac{1}{J}\sum_{j=1}^{J}\theta_j$$

```
mu.hat          <- function(theta) {
  mean(theta)
}
```

And the fact that:

$$\mu|\theta,\sigma,\tau,y \sim N(\hat{\mu},\tau^2/J)$$

We can draw values for μ

```
s.mu        <- function(theta,tau2) {
  mu.hat    <- mu.hat(theta)
  rnorm(1,mu.hat,sqrt(tau2/J))
}
```

1. Finally, we can draw values for θ Using the fact that:

$$\theta_j|\mu,\sigma,\tau,y \sim N(\hat{\theta_j},V_{\theta_j})$$

With:

$$\hat{\theta_j} = \frac{\frac{1}{\tau^2}\mu + \frac{n_j}{\sigma^2}\bar{yj}}{\frac{1}{\tau^2}+\frac{n_j}{\sigma^2}}$$

$$V_{\theta_j} = \frac{1}{\frac{1}{\tau^2}+\frac{n_j}{\sigma^2}}$$

```
theta.hat.j     <- function(j,mu,sigma2,tau2) {
  y.bar.j       <- mean(y[id==LETTERS[j]])
  n.j           <- length(y[id==LETTERS[j]])
  ( (1/tau2) * mu + (n.j/sigma2) * y.bar.j  ) / ( (1/tau2) + (n.j/sigma2)  )
}

V.theta.hat.j   <- function(j,mu,sigma2,tau2) {
  n.j           <- length(y[id==LETTERS[j]])
  ( 1 ) / ( (1/tau2) + (n.j/sigma2) )
}

s.theta         <- function(mu,sigma2,tau2) {
  theta         <- NULL
  for (j in 1:J) {
    t.hat       <- theta.hat.j(j,mu,sigma2,tau2)
    v.t.hat     <- V.theta.hat.j(j,mu,sigma2,tau2)
    theta[j]    <- rnorm(1,t.hat,sqrt(v.t.hat))
  }
  return(theta)
}
```

```
mcmc.gibbs      <- function(chain) {
  res1          <- as.list(NULL)
  sims          <- 200
  param         <- 7
  s.param       <- matrix(NA, ncol = param, nrow = sims )
  colnames(s.param)<-  c("theta1", "theta2", "theta3", "theta4", "mu", "sigma2", "tau2")
  s.param[1,1:4]<- theta.0[chain,]
  s.param[1,7]  <- s.tau.post(theta.0[chain,])
  s.param[1,6]  <- s.sigma.post(theta.0[chain,])
  s.param[1,5]  <- s.mu(theta.0[chain,],s.param[1,7])

  for (s in 2:sims) {
    s.param[s,1:4]<- s.theta(s.param[s-1,5],s.param[s-1,6],s.param[s-1,7])
    s.param[s,7]  <- s.tau.post(s.param[s,1:4])
    s.param[s,6]  <- s.sigma.post(s.param[s,1:4])
    s.param[s,5]  <- s.mu(s.param[s,1:4],s.param[s,7])
  }
return(s.param)
#Warm-up
}
s.param           <- lapply(1:10,function(x) mcmc.gibbs(x))
s.param.1         <- s.param[[1]][101:200, ]

#Transform the variance in to sd.
s.param.1[,6:7]   <- sqrt(s.param.1[,6:7] )

t(apply(s.param.1,2, function(x) quantile(x, c(.025,.25,.5,.75,.975))))
```

```
##          2.5%    25%    50%    75%  97.5%
## theta1 58.539 60.668 61.290 62.136 63.551
## theta2 64.097 65.187 65.966 66.521 67.694
## theta3 65.779 67.124 67.719 68.310 69.505
## theta4 59.654 60.640 61.285 61.767 62.598
## mu     58.590 62.809 63.768 65.404 69.947
## sigma2  1.851  2.169  2.395  2.694  3.328
## tau2    1.825  2.777  3.833  6.389 17.368
```

```
r.hat             <- function(arg1) {
  n               <- dim(arg1)[1]
  m               <- dim(arg1)[2]
  B               <- (n/(m-1)) * sum( ( apply(arg1,2,mean) - mean(arg1) )^2 )
  W               <- (1/m) * (1/(n-1)) * sum( (arg1 - apply(arg1,2,mean))^2 )
  sqrt( (n-1)/(n) + B/(W*n) )
}

 r.hat(sapply(1:10,function(x) s.param[[x]][,4] ))
```

```
## [1] 1.032
```

Simulate 200 draws for each of the 10 chains
Note: Problems with σ and τ, need to review.