A Bayesian Approach to Hierarchical Model Building

Albert E. Parker
Department of Mathematical Sciences
Montana State University

May 3, 2004

A writing project submitted in partial fulfillment
of the requirements for the degree

Master of Science in Statistics

APPROVAL

of a writing project submitted by

Albert E. Parker

This writing project has been read by the writing project director and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the Statistics Faculty.

May 10, 2004
Date

Jim Robison-Cox
Writing Project Director

1 Introduction

The objective of this paper is to demonstrate the Bayesian approach to statistical inference, which can be summed up in a word: posterior. This demonstration touches upon different Bayesian probability models, how these models can be estimated from data, and how to infer something about parameters of interest given a model and data. In section 2, I outline, in my opinion, what differentiates the Bayesian approach from the non-Bayesian's. In section 3, two simple Bayesian models are introduced to motivate the power and flexibility of hierarchical Bayesian models, which I cover in section 4. Section 5 is devoted to demonstrating methods which can be used to approximate the posterior. Bayesian Mixture models are considered in section 6. All of the data sets and analyzes performed in an attempt to elucidate the details of the methods were from Gelman et al [4]. However, the R code which I wrote to implement these analyzes is my own, and a hard copy of most of this code is in Appendix 8, which also contains a URL where all of the code can be found.

2 What's under the hood when performing Bayesian inference?

Bayesian statistical conclusions about k parameters of interest, $\theta \in \Re^k$, are made in terms of probability statements which are conditional on the observed value of $y \in \Re^n$, the data [3, 4]. This probability is called the *posterior density*, written $p(\theta|y)$. The posterior is found by using the definition of conditional probability,

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

Applying this definition once more yields *Bayes Rule*,

$$p(\theta|y) = \frac{p(\theta)p(y|\theta)}{p(y)}, \tag{1}$$

interpreted as the distribution of the parameters given the data. The *sampling distribution* or *likelihood* $p(y|\theta)$ is the probability model which one assumes yielded the data y actually observed. The *prior*, $p(\theta)$, is the density over the population of possible values for θ based on the modeler's knowledge of θ. The marginal density of y is $p(y) = \int p(y,\theta)d\theta = \int p(\theta)p(y|\theta)d\theta$.

Equation (1) is the engine driving the Bayesian perspective. This equation illuminates the three main differences between the Bayesian perspective and the non-Bayesian or *frequentist* perspective. First, from a Bayesian point of view, the parameter vector θ is assumed to be random. This is very different than the frequentist approach taught in many elementary statistics classes, where the parameter is assumed to be a fixed, unknown quantity. Secondly, it can be argued that it is the perceived arbitrariness of the prior distribution imposed upon θ that truly divides statisticians into two camps. Although hard-core Bayesians would maintain that all priors ought to be built on knowledge about the parameters, there are Bayesians who enjoy choosing *conjugate priors*, which yield posteriors of a known parametric type, explained below. On the other hand, non-Bayesians may agree (grudgingly) only with the use of *non-informative priors*. Thirdly, the interpretation of even markedly similar numerical results are different depending on the perspective of the statistician.

A conjugate prior is one where, roughly speaking, the posterior distribution is of the same distribution type or class as the distribution type of the prior. For example, if $p(y|\theta)$ is Binomial(n,θ), then $p(\theta) = \text{Beta}(\alpha,\beta)$ is a conjugate prior since $p(\theta|y)$ is Beta$(y+\alpha, n-y+\beta)$ ([3] p. 298). A critic might point out that such a choice of prior is due to convenience (i.e. to get an analytic expression of the posterior that is well behaved), not necessarily due to a priori knowledge of the data.

When the modeler has little prior knowledge about θ, then it is prudent to use a flat or noninformative distribution for $p(\theta)$. For example, for values of $\theta \in [0,4]$, a noninformative prior is $p(\theta) = \frac{1}{4}$. Using non-informative priors can yield parameter estimates that are similar to the estimates obtained by non-Bayesian methods. For example, in linear regression, the vector of parameters is $\theta = \begin{pmatrix} \beta \\ \sigma \end{pmatrix}$, where $\beta \in \Re^{k-1}$ is a vector of regression coefficients and $\sigma^2 > 0$. One usually assumes that the sampling distribution is $p(y|\beta,\sigma^2) = N(X\beta,\sigma^2 I_n)$ where X is a $n \times (k-1)$ matrix of constants, and for simplicity I assume that $\text{rank}(X) = k-1$. If one chooses a non-informative prior for θ, then the posterior for β is multivariate normal, $p(\beta|y) = N(\hat{\beta},\sigma^2 I_n)$, where $\hat{\beta}$ is the Best Linear Unbiased Estimator of β ([5] p. 85). The 95%*posterior interval estimate* for a particular regression coefficient β_i,

$$\hat{\beta}_i \pm t^{.975}_{n-k+1} SE(\hat{\beta}_i), \tag{2}$$

is the same interval estimate obtained by a frequentist's 95% confidence interval for β_i.

The last example illustrates the point made earlier that, even with the same interval estimate of a parameter, the Bayesian interpretation of the interval is different than the frequentist's. A Bayesian would say that (2) contains the parameter β_i with probability .95. The frequentist would say that once $\hat{\beta}_i$ is computed from data and substituted into (2), then one is 95% *confident* that β_i is in the interval. No statements about probability are appropriate, since the frequentist maintains that the parameter is either in the interval or it is not, and the level of confidence now measures the uncertainty of whether β_i is in the interval. It can be argued that the introduction of this new measure of uncertainty is weird. In some sense, since a probability is already a measure of uncertainty, the Bayesian notion of using a probability to measure the uncertainty of whether the interval contains the parameter is more satisfactory.

A satisfactory way to obtain a prior distribution that is driven by the data is to update a prior by the posterior. Let $p(\theta|y_1)$ be the posterior given that a the data y_1 was observed, and suppose that a new data vector y_2 is observed, such that $y_1, y_2 \sim p(y|\theta)$. The posterior $p(\theta|y_2)$ is found by setting $p(\theta) := p(\theta|y_1)$. This methodology can yield posteriors that are not of a known parametric family, and so numerical techniques are necessary to estimate these posteriors. In this paper, some of these numerical techniques are illustrated.

3 Simple Bayesian Models

To illustrate how one might use the posterior distribution for inference, consider a study performed by the Educational Testing Service to analyze the effects of special coaching programs on SAT test scores ([4] p. 143-4). Assuming that we have a simple random sample of 240 pairs, $\{(z_{i1}, z_{i2})\}$, of SAT scores, the score z_{i1} recorded on individual i before being coached, and the score z_{i2} recorded after being coached. Let $y_i = z_{i2} - z_{i1}$, and suppose that the mean of the differences of these two scores is $\bar{y} = 7.87$, that $\sigma = 64.5$ is known, and that the population mean θ is the parameter of interest. Gelman et al. ([4] p. 42-5) show that if σ^2 is known, so that $p(y|\theta) = N(\theta,\sigma^2)$, and the prior $p(\theta) = N(\mu,\tau^2)$ with μ and τ^2 known, then the posterior is

$$p(\theta|y) = N\left(\frac{\frac{\mu}{\tau^2} + \frac{n}{\sigma^2}\bar{y}}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}, \frac{1}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}\right).$$

The term $\frac{1}{\tau^2}$ is called the *prior precision* since τ^2 is the variance of the prior, which, if large, implies that we have little knowledge of θ. The term $\frac{n}{\sigma^2}$ is called the *data precision*. As $\tau \to \infty$, then the prior precision goes to zero, essentially giving a noninformative prior for θ. The posterior in this instance simplifies to

$$p(\theta|y) = N(\bar{y}, \sigma^2/n)$$

([4] p. 45 and 67). This is precisely the distribution used for inference about θ from the frequentist's point of view. In the present case, we have that $p(\theta|y) = N(7.87, 17.35)$. Thus, a 95% posterior distribution for θ is $(-0.29, 16.04)$, which agrees with the frequentist's 95% confidence interval for θ. To illustrate how one can estimate the posterior when the distribution is not of a known parametric form, I estimate this normal posterior by drawing 100 times from $N(7.87, 17.35)$. The corresponding histogram is in Figure 1(a).

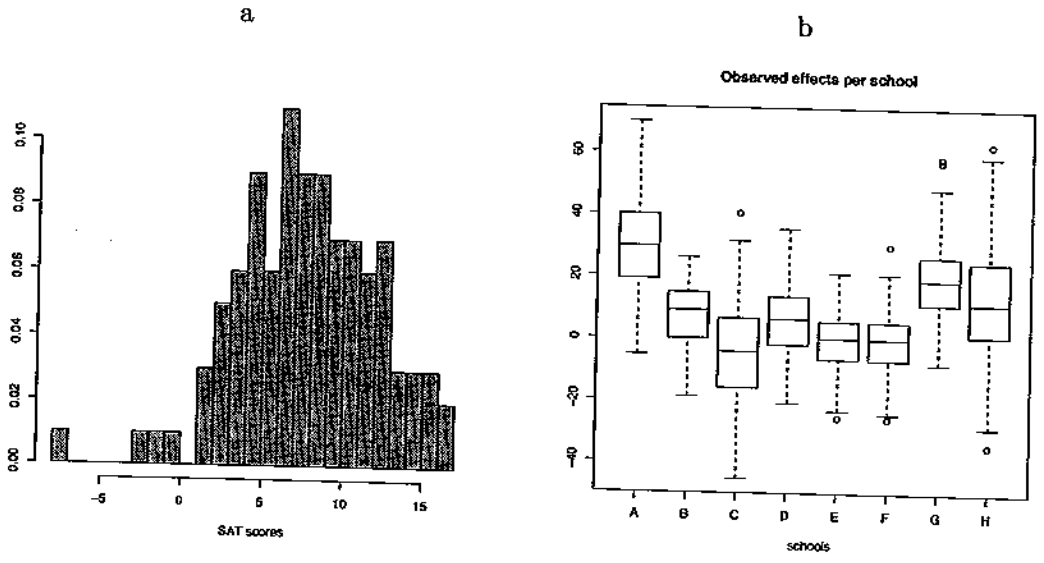

Figure 1: (a) A histogram of 100 draws from $N(7.87, 17.35)$, illustrating how to estimate an arbitrary posterior distribution. (b) Boxplots of 100 draws from separate independent normal estimates for the effects of the coaching programs at eight schools (A through H) on SAT test scores. Note that it is difficult distinguish the effects between any of the schools. The R code which generated these graphs is in Appendix 8.1.

This example can be extended to illustrate how one can use a Bayesian approach to consider differences in means of multiple populations as in an ANOVA. Now suppose that the coaching programs were at eight schools, with school averages given in the following table [6].

School	i	$\bar{y}_{i\cdot}$	s_i^2/n_i
A	1	28.39	14.9
B	2	7.94	10.2
C	3	-2.75	16.3
D	4	6.82	11
E	5	-0.64	9.4
F	6	.63	11.4
G	7	18.01	10.4
H	8	12.16	17.6

Table 1: Average SAT scores from 8 different schools. Reproduced from [4], p. 143.

Following ([4] p. 142), suppose that

$$y_{ij}|\theta_i \sim N(\theta_i, \sigma_i^2)$$

where y_{ij} is the j^{th} observation from the i^{th} school, θ_i is the mean effect of school i, and σ_i^2 is the known variance of school i. Assuming a normal prior for each θ_i, so that $\theta_i \sim N(\mu_i, \sigma_i^2/n_i)$, then, as pointed out above, $p(\theta_i|y) = N(\bar{y}_{i\cdot}, \sigma_i^2/n_i)$. One can approximate the posterior interval estimates for each θ_i by assuming that $\sigma_i^2/n_i = s_i^2/n_i$ from Table 1, and then randomly drawing samples from $p(\theta_i|y)$. The resulting posterior interval estimates, constructed from 100 draws from each normal distribution, are depicted graphically in the boxplots in Figure 1(b). We see that the school effects are not statistically distinguishable due to the large variability of each $\theta_i|y$.

4 Hierarchical Models

In the example involving SAT scores, we assumed that we knew μ and τ, the parameters of the prior distribution. We now present a framework which can be used to consider these as random, called a Bayesian hierarchical model. In a hierarchical model, the distribution of the parameters θ is dependent on another set of parameters, $\phi \in \Re^t$, so that the prior is now given as $p(\theta|\phi)$. This requires a distribution, $p(\phi)$, called a *hyperprior*, to be specified for ϕ. This yields the *joint posterior distribution*, as in (1),

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

where the last equality follows if we assume that the data y depends on ϕ only through θ. Not surprisingly, it can be difficult to get an analytic expression for $p(\theta, \phi) = p(\phi)p(\theta|\phi)$. To deal with this issue, the factorization

$$p(\theta, \phi|y) = p(\phi|y)p(\theta|\phi, y) \tag{3}$$

of the posterior is useful, which follows from the definition of conditional probability. This shows that if one can estimate $p(\phi|y)$ and $p(\theta|\phi, y)$ from data y, then one can use numerical simulations to estimate the posterior by the following algorithm.

Algorithm 1 (Basic Simulation of a posterior distribution) *([4] p. 129)*

1. *Draw vector of hyperparameters $\phi_i \in \Re^t$ from $p(\phi|y)$. If t is large, then it is recommended to perform this step via one of the methods introduced in section 5.*
2. *Draw a vector of parameters $\theta_i \in \Re^k$ from $p(\theta|\phi_i, y)$.*
3. *Perform steps 1-2 N times. The posterior is approximated by the distribution of $\{(\theta_i, \phi_i)\}_{i=1}^N$.*

To illustrate these ideas, I applied a normal hierarchical model to the SAT test score data as did Gelman et al. in [4], p.144-148. Let

- $\bar{y}_{i\cdot}$, the mean of the sample data for school i, be the estimated coaching effect of the special coaching program at school i.
- σ_i^2 be the corresponding sample variance for school i, which we assume is known.

θ_i be the actual coaching effect in school i, which has known variance σ^2 so that the sampling distribution is $p(\bar{y}_i.|\theta_i, \sigma_i^2) = N(\theta_i, \sigma_i^2)$

μ and τ be hyper-parameters, so that the prior is given by

$$\theta_j|\mu, \tau \sim N(\mu, \tau^2)$$

A flat, noninformative prior density is assumed for the hyperprior, $p(\mu, \tau) \propto 1$. By (3), the posterior can be factored as

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

This last form is desirable since each of the three factors on the right hand side have analytic expressions given our hierarchical model (see [4], equations 5.17, 5.20 and 5.21). In this case, $\phi = \begin{pmatrix} \mu \\ \tau \end{pmatrix}$. Step 1 of Algorithm 1 is performed by drawing τ from $p(\tau|y)$, then drawing μ from $p(\mu|\tau, y)$, and then drawing θ from $p(\theta|\mu, \tau, y)$. I wrote R code which applied Algorithm 1 to simulate the joint posterior (see Appendix 8.1). Observe that $\theta \in \Re^8$ in this example. Figure 2 is an attempt to give a graphical display of this simulation of the joint posterior over this eight dimensional space.

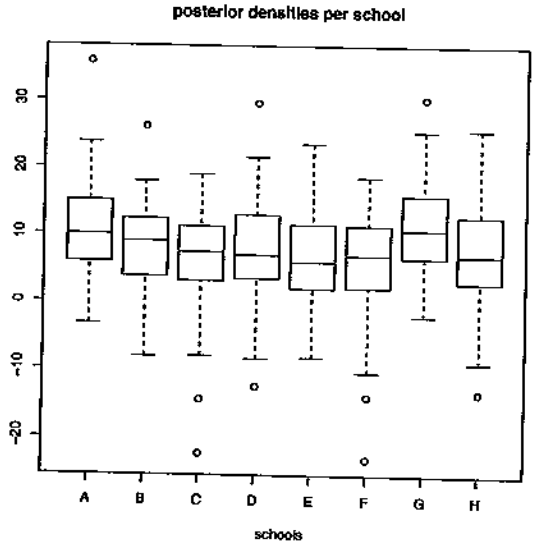

Figure 2: Posterior distribution $p(\theta_i, \mu, \tau|y)$ for the SAT scores for $i = 1, ..., 8$ schools. This suggests that the school effects are more similar than predicted by modelling $\theta_i|y$ with separate normals (Figure 1(b)).

In Figure 3(a), I have shown a simulation of the marginal posterior density $p(\tau|y)$. From this density, we see that values of τ near zero are most probable, $p(\tau > 10|y) < .5$ and $p(\tau > 25|y) \approx 0$. The conditional posterior means $E(\theta_i|\tau, y)$ (for each of the eight schools) as a function of τ is shown in Figure 3(b). Comparing to 3(a), we see that for most of the likely values of τ, the estimated effects are close together, as we noticed in Figure 1(a). As τ becomes larger, corresponding to more variability among the schools, the estimates become more like the means shown in Figure 1(b) ([4] p.145-6) Thus, the Bayesian hierarchical model explains the analysis when we pooled all of the data together (Figure 1(a)), as well as the ANOVA-type analysis (Figure 1(b)).

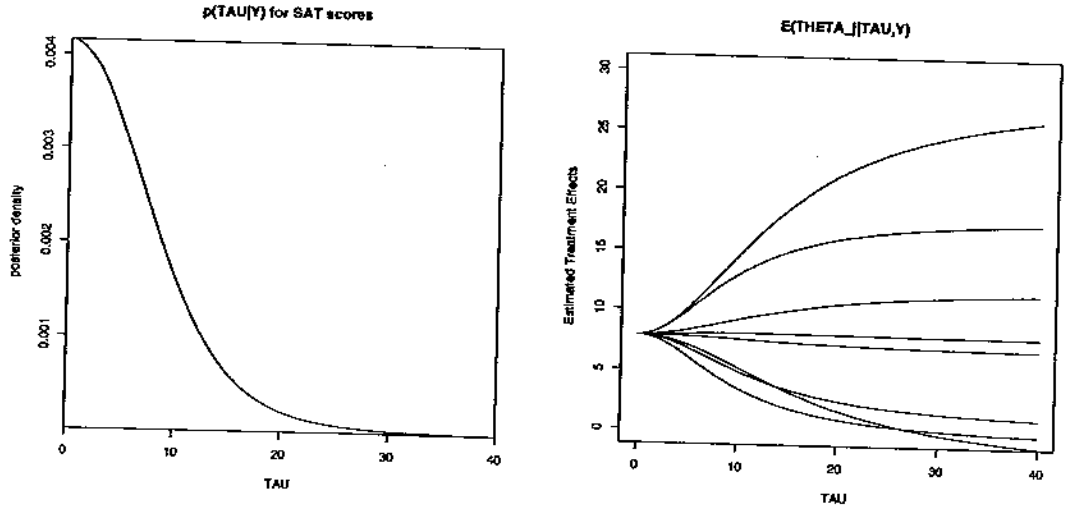

Figure 3: (a) The marginal posterior density function $p(\tau|y)$. (b) The conditional posterior mean $E(\theta_i|\tau, y)$ as a function of τ. .

5 Simulating Complex Hierarchical Models

If the posterior density is complicated or if there are many parameters of interest (that is, if k and t are large), then marginal posteriors for individual parameters can be difficult to obtain numerically using Algorithm 1. One approach to deal with such complicated models is to find the mode of the posterior, then to construct an approximation to the posterior about this mode. I present the method of *conditional maximization* or *step-wise ascent* to find the mode of the posterior, and also *expectation maximization*, which is used to find the mode of a marginal posterior. Multivariate normal and t-distributions are commonly used to approximate the posterior about his mode ([4] p. 275). *Importance resampling* can be used to improve upon this approximation. I also used the *Gibbs sampling approach* to determining posterior approximations. Gibbs sampling is an example of a general class of *Markov chain techniques* [7].

These procedures can be demonstrated by applying the hierarchical normal model examined in the last section to the blood coagulation times, y_{ij}, of 24 animals randomly assigned to four different diets ($i \in \{1, 2, 3, 4\}$) [2].

Diet i	y_{ij}							
1	62	60	63	59				
2	63	67	71	64	65	66		
3	68	66	71	67	68	68		
4	56	62	60	61	63	64	63	59

Table 2: Blood coagulation times in seconds for blood drawn from 24 animals randomly assigned to 4 diets. Reproduced from [4], p. 284.

The following normal hierarchical model for this experiment is similar to that used for the ETS data ([4] p.284-290):

θ_j is the actual coagulation time for animals on diet j, with ~~known~~ variance σ^2, so that

$$y_{ij}|\theta_j, \sigma \sim N(\theta_j, \sigma^2)$$

n_j is the number of animals on diet j

μ and τ are hyper-parameters:

$$\theta_j|\mu, \tau \sim N(\mu, \tau^2)$$

At this point, we deviate from the approach used with the ETS data, and assume the prior

$$p(\mu, \log\sigma, \log\tau) \propto \tau$$

and so the following factorization of the posterior is useful:

$$\begin{aligned} p(\theta, \mu, \log\sigma, \log\tau|y) &\propto p(\theta, \mu, \log\sigma, \log\tau)p(y|\theta, \mu, \log\sigma, \log\tau) \\ &= p(\mu, \log\sigma, \log\tau)p(\theta|\mu, \log\sigma, \log\tau)p(y|\theta, \mu, \log\sigma, \log\tau) \\ &= p(\mu, \log\sigma, \log\tau)p(\theta|\mu, \log\tau)p(y|\theta, \log\sigma) \end{aligned}$$

where the last equality holds since θ is independent of σ^2, and μ and τ affect y through θ and $\log\sigma$. Therefore

$$p(\theta, \phi|y) = p(\theta, \mu, \log\sigma, \log\tau|y) \propto \tau \prod_{j=1}^{4} N_d(\theta_j|\mu, \tau^2) \prod_{j=1}^{4} \prod_{i=1}^{n_j} N_d(y_{ij}|\theta_j, \sigma^2)$$

([4] p. 288) since these are the densities assumed by the model outline above. I use the notation $N_d(\cdot, \cdot)$ to stress that I am using the explicit normal densities in the equation.

Starting at a crude initial parameter estimate, conditional maximization (step-wise ascent) finds the mode of a given joint posterior distribution $p(\theta, \phi)$ by iteratively maximizing $p(\theta, \phi)$ over each individual component of (θ, ϕ) while leaving the other components fixed at their previous values. This process is repeated until some convergence criterion is met, (such as the gradient being close to zero).

I wrote R code (see Appendix 8.2) which implemented conditional maximization on the normal hierarchical model for the blood coagulation time data. Using this code, $(\theta, \mu, \log\sigma, \log\tau|y)$, converges to a mode of $p(\theta, \mu, \log\sigma, \log\tau|y)$ after 8 iterations:

	crude	1st	2nd		7th	8th
theta1	61.000000	61.281553	61.290783	...	61.292425	61.292426
theta2	66.000000	65.870824	65.867621	...	65.866975	65.866975
theta3	68.000000	67.741648	67.734547		67.733177	67.733176
theta4	61.000000	61.147708	61.152770		61.153673	61.153674
mu	64.000000	64.010433	64.011430		64.011563	64.011563
sigma	2.290768	2.169795	2.170377		2.170487	2.170487
tau	3.559026	3.317881	3.310635		3.309290	3.309290
log(jp)	-61.603540	-61.419440	-61.419319	...	-61.419316	-61.419316

It is prudent to mention at this point that when it is difficult to maximize the posterior $p(\theta, \phi|y)$ directly, or if we are only interested in simulating $p(\theta|y)$ for which it is difficult to find a mode, then one might consider the expectation maximization method. Expectation maximization finds the mode of $p(\theta|y)$ by the following algorithm.

Algorithm 2 (Expectation Maximization) *[4] p. 278*

1. *Start with a crude parameter estimate* θ^0.

2. E-step*: Integrate out* ϕ *by computing*

$$E^i(\log p(\theta, \phi|y)) = \int \log\left(p(\theta, \phi|y)\right) p(\theta|\theta^i, y) d\phi.$$

3. M-step*: Let* θ^{i+1} *be the maximizer of* $E^i(\log p(\theta, \phi|y))$

Once the posterior mode is found, a multivariate normal or multivariate t approximation is fit to $p(\phi)$ at the posterior mode. The scale matrix for the multivariate t is the numerically computed Hessian (via finite differences) of the marginal posterior density at the mode. To simulate the posterior density $p(\theta, \phi|y)$, first draw ϕ from the t approximation for $p(\phi)$, then draw θ from $p(\theta|\phi)$. For example, in the blood coagulation times example, we fit a multivariate t_4 approximation for $p(\mu, \log\sigma, \log\tau)$ at the marginal posterior mode found by Conditional Maximization, given in the R output above. To simulate the posterior density $p(\theta, \mu, \log\sigma, \log\tau|y)$, first draw $(\mu, \log\sigma, \log\tau)$ from the t_4 approximation for $p(\mu, \log\sigma, \log\tau)$, then draw θ from $p(\theta|\mu, \log\sigma, \log\tau)$, which, as we have already seen, is simply $\prod_{j=1}^4 N_d(\theta_j|\mu, \tau^2)$ ([4] p. 335-337).

Let ψ be the vector of all parameters in which we are interested, so that $\psi = (\theta, \phi)$. When simulating an approximation to $p(\psi|y)$, such as with a multivariate normal or multivariate t, one can improve upon this approximation by *importance resampling.*

Algorithm 3 (Importance Resampling) *[4] p. 312*

1. *Draw* L *times from the distribution* g.

2. *Draw* $K < L$ *times, without replacement, from* $\{\psi^1, ..., \psi^L\}$*, where the probability of drawing* ψ^i *is proportional to the* resampling weight $\frac{p(\psi=\psi^i|y)}{g(\psi^i)}$.

The last, and most complex algorithm which I introduce here is a Markov chain algorithm, called Gibbs sampling. If one divides up the parameter vector $\theta \in \Re^k$ into d subvectors, $\theta = (\theta_1, ..., \theta_d)$, then this method cycles through the subvectors $\{\theta_i\}_{i=1}^d$, drawing θ_j from $p(\theta_j|\{\theta_i\}_{i\neq j})$. These d draws are performed in each iteration of the Gibbs sampling algorithm.

To monitor convergence of the Gibbs sampler, I consider the estimated *potential scale reduction* $\sqrt{\hat{R}}$, given in [4] p. 332. For values of $\sqrt{\hat{R}|\theta_j}$ close to 1, we conclude that θ_j has converged. One must require that $\sqrt{\hat{R}|\theta_j} \approx 1$ for all j before concluding that the full vector θ has converged.

In an attempt to put together the concepts introduced in this section, I implemented the following algorithm in R to simulate the posterior distribution given the hierarchical model for the blood coagulation times (see Appendix 8.3).

1. Performing Conditional Maximization found a posterior mode at

$$\begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \mu \\ \tau \end{pmatrix} = \begin{pmatrix} 61.3 \\ 65.9 \\ 67.7 \\ 61.2 \\ 2.17 \\ 3.31 \end{pmatrix}$$

2. A multivariate t_4 density is used to approximate $p(\theta, \mu, \log\sigma, \log\tau)$ about this mode.

3. $L = 1000$ draws are taken from the t_4 approximation.

4. The initial values for $K = 10$ Gibbs sequences are obtained via importance resampling, with resampling weight equal to $p(\theta, \mu, \log\sigma, \log\tau|y)$

5. Run the Gibbs Sampler for $M = 100$ iterations, using the initial values of μ, σ, τ found in step 4. These steps, as well as verification of the claims as to the form of the distributions, can be found in ([4] p. 286, 336):

   (a) For each j, draw θ_j from $p(\theta_j|\mu, \sigma, \tau, y)$, which is normal.

   (b) Draw σ^2 from $p(\sigma^2|\theta, \mu, \tau, y)$, an inverse-$\chi^2$.

   (c) Draw μ from $p(\mu|\theta, \sigma, \tau, y)$, which is normal.

   (d) Draw τ^2 from $p(\tau^2|\theta, \mu, \sigma, y)$, which is inverse-$\chi^2$.

The following table shows the summary of posterior inference for the individual parameters from the second halves of the Gibbs sequences (the responsible R code is in Appendix 8.3).

	Quantiles					
	2.5	25	50	75	97.5	sqrt(R)
theta1	58.85	60.37	61.26	61.98	63.64	1.0041
theta2	63.77	65.24	65.87	66.59	67.97	1.0085
theta3	65.91	67.18	67.81	68.57	69.68	1.0030
theta4	59.18	60.42	61.11	61.67	62.78	1.0065
mu	55.88	62.08	63.90	65.83	73.17	0.9986
sigma	1.81	2.15	2.40	2.65	3.54	1.0050
tao	2.11	3.69	5.50	8.34	26.77	1.0186
log(jp)	-95.31	-62.95	-62.19	-64.56	-74.30	

The last column in the table shows the potential scale reduction for each parameter. Values "near" 1 indicate that convergence for that parameter has occurred. In practice, values less than 1.2 are acceptable ([4] p. 332).

6 Mixture Models

Suppose that one wishes to model $y \in \Re^n$ as a mixture of M densities or *mixture components*, $\{p(y|\theta_j)\}_{i=1}^M$. A (finite) mixture model is one in which the sampling distribution for y is

$$p(y|\theta, \lambda) = \sum_{j=1}^{M} \lambda_j p(y|\theta_j),$$

where λ_j is the proportion of the population from mixture component j. Thus, $\sum_j \lambda_j = 1$. This requirement allows the vector $\lambda \in \Re^M$ to be interpreted as a discrete prior distribution on θ, $p(\theta_j) = \lambda_j$.

A mixture can be used to model data drawn from M populations. Introducing another parameter, will make this use clear. Define the random indicator variable ζ_{ij} by

$$\zeta_{ij} = \begin{cases} 1 & \text{if the } i^{th} \text{ observation was from the } j^{th} \text{ population} \\ 0 & \text{otherwise} \end{cases}$$

Then

$$p(y, \zeta|\theta, \lambda) = \Pi_{i=1}^{n} \Pi_{j=1}^{M} (\lambda_j p(y_i|\theta_j))^{\zeta_{ij}},$$

with exactly one $\zeta_{ij} = 1$ for each i ([4] p. 421).

One can use the methods introduced in section 5 to perform inference on the parameters of interest, $\theta \in \Re^k$. To illustrate this, I verified the analysis of Gelman et al. ([4] p. 426-438) on the thirty log response times for 11 schizophrenics and 6 non-schizophrenic individuals [1]. The basis for this study is the following quote:

> Psychology Theory from the last half century and before suggests a model in which schizophrenics suffer from an attention deficit on some trials, as well as general motor retardation; both aspects lead to relatively slower responses for the schizophrenics, **with motor retardation affecting all trials and attentional deficiency only some** [4] p. 426.

Observe that a mixture model seems appropriate since we are interested in modelling data from two populations, schizophrenics and non-schizophrenics. The parameters for the mixture model are as follows:

y_{ij} is the i^{th} log response for individual j

S_j is an indicator variable for individual j: 1 if schizophrenic, 0 otherwise

ζ_{ij} is an indicator variable: 1 if y_{ij} was affected by an attention deficit, 0 otherwise

α_j is the average log response time for individual j. For schizophrenics ($j = 12, ..., 17$), α_j is the average log response time when there is no delay caused by an attention deficit.

τ is the log delay caused by an attention deficit. That is, the average log response time is $\mu + \tau$ when affected by an attention deficit.

σ_y^2 is the common variance for $\alpha_j + \tau\zeta_{ij}$

μ is the mean of α_j for $j = 1, ..., 11$

β is the log delay caused by general motor reflex retardation, so the mean of α_j for $j = 12, ..., 17$ is $\mu + \beta S_j$

σ_α^2 is the common variance for $\mu + \beta S_j$

λ, the mixture component, is the proportion of schizophrenic responses that are delayed due to an attention deficit

Let $\phi = (\sigma_\alpha^2, \beta, \lambda, \tau, \mu, \sigma_y^2)$. A uniform, noninformative prior is assigned to ϕ such that $\sigma_\alpha, \beta, \tau, \sigma_y > 0$ and $\lambda \in [.001, .999]$. Now the-mixture model can be stated as:

$$\begin{aligned} y_{ij}|\alpha_j, \zeta_{ij}, \phi &\sim N(\alpha_j + \tau\zeta_{ij}, \sigma_y^2) \\ \alpha_j|\zeta_{ij}, \phi &\sim N(\mu + \beta S_j, \sigma_\alpha^2) \\ \zeta_{ij}|\phi &\sim Bern(\lambda S_j) \end{aligned}$$

I proceeded with the model analysis as I did in section 5. The location of the R code which implemented these analyzes is given in Appendix 8.4. Since the code was extensive, I broke out the R code into several modules, which correspond to the steps below .

1. Obtain "crude estimates" of the parameters (α, ϕ) as given in ([4] p. 428-9), then divide each parameter by a χ_1^2 random variable in an attempt to spread the points out over the parameter space so that we find all major modes.

2. Expectation maximization, starting from the values found in step 1., finds three modes of $p(\alpha, \phi|y)$ after 100 iterations, a major mode and two minor modes on the edge of the parameter space (R code is schiz.ecm.r).

3. Fit a t_4 approximation at the major mode found in step 2 (R code is schiz.t4.r)

4. Draw $L = 100$ points from the t_4 approximation.

5. The initial values for $K = 10$ Gibbs sequences are obtained via importance resampling, with resampling weight equal to $p(\alpha, \phi|y)$.

6. Perform Gibbs Sampler for $M = 200$ iterations, using the initial values of α, ϕ found in step 5 (R code is schiz.gibbs.r).

The result of this (extensive, slow) simulation is that, under this model, there is strong evidence that the average reaction times are slower for schizophrenics (due to general motor retardation) since a 95% posterior interval for e^{β} is

$$[1.18, 1.62].$$

The response delay due to attention deficiency is infrequent, since the mixture component λ has a 95% posterior interval

$$[.07, .18].$$

However, a 95% posterior interval for e^{τ} is

$$[2.1, 2.61],$$

which indicates that although infrequent, the delay is significantly larger than 0.

7 Conclusion

I performed the analyses summarized here in the Fall of 2000, with Jim Robison-Cox acting as mentor and stat-sage. I am very appreciative that Jim took the time to work with me throughout that semester, with no other reward than getting to look at my mug every week for an hour or two. I had zero statistical training at that time, and so the Bayesian point of view did not seem threatening or different to me. I had nothing to compare it to. In the face of formal training as a statistician, I have outlined my renewed perspective in "What's under the hood?". The great advantage of the Bayesian approach, in my opinion, is the ability to perform inference using (a sequence of) marginal posteriors, which have no need to be of a particular family, due to fast computers, and numerical techniques such as the ones that I have outlined here.

In conclusion, I have attempted to outline, with a very thick pencil, the Bayesian approach to data analysis, and how it differs from the conventional, frequentist approach. The Bayesian perspective depends on probability statements made from a posterior distribution, or a collection of marginal posterior distributions. If these posterior distributions are ugly (complex), or are defined over a large parameter space, then I have outlined some approaches to deal with the computational complexities. I have analyzed three separate data sets in an attempt to elucidate the details of these methods, and my analyses replicate the analyses made by Gelman et al. However, I developed all of the R code used to implement these analyses.

8 APPENDIX: R Code

8.1 Hierarchical Model Construction and Analysis

```
# SAT scores from Rubin (1981) and in Gelam et al (2000), Table 5.2. p 143.
# All page numbers and equation ref's in comments refer to Gelman et al (2000)
# Considering 3 types of models: Seperate, Pooled, Bayesian p(THETA,MHU,TAU|DATA) (5.16)
# MODEL =
#       1:  Seperate Estimates, see 3.3, p 67
#       2:  Pooled Model, (5.13)
#       3:  Posterior Distribution for TAU, p(TAU|DATA)
#       4:  Compute E(theta|tau,y)
#       5:  Compute sd(theta|tau,y)
#       6:  DENSITIES and SAMPLING from the Posterior density p(THETA,MHU,YAU|Y)
#                   p(TAU|Y)*p(MHU|TAU,Y)*p(THETA|MHU,TAU,Y)

# housekeeping
library(MASS)

# The Data
m<-c(28.39,7.94,-2.75,6.82,-.64,.63,18.01,12.16)      # AVGY_j
n<-30                                                  # N_j
sigma<-c(14.9,10.2,16.3,11,9.4,11.4,10.4,17.6)        # SIGMA_j
sigma2<-sigma^2

# 1. Seperate Estimates
# Assuming that p(THETA|Y) is N(M,S^2/N)    (see 3.3, p67)
if (model==1) {
  #s<-sqrt(sigma2/n)                                  # std dev
  s<-sigma
  a<-rnorm(100,m[1],s[1])
  b<-rnorm(100,m[2],s[2])
  c<-rnorm(100,m[3],s[3])
  d<-rnorm(100,m[4],s[4])
  e<-rnorm(100,m[5],s[5])
  f<-rnorm(100,m[6],s[6])
  g<-rnorm(100,m[7],s[7])
  h<-rnorm(100,m[8],s[8])
  boxplot(a,b,c,d,e,f,g,h,xlab="schools",names=c("A","B","C","D","E","F","G","H"),
       main="Observed effects per school")
}

# 2. Pooled Estimate
if (model==2) {
  pooly<-sum(1/sigma2*m)/sum(1/sigma2)      # see 5.13
  vary<-1/sum(1/sigma2)
  print("95% CI for pooled estimate")
  print(qnorm(c(.025,.975),pooly,sqrt(vary)))
  truehist(rnorm(100,pooly,sqrt(vary)),nbins=25,xlab="SAT scores")
}
```

```
# 3. p(TAU|DATA)=postTAU see 5.21 and 5.20
if ((model==3)|(model==4)|(model==5)|(model==6)) {
  size<-1000
  sampsize<-100
  TAU<-seq(0,40,length=size)                    # get grid for TAU
  priorTAU<-dunif(.5,0,40)                  # see page 139 and 144
  postTAU<-seq(0,0,length=length(TAU))      # initialize p(TAU|Y)
  for (i in 1:length(TAU)) {
    den<-sigma2+TAU[i]^2
    V<-1/sum(1/den)
    MHUhat<-sum(m/den)/sum(1/den)
    postTAU[i]<-priorTAU*V^(1/2)*prod(1/den^(1/2)*exp((m-MHUhat)^2/(-2*den)))
  }
  postTAU<-postTAU/sum(postTAU)                 # normalize P(TAU|Y)

  if (model==3) {
  plot(TAU,postTAU,ylab="posterior density",main="p(TAU|Y) for SAT scores",
        ,xaxs="i",yaxs="i",type="l")
  }
}

# 4. Compute E(theta|tau,y)
# 5. Compute sd(theta|tau,y)
if ((model==4)|(model==5)) {
  E<-matrix(0,size,length(m))
  S<-matrix(0,size,length(m))
  for (i in 1:size) {
    Ti2<-TAU[i]^2
    DEN<-1/(sigma2+Ti2)
    V<-1/sum(DEN)                   # (5.20)
    MHUhat<-sum(m*DEN)/sum(DEN)     # (5.20)
    U<-1/(1/sigma2+1/Ti2)           # (5.17)

    E[i,1:length(m)]<-(m/sigma2+MHUhat/Ti2)*U          # from exercise 5.10
    S[i,1:length(m)]<-U+(U/Ti2)^2*V                    # from exercise 5.10
  }
  S<-sqrt(S)

  # plotting
  if (model==4) {
    plot(TAU,E[1:size,1],type="l",ylim=c(0,30),
          ylab="Estimated Treatment Effects",main="E(THETA_j|TAU,Y)")
    for (i in 2:length(m)) {
       lines(TAU,E[1:size,i])
    }
  }

  if (model==5) {
    plot(TAU,S[1:size,1],type="l",ylim=c(0,20),
```

```
          ylab="Estimated Standard Deviations",main="sd(THETA_j|TAU,Y)")
    for (i in 2:length(m)) {
      lines(TAU,S[1:size,i])
    }
  }

  #par(new='FALSE')
}

# 6. DENSITIES and SAMPLING from the Posterior density p(THETA,MHU,YAU|Y)
#            p(TAU|Y)*p(MHU|TAU,Y)*p(THETA|MHU,TAU,Y)
if (model==6) {
  # 1st: Assuming that TAU~uniform on [0,40], p139,140
  TAUgY<-seq(0,0,length=sampsize)       # initializing
  MHUgTY<-seq(0,0,length=sampsize)
  THETA<-matrix(0,length(m),sampsize)
  dTAUgY<-seq(0,0,length=sampsize)
  dMHUgTY<-seq(0,0,length=sampsize)
  dTHETA<-matrix(0,length(m),sampsize)
  jp<-seq(0,0,length=sampsize)
  for (i in 1:sampsize) {
    # 1st: Assuming that TAU~uniform on [0,40], p139,140, get sample of TAU|Y  (5.21)
    # Use inverse cdf here ...
    TAUgY[i]<-sample(TAU,1,replace=TRUE,postTAU)
    tmp<-(TAU==TAUgY[i])*postTAU
    tmp<-tmp[tmp>0]
    dTAUgY[i]<-tmp[1]
    # 2nd: Get MHU|TAU,Y  (5.20)
    DENgY<-sigma2+TAUgY[i]^2
    VgY<-1/sum(1/DENgY)
    MHUhatgY<-sum(m/DENgY)/sum(1/DENgY)
    MHUgTY[i]<-rnorm(1,MHUhatgY,sqrt(VgY))
    dMHUgTY[i]<-dnorm(MHUgTY[i],MHUhatgY,sqrt(VgY))
    # 3rd: Get THETA|MHU,TAO,Y     (5.17)
    for (j in 1:length(m)) {
      s2<-sigma2[j]
      mj<-m[j]
      Ti2<-TAUgY[i]^2
      MHUi<-MHUgTY[i]
      THhat<-(1/s2*mj+1/Ti2*MHUi)/(1/s2+1/Ti2)
      U<-1/(1/s2+1/Ti2)
      THETA[j,i]<-rnorm(1,THhat,sqrt(U))
      dTHETA[j,i]<-dnorm(THETA[j,i],THhat,sqrt(U))
    }
# Posterior Density: Since p(THETA,MHU,TAU|Y)=p(TAU|Y)*p(MHU|TAU,Y)*p(THETA|MHU,TAU,Y)
jp[i]<-dTAUgY[i]*dMHUgTY[i]*prod(dTHETA[1:length(m),i])

}
# Get 95% CI for each model
THETAm<-seq(0,0,length=length(m))
```

```
  THETAv<-seq(0,0,length=length(m))
  ci<-matrix(0,length(m),2)
  for (j in 1:length(m)) {
    x<-THETA[j,1:sampsize]
    x<-x[!is.na(x)]
    THETAm[j]<-mean(x)
    THETAv[j]<-var(x)
    ci[j,1:2]<-qnorm(c(.025,.975),THETAm[j],sqrt(THETAv[j]))
  }


  a<-THETA[1,1:sampsize]
  b<-THETA[2,1:sampsize]
  c<-THETA[3,1:sampsize]
  d<-THETA[4,1:sampsize]
  e<-THETA[5,1:sampsize]
  f<-THETA[6,1:sampsize]
  g<-THETA[7,1:sampsize]
  h<-THETA[8,1:sampsize]
  boxplot(a,b,c,d,e,f,g,h,xlab="schools",names=c("A","B","C","D","E","F","G","H"),
          main="posterior densities per school")
}
```

8.2 Conditional Maximization

```
# blood1.r
#
# BLOOD1(MODEL,MAXITER,S)
# Examining the Coagulation of Blood example on p 284 and p385 of Gelman
#  et # al., which has joint posterior density:
#     theta,mu,log(sigma),log(tao)|y ~ prod(j=1:4)N(thetaj|mu,tao^2)*
#         tao*prod(j=1:4)prod(i=1:n_j)N(y_ij|thetaj,sigma^2)
# using:
#    1. mode finding algorithm (if model==1)
#    2. Gibb's Sampler  (if model==2) and number of Gibb's seq=1 - If
#       >1, see bloodk.r
# INPUTS:
#  MODEL - 1=Mode Finding, 2=Gibb's Sampler
#  MAXITER - # of iterations to run - (I like 20-100)
#  S - Used for Model 2 - Make inference from the S_th spot in the sequence to the end
#      (I like .5)
#
blood1<-function(model,maxiter,S)
{
# Define algorithm parameters
#model<-1
#maxiter<-5

# MODEL specific params
# 1.   MODE FINDING ...
```

```
# 2.   GIBB'S SAMPLING
#S<-2/3  # Make inference from the S_th spot in the Gibb's sequence to the end

# define the data, y_ij
a<-c(62,60,63,59,NA,NA,NA,NA)
b<-c(63,67,71,64,65,66,NA,NA)
c<-c(68,66,71,67,68,68,NA,NA)
d<-c(56,62,60,61,63,64,63,59)
m<-8
n<-4
lenvec<-c(length(a[(!is.na(a))]),length(b[(!is.na(b))]),length(c[(!is.na(c))]),
        length(d[(!is.na(d))]))
N<-sum(lenvec)    # total samples across n experiments
lenmat<-t(matrix(c(4,6,6,8),n,m))
x<-matrix(c(a,b,c,d),m,n)
ymj<-apply(x,2,mean,na.rm=T)

# ********************************************
# Use mode finding algorithm to simulate the jp
# Get first estimates for thetaj, mu, sigma and tao
if (model==1) {
thetaj<-ymj                              # initial estimate for theta
thetam<-t(matrix(thetaj,n,m))
sigmaj2<-apply((x-thetam)^2/(lenmat-1),2,sum,na.rm=T)
sigma2<-mean(sigmaj2)
sigma<-sqrt(sigma2)
mu<-mean(thetaj)
tao2<-var(thetaj)
tao<-sqrt(tao2)

thetahat<-matrix(0,n,maxiter+1)
thetahat[1:n,1]<-thetaj
muhat<-seq(0,0,length=maxiter+1)
muhat[1]<-mu
sigmahat<-seq(0,0,length=maxiter+1)
sigmahat[1]<-sigma
taohat<-seq(0,0,length=maxiter+1)
taohat[1]<-tao
jp<-seq(0,0,length=maxiter+1)
jp[1]<-tao*prod(dnorm(thetaj,mu,tao))*prod(apply(dnorm(x,thetam,sigma),2,prod,na.rm=T))
for (i in 2:(maxiter+1)) {

 # Get conditional modes for each thetaj
 # (see 9.11 and 9.12 on p 286)
 thetahat[1:n,i]<-(1/tao2*mu + lenvec/sigma2*ymj)/(1/tao2 + lenvec/sigma2)
 thetaj<-thetahat[1:n,i]
 thetam<-t(matrix(thetaj,n,m))

 # Get conditional modes for mu (9.15)
 muhat[i]<-1/n*sum(thetaj)
```

```
   mu<-muhat[i]

   # Get conditional modes for sigma   (9.17)
   sigmahat[i]<-sqrt(1/N*sum(apply((x-thetam)^2,2,sum,na.rm=T)))
   sigma<-sigmahat[i]
   sigma2<-sigma^2

   # Get conditional modes for tao
   taohat[i]<-sqrt(1/(n-1)*sum((thetaj-mu)^2))
   tao<-taohat[i]
   tao2<-tao^2

   # Get log(p(theta,mu,log(sigma),log(tao)|y)) =
   #               tao*prod(j=1:4)prod(i=1:n_j)N(y_ij|thetaj,sigma^2)
   jp[i]<-tao*prod(dnorm(thetaj,mu,tao))*prod(apply(dnorm(x,thetam,sigma),2,prod,na.rm=T))

}
print("Using mode finding algorithm to simulate p(theta,mu,sigma,tao|y)")
}

# ******************************************
# Use Gibbs Sampler to simulate the jp
# Get first estimates for thetaj, mu, sigma and tao
if (model==2) {
thetaj<-ymj                          # initial estimate for theta
thetam<-t(matrix(thetaj,n,m))
sigmaj2<-apply((x-thetam)^2/(lenmat-1),2,sum,na.rm=T)
sigma2<-mean(sigmaj2)
sigma<-sqrt(sigma2)
mu<-mean(thetaj)
tao2<-var(thetaj)
tao<-sqrt(tao2)

thetahat<-matrix(0,n,maxiter+1)
thetahat[1:n,1]<-thetaj
muhat<-seq(0,0,length=maxiter+1)
muhat[1]<-mu
sigmahat<-seq(0,0,length=maxiter+1)
sigmahat[1]<-sigma
taohat<-seq(0,0,length=maxiter+1)
taohat[1]<-tao
jp<-seq(0,0,length=maxiter+1)
jp[1]<-tao*prod(dnorm(thetaj,mu,tao))*prod(apply(dnorm(x,thetam,sigma),2,prod,n))
for (i in 2:(maxiter+1)) {

  # Get theta from N(Mtheta,Vtheta) (9.11)
  Mtheta<-(1/tao2*mu + lenvec/sigma2*ymj)/(1/tao2 + lenvec/sigma2)
  Vtheta<-1/(1/tao2 + lenvec/sigma2)
  thetahat[1:n,i]<-rnorm(n,Mtheta,sqrt(Vtheta))
  thetaj<-thetahat[1:n,i]
```

```
    thetam<-t(matrix(thetaj,n,m))

    # Get mu from N(Mmu,Vmu)          (9.14)
    Mmu<-1/n*sum(thetaj)
    Vmu<-tao2/n
    muhat[i]<-rnorm(1,Mmu,sqrt(Vmu))
    mu<-muhat[i]

    # Get sigma from scaled Inv-ci^2(df=Dfsigma,scale=Ssigma)    (9.16)
    Dfsigma<-N
    Ssigma2<-1/N*sum(apply((x-thetam)^2,2,sum,na.rm=T))
    # scaled Inv-chi^2=Df*scale^2/chisq
    sigmahat[i]<-sqrt(Dfsigma*Ssigma2/rchisq(1,Dfsigma))
    # scaled Inv-chi^2=Inv-gamma
    #sigmahat[i]<-1/sqrt(rgamma(1,Dfsigma/2,Dfsigma/2*Ssigma2))
    sigma<-sigmahat[i]
    sigma2<-sigma^2

    # Get tao from scaled Inv-ci^2(df=Dftao,scale=Stao)       (9.18)
    Dftao<-n-1
    Stao2<-1/(n-1)*sum((thetaj-mu)^2)
    # scaled Inv-chi^2=scale^2/chisq
    taohat[i]<-sqrt(Dftao*Stao2/rchisq(1,Dftao))
    # scaled Inv-chi^2=Inv-gamma
    #taohat[i]<-1/sqrt(rgamma(1,Dftao/2,Dftao/2*Stao2))
    tao<-taohat[i]
    tao2<-tao^2

    # Get log(p(theta,mu,log(sigma),log(tao)|y)) =
    #               tao*prod(j=1:4)prod(i=1:n_j)N(y_ij|thetaj,sigma^2) (p 285)
    jp[i]<-tao*prod(dnorm(thetaj,mu,tao))*prod(apply(dnorm(x,thetam,sigma),2,prod,na.rm=T))

}
print("Using Gibb's Sampler to simulate p(theta,mu,sigma,tao|y)")
}

# Compute potential scale reduction, sqrt(R)
#psimean
#B<-N/(n-1)*sum

# Show numerical results of the method

  tot<-matrix(0,n+4,maxiter+1)
  tot[1:n,1:(maxiter+1)]<-thetahat
  tot[n+1,1:(maxiter+1)]<-muhat
  tot[n+2,1:(maxiter+1)]<-sigmahat
  tot[n+3,1:(maxiter+1)]<-taohat
  tot[n+4,1:(maxiter+1)]<-log(jp)
  rownames(tot)<-c("theta1","theta2","theta3","theta4","mu","sigma","tao","log(jp)")
  print(tot)
```

```
 if (model==2) {
   print("Posterior Inference for individual parameters")
   probs<-c(.025,.25,.5,.75,.975)
   suminf<-matrix(0,n+4,6)
   for (i in 1:n)  {
     suminf[i,1:5]<-quantile(thetahat[i,floor((maxiter+1)*S):(maxiter+1)],probs)
   }
   suminf[n+1,1:5]<-quantile(muhat[floor((maxiter+1)*S):(maxiter+1)],probs)
   suminf[n+2,1:5]<-quantile(sigmahat[floor((maxiter+1)*S):(maxiter+1)],probs)
   suminf[n+3,1:5]<-quantile(taohat[floor((maxiter+1)*S):(maxiter+1)],probs)

   dimnames(suminf)<-list(c("theta1","theta2","theta3","theta4","mu","sigma","tao",
                     "log(jp)"),c(2.5,25,50,75,97.5,"sqrt(R)"))
   print(suminf)

 }



 }
```

8.3 Importance Resampling and Gibbs Sampler

```
# bloodk.r
#
# BLOODK(MAXITER,L,K,S)
# Function to examine the Coagulation of Blood example on p 284 (sect
# 9.5) and p385 (sect 11.6) of Gelman
#  et # al., which has joint posterior density:
#     theta,mu,log(sigma),log(tao)|y ~ prod(j=1:4)N(thetaj|mu,tao^2)*
#         tao*prod(j=1:4)prod(i=1:n_j)N(y_ij|thetaj,sigma^2)
# using
#   Gibb's Sampler with more than one Gibb's sequence
#
# INPUTS:
#  MAXITER - # of iterations to run (for each Kth sequence) - (I like 20-100)
#  L - Number of initial samples to make for importance resampling (IR) -
#      set to 1 if don't want IR (I like 1000-2000) (sect 10.5)
#  K - Number of Gibb's sampler sequences (I like 2-10) - if you want
#      to use K=1, use blood1.r
#  S - Make Inference from the Sth spot in the Gibb's sequence to make
#      inference about the parameters (I like .5 - .7).   For example, S=0
#      considers the whole seq, S=.75 considers the last 1/4 of the seq
#      and S=1 only considers the last iterate of the seq
#

bloodk<-function(maxiter,L,K,S)
{
```

```
 # load needed function: getR
 source("getR.r")

 # FOR DEBUGGING FUNCTION - Define algorithm parameters
 #maxiter<-1
 #L<-1  # Number of initial samples to make for importance resampling
 #K<-5    # Number of Gibb's sampler saquences - must be >1
 #S<-.7  # Make inference from the S_th spot in the Gibb's sequence to the end

 # define the data, y_ij
 a<-c(62,60,63,59,NA,NA,NA,NA)
 b<-c(63,67,71,64,65,66,NA,NA)
 c<-c(68,66,71,67,68,68,NA,NA)
 d<-c(56,62,60,61,63,64,63,59)
 m<-8
 n<-4
 lenvec<-c(length(a[(!is.na(a))]),length(b[(!is.na(b))]),length(c[(!is.na(c))]),
        length(d[(!is.na(d))]))
 lenvecmat<-matrix(lenvec,n,K)
 N<-sum(lenvec)    # total samples across n experiments
 lenmat<-matrix(lenvec,m,n*K,,byrow=TRUE)
 x<-matrix(c(a,b,c,d),m,n)
 xmat<-matrix(x,m,n*K)      # block matrix with data X: XMAT=[X|X|X|...|X]
 ymj<-apply(x,2,mean,na.rm=T)
 ymjmat<-matrix(ymj,n,K)

 # Get first estimates for thetaj, mu, sigma and tao
 if (L==1) {                  # 1. start with K crude estimates from p. 285
   thetaj<-ymjmat
   thetam<-matrix(thetaj,m,n*K,byrow=TRUE) # THETAM=[THETA1|...|THETAn|THETA1|...|THETAn]
   sigmaj2<-apply((xmat-thetam)^2/(lenmat-1),2,sum,na.rm=T)
   sigma2<-apply(matrix(sigmaj2,n,K),2,mean)
   sigma<-sqrt(sigma2)
   mu<-apply(thetaj,2,mean)
   tao2<-apply(thetaj,2,var)
   tao<-sqrt(tao2)
}

if (L>1) {     # 2. start with IMPORTANCE RESAMPLING of size K from L
               #    draws from a t_4 distribution
   # t_4 is approx to p(mu,log(sigma),log(tao)) - see p 335 and table
   # 9.4, results from run of conditional maximization and EM
   # start<-matrix(0,L,3)
   lenstartmat<-matrix(lenvec,L,n,byrow=TRUE)
   ymjstartmat<-matrix(ymj,L,n,byrow=TRUE)
   mustart<-rt(L,4)*diff(c(65.29,62.73))/diff(qt(c(.25,.75),4))+64.05  # mu starts
   mustartmat<-matrix(mustart,L,n)

   sigmastart<-abs(rt(L,4)*diff(c(2.12,2.64))/diff(qt(c(.25,.75),4))+2.37) # sigma starts
```

```
    sigma2startmat<-matrix(sigmastart,L,n)^2

    taostart<-abs(rt(L,4)*diff(c(2.62,4.65))/diff(qt(c(.25,.75),4))+3.43) # tao starts
    tao2startmat<-matrix(taostart,L,n)^2

    # An L*n matrix, each row is THETA
    thetastart<-(1/tao2startmat*mustartmat + lenstartmat/sigma2startmat*ymjstartmat)/
                (1/tao2startmat + lenstartmat/sigma2startmat)
    Vstart<-1/(1/tao2startmat + lenstartmat/sigma2startmat)

    tempstart<-matrix(0,L,n)
    for (i in 1:L) {
      tempstart[i,1:n]<-apply(dnorm(x,matrix(thetastart[i,1:n],m,n,byrow=TRUE),
                        sigmastart[i]),2,prod,na.rm=TRUE)
    }
    # from (9.19), p(mu,logsigma,logtao|y) from p 288, an LX1 matrix
    weight<-taostart*apply((dnorm(thetastart,mustartmat,
            sqrt(tao2startmat))*tempstart*sqrt(Vstart)),1,prod)

    #IMPORTANCE RESAMPLING
    startindex<-sample(1:L,K,replace=FALSE,prob=weight)

    thetaj<-ymjmat
    mu<-mustart[startindex]
    sigma<-sigmastart[startindex]
    sigma2<-sigma^2
    tao<-taostart[startindex]
    tao2<-tao^2

}

thetahat<-matrix(0,n,(maxiter+1)*K)
thetahat[1:n,1:K]<-thetaj
muhat<-matrix(0,K,maxiter+1)
muhat[1:K,1]<-mu
sigmahat<-matrix(0,K,maxiter+1)
sigmahat[1:K,1]<-sigma
taohat<-matrix(0,K,maxiter+1)
taohat[1:K,1]<-tao

for (i in 2:(maxiter+1)) {

  # Get theta from N(Mtheta,Vtheta) (9.11)
  mumat<- matrix(mu,n,K,byrow=TRUE)
  sigma2mat<-matrix(sigma2,n,K,byrow=TRUE)
  tao2mat<-matrix(tao2,n,K,byrow=TRUE)

  Mtheta<-(1/tao2mat*mumat + lenvecmat/sigma2mat*ymjmat)/(1/tao2mat
          + lenvecmat/sigma2mat)
  Vtheta<-1/(1/tao2mat + lenvecmat/sigma2mat)
```

```
      thetahat[1:n,(i*K-K+1):(i*K)]<-rnorm(n*K,Mtheta,sqrt(Vtheta))
      thetaj<-thetahat[1:n,(i*K-K+1):(i*K)]
      thetam<-matrix(thetaj,m,n*K,byrow=TRUE)

      # Get mu from N(Mmu,Vmu)          (9.14)
      Mmu<-1/n*apply(thetaj,2,sum)
      Vmu<-tao2/n
      mu<-rnorm(K,Mmu,sqrt(Vmu))
      muhat[1:K,i]<-mu

      # Get sigma from scaled Inv-ci^2(df=Dfsigma,scale=Ssigma)    (9.16)
      Dfsigma<-N
      Ssigma2<-1/N*apply(matrix(apply((xmat-thetam)^2,2,sum,na.rm=T),n,K),2,sum)
      # scaled Inv-chi^2=Df*scale^2/chisq
      sigma<-sqrt(Dfsigma*Ssigma2/rchisq(K,Dfsigma))
      sigmahat[1:K,i]<-sigma
      sigma2<-sigma^2

      # Get tao from scaled Inv-chi^2(df=Dftao,scale=Stao)        (9.18)
      Dftao<-n-1
      Stao2<-1/(n-1)*apply((thetaj-mumat)^2,2,sum)
      # scaled Inv-chi^2=scale^2/chisq
      tao<-sqrt(Dftao*Stao2/rchisq(K,Dftao))
      taohat[1:K,i]<-tao
      tao2<-tao^2

  }
  print("Using Gibb's Sampler to simulate p(theta,mu,sigma,tao|y)")

  # Show numerical results of the method
  if (K==1) {
    tot<-matrix(0,n+4,maxiter+1)
    tot[1:n,1:(maxiter+1)]<-thetahat
    tot[n+1,1:(maxiter+1)]<-muhat
    tot[n+2,1:(maxiter+1)]<-sigmahat
    tot[n+3,1:(maxiter+1)]<-taohat
    tot[n+4,1:(maxiter+1)]<-log(jp)
    rownames(tot)<-c("theta1","theta2","theta3","theta4","mu","sigma","tao","log(jp)")
    print(tot)
  }

  print("Posterior Inference for individual parameters")
  probs<-c(.025,.25,.5,.75,.975)
  suminf<-matrix(0,n+4,6)
  Sindex<-ceiling((maxiter+1)*S)
  if (Sindex==0) Sindex<-1
  seqlen<-maxiter+2-Sindex
  #stlentheta<-(maxiter+1-Sindex)*K+1
  stlentheta<-(Sindex-1)*K+1
  seqlentheta<-K*(seqlen)
```

```
psitheta<-matrix(0,n,seqlentheta)
for (i in 1:n)  {
    psitheta[i,1:seqlentheta]<-thetahat[i,stlentheta:((maxiter+1)*K)]
    suminf[i,1:5]<-quantile(psitheta[i,1:seqlentheta],probs)
    # the arg passed to "getR" is thetai for each of the K Gibb's sequences
    suminf[i,6]<-getR(matrix(psitheta[i,1:seqlentheta],K,seqlen))

}

psimu<-muhat[1:K,Sindex:(maxiter+1)]
suminf[(n+1),1:5]<-quantile(psimu,probs)
suminf[(n+1),6]<-getR(psimu)
psisigma<-sigmahat[1:K,Sindex:(maxiter+1)]
suminf[(n+2),1:5]<-quantile(psisigma,probs)
suminf[(n+2),6]<-getR(psisigma)
psitao<-taohat[1:K,Sindex:(maxiter+1)]
suminf[(n+3),1:5]<-quantile(psitao,probs)
suminf[(n+3),6]<-getR(psitao)

jp<-seq(0,0,length=5)

for (i in 1:5) {
    # Get log(p(theta,mu,log(sigma),log(tao)|y)) =
    #               tao*prod(j=1:4)prod(i=1:n_j)N(y_ij|thetaj,sigma^2) (p 285)
    z<-suminf[1:(n+3),i]
    thetaj<-z[1:n]
    thetam<-matrix(thetaj,m,n,byrow=TRUE)
    mu<-z[5]
    sigma<-z[6]
    tao<-z[7]
    jp[i]<-tao*prod(dnorm(thetaj,mu,tao))*
          prod(apply(dnorm(x,thetam,sigma),2,prod,na.rm=T))
}
suminf[(n+4),1:5]<-log(jp)
suminf[(n+4),6]<-getR(log(jp))
dimnames(suminf)<-list(c("theta1","theta2","theta3","theta4","mu","sigma",
                  "tao","log(jp)"),c(2.5,25,50,75,97.5,"sqrt(R)"))
print(suminf)

}
```

8.4 Mixture Model Analysis

The R code used to analyze the schizophrenic data set was extensive, and so I chose not to include it here. It is available at

http://www.math.montana.edu/~parker/software/splus

References

[1] T. R. Belin and D. B. Rubin. Analysis of a finite mixture model with variance components. Proceedings of the American Statistical Association, Social Statistics Section, 1990.

[2] G. Box, W. G. Hunter, and J. S. Hunter. *Statistics for Experimenters.* New York: Wiley, 1978.

[3] G. Casella and R. L. Berger. *Statistical Inference.* Belmont CA: Duxbury Press, 1990.

[4] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Batesian Data Analysis.* New York: Chapman and Hall, 2000.

[5] J. Robison-Cox. Class notes for advanced linear regression, statistics 506. Montana State University, Bozeman, Montana, 2004.

[6] D. B. Rubin. Estimation in parallel randomized experiments. *Journal of Educational Statistics*, pages 377–401, 1981.

[7] D. Spiegelhalter, A. Thomas, N. Best, and W. Gilks. Bayesian inference using gibbs sampling. BUGS 0.5 Manual, 1995.