Bayesian Modeling in the Social Sciences: an introduction to Markov-Chain Monte Carlo

Simon Jackman[1]

[1]Department of Political Science, Stanford University, Stanford, CA 94305-6044. e-mail: jackman@stanford.edu

1 Philosophical Preliminaries

1.1 End of the Holy War?

- The biggest philosophical debate within statistics is to do with the nature of probability. Two camps are *frequentists* and *subjectivists*.
- Each side has a distinctive view about statistical inference: are we characterizing states of the world, or states of mind? Objectivist vs subjectivist.
- Bayesians are subjectivists, but most introductions to statistics encountered in the social sciences are presented from the frequentist perspective.
- Markov chain Monte Carlo methods have a Bayesian foundation; but have also heralded a new "pragmatism". Bayesian methods are now being widely used by people with little interest in the long "holy war" between frequentists and subjectivists.
- MCMC allows us to do (subjectivist) Bayesian statistics if we want, but for the most part, MCMC is a means to an end, a clever way to get parameter estimates and uncertainty assessments.

1.2 What is Probability?

The **mathematical** definition is uncontroversial:

> ...there is no problem about probability: it is simply a non-negative, additive set function, whose maximum value is unity
>
> Kyburg and Smokler (1980, 3), quoted in Barnett (1982, 72)

and Kolmogorov himself ruled out any questions regarding the *interpretation* of probabilities:

> The theory of probability, as a mathematical discipline, can and should be developed from axioms in exactly the same way as Geometry and Algebra. This means that after we have defined the elements to be studied and their basic relations, and have stated the axioms by which these relations are to be governed, all further

> exposition must be based exclusively on these axioms, independent of the usual concrete meaning of these elements and their relations
>
> Kolmogorov (1956, 1), quoted in Barnett (1982, 71)

Subjective versus objective interpretations of probability:

- **Objective**: probability is a characteristic of a process, defined as the *long-run* or *limiting frequency* of an event, A. That is, let m be the number of occurrences of A, and let n be the number of experiments. Then

$$\Pr(A) = \lim_{n \to \infty} \frac{m}{n}$$

  i.e., the *limit* of a *relative frequency*, the definition of probability proposed by Denis Poisson in 1837 (Leamer, 1978, 25).

  von Mises (1957):

  > ...we may say at once that, up to the present time [1928], no one has succeeded in developing a complete theory of probability without, sooner or later, introducing probability by means of the relative frequencies in long sequences. (p70)
  >
  > The rational concept of probability, which is the only basis of probability calculus, applies only to problems in which either the same event repeats itself again and again, or a great number of uniform elements are involved at the same time... [In] order to apply the theory of probability we must have a practically unlimited sequence of observations.
  >
  > p11, quoted in Barnett (1982, 75-76)

- "objective probability" describes a feature of the coin being flipped, the die being rolled, or the cards being dealt, etc.

- but what about cases where where are studying non-standardized, non-repetitive behavior?

  - What is the probability that a discovered sonata is the work of Beethoven?
  - What is the probability that Andrew Jackson was the eighth president of the United States? (Leamer, 1978, 25)

- What is the probability that the Democrats win the House of Representatives in November 2000?
- What is the probability that I am from New Zealand?
- What is the probability of nuclear war in the next five years? (Lee, 1989, 3)

The *frequentist* notion of probability struggles with these questions. Leamer (1978, 27):

> I just do not see how a frequentist can make meaningful probability statements at all. He can talk about classes of events and their respective physical, possibly objective frequencies. But if we reserve the adjective "probability" for set functions that both obey the [Kolmogorov] probability axioms and also indicate the likelihood of uncertain events, a frequentist statement that two times out of ten we will pull a red ball from the urn is no more a probability statement than the statement that 20% of the balls are red or that the red balls make up 20% of the mass of the balls. Only under certain subjective circumstances can we allow the frequency of .2 to be translated into the statement "The probability of drawing a red ball from the urn is .2"

A more radical position is taken by di Finetti:

> My thesis ... is simply this:
>
> PROBABILITY DOES NOT EXIST
>
> The abandonment of superstitious beliefs about...Fairies and Witches was an essential step along the road to scientific thinking. Probability, too, if regarded as something endowed with some kind of objective existence, is not less a misleading misconception, an illusory attempt to exteriorize or materialize our true probabilistic beliefs.
>
> In investigating the reasonableness of our own modes of thought and behaviour under uncertainty, all we require, and all that we are reasonably entitled to, is consistency among these beliefs, and their reasonable relation to any kind of relevant objective data ("relevant" in as much as subjectively deemed to be so). This is Probability Theory.

di Finetti (1974, x), quoted in Barnett (1982, 68)

- this **subjective** notion of probability is championed by almost all Bayesians (although Bayes himself was ambiguous on the question): a "degree of confidence" (James Bernoulli, 1713) or "degree of belief".
- Laplace in his *Philosophical Essay on Probabilities* (1820): for a hypothesis h and evidence or data e, there is a measurable numerical degree of certainty which "should be entertained in the truth of h in the light of e" (Howson and Urbach, 1993, 52).
- but not just *any* beliefs! Under what conditions can subjective beliefs be considered probabilities?
- when degrees of certainty are located in the closed unit interval, they have the interpretation of probabilities
- these degrees of belief also adhere to the laws of probability, if someone is behaving rationally.
- the Ramsey-de Finetti Theorem: let $p_1, p_2, \ldots$ be a set of *betting quotients* on hypotheses $h_1, h_2, \ldots$. If the p_i do not satisfy the probability axioms, then there is a betting strategy and a set of of S_i of stakes such that whoever follows this betting strategy will lose a finite sum whatever the truth values of the hypotheses turn out to be (Howson and Urbach, 1993, 79).

  n.b., $p = \frac{k}{1+k}$ where k is the odds on h you believe to be fair.

  i.e., if one was using "degrees of belief" as a basis on which to place bets on the truth of sequence of hypothesis, then one would lose money if those "degrees of belief" did not conform to the laws of probability (Kolmogorov's axioms). A reasonably straightforward proof appears in Howson and Urbach (1993, 79-86).
- in particular, how one **updates** beliefs about hypotheses in the light of evidence has to obey laws about conditional probability (Bayes' Rule, see below).

Controversy arises because objectivity is widely believed to be a critical component of scientific inquiry:

> Most modern statisticians will hardly consider the conclusions they arrive at (like the statements made by art critics...) as being of a "subjective" nature, i.e., depending on their mood, their taste, ... [that where one] does not find a significant result, whereas his colleague does, ... [this] is due to differences in their metabolism.... [On an extreme subjectivist view of probability] the statistician...gets a result like that of an ancient priest, whose statements the layman, unable to reproduce the statistician's order of preferences, has to accept without a possibility of criticism. Statistics, in as much as it would remain a science at all, would become a "sacred and secret" science.
>
> van Dantzig (1957) quoted in Barnett (1982, 95)

1.3 Critiques of Classical Statistical Practice

Frequentist inference doesn't permit us to make the conclusions we are used to thinking that it does.

1.3.1 Confidence Intervals

Lee (1989, vii):

> When I first learned a little statistics, I felt confused... Not because the mathematics was difficult...but because I found it difficult to follow the logic by which inferences were arrived from data.... the statement that a 95% confidence interval for an unknown parameter ran from -2 to +2 sounded as if the parameter lay in that interval with 95% probability and yet I was warned that all I could say was that if I carried out similar procedures time after time then the unknown parameters would lie in the confidence intervals I constructed 95% of the time.

Barnett (1982, 182):

> if we ... find that $\bar{x} = 29.8$, how are we to answer the question "how close is 29.8 to μ"? This is a most pertinent question to ask --- some might claim that it is the supreme consideration. Within the classic approach we must rest on any transferred properties of the long-term behavior of the *procedure* itself...
>
> But not everyone is satisfied by this attitude. Some critics suggest it is misconceived! Suppose from data we obtain a 95 per cent confidence interval for μ as $29.5 < \mu < 30.2$. It is suggested that it is of little value to someone need to act in this situation to know that the "95 per cent confidence" attaches not to the specific statement made, but to what proportion of such similarly based statements will be correct in the long run. Such an interpretation may justify the statistician in his professional activities at large, (in the long run his advice is correct on 95 per cent of occasions), but what of the present situation? A statistically naive client will almost inevitably attach the 95 per cent to the statement $29.5 < \mu < 30.2$; the statistician himself can hardly avoid doing so implicitly when he puts the information to any practical use! Any probability interpretation of

> such an attitude must be expressed in non-frequency terms, for example as a statement of *subjective* probability.
>
> For example, a random sample survey of American adults may indicate that mean income in the United States is $35,000. Assuming (rather implausibly) that income is normally distributed, we could estimate a 90% confidence interval for our sample mean, perhaps [$15,000, $55,000] for a modestly sized sample. Using conventional frequentist inference we can conclude that intervals like the one calculated would cover the true (population) mean income 90% of the time for repeated applications of the sampling procedure. The "repeated sampling" inference tells us neither whether the population mean lies within the estimated interval nor even with what probability the mean lies in the interval.... The only available conclusion concerns the long-run behavior... (Western and Jackman, 1994, 413)

But what about *non-repeatable data*? That is, there is no *data-generation process* (DGP) creating data sets for us, just a single set of data. e.g., comparative politics. How can we apply frequentist procedures?

Western and Jackman (1994) considered this problem. There is "uncertainty", but not of the objective, frequentists kind: rather, the researcher's (subjective) uncertainty about the process under study. Classically-trained students of comparative politics struggle with this: e.g.,

- In an analysis of advanced industrial democracies, Lange and Garrett (1987, 268) report that they "adhere to traditional standards [i.e., significance tests] while remaining unsure of their applicability"
- Gorin's (1980, 153) comparative study of inequality "cannot utilize probability theory to ascertain the level of [statistical] significance" for want of "random samples"; he goes on to note that asterisks in his results indicate a ".05 level of significance"

At this point, some comparativists invoke the notion of a "superpopulation", from which the observed histories of specific countries are just one possible realization.

> While this assumption avoids a deterministic theory of the data process, it is highly speculative compared to positive knowledge

about a sampling procedure or, in Fisher's uncompromising phrase, "the physical act of randomization"... Even more troubling, however, is the conclusion that conventional inference allows in this instance. Take the example of a confidence interval for a mean where we can conclude that under repeated realizations (which are acknowledged to be impossible), the interval would cover the true mean 90% of the time. We have no way of knowing whether the current interval is one of the fortunate 90% and no possibility of further replications (Western and Jackman, 1994, 414).

- The fiction of repeated sampling is somehow held out by frequentists as objective; seems ludicrous in the case of comparative politics and arguments that make use of "superpopulations" (somehow not subjective?)
- From a Bayesian perspective, frequentists are making recourse to imaginary data, and then calling their results objective (e.g., Jeffreys, 1961)!
- *p* values also have this property; Lee (1989, vii):

  > It sounded as if the statement that a null hypothesis was rejected at the 5% level meant that there was only a 5% chance of that hypothesis being true, and yet the books warned me this was not a permissable interpretation.

- See Howson and Urbach (1993, ch9) for a thorough critique of conventional significance testing.

1.3.2 Specification Searches

- Leamer's critique of conventional econometric practice in *Specification Searches* 1978 and in numerous articles with provocative titles (e.g., "Let's Take the Con out of Econometrics", 1983).
- *ad hockery* abounds in the study of non-experimental data (subtitle to *Specification Searches* is "Ad Hoc Inference with Nonexperimental Data").
- searching over many different configurations of independent variables, in order to come up with plausible results.

- results are then presented as if they were generated objectively, without any prejudices/hunches/biases as how the results should turn
- "the data are speaking for themselves", "the data are conveying their message..." etc; as if the researcher had nothing to do with it
- one-tailed tests, or arbitrary selection of critical value (i.e., rejection region) for statistical tests.
- **message**: prior beliefs enter into data analysis all the time --- it is inevitably a subjective enterprise --- so why pretend otherwise? Why not incoporate prior beliefs formally (i.e., use the Bayesian approach)?

2 Bayes' Rule

Most simply,

$$\Pr(H|E) = \frac{\Pr(E\&H)}{\Pr(E)} = \frac{\Pr(E|H)\Pr(H)}{\Pr(E)}$$

or in (subjective) words, the degree of belief in proposition H given evidence E is equal to the joint probability of H and E divided by the probability of E.

- $\Pr(H)$ is the **prior** degree of belief in H
- $\Pr(H|E)$ is the **posterior** degree of belief in H, in the sense of what "after looking at the evidence (E)"

Bayes' Rule is an accounting identity that follows from the probability axioms. Mathematically, it is as uncontroversial as the probability axiom on which it rests.

Turns on the axiom relating joint and conditional probabilities:

$$\Pr(A|B) = \frac{\Pr(A\&B)}{\Pr(B)}$$

or

$$\Pr(A|B)\Pr(B) = \Pr(A\&B),$$

where $\Pr(B) \neq 0$.

Thus Bayes' Rule is sometimes called the *rule of inverse probability*, since it shows how a conditional probability B given A can be turned into or inverted into a conditional probability A given B (Leamer, 1978, 39).

Consider the case when A is not a single event, but a set of n mutually exclusive and exhaustive events, (A_n) (Lee, 1989, 9):

$$\Pr(B) = \sum_n \Pr(B|A_n)\Pr(A_n)$$

and

$$\Pr(A_n|B)\Pr(B) = \Pr(A_n\&B) = \Pr(A_n)\Pr(B|A_n)$$

and if $\Pr(B) \neq 0$ then

$$\begin{aligned}
\Pr(A_n|B) &\propto \Pr(A_n)\Pr(B|A_n) \\
&= \frac{\Pr(A_n)\Pr(B|A_n)}{\Pr(B)} \\
&= \frac{\Pr(A_n)\Pr(B|A_n)}{\sum_m \Pr(A_m)\Pr(B|A_m)}
\end{aligned}$$

The first line is one of the more popular ways of stating Bayes' Rule: we shall see that is equivalent to the statement

a posterior is proportional to the prior times the likelihood

Recall that a likelihood can be defined as follows: when the joint probability density function (pdf) $f_n(\mathbf{y}|\theta)$ of the data is regarded as a function of a parameter θ it is called the **likelihood function**.

This is handy: for Bayesians, the likelihood function summarizes the information in the data about θ, and allows the Bayesian to move from prior beliefs to posterior beliefs:

$$\pi(\theta|\mathbf{y}) = \frac{\pi(\theta)\, p(\mathbf{y}|\boldsymbol{\theta})}{p(\mathbf{y})} \quad (\theta \in \Theta) \tag{1}$$

where

$$\begin{aligned} p(\mathbf{y}) &= \int_{\Theta} p(\mathbf{y}, \theta) d\theta \\ &= \int_{\Theta} \pi(\theta) p(\mathbf{y}|\theta) d\theta \end{aligned}$$

Since $p(\mathbf{y}|\theta)$ is (proportional to) the likelihood for the data, we can ignore the denominator and re-write equation (1) as

$$\begin{aligned} \pi(\theta|\mathbf{y}) &\propto \pi(\theta) p(\mathbf{y}|\boldsymbol{\theta}) \\ &\propto \text{prior} \times \text{ likelihood} \end{aligned}$$

It is also possible to interpret this as a **weighted average**: at each point in the parameter space, the posterior $\pi(\theta|\mathbf{y})$ is an average of the likelihood and the prior, but a weighted average, since the posterior is normalized to behave like a probability density (i.e., integrate to 1).

2.1 An Example

In the United Kingdom in 1975, a referendum was to be held as to whether the UK should stay part of the EC. At the recent general election, the proportion supporting Labour (L) was 52%, while the proportion supporting the Conservatives (C) was 48% (ignoring the other parties and non-voters). Many polls indicated that 55% of Labour supporters and 85% of Conservative

supporters intended to vote "Yes" (Y) in the EC referendum, and the remainder intended to vote "No" (N).

Suppose we meet someone and she says is that she intends to vote "Yes" in the referendum. What should we conclude about her partisan support? Applying Bayes' Rule:

$$\begin{aligned} \Pr(L|Y) &= \frac{\Pr(Y|L)P(L)}{\Pr(Y|L)\Pr(L) + \Pr(Y|C)\Pr(C)} \\ &= \frac{.55 \times .52}{(.55 \times .52) + (.85 \times .48)} \approx .41 \end{aligned}$$

2.2 Conjugacy

A prior distribution is said to be *naturally conjugate* with respect to a likelihood if it gives rise to a posterior distribution having the same parametric form as the prior.

- in practice, assumptions about the likelihood's functional form (binomial, normal, Poisson, etc) usually structure the choice of functional form for the prior.
- this is done by exploiting mathematical relations among probability distributions (usually in the exponential family)
- remember: posterior is proportional to a prior times a likelihood
- life is easier is the posterior that results has a known functional form
- e.g., Uniform prior and Binomial Likelihood $\rightarrow$ Beta posterior, because U(0,1) = Beta(1,1).
- e.g., Beta prior and Binomial Likelihood $\rightarrow$ Beta posterior
- e.g., Normal prior and Normal likelihood $\rightarrow$ Normal posterior
- distributional assumptions and conjugacy less of a consideration with "modern Bayes" (computational power allows us to approximate distributions). e.g., Figure 2.4 in Gelman et al. (1995, 41).

2.3 Pooling Analogy

- Conjugate priors also have the property of being interpretable as **additional data**.
- "pooling" an useful analogy for thinking about Bayesian inference: updating prior beliefs with data is equivalent to pooling one set of data with another.

2.4 Inference for a Proportion

How does the "Bayesian mantra"

a posterior is proportional to the prior times the likelihood

apply in the case of a estimating a proportion?

- Data: *y* successes in *n* trials.
- Parameter: θ, probability of success.
- Prior: ignorance regarding a probability/proportion is easy to represent mathematically:

$$f(\theta) = \text{Uniform}(0, 1)$$

i.e., flat over the feasible region of the parameter space.

Digression on the Beta distribution:

1. a uniform distribution on (0,1) is a special case of a Beta distribution, specifically, a Beta (1,1) distribution.
2. More geneally, the Beta distribution is a pdf with support on the unit interval, and so is frequently used for modeling probabilities or proportions:

$$f(\theta; \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha),\Gamma(\beta)}\theta^{\alpha\text{-}1}(1 - \theta)^{\beta\text{-}1}$$

$0 < \theta < 1$, $\alpha > 0$, $\beta > 0$.

3. The parameters of a Beta prior for a probability/proportions can be interpreted as "prior number of observations": $\alpha + \beta$ - 2 = number of "prior observations", and $\alpha/(\alpha + \beta)$ is the prior mean on θ: e.g., Table 2.1 (Gelman et al., 1995, 41).

4. Thus a Uniform(0,1) = Beta(1,1) prior on θ is equivalent to zero previous observations.
5. Elementary calculus reveals that the mode of a Beta(α, β) density is at
$$\frac{\alpha - 1}{\alpha + \beta - 2}$$
6. The variance of a Beta distribution is
$$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

- Likelihood: if each of the *n* trials is independent, the likelihood is said to be *exchangeable*, and the likelihood is simply the joint probability of the data given θ. In this case, we obtain the binomial likelihood function
$$f(y|\theta) = \text{Binomial}(y|n,\theta) = \binom{n}{y}\theta^y(1-\theta)^{n-y}$$
The maximum likelihood estimate of θ is
$$\hat{\theta}_{\text{MLE}} = \frac{y}{n}$$

- Posterior with non-informative (uniform) prior: applying Bayes' Rule:
$$\begin{aligned} f(\theta|y) &\propto f(y|\theta)f(\theta) \\ &\propto \theta^y(1-\theta)^{n-y} \end{aligned}$$
since (1) $\binom{n}{y}$ is not a function of θ and (2) the prior is a constant over the support of the likelihood function.

How can we turn the expression for the posterior into a probability density function (i.e., make it integrate to 1)? We require a constant *c* such that
$$\int_0^1 f(\theta|y)\,d\theta = \int_0^1 c\,\theta^y(1-\theta)^{n-y}\,d\theta = 1$$
i.e.,
$$c = \frac{\Gamma(y+1+n-y+1)}{\Gamma(y+1)\,\Gamma(n-y+1)}$$
where $\Gamma()$ is the gamma function. A well-known result in statistics is that this yields a Beta distribution:
$$f(\theta|y) = c\,\theta^y(1-\theta)^{n-y} \equiv \text{Beta}(y+1, n-y+1),$$
or equivalently, the posterior for θ is the Beta distribution
$$\theta|y \sim \text{Beta}(y+1, n-y+1)$$

2.4.1 Other Types of Prior for θ

See Gelman et al. (1995, 35ff): in general, if we have

1. a **prior** of the form Beta(α, β)
2. a binomial **likelihood**, i.e., $f(y|\theta) \propto \theta^{y}(1 - \theta)^{n-y}$

then the **posterior** is

$$f(\theta|y) = \text{Beta}(\alpha + y, \beta + n - y)$$

2.4.2 Example: male-female birth rates

From Gelman et al. (1995, 39ff): What is the sex ratio among births with *placenta previa* ("an unusual condition of pregnancy in which the placenta is implanted very low in the uterus, obstructing a normal vaginal delivery"). An early study in Germany found that of a total of 980 births, 437 were female. So the MLE of $\theta = 437/980 \approx .446$.

Research Question: How much support does this study provide for the claim that the proportion of female births in the population of *placenta previa* births is less than 0.485, the proportion of female births in the general population? i.e., $\theta < .485$?

Using the assumption of a uniform prior (ignorance) as to the rate of female births θ, we obtain the posterior $f(\theta|y = 437, n = 980) = \text{Beta}(438, 544)$, which is proportional to the likelihood.

Other priors yield other answers; see Figures 1 and 2. Code to produce these graphs appears in A.1.

2.4.3 Simulation from Posterior

We can repeatedly sample from the posterior to obtain arbitrarily precise approximations to the quantities of interest: e.g.,

- median of $f(\theta|y)$, and quantiles of interest
- logit$(\theta) \equiv \ln[\theta/(1 - \theta)]$ (say, if we were relating θ to a set of covariates via a regression-type model, with normal errors; the logit transform makes the large-n normal approximation to the binomial work better).
- male-to-female sex ratio, $\phi = (1 - \theta)/\theta$, and its distribution (see picture).

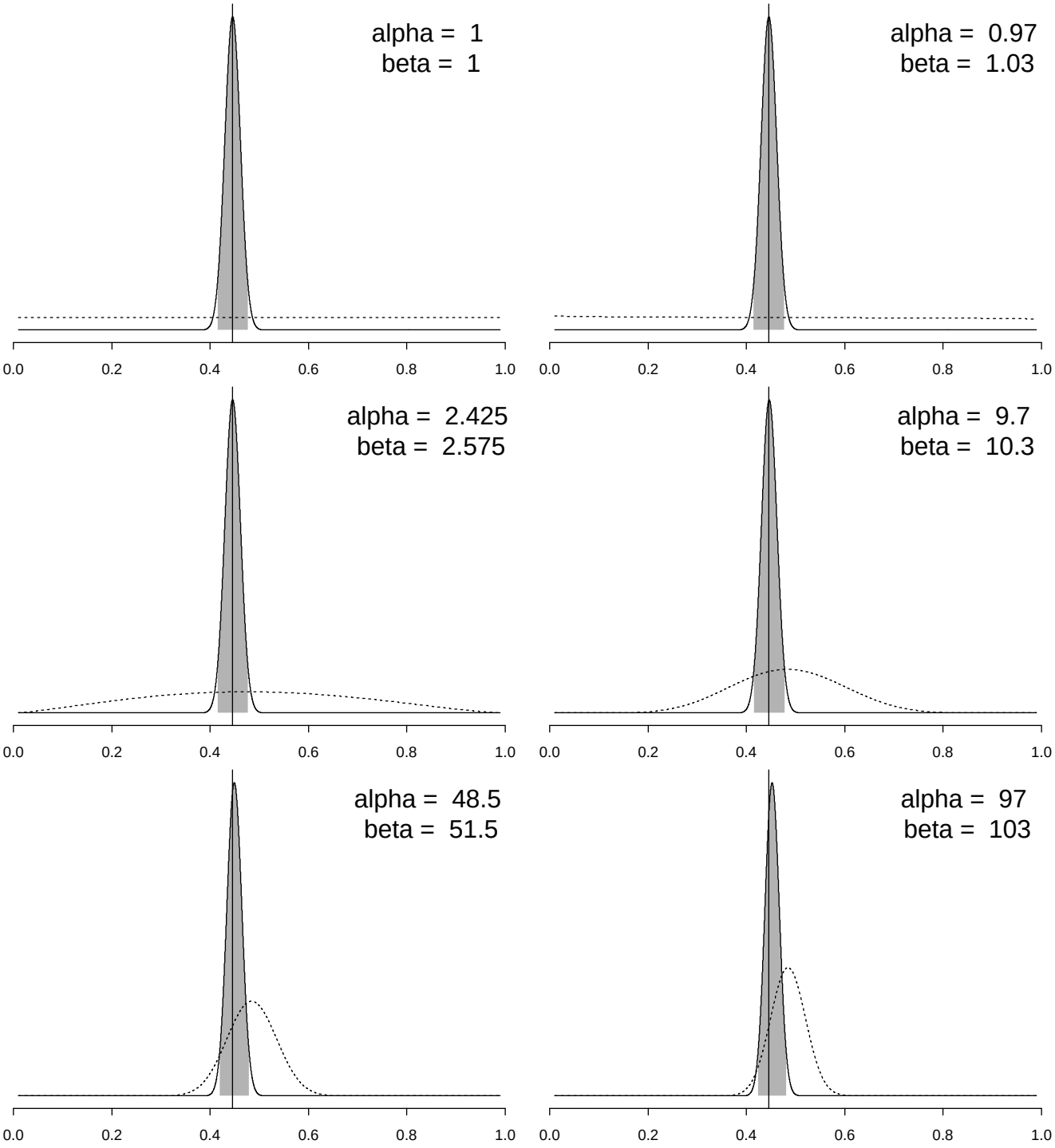

Figure 1: **Posteriors for θ, the probability of a female birth given placenta previa, under a variety of conjugate prior distributions.** This figure graphically replicates the contents of Table 2.1 in Gelman et al. (1995, 41). The dotted lines are the priors, parameterized by the indicated values of α and β, in this case chosen such that θ has a prior mean of .485, but with varying degrees of precision. The shaded areas are 95% posterior intervals for θ, and the solid line indicates the location of the MLE.

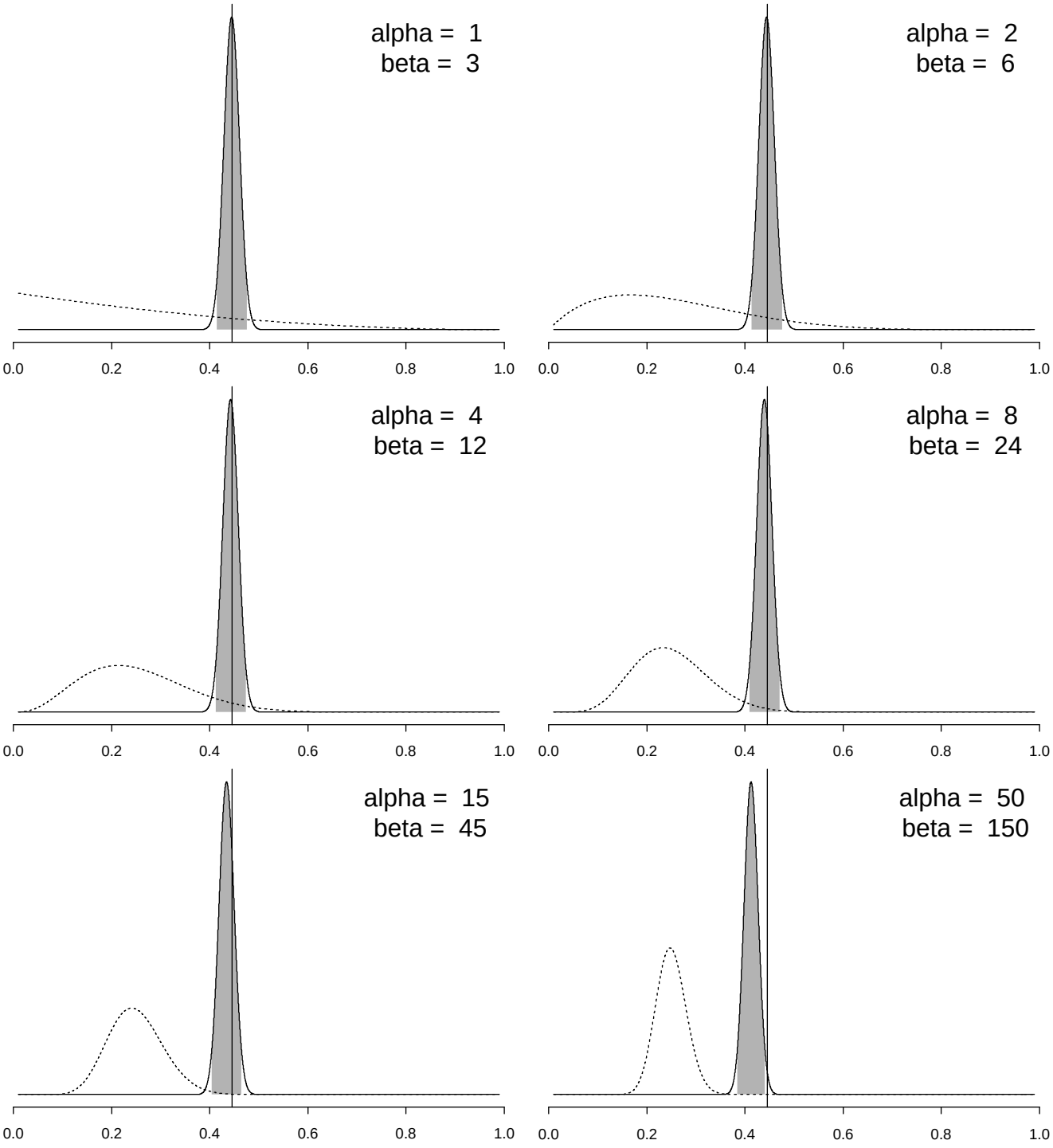

Figure 2: **Posteriors for θ, the probability of a female birth given placenta previa, under a variety of conjugate prior distributions.** The dotted lines are the priors, parameterized by the indicated values of α and β, in this case chosen such that θ has a prior mean of $\alpha/(\alpha+\beta) = .25$, but with varying degrees of precision. The shaded areas are 95% posterior intervals for θ, and the solid line indicates the location of the MLE.

2.5 Inference for the Mean

- **Data: $\mathbf{y} = (y_1, y_2, \ldots, y_n)'$.** That is,

$$y_i \sim N(\theta, \sigma^2), \forall\, i = 1, \ldots, n.$$

- Consider σ^2 fixed.
- **Prior** for θ: $\theta \sim N(\mu_0, \tau_0^2)$, or

$$f(\theta) \propto \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

- **Likelihood:**

$$\begin{aligned} f(\mathbf{y}|\theta, \sigma^2) &= \prod_{i=1}^{n} \phi(y_i|\theta, \sigma^2) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma^2}\right) \\ &= (2\pi\sigma^2)^{-n/2} \exp\left(\frac{-\sum_{i=1}^{n}(y_i - \theta)^2}{2\sigma^2}\right) \\ &\propto \exp\left(\frac{-\sum_{i=1}^{n}(y_i - \theta)^2}{2\sigma^2}\right) \end{aligned}$$

- **Posterior:** applying Bayes' Rule

$$\begin{aligned} f(\theta|\mathbf{y}) &\propto f(\theta) f(\mathbf{y}|\theta) \\ &= f(\theta) \prod_{i=1}^{n} f(y_i|\theta) \\ &\propto \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right) \prod_{i=1}^{n} \exp\left(\frac{-(y_i - \theta)^2}{2\sigma^2}\right) \\ &\propto \exp\left(-\frac{1}{2}\left[\frac{(\theta - \mu_0)^2}{\tau_0^2} + \frac{\sum_{i=1}^{n}(y_i - \theta)^2}{\sigma^2}\right]\right). \end{aligned}$$

i.e., we are again pooling the prior information about θ with the sample information about θ.

The posterior can be re-arranged to yield

$$\theta|y \sim N(\mu_n, \tau_n^2)$$

where

$$\mu_n = \frac{\frac{1}{\tau_0^2}\mu_0 + \frac{n}{\sigma^2}\bar{y}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}}, \qquad \text{i.e., a } \textbf{precision-weighted} \text{ average}$$

and

$$\frac{1}{\tau_n^2} \quad = \quad \frac{1}{\tau_0^2} \quad + \quad \frac{n}{\sigma^2}$$

$$\text{total precision} \;=\; \text{prior precision} \;+\; \text{data precision}$$

- **precision** = 1/**variance**. As variance gets large, precision gets smaller.
- **posterior precision = prior precision + data precision**
- since precision can't be negative, we always "learning" with normal prior and normal data.
- posterior mean is a **precision-weighted average** of the prior mean μ_0 and the sample mean $\bar{y} = \sum y/n$.
- prior ignorance about θ can be expressed by making the prior precision small, or equivalently, by making prior variance τ_0^2 large.
- Thus,

  as $\tau_0 \to \infty$ with n fixed, (i.e., the prior variance/precision increases/decreases)

  or

  as $n \to \infty$ with τ_0 fixed, (i.e., we get more and more data)

  then

  $$f(\theta|\mathbf{y}) \approx N(\bar{y}, \sigma^2/n).$$

  i.e., the data dominate the posterior.
- setting τ_0^2 equal to the (known) variance σ^2 is equivalent to giving the prior distribution the same weight as one extra observation with the value μ_0.

2.6 Inference for the Mean and Variance, Normal data

From Gelman et al. (1995, 71ff):

- **Data: y** $= (y_1, \ldots, y_n)'$. $y_i \sim N(\mu, \sigma^2), \forall\, i$.
- n.b., **two** parameters to carry through the analysis; μ and σ^2; require **priors** for both, i.e., $f(\mu, \sigma^2)$
- conjugacy means that a convenient form for the prior is the *product form*

$$f(\mu, \sigma^2) = f(\mu|\sigma^2) f(\sigma^2)$$

where

$$\begin{aligned} \mu|\sigma^2 &\sim N(\mu_0, \sigma_0^2/\kappa_0) \\ \sigma^2 &\sim \text{Inv-}\chi^2(\nu_0, \sigma_0^2) \end{aligned}$$

- the joint prior density $f(\mu, \sigma^2)$ is given by the convolution of these two densities is known as a "Normal-inverse-chi-squared" density, with four parameters:
  1. the prior location of μ, μ_0
  2. a variance parameter, tapping our uncertainty in the prior location of μ, σ_0^2/κ_0
  3. the degrees of freedom of σ^2, tapped via ν_0
  4. the scale of σ^2, tapped via σ_0^2
- normal data, and so the **likelihood** is

$$\begin{aligned} f(\mathbf{y}|\mu, \sigma^2) &= \prod_{i=1}^{n} \phi(y_i|\mu, \sigma^2) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y_i - \mu)^2}{2\sigma^2}\right) \\ &= (2\pi\sigma^2)^{-n/2} \exp\left(\frac{-\sum_{i=1}^{n}(y_i - \mu)^2}{2\sigma^2}\right) \end{aligned}$$

- using Bayes' Rule, we multiply the joint prior and a Normal likelihood to obtain the joint posterior that is also a "Normal-inverse-chi-squared"

density, with four parameters:

$$\begin{aligned}
\mu_n &= \frac{\kappa_0}{\kappa_0 + n}\mu_0 + \frac{n}{\kappa_0 + n}\bar{y} \\
\kappa_n &= \kappa_0 + n \\
\nu_n &= \nu_0 + n \\
\nu_n\sigma_n^2 &= \nu_0\sigma_0^2 + (n - 1)s^2 + \frac{\kappa_0}{\kappa_0 + n}(\bar{y} - \mu_0)^2
\end{aligned}$$

where

1. μ_n is a weighted average of the prior mean and the sample mean, with weights determined by the relative precisions,
2. ν_n is the prior degrees of freedom plus the sample size
3. $\nu_n\sigma_n^2$ is the posterior sum of squares, combining the prior sum of squares, the sample sum of squares, and the additional uncertainty tapped by the difference between the sample mean and the prior mean.

- the **conditional posterior** $f(\mu|\sigma^2, \mathbf{y})$:

$$\begin{aligned}
\mu|\sigma^2, \mathbf{y} &\sim N(\mu_n, \sigma^2/\kappa_n) \\
&= N\left(\frac{\frac{\kappa_0}{\sigma^2}\mu_0 + \frac{n}{\sigma^2}\bar{y}}{\frac{\kappa_0}{\sigma^2} + \frac{n}{\sigma^2}}, \frac{1}{\frac{\kappa_0}{\sigma^2} + \frac{n}{\sigma^2}}\right)
\end{aligned}$$

- the **marginal posterior** $f(\sigma^2|\mathbf{y})$ is Inverse-χ^2:

$$\sigma^2|\mathbf{y} \sim \text{Inverse-}\chi^2(\nu_n, \sigma_n^2)$$

- **method of composition:** sampling from the two distributions sequentially can be used to build up an arbitrarily precise approximation to the joint distribution. See Figure 3.2 in Gelman et al. (1995, 75). i.e.,

1. sample σ^{2^*} from Inv-$\chi^2(\nu_n, \sigma_n^2)$
2. sample μ^* from $N(\mu_n, \sigma^{2^*}/\kappa_n)$

and store results over many repetitions of this process.

- in this case, we can integrate σ^2 out of the joint posterior to obtain a marginal posterior density for μ that is a t distribution, with location parameter μ_n, scale parameter σ_n^2/κ_n, and degrees of freedom ν_n.

2.7 Bayesian Inference for a Regression, Normal data

- **Data:** $\mathbf{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$.
- **Prior:** $\pi(\boldsymbol{\beta}, \sigma^2) = \pi(\boldsymbol{\beta}|\sigma^2)\pi(\sigma^2)$:

$$\pi(\boldsymbol{\beta}|\sigma^2) \equiv N(\bar{\boldsymbol{\beta}}_0, \sigma^2\mathbf{B}_0) \tag{2}$$

$$\sigma^2 \sim \text{Inverse-}\chi^2(v_0, S_0) \tag{3}$$

- **Non-informative prior:** $\pi(\boldsymbol{\beta}, \sigma^2) \propto 1/\sigma^2$. i.e., a uniform prior on $(\boldsymbol{\beta}, \ln \sigma^2)$.
- **Likelihood:**

$$f(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) \propto \left(\sigma^2\right)^{-n/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mathbf{x}_i\boldsymbol{\beta})^2\right\} \tag{4}$$

- **Marginal Posterior for $\boldsymbol{\beta}$:**

$$\pi(\boldsymbol{\beta}|\sigma^2, \mathbf{y}, \mathbf{X}) \equiv N(\bar{\boldsymbol{\beta}}, \sigma^2\mathbf{B})$$

where

$$\begin{aligned}
\bar{\boldsymbol{\beta}} &= (\text{prior precision} + \text{data precision})^{-1} \\
&\quad \times (\text{prior precision} \times \bar{\boldsymbol{\beta}}_0 + \text{data precision} \times \hat{\boldsymbol{\beta}}_{\text{MLE}}) \\
&= \left(\mathbf{B}_0^{-1} + \mathbf{X}'\mathbf{X}\right)^{-1}\left(\mathbf{B}_0^{-1}\bar{\boldsymbol{\beta}}_0 + \mathbf{X}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}\right) \\
&= \left(\mathbf{B}_0^{-1} + \mathbf{X}'\mathbf{X}\right)^{-1}\left(\mathbf{B}_0^{-1}\bar{\boldsymbol{\beta}}_0 + \mathbf{X}'\mathbf{y}\right),
\end{aligned}$$

and

$$\mathbf{B} = (\text{prior precision} + \text{data precision})^{-1} = \left(\mathbf{B}_0^{-1} + \mathbf{X}'\mathbf{X}\right)^{-1}$$

i.e., the posterior mean is a **precision-matrix-weighted average** of the prior $\bar{\boldsymbol{\beta}}_0$ and the estimate we would get from the data ($\boldsymbol{\beta}_{\text{MLE}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$), where the matrix weights are the prior precision $\mathbf{B}_0^{-1}$ and the data precision $\mathbf{X}'\mathbf{X}$.

n.b., **precision** is the inverse of **variance**.

- **Marginal Posterior for σ^2:**

$$\pi(\sigma^2|\mathbf{XY}) \equiv \text{Inverse-}\chi^2(v_0 + n, S_0 + S)$$

where $S = \mathbf{y}'\mathbf{M}\mathbf{y}$ with $\mathbf{M} = \mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$.

- **Correspondence with familiar least squares results:** note that as the prior mean on $\boldsymbol{\beta}$ goes to $\mathbf{0}$ and $\mathbf{B}$ "gets large", we obtain the MLE. We also obtain the MLE of σ^2 when $\nu_0 = 0$ and $S_0 = 0$.

  In fact, the MLE results exactly with the non-informative prior.

- Posterior is exactly that we would obtain if we pooled one set of data with another. Useful to remind us that we are **averaging** prior information with data.

- This particular form for the prior has the property that it induces a Normal marginal posterior for $\boldsymbol{\beta}$. The posterior has two properties worth noting:

  1. the posterior precision/variance is strictly greater/smaller than the prior precision/variance.
  2. the posterior is unimodal --- doesn't allow for the possibility that after getting data at odds with a prior, we could well wind up "confused" or "uncertain". See For an alternative prior for regression models that allows this possibility, see Leamer (1978, 79).

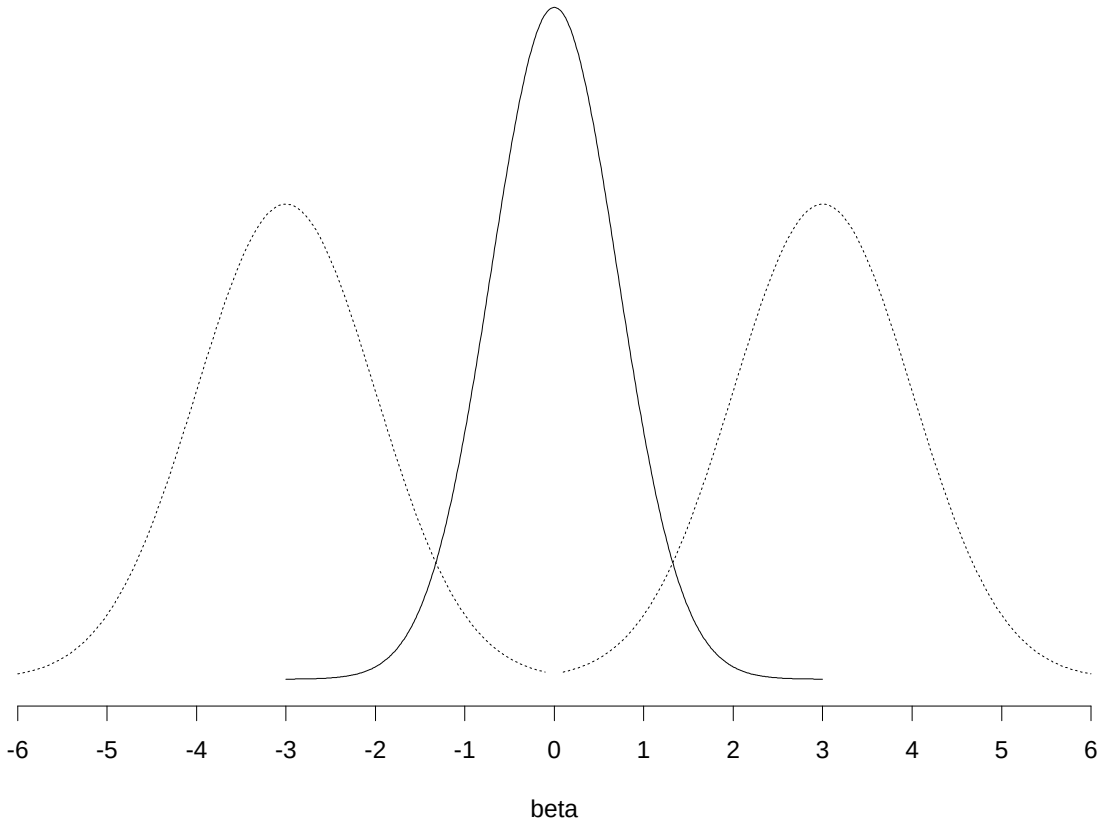

Figure 3: **Bayesian Updating for a Mean, Normal Prior and Data.** The dotted lines indicate the (symmetric) prior and data, a N(-3,1) density and a N(3,1) density. The resulting posterior is a compromise between the two, $N(0, 1/2)$.

3 Simulation Methods

> *Bayesianism has obviously come a long way. It used to be that you could tell a Bayesian by his tendency to hold meetings in isolated parts of Spain and his obsession with coherence, self-interrogation and other manifestations of paranoia. Things have changed...*
>
> Clifford (1993, 53)

Claim:

"everyone" likes likelihood functions!

where

$$\text{"everyone"} = \{\text{most frequentists}, \text{Bayesians}\}$$

- for Bayesians, likelihoods are how we use the data to move from prior to posterior.
- with vague or diffuse priors, likelihoods will dominate the posterior
- but sometimes likelihoods are unwieldy, difficult, or even impossible to compute, let alone maximize with respect to parameters.

Maximizing likelihoods is often easier said than done:

- missing data
- the likelihood is complex, having lots of parameters or a computationally intractable functional form (e.g., multinomial probit model, time series models for qualitative data).
- maximization is slow or difficult
- we know that no unique maximum exists, or that maxima are located on the edge of the parameter space (e.g., finite mixture models),
- even when the MLE can be calculated, the asymptotic properties of MLEs may not "kick in" in samples that are considered "large" --- direct inspection of the log-likelihood or log-posterior may be better than relying on the MLE point estimate and the estimated asymptotic standard error (e.g., correlation coefficients; bimodal distributions on parameters).

- in explicitly Bayesian models, the joint posterior for all the parameters is sometimes very difficult to compute.

Pedigree of MCMC methods:

EM (Dempster, Laird and Rubin, 1977)
↓
Data Augmentation (Tanner and Wong, 1987)
↓
Markov Chain Monte Carlo { Gibbs Sampling / Metropolis-Hastings }

Underlying ideas have been around for a long time:

- relaxation methods in solving systems of linear equations.
- Monte Carlo sampling
- averaging over nuisance parameters

MCMC methods comes to the social sciences via:

- physics and chemistry (e.g., Metropolis et al., 1953)
- image reconstruction (e.g., Geman and Geman, 1984)
- statistical mainstream (e.g., Gelfand and Smith, 1990)

3.1 Imputation/Augmentation

> ...rather than performing a complicated maximization or simulation, one augments the observed data with "stuff" (latent data) which simplifies the calculation and subsequently performs a series of simple maximizations or simulations. This "stuff" can be "missing" data or parameter values. The principle of data augmentation can then be stated as follows: *Augment the observed data* $\mathbf{Y}$ *with latent data* $\mathbf{Z}$ *so that the augmented posterior distribution* $p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{Z})$ *is "simple". Make use of this simplicity in maximizing/marginalizing/calculating/sampling the observed posterior* $p(\boldsymbol{\theta}|\mathbf{Y})$
>
> (Tanner, 1996, 38, emphasis added).

Critical notion is the *posterior identity*:

$$p(\boldsymbol{\theta}|\mathbf{Y}) = \int_{\mathcal{Z}} p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{Z})\, p(\mathbf{Z}|\mathbf{Y})\, d\mathbf{Z} \tag{5}$$

where $\mathbf{Y}$ is observed data, $\mathbf{Z}$ is latent data, and the integration is over $\mathcal{Z}$, the sample space of $\mathbf{Z}$.

Without too much violence to the ideas/math, we can replace the term "posterior" with "likelihood"

The various techniques to be discussed vary in how they exploit this identity:

- EM uses the expected sufficient statistics of $\mathbf{Z}|\mathbf{Y}$ to form an imputation (typically just the mode or mean, and variance);
- MCEM (Monte Carlo EM) or MCMC (Markov chain Monte Carlo) uses **Monte Carlo integration** (averaging over many random draws from $p(\mathbf{Z}|\mathbf{Y})$.

3.2 Monte Carlo and other simulation methods

Consider evaluating the expression

$$J(y) = \int f(y|x)\, g(x)\, dx = E[f(y|x)],$$

i.e., the expected value of some function of y, conditional on x. Note that $f(y)$ could be an identity, in which case $J(y)$ is just the conditional expectation $E(y|x)$.

If $g(x)$ is a probability density from which we can generate random samples, then $J(y)$ can be approximated by **Monte Carlo integration**: i.e.,

$$\hat{J}(y) = \frac{1}{n} \sum_{i=1}^{n} f(y|x_i),$$

where $x_1, \ldots, x_n \overset{\text{iid}}{\sim} g(x)$ are *samples* from the density of x. Moreover,

$$\lim_{n \to \infty} \hat{J}(y) = J(y) \quad \text{(Geweke, 1989)}$$

3.2.1 Computing marginal posterior densities

A Bayesian analysis often results in multi-dimensional joint posterior densities, when substantive interest may focus on a set of parameters.

A common example: the Normal regression model yields the joint posterior (Gelman et al., 1995, 236-7):

$$p(\boldsymbol{\beta}, \sigma^2|y) \sim \text{ Normal-Inverse-}\chi^2,$$

but we want

$$p(\boldsymbol{\beta}|\sigma^2, y) = \int p(\boldsymbol{\beta}, \sigma^2|y) p(\sigma^2|y) d\sigma^2.$$

1. Sample $\sigma^{2(t)}|y$ from its Inverse-χ^2 distribution:

$$\sigma^{2(t)}|y \sim \text{ Inv - } \chi^2(n \text{ - } k, s^2)$$

2. Sample $\boldsymbol{\beta}^{(t)}|\sigma^{2(t)}, y$ from its (conditional) multivariate Normal distribution:

$$\boldsymbol{\beta}^{(t)} \sim N(\hat{\boldsymbol{\beta}}, \sigma^{2(t)} V_{\boldsymbol{\beta}})$$

3. Repeat steps 1 and 2 many times: i.e., $t = 1, 2, \ldots$

3.2.2 Functions of Parameters

Interest will sometimes focus on some function of the model parameters, generically $h(\boldsymbol{\theta})$. For example:

> If each extra dollar of incumbent expenditures yields a β_1 increase in incumbent vote share, how many dollars must a challenger spend so as to offset the increase?

$$\beta_1 - m\beta_2 = 0 \iff m = -\frac{\beta_1}{\beta_2}$$

How to conduct inference on m (a ratio of two Normals)?

The two steps in the previous section yield $\boldsymbol{\beta}^{(t)}|\sigma^{2(t)}, y$. Then calculate

$$m^{(t)} = -\frac{\beta_1^{(t)}}{\beta_2^{(t)}}.$$

The sampled $m^{(t)}$ can then be inspected and summarized for inference purposes (mean, median, standard deviation, quantiles, proportion lying above/below zero, density plots, histograms, etc).

Remark: this procedure is sometimes referred to as a *parametric bootstrap*.

3.2.3 Techniques for Sampling

Two techniques for sampling, very important in MCMC methods:

- **importance sampling**
- **rejection sampling**

The development of quite general algorithms for *adaptive rejection sampling* (Gilks, 1996) has helped spur the developement of a general-purpose software for Markov chain Monte Carlo (see section 9).

4 EM

A technique for obtaining MLEs "more simply", originally pioneered in the context of missing data (e.g., Little and Rubin, 1987) Define

$$Q(\theta^{(t)}, \theta) = \int \ln p(\theta|Y_{\text{obs}}, Y_{\text{mis}}) f(Y_{\text{mis}}|Y_{\text{obs}}, \theta^{(t)}) dY_{\text{mis}},$$

where

- $\ln p(\theta|Y_{\text{obs}}, Y_{\text{mis}})$ is the log-likelihood or log-posterior density for the complete data,
- $f(Y_{\text{mis}}|Y_{\text{obs}}, \theta^{(t)})$ is the predictive density of Y_{mis} (missing "stuff"), given the observed data and the current value of the parameters (θ, ψ).
- the integration is over the sample space of Y_{mis}.
- **E step:** use the current iteration's parameter estimates to generate imputations for the missing "stuff"; this amounts to evaluating this Q function: i.e., computing the *expectation* of the complete-data log-likelihood, averaging over uncertainty as to the missing data.
- **M step:** *maximize* the Q function with respect to θ, yielding updated parameter estimates $\theta^{(t+1)}$, such that

  $$Q(\theta^{(t+1)}, \theta^{(t)}) \geq Q(\theta^{(t)}, \theta^{(t)})$$

  i.e., the log of the incomplete-data likelihood or mode of the log-posterior increases.

  With the new parameter values, the algorithm returns to the *E* step, iterating until convergence in the parameters and/or the likelihood.

Remark: EM does not simply treat the missing data as (additional) parameters in the complete-data log-likelihood. Such an approach yields estimates that do not share MLE's optimal asymptotic properties, save for the trivial case of where the proportion of data that is missing data asymptotically declines to zero Little and Rubin (1983).

Convergence Properties: See Dempster, Laird and Rubin (1977) and Wu (1983). Theorem 7.1 of Little and Rubin (1987):

Every (generalized) EM algorithm *increases* $l(\theta|Y_{obs})$ at each iteration, i.e.,

$$\ln \mathcal{L}(\theta^{(i+1)}|Y_{obs}) \geq \ln \mathcal{L}(\theta^{(i)}|Y_{obs})$$

with equality if and only if

$$Q(\theta^{(i+1)}|\theta^{(i)}) = Q(\theta^{(i)}|\theta^{(i)}).$$

This theorem can be used to show that $l(\theta|Y_{obs})$ is nondecreasing on each iteration of a GEM algorithm, and is strictly increasing on any iteration such that Q increases. Further, a MLE of θ is a fixed point of a GEM algorithm.

Remark: A GEM algorithm simply increases the Q function at each iteration, while an EM algorithm maximizes the Q function with respect to θ.

How does *EM* exploit the posterior identity?:

$$\begin{aligned} f(\mathbf{Y}|\theta) &= f(Y_{obs}, Y_{mis}|\theta) \\ &= f(Y_{obs}|\theta)\, f(Y_{mis}|Y_{obs}, \theta) \end{aligned}$$

The log-likelihood implied by this factorization is

$$\begin{aligned} \mathcal{L}(\theta|\mathbf{Y}) &= \mathcal{L}(\theta|Y_{obs}, Y_{mis}) \\ &= \mathcal{L}(\theta|Y_{obs}) + \ln f(Y_{mis}|Y_{obs}, \theta) \end{aligned}$$

In maximizing this log-likelihood, we'd take the **expectations** over the missing data, conditional on Y_{obs} and the current estimate of θ (Little and Rubin, 1987, 134).

4.1 Example: Probit model for binary data

Model: Consider a probit model for a binary outcome, $y_i \in \{0, 1\}, i = 1, \ldots, n$. We relate the observed binary outcome to covariates via latent regression function

$$y_i^* = \mathbf{x}_i\boldsymbol{\beta} + \varepsilon_i, \tag{6}$$

where

- $\mathbf{x}_i$ is a row vector of observations on k independent variables,
- $\boldsymbol{\beta}$ is a column vector of parameters to be estimated,
- $y_i^* \in \mathbb{R}$ is a latent dependent variable, observed only in terms of its sign, i.e.,

$$y_i = \begin{cases} 0, & \text{if } y_i^* < 0 \\ 1, & \text{if } y_i^* \geq 0 \end{cases}$$

- ε_i $N(0, 1), \forall\, i = 1, \ldots, n$.
- Equation (6) is a standard-looking regression save for the complication that we observe only the sign of the dependent variable, y^*.
- Treat the $\mathbf{y}^*$ as missing data.
- *E* step: Use the current estimate of $\boldsymbol{\beta}$ and other model assumptions to make an *imputation* for each y_i^*
- *M* step: Conditional on the $\mathbf{y}_i^*$ choose $\boldsymbol{\beta}$ so as to maximize the complete-data log-likelihood, updating our estimate of $\boldsymbol{\beta}$.

Details:

- The *Q* function for the probit model is

$$Q(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}) = \int_{\mathbf{y}^*} \ln p(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}, \mathbf{y}^*)\, p(\mathbf{y}^*|\boldsymbol{\beta}^{(t)}, \mathbf{X}, \mathbf{y})\, d\mathbf{y}^*$$

  or the expected value of the complete-data log-likelihood, where the expectation is with respect to the latent dependent variable $\mathbf{y}^*$, conditional on the current estimate of $\boldsymbol{\beta}$, $\boldsymbol{\beta}^{(t)}$ and the observed data $\mathbf{X}$ and $\mathbf{y}$.

- Complete-data log-likelihood:

$$\ln p(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}, \mathbf{y}^*) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{n}\left[\left(y_i^* - \mathbf{x}_i\boldsymbol{\beta}\right)^2\right]$$

given that $\sigma^2 = 1$, by assumption.

- Substituting into the Q function

$$\begin{aligned} Q(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}) &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{n}\left[\left(E[(y_i^* - \mathbf{x}_i\boldsymbol{\beta})|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}]\right)^2\right] \\ &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{n}\left[V[(y_i^* - \mathbf{x}_i\boldsymbol{\beta})|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}] + E[(y_i^* - \mathbf{x}_i\boldsymbol{\beta})|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}]^2\right], \\ &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{n}\left[V(y_i^*|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}) + [E(y_i^*|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}) - \mathbf{x}_i\boldsymbol{\beta}]^2\right]. \end{aligned}$$

- The updated estimate of $\boldsymbol{\beta}$, $\boldsymbol{\beta}^{(t+1)}$, is given by maximizing $Q(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)})$ with respect to $\boldsymbol{\beta}$. The variance and expectation terms do not involve $\boldsymbol{\beta}$ (just $\boldsymbol{\beta}^{(t)}$), and so

$$\begin{aligned} \boldsymbol{\beta}^{(t+1)} &= \arg\min_{\boldsymbol{\beta}} \frac{1}{2}\sum_{i=1}^{n}\left[E(y_i^*|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}) - \mathbf{x}_i\boldsymbol{\beta}\right]^2 \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'E(\mathbf{y}^*|\mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(t)}). \end{aligned}$$

i.e., the updated estimate of $\boldsymbol{\beta}$ is obtained by **running a least-squares regression** of the imputed values for $\mathbf{y}^*$ on the covariates $\mathbf{X}$.

- The imputation for $\mathbf{y}^*$ is

$$E(y_i^*|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}) \equiv y_i^{*(t)} = E[(\mathbf{x}_i\boldsymbol{\beta} + \varepsilon_i)|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}] = \mathbf{x}_i\boldsymbol{\beta}^{(t)} + M_i,$$

where

$$M_i = E_{\boldsymbol{\beta}^{(t)}}(\varepsilon_i|y_i, \mathbf{x}_i, \boldsymbol{\beta}^{(t)}) = \begin{cases} -\phi_i/\Phi_i & \text{if } y_i = 0, \\ \phi_i/(1 - \Phi_i) & \text{if } y_i = 1, \end{cases}$$

and where $\phi_i = \phi(-\mathbf{x}_i\boldsymbol{\beta}^{(t)})$ is the Normal probability density function, and $\Phi_i = \Phi(-\mathbf{x}_i\boldsymbol{\beta}^{(t)})$ is the Normal cumulative distribution function, evaluated at $-\mathbf{x}_i\boldsymbol{\beta}^{(t)}$ (e.g., Johnson, Kotz and Balakrishnan, 1994, 156).

- Convergence of the *EM* algorithm can be monitored by noting the difference between $Q(\boldsymbol{\beta}^{(t+1)}, \boldsymbol{\beta}^{(t)})$ and $Q(\boldsymbol{\beta}^{(t)}, \boldsymbol{\beta}^{(t)})$.
- **Inference:** *EM* does not generate standard errors as a mater of course. The standard errors we obtain from the least squares regression of $\mathbf{y}^*$ on $\mathbf{X}$ are not the standard errors of the MLEs (the least squares regression is heteroscedastic).

  i.e., still require second derivatives of the likelihood function.

Data: I implemented this algorithm for a probit model, using a random subset of 3,000 observations from Nagler's (1994) data on voter turnout, from the 1984 Current Population Survey; predictor variables are education, age, the number of days registraton closes before the election, whether or not a gubernatorial election took place in the respondent's state, and whether the respondent lives in the South. Starting values were drawn from an OLS regression of the observed binary dependent variable on the covariates, and after 30 iterations of the *EM* algorithm the log-likelihood was increasing by steps of less than 10^{-9}.

Figure 4 shows the iterative history of the *EM* algorithm for the log-likelihood, two parameters, and the estimated value of the latent dependent variable for the 1,000th observation. The algorithm converges quite quickly in this case, and after a few iterations has done most of its work, moving away from the OLS starting values towards the maximum likelihood estimates.

Computation: See section A.2.

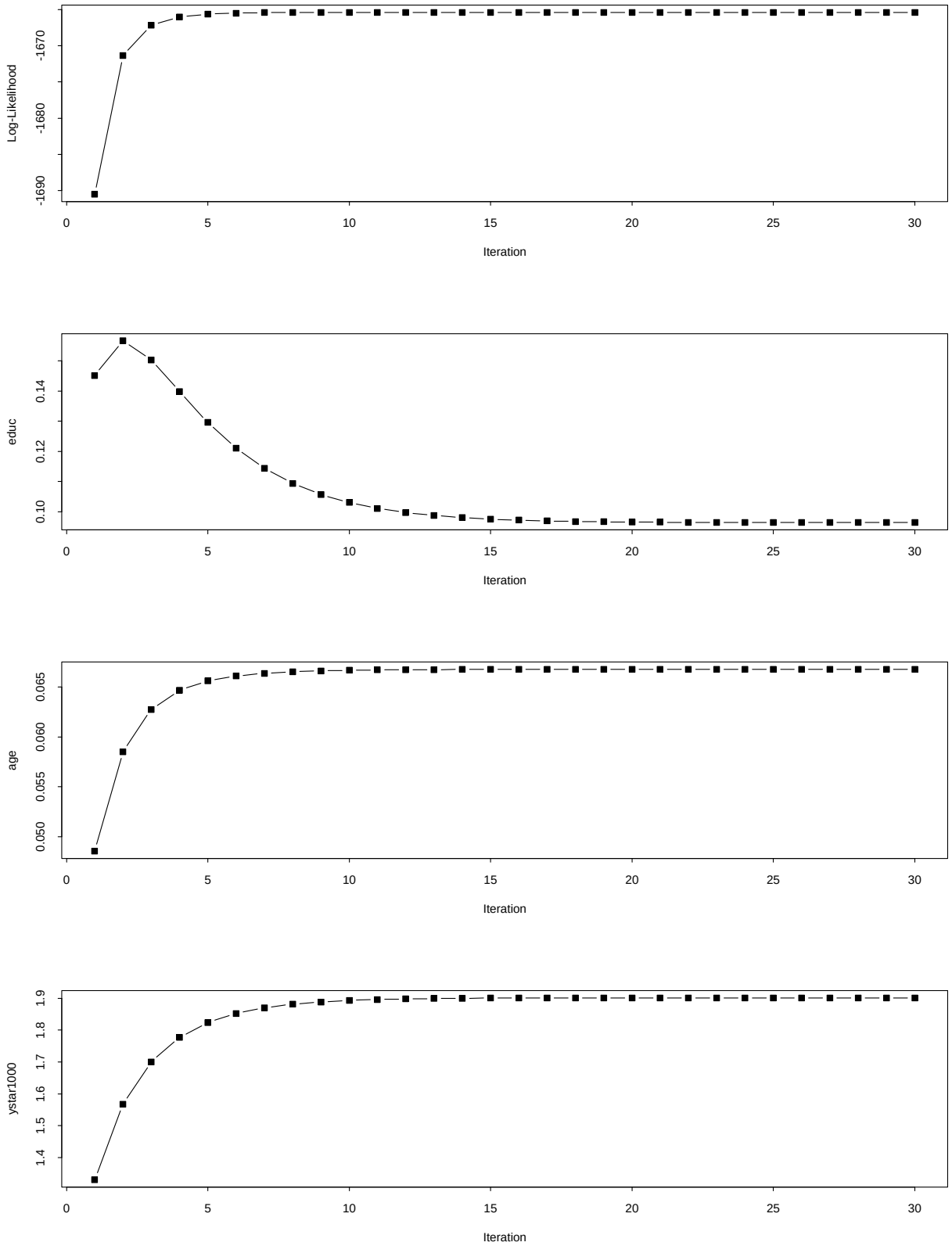

Figure 4: **Iterative History of EM algorithm, Probit Model of Voter Turnout.**

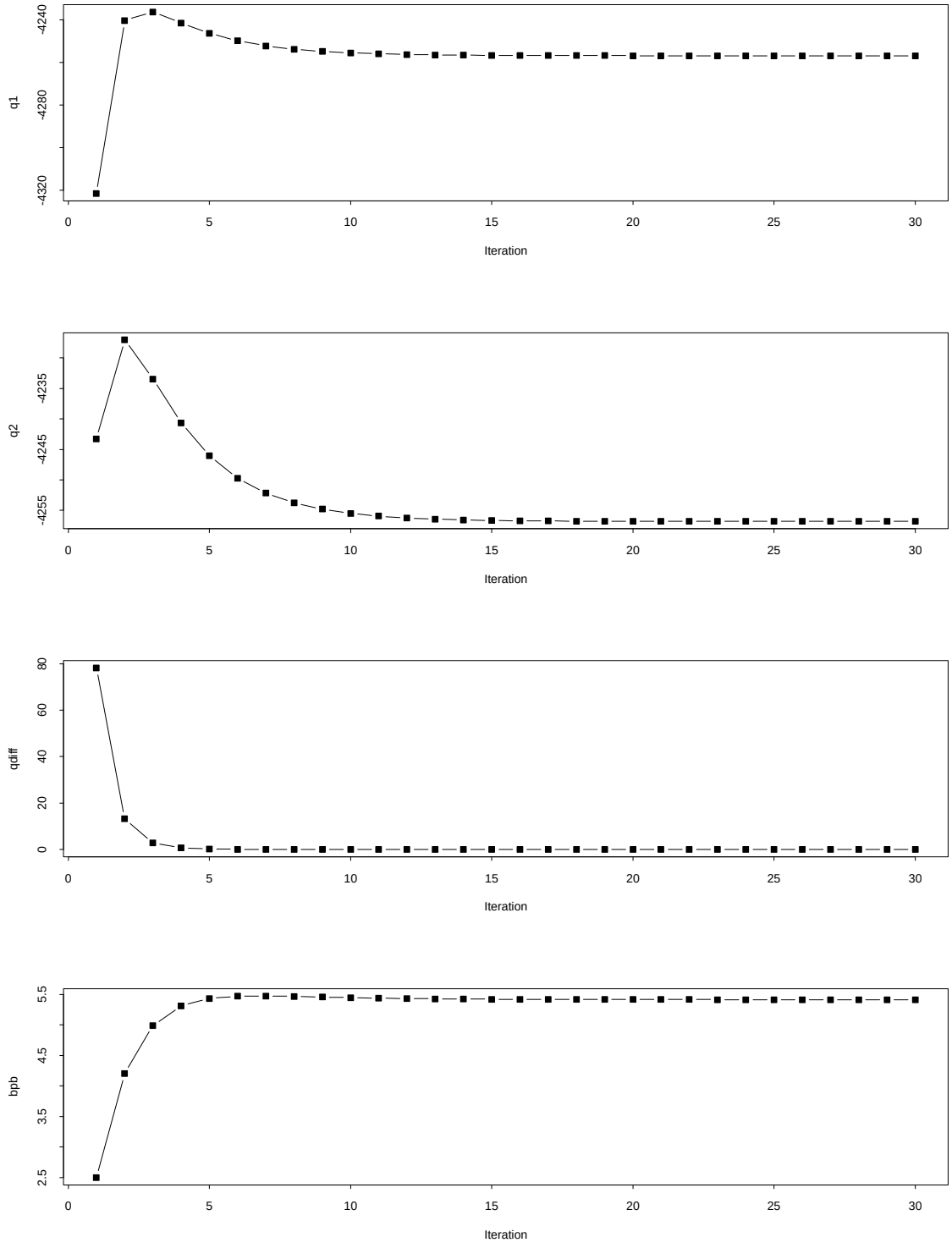

Figure 5: **Convergence Diagnostics, EM Algorithm for probit model.**

4.2 Example: Linear Regression with AR(1) disturbances

Model:

$$\begin{aligned} y_t &= \mathbf{X}_t\boldsymbol{\beta} + u_t, \\ u_t &= \rho\, u_{t\text{-}1} + e_t,\ \ e_t \sim N(0, \sigma^2), \forall\, t = 2, \ldots, T. \end{aligned}$$

Given the normality of e_t, the log-likelihood is

$$\ln \mathcal{L}(\boldsymbol{\beta}, \rho, \sigma^2 \,|\, \mathbf{y}, \mathbf{X}) = \frac{\text{-}T}{2} \ln(2\pi) \text{-} \frac{T}{2} \ln \sigma + \frac{1}{2} \ln(1 \text{-} \rho^2) \text{-} \frac{\mathbf{u}^{*\prime}\mathbf{u}^*}{2\sigma^2}, \tag{7}$$

where

$$\mathbf{u}^* = \mathbf{y}^* \text{-} \mathbf{X}^*\boldsymbol{\beta},$$

and

$$\mathbf{y}^* = \begin{bmatrix} \sqrt{1 \text{-} \rho^2}\, y_1 \\ y_2 \text{-} \rho\, y_1 \\ \vdots \\ y_T \text{-} \rho\, y_{T\text{-}1} \end{bmatrix}, \quad \mathbf{X}^* = \begin{bmatrix} \sqrt{1 \text{-} \rho^2}\, \mathbf{x}_1 \\ \mathbf{x}_2 \text{-} \rho\, \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \text{-} \rho\, \mathbf{x}_{T\text{-}1} \end{bmatrix}$$

are the familiar Prais-Winsten transformations for the first observation Prais and Winsten (1954).

- consider ρ as "missing stuff" (here, a parameter), or the white-noise disturbances $\mathbf{u}^*$ as missing "data" stuff --- unobserved by the analyst, at least initially.
- log-likelihood in (7) can't be calculated, let alone maximized with $\mathbf{u}$ and ρ "missing".

Posterior identity:

$$p(\boldsymbol{\beta}, \sigma^2 \,|\, \mathbf{y}, \mathbf{X}) = \int_{\text{-}1}^{1} p(\boldsymbol{\beta}, \sigma^2 \,|\, \mathbf{y}, \mathbf{X}, \rho)\, p(\rho \,|\, \boldsymbol{\beta}, \sigma^2, \mathbf{y}, \mathbf{X})\, d\rho, \tag{8}$$

where the limits of integration over ρ follow from the assumption of stationarity.

Applying *EM*:

- *E* **step:** make an imputation for ρ conditional on the data and the current estimate of $\boldsymbol{\beta}$ (σ^2 isn't necessary for this step).

- *M* **step:** find estimates of $\boldsymbol{\beta}$ and σ^2 that maximize the log-likelihood in (7) conditional on the imputation for ρ.

Slow motion:

- At the end of iteration i, estimates of $\boldsymbol{\beta}$ and σ^2 are $\boldsymbol{\beta}^{(i)}$ and $\sigma^{2\,(i)}$, respectively.
- Given $u_t = \rho u_{t\text{-}1} + e_t$ and $e_t \sim N(0, \sigma^2), \forall\, t$, the log-predictive density (or log-likelihood) for $\rho^{(i+1)}$ is

$$p(\rho^{(i+1)} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(i)}, \sigma^{2\,(i)}) = -\frac{T\text{ - }1}{2}\left(\ln(2\pi) + \ln \sigma^{2\,(i)}\right) - \sum_{t=2}^{T} \frac{(u_t^{(i)} - \rho^{(i+1)} u_{t\text{-}1}^{(i)})^2}{2\sigma^{2\,(i)}},$$

  where $u_t^{(i)} = y_t - \mathbf{x}_t \boldsymbol{\beta}^{(i)}$.
- The value of $\rho^{(i+1)}$ that maximizes this log-predictive density is

$$\rho^{(i+1)} = \frac{\sum_{t=2}^{T} u_t^{(i)} u_{t\text{-}1}^{(i)}}{\sum_{t=2}^{T} \left(u_{t\text{-}1}^{(i)}\right)^2}, \tag{9}$$

  This is just the coefficient from the regression of $u_t^{(i)}$ on $u_{t\text{-}1}^{(i)}$ (without a constant).
- **M step.** Find $\boldsymbol{\beta}$ and σ^2 that maximize $\ln \mathcal{L}(\boldsymbol{\beta}, \sigma^2 \mid \mathbf{y}, \mathbf{X}, \rho^{(i+1)})$:
  1. $\boldsymbol{\beta}^{(i+1)} \leftarrow$ regression of $\mathbf{y}^{*\,(i+1)}$ on $\mathbf{X}^{*\,(i+1)}$, where $\rho^{(i+1)}$ is used in forming the transformed variables;
  2. $\sigma^{2\,(i+1)} = (\mathbf{e}_{(i+1)}^{*\prime} \mathbf{e}_{(i+1)}^{*})/T$, where $\mathbf{e}_{(i+1)}^{*} = \mathbf{y}^{*\,(i+1)} - \mathbf{X}^{*\,(i+1)} \boldsymbol{\beta}^{(i+1)}$.
- Iterate this algorithm until convergence in the log-likelihood or the parameters; *a la* Cochrane and Orcutt (1949).

- **Data:** monthly approval ratings for Reagan ($T = 96$).
- **Covariates:** inflation rate, unemployment level, change in the S&P 500 stock index, and a dummy for the drop in Reagan's approval associated with the Iran-Contra scandal.
- starting values from OLS, $\rho = 0$.
- **Computation:** See section A.3.

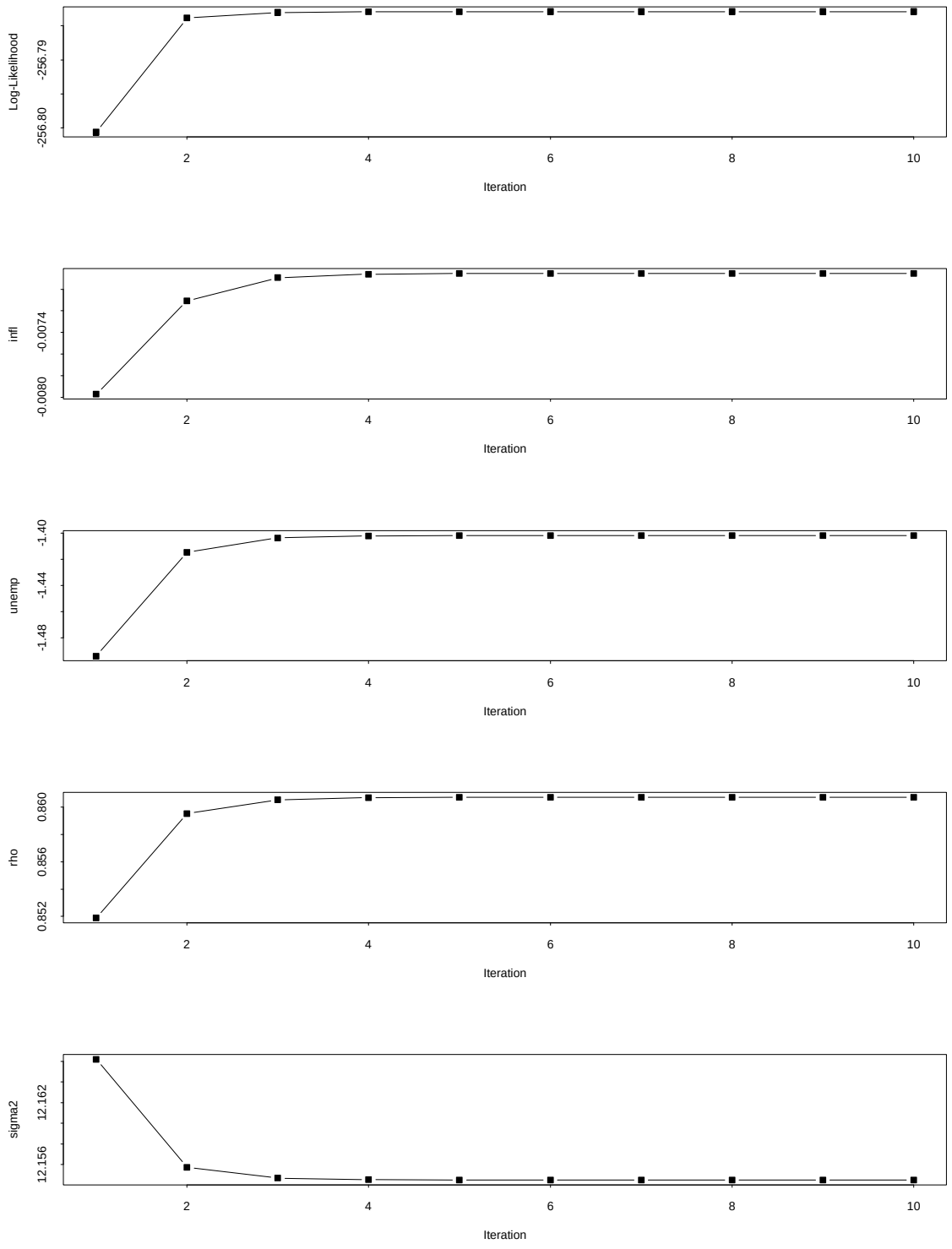

Figure 6: **Iterative History of EM Algorithm, Regression Analysis of Reagan Approval, with AR(1) disturbances.**

4.3 Example: right-censored failure time data

Tanner (1996, 67) presents a simple example where right-censoring is the source of the missing data in a failure time study.

- **Data:** See Figure 7.
- **Model:** $\log t_i = \beta_0 + \beta_1 \frac{1000}{x_i + 273.2} + u_i$, $u_i \sim N(0, \sigma^2), \forall\, i = 1, \ldots, n$.
- Failure times are unobserved (right-censored) in the cases where the motorette had not failed at the end of some known time period, c_i.
- It is clear that simply deleting the censored observations will lead to invalid estimates of the effect of the experimental control on failure time.
- **Log-likelihood:**

$$l(\beta_0, \beta_1, \sigma \mid t, v, Z) = -\frac{N}{2}(\ln 2\pi + \ln \sigma^2) - \sum_{i=1}^{m} \frac{(t_i - \beta_0 - \beta_1 v_i)^2}{2\sigma^2} - \sum_{i=m+1}^{N} \frac{(Z_i - \beta_0 - \beta_1 v_i)^2}{2\sigma^2} \tag{10}$$

  where Z_i is the unobserved log-failure time for case $i > m$.Note that this expression of the log-likelihood differs from standard treatments of right-censoring. The usual formulation (e.g., King (1989, 208--10); Greene (1993, 732)) replaces the term containing Z_i in (10) with

$$\sum_{i=m+1}^{N} \ln \left[1 - \Phi \left(\frac{c_i - \beta_0 - \beta_1 v_i}{\sigma} \right) \right].$$

  and proceeds with direct MLE. i.e., direct MLE is unproblematic here; the censoring example is chosen for illustration only.

- given the presence of the Z_i terms, this log-likelihood can not be evaluated.
- *E* step, imputations for the missing failure times: we know that $Z_i > c_i$, and so an imputation for Z_i is reasonably straightforward to obtain. Applying a standard result on the truncated normal (Maddala (1983, 65); Johnson, Kotz and Kemp (1992, 156)) to this setting yields

$$E(Z_i \mid \beta_0, \beta_1, v_i, \sigma, Z_i > c_i) = \beta_0 + \beta_1 v_i + \lambda \left(\frac{(c_i - \beta_0 - \beta_1 v_i)}{\sigma} \right)$$

where $\lambda(x) = \frac{\phi(x)}{1-\Phi(x)}$, where $\phi(\cdot)$ and $\Phi(\cdot)$ are density and cumulative distribution function of the standard normal distribution, respectively ($\lambda(\cdot)$ is often referred to as the inverse Mills ratio).

- *M* step: run a least squares regression of the complete vector of log-failure times $(t_i, Z_i^{(t)})$, on v_i and a constant to obtain $\boldsymbol{\beta}^{(t+1)}$ (the *M* step). A slight correction to the least squares estimate is required to obtain $\sigma^2_{(t+1)}$ (see Tanner (1996, 42)).
- Iterations continue until convergence in the log-likelihood in (10) or in the parameters.
- Applying the *EM* algorithm to the data in Tanner (1996, Table 4.1) yielded the iterative history for $\beta_0, \beta_1, \sigma^2$, and the log-likelihood shown in Figure 4.3. Starting values for $\boldsymbol{\beta}$ and σ came from a least squares regression on the complete data, ignoring the censoring.
- Code appears in A.4

4.4 Remarks

Many applications of the *EM* algorithm:

- missing data Little and Rubin (1987)
- mixture models -- in time series context, the so-called Hamilton (1990) model.
- unobserved state vector (Kalman filter) -- Watson & Engle (1983); Shumway & Stoffer (1982).

Weakness: no standard errors "built-in" as in (quasi-) Newton methods, which use/approximate 2nd derivatives of the log-likelihood function. Various solutions proposed in the literature, including bootstrapping.

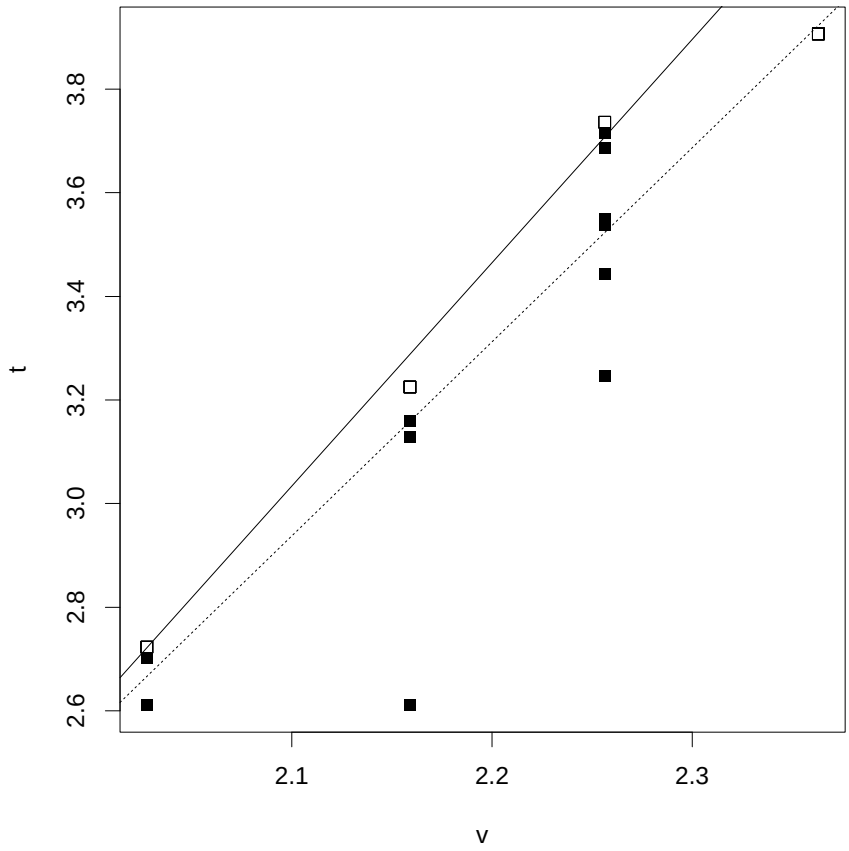

Figure 7: **Censored Failure Time Data.** Right-censored failure times are plotted with an open square. The solid line shows the fit from the *EM* algorithm; the dotted line is the naive least squares estimate.

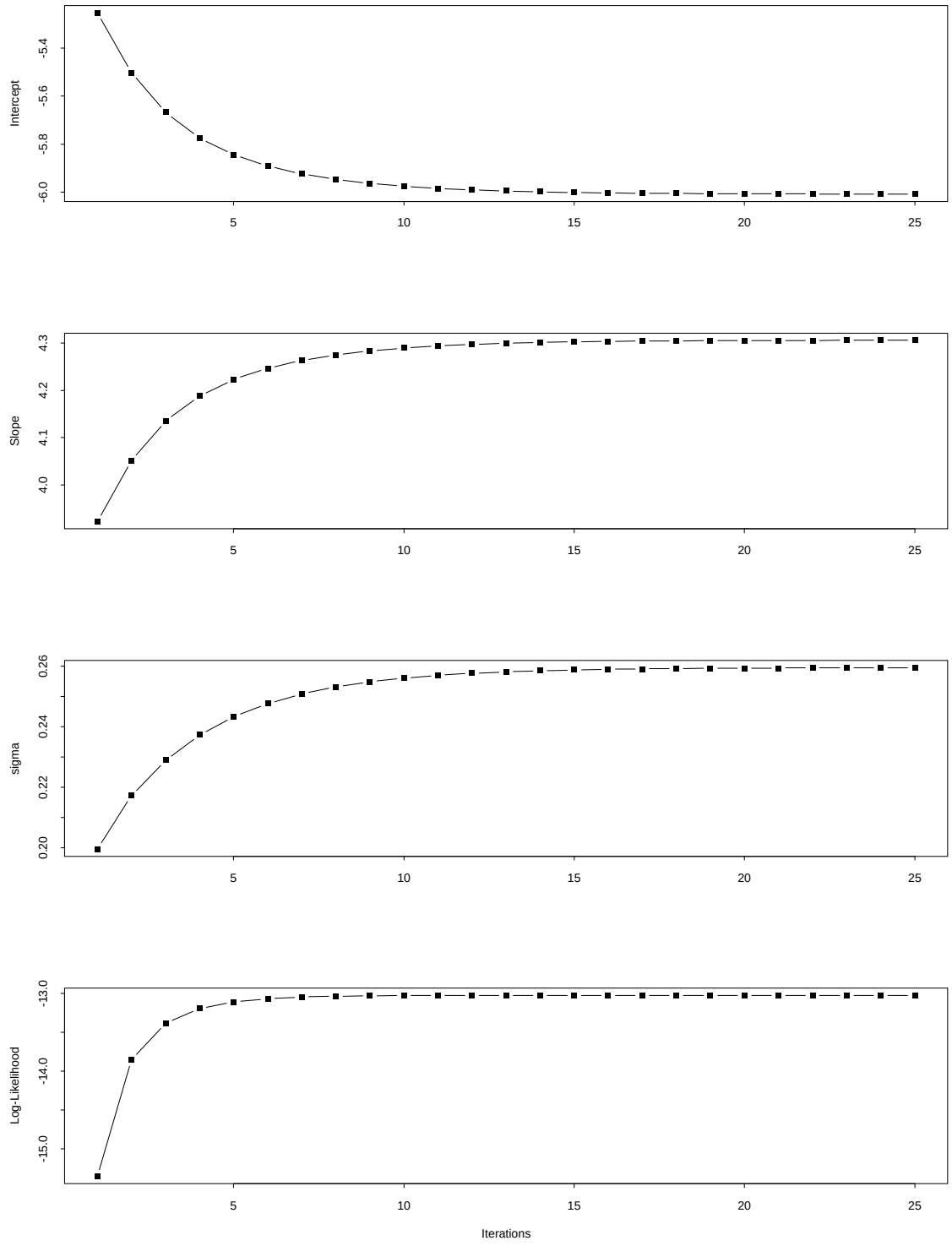

Figure 8: **Iterative History, *EM* algorithm with censored failure time data from Figure 7 (Tanner, 1996, 67).** The *EM* algorithm converges relatively quickly in this example, since the amount of missing information is small relative to the observed data. The consequences of censoring are easily discerned by comparing the starting values of the intercept and slope parameters with their estimated values given an imputation for the censored values. For instance, the effect of the independent variable is underestimated by ignoring the censoring.

4.5 Monte Carlo Implementation of the *E*-Step

Given

$$Q(\theta, \theta^{(i)}) = \int_{\mathcal{Z}} \log p(\theta|Y, Z) p(Z|\theta^{(i)}, Y) dZ$$

- **E step (Monte Carlo):**
  1. Draw $z_1, \ldots z_m \overset{\text{iid}}{\sim} p(Z|\theta^{(i)}, Y)$
  2. Let $\hat{Q}^{(i+1)}(\theta, \theta^{(i)}) = \frac{1}{m} \sum_{j=1}^{m} \log p(\theta|Y, z_j)$.
- **M step:** $\hat{Q}$ is maximized to obtain $\theta^{(i+1)}$.
- sampling (and averaging) from the conditional distribution for the missing data, rather than making an imputation based on the sufficient statistics. Why?
- how big should m be? Relatively small when $\theta^{(i)}$ is far from the true value, but larger when getting closer (Tanner, 1996, 80-81).
- Unlike EM, successive iterations won't converge on the MLE; after a while, the algorithm will be *sampling* in the neighborhood of the maximum. Apparently random fluctuations in the log-likelihood suggest being at a maximum. Suggestions in the statistics literature as to how to diagnose convergence and how large to make m.

Remark: Rubin (1987) terms the z_j *multiple imputations*.

Example: Chan and Ledolter (1995) use this MCEM (Monte Carlo EM) algorithm to estimate a time series model for count data. They also present some results on how this algorithm converges to a neighborhood of the MLE or the mode of a posterior distribution.

4.6 Data Augmentation

From Tanner (1996, ch5):

Data augmentation closely resembles *EM*, except that one *samples* repeatedly from the *both* densities in the posterior identity,

$$p(\theta \mid Y) = \int_Z p(\theta \mid Y, Z)\ p(Z \mid Y)\, dZ,$$

Through sampling the data augmentation explores the entire likelihood (or a distribution proportional to the likelihood) or posterior density.

1. *Imputation Step*. Sample m times from the current approximation to the *predictive distribution* for Z,

$$P(Z \mid Y) = \int_\Psi p^{(i)}(Z \mid \psi, Y)\, p(\psi|Y) d\psi,$$

   where $\psi \in \Psi$ is a parameter(s) characterizing the dependence of the augmented data, Z, on the observed data Y. Denote the sample $z_1, \ldots, z_m$. More specifically:

   (a) Given $p^{(i)}(\theta \mid Y)$, sample a value of θ, θ^*.

   (b) Sample z_j from $p(Z \mid \theta^*, Y)$, with θ^* from the preceding step.

   Repeat these preceding two steps m times, i.e., $j = 1, \ldots, m$. Rubin (1987) calls the z_j *multiple imputations*.

2. *Posterior Step*. Update the current approximation to $p(\theta \mid Y)$ to be the average (or mixture) of augmented posteriors of θ, given the augmented data from step (1)

$$p^{(i+1)}(\theta \mid Y) = \frac{1}{m} \sum_{j=1}^{m} p(\theta \mid z_j, Y),$$

   and return to the imputation step.

   To sample from the mixture of augmented posteriors, Tanner (1996, 92) recommends sampling θ^* from a randomly selected component of the mixture.

- The integration in the posterior identity is performed via "brute force" Monte Carlo methods.

- With m large enough, approximations of posterior densities are "exact" or "near-exact".
- With $m = 1$, have *chained data augmentation*, which is actually the simplest kind of *Gibbs sampling*.

5 Gibbs Sampling

- A multivariate extension of chained data augmentation
- Gather all random quantities --- "stuff" (parameters, latent data, missing data) --- into $\boldsymbol{\theta}$, and **sample** from the conditional distribution for each component of $\boldsymbol{\theta}$.
- A fully stochastic generalization of the techniques encountered so far

Consider $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_d)$. Iteration i of the Gibbs sampler starts with $\boldsymbol{\theta}^{(i)} = (\boldsymbol{\theta}_1^{(i)}, \boldsymbol{\theta}_2^{(i)}, \ldots, \boldsymbol{\theta}_d^{(i)})$ and makes the transition to $\boldsymbol{\theta}^{(i+1)}$ via the following scheme:

a. Sample $\boldsymbol{\theta}_1^{(i+1)}$ from $p(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_2^{(i)}, \boldsymbol{\theta}_3^{(i)}, \ldots, \boldsymbol{\theta}_d^{(i)}, Y)$.
b. Sample $\boldsymbol{\theta}_2^{(i+1)}$ from $p(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_1^{(i+1)}, \boldsymbol{\theta}_3^{(i)}, \ldots, \boldsymbol{\theta}_d^{(i)}, Y)$.
⋮ ⋮
d. Sample $\boldsymbol{\theta}_d^{(i+1)}$ from $p(\boldsymbol{\theta}_d \mid \boldsymbol{\theta}_1^{(i+1)}, \boldsymbol{\theta}_2^{(i+1)}, \ldots, \boldsymbol{\theta}_{d\text{-}1}^{(i+1)}, Y)$.

- The full joint posterior density for all of $\boldsymbol{\theta}$ has been broken down in to a series of conditional densities, thereby circumventing the "curse of [high] dimensionality".
- The sequence of vectors produced by this scheme, $\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(t)}, \ldots$, are a Markov chain.
- Under a fairly wide set of conditions,
  1. $\boldsymbol{\theta}^{(t)}$ converges in distribution to $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_d \mid Y)$, as $t \to \infty$; i.e., the joint posterior distribution of the parameters is the stationary distribution of the Markov chain generated by successive iterations of the Gibbs sampler.
  2. Averaging over a functional of the output of the Markov chain produces a quantity that converges (almost surely) to the functional of the posterior density $p(\boldsymbol{\theta} \mid Y)$ (this the Monte Carlo part of MCMC): i.e.,

$$\frac{1}{t}\sum_{i=1}^{t} f(\boldsymbol{\theta}^{(i)}) \xrightarrow{\text{a.s}} E(f(\theta)).$$

5.1 Historical Remarks

- "Gibbs" = J. Willard Gibbs (1839-1903), with whom we associate "Gibbs free energy" and many other quantities and concepts in statistical mechanics and statistical chemistry.
- "Gibbs distributions" (pdfs over Markov random fields).
- Besag (1974) -- given $\boldsymbol{\theta}$ laid out as a lattice, the joint distribution of the elements of $\boldsymbol{\theta}$ is uniquely determined by the d conditional distributions.
- Many resonances here for Bayesians, well aware of the relationships among marginal and conditional densities.
- Image-reconstruction Geman and Geman (1984); lattice of pixels. Still a very active area of application.
- Comparision with other methods; see Table 1.

Procedure	*Output*	*Inference*
MLE: optimization of likelihood function $\mathcal{L}(\boldsymbol{\theta}\|\mathbf{y}) \propto f(\mathbf{y}\|\boldsymbol{\theta})$.	point estimate: $\hat{\boldsymbol{\theta}}_{\text{MLE}}$	$\text{var}(\theta) \approx -\left[\left.\frac{\partial^2 \mathcal{L}(\boldsymbol{\theta}\|\mathbf{y})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'}\right\|_{\hat{\boldsymbol{\theta}}_{\text{MLE}}}\right]^{-1}$
EM: Let $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = \int \ln[p(\boldsymbol{\theta} \mid Z, Y)]\ p(Z \mid \boldsymbol{\theta}^{(t)}, Y)\, dZ,$ 1. *E* step: Compute $Z^{(t)} = E(Z\|\boldsymbol{\theta}^{(t)}, Y)$ 2. *M* step: $\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \ln[p(\boldsymbol{\theta} \mid Z^{(t)}, Y)]$	point estimate: $\hat{\boldsymbol{\theta}}_{\text{MLE}}$	as for MLE.
MCMC: Let $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_J)'$. 1. Sample $\boldsymbol{\theta}_1^{(t+1)}$ from $p(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_2^{(t)}, \boldsymbol{\theta}_3^{(t)}, \ldots, \boldsymbol{\theta}_J^{(t)}, Y)$. 2. Sample $\boldsymbol{\theta}_2^{(t+1)}$ from $p(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_1^{(t+1)}, \boldsymbol{\theta}_3^{(t)}, \ldots, \boldsymbol{\theta}_J^{(t)}, Y)$. ⋮ … *J*. Sample $\boldsymbol{\theta}_J^{(t+1)}$ from $p(\boldsymbol{\theta}_J \mid \boldsymbol{\theta}_1^{(t+1)}, \boldsymbol{\theta}_2^{(t+1)}, \ldots, \boldsymbol{\theta}_{J\text{-}1}^{(t+1)}, Y)$.	sampled values: $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(T)}$	calculate confidence intervals from observed quantiles of sampled $\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(T)}$.

Table 1: ***Summary of Alternative Methods of Statistical Estimation and Inference.***

5.2 Example: Probit model for binary data

From Albert and Chib (1993):

- Model: $y_i^* = \mathbf{x}_i\boldsymbol{\beta} + \varepsilon_i,\ \ \varepsilon_i \sim\ N(0,1)\ \forall\, i = 1,\ldots,n$

$$\begin{aligned} y_i = 0 \ &\Rightarrow\ y_i^* < 0 \\ y_i = 1 \ &\Rightarrow\ y_i^* \geq 0 \end{aligned}$$

- Prior distributions: $\boldsymbol{\beta} \sim N(\boldsymbol{\beta}_{\text{prior}}, \mathbf{B}_{\text{prior}})$.
- Conditional distributions:

$$\begin{aligned} y_i^* | (y_i = 0, \mathbf{x}_i, \boldsymbol{\beta}) \ &\sim\ N(\mathbf{x}_i\boldsymbol{\beta}, 1) I(y_i^* < 0) \quad \text{(trunc. Normal)} \qquad (11) \\ y_i^* | (y_i = 1, \mathbf{x}_i, \boldsymbol{\beta}) \ &\sim\ N(\mathbf{x}_i\boldsymbol{\beta}, 1) I(y_i^* \geq 0) \quad \text{(trunc. Normal)} \qquad (12) \end{aligned}$$

$$\boldsymbol{\beta} | \mathbf{y}^*, \mathbf{X}, \mathbf{y} \ \sim\ N(\tilde{\boldsymbol{\beta}}, \tilde{\mathbf{B}}), \qquad (13)$$

where

$$\begin{aligned} \tilde{\boldsymbol{\beta}} \ &= \ (\mathbf{B}_{\text{prior}}^{-1} + \mathbf{X}'\mathbf{X})^{-1} (\mathbf{B}_{\text{prior}}^{-1}\boldsymbol{\beta}_{\text{prior}} + \mathbf{X}'\mathbf{y}^*) \\ \tilde{\mathbf{B}} \ &= \ (\mathbf{B}_{\text{prior}}^{-1} + \mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

n.b., with a diffuse prior $\tilde{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}^*$ and $\tilde{\mathbf{B}} = (\mathbf{X}'\mathbf{X})^{-1}$ (the least squares regression estimates).

- Starting values for $\boldsymbol{\beta}$ from least squares regression of binary dependent variable $\mathbf{y}$ on $\mathbf{X}$.
- Gibbs sampler, for iteration t:
  1. Sample $y_i^{*(t)}$ from respective truncated Normals in (11) and (12)
  2. Sample $\boldsymbol{\beta}^{(t)}$ from multivariate Normal in (13)
- Results: See Figure 9 and Table 2.
- Computational details. See A.5 for an implementation in Splus. Figure 9 is produced by the code in A.6.

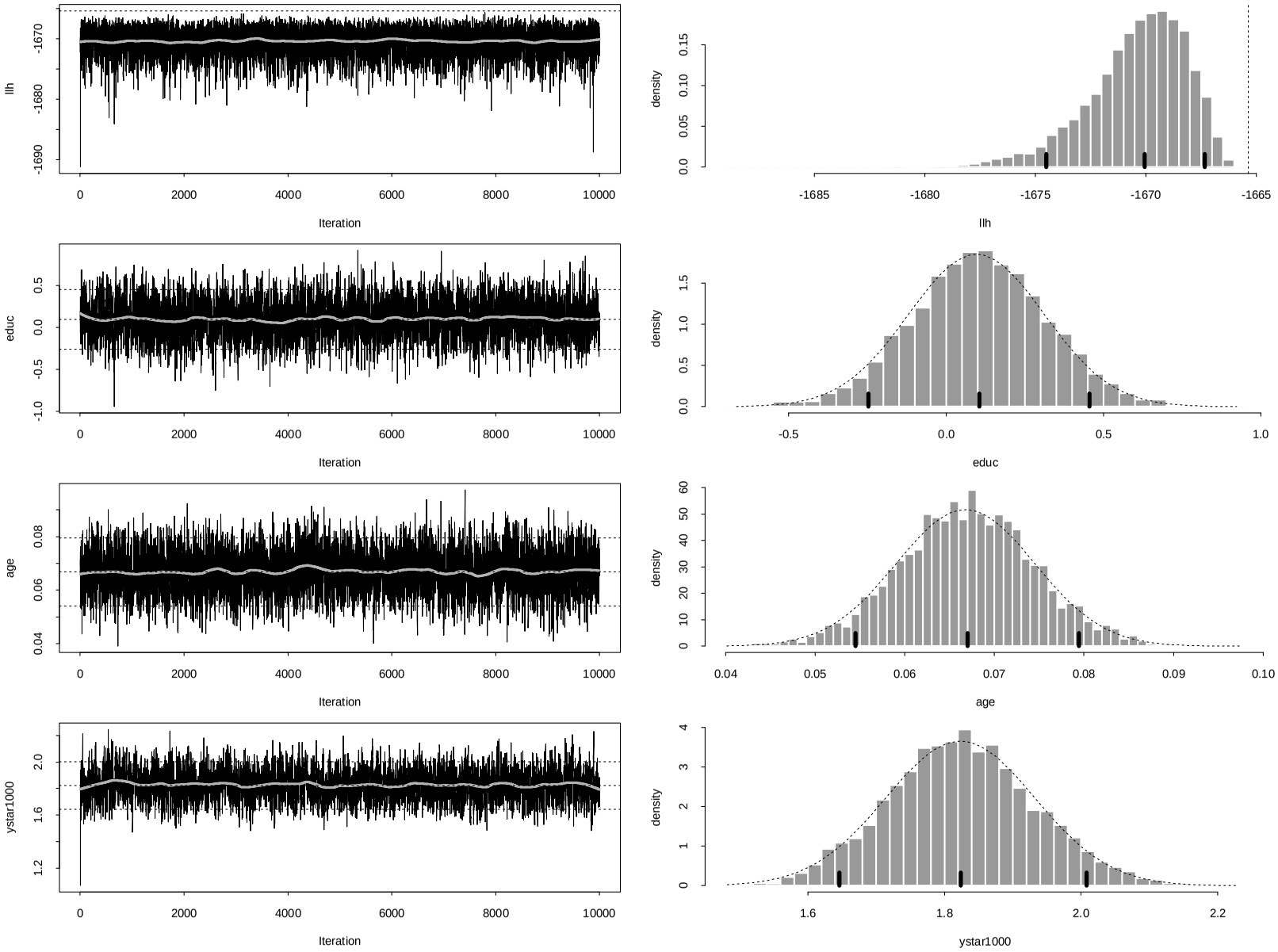

Figure 9: **Output of Gibbs sampler, probit example.** The left hand panels show the iterative history of the Gibbs sampler for the designated quantities, with the dotted lines indicating the location of the MLE, and the thicker grey line indicating a moving average (estimated by loess). The right hand panels show the posterior density of each quantity as a histogram, using the last 5,000 iterations of the Gibbs sampler, with dotted lines indicating the asymptotic Normal density implied by the MLEs; the tick marks on the horizontal axis indicate the 5th, 50th and 95th percentiles of the Gibbs samples.

	MLE	*MCMC*
Intercept	-2.32 (.56) [-3.24, -1.40]	-2.34 - [-3.26, -1.43]
Education	.096 (.22) [-.26, .45]	.11 - [-.25, .46]
Education^2	.021 (.022) [-.015, .057]	.020 - [-.016, .056]
Age	.067 (.008) [.054, .079]	.067 - [.054, .079]
Age^2	-.00047 (.00008) [-.00061, -.00034]	-.00047 - [-.00061, -.00034]
South	-.094 (.061) [-.19, .007]	-.095 - [-.19, .006]
Gubernatorial Election	.065 (.066) [-.044, .17]	.064 - [-.046, .17]
Closing Day	-.021 (.020) [-.053, .012]	-.020 - [-.053, .012]
Education $\times$ *Closing Day*	.0063 (.0081) [-.0071, .020]	.0061 - [-.0073, .020]
Education^2 $\times$ *Closing Day*	-.00061 (.00082) [-.0020, .00074]	-.00059 - [-.0020, .00076]

Table 2: **Comparison of MLEs and Gibbs sampler output, probit model of voter turnout.** Standard errors appear in parentheses for the MLEs. For the Gibbs sampler output, the mean of the last 5,000 samples is reported as the point estimate, no standard error is reported, and a 90% confidence interval is reported in brackets; the 90% confidence interval implied by the MLEs point estimate and standard error (assuming asymptotic Normality) is also reported in brackets.

5.3 Diagnosing Convergence

- MCMC algorithms will get to the desired posterior density for a very wide class of models, even though it may take a long time to get there.
- Determining how long is "sufficiently long" in particular settings is an ongoing topic of research (e.g., Rosenthal, 1995; Polson, 1996; Roberts, 1996).
- (Tierney, 1997, 397) notes that "universally useful, reliable [convergence] diagnostics do not exist, and cannot exist", given the problem-specific Markov chains generated by MCMC.
- (Cowles and Carlin, 1996) provide a comprehensive review of 13 diagnostics.
- Difficulty is that MCMC algorithms produce *samples* from distributions, rather than the value of a function being optimized (refer to Table 1, above):

> Worse yet, the Markov nature of the algorithm means that members of this sample will generally be *correlated* with each other, slowing the algorithm in its attempt to sample from the entire stationary [posterior] distribution and muddying the determination of appropriate Monte Carlo variances for estimates of model characteristics based on the output. ...such high correlations, both within the output for a single model parameter (*autocorrelations*) and across parameters (*cross-correlations*) are not uncommon, caused, for example, by a poor choice of parameterization or perhaps overparameterization. The latter situation can of course lead to "ridges" in the posterior or likelihood surface, long the bane of familiar statistical optimization algorithms (Cowles and Carlin, 1996, 883-4).

- Graphical inspection of the output of an MCMC algorithm is critically important in assessing problems with convergence.
- High within-chain autocorrelations are obvious from a trace plot.
- Multi-modal posterior distributions are also obvious.

- Slow mixing and multi-modal posteriors are not fatal in and of themselves --- the theoretical results guaranteeing convergence to the posterior distribution apply to a wide range of circumstances --- but the MCMC algorithm may have to be run for a very long period in order to reassure oneself that the algorithm is exploring all regions of the parameter space with positive posterior probability.

5.3.1 Geweke diagnostic

- Geweke's (1992) observation that for some function of a scalar output of the MCMC algorithm, say $g(\boldsymbol{\theta})$, the spectral density of the time series $\{g(\boldsymbol{\theta}^{(t)})\}$ can be used to estimate the asymptotic variance of an estimate of the average of the time series.
- This permits comparison of averages from two (or more) stages of the Markov chain (say "early" with n_A iterations and "late" based on the last n_B iterations), which yield estimates $\bar{g}(\boldsymbol{\theta})_A$ and $\bar{g}(\boldsymbol{\theta})_B$.
- The difference of these means divided by the asymptotic standard error of the difference tends to a standard normal distribution as $n \to \infty$ (holding n_A/n and n_B/n constant and $n_A + n_B < n$).
- Cowles and Carlin (1996, 866) discuss the strengths and weaknesses of this diagnostic. In particular, it is unclear how large n_A and n_B should be, relative to n, although Geweke suggested $n_A = .1n$ and $n_B = .5n$.

5.3.2 Parallel Gibbs Samplers

- Gelman and Rubin (1992) recommend starting the Gibbs sampler with **overdispersed starting points**
- By overdispersed it is meant that the variance among the different starting points should be greater than that thought to exist in the target distribution.
- This is especially useful when working with a posterior distribution reasonably thought to be multi-modal.
- i.e., run several Gibbs samplers in parallel.

- Given output from parallel MCMC algorithms, a simple test statistic can be formed by comparing the **within-sequence** and **between-sequence** variation in each scalar component of $\boldsymbol{\theta}$ Gelman and Rubin (1992).

- If the chains have converged on the same posterior density, then the between-sequence variation should be small relative to the within-sequence variation.

- Formally, consider an estimate of the marginal posterior variance of some scalar estimand of interest ψ; i.e.,

$$\widehat{\text{var}}^{+}(\psi|y) = \frac{n\text{-}1}{n}W + \frac{1}{n}B,$$

  where W is the (average) within-chain variance and B is the between-chain variance, for some scalar of interest ψ, conditional on observed data y.

- As $n \rightarrow \infty$ (i.e., the MCMC algorithm is run for longer periods), the contribution of the between-chain variation gets smaller, since it picks up weight $1/n$ in contributing to $\widehat{\text{var}}^{+}(\psi|y)$. Simultaneously, the within-chain variance increasingly dominates this term with additional iterations.

- Gelman and Rubin propose the following statistic as a convergence diagnostic:

$$\sqrt{\hat{R}} = \sqrt{\frac{\widehat{\text{var}}^{+}(\psi|y)}{W}}.$$

  This quantity declines to 1 as $n \rightarrow \infty$, and can be interpreted as the "potential scale reduction" that might result from continuing to run the MCMC algorithm.

- Given streams of output from parallel Gibbs samplers, this statistic can be calculated after a pre-specified number of iterations; Gelman et al. (1995, 332) suggest that values of $\sqrt{\hat{R}}$ below 1.2 are "acceptable", but any determination of convergence will vary from data set to data set.

- This is one of the more simple versions of the Gelman and Rubin convergence diagnostic; more complicated versions and generalizations appear in the statistical literature (e.g., Brooks and Gelman, 1998).

- tradeoff between using finite CPU for multiple chains vs one longer chain.

- I implemented the parallel Gibbs samplers recommendation for the probit turnout example; see Figures 10 and 11.
- Computation: see A.7. Figure 10 is produced by the program in section A.8. Two different implementations using WinBUGS are shown in sections A.9 and A.10.

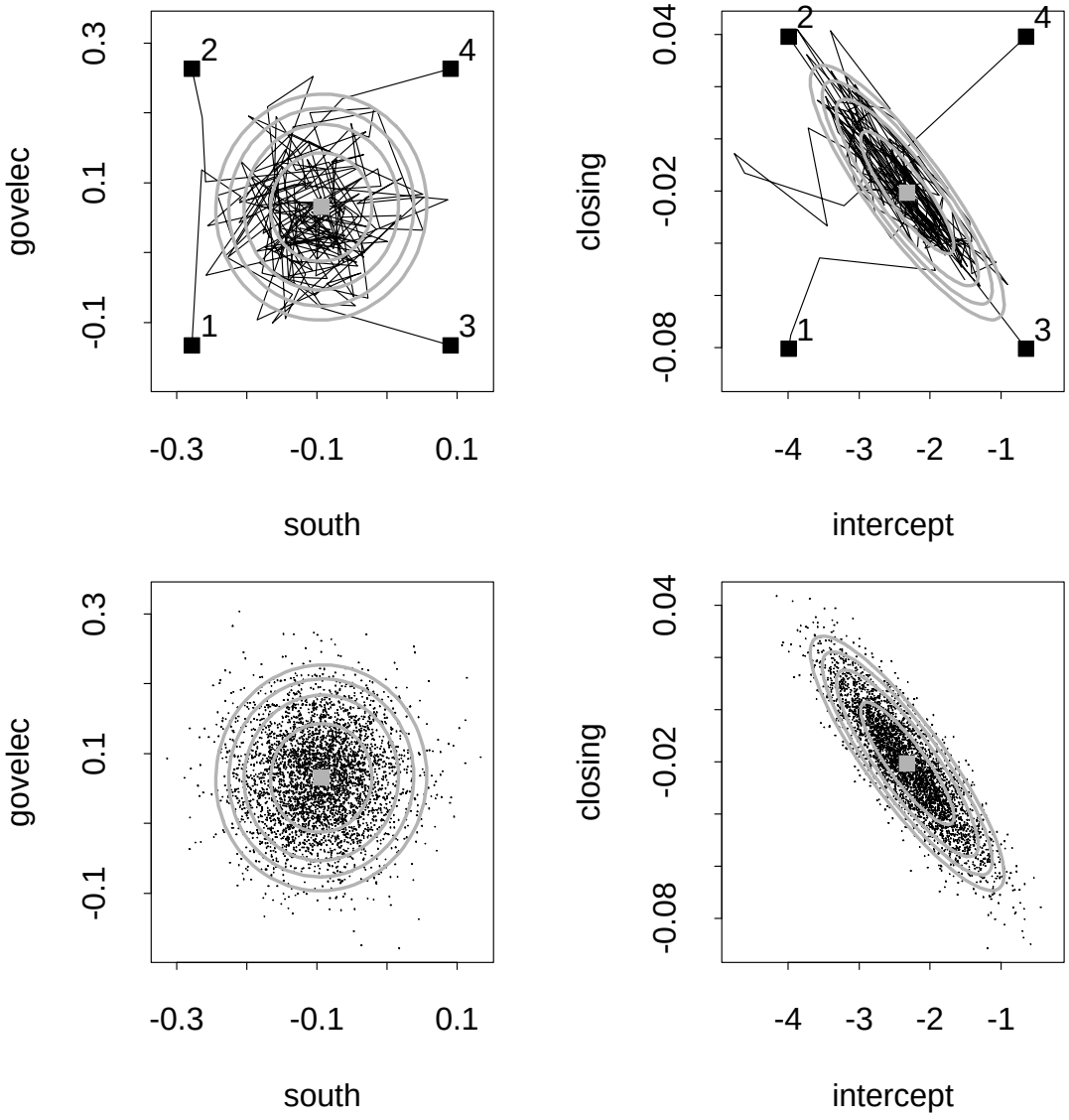

Figure 10: **Output of Gibbs sampler, turnout probit model.** Four chains were run in parallel, starting from widely dispersed starting values. The bottom two plots show the traces of the chains for the first 50 iterations, in two-dimesional subsets of the parameter space. The ellipses indicate likelihood contours (50%, 80%, 90%, 95%), and the square indicates the MLE.

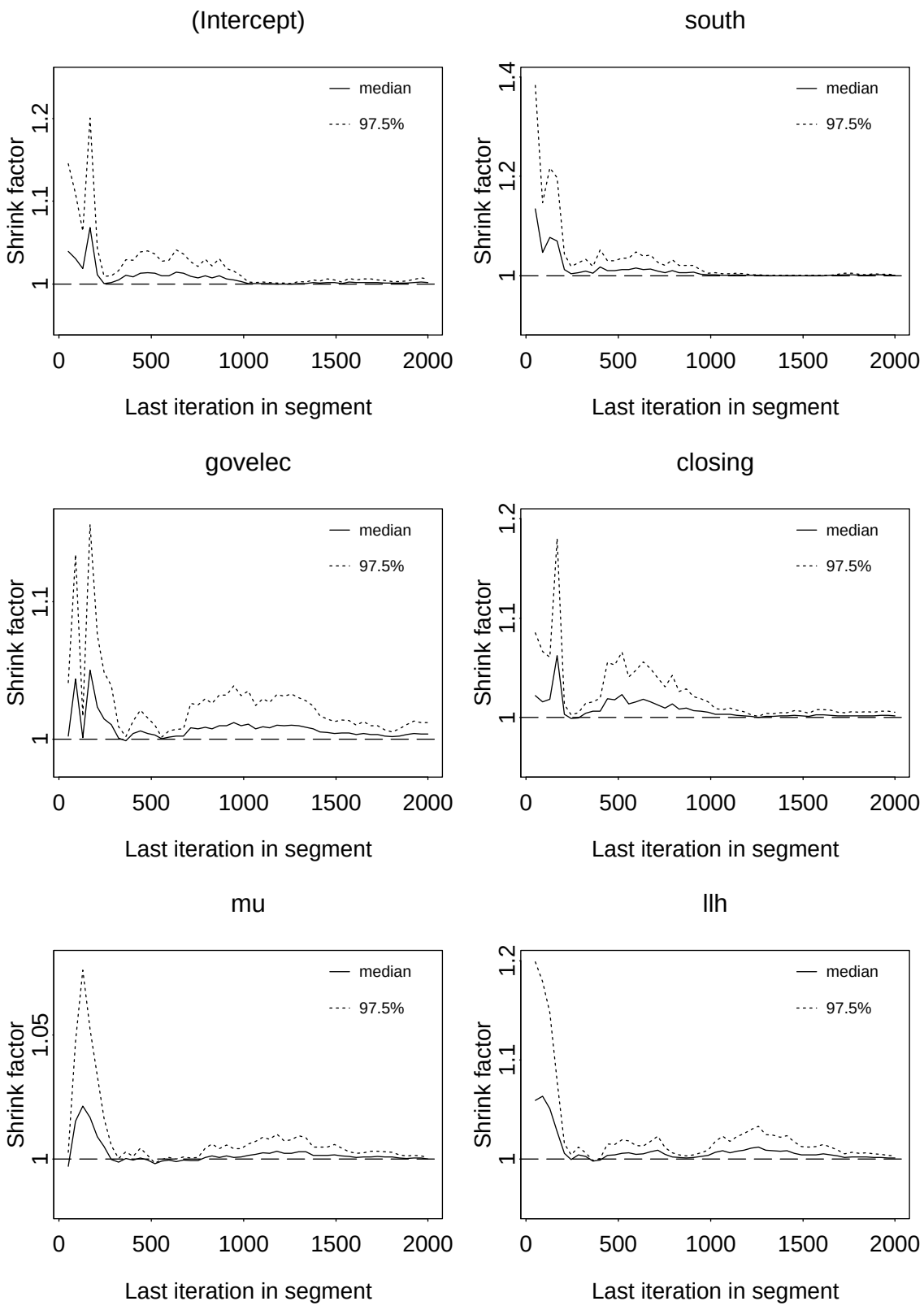

Figure 11: **Gelman and Rubin Shrink Factors, Probit Example.** The Gelman and Rubin test statistic is calculated over the course of the iterations, and plotted as a trace plot. The shrink factors all quickly fall towards 1 for the indicated quantities, suggesting that the MCMC algorithm has converged on the posterior density.

5.4 Example: Linear Regression with AR(1) disturbances

From Chib (1993):

- Model:

$$\begin{aligned} y_t &= \mathbf{X}_t\boldsymbol{\beta} + u_t, \\ u_t &= \rho u_{t\text{-}1} + \varepsilon_t, \quad |\rho| < 1, \\ \varepsilon_t &\sim N(0, \sigma^2) \quad \forall\, t \end{aligned}$$

- $\boldsymbol{\theta} = (\boldsymbol{\beta}, \sigma^2, \rho)$
- priors (possibly informative):

$$\begin{aligned} \boldsymbol{\beta} \mid \sigma^2 &\sim N(\boldsymbol{\beta}_0, \sigma^2 \mathbf{A}_0^{\text{-}1}), \\ \sigma^{\text{-}2} &\sim \Gamma(\frac{\nu_0}{2}, \frac{\delta_0}{2}), \\ \rho &\propto N(\rho_0, R_0^{\text{-}1})\, I_{\rho\in(\text{-}1,1)}, \end{aligned} \tag{14}$$

  i.e., a normal inverse-gamma prior for $(\boldsymbol{\beta}, \sigma^{\text{-}2})$ and a truncated Normal prior on ρ so as to ensure stationarity (I is an indicator function).

- Gibbs sampler:
  1. sample $\boldsymbol{\beta}^{(i+1)}$ from $p(\boldsymbol{\beta} \mid \sigma^2_{(i)}, \rho^{(i)}, \mathbf{y}, \mathbf{X})$, a multivariate Normal,
  2. sample $\sigma^2_{(i+1)}$ from $p(\sigma^2 \mid \boldsymbol{\beta}^{(i+1)}, \rho^{(i)}, \mathbf{y}, \mathbf{X})$, an inverse Gamma,
  3. sample $\rho^{(i+1)}$ from $p(\rho \mid \boldsymbol{\beta}^{(i+1)}, \sigma^2_{(i+1)}, \mathbf{y}, \mathbf{X})$, a truncated Normal.
- Slow motion:
  1. With $\rho^{(i)}$, create transformed data $\mathbf{y}^{*(i)}$ and $\mathbf{X}^{*(i)}$; e.g, $y_t^{*(i)} = y_t \text{-} \rho^{(i)} y_{t\text{-}1}$.
  2. Given that the white-noise disturbance is normal, textbook results on the Bayesian analysis of the linear regression model apply:

$$\boldsymbol{\beta}^{(i+1)} \mid \sigma^2_{(i)}, \rho^{(i)}, \mathbf{y}, \mathbf{X} \sim N(\tilde{\boldsymbol{\beta}}^{(i+1)}, \sigma^2_{(i)} \tilde{\mathbf{A}}^{\text{-}1}_{(i+1)}), \tag{15}$$

  where

$$\tilde{\boldsymbol{\beta}}^{(i+1)} = (\mathbf{A}_0 + \mathbf{X}^{*\prime(i)}\mathbf{X}^{*(i)})^{\text{-}1}(\mathbf{A}_0\boldsymbol{\beta}_0 + \mathbf{X}^{*\prime(i)}\mathbf{y}^{*(i)}) \tag{16}$$

  and

$$\tilde{\mathbf{A}}^{(i+1)} = (\mathbf{A}_0 + \mathbf{X}^{*\prime(i)}\mathbf{X}^{*(i)}). \tag{17}$$

  Sampling $\boldsymbol{\beta}^{(i+1)}$ from this k-variate normal distribution is easy.

3.

$$\sigma^{-2}_{(i+1)} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(i+1)}, \rho^{(i)} \sim \Gamma\left(\frac{T - 1 + v_0 + k}{2}, \frac{\delta_0 + Q_{\boldsymbol{\beta}}^{(i+1)} + d_{\boldsymbol{\beta}}^{(i+1)}}{2}\right) \tag{18}$$

where

$$\begin{aligned} Q_{\boldsymbol{\beta}}^{(i+1)} &= (\boldsymbol{\beta}^{(i+1)} - \boldsymbol{\beta}_0)' \mathbf{A}_0 (\boldsymbol{\beta}^{(i+1)} - \boldsymbol{\beta}_0) \\ d_{\boldsymbol{\beta}}^{(i+1)} &= (\mathbf{y}^{*(i)} - \mathbf{X}^{*(i)} \boldsymbol{\beta}^{(i+1)})' (\mathbf{y}^{*(i)} - \mathbf{X}^{*(i)} \boldsymbol{\beta}^{(i+1)}) \\ &= \mathbf{e}^{*\prime(i+1)} \mathbf{e}^{*(i+1)}. \end{aligned}$$

Sampling from the gamma distribution in (18) and inverting yields a draw from the current approximation to the marginal posterior density of σ^2.

4.

$$f\left(\rho_{(i+1)} \mid \boldsymbol{\beta}_{(i+1)}, \sigma^2_{(i+1)}, \mathbf{y}, \mathbf{X}\right) \propto N(\tilde{\rho}, \tilde{R}^{-1}) I\left(\rho_{(i+1)} \in (-1, 1)\right), \tag{19}$$

where

$$\begin{aligned} \tilde{\rho} &= \tilde{R}^{-1}\left(R_0 \rho_0 + \sigma^{-2}_{(i+1)} \sum_{t=2}^{T} u_t^{(i+1)} u_{t-1}^{(i+1)}\right), \\ \tilde{R} &= \left(R_0 + \sigma^{-2}_{(i+1)} \sum_{t=2}^{T} (u_t^{(i+1)})^2\right), \text{ and} \\ u_t^{(i+1)} &= y_t - \mathbf{x}_i \boldsymbol{\beta}^{(i+1)}. \end{aligned}$$

Draws from the normal density that lie outside the (-1,1) interval are rejected, and the iterations continue with a draw within the interval. A useful check on the plausibility of the stationarity assumption is to note the proportion of draws that fail to meet this constraint.

Application: Reagan approval data

- diffuse priors for each regression coefficient: $\beta \sim N(0, 1000)$
- diffuse priors for $\sigma^2 \sim$ Inverse-$\Gamma(.05, .05)$
- uniform prior on stationary interval [-1,1] for ρ
- Results in Figure 12 and Table 3.
- Computation: section A.11 contains the implementation using Splus, while the much simpler WinBUGS implementation appears in section A.12.

	MLE	*MCMC*
Intercept	63.92	59.28
	(10.72)	-
	[46.29, 81.55]	[27.62, 80.45]
Inflation	-.0064	.11
	(.63)	-
	[-1.04, 1.03]	[-1.11, 1.44]
Unemployment	-1.38	-.72
	(1.31)	-
	[-3.53, .77]	[-3.24, 2.53]
ρ	.86	.91
	(.053)	-
	[.77, .95]	[.81, .99]
σ^2	12.15	12.86
	(1.75)	-
	[9.27, 15.03]	[10.11, 16.49]

Table 3: **Comparison of MLEs and Gibbs sampler output, regression model of Reagan approval with AR(1) disturbances.** The median of the Gibbs sampler output (the last 4,000 of 5,000 samples) is reported as the MCMC point estimate. For the MLEs, standard errors are reported in parentheses; no standard errors are reported for the MCMC output. The 5th and 95th percentiles of the Gibbs samples are reported in square brackets; the 95% confidence interval implied by the MLE point estimate and standard error (assuming asymptotic Normality) is reported in square brackets for the MLEs.

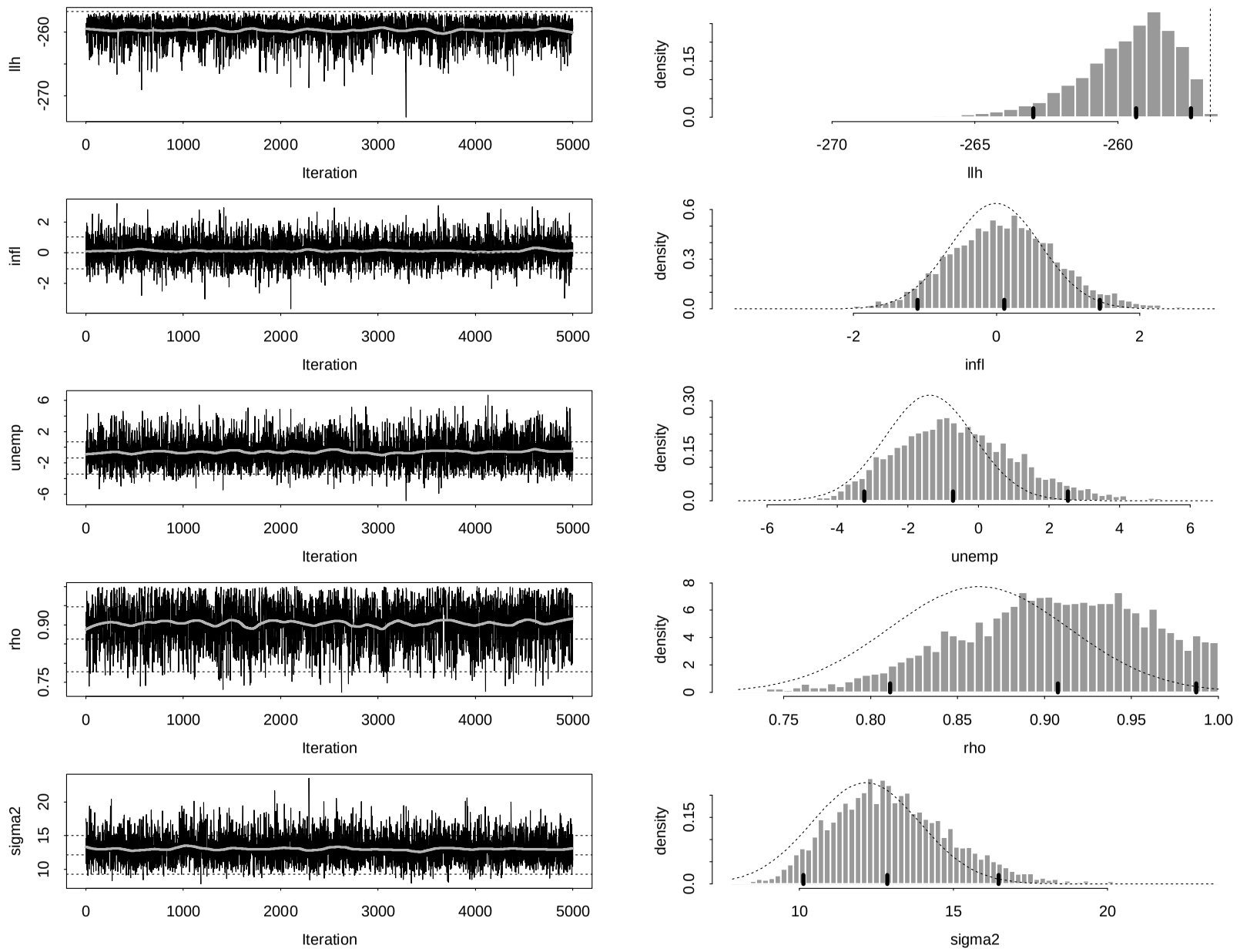

Figure 12: **Gibbs Sampler Output for Regression Model of Reagan Approval with AR(1) disturbances.** Trace plots appear in the left-hand panels; histograms using the last 4,000 iterations appear in the right-hand panels. See Figure 9 for further details.

5.5 Right-Censored Failure Times

For the *uncensored* failure times:

$$t_i|c_i = 0 \sim N(\beta_0 + \beta_1 v_i, \sigma^2)$$

but for the censored failure times:

$$t_i|c_i = 1 \sim N(\beta_0 + \beta_1 v_i, \sigma^2)I(C_i, \infty)$$

where the $I(C_i, \infty)$ indicates left-truncation (i.e., we reject any draw from the given normal distribution below the censoring point C_i).

Diffuse priors:

$$\begin{aligned} \boldsymbol{\beta} &\sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}\right) \\ \sigma^{-2} &\sim \Gamma(.001, .001) \end{aligned}$$

The conditional distributions need to implement the Gibbs sampler are very easy to implement in this instance. Conditional on sampled imputations for the censored failure times, we have essentially a standard Bayesian regression analysis, with a multivariate Normal posterior for $\boldsymbol{\beta}|\sigma^2$, and a inverse-$\chi^2$ posterior for σ^2.

Computation: WinBUGS code appears in section A.13. This example shows how convergence can be very slow for poor parameterizations: the independent univariate normal priors on the slope and the intercept parameter dramatically reduce the ability of the Gibbs sampler to traverse the parameter space; see Figures 13 and 14.

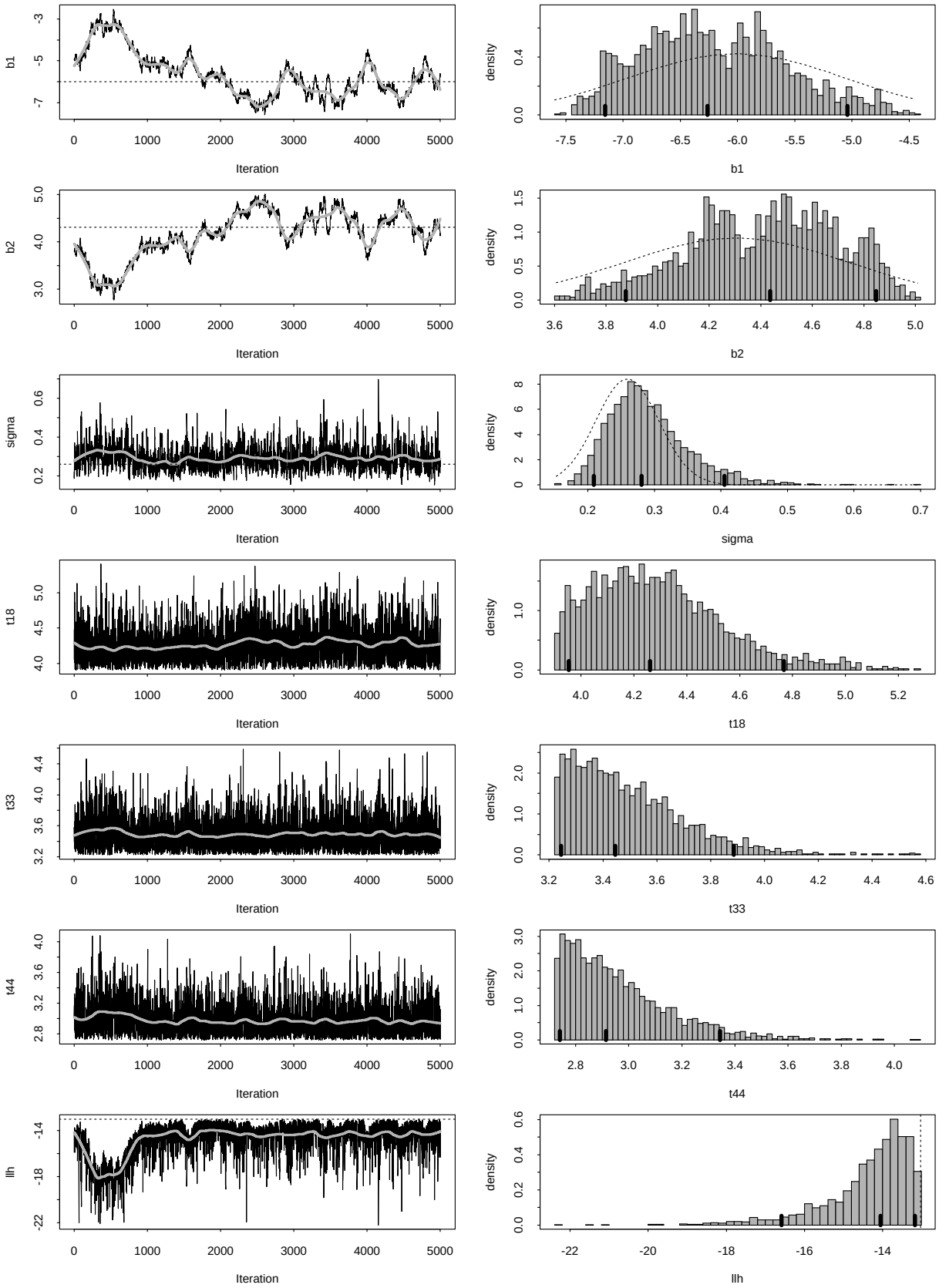

Figure 13: **Summary of Gibbs sampler output, right-censored failure time data.** Five thousand iterations were generated, with the density plots on the right-hand side of the page generated using the last 2,500 Gibbs samples.

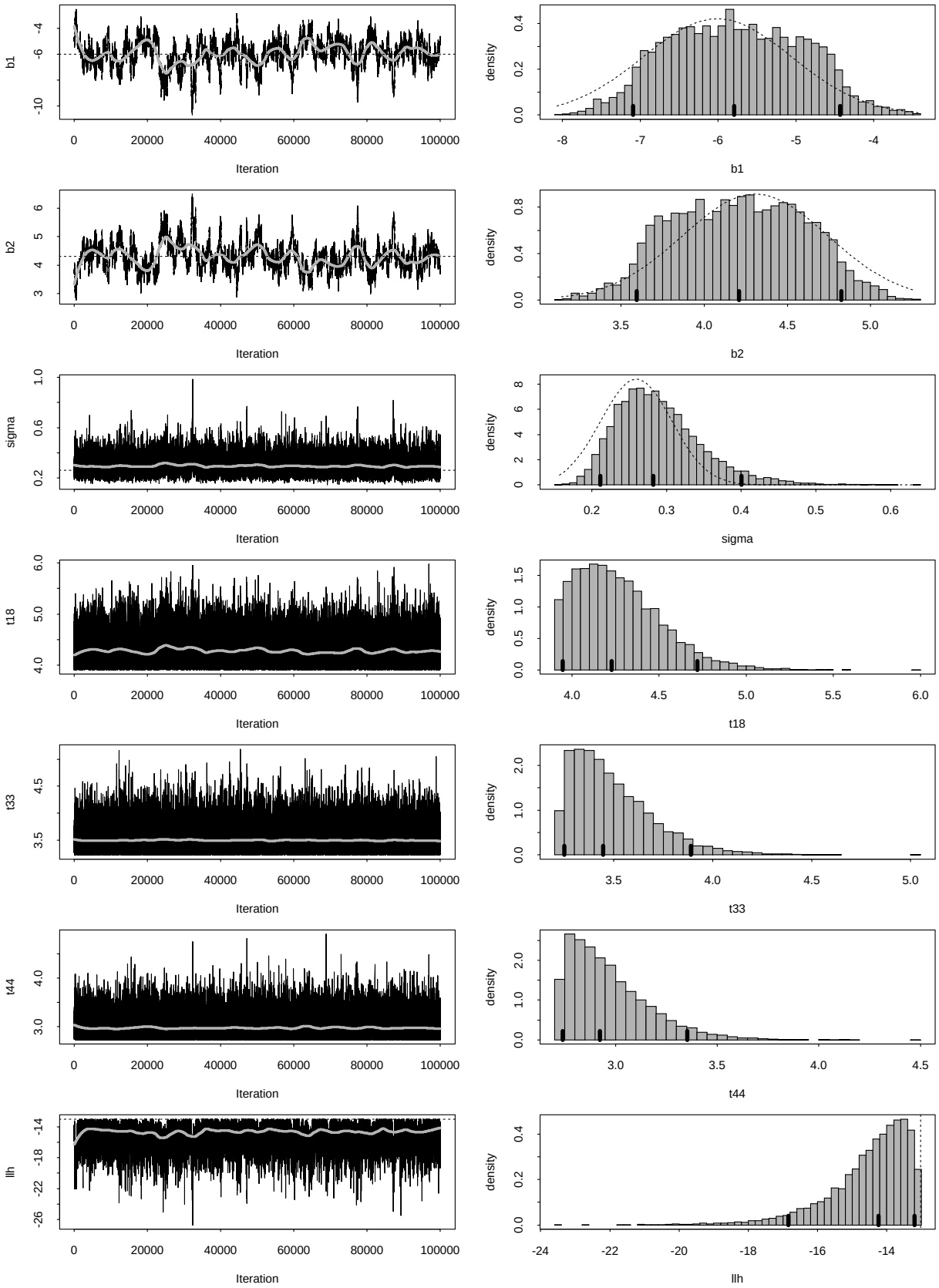

Figure 14: **Summary of Gibbs sampler output, right-censored failure time data.** Fifty thousand iterations were generated, with the density plots on the right-hand side of the page generated using the last 5,000 Gibbs samples. This longer run from the Gibbs sampler removes the apparent bi-modality in the marginal posteriors for the slope and intercept parameters apparent in the shorter-run (Figure 13).

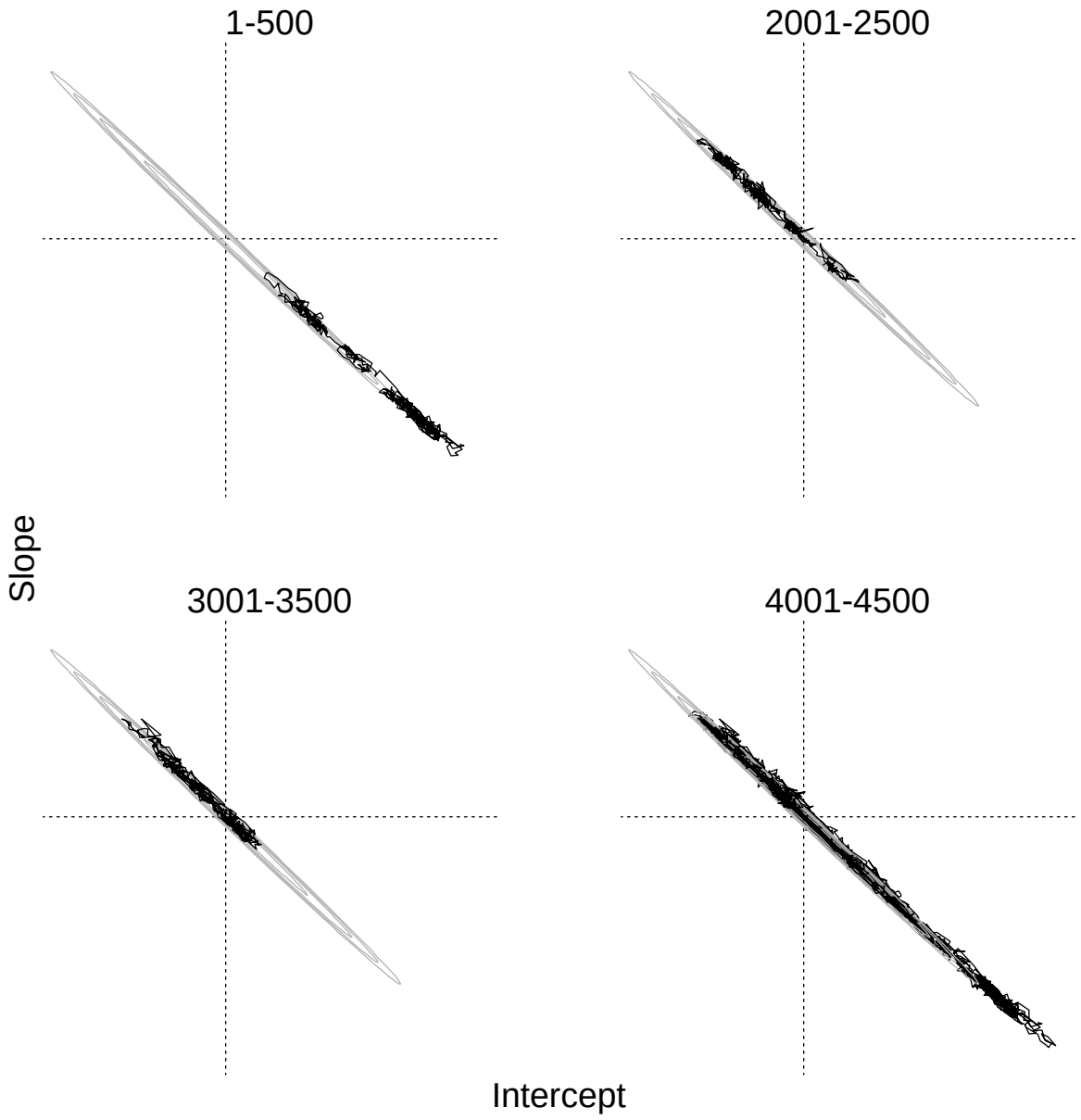

Figure 15: **Two-dimensional trace plot, right-censored failure time data.** Failing to treat the intercept and slope parameters as a block --- that is, updating them separately rather than jointly --- means the sampler can only traverse their joint parameter space slowly, due to the extremely high level of correlation between the parameters.

6 Imputations for Missing Data via MCMC

- As noted previously, an expansive notion of what constitues "missing data" drives EM and MCMC as tools for estimation (and inference).
- EM gives us "plug-in" expected values for whatever it is we happen to be treating as missing data.
- MCMC gives us a sample of "plug-in" values --- what Rubin (1987) refers to as multiple imputations. In this way MCMC allows us to average over our uncertainty in the model's other random quantities when we make inferences about any particular random quantity (be it a missing data point, or a parameter).
- Bottom line: really no distinction between a missing data point and a parameter.
- WinBUGS makes no distinction in a stochastic node. If we have the statement

  ```
  y[i] ~ dnorm(mu[i],tau)
  ```

  but for some i, y_i is missing, then WinBUGS simply samples from $N(\mu_i, \tau^{-2})$, to generate the current iteration's realization of y_i.

  Indeed, *all* of the y_i could be missing, if, say, y_i was some kind of latent variable, as in a probit model.
- A demonstration of this feature of WinBUGS appears in the "Rats" example.
- Missing values of covariates can be handled easily, espiecally if the X variable is categorical.

6.1 Example: missing bivariate Normal data

A hypothetical data set with a long history in statistics appears below in Table 4.

Assume that these data $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2)'$ are drawn from a bivariate Normal distribution with mean $\mu_1 = \mu_2 = 0$, but with an unknown variance-covariance matrix $\mathbf{\Sigma}$. There are just four complete pairs of observations, two with a

1	1	-1	-1	2	2	-2	-2	NA	NA	NA	NA
1	-1	1	-1	NA	NA	NA	NA	2	2	-2	-2

Table 4: ***Twelve Observations from a Bivariate Normal Distribution.***

correlation of 1, and two with a correlation of -1. Ignoring the missing data --- that is, using listwise deletion --- the MLE of the correlation ρ is 0.

Listwise deletion is also inefficient in that the partially observed observations contain information regarding the variance terms σ_1^2 and σ_2^2; if the goal of the analysis was to learn about the correlation between $\mathbf{x}_1$ and $\mathbf{x}_2$, then the loss of the missing data is important, since the variances contribute to the estimate of ρ.

The Gibbs sampler can be used to deal with the missing data problem and the problem of estimating $\boldsymbol{\Sigma}$, and in turn, ρ. The random quantities here are the missing data, denoted $\mathbf{z}$, and $\boldsymbol{\Sigma}$. The Gibbs sampler can be used to learn about the joint distribution of these quantities by iterating the following scheme:

1. Sample from the conditional distribution for the missing data:

   (a) if x_{i2} is observed but x_{i1} is not, sample $x_{i1}^{(t+1)}$ from

   $$f(x_{i1}|x_{i2}, \boldsymbol{\Sigma}^{(t)}) \equiv N\left(\rho^{(t)}\frac{\sigma_1^{(t)}}{\sigma_2^{(t)}}x_{i2}, \sigma_1^{2\,(t)}\left(1 - \rho^{2\,(t)}\right)\right) \tag{20}$$

   (b) if x_{i1} is observed but x_{i2} is not, sample $x_{i2}^{(t+1)}$ from

   $$f(x_{i2}|x_{i1}, \boldsymbol{\Sigma}^{(t)}) \equiv N\left(\rho^{(t)}\frac{\sigma_2^{(t)}}{\sigma_1^{(t)}}x_{i1}, \sigma_2^{2\,(t)}\left(\left(1 - \rho^{2\,(t)}\right)\right)\right) \tag{21}$$

2. Sample $\boldsymbol{\Sigma}^{(t)}$ from its conditional distribution, which (under a non-informative prior) is an inverse-Wishart distribution with scale matrix $\mathbf{S} = \mathbf{X}^{*\prime}\mathbf{X}^{*}$ and degrees of freedom $n = 12$, where $\mathbf{X}^{*}$ is the observed data augmented by $\mathbf{z}^{(t)}$. Details on how to sample from an inverse-Wishart appear in many texts; see, for instance, Gelman et al. (1995, 480).

At each iteration t,

$$\rho^{(t)} = \frac{\sigma_{12}^{(t)}}{\sqrt{\sigma_1^{2\,(t)}\sigma_2^{2\,(t)}}}$$

Analytic results establish that the posterior for ρ is bimodal, with modes close to -1 and 1. Recovering this posterior by simulation methods requires a reasonably large number of iterations. See Figure 16.

Computation: WinBUGS can't handle this kind of problem; there is currently no support for dealing with missing data in a multivariate node. Given the need for a large number of iterations and the lack of support in WinBUGS, I coded this problem in C (see section A.14). Half a million Gibbs samples were generated in 171 seconds (roughly 3,000 samples per second) on a 266MhZ Pentium II running Linux.

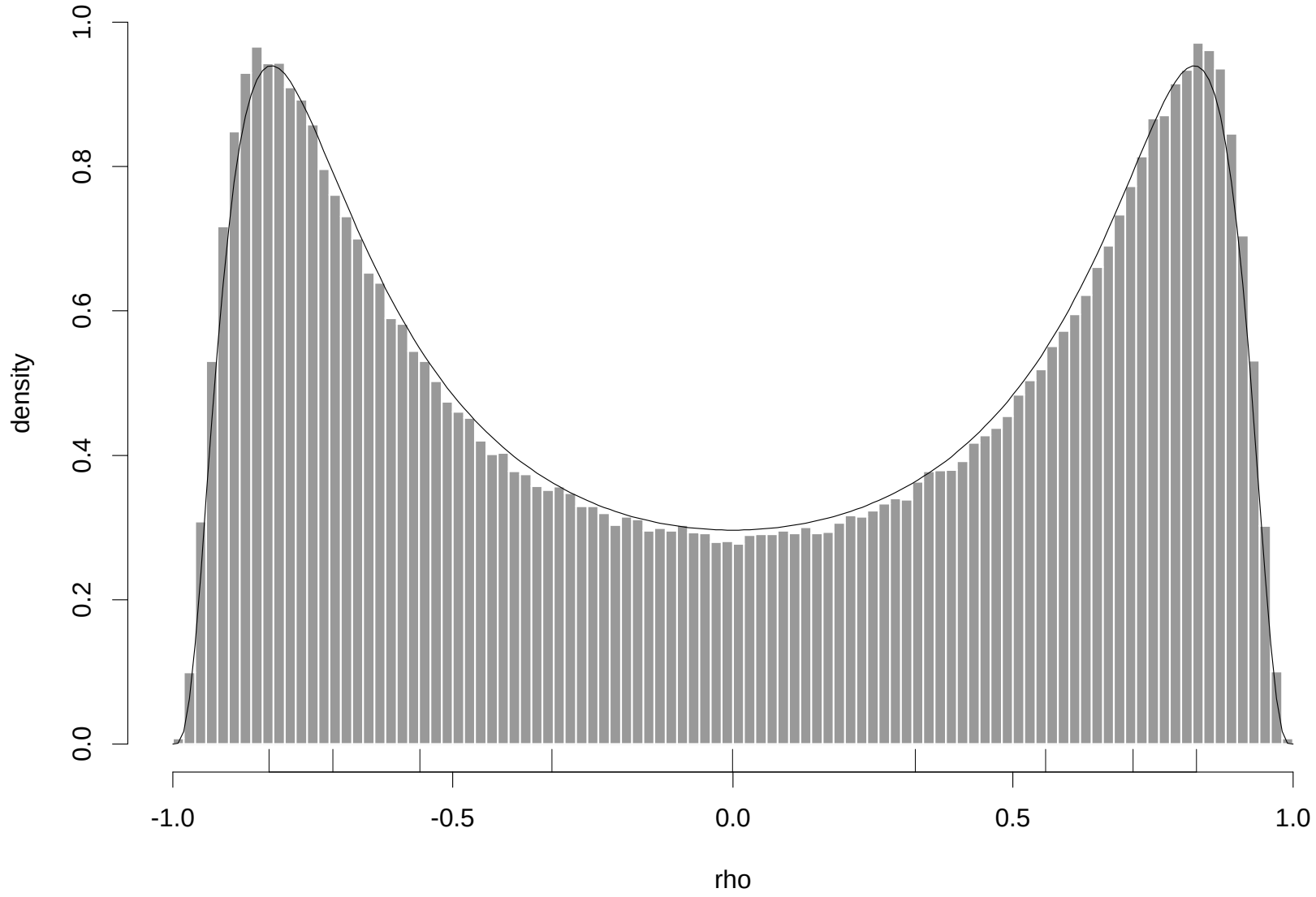

Figure 16: **Posterior density for ρ, estimated by 500,000 Gibbs samples.** The solid line is the exact analytic posterior, which is proportional to $(1 - \rho^2)^{4.5}/(1.25 - \rho^2)^8$ (Tanner, 1996, 96); the tick marks are the observed deciles of the Gibbs samples. Note that the MLE for ρ is 0, which lies in a region of low posterior probability between the two modes, and so is especially misleading in this instance.

7 Mixture Models

- $\mathbf{y} = (y_1, \ldots, y_N)$ has a density that is a *mixture* of M component densities.
- we don't know which component any y_i belongs to.
- might use regression structure unique to each component density:

$$f(y_i|\boldsymbol{\beta}, \mathbf{X}, \boldsymbol{\lambda}) = \lambda_1 f(y_i|\boldsymbol{\beta}_1\mathbf{X}) + \lambda_2 f(y_i|\boldsymbol{\beta}_2\mathbf{X}) + \ldots + \lambda_M f(y_i|\boldsymbol{\beta}_M\mathbf{X}),$$

  where $\sum_{i=1}^{M} \lambda_i = 1$.
- might also condition on $\sigma_m^2, m = 1, \ldots, M$.
- (unobserved/latent) indicators assigning data points to component densities are missing data:

$$\zeta_{im} = \begin{cases} 1 & \text{if the } i\text{th unit is from the } m\text{th component} \\ 0 & \text{otherwise} \end{cases}$$

  treat these as missing data and use *EM* or Gibbs sampling.
- hierarchical structure, in which the mixing proportions $\boldsymbol{\lambda}$ structure the latent indicators

$$f(\boldsymbol{\zeta}_i|\boldsymbol{\lambda}) = \text{Multinomial}(1|\lambda_1, \ldots, \lambda_M)$$

- the likelihood for a mixture of regression regimes is

$$\mathcal{L} = \prod_{i=1}^{n} \prod_{m=1}^{M} [\lambda_m f(y_i|\mathbf{X}, \boldsymbol{\beta}_m, \sigma_m^2)]^{\zeta_{im}} \tag{22}$$

- without constraints, MLE will generally fail (collapsing one of the components onto a single data point, driving the particular $\sigma_m^2 \to 0$ and $f(y_i) \to \infty$); i.e., the likelihood is unbounded on the edge of the parameter space.
- usual constraint is $\sigma_m^2 = \sigma^2, \forall\ m = 1, \ldots, M$, or an informative prior (reflecting beliefs about the distinctiveness of the component distributions).
- A possible alternative to techniques for robust regression (dealing with outliers).

- Also a possible alternative to the widely-used heteroskedastic probit/logit model, or other methods for picking up "varying parameters" or non-constant marginal effects like GAMs or NNs.
- Political science examples include Gelman and King (1990).
- Mixture models have been applied in a large number of settings across many disciplines, including economics Quandt and Ramsey (1973), psychology, engineering and astronomy. Raftery (1996) investigates the number of distinct "discs" in the Milky Way in an exposition of mixture models and MCMC methods.
- "by-product" of estimating a mixture model is categorizing data points into distinct clusters or regimes, mixture models bear important parallels with clustering and discriminant analysis. Accordingly, mixture models have been widely applied in biology, biochemistry, geology and geodesy; Titterington, Smith and Makov (1985, ch2) provides a comphrensive list of applications of mixture models, and is the definitive source on the analysis of mixtures prior to the advent of MCMC methods.

7.1 Demonstration

From Raftery (1996, 177):

I randomly generated 100 observations from each of the following two-component Normal mixtures

$$g(x_i) = \frac{1}{2}N(0,1) + \frac{1}{2}N(6,4), \tag{23}$$

and

$$g(x_i) = \frac{1}{2}N(0,1) + \frac{1}{2}N(2,2). \tag{24}$$

There are five parameters to be estimated here: two means (μ_1, μ_2), two variances (σ_1^2, σ_2^2), and a mixing parameter (λ, equal to .5 in both cases). The sample data and the population mixtures are plotted in Figure 17. The component densities in (23) are much better separated than those in (24), with fairly obvious consequences for the extent to which we can learn about the parameters of each. The "indistinct" mixture in the bottom panel of Figure 17 displays no bi-modality and at a glance appears close to single normal distribution.

The estimation problems inherent in all normal mixture models causes difficulties even with this simple example. The log-likelihood for this two-component normal mixturecauses difficulties for optimization algorithms. Näive attempts to maximize the likelihood in (22) quickly run into difficulties. Standard "tricks of the trade" (maximizing the log-likelihood function with respect to the inverse logit transformation of the mixing parameter λ) do not appreciably improve the performance of unconstrained optimization routines. Algorithms that allow the analyst to impose bounds on the parameter space[1] return sensible results for the more "distinct" mixture in equation (23), but continue to fail for the mixture with quite indistinct components in equation (24), reporting solutions on the edge of the parameter space ($\lambda \to 0$, or $\lambda \to 1$).

Application of *EM* to these two mixtures yields substantially better results. For the "distinct" mixture, *EM* performs quite well, even with the the two variance parameters left unconstrained. A large number of different starting values result in the same set of parameter estimates, reported in Table 5. These parameter estimates are identical to those obtained from constrained optimization of the log-likelihood function, and appear reasonable. For the "indistinct" mixture, constained optimization of the log-likelihood function fails, while *EM* is reasonably robust against choice of starting values,[2] regularly producing the set of parameter estimates reported in Table 5.

The mixtures implied by the parameter estimates are graphed in Figures 18 and 19. The "distinct" mixture is fit quite well (Figure 18), while the difficulty of statistically distinguishing the components of the "indistinct" mixture is apparent in the bottom panel of Figure 19. The estimate of the mixing parameter for the "indistinct" mixture, $\hat{\lambda} = .44$, indicates that roughly 56% of the data belong to the "right-hand" component, while the true figure is a

[1] For instance, the `nlminb()` function in SPLUS.

[2] Without any prior information or constraints labelling the component densities "high" or "low", the *EM* results sometimes reverses the estimated component densities. This is common when the component densities are not well separated. Restricting λ to the half-unit interval remedies this, effectively "labelling" the components as "more prevalent" ($\lambda > .5$) and "less prevalent" ($1 - \lambda > .5$).

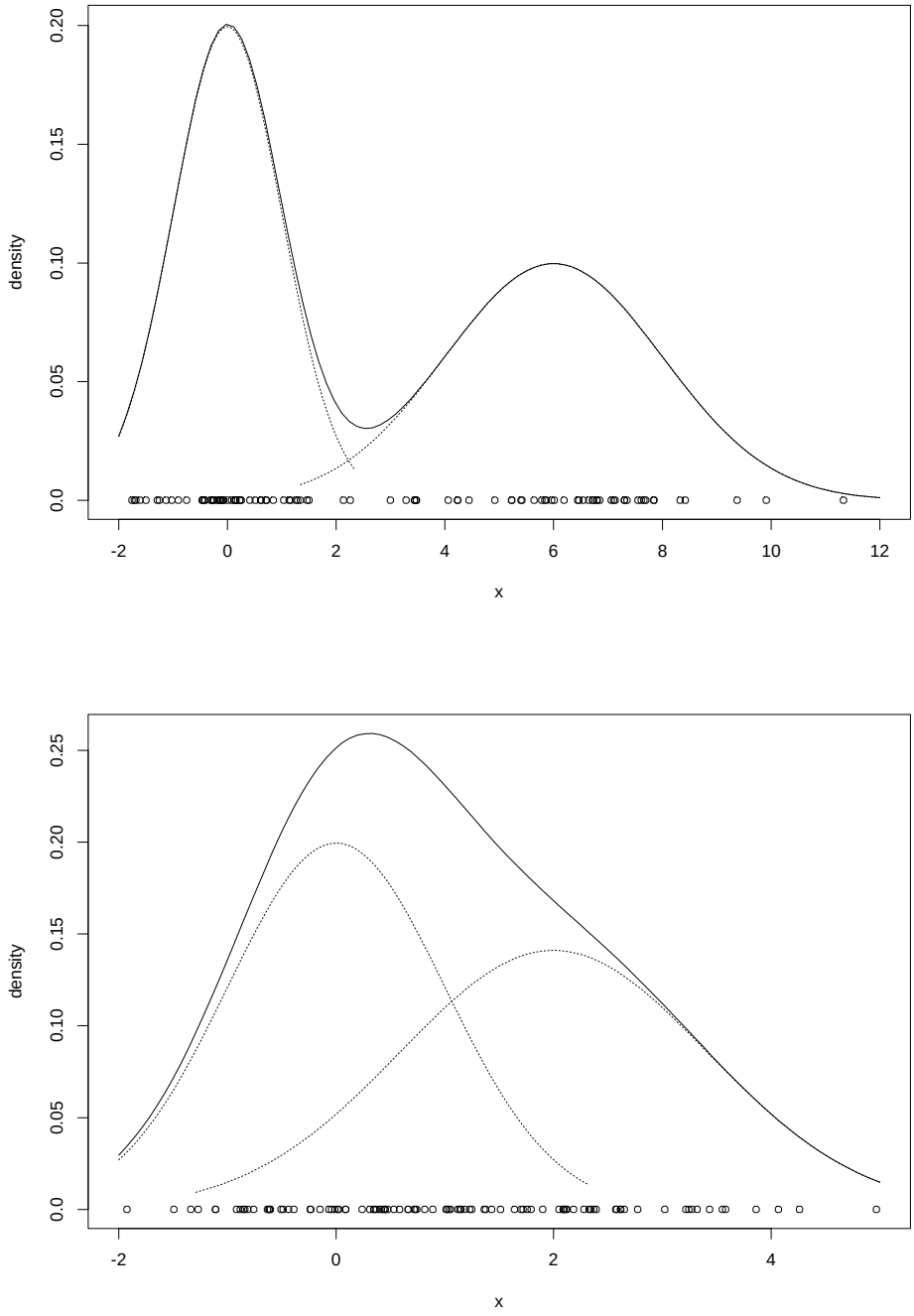

Figure 17: **Two-component normal mixtures.** The mixture in the top panel (equation 23) has components that are quite distinct, compared to the mixture in the bottom panel (equation 24). The component densities are marked by broken lines, and the sampled values (n=100) are shown with the symbols at the bottom of each graph.

Table 5: ***Estimates of Two-Component Univariate Mixtures in Equations (23) and (24).***

	"Distinct" Components, eqn (23)		*"Indistinct" Components, eqn (24)*	
Parameter	*Population*	*EM*	*Population*	*EM*
μ_1	0	-.08	0	-.09
μ_2	6	6.14	2	1.96
σ_1^2	1	.85	1	.67
σ_2^2	4	3.65	2	1.48
λ	.5	.51	.5	.44

n.b., *N*=100.

50-50 split. This rate of misclassification is not too serious, but remains one of the more serious shortcoming of these estimates of the "indistinct" mixture model. The estimates of the other parameters seem tolerable. Nonetheless, misclassificating observations is fairly typical when analyzing data generated by mixtures with poorly separated components.

7.2 Gibbs sampling for mixture models

The *EM* algorithm works reasonably well for the two contrived examples considered above. However, there is no guarantee of stable results when the dimensionality of the parameter space increases, say, in the analysis of mixing regression regimes. In the examples just presented just five parameters were estimates (two means, two variances, and a mixing parameter). When working with mixtures of regression regimes, the number of parameters can increase dramatically. For a mixture of bivariate regression regimes, there are seven parameters (two intercepts, two slopes, two variances, and a mixture parameter), and in general, with *J* components and *K* regressors (including an intercept) per regime there possibly as many as $J(K+2)$ - 1 parameters to be estimated.

It is in these high-dimensional settings that the power of a Bayesian approach to mixtures becomes apparent. As the number of parameters per mixture increases, the component densities need to be increasingly better separated in order for *EM* to distinguish the states. It is in these settings that *ad hoc* parameter constraints have been extensively employed. As I claimed earlier, prior information of some kind is typically needed in order to learn about the parameters of the component densities thought to underlie the data.

MCMC methods make the Bayesian analysis of higher-dimensional mixtures feasible, and in most cases is relatively straightforward to implement. Gibbs sampling mixture models usually amounts to little more than "stochastic *EM*", but sampling the parameters *as well as* the missing data from probability distributions. However, with MCMC methods, the moments of these sampling distributions are generated using Bayes' Rule, thereby merging sample information with the researcher's prior beliefs about the mixture parameters.

The two-component univariate normal mixture model considered above is a well-studied case, and expressions for the marginal posterior distributions needed to implement the Gibbs sampler are straightforward to derive and program. The model parameters consist of means (μ_j), variances (σ_j^2), and mixture parameters ($\lambda_j, \sum_{j=1}^{J} \lambda_j = 1$). Prior distributions for these

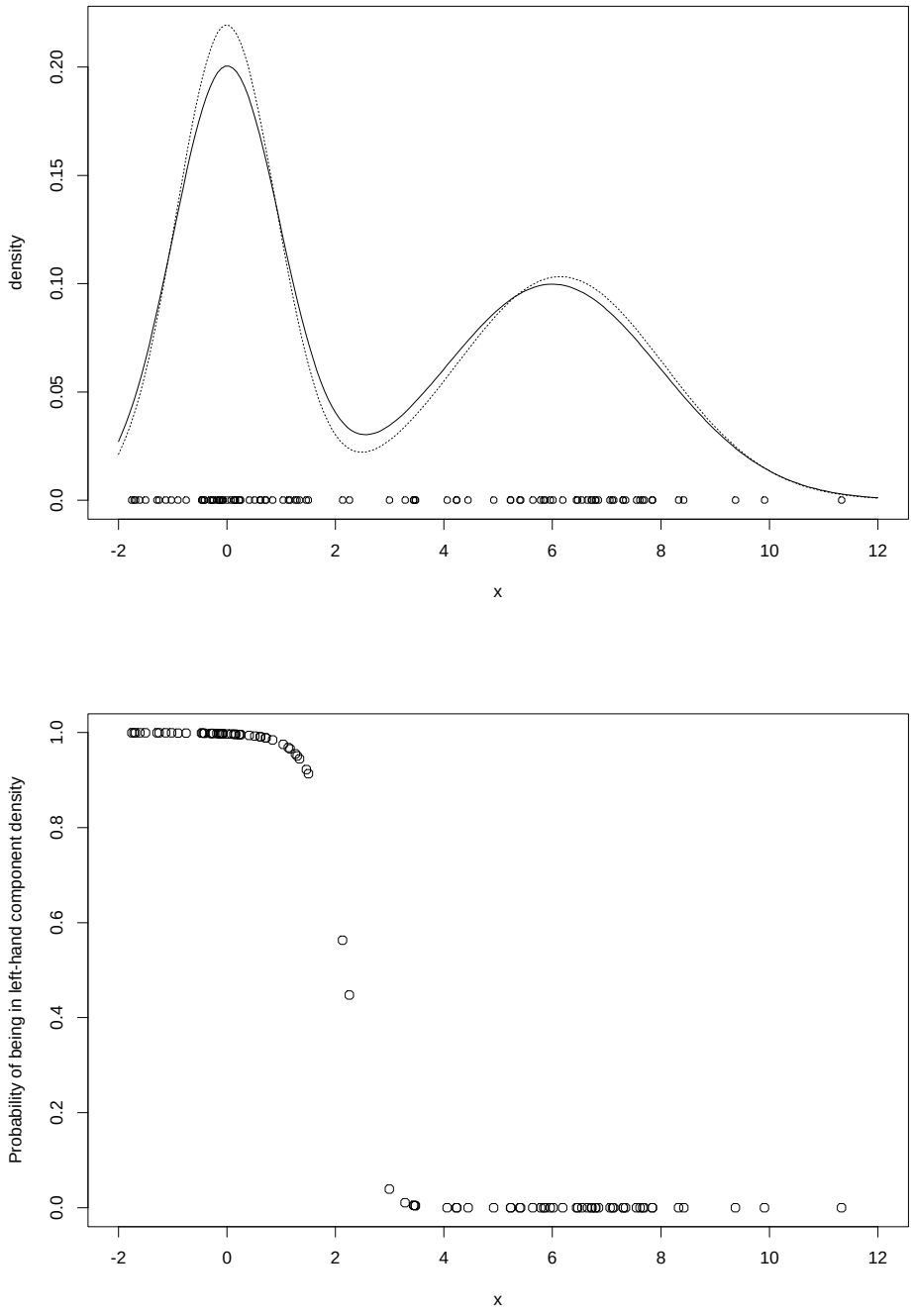

Figure 18: **Population and estimated mixture densities and estimated classification probabilities from the *EM* algorithm, for the "distinct" mixture.** In the top panel, the estimated mixture density is represented with a dashed line. The bottom panel also highlights that the component densities are well-separated, showing that the data cleave neatly into either component, with little ambiguity as to which component generated which observation.

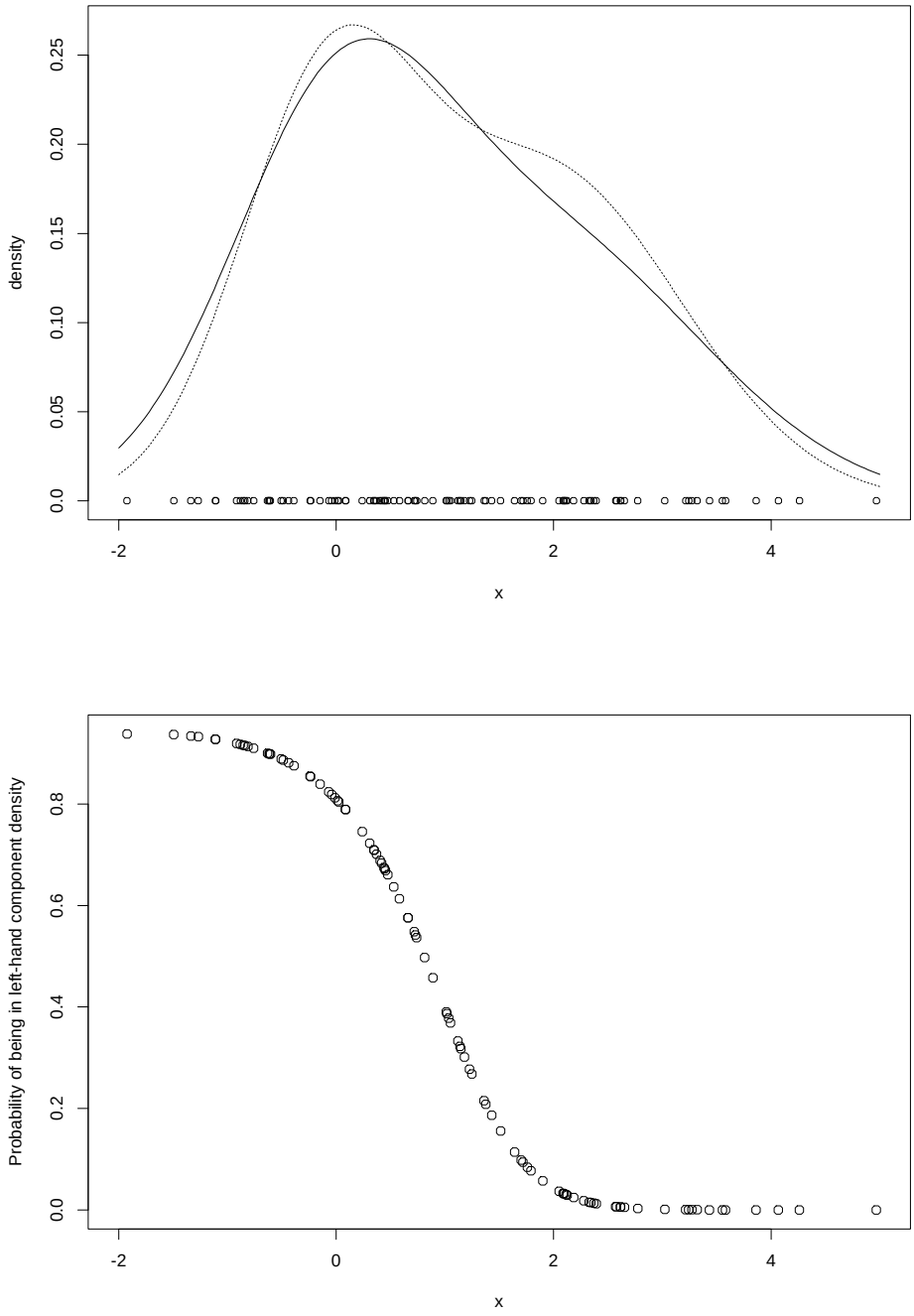

Figure 19: **Population and estimated mixture densities, and estimated classification probabilities from the *EM* algorithm, for the "indistinct" mixture.** The estimated mixture is represented with a dashed line. The large proportion of classification probabilities close to .5 (bottom panel) is typical in poorly-separated mixtures.

parameters that are flexible and yield tractable posterior distributions are

$$\begin{aligned} \mu_j &\sim N\left(\mu_{j0}, s_j^2\right) \\ \sigma_j^2 &\sim \frac{v_j s_j^2}{\chi_{v_j}^2} \\ \boldsymbol{\lambda}' &= (\lambda_1, \ldots, \lambda_J) \\ &\sim \text{Dirichlet}(a_1, \ldots, a_J). \end{aligned}$$

Given these priors and the data, each iteration of the Gibbs sampler consists of a draw from each of the following posterior distributions,

$$\mu_j \sim N\left(\frac{s_j^2 \sum_{\zeta_{ij}=1} x_i + \sigma_j^2 \mu_{j0}}{s_j^2 N_j + \sigma_j^2}, \frac{s_j^2 \sigma_j^2}{s_j^2 N_j + \sigma_j^2}\right), \tag{25}$$

where $N_j = \sum_{i=1}^{N} \zeta_{ij}$;

$$\sigma_j^2 = \frac{v_j s_j^2 + SS_j}{\chi_{v_j+N_j}^2}, \tag{26}$$

where $SS_j = \sum_{\zeta_{ij}=1} (x_i - \mu_j)^2$ and $\chi_{v_j+N_j}^2$ is a draw from a χ^2 distribution with $v_j + N_j$ degrees of freedom;

$$\zeta_{ij} \sim \text{Bernoulli}(z_{ij}), \tag{27}$$

where

$$z_{ij} = \frac{\lambda_j f(x_i|\mu_j, \sigma_j^2)}{\sum_{t=1}^{J} \lambda_t f(x_i|\mu_t, \sigma_t^2)}, \tag{28}$$

$f(\cdot|\mu_t, \sigma_t^2)$ is the Normal density with mean μ_t and variance σ_t^2, and

$$\boldsymbol{\lambda} \sim \text{Dirichlet}(a_1 + N_1, \ldots a_J + N_J). \tag{29}$$

For mixtures of regression regimes, the Gibbs sampler is a generalization of the scheme just outlined, replacing the prior and posterior distributions over the component densities mean parameters with priors and posteriors over J regime-specific vectors of regression coefficients, $\boldsymbol{\beta}_j' = (\beta_{1j}, \ldots, \beta_{kj}), j = 1, \ldots, J$. If priors for the regime-specific regression coefficients take the form

$$\boldsymbol{\beta}_j \sim N\left(\boldsymbol{\beta}_{j0}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}\right), \tag{30}$$

then standard results on the Bayesian analysis of the linear model the posterior is

$$\boldsymbol{\beta}_j \sim N\left(\boldsymbol{\beta}_j^*, \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}^*\right) \tag{31}$$

where

$$\boldsymbol{\beta}_j^* = \left[\sigma_j^{-2} \mathbf{X}_j' \mathbf{X}_j + \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}^{-1}\right]^{-1} \left[\sigma_j^{-2} \mathbf{X}_j' \mathbf{y}_j + \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}^{-1} \boldsymbol{\beta}_{j0}\right], \tag{32}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}^* = \left[\sigma_j^{-2} \mathbf{X}_j' \mathbf{X}_j + \boldsymbol{\Sigma}_{\boldsymbol{\beta}_{j0}}^{-1}\right], \tag{33}$$

and $\mathbf{X}_j$ and $\mathbf{y}_j$ are the original data $\mathbf{X}$ and $\mathbf{y}$, respectively, weighted by the binary indicators ζ_{ij}. For instance, the ith element of $\mathbf{y}_j$ equals y_i if and only if $\zeta_{ij} = 1$ and 0 otherwise, and likewise

for the ith row of $\mathbf{X}_j$. The Gibbs sampler for univariate mixtures described above can be applied by simply substituting the sampling of the mean parameters with the sampling from the k-variate posterior for $\boldsymbol{\beta}_j$ given in equation (31). Also, the sample error sum of squares for the jth component density, SS_j, is redefined as $SS_j = \mathbf{e}_j'\mathbf{e}_j$, where $\mathbf{e}_j = y_j - \mathbf{X}_j\boldsymbol{\beta}_j$.

7.3 Example: District-Level Effects in a Simulation Model of Electoral System.

Gelman and King (1990) use a mixture model to estimate properties of electoral systems, in the first application of MCMC methods in a political science setting. The model is a hierarchical mixture model, where a three-component normal mixture is fit to a vector of observations of Democratic district-level vote shares. The model is hierarchical in the sense that the parameters of the component densities normal are dependent on hyperparameters tapping longer-run features of the electoral system (e.g., the level of over-time variability in election results observed under a given set of district boundaries).

In election t, each district-level result $u_{it} = \text{logit}(v_{it}), v_{it} \in [0, 1]$, is partitioned as follows: an effect specific to each district (γ_i, where i indexes the N districts), a jurisdiction-wide effect (δ_t, constant over all districts, but variable from election-to-election), and random error. If the two systematic components are assumed to be independent, and the error components have identical and independent normal distributions, then the implied model for district-level results u_{it} is:

$$u_{it} \sim N(\alpha_{it}, \sigma^2), \qquad \alpha_{it} = \gamma_i + \delta_t, \tag{34}$$

where u_{it} is the logit of a party's proportion of the votes cast in district i, at election t.

This model is used to simulate hypothetical elections free of unrealistic and restrictive assumptions such as uniform swing; i.e., uniform swing constrains $\gamma_i = 0, \forall\ i$. The simulations consist of repeatedly drawing from the posterior for the model parameters, and building a seats-votes curve with the aggegate seat shares and vote shares implied by each simulation. Two important summary measures, bias and responsiveness are then "read off" from the resulting seats-votes curve.[3] Each simulation consists of adding a given amount of jurisdiction-wide swing (δ_t) to each district-level outcome, and averaging over uncertainty in the district-specific effects (γ_i) by sampling from posterior distribution (the three-component normal mixture).

As in the examples discussed above, we need to know how to assign the data through the three component densities. An unobserved N by 3 matrix, $\boldsymbol{\tau}$, contains this information, where $\tau_{ij} = 1$ if and only if the ith data point is imputed to have been generated by the jth component density. But the entire contents of $\boldsymbol{\tau}$ are missing data. Furthermore, even with an imputation for $\boldsymbol{\tau}$, the eight dimensional parameter vector $\boldsymbol{\theta}^*$ and its likelihood are still reasonably formidable. But conditional on the indicators in $\boldsymbol{\tau}$, each component of the normal mixture can be handled separately. And this is where the chained data augmentation algorithm comes into play. Note that via the posterior identity, the posterior density of θ can

[3] Generally, bias measures asymmetry in a seats-votes curve about the seats-votes duple (s, v) = (.5,.5), but is usually operationalized as the height of the seats-votes curve above or below .5, conditional on v = .5. Responsiveness measures elasticity in the seats-votes curve, or the rate at which changes in a party's vote-share translates into changes in its seat-share.

be factored as

$$P(\boldsymbol{\theta} \mid \mathbf{u}) = \int P(\boldsymbol{\theta} \mid \boldsymbol{\tau}, \mathbf{u})\, P(\boldsymbol{\tau} \mid \boldsymbol{\theta}, \mathbf{u})\, d\boldsymbol{\tau},$$

where the integration is over the space of $\boldsymbol{\tau}$. The *EM* algorithm is used to get starting values (θ^*) for the chained data augmentation algorithm, treating the $\boldsymbol{\tau}$ as missing data. The data augmentation algorithm in this context consists of iterating on

1. sampling $\boldsymbol{\tau}^*$ from $P(\boldsymbol{\tau} \mid \boldsymbol{\theta} = \boldsymbol{\theta}^*, \mathbf{u})$, (multinomial sampling)

2. sampling $\boldsymbol{\theta}^*$ from $P(\boldsymbol{\theta} \mid \boldsymbol{\tau} = \boldsymbol{\tau}^*, \mathbf{u})$ (Normal, inverse χ^2, and Dirichlet sampling)

As in most mixture applications, informative priors on $\boldsymbol{\theta}$ are required. Gelman and King (1990, 279) and Jackman (1994, 354--55) provide details on specifying priors and the Bayesian updating in this instance.

8 Other Applications

- hierarchical models (bio-statistics), "multi-level" models (education and sociology), "random coefficients" (econometrics). Monster area of application in bio-statistics. Recent political science examples:
  - Western's (1998) model of economic growth in a pooled cross-sectional time series setting.
  - King's (1997) method for ecological inference is in large measure a hierarchical model: King samples the precinct-level proportions from posterior distributions resulting from the weighted average of the "global" MLE fit and the precinct-specific data.
- latent dynamic structure in LDVs. e.g., time series of counts or binary outcomes.

9 Software

BUGS (Spiegelhalter et al., 1997) is a very flexible package for **B**ayesian inference **U**sing **G**ibbs **S**ampling. WinBUGS is a much improved version.

- Availabe for free download from http://www.mrc-bsu.cam.ac.uk/bugs
- great for hierarchical models
- very simple syntax
- parser converts model and data to *directed acyclic graphs* (DAGs); figures out which nodes of the graph are stochastic or deterministic; compiles code for relationships among nodes in the DAG.
- Uses conjugacy wherever possible for updating stochastic nodes.
- Log-concavity and adaptive rejection sampling.
- Missing data handled "on-the-fly" for any stochastic node (e.g., dependent variables).

A Programs

Here I present the code used to generate many of the examples and graphs. This code is also available from web site, http://jackman.stanford.edu/mcmc.

A.1 Sensitivity of posterior to prior for a proportion (Splus)

```
###############################################################################
## pictures of different priors/posteriors for placenta previa birth-rate
## data, Gelman et al, p41
##
## simon jackman, dept of political science, stanford university
## july 1999
###############################################################################

n <- 980                                        # number of obs
y <- 437                                        # successes
theta.mle <- y/n                                # MLE

t.seq <- c(seq(.01,.40,by=.01),                 # grid of values for theta
           seq(.4001,.6000,by=.0001),
           seq(.61,.99,by=.01))
lhood <- choose(n,y) * t.seq^y * (1-t.seq)^(n-y) # likelihood over grid

postscript(file="placenta.mle.ps",h=3,w=6.5)    # plot the likelihood function
par(mar=c(2.5,2.5,.1,.1))
plot(t.seq,lhood,type="l",xlab="",ylab="",cex=.5)
mtext("theta",side=1,line=2,cex=.5)
mtext("likelihood",side=2,line=2,cex=.5)

lims <- qbinom(p=c(.025,.975),                  # 95% ci around MLE
               size=n,prob=theta.mle)/n
limseq <- seq(from=lims[1],to=lims[2],length=1000)  # grid over 95% ci
flims <- choose(n,y) * limseq^y * (1-limseq)^(n-y)
polygon(x=c(limseq,rev(limseq)),                # shade 95% highest l'hod
        y=c(flims,rep(0,length(flims))),
        col=4,border=F)

lines(x=rep(theta.mle,2),
      y=c(par()$usr[3],
        choose(n,y)*theta.mle^y *(1-theta.mle)^(n-y)),
      lty=3)

abline(v=.485)
dev.off()

###############################################################################
## a function for sensitivity analysis, takes different prior vals as args
###############################################################################
tempfunc <- function(n=980,y=437,alpha=1,beta=1){
  xseq <- seq(.01,.99,by=.0001)
```

```
  prior <- dbeta(xseq,alpha,beta)         # prior
  post <- dbeta(xseq,y+alpha,n-y+beta)    # posterior
  post.lims <- qbeta(c(.025,.975),y+alpha,n-y+beta)

  limseq <- seq(from=post.lims[1],to=post.lims[2],by=.0001)
  flim <- dbeta(limseq,y+alpha,n-y+beta)

  plot(c(0,1),                            # set up plot region
       c(0,max(c(prior,post))),           # get vertical dimension ok
       xaxs="i",                          # space saver
       axes=F,
       xlab="",ylab="",type="n")

  axis(1,at=seq(0,1,by=.2))               # nice axis
  polygon(x=c(limseq,rev(limseq)),
          y=c(flim,rep(0,length(limseq))),
          col=6,border=F)                 # posterior 95% region
  lines(xseq,prior,lty=2)                 # prior
  lines(xseq,post)                        # posterior
  abline(v=theta.mle)                     # overlay MLE

  ## label
  text(x=.85,y=par()$usr[3] + .95*par()$usr[4],
       paste("alpha = ",alpha,"\nbeta = ",beta),
       cex=.85)
}

# plots for 6 different sets of priors
postscript(file="placenta.sens1.ps",
           horizontal=F,width=7,height=7.5)
par(mfrow=c(3,2))
par(mar=c(2,2,.5,.5))
tempfunc(alpha=1,beta=1)
tempfunc(alpha=.97,beta=1.03)
tempfunc(alpha=2.425,beta=2.575)
tempfunc(alpha=9.7,beta=10.3)
tempfunc(alpha=48.5,beta=51.5)
tempfunc(alpha=97,beta=103)
dev.off()

## plots for an increasingly stringent prior at theta = .25
postscript(file="placenta.sens2.ps",horizontal=F,width=7,height=7.5)
par(mfrow=c(3,2))
par(mar=c(2,2,.5,.5))
tempfunc(alpha=1,beta=3)
tempfunc(alpha=2,beta=6)
tempfunc(alpha=4,beta=12)
tempfunc(alpha=8,beta=24)
tempfunc(alpha=15,beta=45)
tempfunc(alpha=50,beta=150)
dev.off()
```

A.2 EM for binary response (SPlus)

```
##############################################################################
## estimate a probit model using the EM algorithm
##
## treat ystar as missing data
##
## simon jackman, dept of political science, stanford university
## october 1998
##############################################################################

attach("/home/simon/docs/classes/200B/98/.Data")
options(object=1e+09,digits=12)

##############################################################################
## do EM within a function that takes a single argument, obj,
## an object produced by running a regression via the lm command
## e.g., obj <- lm(y~x,y=T,x=T)
##############################################################################
tempfunc <- function(obj){
  maxiter <- 30                         # maximum number of EM iterations
  k <- length(obj$coefficients)
  bout      <- matrix(NA,maxiter,k)     # initialize output
  llhout    <- rep(NA,maxiter)          # initialize output
  q1out     <- rep(NA,maxiter)          # initialize output
  q2out     <- rep(NA,maxiter)          # initialize output
  bpbout    <- rep(NA,maxiter)          # initialize output
  ystar1000 <- rep(NA,maxiter)          # initialize output

  x <- obj$x                            # covariates
  y <- obj$y                            # (binary) dep var
  zeros <- y==0                         # index all my zeros
  ones <- y==1                          # index all my ones
  print(table(zeros,ones))

  ## run a regression for starting values
  xpx <- crossprod(x)                   # do this just once
  ixpx <- solve(xpx)                    # and store
  xpy <- t(x)%*%y
  b <- ixpx%*%xpy                       # ols for start values
  mu <- x%*%b                           # start mu somewhere...predicted values from ols
  bpb <- crossprod(b)
  n <- length(y)
  cat("regression coefficients (starting vals)\n")
  print(b)

  for (iter in 1:maxiter){              # EM iterations start here
    cat(paste("Iteration",iter,"\n"))

    ## E step, imputation for ystar and/or epsilon
    ystar <- rep(NA,n)                  # initialize to NA vector
    phi <- dnorm(-mu)                   # numerator of inverse Mills ratio
    Phi <- pnorm(-mu)                   # denom of inverse Mills ratio
    M <- rep(NA,n)
    M[zeros] <- -phi[zeros]/Phi[zeros]  # E(e|e<-mu), for y=0
```

```
  M[ones] <- phi[ones]/(1-Phi[ones])  # E(e|e>-mu), for y=1
  ystar <- mu + M                     # mu = X%*%beta
  ystar1000[iter] <- ystar[1000]      # keeping track of obs 1000

  ## evaluate q function
  q1 <- -n/2*log(2*pi) - sum(1 - M*mu)/2

  ## M-step, just a regression
  xpystar <- t(x)%*%ystar             # cross-products of X and just-imputed ystar
  b <- ixpx%*%xpystar                 # (X'X)^{-1} X'y^*
  bpbold <- bpb
  bpb <- crossprod(b)
  cat("regression coefficients:\n")
  print(b)

  ## re-evaluate q function
  ## (use old beta(t) for ystar and M, but beta(t+1) for mu)
  mu.new <- x%*%b
  q2 <- -n/2*log(2*pi) - sum(1- M*ystar + (ystar - mu.new)^2)/2
  mu <- mu.new

  ## evaluate incomplete-data (probit) log-likelihood
  llh <- sum(log(pnorm(mu[ones]))) + sum(log(1-pnorm(mu[zeros])))

  cat(paste("  log-likelihood =",llh,"\n"))
  if(iter>1){
    cat(paste("  change in llh  =",llh-llhout[iter-1],"\n"))
    cat(paste("  change in q    =",q2-q1,"\n"))
    cat(paste("  change in bpb  =",bpb-bpbold,"\n"))
  }

  ## output values for tracing iterative history
  bout[iter,] <- b
  llhout[iter] <- llh
  q1out[iter] <- q1
  q2out[iter] <- q2
  bpbout[iter] <- bpb
}

## EM iterations terminated, compute information matrix for inference
#cat("Iterations terminated, computing covariance matrix of estimates\n")
#phi <- dnorm(-mu)
#Phi <- pnorm(-mu)
#info <- matrix(0,nrow=k,ncol=k)
#m <- (phi^2)/(Phi*(1-Phi))

#info <- wcp(x,m)    # weighted cross-product function, see wcp.S

#cat("Information Matrix:\n")
#print(info)
#cat("inverse of information matrix:\n")
#vc <- solve(info)
#print(vc)

#cat("std errors:\n")
```

```
  #se <- sqrt(diag(vc))
  #print(se)

  ## housekeeping, gathering objects for output
  dimnames(bout) <- list(NULL,names(coef(obj)))
  out <- list(b=bout,
      #se=se,
      llh=llhout,
      ystar1000=ystar1000,
              bpb=bpbout,
      q1=q1out,
              q2=q2out)
  out
}
###############################################################################
## end function definition
###############################################################################

## run initial ols, for getting data in shape etc
small.lm <- lm(vote ~ educcat + educsq + age + agesq + south +
               govelec + closing + educlose + educlos2,
               data=nagler.small,
               x=T,y=T)

## this next GLM run gives us the (target) MLEs
small.glm <- glm(vote ~ educcat + educsq + age + agesq + south +
                 govelec + closing + educlose + educlos2,
                 data=nagler.small,
                 family=binomial(link=probit),
                 x=T,y=T)

emout <- tempfunc(small.lm)               ## call the EM function

###############################################################################
## some plotting commands for convergence diagnostics
###############################################################################
postscript("probitemconv.ps")             ## dump to PostScript
par(mfrow=c(4,1))                         ## 4 plots on the page
plot(1:length(emout$q1),emout$q1,
     type="b",pch=15,xlab="Iteration",ylab="q1")
plot(1:length(emout$q2),emout$q2,
     type="b",pch=15,xlab="Iteration",ylab="q2")
plot(1:length(emout$q2),emout$q2-emout$q1,
     type="b",pch=15,xlab="Iteration",ylab="qdiff")
plot(1:length(emout$bpb),emout$bpb,
     type="b",pch=15,xlab="Iteration",ylab="bpb")
dev.off()                                 ## close PostScript file

###############################################################################
## plot iterative history of quantities of interest
###############################################################################
postscript("probitem.ps")
par(mfrow=c(4,1))

plot(1:length(emout$llh),emout$llh,       ## log-likelihood
```

```
     type="b",
     pch=15,
     xlab="Iteration",
     ylab="Log-Likelihood")
plot(1:length(emout$llh),emout$b[,"educcat"], ## educ coefficient
     type="b",
     pch=15,
     xlab="Iteration",
     ylab="educ")
plot(1:length(emout$llh),emout$b[,"age"],     ## age coefficient
     type="b",
     pch=15,
     xlab="Iteration",
     ylab="age")
plot(1:length(emout$llh),emout$ystar1000,     ## y^*_{1000}
     type="b",
     pch=15,
     xlab="Iteration",
     ylab="ystar1000")
dev.off()
```

A.3 EM for regression with AR(1) disturbances (Splus)

```
###############################################################################
## regression with AR(1) disturbances, using EM
##
## simon jackman, dept of political science, stanford university
## october 1998
###############################################################################

###############################################################################
## a function for estimating rho given a regression object
###############################################################################
myrho <- function(fit,print=F){
  e <- resid(fit)
  e1 <- mylag(e,1)
  dw <- sum((e[-1]-e[-length(e)])^2)/crossprod(e)    ## Greene, 3rd ed, p591
  rho <- sum(e[-1]*e[-length(e)])/crossprod(e[-1])   ## Greene, 3rd ed, p600
  if(print){
    cat(paste("rho:         ",rho,"\n"))
    cat(paste("DW Statistic:",dw, "\n"))
  }
  rho
}

###############################################################################
# a function for doing lags properly
###############################################################################
mylag <- function(x,lag=1){
  if (is.matrix(x)){                       # is x a matrix?
    n <- dim(x)[1]
    k <- dim(x)[2]
    z <- rep(NA,k)
    y <- x[1:(n-lag),]
```

```
    for (i in 1:lag)
      y <- rbind(z,y)
  }
  else                                  # assume x is a vector
    {
      n <- length(x)
      y <- c(rep(NA,lag),x[1:(n-lag)])
    }
  y
}

#############################################################################
# a function for transforming variables, by 1st diffs and rho
#############################################################################
mystar <- function(x,rho){
    xstar <- x-(rho*mylag(x,1))
    ## is x a matrix?
    if (is.matrix(x))
      xstar[1,] <- sqrt(1-(rho^2))*x[1,]         # PW transform
    else
      xstar[1] <- sqrt(1-(rho^2))*x[1]           # PW transform
    xstar
}

#############################################################################
# a function to do iterative Cochran-Orcutt (EM) for AR(1) errors
# fit: a lm object, with X matrix and y vector
#############################################################################
mycorcutt <- function(fit,maxiter=10){
  x <- fit$x
  y <- fit$y
  n <- length(y)
  k <- dim(x)[2]
  b <- coef(fit)
  u <- y - x%*%b                                # regression residuals

  rho <- myrho(fit)                             # starting estimate of rho
  cat(paste(rho,"\n"))
  rhostar <- 0                                  # initialize rhostar

  llhout <- rep(NA,maxiter)                     # initialize output
  qout <- rep(NA,maxiter)                       # initialize output
  bout <- matrix(NA,maxiter,k)
  sigmaout <- rep(NA,maxiter)
  rhoout <- rep(NA,maxiter)

  tss <- crossprod(fit$y-mean(fit$y))           # total sum of squares
  for (iter in 1:maxiter){                      # start EM iterations
    rhostar <- sum(u[-1]*u[-n])/sum(u[-1]^2)    # estimate rho
    ystar <- mystar(y,rhostar)                  # transform y
    xstar <- mystar(x,rhostar)                  # transform x
    temp <- lsfit(xstar,ystar,intercept=F)      # GLS
    b <- temp$coef                              # updated b
    e <- temp$residuals                         # white noise residuals
    epe <- crossprod(e)                         # sse on transformed data
```

```
    sigma2 <- epe/n
    u <- y - x%*%b                                  # regression residuals

    ## log-likelihood, Greene, p600
    llh <- -n/2*(log(2*pi)+log(sigma2))
          + log(1-(rhostar^2))/2
          - epe/(2*sigma2)
    ## q function
    q <- -n/2*(log(2*pi)+log(sigma2)) - epe/(2*sigma2)

    ## output iterative history
    llhout[iter] <- llh
    qout[iter] <- q
    bout[iter,] <- b
    sigmaout[iter] <- sigma2
    rhoout[iter] <- rhostar
    yhat <- ystar - e
    r2 <- crossprod(yhat-mean(yhat))/tss         #r2
    cat(paste("Iteration ",iter,
              ": rho = ",round(rhostar,4),
              " llh = ",round(llh,8),
              "\nq = ",round(q,8),
              "\nssu = ",round(epe,4),
              "\nr2 = ",round(r2,4),
              "\n",sep=""))

    if (iter!=1){
      cat(paste("change in llh = ",
                llh-llhout[iter-1],
                "\n"))
      cat(paste("change in q function = ",
                q-qout[iter-1],
                "\n\n"))
    }
  }                                              # end EM loop

  rho.x <- e[-1]
  rho.y <- e[-n]
  rho.u <- rho.y - rhostar*rho.x
  rho.vc <- sum(rho.u^2)/sum(rho.x^2)
  rho.se <- sqrt(rho.vc)                         # standard error on rho

  out <- list(rho=rhoout,                        # package for output
              rho.se=rho.se,
              r2=r2,
              llh=llhout,
              b=bout,
              sigma2=sigmaout,
              maxiter=maxiter)
  out
}
################################################################################
## end function definitions
################################################################################
```

```
## initial OLS regression
reagan.temp <- lm(app~infl+unemp,
  data=reagan,
  x=T,y=T)
## call EM routine
ar1out <- mycorcutt(reagan.temp,maxiter=10)

## show iterative history graphically
postscript("ar1em.ps")                          ## plotting commands
par(mfrow=c(5,1))                               ## 5 plots to page

plot(1:ar1out$maxiter,ar1out$llh,
     xlab="Iteration",
     ylab="Log-Likelihood",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$b[,2],
     xlab="Iteration",
     ylab="infl",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$b[,3],
     xlab="Iteration",
     ylab="unemp",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$rho,
     xlab="Iteration",
     ylab="rho",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$sigma2,
     xlab="Iteration",
     ylab="sigma2",
     type="b",pch=15)

dev.off()
```

A.4 EM for right-censored failure time

```
###############################################################################
## regression with AR(1) disturbances, using EM
##
## simon jackman, dept of political science, stanford university
## october 1998
###############################################################################

###############################################################################
## a function for estimating rho given a regression object
###############################################################################
myrho <- function(fit,print=F){
  e <- resid(fit)
  e1 <- mylag(e,1)
  dw <- sum((e[-1]-e[-length(e)])^2)/crossprod(e)    ## Greene, 3rd ed, p591
```

```
  rho <- sum(e[-1]*e[-length(e)])/crossprod(e[-1])   ## Greene, 3rd ed, p600
  if(print){
    cat(paste("rho:         ",rho,"\n"))
    cat(paste("DW Statistic:",dw, "\n"))
  }
  rho
}

##############################################################################
# a function for doing lags properly
##############################################################################
mylag <- function(x,lag=1){
  if (is.matrix(x)){                      # is x a matrix?
    n <- dim(x)[1]
    k <- dim(x)[2]
    z <- rep(NA,k)
    y <- x[1:(n-lag),]
    for (i in 1:lag)
      y <- rbind(z,y)
  }
  else                                    # assume x is a vector
    {
      n <- length(x)
      y <- c(rep(NA,lag),x[1:(n-lag)])
    }
  y
}

##############################################################################
# a function for transforming variables, by 1st diffs and rho
##############################################################################
mystar <- function(x,rho){
    xstar <- x-(rho*mylag(x,1))
    ## is x a matrix?
    if (is.matrix(x))
      xstar[1,] <- sqrt(1-(rho^2))*x[1,]         # PW transform
    else
      xstar[1] <- sqrt(1-(rho^2))*x[1]           # PW transform
    xstar
}

##############################################################################
# a function to do iterative Cochran-Orcutt (EM) for AR(1) errors
# fit: a lm object, with X matrix and y vector
##############################################################################
mycorcutt <- function(fit,maxiter=10){
  x <- fit$x
  y <- fit$y
  n <- length(y)
  k <- dim(x)[2]
  b <- coef(fit)
  u <- y - x%*%b                              # regression residuals

  rho <- myrho(fit)                           # starting estimate of rho
  cat(paste(rho,"\n"))
```

```
rhostar <- 0                                    # initialize rhostar

llhout <- rep(NA,maxiter)                       # initialize output
qout <- rep(NA,maxiter)                         # initialize output
bout <- matrix(NA,maxiter,k)
sigmaout <- rep(NA,maxiter)
rhoout <- rep(NA,maxiter)

tss <- crossprod(fit$y-mean(fit$y))             # total sum of squares
for (iter in 1:maxiter){                        # start EM iterations
  rhostar <- sum(u[-1]*u[-n])/sum(u[-1]^2)      # estimate rho
  ystar <- mystar(y,rhostar)                    # transform y
  xstar <- mystar(x,rhostar)                    # transform x
  temp <- lsfit(xstar,ystar,intercept=F)        # GLS
  b <- temp$coef                                # updated b
  e <- temp$residuals                           # white noise residuals
  epe <- crossprod(e)                           # sse on transformed data
  sigma2 <- epe/n
  u <- y - x%*%b                                # regression residuals

  ## log-likelihood, Greene, p600
  llh <- -n/2*(log(2*pi)+log(sigma2))
        + log(1-(rhostar^2))/2
        - epe/(2*sigma2)
  ## q function
  q <- -n/2*(log(2*pi)+log(sigma2)) - epe/(2*sigma2)

  ## output iterative history
  llhout[iter] <- llh
  qout[iter] <- q
  bout[iter,] <- b
  sigmaout[iter] <- sigma2
  rhoout[iter] <- rhostar
  yhat <- ystar - e
  r2 <- crossprod(yhat-mean(yhat))/tss          #r2
  cat(paste("Iteration ",iter,
            ": rho = ",round(rhostar,4),
            " llh = ",round(llh,8),
            "\nq = ",round(q,8),
            "\nssu = ",round(epe,4),
            "\nr2 = ",round(r2,4),
            "\n",sep=""))

  if (iter!=1){
    cat(paste("change in llh = ",
              llh-llhout[iter-1],
              "\n"))
    cat(paste("change in q function = ",
              q-qout[iter-1],
              "\n\n"))
  }
}                                               # end EM loop

rho.x <- e[-1]
rho.y <- e[-n]
```

```
  rho.u <- rho.y - rhostar*rho.x
  rho.vc <- sum(rho.u^2)/sum(rho.x^2)
  rho.se <- sqrt(rho.vc)                          # standard error on rho

  out <- list(rho=rhoout,                         # package for output
              rho.se=rho.se,
              r2=r2,
              llh=llhout,
              b=bout,
              sigma2=sigmaout,
              maxiter=maxiter)
  out
}
################################################################################
## end function definitions
################################################################################

## initial OLS regression
reagan.temp <- lm(app~infl+unemp,
  data=reagan,
  x=T,y=T)
## call EM routine
ar1out <- mycorcutt(reagan.temp,maxiter=10)

## show iterative history graphically
postscript("ar1em.ps")                            ## plotting commands
par(mfrow=c(5,1))                                 ## 5 plots to page

plot(1:ar1out$maxiter,ar1out$llh,
     xlab="Iteration",
     ylab="Log-Likelihood",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$b[,2],
     xlab="Iteration",
     ylab="infl",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$b[,3],
     xlab="Iteration",
     ylab="unemp",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$rho,
     xlab="Iteration",
     ylab="rho",
     type="b",pch=15)

plot(1:ar1out$maxiter,ar1out$sigma2,
     xlab="Iteration",
     ylab="sigma2",
     type="b",pch=15)

dev.off()
```

A.5 Gibbs sampler, probit model (Splus)

```
######################################################################
## Gibbs sampler for probit model
##
## Reference:
## @Article{albertchib:jasa,
##   author =   {Albert, James A. and Siddhartha Chib},
##   title =   {Bayesian Analysis of Binary and Polychotomous
##                  Response Data},
##   journal =   jasa,
##   year =   1993,
##  volume =  88,
##  pages =  {669--79}
##}
##
##  simon jackman, dept of political science, stanford univ
##  october 1998
#############################################################################

## a (pretty fast) function for sampling from truncated Normals
## but see the inverse-uniform method in the parallel samplers version
## of this code
rtmvnorm <- function(mu,y,n){
  z <- rnorm(mu,sd=1,n=n)                    # sample unconditionally
  repeat {                                   # loop to meet constraints
    fail1 <- (1:n)[z<0 & y==1]               # index y==1 but z<0
    fail2 <- (1:n)[z>0 & y==0]               # index y==0 but z>0
    fail <- c(fail1,fail2)                   # index all failures
    n.more <- length(fail)
    if (n.more==0) break                     # if no failures, bail out
    z[fail] <- rnorm(mu[fail],sd=1,n=n.more) # else, re-sample failures
  }                                          # repeat until all ok
  z
}

## the Gibbs sampling function (diffuse priors)
probitgibbs <- function(obj,maxiter=1000){
  x <- obj$x
  y <- obj$y

  xpx <- crossprod(x)
  ixpx <- solve(xpx)
  beta <- ixpx%*%crossprod(x,y)     # starting value for beta
  mu <- x%*%beta                    # E(y^*)

  n <- length(y)
  k <- length(beta)

  ystar <- rep(NA,n)                # initialize quantities
  betaout <- matrix(NA,maxiter,k)
  llhout <- rep(NA,maxiter)
  ystarout <- rep(NA,maxiter)
  muout <- rep(NA,maxiter)
```

```
  ## comence Gibbs sampling
  for (iter in 1:maxiter){
    cat(paste("Gibbs sample number",iter," "))

    ystar <- rtmvnorm(mu,y,n)              # sample ystar (latent dep var)

    bstar <- ixpx%*%t(x)%*%ystar           # E(beta | X, y, y^*)
    beta <- mvrnorm(n=1,bstar,ixpx)        # sample beta

    cat("beta:\n")
    print(beta)
    betaout[iter,] <- beta
    ystarout[iter] <- ystar[1000]          # monitor observation 1000
    muout[iter] <- mu[1000]
    mu <- x%*%beta                         # E(y^* | X, y, beta^{t+1})
    Phi <- pnorm(mu)                       # Pr(y=1)
    llhout[iter] <- sum(y*log(Phi) + (1-y)*(log(1-Phi)))  #log-likelihood
    cat(paste("llh:",llhout[iter],"\n"))
  }

  out <- list(b=betaout,ystar=ystarout,llh=llhout,mu=muout)
  out
}
###############################################################################
## end function definitions
###############################################################################

options(memory=1e+09)
probitgibbsout <- probitgibbs(small.lm,maxiter=5000)

## dump output, to port between machines
temp <- list()
temp[[1]] <- cbind(c(1:5000),
              probitgibbsout$b,
              probitgibbsout$mu,probitgibbsout$llh)

dimnames(temp[[1]]) <- list(NULL,c("iter",
                                   names(coef(small.glm)),
                                   "mu","llh"))
data.dump("temp",
          "probitgibbsout.dmp",
          oldStyle=T)
```

A.6 Graphical summaries, Gibbs sampler output, probit model (Splus)

```
###############################################################################
## plot output of Gibbs sampler for probit model
##
## simon jackman, dept of political science, stanford university
## may 1999
###############################################################################
```

```
options(object.size=1e+08)
options(memory=200E+06)

## a function to do a specialized rug
myrug <- function(x,q=c(.05,.50,.95)){
  xtick <- quantile(x,q)                    # get quantiles
  ydim <- par()$usr[3:4]
  yin <- ydim[1] + .12*ydim[2]
  for (i in 1:length(q)){
    lines(x=c(xtick[i],xtick[i]),
          y=c(0,yin),
          lwd=5)
  }
  invisible(NULL)
}

## a function to do the actual plotting
tempfunc <- function(x,start,lab,mle=NULL,llh=NULL){
  xtime <- 1:length(x)
  if (!is.null(llh))
    ylim <- c(min(x),llh)
  else
    ylim <- range(x)
  plot(xtime,x,                             # trace plot
       xlab="Iteration",
       ylim=ylim,ylab=lab,type="l")
  blah <- loess(x~xtime, span=1/10)         # loess for smoothing
  lines.default(x=xtime,y=blah$fitted.values,lwd=3,col=6)
  if (!is.null(mle)){                       # overlay MLE
    abline(h=mle[1],lty=2)
    abline(h=mle[1]+(mle[2]*qnorm(.95)),lty=2)  # overlap MLE CIs
    abline(h=mle[1]+(mle[2]*qnorm(.05)),lty=2)
  }
  if (!is.null(llh))                        # or overlay llh
    abline(h=llh,lty=2)

  ## density (histogram) plot
  x <- x[start:length(x)]            # rest of function works with start:n
  temp <- hist(x,                    # histogram
               nclass=50,            # 50 bins
               prob=T,               # as a density
               ylab="density",
               xlab=lab,
               plot=T)               # plot
  cat("calculated histogram stuff\n")

  if (!is.null(mle)){                # overlay MLE
    cat("MLE:\n")
    print(mle)
    mleseq <- seq(from=min(x),to=max(x),length=100)
    fmle <- dnorm(mleseq,mean=mle[1],sd=mle[2])
    maxy <- max(maxy,max(fmle))
    lines(mleseq,fmle,lty=2)        # MLE implied Normal density
  }
```

```
  #if (!is.null(llh))               # this code useful when max(MLE)>max(hist)
  #  xlim <- c(min(temp$breaks),llh)
  #else
  #  xlim <- range(temp$breaks)
  #  plot(range(temp$breaks),range(temp$counts),
  #        type="n",
  #       ylim=c(0,maxy),
  #       xlim=xlim,xlab=lab,ylab="density")

  myrug(x)                          # do rug to show quantiles
  if(!is.null(llh))
    abline(v=llh,lty=2)             # show llh, for llh plots
}
###############################################################################
## end function definitions
###############################################################################

postscript(file="probitgibbs.ps",
           horizontal=T)                  # landscape
ps.options.send(black.and.white="false")  # this helps for most printers
par(mfrow=c(4,2),mar=c(4,4,1,2))          # 8 panels, squeezed onto page

## get MLEs and standard errors
mle <- summary.glm(small.glm)$coefficients[,1:2]

## call plotting function for various quantities
tempfunc(temp[,"llh"],5001,lab="llh",llh=deviance(small.glm)/-2)
tempfunc(temp[,"educcat"],5001,lab="educ",mle=mle[2,])
tempfunc(temp[,"age"],5001,lab="age",mle=mle[4,])

## get MLE of ystar[1000] and its standard error
mle <- predict(small.glm,newdata=small.glm$x[1000,],se.fit=T)

tempfunc(temp[,"mu"],5001,lab="ystar1000",mle=c(mle$fit,mle$se.fit))

dev.off()
rm(tempfunc)
```

A.7 Parallel Gibbs samplers, probit model (Splus)

```
###############################################################################
## estimate Gibbs sampler for probit model, but run "p" Gibbs samplers
## in parallel
##
## dump output in S format suitable for CODA (see CODA manual)
## a list, with matrix components (one for for each chain).
##
## Reference:
## @Article{albertchib:jasa,
##   author =   {Albert, James A. and Siddhartha Chib},
##   title =   {Bayesian Analysis of Binary and Polychotomous
##                  Response Data},
```

```
##   journal =   jasa,
##   year =   1993,
##  volume =  88,
##  pages =  {669--79}
##}
##
## simon jackman, dept of political science, stanford university
## november 6, 1998 (denver airport...!)
#############################################################################

## a function for sampling  ystar | beta, X, y
## uses inverse-uniform trick
## from Gelfand et al (JASA 1990, p977), Devroye (1986), Greene (1997, 179)
rtnorm <- function(mu,y,n){
  u <- runif(n=n)                           # generate a uniform
  arg <- pnorm(-mu)                         # Phi(-xb)
  p <- rep(NA,n)
  p[y==0] <- (u*arg)[y==0]
  p[y==1] <- (arg + u*(1-arg))[y==1]
  z <- mu + qnorm(p)                        # qnorm is inverse-Phi
  z[z==Inf] <- 50.0                         # protection from silly start vals
  z[z==-Inf] <- -50.0                       # ditto
  z
}

## a function for sampling from the multivariate normal
## Ripley, _Stochastic Simulation_, p87; also see
## Venables & Ripley _MASS_.  This no longer necessary in Splus5
mvrnorm <- function(n = 1, mu, Sigma){
  p <- length(mu)
  if(!all.equal(dim(Sigma), c(p,p)))
    stop("incompatible arguments")
  eS <- eigen(Sigma, sym = T)
  if(!all(eS$values >= 0))
    stop("Sigma is not positive definite")
  X <- mu + eS$vectors %*% diag(sqrt(eS$values)) %*% matrix(rnorm(p*n),p)
  nm <- names(mu)
  if(is.null(nm) && !is.null(dn <- dimnames(Sigma)))
    nm <- dn[[1]]
  if(n == 1)
    drop(X)
  else {
    dimnames(X) <- list(nm, NULL)
    t(X)
  }
}

## the Gibbs sampling function (diffuse priors)
probitgibbs <- function(obj,maxiter=1000){
  x <- obj$x
  y <- obj$y

  xpx <- crossprod(x)
  ixpx <- solve(xpx)
  bmle <- coef(obj)                 # MLEs
```

```
#bmle <- ixpx%*%t(x)%*%y          # naive OLS
n <- length(y)                    # number obs
k <- length(bmle)                 # number predictors
chains <- 4                       # number chains (parallel Gibbs samplers)

## initialize output storage objects
output <- list()
for (i in 1:chains){
  output[[i]] <- matrix(NA,maxiter,1+k+3)
  dimnames(output[[i]]) <- list(NULL,
                                c("iter",
                                  names(coef(obj)),
                                  "mu","ystar","llh")
                                )
  output[[i]][1,1] <- 1
}
mu <- matrix(NA,n,chains)        # E(y^*) = mu = xb

kappa <- c(-3,3)
se <- summary.glm(obj)$coefficients[,2]

# permutation 1 of MLEs for start values
disperse <- rep(0,k)
disperse[1] <- kappa[1]
disperse[8] <- kappa[1]
disperse[6] <- kappa[1]
disperse[7] <- kappa[1]
output[[1]][1,2:(k+1)] <- bmle + disperse*se     # write to output object
mu[,1] <- x%*%(bmle + disperse*se)               # E(y^*) (starting values)
ystar <- rtnorm(mu[,1],y,n)                      # sample ystar
output[[1]][1,1+k+1] <- ystar[1000]
output[[1]][1,1+k+2] <- mu[1000,1]
Phi <- pnorm(mu[,1])                             # P(y=1 | X, y, beta^{1})
output[[1]][1,1+k+3] <- sum(y*log(Phi) + (1-y)*(log(1-Phi))) # llh

# permutation 2 of MLEs for start values
disperse <- rep(0,k)
disperse[6] <- kappa[1]
disperse[7] <- kappa[2]
disperse[1] <- kappa[1]
disperse[8] <- kappa[2]
output[[2]][1,2:(k+1)] <- bmle + disperse*se     # write to output object
mu[,2] <- x%*%(bmle + disperse*se)               # E(y^*) (starting values)
ystar <- rtnorm(mu[,2],y,n)                      # sample ystar
output[[2]][1,1+k+1] <- ystar[1000]
output[[2]][1,1+k+2] <- mu[1000,2]
Phi <- pnorm(mu[,1])                             # P(y=1 | X, y, beta^{1})
output[[2]][1,1+k+3] <- sum(y*log(Phi) + (1-y)*(log(1-Phi))) # llh

# permutation 3 of MLEs for start values
disperse <- rep(0,k)
disperse[6] <- kappa[2]
disperse[7] <- kappa[1]
disperse[1] <- kappa[2]
disperse[8] <- kappa[1]
```

```
  output[[3]][1,2:(k+1)] <- bmle + disperse*se     # write to output object
  mu[,3] <- x%*%(bmle + disperse*se)               # E(y^*) (starting values)
  ystar <- rtnorm(mu[,3],y,n)                      # sample ystar
  output[[3]][1,1+k+1] <- ystar[1000]
  output[[3]][1,1+k+2] <- mu[1000,3]
  Phi <- pnorm(mu[,1])                             # P(y=1 | X, y, beta^{1})
  output[[3]][1,1+k+3] <- sum(y*log(Phi) + (1-y)*(log(1-Phi))) # llh

  # permutation 4 of MLEs for start values
  disperse <- rep(0,k)
  disperse[6] <- kappa[2]
  disperse[7] <- kappa[2]
  disperse[1] <- kappa[2]
  disperse[8] <- kappa[2]
  output[[4]][1,2:(k+1)] <- bmle + disperse*se     # write to output object
  mu[,4] <- x%*%(bmle + disperse*se)               # E(y^*) (starting values)
  ystar <- rtnorm(mu[,4],y,n)                      # sample ystar
  output[[4]][1,1+k+1] <- ystar[1000]
  output[[4]][1,1+k+2] <- mu[1000,4]
  Phi <- pnorm(mu[,1])                             # P(y=1 | X, y, beta^{1})
  output[[4]][1,1+k+3] <- sum(y*log(Phi) + (1-y)*(log(1-Phi))) # llh

  ## commence Gibbs sampling, multiple chains
  for (iter in 2:maxiter){
    cat(paste("Gibbs sample number",iter,"\n"))
    for(chain in 1:chains){                         # loop over chains
      cat(paste("chain",chain,"\n"))
      ystar <- rtnorm(mu[,chain],y,n)               # sample ystar
      bstar <- ixpx%*%t(x)%*%ystar                  # E(beta | X, y, y^*)
      beta  <- mvrnorm(n=1,bstar,Sigma=ixpx)        # sample beta
      cat("beta:\n")
      print(beta)

      ## output iterative history
      output[[chain]][iter,1] <- iter
      output[[chain]][iter,2:(k+1)] <- beta
      output[[chain]][iter,1+k+1] <- ystar[1000]
      output[[chain]][iter,1+k+2] <- mu[1000,chain]
      mu[,chain] <- x%*%beta                        # E(y^* | X, y, beta^{t+1})
      Phi <- pnorm(mu[,chain])                      # P(y=1 | X, y, beta^{t+1})
      output[[chain]][iter,1+k+3] <- sum(y*log(Phi) + (1-y)*(log(1-Phi))) # llh
      cat(paste("llh:",output[[chain]][iter,1+k+3],"\n"))

    }
  }
  output
}

options(memory=1e+09,object.size=1e+08)

probitg2 <- probitgibbs(small.glm,maxiter=2000)

data.dump("probitg2","probitg2.dmp",oldStyle=T)
```

A.8 Graphing output of parallel Gibb samplers, probit model (Splus)

```
options(object.size=1e+08)
options(memory=200E+06)

attach("/home/simon/S/ellipse/.Data")     # attach ellipse routines

mybinaryllh <- function(obj){             # a function to calculate llh
  y <- obj$y
  p <- obj$fitted.values
  llh <- sum(y*log(p) + (1-y)*log(1-p))
  llh
}

mle.llh <- mybinaryllh(small.glm)


postscript(file="probitlook2.ps",
           width=7,height=7)
           #,horizontal=T) # open plotting device
par(mfrow=c(2,2),
    mar=c(4,4,1,2))
    #pty="s")                                      # page layout

## plot joint traces
xmin <- min(unlist(lapply(probitg2,function(z)min(z[,"south"]))))
xmax <- max(unlist(lapply(probitg2,function(z)max(z[,"south"]))))
ymin <- min(unlist(lapply(probitg2,function(z)min(z[,"govelec"]))))
ymax <- max(unlist(lapply(probitg2,function(z)max(z[,"govelec"]))))

plot(probitg2[[1]][1:50,"south"],
     probitg2[[1]][1:50,"govelec"],
     xlim=c(xmin,xmax),
     ylim=c(ymin,ymax),
     xlab="south",ylab="govelec",
     type="l")
lines(probitg2[[2]][1:50,"south"],probitg2[[2]][1:50,"govelec"])
lines(probitg2[[3]][1:50,"south"],probitg2[[3]][1:50,"govelec"])
lines(probitg2[[4]][1:50,"south"],probitg2[[4]][1:50,"govelec"])

## mark starting values
lapply(probitg2,
       function(z)
       points(z[1,"south"],z[1,"govelec"],pch=15)
       )

## text for traces
for (i in 1:4){
  text(probitg2[[i]][1,"south"] + .05*abs(diff(par()$usr[2:1])),
       probitg2[[i]][1,"govelec"] + .05*abs(diff(par()$usr[4:3])),
       as.character(i))
}

## mark MLE
points(coef(small.glm)["south"],
```

```
       coef(small.glm)["govelec"],
       col=6,pch=15)

## likelihood contours
lines(ellipse.glm(small.glm,which=c(6,7),level=.95),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.90),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.80),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.50),col=6,lwd=3)

## plot joint traces
xmin <- min(unlist(lapply(probitg2,function(z)min(z[,"(Intercept)"]))))
xmax <- max(unlist(lapply(probitg2,function(z)max(z[,"(Intercept)"]))))
ymin <- min(unlist(lapply(probitg2,function(z)min(z[,"closing"]))))
ymax <- max(unlist(lapply(probitg2,function(z)max(z[,"closing"]))))

plot(probitg2[[1]][1:50,"(Intercept)"],
     probitg2[[1]][1:50,"closing"],
     xlim=c(xmin,xmax),
     ylim=c(ymin,ymax),
     xlab="intercept",ylab="closing",
     type="l")
lines(probitg2[[2]][1:50,"(Intercept)"],probitg2[[2]][1:50,"closing"])
lines(probitg2[[3]][1:50,"(Intercept)"],probitg2[[3]][1:50,"closing"])
lines(probitg2[[4]][1:50,"(Intercept)"],probitg2[[4]][1:50,"closing"])

## mark starting values
lapply(probitg2,
       function(z)
       points(z[1,"(Intercept)"],z[1,"closing"],pch=15)
       )

## text for traces
for (i in 1:4){
  text(probitg2[[i]][1,"(Intercept)"] + .05*abs(diff(par()$usr[2:1])),
       probitg2[[i]][1,"closing"] + .05*abs(diff(par()$usr[4:3])),
       as.character(i))
}

## mark MLE
points(coef(small.glm)["(Intercept)"],
       coef(small.glm)["closing"],
       col=6,pch=15)

## likelihood contours
lines(ellipse.glm(small.glm,which=c(1,8),level=.95),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.90),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.80),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.50),col=6,lwd=3)

##############################################################################
## plot points from latter iterations
##############################################################################
##############################################################################
## panel 3
##############################################################################
```

```
xmin <- min(unlist(lapply(probitg2,function(z)min(z[,"south"]))))
xmax <- max(unlist(lapply(probitg2,function(z)max(z[,"south"]))))
ymin <- min(unlist(lapply(probitg2,function(z)min(z[,"govelec"]))))
ymax <- max(unlist(lapply(probitg2,function(z)max(z[,"govelec"]))))

plot(probitg2[[1]][,"south"],
     probitg2[[1]][,"govelec"],
     xlim=c(xmin,xmax),
     ylim=c(ymin,ymax),
     xlab="south",ylab="govelec",
     type="n")

for (i in 1:4){
  points(probitg2[[i]][1001:2000,"south"],
         probitg2[[i]][1001:2000,"govelec"],
         cex=.05,pch=16)
}

## mark MLE
points(coef(small.glm)["south"],
       coef(small.glm)["govelec"],
       col=6,pch=15)

## likelihood contours
lines(ellipse.glm(small.glm,which=c(6,7),level=.95),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.90),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.80),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(6,7),level=.50),col=6,lwd=3)

################################################################################
## panel 4
################################################################################
xmin <- min(unlist(lapply(probitg2,function(z)min(z[,"(Intercept)"]))))
xmax <- max(unlist(lapply(probitg2,function(z)max(z[,"(Intercept)"]))))
ymin <- min(unlist(lapply(probitg2,function(z)min(z[,"closing"]))))
ymax <- max(unlist(lapply(probitg2,function(z)max(z[,"closing"]))))

plot(probitg2[[1]][,"(Intercept)"],
     probitg2[[1]][,"closing"],
     xlim=c(xmin,xmax),
     ylim=c(ymin,ymax),
     xlab="intercept",ylab="closing",
     type="n")

for (i in 1:4){
  points(probitg2[[i]][1001:2000,"(Intercept)"],
         probitg2[[i]][1001:2000,"closing"],
         cex=.05,pch=16)
}

## mark MLE
points(coef(small.glm)["(Intercept)"],
       coef(small.glm)["closing"],
       col=6,pch=15)
```

```
## likelihood contours
lines(ellipse.glm(small.glm,which=c(1,8),level=.95),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.90),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.80),col=6,lwd=3)
lines(ellipse.glm(small.glm,which=c(1,8),level=.50),col=6,lwd=3)

closeDevice()
```

A.9 Binary Response Model (WinBUGS)

```
model{
  for (i in 1:N){                 ## loop over observations
    y[i] ~ dbern(p[i]);           ## binary outcome
    logit(p[i]) <- ystar[i];      ## logit link
    ystar[i] <- beta[1]           ## regression structure for covariates
              + beta[2]*educ[i]
              + beta[3]*(educ[i]*educ[i])
              + beta[4]*age[i]
              + beta[5]*(age[i]*age[i])
              + beta[6]*south[i]
              + beta[7]*govelec[i]
              + beta[8]*closing[i]
              + beta[9]*(closing[i]*educ[i])
              + beta[10]*(educ[i]*educ[i]*closing[i]);

    llh[i] <- y[i]*log(p[i]) + (1-y[i])*log(1-p[i]);   # llh contributions
  }

  sumllh <- sum(llh[]);       # sum of log-likelihood contributions

  ## priors
  beta[1:10] ~ dmnorm(mu[] , B[ , ]) ;    # diffuse multivariate Normal prior
                                          # see data file
}
```

A.10 Binary Response Model, truncated normal sampling (WinBUGS)

```
model{
  for (i in 1:N){                 ## loop over observations
     mu[i] <- beta[1]
    + beta[2]*educ[i]
    + beta[3]*(educ[i]*educ[i])
    + beta[4]*age[i]
    + beta[5]*(age[i]*age[i])
    + beta[6]*south[i]
    + beta[7]*govelec[i]
    + beta[8]*closing[i]
    + beta[9]*(closing[i]*educ[i])
    + beta[10]*(educ[i]*educ[i]*closing[i]);
```

```
    ## truncated normal sampling
    ystar[i] ~ dnorm(mu[i],1)I(lo[y[i]+1],up[y[i]+1]);

    probit(p[i]) <- ystar[i];    ## probs, as probit link
    llh[i] <- y[i]*log(p[i]) + (1-y[i])*log(1-p[i]);

  }
  ## truncation points
  lo[1] <- -50; lo[2] <- 0;   # ystar | y=0 ~ N(xb,1)I(-50,0)
  up[1] <-   0; up[2] <- 50;  # ystar | y=1 ~ N(xb,1)I(0,50)

  sumllh <- sum(llh[ ]);

  ## priors
  beta[1:10] ~ dmnorm(mu[] , B[ , ]) ;  ## multivariate Normal prior
}
```

A.11 Regression with AR(1) disturbances (Splus)

```
###############################################################################
## Gibbs sample regression model with AR(1) disturbances
##
## Reference: Chib (1993).
##
## Data are monthly observations of approval for Reagan
##
## simon jackman, dept of political science, stanford univ
## october 1998
###############################################################################
options(memory=200e+06,object.size=1e+09)

## a function for doing lags properly
mylag <- function(x, lag = 1)
{
  if(is.matrix(x)) {       # is x a matrix?
    n <- dim(x)[1]
    k <- dim(x)[2]
    z <- rep(NA, k)
    y <- x[1:(n - lag),  ]
    for(i in 1:lag)
      y <- rbind(z, y)
  }
  else {  ## assume x is a vector
    n <- length(x)
    y <- c(rep(NA, lag), x[1:(n - lag)])
  }
  y
}

## a function for transforming variables, by 1st diffs and rho
mystar <- function(x, rho){
  xstar <- x - (rho * mylag(x, 1))
  ## Prais-Winsten transformation for 1st obersvation
```

```
  if(is.matrix(x))                          ## is x a matrix?
    xstar[1,] <- sqrt(1 - (rho^2)) * x[1,]
  else
    xstar[1] <- sqrt(1 - (rho^2)) * x[1]    # PW transform
  xstar
}

## Gibbs sampler function
ar1gibbs <- function(obj,maxiter=10){
  y <- obj$y
  x <- obj$x
  n <- length(y)                          # number of observations
  k <- dim(x)[2]
  df <- n-k

  beta <- coef(obj)                       # starting values
  rho <- 0.0
  sigma2 <- crossprod(y-x%*%beta)/df
  cat(paste("starting values: sigma2=",sigma2,"\n"))
  cat(paste("starting values: rho   =",rho,"\n"))
  print(beta)

  ## priors
  a0 <- diag(rep(.001,k))                 # prior precision of beta
  b0 <- rep(0.0,k)                        # prior mean of beta
  ia0 <- solve(a0)                        # prior variance of beta
  v.0 <- -k                               # prior equiv number of obs
  sigma2.0 <- 0.0                         # prior sample variance
  delta.0 <- 0.0

  ## initialize output
  bout <- matrix(NA,maxiter,k)
  s2out <- rep(NA,maxiter)
  rhoout <- rep(NA,maxiter)
  llhout <- rep(NA,maxiter)
  nonstat <- 0

  for (iter in 1:maxiter){                      # commence Gibbs sampling
    ## sample rho
    u <- y - x%*%beta                           # regression residuals
    Phi <- sum(u[-1]^2)
    phi <- sum(u[-1]*u[-n])/Phi                 # E(rho | y, X, beta)
    flag <- 0
    repeat {                                    # sample until |rho|<1
      rho <- rnorm(1,mean=phi,sd=sqrt(sigma2/Phi))
      if (abs(rho)<1)
        break
      else
        flag <- 1
    }
    if (flag) nonstat <- nonstat + 1            # count instances non-stat

    ## sample beta
    ystar <- mystar(y,rho)                      # transform y
    xstar <- mystar(x,rho)                      # transform x
```

```
    xpy <- t(xstar)%*%ystar                      # cross-products
    xpx <- crossprod(xstar)                      # data precision
    ixpx <- solve(xpx)                           # data variance
    atilde <- a0 + xpx                           # posterior precision
    iatilde <- solve(atilde)                     # posterior variance
    btilde <- iatilde%*%(a0%*%b0 + xpy)          # posterior mean for beta
    beta <- mvrnorm(mu=btilde,                   # sample beta
    Sigma=as.numeric(sigma2)*iatilde)

    ## sample sigma2
    estar <- ystar - xstar%*%beta                # white noise residuals
    db <- crossprod(estar)                       # sum sq resids
    Qb <- t(beta-b0)%*%a0%*%(beta-b0)
    v <- v.0 + n + k                             # posterior df
    z <- Rgamma(n=1,                             # sample from Gamma
                r=v/2,
                lambda=(delta.0 + Qb + db)/2)
    sigma2 <- 1/z                                # sampled sigma2, IG

    ## log-likelihood
    llh <- -n/2*(log(2*pi) + log(sigma2)) + log(1-(rho^2))/2 - db/(2*sigma2)

    bout[iter,] <- beta                          # beta output
    s2out[iter] <- sigma2                        # sigma2 output
    rhoout[iter] <- rho                          # rho output
    llhout[iter] <- llh

    cat(paste("Iteration",iter,"sigma2 =",sigma2,"rho=",rho,"\n"))
  }                                              # end Gibbs sampler loop

  out <- list(b=bout,s2=s2out,rho=rhoout,llh=llhout,nonstat)
  out
}
##############################################################################
## end function definitions
##############################################################################

## run a regression to get start values, data in good shape, etc etc
reagan.temp <- lm(app~infl+unemp,
  x=T,y=T,data=reagan)

## call Gibbs sampler function
ar1gibbs.out <- ar1gibbs(reagan.temp,maxiter=5000)
```

A.12 Regression with AR(1) disturbances (WinBUGS)

```
###############################################################################
## AR(1) model for Reagan approval.
##
## simon jackman, dept of political science, stanford university
###############################################################################
model {
   # first-observation gets special treatment
  m <- sqrt(1-pow(rho,2));
```

```
  tau1 <- 1/((1-pow(rho,2))*sigma.e);
  mu[1] <- b[1] + b[2]*infl[1] + b[3]*unemp[1]
         + b[4]*dsp500[1] + b[5]*irancontrad[1];
  app[1] ~ dnorm(mu[1],tau1);

  for (t in 2:T){                      ## loop over obs 2 to T
    mu[t] <- b[1]*(1-rho)
    + b[2]*(infl[t] - (rho*infl[t-1]))
    + b[3]*(unemp[t] - (rho*unemp[t-1]))
    + b[4]*(dsp500[t] - (rho*dsp500[t-1]))
    + b[5]*(irancontrad[t] - (rho*irancontrad[t-1]))
    + rho*app[t-1];
    app[t] ~ dnorm(mu[t], tau.e);
  }

  sigma.e <- 1/sqrt(tau.e)           ## convert precision to variance

  ## priors
  rho ~ dunif(-1,1);                 ## uniform prior on stationary interval
  b[1:5] ~ dmnorm(b0[], B[ , ]);     ## multivariate Normal prior
  tau.e ~ dgamma(.05, .05);          ## vague prior on sigma
}

Initial values:

  list(b=c(20,-10,-10,-10, 5),rho=0,tau.e=.01)

  list(b=c(120,20,20,20,20),rho=.9,tau.e=.01)
```

A.13 Right-Censored Failure Times

```
#############################################################################
## right-censored failure time regression
##
## y ~ N(Xb, sigma^2)
## y|censoring ~ N(Xb,sigma^2)I(c,)
##
## data from Tanner 1996, Table 4.1
## Simon Jackman, Dept of Political Science, Stanford University
## August-December 1998
#############################################################################
model{
  for (i in 1:N.OK){                                  # loop over complete observations
    mu[i] <- b[1] + v[i]*b[2];
    t[i] ~ dnorm(mu[i],tau);                          # normal data and log-likelihood
    llh[i] <- -log(2*pi)/2 - log(sigma2)/2 - pow((t[i]-mu[i]),2)/(2*sigma2);
  }

  for (i in (N.OK+1):N){                              # loop over censored observations
    mu[i] <- b[1] + v[i]*b[2];
    t[i] ~ dnorm(mu[i],tau)I(c[i],);                # left-truncated Normal
    llh[i] <- log(1-phi((c[i]-mu[i])/sigma));  # incomplete data llh contrib
  }
```

```
  sumllh <- sum(llh[]);                                  # sum llh contributions

  ## priors
  # b[1:2] ~ dmnorm(b0[], Omega[ , ]);             # multivariate normal
  b0[1] <- 0.0; b0[2] <- 0.0;                     # uninformative priors
  Omega[1,1] <- .005;  Omega[1,2] <- 0.0;
  Omega[2,1] <- 0.0;   Omega[2,2] <- .005;
  tau ~ dgamma(.005,.005);                  # uninformative Gamma
  sigma2 <- 1/tau;                                    # variance is inverse precision
  sigma <- sqrt(sigma2);                          # std error

  ## to do (inefficient!) univariate sampling, comment out the references
  ## to b and Omega, above, and uncomment the following lines
  b[1] ~ dnorm(0,.005);
  b[2] ~ dnorm(0,.005);
}
```

A.14 Severe Missingness in Bivariate Normal Data (C)

```
/*****************************************************************************
 **
 **  routine to Gibbs sample from posterior for Sigma
 **  (inverse-Wishart sampling)
 **  with tanner51 data (Tanner, 3rd edition, Table 5.1)
 **
 ** compile as:
 **
 ** gcc -lm tanner51.c nrutil.o ranlib.o com.o linpack.o
 **        ludcmp.o lubksb.o
 **
 ** usage:
 **
 **  ./a.out <maxiter>
 **
 ** where maxiter is integer number of samples from posterior
 **
 ** simon jackman, dept of political science, stanford university
 ** 1998/99
 **
 *****************************************************************************/

#include <stdio.h>
#include <time.h>
#include <math.h>
#include "nr.h"
#include "nrutil.h"
#include "ranlib.h"

float **xprod(float **x, int n, int p);
float **inverse(float **x, int n);

/*****************************************************************************
 ** Inverse-Wishart sampling routine
 ** uses the approach described in Gelman et al, p480
```

```
 ** (1) takes raw X data, forms A = centered cross-products of X.
 ** (2) if A is k by k, and df degrees of freedom, then
 **      alpha[j,] ~ N(0,A^{-1}), j = 1, ... df
 ** (3) theta = alpha'alpha
 ** (4) return theta^{-1}
 ********************************************************************************/
float **simpwish(float **xstar, int df, int n, int k){
  int i,j,h,l,m;
  float **sigma, **a, **alpha, **sigmaout, *xrow, xp, *mu, *parm, *covm,
    *x, *work;

  a=matrix(1,k,1,k);
  alpha=matrix(1,df,1,k);
  xrow=vector(1,k);
  sigma=matrix(1,k,1,k);
  sigmaout=matrix(1,k,1,k);
  parm=vector(0,k*(k*3)/2 + 1);
  mu=vector(0,k-1);
  covm=vector(0,k*k - 1);
  x=vector(0,k-1);
  work=vector(0,k-1);

  sigma=xprod(xstar,n,k);    /* centered cross-products matrix */
  sigma=inverse(sigma,k);    /* invert */

  mu[0]=0.0; mu[1]=0.0;
  for(i=1,l=0;i<=k;i++){
    for(j=1;j<=k;j++){
      covm[l++]=sigma[i][j];
    }
  }

  setgmn(mu,covm,k,parm);          /* setup for multivariate Normal sampling */
  for(i=1;i<=df;i++) {
    genmn(parm,x,work);            /* multivariate Normal sampling */
    alpha[i][1]=x[0]; alpha[i][2]=x[1];
  }

  for (i=1;i<=df;i++){              /* form xproducts */
    xrow=alpha[i];
    for (j=0,l=1;l<=k;l++){
      xp=xrow[l];
      for (j++,h=0,m=1;m<=l;m++)
        sigmaout[j][++h] += xp*xrow[m];
    }
  }

  for (j=2;j<=k;j++)
    for (l=1;l<j;l++)
      sigmaout[l][j]=sigmaout[j][l];

  sigmaout=inverse(sigmaout,k);         /* invert */

  /* printmat(sigmaout,k,k); */
  return sigmaout;
```

```
}

/*************************************************************************
 ** matrix inversion routine from NR
 *************************************************************************/
float **inverse(float **x, int n)
{
  float **a, **y, d, *col;
  int i, j, *indx;

  col=vector(1,n);
  indx=ivector(1,n);
  y=matrix(1,n,1,n);
  a=matrix(1,n,1,n);

  for(i=1;i<=n;i++){
    for(j=1;j<=n;j++){
      y[i][j]=x[i][j];
      a[i][j]=x[i][j];
    }
  }

  /* fprintf(stdout,"inverse: calling dludcmp, matrix a = \n");
  printmat(a,n,n); */

  ludcmp(a,n,indx,&d);
  for (j=1;j<=n;j++) {
    for(i=1;i<=n;i++) col[i]=0.0;
    col[j]=1.0;
    lubksb(a,n,indx,col);
    for(i=1;i<=n;i++) y[i][j]=col[i];
  }

  return y;
}


/*************************************************************************
 ** cross-products routine
 *************************************************************************/
float **xprod(float **x, int n, int p)
{
  int i, j, k;
  float *mu, **xpx;

  mu = vector(1,p);
  xpx = matrix(1,p,1,p);

  for (j=1;j<=p;j++){
    mu[j]=0.0;
    for (k=1;k<=p;k++){
      xpx[j][k] = 0.0;           /* zero-out entries in sigma */
    }
  }
```

```
  for (i=1;i<=n;i++){
    for (j=1;j<=p;j++){
      mu[j] += x[i][j]/n;        /* calculate the means */
    }
  }

  for (i=1;i<=n;i++){            /* subtract out the means, form xprod */
    for (j=1;j<=p;j++){
      for (k=1;k<=p;k++){
xpx[j][k] += ((x[i][j] - mu[j]) * (x[i][k] - mu[k]));
      }
    }
  }
  return xpx;
}


/*****************************************************************************
 ** the main event...!
 *****************************************************************************/
int main(argc,argv)
     int argc;
     char *argv[ ];
{
  float **x, **xstar, **sigma, rho, sd1, sd2, m1, m2, **sigmanew, **isigma,
        **xpx, **ixpx;
  int i, j, k, l, n, iter, maxiter, df;
  long seed1, seed2;
  time_t nseconds, qseconds, pseconds;
  struct tm *ptr, *localtime();
  char *seedstr, *asctime();
  FILE *outfile, *rhofile;

  /*  C H E C K   A N D   O P E N   O U T P U T   F I L E S  */
  if((outfile = fopen("/home/simon/mcmc/tanner51/log", "w")) == NULL) {
    printf("cannot open log file\n");
    exit(1);
  }

  if((rhofile = fopen("/home/simon/mcmc/tanner51/rho.out", "w")) == NULL) {
    printf("cannot open output file for rho\n");
    exit(1);
  }

  maxiter = atoi(argv[1]);              /* arguments from user invocation */

  nseconds = time(NULL);                /* note start time for seeding rng */
  ptr=localtime(&nseconds);
  seedstr = asctime(ptr);
  fprintf(outfile,"Starting Execution at: %s",seedstr);

  fprintf(outfile,"Getting seeds from timestamp string\n");
  phrtsd(seedstr,&seed1,&seed2);
  fprintf(outfile,"Generated seeds: %12d %12d\n",seed1,seed2);
  setall(seed1,seed2);
```

```
  fprintf(outfile,"...seeded random number generators ok\n");

  /*  I N I T I A L I Z A T I O N  */
  n=12; k=2; df = 11;
  x=matrix(1,n,1,k);
  xstar=matrix(1,n,1,k);
  xpx=matrix(1,k,1,k);
  ixpx=matrix(1,k,1,k);

  /*  D A T A  */
  x[1][1] = 1.0;  x[1][2] =  1.0;
  x[2][1] = 1.0;  x[2][2] = -1.0;
  x[3][1] = -1.0; x[3][2] = 1.0;
  x[4][1] = -1.0; x[4][2] = -1.0;
  x[5][1] = 2.0;  x[5][2] = 9.0;              /* 9.0 is missing code */
  x[6][1] = 2.0;  x[6][2] = 9.0;
  x[7][1] = -2.0; x[7][2] = 9.0;
  x[8][1] = -2.0; x[8][2] = 9.0;
  x[9][1] = 9.0;  x[9][2] = 2.0;
  x[10][1] = 9.0; x[10][2] = 2.0;
  x[11][1] = 9.0; x[11][2] = -2.0;
  x[12][1] = 9.0; x[12][2] = -2.0;

  /*  S T A R T I N G   V A L U E S  */
  sigma = matrix(1,k,1,k);
  sigmanew = matrix(1,k,1,k);
  isigma = matrix(1,k,1,k);

  sigma[1][1]=1.0; sigma[2][2]=1.0;
  sigma[1][2]=0.0; sigma[2][1]=0.0;
  rho = sigma[1][2]/sqrt(sigma[1][1]*sigma[2][2]);
  sd1 = sqrt(sigma[1][1]*(1.0 - rho*rho));
  sd2 = sqrt(sigma[2][2]*(1.0 - rho*rho));
  m1 = rho*sqrt(sigma[1][1]/sigma[2][2]);
  m2 = rho*sqrt(sigma[2][2]/sigma[1][1]);

  /*  C O M M E N C E   G I B B S   S A M P L I N G  */
  for (iter=1;iter<=maxiter;iter++){           /* loop over samples */
    for(i=1;i<=n;i++){                         /* loop over observations */
      xstar[i][1] = x[i][1];
      xstar[i][2] = x[i][2];
      if (x[i][1]==9.0){                       /* x1 is missing, so sample */
xstar[i][1] = gennor(m1*x[i][2],sd1);
      }
      if (x[i][2]==9.0){
xstar[i][2] = gennor(m2*x[i][1],sd2);   /* x2 is missing, so sample */
      }
    }

    sigmanew=simpwish(xstar,df,n,k);             /* sample for Sigma */

    rho = sigmanew[1][2]/sqrt(sigmanew[1][1]*sigmanew[2][2]);
    sd1 = sqrt(sigmanew[1][1]*(1.0 - rho*rho));
    sd2 = sqrt(sigmanew[2][2]*(1.0 - rho*rho));
    m1 = rho*sqrt(sigmanew[1][1]/sigmanew[2][2]);
```

```
    m2 = rho*sqrt(sigmanew[2][2]/sigmanew[1][1]);

    fprintf(rhofile,"%14.8lf\n",rho);           /* output to file */
  }

  /*  E X I T   R O U T I N E */
  pseconds = time(NULL);
  ptr=localtime(&pseconds);
  seedstr = asctime(ptr);
  fprintf(outfile,"Ending Execution at: %s",seedstr);
  fprintf(outfile,
  "Execution took %6.0lf seconds\n",
  difftime(pseconds,nseconds));

  fclose(outfile);
  fclose(rhofile);

}
```

References

Albert, James H. and Siddhartha Chib. 1993. "Bayesian Analysis of Binary and Polychotomous Response Data." *Journal of the American Statistical Association* 88:669--79.

Barnett, Vic. 1982. *Comparative Statistical Inference*. Second ed. New York: Wiley.

Besag, J. 1974. "Spatial interaction and the statistical analysis of lattice systems (with discussion)." *Journal of the Royal Statistical Society, Series B* 41:143--168.

Brooks, Stephen P. and Andrew Gelman. 1998. "General Methods for Monitoring Convergence of Iterative Simulations." *Journal of Computational and Graphical Statistics* 7:434--455.

Chan, K.S. and Johannes Ledolter. 1995. "Monte Carlo EM Estimation for Time Series Models Involving Counts." *Journal of the American Statistical Association* 90:242--252.

Chib, Siddhartha. 1993. "Bayes regression with autoregressive errors: a Gibbs sampling approach." *Journal of Econometrics* 58:275--94.

Clifford, P. 1993. "Discussion on the Meeting on the Gibbs Sampler and Other Markov Chain Monte Carlo Methods." *Journal of the Royal Statistical Society, Series B* 55:53--102.

Cochrane, D. and G.H. Orcutt. 1949. "Application of Least Squares Relationships Containing Autocorrelated Error Terms." *Journal of the American Statistical Association* 44:32--61.

Cowles, Mary Kathryn and Bradley P. Carlin. 1996. "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review." *Journal of the American Statistical Association* 91:883--904.

Dempster, A. P., N. M. Laird and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the *EM* Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1--38.

di Finetti, B. 1974. *Theory of Probability*. Chichester: Wiley. Volume 1.

Gelfand, Alan E. and A. F. M. Smith. 1990. "Sampling based approaches to calculating marginal densities." *Journal of the American Statistical Association* 85:398--409.

Gelman, Andrew and Donald B. Rubin. 1992. "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Sciences* 7:457--511.

Gelman, Andrew and Gary King. 1990. "Estimating the Consequences of Electoral Redistricting." *Journal of the American Statistical Association* 85:274--82.

Gelman, Andrew, John B. Carlin, Hal S. Stern and Donald B. Rubin. 1995. *Bayesian Data Analysis*. London: Chapman and Hall.

Geman, S. and D. Geman. 1984. "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6:721--41.

Geweke, J. 1989. "Bayesian inference in econometric models using Monte Carlo integration." *Econometrica* 57:1317--1339.

Geweke, J. 1992. "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments (with discussion)." In *Bayesian Statistics 4*, ed. J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith. Oxford: Oxford University Press pp. 169--193.

Gilks, Walter R. 1996. "Full conditional distributions." In *Markov Chain Monte Carlo in Practice*, ed. W. R. Gilks, S. Richardson and D. J. Spiegelhalter. London: Chapman and Hall pp. 75--88.

Gorin, Zeev. 1980. "Income Inequality in the Marxist Theory of Development: A Cross-National Test." In *Comparative Social Research*, ed. Richard Tomasson. Greenwich, CT: 3A1.

Greene, William H. 1993. *Econometric Analysis*. Second ed. New York: Prentice-Hall.

Howson, Colin and Peter Urbach. 1993. *Scientific Reasoning: the Bayesian approach*. Second ed. Chicago: Open Court.

Jackman, Simon. 1994. "Measuring Electoral Bias: Australia, 1949--1993." *British Journal of Political Science* 24:319--57.

Jeffreys, H. 1961. *Theory of Probability*. Third ed. Oxford: Clarendon Press.

Johnson, Norman L., Samuel Kotz and Adrienne W. Kemp. 1992. *Univariate Discrete Distributions*. Second ed. New York: Wiley.

Johnson, Norman L., Samuel Kotz and N. Balakrishnan. 1994. *Continuous Univariate Distributions*. Vol. 1 second ed. New York: Wiley.

King, Gary. 1989. *Unifying Political Methodology*. New York: Cambridge University Press.

King, Gary. 1997. *A Solution to the Ecological Inference Problem*. Princeton: Princeton University Press.

Kolmogorov, A. N. 1956. *Information Theory and Statistics*. New York: Chelsea.

Kyburg, Henry E. and Howard E. Smokler. 1980. *Studies in Subjective Probability*. Second ed. Huntington, New York: Kreiger.

Lange, Peter and Geoffrey Garrett. 1987. "The Politics of Growth Reconsidered." *Journal of Politics* 49:257--74.

Leamer, Edward. 1978. *Specification Searches: Ad Hoc Inference with Nonexperimental Data*. New York: Wiley.

Leamer, Edward. 1983. "Let's Take the Con out of Econometrics." *American Economic Review* 23:31--43.

Lee, Peter M. 1989. *Bayesian Statistics: an introduction*. Oxford: Oxford University Press.

Little, Roderick J. A. and Donald B. Rubin. 1983. "On Jointly Estimating Parameters and Missing Data by Maximizing the Complete-Data Likelihood." *The American Statistician* 37:218--220.

Little, Roderick J. A. and Donald B. Rubin. 1987. *Statistical Analysis with Missing Data*. New York: Wiley.

Maddala, G. S. 1983. *Limited-dependent and Qualitative Variables in Econometrics*. New York: Cambridge University Press.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller. 1953. "Equations of state calculations by fast computing machines." *Journal of Chemical Physics* 21:1087--91.

Nagler, Jonathan. 1994. "Scobit: an alternative estimator to logit and probit." *American Journal of Political Science* 38:230--55.

Polson, Nicholas G. 1996. "Convergence of Markov Chain Monte Carlo Algorithms." In *Bayesian Statistics 5*, ed. J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F .M. Smith. Oxford: Oxford University Press pp. 763--773.

Prais, S. J. and C. B. Winsten. 1954. *Trend estimators and serial correlation*. Chicago: Cowles Commission.

Quandt, Richard E. and James B. Ramsey. 1973. "Estimating Mixtures of Normal Distributions and Switching Regressions." *Journal of the American Statistical Association* 73:730--738.

Raftery, Adrian E. 1996. "Hypothesis testing and model selection." In *Markov Chain Monte Carlo in Practice*, ed. W. R. Gilks, S. Richardson and D. J. Spiegelhalter. London: Chapman and Hall pp. 163--187.

Roberts, Gareth O. 1996. "Markov chain concepts related to sampling algorithms." In *Markov Chain Monte Carlo in Practice*, ed. W. R. Gilks, S. Richardson and D. J. Spiegelhalter. London: Chapman and Hall pp. 45--57.

Rosenthal, J. 1995. "Rates of convergence for Gibbs sampling for variance component models." *Annals of Statistics* 23:740--761.

Rubin, Donald B. 1987. *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley.

Spiegelhalter, David J., Andrew Thomas, Nicky Best and Wally R. Gilks. 1997. *BUGS 0.6: Bayesian inference using Gibbs sampling*. Cambridge, UK: MRC Biostatistics Unit.

Tanner, Martin A. 1996. *Tools for Statistical Inference: Methods for the Exploration of Postrior Distributions and Likelihood Functions*. Third ed. New York: Springer-Verlag.

Tanner, Martin and Wing Hung Wong. 1987. "The Calculation of Posterior Distributions by Data Augmentation." *Journal of the American Statistical Association* 82:528--40.

Tierney, Luke. 1997. "Markov Chain Monte Carlo Algorithms." In *Encyclopedia of the Statistical Sciences*, ed. Samuel Kotz, Campbell B. Read and David L. Banks. Vol. 1 (Update) New York: Wiley pp. 392--399.

Titterington, D. M, Adrian F. M. Smith and U. E. Makov. 1985. *Statistical Analysis of Finite Mixture Distributions*. New York: Wiley.

van Dantzig, D. 1957. "Statistical priesthood (Savage on personal probabilities)." *Statistica Neerlandica* 11:1--16.

von Mises, R. 1957. *Probability, Statistics, and Truth*. New York: Academic Press.

Western, Bruce. 1998. "Causal Heterogeneity in Comparative Research: A Bayesian Hierarchical Modelling Approach." *American Journal of Political Science* 42:1233--1259.

Western, Bruce and Simon Jackman. 1994. "Bayesian Inference for Comparative Research." *American Political Science Review* 88:412--23.

Wu, J.C.F. 1983. "On the Convergence Properties of the EM Algorithm." *The Annals of Statistics* 11:95--103.