Errata for *Computational Statistics*, 1st Edition, 3rd Printing

Geof H. Givens and Jennifer A. Hoeting

March 25, 2014

The final paragraph of the preface indicates if you own a 3rd printing copy.

Here is a list of corrections and other notes. We appreciate comments from our careful readers, including Jim Albert, Shan Ba, Jim Brennan, Shoja'eddin Chenouri, Hugh Chipman, Mark Delorey, Stephanie Fitchett, Doug Gorman, Andrew Hill, Michael Höhle, Quiming Huang, Mori Jamshidian, Yueyang Jiang, Wentao Li, Duncan Murdoch and Jason Song.

Corrections are offered in a fourth printing available very soon from Wiley.

Production Error:

- A sofware-related problem introduced by the publisher during the production phase rendered many expressions unreadable on the following pages: 7-8, 76, 171, 178, 216, 224, 232, 263, 272, 337, and 366. Error-free versions of these pages are given at the end of this document.

  If you own a 3rd printing, you may contact Wiley. If you are considering purchasing the book, you can be confident that Wiley will not send you a 3rd printing because they they have placed a hold on all 3rd printing copies remaining in their warehouse and they have initiated an immediate 4th printing.

Chapter 1:

- Page 9, first line below (1.25). Replace "is a convex function" with "is a strictly monotonic function".
- Page 11, two lines below equation (1.30), clearly $c = 1/\int f(\boldsymbol{\theta})L(\boldsymbol{\theta}|\mathbf{x})\, d\boldsymbol{\theta}$.
- Page 12, last paragraph above Example 1.2. The word *influence* is misspelled.

Chapter 2:

- Page 39 first paragraph of section 2.2.2.3 and page 40 first paragraph. Information is gained about the curvature of $\mathbf{g}$ not $\mathbf{g}'$.

Chapter 3:

- The AIC values in this chapter are 2 units too low.
- Section 3.2, last sentence of third paragraph. It is slightly clearer to say "If the neighborhood is defined by allowing as many as k changes to the current candidate solution in order to produce the next candidate, then it is a *k-neighborhood*, and the alteration of those features is called a *k-change*."

- Section 3.5.1.1, third sentence. Replace "a individual" with "an individual".
- Section 3.5.2.2, last sentence of third paragraph. Replace "Such an..." with "Such a..."
- Exercise 3.4. The steady state GA should have $G = 1/P$.

Chapter 4:

- Page 95, the first equation *below* (4.18), there is a log missing in the last term on the right hand side. In other words, the correct equation is $\mathrm{E}\{\log f_{\mathbf{X}}(\mathbf{x}|\boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta}^{(t)}\} = \mathrm{E}\{\log f_{\mathbf{Y}}(\mathbf{y}|\boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta}^{(t)}\} - \mathrm{E}\{\log f_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta}^{(t)}\}$
- Page 98, fifth line up from the bottom of example 4.4, the end of the line should be p_{C}, not p_c.
- Section 4.2.3. We have received the following email from Mori Jamshidian expressing his view of the SEM algorithm.

  > I'm using your text for my computational stats class, and it's been very good, especially in terms of the topics covered. When covering Chapter 4, Section 4.2.3 on EM variance estimation, I noticed that you cover SEM algorithm as one of the main algorithms for EM standard error estimation. In a paper that you have also cited in your book (Jamshidian and Jennrich 2000, JRSS-B) we have noted that SEM does not have a solid theoretical foundation, and have explained why it's prone to all sorts of numerical inaccuracies. Thus, we recommend that the SEM method not be used at all. You mention the method in Jamshidian and Jennrich (2000) as a "more sophisticated numerical differentiation strategy." It turns out that implementation of the methods in Jamshidian and Jennrich (2000) are much simpler than that of SEM, and as we show in our paper they result in highly accurate results. In our view, SEM is a somewhat unsuccessful attempt in using numerical differentiation in the context of EM, as we explain in our paper. Just thought to bring it up, in case you may find this useful for your future editions of the book.
- Page 102, third line of example 4.6 should be $\widehat{p}_{\mathrm{T}} = 0.0132$ not 0.132.
- Page 104, equation (4.49): Omit the δ_i from the denominator of this expression.
- Page 104, the line above equation (4.51) should begin "for $k = 1, \ldots, C$".
- Page 112, the line above equation (4.78) should begin "Finally, note that $\mathbf{b}^{(t)}, \ldots$".

Chapter 6:

- Page 150, example 6.2. Replace $\log \lambda \sim N(4, 0.5^2)$ with $\log \lambda \sim N(\log 4, 0.5^2)$.

Chapter 8:

- pge 237, equation (8.17), there should be a minus sign directly preceding the summation symbol.
- In the first bullet of exercise 8.5, the matrix should be 22×22, not 42×42.

Chapter 9:

- Section 9.2.4, last sentence of first paragraph. The bootstrap estimate of the bias is $\sum_{i=1}^{B}(\widehat{\theta}_i^* - \widehat{\theta})/B = \overline{\theta}^* - \widehat{\theta}$.
- Page 271, bottom paragraph. Delete the sentence beginning "For two-sided intervals..." and the clause after the colon in the subsequent sentence. The improvement offered by the nested bootstrap depends on the accuracy of the original interval and the type of interval. In general, nested bootstrapping can reduce the rate of convergence of coverage probabilities by an additional multiple of $n^{-1/2}$ or n^{-1}. See the cited references.
- Page 272, Example 9.9, fourth paragraph should begin "Let $t_2 \ldots$".

There are currently no other known errors in this printing.

The following pages are error-free versions of the pages damaged by a production snafu at Wiley.

Table 1.2 Notation and description for some common probability distributions of continuous random variables.

Name	Notation and Parameter Space	Density and Sample Space	Mean and Variance
Beta	$X \sim$Beta(α, β) $\alpha > 0$ and $\beta > 0$	$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ $0 \leq x \leq 1$	$\mathrm{E}\{X\} = \frac{\alpha}{\alpha+\beta}$ $\mathrm{var}\{X\} = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$
Cauchy	$X \sim$Cauchy(α, β) $\alpha \in \Re$ and $\beta > 0$	$f(x) = \frac{1}{\pi\beta\left[1+\left(\frac{x-\alpha}{\beta}\right)^2\right]}$ $x \in \Re$	$\mathrm{E}\{X\}$ is non-existent $\mathrm{var}\{X\}$ is non-existent
Chi-square	$X \sim \chi^2_\nu$ $\nu > 0$	$f(x) =$ Gamma$(\nu/2, 1/2)$ $x > 0$	$\mathrm{E}\{X\} = \nu$ $\mathrm{var}\{X\} = 2\nu$
Dirichlet	$\mathbf{X} \sim$Dirichlet$(\boldsymbol{\alpha})$ $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)$ $\alpha_i > 0$ $\alpha_0 = \sum_{i=1}^k \alpha_i$	$f(\mathbf{x}) = \frac{\Gamma(\alpha_0)\prod_{i=1}^k x_i^{\alpha_i-1}}{\prod_{i=1}^k \Gamma(\alpha_i)}$ $\mathbf{x} = (x_1, \ldots, x_k)$ and $0 \leq x_i \leq 1$ $\sum_{i=1}^k x_i = 1$	$\mathrm{E}\{\mathbf{X}\} = \boldsymbol{\alpha}/\alpha_0$ $\mathrm{var}\{X_i\} = \frac{\alpha_i(\alpha_0-\alpha_i)}{\alpha_0^2(\alpha_0+1)}$ $\mathrm{cov}\{X_i, X_j\} = \frac{-\alpha_i\alpha_j}{\alpha_0^2(\alpha_0+1)}$
Exponential	$X \sim$Exp(λ) $\lambda > 0$	$f(x) = \lambda \exp\{-\lambda x\}$ $x > 0$	$\mathrm{E}\{X\} = 1/\lambda$ $\mathrm{var}\{X\} = 1/\lambda^2$
Gamma	$X \sim$Gamma(r, λ) $\lambda > 0$ and $r > 0$	$f(x) = \frac{\lambda^r x^{r-1}}{\Gamma(r)} \exp\{-\lambda x\}$ $x > 0$	$\mathrm{E}\{X\} = r/\lambda$ $\mathrm{var}\{X\} = r/\lambda^2$

Table 1.3 Notation and description for more common probability distributions of continuous random variables.

Name	Notation and Parameter Space	Density and Sample Space	Mean and Variance
Lognormal	$X \sim$Lognormal(μ, σ^2) $\mu \in \Re$ and $\sigma > 0$	$f(x) = \frac{1}{x\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\left(\frac{\log\{x\}-\mu}{\sigma}\right)^2\right\}$ $x \in \Re$	$\mathrm{E}\{X\} = \exp\{\mu + \sigma^2/2\}$ $\mathrm{var}\{X\} = \exp\{2\mu + 2\sigma^2\} - \exp\{2\mu + \sigma^2\}$
Multivariate Normal	$\mathbf{X} \sim N_k(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k) \in \Re^k$ $\boldsymbol{\Sigma}$ positive definite	$f(\mathbf{x}) = \frac{\exp\{-(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})/2\}}{(2\pi)^{k/2}\lvert\boldsymbol{\Sigma}\rvert^{1/2}}$ $\mathbf{x} = (x_1, \ldots, x_k) \in \Re^k$	$\mathrm{E}\{\mathbf{X}\} = \boldsymbol{\mu}$ $\mathrm{var}\{\mathbf{X}\} = \boldsymbol{\Sigma}$
Normal	$X \sim$N(μ, σ^2) $\mu \in \Re$ and $\sigma > 0$	$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right\}$ $x \in \Re$	$\mathrm{E}\{X\} = \mu$ $\mathrm{var}\{X\} = \sigma^2$
Student's t	$X \sim t_\nu$ $\nu > 0$	$f(x) = \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)\sqrt{\pi\nu}}(1 + x^2/\nu)^{-(\nu+1)/2}$ $x \in \Re$	$\mathrm{E}\{X\} = 0$ if $\nu > 1$ $\mathrm{var}\{X\} = \frac{\nu}{\nu+2}$ if $\nu > 2$
Uniform	$X \sim$Unif(a, b) $a, b \in \Re$ and $a < b$	$f(x) = \frac{1}{b-a}$ $x \in [a, b]$	$\mathrm{E}\{X\} = (a + b)/2$ $\mathrm{var}\{X\} = (b - a)^2/12$
Weibull	$X \sim$Weibull(a, b) $a > 0$ and $b > 0$	$f(x) = abx^{b-1} \exp\{-ax^b\}$ $x > 0$	$\mathrm{E}\{X\} = \frac{\Gamma(1+1/b)}{a^{1/b}}$ $\mathrm{var}\{X\} = \frac{\Gamma(1+2/b)-\Gamma(1+1/b)^2}{a^{2/b}}$

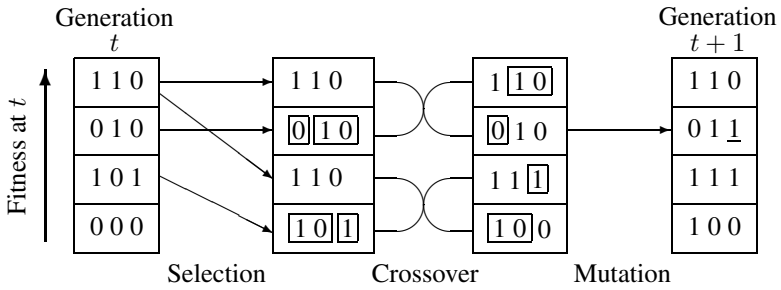

Fig. 3.6 An example of generation production in a genetic algorithm for a population of size $P = 4$ with chromosomes of length $C = 3$. Crossovers are illustrated by boxing portions of some chromosomes. Mutation is indicated by an underlined gene in the final column.

somes per individual and binary chromosome encoding. In generation t, individual '110' has the highest fitness among its generation and is chosen twice in the selection stage. In the crossover stage, the selected individuals are paired off so that each pair recombines to generate two new individuals. In the mutation stage, a low mutation rate is applied. In this example, only one mutation occurs. The completion of these steps yields the new generation.

Example 3.7 (Baseball salaries, continued) The results of applying a simple genetic algorithm to the variable selection problem for the baseball data introduced in Example 3.3 are shown in Figure 3.7. One hundred generations of size $P = 20$ were used. Binary inclusion–exclusion alleles were used for each possible predictor, yielding chromosomes of length $C = 27$. The starting generation consisted of purely random individuals. A rank-based fitness function was used; see equation (3.11) below. One parent was selected with probability proportional to this fitness; the other parent was selected independently, purely at random. Breeding employed simple crossover. A 1% mutation rate was randomly applied independently to each locus.

The horizontal axis in Figure 3.7 corresponds to generation. The AIC values for all twenty individuals in each generation are plotted. The best model found included predictors 2, 3, 6, 8, 10, 13, 14, 15, 16, 24, 25, and 26, yielding an AIC of -416.95. This matches the best model found using random starts local search (Table 3.2). Darwinian survival of the fittest is clearly illustrated in this figure: The twenty random starting individuals quickly coalesce into three effective subspecies, with the best of these slowly overwhelming the rest. The best model was first found in generation 87. □

3.5.1.3 Allele alphabets and genotypic representation The binary alphabet for alleles was introduced in the pioneering work of Holland [291] and continues to be very prevalent in recent research. The theoretical behavior of the algorithm and the relative performance of various genetic operators and other algorithmic variations are better understood for binary chromosomes than for other choices.

where F_j is the cumulative distribution function of each X_{ij} $(j = 1, \ldots, m)$ and the U_{ij} are independent Unif$(0, 1)$ random variables. Since F_j is a cumulative distribution function, its inverse is nondecreasing. Therefore, $h_1(F_1^{-1}(U_{i1}), \ldots, F_m^{-1}(U_{im}))$ is monotone in each U_{ij} for $j = 1, \ldots, m$ whenever h_1 is monotone in its arguments. Moreover, if $U_{ij} \sim$ Unif$(0, 1)$, then $1-U_{ij} \sim$ Unif$(0, 1)$. Hence, $h_2(1-\mathbf{U}_{i1}, \ldots, 1-\mathbf{U}_{im}) = h_1(F_1^{-1}(1 - U_{i1}), \ldots, F_m^{-1}(1 - U_{im}))$ is monotone in each argument and has the same distribution as $h_1(F_1^{-1}(U_{i1}), \ldots, F_m^{-1}(U_{im}))$. Therefore,

$$\hat{\mu}_2(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} h_1\left(F_1^{-1}(1 - U_{i1}), \ldots, F_m^{-1}(1 - U_{im})\right) \tag{6.55}$$

is a second estimator of μ having the same distribution as $\hat{\mu}_1(\mathbf{X})$. Our analysis above allows us to conclude that

$$\text{cov}\{\hat{\mu}_1(\mathbf{X}), \hat{\mu}_2(\mathbf{X})\} \leq 0. \tag{6.56}$$

Therefore, the estimator $\hat{\mu}_{\text{AS}} = (\hat{\mu}_1 + \hat{\mu}_2)/2$ will have smaller variance than $\hat{\mu}_1$ would have with a sample of size $2n$. Equation (6.49) quantifies the amount of improvement. We accomplish this improvement while generating only a single set of n random numbers, with the other n derived from the antithetic principle.

Example 6.6 (Normal expectation) Suppose X has a standard normal distribution and we wish to estimate $\mu = \text{E}\{h(X)\}$ where $h(x) = x/(2^x - 1)$. A standard Monte Carlo estimator can be computed as the sample mean of $n = 100,000$ values of $h(X_i)$ where $X_1, \ldots, X_n \sim$ i.i.d. $N(0, 1)$. An antithetic estimator can be constructed using the first $n = 50,000$ draws. The antithetic variate for X_i is simply $-X_i$, so the antithetic estimator is $\hat{\mu}_{\text{AS}} = \sum_{i=1}^{50,000} [h(X_i) + h(-X_i)]/\,100,000$. In the simulation, $\widehat{\text{cor}}\{h(X_i), h(-X_i)\} = -0.95$, so the antithetic approach is profitable. The standard approach yielded $\hat{\mu}_{\text{MC}} = 1.4993$ with a Monte Carlo standard error of 0.0016, whereas the antithetic approach gave $\hat{\mu}_{\text{AS}} = 1.4992$ with a standard error of 0.0003 (estimated via (6.49) using the sample variance and correlation). Further simulation confirms a more than fourfold reduction in standard error for the antithetic approach. □

Example 6.7 (Network failure probability, continued) Recalling Example 6.5, let the ith simulated network, $\mathbf{X}_i$, be determined by standard uniform random variables $U_{i1}, \ldots, U_{im}$, where $m = 20$. The jth edge in the ith simulated network is broken if $U_{ij} < p$. Now $h(\mathbf{X}_i) = h(U_{i1}, \ldots, U_{im})$ equals 1 if A and B are not connected, and 0 if they are connected. Note that h is nondecreasing in each U_{ij}; therefore the antithetic approach will be profitable. Since $\mathbf{X}_i$ is obtained by breaking the jth edge when $U_{ij} < p$ for $j = 1, \ldots, m$, the antithetic network draw, say $\mathbf{X}_i^*$, is obtained by breaking the jth edge when $U_{ij} > 1 - p$, for the same set of U_{ij} used to generate $\mathbf{X}_i$. The negative correlation induced by this strategy will ensure that $\frac{1}{2n}\left(\sum_{i=1}^{n} h(\mathbf{X}_i) + h(\mathbf{X}_i^*)\right)$ is a superior estimator to $\frac{1}{2n}\sum_{i=1}^{2n} h(\mathbf{X}_i)$. □

6.2 Consider the piecewise exponential envelopes for adaptive rejection sampling of the standard normal density, which is log-concave. For the tangent-based envelope, suppose you are limited to an even number of nodes at $\pm c_1, \ldots, \pm c_n$. For the envelope that does not require tangent information, suppose you are limited to an odd number of nodes at $0, \pm d_1, \ldots, \pm d_n$. The problems below will require optimization using strategies like those in Chapter 2.

(a) For $n = 1, 2, 3, 4, 5$, find the optimal placement of nodes for the tangent-based envelope.

(b) For $n = 1, 2, 3, 4, 5$, find the optimal placement of nodes for the tangent-free envelope.

(c) Plot these collections of envelopes; also plot rejection sampling waste against number of nodes for both envelopes. Comment on your results.

6.3 Consider finding $\sigma^2 = \mathrm{E}\{X^2\}$ when X has the density that is proportional to $q(x) = \exp\{-|x|^3/3\}$.

(a) Estimate σ^2 using importance sampling with standardized weights.

(b) Repeat the estimation using rejection sampling.

(c) Philippe and Robert describe an alternative to importance-weighted averaging that employs a Riemann sum strategy with random nodes [430, 431]. When draws $X_1, \ldots, X_n$ originate from f, an estimator of $\mathrm{E}\{h(X)\}$ is

$$\sum_{i=1}^{n-1} (X_{[i+1]} - X_{[i]}) h(X_{[i]}) f(X_{[i]}), \tag{6.77}$$

where $X_{[1]} \leq \cdots \leq X_{[n]}$ is the ordered sample associated with $X_1, \ldots, X_n$. This estimator has faster convergence than the simple Monte Carlo estimator. When $f = cq$ and the normalization constant c is not known, then

$$\frac{\sum_{i=1}^{n-1} (X_{[i+1]} - X_{[i]}) h(X_{[i]}) q(X_{[i]})}{\sum_{i=1}^{n-1} (X_{[i+1]} - X_{[i]}) q(X_{[i]})} \tag{6.78}$$

estimates $\mathrm{E}\{h(X)\}$, noting that the denominator estimates $1/c$. Use this strategy to estimate σ^2, applying it post hoc to the output obtained in part (b).

(d) Carry out a replicated simulation experiment to compare the performance of the two estimators in parts (b) and (c). Discuss your results.

6.4 Figure 6.10 shows some data on the number of coal-mining disasters per year between 1851 and 1962, available from the website for this book. These data originally appeared in [368] and were corrected in [306]. The form of the data we consider is given in [79]. Other analyses of these data include [378, 443].

where $i = 1, \ldots, I$, $j = 1, \ldots, J_i$, and $k = 1, \ldots, K$. After averaging over k for each i and j, we can rewrite the model (7.27) as

$$Y_{ij} = \mu + \alpha_i + \beta_{j(i)} + \epsilon_{ij}, \qquad i = 1, \ldots, I, \;\; j = 1, \ldots, J_i, \tag{7.28}$$

where $Y_{ij} = \sum_{k=1}^{K} Y_{ijk}/K$. Assume that $\alpha_i \sim N(0, \sigma_\alpha^2)$, $\beta_{j(i)} \sim N(0, \sigma_\beta^2)$, and $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$, where each set of parameters is independent a priori. Assume that σ_α^2, σ_β^2, and σ_ϵ^2 are known. To carry out Bayesian inference for this model, assume an improper flat prior for μ, so $f(\mu) \propto 1$. We consider two forms of the Gibbs sampler for this problem [463]:

(a) Let $n = \sum_i J_i$, $y_{\cdot\cdot} = \sum_{ij} y_{ij}/n$, and $y_{i\cdot} = \sum_j y_{ij}/J_i$ hereafter. Show that at iteration t, the conditional distributions necessary to carry out Gibbs sampling for this model are given by

$$\mu^{(t+1)} \big| \left(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)}, \mathbf{y}\right) \sim N\left(y_{\cdot\cdot} - \frac{1}{n}\sum_i J_i \alpha_i^{(t)} - \frac{1}{n}\sum_{j(i)} \beta_{j(i)}^{(t)}, \frac{\sigma_\epsilon^2}{n}\right),$$

$$\alpha_i^{(t+1)} \big| \left(\mu^{(t+1)}, \boldsymbol{\beta}^{(t)}, \mathbf{y}\right) \sim N\left(\frac{J_i V_1}{\sigma_\epsilon^2}\left(y_{i\cdot} - \mu^{(t+1)} - \frac{1}{J_i}\sum_j \beta_{j(i)}^{(t)}\right), V_1\right),$$

$$\beta_{j(i)}^{(t+1)} \big| \left(\mu^{(t+1)}, \boldsymbol{\alpha}^{(t+1)}, \mathbf{y}\right) \sim N\left(\frac{V_2}{\sigma_\epsilon^2}\left(y_{ij} - \mu^{(t+1)} - \alpha_i^{(t+1)}\right), V_2\right),$$

where $V_1 = \left(\frac{J_i}{\sigma_\epsilon^2} + \frac{1}{\sigma_\alpha^2}\right)^{-1}$ and $V_2 = \left(\frac{1}{\sigma_\epsilon^2} + \frac{1}{\sigma_\beta^2}\right)^{-1}$.

(b) The convergence rate for a Gibbs sampler can sometimes be improved via reparameterization. One approach to reparameterization is called hierarchical centering. For this model, hierarchical centering can be described as follows. Let Y_{ij} follow (7.28), but now let $\eta_{ij} = \mu + \alpha_i + \beta_{j(i)}$ and $\epsilon_{ij} \sim N\left(0, \sigma_\epsilon^2\right)$. Then let $\gamma_i = \mu + \alpha_i$ with $\eta_{ij}|\gamma_i \sim N\left(\gamma_i, \sigma_\beta^2\right)$ and $\gamma_i|\mu \sim N\left(\mu, \sigma_\alpha^2\right)$. As above, assume σ_α^2, σ_β^2, and σ_ϵ^2 are known, and assume a flat prior for μ. Show that the conditional distributions necessary to carry out Gibbs sampling for this model are given by

$$\mu^{(t+1)} \big| \left(\boldsymbol{\gamma}^{(t)}, \boldsymbol{\eta}^{(t)}, \mathbf{y}\right) \sim N\left(\frac{1}{I}\sum_i \gamma_i^{(t)}, \frac{1}{I}\sigma_\alpha^2\right),$$

$$\gamma_i^{(t+1)} \big| \left(\mu^{(t+1)}, \boldsymbol{\eta}^{(t)}, \mathbf{y}\right) \sim N\left(V_3\left(\frac{1}{\sigma_\beta^2}\sum_j \eta_{ij}^{(t)} + \frac{\mu^{(t+1)}}{\sigma_\alpha^2}\right), V_3\right),$$

$$\eta_{ij}^{(t+1)} \big| \left(\mu^{(t+1)}, \boldsymbol{\gamma}^{(t+1)}, \mathbf{y}\right) \sim N\left(V_2\left(\frac{y_{ij}}{\sigma_\epsilon^2} + \frac{\gamma_i^{(t+1)}}{\sigma_\beta^2}\right), V_2\right)$$

where $V_3 = \left(\frac{J_i}{\sigma_\beta^2} + \frac{1}{\sigma_\alpha^2}\right)^{-1}$.

8.2 REVERSIBLE JUMP MCMC

In Chapter 7 we considered MCMC methods for simulating $\mathbf{X}^{(t)}$ for $t = 1, 2, \ldots$ from a Markov chain with stationary distribution f. The methods described in Chapter 7 required that the dimensionality of $\mathbf{X}^{(t)}$ (i.e., of its state space) and the interpretation of the elements of $\mathbf{X}^{(t)}$ do not change with t. In many applications, it may be of interest to develop a chain that allows for changes in the dimension of the parameter space from one iteration to the next. Green's *reversible jump Markov chain Monte Carlo* (RJMCMC) method permits transdimensional Markov chain Monte Carlo simulation [243]. We discuss this approach below in the context of Bayesian model uncertainty. The full generality of RJMCMC is described in many of the references cited here.

Consider constructing a Markov chain to explore a space of candidate models, each of which might be used to fit observed data $\mathbf{y}$. Let $\mathcal{M}_1, \ldots, \mathcal{M}_K$ denote a countable collection of models under consideration. A parameter vector $\boldsymbol{\theta}_m$ denotes the parameters in the mth model. Different models may have different numbers of parameters, so we let p_m denote the number of parameters in the mth model. In the Bayesian paradigm, we may envision random variables $\mathbf{X} = (M, \boldsymbol{\theta}_M)$ which together index the model and parameterize inference for that model. We may assign prior distributions to these parameters, then seek to simulate from their posterior distribution using a MCMC method for which the tth random draw is $\mathbf{X}^{(t)} = \left(M^{(t)}, \boldsymbol{\theta}^{(t)}_{M^{(t)}}\right)$. Here $\boldsymbol{\theta}^{(t)}_{M^{(t)}}$, which denotes the parameters drawn for the model indexed by $M^{(t)}$, has dimension $p_{M^{(t)}}$ that can vary with t.

Thus, the goal of RJMCMC is to generate samples with joint posterior density $f(m, \boldsymbol{\theta}_m|\mathbf{y})$. This posterior arises from Bayes' theorem via

$$f(m, \boldsymbol{\theta}_m|\mathbf{y}) \propto f(\mathbf{y}|m, \boldsymbol{\theta}_m)\, f(\boldsymbol{\theta}_m|m)\, f(m), \tag{8.4}$$

where $f(\mathbf{y}|m, \boldsymbol{\theta}_m)$ denotes the density of the observed data under the mth model and its parameters, $f(\boldsymbol{\theta}_m|m)$ denotes the prior density for the parameters in the mth model, and $f(m)$ denotes the prior density of the mth model. A prior weight of $f(m)$ is assigned to the mth model so $\sum_{m=1}^{K} f(m) = 1$.

The posterior factorization

$$f(m, \boldsymbol{\theta}_m|\mathbf{y}) = f(m|\mathbf{y})\, f(\boldsymbol{\theta}_m|m, \mathbf{y}) \tag{8.5}$$

suggests two important types of inference. First, $f(m|\mathbf{y})$ can be interpreted as the posterior probability for the mth model, normalized over all models under consideration. Second, $f(\boldsymbol{\theta}_m|m, \mathbf{y})$ is the posterior density of the parameters in the mth model.

RJMCMC enables the construction of an appropriate Markov chain for $\mathbf{X}$ that jumps between models with parameter spaces of different dimensions. Like simpler MCMC methods, RJMCMC proceeds with the generation of a proposed step from the current value $\mathbf{x}^{(t)}$ to $\mathbf{X}^*$, and then a decision whether to accept the proposal or to keep another copy of $\mathbf{x}^{(t)}$. The stationary distribution for our chain will be the posterior in (8.5) if the chain is constructed so that

$$f(m_1, \boldsymbol{\theta}_{m_1}|\mathbf{y})a(m_2, \boldsymbol{\theta}_{m_2}|m_1, \boldsymbol{\theta}_{m_1}, \mathbf{y}) = f(m_2, \boldsymbol{\theta}_{m_2}|\mathbf{y})a(m_1, \boldsymbol{\theta}_{m_1}|m_2, \boldsymbol{\theta}_{m_2}, \mathbf{y})$$

$U^{(t+1)}$, but, more generally, chain transitions may be governed by a multivariate vector $\mathbf{U}^{(t+1)}$. We adopt the general case hereafter.

CFTP starts one chain from each state in the state space at some time $\tau < 0$ and transitions each chain forward using proposals generated by q. Proposals are accepted using the standard Metropolis–Hastings ratio. The goal is to find a starting time τ such that the chains have all coalesced by time $t = 0$ when run forwards in time from $t = \tau$. This approach provides $X^{(0)}$, which is a draw from the desired stationary distribution f.

The algorithm to find τ and thereby produce the desired chain is as follows. Let $X_k^{(t)}$ be the random state at time t of the Markov chain started in state k, with $k = 1, \ldots, K$.

1. Let $\tau = -1$. Generate $\mathbf{U}^{(0)}$. Start a chain in each state of the state space at time -1, namely $x_1^{(-1)}, \ldots, x_K^{(-1)}$, and run each chain forward to time 0 via the update $X_k^{(0)} = q\left(x_k^{(-1)}, \mathbf{U}^{(0)}\right)$ for $k = 1, \ldots, K$. If all K chains are in the same state at time 0, then the chains have coalesced and $X^{(0)}$ is a draw from f; the algorithm stops.

2. If the chains have not coalesced, then let $\tau = -2$. Generate $\mathbf{U}^{(-1)}$. Start a chain in each state of the state space at time -2, and run each chain forward to time 0. To do this, let $X_k^{(-1)} = q\left(x_k^{(-2)}, \mathbf{U}^{(-1)}\right)$. Next, you must reuse the $\mathbf{U}^{(0)}$ generated in step 1, so $X_k^{(0)} = q\left(X_k^{(-1)}, \mathbf{U}^{(0)}\right)$. If all K chains are in the same state at time 0, then the chains have coalesced and $X^{(0)}$ is a draw from f; the algorithm stops.

3. If the chains have not coalesced, move the starting time back to time $\tau = -3$ and update as above. We continue restarting the chains one step further back in time and running them forward to time 0 until we start at a τ for which all K chains have coalesced by time $t = 0$. At this point the algorithm stops. In every attempt, it is imperative that the random updating variables be reused. Specifically, when starting the chains at time τ, you must reuse the previously drawn random number updates $\mathbf{U}^{(\tau+1)}, \mathbf{U}^{(\tau+2)} \ldots, \mathbf{U}^{(0)}$. Also note that the same $\mathbf{U}^{(t)}$ vector is used to update all K chains at the tth iteration.

Propp and Wilson show that the value of $X^{(0)}$ returned from the CFTP algorithm for a suitable q is a realization of a random variable distributed according to the stationary distribution of the Markov chain and that this coalescent value will be produced in finite time [438]. Even if all chains coalesce before time 0, you must use $X^{(0)}$ as the perfect sampling draw; otherwise sampling bias is introduced.

Obtaining the perfect sampling draw $X^{(0)}$ from f is not sufficient for most uses. Typically we desire an i.i.d. n-sample from f, either for simulation or to use in a Monte Carlo estimate of some expectation, $\mu = \int h(x) f(x)\, dx$. A perfect i.i.d. sample from f can be obtained by running the CFTP algorithm n times to generate n individual values for $X^{(0)}$. If you only want to ensure that the algorithm is, indeed,

and $b = 0.00802$. The adjusted quantiles are therefore $\beta_1 = 0.038$ and $\beta_2 = 0.986$. The main effect of BC_a was therefore to shift the confidence interval slightly to the right. The resulting interval is $(-0.203, -0.172)$. □

9.3.2.2 The bootstrap t

Another approximate pivot that is quite easy to implement is provided by the *bootstrap* t method, also called the *studentized bootstrap* [153, 159]. Suppose $\theta = T(F)$ is to be estimated using $\widehat{\theta} = T(\widehat{F})$, with $V(\widehat{F})$ estimating the variance of $\widehat{\theta}$. Then it is reasonable to hope that $R(\mathcal{X}, F) = \frac{T(\widehat{F}) - T(F)}{\sqrt{V(\widehat{F})}}$ will be roughly pivotal. Bootstrapping $R(\mathcal{X}, F)$ yields a collection of $R(\mathcal{X}^*, \widehat{F})$.

Denote by $\widehat{G}$ and $\widehat{G}^*$ the distributions of $R(\mathcal{X}, F)$ and $R(\mathcal{X}^*, \widehat{F})$, respectively. By definition, a $1 - \alpha$ confidence interval for θ is obtained from the relation

$$\begin{aligned} &P\big[\xi_{\alpha/2}(\widehat{G}) \le R(\mathcal{X}, F) \le \xi_{1-\alpha/2}(\widehat{G})\big] \\ &\qquad = P\left[\widehat{\theta} - \sqrt{V(\widehat{F})}\xi_{1-\alpha/2}(\widehat{G}) \le \theta \le \widehat{\theta} - \sqrt{V(\widehat{F})}\xi_{\alpha/2}(\widehat{G})\right] \\ &\qquad = 1 - \alpha, \end{aligned}$$

where $\xi_\alpha(\widehat{G})$ is the α quantile of $\widehat{G}$. These quantiles are unknown, because F (and hence $\widehat{G}$) is unknown. However, the bootstrap principle implies that the distributions $\widehat{G}$ and $\widehat{G}^*$ should be roughly equal, so $\xi_\alpha(\widehat{G}) \approx \xi_\alpha(\widehat{G}^*)$ for any α. Thus, a bootstrap confidence interval can be constructed as

$$\left(T(\widehat{F}) - \sqrt{V(\widehat{F})}\xi_{1-\alpha/2}(\widehat{G}^*), \quad T(\widehat{F}) - \sqrt{V(\widehat{F})}\xi_{\alpha/2}(\widehat{G}^*)\right), \tag{9.15}$$

where the percentiles of $\widehat{G}^*$ are taken from the histogram of bootstrap values of $R(\mathcal{X}^*, \widehat{F})$. Since these are percentiles in the tail of the distribution, at least several thousand bootstrap pseudo-datasets are needed for adequate precision.

Example 9.7 (Copper–nickel alloy, continued) Continuing the copper–nickel alloy regression problem introduced in Example 9.3, an estimator $V(\widehat{F})$ of the variance of $\widehat{\beta}_1/\widehat{\beta}_0$ based on the delta method is

$$\left(\frac{\widehat{\beta}_1}{\widehat{\beta}_0}\right)^2 \left(\frac{\widehat{\text{var}}\{\widehat{\beta}_1\}}{\widehat{\beta}_1^2} + \frac{\widehat{\text{var}}\{\widehat{\beta}_0\}}{\widehat{\beta}_0^2} - \frac{2\ \widehat{\text{cov}}\{\widehat{\beta}_0, \widehat{\beta}_1\}}{\widehat{\beta}_0\widehat{\beta}_1}\right), \tag{9.16}$$

where the estimated variances and covariance can be obtained from basic regression results. Carrying out the bootstrap t method then yields the histogram shown in Figure 9.2, which corresponds to $\widehat{G}^*$. The 0.025 and 0.975 quantiles of $\widehat{G}^*$ are -5.77 and 4.44, respectively, and $\sqrt{V(\widehat{F})} = 0.00273$. Thus, the 95% bootstrap t confidence interval is $(-0.197,\ -0.169)$. □

This method requires an estimator of the variance of $\widehat{\theta}$, namely $V(\widehat{F})$. If no such estimator is readily available, a delta method approximation may be used [122].

9.7 PERMUTATION TESTS

There are other important techniques aside from the bootstrap that share the underlying strategy of basing inference on "experiments" within the observed dataset. Perhaps the most important of these is the classic permutation test that dates back to the era of Fisher [170] and Pitman [433, 434]. Comprehensive introductions to this field include [150, 240, 372]. The basic approach is most easily explained through a hypothetical example.

Example 9.9 (Comparison of independent group means) Consider a medical experiment where rats are randomly assigned to treatment and control groups. The outcome X_i is then measured for the ith rat. Under the null hypothesis, the outcome does not depend on whether a rat was labeled as treatment or control. Under the alternative hypothesis, outcomes tend to be larger for rats labeled as treatment.

A test statistic T measures the difference in outcomes observed for the two groups. For example, T might be the difference between group mean outcomes, having value t_1 for the observed dataset.

Under the null hypothesis, the individual labels "treatment" and "control" are meaningless, because they have no influence on the outcome. Since they are meaningless, the labels could be randomly shuffled among rats without changing the joint null distribution of the data. Shuffling the labels creates a new dataset: Although one instance of each original outcome is still seen, the outcomes appear to have arisen from a different assignment of treatment and control. Each of these permuted datasets is as likely to have been observed as the actual dataset, since the experiment relied on random assignment.

Let t_1 be the value of the test statistic computed from the dataset with this first permutation of labels. Suppose all M possible permutations (or a large number of randomly chosen permutations) of the labels are examined, thereby obtaining $t_2, \ldots, t_M$.

Under the null hypothesis, $t_2, \ldots, t_M$ were generated from the same distribution that yielded t_1. Therefore, t_1 can be compared to the empirical quantiles of $t_1, \ldots, t_M$ to test a hypothesis or construct confidence limits. □

To pose this strategy more formally, suppose that we observe a value t for a test statistic T having density f under the null hypothesis. Suppose large values of T indicate that the null hypothesis is false. Monte Carlo hypothesis testing proceeds by generating a random sample of $M - 1$ values of T drawn from f. If the observed value t is the kth largest among all M values, then the null hypothesis is rejected at a significance level of k/M. If the distribution of the test statistic is highly discrete, then ties found when ranking t can be dealt with naturally by reporting a range of p-values. Barnard [20] posed the approach in this manner; interesting extensions are offered in [32, 33].

There are a variety of approaches for sampling from the null distribution of the test statistic. The permutation approach described in Example 9.9 works because "treatment" and "control" are meaningless labels assigned completely at random and

residuals $\left|Y_i - \hat{s}_j^{(i)}(x_i)\right|$ for the three different smooths. The curves in this figure represent $\hat{p}(h_j, x_i)$ for $j = 1, 2$, and 3. The data used in each smooth originate as residuals from alternative smooths using spans of $0.05n$ (dashed), $0.2n$ (dotted), and $0.5n$ (solid), but each set of absolute residuals is smoothed with a span of $0.2n$ to generate the curves shown.

At each x_i, the performances of the three smooths can be assessed using $\hat{p}(h_j, x_i)$ for $j = 1, 2$, and 3. Denote by $\hat{h}_i$ the best of these spans at x_i, that is, the particular span among h_1, h_2, and h_3 that provides the lowest $\hat{p}(h_j, x_i)$. Figure 11.14 plots $\hat{h}_i$ against x_i for our example. The best span can vary abruptly even for adjacent x_i, so next the data in Figure 11.14 are passed through a fixed-span smoother (say, $\hat{s}_2$) to estimate the optimal span as a function of x. Denote this smooth as $\hat{h}(x)$. Figure 11.14 also shows $\hat{h}(x)$.

Now we have the original data and a notion of the best span to use for any given x: namely $\hat{h}(x)$. What remains is to create a final, overall smooth. Among several strategies that might be employed at this point, [180] recommends setting $\hat{s}(x_i)$ equal to a linear interpolation between $\hat{s}_{h^-(x_i)}(x_i)$ and $\hat{s}_{h^+(x_i)}(x_i)$, where among the m fixed spans tried, $h^-(x_i)$ is the largest span less than $\hat{h}(x_i)$ and $h^+(x_i)$ is the smallest span that exceeds $\hat{h}(x_i)$. Thus,

$$\hat{s}(x_i) = \frac{\hat{h}(x_i) - h^-(x_i)}{h^+(x_i) - h^-(x_i)} \hat{s}_{h^+(x_i)}(x_i) + \frac{h^+(x_i) - \hat{h}(x_i)}{h^+(x_i) - h^-(x_i)} \hat{s}_{h^-(x_i)}(x_i). \qquad (11.35)$$

Figure 11.15 shows the final result. The supersmoother adjusted the span wisely, based on the local variability of the data. In comparison, the spline smooth shown in this figure undersmoothed the left side and oversmoothed the right side, for a fixed λ chosen by cross-validation.

Although the supersmoother is a nonlinear smoother, it is very fast compared to most other nonlinear smoothers, including loess.

11.5 CONFIDENCE BANDS

Producing reliable confidence bands for smooths is not straightforward. Intuitively, what is desired is an image that portrays the range and variety of smooth curves that might plausibly be obtained from data like what we observed. Bootstrapping (Chapter 9) provides a method for avoiding parametric assumptions, but it does not help clarify exactly what sort of region should be graphed.

Consider first the notion of a *pointwise confidence band*. Bootstrapping the residuals would proceed as follows. Let $\mathbf{e}$ denote the vector of residuals (so $\mathbf{e} = (\mathbf{I} - \mathbf{S})\mathbf{Y}$ for a linear smoother). Sample the elements of $\mathbf{e}$ with replacement to generate bootstrapped residuals $\mathbf{e}^*$. Add these to the fitted values to obtain bootstrapped responses $\mathbf{Y}^* = \widehat{\mathbf{Y}} + \mathbf{e}^*$. Smooth $\mathbf{Y}^*$ over $\mathbf{x}$ to generate a bootstrapped fitted smooth, $\hat{\mathbf{s}}^*$. Start anew and repeat the bootstrapping many times. Then, for each x in the dataset, a bootstrap confidence interval for $\hat{s}(x)$ can be generated using the percentile method (Section 9.3.1) by deleting the few largest and smallest bootstrap fits at that point. If

node will be selected. If T_0 has $q(T_0)$ terminal nodes, then there are at most $q(T_0)$ subtrees that can be obtained by choosing different values of α.

The best approach for selecting the value for the parameter α in (12.21) relies on cross-validation. The dataset is partitioned into V separate portions of equal size, where V is typically between 3 and 10. For a finite sequence of α values, the algorithm proceeds as follows:

1. Remove one of the V parts of the dataset. This subset is called the validation set.

2. Find the optimal subtree for each value of α in the sequence using the remaining $V - 1$ parts of the data.

3. For each optimal subtree, predict the validation-set responses, and compute the cross-validated sum of squared error based on these validation-set predictions.

Repeat this process for all V parts of the data. For each α, compute the total cross-validated sum of squares over all V data partitions. The value of α that minimizes the cross-validated sum of squares is selected; call it $\hat{\alpha}$. Having estimated the best value for the complexity parameter, we may now prune the full tree for all the data back to the subtree determined by $\hat{\alpha}$.

Efficient algorithms for finding the optimal tree for a sequence of α values (see step 2 above) are available [59, 457]. Indeed, the set of optimal trees for a sequence of α values is nested, with smaller trees corresponding to larger values of α, and all members in the sequence can be visited by sequential recombination of terminal nodes from the bottom up. Various enhancements of this cross-validation strategy have been proposed, including a variant of the above approach that chooses the simplest tree among those trees that nearly achieve the minimum cross-validated sum of squares [533].

Example 12.6 (Stream monitoring, continued) Let us return to the stream ecology example introduced in Example 12.4. A full tree for these data was obtained by splitting until every terminal node has fewer than 10 observations in it or has residual squared error less 1% of the residual squared error for the root node. This process produced a full tree with 53 terminal nodes. Figure 12.10 shows the total cross-validated residual squared error as a function of the number of terminal nodes. This plot was produced using 10-fold cross-validation ($V = 10$). The full tree can be pruned from the bottom up, recombining the least beneficial terminal nodes, until the minimal value of $R_\alpha(T)$ is reached. Note that the correspondence between values of α and tree sizes means that we need only consider a limited collection of α values, and it is therefore more straightforward to plot $R_\alpha(T)$ against $q(T)$ instead of plotting against α. The minimal cross-validated sum of squares is achieved for a tree with five terminal nodes; indeed, this is the tree shown in Figure 12.7.

For this example, the selection of the optimal α, and thus the final tree, varies with different random partitions of the data. The optimal tree typically has between three and thirteen terminal nodes. This uncertainty emphasizes the potential structural