Getting Started with the MCMC Procedure

Funda Gunes and Fang Chen, SAS Institute Inc.

Abstract

With the computing advances of recent decades, Bayesian modeling has been growing in popularity in a variety of scientific fields. The powerful MCMC procedure offers a flexible model-building environment and intuitive syntax that enable you to fit a wide range of Bayesian models. This paper presents a series of examples that show how to use PROC MCMC to fit Bayesian models.

Introduction

The MCMC procedure was first introduced in SAS/STAT 9.2 and has been continuously enhanced with additional features. In SAS/STAT 13.1, PROC MCMC enables you to fit a wide range of statistical models, including linear, generalized linear, and nonlinear models; nested or nonnested random-effects models; and hierarchical models. PROC MCMC also enables you to handle various missing data problems, perform variable selection, and make predictions.

In Bayesian inference, parameters are random variables about which you make probability statements. Bayesian analysis summarizes all the uncertainty about the model parameters through probability distributions. First, you select a model that describes the distribution of the data given the model parameters (likelihood function or the sampling distribution) and you choose prior distributions for the parameters in the model. Then, you update your beliefs about the model parameters by combining the information from the data and the prior distributions to obtain the posterior distribution.

The basic characteristic of the MCMC procedure is its use of Markov chain Monte Carlo methods for drawing sequentially from a target posterior distribution. PROC MCMC generates a large number of samples from the posterior distribution, and then it uses these samples to empirically estimate the posterior distributions of the model parameters.

Using PROC MCMC for Bayesian inference is straightforward. All you need to do is to specify your model, which includes the likelihood function for the data and the prior distributions for the model parameters. PROC MCMC samples from the target posterior distribution. The conditional posterior distributions are automatically derived. PROC MCMC supports many built-in distributions (such as the normal, t, binomial, and univariate truncated distributions) and multivariate distributions (such as multivariate normal and inverse Wishart). Like the NLMIXED procedure, PROC MCMC executes SAS DATA step language, which you use to define nonstandard distributions or functions of parameters that you might be interested in.

One disadvantage of Bayesian analysis is that simulation can be time-consuming. PROC MCMC is equipped with a number of features in order to facilitate performance. For example, beginning with SAS/STAT 13.1, PROC MCMC is multithreaded, taking advantage of all the cores available on your machine.

In addition to the MCMC procedure, SAS/STAT provides built-in Bayesian analysis in four procedures (LIFEREG, LIFETEST, GENMOD, and FMM). These procedures offers you convenient access to Bayesian modeling and inference for generalized linear models, accelerated life failure models, Cox regression models, and piecewise exponential hazard models. They provide a standard set of priors that are appropriate for these analysis. For more information about the built-in Bayesian analyses see Stokes (2014). Furthermore, the new BCHOICE procedure performs Bayesian analysis for discrete choice models, which are commonly used in marketing research to model decision makers' choices among alternative products and services.

This paper first outlines the steps that should be followed in a Bayesian analysis and introduces commonly used MCMC statements. Then it provides several basic examples to illustrate how you can use the MCMC procedure in your work. The first example is a simple linear regression example that illustrates the key features of the procedure. The second example shows how to fit various generalized linear models and illustrates how to use SAS DATA step programming within the MCMC procedure. The third example shows how to fit a nonlinear Poisson regression model and how to use change-of-variables to improve Markov chain mixing. The last example applies hierarchical Bayesian modeling to a meta-analysis study. Additional resources are also provided.

Steps in a Bayesian Analysis

Ideally, Bayesian analysis includes the following steps:

1. Select a probability model and priors for the model parameters.
2. Obtain the estimates of the posterior distribution.
3. Carry out convergence diagnostics by using various plots and convergence tests. If convergence is not achieved, try tricks (such as transforming data, or using change of variables) and check the convergence again. Before you proceed to the next step, you should obtain well-mixing converging chains for all parameters, not just those of interest.
4. Check the fit of the model and evaluate the sensitivity of the results to the priors used.
5. Review the posterior summary and interval statistics. You can also carry out further analysis, such as comparing different models for prediction and inferences.

For an overview of Bayesian statistics see the chapter "Introduction to Bayesian Analysis Procedures" in the *SAS/STAT User's Guide*.

PROC MCMC Statements

The following syntax shows the commonly used PROC MCMC statements:

```
PROC MCMC options;
   PARMS;
   PRIOR;
   Programming statements;
   MODEL;
   RANDOM;
   PREDDIST;
Run;
```

A simple call to PROC MCMC often include the PROC MCMC, PARMS, PRIOR and the MODEL statements. The other frequently used statements are RANDOM and PREDDIST.

You can include options in the PROC MCMC statement to specify the Markov chain simulation details and to control posterior calculation, convergence diagnostics, and plotting. The PARMS statement specifies the model parameters, and the PRIOR statement specifies their prior distributions. The MODEL statement and the programing statements jointly define the likelihood function for each observation.

The RANDOM statement offers a simple way to specify the random-effects parameters in a model. Example 4 illustrates the use of the RANDOM statement. The PREDDIST statement enables you to draw random samples from the posterior predictive distribution of the response variable. This is especially useful for the prediction purposes and for checking whether the fitted model is consistent with the data.

Example 1: Simple Linear Regression Model

This example illustrates the key features of the procedure by showing how you can use the MCMC procedure to fit a simple linear regression model. The following statements create the Class data set, which includes the variables **height** and **weight**:

```
data Class;
   input height weight;
datalines;
69.0 112.5
56.5 84.0

   ... more lines ...
```

```
57.5 85.0
66.5 112.0
;
```

Suppose you want to predict weight from height by using a simple linear regression model, which is specified as follows:

$$\begin{aligned} \text{weight}_i &\sim \text{normal}(\mu,\ \text{var}=\sigma^2) \\ \mu_i &= \beta_0 + \beta_1 height_i, \text{where } i = 1,\ldots,n \end{aligned}$$

The model parameters are the regression coefficients, β_0 and β_1, and the variance parameter, σ^2. They are given the following priors:

$$\begin{aligned} \beta_0, \beta_1 &\sim \text{normal}(0,\ \text{var}=100) \\ \sigma^2 &\sim \text{igamma}(\text{shape}=2,\ \text{scale}=2) \end{aligned}$$

where igamma stands for the inverse-gamma distribution.

The following statements fit the specified model:

```
proc mcmc data=SasHelp.class seed=1 nbi=5000 nmc=10000 outpost=regOut;
   parms beta0 beta1 sigma2;
   prior beta: ~ normal(0, var=100);
   prior sigma2 ~ igamma(shape=2, scale=2);
   mu = beta0+beta1*height;
   model weight ~ normal(mu, var=sigma2);
run;
```

Each of these statements is explained as follows. As for any other SAS procedure, you begin coding by specifying the name of the procedure and the data set.

```
proc mcmc data=SasHelp.class seed=1 nbi=5000 nmc=10000 outpost=regOut;
```

The SEED= option sets the random number seed to ensure Markov chain reproducibility. The NBI= option specifies the number of burn-in iterations, and the NMC= option controls the number of Markov chain iterations after burn-in. The OUTPOST= option saves all posterior samples to the output data set regOut.

It is often easiest to begin by specifying the likelihood function. The MODEL statement specifies the likelihood function for the response variable **weight**.

```
model weight ~ normal(mu, var=sigma2);
```

The $\sim$ sign indicates that a specification of a distribution comes next. In this case, the likelihood has a normal distribution with mean μ and variance σ^2, where the names “mu” and “sigma” are your choice. Next, you can use a programming statement to define the regression mean μ, which is a function of the regression coefficients, β_0 and β_1:

```
mu = beta0+beta1*height;
```

Next, you use the PARMS statement to tell PROC MCMC what names you have created for the model parameters (in this example, beta0, beta1, sigma2):

```
parms beta0 beta1 sigma2;
```

You can use multiple PARMS statements, where each statement defines a block of parameters that are updated sequentially in each iteration. Specifying multiple PARMS statement is convenient if you are working with large number of parameters.

The next step is to specify the prior distributions for the model parameters in the PRIOR statements:

```
prior beta: ~ normal(0, var=100);
prior sigma2 ~ igamma(shape=2, scale=2);
```

The colon after **beta** indicates that symbols in the program that start with **beta** (beta0 and beta1 in this example) have normal priors with mean 0 and variance 100. The variance parameter **sigma2** is given an inverse-gamma prior, and the value of both the shape and scale parameters is 2. This completes the specification of the model. Because the symbol **mu** is used in the MODEL statement, the programing statements that define **mu** must appear before the MODEL statement.

The results of this analysis are shown next. Figure 1 shows the number of observations in the input data and the number of observations used in the analysis. In this case, the SasHelp.class data set has no missing value.

Figure 1 Observation Information

Number of Observations Read	19
Number of Observations Used	19

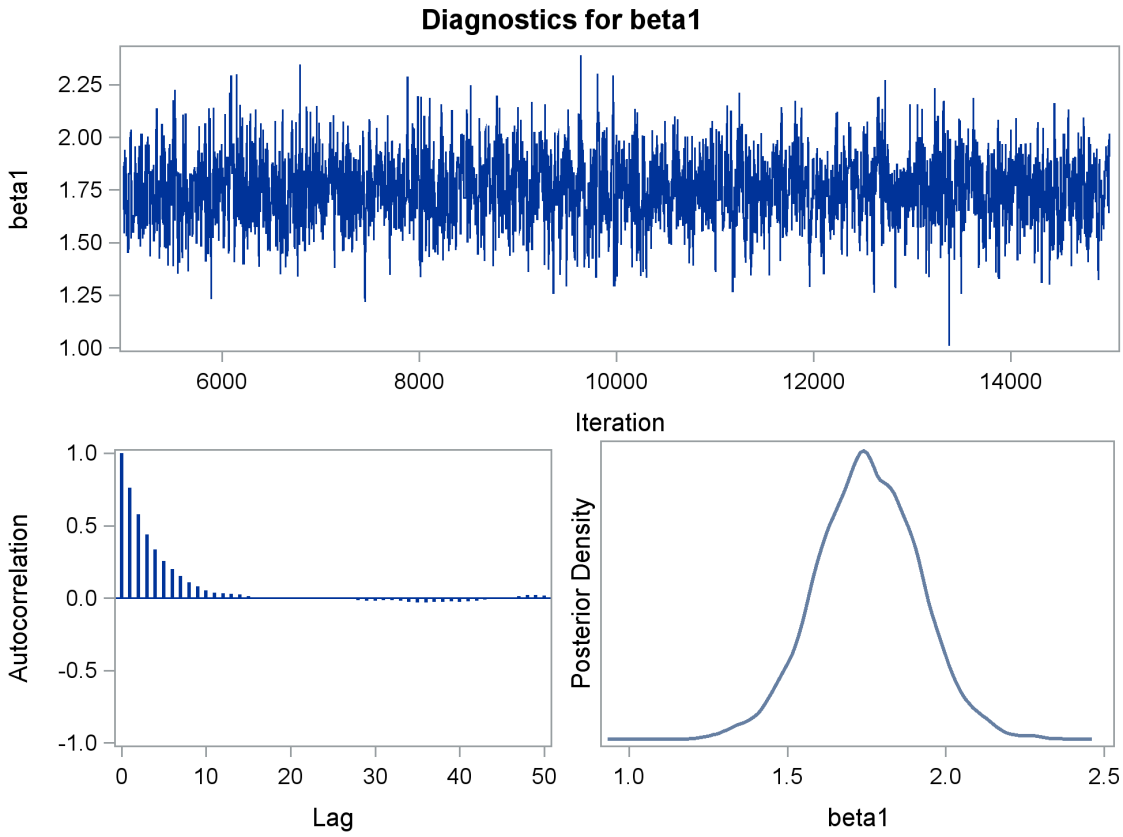
Figure 2 displays the information about the model parameters, including their prior distributions, sampling blocks, the initial values for starting the Markov chain simulation, and the sampling algorithms that are used to draw samples from the related posterior distributions. The variance parameter is sampled according to conjugacy (direct sampling from the conditional posterior), and the regression parameters are sampled using the random walk Metropolis algorithm. PROC MCMC begins the simulation with initial values that are based on the prior distributions, but you can also specify your own initial values in the PARMS statement.

Figure 2 Parameter Information

Parameters				
Block	**Parameter**	**Sampling Method**	**Initial Value**	**Prior Distribution**
1	**sigma2**	Conjugate	0.6667	igamma(shape=2, scale=2)
2	**beta0**	N-Metropolis	0	normal(0, var=100)
	beta1		0	normal(0, var=100)

For each of the model parameters, PROC MCMC generates a convergence diagnostic panel, which includes a trace plot, an autocorrelation plot, and a kernel density plot. The diagnostics panel in Figure 3 indicate good mixing for the slope coefficient β_1.

Figure 3 Diagnostic Plots for the Slope



If no issues of convergence is indicated, you can examine the posterior summary statistics. Figure 4 shows the posterior summary statistics, which include the posterior mean, standard deviation, and highest posterior density (HPD) intervals.

Figure 4 Posterior Summary and Interval Statistics

Posterior Summaries and Intervals					
Parameter	**N**	**Mean**	**Standard Deviation**	**95% HPD Interval**	
beta0	10000	-8.4939	9.5277	-28.1827	9.1692
beta1	10000	1.7539	0.1607	1.4414	2.0734
sigma2	10000	217.5	73.8757	102.4	362.6

Figure 4 shows that, on average, a one-unit increase in height causes a 1.7539-unit increase in weight, and that the probability that the slope coefficient (β_1) lies between 1.4414 and 2.0704 is 95%.

PROC MCMC provides many other posterior summary statistics, autocorrelations and Markov chain convergence diagnostic tools that are not shown here such as the posterior covariance matrix, the posterior correlation matrix and deviance information criteria (DIC). PROC MCMC also provides the following convergence diagnostic tests and statistics:

- Geweke test
- Heidelberger-Welch stationarity and half-width tests
- Raftery-Lewis test
- effective sample size (ESS)

- Monte Carlo standard error (MCSE)

Example 2: Generalized Linear Regression Models with Various Link Functions

This example illustrates how you can use SAS DATA step programming within PROC MCMC to fit various generalized linear models with different link functions.

The following statements create the Beetles data set. The response variable, **y**, is the number of deaths among **n** beetles that have been exposed to an environmental contaminant, **x**.

```
data Beetles;
   input n y x @@;
   datalines;
6  0  25.7   8  2  35.9   5  2  32.9   7  7  50.4   6  0  28.3
7  2  32.3   5  1  33.2   8  3  40.9   6  0  36.5   6  1  36.5
6  6  49.6   6  3  39.8   6  4  43.6   6  1  34.1   7  1  37.4
8  2  35.2   6  6  51.3   5  3  42.5   7  0  31.3   3  2  40.6
;
```

Suppose the likelihood has a binomial distribution,

$$y_i \mid p_i \sim \text{binomial}(n_i,\ p_i)$$

where the success probability, p_i, is the logistic transformation of the regression mean, $\alpha + \beta x_i$:

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta x_i$$

Suppose the regression parameters α and β have diffuse normal priors:

$$\alpha,\ \beta \sim \text{normal}(0,\ \text{var=}10000)$$

The following statements fit the described logistic regression model:

```
proc mcmc data=Beetles seed=2 nmc=20000 nthin=2 outpost=Beetlesout;
   parms alpha beta;
   prior alpha beta ~ normal(0, var=10000);
   p=logistic(alpha+beta*x);
   model y ~ binomial(n, p);
run;
```

The NTHIN=2 option in the PROC MCMC statement keeps every other Markov chain sample and discards the rest to reduce the autocorrelation. The PARMS statement specifies the model parameters, and the PRIOR statement assigns their prior distributions. The programming statement **`p=logistic(alpha+beta*x)`** corresponds to

$$p = \frac{1}{1 + \exp -(\alpha + \beta x)}$$

where the LOGISTIC function is a SAS DATA step function that is supported in PROC MCMC. The MODEL statement specifies the binomial likelihood for the response variable **y**.

Figure 5 shows the posterior summary statistics for this analysis.

Figure 5 Posterior Summary and Interval Statistics

Posterior Summaries and Intervals					
Parameter	**N**	**Mean**	**Standard Deviation**	**95% HPD Interval**	
alpha	10000	-12.5800	2.0454	-15.7692	-8.1235
beta	10000	0.3128	0.0529	0.2003	0.3981

The results in Figure 5 show that the contaminant **x** has a negative effect on survival time of beetles. Suppose you want to find the probability that increased levels of contaminant **x** have a negative effect on survival time of beetles. For this, you need to estimate the following probability

$$P(\beta > 0 \mid data)$$

One way to estimate this probability is to define a function of the model parameter β directly in PROC MCMC. For example, you can add **prob=(beta>0)** to your PROC MCMC code. Because β is a random variable that has a posterior distribution, **prob** is also a random variable that has a posterior distribution. To obtain summary statistics for the **prob** variable, you must use the MONITOR option in the PROC MCMC statement, as in the following statements.

```
proc mcmc data=Beetles seed=2 nmc=20000 nthin=2 outpost=Beetlesout monitor=(prob);
   parms alpha beta;
   prior alpha beta ~ normal(0, var=10000);
   p=logistic(alpha+beta*x);
   model y ~binomial(n, p);
   prob=(beta>0);
run;
```

Another way to obtain posterior information about the function of the parameters is to first run PROC MCMC only with the original model parameters, and then use other SAS procedures (such as FREQ, UNIVARIATE, CAPABILITY, or KDE) or use SAS autocall macros (such as %POSTSUM and %POSTINT) to further analyze the posterior output data set. The following statements show how PROC FREQ is used for to estimate $P(\beta > 0 \mid data)$ using the output posterior data set Beetlesout:

```
proc format;
   value betafmt low-0 = 'beta <= 0' 0<-high = 'beta > 0';
run;

proc freq data=Beetlesout;
   tables beta /nocum;
   format beta betafmt.;
run;
```

Figure 6 shows that all the simulated values of β are greater than 0, so the estimated posterior probability is 100%. The evidence overwhelmingly supports the hypothesis that increased levels of the environmental contaminant increase the probability of death.

Figure 6 PROC FREQ Output

beta	Frequency	Percent
beta > 0	10000	100.00

Suppose you also want to fit a probit model. All you need to do is change the logistic transformation to the following probit transformation:

$$p_i = \Phi(\alpha + \beta x_i)$$

where Φ stands for the cumulative density function (CDF) of the standard normal distribution. This yields the following PROC MCMC code:

```
proc mcmc data=Beetles seed=2 nmc=20000 nthin=2 outpost=Beetlesout;
   parms alpha beta;
   prior alpha beta ~ normal(0, var=10000);
   p=cdf("normal", alpha+beta*x, 0, 1);
   model y ~ binomial(n, p);
run;
```

Example 3: Nonlinear Regression Model

PROC MCMC is not limited to fitting linear or generalized linear models; you can also fit nonlinear regression models. This example illustrates how to fit a nonlinear Poisson regression model and how to improve the mixing of the Markov chain by sampling for the transformed scale of a model parameter.

The following data set contains the number of calls to a technical support help line after a product release and the week in which these calls are received:

```
data calls;
   input weeks calls @@;
   datalines;
1   0   1   2   2   2   2   1   3   1   3   3
4   5   4   8   5   5   5   9   6  17   6   9
7  24   7  16   8  23   8  27
;
```

The number of daily calls can be modeled as a Poisson random variable where the average number of calls is a nonlinear function of the number of weeks that have elapsed since the product's release.

$$\begin{aligned} y_i \mid \alpha + \beta x_i &\sim Poisson(\lambda_i) \\ \lambda_i &= \frac{\gamma}{1 + \exp[-(\alpha + \beta x_i)]} \end{aligned}$$

Suppose the prior distributions of the model parameters are as follows:

$$\begin{aligned} \gamma &\sim \text{gamma}(3.5,\ \text{scale}=12) \\ \alpha &\sim \text{normal}(-5,\ \text{sd}=0.25) \\ \beta &\sim \text{normal}(0.75,\ \text{sd}=0.5) \end{aligned}$$

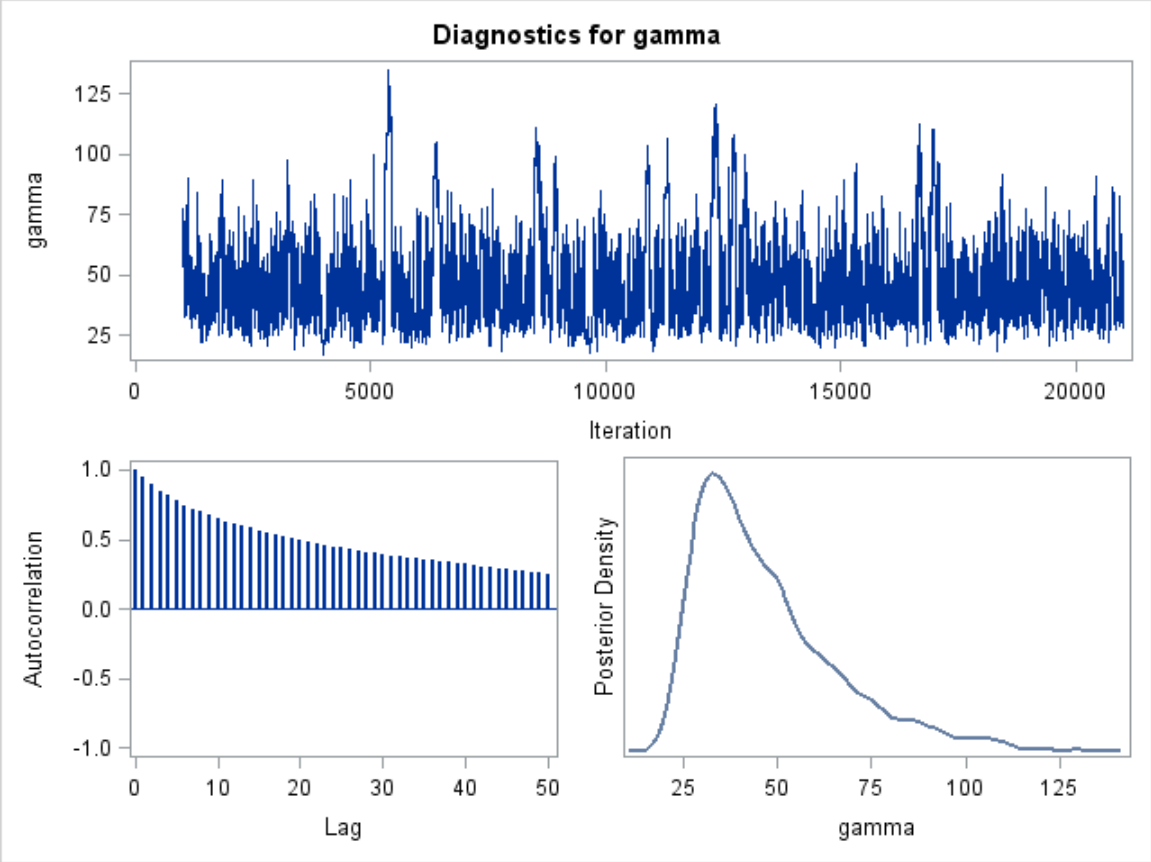
The following PROC MCMC statements fit the described Poisson regression model:

```
proc mcmc data=calls nmc=20000 seed=53197 outpost=callsout  propcov=quanew;
   parms alpha -4 beta 1 gamma 2;
   prior gamma ~ gamma(3.5, scale=12);
   prior alpha ~ normal(-5, sd=0.25);
   prior beta  ~ normal(0.75, sd=0.5);
   lambda=gamma*logistic(alpha+beta*weeks);
   model calls ~ poisson(lambda);
run;
```

The **propcov=quanew** option in the PROC MCMC statement specifies the method used in constructing the initial covariance matrix for the Metropolis-Hastings algorithm. The QUANEW method finds numerically approximated covariance matrices at the optimum of the posterior density function with respect to all continuous parameters, which can greatly increase convergence performance.

The trace plot in Figure 7 for the model parameter γ shows that the mixing is not moving efficiently. In addition, the autocorrelation plot shows that the autocorrelations do not drop off quickly, which indicates a less efficient mixing in the Markov chain.

Figure 7 Diagnostic Plots for the Slope



In Figure 8, the effective sample size (ESS) values of each parameter is relatively low compared to the Markov chain size (20,000). This further indicates the bad efficiency in the Markov chain.

Figure 8 Effective Sample Sizes

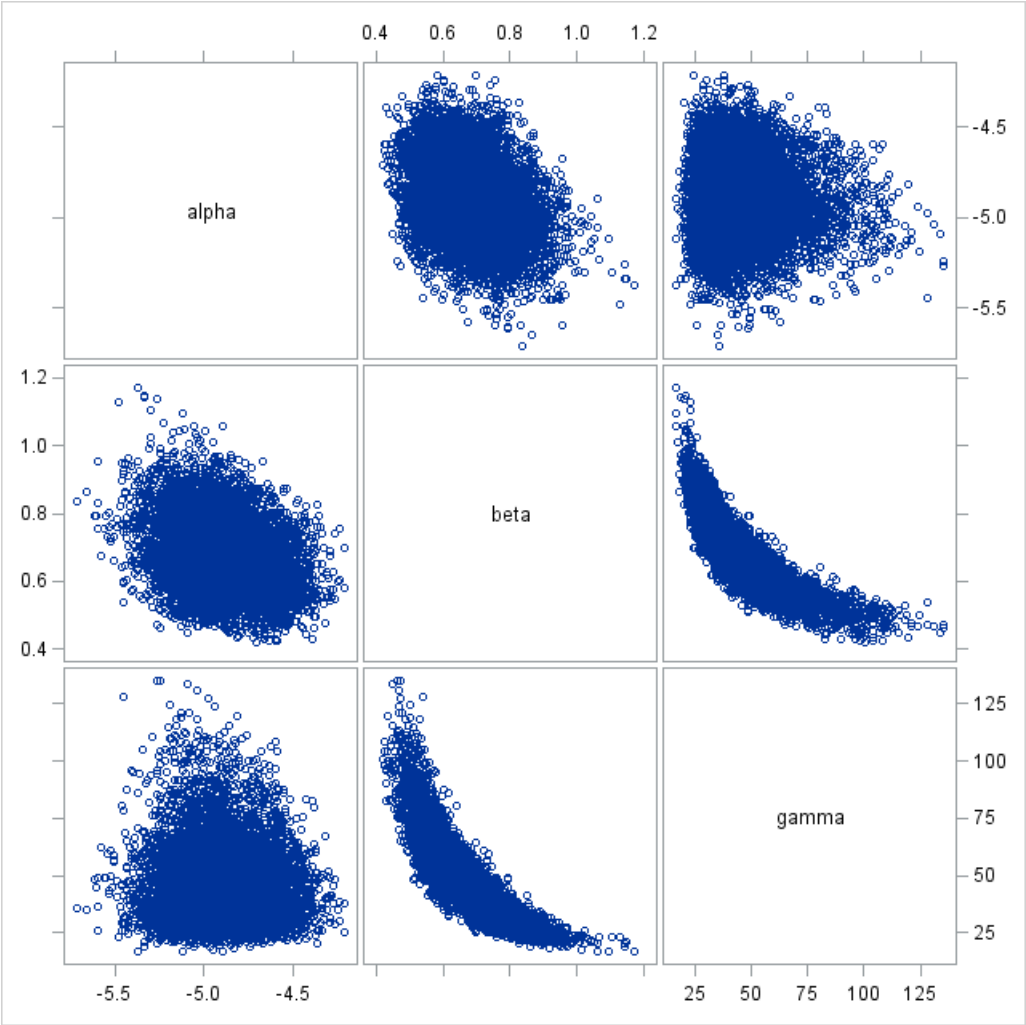
Effective Sample Sizes			
Parameter	**ESS**	**Autocorrelation Time**	**Efficiency**
alpha	841.4	23.7691	0.0421
beta	404.4	49.4592	0.0202
gamma	305.1	65.5494	0.0153

Simple scatter plots are often very useful in further investigating problems in mixing. The following SAS program requests pairwise scatter plots of the model parameters from their posterior samples:

```
proc sgscatter data=callsout;
   matrix alpha beta gamma;
run;
```

In Figure 9, the nonlinearity in the beta and gamma parameters stands out. This nonlinearity in the model parameters often causes the random walk Metropolis sampling algorithm to yield a poorly mixed Markov chain.

Figure 9 Pairwise Scatter Plots of the Parameters



Reparameterization can often improve the mixing. Notice that the gamma parameter in Figure 7 has positive support and that the posterior distribution is skewed to the right. This suggests that the chain might mix more rapidly if you sample from the logarithm of the gamma parameter.

Let $\delta = \log(\gamma)$ then the mean function becomes

$$\lambda_i = \frac{exp(\delta)}{1 + \exp[-(\alpha + \beta x_i)]}$$

This reparameterization yields the following exponential gamma distribution for δ:

$$\delta \sim \text{egamma}(3.5,\ \text{scale=}12)$$

Applying all these changes yields the following PROC MCMC code:

```
proc mcmc data=calls nmc=20000 seed=53197 outpost=callsout  propcov=quanew plots=trace;
   parms alpha -4 beta 1 delta 2;
   prior delta ~ egamma(3.5, scale=12);
   prior alpha ~ normal(-5, sd=0.25);
   prior beta  ~ normal(0.75, sd=0.5);
   lambda=exp(delta)*logistic(alpha+beta*weeks);
   model calls ~poisson(lambda);
run;
```

Figure 10 shows that the Markov chain is moving more efficiently for all model parameters, and Figure 11 shows improved mixing results in better ESSs.

Figure 10 Trace Plots of the Parameters

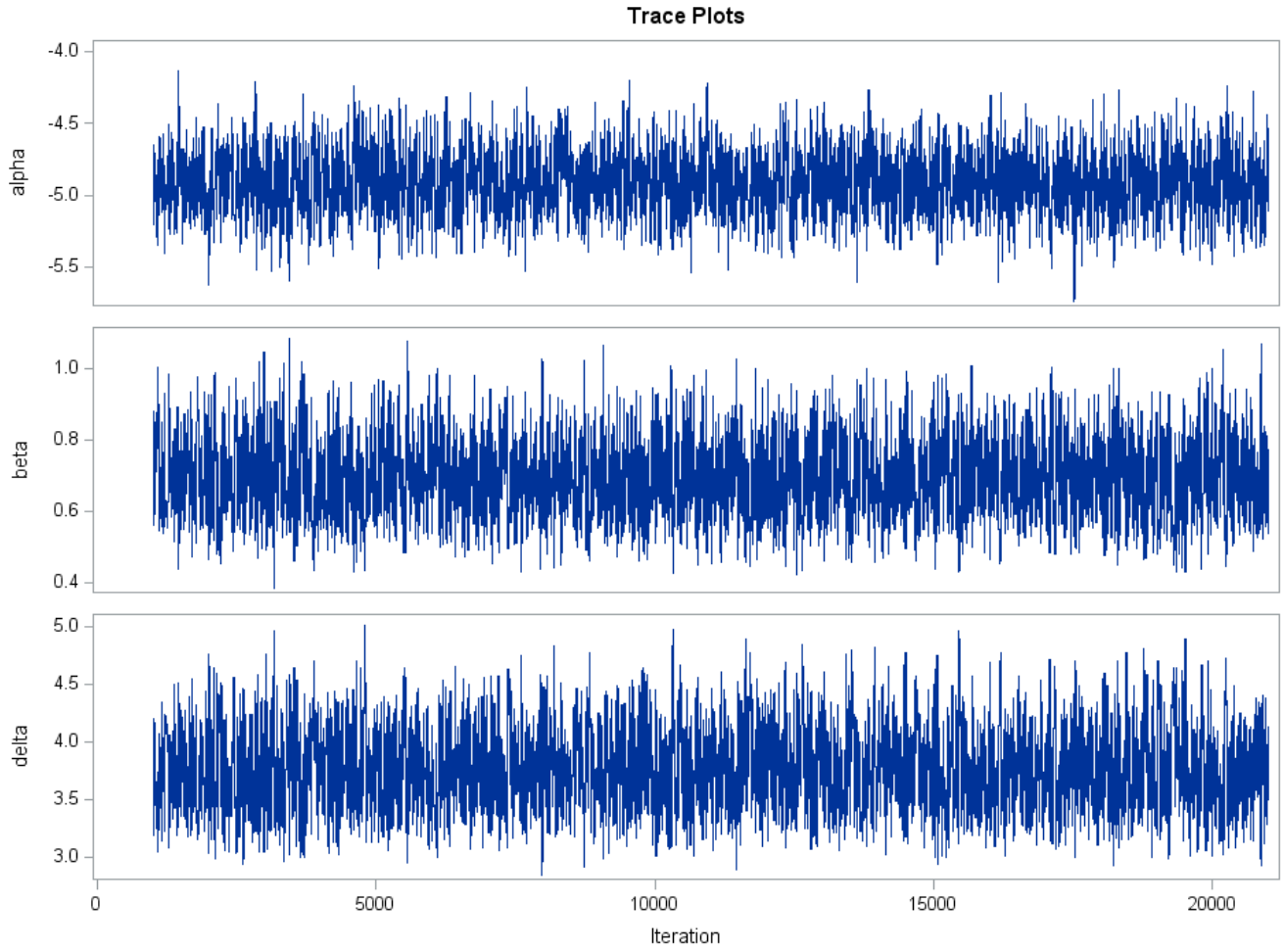


Figure 11 Effective Sample Sizes

Effective Sample Sizes			
Parameter	**ESS**	**Autocorrelation Time**	**Efficiency**
alpha	1690.3	11.8326	0.0845
beta	1506.5	13.2757	0.0753
delta	1396.8	14.3180	0.0698

Example 4: Hierachical Bayesian Modeling for Meta-Analysis

This example illustrates an application of hierarchical Bayesian modeling to a meta-analysis study in medicine and shows the use of the RANDOM statement to specify random-effects parameters. Meta-analysis is an important technique that combines information from different studies. If you have no prior information for believing that any particular study is different from another (exchangeability), you can treat Bayesian meta-analysis as a hierarchical model.

The data in this example, come from a meta-analysis study, which combines results from 22 health care

centers (Yusuf et al. 1985). At each center, patients are randomly assigned to receive beta blockers or a placebo after a myocardial infarction, and the mortality rates are recorded.

The following statements read the data into SAS and create the Multistudies data set, in which **center** shows the center ID; **ctrl** and **ctrlN** are the number of deaths and the total number of patients in the control group; and **trt** and **trtN** are the number of deaths and total number of patients, respectively, in the treatment group.

```
data multistudies;
   input center ctrl ctrlN trt trtN;
   datalines;
   1 3 39 3 38
   2 14 116 7 114

   ... more lines ...

   21 43 364 27 391
   22 39 674 22 680
   ;
```

Suppose the research is interested in estimating both the study-specific effect and the pooled effect of the treatment on the mortality rates.

For each center j $(j = 1, \ldots, 22)$, suppose the number of deaths in the control and treatment groups have the binomial distributions,

$$\begin{aligned} ctrl_j &\sim \text{binomial}(ctrlN_j,\ q_j) \\ trt_j &\sim \text{binomial}(trtN_j,\ p_j) \end{aligned}$$

where p_j and the q_j are the true death probabilities in the treatment and control groups respectively.

The log odds of the treatment group and control group are defined as follows, respectively:

$$\begin{aligned} \phi_j &= \log\left(\frac{q_j}{1-q_j}\right) \\ \phi_j + \theta_j &= \log\left(\frac{p_j}{1-p_j}\right) \end{aligned}$$

Then the treatment effect, θ_j, is the log odds ratio:

$$\theta_j = \log\left(\frac{p_j(1-q_j)}{(1-p_j)q_j}\right)$$

and ϕ_j can be explained as the baseline effect.

Under the assumption that the log odds ratios (θ_j) and the log odds of the control group (ϕ_j) are exchangeable from center to center, you can treat both θ_j and ϕ_j as random-effects parameters that are drawn from the following prior distributions:

$$\begin{aligned} \theta_j &\sim \text{normal}(\mu_\theta,\ \sigma^2_\theta) \\ \phi_j &\sim \text{normal}(\mu_\phi,\ \sigma^2_\phi) \end{aligned}$$

Suppose the following prior distributions are placed on the model parameters:

$$\begin{aligned} \mu_\theta,\ \mu_\phi &\sim \text{normal}(0, 3) \\ \sigma^2_\theta,\ \sigma^2_\phi &\sim \text{igamma}(\text{shape} = 0.01, \text{scale} = 0.01) \end{aligned}$$

where igamma stands for the inverse-gamma distribution.

The following PROC MCMC statements fit the described model:

```
proc mcmc data=multistudies outpost=PostOut nmc=50000 thin=5 monitor=(OR Pooled);

   parms  mu_theta mu_phi s_theta s_phi;

   prior mu: ~ normal(0,sd=3);
   prior s: ~ igamma(0.01,s=0.01);

   random theta ~n(mu_theta, var=s_theta) subject=center;
   random phi ~n(mu_phi, var=s_phi) subject=center;

   q = logistic(phi);
   model ctrl ~ binomial(ctrlN, q);

   p = logistic(theta + phi);
   model trt ~ binomial(trtN, p);

   array OR[22];
   OR[center]=exp(theta);
   Pooled=exp(mu_theta);
run;
```

The RANDOM statement offers a simple way to specify the random-effects parameters in a model. In the preceding program, the RANDOM statements declare **theta** and **phi** as random-effects parameters and specify a normal prior distribution for them. The SUBJECT= option in each RANDOM statement indicates that the grouping index is the data set variable **center**. PROC MCMC defines the number of random effects parameters according to the number of unique values in the **subject=** variable. The RANDOM statements improve the computational and convergence performance by requiring only a single pass through the data set to update all random-effects parameters that are defined by the same RANDOM statement. The procedure steps through the input data set and resolves the **theta** and **phi** with the appropriate reparametrization for that observation.

The programming statement, **OR[center]=exp(theta)**, calculates the odds ratio for each center and stores the result in the OR array. In the next line, the symbol **Pooled** calculates the overall odds ratio for the treatment effect.

Figure 12 displays the posterior summary statistics for the center-specific and pooled odds ratios. The odds ratios provide the multiplicative change in the odds of mortality in the treatment group with respect to the control group. You see some variation among the treatment effects at different centers. The treatment is most effective at center 7 (estimated odds ratio 0.6829) and least effective at center 14 (estimated odds ratio 0.9573). The pooled posterior estimate of the odds ratio is 0.7849 and has a 95% HPD credible interval of (0.6783, 0.8848).

Figure 12 Posterior Summary and Interval Statistics

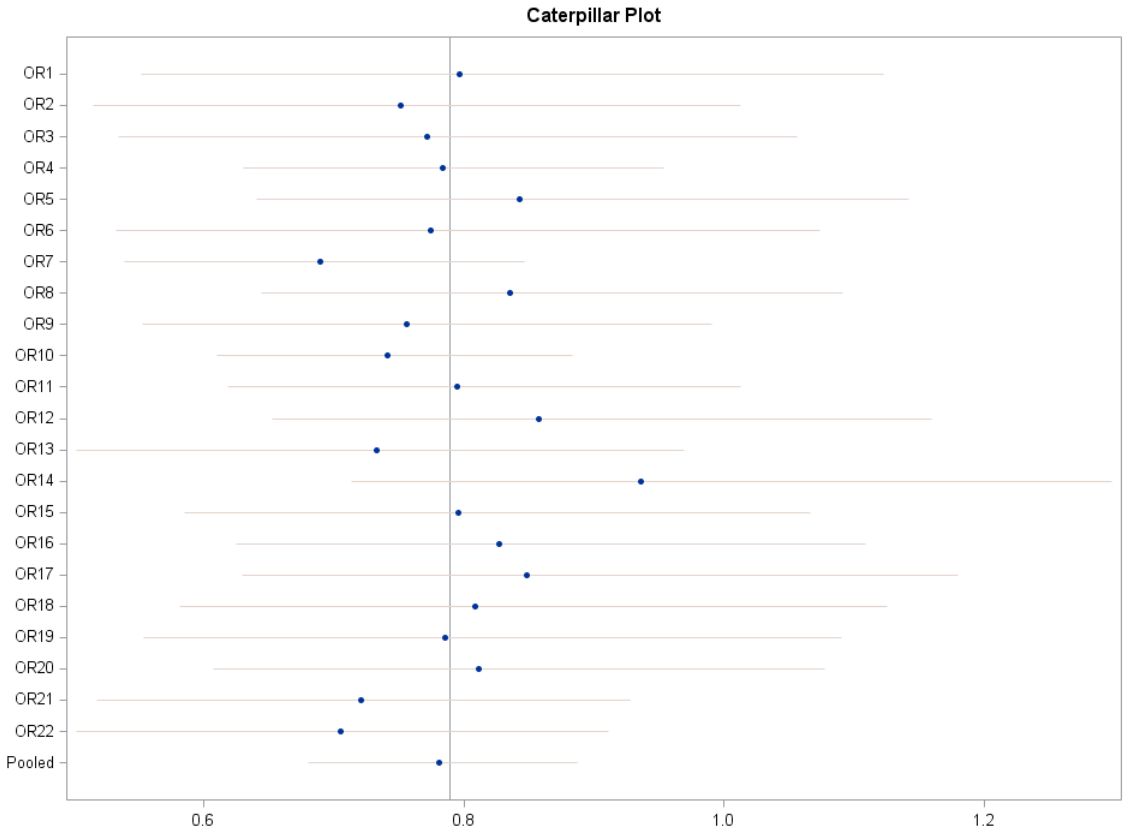
Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
OR1	10000	0.7963	0.1403	0.5296	1.0836
OR2	10000	0.7509	0.1231	0.5109	1.0044
OR3	10000	0.7720	0.1286	0.5148	1.0276
OR4	10000	0.7837	0.0820	0.6259	0.9468
OR5	10000	0.8430	0.1284	0.6089	1.0964
OR6	10000	0.7745	0.1333	0.5148	1.0512
OR7	10000	0.6897	0.0776	0.5373	0.8442
OR8	10000	0.8356	0.1122	0.6223	1.0580
OR9	10000	0.7561	0.1092	0.5412	0.9744
OR10	10000	0.7410	0.0698	0.6071	0.8797
OR11	10000	0.7945	0.0998	0.6093	0.9991
OR12	10000	0.8579	0.1276	0.6349	1.1272
OR13	10000	0.7327	0.1167	0.5046	0.9709
OR14	10000	0.9360	0.1481	0.6788	1.2355
OR15	10000	0.7957	0.1205	0.5551	1.0316
OR16	10000	0.8271	0.1232	0.6006	1.0652
OR17	10000	0.8481	0.1387	0.6028	1.1396
OR18	10000	0.8082	0.1362	0.5516	1.0889
OR19	10000	0.7854	0.1328	0.5226	1.0510
OR20	10000	0.8112	0.1174	0.5894	1.0493
OR21	10000	0.7205	0.1031	0.5133	0.9212
OR22	10000	0.7051	0.1036	0.4916	0.8980
Pooled	10000	0.7804	0.0523	0.6783	0.8848

You can use the %CATER autocall macro to create a caterpillar plot of the odds ratios. The following statement takes uses the PostOut data set and generates a caterpillar plot that includes center-specific and the pooled odds ratio estimates and their 95% HPD intervals.

```
%CATER(data=PostOut, var=OR: Pooled);
```

Figure 13 displays caterpillar plot of the odds ratio estimates.

Figure 13 Caterpillar Plot of the Odds Ratios



For More Information about PROC MCMC

For an overview of Bayesian statistics, see the chapter "Introduction to Bayesian Analysis Procedures" in the *SAS/STAT User's Guide*. You can find additional examples, such as Bayesian LASSO, Poisson regression for overdispersed data and Bayesian quantile regression, in a library of web examples: `http://support.sas.com/rnd/app/examples/STATwebexamples.html`.

Chen (2009) introduced the MCMC procedure when it was first released in SAS/STAT 9.2. This paper describes how to use PROC MCMC for estimation, inference, and prediction, and it includes examples about power priors, sensitivity analysis, and random-effects models. However, this paper was written before the RANDOM statement was introduced in PROC MCMC in SAS/STAT 9.3. The RANDOM statement offers a much simpler way to specify the random-effects parameters in a model. For more information about the RANDOM statement, see Chen (2011).

The Bayesian approach offers a natural model-based solution to the missing data problem by treating missing values as random variables and estimating their posterior distributions. Beginning with SAS/STAT® 12.1, PROC MCMC automatically samples all missing values and incorporates them in the Markov chain for the parameters. You can use the procedure to handle various types of missing data, including data that are missing at random (MAR) and missing not at random (MNAR). PROC MCMC can also perform joint modeling of missing responses and covariates. Chen (2013) reviews the Bayesian approach and describes how the MCMC procedure implements it.

Other important features of PROC MCMC include its support for a wide variety of standard distributions and the PREDDIST statement. PROC MCMC supports a number of standard univariate distributions that have upper or lower truncation options, and it supports multivariate distributions such as the multivariate normal and inverse-Wishart distributions. To simplify the coding of multivariate distributions, PROC MCMC uses intuitive array-based syntax (Chen 2011). In addition, the GENERAL and DGENERAL functions enable you to construct nonstandard prior or likelihood functions. The PREDDIST statement in PROC MCMC enables

you to draw random samples from the posterior predictive distribution of the response variable. This is especially useful for prediction purposes and for checking whether the fitted model is consistent with the data.

Beginning with SAS/STAT 13.1, PROC MCMC is multithreaded. This means that you can get a valuable performance gain by taking advantage of all the cores available in your machine.

ACKNOWLEDGMENTS

The authors are grateful to Anne Baxter, Ed Huddleston and Maura Stokes of SAS Institute Inc. for their valuable assistance in the preparation of this paper.

CONTACT INFORMATION

Funda Gunes
SAS Institute Inc.
SAS Campus Drive
Cary, NC, 27513
funda.gunes@sas.com

Fang Chen
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
fangk.chen@sas.com

REFERENCES

Chen, F. (2009), "Bayesian Modeling Using the MCMC Procedure," in *Proceedings of the SAS Global Forum 2009 Conference*, Cary, NC: SAS Institute Inc.
URL http://support.sas.com/resources/papers/proceedings09/257-2009.pdf

Chen, F. (2011), "The RANDOM Statement and More: Moving On with PROC MCMC," in *Proceedings of the SAS Global Forum 2011 Conference*, Cary, NC: SAS Institute Inc.
URL http://support.sas.com/resources/papers/proceedings11/334-2011.pdf

Chen, F. (2013), "Missing No More: Using the MCMC Procedure to Model Missing Data," in *Proceedings of the SAS Global Forum 2013 Conference*, Cary, NC: SAS Institute Inc.
URL https://support.sas.com/resources/papers/proceedings13/436-2013.pdf

Stokes, M. (2014), "An Introduction to Bayesian Analysis with SAS/STAT Software," in *Proceedings of the SAS Global Forum 2014 Conference*, Cary, NC: SAS Institute Inc.
URL https://support.sas.com/resources/papers/proceedings14/SAS400-2014.pdf

Yusuf, S., Petro, R., Lewis, J., Collins, R., and Sleight, P. (1985), "Beta Blockade During and After Myocardial Infarction: An Overview of Randomized Trials," *Progress in Cardiovascular Diseases*, 27, 335–371.