Paper 257-2009

Bayesian Modeling Using the MCMC Procedure

Fang Chen, SAS Institute Inc, Cary, NC

ABSTRACT

Bayesian methods have become increasingly popular in modern statistical analysis and are being applied to a broad spectrum of scientific fields and research areas. This paper introduces the new MCMC procedure in SAS/STAT 9.2, which is designed for general-purpose Bayesian computations. The MCMC procedure enables you to carry out analysis on a wide range of complex Bayesian statistical models. The procedure uses the Markov chain Monte Carlo (MCMC) algorithm to draw samples from an arbitrary posterior distribution, which is defined by the prior distributions for the parameters and the likelihood function for the data that you specify. This paper describes how to use the MCMC procedure for estimation, inference, and prediction.

INTRODUCTION

This paper introduces the new MCMC procedure in SAS/STAT® software. The MCMC procedure is currently available for SAS 9.2 as an experimental procedure and will become production during 2009.

The MCMC procedure is based on Markov chain Monte Carlo methods; it performs posterior sampling and statistical inference for Bayesian parametric models. The procedure fits single-level or multilevel models. These models can take various forms, from linear to nonlinear models, by using standard or nonstandard distributions. To use the procedure, you declare parameters in the model and specify prior distributions of the parameters and a conditional distribution for the response variable given the parameters. The MCMC procedure enables you to fit models by using either a keyword for a standard form (normal, binomial, gamma) or SAS programming statements to specify a general distribution.

The MCMC procedure uses a random walk Metropolis algorithm to simulate samples from the model you specify. You can also choose an optimization technique (such as the quasi-Newton algorithm) to estimate the posterior mode and approximate the covariance matrix around the mode. The procedure computes a number of posterior estimates, and it also outputs the posterior samples to a data set for further analysis. Successful Bayesian inference that uses this sampling-based approach depends on the convergence of the Markov chain. The MCMC procedure provides a number of convergence diagnostics so you can assess the convergence of the chains.

This paper first provides a brief overview of some relevant concepts in Bayesian methods and sampling-based inference, and then explains the mechanisms that drive the MCMC procedure. A number of examples, ordered from simple to more complex, demonstrate how to use the procedure. The examples show a single-parameter binomial model, inference on functions of parameters, power priors, nonstandard distributions, sensitivity analysis, a random-effects model, and meta-analysis. In addition to the models that are demonstrated in this paper, you can use PROC MCMC to fit zero-inflated Poisson regression, survival analysis, ordinal multinomial models, time series, missing data analysis, and so on. Detailed examples and discussions can be found in the PROC MCMC documentation in the *SAS/STAT User's Guide* and on SAS web site (see the section "CONTACT INFORMATION" on page 22 for more details).

OVERVIEW

Bayesian analysis is a statistical method that makes inference on unknown quantities of interest (which could be parameters in a model, missing data, or predictions) by combining prior beliefs about the quantities of interest and information (or evidence) contained in an observed set of data. In essence, a Bayesian model has two parts: a statistical model that describes the distribution of data ($\mathbf{y}$) given the unknown quantities, and a prior distribution that describes your beliefs about the unknown quantities independent of the data. Most commonly, the unknown quantities of interest are the parameters θ in a model, and the statistical model is the likelihood function $L(\theta;\mathbf{y})$. After collecting the data, you can update your beliefs about θ and calculate the posterior distribution $p(\theta|\mathbf{y})$. The updating from the prior distribution to the posterior distribution is carried out using Bayes' theorem

$$p(\theta|\mathbf{y}) \propto f(\mathbf{y}|\theta)\cdot\pi(\theta) = L(\theta;\mathbf{y})\cdot\pi(\theta)$$

where $f(\mathbf{y}|\theta)$ is the sampling distribution of the response variable and $\pi(\theta)$ is the prior distribution on θ.

Bayesian methods explicitly use probability distributions as a way to quantify uncertainties about the unknown quantities. Probability describes degree of belief rather than long-run frequency. This is a considerable deviation from the classical statistics paradigm. As a result, Bayesian inference is carried out conditional on the observed data and

does not rely on the assumption that a hypothetical infinite population of data exists. These differences give certain advantages to Bayesian methods, such as that all inferences are exact and not approximated and that the results are interpretable so that you can easily answer any scientific questions directly. On the other hand, no readily available asymptotic results (such as the central limit theorem based on an asymptotic increment of data) can be used to facilitate estimations, and Bayesian analysis usually requires a much greater amount of computational resources.

In theory, Bayesian methods are straightforward—the posterior distribution contains everything you need to carry out inference. In practice, the posterior distribution can be difficult to estimate precisely. Without the advantage of asymptotic results, a Bayesian statistician is required to explore the entire parameter space of the posterior distribution, not only where the density is mostly concentrated (area around the mode), but also the tail regions. Most Bayesian analyses resort to sophisticated computations, including the use of simulation methods, in order to provide accurate estimates of the posterior distribution.

Approximating an unknown but potentially complex distribution by drawing samples from it is an effective way to estimate the distribution, although the simulation can take a long time to complete. One popular and very general simulation method is Markov chain Monte Carlo (MCMC). MCMC methods sample successively from a target distribution, with each sample depending on the previous one, hence the notion of the Markov chain. Monte Carlo integration computes an expectation by averaging the Markov chain samples

$$\int_S g(\theta)p(\theta)d\theta \cong \frac{1}{n}\sum_{t=1}^{n} g(\theta^t)$$

where $g(\cdot)$ is a function of interest and θ^t are samples from $p(\theta)$ on its support S.

The MCMC procedure uses a special case of the MCMC method, the random walk Metropolis algorithm (Metropolis and Ulam 1949; Metropolis et al. 1953; Hastings 1970) to generate a sequence of draws from the joint posterior distribution of parameters. The procedure is capable of constructing an optimal proposal distribution in the random walk Metropolis algorithm, and the procedure can be used to generate samples from an arbitrary density. Once you obtain the samples, you can carry out additional statistical inference as desired.

Gelman et al. (2004), a widely popular textbook, provides good overviews, theoretical developments, and data analysis examples in Bayesian statistics. For books with varying difficulty degrees, see DeGroot and Schervish (2002), Lee (2004), Box and Tiao (1992), Leonard and Hsu (1999), Carlin and Louis (2008), Robert (2001), Berger (1985), and Bernardo and Smith (1994). For books that emphasize Monte Carlo methods and different sampling algorithms, see Robert and Casella (2004); Chen, Shao, and Ibrahim (2000); and Liu (2001).

HOW PROC MCMC WORKS

To use PROC MCMC, you specify the model parameters (using the PARMS statements), the prior distributions (using the PRIOR statements), and the conditional distribution of the response variable given the parameters and covariates in the data set (using the MODEL statements). The prior distributions and the likelihood function jointly define a posterior distribution, which becomes the objective function that PROC MCMC uses in the Metropolis algorithm.

By default, PROC MCMC assumes that all observations in the input data set are independent, and the logarithm of the posterior density is calculated as

$$\log(p(\theta|\mathbf{y})) = \log(\pi(\theta)) + \sum_{i=1}^{n} \log(f(y_i|\theta)) + C \tag{1}$$

where θ can be a multidimensional parameter vector, $f(y_i|\theta)$ is the likelihood function for a *single* observation in the data set, n is the sample size, and C is a constant that can be ignored. At each simulation iteration, PROC MCMC steps through the input data set and accumulates the log likelihood from each observation to obtain the joint log likelihood for the data set. Then the procedure evaluates and sums the logarithm of the prior density. The outcome is the value of the objective function, conditional on the current Markov chain sample.

The MCMC procedure is a versatile simulation procedure that enables you to analyze a range of models. You are not restricted to work with models that have the same likelihood function for each observation, as described in Equation(1). You can analyze data sets that have different likelihood functions for subsets of the data, such as

$$\log(f(\mathbf{y}|\theta)) = \sum_{i=1}^{n_1} \log(f_1(y_i|\theta)) + \sum_{i=n_1+1}^{n} \log(f_2(y_i|\theta))$$

where f_1 and f_2 are two different likelihood functions that apply to subsets of the data set. In addition, you can also analyze data that have interobservation dependence, using the log-likelihood function

$$\log(f(\mathbf{y}|\theta)) = \sum_{i=1}^{n} \log(f(y_i|y_{j\neq i}, \theta))$$

where the ith observation could depend on other observations in the data set.

Once the posterior simulation is complete, PROC MCMC calculates the following posterior statistics:

- point estimates: mean, standard deviation, percentiles
- interval estimates: equal-tail and highest posterior density (HPD) intervals
- posterior covariance matrix
- posterior correlation matrix
- deviance information criterion (DIC)

The following Markov chain convergence diagnostic tools are available:

- Geweke test
- Heidelberger-Welch stationarity and half-width tests
- Raftery-Lewis test
- posterior sample autocorrelations
- effective sample size (ESS)
- Monte Carlo standard error (MCSE)

The MCMC procedure provides the following graphical display of the posterior samples:

- trace plot (with optional smoothed mean curve)
- autocorrelation plot
- kernel density plot (with optional fringe plot)

SIMPLE BINOMIAL MODEL

This section provides a simple binomial example in which researchers are interested in evaluating the performance of a medical procedure in a multicenter study. They randomly select five centers for inclusion. One of the study goals is to compare the survival benefit of the medical procedure. In each center, trtN patients are randomly selected and assigned the treatment procedure, and trt deaths are recorded. Similarly, in each center, ctrlN patients are randomly assigned to the control procedure, and ctrl deaths are recorded. In this example, only the treatment arm of the data is used. The following statements create a SAS data set for that portion of the data:

```
data trt;
   input trt trtN;
   datalines;
2  86
2  69
1  71
1 113
1 103
;
```

The simplest model is to ignore any center difference and treat the data as the realization of a shared model, with the same probability applied to all centers. You can use the following binomial model where p is the shared survival probability:

$$\begin{aligned} \text{trt}_i &\sim \text{binomial}(\text{trtN}_i, p) \\ \pi(p) &= \text{uniform}(0,1) \end{aligned} \tag{2}$$

The uniform prior distribution is a noninformative prior distribution on the parameter p.

Setting a shared p in the model is equivalent to pooling all the data from different centers into a single trial, which can be easily done in a DATA step. However, the input data set is kept in the multicenter because it is used later to fit more complex models and to demonstrate the usage of PROC MCMC.

The PROC MCMC statements that fit this one-level binomial model are as follows:

```
ods graphics on;
proc mcmc data=trt seed=17 nmc=10000 outpost=out1;
   parm p 0.2;
   prior p ~ uniform(0,1);
   model trt ~ binomial(trtN,p);
run;
ods graphics off;
```

The ODS GRAPHICS ON statement requests Output Deliver System (ODS) Graphics in addition to the usual tabular output produced by the MCMC procedure. The ODS GRAPHICS OFF statement disables ODS Graphics. You have to specify ODS Graphics in order to produce any posterior diagnostics plots.

The PROC MCMC statement invokes the procedure, and the DATA= option inputs the trt data set. The SEED= option sets the random number seed to ensure Markov chain reproducibility. The NMC= option controls the number of Markov chain simulation. The OUTPOST= option saves the posterior samples to the data set out1.

The PARM statement identifies the model parameter and its starting value. The model contains only one parameter. If no initial values are provided for model parameters, PROC MCMC generates them. The PRIOR statement specifies a uniform prior distribution on p. The uniform distribution is one of the standard distributions that can be used in the procedure. The MODEL statement defines the dependent variable (trt) and its sampling distribution, a binomial distribution with the number of trials trtN and probability p.

The results of this analysis are shown in the following tables. Note that you can convert any PROC MCMC output table to a SAS data set for additional analysis, using the ODS statement (see the section "OUTPUT TABLE NAMES" on page 20 for more details).

The observation table (Figure 1) lists the number of observations in the input data set and the number of observations used from the analysis. By default, all observations that have missing values are excluded from the analysis. Since no missing data are presented, all observations are used.

Figure 1 Observation Information Table

The MCMC Procedure

Number of Observations Read	5
Number of Observations Used	5

The "Parameters" table (Figure 2) lists some basic information about the model parameters. Included are the parameters to be estimated, the sampling algorithm (normal-based random walk Metropolis), initial values, and corresponding prior distributions. These quantities are useful for verifying that you have correctly specified the prior distributions in the model.

Figure 2 Parameter Information Table

Parameters

Parameter	Sampling Method	Initial Value	Prior Distribution
p	N-Metropolis	0.5000	uniform(0,1)

The "Tuning History," "Burn-In History," and "Sampling History" tables (Figure 3) record the history of the Metropolis sampler. In the tuning history, PROC MCMC generates trial samples in consecutive phases and uses them to modify the proposal distribution. The goal is to build a proposal distribution that would give an optimal acceptance probability in the Markov chain, which in turn would produce efficient sampling. The tuned proposal distribution at the end of the tuning phase is used in both the burn-in and sampling stages of the simulation. By default, the burn-in sample size is 1000.

Figure 3 Tuning, Burn-In, and Sampling History Table

Tuning History

Phase	Block	Scale	Acceptance Rate
1	1	2.3800	0.0100
2	1	0.6980	0.0300
3	1	0.2430	0.1240
4	1	0.1193	0.4160

Burn-In History

Block	Scale	Acceptance Rate
1	0.1193	0.4500

Sampling History

Block	Scale	Acceptance Rate
1	0.1193	0.4290

The "Posterior Summaries" table (Figure 4) reports the posterior mean, standard deviation, and quantiles estimates of p. By default, PROC MCMC produces only the quantile estimates, but you can request any percentile estimates. The mean estimate of 0.0181, with a standard deviation of 0.00628, indicates a rather low probability of death for the treatment in all centers. The "Posterior Intervals" table lists the 95% equal-tail and HPD credible interval estimates for the model parameter. The MCMC procedure can compute any requested α-level intervals. Note that the HPD interval should always be smaller than, or at most equal to, the equal-tail interval.

Figure 4 Posterior Summary and Interval Statistics Tables

The MCMC Procedure

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
p	10000	0.0181	0.00628	0.0135	0.0174	0.0218

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
p	0.050	0.00795	0.0323	0.00677	0.0305

Figure 5 lists convergence diagnostics tests. The "Monte Carlo Standard Errors" table displays the Monte Carlo standard errors (MCSE) of the mean estimates. The MCSE-to-posterior standard deviation ratio of 0.022 indicates that only a tiny fraction of the posterior variability is due to the simulation. The "Posterior Autocorrelations" table reports the autocorrelations at selected lags (1, 5, 10, and 50, by default), and they drop off rather quickly, indicating reasonable mixing of the Markov chain. The large p-value of 0.8043 in the "Geweke Diagnostics" table shows that the mean estimate of the Markov chain is stable over time. The "Effective Sample Sizes" table reports the number of effective sample sizes of the parameter. These diagnostics tests do not indicate any nonconvergence of the Markov chain.

Figure 5 Convergence Diagnostics Tables

The MCMC Procedure

Monte Carlo Standard Errors

Parameter	MCSE	Standard Deviation	MCSE/SD
p	0.000138	0.00628	0.0220

Posterior Autocorrelations

Parameter	Lag 1	Lag 5	Lag 10	Lag 50
p	0.6541	0.1399	-0.0017	0.0006

Geweke Diagnostics

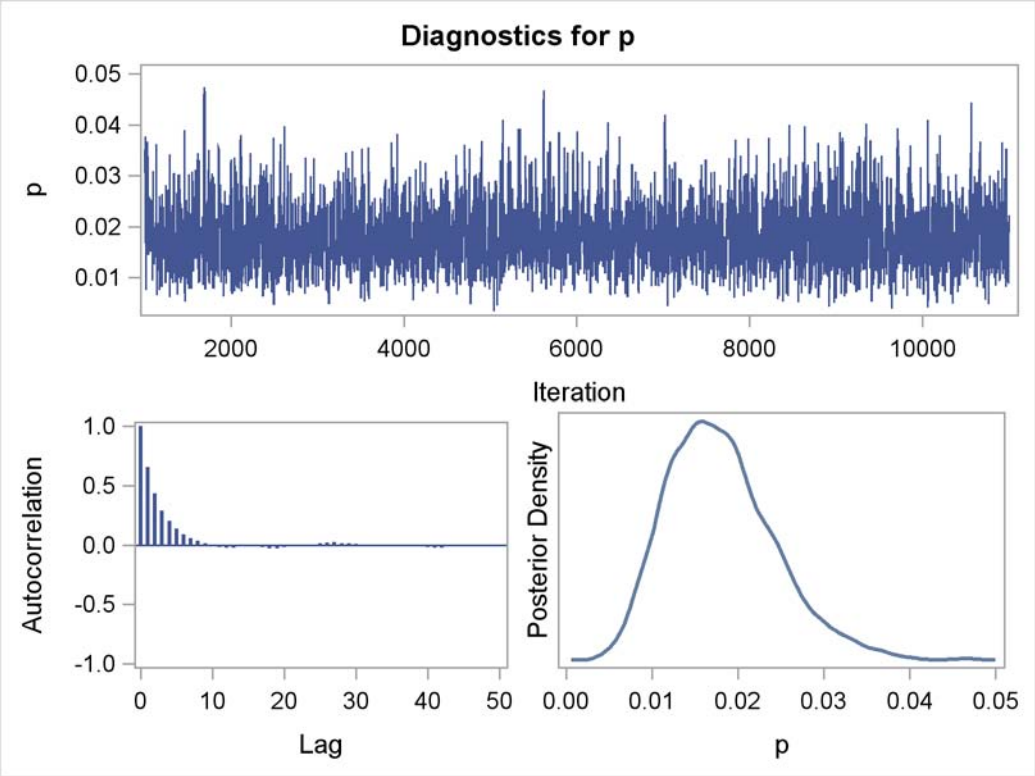
Parameter	z	Pr > |z|
p	0.2478	0.8043

Effective Sample Sizes

Parameter	ESS	Correlation Time	Efficiency
p	2061.5	4.8509	0.2061

The MCMC procedure produces three types of plots (Figure 6) that can aid further convergence checks. The trace plot indicates that the Markov chain has stabilized and appears constant over the graph. In addition, the chain has good mixing and is "dense" in the sense that it quickly traverses the support of the distribution and is able to explore both the tails and the mode areas efficiently. The autocorrelation plot shows a small degree of autocorrelation among the posterior samples, and the kernel density plot estimates the posterior marginal distribution for the parameter.

Figure 6 Diagnostic Plots for p



FUNCTIONS OF PARAMETERS AND PREDICTIONS

Although it is important to obtain parameter estimates, it is often more useful to calculate functions of parameters because they can be used to answer scientific questions directly. In Bayesian analysis, parameters are random variables that have posterior distributions; as a result, functions of parameters are also random variables that have posterior distributions. Consider the following quantities in the binomial example:

- log of the odds: $\log\left(p/(1-p)\right)$
- number of deaths in 242 future trials:

$$x_{\text{pred}} \sim \text{binomial}(242, p)$$

- the probability that x_{pred} exceeds a critical threshold of 4:

$$\Pr\left(x_{\text{pred}} \geq 4 \,|\, \text{Data}\right)$$

Since you already have a data set that contains the posterior samples, you can use the following programming statements to calculate these functions of parameters:

```
data tran (drop=iteration logprior loglike logpost);
   set out1;
   call streaminit(2);
   logodds = log(p/(1-p));
   xpred = rand("binomial", p, 242);
   pcrit = (xpred ge 4);
   output;
   run;
```

The data set tran now contains variables logodds (log of odds), xpred (x_{pred}), and pcrit (p_{critical}). Subsequently, you can use the UNIVARIATE, CAPABILITY, or KDE procedures to analyze the posterior sample. The CALL STREAMINIT statement sets the random number seed used in the RAND function to generate the predictive values of xpred.

Instead of calculating functions of parameters after you obtain the posterior samples, you can calculate them within the same call to PROC MCMC. This would make the program more efficient and would use some of the computing functionality of the MCMC procedure to calculate posterior estimates. To calculate any functions of parameters, you use the following steps:

1. programming statements, similar to those shown previously
2. the MONITOR= option to select the list of symbols (functions of parameters) of interest

The MCMC procedure then proceeds to report posterior summary statistics of these monitored quantities and record the samples in the output data set. The following statements illustrate how to compute the functions of parameters for this example:

```
proc mcmc data=trt seed=17 nmc=10000 outpost=out2
   monitor=(logodds xpred pcrit) diagnostics=none;
   parm p 0.2;

   begincnst;
     call streaminit(2);
   endcnst;

   logodds = log(p/(1-p));
   xpred = rand("binomial", p, 242);
   pcrit = (xpred ge 4);

   prior p ~ uniform(0,1);
   model trt ~ binomial(trtN,p);
run;
```

The MONITOR= option outputs analysis for selected symbols of interest in the program. The output data set contains the posterior samples of these monitored symbols in addition to the parameters declared in the PARM statements. If you want to monitor the model parameters, add the symbol _parms_ (which is shorthand for all model parameters) to the MONITOR= list. Given the reasonable convergence that you saw in the last example, you do not need to reproduce convergence diagnostics. The DIAGNOSTICS=NONE option turns off all diagnostic tests.

Within the BEGINCNST/ENDCNST statements, a single STREAMINIT function call is used to control the random seed in the RAND function call. Any statement within the BEGINCNST/ENDCNST block is executed before the Markov chain sampling. These statements are best used for setting up constants in the model. This STREAMINIT random number seed should not be confused with the seed specified in SEED= option in the PROC MCMC statement. The former controls the reproducibility of the RAND function that appears in the program; the latter controls the reproducibility of posterior sample draws.

The next three programming statements are identical to those seen in the DATA step calls. Because these programming statements do not involve any data set variables, you can enclose them within BEGINPRIOR/ENDPRIOR statements to avoid redundant calculations.

```
beginprior;
   logodds = log(p/(1-p));
   xpred = rand("binomial", p, 242);
   pcrit = (xpred ge 4);
endprior;
```

Programming statements enclosed within this block are not executed for every observation call in the simulation, as is the case in the likelihood calculation. Enclosing programming statements within this block reduces computational redundancy and increase efficiency.

Posterior estimates of the functions of parameters are shown in Figure 7.

Figure 7 Summary and Interval Statistics of Functions Of Parameters in the Binomial Model

The MCMC Procedure

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
logodds	10000	-4.0544	0.3686	-4.2961	-4.0366	-3.7991
xpred	10000	4.3580	2.5956	2.0000	4.0000	6.0000
pcrit	10000	0.5751	0.4944	0	1.0000	1.0000

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
logodds	0.050	-4.8271	-3.3996	-4.7993	-3.3770
xpred	0.050	0	10.0000	0	9.0000
pcrit	0.050	0	1.0000	0	1.0000

POWER PRIORS

This example illustrates an important aspect of Bayesian analysis: incorporation of prior information from historical data. The prior distribution used here is the power prior (Ibrahim and Chen 2000). This example also demonstrates two useful features of PROC MCMC: using nonstandard distributions and assigning different likelihood function to different subsets of the data.

Suppose that you have a historical data set that matches the current data set to some extent. In other words, these two data sets use a similar experiment and have a comparable likelihood function. You can construct a power prior that retains information from the historical data set and uses it as the prior distribution in the current analysis. This type of Bayesian analysis provides a principled and convenient way of updating knowledge from past.

To construct a power prior distribution, you can use the formula

$$p(\theta|D_0, a_0) \propto L(\theta; D_0)^{a_0} \cdot \pi_0(\theta)$$

where

- $D_0 = (n_0, \mathbf{y}_0, \mathbf{X}_0)$ is the historical (or pilot) data
- $L(\theta; D_0)$ is the likelihood of θ based on the historical data
- $\pi_0(\theta)$ is the initial prior for θ, the prior for θ before the historical data D_0 is observed
- a_0 is a discounting parameter, or a scale precision parameter, constrained to $0 \leq a_0 \leq 1$. This parameter a_0 controls the amount of weight you want to put on the historical data: $a_0 = 0$ corresponds to no incorporation of the historical data; $a_0 = 1$ corresponds to the Bayesian update of $\pi(a_0)$.

The posterior distribution of θ is

$$\begin{aligned} p(\theta|D, D_0, a_0) &\propto L(\theta; D) \cdot L(\theta; D_0)^{a_0} \cdot \pi_0(\theta) \\ &\propto \prod_{i=1}^{n} f(y_i|\theta, x_i) \cdot \prod_{j=1}^{n_0} f(y_{0,j}|\theta, x_{0,j})^{a_0} \cdot \pi_0(\theta) \end{aligned} \tag{3}$$

where

- $D = (n, \mathbf{y}, \mathbf{X})$ is the data from the current study
- $\mathbf{y} = \{y_i\}, \mathbf{x} = \{x_i\}$ for $i = 1 \cdots n$
- $\mathbf{y}_0 = \{y_{0,j}\}, \mathbf{x}_0 = \{x_{0,j}\}$ for $j = 1 \cdots n_0$ is the historical data
- $f(\cdot)$ is the likelihood function for a single observation in either the historical or the current data

Combining the two data sets, you can form a new data set D^* and rewrite the posterior distribution in Equation(3) as the following:

$$p(\theta|D^*, a_0) \propto \prod_{i=1}^{n+n_0} f_i(y_i|\theta, x_i) \cdot \pi_0(\theta) \tag{4}$$

$$\text{where } f_i = \begin{cases} f(y_i|\theta, x_i) & \text{for each } i \text{ in the current data set} \\ f(y_{0,i}|\theta, x_{0,i})^{a_0} & \text{for each } i \text{ in the historical data set} \end{cases}$$

The posterior distribution in Equation(4) is easy to specify in PROC MCMC; you can use the IF statement to assign the appropriate likelihood function to different observations in the data set.

Suppose that a pilot study was conducted prior to collecting the data set trt. The following DATA step creates a data set for the trial data from the pilot study, in which 200 patients received the medical procedure and 1 death was observed:

```
data pilot;
   input trt trtN;
   datalines;
1  200
;
```

The following SAS statements create a new combined data set comb. The group variable takes on the values 'current' and 'pilot' for the current data set and the pilot data set, respectively.

```
data comb;
   set trt(in=i) pilot;
   if i then group="current";
   else group="pilot";
   run;
```

Suppose that you do not want to put too much weight on the pilot study, and so you choose $a_0 = 0.2$ in the analysis. The PROC MCMC statements fit a power prior to the binomial model as follows:

```
proc mcmc data=comb seed=17 nmc=10000 outpost=pout;
   parm p 0.2;

   begincnst;
      a0 = 0.2;
   endcnst;

   prior p ~ uniform(0, 1);
   llike = logpdf("binomial", trt, p, trtn);
   if (group = 'pilot') then
      llike = a0 * llike;
   model general(llike);
run;
```

The weight constant a0 is defined within the BEGINCNST/ENDCNST block of statements. The PRIOR statement specifies a uniform prior on p, which is the prior used in the pilot study.

The next programming statement evaluates the logarithm of a binomial density for each observation and assigns it to the symbol llike. LOGPDF is a function that can be used in the procedure—virtually all DATA step functions are supported in PROC MCMC. Next, the IF statement selects observations from the pilot data set (when the condition group = 'pilot' is satisfied) and weights each by a_0. For observations in the current data set, the value of llike is unchanged. This model formulation is exactly what was stated in Equation(4). Using logical control statements, you can easily partition a data set into subsets and assign different log likelihood functions to each observation in the subset.

The MODEL statement declares the log-likelihood function for each observation. The statement uses the GENERAL function which indicates that you are using a nonstandard density or distribution. The argument to the GENERAL function must take the value of the logarithm of the prior or likelihood function.

Note that the response variable, trt, does not appear in the MODEL statement because the calculation of llike already included trt. Thus, you do not need to repeat the response variable in the MODEL statement. However, the following model specification is still permissible:

```
model trt ~ general(llike);
```

Figure 8 lists the posterior summary and intervals of the estimated p using the power prior with weight $a_0 = 0.2$.

Figure 8 Posterior Summary and Interval Statistics of p Using Power Prior

The MCMC Procedure

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
p	10000	0.0172	0.00617	0.0128	0.0164	0.0206

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
p	0.050	0.00748	0.0307	0.00637	0.0294

Posterior summary statistics reported in Figure 4 (analysis of the current data set without any prior information from the pilot study) and Figure 8 are quite similar. You might be interested in examining whether alternative models with different prior specifications would lead to vastly different results. The sensitivity analysis is covered in the next example.

SENSITIVITY ANALYSIS

Sensitivity analysis is the practice of understanding the variation and uncertainty of the posterior inferences as a result of a different prior or model (likelihood function) used in the analysis. For example, you might want to compare power prior analyses of the binomial model with different weight values of a_0. Sensitivity analysis involves running PROC MCMC several times, each time with a different value of a_0, and comparing the results. The BY statement makes it easy to carry out the analysis in one simple procedure call.

Suppose that you want to compare three power prior models with values of a_0 equal to $0, 0.5$, and 1. When $a_0 = 0$, the model corresponds to an analysis that completely discards the pilot study; when $a_0 = 1$, the analysis combines the pilot with the current study and assumes that they both come from the same population. The following statements generate a new data set with a BY-group indicator a0:

```
data sendata;
   set comb;
   do a0 = 0, 0.5, 1;
      output;
   end;

proc sort;
   by a0;
run;
```

The BY statement in PROC MCMC enables you to obtain separate analyses on data in groups defined by the BY variable. The following SAS statements runs three analyses, each with a different weight a_0:

```
ods output postintervals=pi postsummaries=ps;
proc mcmc data=sendata seed=17 nmc=10000 outpost=bout diagnostics=none;
   by a0;
   parm p;
   prior p ~ uniform(0, 1);
   llike = logpdf("binomial", trt, p, trtn);
   if (group = 'pilot') then
      llike = a0 * llike;
   model general(llike);
run;
```

Note that the value of the BY-group variable a0 is used in the calculation of the log likelihood for each observation.

Figure 9 and Figure 10 list posterior summary statistics and interval estimates of p with different a_0 weights in the power prior model. The mean estimates decrease slightly, and the intervals shift towards zero while the length is also decreasing.

Figure 9 Posterior Summary Statistics of p Using Different a_0 Values

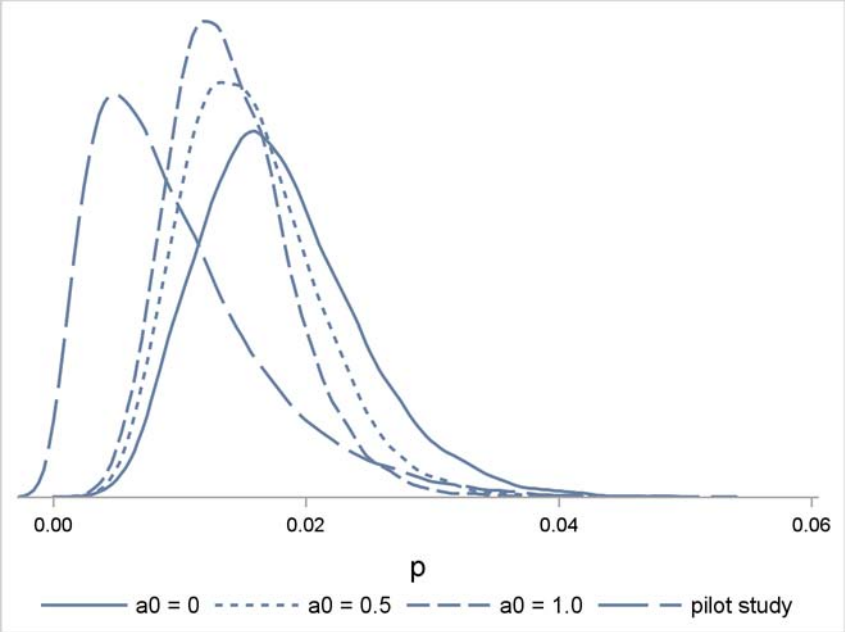
Obs	a0	Parameter	N	Mean	StdDev	P25	P50	P75
1	0.0	p	10000	0.0181	0.00628	0.0135	0.0174	0.0218
2	0.5	p	10000	0.0157	0.00520	0.0120	0.0151	0.0186
3	1.0	p	10000	0.0140	0.00466	0.0106	0.0134	0.0168

Figure 10 Posterior Interval Statistics of p Using Different a_0 Values

Obs	a0	Parameter	Alpha	Credible Lower	Credible Upper	HPDLower	HPDUpper
1	0.0	p	0.050	0.00795	0.0323	0.00677	0.0305
2	0.5	p	0.050	0.00697	0.0278	0.00627	0.0264
3	1.0	p	0.050	0.00638	0.0245	0.00541	0.0228

Figure 11 plots the estimated posterior distributions with different values of a_0. The solid line represents the fit for $a_0 = 0$. In this case, the posterior distribution is identical to the posterior in the first example (Figure 6). The mid-dashed line represents the fit for $a_0 = 1$, and the long-dashed line is an estimate of the posterior distribution that uses only data from the pilot study. The pilot study posterior is very different from the current data set, and the graph clearly shows that a_0 acts as a smoother that bridges the differences between the two data sets.

Figure 11 Posterior Distributions of p with Different Values of a_0



Sensitivity analysis can be more complex than comparing weights. You can use the IF logical control and the GENERAL function to alternate priors or likelihood functions for different BY groups. Suppose that you want to examine the selection of three different priors on θ in a hypothetical model. You can use the following skeleton code as a guide:

```
by a;
if a = 1 then lprior = logpdf(``normal'', theta, 0, 1);
 else if a = 2 then lprior = logpdf(``uniform'', theta, -5, 5);
 else if a = 3 then lprior = logpdf(``gamma'', theta, 1, 1);
prior theta ~ general(lprior);
```

With three levels in the BY-group variable a, PROC MCMC repeats the fit three times, each time applying a different prior (normal, t, or gamma) on theta.

RANDOM-EFFECTS MODEL

Suppose that you have evidence that not all trials share the same probability p, as indicated by Equation(2), and you decide to model separate probabilities p_i for different trials. You now have a random-effects model, where the likelihood formulation is

$$\text{trt}_i \sim \text{binomial}(\text{trtN}_i, p_i)$$

where $i = 1, \cdots, 6$ is the center indicator, including the pilot study. The prior distribution on p_i could be a beta distribution:

$$\pi(p_i) = \text{beta}(a, b)$$

If you know the amount of smoothing that you want to have on the p_i, you can select values for the hyperparameters a and b. For example, choosing $a = 1$ and $b = 1$ (a uniform prior) implies no shrinkage on the probabilities. The random-effects model becomes an independent model, which is equivalent to carrying out a separate analysis for each of the six trials. The empirical Bayes (EB) method (Carlin and Louis 2008) offers another way to choose these hyperparameters. EB optimizes the marginal posterior distribution of the hyperparameters given data (with all the p_i integrated out) and obtains estimates. With enough units/groups in the data, the EB method often provides a posterior that is similar to a fully Bayesian approach.

General speaking, fixing the values for hyperparameters amounts to ignoring information contained in the data on the amount of shrinkage that should be used in the analysis. To let the data decide what is the proper amount of strength to borrow from different centers to reduce the variance estimates, you can treat a and b as parameters and place hyperprior distributions on them. See Gelman et al. (2004) for discussions about the choice of informative and noninformative hyperprior distributions in hierarchical models.

Consider the following diffuse but proper priors on the hyperparameters:

$$\pi(a) = \text{exponential}(\text{scale} = 100)$$
$$\pi(b) = \text{exponential}(\text{scale} = 100)$$

The exponential distribution with scale of 100 has mean 100 and variance 10,000.

To fit a Bayesian random-effects model in PROC MCMC, first you need to create an index indicator in the input data set. The following statements create a new data set re, which contains a center indicator index:

```
data re (drop=group);
   set comb;
   index = _n_;
run;
```

The variable index takes on the value of 1 through 6 in the data set. The following statements fit a random-effects model to the binomial data:

```
proc mcmc data=re nmc=10000 outpost=outm seed=17;
   array p[6];
   parm p: 0.015;
   parm a b;
   prior p: ~ beta(a, b);
   prior a b ~ expon(scale=100);
   model trt ~ binomial(trtN,p[index]);
run;
```

The ARRAY statement allocates an array of size 6, for the six values of p_i, which are the random effects. The PARMS statements declare that p_i, a, and b are model parameters. There are two PARMS statements in this program. When PROC MCMC sees multiple PARMS statements, the procedure updates the parameters by using a blocked Metropolis algorithm, with the updating of each block of parameters conditional on fixed values of others. In this case, in each Markov chain iteration, PROC MCMC updates all the p parameters while holding a and b constant; then it proceeds to update a and b while fixing the p parameters.

The PRIOR statements specify the prior and hyperprior distributions. The expression "p:" is a shorthand notation for all symbols that begin with the letter "p". The order in which the PRIOR statements appear does not impact the program. The MODEL statement declares a binomial likelihood for each observation. The data set variable index is used in the p

array to indicate the correct probability parameter for each response. This way of setting up the model enables PROC MCMC to calculate the correct log likelihood for each observation.

Figure 12 reports the posterior summary and interval statistics of the random effects and the hyperparameters. The posterior p_i estimates are slightly different from each other, but not significantly.

Figure 12 Posterior Summary Statistics of the Random-Effects Model

The MCMC Procedure

Posterior Summaries

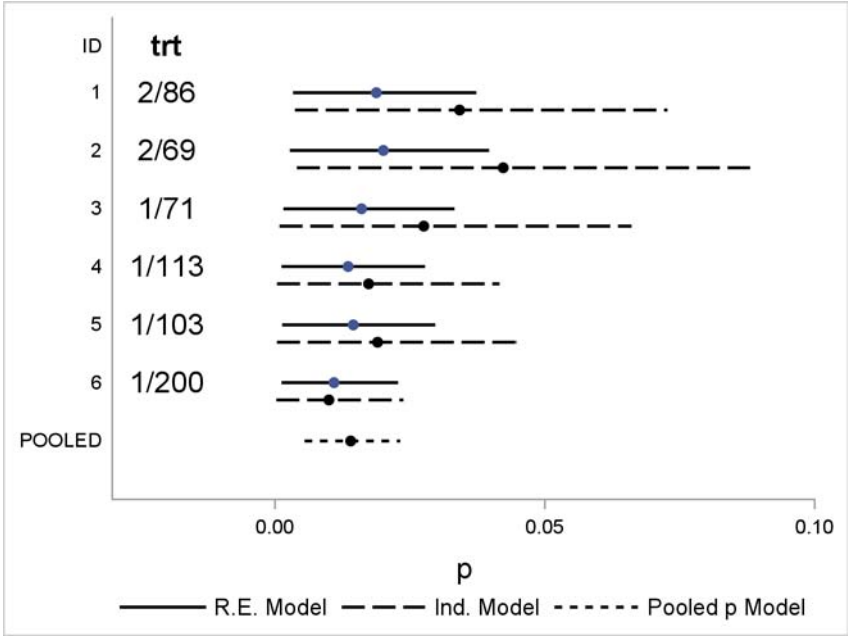
Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
p1	10000	0.0186	0.00935	0.0120	0.0172	0.0234
p2	10000	0.0197	0.0124	0.0122	0.0174	0.0242
p3	10000	0.0162	0.00900	0.00940	0.0147	0.0216
p4	10000	0.0137	0.00744	0.00828	0.0126	0.0177
p5	10000	0.0143	0.00773	0.00861	0.0131	0.0185
p6	10000	0.0101	0.00586	0.00565	0.00932	0.0134
a	10000	3.6526	2.4742	1.9639	3.0473	4.6801
b	10000	226.3	142.1	129.0	191.5	298.0

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
p1	0.050	0.00462	0.0400	0.00273	0.0362
p2	0.050	0.00475	0.0465	0.00252	0.0393
p3	0.050	0.00296	0.0383	0.00185	0.0344
p4	0.050	0.00308	0.0307	0.00141	0.0277
p5	0.050	0.00337	0.0327	0.00135	0.0295
p6	0.050	0.00146	0.0241	0.000568	0.0213
a	0.050	0.7811	10.1398	0.6338	8.5280
b	0.050	43.6428	598.1	23.1890	495.8

Figure 13 provides a better visualization of the estimated p_i. The solid lines indicate 95% credible intervals from the random-effects model estimates, and the dashed lines are from the independent model, in which each trial is modeled separately. The dotted line is from the pooled-p model, where all six trials share the same probability parameter. Pool ID 6 is the pilot data, which has a smaller estimate with a significantly shorter interval. As expected, the independent model estimates have wider ranges, and the random-effects model provides shrinkage that reduces the variances. The pooled-p model has the smallest credible interval because it uses all data to estimate a single p.

Figure 13 95% HPD Intervals from Three Different Fits of the Binomial Data.



META-ANALYSIS

Meta-analysis is a technique that combines information from different studies. If the studies are treated as exchangeable, Bayesian meta-analysis can be understood as a hierarchical model. Studies become different groups in a random-effects model, and the treatment effect from each center could be a random quantity from a larger population. For references on Bayesian meta-analysis, see Sutton and Higgins (2008), Stangl and Berry (2000), and Spiegelhalter, Abrams, and Myles (2004).

Suppose that the trt and pilot data sets are part of a multicenter case-control study. The data used so far come from the treatment arm, and the corresponding data from the control arm is presented here along with the original data from the previous data sets:

```
data multistudies;
   input trt trtN ctrl ctrlN;
   index = _n_;
   datalines;
2  86 1  70
2  69 1  94
1  71 1  78
1 113 2 127
1 103 0  91
1 200 1 142
;
```

Suppose that you are primarily interested in the treatment effect (on the survival benefit) of the medical procedure in each of the centers. Denote the treatment effect for the ith center to be θ_i. Following the specification of a binomial model as outlined in Spiegelhalter, Abrams, and Myles (2004), you can use the following model, where the treatment and control groups each have their own binomial likelihood functions, with treatment probability p_i and control probability q_i for every center:

$$
\begin{aligned}
\text{trt}_i &\sim \text{binomial}(\text{trtN}_i, p_i) \\
\text{ctrl}_i &\sim \text{binomial}(\text{ctrlN}_i, q_i) \\
\text{logit}(p_i) &= \theta_i + \phi_i \\
\text{logit}(q_i) &= \phi_i \\
\pi(\theta_i) &= \text{normal}(\mu, \sigma^2) \\
\pi(\mu)\pi(\phi_i) &= \text{uniform}(-10, 10) \\
\pi(\sigma) &= \text{uniform}(0, 50)
\end{aligned}
$$

Each control group center also has a baseline rate, or the logit of mortality rate, which is ϕ_i. The center treatment effect θ_i measures the effects of the procedure on the patients' survival probability. The treatment effect θ_i is also the logarithm of the odds ratio. θ_i share a normal prior within unknown mean μ and variance σ^2. The hyperprior distributions on μ and σ are noninformative, as is the prior on the baseline rates ϕ_i.

The following statements in PROC MCMC fit the described model:

```
proc mcmc data=multistudies nmc=50000 thin=5 seed=27 outpost=msout monitor=(theta);
   array phi[6];
   array theta[6];

   parm theta: 0.4 mu 0.4;
   parm phi: -4;
   parm sd 1;
   prior sd ~ uniform(0, 50);
   prior theta: ~ n(mu, sd=sd);
   prior mu phi: ~ uniform(-10, 10);
   p = logistic(theta[index] + phi[index]);
   model trt   ~ binomial(trtN, p);
   q = logistic(phi[index]);
   model ctrl ~ binomial(ctrlN, q);
run;
```

The MONITOR= option specifies the θ_i as the symbols of interest. The two ARRAY statements allocate arrays for the treatment effects and the baseline rates for each of the six centers. The PARMS and PRIOR statements declare model parameters and their prior distributions. The LOGISTIC function is used to perform the logistic transformation from the treatment effects and baseline rates to the corresponding success probability. Multiple MODEL statements are allowed in the MCMC procedure.

Figure 14 reports the posterior summary statistics of θ_i. Figure 15 shows 95% credible intervals of odds ratios from the six centers. The trial data are presented on the graph. You can see some variations of the treatment effects, although none significantly favors the treatment, because each interval contains the value 1.

Figure 14 Posterior Summary Statistics

The MCMC Procedure

Posterior Summaries

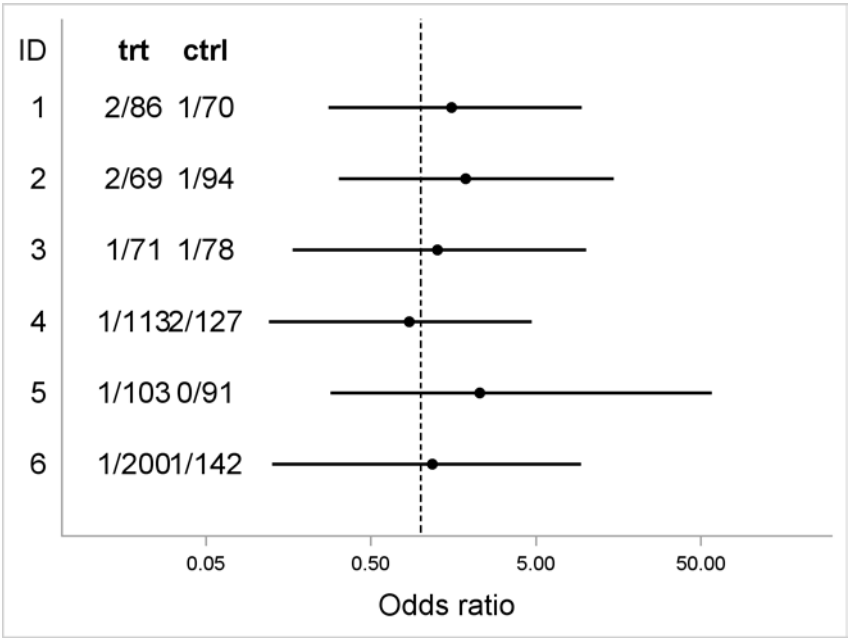
Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
theta1	10000	0.4320	0.8755	-0.1188	0.3622	0.9471
theta2	10000	0.6303	0.9509	0.0459	0.5535	1.1448
theta3	10000	0.2369	0.9943	-0.3609	0.2579	0.8290
theta4	10000	-0.1611	0.9552	-0.6662	-0.0634	0.4713
theta5	10000	0.8274	1.2876	0.0237	0.6348	1.4819
theta6	10000	0.1617	1.0642	-0.4421	0.1996	0.7331

Figure 14 *continued*

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
theta1	0.050	-1.2738	2.2710	-1.2861	2.2447
theta2	0.050	-1.0968	2.7633	-1.1408	2.6906
theta3	0.050	-1.8811	2.2441	-1.7871	2.3125
theta4	0.050	-2.3094	1.4474	-2.1230	1.5491
theta5	0.050	-1.3948	3.9437	-1.2590	4.0613
theta6	0.050	-2.0501	2.2716	-2.0736	2.2363

Figure 15 95% HPD Intervals for Different θ_i



SYNTAX AND DISTRIBUTIONS

The section provides an overview of the statements that are available in the MCMC procedure and the distributions that are supported. Additional information and more details are provided in the complete documentation referenced in the section "CONTACT INFORMATION" on page 22.

PROCEDURE SYNTAX

PROC MCMC *options*;

This statement invokes the procedure. Some useful options are:

DATA= names the input data set.

NBI= specifies the number of burn-in iterations.

NMC= specifies the number of MCMC iterations.

NTU= specifies the number of tuning iterations.

OUTPOST= names the output data set that stores the posterior samples.

PROPCOV= controls options for constructing the initial proposal covariance matrix.

SEED= specifies the random seed for simulation.

THIN= specifies the thinning rate.

ARRAY *arrayname*;

The ARRAY statement enables you to specify SAS arrays.

BEGINCNST;
ENDCNST;

The BEGINCNST and ENDCNST statements must work in pairs. They define a block, within which all programming statements are evaluated only in the setup stage of the simulation. These statements are best used to define constants.

BEGINPRIOR;
ENDPRIOR;

The BEGINPRIOR and ENDPRIOR statements must work in pairs. They define a block, within which all programming statements are evaluated independent of the input data set in each Markov chain iteration. These statements are best used to reduce unnecessary observation-level computations, such as calculating the prior density or functions of parameters.

BY *variables*;

The BY statement invokes standard SAS BY processing.

MODEL *dependent-variable* ~ *distribution;*

The MODEL statement specifies the dependent variable and its conditional distribution (likelihood function) given the parameters. When the GENERAL or DGENERAL functions are used in this statement, you do not need to specify the dependent variable.

PARMS *name* | *name-list* < numbers >

The PARMS statement specifies parameter names and their starting values. Parameters in separate PARMS statements are placed in different Metropolis blocks and updated iteratively in each iteration.

PRIOR *parameters-list* ~ *distribution;*

The PRIOR statement specifies the prior distributions of the model parameters.

Program Statements

Standard SAS programming statements enable you to construct a wide variety of prior distributions and likelihood functions. You can use the following in the program:

- assignment and operators: =, +, -, *, /, <>, <, and so on.
- mathematical functions: PDF, CDF, SDF, LOGPDF, ABS, LOG, LOGISTIC, FACT, BETA, GAMMA, RAND, and so on.
- statements: CALL, DO, IF, PUT, WHEN, and so on.

UDS;

The UDS statement enables you to apply a separate sampling algorithm (called a user-defined sampler), as a Gibbs step, in updating parameters in the model. You need to call PROC FCMP to define a sampling subroutine.

Standard Distributions

The MCMC procedure supports a number of continuous and discrete distributions that you can use to specify either the prior distribution for the parameters or the sampling distribution for the dependent variables. The distributions can be used in either the PRIOR or the MODEL statements. Distribution arguments can be constants, SAS functions and expressions, data set variables, or model parameters. Example statements are as follows:

```
prior alpha ~ cauchy(0, 2);
prior p ~ beta(abs(alpha), constant('pi'));
model y ~ binomial(n, p);
```

Table 1 lists the standard distributions supported by PROC MCMC.

Table 1 Standard Distributions

beta	binary	binomial	cauchy	chisq
expon	gamma	geo	ichisq	igamma
laplace	negbin	normal	pareto	poisson
sichisq	t	uniform	wald	weibull

Some distributions can be parameterized in different ways. For example, the normal distribution can take the scale parameter to be variance, standard deviation, or precision. In such distributions, you must explicitly name the ambiguous parameter. Example statements are as follows:

```
prior beta ~ normal(0, var=sigma2);
prior sigma2 ~ igamma(0.001, is=0.001);
```

Table 2 lists distributions that can be parameterized differently.

Table 2 Distributions with Ambiguous Argument

expon(scale|s = λ)	expon(iscale|is = λ)	
gamma(a, scale|sc = λ)	gamma(a, iscale|is = λ)	
igamma(a, scale|sc = λ)	igamma(a, iscale|is = λ)	
laplace(l, scale|sc = λ)	laplace(l, iscale|is = λ)	
normal(μ, var=σ^2)	normal(μ, sd=σ)	normal(μ, prec=τ)
lognormal(μ, var=σ^2)	lognormal(μ, sd=σ)	lognormal(μ, prec=τ)
t(μ, var=σ^2, df)	t(μ, sd=σ, df)	t(μ, prec=τ, df)

Nonstandard Distributions

The GENERAL and DGENERAL (discrete GENERAL) functions enable you to analyze data that have any prior or likelihood functions, as long as these functions are programmable using the DATA step functions. The argument to the GENERAL and DGENERAL functions must take the value of the logarithm of the prior or likelihood function.

Suppose that you want to specify a noninformative prior on the variance parameter in a normal model:

$$\pi(\sigma^2) \propto \frac{1}{\sigma^2}$$

On the logarithm scale, this prior becomes:

$$\log(\pi(\sigma^2)) = -\log(\sigma^2) + C$$

To specify this prior in PROC MCMC, you can use the following programming statements:

```
lp = -log(sigma2)
prior sigma2 ~ general(lp);
```

Note that the normalizing constant C can be ignored in the program, as long as it is independent of other parameters in the model.

When you use the GENERAL function to specify a joint prior distribution for multiple parameters, make sure that the entire parameter list appears in a single PRIOR statement and not in multiple PRIOR statements. For example, if you want to construct the joint prior distribution

$$\pi(a,b) \propto \frac{1}{a+b}$$

the correct way to program it in PROC MCMC is with the following statements:

```
lp = -log(a+b);
prior a b ~ general(lp);
```

If you use the following statements:

```
lp = -log(a+b);
prior a ~ general(lp);
prior b ~ general(lp);
```

The MCMC procedure interprets your model as

$$\pi(a) \cdot \pi(b) \propto \frac{1}{a+b} \cdot \frac{1}{a+b}$$

which would lead to an unintended posterior distribution.

IMPORTANT: You should be careful using the general functions because PROC MCMC cannot verify that the priors you specify are integrable distributions. You can easily construct a model with priors and a likelihood function that leads to an improper posterior distribution.

Truncated Distributions

Most distributions, with the exception of the binary and uniform distributions, allow for optional LOWER= and UPPER= arguments:

```
prior alpha ~ normal(0, var=1, lower=0);
prior beta ~ beta(2, 1, upper=0.7);
prior gamma ~ igamma(0.001, is=0.001, lower=0.1, upper=3);
```

The truncation arguments can be model parameters only in the normal and general distributions. See the procedure documentation for information about how to construct a prior distribution with model parameters as lower or upper bounds.

OUTPUT TABLE NAMES

The MCMC procedure uses SAS ODS. To convert any table to a SAS data set, use the ODS statement in the following form:

```
ods output 'table-name' = SAS-data-set;
```

TABLE-NAME is one of the names from the table below, and SAS-DATA-SET is a name you select for the output SAS data set.

Table 3 ODS Tables Produced in PROC MCMC

Table Name	Statement / Option
AutoCorr	default
PostSummaries	default
Corr	STATS=CORR
Cov	STATS=COV
DIC	DIC
ESS	default
MCSE	default
Geweke	default
Heidelberger	DIAGNOSTICS=HEIDEL
PostIntervals	default
NObs	default
Parameters	default
PosteriorSample	(for ODS output data set only)
Raftery	DIAGNOSTICS=RAFTERY
SamplingHistory	default
TuningHistory	default

To suppress all displayed output, use the following statement:

```
ods exclude all;
```

To display all output again, use the following statement:

```
ods select all;
```

PROCEDURE CAPABILITY AND LIMITATIONS

The MCMC procedure carries the entire input data set in machine memory during the analysis. This largely dictates the maximum scope of the problem that the procedure can handle. With respect to computation time, the running time is approximately linear to the number of samples in the input data set (nsamples), the number of simulations (nmc), and the number of blocks in the program (number of PARMS statements, nblocks). An analysis would require PROC MCMC to evaluate the log-likelihood function nsamples $\times$ nsim $\times$ nblocks number of times, excluding the tuning and burn-in phases.

The faster a computer evaluates a single log-likelihood function, the faster the program runs. Suppose that a program has nsamples=200, nmc=50,000 and nblocks=3. The MCMC procedure evaluates the log-likelihood function approximately a total number of 3×10^7 times, excluding tuning and burn-in. If the computer can evaluate a single log likelihood 10^6 times per second (typical performance of a PC nowadays), this program takes approximately a half-minute to run. If you want to increase the number of simulations fivefold, the run time also increases approximately fivefold.

This experimental version of PROC MCMC does not parse a statistical model symbolically and apply optimal sampling algorithms accordingly. The procedure can fit a wide range of models, but the random walk Metropolis is not an ideal algorithm for high-dimensional problems, such as large scale multilevel random-effects models. Improvements are planned for future releases that will add specialized algorithms for different scenarios. Also, the MCMC procedure currently does not support standard multivariate distributions, such as the multivariate normal or the inverse Wishart distributions. Nor does the procedure handle missing data automatically. However, you can use programming statements to handle these problems specifically.

CONCLUSIONS

The MCMC procedure offers a flexible environment for fitting a rich variety of linear and nonlinear, single-level and multilevel Bayesian models. The MCMC procedure evaluates Bayesian models, processes SAS programming statements, and uses a self-tuning random walk Metropolis algorithm to generate samples from the target posterior distributions. The syntax is intuitive and enables you to specify many standard distributions directly. In addition, you can use familiar DATA step and macro language programming to construct and estimate nonstandard statistical models.

Besides producing posterior estimates and convergence diagnostics, the MCMC procedure can also compute functions of parameters, make predictions, compare models, evaluate the fitness of a model, and even incorporate user-defined sampling algorithms. The MCMC procedure is an active area of development and additional features will be forthcoming in future releases.

REFERENCES

Berger, J. O. (1985), *Statistical Decision Theory and Bayesian Analysis*, Second Edition, New York: Springer-Verlag.

Bernardo, J. M. and Smith, A. F. M. (1994), *Bayesian Theory*, New York: John Wiley & Sons.

Box, G. E. P. and Tiao, G. C. (1992), *Bayesian Inference in Statistical Analysis*, New York: John Wiley & Sons.

Carlin, B. P. and Louis, T. A. (2008), *Bayesian Methods for Data Analysis*, London: Chapman & Hall/CRC.

Chen, M. H., Shao, Q. M., and Ibrahim, J. G. (2000), *Monte Carlo Methods in Bayesian Computation*, New York: Springer-Verlag.

DeGroot, M. H. and Schervish, M. J. (2002), *Probability and Statistics*, 3rd Edition, Reading, MA: Addison-Wesley.

Gelman, A., Carlin, J., Stern, H., and Rubin, D. (2004), *Bayesian Data Analysis*, Second Edition, London: Chapman & Hall.

Hastings, W. K. (1970), "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, 57, 97–109.

Ibrahim, J. G. and Chen, M.-H. (2000), "Power Prior Distributions for Regression Models," *Statistical Science*, 15, 46–60.

Lee, P. M. (2004), *Bayesian Statistics: An Introduction*, 3rd Edition, Arnold Publishers.

Leonard, T. and Hsu, J. S. (1999), *Bayesian Methods: An Analysis for Statisticians and Interdisciplinary Researchers*, Cambridge: Cambridge University Press.

Liu, J. S. (2001), *Monte Carlo Strategies in Scientific Computing*, Springer-Verlag.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953), "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, 21, 1087–1092.

Metropolis, N. and Ulam, S. (1949), "The Monte Carlo Method," *Journal of the American Statistical Association*, 44.

Robert, C. P. (2001), *The Bayesian Choice*, Second Edition, New York: Springer-Verlag.

Robert, C. P. and Casella, G. (2004), *Monte Carlo Statistical Methods*, second Edition, New York: Springer-Verlag.

Spiegelhalter, D., Abrams, K., and Myles, J. (2004), *Bayesian Approaches to Clinical Trials and Health-Care Evaluation*, Chichester, England: John Wiley & Sons.

Stangl, D. and Berry, D., eds. (2000), *Meta-analysis in Medicine and Health Science*, New York: Chapman & Hall/CRC.

Sutton, A. and Higgins, J. (2008), "Recent Development in Meta-analysis," *Statistics in Medicine*, 27(5), 625–650.

CONTACT INFORMATION

The MCMC procedure requires SAS 9.2. Complete documentation is available on the web at `http://support.sas.com/documentation/cdl/en/statugmcmc/61803/PDF/default/statugmcmc.pdf`.

You can find additional coding examples at `http://support.sas.com/rnd/app/examples/index.html`.

You may send feedback to:

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