Chapter 17:

Estimating Complex Models with Markov Chain Monte Carlo Simulation[1]

Dominique Lord
Zachry Dept. of Civil Engineering
Texas A & M University
College Station, TX

Ned Levine
Ned Levine & Associates
Houston, TX

Byung-Jung Park
Korea Transport Institute
Goyang, South Korea

Srinivas Geedipally
Texas Transportation Institute
Arlington, TX

Haiyan Teng
Houston, TX

Li Sheng
Houston, TX

[1] This chapter was the result of the efforts of several people. Dr. Shaw-pin Miaou of College Station, TX designed the MCMC algorithm for the Poisson-Gamma model. Dr. Byung-Jung Park modified the algorithm to incorporate Poisson-Lognormal and the MCMC binomial model. Dr. Srinivas Geeidpally added the MCMC Normal model. Dr. Dominique Lord provided technical consulting on the dispersion parameters in these models. Dr. Ned Levine developed the block sampling scheme and provided overall project management. Ms. Haiyan Teng and Dr. Li Sheng programmed the routines and added numerous technical improvements to the algorithms.

Table of Contents

Markov Chain Monte Carlo (MCMC)
Simulation of Regression Functions **17.1**
Hill Climbing Analogy 17.1
Bayesian Probability 17.3
Bayesian Inference 17.4
Markov Chain Sequences 17.4
MCMC Simulation 17.6
Step 1: Specifying a Model 17.6
1. Normal Model 17.6
2. Poisson-Gamma Model 17.8
3. Poisson-Lognormal Model 17.9
4. Logit Model 17.9
How to choose a model 17.9
Data with a large number of zeros 17.10
Step 2: Setting Up a Likelihood Function 17.11
Step 3: Defining a Joint Posterior Distribution 17.12
Step 4: Drawing Samples from the Full Conditional Distribution 17.13
Step 5: Summarizing the Results from the Sample 17.16
MCMC Output 17.16
Summary Statistics 17.16
Convergence Statistics 17.17
Example of Estimating Houston Burglaries with the MCMC Poisson-Gamma 17.18
Comparison of MCMC Poisson-Gamma with MLE Poisson-Gamma 17.18
Example of Estimating Houston Burglaries with the MCMC Normal 17.20
Comparison of MCMC Normal with MLE Normal 17.22
Why Run an MCMC when MLE is So Easy? 17.23
Example of Estimating Houston Burglaries with the MCMC Poisson-Lognormal 17.24
Risk Analysis **17.26**
Issues in MCMC Modeling **17.30**
Starting Values of Each Parameter 17.30
Example of Defining Prior Values for Parameters 17.31
Convergence 17.31
Monitoring Convergence 17.36
Statistically Testing Parameters 17.37
Proper Specification of a Model 17.37
Multicollinearity 17.38
Stepwise Variable Entry to Control Multicollinearity 17.41

Table of Contents (continued)

Overfitting 17.42
Condition Number of Matrix 17.43
Overfitting and Poor Prediction 17.43
Improving the Performance of the MCMC Algorithm 17.44
Scaling of the Data 17.45
Block Sampling Method for the MCMC 17.46
Comparison of Block Sampling Method with Full Dataset 17.48
Test 1 17.48
Test 2 17.50
Statistical Testing with Block Sampling Method 17.50
References 17.53

Chapter 17:
Estimating Complex Models with Markov Chain Monte Carlo Simulation

In this chapter, we examine the Markov Chain Monte Carlo (MCMC) method for estimating complex models. We apply it to the family of Poisson models for modeling count data.

Markov Chain Monte Carlo (MCMC) Simulation of Regression Functions

To estimate a regression model from a complex function, we use a simulation approach called *Markov Chain Monte Carlo* (or MCMC). Chapter 12 of the *CrimeStat* manual discussed the Correlated Walk Analysis (CWA) routines. This was an example of a *random walk* whereby each step follows from the previous step. That is, a new position is defined only with respect to the previous position. This is an example of a Markov Chain.

In recent years, there have been numerous attempts to utilize this methodology for simulating regression and other models using a Bayesian approach (Lynch, 2007; Gelman, Carlin, Stern, & Rubin, 2004; Lee, 2004; Denison, Holmes, Mallick & Smith, 2002; Carlin & Louis, 2000; Leonard & Hsu, 1999).

Hill Climbing Analogy

To understand the MCMC approach, let us use a 'hill climbing' analogy. Imagine a mountain climber who wants to climb the highest mountain in a mountain range (for example, Mt. Everest in the Himalaya mountain range). However, suppose a cloud cover has descended on the range such that the tops of mountains cannot be seen; in fact, assume that only the bases of the mountains can be seen. Without a map, how does the climber find the mountain with the highest peak and then climb it? Realistically, of course, no climber is going to try to climb without a map and, certainly, without good visibility. But, for the sake of the exercise, think of how this could be done.

First, the climber could adopt a gradient approach with a systematic walking pattern. For example, he/she takes a step. If the step is higher than the current elevation (i.e., it is uphill), the climber then accepts the new position and moves to it. On the other hand, if the step is at the same or a lower elevation as the current elevation, the step is rejected. After each iteration (accepting or rejecting the new step), the procedure continues. Such a procedure is sometimes called a *greedy algorithm* because it optimizes the decision in incremental steps (local

optimization; Wikipedia, 2010a; Cormen, Leiserson, Rivest, & Stein; 2009; So, Ye, & Zhang, 2007; Dijkstra, 1959).

This strategy can be useful if there is a single mountain to climb (i.e., it is convex throughout or at least in the vicinity of the highest peak). Because generally moving uphill means moving towards the peak of the mountain, this approach will often lead the climber to get to the peak if the mountain is smooth. For a single mountain, a greedy algorithm such as our hill climbing example often works fine. The Maximum Likelihood Estimation (MLE) method is similar to this in that it requires a smooth convex function for which each step upward is assumed to be climbing the mountain. For functions that are smooth and convex, such as the single-parameter exponential family, this algorithm will work very well. The algorithm goes under different names but a common one is the *method of steepest ascent* (Goldfield, Quandt, & Trotter, 1966).

But, if there are multiple mountains (i.e., a range of mountains), how can we be sure that the peak that is climbed is really that of the highest mountain? In other words, again, without a map, for a range of mountains where there are multiple peaks but with only one being the highest, there is no guarantee that this greedy algorithm will find the single highest peak. Greedy algorithms work for simple problems but not necessarily for complex ones. Because they optimize the local decision process, they will not necessarily see the best approach for the whole problem - the global decision process (Goldfield, Quandt, & Trotter, 1966).

In other words, there are two problems that the climber faces. First, he/she does not know where to start. For this a 'map' would be ideal. Second, the search strategy of always choosing the step that goes up does not allow the climber to find alternative routes. Hills or mountains, as we all know, are rarely perfectly smooth; there are crevices and ridges and undulations in the gradient so that a climber will not always be going up in scaling a mountain. Instead, a climber needs to search a larger area in order to find a path that really does go up to the peak (sampling, if you wish).

This is the main reason why the MLE approach cannot estimate the parameters of a complex function since the approach works only for functions that are part of the single-parameter exponential family; they are closed-form functions for which there is a simple maxima that can be estimated. For these functions, which are very common, the MLE is a good approach. These functions are perfectly smooth which will allow a greedy algorithm to work. All of the generalized linear model functions – Ordinary Least Squares (OLS), Poisson, negative binomial, binomial probit, and others, can be solved with the MLE approach.

However, for a two or higher-parameter family, the approach will not work because there may be multiple peaks and a simple optimization approach will not necessarily discover the

highest likelihood. In fact, for a complex surface, MLE may get stuck on a local peak (a local optimum) and not have a way to backtrack in order to find another peak which is truly the highest.

For these, one needs a map for a good starting location and a sampling strategy that allows the exploration of a larger area than just that defined by a greedy algorithm. The 'map' comes from a Bayesian approach to the problem and the alternative search strategy comes from a sampling approach. This is essentially the logic behind the MCMC method.

Bayesian Probability

Let us start with the 'map' and briefly review the information that was discussed in Chapter 14. Bayes Theorem is a formulation that relates the conditional and marginal probability distributions of random variables. The *marginal probability* distribution is a probability independent of any other conditions. Hence, P(A) and P(B) is the marginal probability (or just plain probability) of A and B respectively.

The *conditional probability* is the probability of an event given that some other event has occurred. It is written in the form of P(A|B) (i.e., event A given that event B has occurred). In probability theory, it is defined as:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \tag{17.1}$$

or

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)} \tag{17.2}$$

where the symbol $\cap$ represents the logical concept of 'and' (the Boolean intersection of A and B), which we expressed in words in Chapter 14. We will use the mathematical symbol now.

Bayes Theorem relates the two equivalents of the 'and' condition together.

$$P(B) \times P(A \mid B) = P(A) \times P(B \mid A) \tag{17.3}$$

$$P(A \mid B) = \frac{P(A) \times P(B \mid A)}{P(B)} \tag{17.4}$$

or

$$P(B \mid A) = \frac{P(B) \times P(A \mid B)}{P(A)} \tag{17.5}$$

Bayesian Inference

In the statistical interpretation of Bayes Theorem, the probabilities are estimates of a random variable. Let θ be a parameter of interest and let X be some data. Thus, Bayes Theorem can be expressed as:

$$P(\theta \mid X) = \frac{P(X \mid \theta) \times P(\theta)}{P(X)} \tag{17.6}$$

Interpreting this equation, $P(\theta \mid X)$ is the probability of θ given the data, X. $P(\theta)$ is the probability that θ has a certain distribution and is usually called the *prior probability*. $P(X \mid \theta)$ is the probability that the data would be obtained given that θ is true and is usually called the *likelihood function* (i.e., it is the likelihood that the data will be obtained given θ. Finally, $P(X)$ is the marginal probability of the data, the probability of obtaining the data under all possible scenarios of θ's.

The data are what was obtained from some data gathering exercise (either from experiments or observations). Since the prior probability of obtaining the data (the denominator of the above equation) is not known or cannot easily be evaluated, it is not easy to estimate it. Consequently, often the numerator only is used for estimating the posterior probability since

$$P(\theta \mid X) \propto P(X \mid \theta) \times P(\theta) \tag{17.7}$$

where $\propto$ means 'proportional to'. Because probabilities must sum to 1.0, the final result can be re-scaled so that the probabilities of all entities do sum to 1.0. The prior probability, $P(\theta)$, essentially is the 'map' in the hill climbing analogy discussed above! It points the way towards the correct solution.

The key point behind this logic is that an estimate of a parameter can be updated by additional information. The formula requires that a prior value for the estimate be given with new information being added that is *conditional* on the prior estimate, meaning that it factors in information from the prior. Bayesian approaches are increasingly being used to provide estimates for complex calculations that previously were intractable (Denison, Holmes, Mallilck, & Smith, 2002; Lee, 2004; Gelman, Carlin, Stern, & Rubin, 2004).

Markov Chain Sequences

Now, let us look at an alternative search strategy, the MCMC strategy. Unlike a conventional random number generator that generates independent samples from the distribution

of a random variable, the MCMC technique simulates a Markov chain with a limiting distribution equal to a specified target distribution. In other words, a Markov chain is a sequence of samples generated from a random variable in which the probability of occurrence of each sample depends only on the previous one. More specifically, a conventional random number generator draws a sample of size N and stops. It is non-iterative and there is no notion of the generator converging. We simply require N to be sufficiently large to produce reliable statistics.

An MCMC algorithm, on the other hand, is iterative with the generation of the next sample dependent on the value of the current sample. The algorithm requires us to sample until convergence has been obtained. The initial values of an MCMC algorithm are usually chosen arbitrarily and samples generated from one iteration to the next are correlated (autocorrelation). Consequently, the question of when we can safely accept the output from the algorithm as coming from the target distribution gets complicated and is an important topic in MCMC (convergence monitoring and diagnosis).

The MCMC algorithm involves five conceptual steps for estimating the parameter:

1. The user specifies a functional model and sets up the model parameters.

2. A likelihood function is set up and prior distributions for each parameter are assumed.

3. A joint posterior distribution for all unknown parameters is defined by multiplying the likelihood and the priors as in equation 17.7.

4. Repeated samples are drawn from this joint posterior distribution. However, it is difficult to directly sample from the joint distribution since the joint distribution is usually multi-dimensional. The parameters are, instead, sampled sequentially from their full conditional distributions, one at a time holding all existing parameters constant. This is the *Markov Chain* part of the MCMC algorithm. Typically, because it takes the chain a while to reach an *equilibrium* state, the early samples are thrown out ('burn-in') and the results are summarized based on the M-L samples where M is the total number of iterations and L are the discarded ('burn-in') samples (Miaou, 2006).

5. The estimates for all coefficients are based on the results of the M-L samples, for example the mean, the standard deviation, the median and various percentiles. Similarly, the overall model fit is based on the M-L samples.

MCMC Simulation

Each of these conceptual steps is complex, of course, and involves some detail. The following represents a brief discussion of the steps. In Appendix C, Dominique Lord and Byung-Jung Park presents a more formal discussion of the MCMC method in the context of the Poisson-Gamma-CAR model.

Step 1: Specifying a Model

The MCMC algorithm can be used for many different types of models. In this version of *CrimeStat*, we examine four types of MCMC model: the normal model, two non-spatial Poisson regression models (plus a Logit model that will be discussed in Chapter 18).

The normal model is an MCMC variant on the MLE Ordinary Least Squares. The two Poisson models (Poisson-Gamma and Poisson-Lognormal) are used to test over-dispersion while the NB1 model, discussed in Chapter 16, can be used to test under-dispersion. Figure 17.1 (which is a repeat of Figure 16.3) illustrates three types of dispersion. Note that over-dispersion is more extreme than under-dispersion though both are skewed. One has to use one of the Poisson family models with skewed count data to avoid introducing bias (see Chapter 15 for a discussion of bias from the use of an Ordinary Least Squares model).

Irrespective of the model used, in the Bayesian approach, prior probabilities have to be assigned to all unknown parameters, β, ψ, τ, ν. It is usually assumed that the β_k coefficients follow a *multivariate normal* distribution with $k+1$ dimensions:

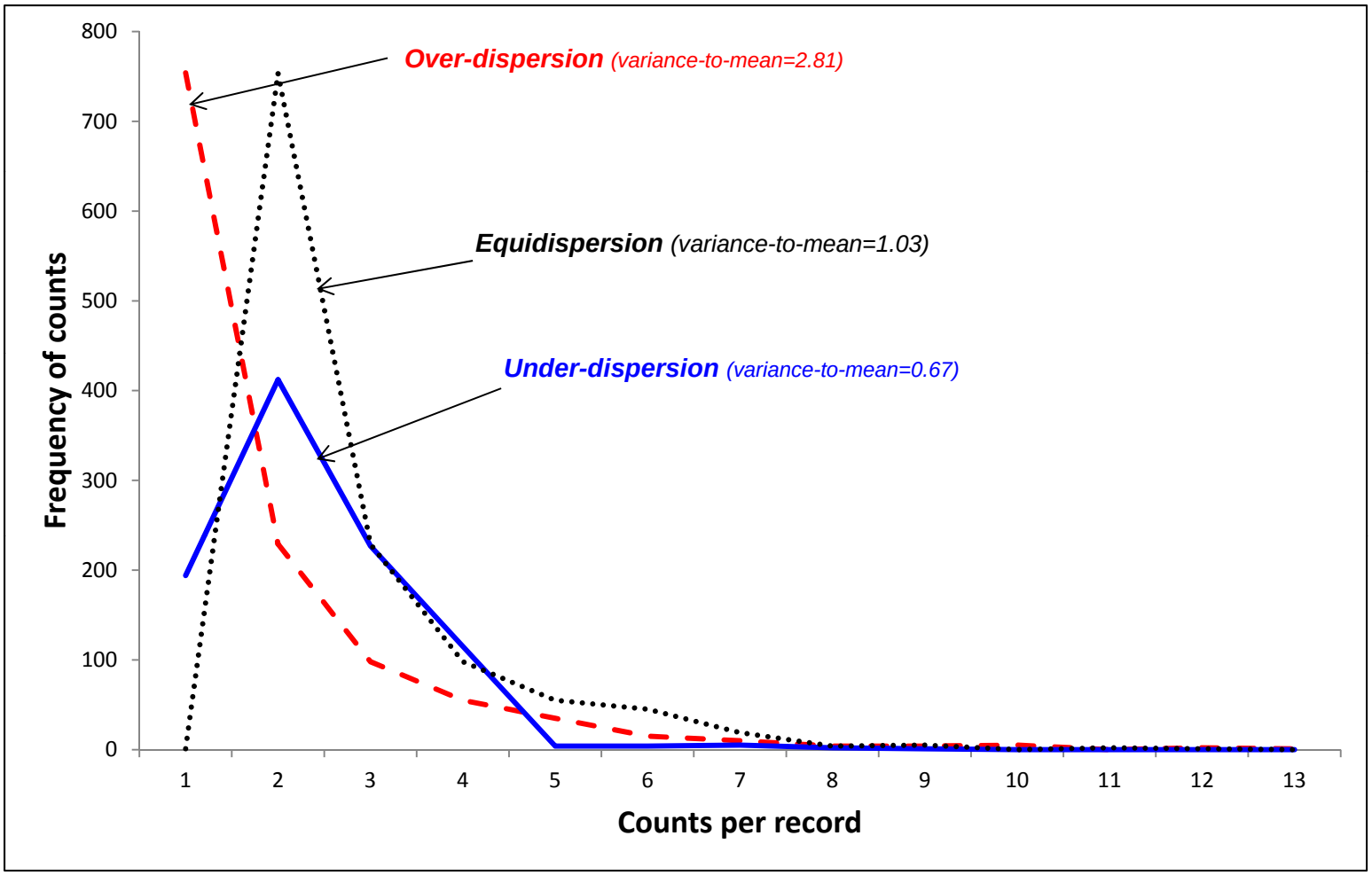
$$\boldsymbol{\beta} \sim MVN_{k+1}(\mathbf{b}_0, \mathbf{B}_0) \tag{17.8}$$

where MVN_{k+1} indicates a multivariate normal distribution with $k+1$ dimensions, and $\mathbf{b}_0$ and $\mathbf{B}_0$ are *hyperparameters* (parameters that define the multivariate normal distribution). For a non-informative prior specification, we usually assume $\mathbf{b}_0 = (0, \cdots, 0)^T$ and a large variance $\mathbf{B}_0 = 100\mathbf{I}_{k+1}$, where $\mathbf{I}_{k+1}$ denotes the $(k+1)$-dimensional identity matrix. Alternatively, independent normal priors can be placed on each of the regression parameters, e.g. $\beta_k \sim N(0, 100)$. If no prior information is known about $\boldsymbol{\beta}$, then sometimes a *flat* uniform prior is also used, $\beta_j \sim U(-\infty, \infty)$.

1. **Normal Model.** This is similar to the Ordinary Least Squares model discussed in Chapter 15 in that it assumes the dependent variable is normally-distributed. However, it is estimated by the MCMC algorithm rather than by MLE.

Figure 17.1:

Skewed Distributions and Type of Dispersion



The dependent variable *y* is a function of an expected mean for observation i and an error term, ε_i:

$$y_i = \lambda_i + \varepsilon_i \tag{17.9}$$

where λ_i is the predicted value of *y* and is a function of *k* independent variables (covariates),

$$\lambda_i = \mathrm{x}_i^T \beta \tag{17.10}$$

$\boldsymbol{\beta}$ is a vector of unknown coefficients for the *k* covariates plus an intercept. The error terms ε are independently and identically distributed as normal. Formally, it is defined as:

$$\varepsilon_i \sim Normal\ (0, \tau) \tag{17.11}$$

with τ being the variance The model error, ε_i, is independent of all covariates. The variance, τ, is assumed to follow a gamma distribution with a mean equal to 1 and a variance equal to $\tau = 1/\psi$ where ψ is a parameter that is greater than 0. The assumption on the uncorrelated error term ε_i is that it is constant for all observations. From equation 17.9, it follows that

$$y_i \sim Normal\ (\lambda_i, \tau) \tag{17.12}$$

2. **Poisson-Gamma Model**. This is similar to the negative binomial model discussed in Chapter 16 except that it is estimated by MCMC rather than by MLE. The Poisson-Gamma model is used when there is over-dispersion in the dependent variable. Formally, it is defined as:

$$y_i \mid \lambda_i \sim Poisson(\lambda_i) \tag{17.13}$$

The Poisson mean λ_i is organized as:

$$\lambda_i = \exp(\mathbf{x}_i^T \boldsymbol{\beta} + \varepsilon_i) \tag{17.14}$$

where exp() is an exponential function, $\boldsymbol{\beta}$ is a vector of unknown coefficients for the *k* covariates plus an intercept, and ε_i is the model error independent of all covariates. The error, $\exp(\varepsilon_i)$, is assumed to follow a gamma distribution with a mean equal to 1 and a variance equal to $\tau = 1/\psi$ where ψ is a parameter that is greater than 0 (Lord, 2006; Cameron & Trivedi, 1998).

3. **Poisson-Lognormal Model.** The Poisson-Lognormal model is an alternative to the Poisson-Gamma. It is useful when there is over-dispersion and when there is a small sample size (less than 50) and the sample mean is low (<1.0; Park & Lord, 2007). It has been used in a number of transportation studies to model motor vehicle crashes (El-Basyouny & Sayed, 2009) and has been adapted to the Bayesian approach by Ma, Kockelman and Damien (2008). Like the Poisson-Gamma model, the Poisson-Lognormal model is defined as:

$$y_i \mid \lambda_i \sim Poisson(\lambda_i) \tag{17.15}$$

The Poisson mean λ_i is organized as:

$$\lambda_i = \exp(\mathbf{x}_i^T \boldsymbol{\beta} + \varepsilon_i) \tag{17.16}$$

where exp() is an exponential function, $\boldsymbol{\beta}$ is a vector of unknown coefficients for the *k* covariates plus an intercept, and ε_i is the model error independent of all covariates. Unlike the Poisson-Gamma model, the error, $\exp(\varepsilon_i)$, is assumed to follow the ***lognormal distribution*** with a mean equal to 0 and a variance equal to $\sigma_\varepsilon^{-2} = \tau_\varepsilon \sim Gamma\ (a_\varepsilon, b_\varepsilon)$.

The reader is referred to Lord and Miranda-Moreno (2008) for additional details about the parameterization of the Poisson-lognormal model.

4. **Logit Model.** *CrimeStat* also includes an MCMC Logit model, but this will be discussed in Chapter 18.

How to Choose a Model

A key issue is how to choose among these alternatives. Overall, the two Poisson-based MCMC models give similar coefficients because the expected value is always estimated with a Poisson function. They differ primarily in the dispersion terms. The user is advised to first run an MLE Poisson model and check the diagnostics box. The diagnostics routine provides information on whether the dependent variable (the count) is significantly skewed while the dispersion parameter from the MLE Poisson model provides information on whether the conditional mean (the mean after controlling for the independent predictors) is still skewed. Further, for a spatial model (discussed in Chapter 19), the diagnostics routine will provide guidelines for the distance decay parameter (alpha).

While more research is clearly needed, a simple set of guidelines are as follows:

A. If the dependent variable is not significantly skewed (as indicated by the significance level of the "*g"* skewness test in the diagnostics routine), then run an OLS model.

B. If the "g" test of the dependent variable shows significant skewness and the ratio of the sample variance to the sample is greater than 2.0, then run an MLE or MCMC negative binomial (Poisson-Gamma) model since the negative binomial is a robust version of the Poisson. This is particularly true when the data set is larger than 50 cases and when the sample mean is 1.0 or greater. Note that the Poisson-lognormal model will provide similar results. However, the negative binomial is the usual model used with skewed data.

C. If the dispersion parameter in the negative binomial model is very close to 0 and is not significant, then the MLE Poisson model can be used. This is a case of equi-dispersion. However, in our experience very few data sets will show actual equi-dispersion. The vast majority are over-dispersed while some are under-dispersed.

D. If the "g" test of the dependent variable shows significant skewness and the ratio of the sample variance to the sample mean is greater than 2.0 but the sample size is less than 50 and the sample mean is less than 1.0, then use the MCMC Poisson-Lognormal model because it is a more robust model than the Poisson-Gamma with small samples and low sample means.

E. Finally, if the "g" test of the dependent variable shows significant skewness but the dispersion parameter in the negative binomial is less than 0, then use the NB1 model that was discussed in Chapter 16. This is a case of under-dispersion where the conditional variance is less than the conditional mean.

F. For all of these tests, the user should be aware of extreme outliers and multicollinearity among the independent variables (i.e., eliminate overlapping, multicollinear variables) as this can cause instability in the coefficients as well as cause models to shift from over-dispersion to under-dispersion, or vice versa.

Data with a Large Number of Zeros

The available Poisson models will handle the vast majority of data sets with count data. However, very occasionally, a data set with an extreme number of zeros will be found (e.g., 70%

or more of the cases have zero for the dependent variable). In cases where the dataset contains a large amount of zeros, traditional models, such as the Poisson-gamma or the Poisson-lognormal, can provide biased estimates or have difficulties converging. To overcome this problem, Poisson and negative binomial zero-inflated (ZI) models could be used (Lambert, 1992), as long as the model properly characterizes the data generating process (Lord et al., 2005). More recently, the Negative Binomial-Lindley (NB-L) distribution has been proposed to model datasets with a large number of zeros (Ghitany et al., 2008; Lord and Geedipally, 2011). The NB-L distribution is, as the name implies, a mixture of the NB and the Lindley distributions (Lindley, 1958; Ghitany et al., 2008). This two-parameter distribution has interesting and thorough theoretical properties in which the distribution is characterized by a single long-term mean that is never equal to zero and a single variance function, similar to the traditional NB distribution. This year, Geedipally et al. (2012) were able to fully develop the NB-L generalized linear model. The model has, in fact, been found to perform much better than the ZI models. The NB-L may be incorporated in a future version of *CrimeStat*.

Step 2: Setting up a Likelihood Function

For any of these types of non-spatial Poisson model, the log likelihood function is set up as a sum of individual logarithms of the model. In the case of the Poisson-Gamma model, the log likelihood function is:

$$L = \sum_{i=1}^{n} \left\{ \left(\sum_{j=0}^{y-1} \ln\left(j+\psi\right) \right) - \ln y_i ! - \left(y_i + \psi\right) \ln\left(1+\psi^{-1} \nu_i e^{\theta_i}\right) + y_i \ln \psi^{-1} + y_i \ln\left(\nu_i + \theta_i\right) \right\} \qquad (17.17)$$

with y_i being the observed (actual) value of the dependent variable, λ_i being the posterior *mean* of each site, $\theta_i = \ln \lambda_i$, ψ is the inverse dispersion parameter, and ν_i is an offset ('at risk') variable.

For the Poisson-Lognormal model, the log likelihood function is:

$$L = \ln\left(\prod_{i=1}^{n} \frac{e^{-(\nu_i \lambda_i)} (\nu_i \lambda_i)^{y_i}}{y_i !} \right) = \sum_{i=1}^{n} \left\{ \left[\nu_i e^{\theta_i} + y_i (\ln \nu_i + \theta_i) - \log \Gamma(y_i + 1) \right] \right\} \qquad (17.18)$$

with y_i being the observed (actual) value of the dependent variable, λ_i being the posterior *mean* of each site, $\theta_i = \ln \lambda_i$, and ν_i is an offset ('at risk') variable.

Step 3: Defining a Joint Posterior Distribution

In the case of the Poisson-Gamma model, the posterior probability, $p(\lambda, \beta, \psi \mid y, a_\omega, b_\omega)$, of the joint posterior distribution is defined as:

$$\pi(\lambda, \beta, \psi|y, a_\omega, b_\omega) \propto f(y|\nu\lambda) \cdot \pi(\lambda|\boldsymbol{\beta}, \psi) \cdot \pi(\beta_1) \cdots \pi(\beta_J) \cdot \pi(\psi|a_\omega, b_\omega) \qquad (17.19)$$

where y_i is the observed value of the dependent variable, **β** are the coefficients of each independent variable, ψ is the inverse dispersion parameter, while α_ω and b_ω are hyperparameters estimated internally in the routine. The equation is not in standard form (Park, 2009). Note that this is a general formulation. The parameters of interest are $(\lambda_1, \cdots \lambda_n)$, $(\beta_1, \cdots \beta_J)$, and ψ.

For the Poisson-Lognormal, the posterior probability, $p(\lambda, \beta, \tau_\varepsilon \mid y, a_\varepsilon, b_\varepsilon)$, of the joint posterior distribution is defined as:

$$\pi(\lambda, \beta, \tau_\varepsilon|y, a_\varepsilon, b_\varepsilon) \propto f(y|\nu\lambda) \cdot \pi(\lambda|\boldsymbol{\beta}, \tau_\varepsilon) \cdot \pi(\beta_1) \cdots \pi(\beta_J) \cdot \pi(\tau_\varepsilon|a_\varepsilon, b_\varepsilon) \qquad (17.20)$$

where y_i is the observed value of the dependent variable, $\boldsymbol{\beta}$ are the coefficients of the independent variable, λ is the Poisson mean, τ_ε is the inverse of the variance and is Gamma distributed, and a and b are hyperparameters that are estimated internally in the routine.

In all the cases, since it is difficult to draw samples of the parameters from the joint posterior distribution, we usually draw samples for each parameter from its full conditional distribution sequentially. This is an iterative process (the Markov Chain part of the algorithm).

Prior distributions for these parameters have to be assigned. In the *CrimeStat* implementation, there is a parameter dialogue box that allows estimates for each of the parameters (including the intercept). On the other hand, if the user does not know which values to assign as prior probabilities, very vague values are used as default conditions to simulate what is known as *non-informative* priors (essentially, vague information). Sometimes these are known as *flat priors* if they assume all values are likely. In *CrimeStat*, we assign a default value for the expected coefficients of 0 and a very large variance. As mentioned, the user can substitute more precise values for the expected value of the coefficients or the variance (based on previous research, for example). Generally, having more precise prior values for the parameters will lead to quicker convergence and a more accurate estimate.

Step 4: Drawing Samples from the Full Conditional Distribution

While there are several approaches to sampling from a joint posterior distribution, the particular sampling algorithm used in *CrimeStat* is a *Metropolis-Hastings* (or MH) algorithm within a Gibbs framework.[2] The MH algorithm is a general procedure for estimating the value of parameters of a function (Hastings, 1970; Metropolis, Rosenbluth, Rosenbluth, Teller & Teller, 1953). It was developed during the U. S. Hydrogen Bomb project by Rosenbluth and his colleagues and improved by Hastings.[3] Gibbs sampling requires the specification of a *full conditional distribution* for each parameter in which the probability is proportional to a simpler distribution that includes only variables (with all constants being dropped). If the full conditionals can be specified, then this is the most efficient. However, since the full conditional distribution itself can be complicated (and becomes more so when a spatial component is added), the MH algorithm can be used to sample from a distribution that represents the *target* distribution. With this algorithm, we do not need to sample directly from the target distribution but from an approximation called a *proposal* distribution (Lynch, 2007). The actual sampling of parameters within those two algorithms is done through either direct sampling or slice sampling depending on the function.[4]

The basic MH algorithm consists of six steps (Train, 2009; Lynch, 2007; Denison, Holmes, Mallick, & Smith, 2002).

1. Define the functional form of the target distribution and establish starting values for each parameter that is to be estimated, θ_0. For the first iteration, the existing value of the parameter, θ_E, will equal θ_0. Set t=1.

2. Draw a candidate parameter from a proposal density, θ_C.

3. Compute the posterior probability of the candidate parameter and divide it by the posterior probability of the existing parameter. Call this R.

4. If R is greater than 1, then accept the proposal density, θ_C.

---

[2] For information on the MH algorithm, see Gelman, Carlin, Stern & Rubin (2004) and Denison, Holmes, Mallick, &Smith, (2002). For information on the Gibbs algorithm, see Lynch (2008); Gelman, Carlin, Stern & Rubin (2004); and Denison, Holmes, Mallick, & Smith (2002).

[3] It is called Metropolis-Hasting because Nicolas Metropolis was the first name listed on the paper. However, the math was developed mostly by Marshall Rosenbluth with the idea proposed by Edward Teller and the programming done by Arianna Rosenbluth (Wikipedia, 2012).

[4] Slice sampling is a way of drawing random samples from a distribution by alternative horizontal and vertical sampling under the density distribution (Radford, 2003).

5. If R is not greater than 1, compare it to a random number drawn from a uniform distribution that varies from 0 to 1, u. If R is greater than u, accept the candidate parameter, θ_C. If R is not greater than u, keep the existing parameter θ_E.

6. Return to step 2 and keep drawing samples until sufficient draws are obtained.

Let us discuss these steps briefly. In the first step, an initial value of the parameter is taken. It is assumed that the functional form of the target population is known and has been defined (e.g., the target is a Poisson-Gamma function, a Poisson-Gamma-CAR, a Poisson-Lognormal-SAR function, a Binomial logit-CAR, etc.). The initial value should be consistent with this function. As mentioned above, a *non-informative* prior value can be selected.

Second, for each parameter in turn, a value is selected from a proposal density distribution. It is considered a 'candidate' since it is not automatically accepted as a draw from the target distribution. The proposal density can take any form that is easy to sample from, such as a normal distribution or a uniform distribution though usually the normal is used. Also, usually the distribution is symmetric though the algorithm can work for non-symmetric proposal distributions, too (Lynch, 2007, 109-112). In the *CrimeStat* implementation, we use a normal distribution. The proposal distribution does not have to be centered over the previous value of the parameter.

Third, the ratio of the posterior probability of the candidate parameter to the posterior probability of the existing parameter is calculated. This is called the *Acceptance* probability and is defined as:

$$\text{Acceptance probability} = R = \frac{f(\theta_C) * g(\theta_E)}{f(\theta_E) * g(\theta_C)} \tag{17.21}$$

The acceptance probability is made up of the product of two ratios. The function f is the target distribution and the function g is the proposal distribution. The first ratio, $f(\theta_C) * f(\theta_E)$, is the ratio of the densities of the target function using the candidate parameter in the numerator relative to the existing parameter in the denominator. That is, with the target function (the function for which we are trying to estimate the parameter values), we calculate the density using the candidate value and then divide this by the density using the existing value. Lynch (2007) calls it the *importance ratio* since the ratio will be greater than 1 if the candidate value yields a higher density (and, consequently, higher probability) than the existing one.

The second ratio, $g(\theta_E) * g(\theta_C)$, is the ratio of the proposal density using the existing value to the proposal density with the candidate value. This latter ratio adjusts for the fact that some candidate values may be selected more often than others (especially with asymmetrical

proposal functions). Note that the first ratio involves the target function densities whereas the second ratio involves the proposal function densities. If the proposal density is symmetric, then the second ratio will only have a very small effect.

Fourth, if R is greater than 1, meaning that the proposal density is greater than the original density, the candidate is accepted. However, if R is not greater than 1, this does not mean that the candidate is rejected but is instead compared to a random draw (otherwise we would have a 'greedy algorithm' that would only find local maxima).

Fifth, a random number, *u*, that varies from 0 to 1 is drawn from a uniform distribution and compared to R. If R is greater than *u*, then the value of the candidate parameter is accepted and becomes the new 'existing' parameter. Otherwise, if R is not greater than *u*, the existing parameter remains. Finally, in the sixth step, we repeat this algorithm and keep drawing samples until the desired sample size is reached.

Now what does this procedure do? Essentially, it draws values from the proposal distribution that increase the probability obtained from the target distribution. That is, generally only candidate values that increase the importance ratio will be accepted. But, this will not happen automatically (as, for example, in a greedy algorithm) since the ratio has to be compared to a random number, *u*, from 0 to 1. In the early steps of the algorithm, the random number may be higher than the existing R since it varies from 0 to 1. Thus, the candidate value is initially rejected more because it does not contribute to a high R ratio.

But, slowly, the acceptance probability will start to be accepted more often than the random draw since the candidate value will slowly approximate the true value of the parameter as it maximizes the target function's probability. Using the hill climbing analogy, the climber will wander around initially going in different directions but will slowly start to climb the hill and, most likely, the hill that is highest in the nearby vicinity. Each step that goes up will be accepted. But, each step that goes down will not necessarily be rejected since it is compared with a random 'step'. Thus, the climber explores other directions than just 'up'. But, over time, the climber will slowly move upward and, probably, more likely climb the highest hill nearby.

It is still possible for this algorithm to find a local 'peak' rather than the highest 'peak' since it explores in the vicinity of the starting location. To truly climb the highest peak, the algorithm needs a good starting value. Where does this 'good' starting value come from? Earlier research can be one basis for choosing a likely starting point. The more a researcher knows about a phenomenon, the better the researcher can utilize that information to ensure that the algorithm starts at a likely place. Without previous research to provide that value, however, Lynch (2007) proposes using the MLE approach to calculate parameters that are used as the initial values. That is, for a common distribution, such as the negative binomial, we use the

MLE negative binomial to estimate the values of the coefficients and intercept and then plug these into the MCMC routine as the initial values for that algorithm. *CrimeStat* allows the defining of initial values for the coefficients in the MCMC routine.

Step 5: Summarizing the Results from the Sample

Finally, after a sufficient number of samples have been drawn, the results can be summarized by analyzing the sample. That is, if a sample is drawn from a target population (using the MH approach or another one, such as the Gibbs method), then the distribution of the sample parameters is our best guess for the distribution of the parameters of the target function. The mean of each parameter would be the best guess for the coefficient value of the parameter in the target function. Similarly, the standard deviation of the sample values would be the best guess for the standard error of the parameter in the target distribution.

Credible intervals can be estimated by taking percentiles of the distribution. This is the Bayesian equivalent to a confidence interval in that it is estimated from a sample rather than from an asymptotic distribution. For example, the 95% credible interval can be calculated by taking the 2.5^{th} and 97.5^{th} percentiles of the sample while the 99% credible interval can be calculated by taking the 0.5^{th} and 99.5^{th} percentiles. There are also other statistics that can be calculated, for example the median (50^{th} percentile and the inter-quartile range (25^{th} and 75^{th} percentiles).

In other words, the entire MCMC sample is used to calculate statistics about the target distribution. Once the MCMC algorithm has reached 'equilibrium', meaning that it approximates the target distribution fairly closely, then a sample of values for each parameter from this algorithm yields an accurate representation of the target distribution.

MCMC Output

Let us discuss the statistics presented in the MCMC output.

Summary Statistics

First, there are the summary statistics represented by the log likelihood, the AIC, the BIC/SC, the Deviance, and Pearson Chi-square indices. Second, there are statistics for model error represented by the MAD and the MSPE; as with the MLE output, quartiles for these error statistics are presented. Third, there are the coefficients, the standard error, and a t-test based on the assumption that the distribution was normal and that the "t" is applicable (an assumption that is not necessarily correct). We present this because it allows a quick evaluation of the 'significance' of an independent variable.

Convergence Statistics

Fourth, in addition to these individual statistics, there are convergence statistics which indicate whether the algorithm converged (Spiegelhalter, Best, Carlin, & Van der Linde, 2002). It is essential for the user to evaluate whether the sequence converged; if it did not, then the coefficients and standard errors are not valid. These statistics are calculated by comparing chains of estimated values for parameters, either with themselves or with the complete series. When there is convergence, the estimates will be similar.

The first convergence statistic is the **Monte Carlo simulation error** (called *MC Error;* Ntzoufras, 2009, 30-40). Two estimates of the value of each parameter are calculated and their discrepancy is evaluated. The first estimate is the mean value of the parameter over all *M-L* iterations (total number of iterations minus the number of burn-in samples discarded). The second estimate is the mean value of the parameter after breaking the *M-L* iterations into *m* chains where *m* is the integer value of the square root of *M-L*.

Let:

$$Mean\theta_K = (\sum_i \theta_i) / K \tag{17.22}$$

$$Mean\theta_M = (\sum_m \theta_m) / m \tag{17.23}$$

and

$$MCError = \frac{\sqrt{Mean\theta_K - Mean\theta_M}}{m(m-1)} \tag{17.24}$$

Generally, the MC error is related to the standard deviation of the parameters. If the ratio is less than 0.05, then the sequence is considered to have converged after the 'burn in' samples have been discarded (Ntzourfras, 2009). As can be seen, the ratios are very low in Table 17.1.

The second convergence statistic is the **Gelman-Rubin convergence diagnostic** (*G-R*, sometimes called the *scale reduction factor*; Gelman, Carlin, Stern & Rubin, 2004; Gelman, 1996; Gelman & Rubin, 1992). Gelman and Rubin called it the *R* statistic, but we will call it the *G-R* statistic. The concept is, again, to break the larger chain into multiple smaller chains and calculate whether the variation within the chains for a parameter approximately equals the total variation across the chains (Carlin & Louis, 2008; Lynch, 2007). That is, when *m* chains are run, each of length *n*, the mean of a parameter θ_m can be calculated for each chain as well as the overall mean of all chains θ_G, the within-chain variance, and the between-chain variance. The G-R statistic is the square root of the total variance divided by the within-chain variance:

$$G-R=\sqrt{\left(\frac{m+1}{m}\right)*\left(\frac{n-1}{n}+\frac{B}{W}\right)-\left(\frac{n-1}{mn}\right)} \tag{17.25}$$

where B is the variance between the means from the m parallel chains, W is the average of the m within-chain variances, and n is the length of each chain (Lynch, 2007; Carlin & Louis, 2000).

The G-R statistic should generally be low for each parameter. If the G-R statistic is under approximately 1.2, then the posterior distribution is commonly considered to have converged (Mitra and Washington, 2007).

Example of Estimating Houston Burglaries with the MCMC Poisson-Gamma

Before we discuss some of the subtleties of the method, let us illustrate this with the Houston burglary example that we have been using in the previous two chapters (Table 17.1). The data came from the Houston Police Department. There were 26,480 burglaries that occurred in 2006 which were allocated to 1,179 Traffic Analysis Zones (TAZ) within the City of Houston. The independent variables were the number of households in 2006 (estimated by the Houston-Galveston Area Council, the metropolitan planning organization) and the median household income for 2000 (from the 2000 U.S. Census).

The MCMC algorithm for the Poisson-Gamma (negative binomial) model was run on the Houston burglary dataset. The total number of iterations was 25,000 with the initial 5,000 being discarded (the 'burn in' period). Thus, the results are based on the final 20,000 samples.

Comparison of MCMC Poisson-Gamma with MLE Poisson-Gamma

By comparing the results of the MCMC Poisson-Gamma estimate on the Houston burglary data set with that from the MLE Poisson-Gamma model from the previous chapter (Table 15.3), we can show that the MCMC method produces very similar results to the MLE when the estimated functions are identical. This is expected since the hyper-priors MCMC are very vague or have large variance. In Table 17.1, the two convergence statistics are very low for all three parameters as well as for the error term. In other words, the algorithm appears to have converged properly and the results are based on a good equilibrium chain.

Second, looking at the likelihood statistics, we see that they are very similar to that of the MLE negative binomial model. The log likelihood value is identical for the two models -4430.8. The AIC and BIC/SC statistics are also almost identical (8869.6 and 8869.8 compared to 8869.6 and 8889.9). The deviance statistic is very similar for the two models - 1,387.5 compared to 1,390.1, as is the Pearson Chi-square statistic – 1,106.4 compared to 1,112.7.

Table 17.1:

Predicting Burglaries in the City of Houston: 2006

MCMC Poisson-Gamma Model

(N= 1,179 Traffic Analysis Zones)

DepVar:	**2006 BURGLARIES**		
N:	1,179		
Df:	1,175		
Type of regression model:	Poisson with Gamma dispersion		
Method of estimation:	MCMC		
Number of iterations:	25,000	Burn in: 5,000	
Likelihood statistics			
Log Likelihood:	-4,430.8		
AIC:	8,869.6		
BIC/SC:	8,889.9		
Deviance:	1,387.5	p≤ 0.0001	
Pearson Chi-Square:	1,106.4	p≤ 0.0001	
Model error estimates			
Mean absolute deviation:	40.0		
1st (highest) quartile:	124.9		
2nd quartile:	19.5		
3rd quartile:	6.2		
4th (lowest) quartile:	9.0		
Mean squared predicted error:	63,007.2		
1st (highest) quartile:	245,857.0		
2nd quartile:	6,527.5		
3rd quartile:	119.4		
4th (lowest) quartile:	156.2		
Dispersion tests			
Adjusted deviance:	1.2	p≤ 0.0001	
Adjusted Pearson Chi-Square:	0.9	p≤ 0.0001	
Dispersion multiplier:	1.5	p≤ 0.0001	Inverse dispersion multiplier: 0.7

Predictor	Mean	Std	t-value[p]	MC error	MC error/ std	G-R stat
INTERCEPT	2.3204	0.086	26.88***	0.002	0.019	1.002
HOUSEHOLDS	0.0012	0.00007	17.57***	0.0000009	0.013	1.001
MEDIAN HOUSEHOLD INCOME	-0.00001	0.00002	-4.92***	0.00000003	0.019	1.002

*** p≤.001

Third, in terms of the model error statistics, the MAD and MSPE are also very similar (40.0 and 63,007.2 compared to 39.6 and 62,031.2; while the difference in the MSPE is 976.0, it is less than 2% of the MSPE for the MLE.[5] Fourth, the over-dispersion tests reveal identical values - adjusted deviance (1.2 for both), adjusted Pearson Chi-square (0.9 for both), and the Dispersion multiplier (both 1.5).

Fifth, the coefficients are identical with the MLE up through third decimal place. For example, for the intercept the MCMC gives 2.3204 compared to 2.3210; that of the two independent variables are identical within the precision of the table. This is not surprising since when we use non-informative priors, it is expected that the posterior estimates will be very close to those estimated by the MLE.

Sixth, the standard errors are identical for all three coefficients. In the MCMC, the standard errors are calculated by taking the standard deviation of the sample. In general, the MCMC will produce similar or slightly larger standard errors. The theoretical distribution assumes that the errors are normally distributed. This may or may not be true depending on the data set. Thus, the MCMC standard errors are non-parametric.

Seventh, a t-test (or more precisely a 'pseudo' t-test) is calculated by dividing the coefficient by the standard error. If the standard errors are normally distributed (or approximately normally distributed), then such a test is valid. On the other hand, if the standard errors are skewed, then the approximate t-test is not accurate. *CrimeStat* outputs additional statistics that list the percentiles of the distributions. These are more accurate indicators of the true confidence intervals and are known as *credible intervals*. We will illustrate these shortly with another example. In short, the pseudo t-test is an approximation to true statistical significance and should be seen as a guide, rather than a definitive answer.

Example of Estimating Houston Burglaries with the MCMC Normal

As an example of the MCMC Normal model, we ran the model on the Houston burglary data set. Keep in mind that this is a skewed data set and that the Normal model is not really appropriate. Table 17.2 presents the results. As a comparison, we repeat the MLE Normal/OLS model from Chapter 15 (Table 15.1).

[5] Frequently, the model error is greater for an MCMC model than an MLE model. Whether this represents true model error or overfitting by the MLE algorithm is not fully understood at this point.

Table 17.2:

Predicting Burglaries in the City of Houston: 2006

MCMC Normal Model

(N= 1,179 Traffic Analysis Zones)

DepVar:	**2006 BURGLARIES**
N:	1,179
Df:	1,175
Type of regression model:	Poisson with Lognormal dispersion
Method of estimation:	MCMC
Number of iterations:	25,000 Burn in: 5,000

Likelihood statistics

Log Likelihood:	-5342.6
AIC:	10,693.2
BIC/SC:	10,713.6
R^2:	0.48

Model error estimates

Mean absolute deviation:	13.5
1st (highest) quartile:	26.5
2nd quartile:	10.6
3rd quartile:	8.2
4th (lowest) quartile:	8.6
Mean squared predicted error:	505.1
1st (highest) quartile:	1,501.7
2nd quartile:	272.3
3rd quartile:	130.5
4th (lowest) quartile:	120.0

Predictor	Mean	Std	t-value[p]	MC error	MC error/ std	G-R stat
INTERCEPT	12.7804	1.235	10.35***	0.020	0.016	1.001
HOUSEHOLDS	0.0255	0.001	32.62***	0.000009	0.011	1.0005
MEDIAN HOUSEHOLD INCOME	-0.0002	0.00003	-7.00***	0.0000004	0.015	1.0004

** p≤.01

*** p≤.001

Table 15.3 (REPEAT):

Predicting Burglaries in the City of Houston: 2006

Ordinary Least Squares: Reduced Model

(N= 1,179 Traffic Analysis Zones)

DepVar:	**2006 BURGLARIES**
N:	1,179
Df:	1,175
Type of regression model:	Ordinary Least Squares
F-test of model:	536.0 $p \leq .0001$
R^2:	0.48
Adjusted R^2:	0.48
Mean absolute deviation:	13.5
1st (highest) quartile:	26.5
2nd quartile:	10.6
3rd quartile:	8.3
4th (lowest) quartile:	8.8
Mean squared predictive error:	505.1
1st (highest) quartile:	1498.8
2nd quartile:	269.5
3rd quartile:	135.1
4th (lowest) quartile:	120.2

Predictor	DF	Coefficient	Stand Error	Tolerance	VIF	t-value	p
INTERCEPT	1	12.8099	1.240	-	-	10.33	0.001
HOUSEHOLDS	1	0.0255	0.0008	0.994	1.006	33.44	0.001
MEDIAN HOUSEHOLD INCOME	1	-0.0002	0.00003	0.994	1.006	-7.03	0.001

Comparison of MCMC Normal with MLE Normal

The MCMC Normal and the MLE Normal produce similar estimates. The log-likelihood statistics are unique to the MCMC model, but the R-squares are identical and the Mean Absolute Deviation and the Mean Squared Predictive Error values are very close to each other in both models. This means that the MCMC Normal converged on the function in a similar manner to the MLE normal. Also, the coefficients estimates for the MCMC Normal are quite close to those produced by the MLE.

Thus, it appears that the MCMC Normal can approximate the MLE Normal under some circumstances. Further, if the dependent variable is truly normally distributed, then the MCMC Normal will produce results that are almost identical.

Note that this is not always the case. When the dependent variable is highly skewed, we have frequently found that the MCMC Normal model will not produce identical results to that of the MLE even if a large number of iterations are run. We are not completely sure why this occurs, but the more skewed the distribution or the more complex the model, the less likely the MCMC Normal will yield the same solution as the MLE Normal. In short, the MCMC Normal is very sensitive to skewness in a data set and is most appropriate when the dependent variable is normally distributed.

Therefore, the user has to be careful in interpreting the MCMC Normal. Before running a spatial regression model using the MCMC Normal (see Chapter 19), users should confirm that the MCMC Normal can replicate an MLE Normal/OLS model. If it does not, they should run an alternative model such as the Poisson-Gamma.

Why Run an MCMC when MLE is So Easy to Estimate?

What we have seen is that the MCMC Poisson-Gamma (negative binomial) model and the MCMC Normal model produced results that were very similar to that of the MLE Poisson-Gamma and MLE Normal models respectively. In other words, simulating the distribution of the Poisson-Gamma function or the MCMC Normal function with the MCMC method has produced results that are completely consistent with a maximum likelihood estimate.

A key question, then, is why bother? The maximum likelihood algorithm works efficiently with functions from the single-parameter exponential family while the MCMC method takes time to calculate. Further, the larger the database, the greater the differential there will be in calculating time. For example in Chapter 16, Table 16.4 presented an MLE negative binomial model of the number of 2006 crimes committed by individual offenders in Manchester as a function of three independent variables – distance from the city center, prior conviction, and age of the offenders. With an Intel Duo core 2.44 GHz processor, the run took 6 seconds for the MLE while it took 86 minutes for the MCMC equivalent! Clearly, the MCMC algorithm is more calculation intensive than the MLE algorithm. If they produce essentially the same results, there is no obvious reason for choosing the slower method over the faster one.

The reason for preferring the MCMC method, however, has to do with the complexity of other models. The MLE approach works particularly well when all the individual functions in a mixed function model belong to the single-parameter exponential family of functions. For more complex functions, however, the method does not work very well. The likelihood functions

need to be worked out explicitly for the MLE approach to work. For example, if other functions for the dispersion were used, such as a Weibul or Gumbel or Cauchy or uniform distribution, the MLE approach would not easily be able to solve such equations since the mathematics are complex and there may not be a single optimal solution.

Further, if we start combining functions in different mixtures, such as Poisson mean, Gamma dispersion but Weibul shape function, the MLE is not easily adapted. An example is spatial regression where assumptions about the mean, the variance and spatial autocorrelation need to be specified exactly. This is a complex model and there is not a simple second derivative that can be calculated for such a function. The existing spatial models have tried to work around this by using a linear form but allowing a spatial autocorrelation term either as a predictive variable (the *spatial lag* model) or as part of the error term (the *spatial error* model; DeSmith, Goodchild, & Longley, 2007; Anselin, 2002). But, they all assume a normally-distributed dependent variable which is rarely found with crime data.

In short, the MCMC method has an advantage over MLE for complex functions. For simpler functions in which the functions are all part of the same exponential family and for which the mathematics has been worked out, MLE is clearly superior in terms of efficiency.

However, the more irregular and complex the function to be estimated, the more the simulation approach has an advantage over the MLE. For example, to estimate a Poisson-Gamma (negative binomial) function takes longer with the MCMC method than with the MLE method and there is no advantage for the MCMC over the MLE. On the other hand, the Poisson-Lognormal model (see below) or the Poisson-Gamma-CAR model (to be discussed in Chapter 19) cannot be estimated by MLE. An even more complex model is a spatial risk model where the 'at risk' variable is constrained to have a coefficient of 1.0 with spatial autocorrelation also being tested; this cannot be estimated with MLE.

Example of Estimating Houston Burglaries with the MCMC Poisson-Lognormal

For an example of a complex mixed function model, let us run the Houston burglary dataset with the Poisson-Lognormal. As mentioned above, the Poisson-Lognormal is an alternative model to the Poisson-Gamma. It is particularly useful when the sample mean is low and there are lots of zeros. The Poisson-lognormal is usually more stable than the Poisson-gamma for these kinds of data.

Table 17.3 shows the results. Compared to Table 17.1 for the Poisson-Gamma model, the log likelihood of the Poisson-Lognormal is more negative (weaker) than for the Poisson-Gamma while the AIC and BIC statistics are higher. In other words, the MCMC Poisson-Gamma fit the data slightly better than the MCMC Poisson-Lognormal though the differences are small.

Table 17.3:
Predicting Burglaries in the City of Houston: 2006
MCMC Poisson-Lognormal Model
(N= 1,179 Traffic Analysis Zones)

DepVar:	**2006 BURGLARIES**		
N:	1,179		
Df:	1,175		
Type of regression model:	Poisson with Lognormal dispersion		
Method of estimation:	MCMC		
Number of iterations:	25,000	Burn in: 5,000	
Likelihood statistics			
Log Likelihood:	-4,650.2		
AIC:	9,308.4		
BIC/SC:	9,328.7		
Deviance:	1,551.9	p≤ 0.0001	
Pearson Chi-Square:	4,685.6	p≤ 0.0001	
Model error estimates			
Mean absolute deviation:	37.5		
1st (highest) quartile:	122.5		
2nd quartile:	20.6		
3rd quartile:	3.3		
4th (lowest) quartile:	4.0		
Mean squared predicted error:	62,216.2		
1st (highest) quartile:	244,906.0		
2nd quartile:	4,489.4		
3rd quartile:	40.4		
4th (lowest) quartile:	63.4		
Dispersion tests			
Adjusted deviance:	1.3	p≤ 0.0001	
Adjusted Pearson Chi-Square:	4.0	p≤ 0.0001	
Dispersion multiplier:	2.0	p≤ 0.0001	Inverse dispersion multiplier: 0.5

Predictor	Mean	Std	t-value[p]	MC error	MC error/ std	G-R stat
INTERCEPT	1.3612	0.092	14.82***	0.002	0.022	1.002
HOUSEHOLDS	0.0013	0.00005	25.30***	0.0000007	0.014	1.000
MEDIAN HOUSEHOLD INCOME	-0.000005	0.00002	-2.92**	0.00000003	0.018	1.001

** p≤.01
*** p≤.001

However, the log-likelihood is the overall probability of the model, not particularly the best fit for the residual errors. Comparing Tables 17.1 and 17.2, we find that the MAD and the MSPE are smaller for the Poisson-Lognormal than for the Poisson-Gamma. The coefficients are very similar. The intercept is smaller in the Poisson-Lognormal while the coefficients for households and for median household income are virtually the same. In short, the Poisson-Lognormal will predict a slightly smaller expected count than the Poisson-Gamma due to the smaller intercept term, but the two sets of estimates are quite similar. In other words, with these data, the Poisson-Lognormal model produces a slightly lower probability but a better fit than the Poisson-Gamma. In this case, we would accept the Poisson-Gamma because the differences are not great. But, there are data sets where the Poisson-Lognormal is definitely better than the Poisson-Gamma (Lord & Miranda-Moreno, 2008).

Risk Analysis

One example of where the MCMC method is better than the MLE method is in ***risk analysis***. Sometimes a dependent variable is analyzed with respect to an exposure variable. For example, instead of modeling just burglaries, a user might want to model burglaries relative to the number of households. In our example in this chapter (Houston burglaries), we have included the number of households as a predictor variable but it is unstandardized, meaning that the estimated effect of households on burglaries cannot be easily compared to other studies that model burglaries relative to households.

For this, a different type of analysis has to be used. Frequently called a *risk analysis*, the dependent variable is related to an exposure measure. The formulation we use is that of Besag, Green, Higdon and Mengersen (1995). Like all the non-linear models that we have examined, the dependent variable, y_i, is modeled as a Poisson function of the mean, λ_i:

$$y_i \mid \lambda_i \sim Poisson(\lambda_i) \tag{17.26}$$

In turn, the mean of the Poisson is modeled as:

$$\mu_i = \nu_i \lambda_i \tag{17.27}$$

where ν_i is an *exposure* measure and λ_i is the *rate* (or risk). The exposure variable is the baseline variable to which the number of events is related. For example, in motor vehicle crash analysis, the exposure variable is usually Vehicle Miles Traveled or Vehicle Kilometers Traveled (multiplied by a power of 10 to eliminate very small numbers, such as per 1000 or per 100 million). In epidemiology, the exposure variable is the population at risk, either the general population or the population of a specific age group perhaps broken down further into gender.

For crime analysis, the exposure variable might be the number of households for residential crimes or the number of businesses for commercial crimes. Choosing an appropriate exposure variable is not a trivial matter. In some cases, there are national standards for exposure (e.g., number of infants for analyzing child mortality; Vehicle Miles Traveled for analyzing motor vehicle crash rates). But, often there are not accepted exposure standards.

In some cases, the exposure variable may be non-linear in order to capture important missing variables. For instance, in highway safety, traffic flow (i.e., the number of vehicle traveling passing a given point in a unit of time) has been found to vary in a non-linear fashion. This characteristic can be explained by the fact vehicle occupancy (i.e., the number of vehicles per unit of length) and vehicle speed, which are directly linked to traffic flow, are variables that are not availabe or routinely collected. Hence, traffic flow tends to show non-linear relationships (see Lord, Manar, &Vizioli, 2005, for more details).

The rate is further structured in the Poisson-Gamma or Poisson-Lognormal models:

$$\mu_i = \nu_i \lambda_i = \nu_i \cdot \exp(\mathrm{x}_i^T \beta + \varepsilon_i) \tag{17.28}$$

where the symbols have the same definitions as in equation 17.18 with the error term, ε_i, being modeled either as a Gamma function (equation 17.9) or as a Lognormal function (equation 6.10).

With the exposure term, the full model is estimated as the same fashion,

$$y_i \sim Poisson(\nu_i \lambda_i) \tag{17.29}$$

$$\lambda_i \sim Gamma\ (\psi, \psi e^{-\mathbf{x}_i^T \boldsymbol{\beta} - \phi_i}) \tag{17.30}$$

$$\lambda_i \sim Lognormal\,[0, Gamma(a_\varepsilon, b_\varepsilon)] \tag{17.31}$$

Note that no coefficient for the exposure variable, ν_i, is estimated (i.e., it is 1.0). It is sometimes called an *offset* variable (or exposure offset). The model is then estimated either with an MLE or MCMC estimation algorithm.

An example is that of Levine (2011) who analyzed the number of motor vehicle crashes in which a male was the primary driver relative to the number of crashes in which a female was the primary driver for each major road segment in the Houston metropolitan area. In the risk model set up, the dependent variable was the number of crashes involving a male primary driver for each road segment while the exposure (offset) variable was the number of crashes involving a female primary driver. The independent variables in the equation were volume-to-capacity ratio

(an indicator of congestion on the road), the distance to downtown Houston, and several road categories (freeway, principal arterial, etc).

To illustrate this type of model, we ran a MCMC Poisson-Gamma model using the number of households as the exposure variable. There was, therefore, only one independent variable, median household income. Table 17.4 shows the results

.

Compared to the non-exposure burglary model (Table 17.1), the model does not fit the data as well. The log likelihood is lower while the AIC and BIC are higher. Further, the MAD and MSPE statistics for model error are much worse.

Further, the dispersion statistics indicate that there is more over-dispersion with the risk model than the simple Poisson-Gamma model. In other words, the exposure variable has not eliminated the dispersion as much as the random effects (non-exposure) model.

Looking at the coefficients, the offset variable (number of households) has a coefficient of 1.0 because it is defined as such. The coefficient for median household income is still negative, but is stronger than in Table 17.1. The effect of standardizing households as the baseline exposure variable has increased the importance of household income in predicting the number of burglaries, controlling for the number of households.

The second part of the table show percentiles for the coefficients, and is preferable for statistical testing than the asymptotic t-test. The reason is that the distribution of parameter values may not be normally distributed or may be very skewed, whereas the t- and other parametric significance tests assume that there is perfect normality. *CrimeStat* outputs a number of percentiles for distribution. We have shown only four of them, the 0.5^{th}, 2.5^{th}, 97.5^{th}, and 99.5^{th} percentiles. The 2.5^{th} and 97.5^{th} represent 95% credible intervals while the 0.5^{th} and 99.5^{th} represent 99% credible intervals.

The way to interpret the percentiles is to check whether a coefficient of 0 (the 'null hypothesis') or any other particular value is outside the 95% or 99% credible intervals. For example, with the intercept term, the 95% credible interval is defined by -2.4365 to -2.1292. For both the intercept and median household income, a coefficient of 0 is outside both the 95% and 99% credible intervals. In other words, both the intercept and median household income are *significantly* different than 0, though the use of the term 'significant' is different than with the usual asymptotic normality assumptions since it is based on the distribution of the parameter values from the MCMC simulation.

Table 17.4:

Predicting Burglaries in the City of Houston: 2006
MCMC Poisson-Gamma Model with Exposure Variable

(N= 1,179 Traffic Analysis Zones)

DepVar:	**2006 BURGLARIES**		
N:	1,179		
Df:	1,176		
Type of regression model:	Poisson with Gamma dispersion		
Method of estimation:	MCMC		
Number of iterations:	25,000	Burn in:	5,000
Distance decay function:	Poisson-Gamma		
Likelihood statistics			
Log Likelihood:	-6,634.4		
AIC:	13,274.8		
BIC/SC:	13,290.0		
Deviance:	5,373.5	p≤ 0.0001	
Pearson Chi-square:	514.4	p≤ 0.0001	
Model error estimates			
Mean absolute deviation:	14,147.3		
Mean squared predicted error:	553,058,555.7		
Dispersion tests			
Adjusted deviance:	4.6	p≤ 0.0001	
Adjusted Pearson Chi-Square:	0.44	p≤ 0.0001	
Dispersion multiplier:	2.3	p≤ 0.0001	Inverse dispersion multiplier: 0.44

Predictor	Mean	Std	t-value[p]	MC error	MC error/ std	G-R stat
Exposure/offset variable:						
HOUSEHOLDS	1.0	-	-	-	-	-
Linear predictors:						
INTERCEPT	3.4624	0.0917	37.75***	0.002	0.020	1.002
MEDIAN HOUSEHOLD INCOME	-0.00009	0.000002	-4.57***	0.00000004	0.020	1.002

*** p≤.001

Percentiles	**0.5th**	**2.5th**	**97.5th**	**99.5th**
INTERCEPT	3.2242	3.2833	3.6389	3.6942
MEDIAN HOUSEHOLD INCOME	-0.00002	-0.00001	-0.00005	-0.00004

In other words, percentiles can be used as a non-parametric alternative to the t- or Z-test. Without making assumptions about the theoretical distribution of the parameter value (which the t- and Z-test do – they are assumed to be normal or near normal for "t"), significance can be assessed empirically.

In summary, in risk analysis, an exposure variable is defined and held constant in the model. Thus, the model is really a risk or rate model that relates the dependent variable to the baseline exposure. The independent variables are now predicting the rate, rather than the count by itself.

Issues in MCMC Modeling

We now turn to four issues in MCMC modeling. The first is the starting values of the MCMC algorithm. The second is the issue of *convergence* to an equilibrium state. The third issue is the statistical testing of parameters and the general problem of overfitting the data while the fourth issue is the performance of the MCMC algorithm with large datasets.

Starting Values of Each Parameter

The MCMC algorithm requires that initial values be provided for each parameter to be estimated. These are called *prior probabilities* even though they do not have to be standardized in terms of a number from 0 to 1. The *CrimeStat* routine allows the defining of initial starting values for each of the parameters and for the overall Φ coefficient in the various spatial regression models (see chapter 18). If the user does not define the initial starting values, then default values are used. Of necessity, these are vague. For the individual coefficients (and the intercept), the initial default values are 0. For the Φ coefficient, the initial default values are defined in terms of its hyperparameters, (Rho = 0.5; Tauphi = 1; alpha = -1). Essentially, these assume very little about the distribution and are, essentially, *non-informative priors*.

The problem with using vague starting values, however, is that the algorithm could get stuck on a local 'peak' and not actually find the highest probability. Even though the MCMC algorithm is not a greedy algorithm, it still explores a limited space. It will generally find the highest peak within its search radius. But, there is no guarantee that it will explore regions far away from its initial location. If the user has some basis for estimating a prior value, then this will usually be of benefit to the algorithm in that it can minimize the likelihood of finding local 'peaks' rather than the highest 'peak'.

Where do the prior values come from? They can come from other research, of course (see Miranda-Moreno et al., 2009). Alternatively, they can come from other methods that have attempted to analyze the same phenomena. Lynch (2007), for example proposes running an

MLE Poisson-Gamma (negative binomial) model and then using those estimates as the prior values for the MCMC Poisson-Gamma. Even if the user is going to run a spatial model (e.g., MCMC Poisson-Gamma-CAR/SAR), the estimates from an MLE model are probably good starting values.

Example of Defining Prior Values for Parameters

We can illustrate this with an example. A model was run on 325 Baltimore County traffic analysis zones (TAZ) predicting the number of crimes that occurred in each zone in 1996. There were four independent variables:

1. Population (1996)
2. Relative median household income index
3. Retail employment (1996)
4. Distance from the center of the metropolitan area (in the City of Baltimore)

The dataset was divided into two groups, group A with 163 TAZs and group B with 162 TAZs. The model was run as a spatial regression (Poisson-Gamma-CAR – see chapter 19) for each of the groups. Table 17.5 shows the results of the coefficients with the standard errors in brackets.

Column 1 shows the results of running the model on group A. Column 2 shows the results of running the model on group B while column 3 shows the results of running the model on group B but using the coefficient estimates from group A as prior values. With the exception of the relative income variable, the coefficients of column C generally fall between the results for group A and group B by themselves. Even the one exception – relative income, is very close to the 'non-informative' estimate for group B.

In other words, using prior values that are based on realistic estimates (in this case, the estimates from group A) have produced results that incorporate that information in estimating the information just from the data. Essentially, this is what equation 17.7, updating the probability estimate of the data given the likelihood based on the prior probability. In short, using prior estimates combines new information with the existing information to update the estimates. Aside from protecting against finding local optima in the MCMC algorithm, the prior information generally improves the knowledge base of the model.

Convergence

In theory, the MCMC algorithm should converge into a stable equilibrium state whereby the true probability distribution is being sampled. With the hill climbing analogy, the climber has found the highest mountain to be climbed and is simply sampling different locations on the

Table 17.5:
The Effects of Starting Values on Coefficient Estimates for Baltimore County Crimes:

Dependent Variable = Number of Crimes in 1996

	(1) **Group A** (N=163 TAZs)	(2) **Group B** (N=162 TAZs)	(3) **Group B** (N=162 TAZs)
Starting values:	**Default/ 'non-informative'**	**Default/ 'non-informative'**	**Group A estimates**
Independent variables			
INTERCEPT	4.3621 *(0.2674)*	4.7727 *(0.2434)*	4.7352 *(0.2489)*
POPULATION	0.00035 *(0.00004)*	0.00034 *(0.00004)*	0.00035 *(0.00004)*
RELATIVE INCOME	-0.0234 *(0.0047)*	-0.0226 *(0.0041)*	-0.0224 *(0.0043)*
RETAIL EMPLOYMENT	0.0021 *(0.0002)*	0.0017 *(0.0002)*	0.0017 *(0.0001)*
DISTANCE FROM CENTER	-0.0590 *(0.0160)*	-0.0898 *(0.0141)*	-0.0881 *(0.0142)*
AVERAGE PHI COEFFICIENT	0.0104 *(0.1117)*	-0.0020 *(0.0676)*	0.0077 *(0.0683)*

mountain to see which one will provide the best path up the mountain. The first iterations in a sequence are thrown away (the 'burn in') because the sequence is assumed to be looking for the true probability distribution. Put another way, the starting values of the MCMC sequence have a big effect on the early draws and it takes a while for the algorithm to move away from those initial values (remember, it is a random walk and the early steps are near the initial starting location).

A key question is how many samples to draw and a second, ancillary question is how many should be discarded as the 'burn in'? Unfortunately, there is not a simple answer to these questions. For some distributions, the algorithm quickly converges on the correct solution and a limited number of draws are needed to accurately estimate the parameters. In the Houston burglary example, the algorithm easily converged with 20,000 iterations after the first 5,000 had been discarded. We have been able to estimate the model accurately after only 4000 iterations with 1000 burn in samples being discarded. The dependent variable is well behaved because it is at the zonal level and the model is simple.

On the other hand, some models do not easily converge to an equilibrium stage. Models with individual level data are typically more volatile. Also, models with many independent variables are complex and do not easily converge. To illustrate, we estimate a model of the residence locations of drunk drivers (DWI) who were involved in crashes in Baltimore County between 1999 and 2001 (Levine & Canter, 2011). The drivers lived in 532 traffic analysis zones (TAZ) in both Baltimore County and the City of Baltimore. The dependent variable was the annual number of drivers involved in DWI crashes who lived in each TAZ and there were six independent variables:

1. Total population of the TAZ
2. The percent of the population who were non-Hispanic White
3. Whether the TAZ was in the designated rural part of Baltimore County (dummy variable: 1 – Yes; 0 – No)
4. The number of liquor stores in the TAZ
5. The number of bars in the TAZ
6. The area of the TAZ (a control variable).

Table 17.6 presents the results. The overall model fit was statistically significant and there was very little over-dispersion (as seen by the dispersion parameter). A "pure" Poisson model could have been used in this case. Of the parameters, the intercept and four of the six independent variables were statistically significant, based on the t-test. The results were consistent with expectations, namely zones (TAZs) with greater population, a greater percentage of non-Hispanic White persons, that were in the rural part of the county, that had more liquor stores, and that had more bars had a higher number of drunk drivers residing in those zones.

However, the convergence statistics were questionable. Two of the parameters had G-R values higher than the acceptable 1.2 level and five of the MC error/standard error values were higher than the acceptable 0.05 level. In other words, it appears that the model did not properly converge. Consequently, we ran the model again with 100,000 iterations and discarded the initial 10,000 'burn in' samples. Table 17.7 shows the results. Comparing tables 17.6 with 17.5, we can see that the overall likelihood statistics was approximately the same as were the dispersion

Table 17.6:

Number of Drivers Involved in DWI Crashes Living in Baltimore County: 1999-2001 MCMC Poisson-Gamma Model with 20,000 Iterations

(N= 532 Traffic Analysis Zones)

DepVar:	**Annual Number of Drivers in DWI Crashes Living in TAZ**		
N:	532		
Type of regression model:	Poisson with Gamma dispersion		
Method of estimation:	MCMC		
Total number of iterations:	25,000	Burn in:	5,000
Likelihood statistics			
Log Likelihood:	-278.7		
AIC:	573.4		
BIC/SC:	607.6		
Deviance:	316.6	p: 0.0001	
Pearson Chi-square:	475.6	p: 0.0001	
Model error estimates			
Mean absolute deviation:	0.32		
Mean squared predicted error:	0.25		
Dispersion tests			
Adjusted deviance:	0.60	p: 0.0001	
Adjusted Pearson Chi-Square:	0.91	p: 0.0001	
Dispersion multiplier:	0.15	p: 0.0001	Inverse dispersion multiplier: 6.77

Predictor	**Mean**	**Std**	**t-value[p]**	**MC error**	**MC error/ std**	**G-R stat**
INTERCEPT	-4.5954	0.476	-9.65***	0.0386	0.081	1.349
POPULATION	0.0004	0.00005	8.70***	0.000003	0.068	1.165
PERCENT WHITE	0.0237	0.005	4.81***	0.0004	0.079	1.283
RURAL	0.6721	0.329	2.04*	0.0184	0.056	1.042
LIQUOR STORES	0.2423	0.125	1.94n.s.	0.0059	0.047	1.028
BARS	0.1889	0.058	3.28**	0.0024	0.041	1.008
AREA	-0.0548	0.033	-1.68n.s.	0.0018	0.055	1.041

n.s. Not significant
** $p \leq .01$
*** $p \leq .001$

Table 17.7:

Number of Drivers Involved in DWI Crashes Living in Baltimore County: 1999-2001 MCMC Poisson-Gamma Model with 90,000 Iterations

(N= 532 Traffic Analysis Zones)

DepVar:	**Annual Number of Drivers in DWI Crashes Living in TAZ**		
N:	532		
Type of regression model:	Poisson with Gamma dispersion		
Method of estimation:	MCMC		
Total number of iterations:	100,000	Burn in:	10,000
Likelihood statistics			
Log Likelihood:	-278.6		
AIC:	573.2		
BIC/SC:	607.4		
Deviance:	317.9	p: 0.0001	
Pearson Chi-square:	479.5	p: 0.0001	
Model error estimates			
Mean absolute deviation:	0.32		
Mean squared predicted error:	0.25		
Dispersion tests			
Adjusted deviance:	0.61	p: n.s.	
Adjusted Pearson Chi-Square:	0.92	p: n.s.	
Dispersion multiplier:	0.14	p: n.s.	Inverse dispersion multiplier: 7.36

Predictor	**Mean**	**Std**	**t-value[p]**	**MC error**	**MC error/ std**	**G-R stat**
INTERCEPT	-4.6608	0.425	-10.96***	0.0222	0.052	1.085
POPULATION	0.0004	0.00005	8.78***	0.000002	0.041	1.041
PERCENT WHITE	0.0243	0.004	5.77***	0.0002	0.050	1.081
RURAL	0.6378	0.324	1.97*	0.0092	0.028	1.005
LIQUOR STORES	0.2431	0.123	1.98*	0.0033	0.027	1.002
BARS	0.1859	0.055	3.36***	0.0011	0.020	1.004
AREA	-0.0515	0.032	-1.63n.s.	0.0009	0.029	1.008

n.s. Not significant
* $p \leq .05$
*** $p \leq .001$

statistics. However, the convergence statistics indicate that the model with 90,000 iterations had better convergence than that with only 20,000. Of the parameters, none had a G-R value greater than 1.2 while only one had an MC Error/Standard error value greater than 0.05, and that only slightly.

This had an effect on both the coefficients and the significance levels. The coefficients were in the same direction for both models but were slightly different. Further, the standard deviations were generally smaller with more iterations and only one of the independent variables was not significant (area, which was a control variable).

In other words, increasing the number of burn-in samples as well as the number of iterations run improved the model. It apparently converged for the second run whereas it had not for the first run. The algorithm did this for two reasons. First, by taking a larger number of iterations, the model was more precise. Second, by dropping more initial iterations during the 'burn in' phase (10,000 compared to 5,000), the series apparently reached an equilibrium state before the sample iterations were calculated. The smaller standard errors suggest that there still was a trend when only 5,000 were dropped but had ceased by the time the first 10,000 iterations had been reached.

The point to remember is that one wants a stable series before drawing a sample. If in doubt, run more during the 'burn in' phase. This increases the calculating time, of course, but the results will be more reliable. Once the MCMC algorithm has reached 'equilibrium', it won't take that many additional samples to produce good estimates. We have estimated that 5,000-10,000 additional samples beyond the 'burn-in' sample will produce good results. One can implement this in stages. For example, run the model with the default 25,000 iterations with 5,000 for the 'burn in' (for a total of 20,000 sample iterations from which to base the conclusions). If the convergence statistics suggest that the series has not yet stabilized, run the model again with more 'burn in' samples and, perhaps, more sample iterations.

Monitoring Convergence

A second concern is how to monitor convergence. There appear to be two different approaches. One is a graphical approach whereby a plot of the parameter values is made against the number of iterations (often called *trace plots*). If the chain has converged, then there should be no visible trend in the data (i.e., the series should be flat). The *WinBugs* software package uses this approach, in addition to the MC Error and G-R statistics (BUGS, 2008). For the time being, we have not included a graphical plot of the parameters in this version of *CrimeStat* because of the difficulties in using this plot with the block sampling approach to be discussed shortly.

Also, graphical visualizations, while useful for informing readers, can be misinterpreted. A series that appears to be stable, such as the Baltimore County DWI crash example given above, may actually have a subtle trend. A series can look stable and yet summary statistics such as the G-R statistic and the MC Error relative to the standard error statistic do not indicate convergence.

On the other hand, summary convergence statistics, such as these two measures, are not completely reliable indicators either since a series may only temporarily be stable. This would be especially true for a simulation with a limited number of runs. Both the G-R and MC Error statistics require that at least 2500 iterations be run, with more being desirable. Further, these statistics are not without controversy. Flegal, Haran, & Jones (2008) argue that MCMC standard errors are needed to allow assessment of the accuracy of the estimate while Gelman (2007), in responding to their concerns, argues that a simulation need only be run sufficiently long so that the estimate is more accurate than its standard error. In other words, the precision defines the number of runs needed once the sequence has achieved equilibrium.

Some authors argue that one needs multiple approaches for monitoring convergence (Carlin and Louis, 2000, 182-183). While we would agree with this approach, for the time being we are utilizing primarily the convergence statistics approach.

Statistically Testing Parameters

With an MCMC model, there are two ways that statistical significance can be tested. The first is by assuming that the sampling errors of the algorithm approximate a normal distribution. Thereby, the t-test would be appropriate. In the output table, the t-value is shown, which is the coefficient divided by the standard error. With a simple model, a dependent variable with higher means and adequate sample, this might be a reasonable assumption for a regular Poisson or Poisson-Gamma function. However, for models with many variables and with low sample means, such an assumption is probably not valid (Lord & Miranda-Moreno, 2008). Further, with the addition of many predictor parameters added, the assumption becomes more questionable.

Consequently, MCMC models tend to be tested by looking at the sampling distribution of the parameter and calculating approximate 95% and 99% credible intervals based on the percentile distribution, as illustrated above in Table 17.4.

Proper Specification of a Model

But statistical testing does not just involve testing the significance of the coefficients, whether by asymptotic *t*- or *Z*-tests or by percentiles. A key issue is whether a model is properly specified. On the one hand, a model can be incomplete since there are other variables that could

predict the dependent variable. The Houston burglary model is clearly underspecified since there are additional factors that account for burglaries, as we suggested above.

But, there is also the problem of *overspecifying* a model, that is, including too many independent variables. While the algorithms – MLE or MCMC, can fit virtually any model that is defined, logically many of these models should have never been tested in the first place.

Multicollinearity

The phenomenon of multicollinearity among independent variables is well known, and most statistical texts discuss this. In chapter 15, we briefly discussed multicollinearity among the independent variables. Now, we will show why multicollinearity can be a problem.

In theory, each independent variable should be statistically independent of the other independent variables. Thus, the amount of variance for the dependent variable that is accounted for by each independent variable should be a unique contribution. In practice, however, it is rare to obtain completely independent predictive variables. More likely, two or more of the independent variables will be correlated. The effect is that the estimated standard error of a predictor variable is no longer unique since it shares some of the variance with other independent variables. If two variables are highly correlated, it is not clear what contribution each makes towards predicting the dependent variable. In effect, multicollinearity means that variables are measuring the same thing.

Multicollinearity among the independent variables can produce very strange effects in a regression model. Among these effects are: 1) if two independent variables are highly correlated, but one is more correlated with the dependent variable than the other, the stronger one will usually have a correct sign while the weaker one will sometimes get flipped around (e.g., from positive to negative, or the reverse); 2) two variables can cancel each other out; each coefficient is significant when it alone is included in a model but neither are significant when they are together; 3) one independent variable can inhibit the effect of another correlated independent variable so that the second variable is not significant when combined with the first one; and 4) if two independent variables are virtually perfectly correlated, many regression routines break down because the matrix cannot be inverted. All these effects indicate that there is non-independence among the independent variables.

Aside from producing confusing coefficients, multicollinearity can overstate the predictability of a model. Since every independent variable accounts for some of the variance of the dependent variable, multicollinearity can cause the overall model to ‘improve’ when it probably has not.

A good example of this is a model that we ran relating the number of 1996 crime trips that originated in each of 532 traffic analysis zones in Baltimore County and the City of Baltimore that culminated in a crime committed in Baltimore County. The dependent variable was, therefore, the number of 1996 crimes originating in the zone while there were six independent variables:

1. Population of the zone (1996)
2. An index of relative median household income of the zone (relative to the zone with the highest income)
3. Retail employment in the zone (1996)
4. Non-retail employment in the zone (1996)
5. The number of miles of the Baltimore Beltway (I-695) that passed through the zone
6. Dummy variable indicating whether the Baltimore Beltway passed through the zone.

The last two variables are clearly highly correlated. If a zone has the Baltimore Beltway passing through it, then it has some miles of that freeway assigned to it. The simple Pearson correlation between the two variables is 0.71. Logically, one should not include highly correlated variables in a model. But, what happens if we do this? Table 17.8 illustrates what can happen. Only the coefficients are shown. In the first model, the Beltway miles variable was used along with population, income, retail employment and non-retail employment. In the second model, the dummy variable for whether the Baltimore Beltway passed through the zone or not was used with the four other independent variables. In the third model, both the Beltway miles and the dummy variable for the Baltimore Beltway were both included along with the four other independent variables.

The coefficients for the intercept and the four other independent variables are very similar (and sometimes identical) across the three models. So, look at the two correlated variables. In the first model, the Beltway miles variable is positive, but not significant. In the second model, the Beltway dummy variable is positive and significant. In the third model, however, when both Beltway variables were included, the Beltway miles variable has become negative while the Beltway dummy variable remains positive and significant.

In other words, including two highly correlated variables has caused illogical results. That is, without realizing that the two variables are, essentially, measuring the same thing, one might conclude that the effect of the Beltway passing through a zone is to increase the likelihood that offenders live in that zone but that the effect of having Beltway miles in the zone decreases the likelihood! Any such conclusion is nonsense, of course. In short, do not include highly correlated variables in the same model.

Table 17.8:
Effects of Multicollinearity on Estimation
MLE Poisson-Gamma Model
(N= 532 Traffic Analysis Zones in Baltimore County)

Dependent variable: Number of 1996 crimes that originated in a zone

	(1)	(2)	(3)
Independent Variables	**Model 1**	**Model 2**	**Model 3**
Intercept	1.6437***	1.5932***	1.5964***
Population	0.00045***	0.00045***	0.00045***
Relative Income	-0.0184***	-0.0188***	-0.0188***
Retail Employment	-0.00024*	-0.00026*	-0.00026*
Non-retail Employment	-0.0001***	-0.00013***	-0.00013***
Beltway miles	0.1864n.s.	---	-0.0397n.s.
Beltway	---	0.3194*	0.3496*

n.s. Not significant
* $p \leq .05$
*** $p \leq .001$

How do we know if two or more variables are correlated? There is a simple tolerance test that is included in the MLE models and in the diagnostics utility for the regression module. Tolerance is defined as (repeating equation 15.18, from Chapter 15)

$$Tol_i = 1 - R^2_{j \neq i} \qquad (17.32)$$

where $R^2_{j \neq i}$ is the R-square associated with the prediction of one independent variable with the remaining independent variables in the model. In the example, the tolerance of both the Beltway miles variable and the Beltway dummy variable was 0.49 whereas when each were in the equation by themselves (models 1 and 2), the tolerance was 0.97. The tolerance test should be the first indicator in suspecting too much overlap in two or more independent variables.

The tolerance test is a simple one and is based on normal (OLS) regression. Consequently, it may be erroneous when one or more of the independent variables are highly

skewed. Nevertheless, it is a good indicator of potential problems. When the tolerance of a variable is low, then the variable should be excluded from the model. Typically, when this happens two or more variables will show a low tolerance and the user can choose which one to remove.

How 'low' is low? There is no simple answer to this, but variables with reasonably high tolerance values can have substantial multicollinearity. For example, if there are only two independent variables in a model and they are correlated 0.3, then the tolerance score is 0.91 ($100 - 0.3^2$). While 0.91 appears high, in fact it indicates that there is 9% of overlap between the two variables. *CrimeStat* prints out a warning message about the degree of multicollinearity based on the tolerance levels. But, the user needs to understand that overlapping independent variables can lead to ambiguous and unreliable results. The aim should be to have truly independent variables in a model since the results are more likely to be reliable over time.

Stepwise Variable Entry to Control Multicollinearity

One solution to limiting the number of variables in a model is to use a *stepwise* fitting procedure. There are three standard stepwise procedures (Der & Everitt, 2002, 88-89). In the first procedure, variables are added one at a time (a *forward selection* model). The independent variable having the strongest linear correlation with the dependent variable is added first. Next, the independent variable from the remaining list of independent variables having the highest correlation with the dependent variable *controlling for* the one variable already in the equation is added and the model is re-estimated. In each step, the independent variable remaining from the list having the highest correlation with the dependent variable controlling for the variables already in the equation is added to the model, and the model is re-estimated. This proceeds until either all the independent variables are added to the equation or else a stopping criterion is met. The usual criterion is only variables with a certain significance level are allowed to enter (called a *p-to-enter*).

Second, a *backward elimination* procedure works in reverse. All independent variables are initially added to the equation. The variable with the weakest coefficient (as defined by the significance level and the t- or Z-test) is removed, and the model is re-estimated. Next, the variable with the weakest coefficient in the second model is removed, and the model is re-estimated. This procedure is repeated until either there are no more independent variables left in the model or else a stopping criterion is met. The usual criterion is that all remaining variables pass a certain significance level (called a *p-to-remove*). This ensures that all variables in the model pass this significance level.

The third method is a combination of these procedures, first adding a variable in a forward selection manner but second removing any variables that are no longer significant or

using a backward elimination procedure but allowing new variables to enter the model if they suddenly become significant.

There are advantages to each approach. A fixed model allows specified variables to be included. If either theory or previous research has indicated that a particular combination of variables is important, then the fixed model allows that to be tested. A stepwise procedure might drop one of those variables. On the other hand, a stepwise procedure usually can obtain the same or higher predictability than a fixed procedure.

Within the stepwise procedures, there are also advantages and disadvantages to each method, though the differences are generally very small. A forward selection procedure adds variables one at a time. Thus, the contribution of each new variable can be seen. On the other hand, a variable that is significant at an early stage could become insignificant at a later stage because of the unique combinations of variables. Similarly, a backward elimination procedure will ensure that all variables in the equation meet a specified significance level. But, the contribution of each variable is not easily seen other than through the coefficients. In practice, one usually obtains the same model with either procedure, so the differences are not that critical.

A stepwise procedure will not guarantee that multicollinearity will be removed entirely. However, it is a good procedure for narrowing down the variables to those that are significant. Then, any co-linear variables can be dropped manually and the model re-estimated.

In the normal and MLE Poisson routines, there is a backward elimination procedure whereby variables are dropped from an equation if their coefficients are not significant.

Overfitting

Overfitting is a more general phenomenon of including too many variables in an equation (Radford, 2006; Nannen, 2003). With the development of Bayesian models, this has become an increasing occurrence because the models, usually estimated with the MCMC algorithm, can fit an enormous number of parameters. Many of these models estimate parameters that are properties of the functions used (called *hyperparameters*) rather than just the variables input as part of the data. In the Poisson-Gamma-CAR model, for example, we estimate the dispersion parameter (ψ) and a general Φ function. Phi (Φ), in turn, is a function of a global component (Rho, ρ), a local component (Tauphi τ_Φ), and a neighborhood component (Alpha -α).

These parameters are part of the functions and are not data. But, since they can vary and are often estimated from the data, there is always the potential that they could be highly correlated and, thereby, cause ambiguous results to occur. Unfortunately, there are not good diagnostics for multicollinearity among the hyperparameters, as there is with the tolerance test.

But, the problem is a real one and one that the user should be cognizant. Sometimes an MCMC or MLE model fails to converge properly, meaning that it either did not finish or else produced inconsistent results from one run to another. We usually assume that the probability structure of the space being modeled is too complex for the model that we are using. And, while that may be true, it is also possible that there is overlap in some of the hyperparameters. In this case, one would be better off choosing a simpler model – one with fewer hyperparameters, than a more complex one.

Condition Number of Matrix

In other words, a user should be very cautious about overfitting models with too many variables, both the data variables and those estimated from functions (the hyperparameters). We have included a condition matrix test for the distance matrix in the Poisson-Gamma-CAR/SAR model. The condition number of a matrix is an indicator of how amenable it is to digital solution (Wikipedia, 2010b). A matrix with a low condition number is said to be well conditioned whereas one with a high number is said to be ill-conditioned. With ill-conditioned matrices, the solutions are volatile and inconsistent from one run to another. How 'high' is high? Numbers higher than, say, 400 are generally ill-conditioned while low condition numbers (say, under 100) are well conditioned. Between 100 and 400 is an ambiguous area. For the Poisson-Gamma-CAR model, if you see a condition number higher than 100, be cautious. If you see one higher than 400, assume the results are completely unreliable with respect to the spatial component.

Overfitting and Poor Prediction

There is also a question about the extent to which a model that is fit is reliable and accurate for predicting a data set which is different. Without going into an extensive literature review, a few guidelines can be given. The Machine Learning computing community concentrates on *training* samples in order to estimate parameters and then using the estimated models to predict a *test* sample (another data set). In general, they have found that simple models do better for prediction than complicated models. One can always fit a particular data set by adding variables or adding complexity to the mathematical function. On the other hand, the more complex the model – the more independent variables in it and the more specified hyperparameters, generally the model will do worse when applied to a new data set. Nannen (2003) called this the *paradox of overfitting*, and it is a rule that a user would be well advised to follow. Try to keep your models simple and reliable. In the long run, simple models with well-defined independent variables will generally do better for prediction.

Improving the Performance of the MCMC Algorithm

Most medium- and large police departments use large datasets, such as calls for service, crime reports, motor vehicle crash reports and other data sets. The largest police departments have huge data sets, constituting millions of records. Further, these data are being collected on a continual basis. *CrimeStat* was developed to handle fairly large data sets and the routines are optimized for this.

However, large data sets pose a problem for multivariate modeling in a number of ways. First, they pose a computing problem in terms of the processing of information. As the number of records goes up, the demand for computer resources increases exponentially. For example, consider the problem of calculating a distance matrix for use in, say, the Poisson-Gamma-SAR model. If each number is represented by 64 bits (double precision), then the amount of memory space required is a function of K^2*64 where K is the number of records. For example, if there are 10,000 records (a relatively small database by police standards), then the amount of memory required will be 10,000*10,000*64 = 6.4 billion bits (or 800 Mb). On the other hand, if the number of records is 100,000, then the memory demand goes up to 80,000 Mb (or 80 Gb). That such databases take a long time to be analyzed is understandable.

Second, large data sets pose problems for interpretation. The 'gold standard' for testing of coefficients or even the overall fit of a model has been to compare the coefficients to 0. This follows from traditional statistics (whom the Bayesians call *frequentists*) whereby a particular statistic (in this case, a regression coefficient) is compared to a 'null hypothesis' which is usually 0. However, with large datasets, especially with extremely large datasets, virtually all coefficients will be significantly different from 0, no matter how they are tested (with t-tests or with percentiles). In this case, 'significance' does not necessarily mean 'importance'. For example, if you have a data set of one million records and plug in a model with 10 independent variables, the chances are that the majority of the variables will be significantly different than 0. This does not mean that the variables are important in any way, only that they account for some of the variance of the dependent variable greater than what would be expected on the basis of chance.

The two problems interact when a user works with a very large dataset. The routines may have difficulty calculating the solution and the results may not necessarily be very meaningful. This will be particularly true for complex models, such as the Poisson-Gamma-CAR which will be discussed in chapter 19. An example will illustrate this. With an Intel 2.4 Ghz computer with a dual core, we ran a model with three independent variables on a scalable dataset; that is, we took a large dataset and sampled smaller subsets of it. We then tested the MCMC Poisson-Gamma and MCMC Poisson-Gamma-CAR models with subsets of different size. Table 17.9 present the results.

As can be seen, the calculation time went up exponentially with the sample size. Further, with the spatial Poisson-Gamma-CAR model, a limit was reached. Because the routine was calculating the distance between each observation and every other observation as part of the spatial weight coefficients, the memory demands blow up very quickly. The non-spatial Poisson-Gamma model can be run on larger datasets (we have run them on sets as large as 100,000 records) but the spatial model cannot be. Even with the non-spatial model, the calculation time for a very large dataset goes up very substantially with the sample size.

Table 17.9:
Effects of Sample Size on Calculations
(Second to Complete)

Sample size	Poisson-Gamma	Poisson-Gamma-CAR
125	23	67
250	43	163
500	81	480
1,000	160	1,569
2,000	305	6,000
4,000	622	25,740
5,000	762	43,740
8,000	1,247	Unable to complete
12,000	1,869	Unable to complete
15,000	2,412	Unable to complete
20,000	3,278	Unable to complete

Scaling of the Data

There are several things that can be done to improve the performance of the MCMC algorithm with large datasets. The first is to scale the data, either by reducing the number of digits that represent each value or by standardizing by Z-scores. There are different ways to scale the data, but a simple one is to move the decimal places. For example, if one of the variables is median household income and is measured in tens of thousands (e.g., 55,000, 135,000), then these values can be divided by 1000 so that they represent ‘per 1000’ (i.e., 55.0 and 135.0 in the example).

To illustrate, we ran a single-family housing value model on a large data set of 588,297 single-family home parcels. The data came from the Harris County Appraisal District and the model related the 2007 assessed value against the square feet of the home, the square feet of the parcel, the distance from downtown Houston and two dummy variables - whether the home had

received a major remodeling between 1985 and 2007 and whether the parcel was within 200 feet of a freeway. The valuations were coded as true dollars and were then re-scaled into units of 'per 1000' (e.g., 45,000 became 45.0). When the data were in real units, the time to complete the run was 20.8 minutes for the MCMC Poisson-Gamma using the Block Sampling Method (see below). When the data were in units of thousandths, the time to complete the run was 15.3 minutes for the MCMC Poisson-Gamma.

In other words, scaling the data by reducing the number of decimal places led to an improvement in calculating time of around 25% for the MCMC model. The effects on an MLE model will be even more powerful due to the different algorithm used. The point is, scaling your data will pay in terms of improving the efficiency of runs.

Block Sampling Method for the MCMC

Another solution is to sample records from the full database and run the MCMC algorithm on that sample. In the MCMC literature, drawing a sub-sample is called 'thinning' the sample (Link & Eaton, 2011). Essentially, a sub-sample is drawn and the MCMC algorithm is run. It is clearly much faster to run a sub-sample than the entire database. However, the problem with this approach, as pointed out by McEachern & Berliner (1994) is that it will be less precise than by running the full database. The reason is that there is sampling error and that the results from any one sub-sample might deviate from the full database.

With the block sampling method, on the other hand, multiple subsamples are drawn with the overall statistics based on a summary of the individual samples. That is, a first sub-sample is drawn and run through the MCMC algorithm. The statistics from the run are calculated. Then, the process is repeated with another sample, and the statistics are calculated on this sample. Then, the process is repeated again and again. We call this the *block sampling method* and it has been implemented in *CrimeStat*. The advantage over a thinned sample is that, because of the Central Limit Theorem, the summary statistics for the repeated samples will converge towards the summary statistics of the full database with much smaller sampling error.

With the block sampling method, the user defines three parameters for controlling the sampling:

1. The block sampling threshold – the size of the database beyond which the block sampling method will be implemented. For example, the default block sampling threshold is set at 6,000 observations, though the user can change this. With this default, any dataset that has fewer than 6,000 records/observations will be analyzed with the full database. However, any dataset that has 6,000 records or more will cause the block sampling routine to be implemented. Note that the user run the entire

dataset, no matter how long it takes, by setting the block sampling threshold to be greater than the number of records in the dataset.

2. Average block size – the expected block size of a sample from the block sampling method. The default is 400 records thought the user can change this. The routine defines a sampling interval, based on *n/N* where *n* is the defined average block size and *N* is the total number of records. For drawing a sample, however, a uniform random number from 0 to 1 is drawn and compared to the ratio of *n/N*. If the number is equal to or less than this ratio (probability), then the record is accepted for the block sample; if the number is greater than this ratio, the record is not accepted for the block sample. Thus, any one sample may not have exactly the number of records defined by the user. But, on average, the average sample size over all runs will be very close to the defined average block size though the variability is high.

3. Number of samples – the number of samples drawn. The default is 25 though the user can change this. We have found that 20-30 samples produce very reasonable results.

The routine then proceeds to implement the block sampling method. For example, if the user keeps the default parameters, then the block sampling method will only be implemented for databases of 6,000 records or more. If the database passes the threshold, then each of the 25 samples are drawn with, approximately, 400 records per sample. The MCMC algorithm is run on each of the samples and the statistics are calculated. After all 25 samples have been run, the routine summarizes the results by averaging the summary statistics (likelihood, AIC, BIC/SC, etc), the coefficients, the standard errors, and the percentile distribution. The results that are printed represent the average over all 25 samples.

GUIDELINE:

Note that MCMC models can take a very long time to calculate. For large datasets, we recommend using the block sampling method. A rough rule-of-thumb is that for non-spatial MCMC models, the block sampling method should be used for 6,000 or more cases while for spatial MCMC models, the block sampling method should be used for 2,000 or more cases. Of course, this will depend on the amount of available RAM as well as the processing speed of the computer.

We have found that this method produces very good approximations to the full database. For several datasets, we have compared the results of the block sampling method with running the full database through the MCMC routine. The means of the coefficients appear to be unbiased estimates of the coefficients for the full database. Similarly, the percentiles appear to be very close, if not unbiased, estimates of the percentiles for the full database. On the other hand, the standard errors appear to be biased estimates of standard errors of the full database.

The reason is that they are calculated from a sample of n observations where the standard errors of the full database are calculated from N observations. An adjusted standard error is produced which approximates the true standard error of the full database. It is defined as;

$$AdjStd.Err = StdErr_{block} * \sqrt{\frac{\bar{n}}{N}} \tag{17.33}$$

where $StdErr_{block}$ is the average standard error from the k samples, N is the total number of records, and $\bar{n}$ is the average block size (the empirical average, not the expected sample size). This is only output when the block sampling method is used.

Comparison of Block Sampling Method with Full Dataset

Test 1

A test was constructed to compare the block sampling method with the full MCMC method on two datasets. The first dataset contained 4000 road segments in the Houston metropolitan area and the model that was run was a traffic model relating vehicle miles traveled (VMT - the dependent variable) against the number of lanes, the number of lane miles, and the volume-to-capacity ratio of the segment. It is not a very meaningful model but was used to test the algorithm.

The dataset was tested with the MCMC model using all records (the full dataset) and the block sampling method. For simplicity, the variables have been called $X_1 \ldots X_k$. The significance levels of the coefficients for the full dataset based on the t-test are shown, since these are based on the estimated standard errors rather than the adjusted standard errors.

Table 17.10 shows the results of the traffic dataset. Comparing the full sample results with the block sample results, the coefficients are very close to each other, within the second decimal place. Similarly, the adjusted standard errors are very close within the third decimal place. On the other hand, the block sampling method took 11.2 minutes to run compared to only

Table 17.10:
Comparing Block Sampling Method with Full Database
MCMC Poisson-Gamma Model
Houston Traffic Dataset
(Time to Complete)

Dependent variable = Vehicle Miles Traveled

	Full dataset		**Block Sampling method**		
	(N=4000)		(n = 402.9)		
Iterations:	20,000		20,000		
Burn in:	5,000		5,000		
Number of samples:	1		20		
Time to complete run:	7.7 minutes		11.2 minutes		
Variable	**Coefficient**	**Std. Error**	**Coefficient**	**Std. Error**	**Adj. Std. Error**
Intercept	4.5414***	0.045	4.5498***	0.140	0.044
X_1	0.6254***	0.022	0.6267***	0.066	0.021
X_2	0.8502***	0.020	0.8618***	0.064	0.020
X_3	2.4163***	0.049	2.3938***	0.154	0.049

Significance of block sampling method based on unadjusted standard error
*** $p \leq .001$

7.7 for the full dataset. With a dataset of this size (N=4000), there was no advantage for the block sampling method even though it produced very similar results.

Now, let's take a more complicated dataset. The second represented 97,429 crimes committed in Manchester, England. It is part of a study on gender differences in crime travel (Levine & Lee, 2012). The model related the journey to crime distance against 14 independent variables involving spatial location, land use, type of crime, ethnicity of the offender, prior conviction history, and gender.

Test 2

Table 17.11 shows the results of the journey to crime dataset. Not all of the variables were significant, according to the t-test of the full dataset. In this case, there were greater discrepancies in the coefficients between the full dataset and the block sampling method. The signs of the coefficients were identical for all parameters except X_{10}, which was not significant. For all parameters, though, the coefficient for the full dataset was within the 95% credible interval of the block sampling method. That is, since this is a sample, the sampling error of the block sampling method incorporates the coefficient for the full dataset for all 16 parameters.

The adjusted standard errors from the block sampling method were quite close to the standard errors of the full dataset; the biggest discrepancy was 0.004 for variable X_6 and is about 15% larger. Most of the adjusted standard errors are within 10% of the standard error for the full dataset, and three are exactly the same. Further, where there is a discrepancy, the adjusted standard errors were slightly larger, suggesting that this is a conservative adjustment.

In short, the block sampling method produced reasonably close results to that of the full dataset for both the coefficients and the standard errors. Given that this model was a very complex one (with 14 independent variables), the fit was good. The biggest advantage of the block sampling method, on the other hand, is the efficiency of it. The block sampling method took 222.7 minutes to run compared to 4,855.1 minutes for the full dataset, an improvement of more than 20 times! Running a large dataset through the MCMC algorithm is a very time consuming process. The block sampling approach produced reasonably close results in a much shorter period of time.

Statistical Testing with Block Sampling Method

Regarding statistical testing of the coefficients, however, we think that the modeled standard errors (or percentiles) be used rather than the adjusted errors. The adjusted standard error is an approximation to the full dataset *if* that dataset had been run. In most cases, it will not have been run. On the other hand, the standard errors estimated from the block sampling method and the percentile distribution were the products of running the individual samples. The errors are larger because the samples were much smaller. But, because this was the method used, statistical inferences should be based on the sample.

What to do if there is a discrepancy? For some datasets, the coefficients from the block sampling method will not be significant whereas they would be if the full dataset was run. In the Manchester example above, only 3 of the coefficients were significant using the block sampling method compared to 14 for the full dataset. This brings up a statistical dilemma. Does one adopt the adjusted standard errors and then re-test the coefficients using the asymptotic t-test or does

Table 17.11:

Comparing Block Sampling Method with Full Database MCMC Poisson-Gamma Model

Manchester Journey to Crime Dataset

(Time to Complete)

Dependent variable = Distance traveled

	Full dataset		**Block Sampling method**		
	(N = 97,429)		(n=402.8)		
Iterations:	100,000		100,000		
Burn in:	10,000		10,000		
Number of samples:	1		30		
Time to complete:	4,855.1 minutes		222.7 minutes		
Variable	**Coefficient**	**Std. Error**	**Coefficient**	**Std. Error**	**Adj. Std. Error**
Intercept	0.2096***	0.018	0.2103	0.321	0.021
X_1	0.8871***	0.025	1.0135*	0.430	0.028
X_2	0.3311***	0.018	0.3434	0.294	0.019
X_3	-0.2274***	0.012	-0.2751	0.199	0.013
X_4	-0.2820***	0.014	-0.3137	0.231	0.015
X_5	0.2525***	0.016	0.3099	0.256	0.016
X_6	0.3560***	0.027	0.3783	0.488	0.031
X_7	0.0753***	0.013	0.1092	0.214	0.014
X_8	0.1766***	0.021	-0.0030	0.374	0.024
X_9	0.1880***	0.023	0.1326	0.406	0.026
X_{10}	0.0135n.s.	0.016	-0.0070	0.268	0.017
X_{11}	-0.5697***	0.016	-0.6759	0.265	0.017
X_{12}	0.0042n.s.	0.014	0.0521	0.226	0.015
X_{13}	-0.2214***	0.016	-0.2755	0.262	0.017
X_{14}	0.0056***	0.001	-0.00004	0.016	0.001
Error	-0.7299***	0.008	-0.7062***	0.139	0.009

Based on asymptotic t-test:

n.s. Not significant

* $p \leq .05$

*** $p \leq .001$

one accept the estimated standard errors and the percentiles? Our opinion is to do the latter. The former is making an assumption (and a big one) that the adjusted standard errors will be a good approximation to the real ones. In these two datasets, this appears to be the case. But, we have no theoretical basis for assuming that. It has just worked out for these and a couple of other datasets that we have tested.

Therefore, the choice for a researcher is to do one of three things if some of the coefficients are not significant using the block sampling method when it appears that they might be if the full dataset would be used.

First, one could always run the full dataset through the MCMC algorithm. If the dataset is large, then it will take a long time to calculate. But, if it is important, then the user should do that. Note that it will be possible to do this only for the Poisson-Gamma model and not for the Poisson-Gamma-CAR/SAR spatial model.

Second, the researcher could try to tweak the MCMC algorithm to increase the likelihood of finding statistical significance for the coefficients increasing the number of iterations to improve the precision of the estimate and by increasing the average sample size of the block sample. If 400 samples were not sufficient, perhaps 600 would be? In doing this, the efficiency advantage of the block sampling method becomes less important compared to improving the accuracy of the estimates.

Third, the researcher can accept the results of the block sampling method and 'live' with the conclusions. If one or more variables was not significant using the block sampling method (which, after all, was based on 20 to 30 samples of around 400 records each), then the variables are probably not important. In other words, running the MCMC algorithm on the full dataset or increasing the sample size of the block samples may find statistical significance in one or more variables. But, the chances are that the variables are not very important, from a statistical perspective.

In our experience, the strongest variables are significant with the block sampling scheme. Perhaps the researcher or analyst should focus on those and build a model around them, rather than scouring for other variables that have very small effect? In short, our opinion is that a smaller, but more robust, model is better than a larger, more volatile one. In terms of understanding, the major variables need to be isolated because they contribute the most to the development of theory. In terms of prediction, the strongest variables will also have the biggest impact. Elegance in a model should be the aim, not a comprehensive list of variables that might be important but probably are not.

References

Anselin, L. (2002). Under the hood: Issues in the specification ad interpretation of spatial regression models, *Agricultural Economics*, 17(3), 247-267.

Besag, J., Green, P., Higdon, D. & Mengersen, K. (1995). Bayesian computation and stochastic systems (with discussion), *Statistical Science*, 10, 3-66.

BUGS (2008). *The BUGS (Bayesian Inference Using Gibbs Sampling) Project*. MRC Biostatistics Unit, University of Cambridge: Cambridge. http://www.mrc-bsu.cam.ac.uk/bugs/. Accessed March 23, 2010.

Cameron, A. Colin & Trivedi, Pravin K. (1998). *Regression Analysis of Count Data*. Cambridge University Press: Cambridge, U.K.

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

Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C. (2009). Ch. 16: Greedy algorithms, *Introduction to Algorithms*, MIT Press: Cambridge, MA.

Denison, D.G.T., Holmes, C. C., Mallick, B. K. & Smith, A. F. M (2002). *Bayesian Methods for Nonlinear Classification and Regression*. John Wiley & Sons, Ltd: Chichester, Sussex.

Der, G. & Everitt, B. S. (2002). *A Handbook of Statistical Analyses using SAS*. Chapman & Hall/CRC: London.

De Smith, M., Goodchild, M. F., & Longley, P. A. (2007). *Geospatial Analysis* (second edition). Matador: Leicester, U.K.

Dijkstra, E. W. (1959). A note on two problems in connection with graphs, *Numerische Mathematik*, 1, 269-271.

El-Basyouny K. & Sayed, T. (2009). Collision prediction models using multivariate Poisson-lognormal regression. *Accident Analysis & Prevention*, 41(4), 820-828.

Flegal, J. M., Haran, M., & Jones, G. L. (2008). Markov Chain Monte Carlo: Can we trust the third significant figure?, *Statistical Science*, 23 (2), 250-60.

References (continued)

Geedipally, S.R., Lord, D., & Dhavala, S.S. (2012). The Negative-Binomial-Generalized-Lindley Generalized Linear Model: Characteristics and application using crash data. *Accident Analysis & Prevention*, 45 (2), 258-265.

Gelman, A. (2007). Markov Chain Monte Carlo standard errors. Note on paper by Flegal, Haran & Jones. http://andrewgelman.com/2007/04/markov_chain_mo/.

Gelman, A. (1996). Inference and monitoring convergence. In Gilks, W. R., S. Richardson, & D. J. Spiegelhalter (eds), *Markov Chain Monte Carlo in Practice*, Chapman and Hall: London.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2004). *Bayesian Data Analysis* (second edition). Chapman and Hall/CRC: Boca Raton, FL.

Gelman, A. & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences (with discussion), *Statistical Science,* 7, 457-511.

Ghitany, M.E., Atieh, B., Nadarajah, S. (2008). Lindley distribution and its application. *Mathematics and Computers in Simulation*, (78), 39-49.

Goldfield, S. M., Quandt, R. E., & Trotter, H. F. (1966). Maximization by quadratic hill-climbing, *Econometrica*, 34 (3), 541-551.

Guikema, S.D. & Coffelt, J. P. (2008). A flexible count data regression model for risk analysis, *Risk Analysis*, 28 (1), 213–223.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov Chains and their applications, *Biometrika*, 57, 97-109.

Lambert, D. (1992). Zero-Inflated Poisson Regression, With an Application to Defects in Manufacturing. *Technometrics*, 34 (1), 1-14.

Lee, P. M. (2004). *Bayesian Statistics: An Introduction* (third edition). Holder Arnold: London.

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

References (continued)

Levine, N. (2011). Spatial variation in motor vehicle crashes by gender in the Houston Metropolitan Area. *Proceedings of the 4th International Conference on Women's Issues in Transportation. Volume II: Technical Papers*, Transportation Research Board: Washington, DC. 12-25. http://onlinepubs.trb.org/onlinepubs/conf/cp46v2.pdf.

Levine, N. & Canter, P. (2011). Linking origins with destinations for DWI Motor Vehicle Crashes: An application of crime travel demand modeling. *Crime Mapping*, 3, 7-41.

Levine, N. & Lee, P. (2012).). "Crime travel of offenders by gender and age in Manchester, England". In press, Michael Leitner (ed), *Crime Modeling and Mapping Using Geospatial Technologies,* Springer.

Lindley, D.V. (1958). Fiducial distributions and Bayes' theorem. *J. R. Stat. Soc.*, (20), 102-107.

Link, W. A. & Eaton, M. J. (2011). On thinning of chains in MCMC. *Methods in Ecology and Evolution*, June. http://onlinelibrary.wiley.com/doi/10.1111/j.2041-210X.2011.00131.x/abstract.

Lord, D. & Geedipally, S.R. (2011). The negative binomial–Lindley distribution as a tool for analyzing crash data characterized by a large amount of zeros. *Accident Analysis & Prevention*, 43 (5), 1738-1742.

Lord, D. (2006). Modeling motor vehicle crashes using Poisson-gamma models: Examining the effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter. *Accident Analysis and Prevention*, 38, 751-766.

Lord, D. & Miranda-Moreno, L. F. (2008). Effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter of Poisson-gamma models for modeling motor vehicle crashes: A Bayesian Perspective. *Safety Science*, 46 (5), 751-770.

Lord, D., Manar, A., & Vizioli, A. (2005). Modeling Crash-Flow-Density and Crash-Flow-V/C Ratio for Rural and Urban Freeway Segments. *Accident Analysis & Prevention*. 37 (1), 185-199.

Lynch, Scott M. (2007). *Introduction to Applied Bayesian Statistics and Estimation for Social Scientists*. Springer: New York.

McEachern, S. N. & Berliner, L. M. (1994). Subsampling the Gibbs Sampler, *The American Statistician*, 48, 188–190.

References (continued)

Ma, J., Kockelman, K. M., & Damien, P. (2008). A multivariate Poisson-lognormal regression model for prediction of crash counts by severity, using Bayesian methods. *Accident Analysis & Prevention*, 40 (3), 964-975.

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

Miaou, S. P. (2006). Coding instructions for the spatial regression models in CrimeStat. Unpublished manuscript. College Station, TX.

Mitra, S. & Washington, S. (2007). On the nature of over-dispersion in motor vehicle crash prediction models, *Accident Analysis and Prevention*, 39, 459-468.

Nannen, V. (2003). *The Paradox of Overfitting*. Artificial Intelligence, Rijksuniversitat: Groningen, Netherlands. http://volker.nannen.com/pdf/the_paradox_of_overfitting.pdf. Accessed March 11, 2010.

Ntzourfras, I. (2009). *Bayesian Modeling using WinBugs*. Wiley Series in Computation Statistics, Wiley: New York.

Park, B. J. (2009). Note on the Bayesian analysis of count data. From Park, Byung-Jung PhD thesis, Texas A & M University: College Station, TX.

Radford, N. (2006). The problem of overfitting with maximum likliehood . CSC 411: Machine Learning and Data Mining, University of Toronto: Toronto, CA. http://www.cs.utoronto.ca/~radford/csc411.F06/10-nn-early-nup.pdf Accessed March 11, 2010.

Radford, N. (2003). Slice sampling, *Annals of Statistics*, 31(3), 705-767.

Sellers, K. S. & Shmueli, G. (2010), A flexible regression model for count data, *Annals of Applied Statistics*, 4 (2), 943-961.

So, A. M., Ye, Y., & Zhang, J. (2007). Greedy algorithms for metric facility location problems. In Gonzalez, T. F. (Ed), *Handbook of Approximation Algorithms and Metaheuristics*, CRC Computer & Information Sciences Series, Chapman & Hall/CRC: Boca Raton, FL, Chapter 39.

References (continued)

Spiegelhalter, D.J., Best, N.G., Carlin, B.P., & Van der Linde, A. (2002) Bayesian measures of model complexity and fit. Journal of the Royal Statistical Society, Series B, Vol. 64, No. 4, pp. 583-639.

Train, K. (2009). *Discrete Choice Methods with Simulation* (2nd edition). Cambridge University Press: Cambridge.

Wikipedia (2012). Metropolis-Hastings algorithm, *Wikipedia*. http://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm. Accessed July 2, 2012.

Wikipedia (2010a). Greedy algorithm, *Wikipedia*. http://en.wikipedia.org/wiki/Greedy_algorithm. Accessed March 12, 2010.

Wikipedia (2010b). Condition number. *Wikipedia.* *http://en.wikipedia.org/wiki/Condition_number*. Accessed March 19, 2010