Metropolis-Hastings (continued)

Algorithm (Metropolis-Hastings sampling). To construct a Markov chain whose equilibrium distribution is $p(\theta|y)$, choose a proposal distribution $g(\theta^*|\theta_t, y)$, define the acceptance probability $\alpha_{MH}(\theta^*|\theta_t, y)$ by (14), and

\[
\text{Initialize } \theta_0; \ t \leftarrow 0 \\
\text{Repeat } \{ \\
\quad \text{Sample } \theta^* \sim g(\theta|\theta_t, y) \\
\quad \text{Sample } u \sim \text{Uniform}(0, 1) \\
\quad \text{If } u \leq \alpha_{MH}(\theta^*|\theta_t, y) \text{ then } \theta_{t+1} \leftarrow \theta^* \\
\quad \text{else } \theta_{t+1} \leftarrow \theta_t \\
\quad t \leftarrow (t + 1) \\
\}
\]

(15)

When the proposal distribution is symmetric in the Metropolis et al. sense, the acceptance probability ratio reduces to \( \frac{p(\theta^*|y)}{p(\theta|y)} \), which is easy to motivate intuitively: whatever the target density is at the current point \( \theta_t \), you want to visit points of higher density more often and points of lower density less often, and it turns out that (14) does this for you in the natural and appropriate way.

As an example of the MH algorithm in action, consider a Gaussian model with known mean \( \mu \) and unknown variance \( \sigma^2 \) applied to the NB10 data in part 2 of the lecture notes.

The likelihood function for \( \sigma^2 \), derived from the sampling model \( (Y_i|\sigma^2) \overset{\text{IID}}{\sim} N(\mu, \sigma^2) \) for \( i = 1, \ldots, n \), is

\[
l(\sigma^2|y) = c \prod_{i=1}^{n} (\sigma^2)^{-\frac{1}{2}} \exp \left[ -\frac{(y_i - \mu)^2}{2\sigma^2} \right] \\
= c (\sigma^2)^{-\frac{n}{2}} \exp \left[ -\frac{\sum_{i=1}^{n} (y_i - \mu)^2}{2\sigma^2} \right].
\]

(16)
MH Sampling (continued)

This is recognizable as a member of the Scaled Inverse $\chi^2$ family $\chi^{-2}(\nu, s^2)$ (e.g., Gelman, Carlin et al. (2003)) of distributions, which (as we saw in part 2 of the lecture notes) is a rescaled version of the Inverse Gamma family chosen so that $s^2$ is an estimate of $\sigma^2$ based upon $\nu$ “observations.”

You can now convince yourself that if the prior for $\sigma^2$ in this model is taken to be $\chi^{-2}(\nu, s^2)$, then the posterior for $\sigma^2$ will also be Scaled Inverse $\chi^2$: with this choice of prior

$$p(\sigma^2|y) = \chi^{-2}\left[\nu + n, \frac{\nu s^2 + \sum_{i=1}^{n}(y_i - \mu)^2}{\nu + n}\right].$$

(17)

This makes good intuitive sense: the prior estimate $s^2$ of $\sigma^2$ receives $\nu$ votes and the sample estimate $\hat{s}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)^2$ receives $n$ votes in the posterior weighted average estimate $\frac{\nu s^2 + n\hat{s}^2}{\nu + n}$.

Equation (17) provides a satisfying closed-form solution to the Bayesian updating problem in this model (e.g., it’s easy to compute posterior moments analytically, and you can use numerical integration or well-known approximations to the CDF of the Gamma distribution to compute percentiles).

For illustration purposes suppose instead that you want to use MH sampling to summarize this posterior.

Then your main choice as a user of the algorithm is the specification of the proposal distribution (PD) $g(\sigma^2|\sigma^2_t, y)$.

The goal in choosing the PD is getting a chain that mixes well (moves freely and fluidly among all of the possible values of $\theta = \sigma^2$), and nobody has (yet) come up with a sure-fire strategy for always succeeding at this task.

Having said that, here are two basic ideas that often tend to promote good mixing:
MH Sampling (continued)

(1) Pick a PD that looks like a somewhat overdispersed version of the posterior you’re trying to sample from (e.g., Tierney (1996)).

Some work is naturally required to overcome the circularity inherent in this choice (if I fully knew \( p(\theta | y) \) and all of its properties, why would I be using this algorithm in the first place?).

(2) Set up the PD so that the expected value of where you’re going to move to (\( \theta^* \)), given that you accept a move away from where you are now (\( \theta_t \)), is to stay where you are now: \( E_g(\theta^*|\theta_t, y) = \theta_t \).

That way, when you do make a move, there will be an approximate left-right balance, so to speak, in the direction you move away from \( \theta_t \), which will encourage rapid exploration of the whole space.

Using idea (1), a decent choice for the PD in the Gaussian model with unknown variance might well be the Scaled Inverse \( \chi^2 \) distribution: \( g(\sigma^2|\sigma_t^2, y) = \chi^{-2}(\nu_*, \sigma_*^2) \).

This distribution has mean \( \frac{\nu_*}{\nu_*-2} \sigma_*^2 \) for \( \nu_* > 2 \).

To use idea (2), then, I can choose any \( \nu_* \) greater than 2 that I want, and as long as I take \( \sigma_*^2 = \frac{\nu_*-2}{\nu_*} \sigma_t^2 \) that will center the PD at \( \sigma_t^2 \) as desired.

So I'll use

\[
g(\sigma^2|\sigma_t^2, y) = \chi^{-2} \left( \nu_*, \frac{\nu_*-2}{\nu_*} \sigma_t^2 \right).
\] (18)

This leaves \( \nu_* \) as a kind of potential tuning constant—the hope is that I can vary \( \nu_* \) to improve the mixing of the chain.
The above figure (motivated by an analogous plot in Gilks et al. (1996)) presents **time series traces** of some **typical output** of the MH sampler with $\nu_* = (2.5, 20, 500)$.

The **acceptance probabilities** with these values of $\nu_*$ are $(0.07, 0.44, 0.86)$, respectively.

The **SD** of the $\chi^{-2}(\nu_*, \frac{\nu_* - 2}{\nu_*} \sigma_t^2)$ distribution is proportional to $\frac{\nu_*^2}{(\nu_* - 2)^2 \sqrt{\nu_* - 4}}$, which decreases as $\nu_*$ increases, and this turns out to be **crucial**: when the proposal distribution SD is too **large** (small $\nu_*$, as in the top panel in the figure), the algorithm tries to make **big jumps** around $\theta$ space (good), but almost all of them get **rejected** (bad), so there are long periods of no movement at all, whereas when the PD SD is too **small** (large $\nu_*$; see the bottom panel of the figure), the algorithm **accepts** most of its proposed moves (good), but they’re so tiny that it takes a **long time to fully explore the space** (bad).
MH Sampling (continued)

Gelman, Roberts, et al. (1995) have shown that in simple problems with approximately normal target distributions, the optimal acceptance rate for MH samplers like the one illustrated here is about 44% when the vector of unknowns is one-dimensional, and this can serve as a rough guide: you can modify the proposal distribution SD until the acceptance rate is around the Gelman et al. target figure.

The central panel of the figure displays the best possible MH behavior in this problem in the family of PDs chosen.

Even with this optimization you can see that the mixing is not wonderful, but contemporary computing speeds enable huge numbers of draws to be collected in a short period of time, compensating for the comparatively slow rate at which the MH algorithm learns about the posterior distribution of interest.

In this example the unknown quantity $\theta = \sigma^2$ was real-valued, but there’s nothing in the MH method that requires this; in principle it works equally well when $\theta$ is a vector of any finite dimension (look back at the algorithm in (15) to verify this).

Notice, crucially, that to implement this algorithm you only need to know how to calculate $p(\theta|y)$ up to a constant multiple, since any such constant will cancel in computing the acceptance probability (15)—thus you’re free to work with unnormalized versions of $p(\theta|y)$, which is a great advantage in practice.
MH Sampling (continued)

There’s even **more flexibility** in this algorithm than might first appear: it’s often possible to identify a set $A$ of **auxiliary variables**—typically these are **latent** (unobserved) quantities—to be sampled along with the parameters, which have the property that they **improve the mixing** of the MCMC output (even though extra time is spent in sampling them).

When the set $(\theta, A)$ of quantities to be sampled is a **vector of length** $k$, there is additional flexibility: you can **block update** all of $(\theta, A)$ at once, or with appropriate modifications of the acceptance probability you can divide $(\theta, A)$ up into **components**, say $(\theta, A) = (\lambda_1, \ldots, \lambda_l)$, and **update the components one at a time** (as Metropolis et al. originally proposed in 1953).

The idea in this **component-by-component** version of the algorithm, which Gilks et al. (1996) call **single-component** MH sampling, is to have $k$ **different** proposal distributions, one for each component of $\theta$.

Each **iteration** of the algorithm (indexed as usual by $t$) has $k$ steps, indexed by $i$; at the beginning of iteration $t$ you **scan** along, updating $\lambda_1$ first, then $\lambda_2$, and so on until you’ve updated $\lambda_k$, which **concludes** iteration $t$.

Let $\lambda_{t,i}$ stand for the **current state** of component $i$ at the end of iteration $t$, and let $\lambda_{-i}$ stand for the $(\theta, A)$ vector with component $i$ **omitted** (the notation gets awkward here; it can’t be helped).

The proposal distribution $g_i(\lambda_i^* | \lambda_{t,i}, \lambda_{t,-i}, y)$ for component $i$ is allowed to **depend** on the most recent versions of all components of $(\theta, A)$; here $\lambda_{t,-i}$ is the **current state** of $\lambda_{-i}$ after step $(i-1)$ of iteration $t$ is finished, so that components 1 through $(i-1)$ have been updated but **not the rest**.
**Gibbs Sampling**

The acceptance probability for the proposed move to $\lambda_i^*$ that creates the correct equilibrium distribution turns out to be

$$\alpha_{MH}(\lambda_i^* | \lambda_{t,-i}, \lambda_{t,i}, y) = \min \left[ 1, \frac{p(\lambda_i^* | \lambda_{t,-i}, y) g_i(\lambda_{t,i} | \lambda_i^*, \lambda_{t,-i}, y)}{p(\lambda_{t,i} | \lambda_{t,-i}, y) g_i(\lambda_i^* | \lambda_{t,i}, \lambda_{t,-i}, y)} \right].$$  \hspace{1cm} (19)

The distribution $p(\lambda_i | \lambda_{-i}, y)$ appearing in (19), which is called the full conditional distribution for $\lambda_i$, has a natural interpretation: it represents the posterior distribution for the relevant portion of $(\theta, A)$ given $y$ and the rest of $(\theta, A)$.

The full conditional distributions act like building blocks in constructing the complete posterior distribution $p(\theta | y)$, in the sense that any multivariate distribution is uniquely determined by its set of full conditionals (Besag (1974)).

An important special case of single-component MH sampling arises when the proposal distribution $g_i(\lambda_i^* | \lambda_{t,i}, \lambda_{t,-i}, y)$ for component $i$ is chosen to be the full conditional $p(\lambda_i^* | \lambda_{t,-i}, y)$ for $\lambda_i$: you can see from (19) that when this choice is made a glorious cancellation occurs and the acceptance probability is 1.

This is Gibbs sampling, independently (re)discovered by Geman and Geman (1984): the Gibbs recipe is to sample from the full conditionals and accept all proposed moves.

Even though it’s just a version of MH, Gibbs sampling is important enough to merit a summary of its own.

**Single-element** Gibbs sampling, in which each real-valued coordinate $(\theta_1, \ldots, \theta_k)$ gets updated in turn, is probably the most frequent way Gibbs sampling gets used, so that’s what I’ll summarize ((20) details Gibbs sampling in the case with no auxiliary variables $A$, but the algorithm works equally well when $\theta$ is replaced by $(\theta, A)$ in the summary).
Gibbs Sampling (continued)

Algorithm (Single-element Gibbs sampling). To construct a Markov chain whose equilibrium distribution is \( p(\theta|y) \) with \( \theta = (\theta_1, \ldots, \theta_k) \),

\[
\text{Initialize } \theta^*, 1, \ldots, \theta^*_{0,k}; \quad t \leftarrow 0 \\
\text{Repeat } \{ \\
\quad \text{Sample } \theta^*_{t+1,1} \sim p(\theta_1|y, \theta^*_{t,2}, \theta^*_{t,3}, \theta^*_{t,4}, \ldots, \theta^*_{t,k}) \\
\quad \text{Sample } \theta^*_{t+1,2} \sim p(\theta_2|y, \theta^*_{t+1,1}, \theta^*_{t,3}, \theta^*_{t,4}, \ldots, \theta^*_{t,k}) \\
\quad \text{Sample } \theta^*_{t+1,3} \sim p(\theta_3|y, \theta^*_{t+1,1}, \theta^*_{t+1,2}, \theta^*_{t,4}, \ldots, \theta^*_{t,k}) \\
\quad \vdots \\
\quad \text{Sample } \theta^*_{t+1,k} \sim p(\theta_k|y, \theta^*_{t+1,1}, \theta^*_{t+1,2}, \theta^*_{t+1,3}, \ldots, \theta^*_{t+1,k-1}) \\
\quad t \leftarrow (t + 1) \\
\} 
\]

(20)

Example: the NB10 Data. Recall from the posterior predictive plot toward the end of part 2 of the lecture notes that the Gaussian model for the NB10 data was inadequate: the tails of the data distribution are too heavy for the Gaussian.

It was also clear from the normal qqplot that the data are symmetric.

This suggests thinking of the NB10 data values \( y_i \) as like draws from a \( t \) distribution with fairly small degrees of freedom \( \nu \).

One way to write this model is

\[
(\mu, \sigma^2, \nu) \sim p(\mu, \sigma^2, \nu) \\
(y_i|\mu, \sigma^2, \nu) \overset{\text{IID}}{\sim} t_\nu(\mu, \sigma^2),
\]

(21)

where \( t_\nu(\mu, \sigma^2) \) denotes the scaled \( t \)-distribution with mean \( \mu \), scale parameter \( \sigma^2 \), and shape parameter \( \nu \).
Model Expansion

This distribution has variance \( \sigma^2 \left( \frac{\nu}{\nu-2} \right) \) for \( \nu > 2 \) (so that shape and scale are mixed up, or \textbf{confounded} in \( t_\nu(\mu, \sigma^2) \)) and may be thought of as the distribution of the quantity \( \mu + \sigma e \), where \( e \) is a draw from the \textbf{standard} \( t \) distribution that is tailed at the back of all introductory statistics books.

However, a \textbf{better way} to think about model (21) is as follows.

It’s a fact from \textbf{basic distribution theory}, probably of more interest to Bayesians than frequentists, that the \( t \) distribution is an \textbf{Inverse Gamma mixture of Gaussians}.

This just means that to generate a \( t \) random quantity you can first draw from an Inverse Gamma distribution and then draw from a Gaussian \textbf{conditional} on what you got from the Inverse Gamma.

(As noted in \textbf{homework 2}, \( \lambda \sim \Gamma^{-1}(\alpha, \beta) \) just means that \( \lambda^{-1} = \frac{1}{\lambda} \sim \Gamma(\alpha, \beta) \)).

In more detail, \((y|\mu, \sigma^2, \nu) \sim t_\nu(\mu, \sigma^2)\) is the same as the \textbf{hierarchical model}

\[
(l|\nu) \sim \Gamma^{-1}\left(\frac{\nu}{2}, \frac{\nu}{2}\right) \\
(y|\mu, \sigma^2, \lambda) \sim N\left(\mu, \lambda \sigma^2\right).
\]  

Putting this together with the \textbf{conjugate prior} for \( \mu \) and \( \sigma^2 \) we looked at earlier in the Gaussian model gives the following HM for the NB10 data:

\[
\nu \sim p(\nu) \\
\sigma^2 \sim \text{SI-}\chi^2(\nu_0, \sigma_0^2) \\
(\mu|\sigma^2) \sim N\left(\mu_0, \frac{\sigma^2}{\kappa_0}\right) \\
(\lambda_i|\nu) \overset{\text{IID}}{\sim} \Gamma^{-1}\left(\frac{\nu}{2}, \frac{\nu}{2}\right) \\
(y_i|\mu, \sigma^2, \lambda_i) \overset{\text{indep}}{\sim} N\left(\mu, \lambda_i \sigma^2\right).
\]

Remembering also from introductory statistics that the Gaussian distribution is the \textbf{limit} of the \( t \) family as \( \nu \to \infty \), you can see that the idea here has been to \textbf{expand} the Gaussian model by embedding it in the richer \( t \) family, of which it’s a special case with \( \nu = \infty \).
Implementing Gibbs

Model expansion is often the best way to deal with **uncertainty in the modeling process**: when you find deficiencies of the current model, **embed it in a richer class**, with the model expansion in directions suggested by the deficiencies (we'll also see this method in action again later).

**The MCMC Dataset.** Imagine trying to do **Gibbs sampling** on model (21), with the parameter vector \( \theta = (\mu, \sigma^2, \nu) \).

Carrying out the iterative program described in (20) above would produce the following **MCMC Dataset**:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Phase</th>
<th>( \mu )</th>
<th>( \sigma^2 )</th>
<th>( \nu )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initializing</td>
<td>( \mu_0 )</td>
<td>( \sigma^2_0 )</td>
<td>( \nu_0 )</td>
</tr>
<tr>
<td>1</td>
<td>Burn-In</td>
<td>( \mu_1(y, \sigma^2_0, \nu_0) )</td>
<td>( \sigma^2_1(y, \mu_1, \nu_1) )</td>
<td>( \nu_1(y, \mu_1, \sigma^2_1) )</td>
</tr>
<tr>
<td>2</td>
<td>Burn-In</td>
<td>( \mu_2(y, \sigma^2_1, \nu_1) )</td>
<td>( \sigma^2_2(y, \mu_2, \nu_1) )</td>
<td>( \nu_1(y, \mu_2, \sigma^2_2) )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>b</td>
<td>Burn-In</td>
<td>( \mu_b )</td>
<td>( \sigma^2_b )</td>
<td>( \nu_b )</td>
</tr>
<tr>
<td>(b + 1)</td>
<td>Monitoring</td>
<td>( \mu_{b+1} )</td>
<td>( \sigma^2_{b+1} )</td>
<td>( \nu_{b+1} )</td>
</tr>
<tr>
<td>(b + 2)</td>
<td>Monitoring</td>
<td>( \mu_{b+2} )</td>
<td>( \sigma^2_{b+2} )</td>
<td>( \nu_{b+2} )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>(b + m)</td>
<td>Monitoring</td>
<td>( \mu_{b+m} )</td>
<td>( \sigma^2_{b+m} )</td>
<td>( \nu_{b+m} )</td>
</tr>
</tbody>
</table>

Looking at iterations 1 and 2 you can see that, in addition to \( y \), the sampler makes use **only of parameter values in the current row and the previous row** (this illustrates the Markov character of the samples).

As we've seen above, at the end of the \((b + m)\) iterations, if you want (say) the **marginal posterior** for \( \mu \), \( p(\mu | y) \), all you have to do is take the \( m \) values \( \mu_{b+1}, \ldots, \mu_{b+m} \) and summarize them in any ways that interest you: their sample mean is your **simulation estimate** of the posterior mean of \( \mu \), their sample histogram (or, better, their **kernel density trace**) is your simulation estimate of \( p(\mu | y) \), and so on.


Practical Issues

Implementation Details. (1) How do you figure out the full conditionals, and how do you sample from them?

(2) What should you use for initial values?

(3) How large should \( b \) and \( m \) be?

(4) More generally, how do you know when the chain has reached equilibrium?

Questions (3–4) fall under the heading of **MCMC diagnostics**, which I’ll cover a bit later, and I’ll address question (2) in the case studies below.

Computing the full conditionals. For a simple example of working out the full conditional distributions, consider the conjugate Gaussian model we looked at earlier:

\[
\begin{align*}
\sigma^2 & \sim \text{SI-} \chi^2(\nu_0, \sigma_0^2) \\
(\mu|\sigma^2) & \sim N\left(\mu_0, \frac{\sigma^2}{\kappa_0}\right) \\
(Y_i|\mu, \sigma^2) & \overset{\text{IID}}{\sim} N(\mu, \sigma^2). 
\end{align*}
\]  

(24)

The full conditional distribution for \( \mu \) in this model is \( p(\mu|\sigma^2, y) \), considered as a function of \( \mu \) for fixed \( \sigma^2 \) and \( y \)—but this is just

\[
\begin{align*}
p(\mu|\sigma^2, y) &= \frac{p(\mu, \sigma^2, y)}{p(\sigma^2, y)} \\
&= \frac{c p(\mu, \sigma^2, y)}{c p(\sigma^2) p(\mu|\sigma^2) p(y|\mu, \sigma^2)} \\
&= c \exp\left[-\frac{\kappa_0}{2\sigma^2} (\mu - \mu_0)^2\right] \prod_{i=1}^n \exp\left[-\frac{1}{2\sigma^2} (y_i - \mu)^2\right]. 
\end{align*}
\]  

(25)
Full Conditionals

From this

\[ p(\mu|\sigma^2, y) = c \exp \left[ -\frac{\kappa_0}{2\sigma^2}(\mu - \mu_0)^2 \right] \exp \left[ -\frac{1}{2\sigma^2} \sum_{i=1}^{n}(y_i - \mu)^2 \right]. \]

Expanding out the squares, collecting powers of \( \mu \), and completing the square in \( \mu \) gives

\[ p(\mu|\sigma^2, y) = c \exp \left[ -\frac{\kappa_0 + n}{2\sigma^2} \left( \mu - \frac{\kappa_0 \mu_0 + n\bar{y}}{\kappa_0 + n} \right)^2 \right], \quad (26) \]

from which it’s clear that the full conditional for \( \mu \) in model (24) is

\[ (\mu|\sigma^2, y) \sim N \left( \frac{\kappa_0 \mu_0 + n\bar{y}}{\kappa_0 + n}, \frac{\sigma^2}{\kappa_0 + n} \right). \quad (27) \]

Similarly, the full conditional for \( \sigma^2 \) in this model, \( p(\sigma^2|\mu, y) \), considered as a function of \( \sigma^2 \) for fixed \( \mu \) and \( y \), is just

\[
\begin{align*}
  p(\sigma^2|\mu, y) &= \frac{p(\sigma^2, \mu, y)}{p(\mu, y)} \\
  &= \frac{c p(\sigma^2, \mu, y)}{c p(\mu|\sigma^2) p(y|\mu, \sigma^2)} \\
  &= c p(\sigma^2) p(\mu|\sigma^2) p(y|\mu, \sigma^2) \\
  &= c \left( \sigma^2 \right)^{-\left(1+\frac{1}{2}\nu_0\right)} \exp \left( \frac{-\nu_0 \sigma_0^2}{2\sigma^2} \right) \cdot \left( \sigma^2 \right)^{-\frac{1}{2}} \exp \left[ -\frac{\kappa_0}{2\sigma^2}(\mu - \mu_0)^2 \right] \cdot \left( \sigma^2 \right)^{-\frac{1}{2}} \exp \left[ -\frac{1}{2\sigma^2} \sum_{i=1}^{n}(y_i - \mu)^2 \right].
\end{align*}
\]

When this is simplified you get
Full Conditionals (continued)

\[ p(\sigma^2|\mu, y) = c \left( \sigma^2 \right)^{-\left( 1 + \frac{\nu_0 + 1 + n}{2} \right)} \exp \left[ -\frac{\nu_0 \sigma_0^2 + \kappa_0 (\mu - \mu_0)^2 + ns_\mu^2}{2\sigma^2} \right], \]

where \( s_\mu^2 = \frac{1}{n} \sum_{i=1}^{n} (y_i - \mu)^2 \).

From the form of this distribution it becomes clear that

\[ (\sigma^2|\mu, y) \sim \text{SI-}\chi^2 \left( \nu_0 + 1 + n, \frac{\nu_0 \sigma_0^2 + \kappa_0 (\mu - \mu_0)^2 + ns_\mu^2}{\nu_0 + 1 + n} \right). \]

Thus in conjugate situations the full conditional distributions have conjugate forms, which are tedious but straightforward to compute.

Both the directness and the tedium of this calculation suggest that it should be possible to write a computer program to work out the full conditionals for you, and indeed at least two such programs now exist:

- **BUGS**, a fairly general-purpose Gibbs sampling program produced by David Spiegelhalter and others at the MRC Biostatistics Unit in Cambridge, UK (Spiegelhalter et al., 1997), and

- **MLwiN**, a program that does both maximum-likelihood and Bayesian calculations in hierarchical (multilevel) models (Rasbash et al. 2000).

BUGS runs under Unix or DOS in a wide variety of hardware configurations, and a Windows version called WinBUGS is also available; we’ll look here at both Unix BUGS and WinBUGS (together with MLwiN if there’s time).

BUGS and WinBUGS are available for free downloading at

[www.mrc-bsu.cam.ac.uk/bugs](http://www.mrc-bsu.cam.ac.uk/bugs);

MLwiN has a nominal charge and can be downloaded from the web page of the Multilevel Models Project,

[multilevel.ioe.ac.uk](http://multilevel.ioe.ac.uk)
Why the Metropolis Algorithm Works

Here’s a sketch of the crucial part of the proof, based on an argument in Gamerman (1997), of the validity of the Metropolis algorithm, in the case of a discrete (finite or countably infinite) state space \( S \) (see chapter 1 in Gilks et al. 1996 for a proof sketch when \( S \) is continuous).

It will be helpful in looking at the proof sketch to specialize the \textbf{Markov chain notation} we’ve been using so far to the case of discrete state spaces, as follows.

A \textbf{stochastic process} \( \{\theta^*_t, t \in T\}, T = \{0,1,\ldots\} \) on a discrete state space \( S \) is a \textbf{Markov chain} iff

\[
P(\theta^*_{t+1} = y | \theta^*_t = x, \theta^*_{t-1} = x_{n-1}, \ldots, \theta^*_0 = x_0) = P(\theta^*_{t+1} = y | \theta^*_t = x)
\]

for all \( t = 0,1,\ldots \) and \( x_0,\ldots,x_{t-1},x,y \in S \).

In general \( P(\theta^*_{t+1} = y | \theta^*_t = x) \) depends on \( x,y \), and \( t \), but if the probability of transitioning from \( x \) to \( y \) at time \( t \) is \textbf{constant} in \( t \) things will clearly be simpler; such chains are called \textbf{homogeneous} (confusingly, some sources call them \textbf{stationary}, but that terminology seems well worth avoiding).

The \textbf{random walk} described earlier is obviously a homogeneous Markov chain, and so are any Markov chains generated by the \textbf{MH algorithm}; I’ll \textbf{assume homogeneity} in what follows.

Under \textbf{homogeneity} it makes sense to talk about the \textbf{transition probability}

\[
P(x,y) = P(\theta^*_{t+1} = y | \theta^*_t = x) \quad \text{for all } t,
\]

which satisfies

\[
P(x,y) \geq 0 \text{ for all } x,y \in S \quad \text{and} \quad \sum_{y \in S} P(x,y) = 1 \text{ for all } x \in S.
\]
Metropolis Proof Sketch

When $S$ is discrete a **transition matrix** $P$ can be defined with element $(i,j)$ given by $P(x_i,x_j)$, where $x_i$ is the $i$th element in $S$ according to whatever **numbering convention** you want to use (the second part of (32) implies that the row sums of such a matrix are always 1; this is the defining condition for a **stochastic matrix**).

Suppose the chain is **initialized** at time 0 by making a **draw** from a probability distribution $\pi_0(x) = P(\theta_0^* = x)$ on $S$ (deterministically starting it at some point $x_0$ is a special case of this); then the probability distribution $\pi_1(y)$ for where it will be at time 1 is

$$
\pi_1(y) = P(\theta_1^* = y) = \sum_{x \in S} P(\theta_0^* = x, \theta_1^* = y) = \sum_{x \in S} P(\theta_0^* = x) P(\theta_1^* = y | \theta_0^* = x) \quad (33) 
$$

which can be written in **vector** and **matrix** notation as

$$
\pi_1 = \pi_0 P, \quad (34)
$$

where $\pi_0$ and $\pi_1$ are regarded as **row vectors**.

Then by the **same reasoning**

$$
\pi_2 = \pi_1 P = (\pi_0 P) P = \pi_0 P^2, \quad (35)
$$

and **in general**

$$
\pi_t = \pi_0 P^t. \quad (36)
$$

For **simple** Markov chains this can be used to work out the **long-run** behavior of the chain as $t \to \infty$, but this becomes **algebraically prohibitive** as the **transition behavior** of the chain increases in **complexity**.