Lecture 17 – Part 2
Bayesian Econometrics

Numerical Methods

• Q: Do we need to restrict our choices of prior distributions to these conjugate families? No. The posterior distributions are well defined irrespective of conjugacy. Conjugacy only simplifies computations.

• Outside the conjugate families, we rely on numerical methods for calculating posterior moments. In these cases, P($\theta \,|\, y$) is not recognized as a (analytical) pdf where we can sample from using a standard library function.

• Another situation where analytical posteriors are difficult to obtain is when the model is no longer linear.

• What do we do in these situations? We simulate the behavior of the model.

Numerical Methods: Simulation

- It is possible to do Bayesian estimation and inference over parameters in these situations, for example, with a nonlinear model.

- Steps:

1. Parameterize the model
2. Propose the likelihood conditioned on the parameters
3. Propose the priors – joint prior for all model parameters
4. As usual, the posterior is proportional to likelihood times prior. (Usually, it requires conjugate priors to be tractable. But, very likely, complicated P($\theta|y$).)
5. Sample –i.e., draw observations- from the posterior to study its characteristics.

Q: How do we draw? **MCMC**.

Numerical Methods: MCMC

- Sampling from the joint posterior P($\theta|y$) may be difficult or impossible. For example, in the CLM, assuming a normal prior for β, and an inverse-gamma prior for σ^2, we get a complicated joint posterior distribution for (β, σ^2).

- To do simulation based estimation, we need joint draws on (β, σ^2). But, if P($\theta|y$) is complicated ⇒ we cannot easily draw from it.

- For these situations, many methods have been developed that make the process easier, including **Gibbs sampling**, **Data Augmentation**, and the **Metropolis-Hastings (MH) algorithm**.

- All three are examples of **Markov Chain-Monte Carlo** (MCMC) methods.

Numerical Methods: MCMC Preliminaries

- **Monte Carlo** (first MC): A simulation. We take quantities of interest of a distribution from simulated draws from the distribution.

Example: Monte Carlo integration

We have a posterior distribution P($\theta \mid y$) that we want to take quantities of interest from. We can evaluate the integral analytically, *I*:

$$I = \int h(\theta)\ \mathrm{P}(\theta)\ d\theta$$

where $h(\theta)$ is some function of θ.

But when P($\theta \mid y$) is complicated, we approximate the integral via MC Integration using the **plug-in estimator**, obtained by simulating M values from P($\theta \mid y$) and calculating:

$$I_M = \sum_{i=1}^{M} h(\theta_i)/M$$

Numerical Methods: MCMC Preliminaries

- From, the LLN, the MC approximation I_M is a consistent (simulation) estimator of the true value *I*. That is, $I_M \rightarrow I$ as $M \rightarrow \infty$.

Q: But, to apply the LLN we need independence. What happens if we cannot generate independent draws?

- Suppose we want to draw from our posterior P($\theta \mid y$), but we cannot sample independent draws from it. But, we may be able to sample draws from P($\theta \mid y$) that are "*slightly*" dependent.

If we can sample slightly dependent draws using a **Markov chain**, then we can still find quantities of interests from those draws.

Numerical Methods: MCMC Preliminaries

• Monte Carlo (first MC): A simulation.

• **Markov Chain** (the second MC): A stochastic process in which future states are independent of past states given the present state.

• Recall that a stochastic process is a consecutive set of random quantities defined on some known state space, Θ.

- Θ: our parameter space

- Consecutive implies a time component, indexed by t.

• A draw θ^t describes the state at time (iteration) *t*. The next draw θ^{t+1} is dependent *only* on θ^t. This is because of the **Markov property**:

$$P(\theta^{t+1} | \theta^t) = P(\theta^{t+1} | \theta^t, \theta^{t-1}, \theta^{t-2}, \dots, \theta^1)$$

Numerical Methods: MCMC Preliminaries

• The state of a Markov chain (MC) is a random variable indexed by t, say, θ^t. The state distribution is the distribution of θ^t: $p_t(\theta)$.

A stationary distribution of the chain is a distribution π such that, if

$$p_t(\theta) = \pi \qquad \Rightarrow p_{t+s}(\theta) = \pi \qquad \text{for all } s.$$

• Under "*certain conditions*" a chain will have the following properties:

– A unique stationary distribution.

– Converge to that stationary distribution π as $t \to \infty$.

– Ergodic. That is, averages of successive realizations of θ will converge to their expectations with respect to π.

A lot of research has been devoted to establish the *certain conditions*.

MCMC – Ergodicity (P. Lam)

• Usual "certain conditions" for ergodicity:

The Markov chain is **aperiodic**, **irreducible** (it is possible to go from any state to any other state), and **positive recurrent** (eventually, we expect to return to a state in a finite amount of time).

Ergodic Theorem

• Let $\theta^{(M)}=\{\theta^1, \theta^2, \theta^3, \ldots, \theta^M\}$ be *M* values from a Markov chain that is aperiodic, irreducible, and positive recurrent –i.e., chain is ergodic–, and E$[g(\theta)] < \infty$. Then, with probability 1:

$$\sum_{i=1}^{M} g(\theta_i)/M \rightarrow \int_\Theta g(\theta)\,\pi(\theta)\,d\theta$$

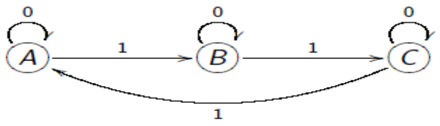
This is the Markov chain analog to the SLLN. It allows us to ignore the dependence between draws of the Markov chain when we calculate quantities of interest from the draws (like MC Integration).

MCMC - Ergodicity (P. Lam)

- **Aperiodicity**

A Markov chain is aperiodic if the only length of time for which the chain repeats some cycle of values is the trivial case with cycle length equal to one.

Let A, B, and C denote the states (analogous to the possible values of θ) in a 3-state Markov chain. The following chain is periodic with period 3, where the period is the number of steps that it takes to return to a certain state.



As long as the chain is not repeating an identical cycle, then the chain is aperiodic.

MCMC - Irreducibility and Stationarity

- **Irreducibility**

A Markov chain is irreducible if there no absorbing states or states in which the chain gets trapped.

- **Stationarity** (Theorem)

If $p_{ij} > 0$ (strictly positive) for all i, j, then the chain is irreducible and there exists a *stationary distribution*, π, such that

$$\lim_{t \to \infty} \pi_0 \mathbf{P}^t = \pi$$

and

$$\pi \, \mathbf{P} = \pi.$$

Since the elements of **P** are all positive and each row sums to one, the maximum eigenvalue of $\mathbf{P}^T$ is 1 and π is determined by the corresponding eigenvector, $\mathbf{R}_1$, and the corresponding row vector from the inverse of the matrix for eigenvectors, $\mathbf{R}^{-1}$.

MCMC - Irreducibility and Stationarity

Proof: Using an eigenvalue decomposition:

$$\mathbf{P} = \mathbf{R} \, \boldsymbol{\Lambda} \, \mathbf{R}^{-1}$$

where **R** is a matrix of eigenvectors and **Λ** is a diagonal matrix of corresponding eigenvalues, λ.

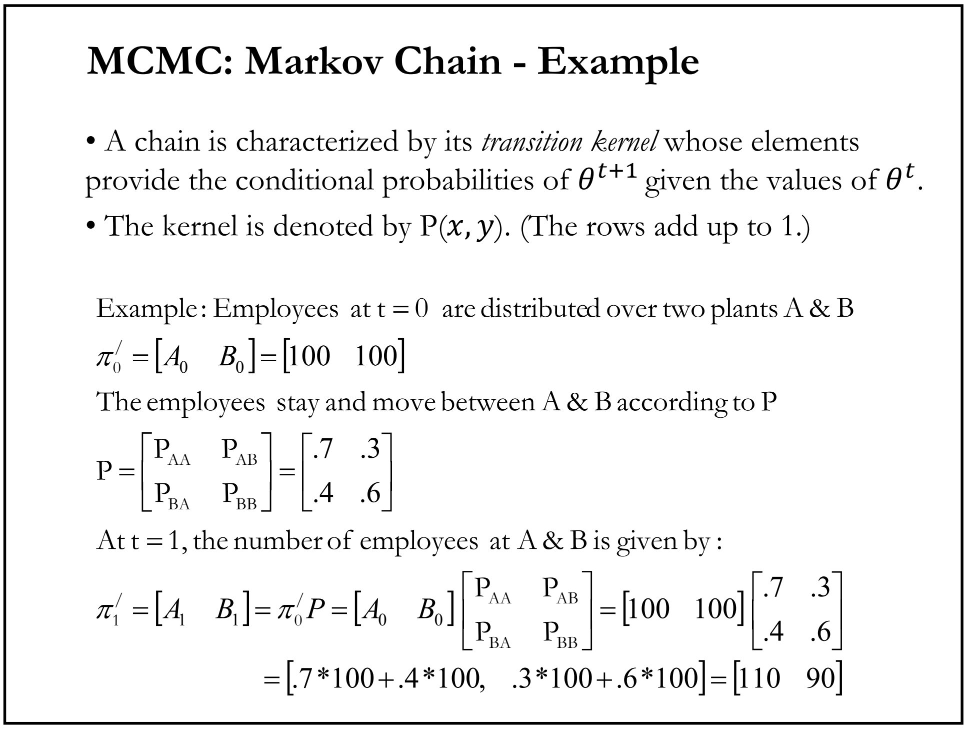
Recall: $(\mathbf{P}^T)^m = \mathbf{R} \, \boldsymbol{\Lambda}^m \, \mathbf{R}^{-1}$, since

$$(\mathbf{P}^T)^m = \mathbf{R} \, \boldsymbol{\Lambda} \, \mathbf{R}^{-1} \ldots \mathbf{R} \, \boldsymbol{\Lambda} \, \mathbf{R}^{-1} = \mathbf{R} \, \boldsymbol{\Lambda}^m \, \mathbf{R}^{-1}.$$

• Then, the long-run steady-state is determined by $\max\{\lambda(\mathbf{P}^T)\}$ and in the direction of the corresponding vector from $\mathbf{R}^{-1}$(if the remaining λ's < 1, then $\lambda^m \to 0$ and their corresponding inverse eigenvectors' influence on direction dies out).

Since $\max\{\lambda(\mathbf{P}^T)\}$=1, with a large M $\Rightarrow \pi_0 \, \mathrm{P}^t \to \pi_0 \times 1 = \pi.$

That is, after many iterations the Markov chain produces draws from a stationary distribution if the chain is irreducible.



MCMC: Markov Chain - Example

- A chain is characterized by its *transition kernel* whose elements provide the conditional probabilities of θ^{t+1} given the values of θ^{t}.
- The kernel is denoted by P(x, y). (The rows add up to 1.)

Example: Employees at t = 0 are distributed over two plants A & B

$$\pi_0^{/} = [A_0 \quad B_0] = [100 \quad 100]$$

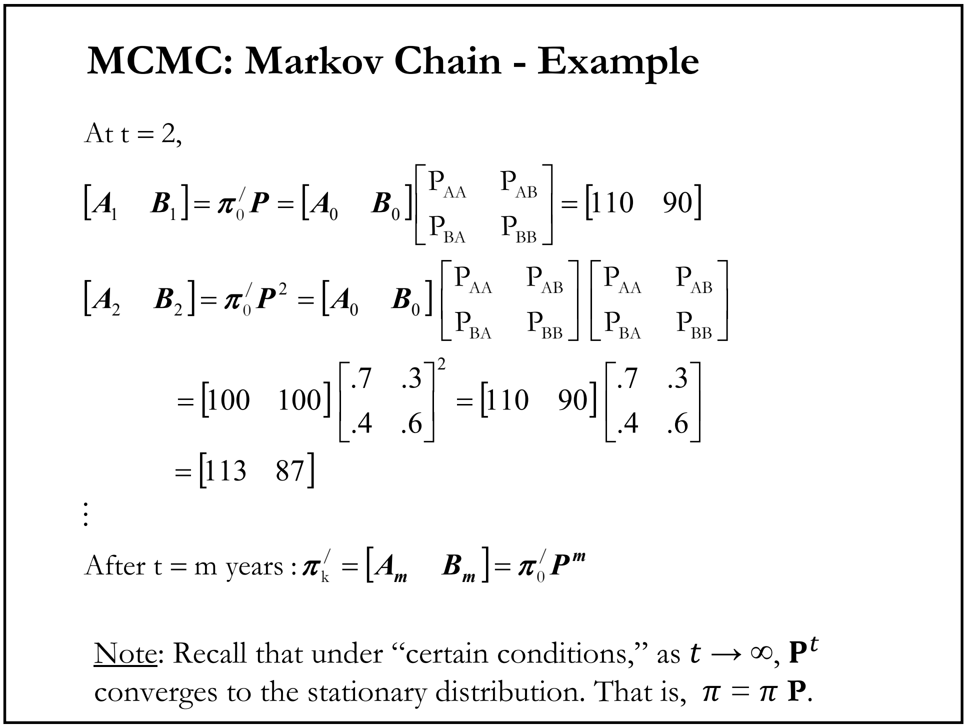
The employees stay and move between A & B according to P

$$\mathrm{P} = \begin{bmatrix} \mathrm{P}_{\mathrm{AA}} & \mathrm{P}_{\mathrm{AB}} \\ \mathrm{P}_{\mathrm{BA}} & \mathrm{P}_{\mathrm{BB}} \end{bmatrix} = \begin{bmatrix} .7 & .3 \\ .4 & .6 \end{bmatrix}$$

At t = 1, the number of employees at A & B is given by:

$$\pi_1^{/} = [A_1 \quad B_1] = \pi_0^{/} P = [A_0 \quad B_0] \begin{bmatrix} \mathrm{P}_{\mathrm{AA}} & \mathrm{P}_{\mathrm{AB}} \\ \mathrm{P}_{\mathrm{BA}} & \mathrm{P}_{\mathrm{BB}} \end{bmatrix} = [100 \quad 100] \begin{bmatrix} .7 & .3 \\ .4 & .6 \end{bmatrix}$$

$$= [.7*100 + .4*100, \quad .3*100 + .6*100] = [110 \quad 90]$$



MCMC: Markov Chain - Example

At t = 2,

$$[\boldsymbol{A}_1 \quad \boldsymbol{B}_1] = \boldsymbol{\pi}_0^{/} \boldsymbol{P} = [\boldsymbol{A}_0 \quad \boldsymbol{B}_0] \begin{bmatrix} \mathrm{P}_{\mathrm{AA}} & \mathrm{P}_{\mathrm{AB}} \\ \mathrm{P}_{\mathrm{BA}} & \mathrm{P}_{\mathrm{BB}} \end{bmatrix} = [110 \quad 90]$$

$$[\boldsymbol{A}_2 \quad \boldsymbol{B}_2] = \boldsymbol{\pi}_0^{/} \boldsymbol{P}^2 = [\boldsymbol{A}_0 \quad \boldsymbol{B}_0] \begin{bmatrix} \mathrm{P}_{\mathrm{AA}} & \mathrm{P}_{\mathrm{AB}} \\ \mathrm{P}_{\mathrm{BA}} & \mathrm{P}_{\mathrm{BB}} \end{bmatrix} \begin{bmatrix} \mathrm{P}_{\mathrm{AA}} & \mathrm{P}_{\mathrm{AB}} \\ \mathrm{P}_{\mathrm{BA}} & \mathrm{P}_{\mathrm{BB}} \end{bmatrix}$$

$$= [100 \quad 100] \begin{bmatrix} .7 & .3 \\ .4 & .6 \end{bmatrix}^2 = [110 \quad 90] \begin{bmatrix} .7 & .3 \\ .4 & .6 \end{bmatrix}$$

$$= [113 \quad 87]$$

$$\vdots$$

After t = m years: $\boldsymbol{\pi}_{\mathrm{k}}^{/} = [\boldsymbol{A}_m \quad \boldsymbol{B}_m] = \boldsymbol{\pi}_0^{/} \boldsymbol{P}^{\boldsymbol{m}}$

<u>Note</u>: Recall that under "certain conditions," as $t \rightarrow \infty$, $\mathbf{P}^{t}$ converges to the stationary distribution. That is, $\pi = \pi\ \mathbf{P}$.

MCMC: General Idea

• We construct a chain, or sequence of values, $\theta^0, \theta^1, \ldots,$ such that for large m, θ^m can be viewed as a draw from the posterior distribution of θ, $P(\theta|\boldsymbol{X})$, given the data $\boldsymbol{X} = \{X_1, \ldots, X_N\}$.

• This is implemented through an algorithm that, given a current value of the parameter vector θ^m, and given $\boldsymbol{X}$, draws a new value θ^{m+1} from a distribution $f(.)$ indexed by θ^m and the data:

$$\theta^{m+1} \sim f(\theta\,|\theta^m, \boldsymbol{X})$$

• We do this in a way that if the original θ^m came from the posterior distribution, then so does θ^{m+1}. That is,

$$\theta^m\,|\boldsymbol{X} \sim P(\theta|\boldsymbol{X}), \qquad \Rightarrow \; \theta^{m+1}\,|\boldsymbol{X} \sim P(\theta|\boldsymbol{X}).$$

MCMC: General Idea

• In many cases, irrespective of where we start −irrespective of θ^0− as $m \to \infty$, it will be the case that the distribution of the parameter conditional only on the data, $\boldsymbol{X}$, converges to the posterior distribution as $m \to \infty$:

$$\theta^m\,|\boldsymbol{X} \xrightarrow{d} p(\theta\,|\boldsymbol{X}),$$

• Then just pick a θ^0 and approximate the mean and standard deviation of the posterior distribution as:

$$\text{E}[\theta\,|\boldsymbol{X}] = 1/(M - M* + 1)\sum_{m=M*}^{M}\theta^m$$

$$\text{Var}(\theta\,|\boldsymbol{X}) = 1/(M - M* + 1)\sum_{m=M*}^{M}\{\theta^m - \text{E}(\theta\,|\boldsymbol{X})\}^2$$

• Usually, the first $M* - 1$ iterations are discarded to let the algorithm converge to the stationary distribution without the influence of starting values, θ^0 (*burn-in*).

MCMC: Burn-in (P. Lam)

• As a matter of practice, most people throw out a certain number of the first draws, $\theta^1, \theta^2, \theta^3, \dots, \theta^{M*}$, known as the *burn-in*. This is to make our draws closer to the stationary distribution and less dependent on the starting point.

• Think of it as a method to pick initial values.

• However, it is unclear how much we should burn-in since our draws are all slightly dependent and we do not know exactly when convergence occurs.

• Not a lot of theory about it.

MCMC: Thinning the Chain (P. Lam)

In order to break the dependence between draws in the Markov chain, some have suggested only keeping every d^{th} draw of the chain. That is, we keep $\theta^{(d)} = \{\theta^{1d}, \theta^{2d}, \theta^{3d}, \dots, \theta^{Md}\}$

• This is known as **thinning**.

- **Pros:**
  - We may get a little closer to *i.i.d.* draws.
  - It saves memory since you only store a fraction of the draws.
- **Cons:**
  - It is unnecessary with ergodic theorem.
  - Shown to increase the variance of your MC estimates.

MCMC - Remarks

- In classical stats, we usually focus on finding the stationary distribution, given a Markov chain.

- MCMC methods turn the theory around: The invariant density is known (maybe up to a constant multiple) –it is the target density, $\pi(.)$, from which samples are desired–, but the transition kernel is unknown.

- To generate samples from $\pi(.)$, the methods find and utilize a transition kernel P(x, y), whose M^{th} iteration converges to π(.) for large M.

MCMC - Remarks

- The process is started at an arbitrary x and iterated a large number of times. After this, the distribution of the observations generated from the simulation is approximately the target distribution.

- Then, the problem is to find an appropriate P(x, y) that works!

- Once we have a Markov chain that has converged to the stationary distribution, then the draws in our chain appear to be like draws from P(θ|y), so it seems like we should be able to use Monte Carlo Integration methods to find quantities of interest.

- Our draws are not independent, which we required for MC Integration to work (remember LLN). For dependent draws, we rely on the Ergodic Theorem.

MCMC - Remarks

• Our draws are dependent, the autocorrelation in the chain can be a problem for the MCMC estimators.

• Compared to MC estimators (MC simulations are the *"gold standard"*, since the draws are independent), MCMC estimators tend to have a higher variance and, then, worse approximations.

MCMC: Gibbs Sampling

• When we sample directly from the *conditional posterior distributions*, the algorithm is known as **Gibbs Sampling** (**GS**).

• The Gibbs sampler partitions the vector of parameters θ into two (or more) blocks or parts, say $\theta = (\theta_1, \theta_2, \theta_3)$. Instead of sampling θ^{m+1} directly from the (known) joint conditional distribution of

$$f(\theta \,|\, \theta^m; \mathbf{X}),$$

it may be easier to sample θ from the (known) *full conditional distributions*, $P(\theta_j \,|\, \theta^m_{-j}; \mathbf{X})$: $(\theta^m_{-j} = \theta^m_i, \theta^m_l)$

- first sample θ_1^{m+1} from $P(\theta_1 \,|\, \theta_2^m, \theta_3^m; \mathbf{X})$.
- then sample θ_2^{m+1} from $P(\theta_2 \,|\, \theta_1^{m+1}, \theta_3^m; \mathbf{X})$.
- then sample θ_3^{m+1} from $P(\theta_3 \,|\, \theta_1^{m+1}, \theta_2^{m+1}; \mathbf{X})$.

• Remark: If θ^m is from the posterior distribution, then so is θ^{m+1}.

MCMC: Gibbs Sampling (P. Lam)

- Q: How can we know the joint distribution simply by knowing the full conditional distributions?

A: The ***Hammersley-Clifford Theorem*** shows that we can write the joint density, P(x, y), in terms of the conditionals P($x \mid y$) and P($y \mid x$).

- Then, how do we figure out the full conditionals?

Suppose we have a posterior P($\theta \mid y$). To calculate the full conditionals for each θ, do the following:

1. Write out the full posterior ignoring constants of proportionality.

2. Pick a block of parameters (say, θ_1). Drop everything that does not depend on θ_1 (we call this block θ_{-1}).

3. Figure out the normalizing constant (and, thus, the full conditional distribution P($\theta_1 \mid \theta_{-1}, y$)).

4. Repeat steps 2 and 3 for all parameters.

MCMC: Gibbs Sampling - Steps

Example: In the previous MVN model, we derived the full conditional posteriors for **μ** & **Σ**:

$$\boldsymbol{p}(\boldsymbol{\mu} \mid \boldsymbol{y}_1,\dots,\boldsymbol{y}_N,\Sigma) \propto \boldsymbol{e}^{-\frac{1}{2}\boldsymbol{\mu}'\boldsymbol{A}_N\boldsymbol{\mu}+\boldsymbol{\mu}'\boldsymbol{b}_N}$$

$$\boldsymbol{p}(\Sigma \mid \boldsymbol{y}_1,\dots,\boldsymbol{y}_N,\boldsymbol{\mu}) \propto |\Sigma|^{-\frac{1}{2}(N+\boldsymbol{v}_0+\boldsymbol{k}+1)}\ \boldsymbol{e}^{-\frac{1}{2}\boldsymbol{tr}([\boldsymbol{S}_0+\boldsymbol{S}_\mu]\Sigma^{-1})}.$$

- Now, we draw $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ using a GS. GS steps:

Step 1: Start with an arbitrary starting value $\boldsymbol{\theta}^0 = (\boldsymbol{\mu}^0, \boldsymbol{\Sigma}^0)$. Set prior values for: $\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0, v_0, \boldsymbol{S}_0$.

Step 2: Generate a sequence of $\boldsymbol{\theta}$'s, following:

- Sample $\boldsymbol{\mu}^{m+1}$ from $p(\boldsymbol{\mu} \mid \boldsymbol{\Sigma}^m; \mathbf{X})$ ~ Normal($\boldsymbol{\mu}_N$, Λ_N)
- Sample $\boldsymbol{\Sigma}^{m+1}$from $p(\boldsymbol{\Sigma} \mid \boldsymbol{\mu}^{m+1}; \mathbf{X})$ ~ IW(v_N, $\boldsymbol{S}_N^{-1}$)

Step 3: Repeat Step 2 for m = 1, 2, ..., M.

MCMC: Gibbs Sampling - Steps

Example: GS steps:

Step 1: Set arbitrary starting value $\boldsymbol{\theta}^0 = (\boldsymbol{\mu}^0, \boldsymbol{\Sigma}^0)$ & $\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0, v_0, \boldsymbol{S}_0$.

Step 2: Generate a sequence of $\boldsymbol{\theta}$'s, following:

- Sample $\boldsymbol{\mu}^{m+1}$ from $P(\boldsymbol{\mu}|\,\boldsymbol{\Sigma}^m; \mathbf{X}) \sim \text{Normal}(\boldsymbol{\mu}_N, \boldsymbol{\Lambda}_N)$
- Sample $\boldsymbol{\Sigma}^{m+1}$from $P(\boldsymbol{\Sigma}\,|\,\boldsymbol{\mu}^{m+1}; \mathbf{X}) \sim \text{IW}(v_N, \boldsymbol{S}_N^{-1})$

Step 3: Repeat Step 2 for m = 1, 2, ... M.

Update Notation:

$\boldsymbol{\Lambda}_N = \mathbf{A}_N^{-1} = (\boldsymbol{\Lambda}_0^{-1} + N\,\boldsymbol{\Sigma}^{-1})^{-1}$

$\boldsymbol{\mu}_N = \mathbf{A}_N^{-1}\,\boldsymbol{b}_N = \boldsymbol{\Lambda}_N\,(\boldsymbol{\Lambda}_0^{-1}\,\boldsymbol{\mu}_0 + N\,\boldsymbol{\Sigma}^{-1}\,\bar{\mathbf{y}})$

$\boldsymbol{S}_{\boldsymbol{\mu}} = \sum_{i=1}^{N} (y_i - \boldsymbol{\mu})\,(y_i - \boldsymbol{\mu})'$

$v_N = N + v_0$

$\boldsymbol{S}_N = \boldsymbol{S}_0 + \boldsymbol{S}_{\boldsymbol{\mu}}$.

MCMC: GS – MVN Example

Example (continuation): We are interested in the monthly correlation between the returns of IBM and DIS from 1990-2016. That is, we want to learn about $\boldsymbol{\Sigma}$.

• **Priors.** We use data on Kellogg's & SPY to set up the priors:

$\boldsymbol{\mu}_0$ = (.0066, 0065)

$\boldsymbol{\Lambda}_0 = S_0$ = rbind(c(.0017, .00065), c(.00065, .0034))

$v_0 = K + 2 = 4$

• **Data**

– $\bar{y}$ = (0.0089, 0.01)

– $\boldsymbol{\Sigma}$ = rbind(c(.0061, .002), c(.002, .0054))

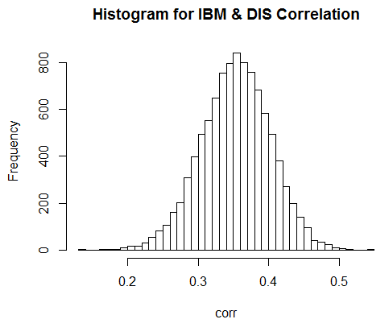
• **Posterior** (M = 10,000; $M^* = M_0$ = 1,000)

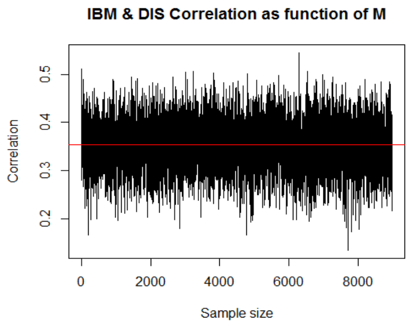
– correlation = ρ = 0.3546. (sample value: 0.3541)

– 95% CI: (0.2567, 0.4477)

MCMC: GS – MVN Example

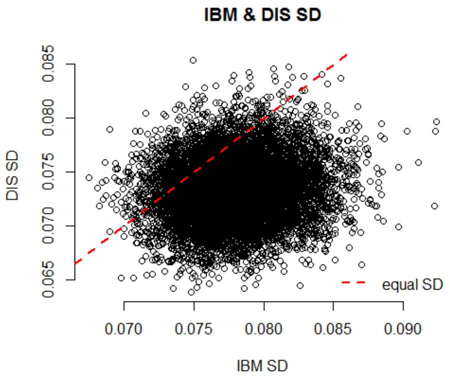
Example (continuation): We also check the histogram and traceplot, plotting the correlation as a function of M.





MCMC: GS – MVN Example

Example (continuation): We may also be interested in knowing which stock is more volatile. We can answer this question by constructing a 95% CI for SD_{IBM} - SD_{DIS}: (-0.00297, 0.012525).



IBM seems to be more volatile (88.12% of the time, IBM has higher SD. But, careful here: this does not mean the difference is 'economically' large).

MCMC: GS – Details

Note: The sequence $\{\theta^{(M)}\}$ is a Markov chain with transition kernel

$$\pi(\theta^{m+1}\,|\,\theta^{m}) = P(\theta_2^{m+1}\,|\,\theta_1^{m+1}; \mathbf{X}) \text{ x } P(\theta_1^{m+1}\,|\,\theta_2^{m}; \mathbf{X})$$

This transition kernel is a conditional distribution function that represents the probability of moving from θ^{m} to θ^{m+1}.

• Under general conditions, the realizations from a Markov Chain as $M \rightarrow \infty$ converge to draws from the ergodic distribution of the chain $\pi(\theta)$ satisfying

$$\pi(\theta^{m+1}) = \int \pi(\theta^{m+1}\,|\,\theta^{m})\; \pi(\theta^{m})\; d\theta^{m}$$

MCMC: GS – Diagnostics

• Convergence can be a problem for MCMC methods. It is important to check the robustness of the results before start using the output:

- Use different θ^{0} (check traceplots for different sequences & GR)
- Use different M_0, M (may use the "effective sample size," ESS)
- Plot θ^{j}as a function of j (check the auto-/cross-correlations in the sequence and across parameters).

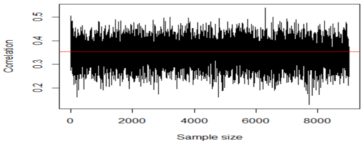
• Run Diagnostics Tests. There are many:

- Geweke (1992): A Z-test, comparing the means of the first 10% of sequence and the last 50% of the sequence.

- Gelman and Rubin (GR, 1992): A test based on comparing different sequences, say *N*. The statistic is called *Shrink factor* is based on the ratio of the variance of the *N* posterior means sequences and the average of the posterior s^2 of the *N* sequences.

MCMC: GS – Diagnostics

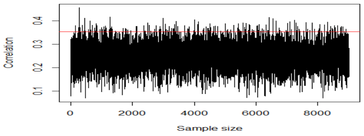
Example: Back to the monthly correlation between the returns of IBM and DIS from 1990-2016. We present traceplots for different θ^0:



$\boldsymbol{\mu}_0$ = (0, 0)

$\boldsymbol{\Lambda}_0 = \mathbf{S}_0$ = rbind(c(.2^2, 0), c(0, .2^2))

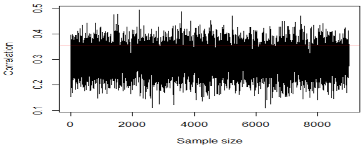
⟹ 95% CI for ρ: (0.2469, 0.4406)



$\boldsymbol{\mu}_0$ = (0, 0)

$\boldsymbol{\Lambda}_0 = \mathbf{S}_0$ = rbind(c(.6^2, -0.1), c(-0.1, .6^2))

⟹ 95% CI for ρ: (0.1460, 0.3500)



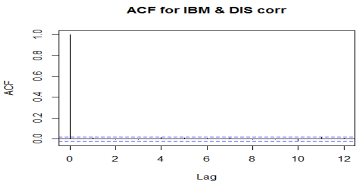
$\boldsymbol{\mu}_0$ = (.04, .04)

$\boldsymbol{\Lambda}_0 = \mathbf{S}_0$ = rbind(c(.05^2, -0.1), c(-0.1, .05^2))

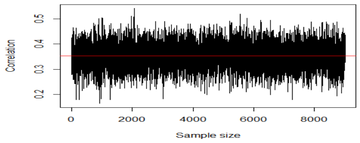
⟹ 95% CI for ρ: (0.1967, 0.3932)

MCMC: GS – Diagnostics

Example: We also present ACF and traceplots for different M_0:

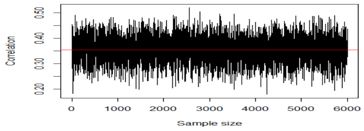


⇒ No autocorrelation problem.



M_0 = 100

⇒ 95% CI for ρ: (0.2565, 0.4478)



M_0 = 4,000

⇒ 95% CI for ρ: (0.2573, 0.4483)

MCMC: GS – Simplicity

• We sample from the known conditional posterior pdf of each parameter. We sample from them using a standard library function.

• Conjugacy is very helpful in this process. For example, if the conditional posterior distributions are all normal, gamma, or beta, then the GS makes sampling from the joint posterior easy.

• To take advantage of the simplicity of the GS, there are situations where drawing from the conditionals is not possible, but it may be possible to convert the problem into one where the GS works (see Albert and Chib's (1993) probit regression set up).

MCMC: GS – Limitations

Three usual concerns:

(1) Even if we have the full posterior joint pdf, it may not be possible or practical to derive the conditional distributions for each of the RVs in the model.

(2) Even if we have the posterior conditionals for each variable, it might be that they are not of a known form, and, thus, there is not a straightforward way to draw samples from them.

(3) There are cases where GS is very inefficient. That is, the "*mixing*" of the GS chain might be very slow, -i.e., the algorithm spends a long time exploring a local region with high density, and takes a very long to explore all regions with significant probability mass.

GS – Example 1: Bivariate Normal

Draw a random sample from bivariate normal $\left[\begin{pmatrix}0\\0\end{pmatrix},\begin{pmatrix}1&\rho\\\rho&1\end{pmatrix}\right]$

(1) Direct approach: $\begin{pmatrix}v_1\\v_2\end{pmatrix}_r = \Gamma\begin{pmatrix}u_1\\u_2\end{pmatrix}_r$ where $\begin{pmatrix}u_1\\u_2\end{pmatrix}$ are two

independent standard normal draws (easy) and $\Gamma=\begin{pmatrix}1&0\\\theta_1&\theta_2\end{pmatrix}$

such that $\Gamma\Gamma'=\begin{pmatrix}1&\rho\\\rho&1\end{pmatrix}$. $\theta_1=\rho,\ \theta_2=\sqrt{1-\rho^2}$.

(2) Gibbs sampler: $v_1 \mid v_2 \sim N\left[\rho v_2, \sqrt{1-\rho^2}\right]$

$v_2 \mid v_1 \sim N\left[\rho v_1, \sqrt{1-\rho^2}\right]$

GS – Example 1: Bivariate Normal

- R Code

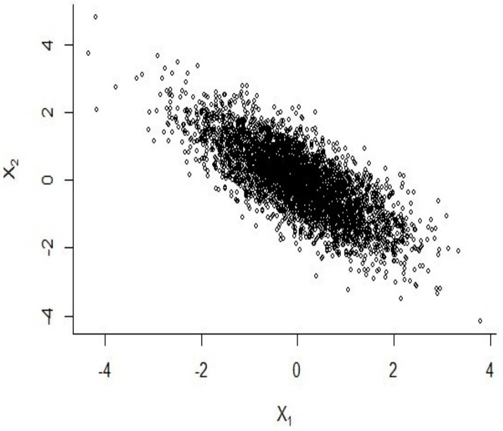
```
# initialize constants and parameters

   M <- 5,000               # length of chain
   burn <- 1,000            # burn-in length
   X <- matrix(0, M, 2)    # the chain, a bivariate sample

   rho <- -.75             # correlation
   mu1 <- 0
   mu2 <- 0
   sigma1 <- 1
   sigma2 <- 1
   s1 <- sqrt(1-rho^2)*sigma1
   s2 <- sqrt(1-rho^2)*sigma2
```

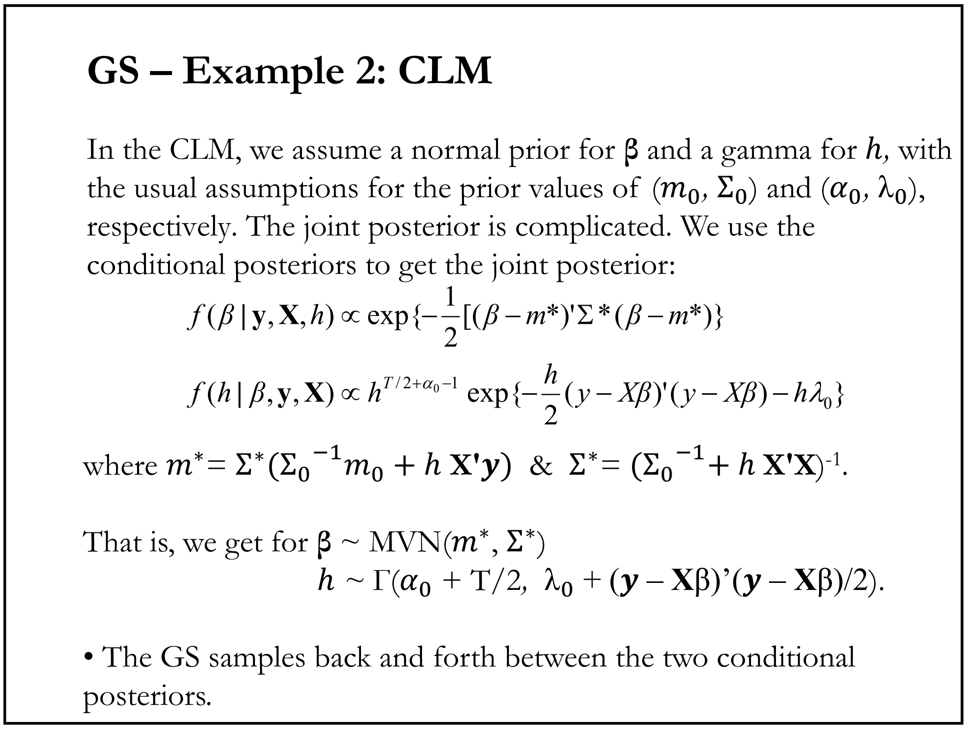
GS – Example 1: Bivariate Normal

```
# generate the chain
   X[1, ] <- c(mu1, mu2)            #initialize
   for (i in 2:M) {
      x2 <- X[i-1, 2]
      m1 <- mu1 + rho * (x2 - mu2) * sigma1/sigma2
      X[i, 1] <- rnorm(1, m1, s1)
      x1 <- X[i, 1]
      m2 <- mu2 + rho * (x1 - mu1) * sigma2/sigma1
      X[i, 2] <- rnorm(1, m2, s2)
   }

   b <- burn + 1
   x <- X[b:M, ]

   # compare sample statistics to parameters
   colMeans(x)
   cov(x)
   cor(x)
   plot(x, main="", cex=.5, xlab=bquote(X[1]),
       ylab=bquote(X[2]), ylim=range(x[,2]))
```

GS – Example 1: Bivariate Normal

```
>     colMeans(x)
[1]  0.03269641 -0.03395135
>     cov(x)
           [,1]       [,2]
[1,]  1.0570041 -0.8098575
[2,] -0.8098575  1.0662894
>     cor(x)
           [,1]       [,2]
[1,]  1.0000000 -0.7628387
[2,] -0.7628387  1.0000000
```





GS – Example 2: CLM

In the CLM, we assume a normal prior for $\boldsymbol{\beta}$ and a gamma for h, with the usual assumptions for the prior values of (m_0, Σ_0) and (α_0, λ_0), respectively. The joint posterior is complicated. We use the conditional posteriors to get the joint posterior:

$$f(\beta \mid \mathbf{y}, \mathbf{X}, h) \propto \exp\{-\frac{1}{2}[(\beta - m^*)'\Sigma^*(\beta - m^*)\}$$

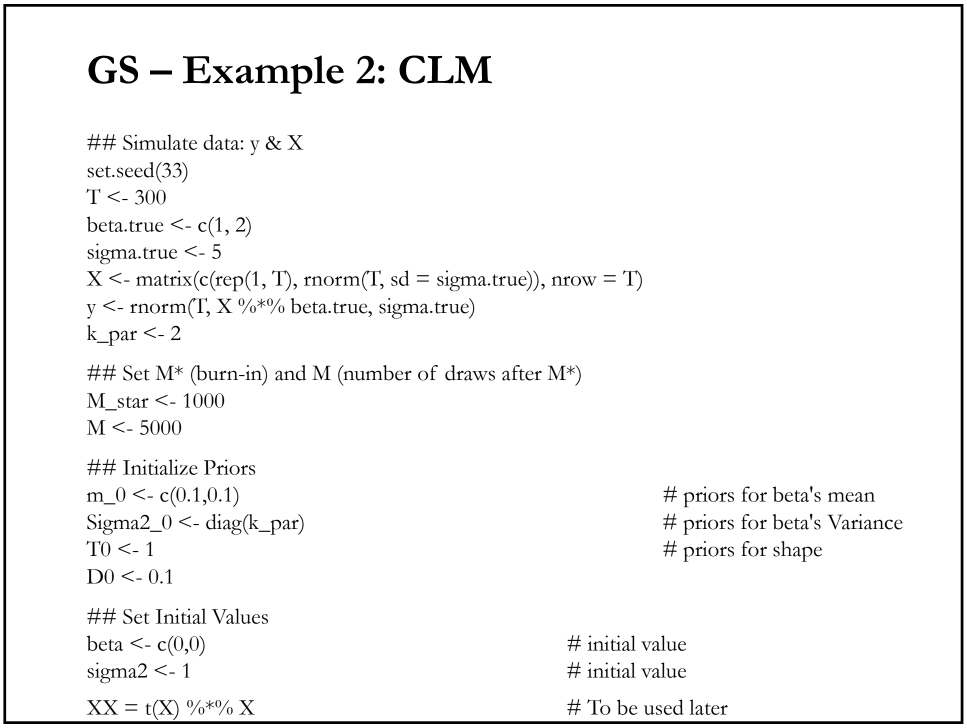
$$f(h \mid \beta, \mathbf{y}, \mathbf{X}) \propto h^{T/2+\alpha_0-1}\exp\{-\frac{h}{2}(y - X\beta)'(y - X\beta) - h\lambda_0\}$$

where $m^* = \Sigma^*(\Sigma_0{}^{-1}m_0 + h\,\mathbf{X'}\boldsymbol{y})$ & $\Sigma^* = (\Sigma_0{}^{-1} + h\,\mathbf{X'X})^{-1}$.

That is, we get for $\boldsymbol{\beta} \sim \text{MVN}(m^*, \Sigma^*)$

$$h \sim \Gamma(\alpha_0 + \text{T}/2,\ \lambda_0 + (\boldsymbol{y} - \mathbf{X}\beta)'(\boldsymbol{y} - \mathbf{X}\beta)/2).$$

- The GS samples back and forth between the two conditional posteriors.



GS – Example 2: CLM

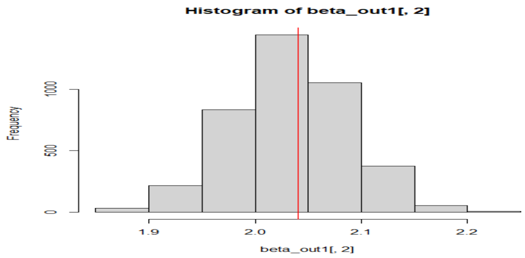
```
## Simulate data: y & X
set.seed(33)
T <- 300
beta.true <- c(1, 2)
sigma.true <- 5
X <- matrix(c(rep(1, T), rnorm(T, sd = sigma.true)), nrow = T)
y <- rnorm(T, X %*% beta.true, sigma.true)
k_par <- 2

## Set M* (burn-in) and M (number of draws after M*)
M_star <- 1000
M <- 5000

## Initialize Priors
m_0 <- c(0.1,0.1)                                    # priors for beta's mean
Sigma2_0 <- diag(k_par)                              # priors for beta's Variance
T0 <- 1                                              # priors for shape
D0 <- 0.1

## Set Initial Values
beta <- c(0,0)                          # initial value
sigma2 <- 1                             # initial value

XX = t(X) %*% X                         # To be used later
```

GS – Example 2: CLM

```
## GS: Iterate over beta and sigma^2 draws
## Assign storage space for holding the GS draws (Output)
beta_out <- matrix(data=NA, nrow= M, ncol= k_par)
sigma_out <- matrix(data = NA, nrow = M, ncol=1)
for (i in 1:M){
   V_beta <- solve(solve(Sigma2_0) + (1/sigma2)*(t(X)%*%X))
   m_beta <- V_beta%*%(solve(Sigma2_0)%*%m_0 + (1/sigma2)*t(X)%*%y)
  beta <- rmvnorm(n=1, m_beta, V_beta)                  # Beta ~ N(m_beta, V_beta)
   resid <- (y - X %*% t(beta))
     T1 <- T0 + T
   D1 <- D0 + t(resid) %*% resid
  sigma2 <- rinvgamma(1, T1, D1)                                    # sigma2 ~IG(T1,D1)
    # save the results.
  beta_out[i,] <- beta
  sigma_out[i,] <- sigma2
}
gs_reg <- lm(y ~ X - 1)
summary(gs_reg)
posterior_means <- apply(beta_out1, 2, mean)
print(cbind(posterior_means, gs_reg$coefficients))
```

GS – Example 2: CLM

```
posterior_means <- apply(beta_out1, 2, mean)
> print(cbind(posterior_means, gs_reg$coefficients))
   posterior_means
X1        1.015181 1.092160
X2        2.032915 2.040219
posterior_sds <- apply(beta_out1, 2, sd)std_errs <-
sqrt(diag(vcov(gs_reg)))print(cbind(posterior_sds, std_errs))
> print(cbind(posterior_sds, std_errs))
   posterior_sds   std_errs
X1    0.27549964 0.27967721
X2    0.05352739 0.05307753
```



GS – Example 3: Logistic Regression

• A standard Bayesian logistic regression model (e.g., modelling the probability of a merger) can be written as follows:

$$y_i \sim Binomial\ (n_i, p_i)$$
$$logit\ (p_i) = X\beta$$
$$\beta_0 \sim N(0, m_0), \beta_1 \sim N(0, m_1)$$

• To use GS, from the complicated posterior, we write down the conditional posterior distributions, as usual. Say, for β_0:

$$p(\beta_0 \mid y, \beta_1) \propto p(y \mid \beta_0, \beta_1) p(\beta_0)$$
$$\propto \prod_i \left(\frac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)} \right)^{y_i} \left(\frac{1}{1 + \exp(\beta_0 + \beta_1 x_i)} \right)^{n_i - y_i} \times \frac{1}{\sqrt{m_0}} \exp(-\frac{\beta_0^2}{2m_0})$$
$$p(\beta_0 \mid y, \beta_1) \sim ?$$

GS – Example 3: Logistic Regression

• But, this distribution is not a standard distribution. It cannot be simply simulated from a standard library function. Thus, the GS cannot be used here.

• But, we can simulate it using MH methods.

• The R package MCMCpack can estimate this model. Also, Bayesian software OpenBUGS, JAGS, WinBUGS, and Stan, which link to R, can fit this model using MCMC.

See R link: https://cran.r-project.org/web/views/Bayesian.html

MCMC: Data Augmentation

- Situation: It is difficult or impossible to sample θ directly from the posterior, $\mathrm{P}(\theta|Y),$ but there exists an unobservable/latent variable Z, such that it is possible to conditionally sample from $\mathrm{P}(\theta|Z,Y)$ & $\mathrm{P}(Z\,|\,\theta,Y)$. Then, we can use the GS to draw from $\mathrm{P}(\theta,Z|Y)$.

- Z is introduced to simplify (sometimes, to improve) the sampler.

- **Data augmentation** (**DA**): Methods for constructing iterative optimization or sampling algorithms through the introduction of unobserved data or latent variables.

- DA was popularized by Dempster, Laird, and Rubin (1977), in their article on the EM algorithm, and by Tanner and Wong (1987). Chib (1992) is the first application in econometrics (in a Tobit model).

MCMC: Data Augmentation

- Typical application in economics/finance: Censored data.

- A DA algorithm starts with the construction of the so-called **augmented data**, Y_{aug}, which are linked to the observed data, Y_{obs}, via a many-to-one mapping M: $Y_{aug} \longrightarrow Y_{obs}$.

- Now, we have "complete data." But, we require that the marginal distribution of Y_{obs} implied by $\mathrm{P}(Y_{aug}|\theta)$ must be the original model $\mathrm{P}(Y_{obs}\,|\,\theta)$. That is, we relate the "observed data" posterior distribution to the "complete data":

$$\begin{aligned}\mathrm{P}(\theta\,|\,Y_{obs},M) &= \int f(\theta,Y_{aug}\,|\,Y_{obs},M)\;dY_{aug}\\ &= \int f(\theta|Y_{aug},Y_{obs},M) f(Y_{aug}\,|\,Y_{obs},M) dY_{aug}\end{aligned}$$

MCMC: Data Augmentation

- Now, we have "complete data." But, we require that the marginal distribution of Y_{obs} implied by $\mathrm{P}(Y_{aug}|\theta)$ must be the original model $\mathrm{P}(Y_{obs}\,|\,\theta)$. That is, we relate the "observed data" posterior distribution to the "complete data":

$$\mathrm{P}(\theta\,|\,Y_{obs}, M) = \int f(\theta, Y_{aug}\,|\,Y_{obs}, M)\; dY_{aug}$$
$$= \int f(\theta|Y_{aug}, Y_{obs}, M) f(Y_{aug}\,|\,Y_{obs}, M) dY_{aug}$$

- We introduce the RV Y_{aug} because it helps. We have a situation where the GS can be used to simulate $\mathrm{P}(\theta\,|\,Y_{obs}, M)$. Two steps:

– Draw Y_{aug} from their joint posterior, $\mathrm{P}(Y_{aug}\,|\,Y_{obs}, M)$

– Draw θ from its completed-data posterior: $\mathrm{P}(\theta\,|\,Y_{obs}, Y_{aug}, M)$

Q: Under which conditions, inference from completed data and inference from observed data are the same?

MCMC: Data Augmentation – Example 1

Suppose we are interested in estimating the parameters of a censored regression. There is a latent variable:

$$Y_i{}^* = X_i\, \beta + \varepsilon_i, \qquad \varepsilon_i\,|X_i \sim iid\, \mathrm{N}(0, 1) \quad i = 1, 2, ..., K, ..., L$$

- We observe $Y_i = \max(0, Y_i{}^*)$, and the regressors X_i. Suppose we observe $(L - K)$ zeroes. (We do not observe the negative Y_i*'s.)

- Assume prior distribution for $\beta \sim \mathrm{N}(\mu, \Omega)$. But, the posterior distribution for β does not have a closed form expression.

<u>Remark</u>: We view both the vector $\boldsymbol{Y}^* = (Y_1{}^*,..., Y_N{}^*)$ & β as unknown RV. With an appropriate choice of $\mathrm{P}(\boldsymbol{Y}^*|\,\text{data}, \beta\,)$ & $\mathrm{P}(\beta\,|\,\boldsymbol{Y}^*)$, we can use a Gibbs sample to get the full posterior $\mathrm{P}(\beta, \boldsymbol{Y}^*|\,\text{data})$.

MCMC: Data Augmentation – Example 1

- The GS consists of two steps:

Step 1 (Imputation): Draw all the missing –i.e., with negative values- elements of $\boldsymbol{Y}^*$ given the current value of the parameter β, say β^m:

$Y_i^* \mid \beta$, data $\sim$ TN($X_i\,\beta^m$, 1; 0) (an $(L-K)$ x 1vector)

if observation i is truncated, where TN(μ, σ^2; c) denotes a truncated normal distribution with mean μ, variance σ^2, and truncation point c (truncated from above). See Botev (2017) for an algorithm to draw from a TN.

Note: Only draw missing elements. That is, if $Y_i > 0$, set $Y_i^* = Y_i$.

Step 2 (Posterior): Draw a new value for the parameter, β^{m+1} given the data and given the (partly drawn) $\boldsymbol{Y}^*$:

$$P(\beta \mid \text{data}, \boldsymbol{Y}^*) \sim N((\boldsymbol{X}'\boldsymbol{X}+\Omega^{-1})^{-1}(\boldsymbol{X}'\boldsymbol{Y}+\Omega^{-1}\mu), (\boldsymbol{X}'\boldsymbol{X}+\Omega^{-1})^{-1})$$

MCMC: Data Augmentation – Example 2

Example: Incomplete univariate data

Suppose that $Y_1, ..., Y_L \sim$ Binomial $(1, \theta)$

Prior for θ $\sim$ Beta(α, β)

Then, the posterior of θ is also Beta:

$$P(\theta|Y) \sim \text{Beta}(\alpha + \textstyle\sum_{i=1}^{L} Y_i,\ \beta + L - \sum_{i=1}^{L} Y_i)$$

Suppose $L-K$ observations are missing. That is, $Y_{obs} = \{Y_1, \dots, Y_K\}$.

Then, $P(\theta|Y_{obs}) \sim \text{Beta}(\alpha + \sum_{i=1}^{K} Y_i,\ \beta + K - \sum_{i=1}^{K} Y_i)$

Step 1: Draw all the missing elements of $\boldsymbol{Y}^*$ given the current value of the parameter θ, say θ^m.

Step 2: Draw a new value for the parameter, θ^{m+1} given the data and given the (partly drawn) $\boldsymbol{Y}^*$.

MCMC: Data Augmentation – Example 3

Example: Auxiliary Variable

Suppose that $Y_1, ..., Y_N \sim iid\, \text{N}(\mu_i, \sigma^2)$ where $i = 0, 1$

We have a mixture of normals, with a mixture proportion, π.

Priors:

$$\pi \sim \text{Beta}(\alpha, \beta).$$
$$\mu_i \sim \text{N}(m, l^{-1})$$

Likelihood:

The pdf of Y_i:

$$p(y_i|\, \mu, \pi) = (1 - \pi) * N(y_i\,|\mu_0, \sigma^2) + \pi * N(y_i\,|\mu_1, \sigma^2)$$

A complicated likelihood, that makes the posterior complicated.

MCMC: Data Augmentation – Example 3

Example (continuation):

We define an equivalent model, using an auxiliary, latent variable, Z_i:

$$\pi \sim \text{Beta}(\alpha, \beta).$$
$$Z_1, ..., Z_N \sim \text{Bernouille}(\pi)$$
$$Y_1, ..., Y_N \sim iid\, \text{N}(\mu(z_i), \sigma^2)$$

The pdf:

$$\begin{aligned} p(y_i|\, \mu, \pi) &= p(y_i|Z_i = 0, \mu, \pi)\, p(Z_i = 0|\, \mu, \pi) + \\ &\quad p(y_i|Z_i = 1, \mu, \pi)\, p(Z_i = 1|\, \mu, \pi) \\ &= (1 - \pi) * N(y_i\,|\mu_0, \sigma^2) + \pi * N(y_i\,|\mu_1, \sigma^2) \end{aligned}$$

We can use this pdf to build a Gibbs sampler:

$$p(\pi|\mathbf{y}, \mathbf{z}, \mu) = \text{Beta}(\pi|\alpha + N_0, \beta + N_1). \quad (N_j = \textstyle\sum_{i=1}^{N} I(Z_i = j))$$

Given $\mathbf{z}$, we know where y_i is drawn from. We have two normals.

MCMC: Data Augmentation – Example 3

Example (continuation): Conditional posteriors:

- $p(\pi|\boldsymbol{y}, \boldsymbol{z}, \mu)$ = Beta($\pi|\alpha + N_0$, β+N_1). ($N_j = \sum_{i=1}^{N} I(Z_i = j)$)

• Given $\boldsymbol{z}$, we know where y_i is drawn from. We have two independent Normal-normal models:

$$\mu_0|\mu_1, \pi, \boldsymbol{y}, \boldsymbol{z} \sim \mathrm{N}(m_0, {l_0}^{-1})$$

$$\mu_1|\mu_0, \pi, \boldsymbol{y}, \boldsymbol{z} \sim \mathrm{N}(m_1, {l_1}^{-1})$$

where

$$l_j = l + N_j\, h \qquad (h = \sigma^2)$$

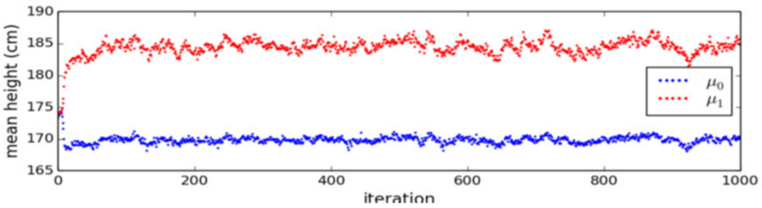
$$m_j = (l\, m + h \textstyle\sum_{i,z_i=k}^{N} y_i)/(l +/N_j\, h)$$

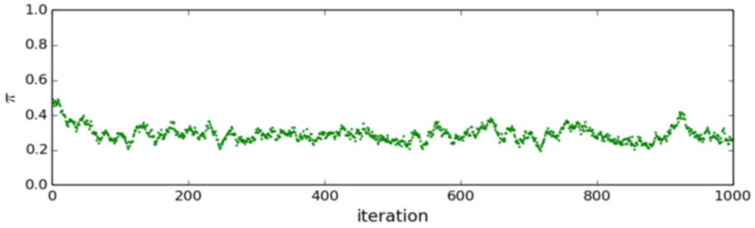
- $p(z|\boldsymbol{y}, \pi, \mu)$ = Bernoulli($z_i\,|\,\alpha_{i,1}/(\alpha_{i,0} + \alpha_{i,1})$

where $\alpha_{i,0} = (1-\pi) * N(y_i|\mu_{0,}\, \sigma^2)$ & $\alpha_{i,1} = \pi\ *\ N(y_i|\mu_1,\ \sigma^2)$

MCMC: Data Augmentation – Example 3

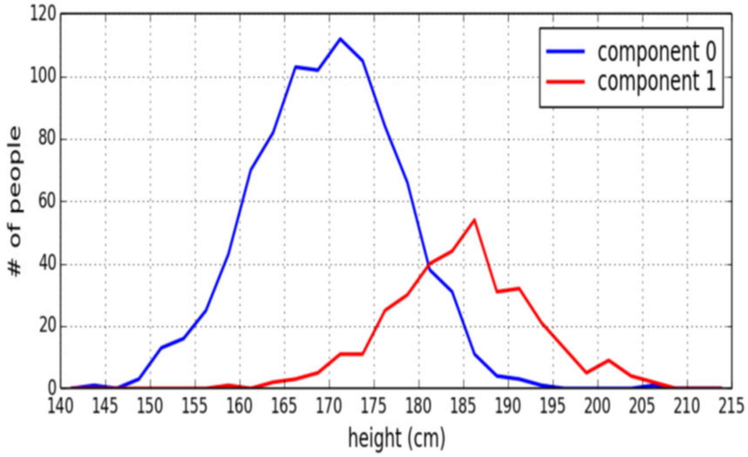
Example (continuation): Application to data of heights of Dutch men (562) and women (695). Traceplots for μ_i and π:





MCMC: Data Augmentation – Example 3

Example (continuation): Histogram of heights:



MCMC: Metropolis-Hastings (MH)

- **MH** is an alternative, and more general, way to construct an MCMC sampler (to draw from the posterior). The Gibbs sampler is a simplified version of the MH algorithm (so simplified, it does not look like it).

- It provides a form of generalized rejection sampling, where values are drawn –i.e., the θ's– from approximate distributions and "corrected" so that, asymptotically they behave as random observations from the *target distribution* –for us, the posterior.

- MH sampling algorithms sequentially draw candidate observations from a '*proposal*' distribution, conditional on the current observations, thus inducing a Markov chain.

MCMC: Metropolis-Hastings (MH)

• We deal with Markov chains: The distribution of the next sample value, say $y = \theta^{m+1}$, depends on the current sample value, say $x = \theta^{m}$.

• In principle, the algorithm can be used to sample from any integrable function. But, its most popular application is sampling from a posterior distribution.

• The MH algorithm jumps around the parameter space, but in a way that the probability to be at a point is proportional to the function we sample from –i.e., the target function.

• Named after Metropolis et al. (1953), which first proposed it and Hastings (1970), who generalized it. Rediscovered by Tanner and Wong (1987) and popularized by Gelfand and Smith (1990).

MCMC: MH – Proposal Distribution

• We want to find a function P(x, y). from where we can sample, that satisfies the *(time) reversibility condition* (*equation of balance*), a sufficient condition for stationarity of $\pi(.)$:

$$\pi(x)\ \mathrm{P}(x, y). = \pi(y)\ \mathrm{P}(y, x).$$

• The *proposal* (or *candidate-generating) density* is denoted $q(x, y)$, where

$$\int q(x, y)\ dy = 1.$$

Interpretation: When a process is at the point x (=θ^{m}), the density generates a value y (=θ^{m+1}) from $q(x, y)$. It tells us how to move from current x to new y. Alternative notation for $q(x, y)$ = $q(y|x)$.

• Idea: Suppose P is the true density. We simulate *y* using $q(x, y)$. We 'accept' it only if it is "*likely*." If it happens that $q(x, y)$ itself satisfies the reversibility condition for all (x, y), we are done.

MCMC: MH – Proposal Distribution

• But, for example, we might find that for some (x, y):

$$\pi(x)\ q(x, y) > \pi(y)\ q(y, x) \qquad \text{(*)}$$

In this case, speaking somewhat loosely, the process moves from x to y too often and from y to x too rarely.

• We want balance. To correct this situation by reducing the number of moves from x to y with the introduction of a probability $(x, y) < 1$ that the move is made:

$$a(x, y) = \text{probability of move from } x \text{ to } y.$$

If the move is not made, the process again returns x as a value from the target distribution.

• Then, transitions from x to y ($y \neq x$) are made according to

$$p_{MH}(x, y) = q(x, y)\ a(x, y) \qquad y \neq x$$

MCMC: MH – Algorithm Rejection Step

Example: We focus on a single parameter θ and its posterior distribution $\pi(\theta)$. We draw a sequence $\{\theta^1, \theta^2, \theta^3, ...\}$ from a MC.

– At iteration m, let $\theta = \theta^m$. Then, propose a move: θ^*. That is, generate a new value θ^* from a proposal distribution $q(\theta^m, \theta^*)$.

– Rejection rule:

Accept θ^* (& let $\theta^{m+1} = \theta^*$) with (acceptance) probability $a(\theta^m, \theta^*)$

Reject θ^* with probability $1-a$ (& set $\theta^{m+1} = \theta^m$)

⇒ We have defined an acceptance function!

Note: It turns out that the acceptance probability, $a(x, y)$, is a function of $\pi(y)/\pi(x)$ –the *importance ratio*. This ratio helps the sampler to visit higher probability areas under the full posterior.

MCMC: MH – Probability of Move

- We need to define $a(x,y)$, the probability of move.

- In our example (*), to get movements from x to y, we define $a(y,x)$ to be as large as possible (with upper limit 1!). Now, the probability of move $a(x,y)$ is determined by requiring that $p_{MH}(x,y)$ satisfies the reversibility condition. Then,

$$\pi(x)\ q(x,y)\ a(x,y) = \pi(y)\ q(y,x)\ a(y,x) = \pi(y)\ q(y,x)$$

$$\Rightarrow a(x,y) = \frac{\pi(y)\ q(y,x)}{\pi(x)\ q(x,y)}$$

Note: If (*) is reversed, we set $a(x,y) = 1$ and $a(y,x)$ as above.

- Then, in order for $p_{MH}(x,y)$ to be reversible, $a(x,y)$ must be

$$a(x,y) = \min\{\frac{\pi(y)\ q(y,x)}{\pi(x)\ q(x,y)}, 1\} \qquad \text{if } \pi(x)\ q(x,y)>0,$$

$$= 1 \qquad \text{otherwise.}$$

MCMC: MH – Probability of Move

- If $q(.)$ is symmetric, then $q(x,y) = q(y,x)$. Then, the probability of move $a(x,y)$ reduces to $\pi(y)/\pi(x)$ –the *importance ratio*. Thus, the acceptance function:
  - If $\pi(y) \geq \pi(x)$, the chain moves to y.
  - Otherwise, it moves with probability given by $\pi(y)/\pi(x)$.

Note: This case, with $q(.)$ symmetric, is called Metropolis Sampling.

- This acceptance function plays 2 roles:

1) It helps the sampler to visit higher probability areas under the full posterior –we do this through the ratio $\pi(y)/\pi(x)$.

2) It should explore the space and avoid getting stuck at one site –i.e., it can reverse its previous move. This constraint is given by the ratio $q(y,x)/q(x,y)$.

MCMC: MH – At Work

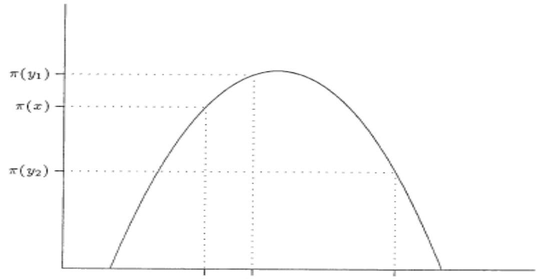


Figure 1. Calculating Probabilities of Move With Symmetric Candidate-Generating Function (see text).

- We consider moves from x (note that $q(x, y)$ is symmetric):
  - A move to candidate y_1 is made with certainty –i.e., $\pi(y_1) > \pi(x)$
    - ⇒ We always say yes to an "up-hill" jump!
  - A move to candidate y_2 is made with probability $\pi(y_2)/\pi(x)$.

Note: The $q(x, y)$ distribution is also called *jumping distribution.*

MCMC: MH – Transition Kernel

- In order to complete the definition of the transition kernel for the MH chain, we consider the possibly non-zero probability that the process remains at x :

$$r(x) = 1 - \int_R q(x, y)\ a(x, y)\ dx.$$

- Then, the transition kernel of the MH chain, denoted by $p_{MH}(x, y)$ is given by:

$$p_{MH}(x, y) = q(x, y)\ a(x, y)\ dy + [\ 1 - \int_R q(x, y)\ a(x, y)\ dy]\ I_x(y)$$

where the indicator function $I_x(y) = 1$ if $x = y$

$= 0$ otherwise.

MCMC: MH Algorithm

- MH Algorithm

We know $\pi(\theta)$ = P($\theta|y$) = P($y\,|\,\theta$) x P(θ)/P(y), a complicated posterior. For example, from the CLM with Y_i *i.i.d.* normal, normal prior for β and gamma prior for h. $\Rightarrow \theta = (\beta, h)$.

Then, $$\frac{\pi(\theta^{m+1})}{\pi(\theta^{m})}=\frac{P(\theta^{m+1}\,|\,y)P(\theta^{m})}{P(\theta^{m}\,|\,y)P(\theta^{m+1})}.$$

P(y), the normalizing constant, plays no role. Again, we ignore it.

Assumptions:

- Symmetric $q(.)$ –i.e., $q(\theta^m, \theta^{m+1}) = q(\theta^{m+1}, \theta^m)$.

$\Rightarrow a(\theta^m, \theta^{m+1}) = \pi(\theta^{m+1})/\pi(\theta^m)$.

- A starting value for θ: θ^0 (m = 0).

MCMC: MH Algorithm

- MH Algorithm – Steps:

(1) Initialized with the starting value θ^0 (m=0):

(2) Generate θ^* from q(θ^m, .) and draw u from U(0, 1).

- If $u \leq a(\theta^m, \theta^*) = \pi(\theta^*)/\pi(\theta^m)$ $\Rightarrow$ set $\theta^{m+1} = \theta^*$.

Else $\Rightarrow$ set $\theta^{m+1} = \theta^m$.

(3) Repeat for m = 1 , 2 ,. . . , M.

- Return the values $\{\theta^{(M)}\} = (\theta^1, \theta^2, \theta^3, \dots, \theta^m, \theta^{m+1}, \dots, \theta^M)$.

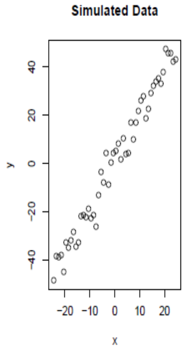
MCMC: MH Algorithm – CLM Example

- (From Florian Hartig) Suppose we have the CLM with

Data: $Y_i = \alpha + \beta X_i + \varepsilon_i$, $\varepsilon_i \,|X_i \sim iid$ N(0, σ^2).

Priors: $\beta \sim$ U(0,10); $\alpha \sim$ N(m=0, σ_0^2=9); & $\sigma^2 \sim U(0.001,30)$

$\Rightarrow \boldsymbol{\theta} = (\alpha, \beta, \sigma^2)$.



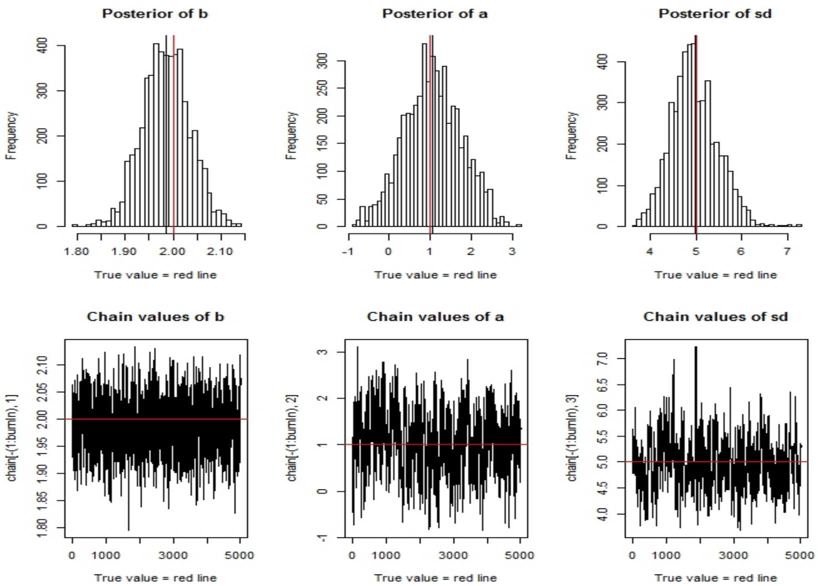
- We simulate the data, with α=1, β=2, σ=5, & T=50.

- Proposal densities: 3 Normals with $\boldsymbol{\theta}^0$ = (2,0,7) & **SD** = (0.1,0.5,0.3).

- Iterations = M =10,000 & Burn-in = M_0 = 5,000.

- OLS: a = 1.188 (0.68), b = 1.984 (.047).

MCMC: MH Algorithm – CLM Example



MCMC: MH – Remarks about q(.)

• We pick proposal distributions, $q(x,y)$, that are easy to sample. But, remarkably, $q(x,y)$ can have almost any form.

• It is usually a good idea to choose $q(x,y)$ close to the posterior, $\pi(.)$.

• There are some exceptions; but assuming that the proposal allows the chain to explore the whole posterior and does not produce a recurrent chain we are OK.

• We tend to work with symmetric $q(x,y)$, but the problem at hand may require asymmetric $q(x,y)$; for instance, to accommodate a particular constraints in the model. For example, to estimate the posterior distribution for a variance parameter, we require that our proposal does not generate values smaller than 0.

MCMC: MH – Special Cases

• Three special cases of MH algorithm are:

1. Random walk metropolis sampling. That is,

$$y = x + z, \qquad z \sim q(z)$$

2. Independence sampling. That is,

$$q(x,y) = q(y).$$

3. Gibbs sampling. (We never reject from the proposals –i.e., the conditional posteriors!)

• Critical decision: Selecting the spread and location of $q(x,y)$. Note that difference choices deliver different *acceptance rates* –i.e., the fraction of candidate draws that are accepted.

Changing the spread and location of $q(x,y)$ to get a desired acceptance rate is called *tuning*.

MCMC: MH – Random Walk Metropolis

- This is a pure Metropolis sampling –see Metropolis et al. (1953).

– Let $q(x, y) = q(|y – x|)$ $q(.)$ is a multivariate symmetric pdf.

– $y = x + z$, where $z \sim q$. (It is called a *random walk* chain!)

- Typical RW proposals: Normal distribution centered around the current value of the parameter –i.e., $q(x, y) \sim$ N(x, s^2), where s^2 is the (fixed) proposal variance that can be *tuned* to give particular acceptance rates. Multivariate t-distributions are also used.

- The RW MH is a good alternative, usual default for the algorithm.

MCMC: MH Algorithm – RW Example

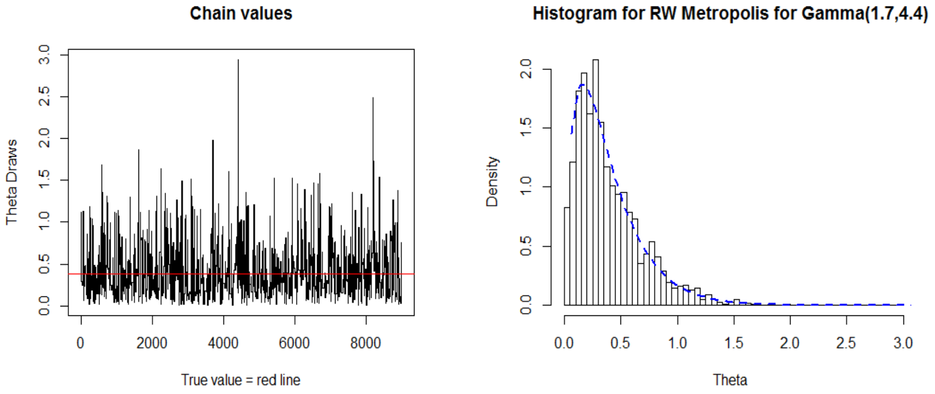
Example: (From P. Lam.) We use a RW MH algorithm to sample from a Gamma(1.7, 4.4) distribution with $q(x, y) \sim$ N(x, [SD=2]^2).

```
mh.gamma <- function(M.sims, start, burnin, cand.sd, shape, rate) {
      theta.cur <- start
      draws <- c()
      theta.update <- function(theta.cur, shape, rate)  {
            theta.can <- rnorm(1, mean = theta.cur, sd = cand.sd)                      # RW θ^{m+1}?
            accept.prob <- dgamma(theta.can, shape, rate)/dgamma(theta.cur, shape, rate)   # a()
             if (runif(1) <= accept.prob)  theta.can                                    # reject?
             else theta.cur
            }
     for (i in 1:M.sims) {
            draws[i] <- theta.cur <- theta.update(theta.cur, shape,rate)
     }
return(draws[(burnin + 1):M.sims])
}
 mh.draws <- mh.gamma(10000, start = 1, burnin = 1000, cand.sd = 2, shape = 1.7, rate = 4.4)
```

MCMC: MH Algorithm – RW Example

Example (continuation):

```
> mean(mh.draws)
[1] 0.3962097                                    # theoretical mean = 1.7/4.4 = 0.3863636
> hist(mh.draws, main="Histogram for RW Metropolis for Gamma(1.7,4.4)",xlab="Theta", breaks=50)
```



MCMC: MH – Independence Sampler

• The independence sampler is so called as each proposal is independent of the current parameter value. That is,

$q(x,y) = q(y)$ (an independent chain –see Tierney (1994).)

That is, all our candidate draws y are drawn from the same distribution, regardless of where the previous draw was.

This leads to acceptance probability $a(x,y) = \min\left(1, \frac{\pi(y)q(x)}{\pi(x)q(y)}\right).$

Note that to determine $a(x, y)$, we use a ratio of importance weights.

• Distributions used for $q(.)$: A Normal based around the ML estimate with inflated variance. A Multivariate-t.

MCMC: MH – Independence Sampler

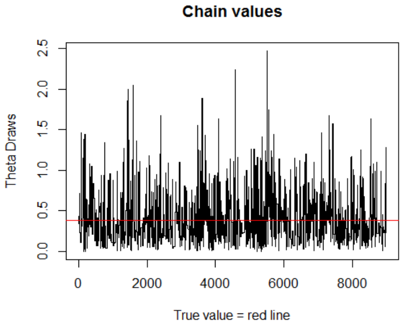
Example: We use an independence sampler MH algorithm to sample from a Gamma(1.7, 4.4) distribution with $q(y)$ ~ N(0, [SD=2]^2).

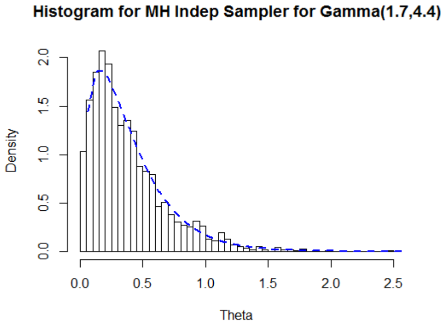
Code in R: Same code, but the theta.update function changes to

```
 theta.update <- function(theta.cur, shape, rate)  {
              theta.can <- rnorm(1, mean = cand.mu, sd = cand.sd)                          # IS θ^{m+1}?
              accept.prob <- dgamma(theta.can, shape, rate)*dnorm(theta.cur, cand.mu, cand.sd)/
(dgamma(theta.cur, shape, rate)*dnorm(theta.can, cand.mu, cand.sd))                         # a(.)
               if (runif(1) <= accept.prob)  theta.can                                      # reject?
               else theta.cur
              }
```

MCMC: MH – Independence Sampler

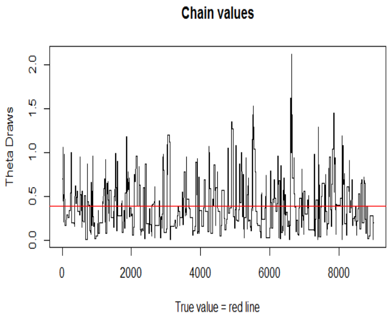
Example (continuation): Below, we draw the traceplot and histogram for the generated draws. The generated ED looks fine. That is, it can be used to calculate quantities of interest.





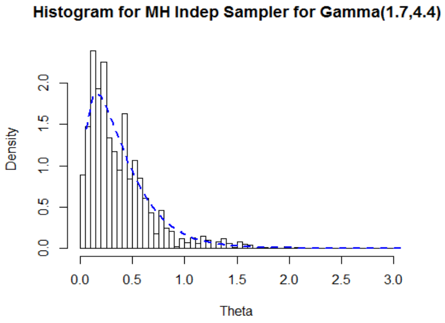
MCMC: MH – Independence Sampler

Example (continuation): Now, suppose we start with x_0=3 and use $q(y)$ ~ N(2, [SD=6]^2). To check the chain, we show below the traceplot and histogram:



Chain values

True value = red line



Histogram for MH Indep Sampler for Gamma(1.7,4.4)

<u>Note</u>: No clear convergence. The chain seems stuck in some values.
⇒ The chain may not be a good approximation to π(θ).

MCMC: MH – Independence Sampler

- The independence sampler can sometimes work very well but can also work very badly!

- The efficiency depends on how close the jumping distribution is to the posterior.

- Generally speaking, the chain will behave well only if the q(.) proposal distribution has heavier tails than the posterior and has similar shape to π(.) .

MCMC: MH – Acceptance Rates

• It is important to monitor the *acceptance rate* (the fraction of candidate draws that are accepted) of the MH algorithm:

– If the acceptance rate is too high, the chain is likely not *mixing* well –i.e., not moving around the parameter space enough.

– If it is too low, the algorithm is too *inefficient* –i.e., rejecting too many draws.

• In general, the acceptance rate falls as the dimension of P$(\theta|y)$ increases (especially, for highly dependent parameters) resulting in slow moving chains and long simulations.

• Simulation times can be improved by using the single component MH algorithm. Instead of updating the whole θ together, θ is divided in parts –say, (β, h)–, with each component updated separately.

MCMC: MH – Acceptance Rates & Tuning

• What is high or low is algorithm specific. One way of finding a '*good*' $q(.)$ is to choose a pdf that gives a particular acceptance rate.

• When we *scale* –i.e., adjust the scale parameters– $q(.)$, say σ, to obtain a particular acceptance rate, we say "we are *tuning* the MH."

• In general, tuning is simple: Proposal jump sizes are

- Increased when acceptance rates are high.
- Decreased when rates are low.

• The above mechanism suggests an *optimal* scale parameter –i.e., the proposal explores the parameter space efficiently.

MCMC: MH – Acceptance Rates & Tuning

- For RW MH, Roberts, Gelman and Gilks (1997), with Gaussian proposals and *i.i.d* components of θ, suggest as "good" acceptance rate:

– 45% for unidimensional problems.
– **23.4%** in the limit (some theoretical support for this result)
– 25% for 6 dimensions.

- 23.4% is often used in practice to tune $q(.)$.

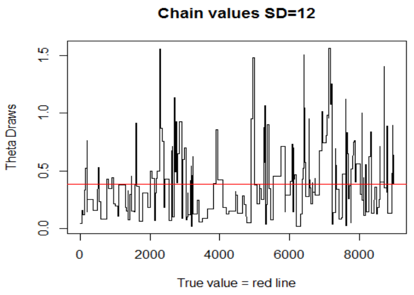
- There is a literature, however, –see, Bedard (2008)– arguing that in many cases 23.4% may be inefficient, for example, hierarchical models.

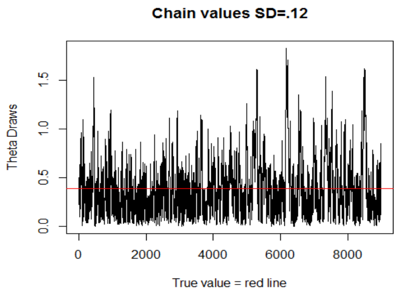
- For Independent MH, Muller (1993) suggests a 'good' rate is close to 100%.

MCMC: MH – Acceptance Rates & Tuning

Example: Back to the RW Metropolis algorithm to sample from a Gamma(1.7, 4.4) distribution with $q(x,y) \sim N(x, SD^2)$. Before, we used SD=2.

Now, we try two extreme SDs to illustrate the usual trade-off:
- SD=12 (too big), with acceptance rate 2.4% ⇒ inefficient.
- SD=.12 (too small), with acceptance rate 86% ⇒ not mixing well.

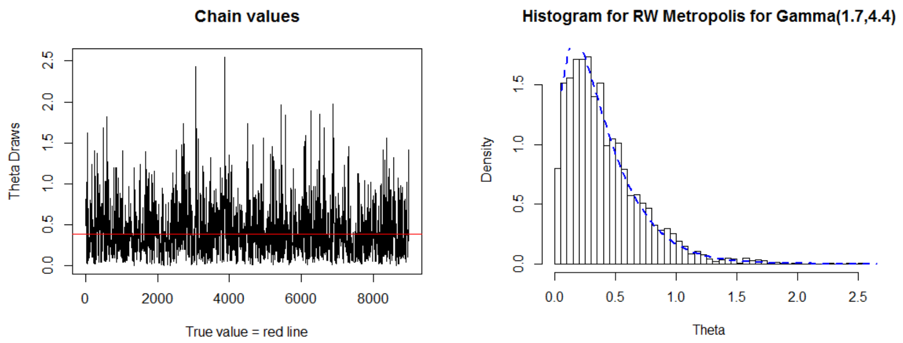




MCMC: MH – Acceptance Rates & Tuning

Example: If we use the Roberts et al's (1994) 23.4% acceptance rate as a target, then we adjust SD to get close to it. When SD=2, the acceptance rate was 14.2% (low).

Tuning SD to **1.2**, we get a 23.8% acceptance rate. The ED generated looks better (see the traceplot and histogram below):



MCMC: MH – Tuning: Adaptive Method

• Adaptive Method (ad hoc)

- Before the burn-in, we have an **adaptation period**, where the sampler improves the proposal distribution. The adaptive method requires a desired acceptance rate, for example, 30% and tolerance, for example, 10% resulting in an acceptable range of (20%, 40%).

- If we are outside the acceptable range, say we reject too much, we scale the proposal distribution, for example, by changing/reducing the spread (say, σ).

- Think of it as a method to find starting values.

• MLwiN uses an adaptive method to construct univariate Normal proposals with an acceptance rate of approximately 50%.

MCMC: MH – Adaptive Method Algorithm

- Run the MH sampler for consecutive batches of 100 iterations. Compare the number accepted, *N* with the desired acceptance rate, *R*. Adjust variance accordingly:

$$\text{If } N \le R, \quad \Rightarrow \sigma_{new} = \sigma_{old} / (2 - \tfrac{N}{R})$$
$$\text{If } N > R, \quad \Rightarrow \sigma_{new} = \sigma_{old} \times (2 - \tfrac{100-N}{100-R})$$

- Repeat this procedure until 3 consecutive values of *N* lie within the acceptable range and then, **mark** (fixed) this parameter. Check other parameters.

- When all the parameters are marked the adaptation period is over.

Note: Proposal SDs are still modified after being marked until adaptation period is over.

MCMC: MH – Autocorrelation

- A usual problem in poor MCMC performance is high autocorrelation, or "*stickiness*" in the chain.

- Estimators based on MC samples (based on *independent* draws from the target) perform better than MCMC samples, which are correlated samples.

- The SE of both estimators is given by:

$\text{Var}_{\text{MC}}[\bar{\theta}\,|\,y] = \text{Var}[\theta\,|\,y]/M$

$\text{Var}_{\text{MCMC}}[\bar{\theta}\,|\,y] = \text{Var}[\theta\,|\,y]/M + 1/M\, \Sigma_{m \neq t}\, \text{E}\{(\theta^m - \theta_{\text{true}})\ (\theta^t - \theta_{\text{true}})\}$

where $\bar{\theta}$ approximates $\text{E}[\theta\,|\,y] = \theta_{\text{true}}$. The 2nd term in $\text{Var}_{\text{MCMC}}\ [\bar{\theta}\,|\,y]$ is, in general, positive (& higher than $\text{Var}_{\text{MC}}[\bar{\theta}\,|\,y]$).

MCMC: MH – Autocorrelation

- Thus, we expect the MCMC approximation to be worse. The higher the autocorrelation, the less information we have in the chain. We may need a very large M to get enough information to estimate quantities of interest from $\pi(\theta)$.

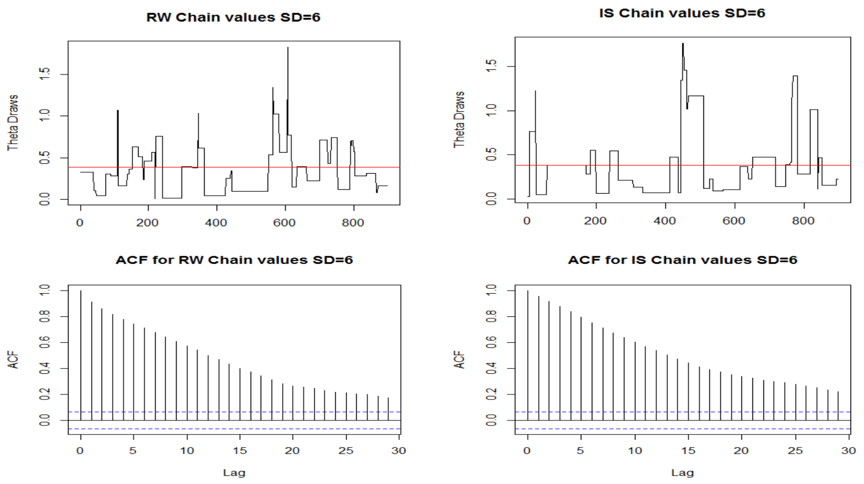
- That is, a chain with high autocorrelation moves around the parameter space Θ slowly, taking a long time to achieve the correct balance of samples to approximate $\pi(\theta)$.

- It is common practice to adjust the level of correlation by adjusting $q(.)$, usually, tuning σ.

- Sample autocorrelation function (ACF) plots are used to determine how much autocorrelation is in the chain.

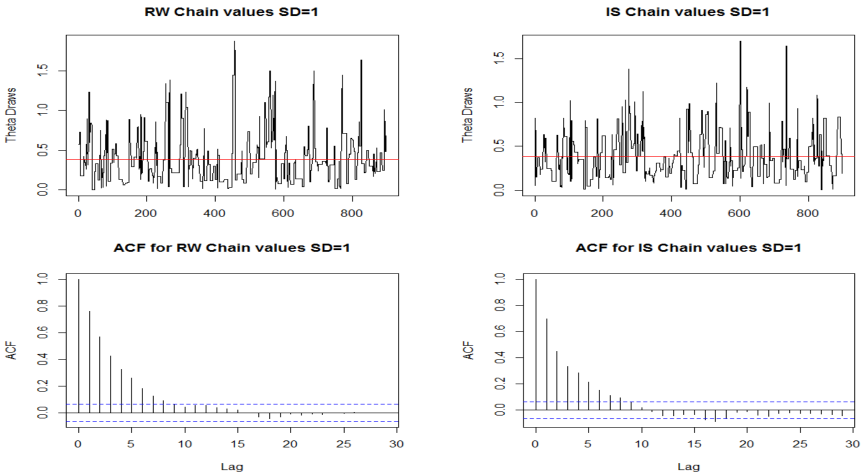
MCMC: MH – Autocorrelation: Tuning

Example: MH algorithm sampling from a Gamma(1.7, 4.4) pdf with $q(.)$ ~ N(., SD^2). We plot the first 1,000 values & ACF (SD=6).



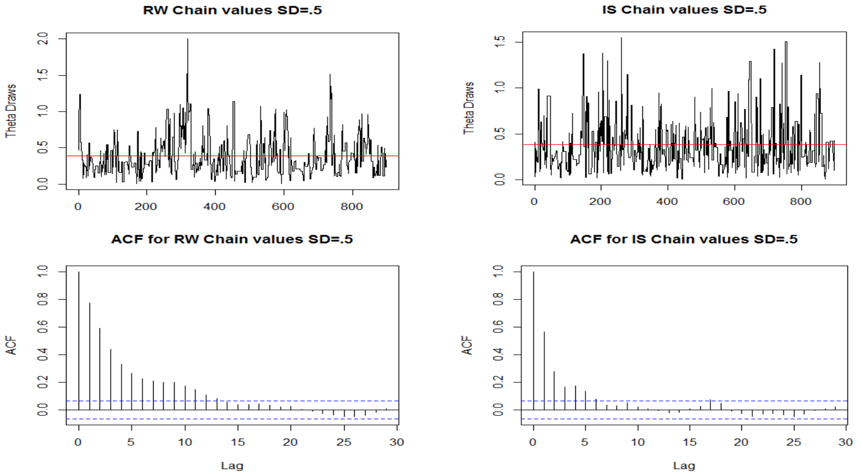
MCMC: MH – Autocorrelation: Tuning

Example (continuation): Now, we plot the first 1,000 values & ACF (SD=1).



MCMC: MH – Autocorrelation: Tuning

Example (continuation): Now, we plot the first 1,000 values & ACF (SD=.5).



MCMC: MH – Autocorrelation - Remarks

• In practice, we adjust the level of correlation by scaling $q(.)$.

• By looking at the previous graphs, it is tempting to reduce σ to get a lower autocorrelation (and faster convergence). But, as σ → 0, the acceptance rate goes to 1. That is, the chain never moves (see RW chain with SD = 0.5).

• That is, there is a trade-off when adjusting σ to control for low autocorrelation; we want σ to be:

- large enough so that the chain moves quickly throughout Θ.

- but not so large σ that the rejection rate is too high.

MCMC: MH – Diagnostics and Convergence

• Similar diagnostics tools as the ones discussed for the Gibbs Sampler.

• Convergence is a practical problem for MCMC methods.

– Converge can be slow ⇒ Let the MH algorithm run.

– There are some formal tests –see Robert and Casella (2004). In the usual complicated setups they tend to have a Type II error problem (accept convergence too much/too quickly) ⇒ Rely on graphs (traceplots & histograms, correlograms).

Practical advise: Run the algorithm until some iteration *M**, where it looks like the chain is stationary. Then, run it *M* more times to check! Discard the first *M** iterations. Keep the rest to approximate $\pi(.)$.

MCMC: MH – Remarks

• MH sampling produces a chain, $\{\theta^{(M)}\}$, with $\pi(.)$ as limiting distribution. The chain allows us to calculate quantities of interest of $\pi(.)$. (moments, C.I., etc.) when *i.i.d.* simulations cannot be used.

• Pros:

– We need less information about $\pi(.)$ than other methods.

– With little tuning, the algorithm works reasonably well.

– Large dimensions problems can be broken into sets of smaller ones.

• Cons:

– Results are only asymptotic (when $M \to \infty$).

– Convergence may be very slow. For practical purposes, the algorithm may not converge.

– Detecting slow convergence may be difficult.

Application 1: Bivariate Normal

• We want to simulate values from

$$f(x) \propto \exp\left\{-(1/2)\left(\mathbf{x}'\mathbf{\Sigma}^{-1}\mathbf{x}\right)\right\}; \quad \Sigma=\begin{pmatrix} 1 & .9 \\ .9 & 1 \end{pmatrix}$$

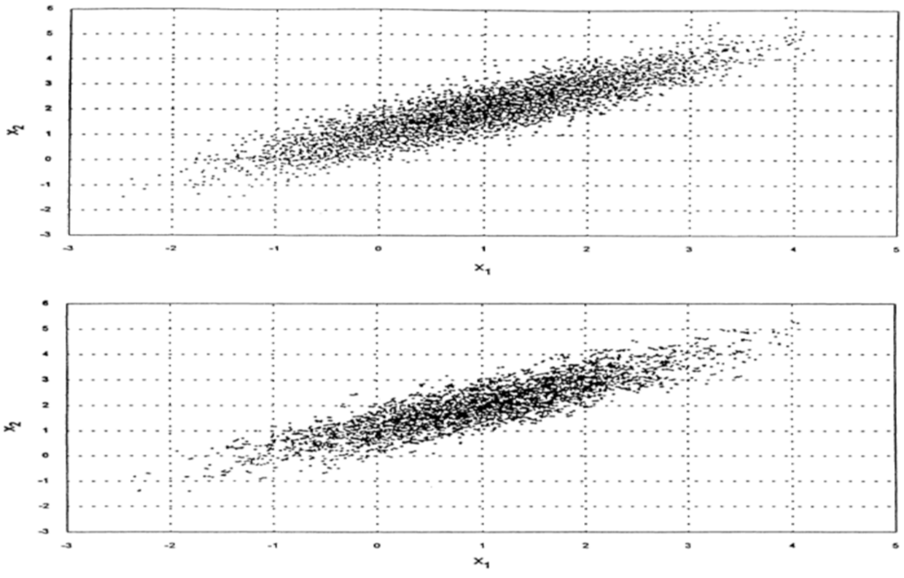
• Proposal distribution: RW chain: $y = x + z$, $z \sim$ bivariate Uniform on $(-\delta_i, \delta_i)$, for i = 1 ,2. (δ_i controls the spread)

To avoid excessive move, let δ_1=.75 and δ_2=1.

• The probability of move (for a symmetric proposal) is:

$$\alpha(x, y) = \min\left(1, \frac{\exp\{-\frac{1}{2}(y-\mu)'\Sigma^{-1}(y-\mu)\}}{\exp\{-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\}}\right).$$

Application 1: Bivariate Normal



Application 2: The Probit Model (Greene)

- The Probit Model:

(a) $y_i^* = \mathbf{x}_i'\boldsymbol{\beta} + \varepsilon_i$

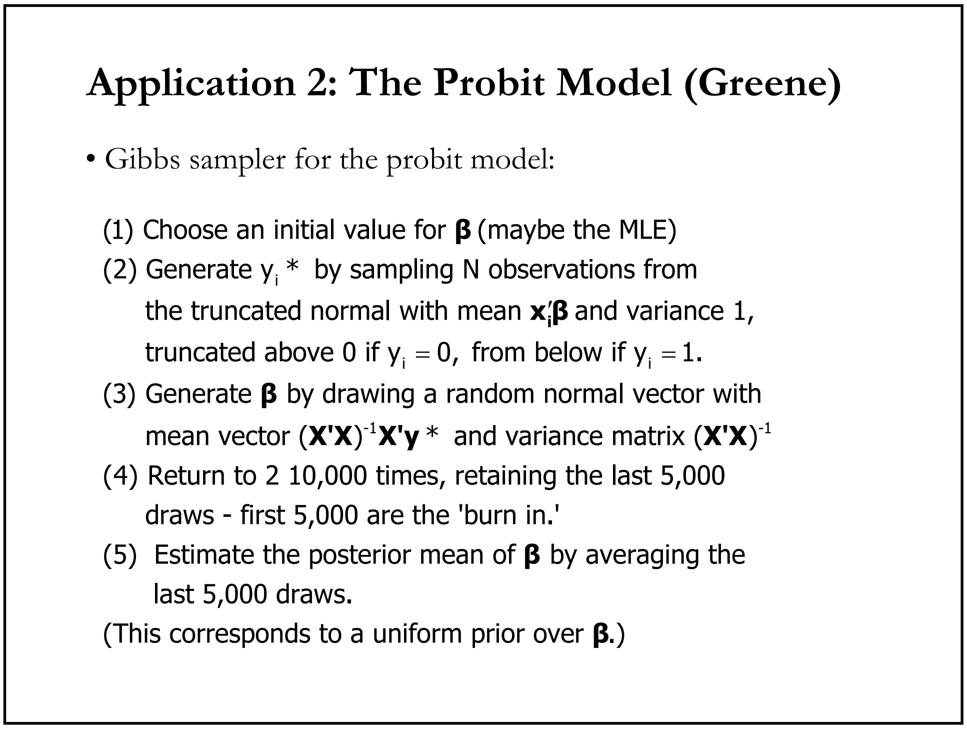
$\varepsilon_i \sim N[0,1]$

(b) $y_i = 1$ if $y_i^* > 0$, 0 otherwise

Consider estimation of $\boldsymbol{\beta}$ and y_i^* (data augmentation)

(1) If y* were observed, this would be a linear regression
(y_i would not be useful since it is just sgn(y_i*).)
We saw in the linear model before, p($\boldsymbol{\beta}$ | y_i^*, y_i)

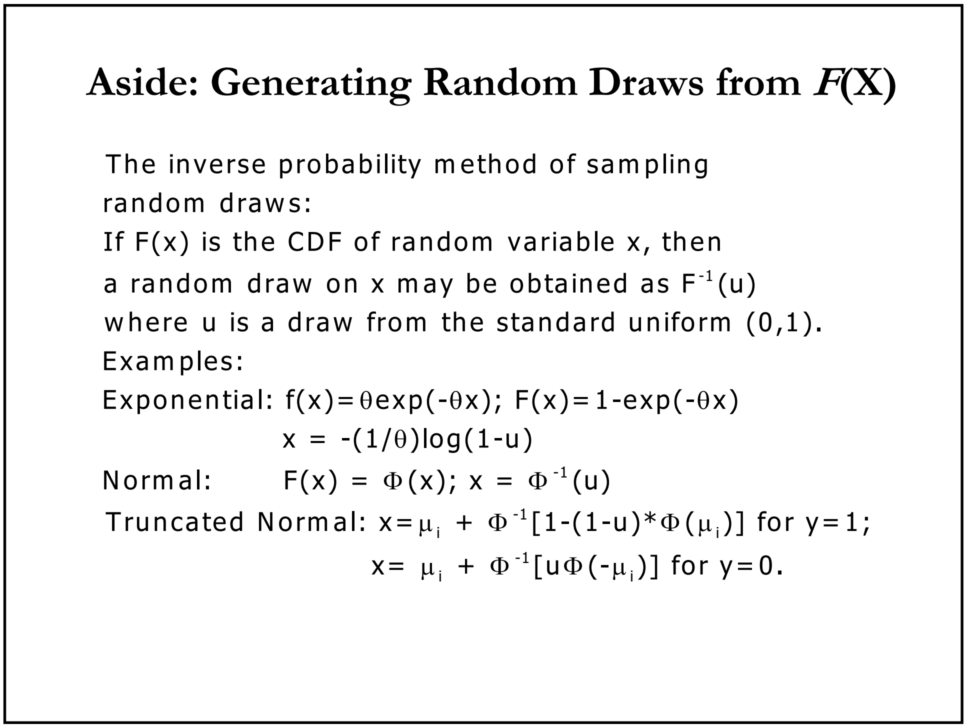
(2) If (only) $\boldsymbol{\beta}$ were observed, y_i^* would be a draw from the normal distribution with mean $\mathbf{x}_i'\boldsymbol{\beta}$ and variance 1.
But, y_i gives the sign of y_i^* . $y_i^* \mid \boldsymbol{\beta}, y_i$ is a draw from the truncated normal (above if y=0, below if y=1)



Application 2: The Probit Model (Greene)

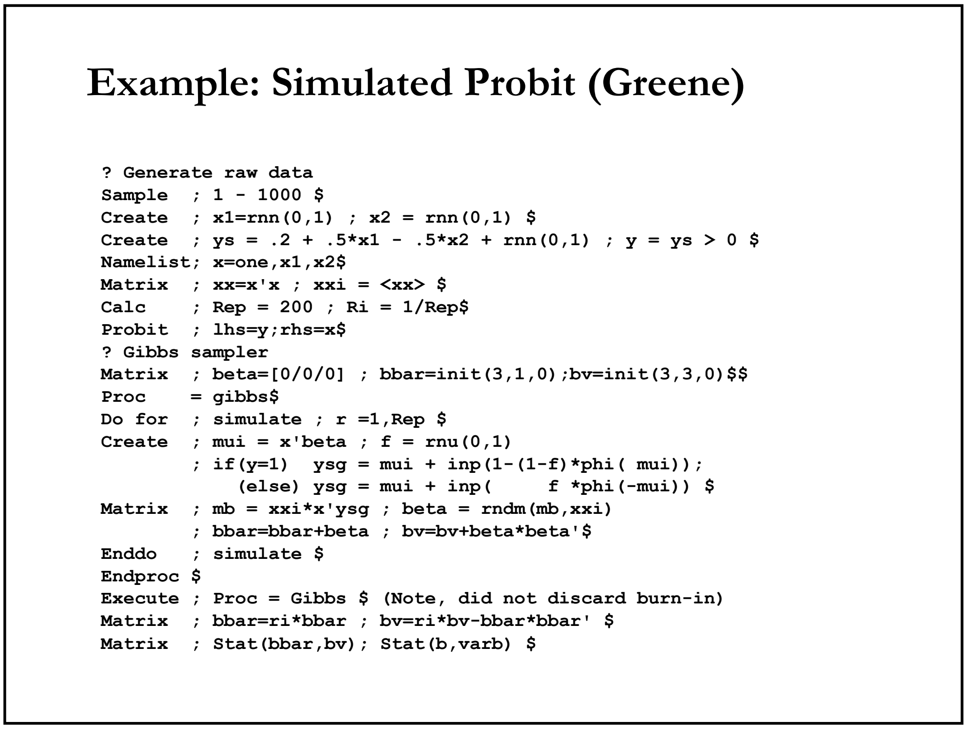
- Gibbs sampler for the probit model:

(1) Choose an initial value for $\boldsymbol{\beta}$ (maybe the MLE)
(2) Generate y_i* by sampling N observations from
the truncated normal with mean $\mathbf{x}_i'\boldsymbol{\beta}$ and variance 1,
truncated above 0 if $y_i = 0$, from below if $y_i = 1$.
(3) Generate $\boldsymbol{\beta}$ by drawing a random normal vector with
mean vector $(\mathbf{X'X})^{-1}\mathbf{X'y}*$ and variance matrix $(\mathbf{X'X})^{-1}$
(4) Return to 2 10,000 times, retaining the last 5,000
draws - first 5,000 are the 'burn in.'
(5) Estimate the posterior mean of $\boldsymbol{\beta}$ by averaging the
last 5,000 draws.
(This corresponds to a uniform prior over $\boldsymbol{\beta}$.)



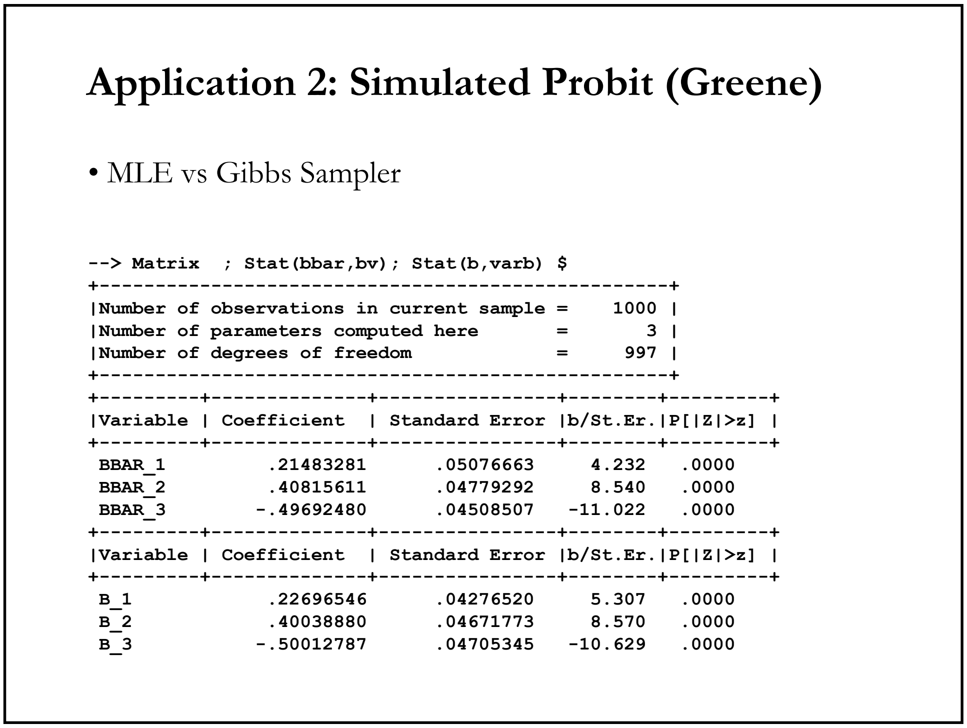
Aside: Generating Random Draws from *F*(X)

The inverse probability method of sampling
random draws:
If F(x) is the CDF of random variable x, then
a random draw on x may be obtained as $F^{-1}(u)$
where u is a draw from the standard uniform (0,1).
Examples:
Exponential: $f(x)=\theta\exp(-\theta x)$; $F(x)=1-\exp(-\theta x)$
$x = -(1/\theta)\log(1-u)$
Normal: $F(x) = \Phi(x)$; $x = \Phi^{-1}(u)$
Truncated Normal: $x=\mu_i + \Phi^{-1}[1-(1-u)*\Phi(\mu_i)]$ for $y=1$;
$x= \mu_i + \Phi^{-1}[u\Phi(-\mu_i)]$ for $y=0$.



Example: Simulated Probit (Greene)

```
? Generate raw data
Sample   ; 1 - 1000 $
Create   ; x1=rnn(0,1) ; x2 = rnn(0,1) $
Create   ; ys = .2 + .5*x1 - .5*x2 + rnn(0,1) ; y = ys > 0 $
Namelist; x=one,x1,x2$
Matrix   ; xx=x'x ; xxi = <xx> $
Calc     ; Rep = 200 ; Ri = 1/Rep$
Probit   ; lhs=y;rhs=x$
? Gibbs sampler
Matrix   ; beta=[0/0/0] ; bbar=init(3,1,0);bv=init(3,3,0)$$
Proc     = gibbs$
Do for   ; simulate ; r =1,Rep $
Create   ; mui = x'beta ; f = rnu(0,1)
         ; if(y=1)  ysg = mui + inp(1-(1-f)*phi( mui));
             (else) ysg = mui + inp(     f *phi(-mui)) $
Matrix   ; mb = xxi*x'ysg ; beta = rndm(mb,xxi)
         ; bbar=bbar+beta ; bv=bv+beta*beta'$
Enddo    ; simulate $
Endproc $
Execute ; Proc = Gibbs $ (Note, did not discard burn-in)
Matrix   ; bbar=ri*bbar ; bv=ri*bv-bbar*bbar' $
Matrix   ; Stat(bbar,bv); Stat(b,varb) $
```



Application 2: Simulated Probit (Greene)

- MLE vs Gibbs Sampler

```
--> Matrix  ; Stat(bbar,bv); Stat(b,varb) $
+---------------------------------------------------+
|Number of observations in current sample =    1000 |
|Number of parameters computed here       =       3 |
|Number of degrees of freedom             =     997 |
+---------------------------------------------------+
+---------+--------------+----------------+--------+---------+
|Variable | Coefficient  | Standard Error |b/St.Er.|P[|Z|>z] |
+---------+--------------+----------------+--------+---------+
 BBAR_1         .21483281      .05076663     4.232   .0000
 BBAR_2         .40815611      .04779292     8.540   .0000
 BBAR_3        -.49692480      .04508507   -11.022   .0000
+---------+--------------+----------------+--------+---------+
|Variable | Coefficient  | Standard Error |b/St.Er.|P[|Z|>z] |
+---------+--------------+----------------+--------+---------+
 B_1            .22696546      .04276520     5.307   .0000
 B_2            .40038880      .04671773     8.570   .0000
 B_3           -.50012787      .04705345   -10.629   .0000
```

Application 3: Stochastic Volatility (SV)

In the SV model, we have

$$h_t = \omega + \phi h_{t-1} + \eta_t ; \qquad \eta_t \sim N(0, \sigma_\eta^2)$$

Or in logs,

$$\log\ h_t = \omega + \phi \log\ h_{t-1} + \eta_t$$

• We have 3 SV parameters to estimate φ =(ω, ϕ, σ_η^2) and the latent h_t.

• The difference with ARCH models: The shocks that govern the volatility are not necessarily mean ε_t's. There is a volatility shock.

• SVOL Estimation is based on the idea of hierarchical structure:

- f($y|h_t$)	(distribution of the data given the volatilities)
- f(h_t|φ)	(distribution of the volatilities given the parameters)
- f(φ)	(distribution of the parameters)

Application 3: Stochastic Volatility (SV)

Bayesian Goal: To get the posterior f(h_t, φ|**y**)

Priors (Beliefs):
Normal-Gamma for f(φ). (Standard Bayesian regression model)
- Inverse-Gamma for f(σ_η^2)
- Impose (assume) stationarity of h_t. (Truncate ϕ as necessary)

Algorithm: MCMC
Augment the parameter space to include h_t.
With a proper prior for f(h_t, φ), the MCMC provides inference about the joint posterior f(h_t, φ| y).

• Classic reference: Andersen (1994), *Mathematical Finance*.

• Application to interest rates: Kalimipalli and Susmel (2004, *JEF*).

Application 3: Stochastic Volatility (SV)

• Gibbs Algorithm for Estimating SV Model --from K&S (2004).

$$\Delta r_t - (\hat{a}_0 + \hat{a}_1 r_{t-1}) \equiv RES_t$$

$$RES_t = \sqrt{h_t r_{t-1}^{2\alpha}}\,\varepsilon_t, \qquad \alpha = 0.5$$

$$\ln(h_t) = \omega + \phi_1 \ln(h_{t-1}) + \sqrt{\sigma_\eta^2}\;\eta_{t-1}$$

- In the SV model, we estimate the parameter vector and 1 latent variable: $\theta = \{\omega, \sigma_\eta, \phi_1,\}$ and $H_t = \{h_1,...,h_t\}$.
- Parameter set therefore consists of $\Theta = \{H_t, \theta\}$ for all t.

• Using Bayes theorem to decompose the joint posterior density as follows.

$$f(H_n,\theta) \propto f(Y_n|H_n) f(H_n|,\theta) f(\theta)$$

Application 3: Stochastic Volatility (SV)

$$f(H_n,\theta) \propto f(Y_n|H_n) f(H_n|,\theta) f(\theta)$$

• Next draw the marginals $f(H_t|Y_t,\theta)$ & $f(\theta|Y_t,H_t)$, using a Gibbs sampling algorithm:

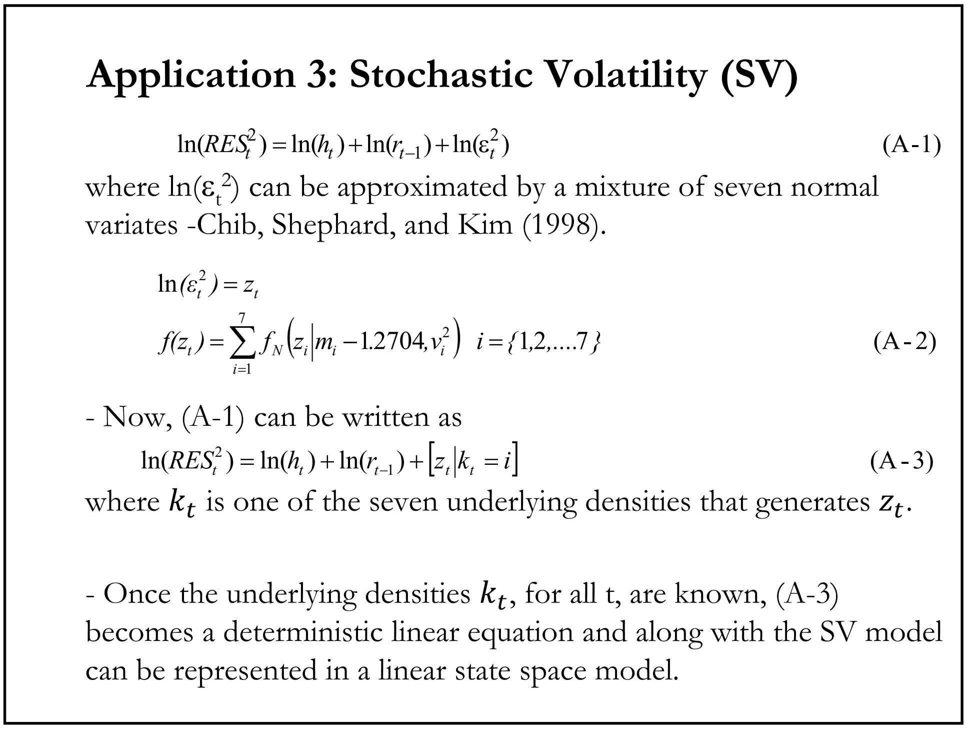
Step 1: Specify initial values $\theta^{(0)} = \{\omega^{(0)}, \sigma_\eta^{(0)}, \phi^{(0)}\}$. Set $i = 1$.

Step 2:

Draw the underlying volatility using the multi-move simulation sampler –see, De Jong and Shephard (1995)--, based on parameter values from **step 1**.

- The multi-move simulation sampler draws H_t for all the data points as a single block. Recall we can write:

$$\ln(RES_t^2) = \ln(h_t) + \ln(r_{t-1}) + \ln(\varepsilon_t^2) \qquad \text{(A-1)}$$



Application 3: Stochastic Volatility (SV)

$$\ln(RES_t^2) = \ln(h_t) + \ln(r_{t-1}) + \ln(\varepsilon_t^2) \qquad \text{(A-1)}$$

where $\ln(\varepsilon_t^{\,2})$ can be approximated by a mixture of seven normal variates -Chib, Shephard, and Kim (1998).

$$\ln(\varepsilon_t^2) = z_t$$

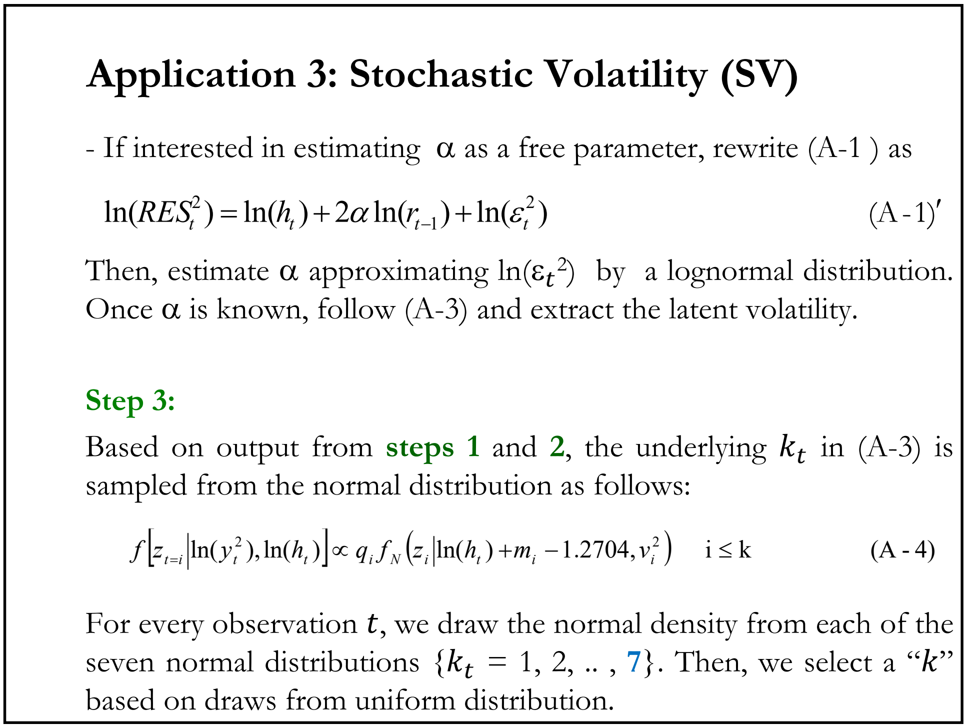
$$f(z_t) = \sum_{i=1}^{7} f_N\left(z_i \middle| m_i - 1.2704, v_i^2\right) \quad i = \{1,2,....7\} \qquad \text{(A-2)}$$

- Now, (A-1) can be written as

$$\ln(RES_t^2) = \ln(h_t) + \ln(r_{t-1}) + \left[z_t \middle| k_t = i\right] \qquad \text{(A-3)}$$

where k_t is one of the seven underlying densities that generates z_t.

- Once the underlying densities k_t, for all t, are known, (A-3) becomes a deterministic linear equation and along with the SV model can be represented in a linear state space model.



Application 3: Stochastic Volatility (SV)

- If interested in estimating α as a free parameter, rewrite (A-1) as

$$\ln(RES_t^2) = \ln(h_t) + 2\alpha \ln(r_{t-1}) + \ln(\varepsilon_t^2) \qquad \text{(A-1)}'$$

Then, estimate α approximating $\ln(\varepsilon_t^{\,2})$ by a lognormal distribution. Once α is known, follow (A-3) and extract the latent volatility.

Step 3:

Based on output from **steps 1** and **2**, the underlying k_t in (A-3) is sampled from the normal distribution as follows:

$$f\left[z_{t=i} \middle| \ln(y_t^2), \ln(h_t)\right] \propto q_i f_N\left(z_i \middle| \ln(h_t) + m_i - 1.2704, v_i^2\right) \quad \text{i} \leq \text{k} \qquad \text{(A-4)}$$

For every observation t, we draw the normal density from each of the seven normal distributions $\{k_t = 1, 2, .. , 7\}$. Then, we select a "k" based on draws from uniform distribution.

Application 3: Stochastic Volatility (SV)

Step 4:

Cycle through the conditionals of parameter vector $\theta = \{\omega, \sigma_\eta, \phi_1\}$ for the volatility equation using Chib (1993), using output from **steps 1-3**. Assuming that $f(\theta)$ can be decomposed as:

$$f(\theta|Y_n, H_n) \propto f(\omega|Y_n, H_n, \theta_{-\omega}) f(\sigma_\eta^2|Y_n, H_n, \theta_{-\sigma^2}) f(\phi|Y_n, H_n, \theta_{-\phi}) \qquad \text{(A-5)}$$

where θ_{-j} refers to the θ parameters excluding the jth parameter.

- The conditional distributions (normal for ω and ϕ, inverse gamma for σ_η^2) are described in Chib (1993). You need to specify the prior means and standard deviations.

Step 5: Go to **step 2**. (Now, Set i =2.)

Conclusions (Greene)

- Bayesian vs. Classical Estimation
  - In principle, different philosophical views and interpretation
  - As practiced, just two different algorithms
  - The religious debate is a red herring –i.e., misleading.

- Gibbs Sampler. A major technological advance
  - Useful tool for both classical and Bayesian
  - New Bayesian applications appear daily

References:

Gelman *et al.* (2016), "**Bayesian Data Analysis**" (3rd edition).

Kruschke, John (2014), **Doing Bayesian Data Analysis**: A Tutorial with R, JAGS, and Stan (2nd edition).

Standard Criticisms (Greene)

- Of the Classical Approach
  - Computationally difficult (ML vs. MCMC)
  - It is difficult to pay attention to heterogeneity, especially in panels when *N* is large.

  Responses: None are true. See, e.g., Train (2003, Ch. 10)

- Of Classical Inference in this Setting
  - Asymptotics are “only approximate” and rely on “imaginary samples.” Bayesian procedures are “exact.”

  Response: The inexactness results from acknowledging that we try to extend these results outside the sample. The Bayesian results are “exact” but have no generality and are useless except for this sample, these data and this prior. (Or are they? Trying to extend them outside the sample is a distinctly classical exercise.)

Standard Criticisms (Greene)

- Of the Bayesian Approach
  - Computationally difficult.

  Response: Not really, with MCMC and Metropolis-Hastings
  - The prior (conjugate or not) is a hoax. It has nothing to do with “prior knowledge” or the uncertainty of the investigator.

  Response: In fact, the prior usually has little influence on the results. (Bernstein and von Mises Theorem)

- Of Bayesian ‘Inference’
  - It is not statistical inference
  - How do we discern any uncertainty in the results?

  This is precisely the underpinning of the Bayesian method. There is no uncertainty. It is ‘exact.