# Markov chain Monte Carlo

- **Markov chain Monte Carlo (MCMC) methods introduced below can be used to generate a draw from a distribution that approximates some target distribution $f(\cdot)$**
- **The basic idea is that we know that $\mathbf{X} \sim f(\mathbf{x})$, but we can't get $f(\mathbf{x})$ in closed form or we can't simulate easily from $f(\mathbf{x})$.**
- **The MCMC methods introduced below can be used to generate a draw from a distribution that approximates $f$, but they are more properly viewed as methods for generating a sample from which expectations of functions of X can reliably be estimated.**

# Brief review of Markov chains: Definitions

1. **A discrete-time, discrete-space Markov chain is $X^{(0)}, X^{(1)}, \ldots$ where $X^{(t)}$ obeys the *Markov property* that**

$$P\left[X^{(t)} \middle| x^{(0)}, \ldots, x^{(t-1)}\right] = P\left[X^{(t)} \middle| x^{(t-1)}\right]$$

2. **A Markov chain is governed by a *transition probability matrix*, $\mathbf{P}$, with the $ij$th element equaling $P\left[X^{(t+1)} = j \middle| X^{(t)} = i\right]$.**

3. **Let $\boldsymbol{\pi}$ denote a vector of probabilities with $i$th element $\pi_i$ denoting the marginal probability that $X^{(t)} = i$. Then the marginal distribution of $X^{(t+1)}$ must be $\boldsymbol{\pi}^T\mathbf{P}$.**

   **If $\boldsymbol{\pi}^T\mathbf{P} = \boldsymbol{\pi}^T$ then $\boldsymbol{\pi}$ is a *stationary distribution* for $\mathbf{P}$.**

4. **A Markov chain is *irreducible* if any state $j$ can be reached from any state $i$ in a finite number of steps for all $i$ and $j$.**

5. **A Markov chain is *periodic* if it can visit certain portions of the state space only at regularly spaced intervals.**

# Brief review of Markov chains: Stationary distribution

**If $X^{(1)}, X^{(2)}, \ldots$ are realizations from an irreducible and aperiodic Markov chain with stationary distribution $\boldsymbol{\pi}$, then**

1. **$X^{(n)}$ converges in distribution to the distribution given by $\boldsymbol{\pi}$**
2. **For any function $h$**

$$\frac{1}{n}\sum_{t=1}^{n} h(X^{(t)}) \to \mathrm{E}_{\boldsymbol{\pi}}\{h(X)\}$$

   **almost surely as $n \to \infty$, provided $\mathrm{E}_{\boldsymbol{\pi}}\{|h(X)|\}$ exists.**

**This is one form of the *ergodic theorem*, which is a generalization of the strong law of large numbers.**

# Markov chain Monte Carlo

- **Let $\left\{\mathbf{X}^{(t)}\right\}$ denote a Markov chain for $t = 0, 1, 2, \ldots$, where $\mathbf{X}^{(t)} = \left(X_1^{(t)}, \ldots, X_p^{(t)}\right)$ and the state space is either continuous or discrete.**
- **The MCMC sampling strategy is to construct an irreducible, aperiodic Markov chain for which the stationary distribution equals the target distribution, $f$.**
- **For sufficiently large $t$, a realization, $\mathbf{X}^{(t)}$, from this chain will have approximate marginal distribution $f$.**
- **There are various strategies for constructing a MCMC algorithm including the Metropolis-Hastings algorithm, the Gibbs sampler, etc.**
- **A popular application of MCMC methods is to facilitate Bayesian inference where $f$ is a Bayesian posterior distribution for parameters $\mathbf{X}$**

## Metropolis-Hastings algorithm

**The method begins at $t = 0$ with the selection of $\mathbf{X}^{(0)} = \mathbf{x}^{(0)}$ drawn at random from some starting distribution, $g$, with the requirement that $f\left(\mathbf{x}^{(0)}\right) > 0$.**

**Given $\mathbf{X}^{(t)} = \mathbf{x}^{(t)}$, the algorithm generates $\mathbf{X}^{(t+1)}$ as follows.**

1. **Sample a candidate value, $\mathbf{X}^*$, from a *proposal distribution* $g\left(\cdot \middle| \mathbf{x}^{(t)}\right)$.**
2. **Compute the *Metropolis-Hastings ratio*, $R\left(\mathbf{x}^{(t)}, \mathbf{X}^*\right)$, where**

$$R(\mathbf{u}, \mathbf{v}) = \frac{f(\mathbf{v})\, g(\mathbf{u} | \mathbf{v})}{f(\mathbf{u})\, g(\mathbf{v} | \mathbf{u})}. \tag{92}$$

3. **Sample a value for $\mathbf{X}^{(t+1)}$ according to the following:**

$$\mathbf{X}^{(t+1)} = \begin{cases} \mathbf{X}^* & \text{with probability } \min\left\{R\left(\mathbf{x}^{(t)}, \mathbf{X}^*\right), 1\right\} \\ \mathbf{x}^{(t)} & \text{otherwise.} \end{cases} \tag{93}$$

4. **Increment $t$ and return to step 1.**

# Metropolis-Hastings algorithm

**A few details:**

- **If the proposal distribution is chosen so that the chain is irreducible and aperiodic, then the unique limiting stationary distribution of the chain generated by the Metropolis-Hastings algorithm is the target distribution.**

- **Sequence $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots$ will likely include multiple copies of some points in the state space.**

  **It is important to include these copies in the chain and in any estimates based on the output, since the frequencies of sampled points are used to correct for the fact that the proposal density differs from the target density.**

# Metropolis-Hastings algorithm

**The distribution of realizations from the Metropolis-Hastings chain approximates the stationary distribution of the chain as $t$ progresses, therefore**

$$\mathbf{E}\left\{h(\mathbf{X})\right\} \approx \frac{1}{n}\sum_{i=1}^{n} h\left(\mathbf{x}^{(i)}\right)$$

**Some useful quantities that can be estimated include**

- **means:** $\mathrm{E}\left\{\mathrm{h}(\mathbf{X})\right\}$
- **variances:** $\mathrm{E}\left\{(\mathrm{h}(\mathbf{X}) - \mathrm{E}\{\mathrm{h}(\mathbf{X})\})^2\right\}$
- **tail probabilities:** $\mathbf{E}\left\{1_{\{h(\mathbf{X})\leq q\}}\right\}$ **for constant** $q$**, where** $1_{\{A\}} = 1$ **if** $A$ **is true and 0 otherwise**
- **Virtually any other statistic that may be of interest!**

**Can also estimate** $f$ **itself using density estimation methods**

# Choosing a proposal distribution

**A well chosen proposal distribution produces**

- **candidate values that cover the support of the stationary distribution in a reasonable number of iterations**
- **candidate values that are not accepted or rejected too frequently**

**Both of these factors are related to the spread of the proposal distribution:**

- **Too diffuse: candidate values will be rejected frequently and the chain will require many iterations to adequately explore the space of the target distribution**
- **Too focused: the chain will remain in one small region of the target distribution for many iterations while other regions of the target distribution will not be adequately explored**

## Independence chains

**Suppose that the proposal distribution for the Metropolis-Hastings algorithm is chosen such that for some fixed density $g$**

$$g\left(\mathbf{x}^* \middle| \mathbf{x}^{(t)}\right) = g(\mathbf{x}^*)$$

**This yields an independence chain, where each candidate value is drawn independently of the past.**

**In this case, the Metropolis-Hastings ratio is**

$$R\left(\mathbf{x}^{(t)}, \mathbf{X}^*\right) = \frac{f\left(\mathbf{X}^*\right) g\left(\mathbf{x}^{(t)}\right)}{f\left(\mathbf{x}^{(t)}\right) g\left(\mathbf{X}^*\right)}$$

**The resulting Markov chain is irreducible and aperiodic if $g(\mathbf{x}) > 0$ whenever $f(\mathbf{x}) > 0$.**

## Bayesian inference and MCMC

**In the Bayesian paradigm, parameters are assumed to follow some probability distribution. Suppose that data Y has a distribution parameterized by $\boldsymbol{\theta}$.**

**Bayes Theorem:**

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{y}) &= \frac{p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})}{\int p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})\, d\boldsymbol{\theta}} \\ &= \frac{p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})}{p(\mathbf{y})} \\ &\propto p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta}) \end{aligned}$$

**where**

- $p(\boldsymbol{\theta}|\mathbf{y})$ **is the** *posterior density* **of** $\boldsymbol{\theta}$
- $p(\mathbf{y}|\boldsymbol{\theta})$ **is the** *likelihood* **or sampling distribution**
- $p(\boldsymbol{\theta})$ **is the** *prior density*

**In Bayesian applications we use MCMC to simulate from the posterior distribution $p(\boldsymbol{\theta}|\mathbf{y})$, so the target distribution is typically the posterior distribution.**

**Simple strategy: use the prior as a proposal distribution in an independence chain.**
**In our Metropolis-Hastings notation,**

$$f(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\mathbf{y})$$

**and**

$$g(\boldsymbol{\theta}^*) = p(\boldsymbol{\theta}^*).$$

**Then the Metropolis-Hastings ratio is given by:**

$$R\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^*\big) = \frac{p(\mathbf{y}|\,\boldsymbol{\theta}^*)}{p\big(\mathbf{y}\big|\boldsymbol{\theta}^{(t)}\big)}$$

**Comments:**

- **If proposal distribution= prior distribution, the Metropolis-Hastings ratio equals the likelihood ratio.**
- **By definition, the support of the prior covers the support of the target posterior, so the stationary distribution of this chain is the desired posterior.**
- **More sophisticated algorithms often have better performance.**

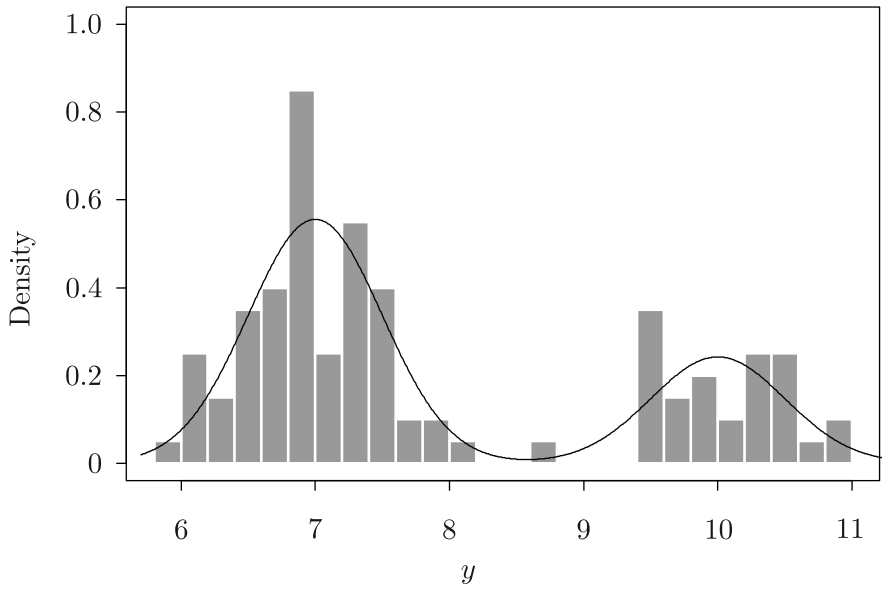
# MCMC Example 1: Estimating a mixture parameter

**Suppose we have observed data $y_1, y_2, \ldots, y_{100}$ sampled independently and identically distributed from the mixture distribution**

$$\delta N(7, 0.5^2) + (1 - \delta) N(10, 0.5^2) \tag{94}$$

- **Mixture densities are common in real-life applications where, for example, the data may come from more than one population**
- **We will use MCMC techniques to construct a chain whose stationary distribution equals the posterior density of $\delta$ assuming a Unif(0,1) prior distribution for $\delta$**

# MCMC Example 1: Estimating a mixture parameter




**Histogram of 100 observations simulated from the mixture distribution (94). The data were generated with $\delta = 0.7$, so we should find that the posterior density is concentrated in this area.**

# MCMC Example 1: Estimating a mixture parameter

**Try two different MCMC proposal densities:**

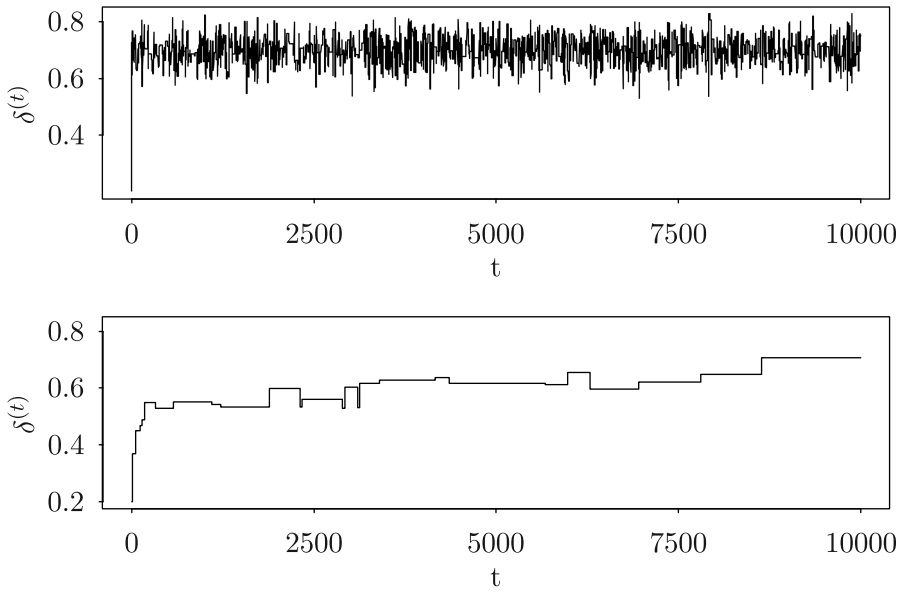
1. **Beta(1,1): equivalent to a Unif(0,1) distribution**
2. **Beta(2,10):**
   - **skewed right with mean of 0.167**
   - **values of $\delta$ near 0.7 are unlikely to be generated from the proposal distribution**

**Generated 10,000 iterations of Metropolis-Hastings algorithm for each proposal.**

**Some output:**

- **Sample paths:**
  - **a plot of the chain realizations, $\delta^{(t)}$, against the iteration number, $t$**
  - **useful for investigating the behavior of the Markov chain**
- **Histogram of realizations**

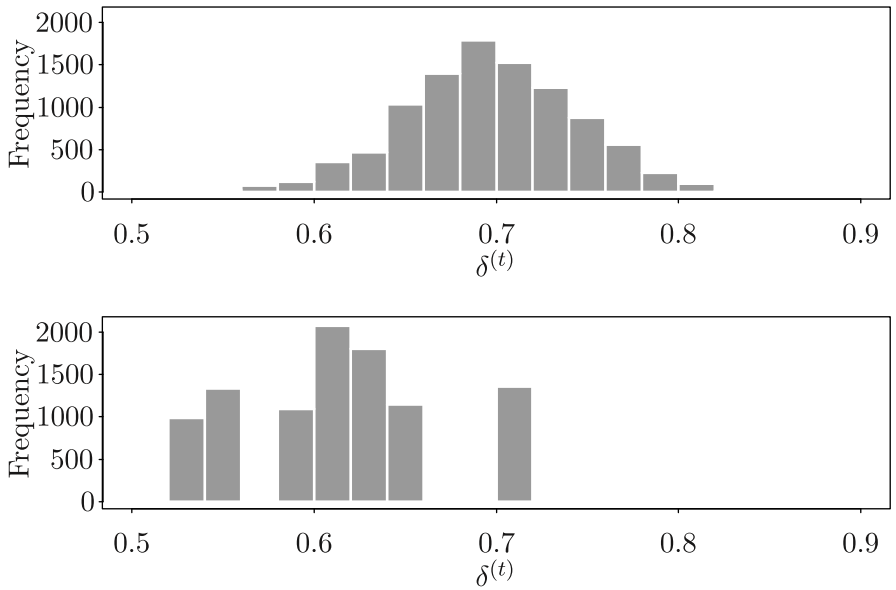
# MCMC Example 1: Estimating a mixture parameter




**Sample paths for $\delta$ from independence chains with proposal densities Beta(1,1) (top) and Beta(2,10) (bottom)**

**Burn-in period: Since initial iterates retain strong dependence on the starting value, eliminate them in estimation**

# MCMC Example 1: Estimating a mixture parameter




**Histograms of $\delta^{(t)}$ for iterations 201-10,000 of independence chains with proposal densities Beta(1,1) (top) and Beta(2,10) (bottom)**

## Random walk chains

**Random walk chain: Let $\mathbf{X}^*$ be generated by drawing**

$$\boldsymbol{\epsilon} \sim h(\boldsymbol{\epsilon})$$

**for some density $h$ and then setting**

$$\mathbf{X}^* = \mathbf{x}^{(t)} + \boldsymbol{\epsilon}.$$

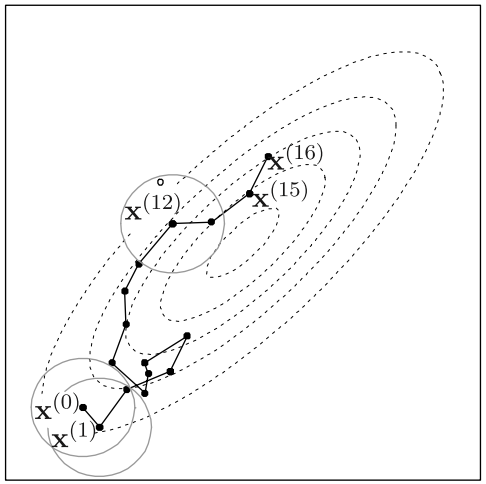
**Note that the proposal is then given by**

$$g\left(\mathbf{x}^* | \, \mathbf{x}^{(t)}\right) = h\left(\mathbf{x}^* - \mathbf{x}^{(t)}\right)$$

**Common choices for $h$:**

- **uniform distribution over a ball centered at the origin**
- **scaled standard normal distribution**
- **scaled Student's $t$ distribution**

# Random walk chains




**Hypothetical random walk chain for sampling a two dimensional target distribution (dotted contours) using proposed increments sampled uniformly from a disk centered at the current value.**

# Gibbs Sampling

- **Specifically adapted for multidimensional target distributions.**
- **Goal: still to construct a Markov chain whose stationary distribution—or some marginalization thereof—equals the target distribution, $f$**
- **Sequentially samples from univariate conditional distributions, which are often available in closed form**

## Basic Gibbs sampler

**Suppose it is easy to sample from the univariate conditional distributions:**

$$X_i|X_1,\ldots,X_{i-1},X_{i+1},\ldots,X_p \sim f\left(x_i|\, x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_p\right)$$

**A basic Gibbs sampler:**

1. **Select starting values $\mathbf{x}^{(0)}$ and set $t = 0$.**
2. **Generate, in turn,**

$$\begin{aligned}
X_1^{(t+1)}\big|\cdot &\sim f\left(x_1\big|x_2^{(t)},\ldots,x_p^{(t)}\right)\\
X_2^{(t+1)}\big|\cdot &\sim f\left(x_2\big|x_1^{(t+1)},x_3^{(t)},\ldots,x_p^{(t)}\right)\\
&\vdots\\
X_{p-1}^{(t+1)}\big|\cdot &\sim f\left(x_{p-1}\big|x_1^{(t+1)},x_2^{(t+1)},\ldots,x_{p-2}^{(t+1)},x_p^{(t)}\right)\\
X_p^{(t+1)}\big|\cdot &\sim f\left(x_p\big|x_1^{(t+1)},x_2^{(t+1)},\ldots,x_{p-1}^{(t+1)}\right)
\end{aligned}$$

   **where $|\cdot$ denotes conditioning on the most recent updates to all other elements of X.**

3. **Increment $t$ and go to step 2.**

# Notes and enhancements to the Gibbs sampler

- **The completion of step 2 for all components of X is called a *cycle***
- **Can reorder the conditional distribution updates with each cycle**
- **Can block or group elements of X and update them jointly.**
  **For example when $p = 4$, perhaps it is useful to consider**

$$X_2^{(t+1)}, X_3^{(t+1)} \Big| \cdot \sim f\left(x_2, x_3 \Big| x_1^{(t+1)}, x_4^{(t)}\right)$$

- **Can show Gibbs sampling is a Metropolis-Hastings algorithm with time inhomogeneous proposal distributions and with Metropolis-Hasting ratio $R$ always equal to 1.**

# Hybrid Gibbs sampling

**A *hybrid MCMC* algorithm might proceed with the following sequence of updates with $p = 6$:**

1. **Update** $X_1^{(t+1)} \Big| \left( x_2^{(t)}, x_3^{(t)}, x_4^{(t)}, x_5^{(t)}, x_6^{(t)} \right)$ **with a Gibbs step.**
2. **Update** $\left( X_2^{(t+1)}, X_3^{(t+1)} \right) \Big| \left( x_1^{(t+1)}, x_4^{(t)}, x_5^{(t)}, x_6^{(t)} \right)$ **with a Metropolis step.**
3. **Update** $X_4^{(t+1)} \Big| \left( x_1^{(t+1)}, x_2^{(t+1)}, x_3^{(t+1)}, x_5^{(t)}, x_6^{(t)} \right)$ **with a step from a random walk chain.**
4. **Update** $\left( X_5^{(t+1)}, X_6^{(t+1)} \right) \Big| \left( x_1^{(t+1)}, x_2^{(t+1)}, x_3^{(t+1)}, x_4^{(t+1)} \right)$ **with a Gibbs step.**

**The Metropolis-Hastings steps within a Gibbs algorithm are typically useful when the univariate conditional density for one or more elements of X is not available in closed form.**

# MCMC algorithm implementation

**All MCMC algorithms described above have the correct limiting stationary distribution.**

**Mixing:**

- **How quickly does the chain forget its starting value?**
- **How quickly does the chain fully explore the support of the target distribution?**
- **How far apart in a sequence do observations need to be before they can be considered to be approximately independent?**

**Stationarity: has the chain has run sufficiently long so that**

- **it is reasonable to believe that the output adequately represents the target distribution?**
- **the output can be used reliably for estimation?**

# Ensuring good mixing and convergence

**Strategies and diagnostics**

1. **Number of chains**
2. **Simple graphs to assess mixing and convergence**
3. **Reparameterization**
4. **Burn-in and run length**

# 1. Number of chains

**A difficult problem to diagnose: chain has become stuck in one or more modes of the target distribution.**

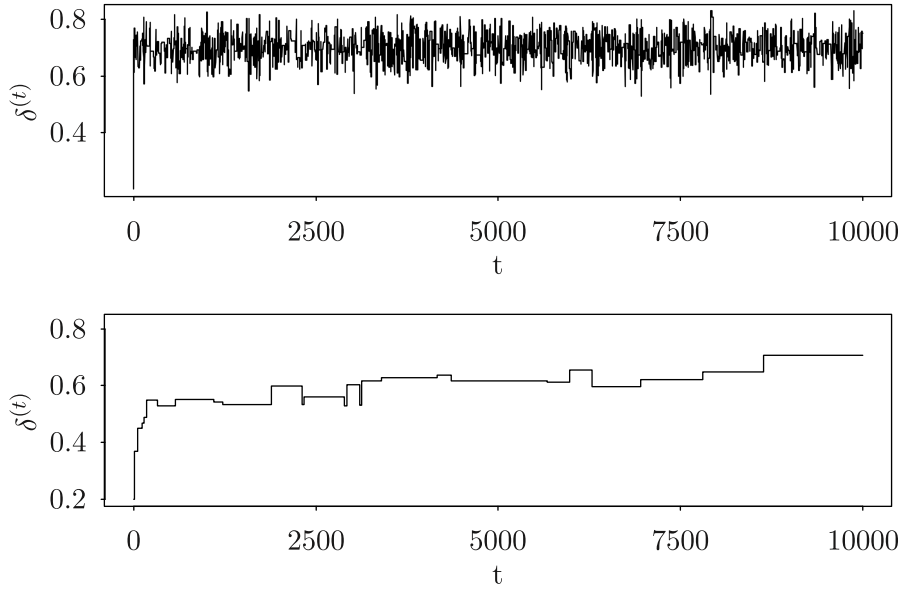
**Partial solution: run multiple chains from diverse starting values and then compare the within- and between-chain behavior.**

**Diagnostics: sample path plots and Gelman-Rubin statistic**

## 2. Simple graphs to assess mixing and convergence

**Sample paths: plot of the iteration number versus the realizations of $\mathbf{X}^{(t)}$ for $t = 0, 1, \ldots$**

- **also called trace or history plots**
- **Want to see a very wiggly line that (perhaps) moves away from the starting value.**



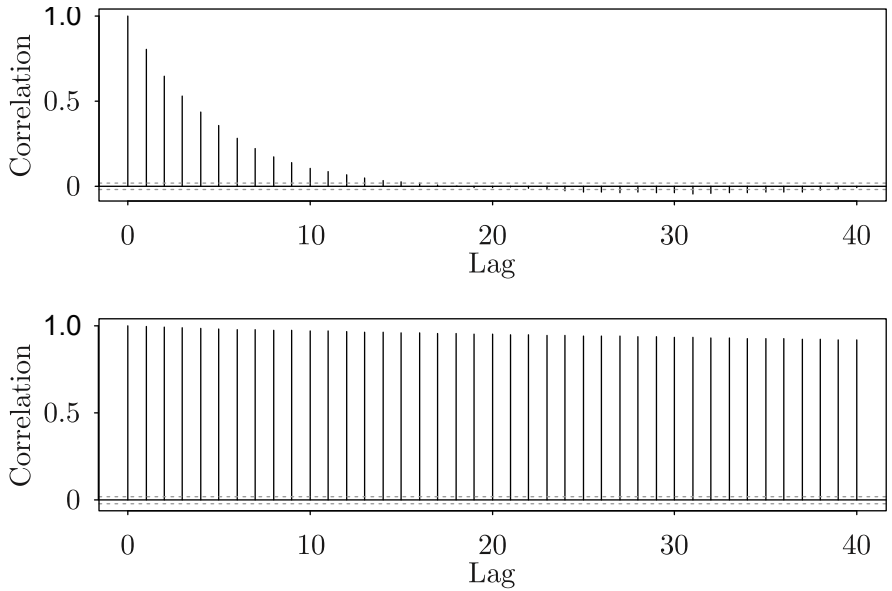

**Sample paths for $\delta$ from independence chains with proposal densities Beta(1,1) (top) and Beta(2,10) (bottom)**

## 2. Simple graphs to assess mixing and convergence

**autocorrelation function (ACF) plot:**

- **A plot of the lag versus the correlation between iterates at that lag.**
- **Slow decay in the acf suggests poor mixing.**
- **Multi-parameter problems:**
  - **examine cross-correlations between parameters that might be related**
  - **High cross-correlations may indicate poor mixing of the chain**

# 2. Simple graphs to assess mixing and convergence




**Autocorrelation plots for example on estimating a mixture parameter with with proposal densities Beta(1,1) (top) and Beta(2,10) (bottom).**

# 3. Reparameterization

- **The mixing rates of both the Gibbs sampler and Metropolis-Hastings algorithms can be improved via reparameterization of the model.**
- **High correlation between the elements of X can lead to slow convergence. Reparameterization of the model can reduce the correlation and thus speed convergence.**

**Reparameterization approaches are typically adapted for specific models:**

- **If there are continuous covariates in a linear model, center and scale the covariates to reduce correlations between the parameters in the model.**
- **Hierarchical centering: parameters in a hierarchical Bayesian model are moved down the hierarchy. Particularly useful in models with random effects.**

# 4. Burn-in and run length

**Burn-in**

- **Initial MCMC iterates will not have exactly the correct marginal distribution, and the dependency on the initial point (or distribution) from which the chain was started may remain strong.**
- **To reduce the severity of this problem, the first several thousand values from the chain are typically discarded.**

**Gelman and Rubin statistic, $R$**

- **Compares the output of several chains**
- **If within and between chain variances are similar in magnitude, suggests that the chains are stationary**
- **Roughly: want $\sqrt{R} < 1.2$**
- **If not: increase run-length, reparameterize the model, etc.**

## 4. Burn-in and run length: Gelman and Rubin statistic

**Run the MCMC algorithm $J$ separate times to produce equal-length chains ($J \geq 2$) with starting values dispersed over the support of the target density.**

**Let**

- **$L$ denote the length of each chain after discarding $D$ burn-in iterates.**
- **The variable of interest is $X$, and its value at the $t$th iteration of the $j$th chain is $x_j^{(t)}$.**
- **For the $j$th chain, the $D$ values $x_j^{(0)}, \ldots, x_j^{(D-1)}$ are discarded and the $L$ values $x_j^{(D)}, \ldots, x_j^{(D+L-1)}$ are retained.**

# 4. Burn-in and run length: Gelman and Rubin statistic

**Let**

$$\bar{x}_j = \frac{1}{L}\sum_{t=D}^{D+L-1} x_j^{(t)} \text{ and } \bar{x}_. = \frac{1}{J}\sum_{j=1}^{J} \bar{x}_j$$

**and define the between-chain variance as**

$$B = \frac{L}{J-1}\sum_{j=1}^{J} (\bar{x}_j - \bar{x}_.)^2$$

**Now if the within-chain variance for the $j$th chain is** $s_j^2 = \frac{1}{L-1}\sum_{t=D}^{D+L-1}\left(x_j^{(t)} - \bar{x}_j\right)^2$**, then let**

$$W = \frac{1}{J}\sum_{j=1}^{J} s_j^2$$

**represent the mean of the $J$ within-chain estimated variances.**

**Finally, let**

$$R = \frac{\frac{L-1}{L}W + \frac{1}{L}B}{W}.$$

## 4. Burn-in and run length: Gelman and Rubin statistic

- **If all the chains are stationary, then both the numerator and the denominator should estimate the marginal variance of $X$.**
- **If there are notable differences between the chains, then the numerator will exceed the denominator.**
- **As $L \to \infty$, $\sqrt{R} \to 1$.**
- **In practice, some authors suggest that $\sqrt{R} < 1.2$ is acceptable.**

**Remedies:**

- **If the chosen burn-in period did not yield an acceptable result, then $D$ should be increased, $L$ should be increased, or preferably both.**
- **If the $x_j^{(t)}$ iterates are transformed so that their distribution is approximately normal, the performance of this diagnostic is improved.**
- **Reparameterize the model and rerun the chain.**

# MCMC software

**WinBUGS:**

- **WinBUGS is free software to carry out Gibbs sampling (www.mrc-bsu.cam.ac.uk/bugs/)**
- **Programming your own MCMC algorithm is the best way to learn what is going on in MCMC, but WinBUGS makes life a lot easier.**
- **Now available: code to run WinBUGS from within R**

**Other software available in R and other languages, but WinBUGS is the current popular choice**

# Output of MCMC procedures

**Marginalization: If $\{\mathbf{X}^{(t)}\}$ represents a $p$-dimensional Markov chain, then the limiting distribution of $\{X_i^{(t)}\}$ is the $i$th marginal of $f$. If you are focused only on a property of this marginal, discard the rest of the simulation and analyze the realizations of $X_i^{(t)}$.**

**Mean: The most commonly used estimator is based on an empirical average. Discard the burn-in; then calculate the desired statistic by taking**

$$\frac{1}{L}\sum_{t=D}^{D+L-1} h\left(\mathbf{X}^{(t)}\right)$$

**Standard deviation: sample standard deviation of realizations (after burn-in)**

**Probability estimates: The probability of any event can be estimated by the frequency of that event in the chain.**

# Output of MCMC procedures

**Monte Carlo standard error: There are several ways to compute this.**

*Batch Method*:

1. **Separate the realizations into batches with, say, 50 consecutive iterations in each batch.**
2. **Compute the mean of each batch.**
3. **The estimated standard error is the standard deviation of these means divided by the square root of the number of batches.**

**Density/histogram:**

- **Histogram of the raw iterates.**
- **A kernel density estimate of the marginal posterior distributions.**

**Quantiles: Compute empirical quantiles**

**Other summary statistics: Be creative!**

# Practical implementation advice

**Some rules of thumb:**

- **Mixing: For a Metropolis algorithm with normal target and proposal distributions, it has been suggested that a user should aim for acceptance rates of approximately**
  - **45% for one- or two-dimensional problems**
  - **23% for higher-dimensional problems**
- **Burn-in lengths: 0-50,000**
- **Chain lengths: 5,000-5,000,000**
  - **More samples means more accurate estimates of the posterior distribution.**
  - **One rule of thumb: the simulation should be run until the Monte Carlo error for each parameter of interest is less than 5% of the sample standard deviation.**
  - **Chain lengths will continue to increase as computing power grows.**

# MCMC Example 2: Fur seal pup capture-recapture analysis

- **After centuries of severe population reductions due to commercial and subsistence hunting, the abundance of fur seals in New Zealand has been increasing in recent years.**
- **Our goal is to estimate the number of pups in a fur seal colony using a capture-recapture approach.**
- **Estimation: can estimate population size and survival rates, etc.**
- **Consider, for example, high recapture rates suggest that the true population size does not greatly exceed the total number of unique individuals ever captured.**

# MCMC Example 2: Fur seal pups

**Let**

- $N$ **be the unknown population size to be estimated**
- $I$ **be the number of census attempts**
- $\mathbf{c} = (c_1, \ldots, c_I)$ **be the total number of captures (including recaptures) at each attempt**
- $r$ **be the total number of distinct animals captured during the study**

**We assume that the population is closed during the period of the sampling which means that deaths, births and migrations are inconsequential during this period.**

# MCMC Example 2: Fur seal pups

- **We consider a model with separate, unknown capture probabilities for each census effort, $\alpha = (\alpha_1, \ldots, \alpha_I)$.**
- **This model assumes that all animals are equally catchable on any one capture occasion, but capture probabilities may change over time.**

**The likelihood for this model is**

$$L\left(N, \boldsymbol{\alpha} | \mathbf{c}, r\right) \propto \frac{N!}{(N-r)!} \prod_{i=1}^{I} \alpha_i^{c_i} \left(1 - \alpha_i\right)^{N - c_i} \tag{95}$$

**This model is sometimes called the $M(t)$ model**

# MCMC Example 2: Fur seal pups

- **In a study conducted on the Otago Peninsula on the South Island of New Zealand, fur seal pups were marked and released during $I = 7$ census attempts during one season.**
- **It is reasonable to assume the population of pups was closed during the study period.**
- **$r = \sum_{i=1}^{7} m_i = 84$ unique fur seals were observed during the sampling period.**

**Number of pups captured and recaptured during 7 census efforts in one season**

| | | Census attempt, $i$ | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| **Number captured** | $c_i$ | **30** | **22** | **29** | **26** | **31** | **32** | **35** |
| **Number newly caught** | $m_i$ | **30** | **8** | **17** | **7** | **9** | **8** | **5** |

# MCMC Example 2: Fur seal pups, priors

**For estimation, one might adopt a hierarchical Bayesian framework where $N$ and $\boldsymbol{\alpha}$ are assumed to be a priori independent with the following priors:**

- **$N$: noninformative Jeffreys prior $f(N) \propto 1/N$**
- **capture probabilities: $f(\alpha_i|\theta_1, \theta_2) = \text{Beta}(\theta_1, \theta_2)$ for $i = 1, \ldots, 7$, assumed to be a priori exchangeable**

**Previous analyses with the $M(t)$ model have indicated sensitivity to the prior distribution for the capture probabilities. To mitigate this sensitivity, we introduce a hyperprior for $(\theta_1, \theta_2)$:**

$$f(\theta_1, \theta_2) \propto \exp\{-(\theta_1 + \theta_2)/1000\},$$

**with $(\theta_1, \theta_2)$ assumed to be a priori independent of the remaining parameters.**

# MCMC Example 2: Fur seal pups, Gibbs

**Goal:**

$$p(N, \alpha_1, \ldots, \alpha_7, \theta_1, \theta_2 | \mathbf{c}, r) \propto p(\mathbf{c}, \mathbf{m} | N, \alpha_1, \ldots, \alpha_7, \theta_1, \theta_2)$$
$$p(N) \left[ \prod_{i=1}^{7} p(\alpha_i | \theta_1, \theta_2) \right] p(\theta_1, \theta_2)$$

**A Gibbs sampler can then be constructed by simulating from the conditional posterior distributions**

$$N - 84 | \cdot \sim \textbf{NegBin}\left(84, 1 - \prod_{i=1}^{7} (1 - \alpha_i)\right)$$
$$\alpha_i | \cdot \sim \textbf{Beta}(c_i + \theta_1, N - c_i + \theta_2) \quad \textbf{for } i = 1, \ldots, 7$$
$$\theta_1, \theta_2 | \cdot \sim k \left[ \frac{\Gamma(\theta_1 + \theta_2)}{\Gamma(\theta_1)\Gamma(\theta_2)} \right]^7 \prod_{i=1}^{7} \alpha_i^{\theta_1} (1 - \alpha_i)^{\theta_2} \exp\{-(\theta_1 + \theta_2)/1000\}$$

**where $|\cdot$ denotes conditioning on the remaining parameters and the data, and $k$ is an unknown constant.**

# MCMC Example 2: Fur seal pups, Gibbs

**Problem: it is difficult to produce a chain for $(\theta_1, \theta_2)$ with adequate mixing and convergence behavior.**

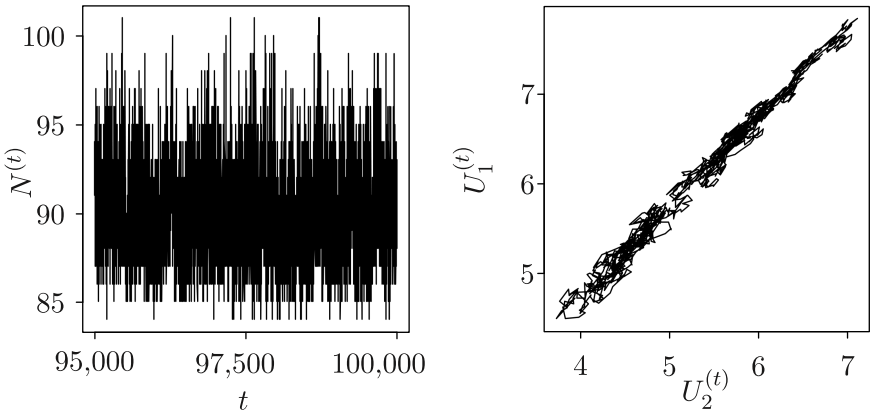
**To improve performance, transform $(\theta_1, \theta_2)$ to**

$$\mathbf{U} = (U_1, U_2) = (\log \theta_1, \log \theta_2).$$

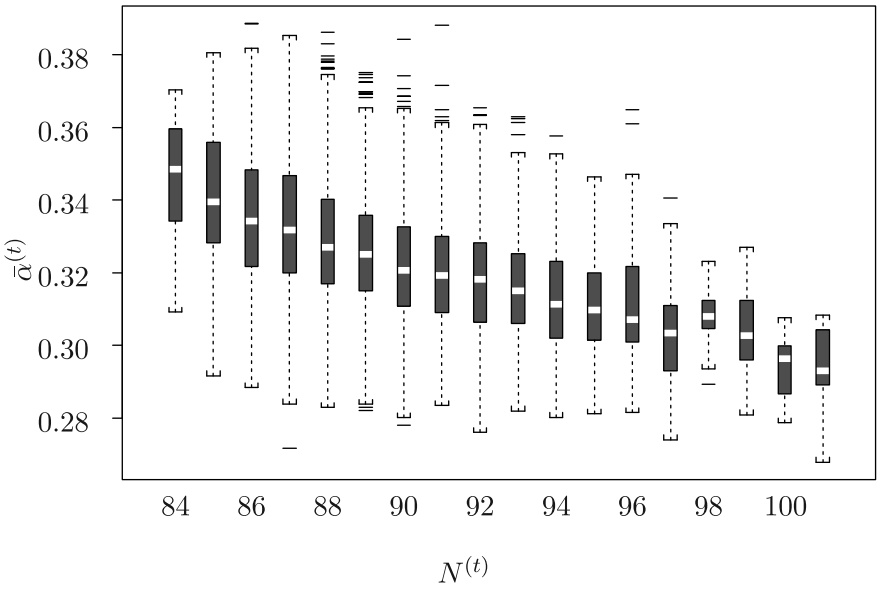
**This requires transformation of the conditional densities.**

# MCMC Example 2: Fur seal pups, Gibbs

- **The results below are based on a chain of 100,0000 iterations with the first 1,000 iterations discarded for burn-in.**
- **Based on five runs of 100,000 iterations each, the Gelman and Rubin statistic for $N$ is equal to 1.00047 which suggests the $N^{(t)}$ chain is roughly stationary.**
- **Sample paths for $N$ (left panel) and U (right panel) for final 5,000 iterations in the seal pup example.**

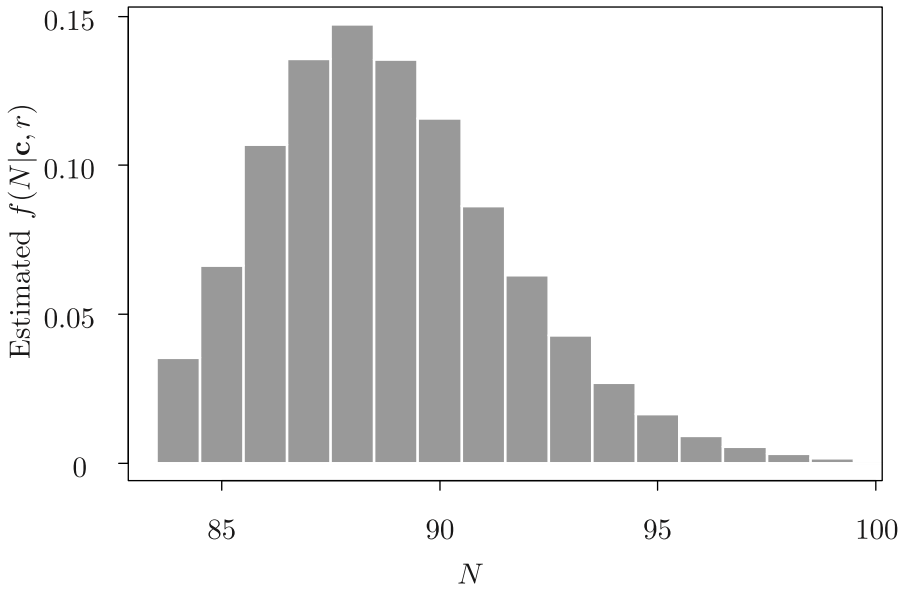


# MCMC Example 2: Fur seal pups, Gibbs




**Split boxplot of $\bar{\alpha}^{(t)}$ against $N^{(t)}$ for the seal pup example.**

# MCMC Example 2: Fur seal pups, Gibbs




**Estimated marginal posterior probabilities for $N$ for the seal pup example.**

**The posterior mean of $N$ is 90 with a 95% highest posterior density interval of (84, 95).**

# Part II: Monte Carlo Integration Methods


6. Introduction to integration

7. Monte Carlo Integration

8. Markov Chain Monte Carlo Methods

9. Advanced MCMC Methods

**Slice sampling and other auxiliary variable methods, Reversible jump MCMC, Perfect sampling, Langevin Metropolis-Hastings Algorithm, Hit-and-run algorithm, Multiple-try Metropolis-Hastings Algorithm**

10. Integration: Concluding remarks

# Slice sampling and other auxiliary variable methods

- **In some cases standard MCMC methods can take too long to mix properly to be of practical use**
- **One remedy: augment the state space of the variable of interest**
- **Auxiliary variable methods can lead to chains which mix faster and require less tuning than standard MCMC methods**

**Consider a target distribution $f$ which can be evaluated but not easily sampled. Construct an auxiliary variable algorithm as follows:**

1. **Augment the state space of $\mathbf{X}$ by the state space of a vector of auxiliary variables, $\mathbf{U}$.**
2. **Construct a Markov chain over the joint state space of $(\mathbf{X}, \mathbf{U})$ having stationary distribution $(\mathbf{X}, \mathbf{U}) \sim f(\mathbf{x}, \mathbf{u})$ which marginalizes to the target $f(\mathbf{x})$.**
3. **When simulation has been completed, discard $\mathbf{U}$ realizations and base inferences only on the marginal distribution of $\mathbf{X}$.**

## Slice sampling

**Consider MCMC for a univariate variable $X \sim f(x)$, and suppose that it is impossible to sample directly from $f$.**

- **Introducing any univariate auxiliary variable $U$ would allow us to consider a target density for $(X, U) \sim f(x, u)$.**
- **Writing $f(x, u) = f(x)f(u|x)$ suggests an auxiliary variable Gibbs sampling strategy that alternates between updates for $X$ and $U$.**
- **The trick is to choose a $U$ variable that speeds MCMC mixing for $X$.**

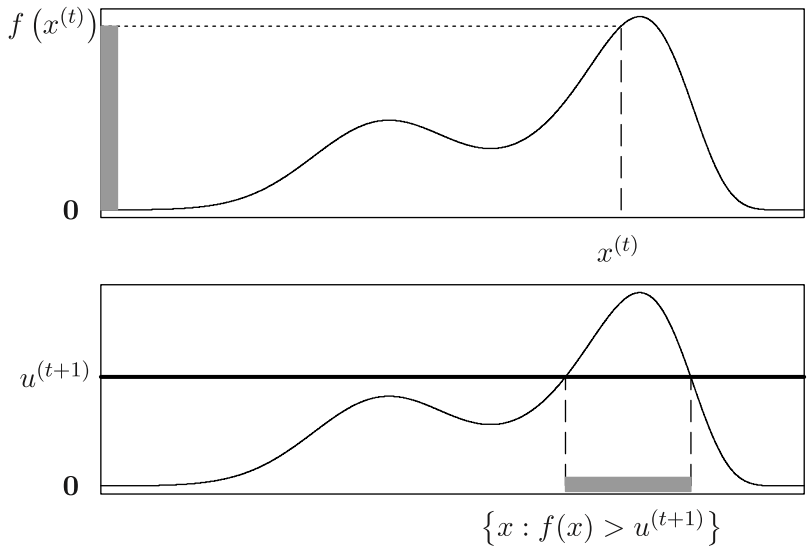
**At iteration $t+1$ of the slice sampler we alternately generate $X^{(t+1)}$ and $U^{(t+1)}$ according to**

$$U^{(t+1)} \Big| x^{(t)} \sim \textbf{Unif}\left(0, f\left(x^{(t)}\right)\right), \tag{96}$$

$$X^{(t+1)} \Big| u^{(t+1)} \sim \textbf{Unif}\left\{x : f(x) \geq u^{(t+1)}\right\}. \tag{97}$$

**The sampling in (97) can be challenging if $f$ is not invertible. One approach to implementing equation (97) is to use rejection sampling.**

# Illustration of slice sampling

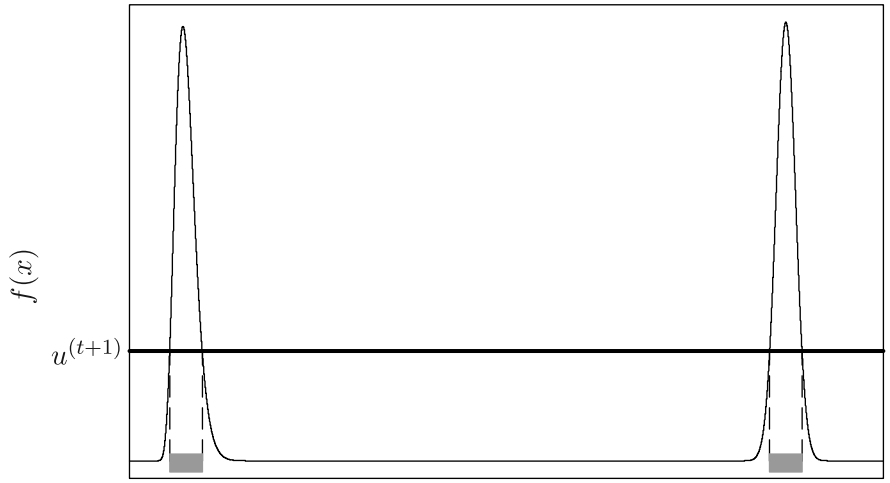



**At iteration $t+1$, the algorithm starts at $x^{(t)}$.**

**Upper panel: Draw $U^{(t+1)} \sim \text{Unif}\big(0, f\left(x^{(t)}\right)\big)$. In the top panel this corresponds to sampling along the vertical shaded strip.**

**Lower panel: Draw $X^{(t+1)} \big| \big(U^{(t+1)} = u^{(t+1)}\big)$ uniformly from the set of $x$ values for which $f(x) \geq u^{(t+1)}$. In the lower panel this corresponds to sampling along the horizontal shaded strip.**

# Example: univariate multimodal target distribution




**Generate samples from this multimodal target distribution using a**

- **Standard Metropolis–Hastings algorithm: algorithm may find one mode of the distribution, but it may take many iterations to find the other mode.**
- **Slice sampler: The horizontal shaded areas indicate the set defined in (97) from which $X^{(t+1)}\big|u^{(t+1)}$ is uniformly drawn. Hence the slice sampler will have about a 50% chance of switching modes each iteration. Therefore the slice sampler will mix much better with many fewer iterations required.**

# Reversible jump MCMC (RJMCMC)

- **Standard MCMC algorithms require that the dimensionality of $\mathbf{X}^{(t)}$ and the interpretation of the elements of $\mathbf{X}^{(t)}$ do not change with $t$**
- **In many applications, it may be of interest to develop a chain that allows for changes in the dimension of the parameter space from one iteration to the next.**
- **Examples where RJMCMC methods are useful:**
  - **model selection**
  - **selection of the number of components in a mixture distribution**
  - **knot selection in nonparametric regression**

# RJMCMC: Notation

**Consider constructing a Markov chain to explore a space of candidate models, each of which might be used to fit observed data y. Let**

- $\mathcal{M}_1, \ldots, \mathcal{M}_K$ **= a collection of models under consideration.**
- $\boldsymbol{\theta}_m$ **= parameters in the** $m$**th model.**
- $p_m$ **= number of parameters in the** $m$**th model.**
- **In the Bayesian paradigm, we may envision random variables** $\mathbf{X} = (M, \boldsymbol{\theta}_M)$ **which together index the model and parameterize inference for that model.**

**We may assign prior distributions to these parameters, then seek to simulate from their posterior distribution using a MCMC method for which the** $t$**th random draw is** $\mathbf{X}^{(t)} = \left(M^{(t)}, \boldsymbol{\theta}^{(t)}_{M^{(t)}}\right)$**.**

- $\boldsymbol{\theta}^{(t)}_{M^{(t)}}$**= parameters drawn for the model indexed by** $M^{(t)}$
- $p_{M^{(t)}}$**= dimension** $\boldsymbol{\theta}^{(t)}_{M^{(t)}}$

# RJMCMC: Overview

**The goal of RJMCMC is to generate samples with joint posterior density**

$$f\left(m, \boldsymbol{\theta}_m | \mathbf{y}\right)$$

**This posterior arises from Bayes' theorem via**

$$f\left(m, \boldsymbol{\theta}_m | \mathbf{y}\right) \propto f\left(\mathbf{y} | m, \boldsymbol{\theta}_m\right) f\left(\boldsymbol{\theta}_m | m\right) f(m),$$

**where**

- $f\left(\mathbf{y} | m, \boldsymbol{\theta}_m\right)$ **denotes the density of the observed data under the** $m$**th model and its parameters**
- $f\left(\boldsymbol{\theta}_m | m\right)$ **denotes the prior density for the parameters in the** $m$**th model**
- **A prior weight of** $f(m)$ **is assigned to the** $m$**th model with** $\sum_{m=1}^{K} f(m) = 1$**.**

# RJMCMC: Overview

**The posterior factorization**

$$f\left(m, \boldsymbol{\theta}_m | \mathbf{y}\right) = f\left(m|\mathbf{y}\right) f\left(\boldsymbol{\theta}_m | m, \mathbf{y}\right)$$

**suggests two important types of inference:**

1. $f\left(m|\mathbf{y}\right)$ **is the posterior probability for the** $m$**th model, normalized over all models under consideration**
2. $f\left(\boldsymbol{\theta}_m | m, \mathbf{y}\right)$ **is the posterior density of the parameters in the** $m$**th model.**

**RJMCMC enables the construction of an appropriate Markov chain for** $\mathbf{X} = (M, \boldsymbol{\theta}_M)$ **that jumps between models with parameter spaces of different dimensions.**

**The key to the RJMCMC algorithm is the introduction of auxiliary random variables at times** $t$ **and** $t+1$ **with dimensions chosen so that the augmented variables (namely X and the auxiliary variables) at times** $t$ **and** $t+1$ **have equal dimensions.**

# RJMCMC: Dimension matching

**Consider a problem with two models, 1 and 2, such that:**

- **Model $\mathcal{M}_1$ has $p_1$ parameters**
- **Model $\mathcal{M}_2$ has $p_2$ parameters**
- $p_2 > p_1$

**Moving between models:**

- **Proposing a move from $\mathcal{M}_1$ to $\mathcal{M}_2$:**

  **Generate $\boldsymbol{\theta}_2$ from an invertible deterministic function of both $\boldsymbol{\theta}_1$ and an independent random component $\mathbf{U}_1$. We can write**

$$\boldsymbol{\theta}_2 = \mathbf{q}_{1,2}(\boldsymbol{\theta}_1, \mathbf{U}_1)$$

- **Proposing a move from $\mathcal{M}_2$ to $\mathcal{M}_1$:**

  **Carry out via the inverse transformation, $(\boldsymbol{\theta}_1, \mathbf{U}_1) = \mathbf{q}_{1,2}^{-1}(\boldsymbol{\theta}_2) = \mathbf{q}_{2,1}(\boldsymbol{\theta}_2)$. Note that $\mathbf{q}_{2,1}$ is an entirely deterministic way to propose $\boldsymbol{\theta}_1$ from a given $\boldsymbol{\theta}_2$.**

# RJMCMC: The algorithm

**Assume that the chain is currently visiting model $m^{(t)}$, so the chain is in the state $\mathbf{x}^{(t)} = \left(m^{(t)}, \boldsymbol{\theta}^{(t)}_{m^{(t)}}\right)$. The next iteration of the RJMCMC algorithm can be summarized as follows:**

1. **Sample a candidate model $M^* \big| m^{(t)}$ from a proposal density with conditional density $g(\cdot|m^{(t)})$. The candidate model requires parameters $\boldsymbol{\theta}_{M^*}$ of dimension $p_{M^*}$.**

2. **Given $M^* = m^*$, generate an augmenting variable $\mathbf{U}|\left(m^{(t)}, \boldsymbol{\theta}^{(t)}_{m^{(t)}}, m^*\right)$ from a proposal distribution with density $h(\cdot|m^{(t)}, \boldsymbol{\theta}^{(t)}_{m^{(t)}}, m^*)$. Let**

$$(\boldsymbol{\theta}^*_{m^*}, \mathbf{U}^*) = \mathbf{q}_{t,*}\left(\boldsymbol{\theta}^{(t)}_{m^{(t)}}, \mathbf{U}\right),$$

   **where $\mathbf{q}_{t,*}$ is an invertible mapping from $\left(\boldsymbol{\theta}^{(t)}_{m^{(t)}}, \mathbf{U}\right)$ to $(\boldsymbol{\theta}^*_{M^*}, \mathbf{U}^*)$ and the auxiliary variables have dimensions satisfying $p_{m^{(t)}} + p_{\mathbf{U}} = p_{m^*} + p_{\mathbf{U}^*}$**

3. **For a proposed model, $M^* = m^*$, and the corresponding proposed parameter values $\boldsymbol{\theta}^*_{m^*}$, compute the Metropolis–Hastings ratio given by**

$$\frac{f(m^*, \boldsymbol{\theta}^*_{m^*}|\mathbf{y})g(m^{(t)}|m^*)h(\mathbf{u}^*|m^*, \boldsymbol{\theta}^*_{m^*}, m^{(t)})}{f(m^{(t)}, \boldsymbol{\theta}^{(t)}_{m^{(t)}}|\mathbf{Y})g(m^*|m^{(t)})h(\mathbf{u}|m^{(t)}, \boldsymbol{\theta}^{(t)}_{m^{(t)}}, m^*)} \left|\mathbf{J}(t)\right|, \tag{98}$$

**where**

$$\mathbf{J}(t) = \left.\frac{d\mathbf{q}_{t,*}(\boldsymbol{\theta}, \mathbf{u})}{d(\boldsymbol{\theta}, \mathbf{u})}\right|_{(\boldsymbol{\theta},\mathbf{u})=\left(\boldsymbol{\theta}^{(t)}_{m^{(t)}},\mathbf{U}\right)}. \tag{99}$$

**Accept the move to the model $M^*$ with probability equal to the minimum of 1 and the expression in (98). If the proposal is accepted, set $\mathbf{X}^{(t+1)} = (M^*, \boldsymbol{\theta}^*_{M^*})$. Otherwise, reject the candidate draw and set $\mathbf{X}^{(t+1)} = \mathbf{x}^{(t)}$.**

**The last term in (98) is the absolute value of the determinant of the Jacobian matrix arising from the change of variables from $\left(\boldsymbol{\theta}^{(t)}_{m^{(t)}}, \mathbf{U}\right)$ to $(\boldsymbol{\theta}^*_{m^*}, \mathbf{U}^*)$. If $p_{M^{(t)}} = p_{M^*}$, then (98) simplifies to the standard Metropolis–Hastings ratio (92).**

4. **Discard U and U$^*$. Return to step 1.**

# RJMCMC: Simple example with two models

**Consider a problem with $K = 2$ possible models:**

- **Model $\mathcal{M}_1$ has a one-dimensional parameter space $\boldsymbol{\theta}_1 = \alpha$**
- **Model $\mathcal{M}_2$ has a two-dimensional parameter space $\boldsymbol{\theta}_2 = (\beta, \gamma)$. Thus $p_1 = 1$ and $p_2 = 2$.**

**Let $m_1 = 1$ and $m_2 = 2$ (model indexes).**

**If the chain is currently in state $(1, \boldsymbol{\theta}_1)$ and $\mathcal{M}_2$ is proposed:**

**A random variable $U \sim h(u|1, \theta_1, 2)$ is generated from some proposal density $h$. Let $\beta = \alpha - U$ and $\gamma = \alpha + U$, so $\mathbf{q}_{1,2}(\alpha, u) = (\alpha - u, \alpha + u)$ and $\left|\frac{d\mathbf{q}_{1,2}(\alpha,u)}{d(\alpha,u)}\right| = 2$.**

**For a proposed move from $\mathcal{M}_1$to $\mathcal{M}_2$, the Metropolis–Hasting ratio (98) is equal to**

$$\frac{f(2, \beta, \gamma|\mathbf{Y})\, g(1|2)}{f(1, \alpha|\mathbf{Y})\, g(2|1)\, h(u|1, \theta_1, 2)} \times 2. \tag{100}$$

# RJMCMC: Simple example with two models

**If the chain is currently in state $(2, \boldsymbol{\theta}_2)$ and $\mathcal{M}_1$ is proposed:**

**The inverse mapping is given by $(\alpha, u) = \mathbf{q}_{2,1}(\beta, \gamma) = \left(\frac{\beta+\gamma}{2}, \frac{\beta-\gamma}{2}\right)$. Therefore $\left|\frac{d\mathbf{q}_{2,1}(\beta,\gamma)}{d(\beta,\gamma)}\right| = \frac{1}{2}$, and $U^*$ is not required to match dimensions. This transition is entirely deterministic, so we replace $h(u^*|2, \theta_2, 1)$ in (98) with 1.**

**For a proposed move from $\mathcal{M}_2$ to $\mathcal{M}_1$, the Metropolis–Hastings ratio equals the reciprocal of (100)**

## Challenges to implementing RJMCMC

- **The number of dimensions can be enormous, so it can be critical to select an appropriate proposal distribution $h$ and to construct efficient moves between the different dimensions of the model space.**
- **Diagnosis of convergence is difficult**

# Other advanced MCMC algorithms

## Perfect Sampling

- **Generates a chain that has exactly reached the stationary distribution**
- **Method has promise, but challenges in implementation**

## Langevin Metropolis-Hastings algorithm

- **A random walk with drift. The drift favors proposals that move towards modes of the target distribution**
- **A random walk with drift can be generated using the proposal**

$$\mathbf{X}^* = \mathbf{x}^{(t)} + \mathbf{d}^{(t)} + \sigma\boldsymbol{\epsilon}^{(t)}$$

**where**

$$\mathbf{d}^{(t)} = \left(\frac{\sigma^2}{2}\right) \left.\frac{\partial \log f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(t)}}$$

**and $\boldsymbol{\epsilon}^{(t)}$ is a $p$-dimensional standard normal random variable.**

- **The scalar $\sigma$ is a tuning parameter whose fixed value is chosen by the user to control the magnitude of proposed steps.**

# Other advanced MCMC algorithms

## Hit-and-run algorithm

- Proposed move away from $\mathbf{x}^{(t)}$ is generated in two stages:
  1. Choose a direction to move
  2. Choose a distance to move in the chosen direction
- Resulting Markov chain isn't time invariant
- Can be useful when the state space of $\mathbf{X}$ is sharply constrained

## Multiple-try Metropolis-Hastings algorithm

Use when chain is slow to converge or trapped in a local mode of $f$

- Generate a large number of candidates and
- Choose among the candidates in a manner that ensures that the chain retains the correct limiting stationary distribution