Markov Chain Monte Carlo Methods

JEFF GILL

Distinguished Professor

Departments of Government and Mathematics & Statistics

American University

Bureaucratic Politics Example

- Contains *every* federal political appointee to full-time positions requiring Senate confirmation from November, 1964 through December, 1984 (collected by Mackenzie and Light, ICPSR Study Number 8458, Spring 1987).

- The survey queries various aspects of the Senate confirmation process, acclamation to running an agency or program, and relationships with other functions of government.

- The authors needed to preserve anonymity so they embargoed some variables and randomly sampled 1,500 down to 512.

- These latter issues are dealt with in Gill and Casella (JASA 2009).

Bureaucratic Politics Example

- **Outcome Variable:** `stress` as a surrogate measure for self-perceived effectiveness and job-satisfaction, measured as a five-point scale from "not stressful at all" to "very stressful."

- **Explanatory Variables:**
  - `Government Experience`,
  - `Ideology`,
  - `Committee Relationship`,
  - `Career.Exec-Compet`,
  - `Career.Exec-Liaison/Bur`,
  - `Career.Exec-Liaison/Cong`,
  - `Career.Exec-Day2day`,
  - `Career.Exec-Diff`,
  - `Confirmation Preparation`,
  - `Hours/Week`,
  - `President Orientation`.

Ordered Logit Model

- A Bayesian random effects specification for ordered survey outcomes, so latent variable thresholds for $\mathbf{Y}$ are assumed on the ordering:

$$\mathbf{U}_i: \ \theta_0 \underset{c=1}{\Longleftrightarrow} \theta_1 \underset{c=2}{\Longleftrightarrow} \theta_2 \underset{c=3}{\Longleftrightarrow} \theta_3 \ldots \theta_{C-1} \underset{c=C}{\Longleftrightarrow} \theta_C$$

- The vector of (unseen) utilities across individuals in the sample, $\mathbf{U}$, is determined by a linear additive specification of explanatory variables: $\mathbf{U} = -\mathbf{X}'\boldsymbol{\gamma} + \boldsymbol{\eta}$, where $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_p]$ does not depend on the θ_j, and $\boldsymbol{\eta} \sim F_{\boldsymbol{\eta}}$.

- This means that the probability that individual i in the sample is observed to be in category r or lower is:

$$P(\mathbf{Y}_i \leq r|\mathbf{X}_i) = P(\mathbf{U}_i \leq \theta_r) = P(\boldsymbol{\eta} \leq \theta_r + \mathbf{X}_i'\boldsymbol{\gamma}) = F_{\boldsymbol{\eta}_i}(\theta_r + \mathbf{X}_i'\boldsymbol{\gamma}),$$

  which is differently signed than in `R` : "$logitP(Y <= k|x) = zeta\_k - eta$" from the help page.

- Specifying a logistic distributional assumption on the errors and adding the random effect term produces this logistic cumulative specification for the whole sample:

$$F_{\boldsymbol{\eta}}(\theta_r + \mathbf{X}'\boldsymbol{\gamma} + \mathbf{b}) = P(\mathbf{Y} \leq r|\mathbf{X}) = [1 + \exp(-\theta_r - \mathbf{X}'\boldsymbol{\gamma} + \mathbf{b})]^{-1}$$

Model Priors

- The software we will use is for Bayesian hierarchial models, meaning that we have to stipulate an assumed distribution for the unknown parmaeters *before* observing and analyzing the data.

- Most social scientists give fairly vague prior statements as a way not to worry about this.

- Prior distributions are either semi-informed or skeptical:

$p(\gamma_k) \sim \mathcal{N}(\mu_{\gamma_k}, \sigma^2_\gamma),\ k = 1, \ldots, p$	for each of the p explanatory variables,
$p(\theta_j) \sim \mathcal{N}(0, \sigma^2_\theta),\ j = 1, \ldots, C-1$	for the four latent variable thresholds,
$b_i \sim \mathcal{N}(0, \tau)$	for the random effects term,
$\tau \sim \mathcal{IG}(\delta_1, \delta_2)$	for the random effects hyperprior,

Model Posterior

- All this produces a posterior distribution according to:

$$\pi(\boldsymbol{\gamma}, \boldsymbol{\theta}|\mathbf{X}, \mathbf{Y}) \propto L(\boldsymbol{\gamma}, \boldsymbol{\theta}|\mathbf{X}, \mathbf{Y}) p(\boldsymbol{\theta}) p(\boldsymbol{\gamma}) p(b|\tau) p(\tau)$$

$$\propto \prod_{i=1}^{n} \prod_{j=1}^{C-1} \prod_{k=1}^{p} [\Lambda(\theta_j - \mathbf{X}_i'\boldsymbol{\gamma} + \mathbf{b}_i) - \Lambda(\theta_{j-1} - \mathbf{X}_i'\boldsymbol{\gamma} + \mathbf{b}_i)]^{z_{ij}}$$

$$\times \exp\left(-\frac{(\boldsymbol{\gamma}_k - \mu_{\gamma_k})^2}{2\sigma_\gamma^2} - \frac{\theta_j^2}{2\sigma_\theta^2} - \frac{b_i^2}{2\tau^2} - \frac{\delta_2}{\tau} \right) \tau^{-(\delta_1+1)}$$

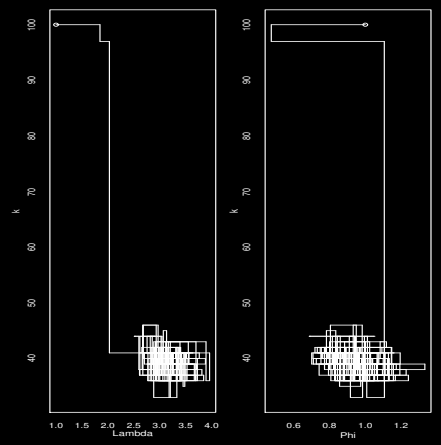
  which is kind of ugly (and hard marginalize).

- While this form tells us everything we need to know about the *joint distribution* of the model parameters, we need to marginalize (integrate) it for every one of these parameters to create an informative regression table.

- Solution: Gibbs sampling (a type of MCMC), which is a special application of a special kind of stochastic process..

Technologies that have changed my life:

- gene sequencing
- iPhones
- online banking
- Gibbs sampling



Posterior Summary, Model for Survey of Political Executives

	Mean	Std.Err.	95% HPD Intervals
Explanatory Variables:			
Constant Term	1.20215	2.24169	
Government Experience	-0.65500	0.60664	
Ideology	-0.49140	0.32964	
Committee Relationship	1.14550	0.39131	
Career.Exec-Compet	1.73300	0.85088	
Career.Exec-Liaison/Bur	-2.81800	0.55620	
Career.Exec-Liaison/Cong	1.03675	0.49193	
Career.Exec-Day2day	-0.76595	0.38019	
Career.Exec-Diff	0.27780	0.29838	
Confirmation Preparation	1.02440	0.45753	
Hours/Week	-0.72720	0.42732	
President Orientation	1.94950	0.86787	
Threshold Intercepts:			
None|Little	-5.93500	1.85782	
Little|Some	-2.69250	1.59126	
Some|Significant	1.19300	1.52653	
Significant|Extreme	8.40450	2.10379	

$\hat{\sigma} = 6.04350$ (1.20325), dashed vertical line at zero.

Core BUGS Code for this Model

```
for (i in 1:N) {
  b[i] ~ dnorm(0.0, tau)
  mu[i] <- theta[1]
    + theta[2]*previous.empl[i]     + theta[3]*ideology[i]
    + theta[4]*senate.relat[i]     + theta[5]*confirm.prep[i]
    + theta[6]*hours.week[i]      + theta[7]*career.exec.compet[i]
    + theta[8]*career.exec.liason.bur[i] + theta[9]*career.exec.liason.cong[i]
    + theta[10]*career.exec.day2day[i]  + theta[11]*career.exec.diff[i]
    + theta[12]*president.orient[i]
    for (j in 1:Ncut) {logit(Q[i, j]) <- -(k[j] + mu[i] - b[i])} #cum prob lower than j
    # probability that response = j
    p[i, 1] <- max( min(1 - Q[i, 1], 1), 0)
    for (j in 2 : Ncut) { p[i,j] <- max( min(Q[i, j - 1] - Q[i, j],1), 0) }
    p[i, (Ncut+1)] <- max( min(Q[i, Ncut], 1), 0)
    stress[i] ~ dcat(p[i, ])
}
```

Getting Started with the BUGS Language

- Explanatory variables: `contracting`, `gov.influence`, `leg.influence`, `elect.board`, `years.tenure`, `education`, `party.ID`, `category2`, `category3`, `category4`, `category5`, `category6`, `category7`, `category8`, `category9`, `category10`, `category11`, `category12`, `med.time`, `medt.contr`, `gov.ideology`, `lobbyists`, `nonprofits`.

- Data for JAGS needs to be in list form:

```
asap.jags.list <- list(STATES <- 50, SUBJECTS <- 713,
  state.id <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...),
  contracting <- c(6, 2, 0, 0, 0, 1, 0, 3, 3, 6, 0, 1, 5, 1, 1, 1, 0,...),
  :
  nonprofits <- c(1.9783, 0.509, 2.0701, 1.3639, 15.6682, 2.7968,...)
)
```

- Or read this in from a file that you've constructed.

Getting Started with the BUGS Language

- Markov chains require starting points, so create the following to modify:

```
"tau" <- 1
"tau.alpha" <- 1
"alpha" <- c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
      0,0,0,0,0,0,0,0,0,0,0,0)
"beta" <- c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
"gamma" <- c(0,0,0)
```

- Then make them different.

The First Part of the JAGS Code

```
model {
 for (i in 1:SUBJECTS) {
  mu[i] <- alpha[state.id[i]]
   + beta[1]*contracting[i] + beta[2]*gov.influence[i] + beta[3]*leg.influence[i]
   + beta[4]*elect.board[i] + beta[5]*years.tenure[i] + beta[6]*education[i]
   + beta[7]*party.ID[i]  + beta[8]*category2[i]  + beta[9]*category3[i]
   + beta[10]*category4[i] + beta[11]*category5[i]  + beta[12]*category6[i]
   + beta[13]*category7[i] + beta[14]*category8[i]  + beta[15]*category9[i]
   + beta[16]*category10[i] + beta[17]*category11[i] + beta[18]*category12[i]
   + beta[19]*med.time[i]  + beta[20]*medt.contr[i]
  grp.influence[i] ~ dnorm(mu[i],tau)
 }
 for (j in 1:STATES) {
  eta[j] <- gamma[1]*gov.ideology[j] + gamma[2]*lobbyists[j]
      + gamma[3]*nonprofits[j]
  alpha[j] ~ dnorm(eta[j],tau.alpha)
 }
```

The Second Part of the JAGS Code

```
 beta[1]  ~ dnorm(0.070,1)   # PRIOR MEANS FROM KELLEHER AND YACKEE 2009, MODEL 3
 beta[2]  ~ dnorm(-0.054,1)
 beta[3]  ~ dnorm(0.139,1)
 beta[4]  ~ dnorm(0.051,1)
 beta[5]  ~ dnorm(0.017,1)
 beta[6]  ~ dnorm(0.056,1)
 :
 beta[18] ~ dnorm(0.0,1)    # DIFFUSE PRIORS
 beta[19] ~ dnorm(0.184,1)   # PRIOR MEANS FROM KELLEHER AND YACKEE 2009, MODEL 3
 beta[20] ~ dnorm(0.156,1)
 gamma[1] ~ dnorm(0.0,1)    # DIFFUSE PRIORS
 gamma[2] ~ dnorm(0.0,1)
 gamma[3] ~ dnorm(0.0,1)
 tau    ~ dgamma(1.0,1)
 tau.alpha ~ dgamma(1.0,1)
}
```

Running JAGS From R

```
# LOAD LIBRARY AND SOURCE FILES
library(rjags); library(arm); library(coda); library(superdiag)

# DEFINE THE MODEL
asap.model2.rjags <- function() {
 for (i in 1:SUBJECTS) {
 :
}

# SAVE MODEL TO A FILE
write.model(asap.model2.rjags, "Article.JPART/asap.model2.rjags")
```

Running JAGS From R

```
# RUN THE SAMPLER AND COLLECT coda SAMPLES
asap2.model <- jags.model(file="Article.JPART/asap.model2.rjags",
  inits=asap.inits, data=asap.jags.list, n.chains=3, n.adapt=5000)
update(asap2.model, n.iter=2500)
asap2.mcmc <- coda.samples(model=asap2.model, variable.names=names(asap.jags.list),
  n.iter=2500)
summary(asap2.mcmc)

# CHECK CONVERGENCE
superdiag(as.mcmc.list(asap2.mcmc), burnin=0)

# GET THE DEVIANCE AND THE DIC
asap2.dic <- dic.samples(asap2.model, n.iter=25000, type="pD")
```

What is a Stochastic Process?

♢ A type of stochastic process that will help us estimate posterior quantities.

♢ A *stochastic process* is a consecutive set of random quantities defined on some known state space, Θ, indexed so that the order is known: $\{\theta^{[t]} : t \in T\}$.

♢ Frequently, but not necessarily, T is the set of positive integers implying consecutive, even-spaced time intervals: $\{\theta^{[t=0]}, \theta^{[t=1]}, \theta^{[t=2]}, \ldots\}$.

♢ A stochastic process must also be defined with respect to a *state space*, Θ, which identifies the range of possible values of θ. This state space is either discrete or continuous depending on how the variable of interest is measured.

What is a Markov Chain?

♢ A *Markov chain* is a stochastic process with the property that any specified state in the series, $\theta^{[t]}$, is dependent only the previous value of the chain, $\theta^{[t-1]}$.

♢ Therefore values are *conditionally* independent of all other previous values: $\theta^{[0]}, \theta^{[1]}, \ldots, \theta^{[t-2]}$.

♢ Formally:

$$P(\theta^{[t]} \in A | \theta^{[0]}, \theta^{[1]}, \ldots, \theta^{[t-2]}, \theta^{[t-1]}) = P(\theta^{[t]} \in A | \theta^{[t-1]}),$$

where A is any identified set (an event or range of events) on the complete state space. (We will use this A notation extensively.)

♢ Colloquially:

> "A Markov chain wanders around the state space remembering only where it has been in the last period."

Why Is this Useful?

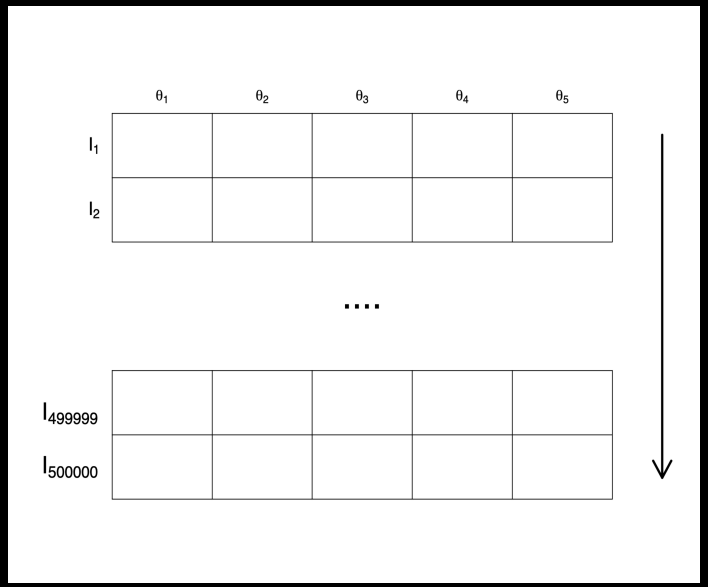
♢ This "short-term" memory property is very useful because when the chain eventually finds the region of the state space with highest density, it will wander around there producing a sample that is only modestly nonindependent.

♢ If this is the posterior region, then we can use these "empirical" values as legitimate posterior sample values.

♢ Thus difficult posterior calculations can be done with MCMC by letting the chain wander around "sufficiently long", thus producing summary statistics from recorded values.

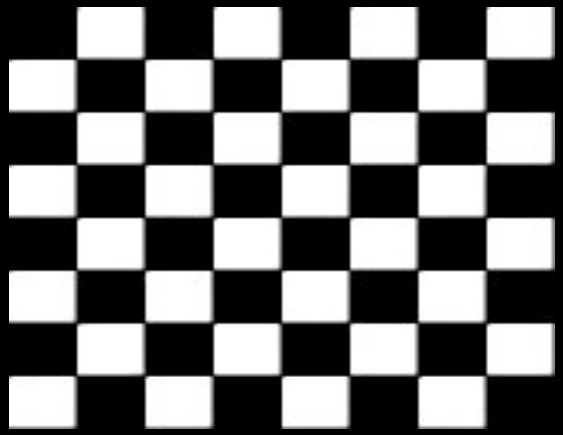
Returning to the Big Picture

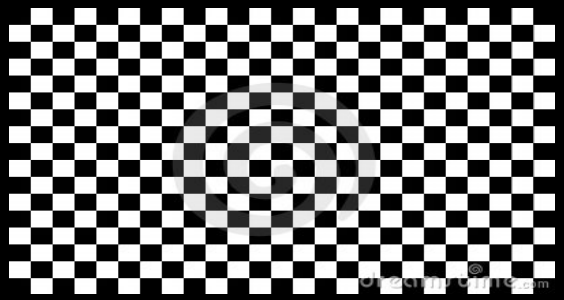
- So the Markov chain explores the multidimensional state space recording where it has been for us.
- This means that we can take the empirical values for each dimension (model parameter) and summarize them any way we want.

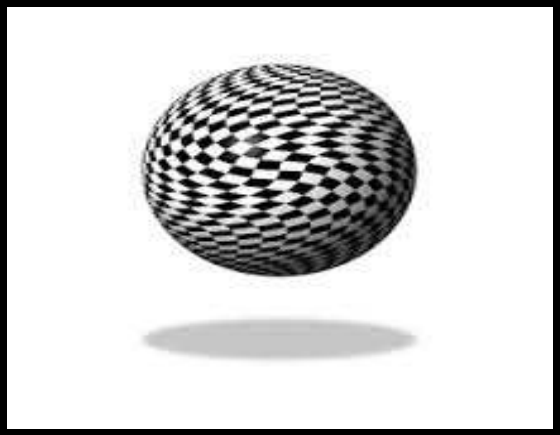


How Does it Move?

- How does the Markov chain decide to move?
- Define the *transition kernel*, K, as a general mechanism for describing the probability of moving to some other specified state based on the current chain status.
- $K(\theta, A)$ is a defined probability measure for all θ points in the state space to the set $A \in \Theta$.
- So $K(\theta, A)$ maps potential transition events to their probability of occurrence.







What is a Discrete Space Markov Chain Kernel?

♢ When the state space is discrete, K is a matrix mapping, $k \times k$ for k discrete elements in A, where each cell defines the probability of a state transition from the first term to all possible states:

$$P_A = \begin{bmatrix} p(\theta_1,\theta_1) & p(\theta_1,\theta_2) & \dots & p(\theta_1,\theta_{k-1}) & p(\theta_1,\theta_k) \\ p(\theta_2,\theta_1) & p(\theta_2,\theta_2) & \dots & p(\theta_2,\theta_{k-1}) & p(\theta_2,\theta_k) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ p(\theta_k,\theta_1) & p(\theta_k,\theta_2) & \dots & p(\theta_k,\theta_{k-1}) & p(\theta_k,\theta_k) \end{bmatrix}$$

where the row indicates where the chain is at this period and the column indicates where the chain is going in the next period.

♢ Each matrix element is a well-behaved probability, $p(\theta_i,\theta_j) \geq 0,\ \forall i,j \in A$. When the state space is continuous, then K is a conditional PDF: $f(\theta|\theta_i)$.

♢ Rows of P_A sum to one and define a conditional PMF since they are all specified for the same starting value and cover each possible destination in the state space: for row i: $\sum_{j=1}^{k} p(\theta_i,\theta_j)$.

What is a Continuous Space Markov Chain Kernel?

♢ When the state space is continuous, then K is a conditional PDF: $f(\theta|\theta_i)$.

♢ K is a conditional PDF: $f(\theta|\theta_i)$, meaning a properly defined probability statement for all $\theta \in A$, given some given current state θ_i.

♢ Continuous state space Markov chains have more involved theory; so it's often convenient to think about discrete Markov chains at first.

What is a Markov Chain? (cont.)

♢ Transition probabilities between two selected states for arbitrary numbers of steps m can be calculated multiplicatively.

♢ The probability of transitioning from the state $\theta_i = x$ at time 0 to the state $\theta_j = y$ in exactly m steps is given by the multiplicative series:

$$p^m(\theta_i^{[0]} = x, |\theta_j^{[m]} = y) = \underbrace{\sum_{\theta_1}\sum_{\theta_2}\cdots\sum_{\theta_{m-1}}}_{\text{all possible paths}} \underbrace{p(\theta_i,\theta_1)p(\theta_1,\theta_2)\cdots p(\theta_{m-1},\theta_j)}_{\text{transition products}}.$$

♢ So $p^m(\theta_i^{[0]} = x, |\theta_j^{[m]} = y)$ is also a stochastic transition matrix that specifies the product of all the required intermediate steps where we sum over all possible paths that reach y from x.

A Two State Markov Chain

♢ A two-dimensional state space: a discrete vote choice between two political parties, a commercial purchase decision between two brands, etc.

♢ Voters/consumers/predators/viruses/etc. who normally select θ_1 have an 80% chance of continuing to do so, and voters/consumers who normally select θ_2 have only a 40% chance of continuing to do so.

♢ The transition matrix P:

$$\begin{array}{c} \\ \text{current period} \end{array} \begin{array}{c} \overbrace{}^{\text{next period}} \\ \theta_1 \quad \theta_2 \end{array} \begin{array}{c} \\ \left\{ \begin{array}{c} \theta_1 \\ \theta_2 \end{array} \right. \end{array} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix}.$$

A Two State Markov Chain

♢ Assign a starting point:

$$S_0 = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix};$$

♢ To get to the first state, we simply multiply the initial state by the transition matrix:

$$S_1 = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.7 & 0.3 \end{bmatrix} = S_1.$$

A Two State Markov Chain

♢ This series continues multiplicatively as long as we like:

$$\text{Second state:} \quad S_2 = \begin{bmatrix} 0.7 & 0.3 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.74 & 0.26 \end{bmatrix}$$

$$\text{Third state:} \quad S_3 = \begin{bmatrix} 0.74 & 0.26 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.748 & 0.252 \end{bmatrix}$$

$$\text{Fourth state:} \quad S_4 = \begin{bmatrix} 0.748 & 0.252 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.7496 & 0.2504 \end{bmatrix}.$$

A Two State Markov Chain

♢ The **Big Picture**: Imagine that this stationary distribution was the articulation of some PMF or PDF that we could not analytically describe but would like to. If we could run this Markov chain sufficiently long we would eventually get the stationary distribution *for any point in the state space*.

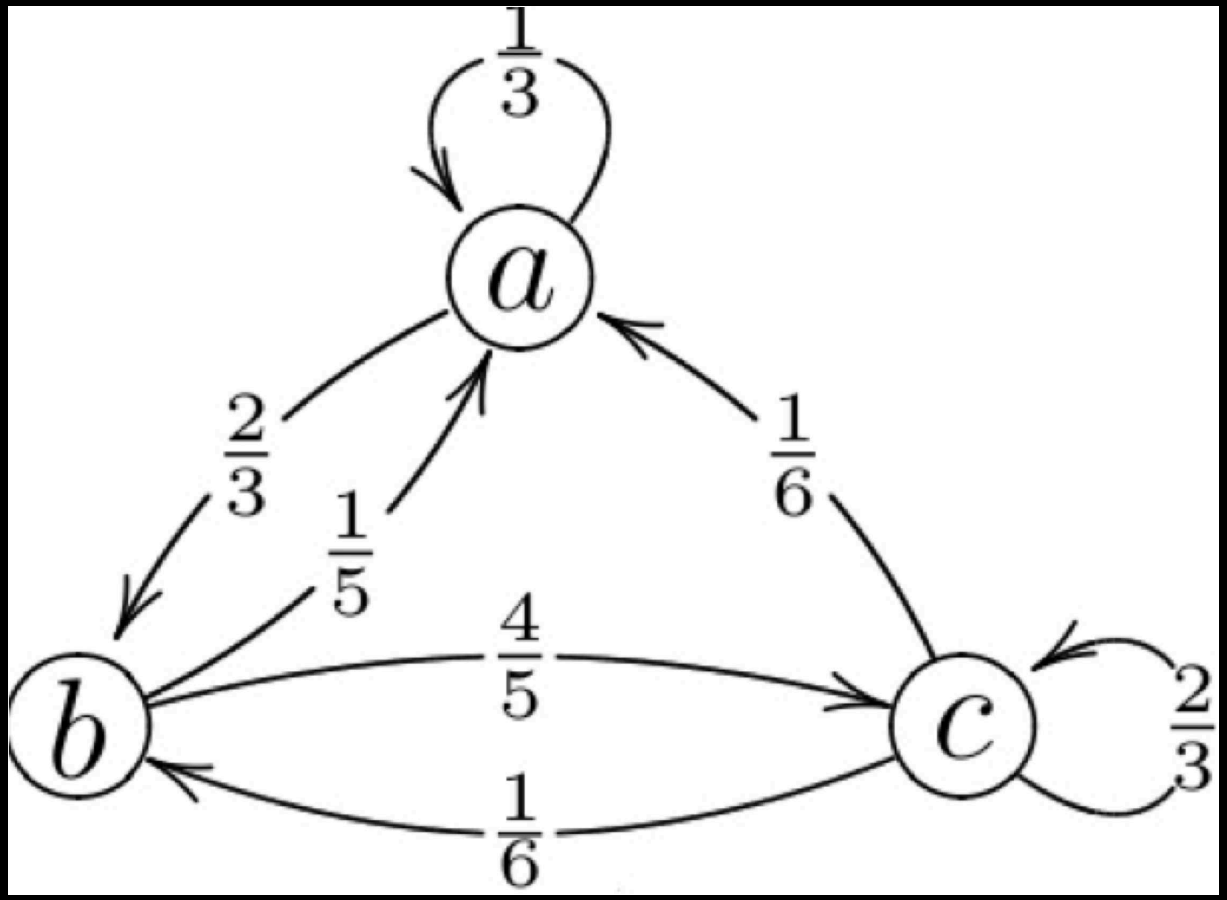
Actually, for this simple example we could solve directly for the steady state $S = [s_1, s_2]$ by stipulating:

$$\begin{bmatrix} s_1 & s_2 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{bmatrix} = \begin{bmatrix} s_1 & s_2 \end{bmatrix},$$

and solving the resulting two equations for the two unknowns.

♢ This operation of running a Markov chain until it reaches its stationary distribution is exactly the process employed in MCMC.

Diagramatic Markov Chain



Another Simple Example

- Define the intended algorithm as some means of shuffling cards.
- The objective (stationary distribution) is a uniformly random distribution in the deck: for any given order to the stack, the probability of any one card occupying any one position is 1/52.
- Algorithm: take the top card and insert it uniformly randomly at some other point in the deck, continue.
- Is this a stochastic process?
- Is this a Markov chain?
- What is the limiting distribution?
- See BTW, Bayer and Diaconis (1992).

Another Simple Example

- Define the intended algorithm as some means of shuffling cards.
- The objective (stationary distribution) is a uniformly random distribution in the deck: for any given order to the stack, the probability of any one card occupying any one position is 1/52.
- Algorithm: take the top card and insert it uniformly randomly at some other point in the deck, continue.
- Is this a stochastic process?
- Is this a Markov chain?
- What is the limiting distribution?
- See BTW, Bayer and Diaconis (1992).

Another Simple Example (cont.)

- For simplicity (without loss of generality), set $n = 3$ cards.
- Sample space ($3!$ elements): $\{[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]\}$.
- Transition kernel:

$$P = \begin{bmatrix} 1/3 & 0 & 1/3 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 0 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 1/3 & 1/3 & 0 & 1/3 \end{bmatrix}$$

- Starting point: $[1,2,3]$.
- Assume no periodicity, obviously it is irreducible and closed (therefore positive reccurent), therefore this transition kernel defines an ergodic Markov chain.

Another Simple Example (cont.)

R code:

```
P <- matrix(c(1/3,0,1/3,0,1/3,0,0,1/3,1/3,0,1/3,0,1/3,0,1/3,0,0,1/3,
      1/3,0,0,1/3,0,1/3,0,1/3,0,1/3,1/3,0,0,1/3,0,1/3,0,1/3),nrow=6)
P
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 0.3333333 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
[2,] 0.0000000 0.3333333 0.0000000 0.0000000 0.3333333 0.3333333
[3,] 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000
[4,] 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.3333333
[5,] 0.3333333 0.3333333 0.0000000 0.0000000 0.3333333 0.0000000
[6,] 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.3333333
```

Another Simple Example (cont.)

```
MC.multiply <- function(P.in,N) {
   S <- c(1,0,0,0,0,0)%*%P.in
   for (i in 2:N) {
     S <- S%*%P.in
     print(S)
   }
}

MC.multiply(P,15)

     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.2222222 0.1111111 0.2222222 0.2222222 0.1111111 0.1111111
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1851852 0.1481481 0.1851852 0.1851852 0.1481481 0.1481481
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1728395 0.1604938 0.1728395 0.1728395 0.1604938 0.1604938
```

```
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1687243 0.1646091 0.1687243 0.1687243 0.1646091 0.1646091
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1673525 0.1659808 0.1673525 0.1673525 0.1659808 0.1659808
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1668953 0.1664380 0.1668953 0.1668953 0.1664380 0.1664380
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1667429 0.1665905 0.1667429 0.1667429 0.1665905 0.1665905
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666921 0.1666413 0.1666921 0.1666921 0.1666413 0.1666413
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666751 0.1666582 0.1666751 0.1666751 0.1666582 0.1666582
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666695 0.1666638 0.1666695 0.1666695 0.1666638 0.1666638
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666676 0.1666657 0.1666676 0.1666676 0.1666657 0.1666657
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666670 0.1666664 0.1666670 0.1666670 0.1666664 0.1666664
```

```
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666668 0.1666666 0.1666668 0.1666668 0.1666666 0.1666666
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666667 0.1666666 0.1666667 0.1666667 0.1666666 0.1666666
     [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] 0.1666667 0.1666666 0.1666667 0.1666667 0.1666666 0.1666666
```

The Chapman-Kolmogorov Equations

- Recall the probability of transitioning from the state $\theta_i = x$ at time 0 to the state $\theta_j = y$ in exactly m steps:

$$p^m(\theta_i^{[0]} = x, |\theta_j^{[m]} = y) = \underbrace{\sum_{\theta_1}\sum_{\theta_2}\cdots\sum_{\theta_{m-1}}}_{\text{all possible paths}} \underbrace{p(\theta_i,\theta_1)p(\theta_1,\theta_2)\cdots p(\theta_{m-1},\theta_j)}_{\text{transition products}}.$$

- These can be strung out to show how successive events are bound together probabilistically:

$$p^{m_1+m_2}(x,y) = \sum_{\text{all } z} p^{m_1}(x,z)p^{m_2}(z,y) \qquad \text{discrete case}$$

$$p^{m_1+m_2}(x,y) = \int_{\text{range } z} p^{m_1}(x,z)p^{m_2}(z,y)dz \qquad \text{continuous case.}$$

- The Chapman-Kolmogorov equations are particularly elegant for the discrete case because they can be represented as a series of transition matrix multiplications:

$$p^{m_1+m_2} = p^{m_1}p^{m_2} = p^{m_1}p^{m_2-1}p = p^{m_1}p^{m_2-2}p^2 = \ldots.$$

Marginal Distributions

- We want the *marginal* distribution at some step m th from the transition kernel.

- For the discrete case the marginal distribution of the chain at the m step is obtained by inserting the current value of the chain, $\theta_i^{[m]}$, into the row of the transition kernel for the m^{th} step, p^m:

$$\pi^m(\theta) = [p^m(\theta_1), p^m(\theta_2), \ldots, p^m(\theta_k)].$$

- So the marginal distribution at the first step of discrete Markov chain is given by:

$$\pi^1(\theta) = p^1\pi^0(\theta),$$

where π^0 is the initial starting value assigned to the chain and $p^1 = p$ is a transition matrix.

Marginal Distributions in the Future

- The marginal distribution at some (possibly distant) step m for a given starting value is:

$$\pi^m = p\pi^{m-1} = p(p\pi^{m-2}) = p^2(p\pi^{m-3}) = \ldots = p^m\pi^0.$$

- Since successive products of probabilities quickly result in lower values, the property above shows how Markov chains eventually "forget" their starting points.

- The marginal distribution for the continuous case is only slightly more involved since we cannot just list as a vector the quantity:

$$\pi^m(\theta_j) = \int_\theta p(\theta, \theta_j)\pi^{m-1}(\theta)d\theta,$$

  which is the marginal distribution of the chain, given that it is currently on point θ_j at step m.

- Now we need a set of conditions that assures us that a given Markov chain will reach its stationary distribution, where the marginal distribution is constant and permanent.

Markov Chain Properties: Homogeneity

- *Homogeneity*
  - A Markov chain is said to be *homogeneous* at step m if the transition probabilities at this step do not depend on the value of m.
  - With non-homogeneous Markov chains the transition matrix is not constant but a function of time.
  - These are useful for modeling probabilistic decay processes, but not for our purposes.
  - The samplers that we use/pick are known to obtain the property, subject to machine implementation.
  - Counter-example: at the starting point a chain cannot be homogeneous since the marginal distribution for the first step is clearly not independent of the initial values that are hand-picked.

Markov Chain Properties: States

- *Some Characteristics of* States*:*
  - A state is *absorbing* if once the chain enters this state it cannot leave: $p(A, A^c) = 0$.
  - A state is *transient* if the probability of the chain not returning to this state is non-zero: $1 - p(A, A) > 0$. Also, phrases as *a finite number of visits in infinite time.*
  - State A is *closed* to state B if a Markov chain on A cannot reach B: $p(A, B) = 0$. State A is closed in the general sense if it is absorbing.

Markov Chain Properties: Irreducibility

- *Irreducibility*
  - A state is *irreducible* if for every θ_i and θ_j in this state, the two sub-states "communicate."
  - A Markov chain is *irreducible* on A if every reached point or collection of points (necessarily a subspace in the continuous case) can be reached from every other reached point or collection of points:

$$p(\theta_i, \theta_j) \neq 0, \ \forall \theta_i, \theta_j \in A.$$

  - A convenient way to remember the principle behind irreducibility is the notion that you could reduce the set if you wanted to, but that *you do not want to* because then there will be points that cannot be reached from other points.

Markov Chain Properties: Recurrence

- *Recurrence*
  - *If a state is closed, discrete or continuous but finite, and irreducible, then this state and all sub-spaces within this subspace are recurrent.*
  - Markov chains operating on recurrent state spaces are recurrent.
  - Recurrence is good!
  - Formal definition:
    A irreducible Markov chain is called *recurrent* with regard to a given state, A, which is a single point or a defined collection of points (required for the bounded-continuous case), if the probability that the chain occupies A infinitely often over unbounded time is nonzero.

Markov Chain Properties: Recurrence

- *Recurrence*
  - Colloquial definition:
    When a chain moves into a recurrent state, it stays there forever and visits every subspace infinitely often.
  - A Markov chain is *positive recurrent* if the mean time to return to A is bounded.
  - Otherwise it is called *null recurrent*.
  - Note: irreducible chains are either recurrent or transient.

Markov Chain Properties: Harris Recurrence

- *Harris Recurrence*
  - If we only had to deal with discrete or finite state spaces, then standard recurrence would be enough.
  - With unbounded-continuous state spaces it is necessary to have a stricter definition that guarantees that the probability of visiting subspace A infinitely often in the limit is still one.
  - Define η_A as the number of visits to A in the limit.
  - Require $P(B) > 0 \, \forall \, B \subset A$.
  - The set A is Harris recurrent if $P(\eta_B = \infty) = 1, \quad \forall B \in A$.
  - An irreducible Markov chain is Harris recurrent if it every possible subspace is Harris recurrent.
  - An aperiodic, irreducible chain with an invariant distribution on an unbounded continuous state space that is *not* Harris recurrent has a positive probability of getting stuck forever in an area bounded away from convergence, given a starting point there.

Markov Chain Properties: Linkages

- *Recurrence and Irreducibility*
  - The union of a set of recurrent states (nonempty, and bounded or countable) or Harris recurrent states is a new state that is closed and irreducible.
  - The linkage between recurrence and irreducibility is important in defining a subspace that captures a Markov chain and at the same time assures that this Markov chain will explore all of the subspace.
  - **Important result:**
    - Given a Markov chain on continuous state space,
    - when the chain wanders into a closed, irreducible set of Harris recurrent states,
    - it stays there forever and visits every single sub-state (region) with probability one.

Markov Chain Properties: Stationarity

- *Stationarity*
  - Define $\pi(\theta)$ as the stationary distribution of the Markov chain for θ on the state space A.
  - Recall that $p(\theta_i, \theta_j)$ is the probability that the chain will move from θ_i to θ_j at some arbitrary step t,
  - and $\pi^t(\theta)$ is the corresponding marginal distribution.
  - The stationary distribution satisfies:

$$\sum_{\theta_i} \pi^t(\theta_i) p(\theta_i, \theta_j) = \pi^{t+1}(\theta_j) \qquad \text{Discrete case}$$

$$\int \pi^t(\theta_i) p(\theta_i, \theta_j) d\theta_i = \pi^{t+1}(\theta_j) \qquad \text{Continuous case.}$$

    meaning that application of the kernel to the current distribution probabilities returns the same probabilities.

Markov Chain Properties: Stationarity

- The marginal distribution remains fixed when the chain reaches the stationary distribution and we might as well drop the superscript designation for iteration number and just use $\pi(\theta)$.
- In shorthand: $\pi = \pi p$.
- Once the chain reaches its stationary distribution, it stays in this distribution and moves about, or "mixes," throughout the subspace according to marginal distribution, $\pi(\theta)$, forever.
- Recurrence gives the range restriction property whereas stationarity gives constancy of the probability structure that dictates movement.

Markov Chain Properties: Periodicity

- *Periodicity*
  - The *period* of a Markov chain is the length of time required to repeat an identical cycle of chain values.
  - Actually, periodicity is bad and we want an *aperiodic* Markov chain: the repeating length is the trivial value of one.
  - The problem with periodicity is that it destroys the probabilistic property of the transition kernel.
  - Update of our status: we have recurrence to make sure that we revisit states infinitely often in the limit, we have stationarity to make sure that eventually we will obtain constancy of the probability structure, but we don't have any guarantee about what kind of distribution we obtain in this limit.
  - Caveat: all Markov chains run on computers are actually periodic ("Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin." –John von Neumann [1903-1957])

Markov Chain Properties: Ergodicity

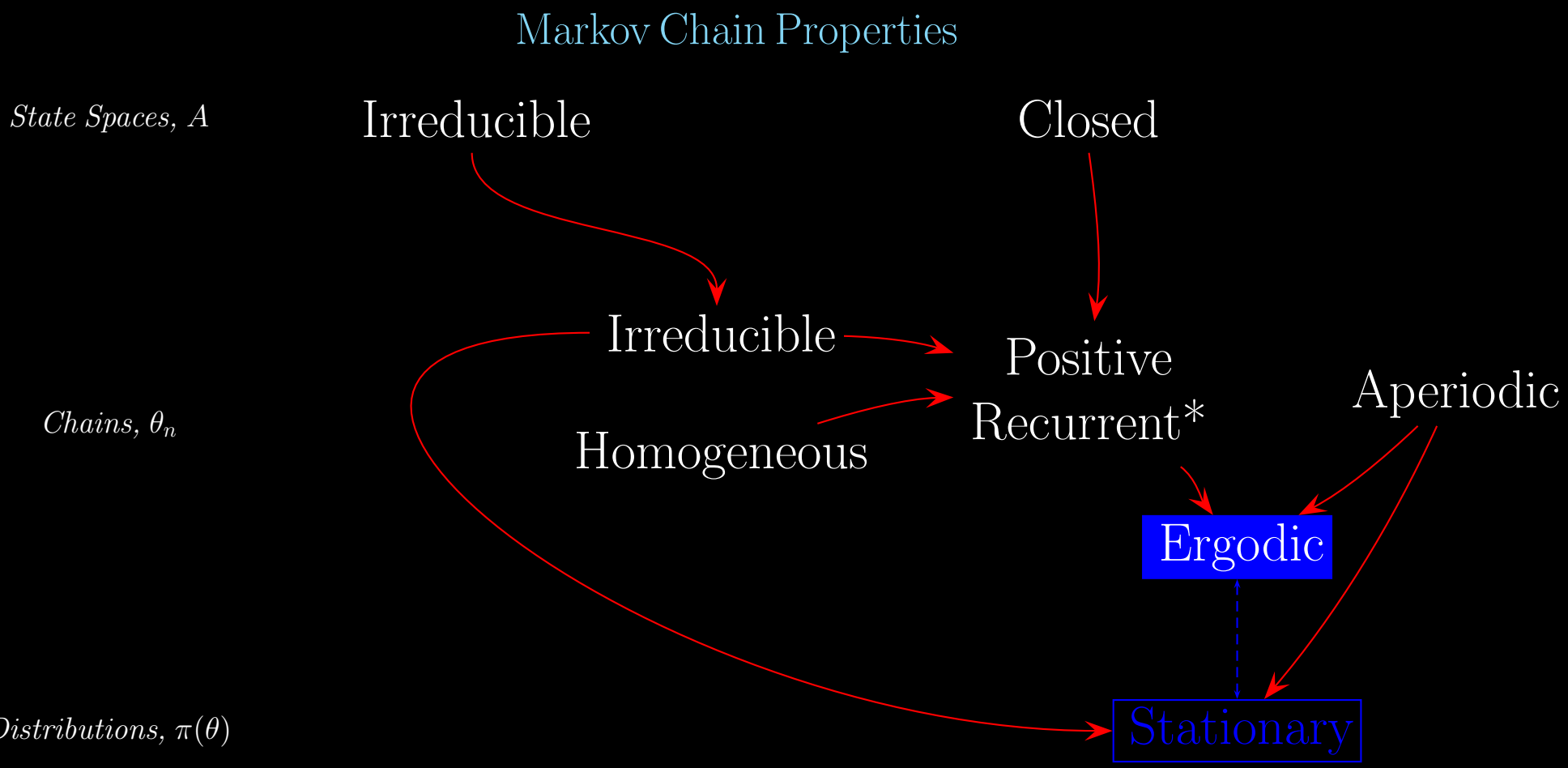
- If a chain is positive Harris recurrent (unbounded continous case) or recurrent (discrete/bounded continouus case), and aperiodic, then we call it *ergodic*. Ergodic Markov chains have the property:

$$\lim_{n\to\infty} K^n(\theta_i, \theta_j) = \pi(\theta_j),$$

  for all θ_i, and θ_j in the subspace

- What this means: *once a specified chain is determined to have reached its ergodic state, sample values behave as if they were produced by the posterior of interest from the model.*

- The *ergodic theorem* is the equivalent of the strong law of large numbers but for Markov chains, since it states that any specified function of the posterior distribution can be estimated with samples from a Markov chain in its ergodic state because averages of sample values give strongly consistent parameter estimates.



*Discrete or bounded continuous: $E[\eta_A] = \infty \; \forall \theta \in A$. Continuous: $P(\theta_n \in A) = 1, \; \forall A \in H$ where $P > 0$ (Harris). Positive recurrence: mean return time to A is bounded.

Markov Chain Properties

- *Consequences of Ergodicity*
  - Suppose $\theta_{i+1}, \ldots, \theta_{i+n}$ are n (not necessarily consecutive) values from a Markov chain that has reached its ergodic distribution, a statistic of interest, $h(\theta)$, can be calculated empirically:

$$\hat{h}(\theta_i) = \frac{1}{n} \sum_{j=i+1}^{i+n} h(\theta_j) = \int h(\theta) d\pi(\theta)$$

    This is the ergodic theorem.

  - For finite quantities this converges almost surely:

$$p[\hat{h}(\theta_i) \rightarrow h(\theta), \text{ as } n \rightarrow \infty] = 1$$

  - Even though Markov chain values, by their very definition, have serial dependence, the mean of the chain values provides a strongly consistent estimate of the true parameter.

Markov Chain Properties

- *Flavors of Ergodicity*
  - Nothing yet said about the *rate of convergence*.
  - **Geometric Ergodicity**: a Markov chain with stationary distribution π at time t, a positive constant $r \in (0, 1)$, and a non-negative, real-valued function of θ such that:

$$||P(\theta^{[t]}) - \pi|| \leq f(\theta) r^t.$$

  - **Uniform Ergodicity**: replace $f(\theta)$ with a positive constant k:

$$||\hat{h}(\theta_i) - h(\theta)|| \leq k r^t.$$

  - Where

$$\|f(\boldsymbol{\theta}_t) - \pi(\boldsymbol{\theta})\| = \frac{1}{2} \sup_{\boldsymbol{\theta} \in A} A \int_{\boldsymbol{\Theta}} |f(\boldsymbol{\theta}_t) - \pi(\boldsymbol{\theta})| \, d\boldsymbol{\theta},$$

    which is also half of the L_1 distance (the $1/2$ is sometimes omitted by authors).

Markov Chain Properties

- *Flavors of Ergodicity (cont.)*
  - No matter which flavor of ergodicity applies, we get the important theoretical result from the CLT.
  - For a given empirical estimator $\hat{h}(\theta_i)$ with bounded limiting variance, we get:

$$\sqrt{n}\frac{\hat{h}(\theta_i) - h(\theta)}{\sqrt{Var(\hat{h}(\theta_i))}} \underset{n\to\infty}{\longrightarrow} \mathcal{N}(0,1)$$

  since $\hat{h}(\theta_i) = \frac{1}{n}\sum_{j=i+1}^{i+n} h(\theta_j)$ is really a mean.

The Gibbs Sampler

♢ Recall the conditional exponential pdfs:

$$f(x|y) \propto y\text{exp}[-yx], \quad f(y|x) \propto x\text{exp}[-xy], \quad 0 < x, y < B < \infty.$$

where we want to describe the marginal distributions of x and y.

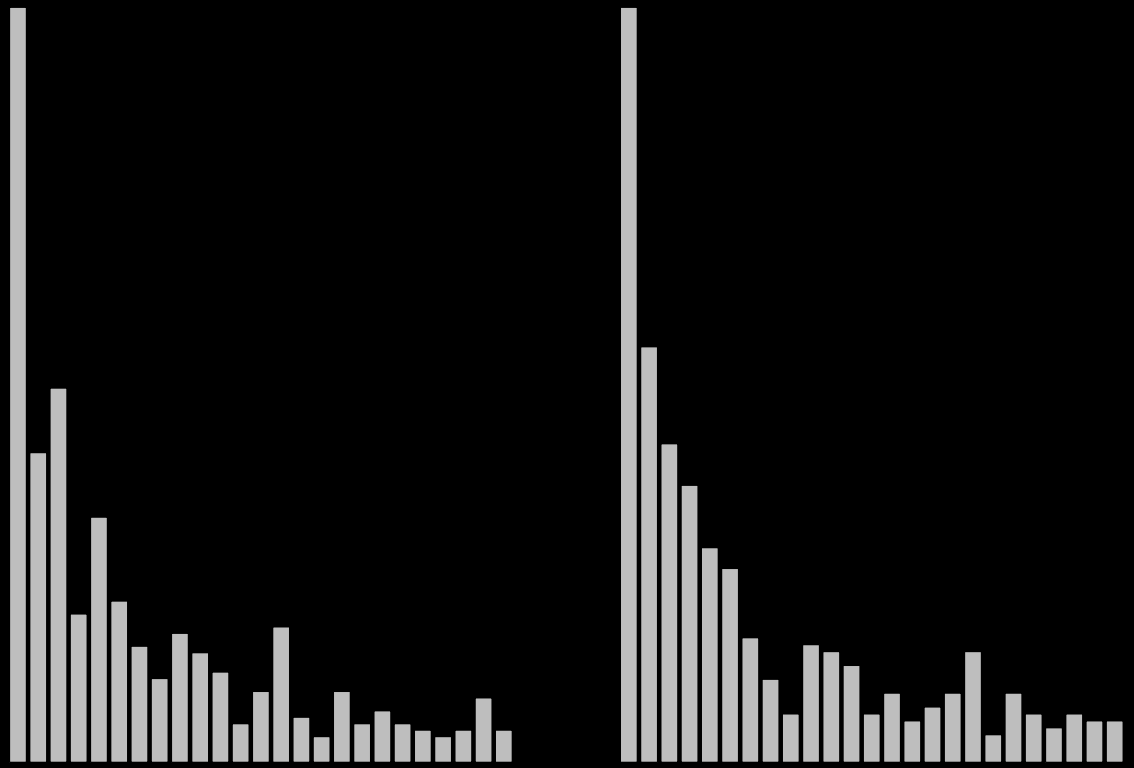
♢ For two parameters, x and y, this involves a starting point, $[x_0, y_0]$, and the cycles defined by drawing random values from the conditionals according to:

$$\begin{aligned}
x_1 &\sim f(x|y_0), & y_1 &\sim f(y|x_1) \\
x_2 &\sim f(x|y_1), & y_2 &\sim f(y|x_2) \\
x_3 &\sim f(x|y_2), & y_3 &\sim f(y|x_3) \\
&\vdots & &\vdots \\
&\vdots & &\vdots \\
x_m &\sim f(x|y_{m-1}), & y_m &\sim f(y|x_m).
\end{aligned}$$

The Gibbs Sampler, Conditional Exponential Distributions

```
B <- 5; k <- 15; m <- 500; x <- NULL; y <- NULL
while (length(x) < m) {
  x.val <- c(runif(1,0,B),rep((B+1),length=k))
  y.val <- c(runif(1,0,B),rep((B+1),length=k))
  for (j in 2:(k+1)) {
    while(x.val[j] > B) x.val[j] <- rexp(1,y.val[j-1])
    while(y.val[j] > B) y.val[j] <- rexp(1,x.val[j])
  }
  x <- c(x,x.val[(k+1)])
  y <- c(y,y.val[(k+1)])
}
```

The Gibbs Sampler (cont.)



The Gibbs Sampler (cont.)

♢ The Gibbs sampler is a transition kernel created by a series of full conditional distributions.

♢ It is a Markovian updating scheme based on conditional probability statements.

♢ If the limiting distribution of interest is $\pi(\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is an k length vector of coefficients to estimate, then the objective is to produce a Markov chain that cycles through these conditional statements moving toward and then around this distribution.

♢ The set of full conditional distributions for $\boldsymbol{\theta}$ are denoted $\boldsymbol{\Theta}$ and defined by $\pi(\boldsymbol{\Theta}) = \pi(\theta_i|\boldsymbol{\theta}_{-i})$ for $i = 1, \ldots, k$, where the notation $\boldsymbol{\theta}_{-i}$ indicates a specific parametric form from $\boldsymbol{\Theta}$ without the θ_i coefficient.

The Gibbs Sampler (cont.)

♢ Steps:

1. Choose starting values: $\boldsymbol{\theta}^{[0]} = [\theta_1^{[0]}, \theta_2^{[0]}, \ldots, \theta_k^{[0]}]$.
2. At the j^{th} iteration, $j = 1, \ldots m$ complete the single cycle by drawing values from the k distributions given by:

$$
\begin{array}{lcllllllll}
\theta_1^{[j]} & \sim & \pi(\theta_1 & | & \theta_2^{[j-1]}, & \theta_3^{[j-1]}, & \theta_4^{[j-1]}, & \ldots, & \theta_{k-1}^{[j-1]}, & \theta_k^{[j-1]}) \\
\theta_2^{[j]} & \sim & \pi(\theta_2 & | & \theta_1^{[j]}, & \theta_3^{[j-1]}, & \theta_4^{[j-1]}, & \ldots, & \theta_{k-1}^{[j-1]}, & \theta_k^{[j-1]}) \\
\theta_3^{[j]} & \sim & \pi(\theta_3 & | & \theta_1^{[j]}, & \theta_2^{[j]}, & \theta_4^{[j-1]}, & \ldots, & \theta_{k-1}^{[j-1]}, & \theta_k^{[j-1]}) \\
 & & \vdots & & & & & & & \\
\theta_{k-1}^{[j]} & \sim & \pi(\theta_{k-1} & | & \theta_1^{[j]}, & \theta_2^{[j]}, & \theta_3^{[j]}, & \ldots, & \theta_{k-2}^{[j]}, & \theta_k^{[j-1]}) \\
\theta_k^{[j]} & \sim & \pi(\theta_k & | & \theta_1^{[j]}, & \theta_2^{[j]}, & \theta_3^{[j]}, & \ldots, & \theta_{k-2}^{[j]}, & \theta_{k-1}^{[j]})
\end{array}
$$

3. Increment j and repeat until convergence.

Gibbs Sampler Theory

- Properties of the Gibbs sampler:
  - Since the Gibbs sampler conditions only on values from the last iteration of its chain values, it clearly has the Markovian property.
  - The Gibbs sampler has the true posterior distribution of the parameter vector as its limiting distribution: $\boldsymbol{\theta}^{[i]} \underset{i=1 \longrightarrow \infty}{\xrightarrow{d}} \boldsymbol{\theta} \sim \pi(\boldsymbol{\theta})$.
  - The Gibbs sampler is a homogeneous Markov chain: the consecutive probabilities are independent of n, the current length of the chain.
  - The Gibbs sampler converges at a geometric rate: the total variation distance between an arbitrary time and the point of convergence decreases at a geometric rate in time (t).
  - The Gibbs sampler is ergodic.

The Gibbs Sampler, Realistic Example

- Consider a series of coal mine disasters over a 112-year history in the U.K. with relatively high disaster counts in the early era and relatively low disaster counts in the late era. Thus the question from a public policy perspective is when did improvements in technology and safety practices have an actual effect on the rate of serious accidents?

- The data, covering the years 1851 to 1962, are given (in sequential rows) by:

1851-1866	4	5	4	1	0	4	3	4	0	6	3	3	4	0	2	6	1-16
1867-1882	3	3	5	4	5	3	1	4	4	1	5	5	3	4	2	5	17-32
1883-1898	2	2	3	4	2	1	3	2	2	1	1	1	1	3	0	0	33-48
1899-1914	1	0	1	1	0	0	3	1	0	3	2	2	0	1	1	1	49-64
1915-1930	0	1	0	1	0	0	0	2	1	0	0	0	1	1	0	2	65-80
1931-1946	3	3	1	1	2	1	1	1	1	2	4	2	0	0	0	1	81-96
1947-1962	4	0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	97-112

The Gibbs Sampler, Realistic Example (cont.)

- This is an example of a *change-point problem* where the objective is to estimate when the parameterization of the underlying data-generation process changes.

- The statistical objective is to estimate the point where the Poisson parameter changes and obtain estimated values for this parameter before and after the change.

- Specifically, $y_1, y_2, \ldots, y_n$ are a series of count data where there exists the possibility of a change-point at some period, k, such that:

$$y_i|\lambda \sim \mathcal{P}(\lambda) \qquad i = 1, \ldots, k$$
$$y_i|\phi \sim \mathcal{P}(\phi) \qquad i = k+1, \ldots, n.$$

  So now there are three parameters to estimate: λ, ϕ, and k, where k plays a very different role than standard parameters.

The Gibbs Sampler, Realistic Example (cont.)

- Priors:

$$\lambda \sim \mathcal{G}(\alpha, \beta) \qquad \phi \sim \mathcal{G}(\gamma, \delta) \qquad k \sim \text{discrete uniform on } [1, 2, \ldots, n],$$

where the prior parameters are assigned according to: $\alpha = 4$, $\beta = 1$, $\gamma = 1$, $\delta = 2$. Since the mean of a gamma distribution is the product of its parameters, this assignment of parameters roughly resembles the mean of the first 50% of the data ($\alpha\beta$), and the second 50% of the data ($\gamma\delta$).

- Joint posterior:

$$\pi(\lambda, \phi, k|\mathbf{y}) \propto L(\lambda, \phi, k|\mathbf{y})\pi(\lambda|\alpha, \beta)\pi(\phi|\gamma, \delta)\pi(k)$$

$$= \left(\prod_{i=1}^{k} \frac{e^{-\lambda}\lambda^{y_i}}{y_i!}\right)\left(\prod_{i=k+1}^{n} \frac{e^{-\phi}\phi^{y_i}}{y_i!}\right)\left(\frac{\beta^{\alpha}}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}\right)$$

$$\times \left(\frac{\delta^{\gamma}}{\Gamma(\gamma)}\phi^{\gamma-1}e^{-\delta\phi}\right)\frac{1}{n}$$

$$\propto \lambda^{\alpha-1+\sum_{i=1}^{k} y_i}\phi^{\gamma-1+\sum_{i=k+1}^{n} y_i}\exp[-(k+\beta)\lambda - (n-k+\delta)\phi].$$

The Gibbs Sampler, Realistic Example (cont.)

- Conditional posterior distributions:

$$\lambda|\phi,k \sim \mathcal{G}(\alpha+\sum_{i=1}^{k} y_i, \beta+k) \qquad \phi|\lambda,k \sim \mathcal{G}(\gamma+\sum_{i=k+1}^{n} y_i, \delta+n-k),$$

$$\begin{aligned}
p(\mathbf{y}|k,\lambda,\phi) &= \left(\prod_{i=1}^{k}\frac{e^{-\lambda}\lambda^{y_i}}{y_i!}\right)\left(\prod_{i=k+1}^{n}\frac{e^{-\phi}\phi^{y_i}}{y_i!}\right) \\
&= \left(\prod_{i=1}^{n}\frac{1}{y_i!}\right)\left(e^{-k\lambda}\lambda^{\sum_{i=1}^{k} y_i}\right)e^{-(n-k)\phi}\left(\prod_{i=k+1}^{n}\phi^{y_i}\right) \\
&= \left(\prod_{i=1}^{n}\frac{1}{y_i!}\right)\lambda^{\sum_{i=1}^{k} y_i}e^{-n\phi}e^{k(\phi-\lambda)}\left(\prod_{i=k+1}^{n}\phi^{y_i}\right)\left(\prod_{i=1}^{k}\frac{\phi^{y_i}}{\phi^{y_i}}\right) \\
&= \left(\prod_{i=1}^{n}\frac{e^{-\phi}\phi^{y_i}}{y_i}\right)\left(e^{k(\phi-\lambda)}\left(\frac{\lambda}{\phi}\right)^{\sum_{i=1}^{k} y_i}\right) \\
&= f(\mathbf{y},\phi)L(\mathbf{y}|k,\lambda,\phi).
\end{aligned}$$

- This latter provides two functions, the first of which is free of k.

The Gibbs Sampler, Realistic Example (cont.)

- Since our objective is simply a full conditional statement for k, we will use only $L(\mathbf{y}|k, \lambda, \phi)$.
- Suppressing the conditioning on λ and ϕ for notational clarity only, apply Bayes' Law with a generic distribution for k, $p(k)$:

$$p(k|\mathbf{y}) = \frac{f(\mathbf{y}, \phi)L(\mathbf{y}|k)p(k)}{\sum_{\ell=1}^{n} f(\mathbf{y}, \phi)L(\mathbf{y}|k_\ell)p(k_\ell)} = \frac{L(\mathbf{y}|k)p(k)}{\sum_{\ell=1}^{n} L(\mathbf{y}|k_\ell)p(k_\ell)}.$$

- Here we took advantage of the discrete feature of k and summed over all possible values with the index ℓ.
- Due to the discrete prior, this simplifies even more such that with proportionality now $p(k|\mathbf{y}) \propto L(\mathbf{y}|k)$.
- So each iteration of the Gibbs sampler will calculate an n-length probability vector for k and draw a value accordingly.

R Code for the Realistic Example, (cont.)

```
bcp <- function(theta.matrix,y,a,b,g,d) {
  n <- length(y)
  k.prob <- rep(NA,length=n)
  for (i in 2:nrow(theta.matrix)) {
    lambda <- rgamma(1,a+sum(y[1:theta.matrix[(i-1),3]]),
              b+theta.matrix[(i-1),3])
    phi  <- rgamma(1,g+sum(y[theta.matrix[(i-1),3]:n]),
              d+length(y)-theta.matrix[(i-1),3])
    for (j in 1:n) k.prob[j] <- exp(j*(phi-lambda))*
                   (lambda/phi)^sum(y[1:j])
    k.prob <- k.prob/sum(k.prob)
    k   <- sample(1:n,size=1,prob=k.prob)
    theta.matrix[i,] <- c(lambda,phi,k)
  }
  return(theta.matrix)
}
```

The Gibbs Sampler, Realistic Example (cont.)

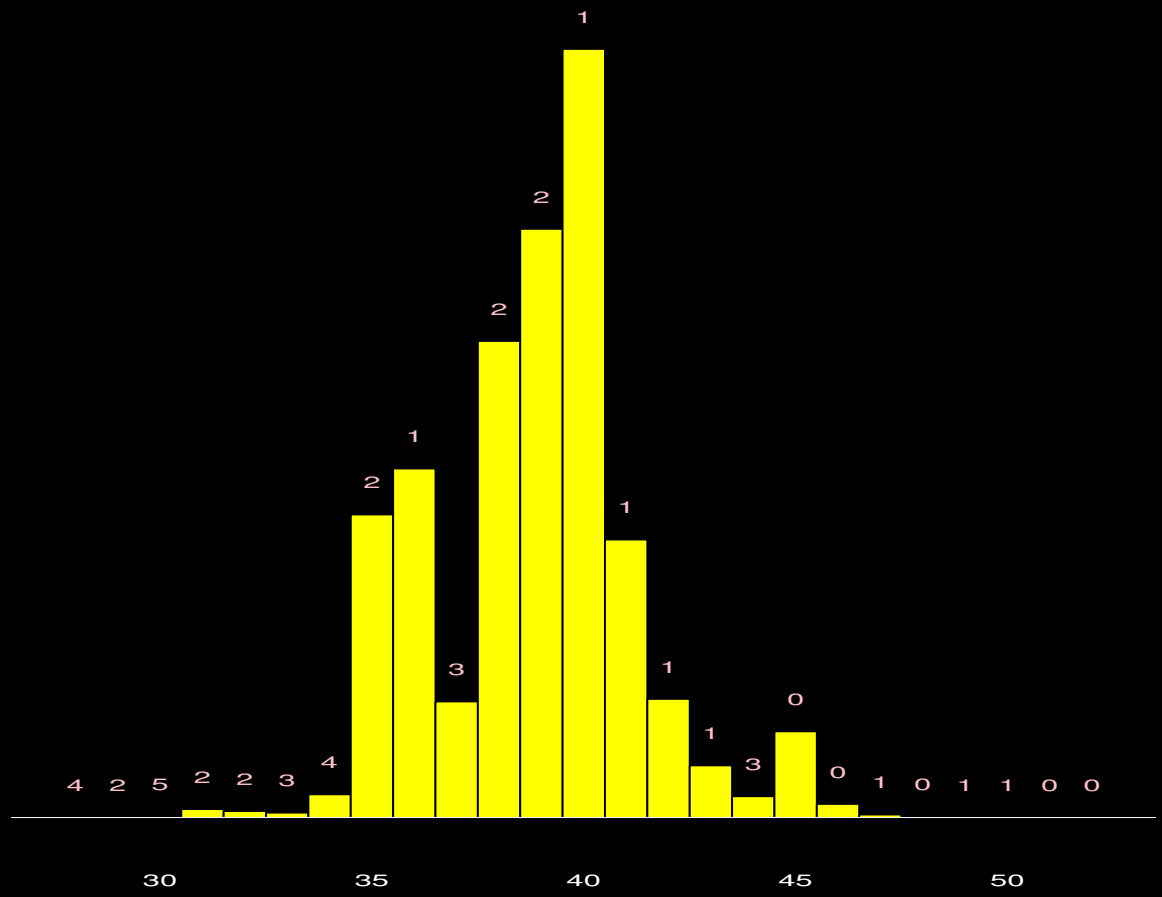
- Setup and Running:

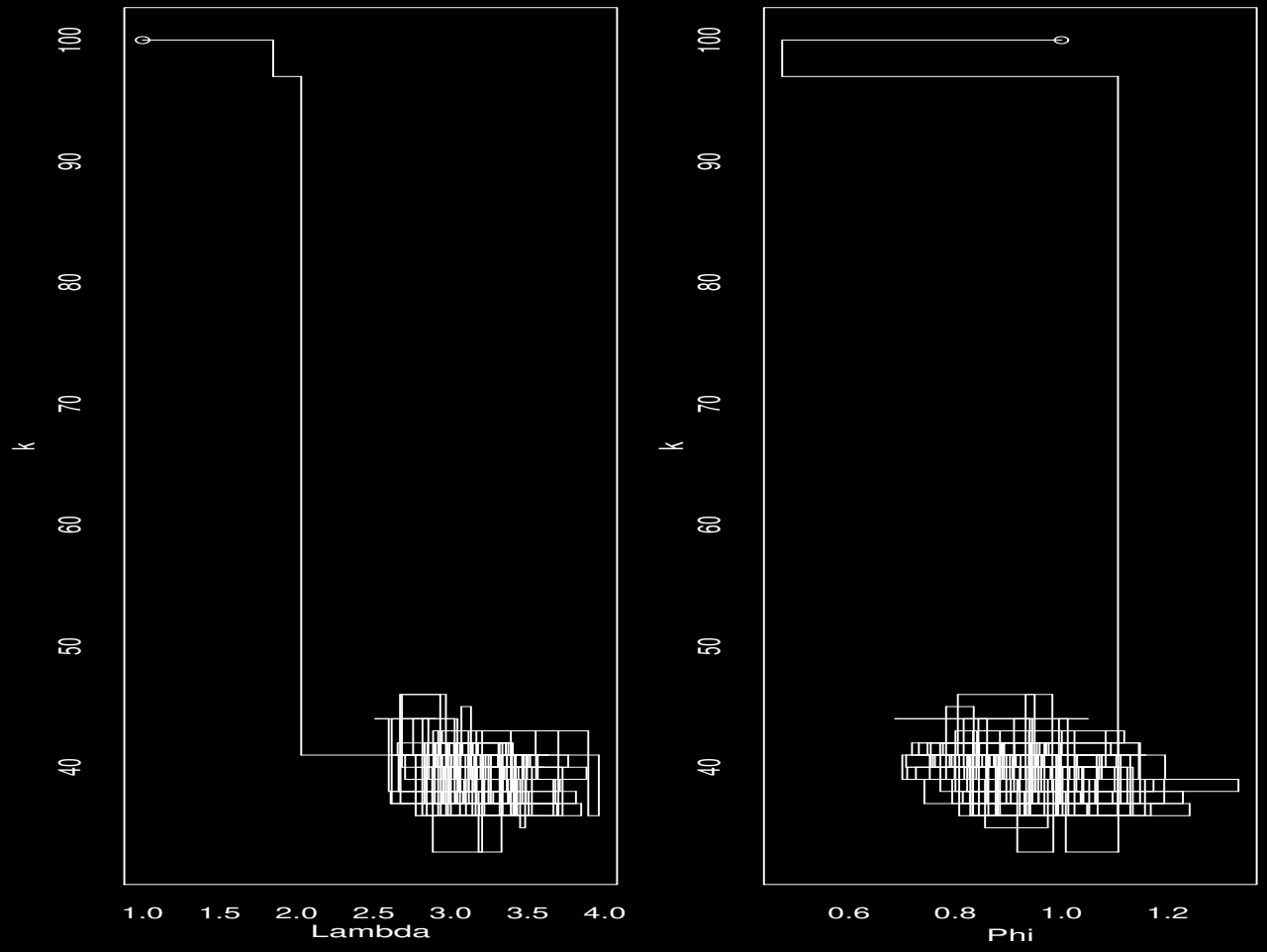
```
coal.mining.disasters <- c(4,5,4,0,1,4,3,4,0,6,3,3,4,0,2,6,
            3,3,5,4,5,3,1,4,4,1,5,5,3,4,2,5,
            2,2,3,4,2,1,3,2,2,1,1,1,1,3,0,0,
            1,0,1,1,0,0,3,1,0,3,2,2,0,1,1,1,
            0,1,0,1,0,0,0,2,1,0,0,0,1,1,0,2,
            3,3,1,1,2,1,1,1,1,2,4,2,0,0,1,4,
            0,0,0,1,0,0,0,0,0,1,0,0,1,0,1)
num.reps <- 2000
coal.mat <- matrix(NA,ncol=3,nrow=num.reps)
coal.mat[1,] <- c(1,1,100)
alpha <- 4; beta <- 1; gamma <- 1; delta <- 2
coal.mat <- bcp(coal.mat,coal.mining.disasters,alpha,beta,gamma,delta)
summary(coal.mat[1000:2000,])
```

The Gibbs Sampler, Realistic Example (cont.)

- This function is run for 2,000 iterations summarizing the last 1,000 chain values.

Quantile	λ	ϕ	k
Minimum	2.286	0.5975	32.00
First quartile	2.946	0.8616	39.00
Median	3.132	0.9359	40.00
Third quartile	3.339	1.0131	41.00
Maximum	4.337	1.4018	47.00
Mean	3.154	0.9447	39.77


4 2 5 2 2 3 4 2 1 3 2 2 1 1 1 3 0 0 1 0 1 1 0 0
30
35
40
45
50


k
100
90
80
70
60
50
40
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Lambda
k
100
90
80
70
60
50
40
0.6
0.8
1.0
1.2
Phi

Bayesian Tobit Model for Death Penalty Support

- Use the Tobit model (Tobin 1958) to look at social and political influences on U.S. state decisions to impose the death penalty since the Supreme Court ruled the practice constitutional in *Furman v. Georgia* 1972.

- Does the ideological, racial and religious makeup, political culture, and urbanization are causal effects for state-level death sentences from 1993 to 1995.

- The Tobit model is necessary to account for censoring here because 15 states did not have capital punishment provisions on the books in the studied period.

Bayesian Tobit Model for Death Penalty Support

- If $\mathbf{z}$ is a latent outcome variable in this context with the assumptions

$$\mathbf{z} = \mathbf{x}\boldsymbol{\beta} + \boldsymbol{\eta}$$

and

$$z_i \sim \mathcal{N}(\mathbf{x}\boldsymbol{\beta}, \sigma^2),$$

then the observed outcome variable is produced according to:

$$y_i = z_i \qquad \text{if} \qquad z_i > 0,$$

and

$$y_i = 0 \qquad \text{if} \qquad z_i \leq 0.$$

- The likelihood function is then:

$$L(\boldsymbol{\beta}, \sigma^2 | \mathbf{y}, \mathbf{X}) = \prod_{y_i=0} \left[1 - \Phi\left(\frac{x_i\boldsymbol{\beta}}{\sigma}\right)\right] \prod_{y_i>0} (\sigma^{-1}) \exp\left[-\frac{1}{2\sigma^2}(y_i - x_i\boldsymbol{\beta})^2\right].$$

Bayesian Tobit Model for Death Penalty Support

- A flexible parameterization for the priors is given by

$$\boldsymbol{\beta}|\sigma^2 \sim \mathcal{N}(\boldsymbol{\beta}_0, \mathbf{I}\sigma^2 B_0^{-1}) \qquad \sigma^2 \sim \mathcal{IG}\left(\frac{\gamma_0}{2}, \frac{\gamma_1}{2}\right)$$

  with vector hyperparameter $\boldsymbol{\beta}_0$, scalar hyperparameters B_0, $\gamma_0 > 2$, $\gamma_1 > 0$, and appropriately sized identity matrix $\mathbf{I}$.

- Substantial prior flexibility can be achieved with varied levels of these parameters, although values far from those implied by the data will make the Gibbs sampler algorithm run very slowly.

- The resulting joint posterior, $\pi(\boldsymbol{\beta}, \sigma^2, z|\mathbf{y}, \mathbf{X})$, is now analytically *intractible*, even with this basic model.

Bayesian Tobit Model for Death Penalty Support

- The full conditional distributions for Gibbs sampling are given for the $\boldsymbol{\beta}$ block, σ^2, and the individual $z_i|y_i = 0$ as:

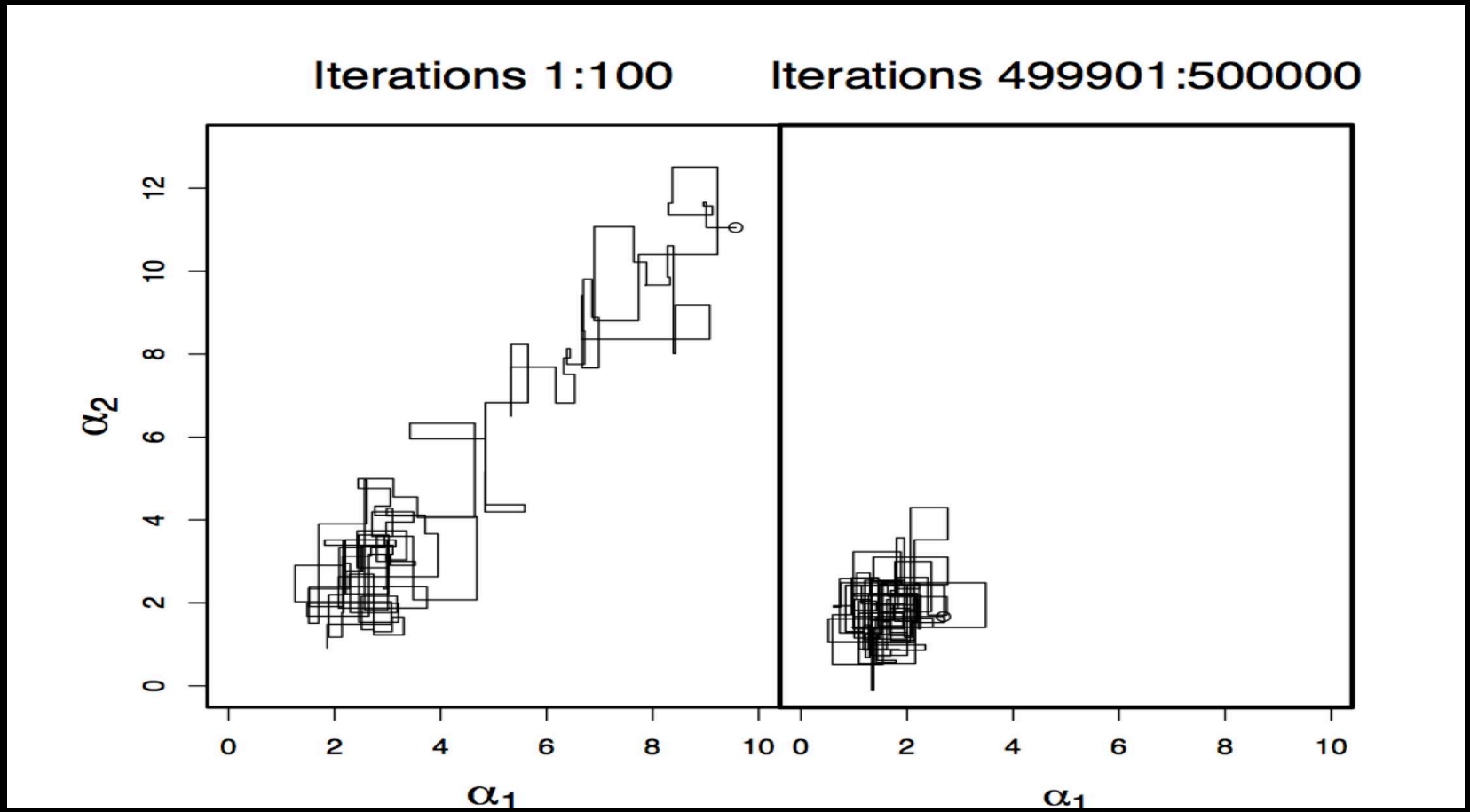
$$\boldsymbol{\beta}|\sigma^2, \mathbf{z}, \mathbf{y}, \mathbf{X} \sim \mathcal{N}\Big((B_0 + \mathbf{X}'\mathbf{X})^{-1})(\boldsymbol{\beta}_0 B_0 + \mathbf{X}'\mathbf{z}), \\ (\sigma^{-2}B_0 + \sigma^{-2}\mathbf{X}'\mathbf{X})^{-1}) \Big)$$

$$\sigma^2|\boldsymbol{\beta}, \mathbf{z}, \mathbf{y}, \mathbf{X} \sim \mathcal{IG}\left(\frac{\gamma_0 + n}{2}, \frac{\gamma_1 + (\mathbf{z} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{z} - \mathbf{X}\boldsymbol{\beta})}{2} \right)$$

$$z_i|y_i = 0, \boldsymbol{\beta}, \sigma, \mathbf{X} \sim \mathcal{TN}(\mathbf{X}\boldsymbol{\beta}, \sigma^2) I_{(-\infty,0)},$$

where $\mathcal{TN}()$ denotes the truncated normal and the indicator function $I_{(-\infty,0)}$ provides the bounds of truncation.

Gibbs Sampling Illustration



Posterior Summary, Tobit Model for Death Penalty Support

	$\bar{\beta}$	σ_β
Constant	-6.7600	3.5630
Past Rates	25.5586	8.0697
Political Culture	0.7919	0.1398
Current Opinion	5.9499	1.0805
Ideology	0.2638	1.0961
Murder Rate	0.1800	0.0764

The Metropolis-Hastings Algorithm

- Another type of stochastic process that will help us estimate posterior quantities.
- Background:
  - The original work by Metropolis et al. postulated a two-dimensional enclosure with $n = 10$ molecular particles.
  - They sought to estimate the state-dependent total energy of the system at equilibrium.
  - Of course there is an incredibly large number of locations for the molecules in the system that must be accounted for and this number grows exponentially with time.

The Metropolis-Hastings Algorithm

- Continued:
  - Their idea is to *simulate* this system probabilistically by generating moves that are more likely than others based on positions that are calculated using uniform probability generated candidate jump points.
  - Therefore the simulation produces an estimated force based on a statistical, rather than deterministic, arrangement of particles.
  - Obviously much easier.

Metropolis-Hastings Circa 1953

- Assumptions:
  - We want to describe the posterior: $\pi(\theta)$, which is difficult.
  - Candidate values will be generated from the distribution $q(\theta'|\theta)$ where $q(.|\theta)$ a valid PDF for all values of θ, meaning:
$$\int q(\theta', \theta) d\theta = 1$$
  - Also assume that this candidate generating (instrumental, jumping, or proposal) distribution is symmetric in its arguments:
$$q(\theta|\theta') = q(\theta'|\theta).$$
    Otherwise the Markov chain is not irreducible.
  - Note that the support of $\pi(\theta)$ and $q(\theta'|\theta)$ must be equivalent.

Metropolis-Hastings Circa 1953 (cont.)

- A a single Metropolis iteration from the symmetric form has the following steps:
  1. Sample θ' from $q(\theta'|\theta)$, where θ is the current location.
  2. Sample u from $u[0:1]$.
  3. If
  $$a(\theta', \theta) = \frac{\pi(\theta')}{\pi(\theta)} > u$$
  then accept θ'.
  4. Otherwise keep θ.

- The result is:
$$\theta_0, \theta_1, \theta_2, \ldots, \theta_n$$
where consecutive values are not necessarily unique.

Metropolis-Hastings Circa 1970

- Background:
  - It turns out that the symmetric requirement of the instrumental distribution was an annoying restriction.
  - Hastings (1970) as well as Peskun (1970) generalized the Metropolis et al. version by suggesting other jumping distributions such as the normal or Poisson.
  - Question: can an asymmetric instrumental distribution work if there is some sort of compensation in the acceptance ratio?
  - Yes (obviously), but there are conditions.

Metropolis-Hastings Circa 1970

- Generalizing:
  - Define again $K(\theta', \theta)$ as the kernel of the Metropolis-Hastings algorithm going from θ to θ'.
  - Sometimes $K(\theta', \theta)$ is labeled as $A(\theta', \theta)$ and called the *actual transaction function* from θ to θ' to distinguish it from $a(\theta', \theta)$.
  - For Metropolis algorithms, there is a two-step process:

$$A(\theta', \theta) = p(\text{actually moving}) = p(\text{proposal}) \times p(\text{acceptance}) = q(\theta'|\theta) \times \min\left[1, \frac{\pi(\theta')}{\pi(\theta)}\right]$$

  - We now require that the transition kernel satisfy the *detailed balance equation*:

$$A(\theta', \theta)\pi(\theta') = A(\theta, \theta')\pi(\theta),$$

    also called the *reversibility condition*, which can be rewritten:

$$\pi(\theta') = \frac{A(\theta, \theta')\pi(\theta)}{A(\theta', \theta)}$$

Metropolis-Hastings Circa 1970

- Generalizing, cont.:
  - Why? Because if the detailed balance equation holds, then:

$$\int A(\theta, \theta')\pi(\theta)d\theta = \int A(\theta', \theta)\pi(\theta')d\theta = \pi(\theta') \int A(\theta', \theta)d\theta = \pi(\theta')$$

  - So the detailed balance equation assures invariance, and we can obviously go the other direction too:

$$\int A(\theta', \theta)\pi(\theta')d\theta' = \int A(\theta, \theta')\pi(\theta)d\theta' = \pi(\theta) \int A(\theta, \theta')d\theta' = \pi(\theta)$$

  - Using $A(\theta', \theta)\pi(\theta') = A(\theta, \theta')\pi(\theta)$ we can rewrite the candidate posterior as:

$$\pi(\theta') = \frac{A(\theta, \theta')\pi(\theta)}{A(\theta', \theta)}$$

    and this means that irreducibility is still assured and we have a more general version of symmetry with *reversible* chains.

Metropolis-Hastings Circa 1970

- Generalizing, cont.:
  - Thus:
$$\frac{A(\theta', \theta)}{A(\theta, \theta')} = \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\pi(\theta')}{\pi(\theta)}.$$
  - This suggests, therefore, replacing:
$$a(\theta', \theta) = \frac{\pi(\theta')}{\pi(\theta)} > u \qquad \text{with:} \qquad a(\theta', \theta) = \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\pi(\theta')}{\pi(\theta)} > u.$$
  - Which means that we can use the same Metropolis-Hastings engine but replace symmetry with reversibility.
  - Examples:

    random walk chain, $\theta' = \theta + f(\tau)$
    (where τ is some convenient random variable form).
    independence chain, $\theta' = f(\tau)$
    (ignores current position entirely in the candidate generation step).

Metropolis-Hastings Circa 1970

- What does this imply?:

$$p(\text{actually moving}) = p(\text{proposal}) \times p(\text{acceptance})$$

$$A(\theta, \theta') = q(\theta'|\theta)\min\left[1, \frac{q(\theta|\theta')}{q(\theta'|\theta)}\frac{\pi(\theta')}{\pi(\theta)}\right]$$

which includes the reversibility assumption, now given that $\theta \neq \theta'$,

$$\begin{aligned}\pi(\theta)A(\theta, \theta') &= \pi(\theta)q(\theta'|\theta)\min\left[1, \frac{q(\theta|\theta')}{q(\theta'|\theta)}\frac{\pi(\theta')}{\pi(\theta)}\right] \\ &= \min\left[\pi(\theta)q(\theta'|\theta), \pi(\theta')q(\theta|\theta')\right]\end{aligned}$$

which is clearly now symmetric in θ and θ' because:

$$\begin{aligned}\pi(\theta')A(\theta', \theta) &= \pi(\theta')q(\theta|\theta')\min\left[1, \frac{q(\theta'|\theta)}{q(\theta|\theta')}\frac{\pi(\theta)}{\pi(\theta')}\right] \\ &= \min\left[\pi(\theta')q(\theta|\theta'), \pi(\theta)q(\theta'|\theta)\right]\end{aligned}$$

The Metropolis-Hastings Algorithm, Steps

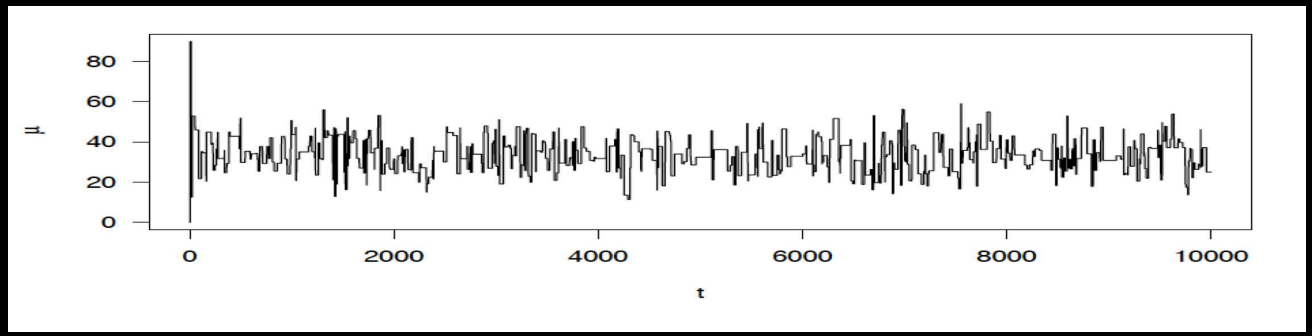
- The full algorithm:

1. transform (if necessary) θ, ($j \in J$ parameters in $\boldsymbol{\theta}$) so that it has the posterior distribution $\pi(\theta)$ with support over $\Re$.
2. at the t^{th} step in the chain, draw θ' from an candidate distribution, $q_t(\theta'|\theta)$, over the same support, such a normal random variable centered at the current value of θ.
3. define the *acceptance ratio*:
$$a(\theta', \theta) = \frac{q_t(\theta|\theta')}{q_t(\theta'|\theta)} \frac{\pi(\theta')}{\pi(\theta^{[t]})}.$$
4. make a decision about the $t + 1^{st}$ point in the chain probabilistically:
$$\theta^{[t+1]} = \begin{cases} \theta' & \text{with probability} \quad P\left[\min\left(a(\theta', \theta), 1\right)\right] \\ \theta^{[t]} & \text{with probability} \quad 1 - P\left(\min\left(a((\theta', \theta), 1)\right]\right. . \end{cases}$$

The Metropolis-Hastings Transition Kernel

- Three possible events can occur:
  1. sample a value of higher density and move with probability one,
  2. sample a value of lower density but move anyway by drawing a small uniform random variable in Step 2 above,
  3. or draw a uniform random variable larger than the ratio of posteriors and therefore stay in the same location.

- So unlike the Gibbs sampler, movement is not required on each iteration.



The Metropolis-Hastings Transition Kernel (cont.)

- Transitioning from $\theta = \theta^{[k]}$ to the current value (that is, not moving at all) can occur two ways: a successful transition to the current state and a failed transition to a different state:

$$p(\theta,\theta) = \underbrace{p(\theta^{[k+1]}=\theta), I(\theta^{[k+1]})|\theta^{[k]}=\theta)}_{\text{moving back to same point}} + \underbrace{p(\theta^{[k+1]}\neq\theta), \neg I(\theta^{[k+1]})|\theta^{[k]}=\theta)}_{\text{not moving}}$$

$$= q(\theta,\theta) + \sum_{\theta'\neq\theta} q(\theta',\theta)\left(1 - \min\left[1, \frac{q(\theta,\theta')\pi(\theta')}{q(\theta',\theta)\pi(\theta)}\right]\right).$$

The Metropolis-Hastings Transition Kernel (cont.)

- **Key Point**: the choice for the jumping distribution variance really matters.
  - If the variance is too large, then the jumping distribution will be too wide relative to the target distribution and each successive step will move too far in some direction
  - If the jumping distribution variance is too small causing overly cautious small steps through the sample space and the chain converge slowly and mix poorly through the limiting distribution once we have converged.

Simple Metropolis-Hastings Example

- Suppose we want to simulate the bivariate normal distribution: $\mathcal{N}\left[\begin{array}{c|cc} 1 \\ 2 & 1.0 & 0.9 \\ & 0.9 & 1.0 \end{array}\right]$.
- Lets disadvantage the algorithm somewhat by sampling from a bivariate t-distribution, starting with four overdispersed points in the sample space.
- A simple example, so it better work!

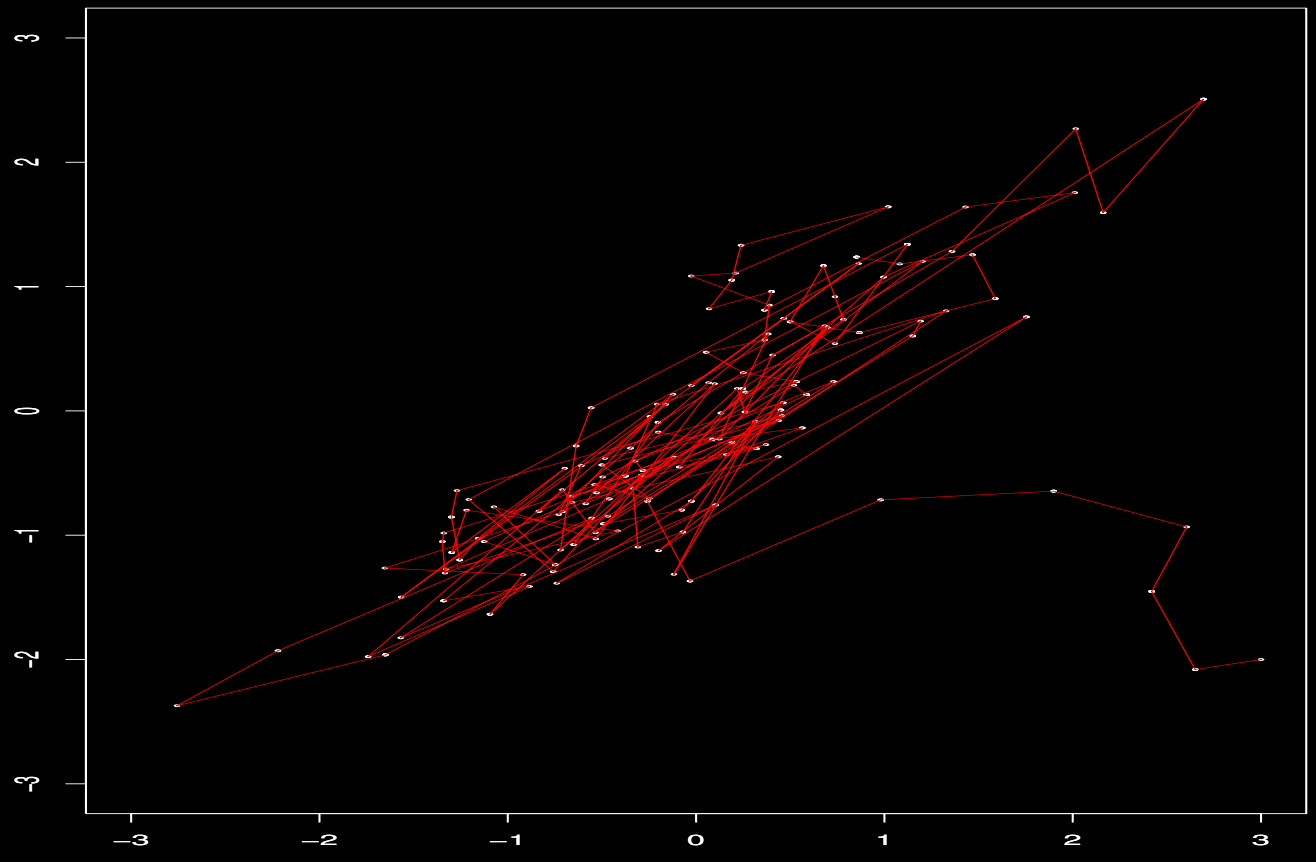
Simple Metropolis-Hastings Example

```
metropolis <- function(num.iter,I.mat) {
   theta.matrix <- rbind(c(3,-2),matrix(NA,nrow=num.iter,ncol=2))
   for (i in 2:nrow(theta.matrix)) {
     theta.prime <- mvrnorm(1,theta.matrix[(i-1),],I.mat)/(sqrt(rchisq(2,5)/5))
     a <-dmultinorm(theta.prime[1],theta.prime[2],c(0,0),I.mat)/
       dmultinorm(theta.matrix[(i-1),1],theta.matrix[(i-1),2], c(0,0),I.mat)
     if (a > runif(1)) theta.matrix[i,] <- theta.prime
     else theta.matrix[i,] <- theta.matrix[(i-1),]
   }
   theta.matrix
}
```

Simple Metropolis-Hastings Example

```
mcmc.mat <- metropolis(300,matrix(c(1,0.9,0.9,1),ncol=2))
par(mfrow=c(1,1),mar=c(3,2,2,2),oma=c(3,3,1,1),col.axis="white",col.lab="white",
  col.sub="white",col="white",bg="black")
plot(mcmc.mat,pch=1,cex=0.3,xlim=c(-3,3),ylim=c(-3,3))
lines(mcmc.mat,lwd=0.5,col="red")
axis(side=1,at=-3:3,NULL,col="white"); axis(side=2,at=-3:3,NULL,col="white")
```

Simple Metropolis-Hastings Example



A Not-Simple Metropolis-Hastings Example

- "New Findings from Terrorism Data: Dirichlet Process Random Effects Models for Latent Groups." (Kyung, Gill and Casella, Journal of the Royal Statistical Society-C, 2010).

- We develop a logistic random effects specification using a Dirichlet process to model the random effects, uncovering latent information that would not otherwise have been revealed

- In generating the Dirichlet process parameters we first update the partition matrix $\mathbf{A}$ and then the precision parameter m.

A Not-Simple Metropolis-Hastings Example

- Given the model parameters at iteration t, $\theta^{(t)} = (\boldsymbol{\eta}^{(t)}, \boldsymbol{\beta}^{(t)}, \tau^{2(t)})$, we generate

$$\mathbf{q}^{(t+1)} = (q_1^{(t+1)}, \ldots q_n^{(t+1)}) \sim \text{Dirichlet}(n_1^{(t)} + 1, \ldots, n_k^{(t)} + 1, 1, \ldots, 1)$$

$$\mathbf{A}^{(t+1)} \sim P(\mathbf{A}') f(\mathbf{y}|\theta^{(t+1)}, \mathbf{A}') \binom{n}{n_1 \ \cdots \ n_{k'}} \prod_{j=1}^{k'} [q_j^{(t+1)}]^{n'_j}$$

where $n_j > 0$, $n_1 + \cdots + n_{k'} = n$, and the probability of the candidate is given by

$$P\left(\mathbf{A}^{(t+1)}\right) = \frac{\Gamma(n)}{\Gamma(n - k^{(t+1)} + 1)} \frac{\Gamma(n_{k^{(t+1)}}^{(t+1)} + n - k^{(t+1)} + 1)}{\Gamma(2n)} \prod_{j=1}^{k^{(t+1)}-1} \Gamma(n_j^{(t+1)} + 1)$$

- So values of $\mathbf{q}^{(t+1)}$ and $\mathbf{A}^{(t+1)}$ at step $t+1$ and jointly accepted or rejected.

Table 1: Global Suicide Attack Data–coefficients, standard errors, and HPD intervals from a Bayesian random effects logit model (BRELM), and from the generalized linear mixed Dirichlet model (GLMDM) using a logit link.

	BRELM				GLMDM Logit			
Coefficient	COEF	SE	95% HPD		COEF	SE	95% HPD	
Intercept	-6.457	4.232	-21.605	-3.407	-4.105	0.559	-5.276	-3.079
YEAR - 1998	0.303	0.228	0.135	1.137	0.195	0.039	0.121	0.273
MULT.INCIDENT	-0.802	0.488	-2.222	-0.142	-0.585	0.221	-1.028	-0.162
MULTI.PARTY	-0.945	0.690	-3.289	-0.225	-0.626	0.229	-1.088	-0.189
SUSP.UNCONFIRM	-0.109	0.344	-0.928	0.472	-0.061	0.198	-0.455	0.331
SUCCESSFUL	-1.035	0.705	-3.308	-0.262	-0.695	0.245	-1.172	-0.210
ATTACK.TYPE	0.122	0.135	-0.122	0.466	0.098	0.073	-0.046	0.240
WEAPON.TYPE	2.714	1.673	1.346	7.769	1.725	0.320	1.162	2.422
TARGET.TYPE	-0.073	0.330	-0.749	0.527	-0.038	0.185	-0.434	0.323
NUM.FATAL	-0.019	0.025	-0.085	0.017	-0.013	0.012	-0.036	0.009
NUM.INJUR	0.030	0.030	0.010	0.126	0.017	0.004	0.008	0.025
PSYCHOSOCIAL	0.824	0.633	0.216	3.044	0.555	0.192	0.188	0.944
PROPERTY.DAMAGE	0.439	0.305	0.122	1.406	0.297	0.094	0.114	0.483

A Not-Simple Metropolis-Hastings Example

The Hit-and-Run Algorithm

- See (Bélisle, Romeijn, and Smith 1993; Chen and Schmeiser 1993).
- A special case of the Metropolis-Hastings algorithm.
- Separates the move decision into a direction decision and a distance decision.
- This makes it especially useful in tightly constrained parameter space because we can tune the jumping rules to be more efficient.
- Also helpful when there are several modes of nearly equal altitude.
- This is an example of the flexibility of the Metropolis-Hastings algorithm.

Hit-and-Run Algorithm, Steps

- From an arbitrary point $\boldsymbol{\theta}_t$, at time t:

Step 1: Generate a multidimensional direction, $\mathbf{Dr}_t$, on the surface of a unit hypersphere from the distribution $f(\mathbf{Dr}|\boldsymbol{\theta}^{[t]})$.

Step 2: Generate a signed distance, Ds_t, from density $g(Ds|\mathbf{Dr}_t, \boldsymbol{\theta})$.

Step 3: Set the candidate jumping point to: $\boldsymbol{\theta}' = \boldsymbol{\theta}^{[t]} + Ds_k\mathbf{Dr}_t$ and calculate:

$$a(\boldsymbol{\theta}', \boldsymbol{\theta}^{[t]}) = \frac{\pi(\boldsymbol{\theta}'|\mathbf{X})}{\pi(\boldsymbol{\theta}^{[t]}|\mathbf{X})}$$

Step 4: Move to $\boldsymbol{\theta}^{[5+1]}$ according to:

$$\boldsymbol{\theta}^{[t+1]} = \begin{cases} \boldsymbol{\theta}' & \text{with probability} \quad P\left[\min\left(a(\boldsymbol{\theta}', \boldsymbol{\theta}^{[t]}), 1\right)\right] \\ \boldsymbol{\theta}^{[t]} & \text{with probability} \quad 1 - P\left(\min\left(a(\boldsymbol{\theta}', \boldsymbol{\theta}^{[t]}), 1\right)\right]. \end{cases}$$

Hit-and-Run Algorithm, Required Assumptions

- For all $\mathbf{Dr}_i$, $f(\mathbf{Dr}|\boldsymbol{\theta}^{[t]}) > 0$.
- $g(Ds|\mathbf{Dr}, \boldsymbol{\theta})$ must be strictly greater than zero and have the property:

$$g(Ds|\mathbf{Dr}, \boldsymbol{\theta}) = g(-Ds| - \mathbf{Dr}, \boldsymbol{\theta}).$$

- The new form of the detailed balance equation is:

$$g(||\boldsymbol{\theta}^{[t]} - \boldsymbol{\theta}'||)a(\boldsymbol{\theta}', \boldsymbol{\theta}^{[t]})\pi(\boldsymbol{\theta}^{[t]}|\mathbf{X}) = g(||\boldsymbol{\theta}' - \boldsymbol{\theta}^{[t]}||)a(\boldsymbol{\theta}^{[t]}, \boldsymbol{\theta}')\pi(\boldsymbol{\theta}'|\mathbf{X})$$

- *Then this is an ergodic Markov chain with stationary distribution* $\pi(\boldsymbol{\theta}|\mathbf{X})$.

Hit-and-Run Algorithm, Comments

- Typically $f(\mathbf{Dr}|\boldsymbol{\theta}^{[t]})$ is chosen to be uniform but others are possible (Von Mises?).
- The $a(\boldsymbol{\theta}', \boldsymbol{\theta}^{[t]})$ criterion can be made much more general.
- One advantage to this algorithm over standard M-H is that $g(Ds|\mathbf{Dr}, \boldsymbol{\theta})$ is also flexible and disengaged from the direction decision. Very tunable.
- Additional development: *adaptive directional sampling*.

Hit-and-Run Example

- Lets use this algorithm to simulate from a bivariate normal.
- Not really necessary of course, but shows integrity of the process.
- Actually H-R is really good with *problematic* forms.

Hit-and-Run Example

```
hit.run <- function(num.iter,I.mat) {
  theta.mat <- rbind(c(0,0),matrix(NA,nrow=num.iter,ncol=2))
  for (i in 2:nrow(theta.mat)) {
    rad.draw <- runif(1,0,2*pi)
    if ((rad.draw > 0 & rad.draw < pi/2) | (rad.draw > pi & rad.draw < 3*pi/2))
       distance <- rgamma(1,1,3)
    else  distance <- rgamma(1,1,6)
    xy.theta <- c(distance*cos(rad.draw),distance*sin(rad.draw)) + theta.mat[(i-1),]
    a <- dmultinorm(xy.theta[1], xy.theta[2],      c(0,0),I.mat)/
      dmultinorm(theta.mat[(i-1),1], theta.mat[(i-1),2],c(0,0),I.mat)
    u <- runif(1)
    if (runif(1) < a) { theta.mat[i,] <- xy.theta }
    else theta.mat[i,] <- theta.mat[(i-1),]
  }
  theta.mat
}
```

Hit-and-Run Example

```
mcmc.mat <- hit.run(10000,matrix(c(1,0.95,0.95,1),ncol=2))
postscript("Class.Multilevel/hit.run.contours.ps")
par(mfrow=c(1,1),mar=c(3,2,2,2),oma=c(3,3,1,1),col.axis="white",col.lab="white",
  col.sub="white",col="white",bg="black")
plot(mcmc.mat[(nrow(mcmc.mat)/2+1):nrow(mcmc.mat),],pch=1,cex=0.1,
  xlim=c(-4,4),ylim=c(-4,4))

alpha.grid <- seq(-4,4,length=300)
beta.grid <- seq(-4,4,length=300)
density <- outer(alpha.grid,beta.grid,dmultinorm,mu.vector=c(0,0),
  sigma.matrix=matrix(c(1,0.95,0.95,1),ncol=2))
contour(alpha.grid,beta.grid,density,levels=c(.2,.1,.05),col="red",add=TRUE)
dev.off()
```

Hit-and-Run Example