STAT3016/4116/7016:
Introduction to Bayesian Data Analysis

RSFAS, College of Business and Economics, ANU

Posterior approximation with the Gibbs sampler

Introduction

The models we have looked at so far have led to standard posterior distributions from which it is easy to sample directly from.

For many multiparameter models, the joint posterior distribution is non-standard and difficult to sample directly from.

But it may be easy to sample from the full conditional distribution of each parameter.

For these cases, we can use the Gibbs sampler to approximate the joint posterior distribution.

Gibbs sampling

The Gibbs sampler is an iterative algorithm. Suppose the parameter vector of interest is $\phi = (\phi_1, ..., \phi_d)$. At each iteration 't' of the Gibbs sampler , each ϕ_j^t (j=1,...,d) is sampled from the conditional distribution

$$p(\phi_j | \phi_{-j}^{t-1}, y)$$

where ϕ_{-j}^{t-1} represents all component of ϕ, except ϕ_j at their current values:

$$\phi_{-j}^{t-1} = (\phi_1^t, \phi_2^t, ..., \phi_{j-1}^t, \phi_{j+1}^{t-1}, ..., \phi_d^{t-1})$$

The iterations continue until the distribution converges to the target joint posterior distribution $p(\phi_1, ..., \phi_d | y)$

A semiconjugate prior distribution for the normal model

Let's modify the normal model and assume joint prior independence $p(\theta, \sigma^2) = p(\theta)p(\sigma^2)$. Moreover, let's assume the following "semiconjugate" prior distribution:

$$\theta \sim \text{normal}(\mu_0, \tau_0^2)$$
$$1/\sigma^2 \sim \text{gamma}(\nu_0/2, \nu_0\sigma_0^2/2)$$

$\sigma^2|y \rightarrow$ not gamma
$\sigma^2|y, \theta \rightarrow$ inv.-gamma
$\theta|\sigma^2, y \rightarrow$ normal

where τ_0^2 is not proportional to σ^2.

In this case, we can show that the marginal posterior distribution of $1/\sigma^2$ is not a gamma distribution, or any other standard distribution from which we can easily sample. But we can derive the conditional posterior distribution of $1/\sigma^2$ given θ and y which is:

$$\sigma^2|\theta, y_1, ..., y_n \sim \text{inv-gamma}(\nu_n/2, \nu_n\sigma_n^2/2)$$

where $\nu_n = \nu_0 + n$ and $\sigma_n^2 = \frac{\nu_0\sigma_0^2 + (n-1)s^2 + n(\bar{y}-\theta)^2}{\nu_n}$

Gibbs sampling - the normal model

The distributions $p(\theta|\sigma^2, y_1, ..., y_n)$ and $p(\sigma^2|\theta, y_1, ..., y_n)$ are called the full conditional distributions of θ and σ^2 respectively.

To implement the Gibbs sampler to obtain posterior draws of θ and σ^2, suppose the current state of the parameters is $\phi^{(s)} = \{\theta^{(s)}, \sigma^{2(s)}\}$. We generate a new state as follows:

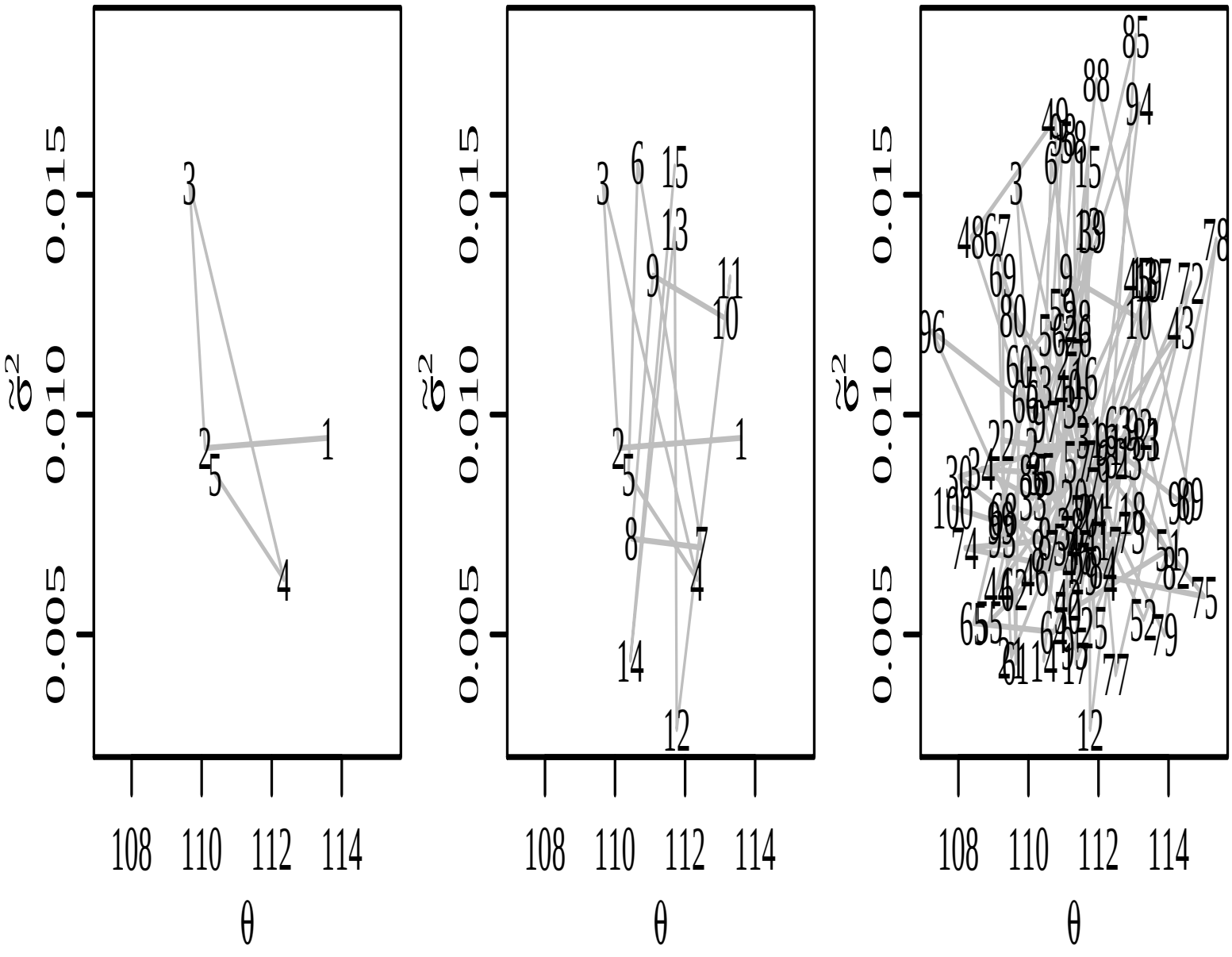
1. sample $\theta^{(s+1)} \sim p(\theta|\sigma^{2(s)}, y_1, ..., y_n)$;
2. sample $\sigma^{2(s+1)} \sim p(\sigma^2|\theta^{(s+1)}, y_1, ..., y_n)$;
3. let $\phi^{(s+1)} = \{\theta^{(s+1)}, \sigma^{2(s+1)}\}$

From the Gibbs sampler, we generate a *dependent* sequence of parameters $\{\phi^{(1)}, ..., \phi^{(S)}\}$

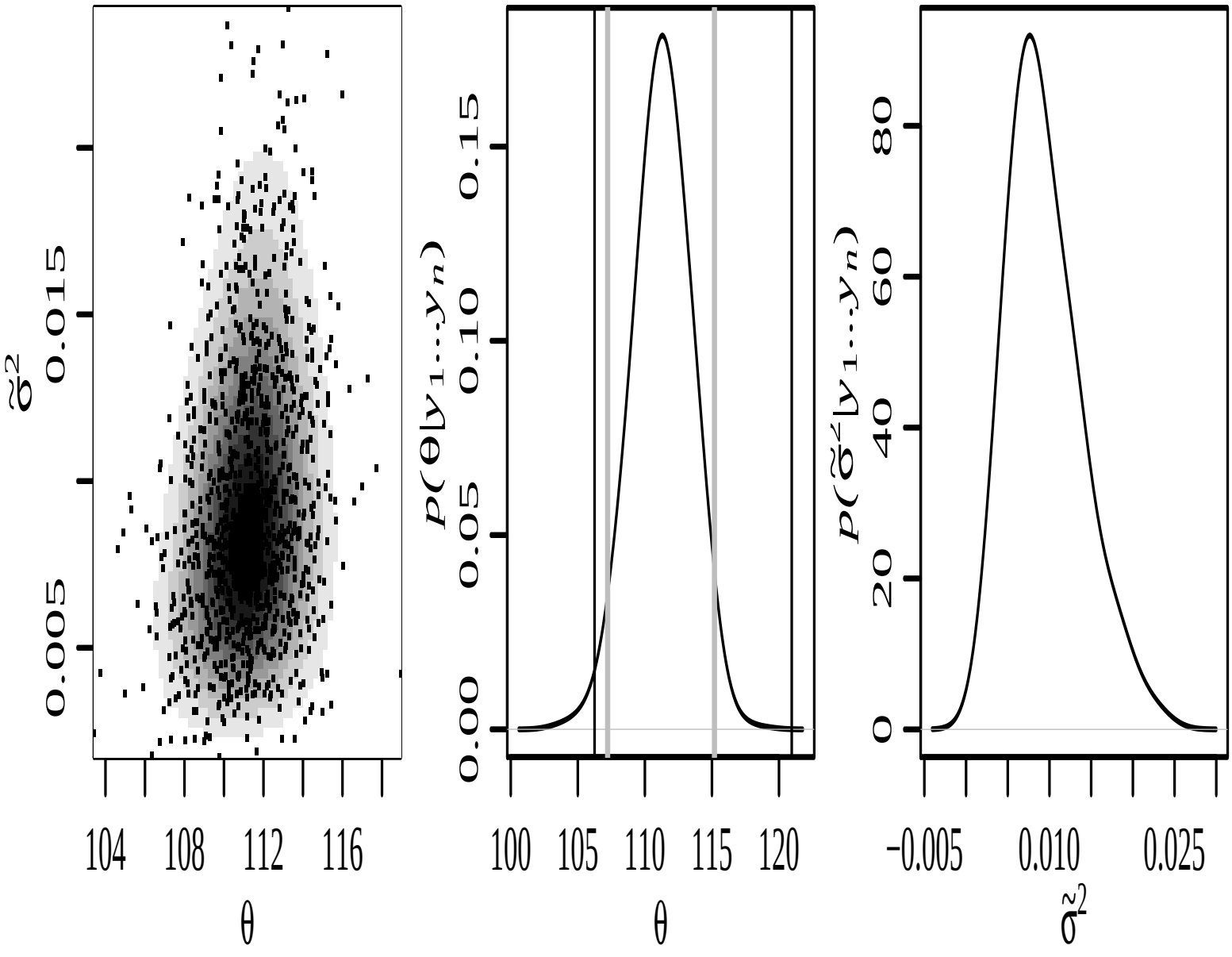
Gibbs sampling - implementation in R - Cars speed data example

```
S<-1000
PHI<-matrix(nrow=S,ncol=2)
PHI[1,]<-phi<-c( mean.y, 1/var.y)

### Gibbs sampling
for(s in 2:S) {

# generate a new theta value from its full conditional
mun<-  ( mu0/t20 + n*mean.y*phi[2] ) / ( 1/t20 + n*phi[2] )
t2n<- 1/( 1/t20 + n*phi[2] )
phi[1]<-rnorm(1, mun, sqrt(t2n) )

# generate a new 1/sigma^2 value from its full conditional
nun<- nu0+n
s2n<- (nu0*s20 + (n-1)*var.y + n*(mean.y-phi[1])^2 ) /nun
phi[2]<- rgamma(1, nun/2, nun*s2n/2)

PHI[s,]<-phi          }
```

Gibbs sampling - implementation in R



Gibbs sampling - implementation in R



Properties of the Gibbs sampler

Suppose we have a vector of parameters $\phi = \{\phi_1, ..., \phi_d\}$. Information about ϕ is measured with $p(\phi)$. Running the Gibbs sampler algorithm we can generate a *dependent* sequence of vectors:

$$\phi^{(1)} = \left\{\phi_1^{(1)}, ..., \phi_d^{(1)}\right\}$$

$$\phi^{(2)} = \left\{\phi_1^{(2)}, ..., \phi_d^{(2)}\right\}$$

$$.....$$

$$\phi^{(S)} = \left\{\phi_1^{(S)}, ..., \phi_d^{(S)}\right\}$$

$\phi^{(s)}$ depends on $\phi^{(0)}, ..., \phi^{(s-1)}$ only through $\phi^{(s-1)}$. This is the Markov property and the sequence is called a Markov Chain.

Properties of the Gibbs sampler

Under some conditions

$$Pr(\phi^{(s)} \in A) \to \int_A p(\phi) d\phi \text{ as } s \to \infty$$

That is, the sampling distribution of $\phi^{(s)}$ converges to the target distribution. More importantly for most functions g of interest

$$\frac{1}{S}\sum_{s=1}^{S} g(\phi^{(s)}) \to E[g(\phi)] = \int g(\phi)p(\phi)d\phi \text{ as } s \to \infty$$

So $\frac{1}{S}\sum_{s=1}^{S} g(\phi^{(s)})$ is our Markov Chain Monte Carlo (MCMC) approximation for $E[g(\phi)]$

(Cars speed example)

```
> mean(PHI[,1])
[1] 111.278
> sum(PHI[,1]<115 & PHI[,1]>107)/S
[1] 0.946
```

Bivariate normal distribution example

Exercise 1: Consider a single observation (y_1, y_2) from a bivariate normally distributed population with unknown mean $\theta = (\theta_1, \theta_2)$ and known covariance matrix $\left(\begin{smallmatrix} 1 & \rho \\ \rho & 1 \end{smallmatrix}\right)$. With a uniform prior distribution on θ, the posterior distribution is

$$\begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \Big| y \sim N\left(\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \right)$$

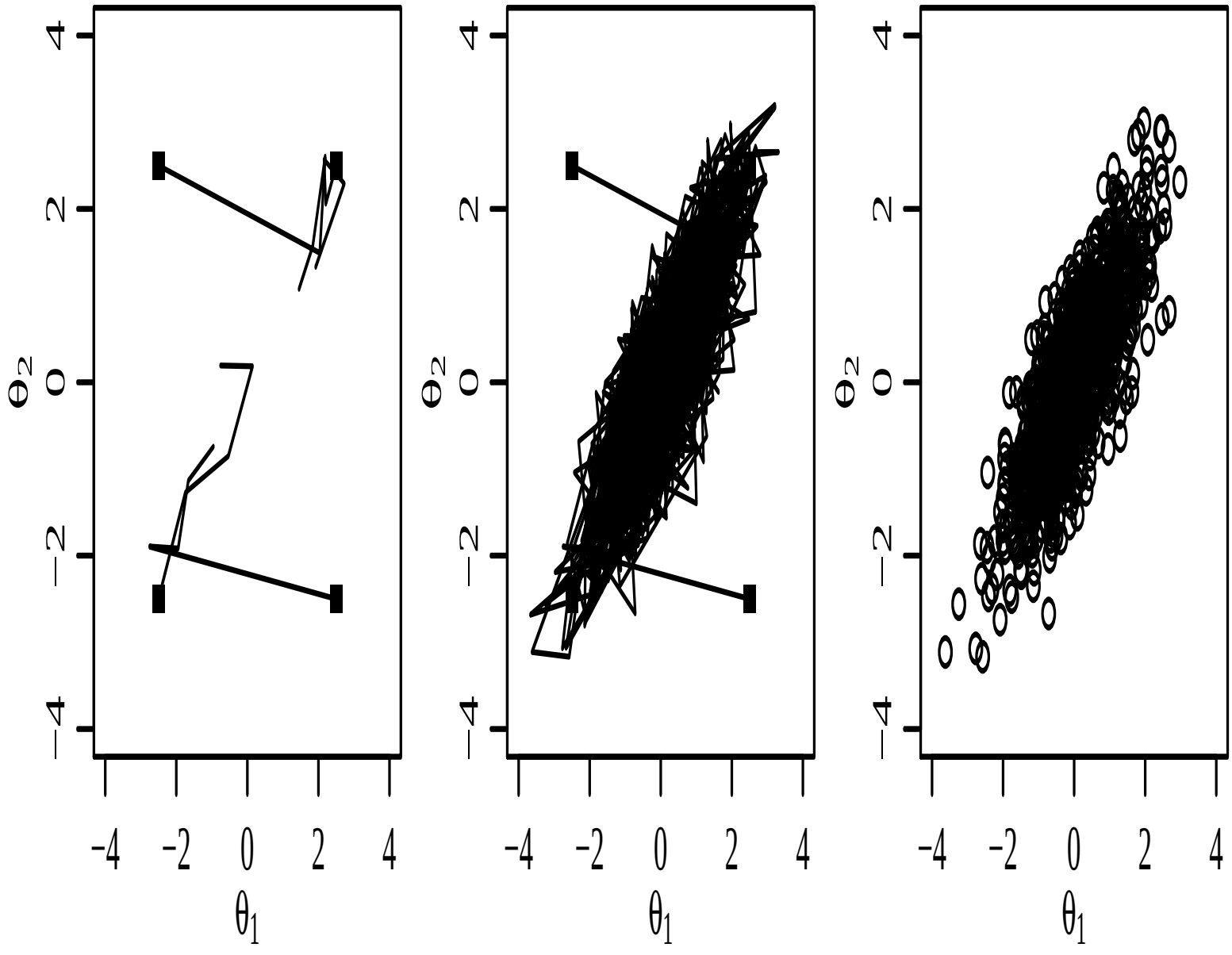
Let $\rho = 0.8$ and let $(y_1, y_2) = (0, 0)$. Use the Gibbs sampler to generate 500 posterior draws of (θ_1, θ_2).

You will need the following conditional distributions (derived from the properties of the multivariate normal distribution)

$$\theta_1 | \theta_2, y \sim N(y_1 + \rho(\theta_2 - y_2), 1 - \rho^2)$$

$$\theta_2 | \theta_1, y \sim N(y_2 + \rho(\theta_1 - y_1), 1 - \rho^2)$$

Bivariate normal distribution example



Binomial-Beta-Poisson example

Exercise 2: Let $y \sim Bin(n, \theta)$, where both n and θ are unknown parameters. Assume a Beta(a,b) prior for θ and a Pois(λ) prior for n.

Write out the steps of the Gibbs sampling algorithm to obtain posterior draws of (n, θ).

Check your work by implementing a Monte Carlo study with some values of n, θ and y of your choice.

Important note

Monte Carlo and MCMC sampling algorithms:

- are not models.
- they do not generate "more information" than is in $\mathbf{y}$ and $p(\phi)$.
- they are simply "ways of looking at" $p(\phi|\mathbf{y})$

We use MCMC sampling algorithms because for many models $p(\phi|\mathbf{y})$ is hard to write down. A useful way to "look at" $p(\phi|\mathbf{y})$ is to study Monte Carlo samples from $p(\phi|\mathbf{y})$.

(note: modelling and estimation are satisfied once we have specified the prior and sampling model)

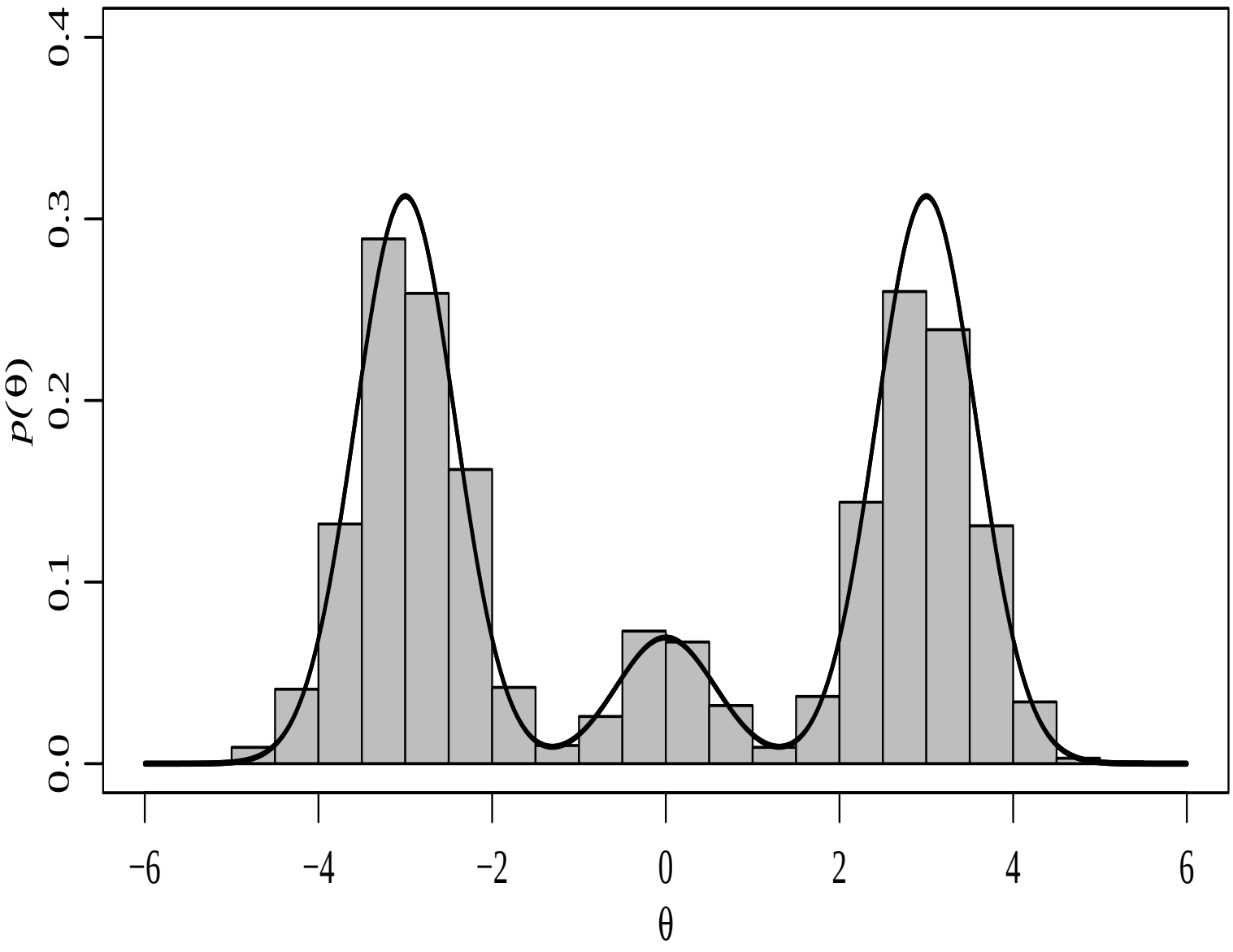
Introduction to MCMC diagnostics

How do we know that the simulated sequence $\{\phi^{(1)}, ..., \phi^{(S)}\}$ from the MCMC algorithm has converged to the target distribution?

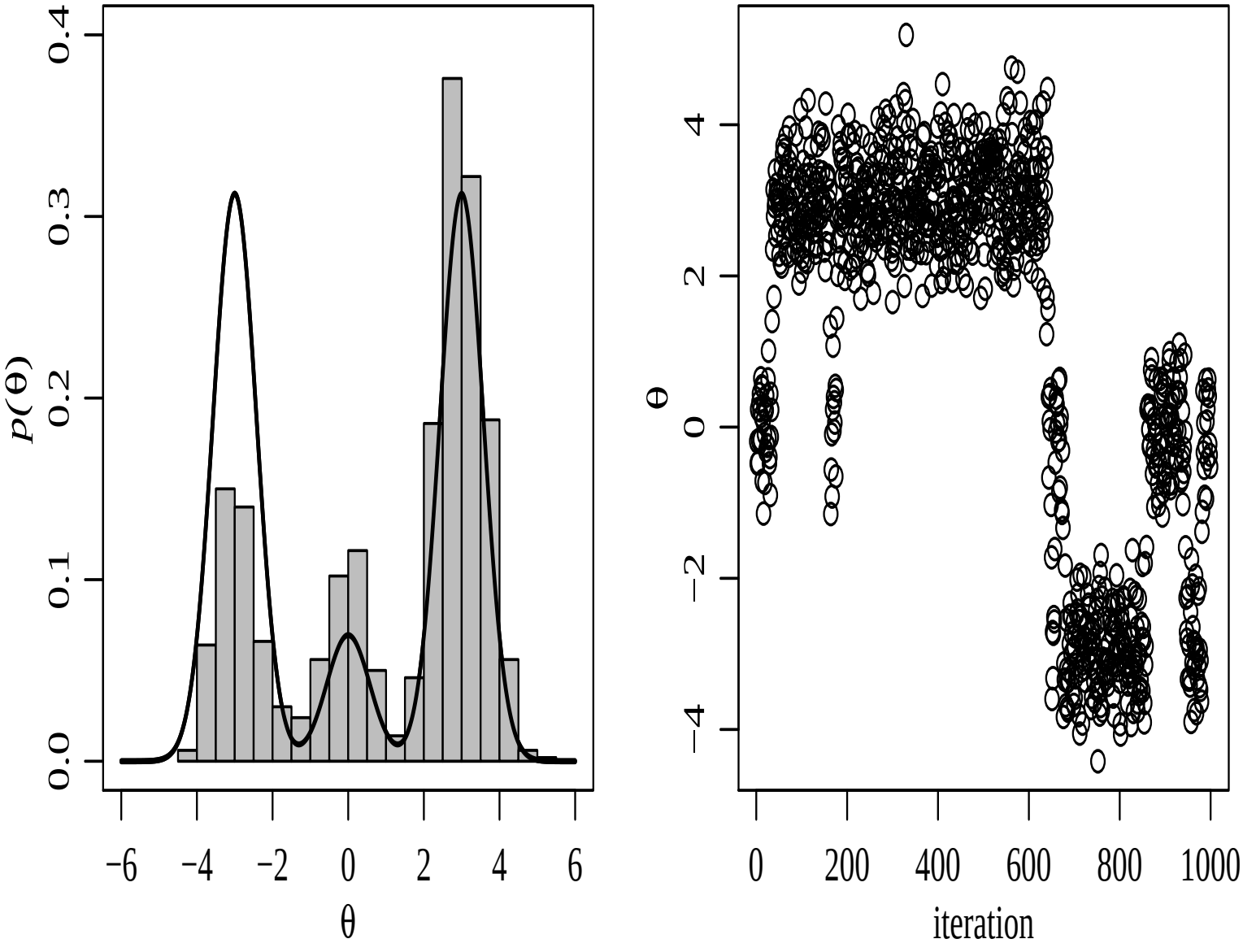
Example: Consider a joint probability distribution on $\delta = \{1, 2, 3\}$ and $\theta \in \mathbb{R}$. The target density is $\{Pr(\delta = 1), Pr(\delta = 2), Pr(\delta = 3)\} = (0.45, 0.10, 0.45)$ and $p(\theta|\delta) = \text{dnorm}(\theta, \mu_\delta, \sigma_\delta)$ where $(\mu_1, \mu_2, \mu_3) = (-3, 0, 3)$ and $(\sigma_1^2, \sigma_2^2, \sigma_3^2) = (1/3, 1/3, 1/3)$. What type of distribution is this? What do the δ represent??

Exercise 3: Generate Monte Carlo samples and MCMC samples with the Gibbs sampler for (θ, δ). Create 1000 samples of each. Compare your samples of θ to the exact marginal density of θ. (HINT: to implement the Gibbs sampler, you need $Pr(\delta = d|\theta)$ for $d \in \{1, 2, 3\}$)

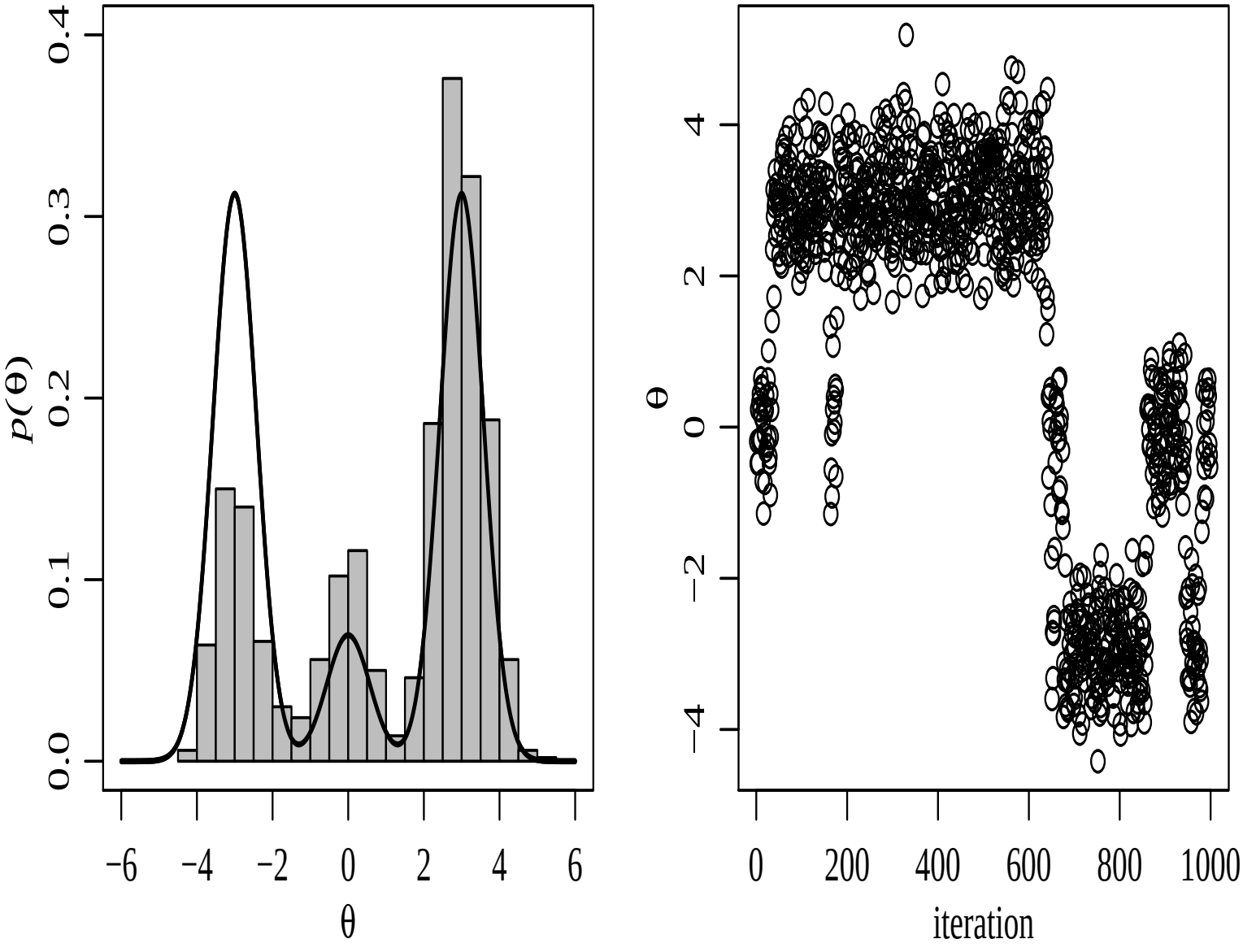
Monte Carlo Approximation



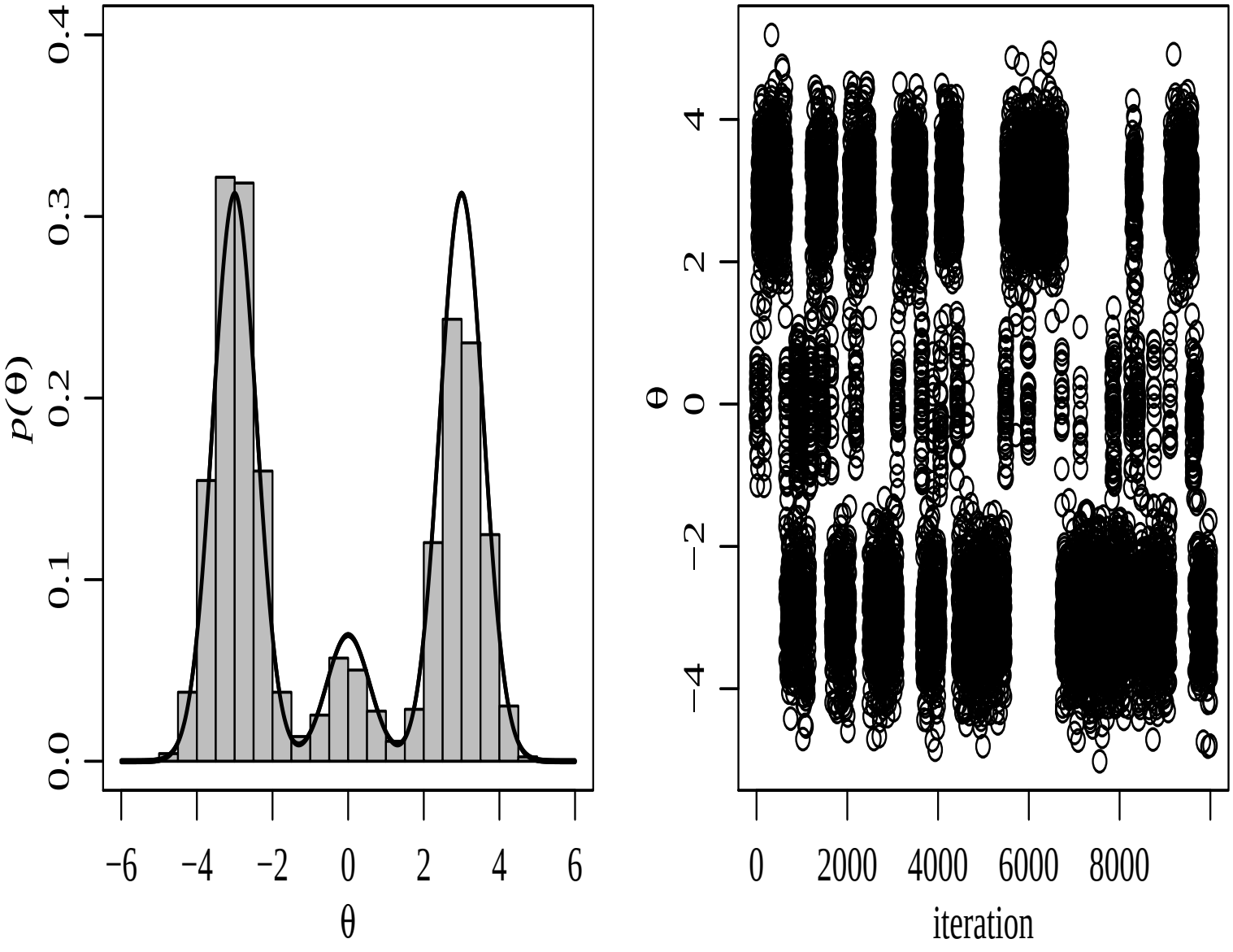
Histogram and traceplot of 1000 Gibbs samples



Histogram and traceplot of 1000 Gibbs samples



Histogram and traceplot of 10000 Gibbs samples



Stationarity and zero auto-correlation

Consider A_1 and A_2 and A_3 to be three disjoint subsets of the parameter space (corresponding to the regions near the three modes of the normal mixture distribution in Exercise 2). Consider the sequence $\left\{\phi^{(1)}, ..., \phi^{(S)}\right\}$ as the trajectory of a particle ϕ moving around the parameter space. We want the amount of time the particle spends in a particular region to be proportional to the target probability $\int_A p(\phi)d\phi$. For our example, this means $Pr(A_2) < Pr(A_1) \approx Pr(A_3)$.

Therefore we need our particle to

1. move out of A_2 and into higher probability regions; (stationarity/ convergence)
2. move between A_1 and A_3, and any other sets of high probability. (zero auto correlation)

Stationarity

Stationarity: Start the Gibbs sampler from different starting points and check that the samples converge to the same distribution.

To do so, we can check that samples taken in one part of the chain have a similar distribution to samples taken from another part of the chain. Apply a test (such as the Kolomogorov-Smirnov) to determine equality of the distributions.

Autocorrelation

if AC↑, dependency of generated parameters are high.

Autocorrelation: how quickly does the particle move around the parameter space (speed of mixing). Ideally we want perfect mixing- the chain jumps between different regions in one step.

Is $\mathrm{Var}_{\mathrm{MCMC}}[\bar{\phi}]$, $>$, $<$ or $=$ to $\mathrm{Var}_{\mathrm{MC}}[\bar{\phi}]$ (where $\bar{\phi} = \sum \phi^{(s)}/S$)?

To check the correlation at different lags 't', use the `acf()` in R. We want low-values of the autocorrelation. High values mean the chain moves around the parameter space slowly and we need more MCMC samples to attain a given level of precision for our approximation.

Autocorrelation - effective sample size

This Effective Sample Size gives an estimate of the equivalent number of independent iterations that the chain represents.

MC has effective sample size of the # of draws you pick.

$$M = \frac{n}{1 + 2\sum_{k=1}^{\infty} \rho_k}$$

$k \in (1, \infty)$

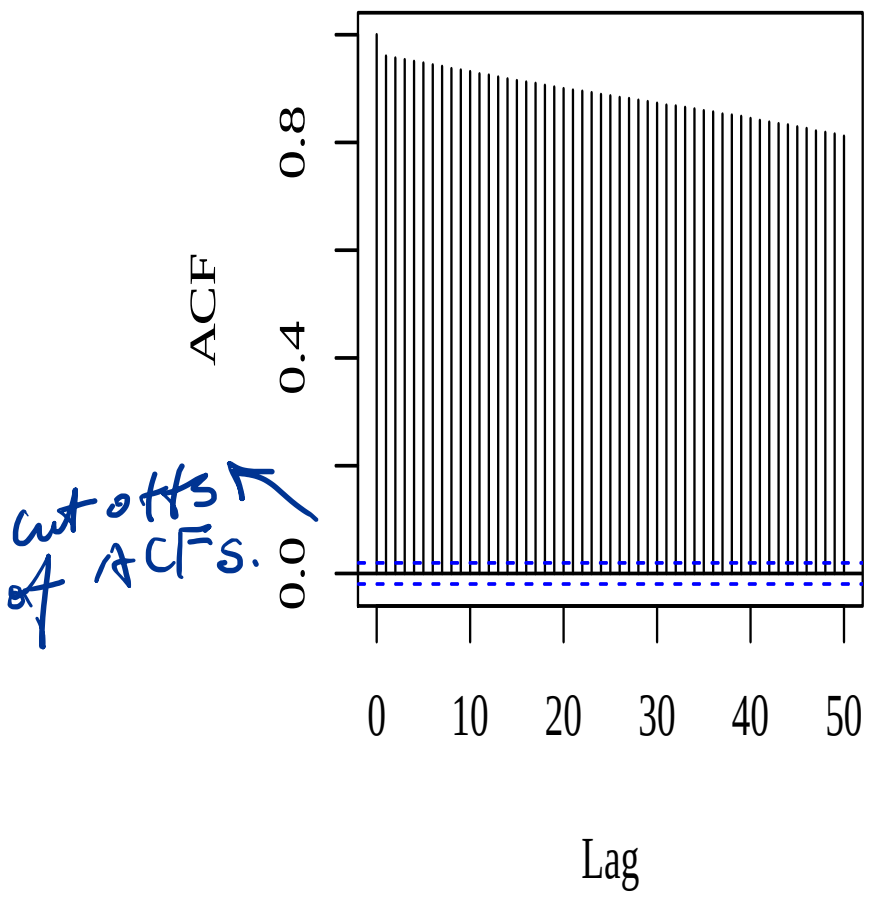
ρ_k autocorrelation of kth draw

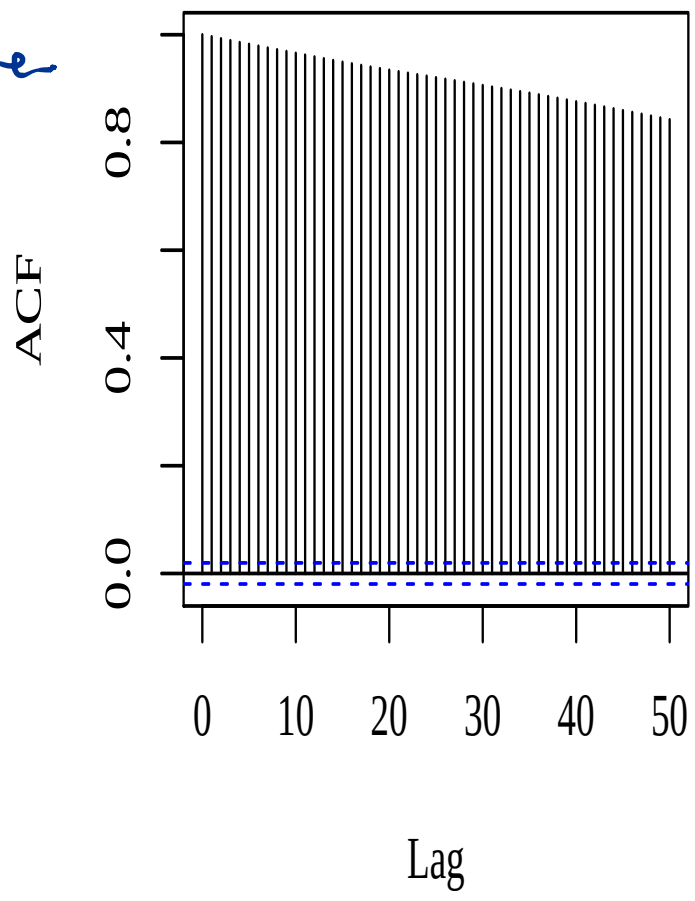
$\rho_k \uparrow$, $M \downarrow$

In R: 'coda' package, `effectiveSize()`

High values of M (close to the number of posterior draws n or $M > 1000$) are preferred.

MCMC diagnostics - ACF Plots - normal mixture example





cutoffs of ACFs.

all for beyond cutoffs here

high ACF for both paras. also M is ↓

MCMC diagnostics - effective sample size - - normal mixture example

see ch6. exercise codes.

(library (coda)

```
> effectiveSize(THD.MCMC[,1])
    var1
18.42419
> effectiveSize(THD.MCMC[,2])
    var1
16.12003
```

MCMC diagnostics

③ e.g. binomial model.
we don't generate p directly.
b/c $p \in (0, 1)$. too small range.
instead, we reparametrize logit(p)
↓
expand the range to $(-\infty, \infty)$

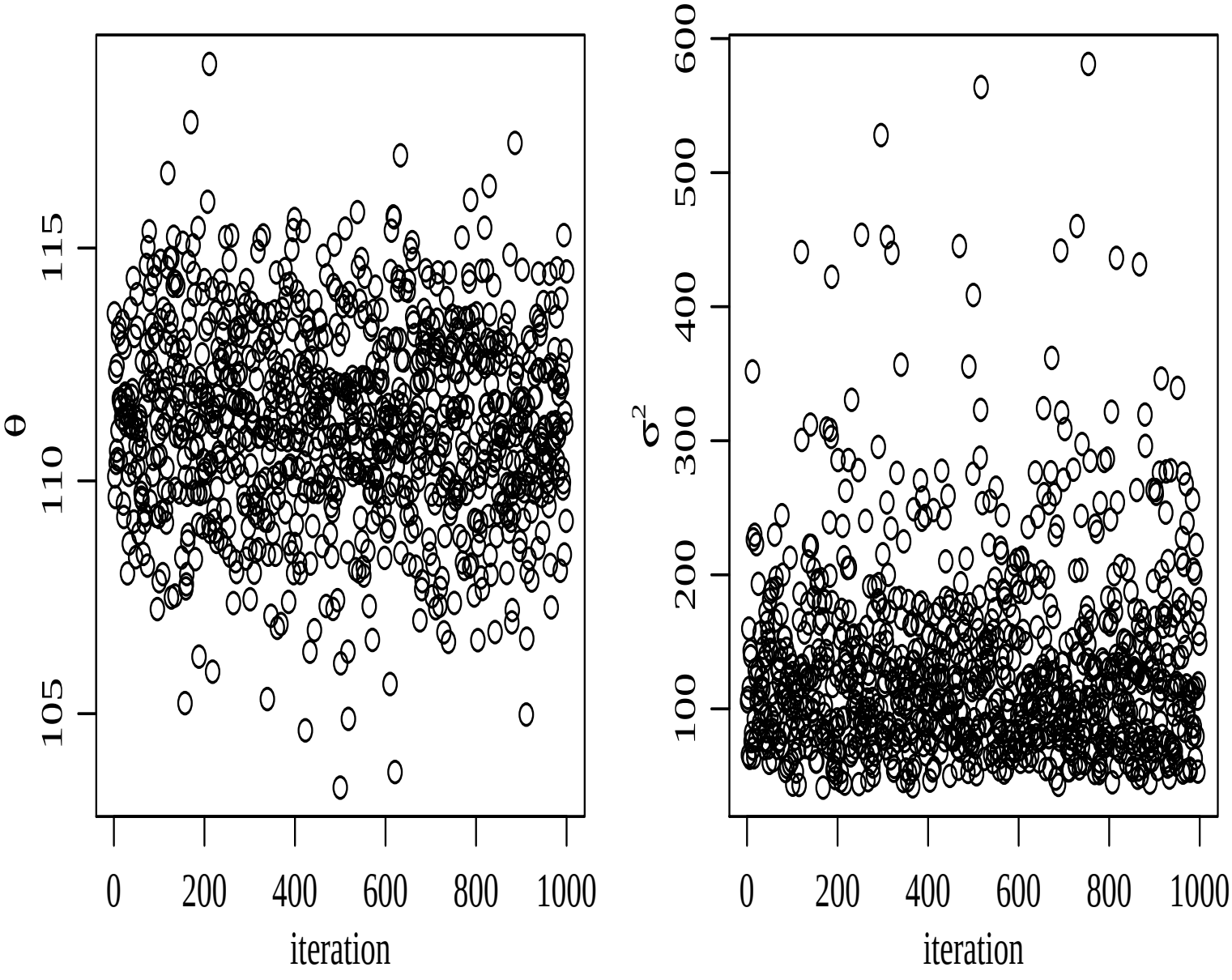
If the autocorrelation is high and the effectiveSize is low, what might you do to improve the computational efficiency of your Gibbs sampling algorithm? Focus on thinning.

① Thinning, we do not use all the draws. ~~use~~ e.g. use 1 out of every 100 draws. (we create lag intentionally)
when lag ↑, acf ↓. more independent draws.

② Burn-in period. cut off some draws at a certain point.
(very similar to thinning).
~~idea:~~ If the starting value is bad, burn-in is good.
~~Because~~ want to discard the first few thousands/hundreds of inaccurate draws.

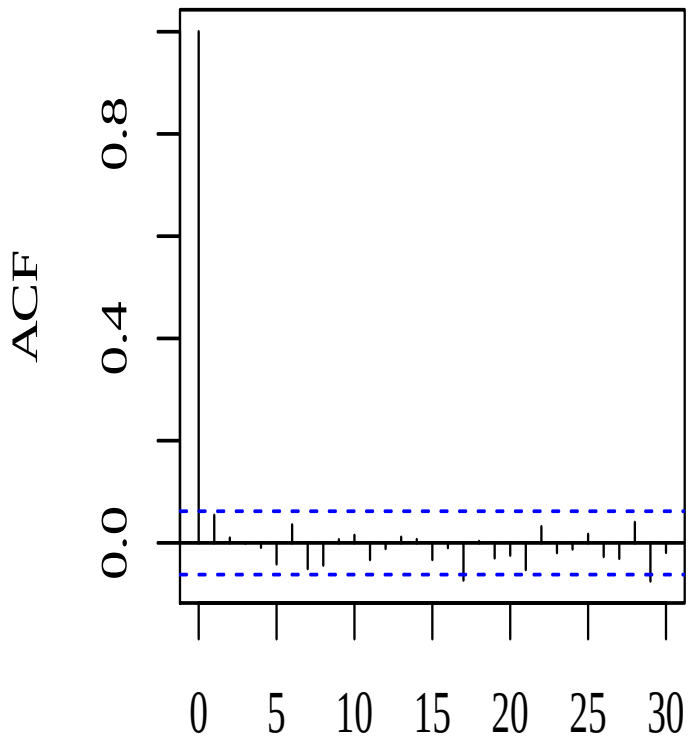
③ Parameter expansion/Reparametrization. (not gonna expand this topic)

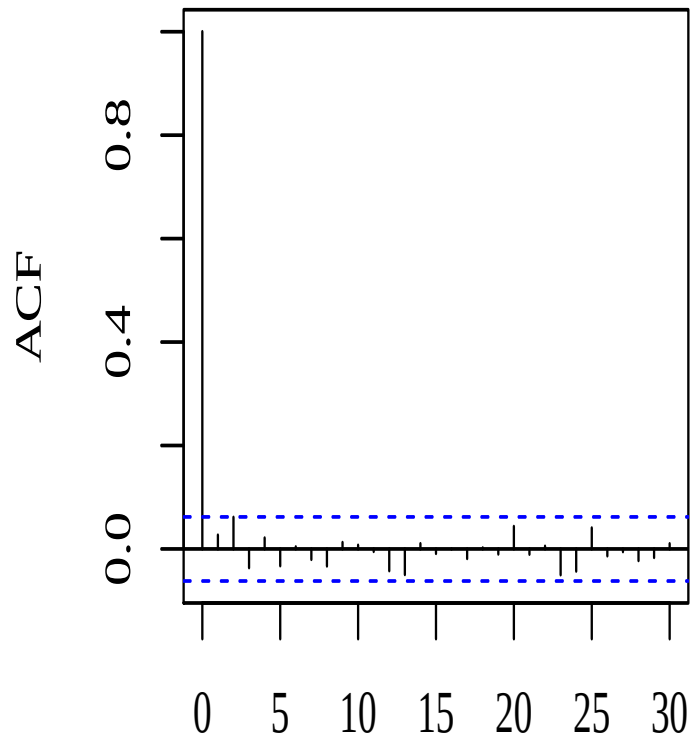
MCMC diagnostics - Cars speed Gibbs sampler



MCMC diagnostics - Cars speed Gibbs sampler

As M required large, beg selected good, then we have to increase S (size of sample draws)





MCMC diagnostics - Cars speed Gibbs sampler

```
> effectiveSize( PHI[,1] )
    var1
896.8363
> effectiveSize(1/PHI[,2] )
    var1
838.1677
```