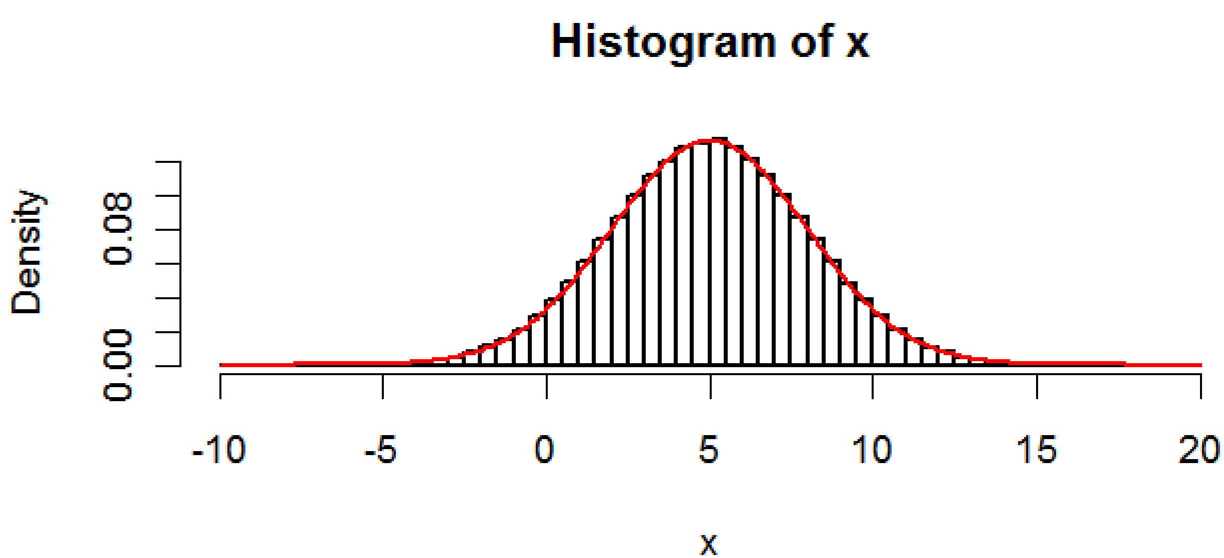
MCMC:
Metropolis Algorithm

Idea:

Random samples from the posterior

- Approximate PDF with the histogram
- Performs *Monte Carlo Integration*
- Allows all quantities of interest to be calculated from the sample (mean, quantiles, var, etc)



	TRUE	Sample
mean	5.000	5.000
median	5.000	5.004
var	9.000	9.006
Lower CI	-0.880	-0.881
Upper CI	10.880	10.872

Outline

- Different numerical techniques for sampling from the posterior
  - Rejection Sampling
  - Inverse Distribution Sampling
  - **Markov Chain-Monte Carlo (MCMC)**
    - **Metropolis**
    - **Metropolis-Hastings**
    - Gibbs sampling
- Sampling conditionals vs full model
- Flexibility to specify complex models

Markov Chain Monte Carlo

1) Start from some initial parameter value
2) Evaluate the unnormalized posterior
3) Propose a new parameter value
4) Evaluate the new unnormalized posterior
5) Decide whether or not to accept the new value
6) Repeat 3-5

The idea of MCMC

- Can we find a transition rule, p, such that the stationary distribution
  - Exists
  - Equals the posterior (target distribution)

Metropolis Algorithm

1) Start from some initial parameter value θ^c

2) Evaluate the unnormalized posterior $p(\theta^c|X)$

3) Propose a new parameter value $\theta^/$
   Random draw from a "jump" distribution centered on the current parameter value

4) Evaluate new unnormalized posterior $p(\theta^/|X)$

5) Decide whether or not to accept the new value
   Accept new value with probability

$$a = p(\theta^/) / p(\theta^c)$$

6) Repeat 3-5

$$p(\theta^c \rightarrow \theta^*) = P(\theta^*) / P(\theta^c)$$

Accept new value with probability

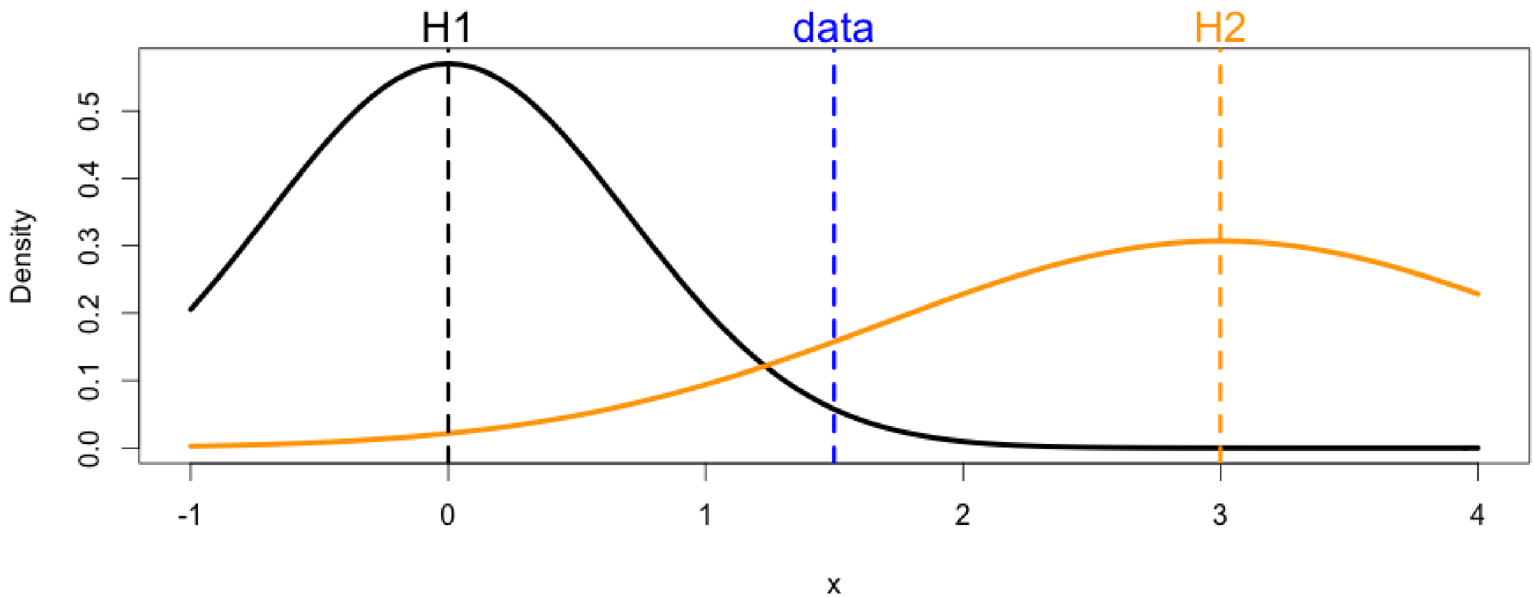
$$a = P(\theta^*) / P(\theta^c)$$

means

$P(\theta^*) > P(\theta^c)$:	$a > 1$	accept always
$P(\theta^*) < P(\theta^c)$:	$0 < a < 1$	accept sometimes

```
if
  runif(1,0,1) < a
  accept
else
  reject
```

& STAY AT CURRENT VALUE



$$P(\mu_1|X)=\frac{P(X|\mu_1)P(\mu_1)}{\int P(X|\mu)P(\mu)} \qquad P(\mu_2|X)=\frac{P(X|\mu_2)P(\mu_2)}{\int P(X|\mu)P(\mu)}$$

$$\frac{P(\mu_1|X)}{P(\mu_2|X)}=\frac{\dfrac{P(X|\mu_1)P(\mu_1)}{\int P(X|\mu)P(\mu)}}{\dfrac{P(X|\mu_2)P(\mu_2)}{\int P(X|\mu)P(\mu)}}=\frac{P(X|\mu_1)P(\mu_1)}{P(X|\mu_2)P(\mu_2)}$$

Metropolis Algorithm

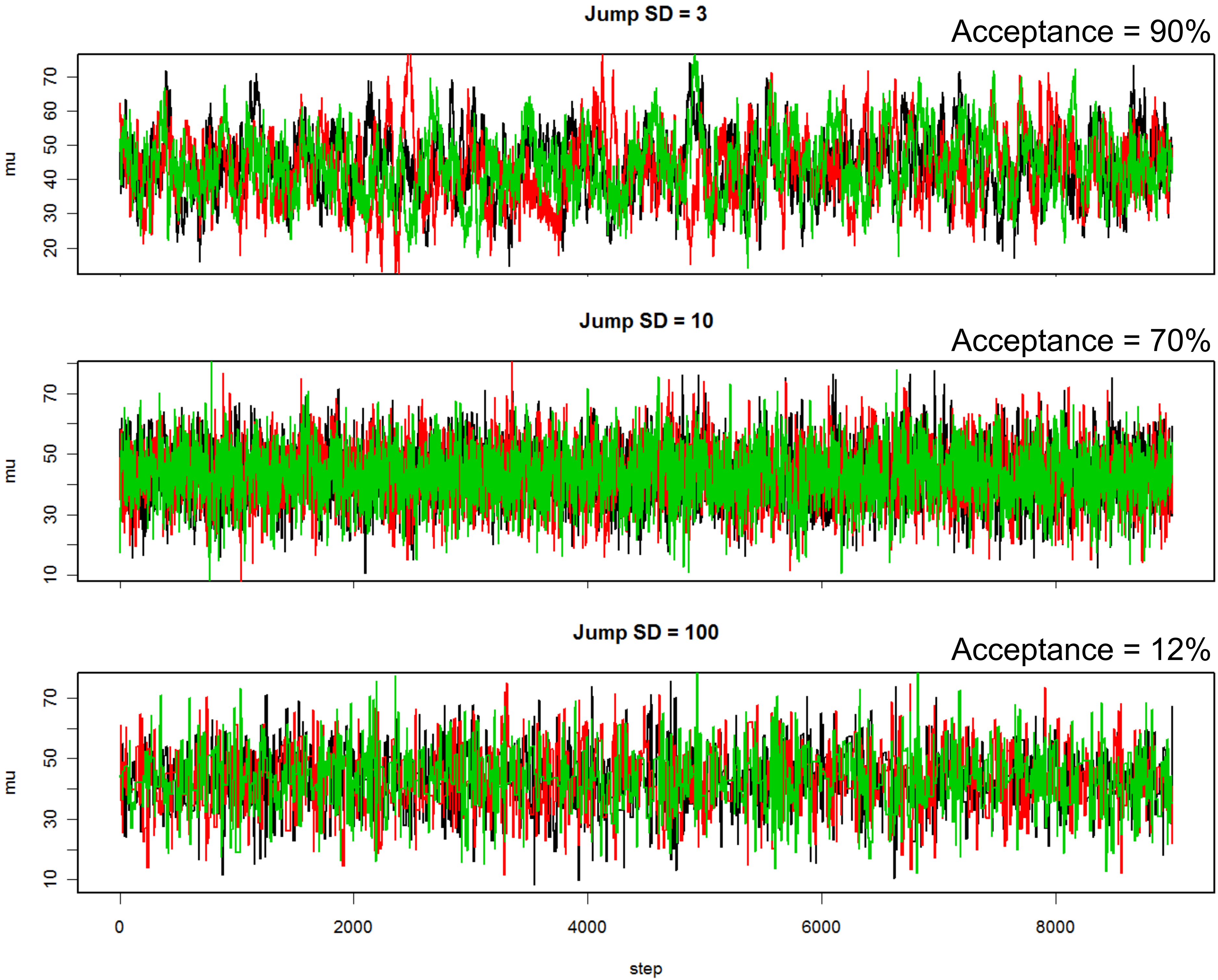
- Most popular form of MCMC
- Can be applied to most any problem
- Implementation requires little additional thought beyond writing the model
- Evaluation/Tuning does require the most skill & experience
- Indirect Method
  - Requires a second distribution to propose steps

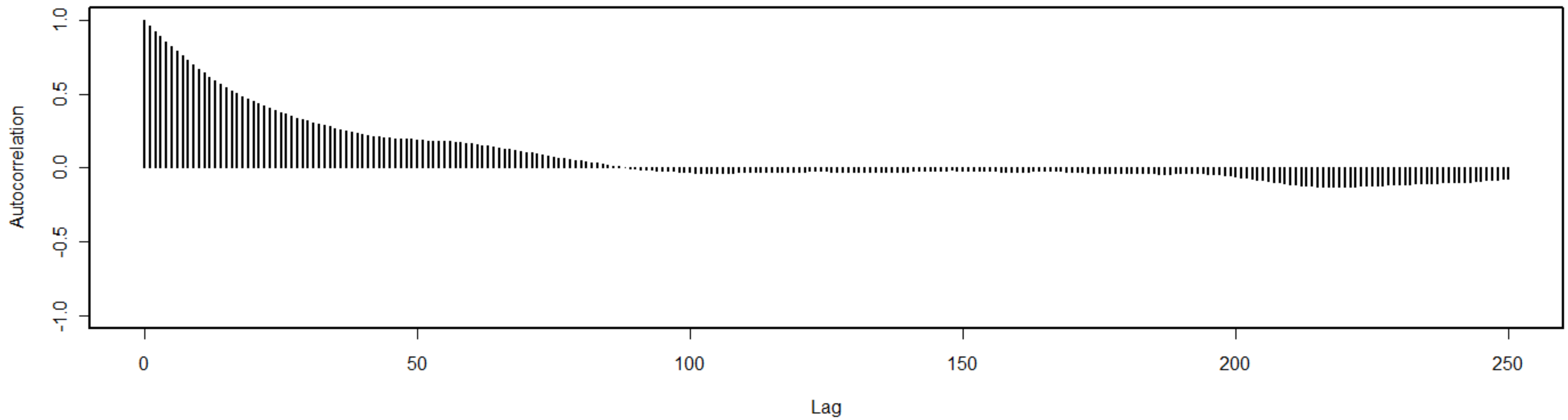
Jump distribution

- For Metropolis, Jump distribution J must be SYMMETRIC
  $J(\theta^{/} \mid \theta^{c}) = J(\theta^{c} \mid \theta^{/})$
- Most common Jump distribution is the Normal
  $J(\theta^{/} \mid \theta^{c}) = N(\theta^{/} \mid \theta^{c}, \upsilon)$
- User must set the variance of the jump
  - Trial-and-error
  - Tune to get acceptance rate 30-70%
  - Low acceptance = decrease variance (smaller step)
  - Hi acceptance = increase variance (bigger step)

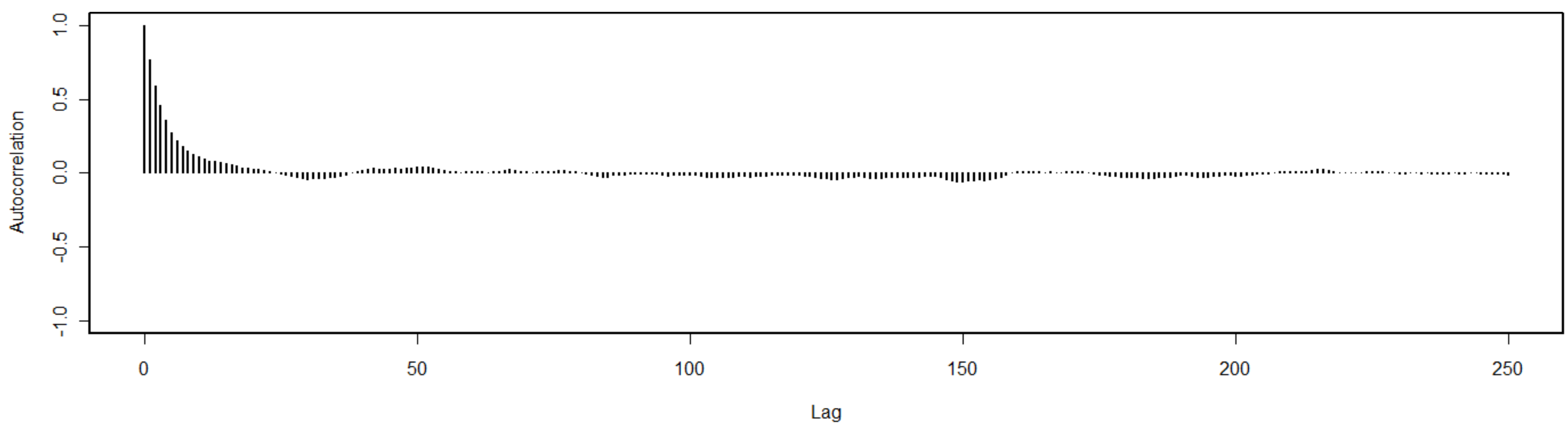
Example

- Normal with known variance, unknown mean
  - Prior: N(53,10000)
  - Data: y = 43
  - Known variance: 100
  - Initial conditions, 3 chains starting at -100, 0, 100
  - Jump distribution = Normal
  - Jump variance = 3,10,30

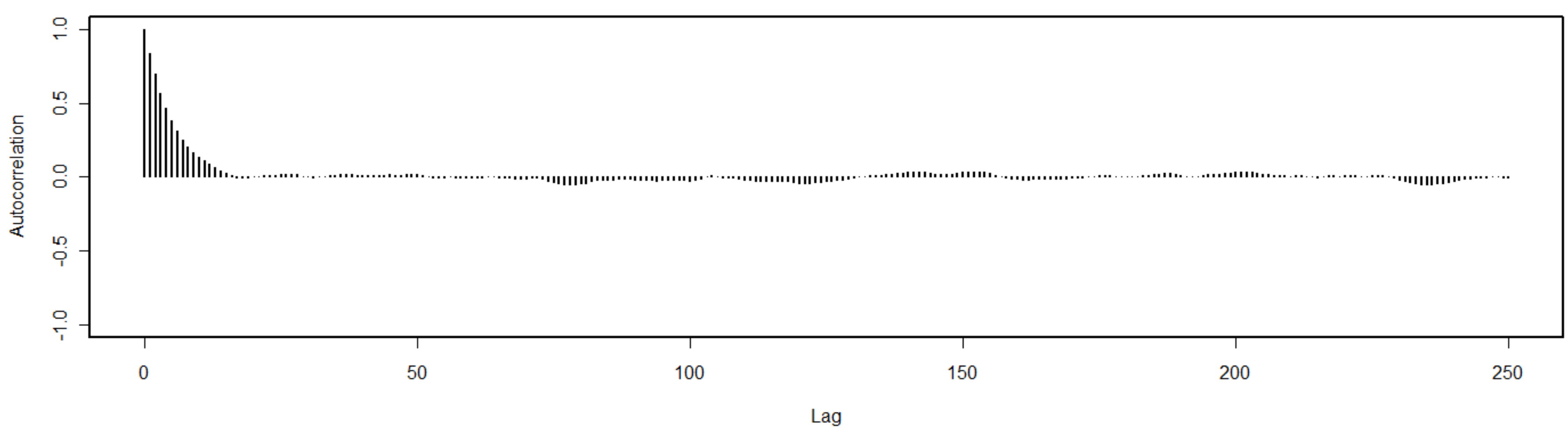

Jump SD = 3
Acceptance = 90%
Jump SD = 10
Acceptance = 70%
Jump SD = 100
Acceptance = 12%
mu
step
0
2000
4000
6000
8000



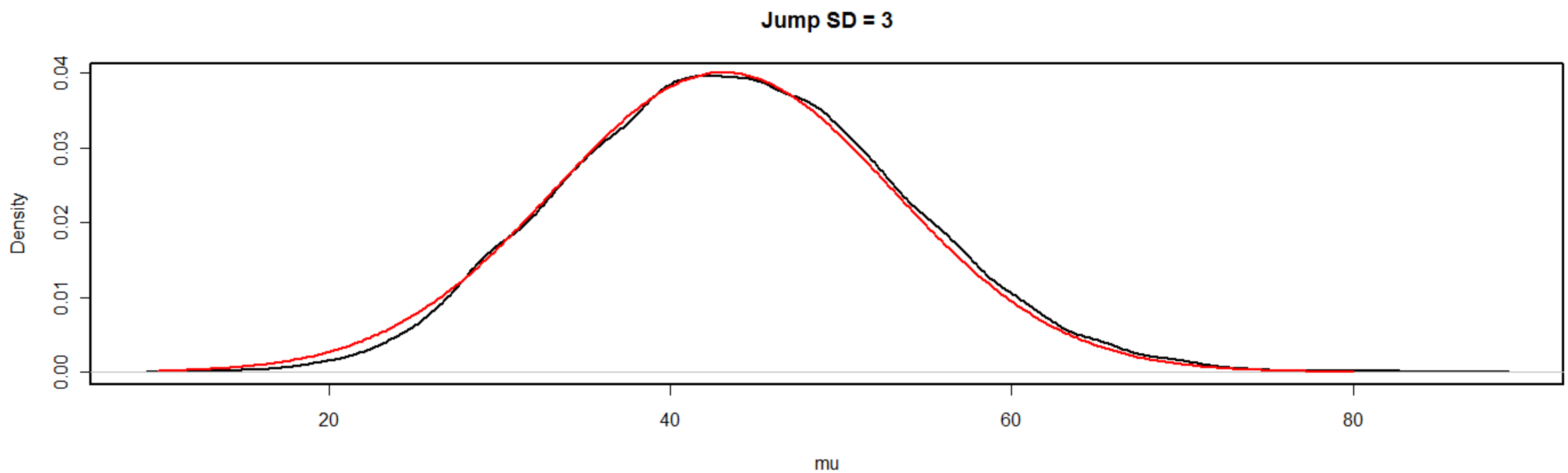
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
50
100
150
200
250
Lag

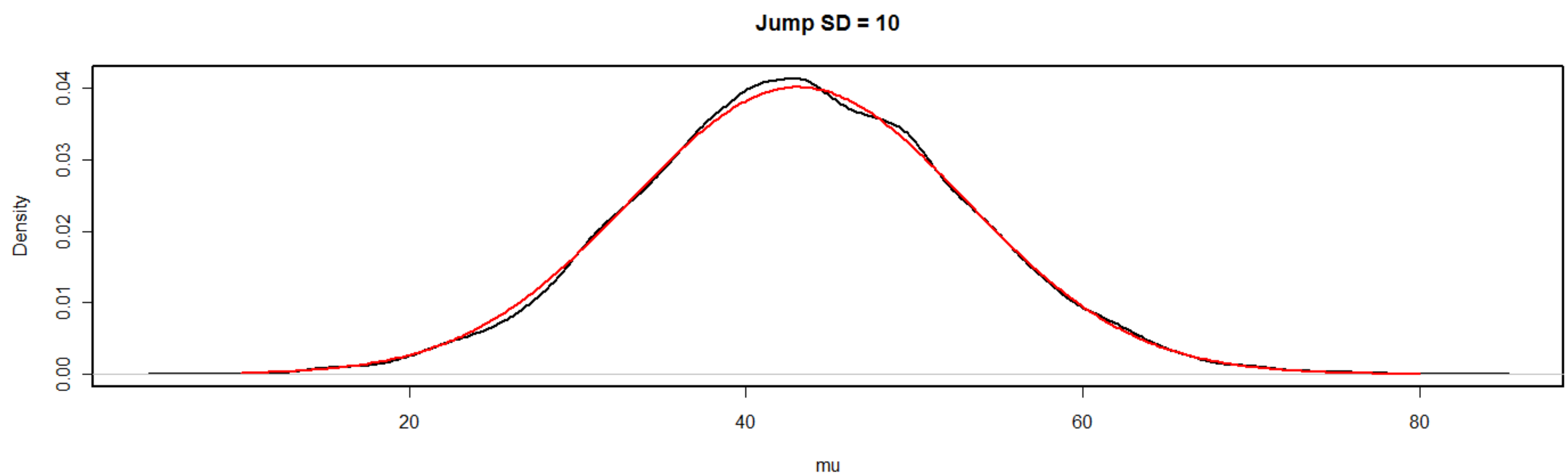


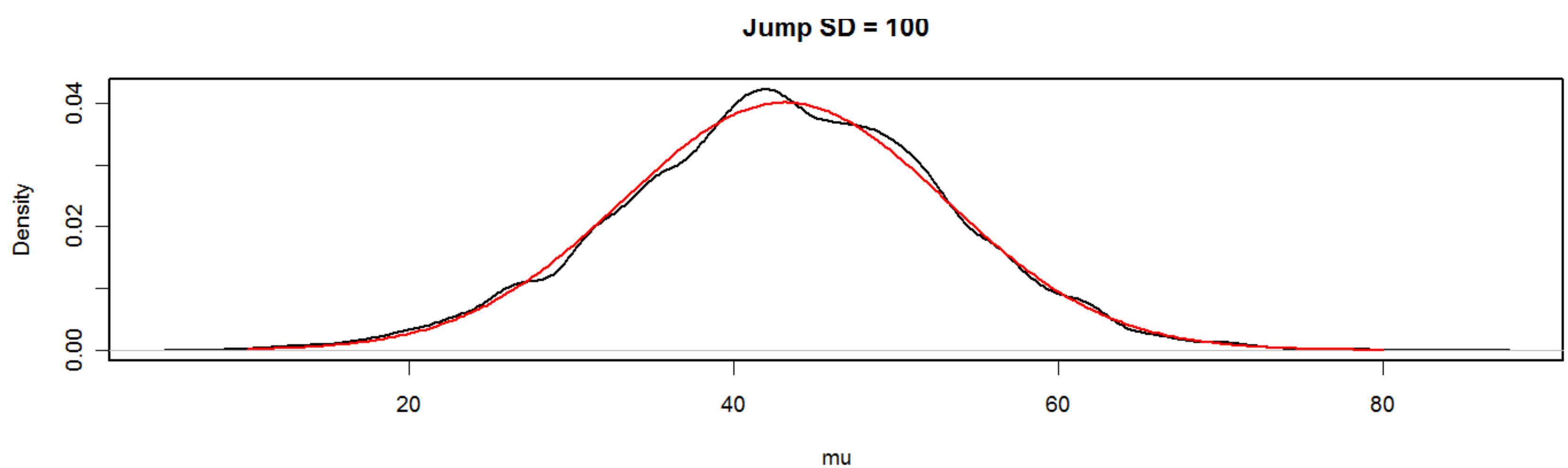
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
50
100
150
200
250
Lag

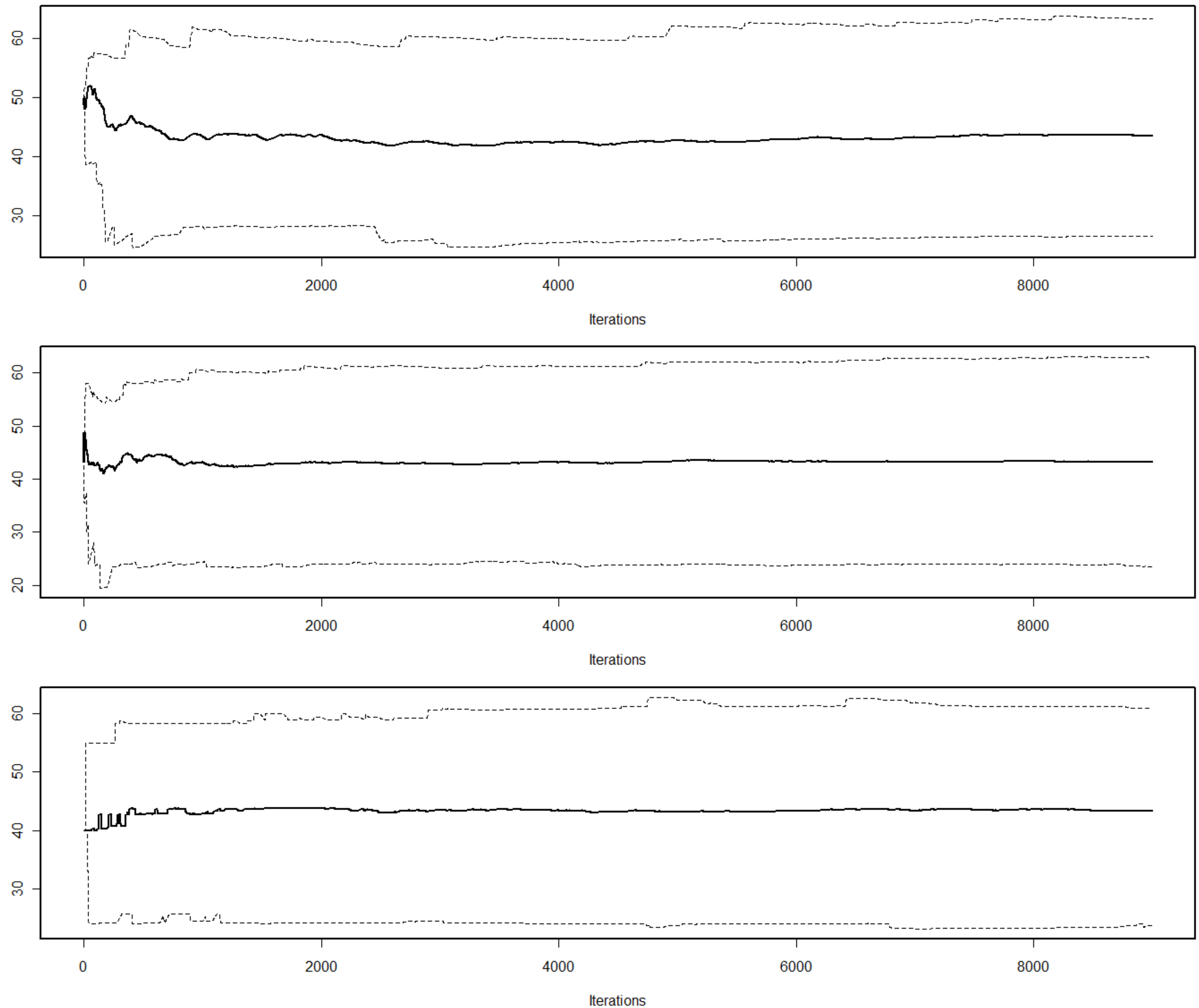


Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
50
100
150
200
250
Lag

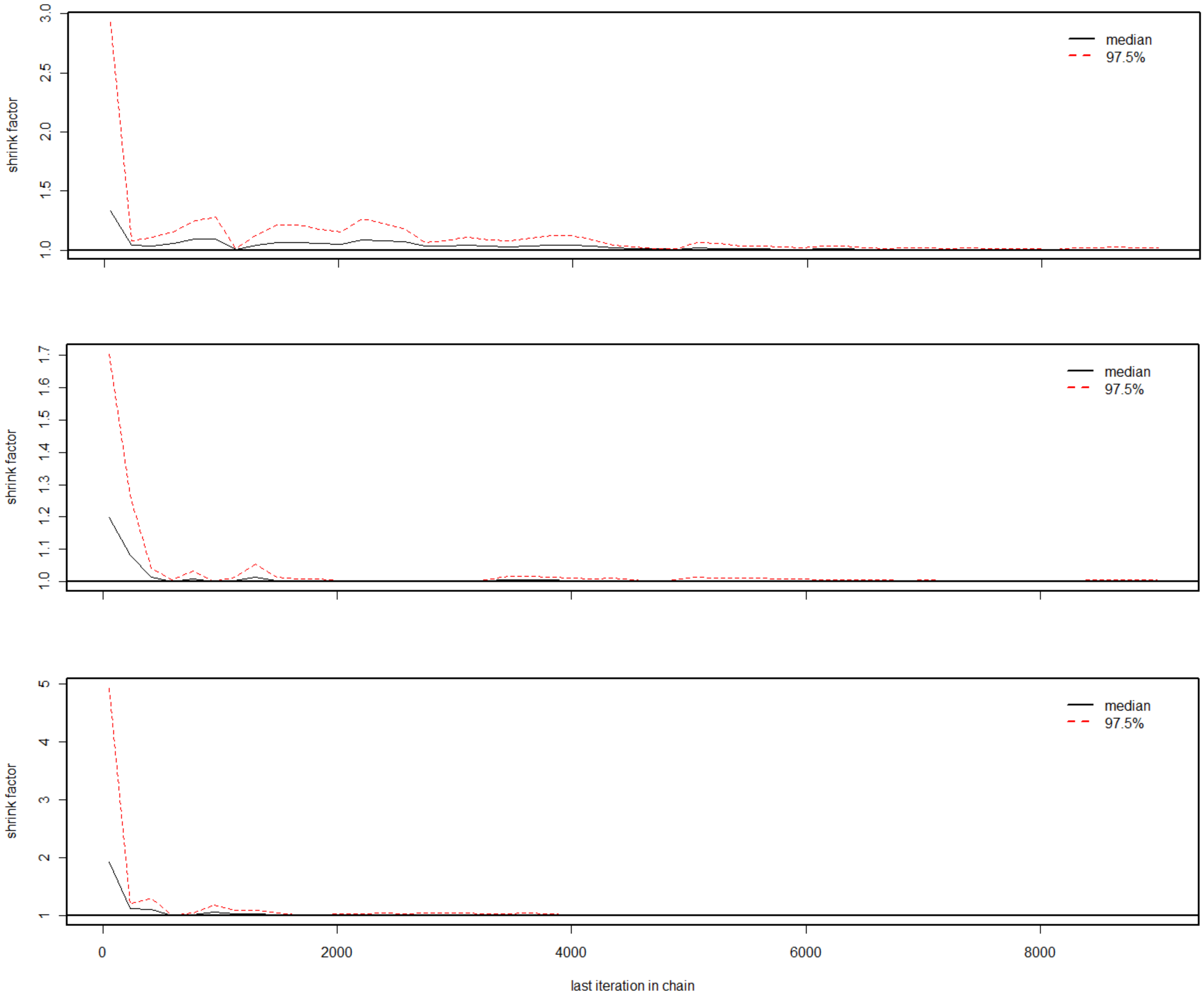

Jump SD = 3
Density
0.00
0.01
0.02
0.03
0.04
20
40
60
80
mu


Jump SD = 10
Density
0.00
0.01
0.02
0.03
0.04
20
40
60
80
mu


Jump SD = 100
Density
0.00
0.02
0.04
20
40
60
80
mu



Iterations
Iterations
Iterations


median
97.5%
shrink factor
3.0
2.5
2.0
1.5
1.0
1.7
1.6
1.5
1.4
1.3
1.2
1.1
5
4
3
2
1
0
2000
4000
6000
8000
last iteration in chain

Multivariate example

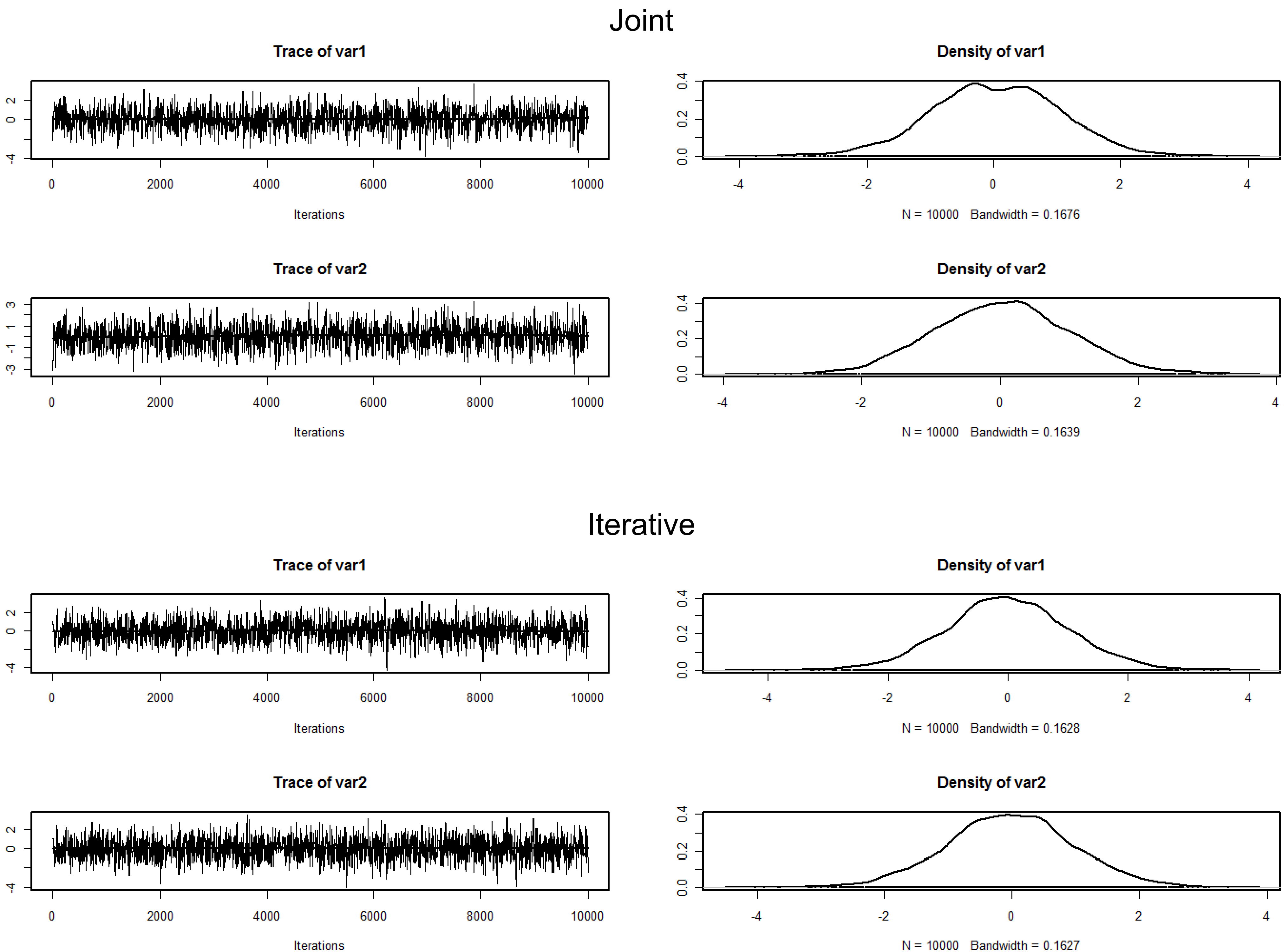
- Bivariate Normal $N_2\left(\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & 0.5\\0.5 & 1\end{bmatrix}\right)$
- Option 1: Draw from joint distribution

$$J = N_2\left(\begin{bmatrix}\theta_1^*\\\theta_2^*\end{bmatrix}\middle|\begin{bmatrix}\theta_1^c\\\theta_2^c\end{bmatrix}, V\right)$$

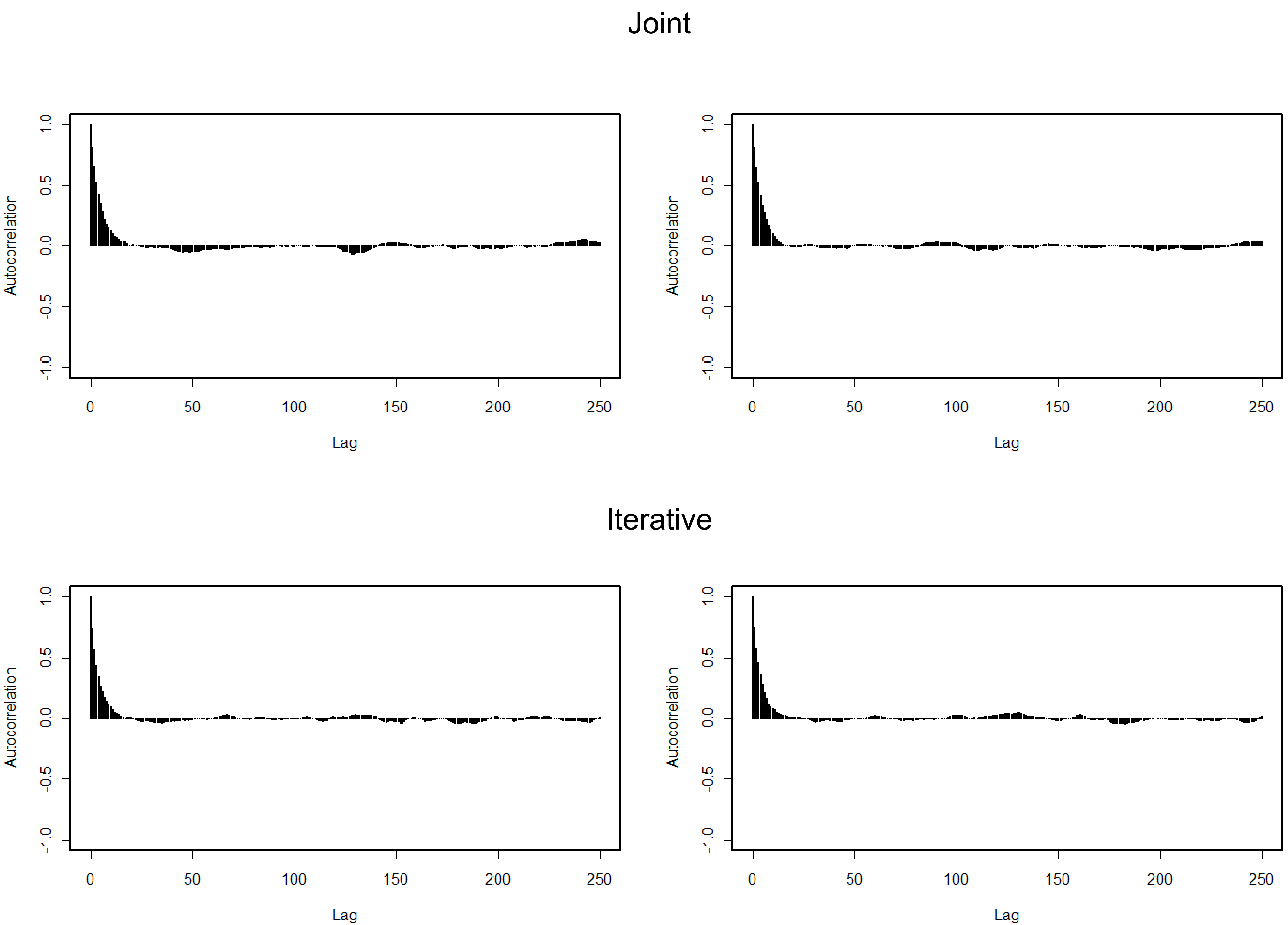
- Option 2: Draw from each parameter iteratively

$$J_1 = N(\theta_1^* \mid \theta_1^c, V_1)$$

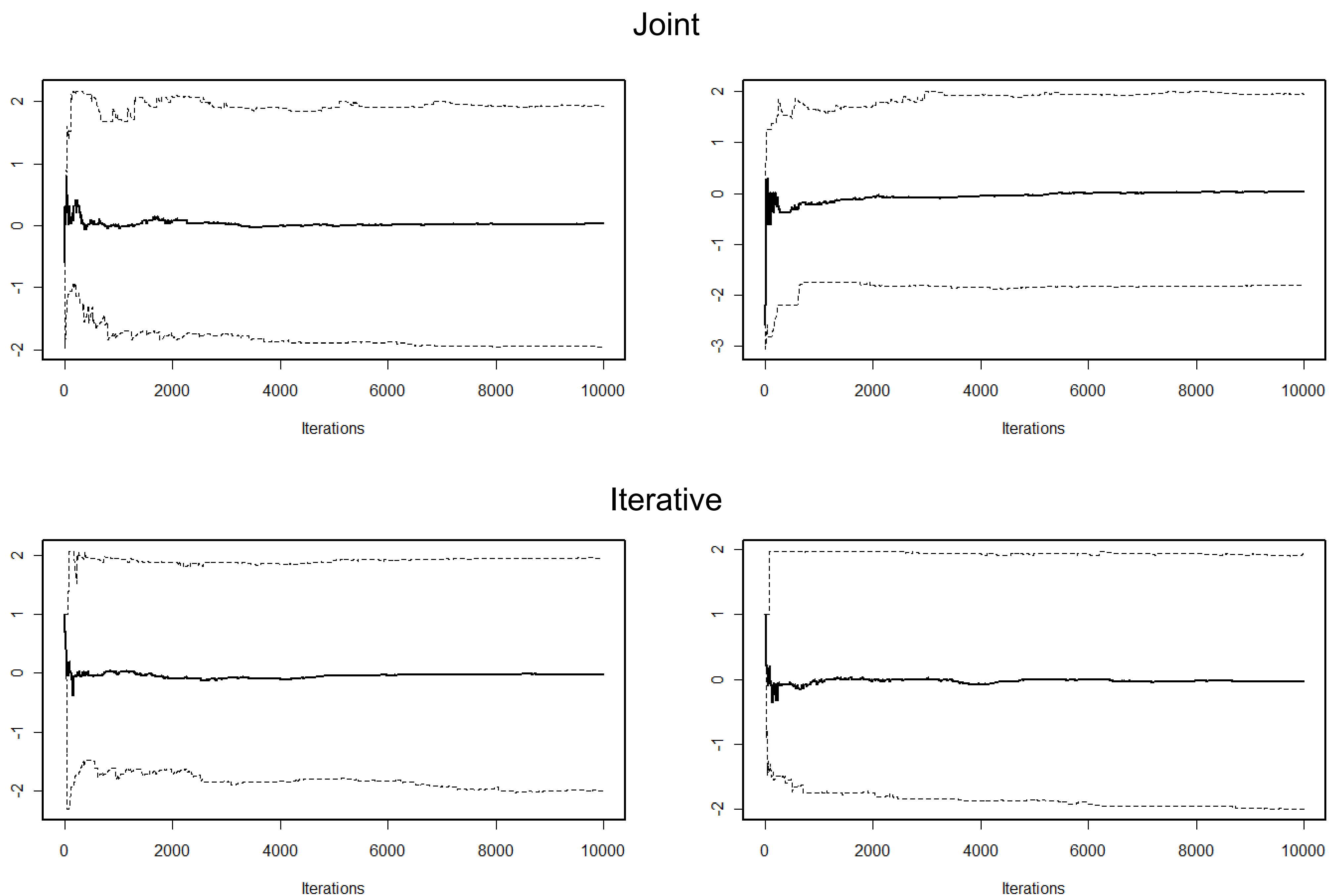
$$J_2 = N(\theta_2^* \mid \theta_2^c, V_2)$$



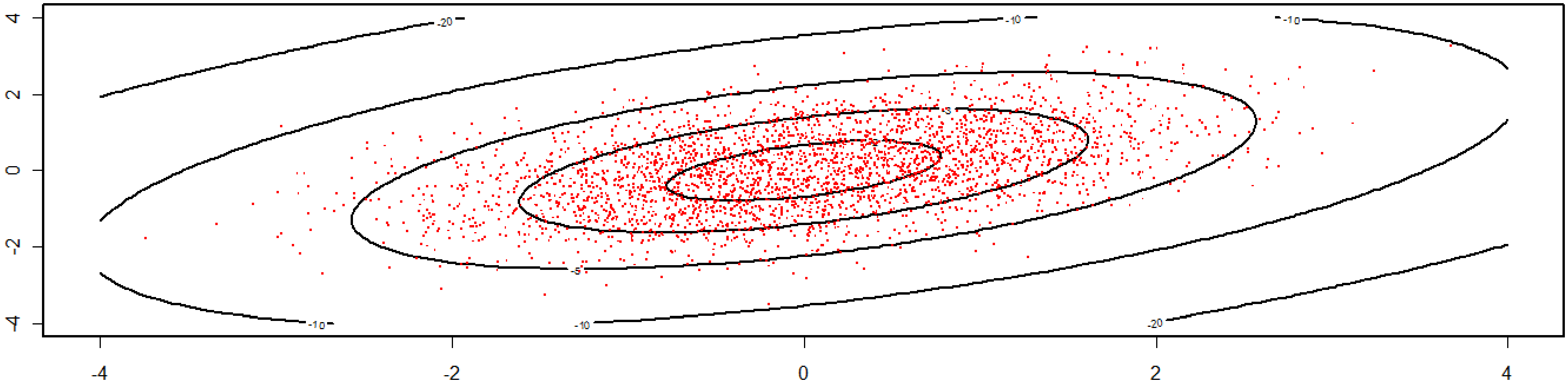
Joint
Trace of var1
Iterations
Density of var1
N = 10000 Bandwidth = 0.1676
Trace of var2
Iterations
Density of var2
N = 10000 Bandwidth = 0.1639
Iterative
Trace of var1
Iterations
Density of var1
N = 10000 Bandwidth = 0.1628
Trace of var2
Iterations
Density of var2
N = 10000 Bandwidth = 0.1627



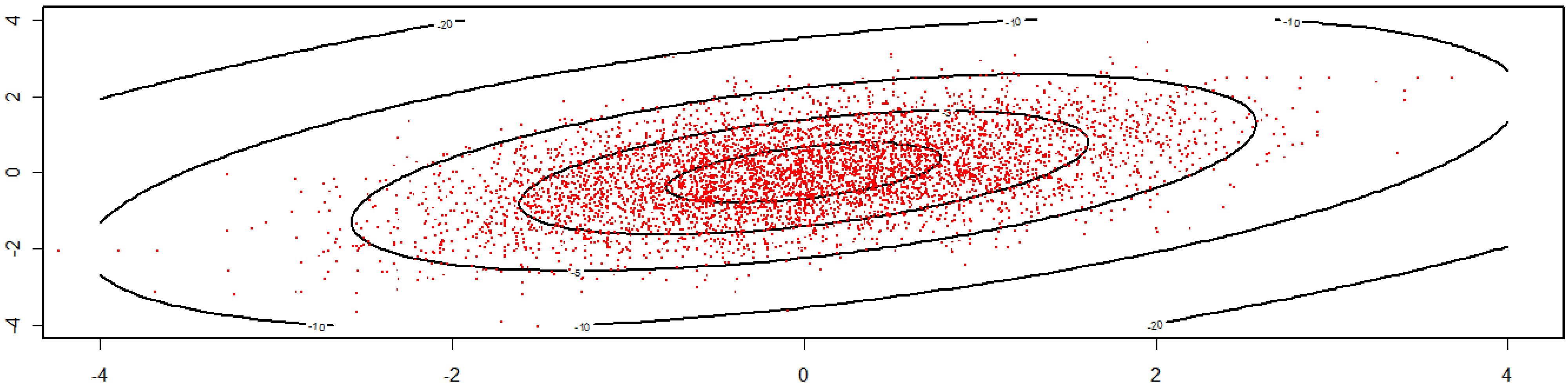
Joint
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
50
100
150
200
250
Lag
Iterative


Joint
Iterative
Iterations
Iterations
Iterations
Iterations
0
2000
4000
6000
8000
10000
-3
-2
-1
0
1
2

Joint



Iterative

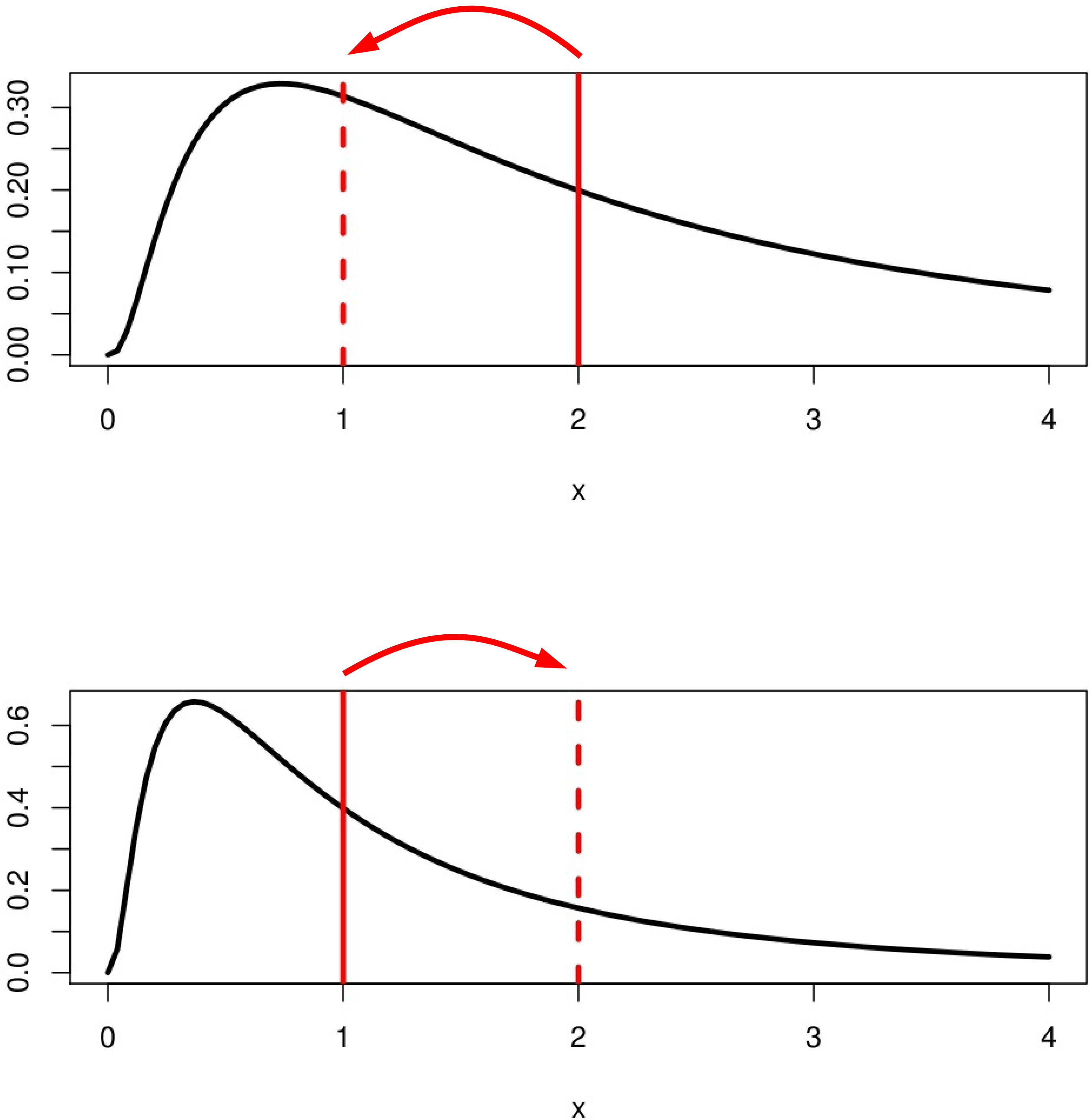


Metropolis-Hastings

- Generalization of Metropolis
- Allows for asymmetric Jump distribution
- Acceptance criteria

$$a = \frac{p(\theta^*)/J(\theta^*|\theta^c)}{p(\theta^c)/J(\theta^c|\theta^*)}$$

- Most commonly arise due to bounds on parameter values / non-normal Jump distributions


0.00
0.10
0.20
0.30
0
1
2
3
4
x
0.0
0.2
0.4
0.6
0
1
2
3
4
x