Markov Chain Monte Carlo Methods: Metropolis-Hastings Algorithm

Mihir Arjunwadkar

Centre for Modeling and Simulation
Savitribai Phule Pune University

We will assume that the theory of

discrete-state-space

Markov chains we sketched applies to

continuous-state-space

Markov chains as well.

Two archetypal problems

Given an arbitrary (multivariate) PDF f,

- How does one sample f?
- How does one estimate expected values with respect to f?

Markov chain Monte Carlo (MCMC) methods address both these problems.

Basic MCMC: One Metropolis-Hastings step

Input

1. The *target* or *desired* (multivariate) density $f(x)$
2. A friendly & easily sampleable *proposal* density $q(y|x)$
3. An initial / starting state x_1

Basic MCMC: One Metropolis-Hastings step

$$x_1 \xrightarrow{MH} x_2 \xrightarrow{MH} \cdots \xrightarrow{MH} x_i \xrightarrow{MH} x_{i+1} \xrightarrow{MH} \cdots$$

Algorithm to generate x_{i+1} from x_i

1. Generate (sample) a *proposal* (or *candidate*):
$$y \sim q(y|x_i)$$
2. Compute acceptance probability:
$$a(y|x_i) = \min\left\{1, \frac{f(y)}{f(x_i)}\frac{q(x_i|y)}{q(y|x_i)}\right\}$$
3. Accept or reject probabilistically:
$$x_{i+1} = \begin{cases} y & \text{with probability } a(y|x_i) \\ x_i & \text{with probability } 1 - a(y|x_i) \end{cases}$$

"Accepting with probability a" ::
Generate $u \sim \text{Uniform}(0,1)$. If $u \leq a$, then accept, else reject.

Metropolis: Univariate example 1

- Desired PDF

$$f(x) = \frac{1}{\pi} \frac{1}{1 + x^2}$$

- Proposal PDF: $N(x, b^2)$; i.e.,

$$q(y|x) = \frac{1}{\sqrt{2\pi b^2}} \exp\left(-\frac{1}{2} \left(\frac{y - x}{b} \right)^2 \right).$$

b: the *step size/length* parameter; a tunable parameter.

This is an illustrative example (AoS2004): A transformation-based efficient sampler is available for this $f(x)$

Metropolis: Univariate example 1

- Because this proposal is *symmetric*; i.e.,

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

we have

$$a(y|x) = \min\left\{1, \frac{f(y)}{f(x)}\right\} = \min\left\{1, \frac{1+x^2}{1+y^2}\right\}.$$

- We will study the behaviour of the algorithm for
  - various starting points x_1, and
  - different values of b.

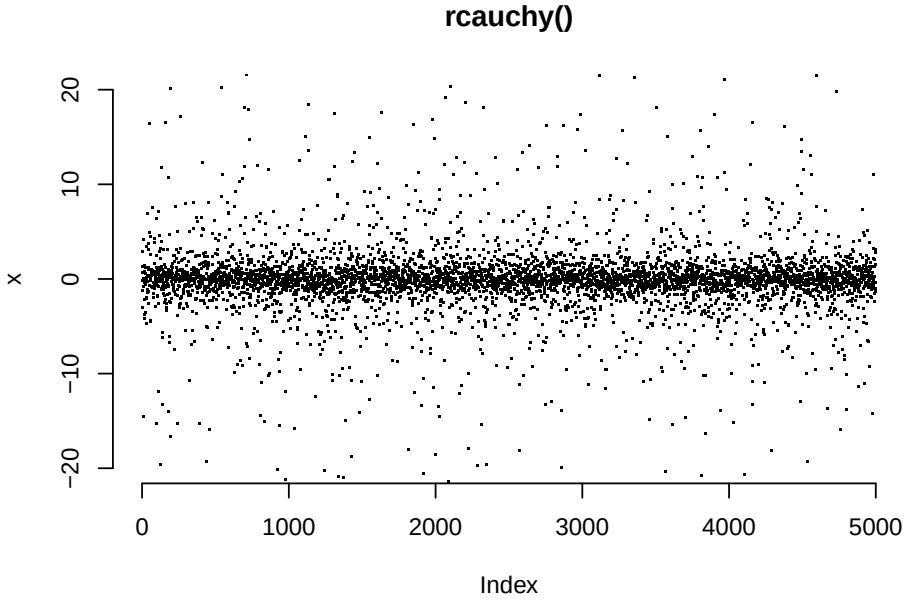
This is an illustrative example (AoS2004): A transformation-based efficient sampler is available for this $f(x)$

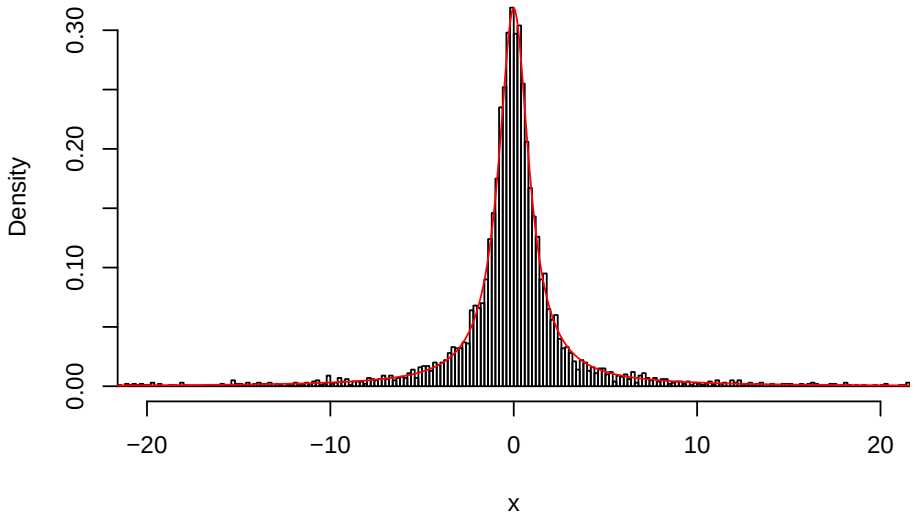
MCMC for Cauchy: Illustrative non-MCMC implementation in R

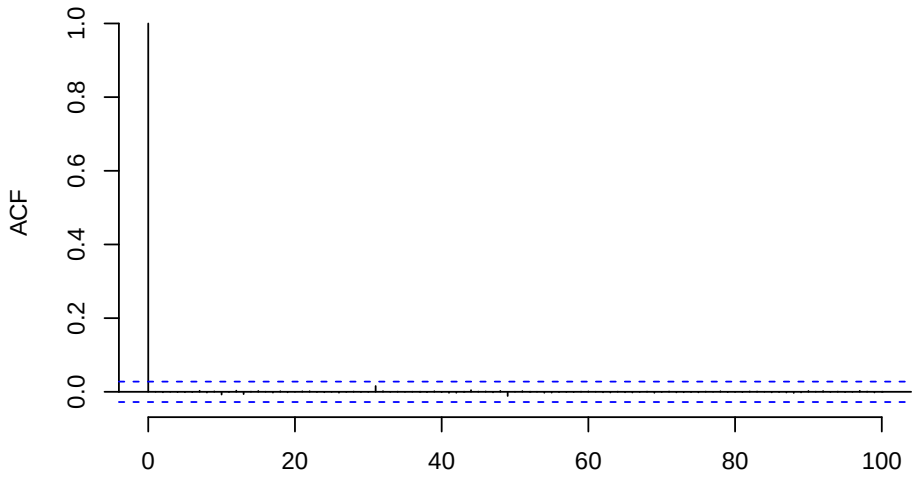
```
rcauchy.mcmc <- function( n, xo = 0, b = 1 )
 {
  proposal <- function( current, b ) { rnorm( 1, current, b ) }

  accept <- function( proposed, current )
              {
               runif( 1 ) <= ( 1 + current^2 ) / ( 1 + proposed^2 )
              }

  x <- rep( xo, n )

  for ( i in 2:n )
   {
    y    <- proposal( x[i-1], b )                    # generate proposal
    x[i] <- if ( accept( y, x[i-1] ) ) y else x[i-1] # accept or reject
   }

  x
 }
```

Metropolis: Univariate example 1

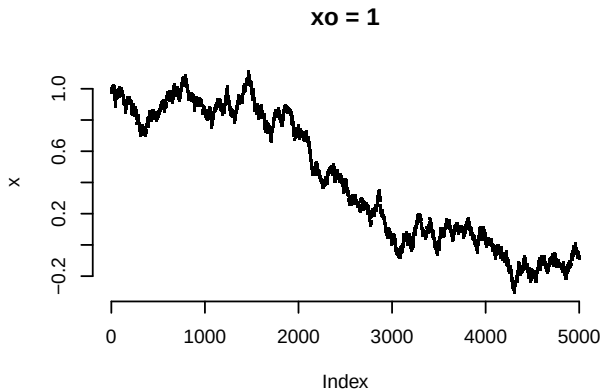
IID Cauchy random numbers, for comparison

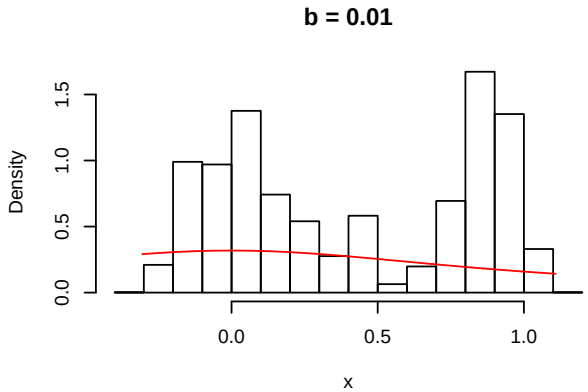


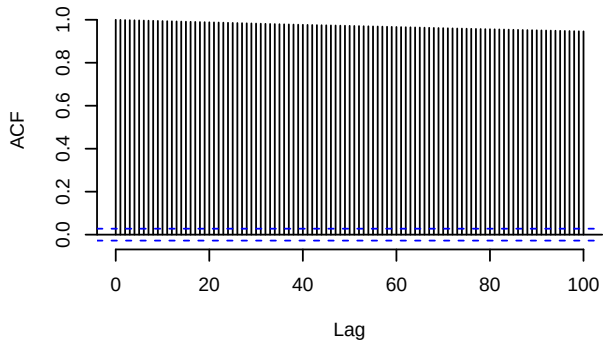


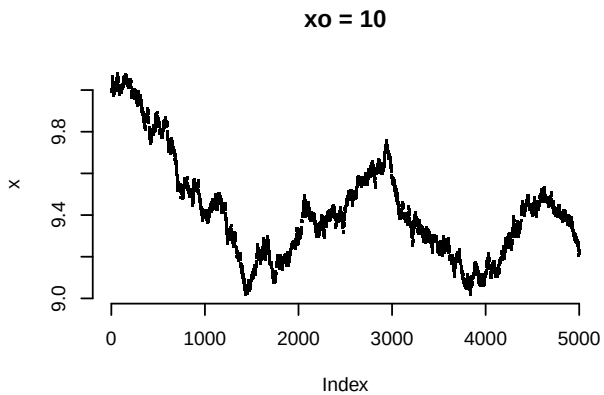


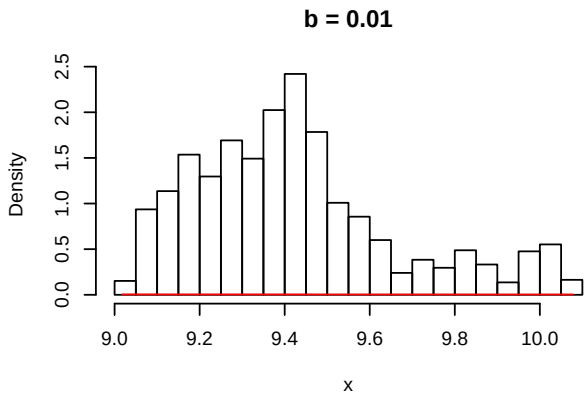
Metropolis: Univariate example 1

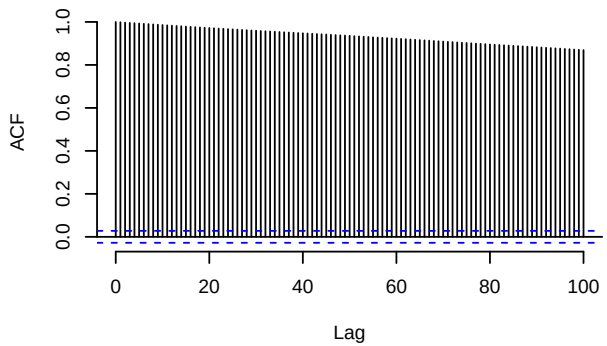


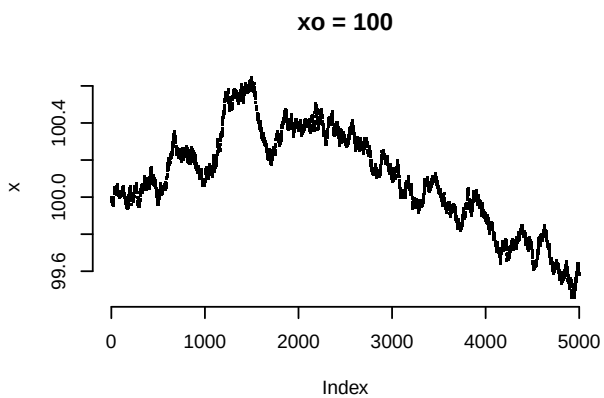


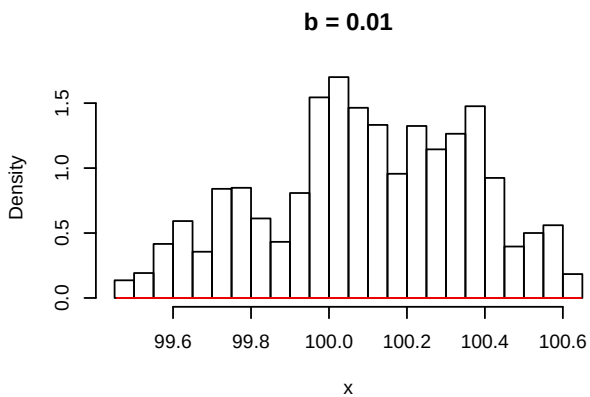


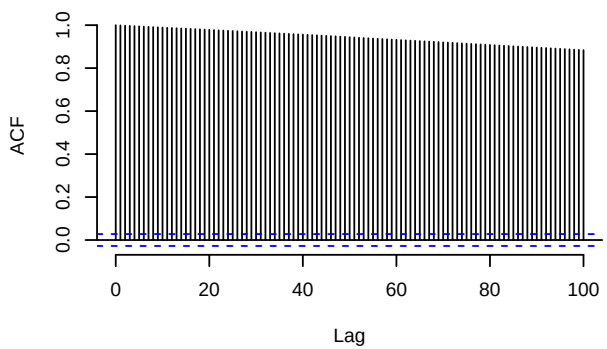




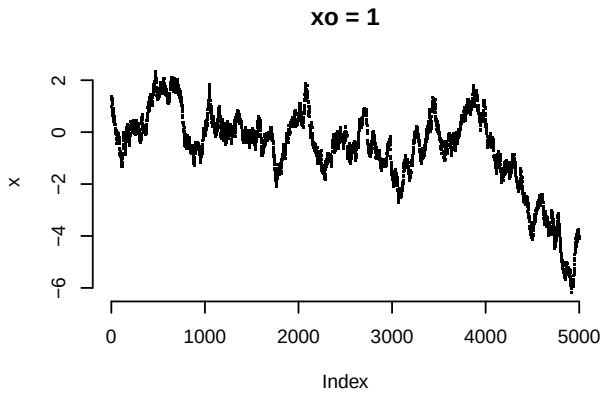


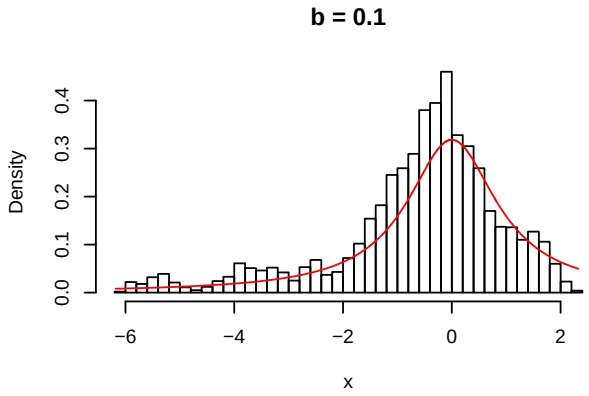


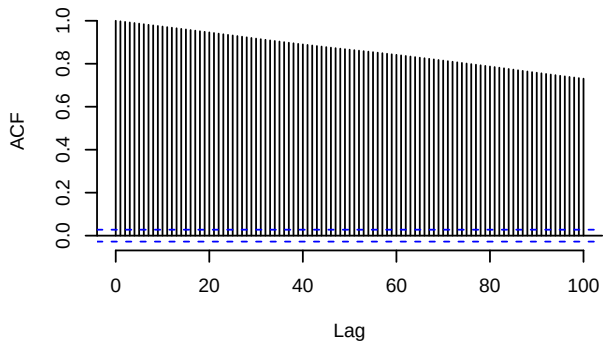


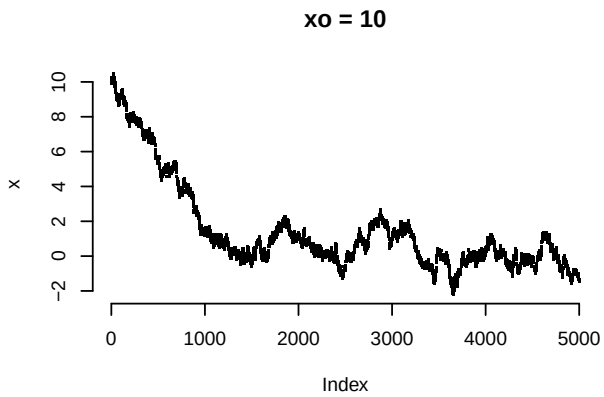


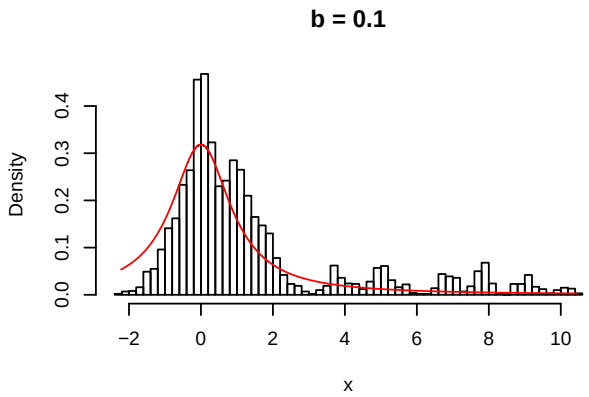
Metropolis: Univariate example 1

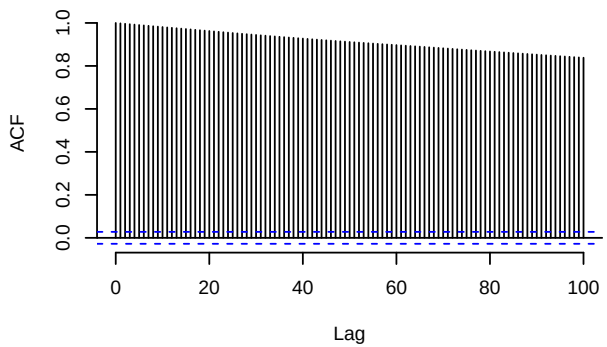


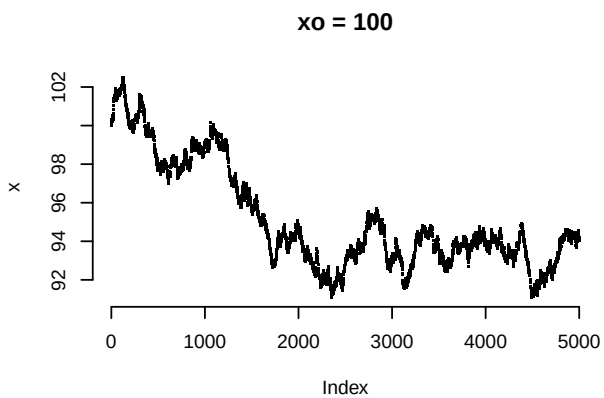


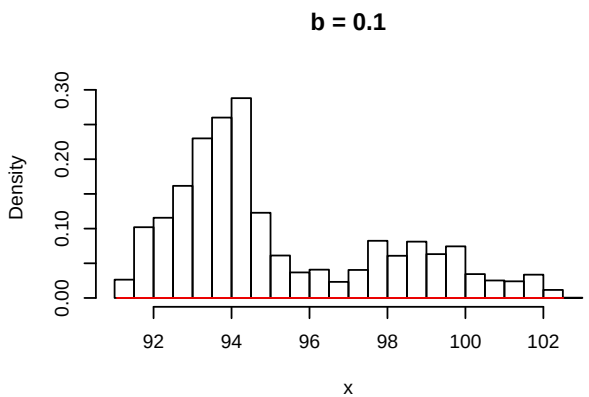


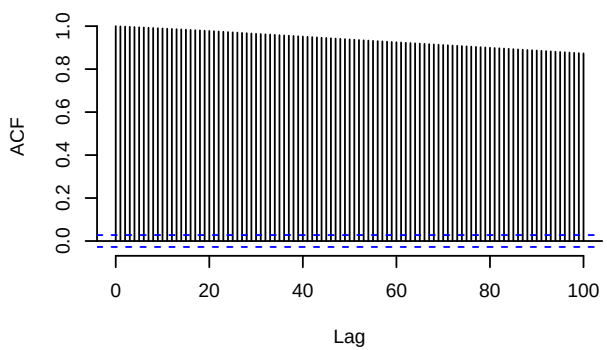




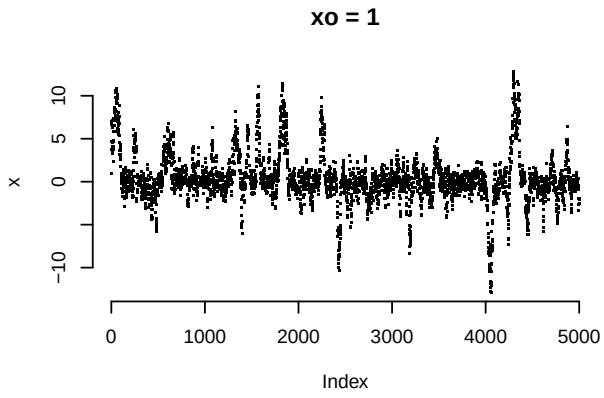


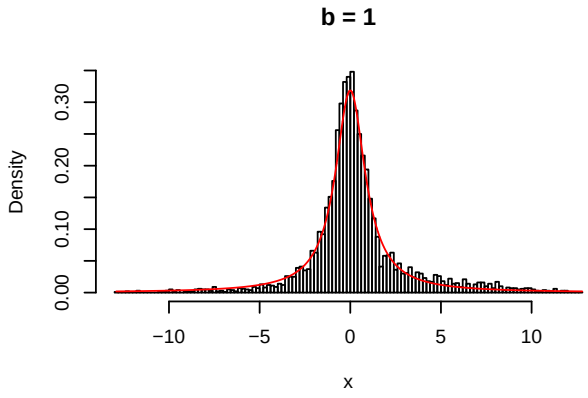


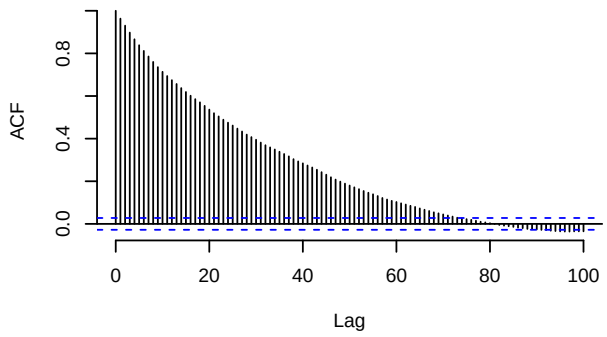


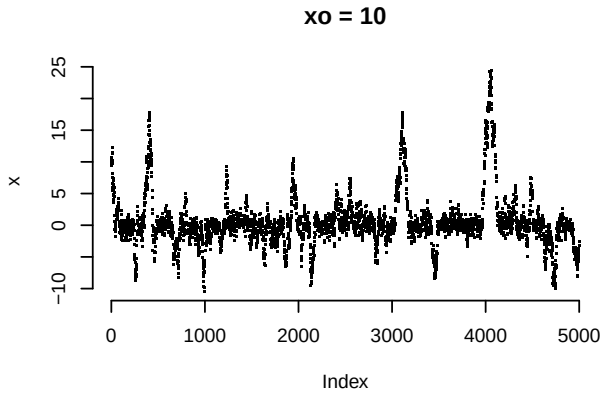


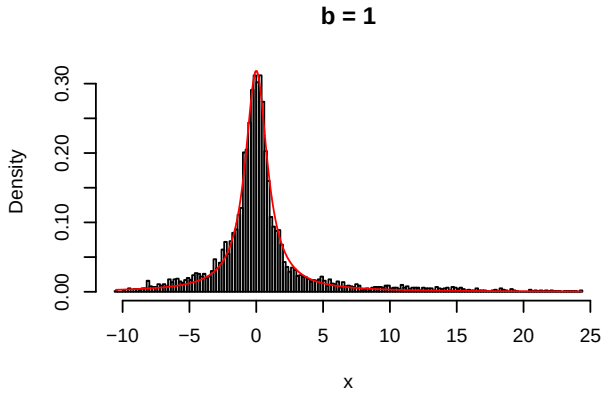
Metropolis: Univariate example 1

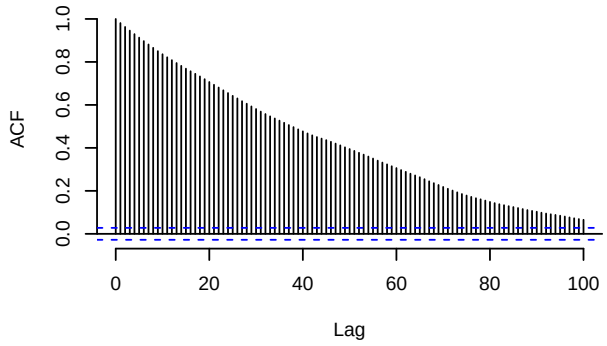


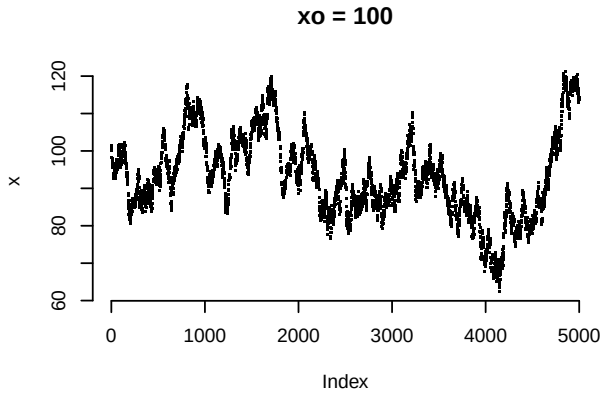


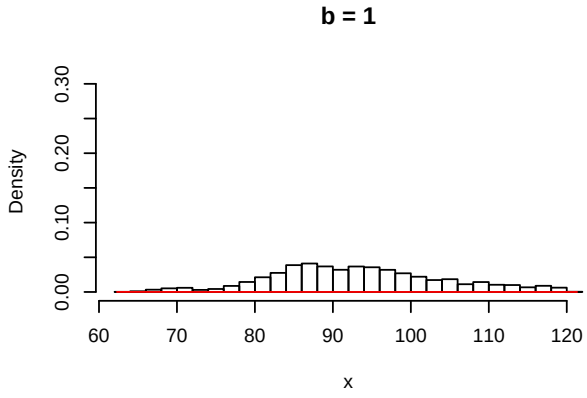


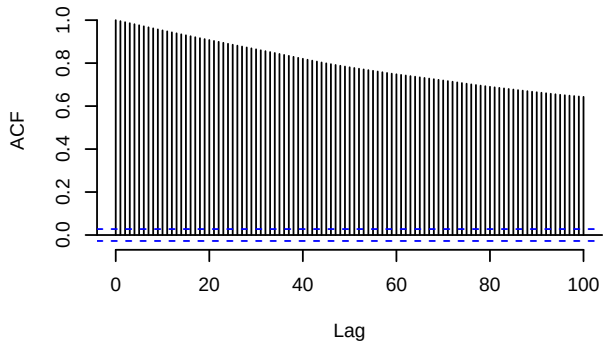




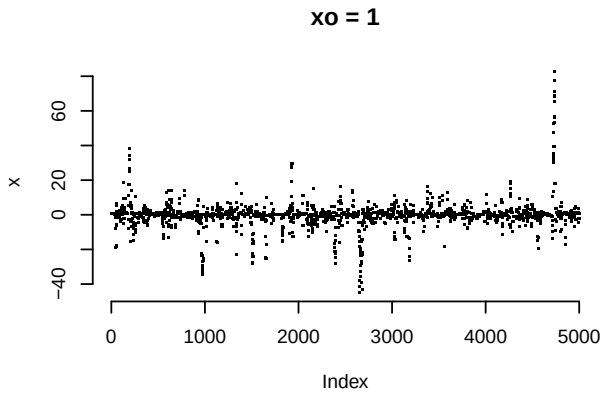


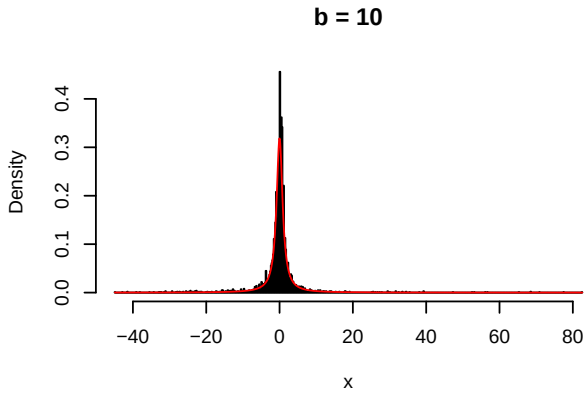


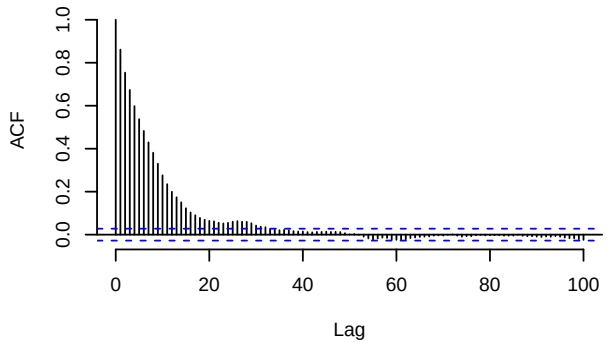


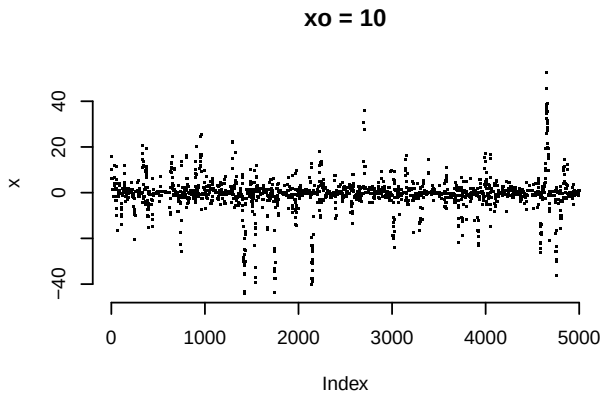


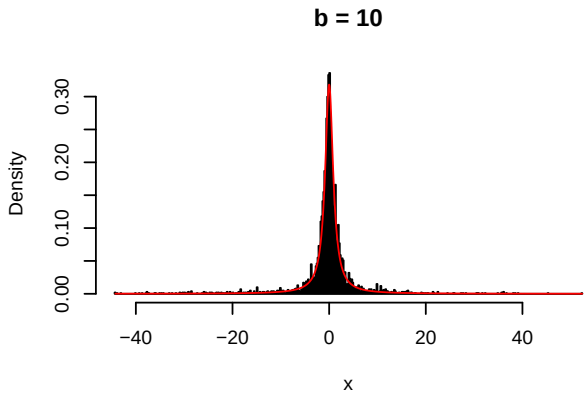
Metropolis: Univariate example 1

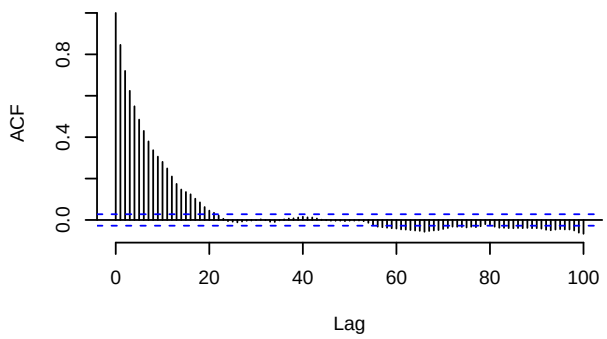


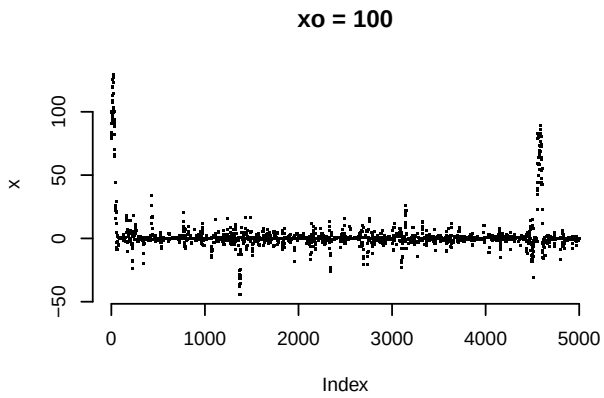


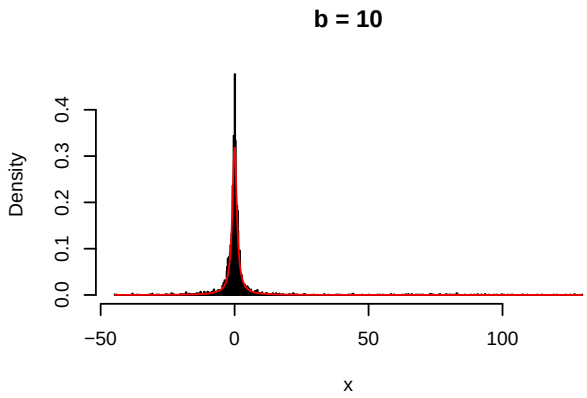


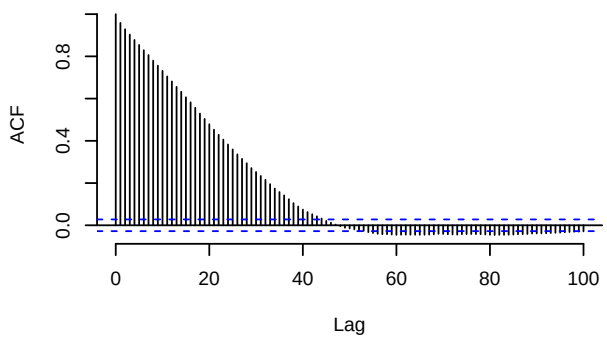




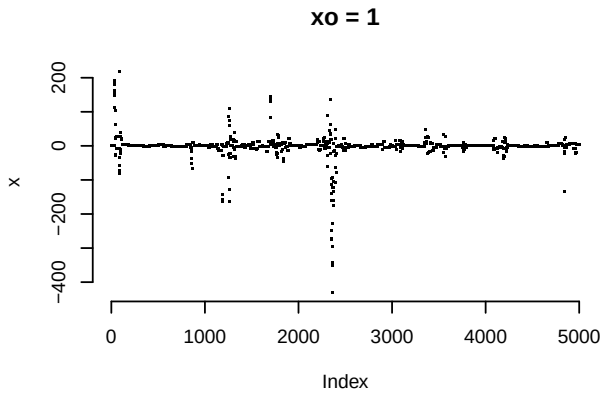


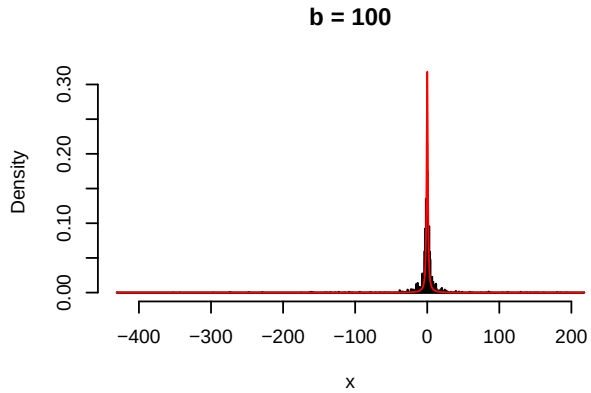


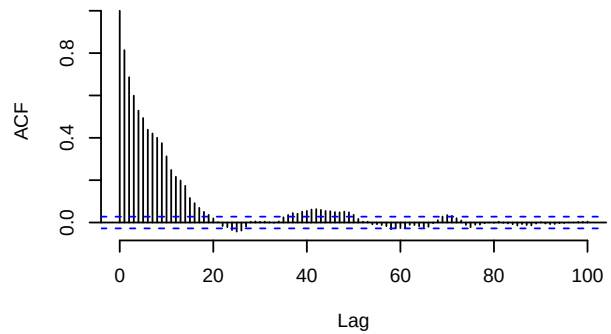


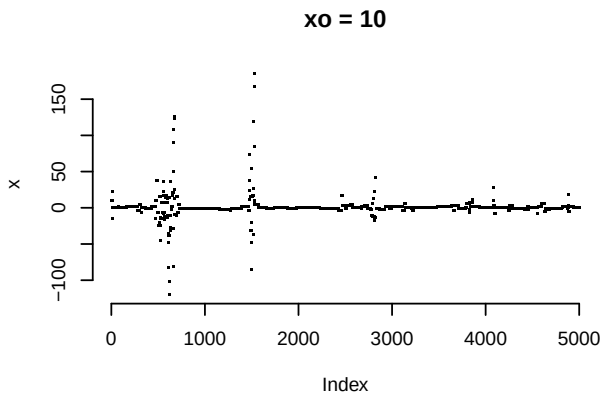


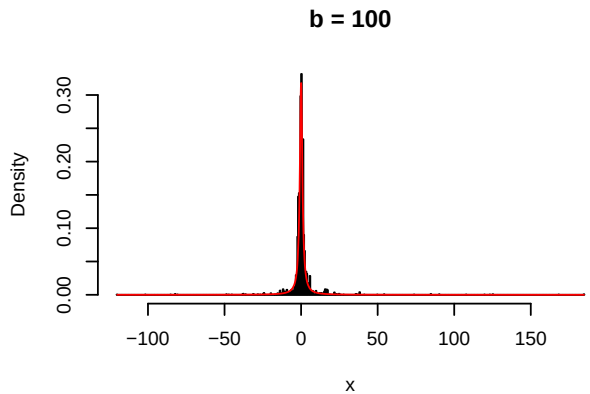
Metropolis: Univariate example 1

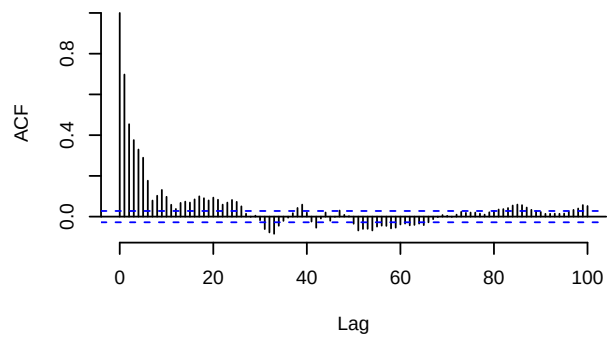


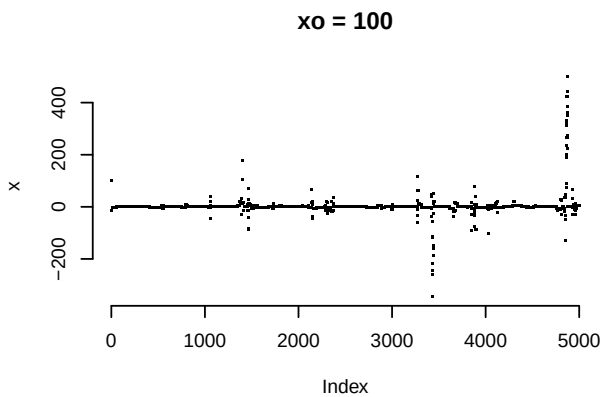


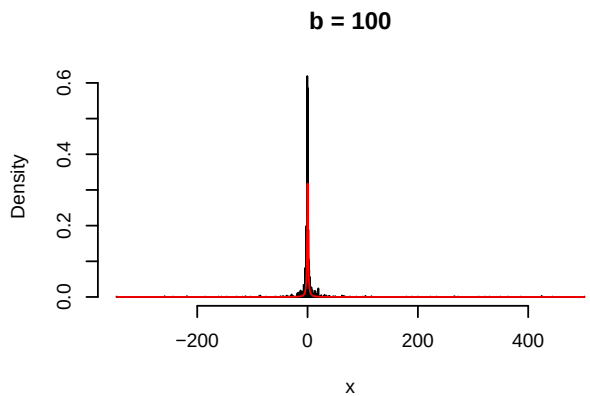


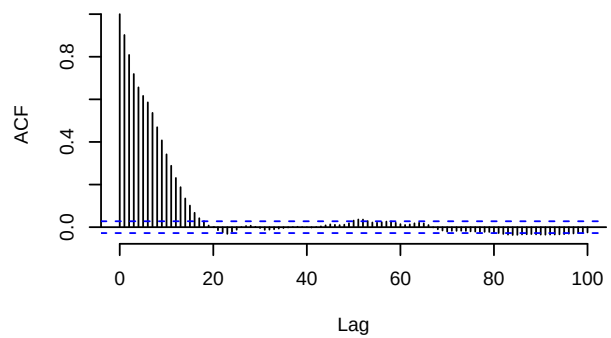




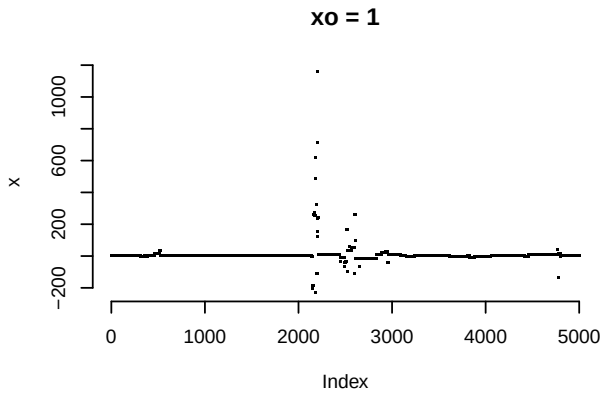


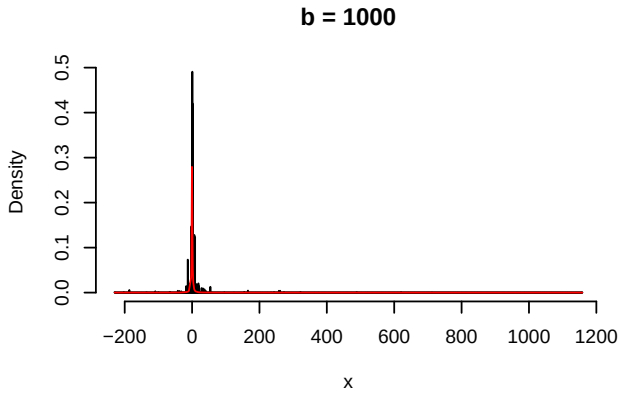


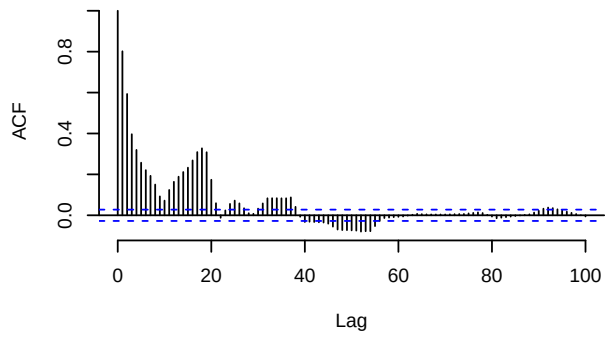


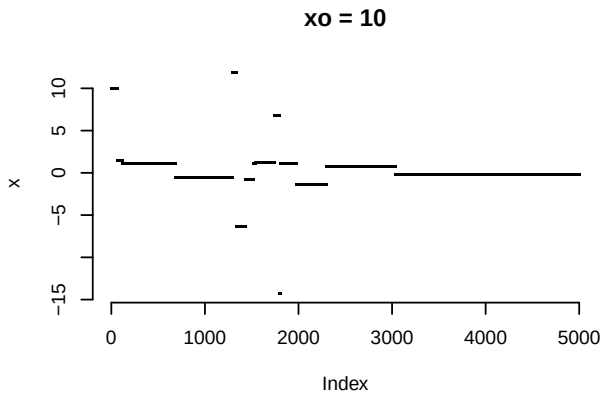


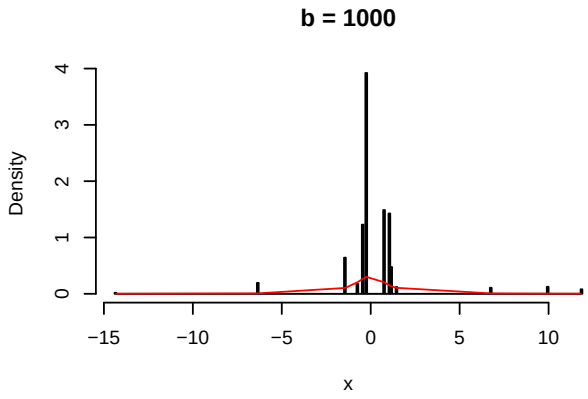
Metropolis: Univariate example 1

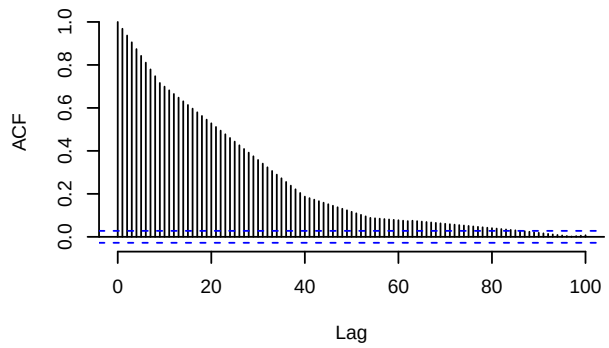


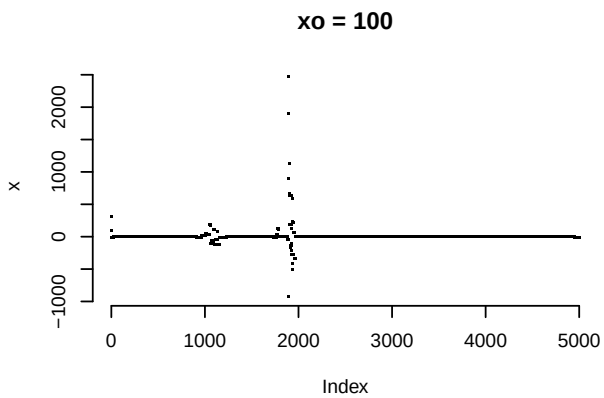


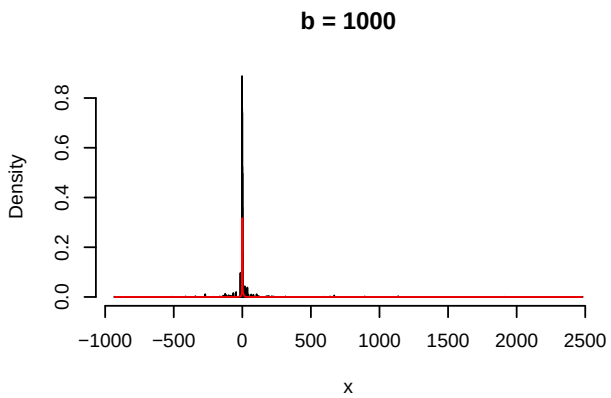


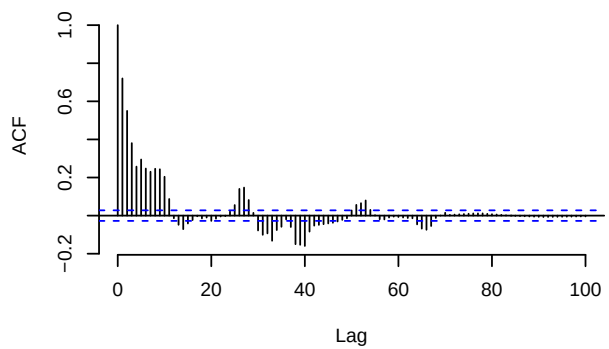












Metropolis: Univariate example 1

Observations

- b small :: ACF large :: long equilibration / burn-in time
- b large :: high rejection rate :: flat regions in x vs. time plot
- b large :: Markov chain may not equilibrate at all
- x_1 away from 0 :: longer equilibration time, by and large
- Non-equilibrium initial transient: Substantial trend variation, time-varying (windowed) averages
- Equilibrium: Flat trend, well-defined / time-independent (windowed) averages

- Desired PDF: Gaussian mixture

$$f_X(x) = \sum_{i=1}^{k} w_i \phi(x, \mu_i, \sigma_i)$$

where
μ_i: mean of the ith component
σ_i: standard deviation of the ith component, $\sigma_i > 0$
w_i: weight of the ith component, $\sum_{i=1}^{k} w_i = 1$
$\phi(\cdot, \mu, \sigma)$: Normal(μ, σ) PDF

- Proposal PDF: $N(x, b^2)$; i.e.,

$$q(y|x) = \frac{1}{\sqrt{2\pi b^2}} \exp\left(-\frac{1}{2}\left(\frac{y-x}{b}\right)^2\right)$$

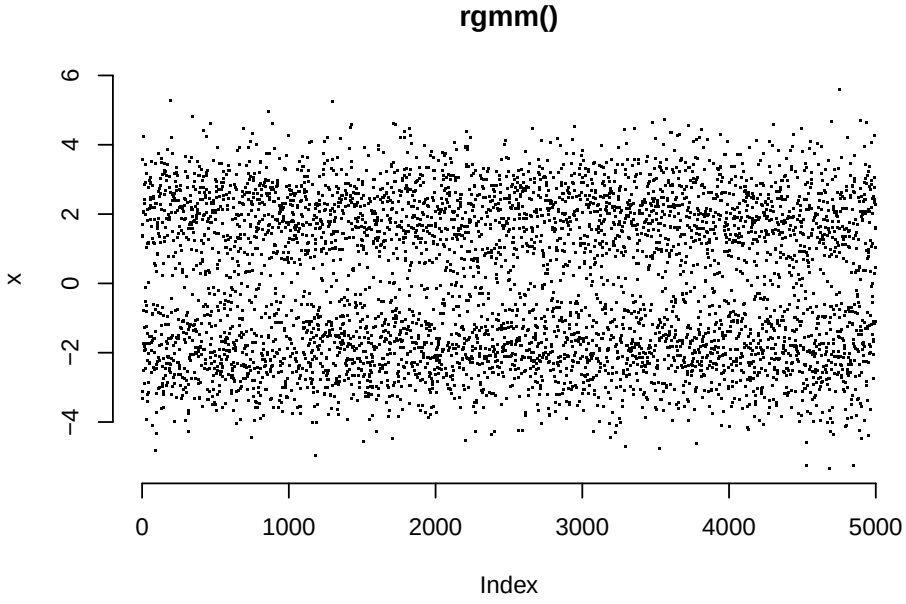
b: the *step size/length* parameter; a tunable parameter.

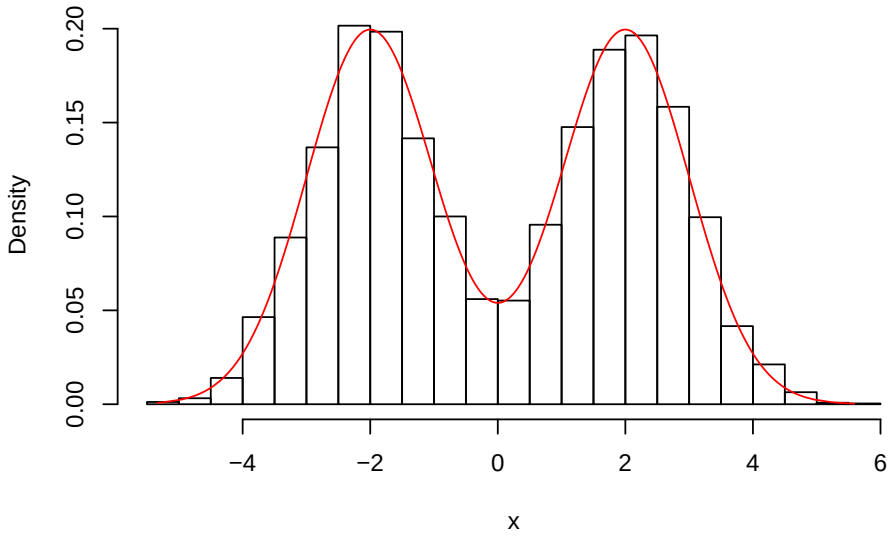
GMM: Illustrative implementation in R

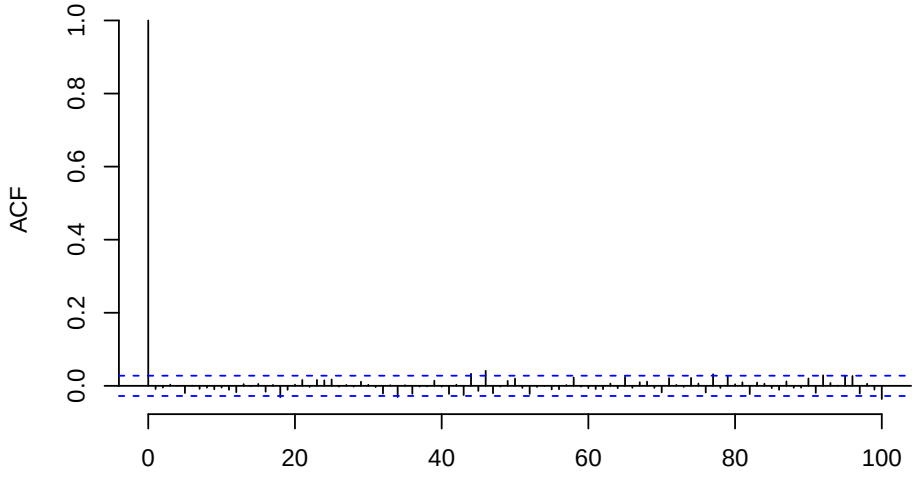
```
dgmm <- function( x, par )
 {
  # k-component univariate Gaussian mixture density
  # par: mixture parameters, columns == (mean,sd,weight),
  #                          rows == mixture components

  stopifnot( is.matrix( par ), ncol( par ) == 3, all( par[,2] > 0 ),
             all( par[,3] >= 0 ), sum( par[,3] ) == 1 )

  sapply( x, function( t ) { sum( par[,3] * dnorm( t, par[,1], par[,2] ) ) } )
 }

rgmm <- function( n, par = matrix( c( -2, 1, 0.5, +2, 1, 0.5 ),
                                   ncol = 3, byrow = TRUE ) )
 {
  # iid random numbers from a k-component univariate Gaussian mixture
  # par: mixture parameters, columns == (mean,sd,weight),
  #                          rows == mixture components

  stopifnot( is.matrix( par ), ncol( par ) == 3, all( par[,2] > 0 ),
             all( par[,3] >= 0 ), sum( par[,3] ) == 1 )

  i <- sample( 1:nrow( par ), n, prob = par[,3], replace = TRUE )

  rnorm( n, par[i,1], par[i,2] )
 }
```

Metropolis: Univariate example 2

IID GMM random numbers, for comparison



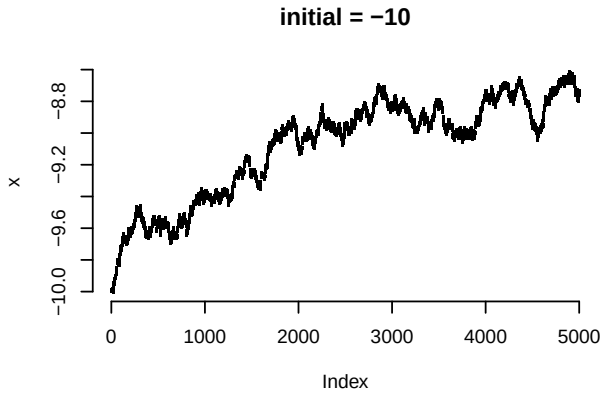


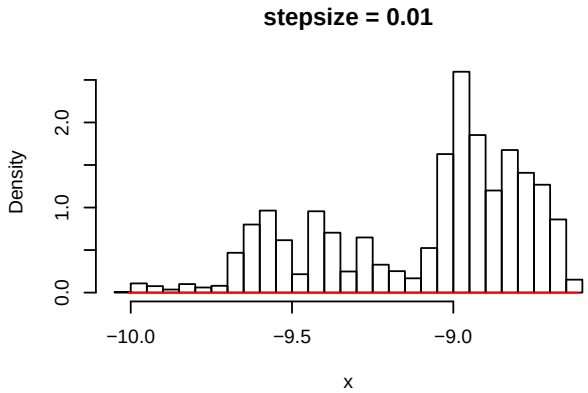


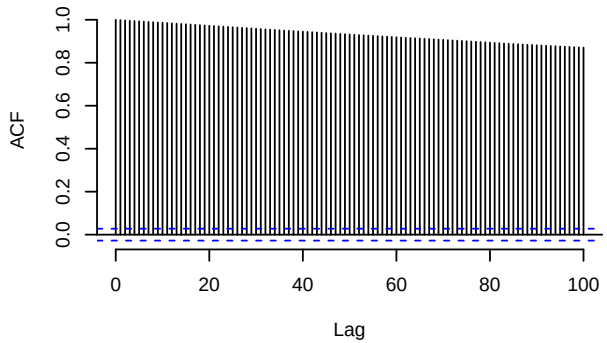
Metropolis MCMC for GMM: Illustrative implementation in R

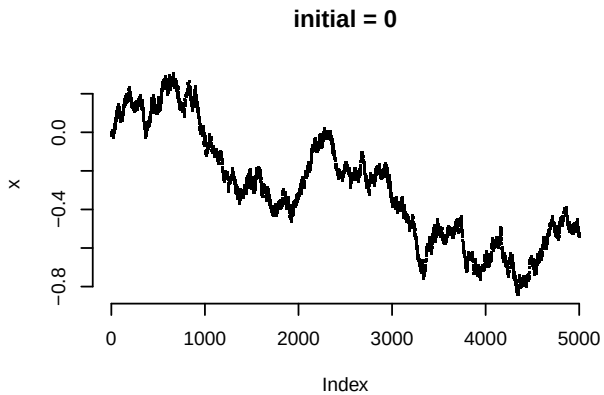
```
source( 'gmm.r' )

rgmm.mcmc <- function( n, initial = 0, stepsize = 1,
                       par = matrix( c( -2, 1, 0.5, +2, 1, 0.5 ),
                                     ncol = 3, byrow = TRUE ) )
 {
  proposal <- function( current, stepsize ) { rnorm( 1, current, stepsize ) }

  accept.m <- function( proposed, current, ... )
   {
    # Metropolis form for a symmetric proposal PDF
    runif( 1 ) <= dgmm( proposed, ... ) / dgmm( current, ... )
   }

  x <- rep( initial, n )

  for ( i in 2:n )
   {
    # generate proposal
    y <- proposal( x[i-1], stepsize )

    # accept or reject
    x[i] <- if ( accept.m( y, x[i-1], par ) ) y else x[i-1]
   }

  x
 }
```

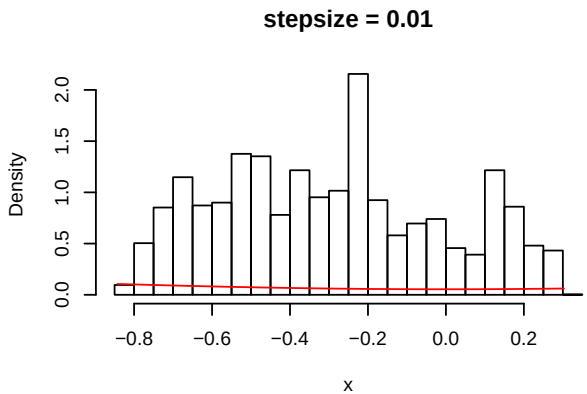
Metropolis: Univariate example 2

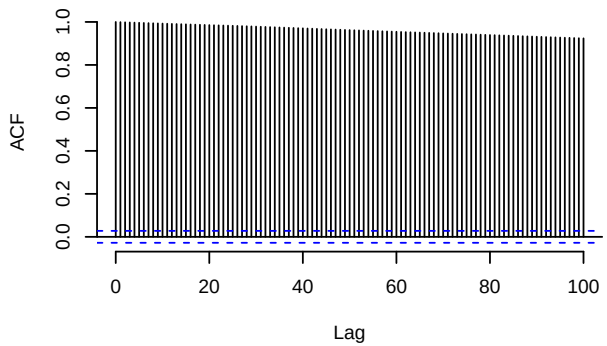


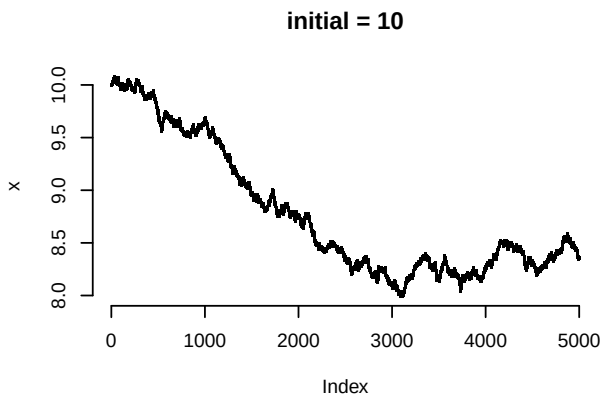


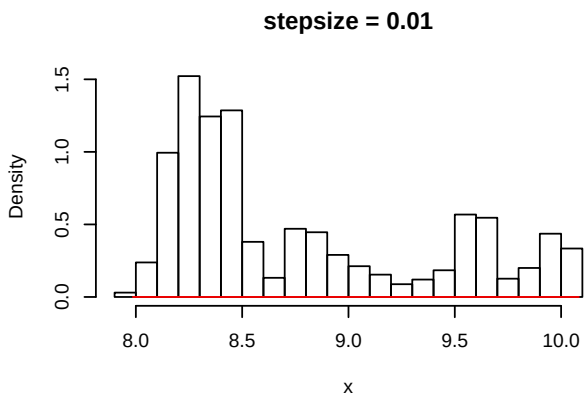


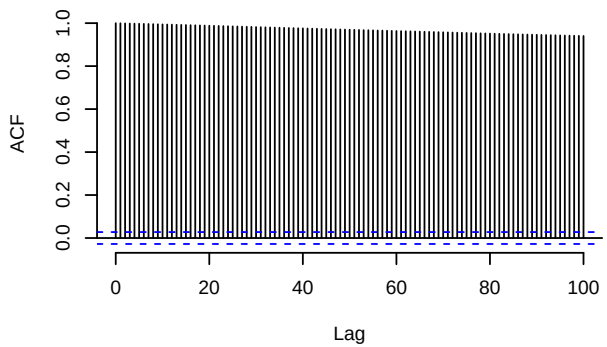




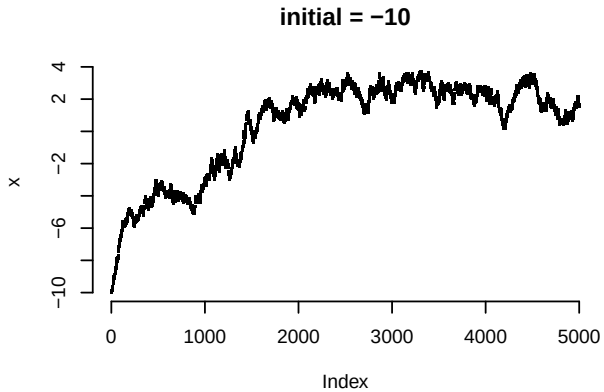


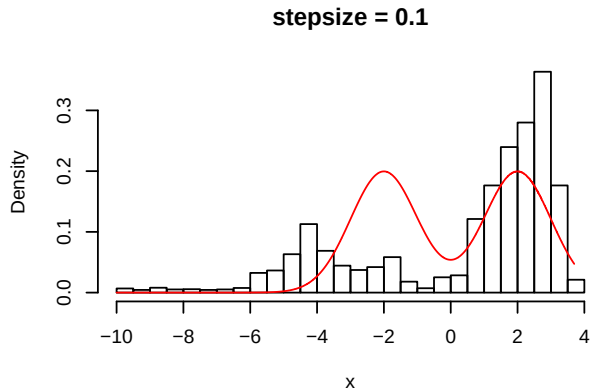


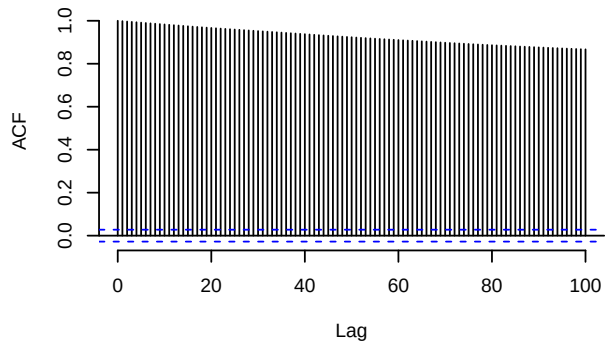


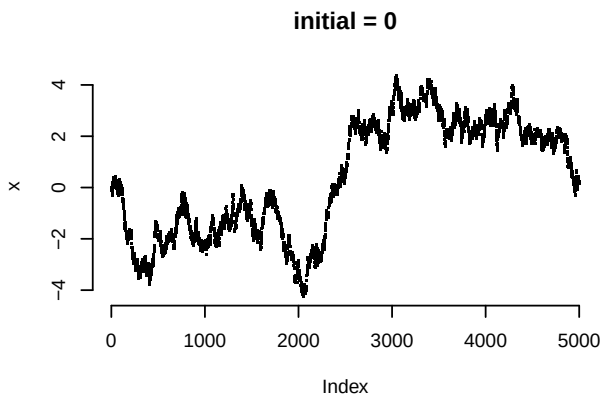


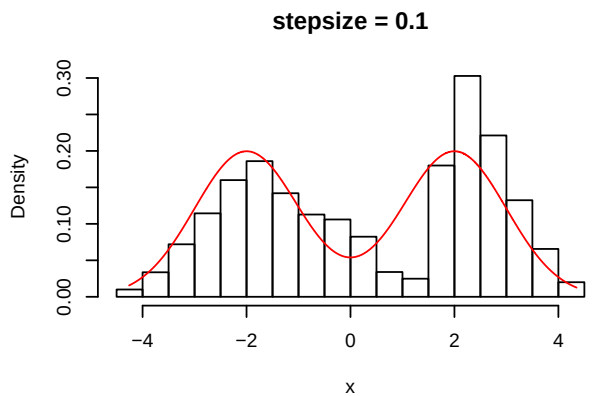
Metropolis: Univariate example 2

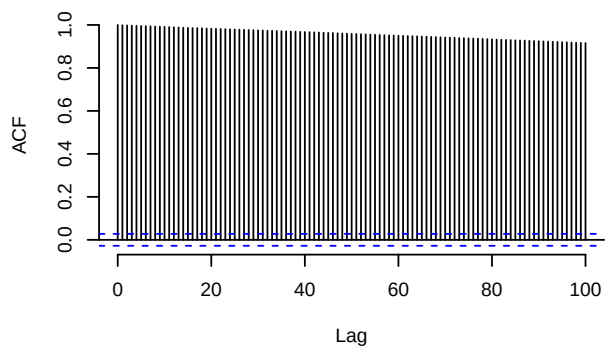


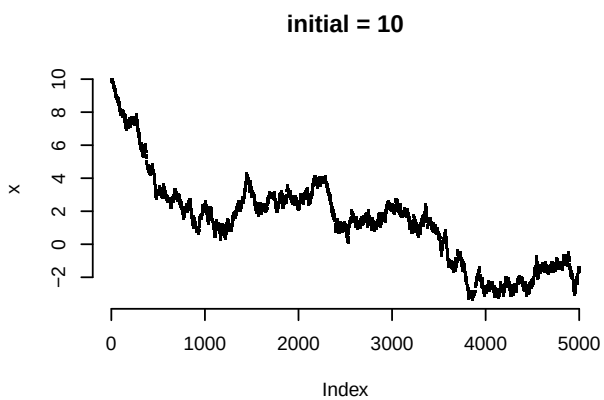


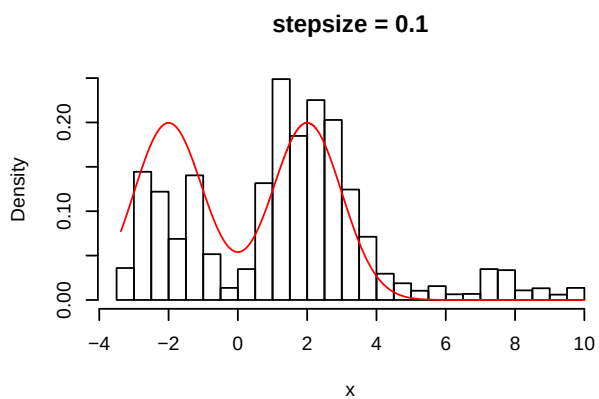


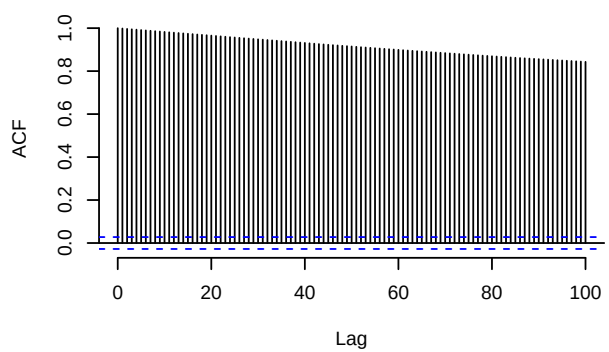




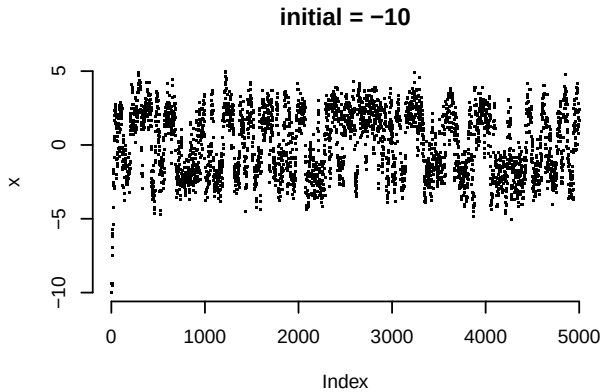


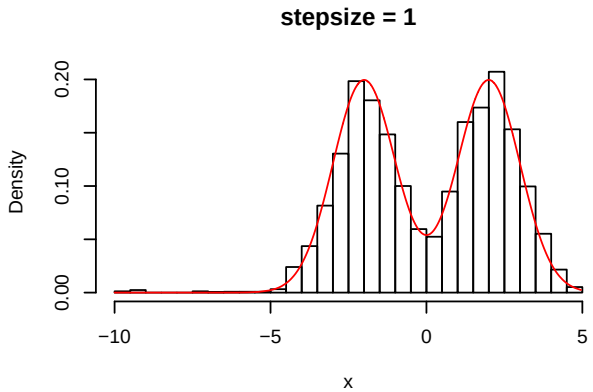


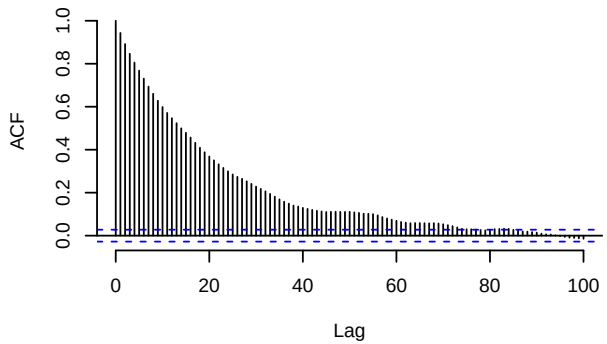


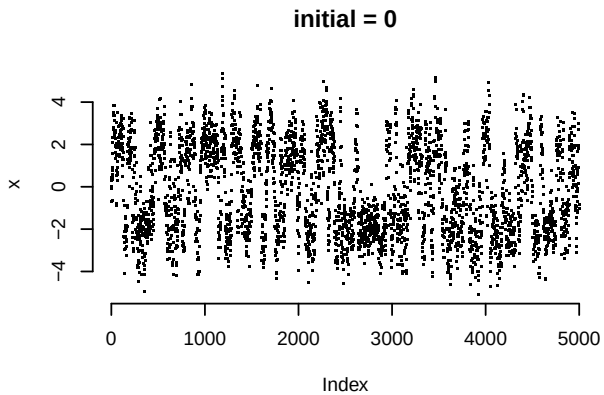


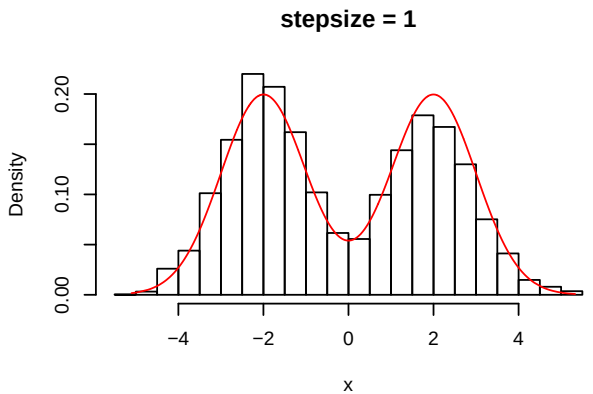
Metropolis: Univariate example 2

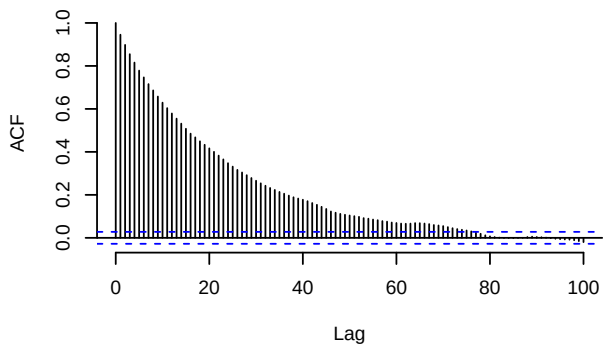


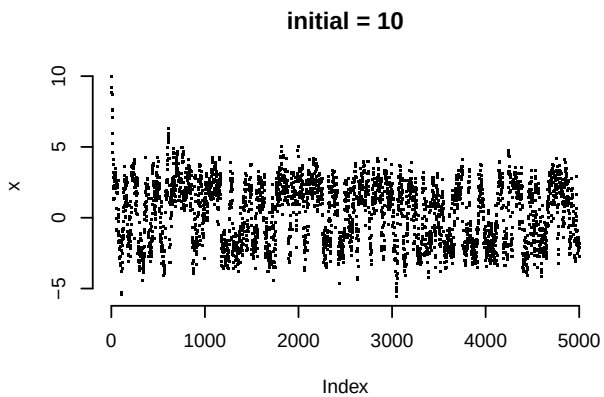


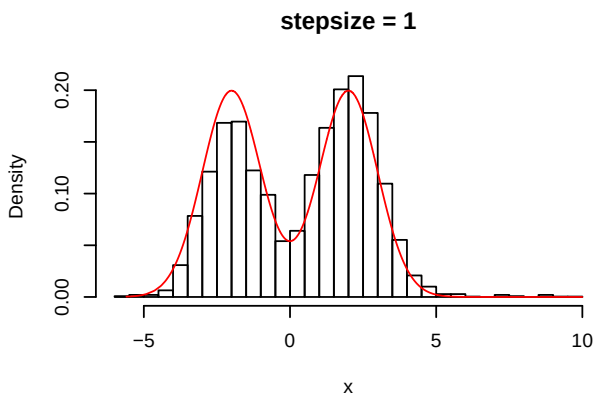


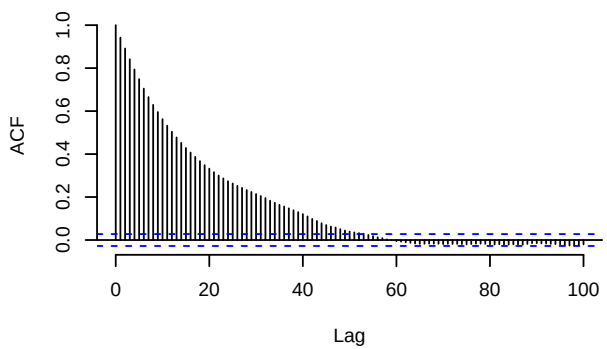




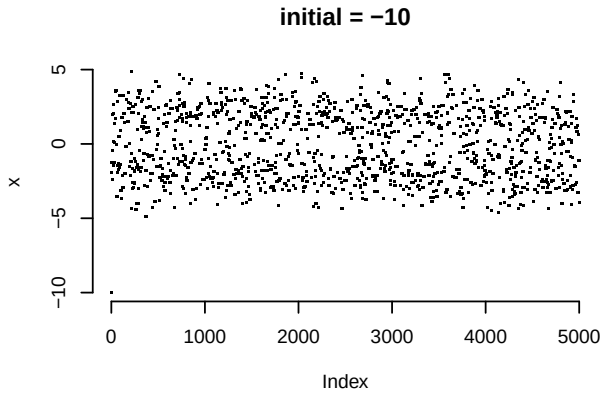


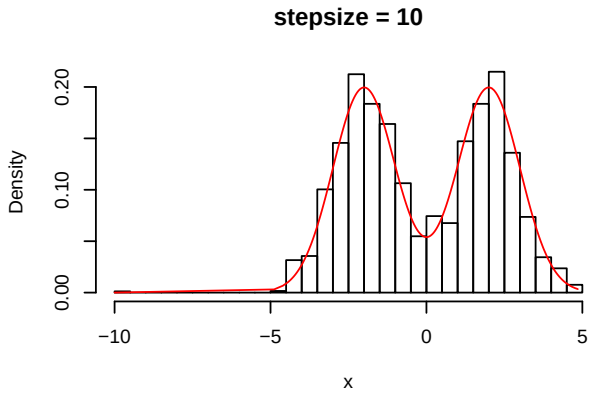


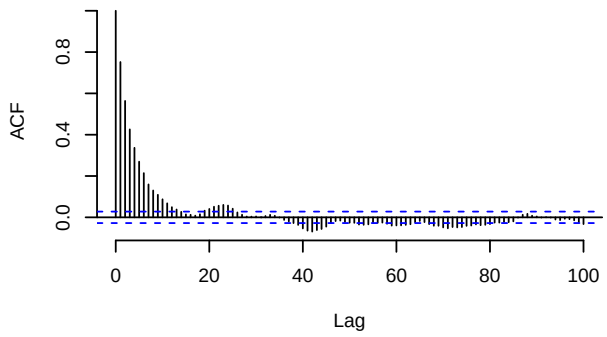


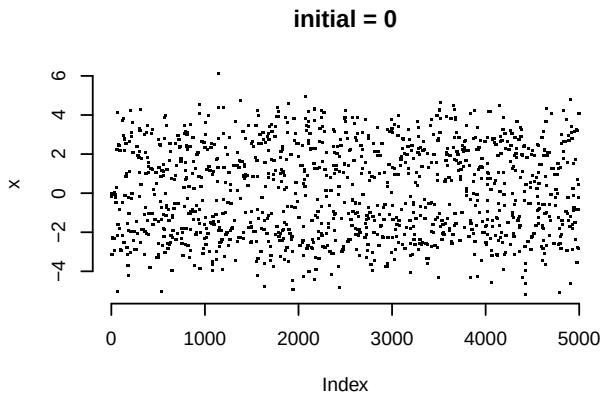


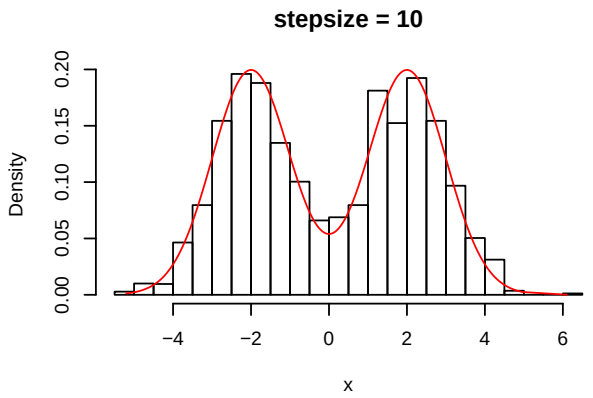
Metropolis: Univariate example 2

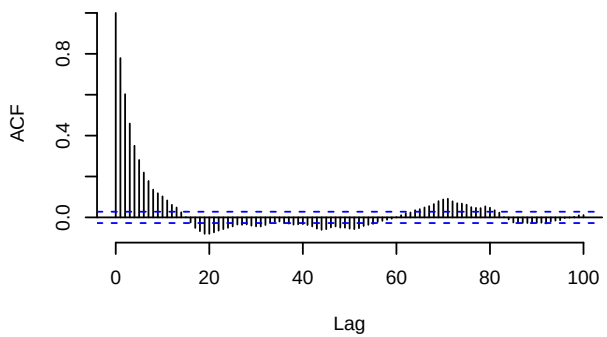


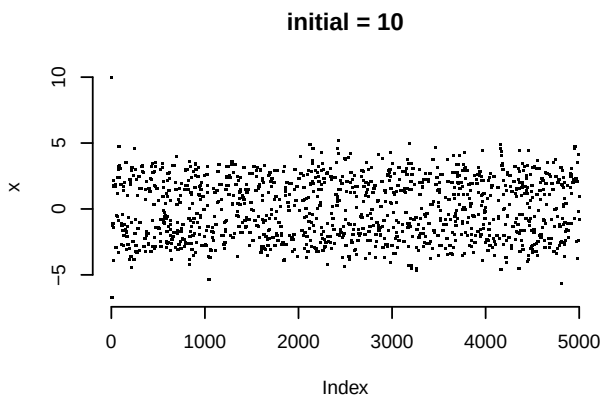


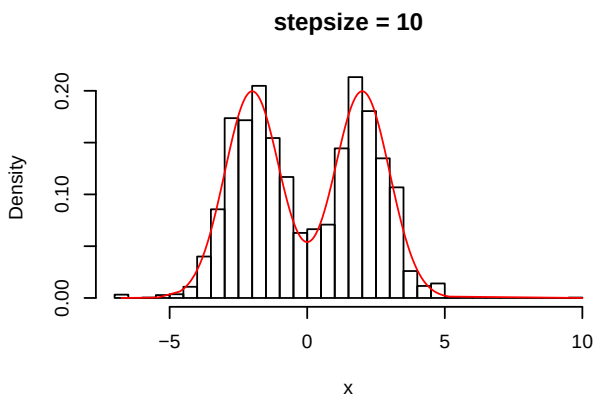


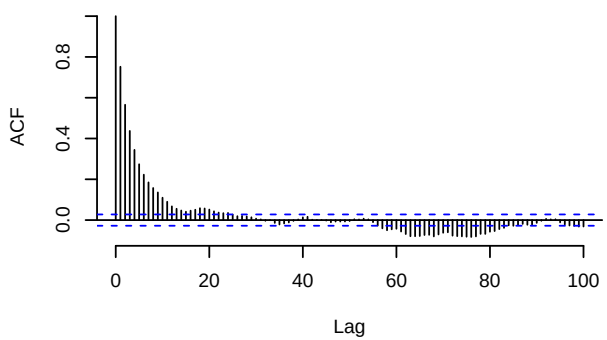




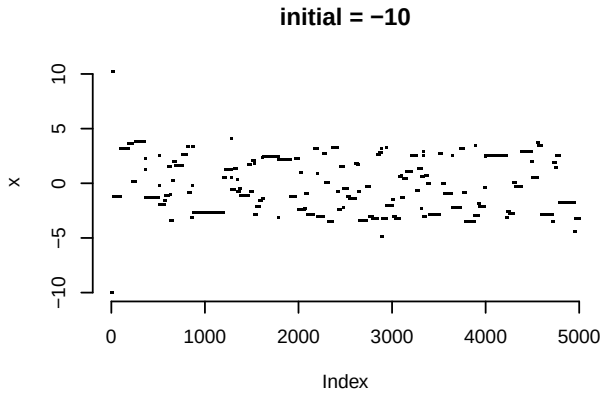


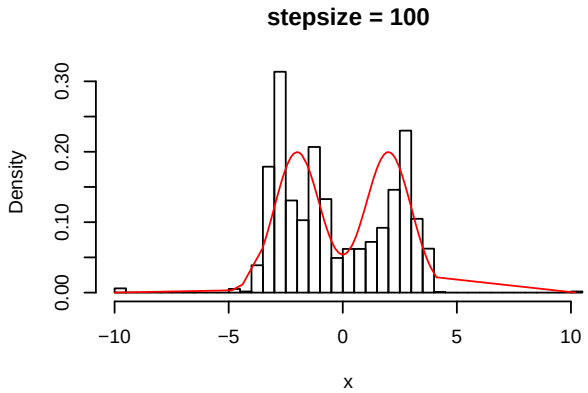


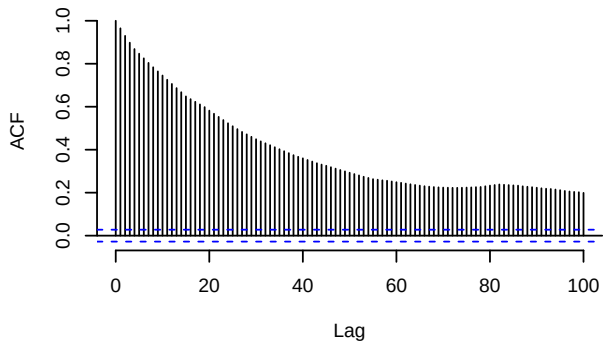


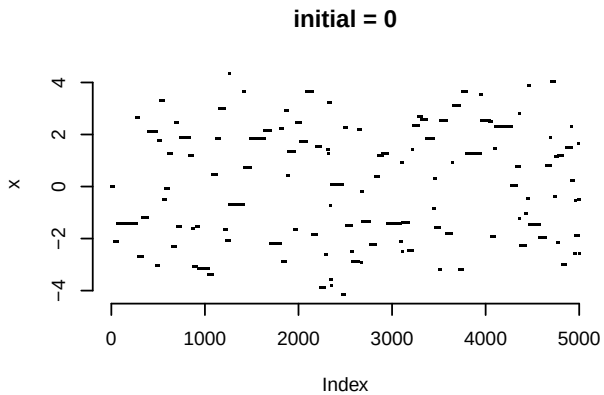


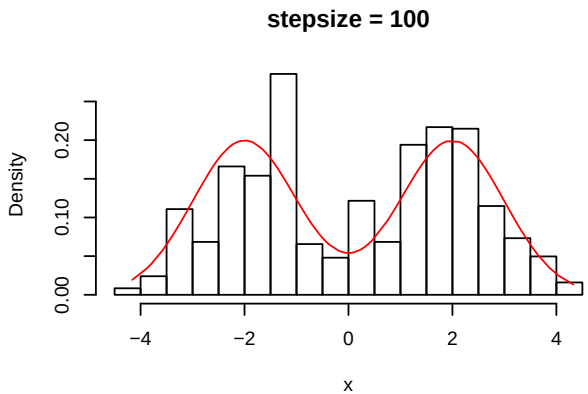
Metropolis: Univariate example 2

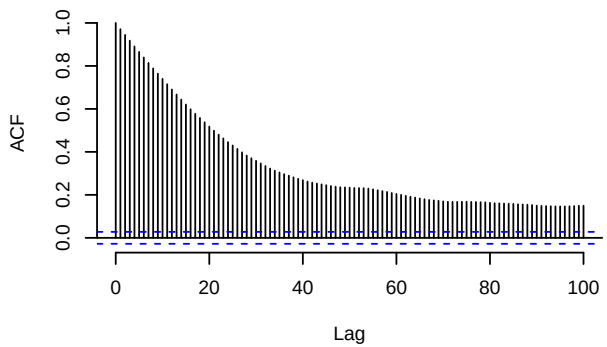


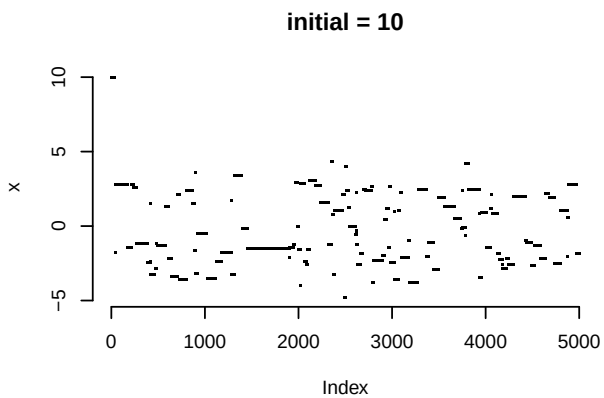


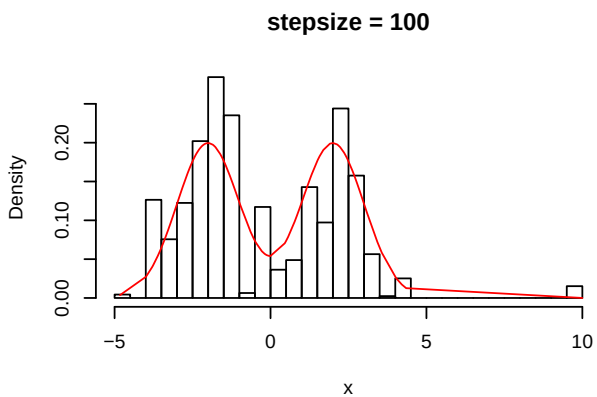


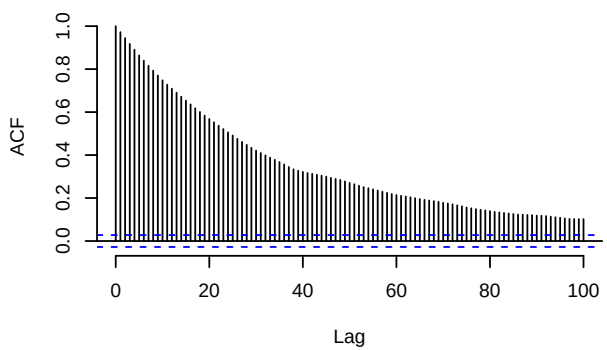












Why is Metropolis-Hastings expected to work?

Recall detailed balance for discrete-state-space Markov chains: $\pi_i \mathcal{P}_{ij} = \pi_j \mathcal{P}_{ji}$ for every pair of states i, j

For the Markov chain set up by M-H, the Markov matrix / kernel is

$$\mathcal{P}(y|x) = q(y|x) \times a(y|x).$$

The condition of detailed balance for M-H, if it holds, would be

$$\mathcal{P}(y|x) \times f(x) = \mathcal{P}(x|y) \times f(y).$$

We will show that this indeed is the case.

Why is Metropolis-Hastings expected to work?

LHS

$$\begin{aligned}
\mathcal{P}(y|x)f(x) &= q(y|x)a(y|x)f(x) \\
&= f(x)q(y|x) \times \min\left\{1, \frac{f(y)}{f(x)}\frac{q(x|y)}{q(y|x)}\right\} \\
&= \min\left\{f(x)q(y|x), f(y)q(x|y)\right\}.
\end{aligned}$$

RHS

$$\begin{aligned}
\mathcal{P}(x|y)f(y) &= q(x|y)a(x|y)f(y) \\
&= f(y)q(x|y) \times \min\left\{1, \frac{f(x)}{f(y)}\frac{q(x|y)}{q(y|x)}\right\} \\
&= \min\left\{f(y)q(x|y), f(x)q(x|y)\right\}.
\end{aligned}$$

LHS $=$ RHS $\implies$ detailed balance satisfied by Metropolis-Hastings

Note: (1) $c\min\{a, b\} = \min\{ca, cb\}$ for $c > 0$, and (2) $\min\{a, b\} = \min\{b, a\}$

Basic MCMC: Schematic workflow

Input

1. The *target* or *desired* (multivariate) density $f(x)$
2. A friendly & easily sampleable *proposal* density $q(y|x)$
3. An initial / starting state x_1

Schematic MCMC workflow

1. Using Metropolis-Hastings, generate a Markov chain sample $x_1, \ldots, x_M, x_{M+1}, \ldots, x_{M+N}$.
2. Discard the first M (TBD later).
3. Random sample: $x_{M+1}, \ldots, x_{M+N}$.
4. Estimate of $I = \int h(x)f(x)dx$: $\widehat{I}_N = \frac{1}{N} \sum_{i=M+1}^{N} h(x_i)$ etc., as in Monte Carlo integration.

Metropolis-Hastings: What it does & what it doesn't

- M-H sets up a Markov chain – by design.
- M-H ensures the condition of detailed balance so that the desired PDF $f(x)$ is *a* stationary PDF of the Markov chain: Recall that detailed balance is *sufficient* to ensure stationarity, but not *necessary*.
- M-H *does not* ensure that $f(x)$ is the limiting PDF of the Markov chain: This part needs to be ensured through wise choice of the proposal PDF $q(y|x)$ and any tunable parameters therein.
- Convergence theorems related to ergodic Markov chains ensure that expected values can be estimated via long-time averaging over a realization of the chain.

If the proposal PDF is *symmetric*; i.e.,

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

then

$$a(y|x) = \min\left\{1, \frac{f(y)}{f(x)}\right\}.$$

This is the (original) Metropolis algorithm.

Proposed uphill moves are always accepted.
Proposed downhill moves are not necessarily rejected.