Computational Statistics - MCMC

Sprint Lead: Chris Nemeth

Group 2

Max Howell, Robert Lambert, Adam Page, Wanchen Yue

Introduction

What is MC & MC?

Markov Chain:

- Transition kernel K:

$$K(A|x) = \mathbb{P}(X_t \in A|X_{t-1} = x).$$

- This kernal then has transition density p:

$$K(A|x) = \int_A p(y|x)\mathrm{d}y$$

- The chain then has an invariant distribution Π with density $\pi(x)$:

$$\Pi(A) = \int K(A|x)\pi(x)\mathrm{d}x$$

What is MC & MC?

Markov Chain:

- Transition kernel K:

$$K(A|x) = \mathbb{P}(X_t \in A | X_{t-1} = x).$$

- This kernal then has transition density p:

$$K(A|x) = \int_A p(y|x)\mathrm{d}y$$

- The chain then has an invariant distribution Π with density $\pi(x)$:

$$\Pi(A) = \int K(A|x)\pi(x)\mathrm{d}x$$

Monte Carlo (Integration):

- Mathematically:

$$\theta = \mathbb{E}\left[\phi(X)\right] = \int \phi(x)\pi(x)\mathrm{d}x$$

- Allows us to take a sample average:

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$$

What is MCMC?

Joining the two MCs together:

- We know a distribution, $\pi(x)$, up to some constant of proportionality.
- Construct a Markov Chain whose stationary distribution is $\pi(x)$.
- Simulate N samples, $x_1, \ldots, x_N$. Remove the burn-in period.
- Using Monte Carlo we can estimate posterior expectations and probabilities.

This is the base idea of MCMC.

Metropolis-Hastings

For some proposal density, q, we have an acceptance probability that we 'move' states of,

$$\alpha(x|y) = \min\left\{1, \frac{\pi(x)q(y|x)}{\pi(y)q(x|y)}\right\},$$

where π is the stationary distribution of our Markov chain.

Metropolis-Hastings

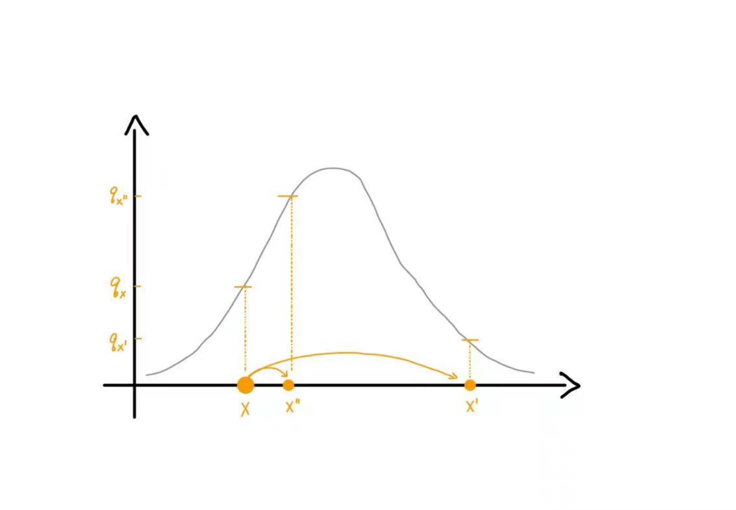


Figure 1: Proposal Distribution Illustrated

MCMC Algorithms

MCMC Algorithms

There exist a number of algorithms that can give us a proposal density for our Markov chain's dynamics.

MCMC Algorithms

There exist a number of algorithms that can give us a proposal density for our Markov chain's dynamics.

By using the Metropolis-Hastings algorithm we can obtain an acceptance probability for each proposed change of state within the Markov Chain.

Random Walk

The most primitive MCMC algorithm is the Random Walk Metropolis Algorithm - for which the proposal distribution is a random walk.

Random Walk

The most primitive MCMC algorithm is the Random Walk Metropolis Algorithm - for which the proposal distribution is a random walk.

If in the current state x, the proposed next state x' is obtained using the random variable X':

$$X' = x + \epsilon$$

Where ϵ is a zero mean random variable.

Random Walk

The most primitive MCMC algorithm is the Random Walk Metropolis Algorithm - for which the proposal distribution is a random walk.

If in the current state x, the proposed next state x' is obtained using the random variable X':

$$X' = x + \epsilon$$

Where ϵ is a zero mean random variable.

This transition is accepted with probability $\alpha(x' \mid x)$, obtained using the Metropolis-Hastings algorithm.

Random Walk

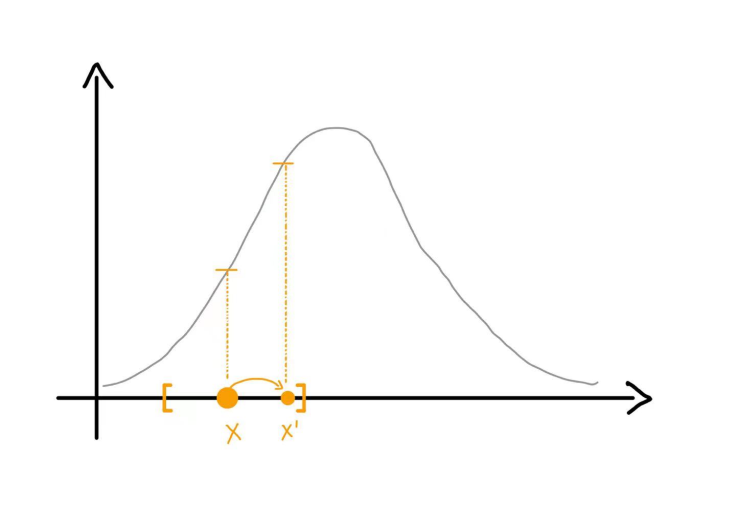


Figure 2: Visualisation of RWM

Random Walk

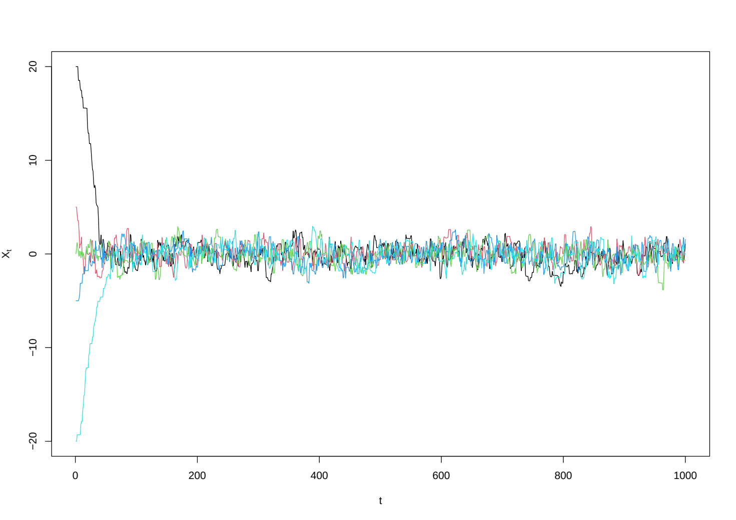


Figure 3: Trace plot of RW for N(0,1) with starting values: 20,5,0,-5,-20

Random Walk

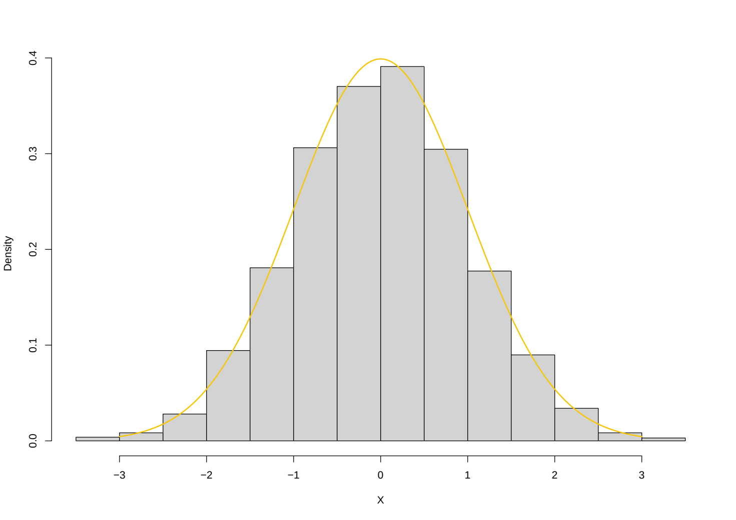


Figure 4: Histogram of RW for N(0,1)

Metropolis-adjusted Langevin algorithm

Even with appropriately tuned parameters, RWM proposes a new state x' at random.

Metropolis-adjusted Langevin algorithm

Even with appropriately tuned parameters, RWM proposes a new state x' at random.

If we instead, try to model our Markov chain as a discretised Langevin diffusion, we can use Langevin dynamics to propose a new state based upon a random walk displaced by some factor of the gradient of the proposed density.

Metropolis-adjusted Langevin algorithm

Even with appropriately tuned parameters, RWM proposes a new state x' at random.

If we instead, try to model our Markov chain as a discretised Langevin diffusion, we can use Langevin dynamics to propose a new state based upon a random walk displaced by some factor of the gradient of the proposed density.

$$x' = x + \frac{h}{2}\nabla log(\pi(x)) + \sqrt{h}\epsilon_t$$

Where $\epsilon_t \sim \mathcal{N}(0, 1)$.

Metropolis-adjusted Langevin algorithm

If at every timepoint we were to select the proposed x' then we would be carrying out the Unadjusted Langevin Algorithm (ULA).

Metropolis-adjusted Langevin algorithm

If at every timepoint we were to select the proposed x' then we would be carrying out the Unadjusted Langevin Algorithm (ULA).

Whilst not producing samples exactly from the proposed density $\pi(x)$ it is particularly useful in Big Data settings.

Metropolis-adjusted Langevin algorithm

If at every timepoint we were to select the proposed x' then we would be carrying out the Unadjusted Langevin Algorithm (ULA).

Whilst not producing samples exactly from the proposed density $\pi(x)$ it is particularly useful in Big Data settings.

If were instead to accept the proposed x' with probability α (obtained from using the MH algorithm) we would be using the Metropolis-adjusted Langevin algorithm which does sample from the proposed density $\pi(x)$.

$$q\left(x' \mid x\right) = N\left(x'; x + \frac{h}{2}\nabla \log \pi(x), h\right)$$

Metropolis-adjusted Langevin algorithm

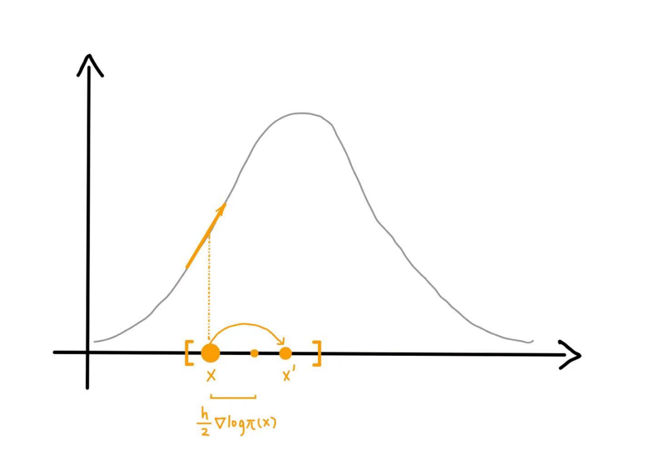


Figure 5: MALA Visualisation

Metropolis-adjusted Langevin algorithm

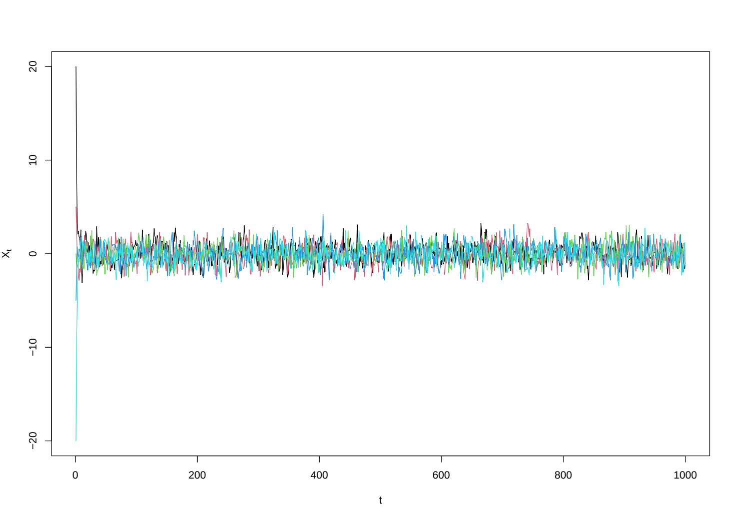


Figure 6: Trace plot of MALA for N(0,1) with starting values: 20,5,0,-5,-20

Metropolis-adjusted Langevin algorithm

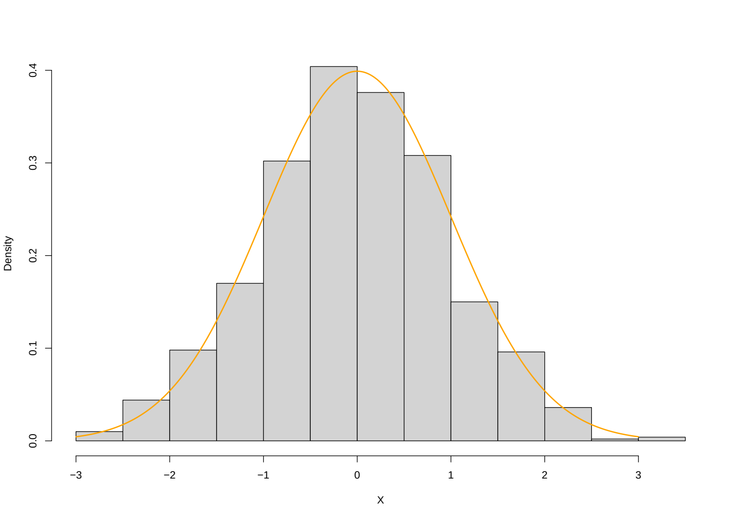


Figure 7: Histogram of MALA for N(0,1)

Gibbs Sampling

For $x = (x_1, x_2, ..., x_d)$ the joint distribution of x may be complicated. But if some conditional distributions are known this can be taken advantage of.

Gibbs Sampling

For $x = (x_1, x_2, ..., x_d)$ the joint distribution of x may be complicated. But if some conditional distributions are known this can be taken advantage of.

We make a proposal of

$$q(x' \mid x) = \pi(x_i \mid x_{-i})$$

.

Gibbs Sampling

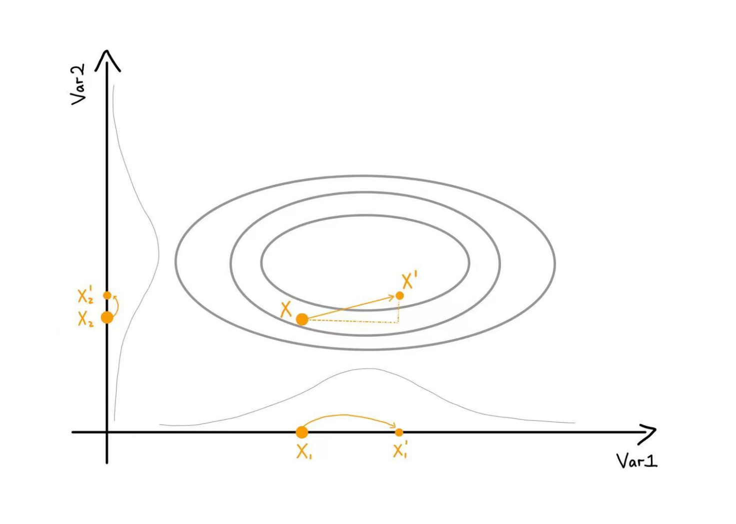


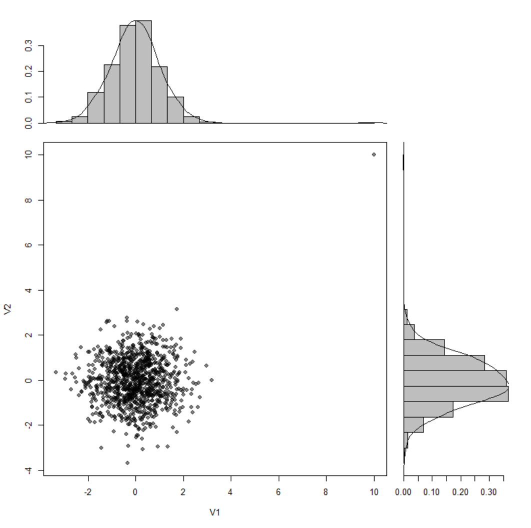
Figure 8: Gibbs Visualisation

Gibbs Sampling

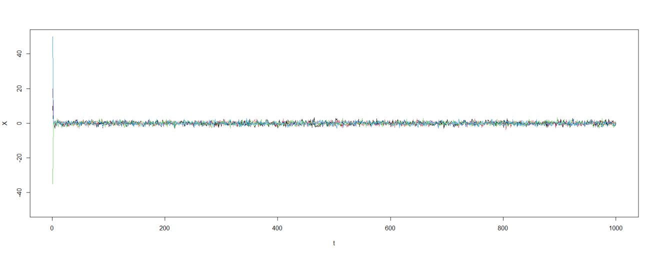
In the context of Metropolis-Hastings the acceptance probability is always 1.

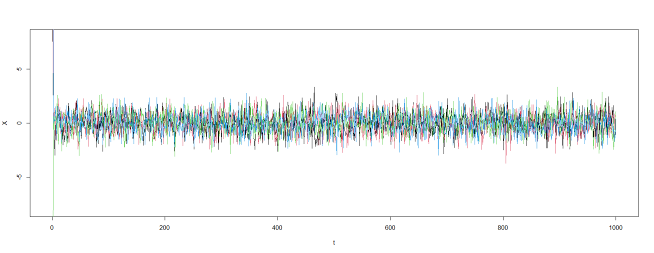
$$
\begin{aligned}
\alpha(x' \mid x) &= \min\left\{1, \frac{\pi(x')q(x \mid x')}{\pi(x)q(x' \mid x)}\right\} \\
\alpha(x_i \mid x_{-i}) &= \min\left\{1, \frac{\pi(x_i)\pi(x_{-i} \mid x_i)}{\pi(x_{-i})\pi(x_i \mid x_{-i})}\right\} \\
&= \min\left\{1, \frac{\pi(x_i) \times \frac{\pi(x_{-i}, x_i)}{\pi(x_i)}}{\pi(x_{-i}) \times \frac{\pi(x_i, x_{-i})}{\pi(x_-i)}}\right\} \\
&= \min\left\{1, \frac{\pi(x)}{\pi(x)}\right\} \\
&= 1
\end{aligned}
$$

Gibbs Sampling-Implementation

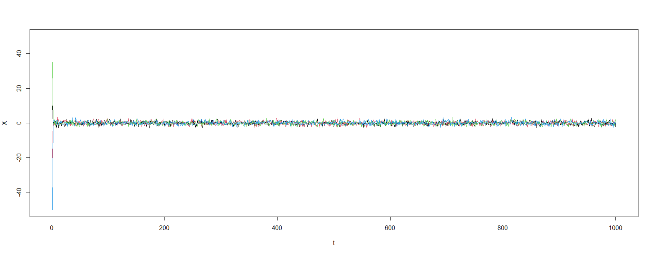


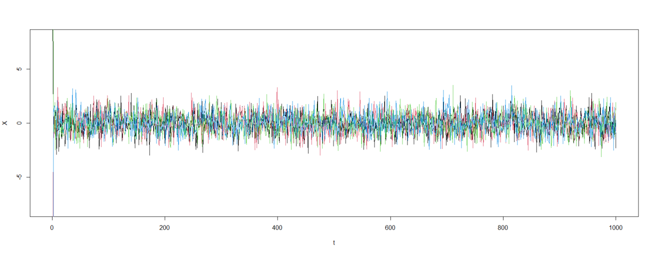
Gibbs Sampling-Implementation X_1





Gibbs Sampling-Implementation X_2

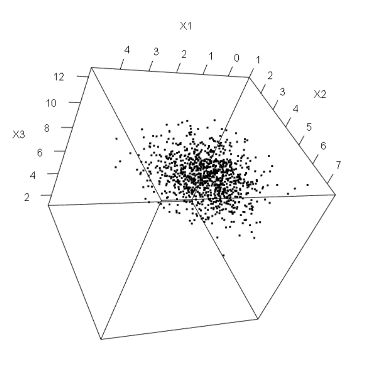




Gibbs Sampling - 3D case

$$\mu = \begin{pmatrix} 2 \\ 5 \\ 10 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.4 \\ 0.2 & 0.4 & 1 \end{pmatrix}$$



Comparisons

Target Distribution

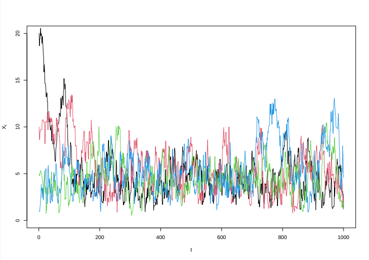
In order to test the limitations of each of the algorithms, we trial them using a number of different target distributions:

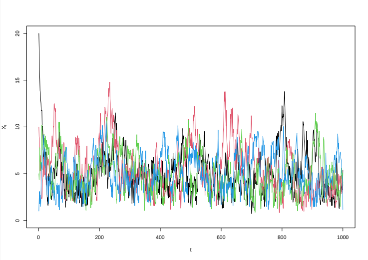
Target Distribution

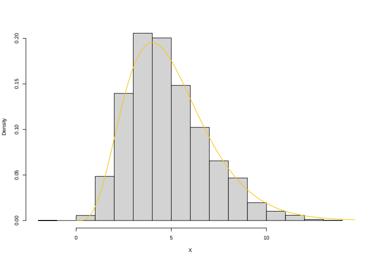
In order to test the limitations of each of the algorithms, we trial them using a number of different target distributions:

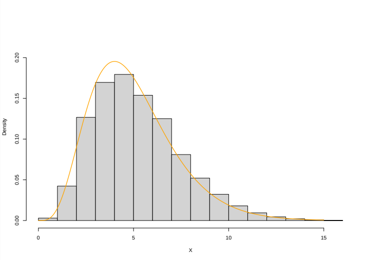
- Skewed distribution
- Heavy tailed distribution
- Multi-mode distribution

Target Distribution: Gamma(5,1)

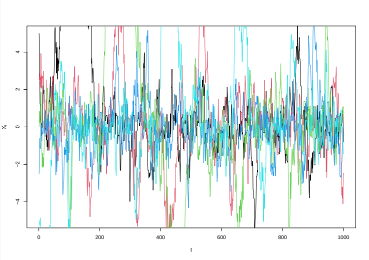


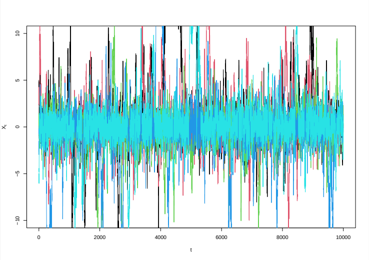


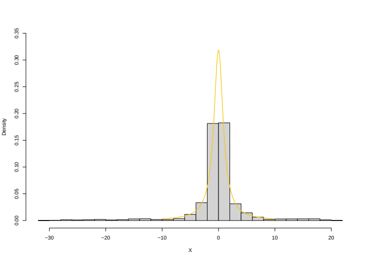


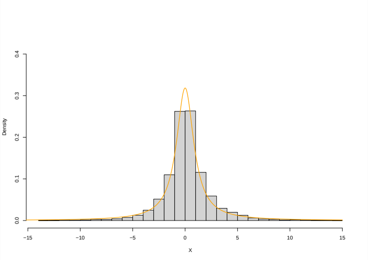


Target Distribution: Student t(1)

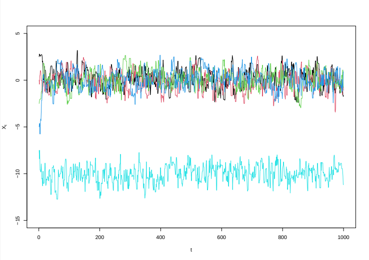


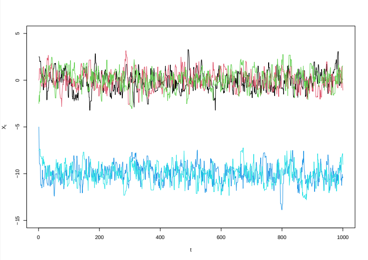


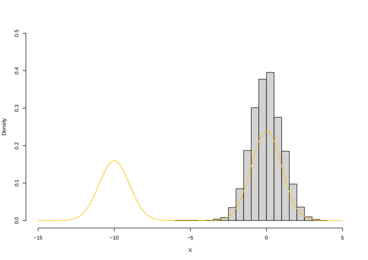


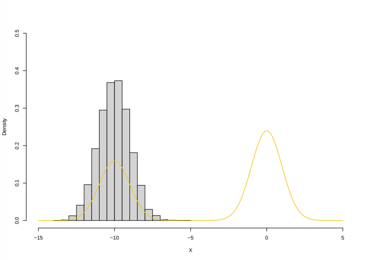


Target Distribution: Bimodal(-10,0) - RW

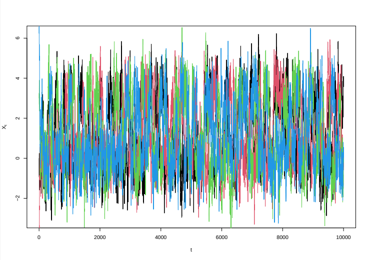


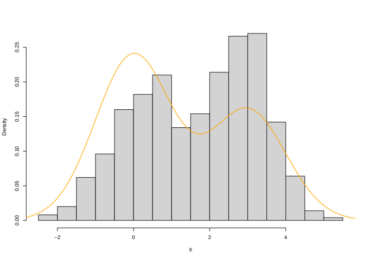


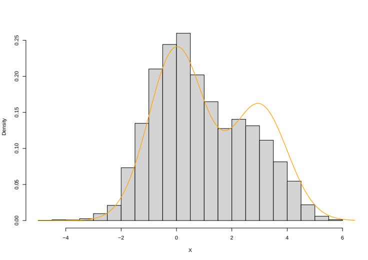


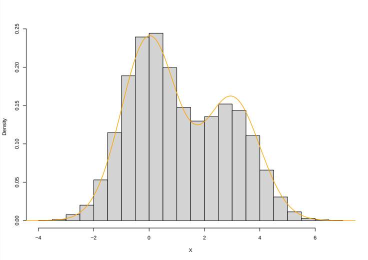


Target Distribution: Bimodal(0,3) - MALA

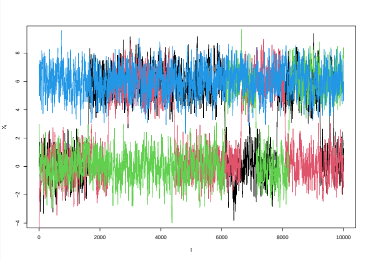


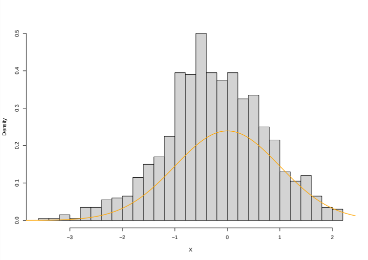


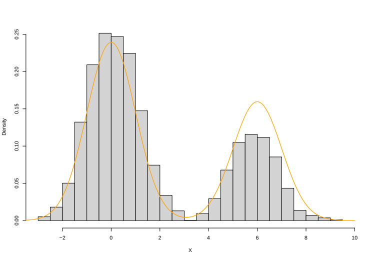


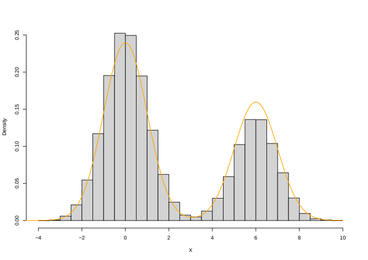


Target Distribution: Bimodal(0,6) - MALA

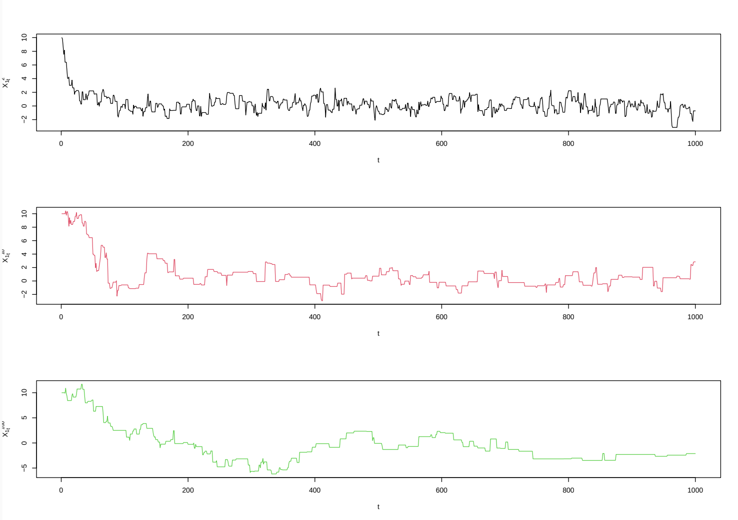




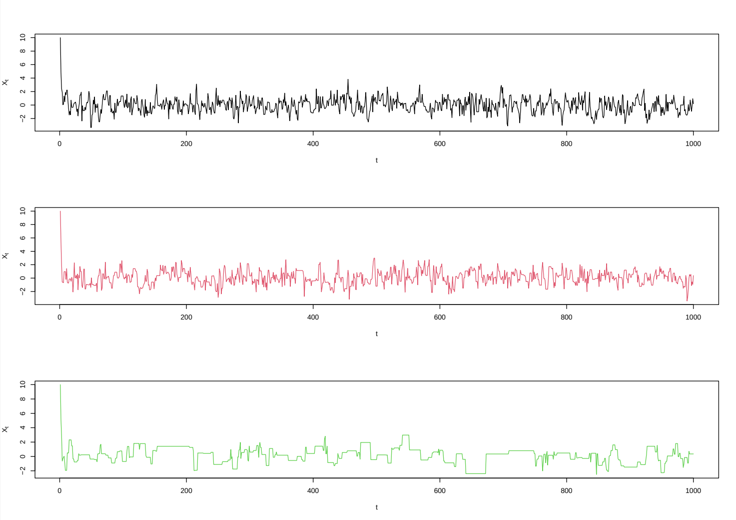




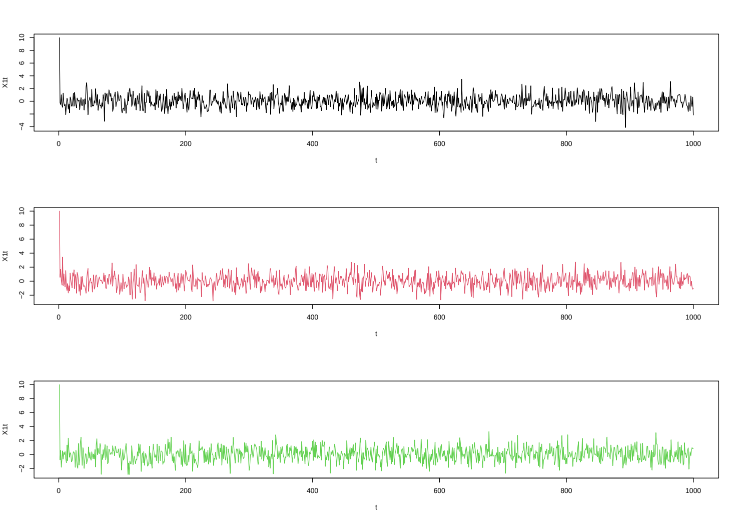
Dimensions: RW d=2,10,100



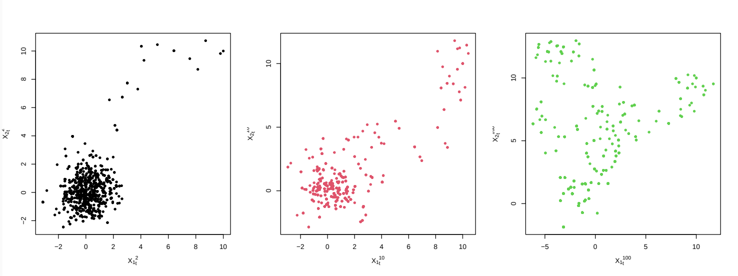
Dimensions: MALA d=2,10,100



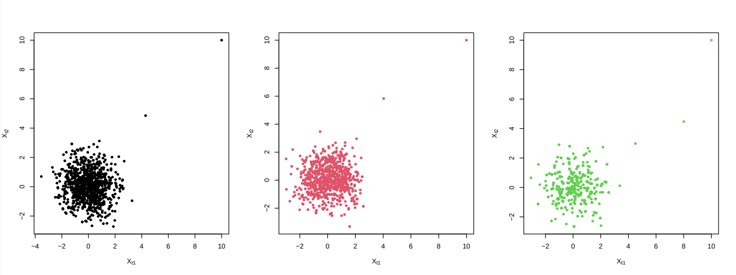
Dimensions: Gibbs d=2,10,100



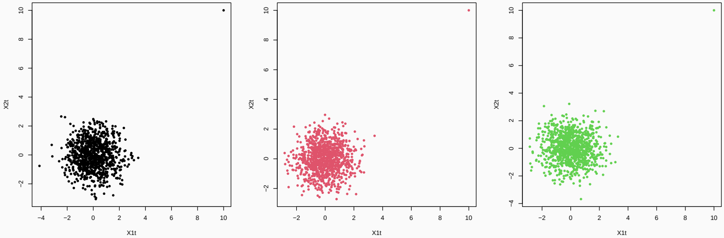
Dimensions: RW d=2,10,100



Dimensions: MALA d=2,10,100



Dimensions: Gibbs d=2,10,100



Convergence Diagnostics

Asymptotic Distribution

CLT for Markov Chain:

$$\sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n} g\left(X_i\right) - \mathrm{E}(g(X))\right) \xrightarrow{d} N\left(0, \sigma^2\right) \quad (n \to \infty)$$

$$\begin{aligned}\sigma^2 &= \operatorname{Var}\left(g\left(X_0\right)\right) + 2\sum_{k=1}^{\infty} \operatorname{cov}\left(g\left(X_0\right), g\left(X_k\right)\right) \\ &= \operatorname{Var}(g(X_0)[1 + 2\sum_{k=1}^{\infty} \operatorname{corr}(g(X_0), g(X_k))]\end{aligned}$$

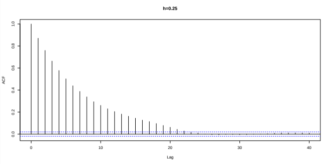
Effective Sample Size

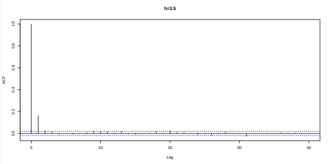
Size k of an iid sample $Y_1, \ldots Y_k \sim f$ whose average $\frac{1}{k}\sum_{i=1}^{k} g(Y_i)$ has the same variance as $\frac{1}{n-b}\sum_{i=1}^{k} g(X_i)$

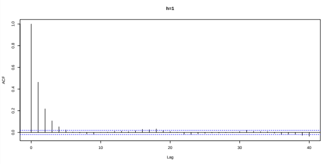
$$k = \frac{n-b}{1 + 2\sum_{k=1}^{\infty} \operatorname{corr}\left(g\left(X_0\right), g\left(X_k\right)\right)}$$

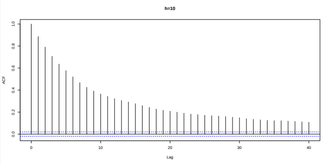
Step Size Tuning: MALA

```
h: 0.25 ESS: 686.5147 Accept Rate: 0.9916
h: 1 ESS: 3433.997 Accept Rate: 0.9266
h: 3.5 ESS: 7196.557 Accept Rate: 0.5604
h: 10 ESS: 325.0788 Accept Rate: 0.1602
```

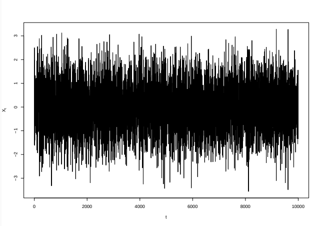


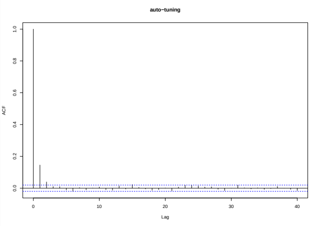






Step Size Tuning: MALA





$$h_{k+1} = h_k + \frac{h_k}{k}(R_k - \hat{\alpha})$$

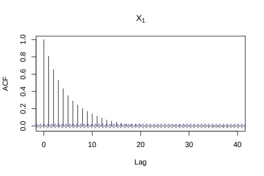
$$\hat{\alpha} \approx 0.576$$

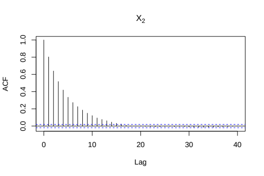
$$\hat{h} \approx 3.3401$$

Figure 9: Starting h = 10, Effective sample size = 7187

Correlation

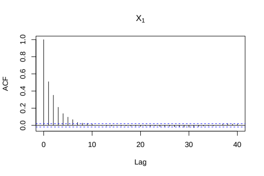
RW

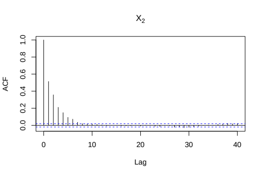




ESS: X1=1054; X2=1077;

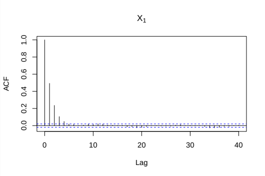
MALA

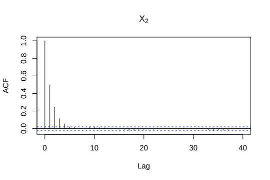




ESS: X1=2530; X2=2481;

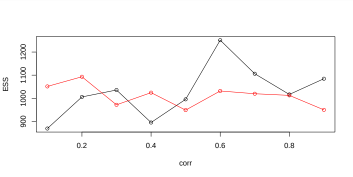
Gibbs

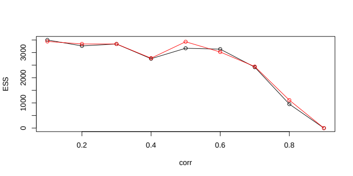


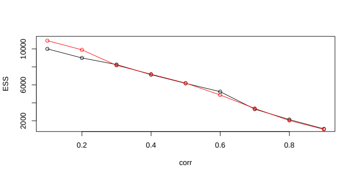


ESS: X1=3396; X2=3344;

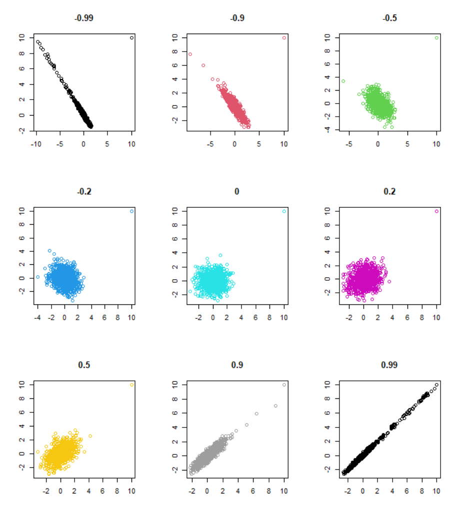
Correlation







Gibbs Sampling with High Correlation



X_1	X_2
24.09	19.48
125.55	113.87
692.36	610.60
1000.00	907.24
1119.07	1000.00
858.51	1000.00
539.68	620.75
109.73	84.94
9.26	9.29

Conclusion

Summary

- Implementation of RWM, MALA, Gibbs Sampler.
- Experimentation with dimensionality, correlation of variables, target distributions, hyper-parameter tuning.
- Further research directions.
    - Hamiltonian Dynamics - HMC, NUTS

Thank you for listening. Questions?