Bayesian Statistics

Fabio Sigrist

ETH Zurich, Autumn Semester 2019

Today's topics

- Adaptive MCMC
- Hamiltonian Monte Carlo

Adaptive MCMC

Random walk Metropolis algorithm

In the **random walk Metropolis (RWM) algorithm**, one generates proposals as $\boldsymbol{Y^t \sim \mathcal{N}(X^{t-1}, \Sigma)}$ where Σ is an arbitrary positive definite covariance matrix.

- In theory, the choice of Σ is irrelevant. For any choice of Σ, we obtain a consistent estimate for $\int h(x)\pi(x)dx$

- In practice, the choice of Σ has a large influence on the quality of the approximation for any finite number N of steps

Clicker question

Optimal choices for Σ

How to choose Σ?

1. For some cases, it has been shown that if π is a p-dimensional distribution with covariance matrix $\mathrm{Cov}_\pi(X)$, then the "optimal" choice of Σ is
$$\Sigma = \frac{2.38^2}{p}\mathrm{Cov}_\pi(X)$$

2. A similar result says that the "optimal" choice is such that the average acceptance rate after the burn-in phase is 0.234, that is
$$\int \pi(x) \underbrace{\int q(x,y)\min\left(1, \frac{\pi(y)}{\pi(x)}\right) dy}_{\text{Acceptance probability for } X^{t-1}=x} dx = 0.234$$

See blackboard

Optimal choices for Σ

- These criteria can be used as **rules of thumb**

- **Problem**: the criteria cannot be used directly because they depend on the unknown target π

- **Possible solution**:
  1. Run an exploration phase where (i) you try out various values of Σ for an ”optimal” acceptance rate or (ii) estimate $\mathrm{Cov}_\pi(X)$ in order to obtain an ”optimal” estimate for Σ

  2. Then, run the algorithm with a fixed Σ determined from the experience gained in the exploration phase

Optimal choices for Σ

Idea of adaptive MCMC: combine the two phases by using a varying Σ^t which depends on the sequence of values $(X^0, X^1, \ldots, X^{t-1})$ generated so far

1. For the first criterion, one can take

$$\Sigma^t = \frac{2.38^2}{p}\frac{1}{t-1}\sum_{s=0}^{t-1}(X^s - \bar{X}^{t-1})(X^s - \bar{X}^{t-1})^T, \quad \bar{X}^{t-1} = \frac{1}{t}\sum_{s=0}^{t-1} X^s$$

2. For the second criterion, we only want to optimize the scale of Σ where the shape is fixed, e.g. $\Sigma = \sigma^2 I_p$. We take

$$\sigma^{2,t} = \begin{cases} r_t \sigma^{2,t-1} & \text{if} \quad \frac{1}{t-1}\sum_{s=0}^{t-2}\min(1, \frac{\pi(Y^{s+1})}{\pi(X^s)}) > 0.234, \\ \frac{1}{r_t}\sigma^{2,t-1} & \text{if} \quad \frac{1}{t-1}\sum_{s=0}^{t-2}\min(1, \frac{\pi(Y^{s+1})}{\pi(X^s)}) < 0.234. \end{cases}$$

Here Y^s is the proposed value in step s and $r_t \downarrow 1$

Hamiltonian Monte Carlo

aka hybrid Monte Carlo

MCMC using Hamiltonian dynamics

- In some situations, algorithms such as the Gibbs sampler or the random walk Metropolis algorithm explore the target density π only slowly

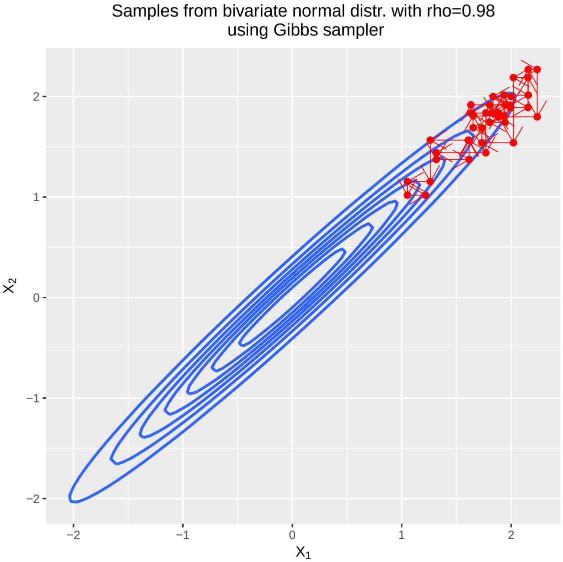
- Illustrative **example**: simulate from a bivariate normal distribution

$$(X_1, X_2) \sim N\left(0, \left(\begin{array}{cc} 1 & \rho \\ \rho & 1 \end{array}\right)\right), \quad -1 < \rho < 1,$$

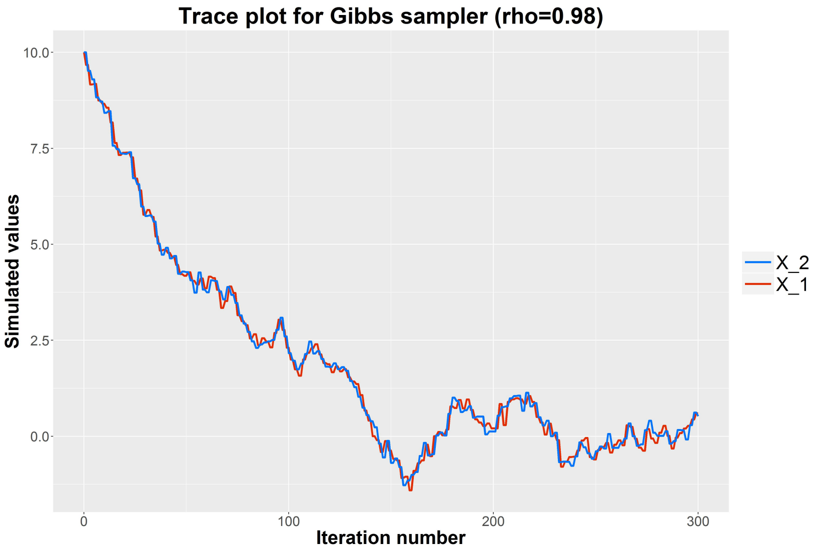
  with high correlation ρ, e.g., $\rho = 0.98$

- The Gibbs sampler makes only small moves and the random walk Metropolis (RWM) algorithm makes either small moves or has a low acceptance probability for proposals with big moves

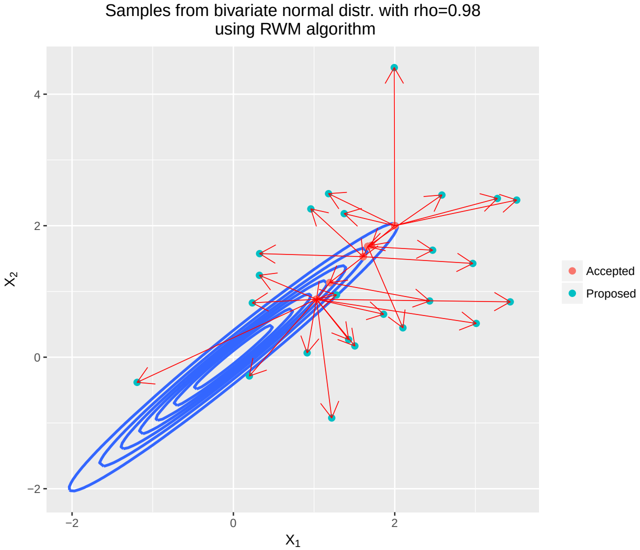
Gibbs sampler for bivariate normal distribution



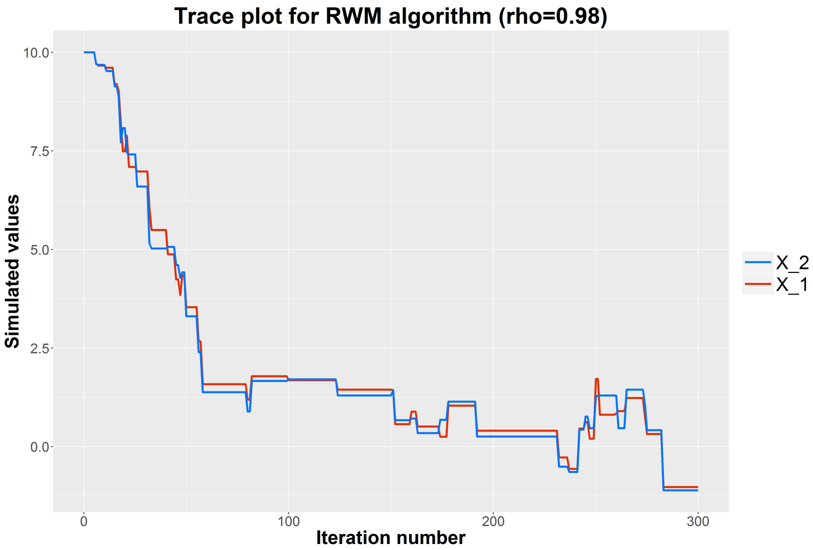
Gibbs sampler for bivariate normal distribution



RWM algorithm for bivariate normal distribution



RWM algorithm for bivariate normal distribution



MCMC using Hamiltonian dynamics

- **Hamiltonian Monte Carlo (HMC)** allows for making big moves that are still accepted with high probability
- **Assumption:**
  - $X \in \mathbb{R}^p$
  - We can evaluate the gradient of $\log \pi$ efficiently

MCMC using Hamiltonian dynamics

- Consider new target $\tilde{\pi}$ on a space with doubled dimension

$$\tilde{\pi}(x, u) \propto \pi(x) \exp\left(-\sum_{i=1}^{p} \frac{u_i^2}{2m_i}\right)$$

- The $\boldsymbol{U_i}$'s can be thought of as **auxiliary variables** that allow the chain to make big moves

- If $(X, U) \sim \tilde{\pi}$, then $X \sim \pi$

- HMC is based on a deterministic, **invertible map $\boldsymbol{G(x, u)}$** that is **volume preserving** and **keeps $\boldsymbol{\tilde{\pi}}$ invariant**

$$\left|\det \frac{\partial G(x, u)}{\partial x \partial u}\right| = 1, \quad \tilde{\pi}(G(x, u)) = \tilde{\pi}(x, u), \forall x, u$$

Construction of map G

Comments

- Invertibility is needed for reversibility
- Volume preservation is needed for a simple form of the Metropolis-Hastings acceptance ratio

Clicker question

Construction of map G

- The construction of G is based on Hamiltonian mechanics
- The **Hamiltonian $\boldsymbol{H(x, u)}$** is defined as

$$H(x, u) = -\log \pi(x) + \sum_{i=1}^{p} \frac{u_i^2}{2m_i}$$

I.e.

$$\tilde{\pi}(x, u) \propto \exp\left(-H(x, u)\right)$$

- **Physical interpretation**
  - x is the position
  - u is the momentum
  - $-\log \pi(x)$ is the potential energy
  - $\sum_{i=1}^{p} \frac{u_i^2}{2m_i}$ the kinetic energy
  - $H(x, u)$ is the total energy in the system

Construction of map G

- The **transformation $\boldsymbol{G(x, u)}$** is given by the solution of the ordinary differential equation (ODE)

  $$\begin{aligned} \frac{dx_i}{dt'} &= \frac{\partial H(x,u)}{\partial u_i} = \frac{u_i}{m_i} \\ \frac{du_i}{dt'} &= -\frac{\partial H(x,u)}{\partial x_i} = \frac{\partial \log \pi(x)}{\partial x_i}, \qquad 0 \leq t' \leq T, \end{aligned}$$

  with initial condition (x, u)

- $G(x, u)$ is volume preserving and keeps $\tilde{\pi}$ invariant

  See blackboard

Discretization of ODEs

- In practice, we need to solve the differential equation by some **discretization method**
- The so-called **leap frog method** induces only small changes to $\tilde{\pi}$, it preserves volume exactly and it is time-reversible, i.e., its implied mapping G is invertible
- The exact **invariance of $\tilde{\pi}$ is restored by a Metropolis-Hastings step at the end** with acceptance ratio

$$a((x,u),(x^*,u^*)) = \min\left(1, \exp\left(-H(x^*,u^*) + H(x,u)\right)\right),$$

where (x^*,u^*) is the newly proposed value and (x,u) the current one

Resampling of momentum variables

- **There is still an issue:** So far, using G to sample from the joint distribution of (X, U) leaves the density $\tilde{\pi}$ unchanged or almost unchanged:

  $$\tilde{\pi}(G(x, u)) = \tilde{\pi}(x, u)$$

  This means that the implied kernel does not sample from the whole space

- **Solution:** First, simulate an independent new component U and then apply the map G
  - If $(X, U) \sim \tilde{\pi}$ and $U' \sim N(0, \text{diag}(m_i))$ is independent of (X, U), then also $(X, U') \sim \tilde{\pi}$
  - This step can be considered as a Gibbs step

HMC algorithm

Algorithm (Hamiltonian Monte Carlo algorithm)

Simulate (X_0, U_0)
For $t = 1, 2, \ldots$

1a. *Simulate* $U' \sim N(0, diag(m_i))$

1b. *Use the leap frog method (or any other method that results into an invertible G that is volume preserving) to generate a proposal* $(X^*, U^*) = G(x_{t-1}, U')$

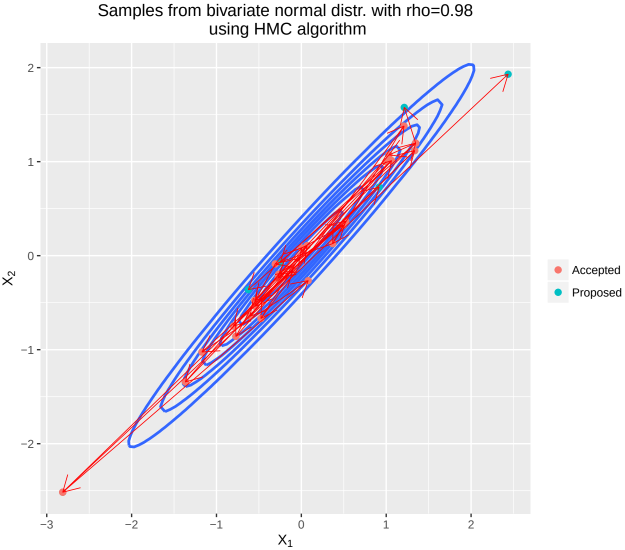
2. *Simulate* $V \sim uniform(0, 1)$*. If* $V \leq a((x_{t-1}, U'), (X^*, U^*))$ *set* $X_t = X^*$*, otherwise* $X_t = x_{t-1}$

Comments on HMC algorithm

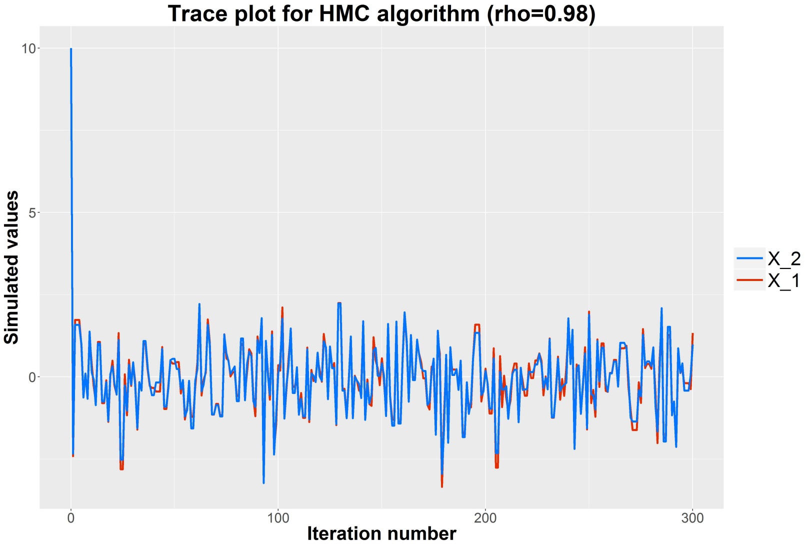
- T, the m_i's, and the step size ε of the discretization in the leap frog method are (important) **tuning parameters**

- **Intuition** on why HMC works: *see blackboard*

HMC algorithm for bivariate normal distribution



HMC algorithm for bivariate normal distribution



Comments on choice of tuning parameters

- If ε **is too large**, then the discretization is inaccurate and one has low acceptance probabilities

  If ε **is too small**, one wastes computational resources

- If ***T*** **is too small**, then the successive samples will be close together which results in undesirable random walk behavior (benefit of HMC is lost)

  If ***T*** **is too large**, one wastes computational resources and the HMC algorithm might produce trajectories that loop back to the initial values (”U-turns”)

HMC extensions

- The so-called **No-U-Turn Sampler (NUTS)** can choose the crucial tuning parameters T and ε automatically. This is implemented in the software Stan

- Instead of assuming $U \sim N(0, \text{diag}(m_i))$, one can assume $U \sim N(0, M)$.

  Riemannian manifold Hamiltonian Monte Carlo (RMHMC) allows for position dependent ”mass matrices” $M(x)$, at the expense of computational cost