Plan for today

- MCMC: what have we learned
- More on MCMC
- Special cases of the MH algorithm
- The Gibbs sampler

MCMC what we have learned:

- Problem: Sample from $\pi(x)$, $x \in S$.
- MCMC idea:
  - Construct Markov chain with $\pi(x)$ as limiting distribution.
  - Simulate the Markov chain for a long time so that it has time to converge.
  - Most MCMC samplers are based on reversible Markov chains $\Rightarrow$ Their convergence is proved by checking the detailed balance equation.

Review: Metropolis-Hastings construction

- $$P(y \mid x) = \begin{cases} Q(y \mid x)\alpha(y \mid x), & y \neq x \\ 1 - \sum_{z \neq x} Q(z \mid x)\alpha(z \mid x), & y = x \end{cases}$$
- $$\alpha(y \mid x) = \min\left\{1, \frac{\pi(y)}{\pi(x)} \cdot \frac{Q(x \mid y)}{Q(y \mid x)}\right\}$$

Review: Metropolis-Hastings algorithm

Init $x_0 \sim g(x_0)$
for $i = 1, 2, \dots$ **do**
  Generate a proposal $y \sim Q(y|x_{i-1})$
  $u \sim \mathrm{U}(0, 1)$
  if $u < \min \underbrace{\left(1, \frac{\pi(y)}{\pi(x_{i-1})} \times \underbrace{\frac{Q(x_{i-1}|y)}{Q(y|x_{i-1})}}_{\text{Proposal ratio}}\right)}_{\text{Acceptance probability } \alpha}$ **then**
    $x_i \leftarrow y$
  else
    $x_i \leftarrow x_{i-1}$
  end if
end for

What about

- Irreducible: Must be checked in each case. Must choose $Q(y \mid x)$ so that this is ok.

What about

- Irreducible: Must be checked in each case. Must choose $Q(y \mid x)$ so that this is ok.
- Aperiodic: Sufficient that $P(x \mid x) > 0$ for one $x \in S$, so sufficient that $\alpha(y \mid x) < 1$ for one pair $y, x \in S$.

What about

- Irreducible: Must be checked in each case. Must choose $Q(y \mid x)$ so that this is ok.
- Aperiodic: Sufficient that $P(x \mid x) > 0$ for one $x \in S$, so sufficient that $\alpha(y \mid x) < 1$ for one pair $y, x \in S$.
- Positive recurrent: for finite S, irreducibility is sufficient. More difficult in general, but if Markov chain is not recurrent we will see this as drift in the simulations. (In practice usually no problem).

What about continuous distributions?

See Notes

Metropolis-Hastings algorithm

Elements of the problem:

- Target distribution $\pi(x)$: Given by the problem
- Proposal distribution $Q(y|x)$: Chosen by the user
- Acceptance probability $\alpha(y|x)$: Derived in order to fullfill the detailed balance condition.

Remarks on the Metropolis-Hastings algorithm

- Under some regularity conditions it can be shown that the Metropolis-Hasting algorithm converges to the target distribution regardless of the specific choice of $Q(y|x)$.

For more comments and details see: Chib, S. and Greenberg, E. (1995), *Understanding the Metropolis-Hastings algorithm, The American Statistician, 49: 327–335*

Remarks on the Metropolis-Hastings algorithm

- Under some regularity conditions it can be shown that the Metropolis-Hasting algorithm converges to the target distribution regardless of the specific choice of $Q(y|x)$.
- However, the speed of convergence and the dependence between the successive samples depends strongly on the proposal distribution.

For more comments and details see: Chib, S. and Greenberg, E. (1995), *Understanding the Metropolis-Hastings algorithm, The American Statistician, 49: 327–335*

Remarks on the Metropolis-Hastings algorithm

- Under some regularity conditions it can be shown that the Metropolis-Hasting algorithm converges to the target distribution regardless of the specific choice of $Q(y|x)$.
- However, the speed of convergence and the dependence between the successive samples depends strongly on the proposal distribution.
- Since we only need to compute the ratio $\pi(y)/\pi(x)$, the proportionality constant is irrelevant.

For more comments and details see: Chib, S. and Greenberg, E. (1995), *Understanding the Metropolis-Hastings algorithm, The American Statistician, 49: 327–335*

Remarks on the Metropolis-Hastings algorithm

- Under some regularity conditions it can be shown that the Metropolis-Hasting algorithm converges to the target distribution regardless of the specific choice of $Q(y|x)$.
- However, the speed of convergence and the dependence between the successive samples depends strongly on the proposal distribution.
- Since we only need to compute the ratio $\pi(y)/\pi(x)$, the proportionality constant is irrelevant.
- Similarly, we only care about $Q(.)$ up to a constant.

For more comments and details see: Chib, S. and Greenberg, E. (1995), *Understanding the Metropolis-Hastings algorithm, The American Statistician, 49: 327–335*

Remarks on the Metropolis-Hastings algorithm

- Under some regularity conditions it can be shown that the Metropolis-Hasting algorithm converges to the target distribution regardless of the specific choice of $Q(y|x)$.
- However, the speed of convergence and the dependence between the successive samples depends strongly on the proposal distribution.
- Since we only need to compute the ratio $\pi(y)/\pi(x)$, the proportionality constant is irrelevant.
- Similarly, we only care about $Q(.)$ up to a constant.
- Often it is advantageous to calculate the acceptance probability on log-scale, which makes the computations more stable.

For more comments and details see: Chib, S. and Greenberg, E. (1995), *Understanding the Metropolis-Hastings algorithm, The American Statistician, 49: 327–335*

Special cases of the Metropolis-Hastings algorithm

Depending on the choice of $Q(x|x_{i-1})$ different special cases result. In particular, two classes are important

- The independence proposal
- The Metropolis algorithm
  - Random walk proposals

Independence proposal

- The proposal distribution does not depend on the current value x_{i-1}

$$Q(x|x_{i-1}) = Q(x).$$

- $Q(x)$ is an approximation to $\pi(x)$
  $\Rightarrow$ Acceptance rate should be close to 1.
- The sampler is closer to rejection sampler. However, here if we reject, then we retain the sample.

Experience:

- Performance is either very good or very bad, usually very bad.
- The tails of the proposal distribution should be at least as heavy as the tails of the target distribution.

The Metropolis algorithm

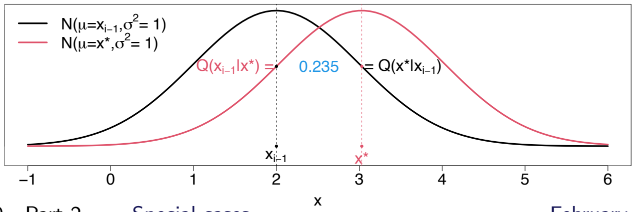
The proposal density is symmetric around the current value, that means

$$Q(x_{i-1}|y) = Q(y|x_{i-1}).$$

Hence,

$$\alpha = \min\left(1, \frac{\pi(y)}{\pi(x_{i-1})} \times \frac{Q(x_{i-1}|y)}{Q(y|x_{i-1})}\right) = \min\left(1, \frac{\pi(y)}{\pi(x_{i-1})}\right)$$

A particular case is the random walk proposal, defined as the current value x_{i-1} plus a random variate of a 0-centred symmetric distribution.



Examples for random walks proposal

Assume x is scalar.

Then all proposal kernels, which add a random variable generated from a zero-symmetrical distribution to the current value x_{i-1}, are random walk proposals. For example:

$$y \sim \mathcal{N}(x_{i-1}, \sigma^2)$$

$$y \sim t_\nu(x_{i-1}, \sigma^2)$$

$$y \sim \mathcal{U}(x_{i-1} - d, x_{i-1} + d)$$

See R-code `demo_mcmcRW_2D.R`.

Efficiency of the Metropolis-Hastings algorithm

The efficiency and performance of the Metropolis-Hastings algorithm depends crucially on the relative frequency of acceptance.

Efficiency of the Metropolis-Hastings algorithm

The efficiency and performance of the Metropolis-Hastings algorithm depends crucially on the relative frequency of acceptance.

An acceptance rate of one is not always good:

- Too large acceptance rate $\Rightarrow$ slow target density exploration
- Too small acceptance rate $\Rightarrow$ large moves proposed, but rarely accepted

Efficiency of the Metropolis-Hastings algorithm

The efficiency and performance of the Metropolis-Hastings algorithm depends crucially on the relative frequency of acceptance.

An acceptance rate of one is not always good:

- Too large acceptance rate $\Rightarrow$ slow target density exploration
- Too small acceptance rate $\Rightarrow$ large moves proposed, but rarely accepted

Tuning the acceptance rate:

- For random walk proposals, acceptance rates between 20% and 50% are typically recommended. They can be achieved by changing the variance of the proposal distribution.
- For independence proposals a high acceptance rate is desired, which means the proposal density is close to the target density.

Example: Random walk proposal

Exploration of a standard Gaussian distribution ($\mathcal{N}(0,1)$) using a random walk Metropolis algorithm. As proposal assume a Gaussian distribution with variance σ^2, where.

- $\sigma = 0.24$
- $\sigma = 2.4$
- $\sigma = 24$

See R-code `demo_mcmcRW.R`.

Example of Rao (1973)

The vector $\boldsymbol{y} = (y_1, y_2, y_3, y_4) = (125, 18, 20, 34)$ is multinomial distributed with probabilities

$$\left\{\frac{1}{2} + \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right\}$$

We would like to simulate from the posterior distribution (assuming a uniform prior)

$$f(\theta|\boldsymbol{y}) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4}.$$

using MCMC and compare two proposal kernels:

1. independence proposal
2. random walk proposal

See R-code `demo_mcmcRao.R`.

Rao: Independence proposal

$$\theta^\star \sim \mathcal{N}(\mathrm{Mod}(\theta|\boldsymbol{y}), F^2 \times I_p^{-1}), \tag{5}$$

where $\mathrm{Mod}(\theta|\text{data})$ denotes the posterior mode, I_p the negative curvature of the log posterior at the mode, and F a factor to blow up the standard deviation.

Rao: Random walk proposal

$$\theta^\star \sim \mathrm{U}(\theta^{(k)} - d, \theta^{(k)} + d),$$

where $\theta^{(k)}$ denotes the current state of the Markov chain and $d = \sqrt{12}/2 \cdot 0.1$.

Comments on the Metropolis-Hasting algorithm

- A trivial special case results when

$$Q(x^\star | x_{i-1}) = \pi(x^\star),$$

  That means, we propose realisations from the target distribution. Then $\alpha = 1$ and all proposals are accepted.
- The advantage of the MH-algorithm is that arbitrary proposal kernels can be used. The algorithm will always converge to the target distribution.
- However, the speed of convergence and the dependence between the successive samples depends strongly on the proposal distribution.

Numerical Note

How to compute

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

Naive strategy: Compute $\pi(y)$, $\pi(x)$, $Q(y|x)$, $Q(x|y)$. Then compute the ratio.

Numerical Note

How to compute

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

Naive strategy: Compute $\pi(y)$, $\pi(x)$, $Q(y|x)$, $Q(x|y)$. Then compute the ratio.
Solution:

- Simplify the expression as much as possible
- Compute everything in log-scale

MCMC and iterative conditioning

MH-algorithms are sometimes applied iteratively on components of x.

Let $\boldsymbol{x}$ be decomposed by several (for simplicity scalar) components.

$$\boldsymbol{x} = (x^1, \ldots, x^p)$$

Now the MH-algorithm is applied iteratively on the components x^j, conditioning on the current values of $\boldsymbol{x}^{-j}$ with

$$\boldsymbol{x}^{-j} = (x^1, \ldots, x^{j-1}, x^{j+1}, \ldots, x^p)$$

MCMC and iterative conditioning

To be concrete, one uses

- a proposal kernel $Q(y^j | x^j_{i-1}, \boldsymbol{x}^{-j}_{i-1})$, $j = 1, \ldots, p$.
- with acceptance probability

$$\alpha = \min\left(1, \frac{\pi(y^j | \boldsymbol{x}^{-j}_{i-1})}{\pi(x^j_{i-1} | \boldsymbol{x}^{-j}_{i-1})} \times \frac{Q(x^j_{i-1} | y^j, \boldsymbol{x}^{-j}_{i-1})}{Q(y^j | x^j_{i-1}, \boldsymbol{x}^{-j}_{i-1})}\right)$$

This algorithm converges to the stationary distribution with density $\pi(\boldsymbol{x})$, as long as all components are updated arbitrarily often.

Iterative conditioning: Conditional densities

In this case, the acceptance probability α only uses the full conditional densities $\pi(x^j | \boldsymbol{x}^{-j})$, $j = 1, \ldots, p$, and not the joint density $\pi(\boldsymbol{x})$.
Both are related as follows

$$\pi(x^j | \boldsymbol{x}^{-j}) = \frac{\pi(\boldsymbol{x})}{\pi(\boldsymbol{x}^{-j})} \propto \pi(\boldsymbol{x})$$

Thus, the (non-normalised) conditional densities of $x^j | \boldsymbol{x}^{-j}$ can be directly derived from $\pi(\boldsymbol{x})$ by omitting all multiplicative factors, that do not depend on x^j.

Gibbs sampling

It seems natural to use the conditional densities as proposal kernels, i.e.

$$Q(y^j | x^j_{i-1}, \boldsymbol{x}^{-j}_{i-1}) = \pi(x^j | \boldsymbol{x}^{-j}_{i-1}).$$

In this case, we get $\alpha = 1$, which leads to the well known Gibbs sampler. Gibbs sampling updates parameters iteratively by sampling from the corresponding full conditional distributions.

Gibbs sampling

Let $x = (x^1, \dots, x^p)$, $x \sim \pi(x)$, p proposal distributions are defined by:

- propose $y^j \sim \pi(y^j | x^{-j})$
- keep $y^k = x^k$ for $k \neq j$

Notation:

- $x = (x^1, \dots, x^p)$
- $x^{-j} = (x^1, \dots, x^{j-1}, x^{j+1}, \dots, x^p)$
- $y = (y^1, \dots, y^p) = (x^1, \dots, x^{j-1}, y^j, x^{j+1}, \dots, x^p)$

What is the acceptance probability for the Gibbs sampling?

Gibbs-Sampling algorithm

Idea: Sequentially sample from univariate conditional distributions

1. Select starting values $\boldsymbol{x}_0$ and set $i = 0$.
2. Repeatedly:

$$
\begin{aligned}
&\text{Sample} \quad x_{i+1}^1 | \cdot \sim \pi(x^1 | x_i^2, \dots, x_i^p) \\
&\text{Sample} \quad x_{i+1}^2 | \cdot \sim \pi(x^2 | x_{i+1}^1, x_i^3, \dots, x_i^p) \\
&\vdots \\
&\text{Sample} \quad x_{i+1}^{p-1} | \cdot \sim \pi(x^{p-1} | x_{i+1}^1, x_{i+1}^2, \dots, x_{i+1}^{p-2}, x_i^p) \\
&\text{Sample} \quad x_{i+1}^p | \cdot \sim \pi(x^p | x_{i+1}^1, \dots, x_{i+1}^{p-1})
\end{aligned}
$$

   where $|\cdot$ denotes conditioning on the most recent updates of all other elements of $\boldsymbol{x}$.

3. Increment i and go to step 2.