LECTURE 16: MARKOV CHAIN MONTE CARLO (CONTD)

STAT 545: Intro. to Computational Statistics

Vinayak Rao

Purdue University

November 14, 2016

Markov chain Monte Carlo

Obtain dependent samples $(x_1, x_2, \ldots)$ by running an irreducible, aperiodic, positive recurrent Markov chain on $\mathcal{X}$.

Ensure right stationary distribut. via detailed balance:

$$\pi(x_{n+1})\mathcal{K}(x_n, x_{n+1}) = \pi(x_n)\mathcal{K}(x_{n+1}, x_n)$$

Means that $p(x_n = a, x_{n+1} = b) = p(x_n = b, x_{n+1} = a)$ for any n, a, b.

Markov chain Monte Carlo

We saw two simple MCMC algorithms

An iteration of the Metropolis-Hastings algorithm:

- Propose a new point x^* according to $q(x^*|x_n)$.
- Accept with prob. $\alpha = \min\left(1, \frac{\pi(x^*)q(x_n|x^*)}{\pi(x_n)q(x^*|x_n)}\right)$

We saw two simple MCMC algorithms

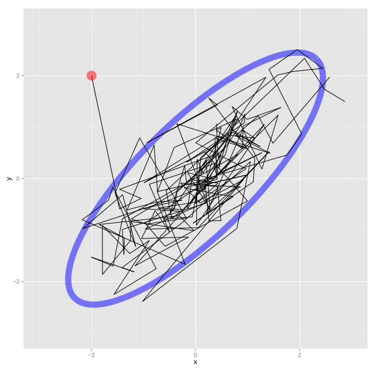
An iteration of the Metropolis-Hastings algorithm:

- Propose a new point x^* according to $q(x^*|x_n)$.
- Accept with prob. $\alpha = \min\left(1, \frac{\pi(x^*)q(x_n|x^*)}{\pi(x_n)q(x^*|x_n)}\right)$

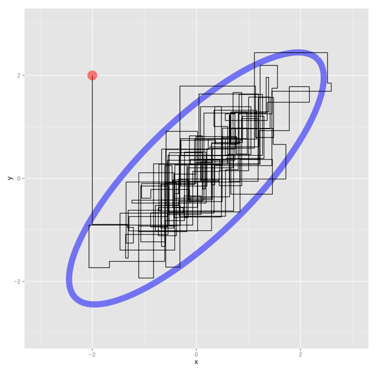
An iteration of the Gibbs sampling algorithm:

- Let the state at iteration n be $\mathbf{x}_n = (x_n^1, x_n^2, \ldots, x_n^d)$.
- Set $\mathbf{y} = \mathbf{x}_n$.
- Update $\mathbf{y}$ by resampling $y^i \sim \pi(\cdot|\mathbf{y}^{\neg i}) \quad \text{for } i \in \{1, \ldots, d\}$
- Set $= \mathbf{x}_{n+1} = \mathbf{y}$.

MCMC



Metropolis-Hastings



Gibbs

$\mathcal{K}_i$: kernel that samples component i from its conditional distrib.

$$\mathcal{K}_i(\mathsf{x}_{n+1}, \mathsf{x}_n) = \begin{cases} 0, & \text{if } \mathsf{x}^j_{n+1} \neq \mathsf{x}^j_n \text{ for any } j \neq i, \\ \pi(\mathsf{x}^i_{n+1} | \mathsf{x}^{\neg i}_n), & \text{otherwise} \end{cases}$$

$\mathcal{K}_i$: kernel that samples component i from its conditional distrib.

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } \mathbf{x}^j_{n+1} \neq \mathbf{x}^j_n \text{ for any } j \neq i, \\ \pi(\mathbf{x}^i_{n+1} | \mathbf{x}^{\neg i}_n), & \text{otherwise} \end{cases}$$

Clearly $\mathcal{K}_i$ has π as its stationary distribution:
If $\mathbf{x}_n \sim \pi$, then $\mathbf{x}_{n+1} \sim \pi$.

A few points on the Gibbs sampler

$\mathcal{K}_i$: kernel that samples component i from its conditional distrib.

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } \mathbf{x}^j_{n+1} \neq \mathbf{x}^j_n \text{ for any } j \neq i, \\ \pi(\mathbf{x}^i_{n+1} | \mathbf{x}^{\neg i}_n), & \text{otherwise} \end{cases}$$

Clearly $\mathcal{K}_i$ has π as its stationary distribution:
If $\mathbf{x}_n \sim \pi$, then $\mathbf{x}_{n+1} \sim \pi$.
However, it is not irreducible. For this, we must either:

- Cycle through components: $\mathcal{K} = \mathcal{K}_1 \circ \mathcal{K}_2 \circ \cdots \circ \mathcal{K}_d$ (Composition of kernels)
- Randomly pick components: $\mathcal{K} = \nu_1 \mathcal{K}_1 + \nu_2 \mathcal{K}_2 + \cdots + \nu_d \mathcal{K}_d$ (Mixture of kernels)

In general, composing or mixing kernels with π as the stationary distribution gives a new kernel with π as stationary.

You should be able to prove this starting with two kernels.

New Markov kernels from old

In general, composing or mixing kernels with π as the stationary distribution gives a new kernel with π as stationary.

You should be able to prove this starting with two kernels.

Thus can alternate between or randomly perform MH and Gibbs.
Can cycle randomly or deterministically between MH kernels with different proposal distributions

New Markov kernels from old

In general, composing or mixing kernels with π as the stationary distribution gives a new kernel with π as stationary.

You should be able to prove this starting with two kernels.

Thus can alternate between or randomly perform MH and Gibbs.
Can cycle randomly or deterministically between MH kernels with different proposal distributions

Careful though:
Should not depend on some statistic of all earlier samples

In general, composing or mixing kernels with π as the stationary distribution gives a new kernel with π as stationary.

You should be able to prove this starting with two kernels.

Thus can alternate between or randomly perform MH and Gibbs.
Can cycle randomly or deterministically between MH kernels with different proposal distributions

Careful though:
Should not depend on some statistic of all earlier samples

Metropolis-within-Gibbs

Sometimes Gibbs sampling from the conditionals isn't easy:

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } x^j_{n+1} \neq x^j_n \text{ for any } j \neq i, \\ \pi(x^i_{n+1} | \mathbf{x}^{\neg i}), & \text{otherwise} \end{cases}$$

Metropolis-within-Gibbs

Sometimes Gibbs sampling from the conditionals isn't easy:

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } x_{n+1}^j \neq x_n^j \text{ for any } j \neq i, \\ \pi(x_{n+1}^i | \mathbf{x}^{\neg i}), & \text{otherwise} \end{cases}$$

Use a 'sub-kernel' that has $\pi(\mathbf{x}_{n+1}^i | \mathbf{x}^{\neg i})$ as it's invariant distribution:

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } x_{n+1}^j \neq x_n^j \text{ for any } j \neq i, \\ k_i(x_{n+1}^i | \mathbf{x}_n), & \text{otherwise} \end{cases}$$

Metropolis-within-Gibbs

Sometimes Gibbs sampling from the conditionals isn't easy:

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } x^j_{n+1} \neq x^j_n \text{ for any } j \neq i, \\ \pi(x^i_{n+1} | \mathbf{x}^{\neg i}), & \text{otherwise} \end{cases}$$

Use a 'sub-kernel' that has $\pi(\mathbf{x}^i_{n+1} | \mathbf{x}^{\neg i})$ as it's invariant distribution:

$$\mathcal{K}_i(\mathbf{x}_{n+1}, \mathbf{x}_n) = \begin{cases} 0, & \text{if } x^j_{n+1} \neq x^j_n \text{ for any } j \neq i, \\ k_i(x^i_{n+1} | \mathbf{x}_n), & \text{otherwise} \end{cases}$$

Common to define $k_i(x^i_{n+1} | \mathbf{x}_n)$ via a MH-step:

Propose $x^i_{n+1} \sim q(\cdot | \mathbf{x}_n) = q(\cdot | \{\mathbf{x}^{\neg i}, x^i_n\})$

Accept with prob. $\min\left(1, \frac{\pi(\mathbf{x}^i_{n+1} | \mathbf{x}^{\neg i}) q(x^i_n | \{\mathbf{x}^{\neg i}, x^i_{n+1}\})}{\pi(\mathbf{x}^i_n | \mathbf{x}^{\neg i}) q(x^i_{n+1} | \{\mathbf{x}^{\neg i}, x^i_n\})}\right)$

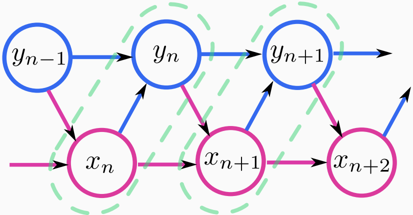
Data augmentation schemes

Data augmentation:

Introduce auxiliary variables we do not care about.

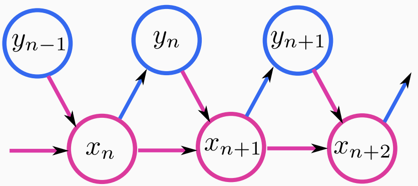
Data augmentation:
Introduce auxiliary variables we do not care about.

If we want to run a Markov chain on $\mathcal{X}$.
Introduce auxiliary variables $y \in \mathcal{Y}$ and run chain on $\mathcal{X} \times \mathcal{Y}$.



Data augmentation:
Introduce auxiliary variables we do not care about.

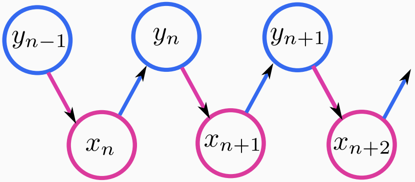
If we want to run a Markov chain on $\mathcal{X}$.
Introduce auxiliary variables $y \in \mathcal{Y}$ and run chain on $\mathcal{X} \times \mathcal{Y}$.

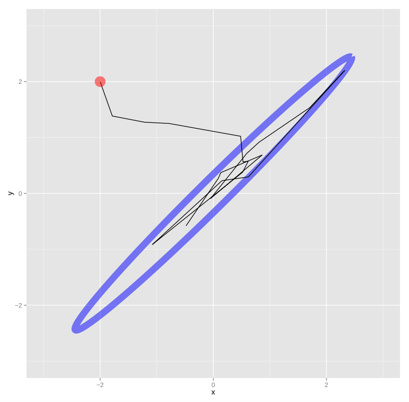


Data augmentation schemes

Data augmentation:
Introduce auxiliary variables we do not care about.

If we want to run a Markov chain on $\mathcal{X}$.
Introduce auxiliary variables $y \in \mathcal{Y}$ and run chain on $\mathcal{X} \times \mathcal{Y}$.



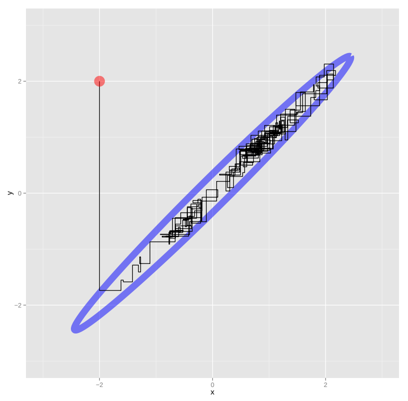


MH is a random walk on $\mathcal{X}$.

Make 'blind proposals' and evaluate with π.

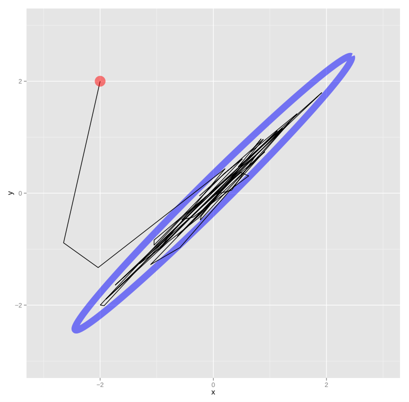
N steps takes you $\sqrt{N}$-distance.

MCMC



Gibbs sampler explores $\mathcal{X}$-space one component at a time.

MCMC



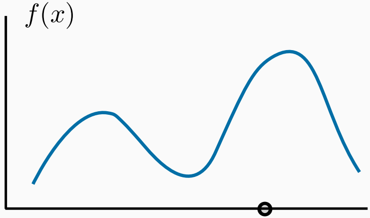
Hamiltonian Monte Carlo: Exploit gradient-information about $\pi(x) = f(x)/Z$ to explore $\mathcal{X}$ more efficiently

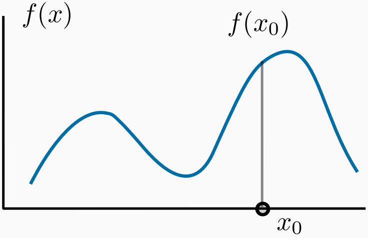
Slice sampling

Uniformly sampling below $f(x)$ gives samples from $\pi(x) \propto f(x)$.

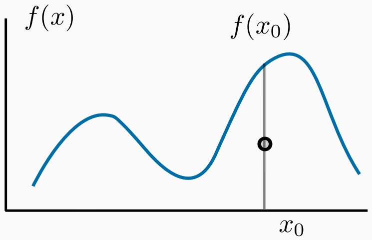
Slice sampling augments the x with a height y.

Runs a Markov chain by updating (x, y).

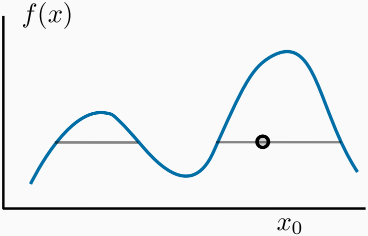




Sample $y_0 \sim P(\cdot|x_0) = \mathrm{Unif}(0, f(x_0))$


$f(x)$
$f(x_0)$
x_0

Slice sampling

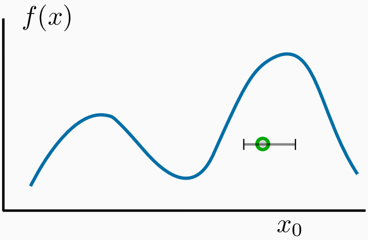


Given y_0, sample

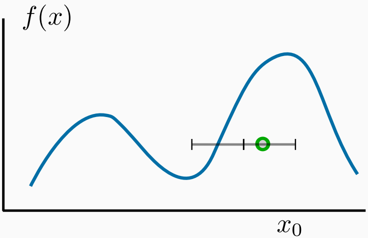
$$x_1 \sim p(\cdot|y_0) = \mathrm{Unif}(L_{y_0}), \text{where } L_x = \{x : f(x) > y_0\})$$

Not easy.

Make conditional updates: $x_1 \sim p(\cdot|x_0, y_0)$

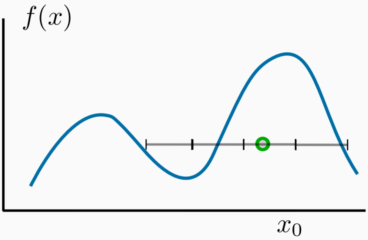


Randomly locate a window R around x_0; pick $x_1 \sim \text{Unif}(R \cap L_y)$.
Observe: if x_0 (green), is $\text{Unif}(y_0)$, then so is x_1 (red). Randomly locating window ensures reversibility.
NOT irreducible: can't jump between modes with long 0-prob



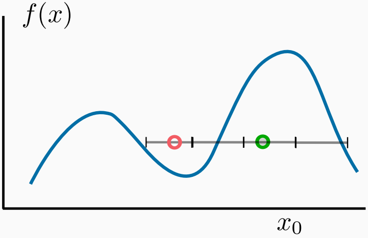
Fix: grow window until both ends greater than y_0. Different possible strategies. A simple one:

- Let current window length be L.
- Double the length by adding $L/2$ segments to *random* sides.
- Repeat till both ends lie above f.

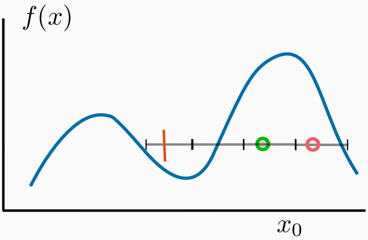


Asymmetric growth allows us to reach any mode with nonzero probability.

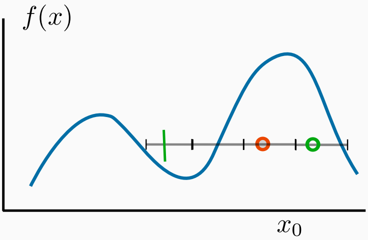
Slice sampling



Now, uniformly pick points inside window. If point is invalid, truncate window and repeat.



When we obtain a valid point, the preceding window can be viewed as a random window with a bunch of auxiliary variables.



Note the overall process is reversible