Lecture 4:
Monte Carlo methods

Monte Carlo

- **Monte Carlo methods** (or **Monte Carlo experiments**) are a class of computational algorithms that rely on repeated random sampling to compute their results

- Monte Carlo methods for random number generation
  - Metropolis-Hastings algorithm
  - Gibbs sampler

- Monte Carlo methods in statistical inference
  - Estimating Integrals
  - Variance estimation
  - Variance reduction
    - Importance sampling
    - Control variates

Markov Chain Monte Carlo (MCMC)

- Motivation: We already know methods to generate
  - univariate distributions (inverse CDF, acceptance/rejection)
  - multivariate normal

but what about general multivariate distribution?

MCMC allows to do that!

Short about bayesian inference

Suppose that a dataset *D* was obtained by sampling from some unknown distribution $f(x|\vartheta)$. How to find ϑ?

- Frequentists : ϑ is unknown parameter, compose likelihood $p(D|\vartheta)$ function, find maximum → get ϑ

- Bayesians: ϑ is a random variable, it has prior probability $p(\vartheta)$ (which reflects our guesses about possible ϑ and their probabilities, before data collected)
- After data is collected, the Bayes' theorem says:

$$p(\theta \mid D) = \frac{p(D \mid \theta)\, p(\theta)}{p(D)} = \frac{p(D \mid \theta)\, p(\theta)}{\int p(D \mid \theta)\, p(\theta) d\theta}$$

Short about bayesian inference

Task: knowing $p(D|\vartheta)$ and $p(\vartheta)$, generate random samples from $p(\vartheta|D)$.

Problems:

1. As previously, it is multivariate distribution of general type
2. Integral is often difficult or impossible to compute

MCMC:

1. First problem is solved
2. Integral computation will not be needed in MCMC

MCMC: Example

- **Linear regression** with an error term distributed in some way (normally, student ,…). Assume normal distribution for ε

$$Y = \beta X + \varepsilon$$

- How to find credible interval for β if you know σ?
  - $P(Y|X,\beta\ ) = \prod_{i=1}^{N} f_N(Y_i|\mu = \beta X_i, \sigma = \sigma)$
  - To get $P(\beta|Y,X\ )$, sample with MCMC by using $P(Y|X,\beta\ )$ if you do not have any prior knowledge about β, otherwise include prior
  - Use MCMC sample to compute quantiles

Note: In the case of normal distribution, the interval can be computed analytically.

Markov chains

- Markov chain is a sequence $\mathbf{X_0}$, $\mathbf{X_1}$,… of random variables such that next value depends only on the previous one

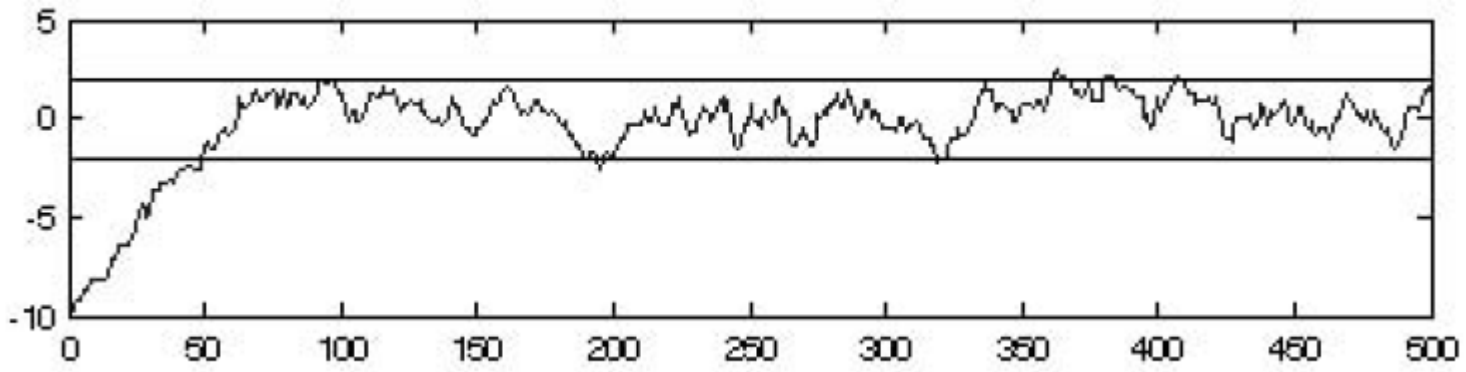
- $P(X_{t+1} \mid X_t)$ is called transition kernel, assume it does not depend on *t*

Theorem Under certain conditions, Markov chain will converge to the stationary distribution ϕ (not sensitive to $\mathbf{X_0}$), i.e.

$$all\ X_i \in \phi \quad for\ i = k, k+1, \ldots$$

First *k-1* samples are normally discarded, they are called **burn-in period**

Example: univariate X



Metropolis-Hastings algorithm

Given:

- PDF $\pi(\mathbf{x})$ that we need to obtain samples from
- *Proposal distribution* $q(. \mid X_t)$ – it may have almost any, but *regular* form
  - Ex: $q(. \mid X_t)$ is a normal distr. with mean X_t and fixed cov. matr.

"Regular form":

- It's enough that the proposal distr. has the same support with nonzero density as π

Metropolis-Hastings algorithm

PROCEDURE - METROPOLIS-HASTINGS SAMPLER

1. Initialize the chain to $\boldsymbol{X}_0$ and set $t = 0$.
2. Generate a candidate point $\boldsymbol{Y}$ from $q(.|\boldsymbol{X}_t)$.
3. Generate U from a uniform $(0, 1)$ distribution.
4. If $U \leq \alpha(\boldsymbol{X}_t, Y)$ (Equation α) then set $\boldsymbol{X}_{t+1} = \boldsymbol{Y}$, else set $\boldsymbol{X}_{t+1} = \boldsymbol{X}_t$.
5. Set $t = t + 1$ and repeat steps 2 through 5.

- Equation α:

$$\alpha(\boldsymbol{X}_t, \boldsymbol{Y}) = \min\left\{1, \frac{\pi(\boldsymbol{Y})q(\boldsymbol{X}_t|\boldsymbol{Y})}{\pi(\boldsymbol{X}_t)q(\boldsymbol{Y}|\boldsymbol{X}_t)}\right\}$$

Metropolis-Hastings algorithm

Comments:

- The chain will converge to π(**x**)
- In Bayesian inference the integral term will be cancelled, look the formula for α
- Observe, that in some cases the chain does not move
- The variables in the obtained sample are dependent

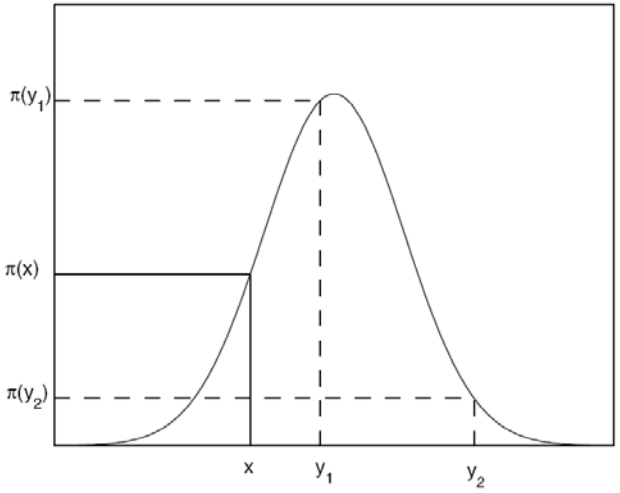
- If $q(Y \mid X) = q(\|X - Y\|)$, the formula transforms to *Random-walk Monte Carlo*

$$\alpha(X_t, Y) = \min\left\{1, \frac{\pi(Y)}{\pi(X_t)}\right\}$$

Choice of proposal distribution

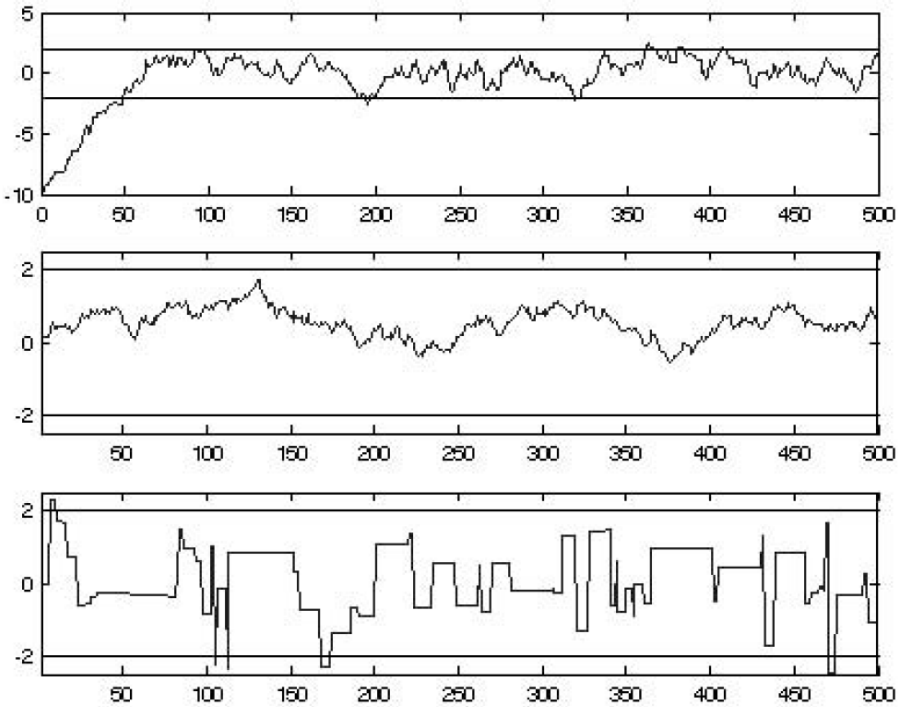
- In Random-Walk Monte Carlo,

If $\pi(Y) \geq \pi(X_t)$, the chain moves to next point, otherwise moved with some probability.



Choice of proposal distribution

- Proposal density should be selected with care!
- Example: q is normal with σ=0.5 0.1 10



Gibbs sampler

- Another way to generate multivariate random numbers
- Uses conditional distributions → need random number generators that sample from univariate distributions

PROCEDURE - GIBBS SAMPLER

1. Generate a starting point $\mathbf{X}_0 = (X_{0,1}, \ldots, X_{0,d})$. Set $t = 0$.
2. Generate a point $X_{t,1}$ from

$$f(X_{t,1} | X_{t,2} = x_{t,2}, \ldots, X_{t,d} = x_{t,d}).$$

Generate a point $X_{t,2}$ from

$$f(X_{t,2} | X_{t+1,1} = x_{t+1,1}, X_{t,3} = x_{t,3}, \ldots, X_{t,d} = x_{t,d}).$$

. . .

Generate a point $X_{t,d}$ from

$$f(X_{t,d} | X_{t+1,1} = x_{t+1,1}, \ldots, X_{t+1,d-1} = x_{t+1,d-1}).$$

3. Set $t = t + 1$ and repeat steps 2 through 3.

Gibbs sampler

- At each iteration of step 2
  - the random numbers are generated from univariate distr. (since *d-1* parameters are fixed)
  - Only one component of X_t is updated
- The convergence can be slow
- However, very useful in high dimensions compared to Metropolis-Hastings
- Also useful when X has more limited domain than proposal distribution, for ex. $X_i > 0$ but $Y_i \sim Normal(.,.)$

Example

- Use Gibbs to generate from $N\left(\mu = \begin{pmatrix}1\\2\end{pmatrix}, \Sigma = \begin{pmatrix}1 & 0.5\\0.5 & 1\end{pmatrix}\right)$

Gibbs sampler

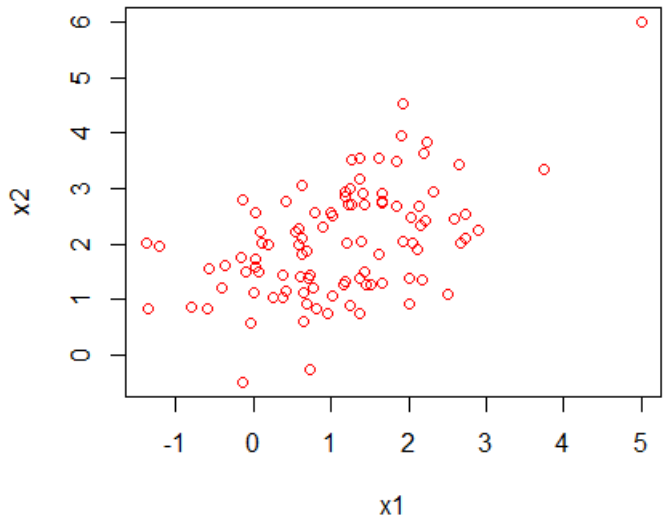
- Theoretical result:

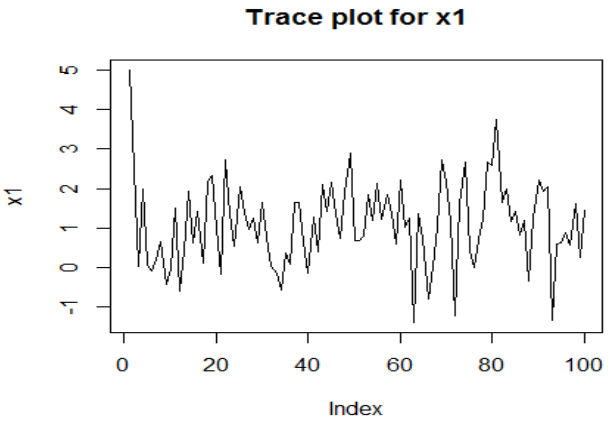
$$X_1 \mid X_2 = x_2 \sim \mathcal{N}\left(\mu_1 + \frac{\sigma_1}{\sigma_2}\rho(x_2 - \mu_2), (1-\rho^2)\sigma_1^2\right).$$

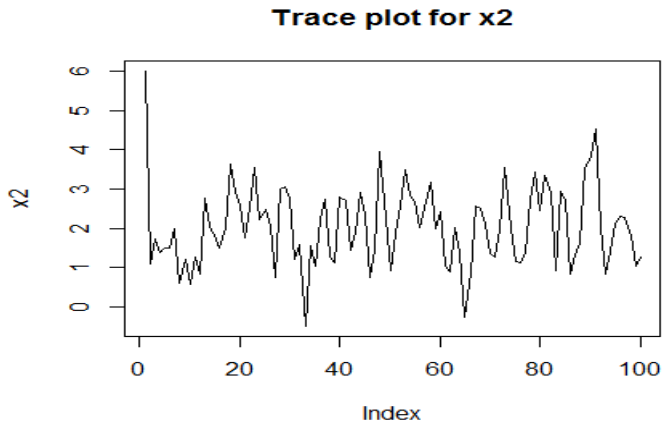
- Possible Gibbs sampler:
  1. Set $t = 0$
  2. Starting point $X = (x_{t,1} = 5, x_{t,2} = 6)$
  3. Generate $x_{t+1,1} \sim N(1 + \frac{1}{1}0.5(x_{t,2} - 2), (1 - 0.25) \cdot 1)$
  4. Generate $x_{t+1,2} \sim N(2 + \frac{1}{1}0.5(x_{t+1,1} - 1), (1 - 0.25) \cdot 1)$
  5. Set $t = t + 1$ and go to step 3 until *n* samples are obtained

Gibbs sampler

- Results, n=100







Convergence monitoring

- When should we stop the chain, i.e. when the convergence to the target distr. is attained?

- Typically, a sample is generated to make further inference (mean, quantiles etc)

Convergence monitoring

Gelman-Rubin method

Assume, we estimate *v(**X**)*

- Generate *k* sequences of length *n* with different starting points
- Compute between- and within- sequence variances:

$$B = \frac{n}{k-1}\sum_{i=1}^{k}(\bar{v}_{i.} - \bar{v}_{..})^2 \qquad W = \frac{1}{k}\sum_{i=1}^{k} s_i^2 \qquad s_i^2 = \frac{1}{n-1}\sum_{j=1}^{n}(v_{ij} - \bar{v}_{i.})^2$$

- Compute overall variance estimate:

$$\hat{\mathrm{var}}(v) = \frac{n-1}{n}W + \frac{1}{n}B$$

- Compute Gelman-Rubin factor

$$\sqrt{\hat{R}} = \sqrt{\frac{\hat{\mathrm{var}}(v)}{W}}$$

- If the factor is close to 1, i.e. around 1.0 - 1.2, then the convergence is achieved

Convergence monitoring

- Code
  - Assume that we have *n* chains as vectors matrix X (X[,1],..X[,n])

```
library(coda)
f=mcmc.list()
for (i in 1:n) f[[i]]=as.mcmc(X[,i])
gelman.diag(f)
```

```
> gelman.diag(f)
Potential scale reduction factors:

     Point est. Upper C.I.
[1,]          1          1
```

Should be close to 1!

MC for inference

Estimation of definite integral

$$\theta = \int_D f(x)\mathrm{d}x$$

If can not estimate in closed form,

- Decompose into

$$f(x) = g(x)p(x) \quad \text{such that} \quad \int_D p(x)\mathrm{d}x = 1$$

- Then, $\theta = \mathrm{E}(g(X)) = \int_D g(x)p(x)\mathrm{d}x$

- Estimator $\widehat{\theta} = \frac{\sum g(x_i)}{m}$

MC for inference

Estimation of definite integral

1. Decompose the function $f(x) = g(x)p(x)$ where $g(x)$ is some pdf.
2. Simulate sample $x_1,\ldots x_m$ from *p(x)*
3. Estimate integral as

$$\widehat{\theta} = \frac{\sum g(x_i)}{m}$$

Comments

- The estimated integral depends on *m* and *g* → how uncertain is it? → a variance estimate would be needed
- Decomposition is not unique → one decomposition is more useful than another
- You need to be able to generate from *p(x)*.
- In Bayesian inference, use MCMC samples from $p(\theta|D)$ to compute point estimators (posterior mean) $\theta^* = \int \theta p(\theta|D) \approx \frac{1}{m}\sum\theta_i$

Variance estimation

- Variance of integral estimation

$$\widehat{\mathrm{V}}(\widehat{\theta}) = \frac{\sum \left(g(x_i) - \overline{g(x)} \right)^2}{m(m-1)}$$

- However, since $\mathbf{x}_i$ are correlated in MCMC, this estimator is biased

- Instead, take longer chain and use batch means instead of $\mathbf{x}_i$

Importance sampling

- Which importance function $p(x)$ would reduce the variance of the integral mostly?

Theorem $p(x) \propto |f(x)|$ gives the lowest variance of the estimated integral

Control variates

- Another way to reduce the variance of the integral
- Idea
  - Assume that two random variables $\hat{\theta}\ and\ \hat{\mu}$ are correlated, i.e. $cov(\hat{\theta}, \hat{\mu}) \neq 0$, and $E\hat{\mu} = M$
  - Then estimator $\theta^* = \hat{\theta} + c(\hat{\mu} - M)$ reduces the variance without influencing the mean, i.e. $E\theta^* = E\hat{\theta}$ and $Var(\theta^*) < Var(\hat{\theta})$
  - The optimal $c = -\frac{cov(\hat{\theta},\hat{\mu})}{var(\hat{\mu})}$

Control variates

- How to use in integral estimation:
  - Need to estimate $\theta = \int g(x)p(x)dx$
  - Assume, we can estimate analytically $M = \int h(x)p(x)dx$
  - Sample x_i from pdf $p(x)$ and consider random vars: $\hat{\theta} = \frac{1}{m}\sum g(x_i)$, $\hat{\mu} = \frac{1}{m}\sum h(x_i)$
  - Use $\hat{\theta}$ and $\hat{\mu}$ and M to estimate:

$$\theta^* = \hat{\theta} + c(\hat{\mu} - M)$$

  - Estimate *c* by using sample variances and sample covariances of $h(x_i)$, $g(x_i)$

Control variates

- Example
  - Estimate $\mathrm{I} = \int_0^1 \frac{1}{x+1} dx$
  - Use $g(x) = \frac{1}{x+1}, p(x) = 1$ (uniform distribution)
  - Use $h(x) = x + 1.$
  - $M = \int h(x)p(x)dx = \int_0^1 (x+1)dx = 1.5$
- Algorithm
  - Generate $x_1, \ldots x_n$ from $U[0,1]$
  - Set $c \approx 0.477$
  - Estimate $I \approx \frac{1}{n}\sum_1^n \frac{1}{x_i+1} - c\left(\frac{1}{n}\sum_1^n (x_i+1) - 1.5\right)$

Source: Wikipedia

	Estimate	Variance
Classical estimate	0.69475	0.01947
Control variates	0.69295	0.00060