Monte Carlo Methods

732A90
Computational Statistics

Krzysztof Bartoszek
(krzysztof.bartoszek@liu.se)

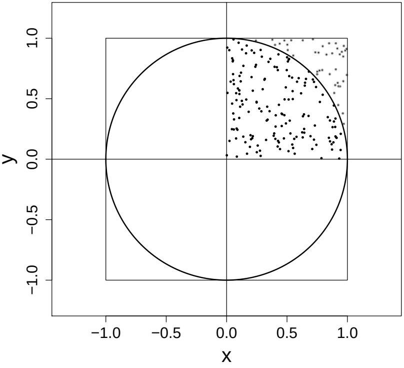
8 II 2017

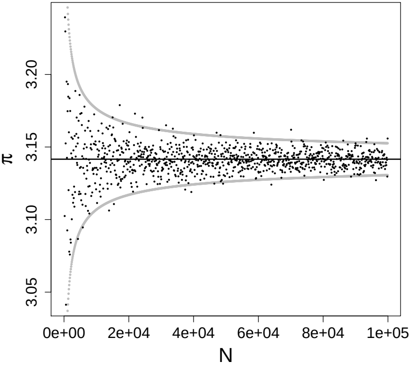
Department of Computer and Information Science
Linköping University

What is the area of the unit circle?

```
f.circArea<-function(N){
    m.xy<-cbind(runif(N),runif(N))
    4*sum(apply(m.xy,1,function(xy){xy[1]^2+xy
        [2]^2<1}))/N
}
```

$$3.141 \approx \pi = \oint 1 \mathrm{d}x$$





Monte Carlo methods: outline

- **Monte Carlo methods** are a class of computational algorithms that use repeated random sampling to compute their results.
- Monte Carlo methods for random number generation
  - Metropolis–Hastings algorithm
  - Gibbs sampler
- Monte Carlo methods for statistical inference
  - Estimate integrals (we already did!)
  - Variance estimation
  - Variance reduction: importance sampling, control variates

Previous lecture: Generate

- univariate distributions (inverse CDF, acceptance/rejection)
- multivariate normal

but general multivariate distribution?

MCMC

Bayesian inference: Recap

A dataset D is obtained by sampling from a distribution $f(\cdot|\theta)$. How to estimate θ?

- *Frequentists*: θ is an unknown but fixed parameter, compose likelihood $\mathcal{L}(D|\theta)$ and find θ that maximizes it.

- *Bayesians*: θ is a random variable with **prior** probability law $p(\theta)$ before observing D

- After observing D, Bayes' theorem gives

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D|\theta)p(\theta)}{\int p(D|\theta)p(\theta)\mathrm{d}\theta}$$

Bayesian inference: Recap

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D|\theta)p(\theta)}{\int p(D|\theta)p(\theta)\mathrm{d}\theta}$$

We know: $p(D|\theta)$ (the model), $p(\theta)$ (the prior)
We need: simulate from $p(\theta|D)$ (the posterior)

1. General (multivariate) type distribution
2. Integral can be impossible to compute

1. MCMC solves this
2. Not needed (given D it is constant)

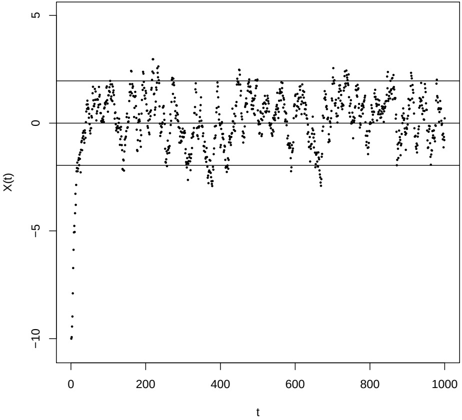
Markov Chains: Recap

- A Markov chain is a sequence X_0, X_1, ... of random variables such that the distribution of the next value depends only on the current one (and parameters).
- $P(X_{t+1}|X_t)$ is called a **transition kernel**. Assume it does not depend on t (**time homogeneous**).

- A Markov chain is **stationary**, with stationary distribution Φ, if $\forall_k \; X_k \sim \Phi$
- One shows (not trivial in general) that under *certain* conditions a Markov chain will converge to the stationary distribution in the limit.

Markov Chains: Example

$$X(t+1) = e^{-1}X(t) + \epsilon \ , \epsilon \sim \mathcal{N}(0, \tfrac{5}{2} \cdot (1 - e^{-2}))$$



Discard first $K-1$ samples: **burn–in period**

MCMC: Example

Linear regression with residual normally/student/etc. distributed

$$Y = \beta X + \epsilon$$

How to find credible interval for β if we know $\mathrm{Var}\left[\epsilon\right] = \sigma^2$?

1. $$P(Y|X,\beta) = \prod_{i=1}^{N} f(Y_i|\text{mean} = \beta X_i, \text{var} = \sigma^2)$$
2. Obtain $P(\beta|Y,X)$ by drawing from $P(Y|X,\beta)P(\beta)$ **in a clever way**.
3. The prior ?
4. Use the MCMC sample to obtain quantiles.

Normal residual: analytical solution

Metropolis–Hastings algorithm

We have

- A PDF $\pi(x)$ that we want to sample from.
- A **proposal distribution** $q(\cdot|X_t)$ that has a **regular** form w.r.t. to $\pi(\cdot)$
  E.g. $q(\cdot|X_t)$ is normal with mean X_t and given variance
- *Regular* form: suffices that the proposal has the same support as π.

Metropolis–Hastings Sampler

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

Initialize chain to X_0, $t = 0$
while $t < t_{\max}$ **do**
    Generate a candidate point $Y \sim q(\cdot|X_t)$
    Generate $U \sim Unif(0, 1)$
    if $U < \alpha(X_t, Y)$ **then**
        $X_{t+1} = Y$
    else
        $X_{t+1} = X_t$
    end if
    $t = t + 1$
end while

Metropolis–Hastings Sampler: Properties

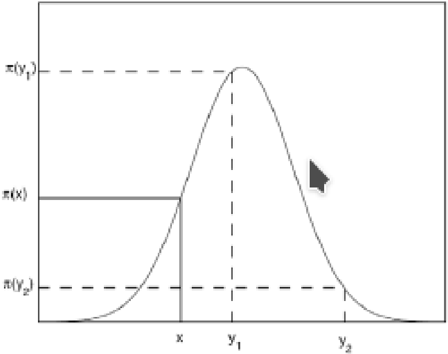
- Informally: "The chain $(X_t)_{t=0}^{\infty}$ will converge to $\pi(\cdot)$."
- The chain might not move sometimes.
- The values of the chain are dependent.
- If $q(X_t|Y) = q(Y|X_t)$ (i.e. symmetric proposal) we get **Random–walk Monte Carlo**:

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

Choice of proposal distribution

- In Random–Walk Monte Carlo

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

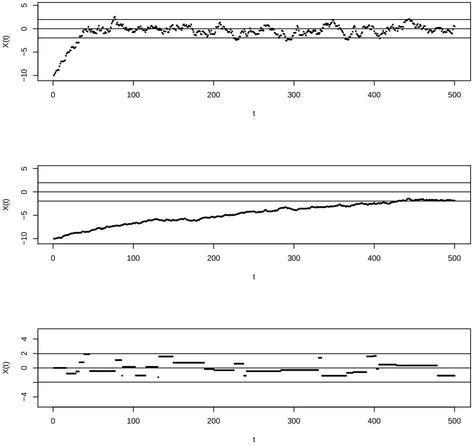


Choice of proposal dist.: **target:** $\pi(\cdot) = \mathcal{N}(0,1)$

```
f.MCMC.MH<-function(nstep,X0,props){
    vN<-1:nstep
    vX<-rep(X0,nstep);
    for (i in 2:nstep){
        X<-vX[i-1]
        Y<-rnorm(1,mean=X,sd=props)
        u<-runif(1)
        a<-min(c(1,(dnorm(Y)*dnorm(X,mean=Y,sd=
            props))/(dnorm(X)*dnorm(Y,mean=X,sd=
            props))))
        if (u <=a){vX[i]<-Y}else{vX[i]<-X}
    }
    plot(vN,vX,pch=19,cex=0.3,col="black",xlab="t",
       ylab="X(t)",main="",ylim=c(min(X0-0.5,-5),
      max(5,X0+0.5)))
    abline(h=0)
    abline(h=1.96)
    abline(h=-1.96)
}
```

Choice of proposal distribution

q normal with sd: `props=` 0.5, 0.1 and 20



Gibbs sampler: alternative to Metropolis–Hastings

We want to generate from a distribution on $\mathbb{R}^d$.

1: Initialize chain to $X_0 = (X_{0,1}, \ldots, X_{0,d})$, $t = 0$
2: **while** $t < t_{\max}$ **do**
3: **for** $i = 1, \ldots, d$ **do**
4: Generate

$$X_{t+1,i} \sim f(\cdot | X_{t+1,1}, \ldots, \mathbf{X_{t+1,i-1}}, \mathbf{X_{t,i+1}}, \ldots, X_{t,d})$$

5: **end for**
6: $t = t + 1$
7: **end while**

Gibbs sampler

- At each iteration inside the `for` loop univariate random numbers are generated.
- Only one element is updated.
- **WE NEED TO KNOW THE CONDITIONAL MARGINAL DISTRIBUTIONS**.
- Convergence may be slow.
- Can be useful in high dimensions (i.e. proposal density may be difficult to find in another way).

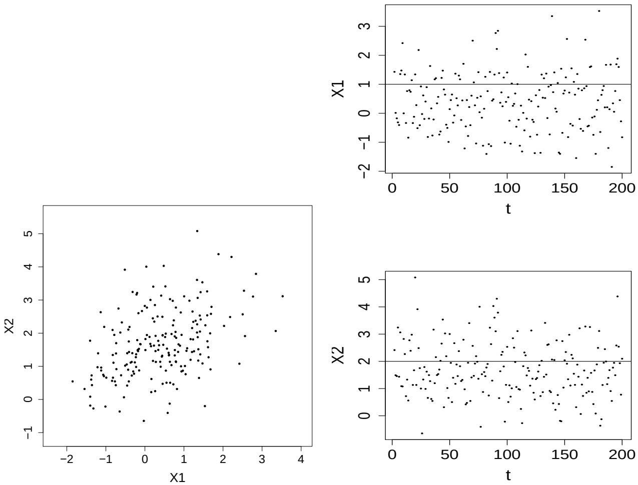
Gibbs sampler: target: d–dim $\mathcal{N}(\mu, \Sigma)$

```
f.MCMC.Gibbs<-function(nstep,X0,vmean,mVar){
    d<-length(vmean); mX<-matrix(0,nrow=nstep,ncol=
        d); mX[1,]<-X0
    for (i in 2:nstep){
        X<-mX[i-1,];Y<-rep(0,d)
        Y[1]<-rnorm(1,mean=vmean[1]+(mVar[1,-1]%*%
            solve(mVar[-1,-1]))%*%(X[2:d]-vmean
            [-1]),sd=sqrt(mVar[1,1]-mVar[1,-1]%*%
            solve(mVar[-1,-1])%*%mVar[-1,1]))
        for (j in 2:(d-1)){Y[j]<-rnorm(1,mean=vmean
            [j]+(mVar[j,-j]%*%solve(mVar[-j,-j]))%*
            %(c(Y[1:(j-1)],X[(j+1):d])-vmean[-j]),
            sd=sqrt(mVar[j,j]-mVar[j,-j]%*%solve(
            mVar[-j,-j])%*%mVar[-j,j]))}
        Y[d]<-rnorm(1,mean=vmean[d]+(mVar[d,-d]%*%
            solve(mVar[-d,-d]))%*%(Y[1:(d-1)]-vmean
            [-d]),sd=sqrt(mVar[d,d]-mVar[d,-d]%*%
            solve(mVar[-d,-d])%*%mVar[-d,d]))
        mX[i,]<-Y
    };mX}
```

Gibbs sampler: Example (code: see R scripts)

Generate from

$$\mathcal{N}([1\ 2]^T, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix})$$



Convergence monitoring

- When should we stop the chain? When are we (nearly) at the stationary distribution?
- Typically such a sample is generated to make further inference.

Convergence monitoring: Gelman–Rubin method

We want to estimate $v(\theta)$.

1. Generate k sequences of length n with different starting points.
2. Compute between– and within– sequence variances:

$$B = \frac{n}{k-1}\sum_{i=1}^{k}(\overline{v}_{i\cdot} - \overline{v}_{\cdot\cdot})^2 \quad W = \sum_{i=1}^{k}\frac{s_i^2}{k} \quad s_i^2 = \sum_{j=1}^{n}\frac{(\overline{v}_{ij} - \overline{v}_{i\cdot})^2}{n-1}$$

3. Overall variance estimate: $\hat{\mathrm{Var}}[v] = \frac{n-1}{n}W + \frac{1}{n}B$
4. Gelman–Rubin factor:

$$\sqrt{R} = \sqrt{\frac{\hat{\mathrm{Var}}[v]}{W}}$$

5. Values much larger than 1 indicate lack of convergence
6. See `?coda::gelman.diag`

```
library(coda)
f1<-mcmc.list();f2<-mcmc.list();n<-100;k<-20
X1<-matrix(rnorm(n*k),ncol=k,nrow=n)
X2<-X1+(apply(X1,2,cumsum)*(matrix(rep(1:n,k),ncol=
    k)^2))
for (i in 1:k){f1[[i]]<-as.mcmc(X1[,i]);f2[[i]]<-as
    .mcmc(X2[,i])}
print(gelman.diag(f1))
# Potential scale reduction factors:
#       Point est. Upper C.I.
#[1,]       0.999       1.01

print(gelman.diag(f2))
# Potential scale reduction factors:
#       Point est. Upper C.I.
#[1,]        1.82       2.38
```

MC for inference

- Estimation of a definite integral

$$\theta = \int_D f(x)\mathrm{d}x \quad \left(\text{recall} \quad \pi = \int_{\bigcirc} 1\mathrm{d}x\right)$$

- Decompose into:

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

- Then, if $X \sim p(\cdot)$

$$\theta = \mathrm{E}\left[g(X)\right] = \int_D g(x)p(x)\mathrm{d}x$$

- $$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} g(x_i), \quad \forall_i x_i \sim p(\cdot)$$

MC for inference

- Decomposition is not unique, some will be better (lower variance) others worse. $p(x) \propto |f(x)|$: minimal
- Can we easily generate from $p(\cdot)$?
- Bayesian inference: use MCMC samples from $p(\theta|D)$ to obtain a point estimator

$$\theta^* = \int \theta p(\theta|D) \approx \frac{1}{n} \sum_{i=1}^{m} \theta_i$$

- $\hat{\theta}$ depends on n and $g(X)$, how variable will it be?

$$\widehat{\mathrm{Var}\left[\hat{\theta}\right]} = \frac{1}{n(n-1)} \sum_{i=1}^{n} \left(g(x_i) - \overline{g(x)}\right)^2$$

- MCMC: estimator biases as chain correlated, use longer chain and batch mean instead of x_i.

Summary

1. Generating data from a general multivariate distribution
2. Markov Chain Monte Carlo: Metropolis–Hastings algorithm, Gibbs sampling
3. Convergence: Gelman–Rubin method
4. Estimation of integral