MSiA 400 Lab 3

Monte Carlo Methods & MCMC

Huiyu Wu

10/17/2022

NORTHWESTERN UNIVERSITY

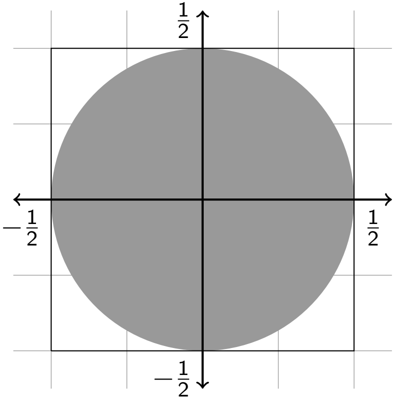
Monte Carlo Methods

- Rely on random sampling to obtain numerical results
- Use randomness to solve problems that may be deterministic
- Common applications
  - Optimization
  - Numerical Integration
  - Generating draws from probability distributions

Monte Carlo Integration

- Goal: compute $I = \int_a^b h(x)dx$
- Monte Carlo Method
  - Let $h(x) = g(x)p(x)$, where $p(x)$ is a pdf defined on $[a, b]$
  - $I = \int_a^b h(x)dx = \int_a^b g(x)p(x)dx = E_p[g(x)]$
  - $x_1, \cdots, x_n \sim p(x)$ iid
  - $I = E_p[g(x)] \approx \frac{1}{n} \sum_{i=1}^{n} g(x_i)$

Monte Carlo Integration Estimating Pi



- Circle inscribed in unit square
- Area of circle $= \frac{\pi}{4}$
- Area of square $= 1$
- $X = (X_1, X_2)$, drawn iid from unif $\left(-\frac{1}{2}, \frac{1}{2}\right)$
- $P\left(X_1^2 + X_2^2 \leq \frac{1}{4}\right) = \frac{\pi}{4}$
- $\pi \approx \frac{4}{n} \sum_{i=1}^{n} \mathbb{1}\left(x_{1i}^2 + x_{2i}^2 \leq \frac{1}{4}\right)$

Monte Carlo Integration Estimating Pi in R

```
set.seed(400)  # Seed RNG
n = 10000
x1 = runif(n,-0.5,0.5)
x2 = runif(n,-0.5,0.5)
z = x1^2+x2^2
Pi = 4*sum(z<=0.25)/n
Pi
```

```
## [1] 3.1412
```

```
pi
```

```
## [1] 3.141593
```

Markov Inequality

- Statement: $P\left(Y \geq a\right) \leq \frac{E(Y)}{a}$
- Conditions: Y is non-negative and $a > 0$.
- What if: $Y = \left(X - \mu\right)^2$ and $a = \delta^{-1}\sigma^2$

Error Estimation

- Chebyshev Inequality: $P\left((X-\mu)^2 \geq \frac{\sigma^2}{\delta}\right) \leq \delta$
- This implies: $P\left((\text{error})^2 \geq \frac{\text{var}[p(x)]}{n\delta}\right) \leq \delta$
- Number of samples needed to n to meet tolerance with $(1-\delta)$ confidence: $n \geq \frac{\text{var}[p(x)]}{(\text{tolerance})^2\delta}$
- How many samples do we need to 99% confident that we can compute pi to 2 decimal places? Note: $\text{var}\left[p(x)\right] = \frac{1}{12^2}$

$$n \geq \frac{100 * 100^2}{144} \approx 6944$$

Markov Chain Monte Carlo (MCMC)

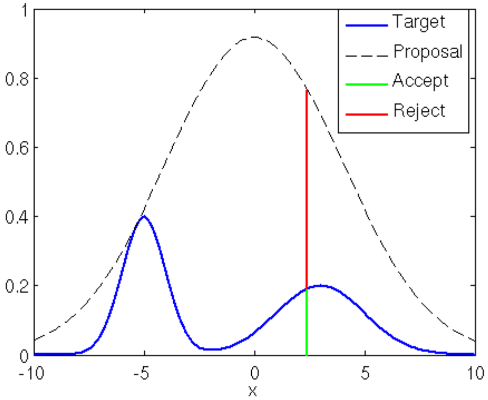
- What if $p(\cdot)$ is difficult to sample from?
- Idea: construct a Markov chain, where the equilibrium distribution is the desired distribution $p(\cdot)$.
- Detailed balance: $\pi_i p_{ij} = \pi_j p_{ji}$
- Bayes' Theorem: $p(x|y) = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx} \propto p(y|x)p(x)$

Rejection Method

Algorithm 1: Rejection Methods

1 Choose $q(\cdot)$, approximation of $p(\cdot)$
2 Find M st $p(x) \leq Mq(x), \ \forall x$
3 Draw y from $q(\cdot)$
4 Draw $u \sim \text{unif}(0,1)$
5 Compute acceptance ratio $\alpha = \frac{p(x)}{Mq(x)}$
6 **if** $u \leq \alpha$ **then**
7   Accept $x = y$
8 **else**
9   Reject: return to line 3, and try again

Intuition



Metropolis Algorithm

Algorithm 2: Metropolis Algorithm

Choose x_0
Choose $q(x|y)$ symmetric (i.e., $q(x|y) = q(y|x)$)
Choose $f(x) \propto p(x)$
for $t = 0, 1, \cdots$ **do**
    Draw x' from $q(\cdot|x_t)$
    Compute acceptance ratio $\alpha = \frac{f(x')}{f(x_t)}$
    Draw $u \sim \text{unif}(0, 1)$
    if $u \leq \alpha$ **then**
        Accept $x_{t+1} = x'$
    else
        Reject, set $x_{t+1} = x_t$

Metropolis-Hastings Algorithm

Algorithm 3: Metropolis-Hastings Algorithm

Choose x_0
Choose $q(x|y)$ (general)
Choose $f(x) \propto p(x)$
for $t = 0, 1, \cdots$ **do**
    Draw x' from $q(\cdot|x_t)$
    Compute acceptance ratio $\alpha = \frac{f(x')q(x_t|x')}{f(x_t)q(x'|x_t)}$
    Draw $u \sim \text{unif}(0, 1)$
    if $u \leq \alpha$ **then**
        Accept $x_{t+1} = x'$
    else
        Reject, set $x_{t+1} = x_t$

MCMC

- Metropolis & Metropolis-Hastings algorithms generate sample points $\{x_0, \cdots, x_t, \cdots\}$
- These form a Markov chain as the probability α of transitioning from x_{t-1} to x_t does not depent on $\{x_0, \cdots, x_{t-2}\}$
- Stationary distribution of Markov chain equals target distribution $p(\cdot)$

MCMC

- Typical implementation
  - Burn-in period k – samples for algorithm to approach desired distribution
  - Keep every m samples – thinning method to reduce auto-correlation
- Diagnostics
  - Plot of sample points: poor mixing and/or patterns vs. well mixing
  - Serial autocorrelations, time lag
  - Formal tests: Geweke test, Raftery-Lewis test

MCMC

- Choice of $q(\cdot|\cdot)$
  - Random-walk chain: new value $x' = x_t + z$,

    $$q(x'|x_t) = r(x' - x_t) = r(z)$$

    If $r(z) = r(-z)$, then $q(\cdot|\cdot)$ is symmetric.(ex: normal)
  - Independent chain: new value x' drawn independent from previous point

    $$q(x'|x_t) = s(x') \text{ for any distribution } s(x')$$

    Generally not symmetric

Gibbs Sampling

- Simpler to sample from conditional than from joint.

Algorithm 4: Gibbs Sampler

Choose $\mathbf{x}^{(0)} = \left(x_1^{(0)}, \cdots, x_n^{(0)}\right)$

for $t = 0, 1, \cdots$ **do**

  for $j = 0, \cdots, n$ **do**

    Draw $x_j^{(t+1)}$ from $p\left(\cdot | x_1^{(t+1)}, \cdots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \cdots, x_n^{(t)}\right)$

MCMC Example

- Poisson regression
- No closed-form solution for optimal weights β^*, must be found using numerical methods

$$y_i | x_i \sim \text{Poisson}\left(\mu_i = \exp\left(x_i^T \beta\right)\right)$$

$$p\left(\beta | y, X\right) \propto \exp\left(\sum_{i=1}^{n} \left[-\exp\left(x_i^T \beta\right) + y_i x_i^T \beta\right]\right) \pi\left(\beta\right)$$

MCMC Example in R

- Puffin Dataset (38 observations at Great Island, Newfoundland)
  - Nest nesting frequency (burrows per 9 square meters)
  - Grass grass cover (percentage)
  - Soil mean soil depth (in centimeters)
  - Angle angle of slope (in degrees)
  - Distance distance from cliff edge (in meters)

```
library(LearnBayes)
```

```
puffin[1:5,]
```

```
##   Nest Grass Soil Angle Distance
## 1   16    45 39.2    38        3
## 2   15    65 47.0    36       12
## 3   10    40 24.3    14       18
## 4    7    20 30.0    16       21
## 5   11    40 47.6     6       27
```

MCMC Example in R

```
#Frequentist view
pfit = glm(Nest~Grass+Soil+Angle+Distance,poisson,data=puffin)
summary(pfit)
```

```
##
## Call:
## glm(formula = Nest ~ Grass + Soil + Angle + Distance, family = poisson,
##     data = puffin)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3262  -1.2984  -0.6617   0.8119   2.5304
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.069973   0.452568   6.783 1.17e-11 ***
## Grass        0.005441   0.003104   1.753  0.07960 .
## Soil         0.033441   0.010822   3.090  0.00200 **
## Angle       -0.030077   0.010724  -2.805  0.00504 **
## Distance    -0.089399   0.010680  -8.371  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 310.427  on 37  degrees of freedom
## Residual deviance:  68.765  on 33  degrees of freedom
## AIC: 183.38
##
## Number of Fisher Scoring iterations: 6
```

MCMC Example in R

```r
#Bayesian method
library(MCMCpack)
```

```r
#Assumes normal prior
Bpfit = MCMCpoisson(Nest~Grass+Soil+Angle+Distance,data=puffin,burnin=1000,mcmc=25000,thin=25)
summary(Bpfit)
```

```
##
## Iterations = 1001:25976
## Thinning interval = 25
## Number of chains = 1
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                  Mean       SD  Naive SE Time-series SE
## (Intercept)  3.066791 0.468017 1.480e-02      0.0164472
## Grass        0.005422 0.003108 9.827e-05      0.0001072
## Soil         0.033598 0.011199 3.541e-04      0.0004051
## Angle       -0.030116 0.011293 3.571e-04      0.0004012
## Distance    -0.090009 0.011227 3.550e-04      0.0004010
##
## 2. Quantiles for each variable:
##
##                  2.5%       25%       50%       75%     97.5%
## (Intercept)  2.196294  2.734165  3.059824  3.364891  3.993143
## Grass       -0.000678  0.003212  0.005457  0.007519  0.011188
## Soil         0.010709  0.026503  0.034166  0.041186  0.055244
## Angle       -0.051478 -0.037771 -0.029952 -0.022732 -0.007356
## Distance    -0.111761 -0.097584 -0.089772 -0.082416 -0.067911
```