

# DSL Seminar: MCMC (3)

Kyung-han Kim

Data Science Lab

July, 2023

# 목차

- Markov Chain Monte Carlo (MCMC)란 무엇인가
- MCMC의 사용 목적 (통계전산, 베イズ통계)
- Metropolis-Hastings Algorithm
- R로 Metropolis-Hastings algorithm 구현하기

# Review

- Markov Chain
- Monte Carlo Integration

# 난수 생성의 새로운 방식

- 균일분포 ( $U(0, 1)$ )에서는 pseudo-random number를 생성할 수 있습니다.
- 이를 바탕으로 Inverse-CDF method, Importance sampling, Rejection sampling, Variable transformation 등을 활용하면 이항분포, 지수분포, 감마분포, 정규분포 등 다양한 확률분포로부터 난수를 생성할 수 있습니다.
- 그렇다면 위와 같이 정형화된 확률분포가 아닌, pdf가 아주 복잡한 확률분포에서는 어떻게 난수를 생성할 수 있을까요? → MCMC!

# Markov Chain Monte Carlo (MCMC)

- MCMC는 어떤 irreducible aperiodic Markov Chain의 stationary distribution  $\pi$ 로부터 random sample을 만들어 내는 기법입니다.
- MCMC는 Markov Chain을 사용하는데, Markov chain에서 한 시점의 값은 직전 시점의 값에 영향을 받습니다.  
이러한 Markov chain의 성질은 iid sampling을 불가능하게 만듭니다.  
: MCMC는 **dependent sampling** 기법입니다.
- 또한 이 dependency 때문에, 앞쪽 sample들은 **초깃값의 영향**을 강하게 받습니다.
- 실제로 사용할 때에는, 우리가 sample을 만들어 내기 원하는 확률분포인 target distribution이  $\pi$ 라고 간주하고 MCMC를 사용합니다.

# MCMC의 사용 목적

- 두 수업 모두에서 MCMC를 사용하는 기본 목적은 동일합니다.
- MCMC는 iid sampling이 불가능할 정도로 복잡한 확률분포에서 표본을 얻어내기 위해 사용하는 기법입니다.
- 통계전산에서 target distribution은 iid sampling이 까다로운 모든 확률분포를 대상으로 합니다.
- 베이지통계에서 target distribution은 주로 posterior distribution입니다.
  - $(\text{Posterior}) \propto (\text{Prior}) \times (\text{Likelihood})$

# Metropolis-Hastings Algorithm (Discrete)

- 가장 대중적으로 널리 사용하는 MCMC 기법입니다.
- 확률변수  $X$ 의 pmf가  $\pi$ 라고 할 때, M-H algorithm은 아래의 방법대로 Markov chain을 만들어 냅니다:
  - 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
  - 2 Sample  $X' \sim Q(X'|X^{(t)})$
  - 3 Sample  $U \sim U(0, 1)$
  - 4 If  $U < \frac{\pi(X')Q(X^{(t)}|X')}{\pi(X^{(t)})Q(X'|X^{(t)})}$ , then  $X^{(t+1)} = X'$ . else,  $X^{(t+1)} = X^{(t)}$
  - 5  $t = t + 1$ , Go to 2.
- 각각의 step의 의미를 살펴 봅시다.

# M-H Algorithm: step-by-step (1)

1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .

- Markov chain을 결정짓기 위한 조건 중 하나가 Initial value ( $X^{(0)}$ ) 입니다.
- 시점을 0으로 초기화하고, 초깃값을 설정하는 단계입니다.
- $t$ 를 계속 증가시킬 경우 초깃값은 최종 결과에 큰 영향을 미치지 않지만, 되도록 reasonable하게 정해주는 것이 좋습니다.



# M-H Algorithm: step-by-step (2)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
- 2 Sample  $X' \sim Q(X'|X^{(t)})$ 
  - $Q$ 는 Irreducible Markov Chain의 Transition probability matrix입니다.
  - 이  $Q$ 를 이용해서 다음 시점에서의 값이 될 후보 값인  $X'$ 를 생성합니다.
    - 그런 이유로  $Q$ 를 proposal이라고 부릅니다.
  - Transition probability matrix는 state가  $i$ 에서  $j$ 로 이동할 확률인  $p_{ij}$ 로 이루어져 있습니다.
  - 이전 시점 값 ( $X^{(t)} = i$ )을 알아야 다음 시점 값으로 이동할 확률을 알 수 있기 때문에  $X^{(t)}$ 를 given으로 표시합니다. (dependent sampling)
  - $Q$ 는 우리가 마음대로 정해서 사용할 수 있으나, 되도록 symmetric한 것이 좋습니다. (Metropolis Algorithm)
  - $Q$ 가 symmetric일 경우 Step(4)에서  $\alpha$  계산이 용이해집니다!

# M-H Algorithm: step-by-step (3)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
- 2 Sample  $X' \sim Q(X'|X^{(t)})$
- 3 Sample  $U \sim U(0, 1)$ 
  - Metropolis-Hastings algorithm에는 다음 시점이 될 만한 값을 생성해 놓고, 그 값을 사용할 것인지 여부를 결정하는 단계가 있습니다.
  - 이  $U$ 는 해당 단계에서 사용하기 위해 미리 생성하는 값입니다.
  - 다음 단계에서  $U$ 와 threshold 값을 비교해서 무엇이 더 큰지에 따라  $X'$ 를 사용할지 말지 정합니다.

# M-H Algorithm: step-by-step (4)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
- 2 Sample  $X' \sim Q(X'|X^{(t)})$
- 3 Sample  $U \sim U(0, 1)$
- 4 If  $U < \frac{\pi(X')Q(X^{(t)}|X')}{\pi(X^{(t)})Q(X'|X^{(t)})}$ , then  $X^{(t+1)} = X'$ . else,  $X^{(t+1)} = X^{(t)}$ 
  - $\alpha = \frac{\pi(X')Q(X^{(t)}|X')}{\pi(X^{(t)})Q(X'|X^{(t)})}$  가  $U$ 와 비교되는 threshold 값입니다.
  - 만약  $X'$ 가  $X^{(t)}$  보다 그럴싸하다면,  $\pi(X') > \pi(X^{(t)})$ , 다시 말해, pmf 값이  $x = X'$  일 때가  $x = X^{(t)}$  일 때보다 클 것입니다. 이 경우  $\alpha > 1$ 을 기대할 수 있습니다.
  - 반대의 경우  $\alpha < 1$ 일 가능성이 클 것입니다.
  - 이번 단계의 목적은  $\min(1, \alpha)$ 의 확률로  $X'$ 를 채택하는 것입니다.
    - $\alpha > 1 \rightarrow \min(1, \alpha) = 1$ : 무조건 사용
    - $\alpha < 1 \rightarrow \min(1, \alpha) = \alpha$ :  $\alpha$ 의 확률로 사용 ( $1 - \alpha$ : 기존 값 유지)
  - $P(U < \alpha) = \min(1, \alpha)$ 이므로  $U$ 와  $\alpha$ 를 비교해 위의 목적을 달성!
  - $Q$ 는 두 가지 목적으로  $\alpha$  계산식에 포함됩니다. (symmetric이면 약분)
    - 1) Detailed Balance Condition을 만족시키기 위해
    - 2)  $Q$ 가 asymmetric한 경우 그러한 제약사항을 보정해 주기 위해

# M-H Algorithm: step-by-step (5)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
- 2 Sample  $X' \sim Q(X'|X^{(t)})$
- 3 Sample  $U \sim U(0, 1)$
- 4 If  $U < \frac{\pi(X')Q(X^{(t)}|X')}{\pi(X^{(t)})Q(X'|X^{(t)})}$ , then  $X^{(t+1)} = X'$ . else,  $X^{(t+1)} = X^{(t)}$
- 5  $t = t + 1$ , Go to 2.
  - 다음 시점의 값이 결정되었으므로 시점 ( $t$ )을 1 늘려 주고 다시 위의 절차를 반복해 또다른 sample을 얻습니다.
  - 초깃값은 다시 지정해 줄 필요가 없으므로 2로 돌아갑니다.

# M-H Algorithm: 단계별 의미

- 1 초기값을 설정한다.
- 2 다음 sample이 될 후보 값을 뽑는다.
- 3 다음 sample을 사용할 확률과 비교할  $U$  값을 뽑는다.
- 4 후보 값을 사용할 확률  $\alpha$ 와  $U$ 를 비교해 사용 여부를 정한다.  
사용하기로 나왔다면 다음 시점 값은 후보 값이 되고,  
아니라면 이전 시점 값을 그대로 다음 시점에도 사용한다.
- 5 시점을 1 늘리고 다시 돌아가 절차를 반복한다.

# Metropolis-Hastings Algorithm (Continuous)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
  - 2 Sample  $X' \sim q(X'|X^{(t)})$ .
  - 3 Sample  $U \sim U(0, 1)$ .
  - 4 If  $U < \frac{\pi(X')q(X^{(t)}|X')}{\pi(X^{(t)})q(X'|X^{(t)})}$ , then  $X^{(t+1)} = X'$ . Else,  $X^{(t+1)} = X^{(t)}$ .
  - 5  $t = t + 1$ , Go to 2.
- Transition probability matrix  $Q$ 가 proposal density  $q$ 로 바뀌는 것 이외에는 모든 것이 동일합니다.
  - 그렇지만 그 하나가 매우 많은 것을 바꾸기 때문에 조금 더 자세히 살펴 보려고 합니다.

# M-H Algorithm: step-by-step (2) (Continuous)

- 1 Initialize  $t = 0, X^{(0)} = x_0 \in \mathcal{X}$ .
- 2 Sample  $X' \sim q(X'|X^{(t)})$ .
  - $q(X'|X^{(t)})$ : 평균이  $X^{(t)}$  인 proposal density  $q$ 에서  $X'$ 를 생성한다.
  - Proposal density는 기본적으로 임의로 설정할 수 있습니다.  
주로 정규분포를 많이 사용합니다.
    - 평균은  $X^{(t)}$ 라고 치자. 분산은? (acceptance rate와 관련 있음)
  - Proposal density가 symmetric하면 계산이 용이합니다.
  - 우리가 원하는 target distribution에서는 iid sample을 만들기 힘들니,  
보다 쉽게 난수를 생성할 수 있는 다른 확률분포 (proposal)에서 후보군을  
만들어 낸 다음에, 그 중 적합한 것들만 골라 내서 사용하려는 목적입니다!
  - Target distribution을 따르는 모수에 제약조건이 있다면 그 조건에 맞게  
sampling을 해 줘야 합니다.
    - Ex] 무조건 양수여야 한다.

# Minor problem: Underflow

- Underflow: 값이 너무 0에 가까워져서 컴퓨터가 이를 제대로 인식하지 못하는 오류
- $\alpha = \frac{\pi(X')q(X^{(t)}|X')}{\pi(X^{(t)})q(X'|X^{(t)})}$  를 계산할 때 경우에 따라  $\pi(\cdot)$  값들이 너무 작아져서 Underflow가 발생,  
 $\alpha = \frac{0}{0} = \text{NaN}$ 으로 출력되는 오류가 생길 수 있습니다.
  - $\pi(\cdot)$ 가 posterior distribution으로 설정되는 베이지스통계에서 더 자주 발생.
- 이럴 경우  $\alpha$ 가 아닌  $\log \alpha$ 를 계산하고, 이를  $\log U$ 와 비교해 해결합니다.
- 베이지스통계 과제를 통해 추후 더 자세히 살펴 보겠습니다.



# ChatGPT: What is M-H algorithm?



Explain me about Metropolis-Hastings algorithm.



The Metropolis-Hastings algorithm is a Markov Chain Monte Carlo (MCMC) method for generating samples from a target distribution, which is often complex and high-dimensional. It is particularly useful when the target distribution cannot be directly sampled from, but can be evaluated up to a normalizing constant.



The algorithm starts with an initial state, and then repeatedly proposes a new state from a proposal distribution. The proposal is accepted or rejected with a probability determined by the target distribution and the proposal distribution. If the proposal is accepted, the new state becomes the current state. Otherwise, the current state is kept.

The algorithm is named after Nicholas Metropolis and W. Keith Hastings, who first described it in a 1953 paper. The Metropolis-Hastings algorithm is a popular method for Bayesian inference and other applications in statistics, physics, and machine learning.

Figure 1: ChatGPT: Metropolis-Hastings Algorithm

# MCMC 수렴 여부 확인하기

- 앞서 언급한 절차대로 MCMC를 잘 구성하더라도, 그 결과물로 얻게 되는 Markov chain 값 (=random sample from  $\pi(\cdot)$ )들은  $\pi(\cdot)$ 에 수렴하지 않을 수도 있습니다.
- 그래서 MCMC로 sample을 만들어 낸 후에는, 해당 sample이 내가 원하는 target distribution  $\pi$ 에 잘 수렴했는지, 수렴했다면 언제쯤부터 수렴하는지를 직접 확인하는 절차가 필요합니다.
- 과제에서는 주로 아래의 내용들을 확인하기를 요구합니다:
  - Trace plot (Time series plot)
  - Histogram
  - Auto-correlation plot
  - Acceptance probability
  - Effective Sample Size (ESS)
- 베이지통계에서는 Sampling이 성공적이었다는 전제 하에, 아래의 내용을 추가로 확인해 보기를 요구하기도 합니다.
  - Posterior mean
  - 95% HPD interval

# (1) Trace plot (시계열그림)

- Trace plot을 통해 시간에 따른 sample 값의 변화를 확인합니다.
- MCMC가 잘 수렴했다면, 일정 시점 이후의 sample들은 특정 값 주위에서 큰 변동 없이 움직여야 합니다.
- ts.plot() 함수를 이용합니다.

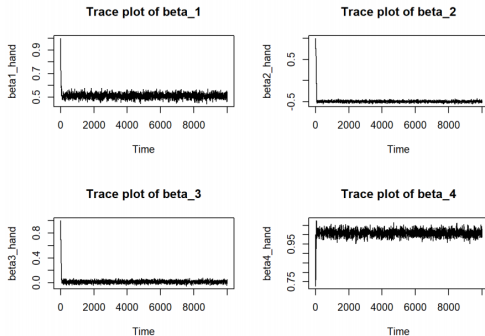


Figure 2: Example of well-converged trace plot

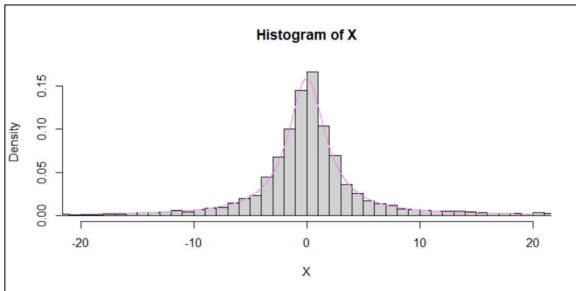
# (1)-1 Burn-in period

- 앞선 trace plot에서 볼 수 있듯이, MCMC가 수렴하는 데까지는 일정 정도의 iteration (time)이 필요합니다.  
수렴하기 이전의 sample들은 우리의 target distribution  $\pi$ 에서 나왔다고 보기 어렵습니다.
- 따라서 수렴 이전의 sample은 사용하는 것이 부적절하다고 판단, 만들어진 MCMC sample의 앞부분 일부를 제거하는 과정이 필요합니다.
- 이 과정을 **Burn-in**이라고 하며, MCMC에서 필수적인 요소입니다!
- 아래의 경우 감점 사유에 해당될 수 있습니다:
  - 통계전산 과제에서 burn-in을 사용하라고 했음에도 누락하는 경우
  - 베이지통계 과제에서는 따로 언급이 없더라도 burn-in을 하는 것을 권장 (불시에 감점될 수 있음)
  - burn-in을 적용했더라도 burn-in period가 reasonable하지 않은 경우
  - Number of Iteration이 불충분한 경우 (권장:  $n=10,000$ )

## (2) Histogram (히스토그램)

- 히스토그램을 통해 MCMC sample이 target distribution과 분포가 일치하는지 확인할 수 있습니다. (hist() 함수를 사용)
  - 단, target distribution을 직접 그릴 수 있는 경우에 한해 요구합니다.

다음은 히스토그램을 그려 보겠습니다. 문제에서 주어진 코시분포의 pdf까지 겹쳐서 그려보겠습니다.



보라색으로 그려진 곡선이 코시분포입니다.

히스토그램이 pdf와 상당히 유사한 형태를 보이며 sampling이 이루어졌다는 것을 볼 수 있습니다.

Figure 3: Histogram with target density function

### (3) Auto-correlation plot (자기상관그림)

- MCMC는 더이상 iid sampling이 아니기 때문에, sample들 간에 correlation이 있습니다.
- 이 correlation의 정도를 시각화해 주는 것이 Auto-correlation plot입니다.
- Autocorrelation은 빠르게 0에 가깝게 감소하는 것이 바람직합니다.

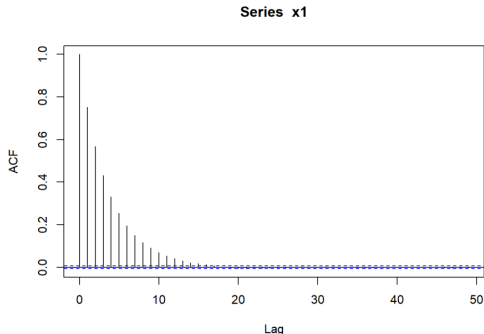


Figure 4: Example of Autocorrelation plot

## (4) Acceptance Probability/Rate

- Metropolis-Hastings algorithm에서는 random sample이 만들어졌다고 해서 해당 sample을 무조건 사용하지는 않습니다.
- Threshold 값인  $\alpha$ 를  $\pi$ 와  $q$ 로 계산해 새롭게 제안된 sample인  $X'$ 을 실제로 사용할 확률을 구하게 됩니다.
- 이 때  $X^{(t+1)} = X'$ 가 되면 이를 'accept'라고 합니다.
- Acceptance rate =  $\frac{\text{Total number of accept}}{\text{Total number of Iteration}}$  로 정의됩니다.
- R에서 acceptance rate를 구하는 방법에는 2가지가 있습니다:
  - Metropolis-Hastings algorithm 코드 내에서 accept 변수를 지정,  $X'$ 가 accept될 때마다  $\text{accept} = \text{accept} + 1$ .  
이 때 acceptance rate는  $\frac{\text{accept}}{n\_iter}$  가 된다.
  - $\text{length}(\text{unique}(\text{sample})) / \text{length}(\text{sample})$ 로 구하기! (편해서 추천)
- 바람직한 acceptance rate는 20%에서 50% 사이입니다.  
Acceptance rate 100%는 결코 바람직하지 못합니다. 왜일까요?

## (5) Effective Sample Size (ESS)

- MCMC를 이용한 표본 생성은 Dependent sampling입니다.
- Dependent sampling은 기본적으로 independent sampling보다 성능이 좋지 못합니다.
- 그렇다면 구체적으로 얼마나 성능이 떨어질까요?
- ESS는 현재 내가 가진 MCMC sample이, 길이가 어느 정도인 independent sample과 비슷한 효과를 내는지 알려 줍니다.
- Ex] MCMC로 sample을 10,000개 만들었는데  $ESS=2,000$  이다  
= 내가 가진 MCMC sample은 iid sample 2,000개와 비슷하다.
- ACF 값이 클수록 ESS는 작아집니다.
- $ESS < 100$  인 경우 감점 사유입니다. (베이지스통계)