Lecture 3. Generation of RVs

Tiejun Li[1,2]

[1]School of Mathematical Sciences (SMS),
&
[2]Center for Machine Learning Research (CMLR),
Peking University,
Beijing 100871,
P.R. China
tieli@pku.edu.cn

Office: No. 1 Science Building, Room 1376E

Table of Contents

Basic MC method

The MC method for integration is as follows:

$$I(f) = \int f(x)p(x)dx \quad \longrightarrow \quad I_N(f) = \frac{1}{N}\sum_{i=1}^{N} f(X_i), \quad X_i \sim p(x) \ \ i.i.d.$$

From the WLLN, $I_N(f) \to I(f)$ in probability.

Problem: How to generate the random variables X_i?
$(i = 1, \ldots, N)$

Generation of RVs

- The first step to apply Monte Carlo method is to generate random variables. In computer simulations the random variables are replaced with pseudo-random variables for the reason of repeatability.

Generation of RVs

- The first step to apply Monte Carlo method is to generate random variables. In computer simulations the random variables are replaced with pseudo-random variables for the reason of repeatability.
- We will show in the continued texts that the arbitrary distribution can be generated based on the uniform distributions.

Generation of RVs

- The first step to apply Monte Carlo method is to generate random variables. In computer simulations the random variables are replaced with pseudo-random variables for the reason of repeatability.
- We will show in the continued texts that the arbitrary distribution can be generated based on the uniform distributions.
- Let us start with generating the uniform distribution $\mathcal{U}[0,1]$. We recommend (Book: Numerical Recipes in C.) for the codes to be used in practice.

Uniform distribution: PRNG

- The most commonly used pseudo-random number generator (PRNG) for $\mathcal{U}[0,1]$ is based on the linear congruential generator (LCG) and its different kinds of variants. It has the following simple form

$$X_{n+1} = aX_n + b(\mathrm{mod}\ m)$$

where a, b and m are chosen natural numbers beforehand, and X_0 is the seed.

Uniform distribution: PRNG

- The most commonly used pseudo-random number generator (PRNG) for $\mathcal{U}[0,1]$ is based on the linear congruential generator (LCG) and its different kinds of variants. It has the following simple form

$$X_{n+1} = aX_n + b(\mathrm{mod}\ m)$$

  where a, b and m are chosen natural numbers beforehand, and X_0 is the seed.
- The obtained sequence X_n/m is the desired pseudo-random number satisfies $\mathcal{U}[0,1]$. The *period* for a typical sequence produced by the above recursion formula is defined as the length of the repeating cycle.

Uniform distribution: Theorem on LCG

It is proven in (Book: The Art of Computer Programming) that

Theorem

The period of a LCG is m if and only if
(i) b and m are relatively prime;
(ii) every prime factor of m divides $a - 1$;
(iii) if $m \Big| 4$, then $(a - 1) \Big| 4$.

To achieve the goal of full period, a good choice in computer implementation is $m = 2^k$, $a = 4c + 1$, and b is odd.

Uniform distribution: Magic 16807

The LCG is also discussed when $b = 0$.

- In 1969, Lewis, Goodman and Miller proposed the following pseudo-random number generator

$$X_{n+1} = aX_n \pmod{m},$$

with $a = 7^5 = 16807, m = 2^{31} - 1$.

Uniform distribution: Magic 16807

The LCG is also discussed when $b = 0$.

- In 1969, Lewis, Goodman and Miller proposed the following pseudo-random number generator

  $$X_{n+1} = aX_n \pmod{m},$$

  with $a = 7^5 = 16807, m = 2^{31} - 1$.
- This generator has passed all new theoretical tests in subsequent years, and resulted in a lot of successful use. They called it "Minimal standard generator" against which other generators should be judged.

Uniform distribution

- It is implemented in the function "ran0()" in the book *Numerical Recipes*. The period of "ran0()" is about 2.1×10^9.

Uniform distribution

- It is implemented in the function "ran0()" in the book *Numerical Recipes*. The period of "ran0()" is about 2.1×10^9.
- With shuffling algorithm by combining sequences with different periods, a more powerful pseudo-random number generator "ran2()" with period about 2.3×10^{18} is constructed.

Uniform distribution

- It is implemented in the function "ran0()" in the book *Numerical Recipes*. The period of "ran0()" is about 2.1×10^9.
- With shuffling algorithm by combining sequences with different periods, a more powerful pseudo-random number generator "ran2()" with period about 2.3×10^{18} is constructed.
- The authors claim that they will pay $\$1{,}000$ for the first person who may convince them by finding a statistical test that this generator fails in a nontrivial way!

Uniform distribution: Limitations of LCG

- More general LCG generators take the following form

$$X_{n+1} = a_0 X_n + a_1 X_{n-1} + \cdots + a_j X_{n-j} + b \pmod{m}.$$

These generators are characterized by the period τ, which in the best case can not proceed $m^{j+1} - 1$. The length of τ depends on the choice of a_j, b and m.

Uniform distribution: Limitations of LCG

- More general LCG generators take the following form

$$X_{n+1} = a_0 X_n + a_1 X_{n-1} + \cdots + a_j X_{n-j} + b \pmod{m}.$$

  These generators are characterized by the period τ, which in the best case can not proceed $m^{j+1} - 1$. The length of τ depends on the choice of a_j, b and m.
- One important fact about the LCG is that it shows very poor distributions of s-tuples, i.e. the vectors $(X_n, X_{n+1}, \ldots, X_{n+s-1})$. Marsaglia proved the important fact.

Uniform distribution: Limitations of LCG

- More general LCG generators take the following form

$$X_{n+1} = a_0 X_n + a_1 X_{n-1} + \cdots + a_j X_{n-j} + b \pmod{m}.$$

  These generators are characterized by the period τ, which in the best case can not proceed $m^{j+1} - 1$. The length of τ depends on the choice of a_j, b and m.
- One important fact about the LCG is that it shows very poor distributions of s-tuples, i.e. the vectors $(X_n, X_{n+1}, \ldots, X_{n+s-1})$. Marsaglia proved the important fact.

Theorem (Marsaglia, PNAS, 1968)

The s-tuples $(X_n, X_{n+1}, \ldots, X_{n+s-1})$ obtained via (5) lie on a maximum of $(s!m)^{\frac{1}{s}}$ equidistant parallel hyperplanes within the s-dimensional hypercube $(0,1)^s$.

Uniform distribution: Remarks

- When s is large, the deviation with respect to the uniform distribution is apparent:

$$(s!m)^{\frac{1}{s}} \sim s/e, \quad s \gg 1$$

Though the LCG has this drawback, it is still the most widely used pseudo random number generator in practice. The nonlinear generators are also discussed to overcome this limitation.

Uniform distribution: Remarks

- When s is large, the deviation with respect to the uniform distribution is apparent:

$$(s!m)^{\frac{1}{s}} \sim s/e, \quad s \gg 1$$

  Though the LCG has this drawback, it is still the most widely used pseudo random number generator in practice. The nonlinear generators are also discussed to overcome this limitation.
- Some very recent mathematical softwares adopt the so-called Mersenne Twister generator, which avoids the linear congruential steps and has the period up to $2^{19937} - 1$.

Uniform distribution: Remarks

- When s is large, the deviation with respect to the uniform distribution is apparent:

  $$(s!m)^{\frac{1}{s}} \sim s/e, \quad s \gg 1$$

  Though the LCG has this drawback, it is still the most widely used pseudo random number generator in practice. The nonlinear generators are also discussed to overcome this limitation.
- Some very recent mathematical softwares adopt the so-called Mersenne Twister generator, which avoids the linear congruential steps and has the period up to $2^{19937} - 1$.
- We remark here that since the generation of RVs are essential for the success of the algorithm, one must use the reliable RV generators from available well-accepted codes or libraries!

Statistical testing

- It is very difficult to distinguish whether a given sequence is generated from deterministic methods or stochastic methods. The practical way to handle this issue is to judge whether it can pass the corresponding statistical testing if the sequence is assumed to be random. That is the principle under which the pseudo-random number generator works.

Statistical testing

- It is very difficult to distinguish whether a given sequence is generated from deterministic methods or stochastic methods. The practical way to handle this issue is to judge whether it can pass the corresponding statistical testing if the sequence is assumed to be random. That is the principle under which the pseudo-random number generator works.
- One may be referred to the book (The Art of Computer Programming) or the document

  https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf

  maintained by the National Institute of Standards and Technology for more details on the empirical tests for PRNG.

Statistical testing

- Equi-distribution test: The interval $(0, 1)$ is divided into K subintervals. Count the number N_j of points falling into the j-th interval and perform χ^2-test to the counts.

Statistical testing

- Equi-distribution test: The interval $(0, 1)$ is divided into K subintervals. Count the number N_j of points falling into the j-th interval and perform χ^2-test to the counts.
- Serial test: Consider the s-vector

$$\boldsymbol{X}_n = (X_n, X_{n+1}, \ldots, X_{n+s-1})$$

  in s-dimensional space $(s > 2)$. The s-hypercube is divided into r^s equi-partitions and perform χ^2-test to the number of counts in each sub-hypercube.

Statistical testing

- Equi-distribution test: The interval $(0,1)$ is divided into K subintervals. Count the number N_j of points falling into the j-th interval and perform χ^2-test to the counts.
- Serial test: Consider the s-vector

$$\boldsymbol{X}_n = (X_n, X_{n+1}, \dots, X_{n+s-1})$$

  in s-dimensional space $(s > 2)$. The s-hypercube is divided into r^s equi-partitions and perform χ^2-test to the number of counts in each sub-hypercube.
- Runs test: Count the statistics of runs-up or runs-down of a sequence from $\mathcal{U}[0,1]$ samples. For example, Let $S = (5,4,6,7,3,2)$, then $R = (-,+,+,-,-)$. We say R has 3 runs and 1 runs-down of length 1, 1 runs-down of length 2, and 1 runs-up of length 2. Different testings can be performed for the statistics of R.

Table of Contents

Inverse Transformation Method

The general random variables $Y \in \mathbb{R}$ can be generated from $\mathcal{U}[0,1]$ in principle based on the following well-known proposition.

Proposition (Inverse Transformation Method)

Suppose the distribution function of Y *is* $F(y)$, *i.e.* $\mathbb{P}(Y \leq y) = F(y)$, *which is strictly increasing.* $X_i \sim \mathcal{U}[0,1]$, *then* $Y_i := F^{-1}(X_i)$ *is the desired random variables.*

Inverse Transformation Method

The geometric interpretation of the above proposition is clear from the following figure.

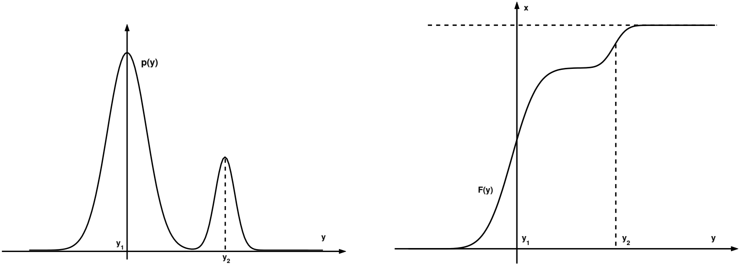

Figure: Left panel: The pdf of Y. Right panel: The distribution function $F(y)$

Inverse Transformation Method

Proof.

If $F(y)$ is strictly increasing, we have the following simple proof for any $y \in \mathbb{R}$

$$\mathbb{P}(Y \leq y) = \mathbb{P}(F^{-1}(X) \leq y) = \mathbb{P}(X \leq F(y)) = F(y).$$

$\square$

Question: How to deal with the case when there are atoms in the distribution of Y or some parts have zero probability?

Inverse Transformation Method

- Generation of $\mathcal{U}[a,b]$:
  The distribution function
  $$F(y) = \frac{y-a}{b-a}, \qquad y \in [a,b],$$
  then $F^{-1}(x) = (b-a)x + a$, so we can take $X_i \sim \mathcal{U}[0,1]$, $Y_i = (b-a)X_i + a$.

Inverse Transformation Method

- Generation of $\mathcal{U}[a,b]$:
  The distribution function

$$F(y) = \frac{y-a}{b-a}, \qquad y \in [a,b],$$

  then $F^{-1}(x) = (b-a)x + a$, so we can take $X_i \sim \mathcal{U}[0,1]$, $Y_i = (b-a)X_i + a$.
- Exponential distribution:
  The distribution function

$$F(y) = 1 - e^{-\lambda y}$$

  then $F^{-1}(x) = -\ln(1-x)/\lambda,\ x \in [0,1]$, so we can take

$$Y_i = -\frac{1}{\lambda}\ln X_i, \qquad (i = 1, 2, \dots)$$

  where $X_i \sim \mathcal{U}[0,1]$ since $1 - X_i \sim \mathcal{U}[0,1]$ also.

Inverse Transformation Method

Now let us investigate the possibility of generating $N(0,1)$ via inverse transformation method. We have

$$F(x) = \int_{-\infty}^{x} p(y)dy = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\Big(\frac{x}{\sqrt{2}}\Big).$$

where $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ is the error function. So

$$F^{-1}(x) = \sqrt{2}\mathrm{erf}^{-1}(2x-1).$$

It is difficult to implement with this formula since it involves the solution of transcendental equations!

Table of Contents

Box-Muller method for Gaussian RVs

- Box-Muller method. Consider a two dimensional Gaussian distributed vector with independent components. With polar coordinates $x = r\cos\theta$, $y = r\sin\theta$, we have

$$\frac{1}{2\pi}e^{-\frac{x_1^2+x_2^2}{2}}dx_1dx_2 = \left(\frac{1}{2\pi}d\theta\right)\cdot(e^{-\frac{r^2}{2}}rdr).$$

Box-Muller method for Gaussian RVs

- Box-Muller method. Consider a two dimensional Gaussian distributed vector with independent components. With polar coordinates $x = r\cos\theta$, $y = r\sin\theta$, we have

$$\frac{1}{2\pi}e^{-\frac{x_1^2+x_2^2}{2}}dx_1dx_2 = \left(\frac{1}{2\pi}d\theta\right)\cdot(e^{-\frac{r^2}{2}}rdr).$$

- Make transformation from a 2D Gaussian to (Θ, R). The measure $1/2\pi d\theta$ corresponds to $\mathcal{U}[0,2\pi]$ in θ space, and $e^{-\frac{r^2}{2}}rdr$ corresponds to the distribution in r-direction with

$$F(r) = \int_0^r e^{-\frac{s^2}{2}}sds = 1 - e^{-\frac{r^2}{2}}.$$

  $F^{-1}(r)$ is easy to be obtained with inverse transformation. So

$$Z_i = R_i\cos\Theta_i,$$

  where $R_i = \sqrt{-2\ln X_i}$, $\Theta_i = 2\pi Y_i$ and $X_i, Y_i \sim \mathcal{U}[0,1] i.i.d.$.

Box-Muller method for Gaussian RVs

Remark

Another approximately generating Gaussian random variable is by central limit theorem

$$X_n = \sqrt{12/N}\Big(\sum_{k=1}^{N}\xi_k - \frac{N}{2}\Big)$$

where $\xi_k \sim \mathcal{U}([0,1])$ *i.i.d.. The CLT asserts that* $N = 12$ *is sufficiently large for many purposes.*

Table of Contents

Composition of random variables

Some distributions can be obtained by the composition of simple random variables instead of the direct application of the previous principles. Here are some examples.

- Sampling the hat pdf.
  Suppose the pdf is

$$f(z) = \begin{cases} z, & 0 < z < 1, \\ 2 - z, & 1 \leq z < 2. \end{cases}$$

  It is interesting to observe that Z has the same distribution with $X + Y$, where X and Y are i.i.d. with distribution $\mathcal{U}[0, 1]$. This suggests that sampling Z can be obtained by the summation of two uniform RVs ξ_1 and ξ_2.

Composition of random variables

- Sampling a random variable raised to a power.
  Let $X_1, \ldots, X_n$ be drawn i.i.d. from the CDF $F_1(x_1), \ldots, F_n(x_n)$. If we set Z to be the largest number among X_i, i.e.

  $$Z = \max\{X_1, \ldots, X_n\}.$$

  Then the CDF of Z will be $F(z) = \prod_{i=1}^{n} F_i(z)$. Suppose we want to generate $Z \sim p(z) = nz^{n-1}$, where $z \in [0, 1]$. Then $F(z) = z^n$ and we can take X_i are $\mathcal{U}[0, 1]$ RVs.

Composition of random variables

- Sampling the mixture models.
  Suppose the pdf

$$f(x) = \sum_{i=1}^{n} \alpha_i g_i(x), \quad \alpha_i \geq 0,\ g_i(x) \geq 0.$$

  We can rewrite it as

$$f(x) = \sum_{i=1}^{n} \beta_i h_i(x), \quad \beta_i = \alpha_i \int g_i(x)dx, \quad h_i(x) = \frac{g_i(x)}{\int g_i(x)dx},$$

  so we have the relation

$$\int h_i(x)dx = 1, \quad \sum_{i=1}^{n} \beta_i = 1.$$

  The sampling of X can be obtained by first sample the index I according to the distribution $\{\beta_i\}_{i=1}^{n}$, and then sample X according to the pdf $h_I(x)$. The rationale behind this is simply by the definition of conditional probability.

Table of Contents

Acceptance-Rejection method

- Though the inverse transformation method gives one approach to generate arbitrary RVs in principle, we have found that it encounters difficulty in implementations if there is no closed form inverse of the CDF.

Acceptance-Rejection method

- Though the inverse transformation method gives one approach to generate arbitrary RVs in principle, we have found that it encounters difficulty in implementations if there is no closed form inverse of the CDF.
- Next we present acceptance-rejection method, which is another general methodology to sample arbitrary RVs.

Acceptance-Rejection method

The aim is to generate RV with density $0 \le p(x) \le d$, $a \le x \le b$. The idea is to lift the state space into a higher dimensional space as shown in Figure.

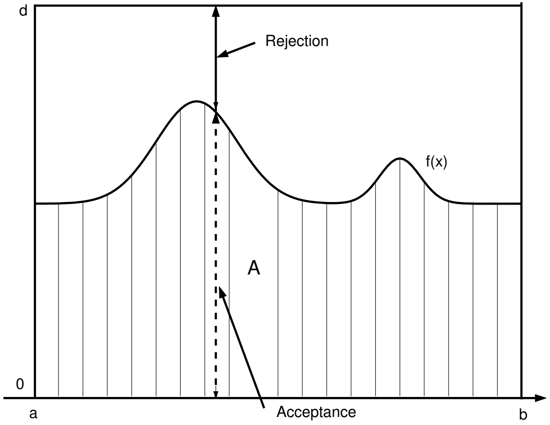

Figure: Schematics of the Acceptance-Rejection method

Acceptance-Rejection method

- Suppose we can sample a uniformly distributed two dimensional random variable (X, Y) in the shaded domain A, where

$$A := \{(x, y) \ : \ x \in [a, b], y \in [0, p(x)]\}.$$

Acceptance-Rejection method

- Suppose we can sample a uniformly distributed two dimensional random variable (X, Y) in the shaded domain A, where
$$A := \{(x, y) \ : \ x \in [a, b], y \in [0, p(x)]\}.$$
- The pdf is $\chi_A(x, y)$ and its X-marginal distribution
$$p_X(x) = \int_0^{p(x)} \chi_A(x, y) dy = \int_0^{p(x)} 1 dy = p(x),$$
which is exactly the desired distribution.

Acceptance-Rejection method

- Suppose we can sample a uniformly distributed two dimensional random variable (X, Y) in the shaded domain A, where
$$A := \{(x, y) \ : \ x \in [a, b], y \in [0, p(x)]\}.$$
- The pdf is $\chi_A(x, y)$ and its X-marginal distribution
$$p_X(x) = \int_0^{p(x)} \chi_A(x, y) dy = \int_0^{p(x)} 1 dy = p(x),$$
which is exactly the desired distribution.
- The uniform distribution in domain A can be easily obtained by the inheritance from the uniform distribution in $[a, b] \times [0, d]$. This naturally leads to the *Acceptance-Rejection* algorithm.

Acceptance-Rejection method

Algorithm (Acceptance-Rejection method)

Generate $X \sim p(x)$.

Step1. *Generate* $X_i \sim \mathcal{U}[a, b]$.

Step2. *Generate a decision-making RV* $Y_i \sim \mathcal{U}[0, d]$.

Step3. *If* $0 \leq Y_i < p(X_i)$, *accept; otherwise, reject.*

Step4. *Back to Step1.*

Acceptance-Rejection method

- For the unbounded random variables, we should introduce more general *comparison functions*. We draw a curve $f(x)$ which lies everywhere above the original distribution density function $p(x)$. This $f(x)$ is called comparison function.

Acceptance-Rejection method

- For the unbounded random variables, we should introduce more general *comparison functions*. We draw a curve $f(x)$ which lies everywhere above the original distribution density function $p(x)$. This $f(x)$ is called comparison function.
- Suppose we can generate the uniform distribution in the two dimensional domain covered by $f(x)$, we can apply the acceptance-rejection strategy to reduce it to the uniform distribution in the domain covered by $p(x)$.

Acceptance-Rejection method

- For the unbounded random variables, we should introduce more general *comparison functions*. We draw a curve $f(x)$ which lies everywhere above the original distribution density function $p(x)$. This $f(x)$ is called comparison function.
- Suppose we can generate the uniform distribution in the two dimensional domain covered by $f(x)$, we can apply the acceptance-rejection strategy to reduce it to the uniform distribution in the domain covered by $p(x)$.
- Then the X-marginal distribution assures us the correct sampling. Now let us consider the generation of uniform RVs with the support covered by $f(x)$ in 2D.

Acceptance-Rejection method

Suppose we have $\int_{-\infty}^{\infty} f(x)dx = A$ and we have the concrete form for $F^{-1}(x)$, where $F(x) = \int_{-\infty}^{x} f(x)dx$. Then we consider the decomposition of uniform measure in $x \in (-\infty, \infty), y \in [0, f(x)]$

$$\frac{1}{A} f(x) dx \frac{1}{f(x)} dy.$$

This introduces a strategy for generating 2D uniform distribution by conditional sampling.

Acceptance-Rejection method

Suppose we have $\int_{-\infty}^{\infty} f(x)dx = A$ and we have the concrete form for $F^{-1}(x)$, where $F(x) = \int_{-\infty}^{x} f(x)dx$. Then we consider the decomposition of uniform measure in $x \in (-\infty, \infty), y \in [0, f(x)]$

$$\frac{1}{A} f(x)dx \frac{1}{f(x)} dy.$$

This introduces a strategy for generating 2D uniform distribution by conditional sampling.

Algorithm (Acceptance-Rejection method with general comparison function)

Generate the unbounded $X \sim p(x)$.

Step1. *Generate $X_i = F^{-1}(AZ_i)$, where $Z_i \sim \mathcal{U}([0,1])$;*

Step2. *Generate decision-making RV $Y_i \sim \mathcal{U}[0, f(X_i)]$;*

Step3. *If $0 \leq Y_i < p(X_i)$, accept; otherwise, reject;*

Step4. *Back to Step1.*

Acceptance-Rejection method

- For bell-shaped random variables, the commonly used comparison function is the Cauchy distribution (or Lorentzian function) because of the slow decay when y is large

  $$p(y) = \frac{1}{\pi(1+y^2)}.$$

  One can check the first and second moments of the Cauchy distribution are both infinity though the principal integral of $p(y)$ is 0 because of symmetry.

Acceptance-Rejection method

- For bell-shaped random variables, the commonly used comparison function is the Cauchy distribution (or Lorentzian function) because of the slow decay when y is large

$$p(y) = \frac{1}{\pi(1+y^2)}.$$

  One can check the first and second moments of the Cauchy distribution are both infinity though the principal integral of $p(y)$ is 0 because of symmetry.

- Since the standard deviation typically characterize the width of the "shoulder" near the center, the infinite second moment gives the reason why it is the usual candidate of comparison functions. Its inverse indefinite integral is jus the tangent function.

Acceptance-Rejection method

- The comparison function is often chosen as the rescaled Cauchy function

$$f(x) = \frac{c_0}{1 + (x - x_0)^2/a_0^2} = c_0 p\Big(\frac{x - x_0}{a_0}\Big).$$

One can adjust the values of x_0, a_0 and c_0 such that it is everywhere greater than $p(x)$.

Acceptance-Rejection method

For the discrete random variables such as the Poisson and binomial distribution. one can extend it into a continuous distribution. With Poisson distribution as an example, we can extend it to $\mathbb{R}$ as

$$q(m) = \frac{x^{[m]} e^{-x}}{[m]!},$$

where $[m]$ represents the largest integer less than m. When x is large enough, we can take Cauchy function as the comparison function.