International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com || Volume 08– Issue 10 || October 2023|| PP. 83-87

An algorithm of the Monte Carlo method to simulate a given distribution by the cumulative distribution function inversion

Le Bich Ngoc[1]
[1]*(Department of Fundamental and Applied Sciences, Thai Nguyen University of Technology, Vietnam)*

Abstract: The paper introduces the simulation algorithm of a given distribution by using the cumulative distribution function inversion. Implementing such simulation algorithm is performed using a practical approach in which the Newton-Raphson method is applied to find an approximation to a sample of the given probability distribution. This approximation enables the simulation in the situation where the inverse of the cumulative distribution function is expensive or impossible to find explicitly.

Keywords: algorithm of the distribution simulation, cumulative distribution function inversion, experimental simulation, Monte Carlo simulation, random variable simulation

I. INTRODUCTION TO MONTE CARLO SIMULATION

The simulation of a given probability distribution becomes valuable in many applied science sectors. An efficient simulation method is needed to reduce the computational cost for the reason of practice with the adaptation in real-time. The Monte Carlos simulation has advantages when dialing with complex systems, where analytical solutions are difficult to obtain or when there is inherent uncertainty. They provide a means to explore a range of possible outcomes, make informed decisions, and assess risks in a wide array of applications. One of the most valuable applications of Monte Carlo simulation in this context is for risk analysis, particularly in financial modeling and decision-making. Monte Carlo simulations are widely used in the pricing of financial options, such as European and American options ([1-3]). By simulating the possible future price paths of underlying assets, financial analysts can estimate the option's value and assess the associated risks. Banks and financial institutions use Monte Carlo simulations to model credit risk. They simulate the probability of default and potential losses associated with a portfolio of loans or bonds. Manufacturing industries use Monte Carlo simulations to model quality control processes, ensuring that products meet specified quality standards even when there are variations in manufacturing conditions. The Monte Carlo method is a preferable option to reduce the computational cost which is only of a linear order of increase with respect to the dimension.

For a given random variable X in the probability space $(\Omega, \mathcal{F}, P)$. Consider the cumulative distribution function (c.d.f) $F: \mathbb{R} \longrightarrow [0,1], x \longmapsto F(x) = P(X \le x)$. We define the left-continuous inverse of the c.d.f F to be the function: $F^{\leftarrow}: [0,1] \longrightarrow \mathbb{R}, u \longmapsto F^{\leftarrow}(u) = \inf\{y \in \mathbb{R}: F(y) \le u\}$. If F is a bijection, then $F^{\leftarrow}$ is identical to the inverse function F^{-1} of F. The following result ([4], Theorem 2.3) provides the simulation algorithm to X.

Theorem. Let F be a c.d.f of a random variable with the inverse $F^{\leftarrow}$. Let U be an uniform random variable over the interval $[0,1]$. Then, $X = F^{\leftarrow}(U)$ has the c.d.f F.

Proof. We have, for all $x \in \mathbb{R}$,

$$P(X \le x) = P(\inf\{y \in \mathbb{R}: F(y) \le U\} \le x) = P(U \le F(x)) = \int_0^{F(x)} dt = F(x).$$

This shows that X has the c.d.f F. Here, the last identity is followed from the fact that $F(x) \in [0,1]$.

Example 1. (Exponential distribution) Let X be the exponential distribution with parameter $\lambda > 0$. The c.d.f of X is $F(x) = \int_0^x \lambda e^{-\lambda t} dt$. Then, $F^{\leftarrow}(U) = F^{-1}(U) = -\frac{1}{\lambda}\ln(1-U)$ is the random variable with the same distribution for the uniform random variable U over the interval $[0,1]$. So, $X = F^{\leftarrow}(U)$ is the simulation to X.

Remark. We can use the simulation $X = -\frac{1}{\lambda}\ln U$, where U is the uniform random variable over the interval $[0,1]$. This function is constructed on the basis of the inverse $F^{\leftarrow}(u) = -\frac{1}{\lambda}\ln(1-u)$ since $(1-U)$and U are also uniformly distributed over $[0,1]$.

Example 2. (Cauchy distribution) The Cauchy distribution with the parameter $\sigma > 0$ has the probability density function $f(x) = \frac{\sigma}{\pi(x^2+\sigma^2)}$. Then, $X = F^{\leftarrow}(U) = F^{-1}(U) = \sigma \tan \pi U$ has the c.d.f F.

II. IMPLEMENTATION

The implementation of the c.d.f inversion requires to find the explicit function $F^{\leftarrow}$. This can only be possible if F is bijective. However, the numerical approach to find the solution $x = F^{\leftarrow}(u)$ of the equation

$F(x) = u$ can be used. In that scene, the Newton-Raphson method is usually suitable to dial with that numerical approach. We propose here an algorithm to implement that idea.

Algorithm. Input: Given a positive probability density function $f: \mathbb{R} \longrightarrow \mathbb{R}_+ = [0, \infty)$, so its c.d.f is $F(x) = \int_0^x f(t)\,dt$, given a prescribed threshold $\varepsilon > 0$.

Output: x to be an approximation to the solution of the equation $F(x) = u$ within the prescribed threshold.

```
Step 1: Draw a sample $u$ from the uniform random variable $U$ over the interval $[0,1]$.
Step 2: Take the initial approximation $x_0 = u$.
Initiate the first approximation $l = 0$.
While $0 \leq l \leq Nmax$: generate the number
    $$x_{l+1} = x_l - \frac{F(x_l) - u}{f(x_l)},$$
  If $|x_{l+1} - x_l| < \varepsilon$, then set $x = x_{l+1}$ and exit the While-loop and get Output: The procedure succeeds, go to Step 3;
  else, $l = l + 1$.
  Step 3: If $l > Nmax$, Output message: The procedure fails;
          else, end the procedure successfully.
```

The error of the approximation is estimated on the basis of absolute error where the convergence of the sequence $(x_l)_{l \geq 0}$ is guaranteed.

Below, we present two Matlab programs to implement the algorithm.

Program 1: $F^{\leftarrow}$ is assumed to be known (It can be explicitly determined)

```
function outp =  invcdf(invF,n)
% n is the sample size
% invF is the inverse of the c.d.f F
for j=1:n
    u=rand();
    outp(j) = invF(u);
end
```

Program 2: $F^{\leftarrow}$ is assumed to be unknown or is not needed to find explicitly

```
function  outp=invcdf(f,n,xL,maxiter,tol)
% n is the sample size
% xL lower limit
% maximum number of iterations
for j=1:n
       u=rand();
       x0 = u;
for i=1:maxiter
           Fx0 = integral(@(t) f(t),xL,x0);
           x1 = x0 - (Fx0-u)/f(x0);
if abs(x1-x0)<tol
break
end
           x0 = x1;
end
       outp(j) = x1;
end
```

III. NUMERICAL EXPERIMENTS

The section aims to present some numerical experiments of the algorithm proposed. This results shows the time consumption of the implementations introduced above. The experiments are perform on a computer with a CPU of 8GB,

Example 3. Consider the exponential distribution with several values of the parameter $\lambda > 0$, with the c.d.f is$f(x) = \lambda e^{-\lambda x}$. We will use in the place of $F^{\leftarrow}$ the simulation as one given in Example 1 for Program 1.

That is, $X = -\frac{1}{\lambda} \ln U$, for $U \sim \mathcal{U}[0,1]$.

The computations are given in Table 1.

λ	Sample size n	Time consume (in second) of Program 1	Time consume (in second) of Program 2 $maxiter = 10, tol = 10^{-3}$
$\lambda = 0.5$	$n = 10,100,1000$	0.008373, 0.009154, 0.012558	0.028497, 0.179688, 1.444786 (Figure 1)
$\lambda = 1$	$n = 10,100,1000$	0.007831, 0.005624, 0.011675	0.026016, 0.146433, 1.222732
$\lambda = 2$	$n = 10,100,1000$	0.008804, 0.004679, 0.007208	0.019931, 0.132912, 1.070706
$\lambda = 21$	$n = 10,100,1000$	0.005680, 0.007002, 0.008162	0.225716, 1.492402, 13.883603

Table 1. Computational time for simulating the exponential random variable

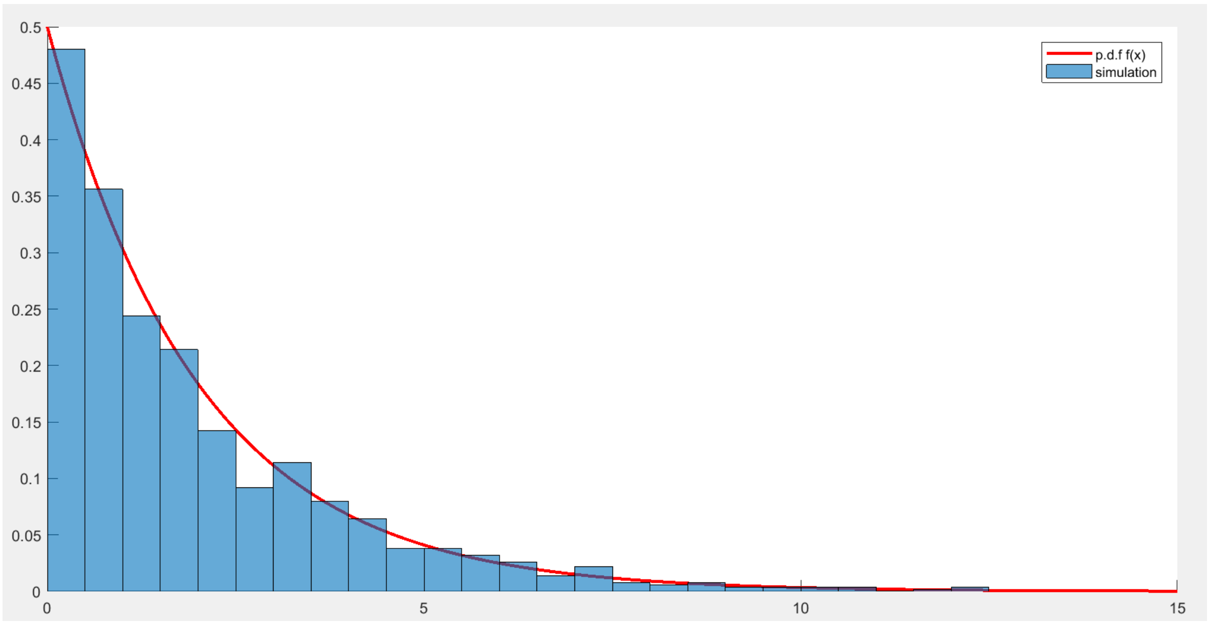

Figure 1. Histogram for the exponential random variable simulated by the program 2 with $\lambda = 0.5, n = 1000$, $maxiter = 10, tol = 0.001$.

Example 4. Consider the random variable X which is the Cauchy distribution with the parameter $\sigma > 0$ with the probability density function $f(x) = \frac{\sigma}{\pi(x^2+\sigma^2)}$. Simulate $X = F^{\leftarrow}(U) = F^{-1}(U) = \sigma \tan \pi U$ with several values of σ. The computational time of Program 1 and Program 2 are shown in Table 2.

σ	Sample size n	Time consume (in second) of Program 1	Time consume (in second) of Program 2 $maxiter = 10, tol = 10^{-3}$
$\sigma = 0.5$	$n = 10,100,1000$	0.023061, 0.025929, 0.030998	0.095558, 0.675244, 5.623255
$\sigma = 1$	$n = 10,100,1000$	0.022628, 0.027330, 0.034090	0.105147, 1.068485, 6.319283
$\sigma = 2$	$n = 10,100,1000$	0.040685, 0.028570, 0.032059	0.091704, 0.921268 (Figure 2), 6.628591
$\sigma = 20$	$n = 10,100,1000$	0.026457, 0.031282, 0.029755	0.158800, 0.855429, 7.552468

Table 2. Simulation of the Cauchy distribution with several values of $\sigma > 0$.

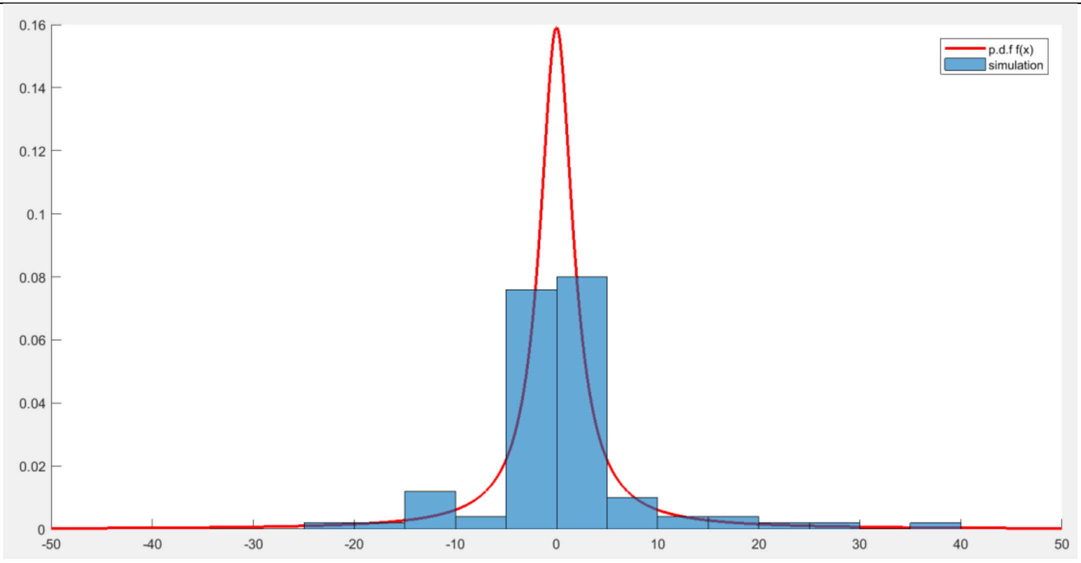

Figure 2. Histogram of the Cauchy distribution with $\sigma = 2$ simulated from Program 2 with $n = 100, maxiter = 10, tol = 10^{-3}$.

Example 5. (Normal distribution) Simulation of the standard normal distribution with the p.d.f. given by $f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$. Program 2 is used to perform the procedure. The computational time is shown in Table 3.

Sample size n	Computational time (in seconds) of the standard normal distribution with Program 2 $maxiter = 10, tol = 10^{-3}$
$n = 10$	0.091017
$n = 100$	0.579677
$n = 1000$	4.106882

Table 3. Computational time for simulating the standard normal distribution with $maxiter = 10, tol = 10^{-3}$.

The algorithm constructed has shown its advantage in situation that the explicit inversion of the c.d.f is hard or impossible to find. The above experiments presents that this algorithm has a good efficiency and the implementation in Matlab is simple with a low computational cost. The output from this implementation attempts a good accuracy when performing on various distributions.

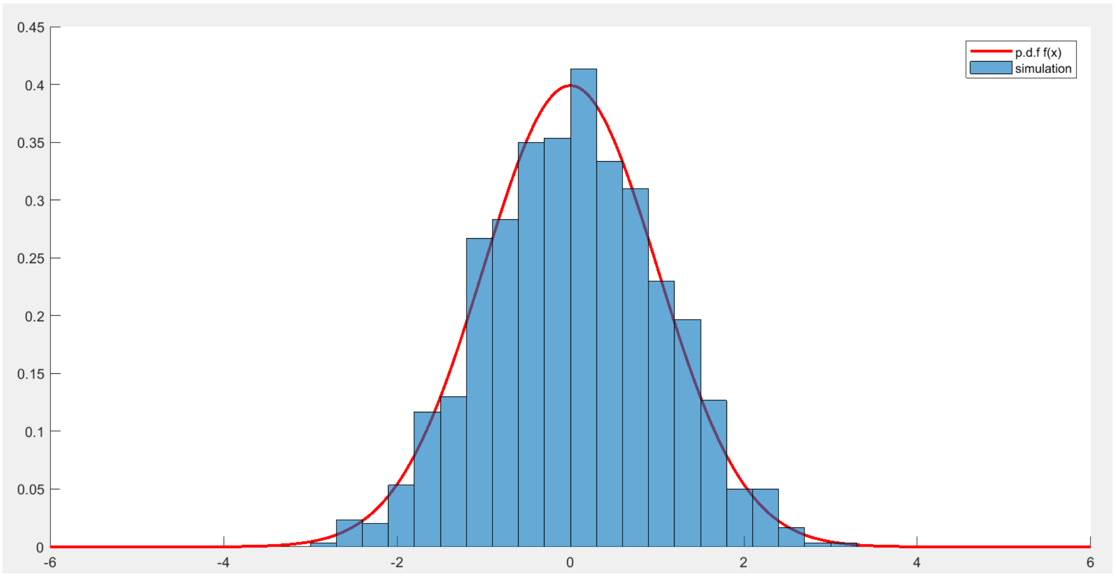

Figure 3. Histogram of the standard normal distribution simulated by Program 2 with $maxiter = 10, tol = 10^{-3}, n = 1000$.

IV. CONCLUSION

The algorithm constructed for the simulation of a distribution by the cumulative distribution function inversion method shows its simplicity and efficiency in performing. The advantages of the techniques is verified especially for the circumstance where an explicit inverse of the cumulative distribution function is difficult or impossible to present. Besides, the paper also presents the implementation for the algorithm in Matlab language. The numerical experiments is presented to prove the advantage of the algorithm proposed.

REFERENCES

Journal Papers:

[1] Black, F., & Scholes, M., The Pricing of Options and Corporate Liabilities, *Journal of Political Economy, 81(3),* 1973, 637-654.

[2] Broadie, M., & Glasserman, P., Pricing American-style securities using simulation." *Journal of Economic Dynamics and Control, 21(8-9),* 1997, 1323-1352

[3] Longstaff, F. A., & Schwartz, E. S., Valuing American Options by Simulation: A Simple Least-Squares Approach, *The Review of Financial Studies, 14(1),* 2001, 113-147.

Books:

[4] C. Graham, D. Talay, *Stochastic simulation and Monte Carlo Methods* (Springer-Verlag, 2013).