Approximation of Standard Normal Distribution by the Monte Carto Methods

Zheng Fang[*]
Qian Zhao[†]

2018/1/28

Abstract

This article is mainly desired to using the Monte Carlo methods to approximate the distribution of standard normal distribution, which is also know as $N(0,\ 1)$

Introduction

Monte Carlo methods (or Monte Carlo experiments) are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. Their essential idea is using randomness to solve problems that might be deterministic in principle. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other approaches.

In principle, Monte Carlo methods can be used to solve any problem having a probabilistic interpretation. By the law of large numbers, integrals described by the expected value of some random variable can be approximated by taking the empirical mean (a.k.a. the sample mean) of independent samples of the variable. When the probability distribution of the variable is parametrized, mathematicians often use a Markov chain Monte Carlo (MCMC) sampler. The central idea is to design a judicious Markov chain model with a prescribed stationary probability distribution. That is, in the limit, the samples being generated by the MCMC method will be samples from the desired (target) distribution. By the ergodic theorem, the stationary distribution is approximated by the empirical measures of the random states of the MCMC sampler.

Math Equations

CDF of standard normal distribution:

$$\Phi(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \tag{1}$$

Approximated CDF by Monte Carlo methods:

$$\hat{\Phi(t)} = \frac{1}{n} \sum_{1}^{n} I(X_i \leq t) \tag{2}$$

[*]zheng.fang@uconn.edu; Master student at Department of Mathmetics, University of Connecticut.
[†]qian.4.zhao@uconn.edu; Master student at Department of Mathmetics, University of Connecticut.

Data Generated by Computer

By using function rnorm, we generate a N dimantion vetor X of ramdom numbers that according to the standard normal distribution, where N = {10^2, 10^3, 10^4}. below is one example of vector X when N = 100, and each experiment should repeat 100 times.

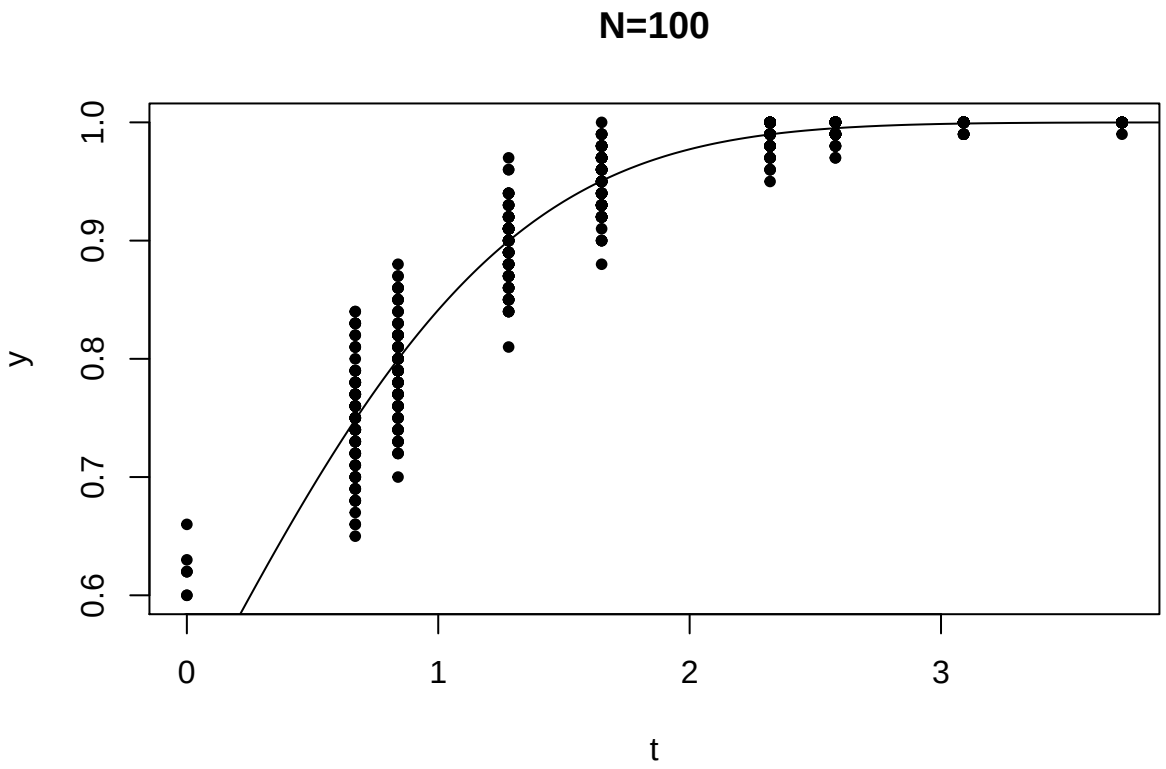
```
set.seed(1)
X <- rnorm(100, mean = 0, sd = 1)
knitr::kable(X)
```

x
-0.6264538
0.1836433
-0.8356286
1.5952808
0.3295078
-0.8204684
0.4874291
0.7383247
0.5757814
-0.3053884
1.5117812
0.3898432
-0.6212406
-2.2146999
1.1249309
-0.0449336
-0.0161903
0.9438362
0.8212212
0.5939013
0.9189774
0.7821363
0.0745650
-1.9893517
0.6198257
-0.0561287
-0.1557955
-1.4707524
-0.4781501
0.4179416
1.3586796
-0.1027877
0.3876716
-0.0538050
-1.3770596
-0.4149946
-0.3942900
-0.0593134
1.1000254
0.7631757
-0.1645236
-0.2533617
0.6969634

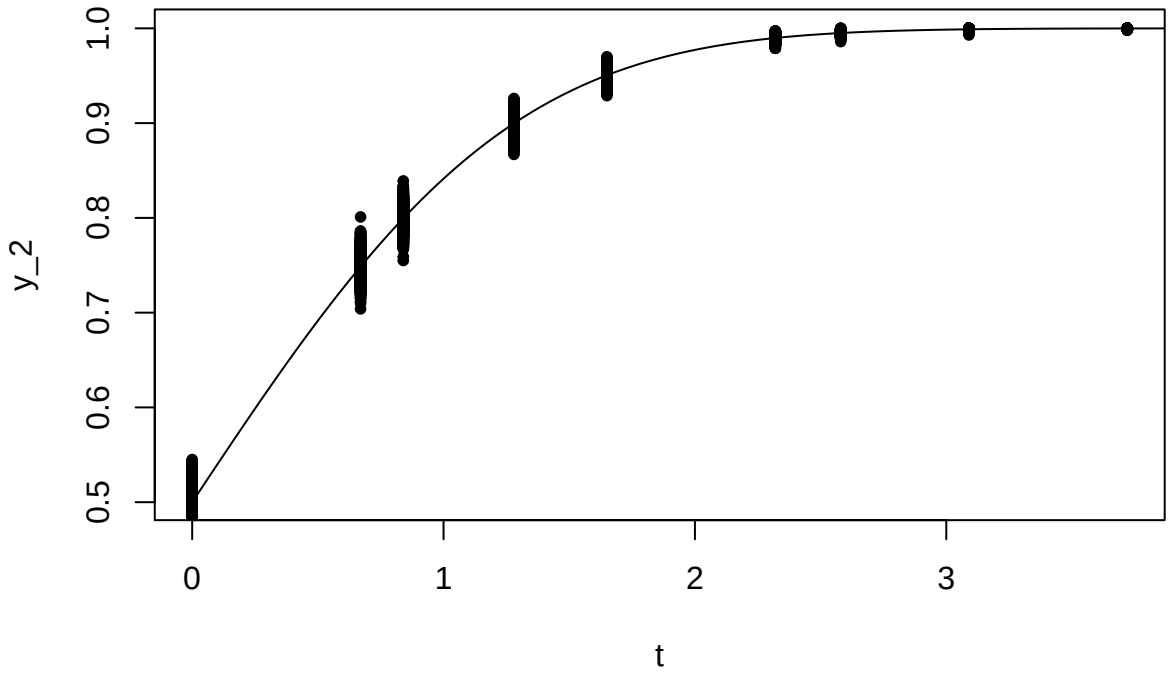
x
0.5566632
-0.6887557
-0.7074952
0.3645820
0.7685329
-0.1123462
0.8811077
0.3981059
-0.6120264
0.3411197
-1.1293631
1.4330237
1.9803999
-0.3672215
-1.0441346
0.5697196
-0.1350546
2.4016178
-0.0392400
0.6897394
0.0280022
-0.7432732
0.1887923
-1.8049586
1.4655549
0.1532533
2.1726117
0.4755095
-0.7099464
0.6107264
-0.9340976
-1.2536334
0.2914462
-0.4432919
0.0011054
0.0743413
-0.5895209
-0.5686687
-0.1351786
1.1780870
-1.5235668
0.5939462
0.3329504
1.0630998
-0.3041839
0.3700188
0.2670988
-0.5425200
1.2078678
1.1604026
0.7002136
1.5868335

x
0.5584864
-1.2765922
-0.5732654
-1.2246126
-0.4734006

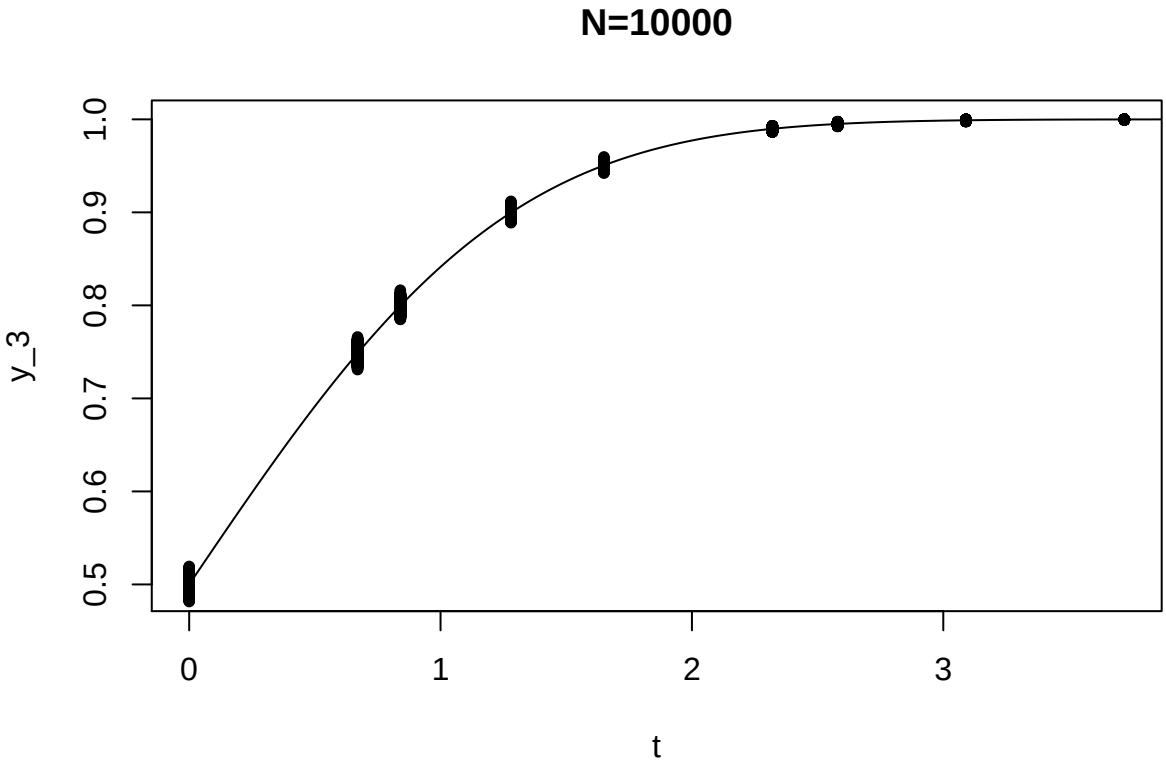
Figures




```
t <- c(0.0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72)
y_2 <- matrix(data = "numeric", nrow = 1000, ncol = 9)
X_2 <- rnorm(1000, mean = 0, sd = 1)
for (i in 1:9){
  y_2[1,i] <- 1/1000 * sum( I(X_2 <= t[i]) )}
plot(t, y_2[1, ], ylab = "y_2", main = "N=1000", pch = 20)
for (j in 2:1000){
  X_2 <- rnorm(1000, mean = 0, sd = 1)
  for (i in 1:9){
    y_2[j,i] <- 1/1000 * sum( I(X_2 <= t[i]) )}
  points(t, y_2[j, ], pch = 20)}
m <- seq(0,4,by = .02)
n <- pnorm(m, mean = 0, sd = 1)
lines(m,n)
```




```
t <- c(0.0, 0.67, 0.84, 1.28, 1.65, 2.32, 2.58, 3.09, 3.72)
y_3 <- matrix(data = "numeric", nrow = 10000, ncol = 9)
X_3 <- rnorm(10000, mean = 0, sd = 1)
for (i in 1:9){
  y_3[1,i] <- 1/10000 * sum( I(X_3 <= t[i]) )}
plot(t, y_3[1, ], ylab = "y_3", main = "N=10000", pch = 20)
for (j in 2:10000){
  X_3 <- rnorm(10000, mean = 0, sd = 1)
  for (i in 1:9){
    y_3[j,i] <- 1/10000 * sum( I(X_3 <= t[i]) )}
  points(t, y_3[j, ], pch = 20)}
m <- seq(0,4,by = .02)
n <- pnorm(m, mean = 0, sd = 1)
lines(m,n)
```



Note that, the lines in above figures are the probability line of $N(0,1)$

Conclusion

By the incresing of the N, the results of the Monte Carlo methods is closer to the standard normal distribution.