A Fast Algorithm for the Generation of Random Numbers with Exponential and Normal Distributions

Julio F. Fernández¹,² and Juan Rivero²

¹ Instituto Carlos I de Física Teórica y Computacional and Departamento de Física Aplicada, Universidad de Granada, 18071-Granada, Spain.
² Instituto Venezolano de Investigaciones Científicas, Apartado 21827, Caracas 1020A, Venezuela

Abstract: Algorithms for the generation of pseudorandom numbers with normal and exponential distributions are described here. No transcendental functions need to be evaluated; furthermore, only two uniform deviates per generation are required; no tables are used. These algorithms are much faster than other exponential and normal random number generators.

1. Introduction

Many simulations in computational physics require random numbers with a given distribution. Exponential and normal distributions are often needed. Existing random number generators are often inaccurate and they are time consuming. It has been shown recently that the inaccuracy issue can sometimes be reasonably serious [1]. Exponential and normally distributed random numbers are generated particularly slowly because one or more transcendental functions and/or several uniform deviates must be evaluated for each random number generated [2,3]. New algorithms for the generation of pseudorandom numbers with normal and exponential distributions are described in this lecture. No transcendental functions need to be evaluated; furthermore, only two uniform deviates per generation are required. Its accuracy is easy to control. No tables are used. It is much faster than other generators.

As part of an introduction for students who are unfamiliar with this subject, one of the simplest methods to generate exponentially distributed random numbers is explained next. Take a random number \( x \) (supplied, for instance by a built in generator in your computer), distributed uniformly in the interval \( (0, 1) \), that is, \( P(x) = 1 \), for \( 0 < x < 1 \), and \( P(x) = 0 \), for \( x < 0 \) and \( x > 1 \). There is a function \( y(x) \) that transforms the uniform deviate \( x \) into the desired exponential deviate \( y \); it must fulfill

\[
P(x)|dx| = P(y)|dy|
\]  

(1.1)
where \( P(y) \) is the desired exponential distribution \( (e^{-y}) \). Substitution of \( P(x) \) and \( P(y) \) into the above equation gives

\[
\left| \frac{dx}{dy} \right| = e^{-y} \tag{1.2}
\]

The solution to this equation is \( x = c \pm e^{-y} \), where \( c \) is an arbitrary constant. Since we know that \( x \) is within \((0, 1)\), it follows that either

\[
x = e^{-y} \tag{1.3}
\]

or

\[
x = 1 - e^{-y}. \tag{1.4}
\]

The first of these equations gives

\[
y = -\ln x \tag{1.5}
\]

which is the desired transformation. (The other solution, \( y = \ln(1 - x) \) entails an additional subtraction, and is therefore of little interest.) Unfortunately, this is a slow algorithm. It takes most computers about 10 times longer to compute a logarithm than it does to do simple arithmetic operations, such as sums, subtractions and multiplications.

Implementation of this idea for the generation of normal deviates turns out to be somewhat more complicated. Equation (1.2) gives

\[
\left| \frac{dx}{dy} \right| = \frac{1}{\sqrt{2\pi}}e^{-y^2/2} \tag{1.6}
\]

Unfortunately this expression integrates to an error function, which is not included in the list of functions given in most computer languages. This problem is avoided by the Box-Muller procedure, which is a generalization of the above method [3]. It requires the evaluation of a logarithm and a square root (it takes nearly as much computer time to compute a square root as a logarithm does).

There are other ways to generate exponential and normal deviates. We will not describe them here. We next list a few of them and state their main shortcomings. A method often used to generate exponentially distributed random numbers is the comparison method [4]. Its main drawback is that it requires, on the average, that one generate 4.3 uniform random deviates per exponential random number computed (it is therefore slow). One can also index a table of the desired distribution with a uniform random number. This method is not accurate, due to the necessarily coarse resolution of the table with respect to machine precision [2]. In Ref. [5], a method to generate random numbers by means of memory lookup is proposed and implemented in hardware; but this implementation is obviously not universally available.

In addition to the Box-Muller method (mentioned already), there exist several algorithms for the generation of normal deviates. The comparison method may be used [4]. As in the exponential case, it demands several uniform deviates per normal deviate. The sum method [3], which is based on the central limit