Algorithm for normal random numbers

Julio F. Fernández¹ and Carlos Criado²

¹Instituto de Ciencia de Materiales de Aragón
Consejo Superior de Investigaciones Científicas
and Universidad de Zaragoza, 50009-Zaragoza, Spain

²Departamento de Física Aplicada I, Universidad de Málaga
29071-Málaga, Spain

(March 24, 2022)

We propose a simple algorithm for generating normally distributed pseudo random numbers. The algorithm simulates \( N \) molecules that exchange energy among themselves following a simple stochastic rule. We prove that the system is ergodic, and that a Maxwell like distribution that may be used as a source of normally distributed random deviates follows in the \( N \to \infty \) limit. The algorithm passes various performance tests, including Monte Carlo simulation of a finite 2D Ising model using Wolff’s algorithm. It only requires four simple lines of computer code, and is approximately ten times faster than the Box-Muller algorithm.
Pseudo random number (PRN) generation is a subject of considerable current interest. Deterministic algorithms lead to undesirable correlations, and some of them have been shown to give rise to erroneous results for random walk simulations, Monte Carlo (MC) calculations, and growth models. Most of the interest has been focused on PRN’s with uniform distributions. Less attention has been paid to non-uniform PRN generation.

Sequences of random numbers with Gaussian probability distribution functions (pdf’s) are needed to simulate on computers gaussian noise that is inherent to a wide variety of natural phenomena. Their usefulness transcends physics. For instance, numerical simulations of economic systems that make use of so called geometric Brownian models (in which noise is multiplicative) also need a source of normally distributed PRN’s. There are several algorithms available for PRN’s with Gaussian pdf’s. Some, such as Box-Muller’s algorithm, require an input of uniform PRN’s, and their output often suffers from the pitfalls of the latter. Robustness is therefore a relevant issue. In addition, Box-Muller’s algorithm is slow and can consequently consume significant fractions of computer simulation times. The comparison method demands several uniform PRN’s per normal PRN, and is therefore also slow. Use of tables is not a very accurate method. Algorithms that are related, but not equivalent, to the one we propose here have been published, but they are somewhat cumbersome to use. In addition, no proof of their validity has been given.

We propose here a new algorithm for the generation of normally distributed PRN’s that is quite simple and fast. It is a stochastic caricature of a closed classical system of $N$ particles. Their velocities provide a source of PRN’s. We prove that, for any initial state, their pdf becomes Maxwellian in the $N \to \infty$ limit, after an infinite number of two-particle “collisions” take place. To this end, we first prove that our system is ergodic [14,15]. The end result is expected to take place in Statistical Physics, we expect that sufficient number of iterations will lead to an approximately Gaussian pdf of register values, from which the desired Gaussian pdf’s may be drawn. (See also Ref. [12] is not a very accurate method. Algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s is believed to make use of so called geometric Brownian models (in which noise is multiplicative) also need a source of normally distributed PRN’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exists a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed PRN’s [7]. There exist a large number of algorithms that require an input of uniform PRN’s, and their output of Gaussian pdf’s [8]. Some, such as Box-Muller’s algorithm, need a source of normally distributed
Clearly, \( p(v) \to C \exp(-v^2/2) \) in the \( N \to \infty \) limit, which is the desired result.

We prove below, in three stages, that \( P_n(v) \) does indeed become homogeneous over spherical surface \( S_{N-1} \), if \( N \geq 3 \), in the \( n \to \infty \) limit. We first prove \( P_n(v) \leftrightarrow P_n(u) \) as \( n \to \infty \) if \( v \) and \( u \) are related. [From here on, we say that points \( v \) and \( u \) are related if successive transformations (1-3) of \( v \) can lead to \( u \).] We then prove that the system’s “orbit” covers \( S_{N-1} \) densely [that is, that any point \( v \in S_{N-1} \) can be brought arbitrarily close to any other point \( u \in S_{N-1} \) by applying transformations (1-3) to \( v \) a sufficient number of times]. Then, the desired result follows easily. It may help to place the significance of the proof that follows into proper perspective to note that if in Eq. (1) \( j \to U_i [1, N] \) is replaced by \( j = i + 1 \ mod N \), the system becomes then non-ergodic, as can be easily checked numerically.

To start the proof, let kernel \( K(v, v') \) be defined by \( P_{n+1}(v) = \int K(v, v') P_n(v') \, dv' \), and let

\[
F_n = \int \{ P_{n+1}^2(v) - P_n^2(v) \} \, dv. \tag{0.5}
\]

Note first that \( F_n < 0 \) implies that \( P_{n+1}(v) \) is more uniform than \( P_n(v) \), in the sense that \( \int dv \, [P_{n+1}(v) - \overline{P}]^2 < \int dv \, [P_n(v) - \overline{P}]^2 \), where \( \overline{P} = 1/\int \, dv \). It follows from the definition of \( K(v, v') \) that

\[
F_n = \int \, dv \, [\int dv_1 K(v, v_1) P_n(v_1)]^2 - P_n^2(v). \tag{0.6}
\]

Making use of the detailed balance condition, \( K(v, v') = K(v', v) \), which our system satisfies, and the relation \( \int \, dv \, K(v, v') = 1 \), Eq. (0.6) can be cast into

\[
F_n = -\frac{1}{2} \int \, dv \int dv_1 \int dv_2 \, Q(v, v_1, v_2), \tag{0.7}
\]

where, \( Q = K(v, v_1) K(v, v_2)[P_n(v_1) - P_n(v_2)]^2 \). Therefore, in the \( n \to \infty \) limit, \( P_n(v) \) becomes constant over each set in \( S_{N-1} \) within which any two points \( v, u \) are related.

We now prove that the system’s orbit covers \( S_{N-1} \) densely. Let \( H_N \) be the group of transformations in \( N \) dimensions defined by Eqs. (1-3). We first show that any rotation in \( 3D \) that is, any element of \( SO(3) \) can be approximated arbitrarily close by elements of \( H_N \).

The proof is extended to higher dimensions by induction. Note first that \( H_2 \) does not belong to the set of finite rotation groups in \( 3D \) [that is, any element of \( SO(3) \)] can be approximated arbitrarily close by elements of \( H_3 \).

The proof is extended to higher dimensions by induction. Note first that \( H_2 \) does not belong to the set of finite rotation groups in \( 3D \) [that is, any element of \( SO(3) \)] can be approximated arbitrarily close by elements of \( H_3 \). The proof is extended to higher dimensions by induction. Note first that \( H_2 \) does not belong to the set of finite rotation groups in \( 3D \) [that is, any element of \( SO(3) \)] can be approximated arbitrarily close by elements of \( H_3 \). The proof is extended to higher dimensions by induction. Note first that \( H_2 \) does not belong to the set of finite rotation groups in \( 3D \) [that is, any element of \( SO(3) \)] can be approximated arbitrarily close by elements of \( H_3 \). The proof is extended to higher dimensions by induction. Note first that \( H_2 \) does not belong to the set of finite rotation groups in \( 3D \) [that is, any element of \( SO(3) \)] can be approximated arbitrarily close by elements of \( H_3 \).
ity density at $v$ is approximately $N^{-1} v^2 (3 - v^2/2)/2$ for very large $N$. (It is pointless to require this error to be too small since a PRN is expected to be generated beyond $x$ with a small probability $q$.) It then follows that $[\ln(M/v)]^2 \lesssim N\delta P/P$ must be satisfied by $N$. Thus, approximately $10^6$ registers are sufficient in order to generate as many as $10^{15}$ PRN’s, with a roughly 10% error in the probability for the largest PRN in the sequence. For results obtained from a sequence of $10^{10}$ PRN’s generated with 1024 registers, see Fig. 1.

FIG. 2. Number $n(v)$ of PRN’s within $v - \Delta v/2$ and $v + \Delta v/2$, for $\Delta v = 0.1$, starting from initial conditions $v_i = 1$, for all $i \in [1, N]$, after transformation (1-3) is iterated $2n_p N$ times (that is, after each register interacts, on the average, $2n_p$ times). The $\square$ and $\bullet$ stand for $n_p = 2, 10$, respectively, for $N = 1024$. The $\bigcirc$ and $\varnothing$ stand for $n_p = 2, 4,$ and 10 respectively, for $N = 1048576$. The two straight lines stand for $C \exp(-v^2/2)$ for two values of $C$.

Our algorithm must be applied a number $n_p N$ of times before it is ready for use unless all $v_i$ are initialized with “equilibrium” values (stored from some previous computer run). The distribution of all register values then evolves towards equilibrium, as illustrated in Fig. 2. Deviations from equilibrium are statistically insignificant for $n_p \gtrsim 2$ and $N = 1024$, and for $n_p \gtrsim 4$ and $N = 1048576$. Since $n_p$ is expected to increase as $\ln N$, $n_p = 8$ should provide ample warm up for any foreseeable applications.

The number of PRN’s that must be generated before each PRN in sequence $v_1, v_2, \ldots, v_N$ returns within distance $r$ from its initial value is exponential in $N$. More specifically, we estimate it to be $(\tau/\sqrt{N}) (1/r)^N$ for $N \gg 1$, where $\tau$ is the period of the algorithm used to select $i$ and $j$ in Eq. (1). The estimation is based on $P_n(v) \to \text{constant over } S_{N-1}$ as $n \to \infty$. Thus, an effectively infinite recurrence time follows for any reasonable value of $N$.

Correlations between a finite number of PRN’s clearly vanish as $N \to \infty$, since $i$ and $j$ in Eq. (1) are supposedly independent PRN’s. We have searched for correlations in $m$ successively generated PRN’s $v_1, v_2, \ldots, v_m$, for $m = 3, 4, \ldots, 6$, performing a chi-square isotropy test over the corresponding $m$-dimensional space. An $m$-tuple $v = (v_1, v_2, \ldots, v_m)$ was said to belong to the $i$-th cone, of 1024 randomly oriented cones with axes $w_1, w_2, \ldots, w_{1024}$, if $0.99 \leq v.w_i \leq 1$. No significant deviations from isotropy were observed for $10^6$ generated $m$-tuples.

Implementation of Wolff’s algorithm [19] in MC calculations of the Ising model’s critical behavior is a demanding test that some well known uniform PRN generators have failed [3]. Large clusters are then flipped as a whole, and this tests correlations in very long sequences. We have used normal PRN’s generated by our algorithm as input into a MC simulation of an Ising system of $16 \times 16$ spins at the critical temperature. [For that, we note that $v_1^2 + v_2^2 > 2x$ as often as $u > \exp(-x)$ if $v_1$ and $v_2$ ($u$) are PRN’s with Gaussian (uniform) pdf’s, respectively.]

The energy obtained is shown in Fig. 3 as a function of the number of registers $N$. The following uniform PRN algorithms were used to select $i$ and $j$ in Eq. (1): ggl [3], R(250,103,xor) [23], and Ran3 [24]. We tried the latter two algorithms, which have been shown to lead by themselves to unacceptable results for the Ising model [3], in order to test our algorithm’s robustness. The results shown in Fig. 3 are gratifying.

Similarly, the specific heat $c$ and magnetization $m$ fluctuations data points obtained follow approximately the relations $c \simeq c_0 + 8.4/N$, and $\langle m^2 \rangle \simeq \chi_0 + 33/N$, respectively, where $c_0 = 1.497(1)$ and $\chi_0 = 0.5454(2)$, in agreement with the known exact values [3] [21].
Double precision is recommended. It prevents excessive drift of the sum $\sum v_2 i$ away from its assigned value. Even then, single precision accuracy is to be expected at the end of a sequence of some $10^{16}$ PRN's, unless the sum is normalized several times during the run.

In summary, we have shown that implementation of Eqs. (1-3) provides a source of PRN’s with an approximately Gaussian pdf. Some $10^4$ registers (molecules) are sufficient for some purposes, but up to $10^5$ or more may be necessary for more demanding tasks. (Having to make a decision about the number of registers to be used may sometimes be an unwelcomed task. On the other hand, it is a virtue of the algorithm, that one can control, through the value of $N$, how close the output is to be from sequences of truly independent random numbers with Gaussian pdf’s.) Initial warm ups for arbitrary initial conditions are necessary; it is sufficient to let each register initially interact an average number of, say, 8 times. The system’s recurrence time was shown to be exponential in $N$, and therefore effectively infinite. Its behavior appears to be robust. The proposed algorithm runs an order of magnitude faster on computers than the most often used Box-Muller method. For a fortran code of our algorithm or other questions, please write JFF@Pipe.Unizar.Es.

Continuous help from Dr. Pedro Martínez with computer systems is deeply appreciated by JFF. We are indebted to Prof. P. Grassberger for an important suggestion. JFF and CC are grateful for partial financial support from DGICYT of Spain, through grants No. PB95-0797 and PB97-1080, respectively.

[1] For a general introduction to generation of uniform PRN sequences, see, D.E. Knuth, *The Art of Computer Programming*, Vol. 2, second edition (Addison-Wesley, Reading, MA, 1981); G. Marsaglia, in *Computer Science and Statistics: The Interface*, edited by L. Billard (Elsevier-North Holland, Amsterdam, 1985); P. L’Ecuyer, Ann. Oper. Res. 53, 77 (1994); H. Niederreiter, J. Comp. Appl. Math. 56, 159 (1994).

[2] P. Grassberger, Phys. Lett A 181, 43 (1993); L. N. Shchur, J. R. Heringa, and H. W. J. Blöte, Physica A, 241, 579 (1997).

[3] A. M. Ferrenberg, D. P. Landau, and Y. J. Wong, Phys. Rev. Lett. 69, 3382 (1992); I. Vattulainen, T. Ala-Nissila, and K. Kankaala Phys. Rev. Lett. 73, 2513 (1994); F. James, Europhys. Lett., 24, 24 (1993); L. N. Shchur and H. W. J. Blöte, Phys. Rev. B, 55, R4905 (1997).

[4] P. J. Resende and B. V. Costa, Phys. Rev. E 58, 5183 (1998).

[5] R. M. D’Souza, Y. Bar-Yam, and M. Kardar, Phys. Rev. E, 57, 5044 (1998).

[6] For old work, see N. Wax, *Noise and Stochastic Processes* (Dover Publications Inc., 1954); for a more recent general account, see, C.W. Gardiner, *Handbook of Stochastic Methods* (Springer-Verlag, Berlin, 1990).

[7] A. R. Bergstrom, Econometric Theory 13, 467 (1997); C. Rose, J. of Economic Dynamics and Control 19, 1391 (1997).

[8] See for instance, Devroye, *Non-Uniform Random Variate Generation* (Springer-Verlag, New York, 1986); D.E. Knuth, *The Art of Computer Programming*, Vol. 2, second edition (Addison-Wesley, Reading, MA, 1981) p. 117.

[9] See for instance, P. Bratley, B. L. Fox and L. E. Schrage, *A Guide to Simulation* (Springer-Verlag, New York, 1983) pp. 223-224.

[10] J. F. Fernández and J. Rivero, Comput. Phys. 10, 83 (1996).

[11] G.E. Forsythe, Math. of Comp. 26, 817, (1972).

[12] See, for instance, R. Toral and A. Chakrabarti, *Comput. Phys. Commun.* 74, 327 (1993) and references therein.

[13] C. S. Wallace, ACM Trans. Math. Software 22, 119 (1996).

[14] See, for instance, J. Lebowitz and O. Penrose, Physics Today 26, 23 (1973).

[15] V. I. Arnold and A. Avez, *Ergodic Problems of Classical Mechanics* (W. A. Benjamin, Inc., New York, 1968).

[16] H. S. M. Coxeter *Introduction to Geometry* (John Wiley and Sons, New York, 1969); A. I. Kostrikin, *Introduction to Algebra* (MacGraw-Hill, New York, 1992).

[17] See, for instance, R. Gilmore, *Lie groups, Lie algebras, and some of their applications* (John Wiley and Sons, New York, 1974), p. 209.

[18] H. Goldstein, *Classical Mechanics* (Addison-Wesley, Reading, 1959) pp. 124-132.

[19] U. Wolff, Phys. Rev. Lett. 62, 361 (1989).

[20] W. H. Press, B. P. Flannery, S.A. Teukolsky, and W. T. Vetterling, *Numerical Recipes* (Cambridge University Press, Cambridge, UK, 1989).

[21] A. E. Ferdinand and M. E. Fisher, Phys. Rev. 185, 832 (1969).