STRG #9

Statistical Techniques Research Group
Section of Mathematical Statistics
Department of Mathematics
Princeton University, Princeton, N.J.

FINE HALL LIBRARY
PRINCETON UNIVERSITY

Technical Report No. 9
February 1958

GENERATION OF RANDOM NORMAL DEVIATES ON LARGE-SCALE COMPUTERS

by

Mervin E. Muller

Department of Army Project No. 5B99-01-004
Ordnance R and D Project No. PB2-0001
OOR Project No. 1715
Contract No. DA 36-034-ORD 2297

FINE HALL LIBRARY
PRINCETON UNIVERSITY

SUMMARY

A new method for generating normal deviates is introduced. A rapid and reliable application for determining a normal deviate by solving the inverse relationship of the Normal Distribution Function is developed for use on a large binary computer with index registers. The necessary coefficients are given in Tables II, III, IV, and V. Section 3 contains a review of known methods for generating normal deviates. A comparison of the various methods is given in Section 4.

The new direct method gives higher accuracy than previous methods of comparable speed. The detailed inverse technique proposed yields accuracy comparable with, or better than most previous proposals using about one-quarter the computing time.

FINE HALL LIBRARY
PRINCETON UNIVERSITY

0. Introduction

Many applications of electronic computers, for example (10), (15), (17), (20) require the efficient generation of large numbers of random normal deviates.

Tables of random normal deviates are of course available, for example (21), (26), but they are not sufficiently extensive for many purposes and an outside source of this kind cannot usually be used effectively by the computer. What is required is some method of generation which can be rapidly carried out by the machine itself.

Methods are available by which pseudo random numbers may be produced, see for example (4), (8), (9), (13), (14), (16), (23), (25). Judging by the results, given for example by (8), (9), (20), (24), the most satisfactory procedure, now in use, for generating random numbers is the one based on "residue class" techniques, see for example (16), (23).

We shall not consider here the validity of employing deterministic methods for generating random numbers but assume that some machine method of satisfactorily producing random numbers is available from which random normal deviates are to be produced. A number of different ways of generating random normal deviates is known, for example (9), (24).

One purpose of the present paper is to give what is believed to be a new method for generating normal deviates due to Dr. G.E.P. Box and the author and to compare it with other methods. Secondly, though it is known that one can generate a normal deviate by first generating a uniform deviate, say U_i, from the unit interval $[0, 1]$ and then solving the inverse relationship for the normal deviate X_i (namely finding X_i from $U_i = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{X_i} e^{-t^2/2}\, dt$), and efficient and reliable application of this method is not available.

Consequently, this problem is considered in detail too. We also shall briefly mention the generation of other random variables.

1. Direct Method:

1.1 Method.

The following theorem may be used to generate a pair of independent random normal deviates from the same normal distribution starting from a pair of random numbers.

Theorem: Let U_1, U_2 be independent random variables from the same rectangular density function on the interval $[0, 1]$. Consider the random variables:

$$(I) \qquad \begin{aligned} X_1 &= (-2 \log_e U_1)^{1/2} \cos 2\pi U_2 \\ X_2 &= (-2 \log_e U_1)^{1/2} \sin 2\pi U_2 \end{aligned}$$

Then (X_1, X_2) will be a pair of independent random variables from the same normal distribution with mean zero, and unit variance.

Proof: From (I), (giving attention to principal values), one obtains at once the inverse relationships:

$$U_1 = e^{-(X_1^2 + X_2^2)/2},$$

$$U_2 = \frac{1}{2\pi} \arctan \frac{X_2}{X_1}.$$

It follows, after evaluating the Jacobian of the transformation, that the joint density function of X_1, X_2 is

$$f(X_1, X_2) =$$

$$\frac{1}{2\pi} e^{-(X_1^2 + X_2^2)/2} = \frac{1}{\sqrt{2\pi}} e^{-X_1^2/2} \cdot \frac{1}{\sqrt{2\pi}} e^{-X_2^2/2} = f(X_1)\, f(X_2);$$

thus the desired conclusions, including the independence of X_1 and X_2 follows.

The above result is motivated by the following considerations: The probability density of $f(X_1, X_2)$ is constant on circles, so $\theta = \arctan X_2/X_1$ is uniformly distributed $[0, 2\pi]$. Further, the square of the length of the radius vector $r^2 = X_1^2 + X_2^2$ has a Chi-squared distribution with two degrees of freedom. If U has a rectangular density on $[0, 1]$ then $-2 \log_e U$ has a Chi-squared distribution with two degrees of freedom. Proceeding in the reverse order we arrive at (1).

1.2 Generalizations.

Since X_1, X_2 are independent variables the technique suggested above can be used to generate n-dimensional normal deviates.

The usual transformation to obtain independent arbitrary n-variate normally distributed variables Y_1, Y_2 ..., Y_n with means μ_1, μ_2, ..., μ_n, variances σ_1^2, σ_2^2, ..., σ_n^2, and correlation matrix ρ follows at once by generating n-normal deviates and using a Jacobi-Toeplitz decomposition, see (26). For example, in the important case of the bivariate distribution, with means μ_1, μ_2, variances σ_1^2, σ_2^2 and correlation ρ, we have:

$$Y_1 = \mu_1 + \sigma_1 X_1 ,$$

$$Y_2 = \mu_2 + C_1 X_1 + C_2 X_2 ,$$

where $C_1 = \sigma_2 \rho$ and $C_2 = \sigma_2 (1 - \rho^2)^{1/2}$.

1.3 Convenience and Accuracy.

The method suggested here grew out of the desire to have a way of generating normal deviates which would be reliable in the tails of the distribution. Since most computing centers have library programs to compute values of trigonometric functions, logarithms, and square roots this approach requires little additional machine program writing. The accuracy obtainable here depends essentially on

the precision of the available library programs, whereas that of other methods cannot so readily be increased.

2. Inverse Method.

2.1 Introduction

In principle, the inverse method of generating a normal deviate X_1 from a uniform deviate U_1 is to find the inverse relationship $X = X(U)$ given that $U = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{X} e^{-t^2/2} dt$. The actual determination of $X(U)$ offers certain numerical difficulties when it is desired to generate reliable normal deviates, especially for large values of X. We have considered this problem for two aspects. First, speed and secondly and equally important, reliability. We have insisted that the maximum absolute error in X should be less than 4×10^{-4} in the range $-5 \leq X \leq 5$, where Prob $\{ -5 \leq X \leq 5 \} > 1 - 6 \times 10^{-7}$ so that X is correct to within 4×10^{-4} except for an event of probability less than 6×10^{-7}. The details of the procedure have been carried out for application on a large size binary machine with index registers, where it is possible to utilize memory space in order to obtain greater speed.

The relation $X = X(U)$ is approached in stages. The interval $[0, 1]$ for U is subdivided so that over each sub-interval it is possible to obtain a reliable and fast procedure for computing X. Over most of $[0, 1]$ $X = X(U)$ is approximated by Chebyshev polynomials. As X becomes large in absolute value it is necessary to increase the degree of the approximating polynomial. However, even though the degree of the polynomial increases, the frequency with which these approximations are needed decreases, hence this method will use, on the average, a low order approximating polynomial for $X = X(U)$. Due to symmetry it is actually only necessary to study $X = X(U)$ for $(1/2 \leq U \leq 1)$.

For speed in computing it is best to have the Chebyshëv polynomials of as low degree as possible, subject to the specified level of accuracy. Following this approach one could find sub-intervals of greatest or optimum width, see (6). Two drawbacks then become apparent. If on a binary computer the sub-interval widths are not some negative power of two, considerable computing time is spent in mapping a given sub-interval onto the range [- 1, 1] which is needed when using a Chebyshëv approximation. More important, if the intervals are not of uniform width and a negative power of two, considerable machine time, and memory space, is required to select the appropriate approximation. For, it will not then be possible to use an index register to select the appropriate approximation. Taking these facts into consideration, and taking into account memory space requirements, it was decided to have the widths of the sub-intervals equal $\frac{1}{128}$. With this choice of interval width the largest part of [0, 1], namely $\frac{1}{8} \leq U \leq \frac{7}{8}$, could be approximated by linear functions. Quadratic and quartic approximations are used for $\frac{7}{8} \leq U \leq \frac{127}{128}$. For ($\frac{127}{128} < U \leq 1$), and by symmetry ($0 \leq U \leq \frac{1}{128}$), $X = X(U)$ has a singularity of logarithmic type, consequently for U in this sub-interval an approximation of more subtle type than Chebyshëv polynomials is needed.

2.2 Use of Polynomials

Since the techniques used in approximating $X = X(U)$ have possible applicatic when considering the generation of other possible pseudo random variables we shall review the techniques in detail.

To approximate $X = X(U)$ in the jth sub-interval, ($\frac{64 + j - 1}{128} \leq U \leq \frac{64 + j}{128}$), $j = 1(1)64$, by Chebyshëv polynomials we shall make use of the technique of Interpolating Chebyshëv Polynomials as developed by C. Lanczos, see for example (11), (12). If a polynomial approximation is appropriate for a given sub-interval this approach will approximately minimize the maximum error of the approximation.

To begin, the jth sub-interval is transformed onto [- 1, 1] by using

$$r = 256\,U - 127 - 2j.$$

Let $r = \cos\theta$, $0 \le \theta \le \pi$. Let $T_i(r) = \cos[i(\cos^{-1} r)]$ denote the ith Chebyshëv polynomial. Thus for the jth sub-interval,

$$X = X_j(r) = 1/2\, d_{oj} + \sum_{i=1}^{\infty} d_{ij} T_i(r).$$

$X_j(r)$ will be approximated by

$$X_{nj}(r) = 1/2\, d_{oj} + \sum_{i=1}^{n} d_{ij}\, T_i(r),$$

where the error after using terms up through nth degree is,

$$\epsilon_{nj}(r) = X_j(r) - X_{nj}(r) = \sum_{i=n+1}^{\infty} d_{ij}\, T_i(r).$$

If convergence is sufficiently rapid $\epsilon_{nj}(r)$ is estimated by $d_{n+1,j}\, T_{n+1}(r)$. Consequently we seek the set, $\{ r_k \; k = 0, 1, 2, \ldots, n \}$, such that $T_{n+1}(r_k) = 0$, or equivalently, θ_k's such that $\cos(n+1)\,\theta_k = 0$. This in turn requires that $\theta_k = \frac{\pi}{2}\left(\frac{2k+1}{n+1}\right)$. Thus $r_k = \cos\frac{\pi}{2}\left(\frac{2k+1}{n+1}\right)$.

The coefficients d_{ij} are then obtained from,

$$d_{ij} = \frac{2}{n+1} \sum_{k=0}^{n} X(r_k)\, T_i(r_k),$$

$$= \frac{2}{n+1} \sum_{k=0}^{n} X(r_k) \cos\frac{i\pi}{2}\left(\frac{2k+1}{n+1}\right).$$

In carrying out the evaluations for $X_j(U_k)$ $k = 0, \ldots, n$, for the jth interval it is necessary to evaluate $X = X(U)$ for U_{kj}'s where

$$U_{kj} = \frac{r_k + 127 + 2j}{256}.$$

The values of $X(U_{kj})$ were obtained by cubic inverse interpolation in the National Bureau of Standard Tables of the Normal Probability Function, (22). These values are included as a table in the Appendix.

We then obtain,

$$X_{nj}(r) = 1/2\, d_{oj} + \sum_{i=1}^{n} d_{ij}\, T_i(r).$$

However, for computing purposes it is more convenient to express the $T_i(r)$'s as polynomials in r, see Lanczos (11), (12), for example $T_1(r) = r$, $T_2(r) = 2r^2 - 1$. One then obtains

(1) $X_{nj}(r) = \sum_{i=0}^{n} a_{ij} r^i$. The corresponding coefficients a_{ij} are given in section 2.3

For the linear cases the essential computations for the a_{ij}'s are concerned with finding the inverse values, namely $X = X(U_{kj})$. However, for the linear cases the a_{ij}'s given in section 2.3 were not obtained in this manner since we had previously needed to have values of X corresponding to $U_s = \frac{128 + s}{256}$, $s = 0, 1, 2, \ldots, 128$. Consequently we avoided the additional labour of finding the necessary X's by utilizing the available $X(U_s)$'s. Thus, while not fitting the straight lines by first degree Chebyshëv polynomials we were able to obtain the desired level of precision as follows: In the jth interval, where it is appropriate to fit a straight line, consider U_j, U_{j+1}, and $U_{j,\frac{1}{2}} = \frac{U_j + U_{j+1}}{2}$. Let X_j, X_{j+1}, $X_{j,\frac{1}{2}}$ denote the corresponding X values. The jth line is fitted such that the deviations at the ends of the class interval are equal and such that the deviations at the mid-point, namely $U_{j,\frac{1}{2}}$, is equal in magnitude to the deviations at the end points, but is of opposite sign. This approach gives essentially a Chebyshëv-type approximation in that we are striving to minimize the maximum error in a given sub-interval. By straightforward computations one then obtains that

$$a_{oj} = \frac{2(X_{j+1} + X_{j,\frac{1}{2}}) - (X_{j+1} - X_j)(255 + 4j)}{4},$$

$$a_{ij} = 128(X_{j+1} - X_j).$$

In the selection of the width of a sub-interval, w_j, or for determining the appropriate degree of the Chebyshëv Polynomial for a given sub-interval, the following result is used, see (6) for more general details,

$$w_j = 4\left[\frac{(n+1)!}{2}\,\epsilon\,\left|X^{(n+1)}(c_j)\right|^{-1}\right]^{\frac{1}{n+1}}$$

where ϵ is the largest absolute error that is tolerable, in our case $\epsilon = 4 \times 10^{-4}$, and where $X^{(n+1)}(U)$ is the (n+1)st derivative of $X = X(U)$ with respect to U.

As a result it was necessary to be able to express $X^{(n+1)}(U)$ in terms of the derivatives of U with respect to X. Recall that,

$$U = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{X} e^{-t^2/2}\,dt, \quad \frac{dU}{dX} = \frac{1}{\sqrt{2\pi}}e^{-X^2/2} = Q(X).$$

Though the derivations for the derivatives of the inverse function $X = X(U)$ are an immediate application of implicit differentiation they are included below since they are rather tedious to obtain.

TABLE I

Derivatives of the Inverse Function $X = X(U)$

Order of Derivative	
1.	$[Q(X)]^{-1}$
2.	$[Q(X)]^{-2}X$
3.	$[Q(X)]^{-3}(2X^2 + 1)$
4.	$[Q(X)]^{-4}(6X^3 + 7X)$
5.	$[Q(X)]^{-5}(24X^4 + 46X^2 + 7)$
6.	$[Q(X)]^{-6}(120X^5 + 326X^3 + 127X)$
7.	$[Q(X)]^{-7}(720X^6 + 2{,}556X^4 + 1{,}740X^2 + 127)$
8.	$[Q(X)]^{-8}(5{,}040X^7 + 22{,}212X^5 + 22{,}404X^3 + 4{,}369X)$
9.	$[Q(X)]^{-9}(40{,}320X^8 + 212{,}976X^6 + 290{,}292X^4 + 102{,}164X + 4{,}369)$
10.	$[Q(X)]^{-10}(362{,}880X^9 + 2{,}240{,}344X^7 + 3{,}890{,}484X^5 + 2{,}080{,}644X^3 + 243{,}649X)$.

2.3 Table of Coefficients

The following tables provide the necessary coefficients for the approximations given by equation (1) of section 2.2 .

TABLE II

Linear Cases

j	a_{oj}	a_{1j}
1	-1.25339 449	2.50678 851
2	-1.25388 344	2.50775 044
3	-1.25487 723	2.50967 688
4	-1.25639 365	2.51257 300
5	-1.25845 203	2.51644 669
6	-1.26107 329	2.52130 843
7	-1.26428 016	2.52717 152
8	-1.26809 727	2.53405 216
9	-1.27255 126	2.54196 946
10	-1.27767 105	2.55094 571
11	-1.28348 794	2.56100 644
12	-1.29003 594	2.57218 075
13	-1.29735 187	2.58450 135
14	-1.30547 572	2.59800 495
15	-1.31445 096	2.61273 256
16	-1.32432 485	2.62872 974
17	-1.33514 882	2.64604 704
18	-1.34698 984	2.66475 763
19	-1.35986 540	2.68485 495
20	-1.37390 816	2.70651 014
21	-1.38914 726	2.72973 042
22	-1.40567 396	2.75461 642
23	-1.42357 635	2.78126 053
24	-1.44295 134	2.80976 489
25	-1.46390 578	2.84024 264
26	-1.48655 769	2.87281 914
27	-1.51103 773	2.90763 359
28	-1.537749 093	2.94484 078
29	-1.56607 868	2.98461 325
30	-1.59698 101	3.02714 356
31	-1.63039 945	3.07264 746
32	-1.66656 010	3.12136 699
33	-1.70571 765	3.17357 468
34	-1.74815 989	3.22957 838
35	-1.79421 327	3.28972 698
36	-1.84424 956	3.35441 736
37	-1.89869 401	3.42410 298
38	-1.95803 516	3.49930 401
39	-2.02283 720	3.58062 028
40	-2.09375 508	3.66874 687
41	-2.17155 347	3.76449 389
42	-2.25713 085	3.86881 137
43	-2.35154 997	3.98282 083
44	-2.45607 734	4.10785 604
45	-2.57223 456	4.24551 603
46	-2.70077 415	4.39645 556
47	-2.84833 659	4.56815 521
48	-3.01223 276	4.75713 372
49	-3.19590 941	4.96703 562
50	-3.40829 673	5.20760 458
51	-3.65403 652	5.48350 516
52	-3.93953 404	5.80124 828
53	-4.27256 248	6.16869 364
54	-4.67044 630	6.60394 516
55	-5.15058 375	7.12472 205
56	-5.74560 472	7.76467 397

TABLE III

Quadratic Cases

j	a_{oj}	a_{1j}	a_{2j}
57	1.56668 859	-0.03343 48405	.00087 5575
58	1.63732 538	-0.03745 15701	.00114 804
59	1.71722 812	-0.04284 26652	.00157 546
60	1.80989 233	-0.05049 00254	.00230 549
61	1.92135 077	-0.06226 30013	.00372 027
62	2.06352 790	-0.08299 31005	.00708 977

TABLE IV

Quartic Cases

j	a_{oj}	a_{1j}	a_{2j}	a_{3j}	a_{4j}
63	+2.26622 681	+0.12757 931	+0.01844 432	+0.00424 42872	+0.00104 0

2.4 The Interval $(\frac{127}{128} \leq U < 1)$.

In view of the fact that X(U) may be looked upon as a function having a singularity of logarithmic type in this interval, it is necessary to abandon the use of a polynomial type approximation here. A satisfactory rational approximation is obtained by using a truncated continued fraction expansion, see for example (7), (19). For any given value of U, the corresponding value of X is approximated by the following recurrence relation. The k^{th} convergant to X(U) is given by

$$X_k(U) = \frac{M_k(U)}{N_k(U)},$$

where $M_k(U)$ and $N_k(U)$ are determined as follows:

(1) $M_{k+1}(U) = d_k M_k(U) + (U - U_{k-1})M_{k-1}(U)$; $M_o(U) = 1$; $M_1(U) = d_o$;

(2) $N_{k+1}(U) = d_k N_k(U) + (U - U_{k-1})N_{k-1}(U)$; $N_o(U) = 0$; $N_1(U) = 1$;

and where d_k denotes a selected value of the k^{th} inverted divided difference of X(U)

TABLE V

Constants for Rational Approximation

k	U_k	d_k
0	.99223 97464	+.24200 000 x 10^{+1}
1	.99461 38540	+.18262 366 x 10^{-1}
2	.99653 30262	-.65518 108 x 10^{0}
3	.99813 41867	+.23997 757 x 10^{-1}
4	.99903 23968	-.26737 146 x 10^{0}
5	.99931 28620	+.16541 263 x 10^{-1}
6	.99966 30707	-.14194 984 x 10^{0}
7	.99984 08914	+.99732 778 x 10^{-2}
8	.99992 76519	-.60049 158 x 10^{-1}
9	.99996 83287	+.51181 541 x 10^{-2}
10	.99998 6654251	-.23299 296 x 10^{-1}
11	.99999 4587456	+.24107 770 x 10^{-2}
12	.99999 7887545	-.86334 192 x 10^{-2}
13	.99999 9206672	+.10076 316 x 10^{-2}
14	.99999 97133484	-.30828 145 x 10^{-2}

In order to insure sufficient precision it is suggested that the machine program test to see if $U > .99999$. When U is greater than $.99999$ it is suggested that the additional significant figures for U_k, $k = 10(1)14$ be utilized. Double precision operations are not necessary if these U_k are stored with the first three nines suppressed; U must then be shifted the appropriate number of places before performing $U - U_k$.

Following (7), page 406, $X_k(U)$ can be computed from

$$(3)\quad X_k(U) = d_o + \sum_{n=1}^{k} \frac{(-1)^{n+1}(U - U_o)(U - U_1) \ldots (U - U_{n-1})}{N_n(U)\, N_{n+1}(U)}$$

To insure that $X(U)$ can be approximated to within 4×10^{-4} it is necessary to stop at an appropriate value of $k = k(U)$ since the approximation being employed is essentially a divergent expansion. Numerical evaluations verified that it is sufficient to have the machine program terminate as soon as one of the following three conditions is satisfied, namely: (i) select $X_k(U)$ as the approximation to $X(U)$ if $r_k = \left| X_k(U) - X_{k-1}(U) \right| < 4 \times 10^{-4}$, (ii) select $X_k(U)$ as the approximation to $X(U)$ if $r_k - r_{k+1} < 0$, and (iii) if $k = 14$ terminate the process and select $X_{14}(U)$ as the approximation to $X(U)$.

The value of k increases as U increases in the interval $(\frac{127}{128} < U < 1 - 3 \times 10^{-7})$, and consequently so does the computing time. However, this sub-interval will, on the average, increase the necessary computing time very little. It should be kept in mind that while this method is very appropriate and convenient for a machine which performs "floating point" multiplication, this approach would create serious scaling difficulties for a "fixed point" mode of operation.

3. Review of Other Methods.

All the methods to be considered have the common feature that they require the availability of independent and uniformly distributed deviates U.

3.1 Central Limit Approach

By appealing to the central limit theorem of probability, e.g. (3), we can know that sums of an arbitrary number of U's will be asymptotically normally distributed. Though this method is easy to use, poor results are obtained in the tails of the distribution, see for example (10). Inspection of the following table illustrates this point, the table was obtained by computing selected values of the distribution function of the sum of twelve uniform deviates and comparing these values with the correct values for the normal distribution function.

TABLE VI

Normal deviates exceeded with certain probabilities compared with sums of twelve uniform deviates exceeded with the same probabilities (means and variances equated

Sum of Twelve Deviates	Probability of a larger deviate	Normal deviates with same mean and variance with same probability	Difference of Deviates
0.0000	0.500000	0.0000	0.0000
0.2000	0.421711	0.1975	0.0025
0.4000	0.346338	0.3952	0.0048
0.6000	0.276483	0.5933	0.0067
0.8000	0.214180	0.7920	0.0080
1.0000	0.160727	0.9915	0.0085
1.2000	0.116639	1.1912	0.0088
1.4000	0.817077×10^{-1}	1.3937	0.0063
1.6000	0.557457×10^{-1}	1.5969	0.0031
1.8000	0.357846×10^{-1}	1.8018	-0.0018
2.0000	0.222756×10^{-1}	2.0089	-0.0089
2.2000	0.132681×10^{-1}	2.2183	-0.0183
2.4000	0.754029×10^{-2}	2.4304	-0.0304
2.6000	0.407497×10^{-2}	2.6458	-0.0458
2.8000	0.208611×10^{-2}	2.8648	-0.0648
3.0000	0.100700×10^{-2}	3.0882	-0.0882
3.2000	0.455824×10^{-3}	3.3165	-0.1165
3.4000	0.192173×10^{-3}	3.5505	-0.1505
3.6000	0.748223×10^{-4}	3.7917	-0.1917
3.8000	0.266137×10^{-4}	4.0410	-0.2410
		4.3004	-0.3004

The importance of extreme or tail values for particular applications can be seen for example in the material presented in (1).

3.2 Rejection Approach

For some distributions rejection-type techniques, see for example (2), (10), (18), are acceptable, however for the normal distribution this approach is very inefficient, especially if precise tail values are important.

This technique is attributed to von Neumann. The presentation follows that in (24). One proceeds to generate normal deviates in the truncated region $-b \leq X \leq b$ as follows: Generate uniform deviates U_1 and U_2. Each time compute $Y = -2b^2 (U_1 - \frac{1}{2})^2$. If $\log_e U_2 \leq Y$ then the value $X = b(2U_1 - 1)$ is used as a normal deviate.

If $U_2 > Y$ then one rejects the pair (U_1, U_2) and repeate the above process. The inefficiency of the process can be appreciated by realizing that the probability that a pair (U_1, U_2) will be used to generate a normal deviate, namely $\text{Prob}\{ U_2 \leq e^{-2b^2(U_1 - \frac{1}{2})^2} \}$, equals $\int_0^1 e^{-2b^2(U - \frac{1}{2})^2} dU$, which is asymptotically $\frac{1}{b}\sqrt{\frac{\pi}{2}}$.

3.3 Hastings' Approach

Pade-type or rational approximations to transform a uniform deviate to a normal deviate have been suggested by several people. The best known version is due to C. Hastings (5), page 192. Using this approach one obtains a normal deviate X from a uniform deviate $U = q$ as follows:

$$X = X(q) = \eta - \left(\frac{a_o + a_1\eta + a_2\eta^2}{1 + b_1\eta + b_2\eta^2 + b_3\eta^3} \right)$$

where

$$\eta = \sqrt{\ln 1/q^2}, \quad q = \frac{1}{\sqrt{2\pi}} \int_{X(q)}^{\infty} e^{-(1/2)t^2} \quad 0 < q \leq .5,$$

and

$$a_o = 2.515517 \qquad b_1 = 1.432788$$
$$a_1 = 0.802853 \qquad b_2 = 0.189269$$
$$a_2 = 0.010328 \qquad b_3 = 0.001308$$

Except for certain important sub-intervals such as $(1.48 \leq X \leq 2.42)$, the absolute value of the error is less than 4×10^{-6} and even on these intervals the absolute values of the error is less than 6×10^{-4}.

3.4 Teichroew's Approach

A fixed number of uniform deviates is summed, then using an interpolating Chebyshev polynomials an improved approximate normal deviate is obtained. The complete details have been obtained by Teichroew (24). Teichroew calls this the method of "Approximation by Curve Fitting". He proceeds to find a normal deviate as follows:

It is desired to find $y = m(\theta)$ where

$$\int_{-\infty}^{y} \frac{1}{\sqrt{2\pi}} e^{-t^2/2}\, dt = \int_{o}^{\theta} \phi_\gamma(t)\, dt ,$$

and where ϕ_γ is the density function of the sum of γ uniform deviates. This approach requires that the range of θ be restricted. For the restricted range $\theta_L \leq \theta \leq \theta_U$, $y = m(\theta)$ is approximated by determining an interpolating polynomial. A Chebyshëv polynomials of degree $k - 1$ is fitted so that its values coincide with the values of $m(\theta)$ at the k roots of the Chebyshëv polynomial of degree k.

Though Teichroew has obtained the coefficients of the Chebyshëv polynomials for $\gamma = 6$, 8 and 12, we shall describe the case $\gamma = 12$ since this value of γ gives the most satisfactory results. As before U_i denotes a uniform deviate and let $\theta = \sum_{i=1}^{12} U_i$ so $0 \leq \theta \leq 12$. However, θ must be restricted, here Teichroew chooses $\theta_L = 2$, $\theta_U = 10$, i.e., $2 \leq \theta \leq 10$ or $-1 \leq \frac{\theta - 6}{4} \leq 1$. (Note: Prob $\{\theta_L \leq \theta \leq \theta_U\} \geq 1 - (2)10^{-5}$), so there is a very small probability of being outside this range. His program was prepared for S.W.A.C., and arranged so that if θ fell outside (2, 10) the machine would halt, and then type out this fact. Let $r = \frac{\theta - 6}{4}$. Then an approximate normal deviate X is obtained as $X = \sum_{j=0}^{9} d_{2j+1} T_{2j+1}(r)$.

For machine computation it is not economical or convenient to find X in the above form, a rearranged series in r is obtained. If we truncate the series after the term $T_9(r)$, this series will be

$X = a_1 r + a_3 r^3 + a_5 r^5 + a_7 r^7 + a_9 r^9$, where the coefficients are as follows:

$a_1 = 3.949846138$

$a_3 = 0.252408784$

$a_5 = 0.076542912$

$a_7 = -0.008355968$

$a_9 = 0.029899776$

4. Some Comparisons.

Though the following timing and memory space considerations may be valid for other computers they have been derived for a floating point binary machine with index registers, in this case the IBM 704. Further, the timing and memory space requirements have been evaluated subject to existing library subroutines which are available for the 704 through the SHARE organization. It is possible that some computing time or memory space could be saved if specialized function subroutines were written, however the present approach has the advantage that if a given sampling, or Monte Carlo problem requires some of the same library function subroutines they will be already available in the memory. The memory space requirements indicated below include the necessary function subroutines. For each method a normal deviate is formed for use as a "floated" normal deviate.

Method	Time per deviate in miliseconds	Precision in units of X	Precision except for probability less than	Memory Space (reusable temporary locations)
Inverse	(Average) = 1.395 (Standard deviation) = 1.261 (87.5% of cases) $\leq$ 0.996	4×10^{-4}	6×10^{-7}	202(4)
Sum of 12 uniform deviates	5.052	See Table VI	See Table VI	25(4)
Direct	6.601	5×10^{-7}	4×10^{-8}	175(7)
Teichroew's $\gamma = 12$ $(\gamma = 8)$	 6.948 6.278	 2×10^{-4} 2×10^{-4}	 2×10^{-5}	 46(4) 49(4)
Hastings'	6.968	6×10^{-4}	4×10^{-8}	104(8)
Rejection	(Average) = 16.360 (Standard deviation) = 28.201	5×10^{-7}	6×10^{-7}	76(5)

5. Other Random Deviates.

The occasion sometimes arise for random deviates other than the normal deviates, in particular the one which can be generate from the normal deviates, for example deviates from the F-distribution, Chi-squared, and t-distribution.

Observations from the Chi-squared distribution with $2k$ degrees of freedom can of course be generated by adding together the k terms, $\sum_{i=1}^{k} (-2 \log U_i)$ and for Chi-squared with $2k + 1$ degrees of freedom one may add the square of a normal deviate generated by the above method. Deviates from the F-distribution and for the t-distribution may be obtained by calculating the appropriate ratio of deviates generated as above.

Acknowledgements.

The author wishes to express his appreciation to Dr. G.E.P. Box and Dr. John W. Tukey for their interest and generous comments concerning this work. The author is also indebted to Mrs. A. Schay for her excellent help with the numerical computations.

References:

1. G.E.P. Box and S.L. Andersen, "Permutation Theory in the Derivation of Robust Criteria and the Study of Departures from Assumption", J. Royal Stat. Soc., Series B v XVII(1955), p. 1-34.

2. James W. Butler, "Machine Sampling from Given Probability Distributions", Proc. Gainesville Monte Carlo Symposium, p. 249-264, Wiley, 1956.

3. H. Cramer, Mathematical Methods of Statistics, Princeton University Press, 1946.

4. George E. Forsythe, "Generation and Testing of Random Digits at The National Bureau of Standards, Los Angeles", Monte Carlo Method, National Bureau of Standards, AMS 12, 1951, p. 34-35.

5. C. Hastings, Approximations for Digital Computations, Princeton University Press, 1955.

6. J. O. Harrison, "Piecewise Polynomial Approximations for Large Scale Digital Calculations", MTAC, v. III, (1949), p. 400-407.

7. F. B. Hildebrand, Introduction to Numerical Analysis , Mc Graw - Hill, 1956.

8. D. L. Johnson, "Generating and Testing Pseudo Random Numbers on the IBM Type 701", MTAC, v. X, (1956), p. 8-13.

9. M. L. Juncosa, "Random number generation of the BRL High-Speed computing machines", Ballistic Research Laboratories, Aberdeen Proving Ground, Maryland, Report 855, 1953.

10. Herman Kahn, "Applications of Monte Carlo", RAND Project Report, 1954.

11. C. Lanczos, "Trigonometric Interpolation of Empirical And Analytical Functions", J. Math. Phys., v. 17, (1938), p. 123-199.

12. C. Lanczos, Tables of Chebyshëv Polynomials $S_n(X)$ and $C_n(X)$, National Bureau of Standards, AMS 9 (1952).

13. D. H. Lehmer, "Mathematical Methods in Large Scale Computing Units", Proc. Second Symposium on Large-Scale Digital Calculating Machinery, (1949) p. 141-146. Harvard Univ. Press, Cambridge, Mass., 1951.

14. D. H. Lehmer, Mathematical Reviews, v. 15, (1949), p. 559.

15. Monte Carlo Methods, National Bureau of Standards, AMS 12, (1951) p. 34-35.

16. J. Moshman, "The Generation of Pseudo-Random Numbers on a Decimal Calculator", J. Assoc. for Computing Machinery, v. 1 (1954) p. 88-91.

17. M. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem", Ann. Math. Stat., v. 27, (1956) p. 569-589.

18. J. von Neumann, "Various Techniques Used in Connection with Random Digits", Monte Carlo Method, National Bureau of Standards, AMS 12, (1951) p. 36-38.

19. N. E. Norlund, Vorlesungen Über Differenzenrechnung, Springer, 1924.

20. Proc. Symposium on Monte Carlo Methods, Gainesville, 1954. Wiley 1956.

21. The RAND Corp., One Million Random Digits and 100,000 Normal Deviates, The Free Press, Glencoe, Illinois, 1955.

22. Tables of Normal Probability Functions, National Bureau of Standards, AMS 23, (1953).

23. O. Taussky and J. Todd, "Generation and Testing of Pseudo-Random Numbers", Proc. Gainesville Monte Carlo Symposium, p. 15-27, Wiley, 1956.

24. D. Teichroew, "Distribution Sampling with High Speed Computers", Ph. D. Thesis, Univ. of North Carolina, 1953.

25. D. F. Votaw and J. A. Rafferty, "High Speed Sampling", MTAC v. 5 (1951) p. 1-8.

26. H. Wold, Random Normal Deviates, Tracts for Computers No. XXV Cambridge Univ. Press, 1948.

Appendix

This table gives values of X(U) corresponding to the Normal Distribution for $F(U_j) = \frac{1}{2} + \frac{j}{256}$, $j = 1(1)\ 127$. The values were obtained from the National Bureau of Standard Tables of the Normal Probability Function, (22), by using the inversion formula:

$$X(U) = X_o + \frac{a}{2Q_o} + \frac{X_o}{2}\left(\frac{a}{2Q_o}\right)^2 + \frac{(2X_o^2 + 1)}{6}\left(\frac{a}{2Q_o}\right)^3,$$

where U_o is the nearest tabulated entry to U, and $U - U_o = a$.

TABLE VII

Inverse Values for the Normal Distribution

j	$F(x_j) = \frac{1}{2} + j/256$	x_j	j	$F(x_j) = \frac{1}{2} + j/256$	x_j
1	0.50390625	0.00979167	26	0.60156250	0.25739353
2	0.50781250	0.01958429	27	0.60546875	0.26752821
3	0.51171875	0.02937878	28	0.60937500	0.27769044
4	0.51562500	0.03917609	29	0.61328125	0.28788143
5	0.51953125	0.04897716	30	0.61718750	0.29810241
6	0.52343750	0.05878294	31	0.62109375	0.30835463
7	0.52734375	0.06859437	32	0.62500000	0.31863936
8	0.53125000	0.07841241	33	0.62890625	0.32895791
9	0.53515625	0.08823802	34	0.63281250	0.33931161
10	0.53906250	0.09807215	35	0.63671875	0.34970180
11	0.54296875	0.10791578	36	0.64062500	0.36013003
12	0.54687500	0.11776987	37	0.64453125	0.37059729
13	0.55078125	0.12763542	38	0.64843750	0.38110545
14	0.55468750	0.13751340	39	0.65234375	0.39165587
15	0.55859375	0.14740482	40	0.65625000	0.40225007
16	0.56250000	0.15731068	41	0.66015625	0.41288960
17	0.56640625	0.16723201	42	0.66406250	0.42357608
18	0.57031250	0.17716982	43	0.66796875	0.43431116
19	0.57421875	0.18712516	44	0.67187500	0.44509652
20	0.57812500	0.19709908	45	0.67578125	0.45593392
21	0.58203125	0.20709265	46	0.67968750	0.46682512
22	0.58593750	0.21710695	47	0.68359375	0.47777199
23	0.58984375	0.22714306	48	0.68750000	0.48877641
24	0.59375000	0.23720211	49	0.69140625	0.49984034
25	0.59765625	0.24728522	50	0.69531250	0.51096581

TABLE VII - continuation -

j	$F(x_j) = \frac{1}{2} + j/256$	x_j	j	$F(x_j) = \frac{1}{2} + j/256$	x_j
51	0.69921875	0.52215488	81	0.81640625	0.90175411
52	0.70312500	0.53340971	82	0.82031250	0.91655667
53	0.70703125	0.54473251	83	0.82421875	0.93156283
54	0.71093750	0.55612559	84	0.82812500	0.94678176
55	0.71484375	0.56759132	85	0.83203125	0.96222320
56	0.71875000	0.57913216	86	0.83593750	0.97789754
57	0.72265625	0.59075066	87	0.83984375	0.99381591
58	0.72656250	0.60244945	88	0.84375000	1.00999017
59	0.73046875	0.61423129	89	0.84765625	1.02643306
60	0.73437500	0.62609901	90	0.85156250	1.04315826
61	0.73828125	0.63805558	91	0.85546875	1.06018048
62	0.74218750	0.65010407	92	0.85937500	1.07750557
63	0.74609375	0.66224768	93	0.86328125	1.09518065
64	0.75000000	0.67448975	94	0.86718750	1.11319428
65	0.75390625	0.68683375	95	0.87109375	1.13157656
66	0.75781250	0.69928330	96	0.87500000	1.15035938
67	0.76171875	0.71184220	97	0.87890625	1.16953661
68	0.76562500	0.72451438	98	0.88281250	1.18916435
69	0.76953125	0.73730400	99	0.88671875	1.20926123
70	0.77343750	0.75021538	100	0.89062500	1.22984876
71	0.77734375	0.76325304	101	0.89453125	1.25099172
72	0.78125000	0.77642176	102	0.89843750	1.27268865
73	0.78515625	0.78972652	103	0.90234375	1.29502241
74	0.78906250	0.80317257	104	0.90625000	1.31801090
75	0.79296875	0.81676542	105	0.91015625	1.34171784
76	0.79687500	0.83051088	106	0.91406250	1.36620382
77	0.80078125	0.84441508	107	0.91796875	1.39153749
78	0.80468750	0.85848447	108	0.92187500	1.41779714
79	0.80859375	0.87272589	109	0.92578125	1.44507258
80	0.81250000	0.88714656	110	0.92968750	1.47345903