MATHEMATICS OF COMPUTATION
VOLUME 45, NUMBER 172
OCTOBER 1985, PAGES 559–568

Accuracy in Random Number Generation*

By John F. Monahan

Abstract. The generation of continuous random variables on a digital computer encounters a problem of accuracy caused by approximations and discretization error. These in turn impose a bias on the simulation results. An ideal discrete approximation of a continuous distribution and a measure of error are proposed. Heuristic analysis of common methods for transforming uniform deviates to other continuous random variables is discussed. Comments and recommendations are made for the design of algorithms to reduce the bias and avoid overflow problems.

1. Introduction. In stochastic simulation and Monte-Carlo integration, a given function h is integrated with respect to a probability distribution F by obtaining random variables X_i, $i = 1,\ldots,n$, independently and identically distributed (iid) with distribution function (df) F and applying the strong law of large numbers to the mean

$$\lim n^{-1}\sum h(X_i) = \int h(x)\,dF(x) \quad \text{a.s.}$$

The procedure is straightforward and intuitive; moreover, any reasonable behavior of h can be tolerated. For integration in many dimensions, the curse of dimensionality can be avoided and a convergence rate of $O_p(n^{-1/2})$ is typical [2, Chapter 5].

This theory, however, must be tempered with the reality of digital computation. Floating-point arithmetic uses a finite set of numbers to approximate the real line. Hence, all random variables obtained on a digital computer must have a discrete distribution whose support is limited to the number system defined by the floating-point arithmetic. As a result, the intended distribution F is approximated by G, the df of the generated variables X_i. The sample mean of $h(X_i)$ is then converging to $\int h\,dG$, producing a bias that depends both upon the function h and upon the accuracy of the approximation G to F, whether the algorithm used to generate was theoretically exact or approximate.

In the past, the effects of approximations of continuous distributions have been ignored. Since the statistical accuracy of a mean is $O_p(n^{-1/2})$, a small bias cannot be detected unless the replication size is very large. With the plummeting cost of computation comes the possibility of an astronomical number of replications. Thus, the problem of accuracy must be addressed.

Received July 27, 1984.

1980 *Mathematics Subject Classification*. Primary 65C10; Secondary 65G10.

Key words and phrases. Random number generation, discretization error, approximation error, relative accuracy, floating-point arithmetic.

*An earlier version of this paper was presented at the Joint National Meeting of ORSA/TIMS, New Orleans, Louisiana, April, 1979.

0025-5718/85 $1.00 + $.25 per page

In the next section, best discrete approximations and the measures of error are presented. In Section 3, the problem of the discrete approximation of the uniform distribution and the effect of approximation error on integration error are discussed. Transformations are analyzed in Section 4. Heuristic analysis of some common algorithms for generating random variables is given in Section 5. Comments and recommendations are made in Section 6.

An important point must be made before continuing. In this paper, the randomness of pseudorandom number generators is never to be an issue. An inexhaustible source of random bits (or discrete uniform integers) will be assumed. Methods of approximating the uniform distribution on an interval of the real line, however, will be discussed in Section 3.

2. The Ideal Approximation and Measure of Error. The problem to be addressed here is the best approximation of a continuous distribution on the real line with a discrete distribution on the discrete set of numbers S arising from floating-point arithmetic to represent the real line. The best approximation will of course depend on the criterion; several will be proposed. In each case, an approximating distribution G or a measure of error between G and F (the intended) is proposed.

Proposal A. Approximate F with G such that $G(x) = F(x)$ for all x in S.

Although this appears to be quite reasonable, Proposal A is really arbitrary. The predominant convention is that distribution functions are continuous on the right, so $F(x) = \Pr(X \leqslant x)$. The approximant G is thus stochastically larger than F. Alternatively, a left-continuous df can be defined as $\Pr(X < x)$, [10], yielding a stochastically smaller approximant.

Proposal B. Measure error by the Kolmogorov-Smirnov distance,

$$e_B(F, G) = \sup|F(x) - G(x)|.$$

The primary difficulty with Proposal B is that it is sensitive only to the greatest difference. A distribution G that attains the same greatest difference has the same distance as another, whether the extreme is achieved many times or only at one point. The second problem is that the best approximation denoted by G^* will put mass on a point x_i in S that is half of the mass in the interval formed by its neighbors:

$$G^*(x_i) - G^*(x_{i-1}) = \tfrac{1}{2}\{F(x_{i+1} - 0) - F(x_{i-1})\}.$$

No matter how tightly a probability mass is spread around a point x_i, it will share half of the mass with its neighbors.

Proposal C. Let r: $R^1 \to S$ denote the function that produces the closest point in S to a real number x in R^1. Then if X has df F, then $X^* = r(X)$ is the ideal approximation.

Proposal C is fundamental to rounding arithmetic systems and will serve as the definition of the *ideal approximation*. The ideal approximation for F will be denoted by F^*.

An algorithm for generating the ideal approximation for the uniform distribution on $(0, 1)$ is given in Section 3. Ideal approximations to certain other distributions are possible.

Remark 1. The order statistics from an ideal approximation are ideal for the order statistics of the intended distribution.

Remark 2. The exponential distribution can be generated ideally using the von Neumann algorithm [14] and a tie-breaking scheme [11, p. 1066].

Proposal D. Measure error with the L_1 distance between probability densities $\int |f - g|\, dx$ or between probability functions $\Sigma|f_i - g_i|$.

This measure was proposed by Devroye [3] and is compatible with Proposal C. It is sensitive to how often an error is made, in contrast to the K-S error. However, it is not sensitive to how far. To illustrate, let S be the integers and let F put mass equally on $1, 2, 3, 4$. Let G_1 put mass $0, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}$ and G_2 put mass $0, \frac{1}{4}, \frac{1}{4}, \frac{1}{2}$ on those four integers, respectively. Although G_1 is obviously closer to F, it is just as far from F as G_2, as measured by Proposal D.

Proposal E. Measure error by $e(F, G)$ given by

$$e(F, G) = \int_{x_{-\infty}}^{x_{+\infty}} |F(x) - G(x)|dx,$$

where $x_{-\infty}$ and $x_{+\infty}$ are the first and last points in S.

The error given in Proposal E can be viewed as measuring the probability mass that is errant and the distance to be moved to correct, giving a sense of work to $e(F, G)$. The integral is truncated at the endpoints of S because F may not admit a mean and the behavior of F outside the range of S should have no effect on the approximant G. This measure of error is the one adopted in this paper and will be used subsequently.

Knuth and Yao [8] have investigated the problem of generating a random variable with a given distribution with the fewest random bits used as input. Clearly, the ideal approximation of any distribution can be obtained with enough effort. Only in special cases, as with the von Neumann algorithm for the exponential, will an algorithm giving the ideal approximation be in common use. Given the value of simplicity in algorithms for generating random numbers, the current algorithms will not be supplanted.

3. The Uniform Distribution and Integration Bias. The uniform distribution on the interval $(0, 1)$ is the fundamental distribution from which all other distributions can be obtained. On a computer using normalized floating-point rounded arithmetic with d digits in base b, the fundamental interval is not $(0, 1)$, but actually $[b^{-1}, 1)$, due to the representation of a number in normalized form (that is, no leading zero digits unless the number is zero). As a result, the fundamental distribution to be generated is uniform on $[b^{-1}, 1)$.

Generating the ideal approximation to the uniform $(0, 1)$ distribution can be viewed as having two steps. First, generate a random variable uniformly on $[b^{-1}, 1)$; then generate a random exponent with the geometric distribution to cover all of $[0, 1]$. Underflows in Step 5 below are set to zero and not flagged. The output of the following algorithm has mean $\frac{1}{2}(1 - b^{-2E-2})$ and variance $\frac{1}{12} + O(b^{-2d})$, where E represents the limit on the exponent. The error can be computed

$$e(F, F_A) = \tfrac{1}{4}b^{-2d}(1 - b^{-2E})/(1 - b^{-2}).$$

Algorithm A. Generate a uniform [0, 1] deviate

(1) Generate a random integer M from the discrete uniform distribution on $[b^{d-1}, b^d - 1]$.

(2) With probability $\frac{1}{2}$, add one to M, yielding M^*. (Notice that M^* is ideal for uniform (b^{d-1}, b^d).)

(3) Make a floating point W from M^* divided by b^d.

(4) Generate J with the geometric distribution with parameter $(1/b)$ so $\Pr(J = j) = b^{-j}(1 - 1/b)$ for $j = 0, 1, 2, \ldots$.

(5) Deliver $X = W^* b^{-J}$ as the desired deviate.

The common algorithm (call it Algorithm B) for generating the uniform distribution is to actually generate a random integer in $[0, b^d - 1]$ and then divide it by b^d and make it a floating-point number. This approximating distribution for the uniform then has a mean of $(1 - b^{-d})/2$ and $(1 - b^{-2d})/12$ for the variance; the error from Proposal E is $b^{-d}/2$. The errors made at this step lead to difficulties when these variates are used to generate from other distributions, which will be discussed later. Algorithm A differs from Algorithm B in two ways: normalization and centering. The usual practice is stochastically smaller than it should be and neglects the new digits created when the number is normalized because of leading zero digits.

The centering in step 2 of Algorithm A is distinctive and yields a distribution for W that is equivalent to a trapezoidal integration rule on $[b^{-1}, 1]$ with $b^{d-1}(b - 1)$ subintervals, in contrast to the Algorithm B left-handed Riemann sum. These characterizations allow the use of standard techniques [2, Chapter 2] to analyze the bias of Monte-Carlo integration of a function h using an approximation G to a distribution F:

$$\text{bias} = \int h(x)\, dF(x) - \int h(x)\, dG(x).$$

If the function h to be integrated is differentiable and vanishes for large $|x|$, then the error measure from Proposal E can be used to bound the bias

$$\left| \int h(x)\, dF(x) - \int h(x)\, dG(x) \right| \leqslant e(F, G) \sup |h'(x)|.$$

Since the floating-point arithmetic used here satisfies

$$|r(x) - x| \leqslant U|x|,$$

where $U = \frac{1}{2} b^{1-d}$ is the common unit of relative accuracy [1, p. 45], then an expression for the bias for an ideal approximation can be obtained:

$$\left| \int h(x)\, dF(x) - \int h(x)\, dF^*(x) \right| \leqslant \int |h(r(x)) - h(x)| dF(x)$$

$$\leqslant U \int |x h'(x^*)| dF(x),$$

where x^* lies between $r(x)$ and x. This expression reinforces intuition suggesting that the best place for a function h to behave its worst is near zero, where the number system is most accurate.

The result above relating relative accuracy can be used to obtain the inequality $|F(x) - F^*(x)| \leqslant U|x| f(x)$ in order to obtain an approximation to the error of the

ideal approximation

$$e(F, F^*) \leqslant U\int |x|f(x)\,dx = UE\{|X|\}$$

indicating obvious problems that confront approximating a distribution that does not admit a mean.

4. Transformations. The simplest method for generating a random variable from a distribution G is to transform a uniform deviate X by $H(X)$, where H is the inverse of G or its equivalent. Although the probability integral transform is seen in some quarters to solve the entire problem of random number generation, these transformations can illustrate the fundamental computational problems that motivate this paper. First of all, writing G^{-1} merely subsumes but does not solve the difficult problem of computing functions using arithmetic operations: additions, multiplications, etc. In most of the important cases, an approximation of G^{-1} is really computed and it embodies an approximation error, as well as roundoff error. This will be analyzed first, to be followed by the analysis of the effect of the variation in accuracy in floating-point arithmetic.

For the moment, ignore all other sources of error, except for the approximation of $H(x)$ by $\hat{H}(x)$ with a *relative error* of w. Then, if F represents the df of X then, considering only the worst case,

$$\Pr(\hat{H}(X) \leqslant y) = \Pr((1 + w)H(X) \leqslant y) = F(H^{-1}\{y/(1 + w)\}),$$

we can compute the error in $G = F(H^{-1})$ as

$$e(G_H, G_{\hat{H}}) = \int |G(y) - G(y/(1 + w))|\,dy = w/(1 + w).$$

If instead of relative error, we deal with *absolute error*, then we can compute

$$e(G_H, G_{\hat{H}}) = \int |G(y) - G(y + w)|\,dy = w.$$

Now assume that the computation of H is exact, in the sense that the computed value y is the floating-point representation of the number $H(x)$ for an x in S; that is, $y = r(H(x))$. Also assume that the input variable X is generated according to the ideal discrete approximation of F in S, and that the target distribution for Y is G. Then the discretization error is due to the varying degrees of accuracy in the representation of the domain variable x and in the range variable y. We can view H as a mapping from S to S (or subsets) that is sometimes one-to-one, sometimes many-to-one. There may be points in S that are in $r(\{\text{support of } G\})$, but are not points of $H(\{\text{support of } F^*\})$. I will refer to this as a consequence of H being a one-to-many function, in that a set of consecutive points are mapped into a set of points whose "closure" (adding intermediate points skipped) is much larger in number. To illustrate, in Figure 1 is given the ideal approximation of the uniform distribution on (0, 1) in one-digit base 10 arithmetic, with exponents only -1, 0, 1, and the biggest number is 10. The third column gives the probability masses of this distribution under the transformation $H(x) = 2x - 1$, so that the target distribution G is uniform $(-1, +1)$. The point $x = \frac{1}{2}$ is mapped to zero and there are no points in the domain that can reach the neighboring points in the interval $(-.1, .1)$. In most

x_i	ideal uniform	H_1	ideal exponential	H_2	H_3
−1.		.025			
−.9		.05			
−.8		.075			
−.7					
−.6		.1			
−.5					
−.4		.1			
−.3					
−.2		.1			
−.1					
−.09					
−.08					
−.07					
−.06					
−.05					
−.04					
−.03					
−.02					
−.01					
0.	.005	.1	.005	.05	.005
.01	.01		.01		.01
.02	.01		.01		.01
.03	.01		.009		.01
.04	.01		.01		.01
.05	.01		.01		.01
.06	.01		.009		.01
.07	.01		.009		.01
.08	.01		.009		.01
.09	.01		.01		.01
.1	.055		.049	.1	.055
.2	.1	.1	.082	.1	.1
.3	.1		.074		
.4	.1	.1	.067	.1	.1
.5	.1		.061	.1	.1
.6	.1	.1	.055		
.7	.1		.050	.1	.1
.8	.1	.1	.045		
.9	.1		.041	.1	.1
1.	.05	.05	.164	.1	.1
2.			.141	.165	.2
3.			.052	.05	
4.			.019	.02	
5.			.007	.01	
6.			.0026		
7.			.00095		
8.			.00035		
9.			.0001		
10.			.00007	.005	.05

FIGURE 1

Probability functions of some discrete approximations of continuous distributions.

other regions, the transformation H is one-to-two. Let this ratio of the mapping H be represented by R such that $R = \frac{1}{2}$ if H is locally two-to-one and $R = 2$ if locally one-to-two, then we can write it as

$$R(x) = s(x)|H'(x)|/s(H(x)),$$

where $s(x)$ measures the length of the interval $r^{-1}(\{x\})$ of points that are rounded to x. Notice that for $x = \frac{1}{2}$, we have $(.1)(2)/(.01) = 20$, which is really the case for $H(x) = 2x - 1$. For a simple multiplication, $H(x) = hx$, then for most values of x the ratio is

$$R(x) = s(x)|h|/s(hx) = U|x|\,|h|/U|xh| = 1.$$

The error can then be computed as the distance a probability mass is moved to correct $(R^2(x) - 1)/4s(H(x))\Pr(X = x)$ so that an approximation can be written as

$$\hat{e}(G_H) = U/4\int_{\{R(x)\geq 1\}} \left[\left(xH'(x)/H(x)\right)^2 - 1\right] H(x)\,dF(x).$$

The contribution from the region where $R(x) \leqslant 1$ is negligible. This approximation works reasonably well, except for $H(x)$ near the origin, where $s(0)$ is not 0, but depends on the limit on the exponent.

Example 1. $\hat{e}(G_H)$ is infinite for $H_1(x) = 2x - 1$.

Example 2. $\hat{e}(G_H)$ is infinite for $H_2(x) = -\log_e(x)$.

Example 3. $\hat{e}(G_t)$ is infinite for $H_3(x) = -\log_e(1 - x)$.

Example 4. $R(x) < 1$ for $H_4(x) = x^{1/2}$.

Example 5. $R(x) = 1$ for $H_5(x) = 1/x$.

The first three examples are not recommended methods. For generating a random variable uniformly on $(-1, +1)$, it is best to take a uniform $(0, 1)$ variate and attach a random sign. As has been stated before, the von Neumann algorithm can generate an ideal exponential deviate, so that neither H_2 nor H_3 are recommended. It is true that the approximation exaggerates any errors near the origin. However, since high accuracy should be expected there, errors commited there deserve severe punishment. In the case of H_3, however, a few points below one are not able to cover a large part of the real line.

Although H_4 may be a recommendable transformation, the distribution of the maximum of two uniforms can be generated ideally (Remark 1 and also [8, p. 364ff]).

5. Heuristics. There are four general methods for generating random variables with a target distribution F using a sequence of uniform $(0, 1)$ deviates. One of these methods, transformations, has just been discussed. Analysis of the other methods, acceptance-rejection, ratio of uniforms, and Forsythe's algorithm, bogs down before any conclusions are available. Consequently, I will only make some heuristic comments, summarizing some of the analysis of these methods.

Acceptance-rejection is the most common method and is the most difficult to analyze. Obviously, any errors made in generating the envelope distribution G cannot be recovered, so that it is important to generate it ideally if possible. The only other place for improvement is in the comparison of the uniformly distributed Y with the ratio $cf(X)/g(X)$. When this ratio is small, generating Y ideally on $(0, 1)$ will do the best possible. However, a ratio near unity is best, but the comparison will be less accurate, unless computation of the complement is easy. In any case, generating X from G is most important; the comparison is secondary.

The ratio of uniforms method [7] does a good job of generating from distributions with infinite support. If the region C_f lies in a box and both U and V are generated ideally, then the output variable $X = V/U$ should be close to the ideal. Here, the region near the origin can be covered by small values of V, and the tails by small values of U. As with acceptance-rejection, the comparison should have only a secondary effect on accuracy.

In the Forsythe algorithm [4], [5], [11], the most important steps are in computing the initial random variable X as well as possible and to avoid loss of accuracy in using the transformation G, following the same type of analysis as in Section 4. Since the comparisons are done with uniform deviates, this step is relatively easy.

6. Implementations. The practical question yet to be answered is what problems can be encountered at what replication size. Following the recommendations given here and sound numerical analysis, there are two types of problems: integration bias and overflow interrupts. The replication size at which adjustments must be made is different for the two cases.

If the recommendations in Sections 3, 4, and 5 are followed, then the error in random number generation should be simply a scale factor times that of the ideal distribution. Thus, the results for the integration bias given at the end of Section 3 can be applied. In general, this means that the bias should be on the order of the machine unit U. To detect such a bias, the replication size needs to be $O(U^{-2})$, with the scale factor determined by the variance. Ignoring those two important scale factors, for 6 digit, base 16 rounded arithmetic, such as on an IBM mini- or mainframe, $U = 10^{-6}$, so that 10^{12} replications would be needed to detect such an error. Obviously, those two scale factors could reduce that by orders of magnitude.

Certain types of experiments can produce large scale factors. Importance sampling experiments can be very sensitive to tail behavior, as well as statistical studies of robustness. Computing critical values of a test by Monte-Carlo is a formidable problem statistically and at least as sensitive to tail behavior as the test itself. Generating normal random variables by summing 12 uniforms and subtracting 6 is not a good algorithm in terms of accuracy or speed. It may perform well in some applications. In the analysis of the range of a sample as an estimator of scale, it will be a poor approximation which will require only a moderate number of replications to encounter problems.

The second problem looming in Monte-Carlo studies is the possibility of an interrupt due to overflow. Unlike other areas of numerical analysis, this can not always be avoided through discipline and good grooming. If we intend to generate random variables with a stable distribution, say, the probability of obtaining a variable larger than the largest representable number may not be infinitesimal in theory. A method that generates an overflow is not in principle incorrect.

In practice, the questions are what are the probabilities and how to trap errors. If the probabilities are extremely small, then an error trap would be unnecessary computation. Consider H_2 for generating the exponential, or the general ratio of uniforms method. If we use Algorithm B for generating the uniform variable, then the probability of an overflow is b^{-d}, which is much too large when I once generated 5,000,000 exponentials (using H_2) on an IBM machine in a single experiment.

However, if the uniforms are generated ideally, that probability drops to 10^{-163} on the same machine.

Let me summarize my recommendations:

(1) Generate the uniform and exponential distributions ideally.

(2) Use random bits to assign a random sign, avoid $2x - 1$.

(3) Avoid transformations whose error estimates are large.

(4) Avoid or trap overflows whose probability is U. A preferred level is b^{-E}; U^2 may be acceptable.

(5) Generate double-precision random variables when the replication size reaches $U^{-1/3}$ to be safe.

I do not expect the reader to rerun any experiments after reading this paper. The replication size is seldom large enough to encounter any of the difficulties mentioned here. However, I do anticipate problems as the cost of computation allows the replications to grow and grow.

How did I avoid any overflow with my 5,000,000 exponentials? If I had followed Algorithm B precisely, the probability of an overflow would have been .258 for each of the four experiments. However, I had used Schrage's [13] implementation of the Lewis, Goodman, Miller [9] algorithm, which actually generates integers from 1 to $2^{31} - 1$ so I had no possibility of obtaining zero or one. Gentle [6, p. 160] argues that a subprogram to generate a uniform (0, 1) pseudorandom variable should never give either a zero or a one. The main concern is using F^{-1} for a distribution with infinite support. I would argue that using F^{-1} for values of the argument near one is bad numerical analysis. For example, the Odeh and Evans [12] algorithm for the inverse of the normal df is properly designed to accept a value on $(0, \frac{1}{2})$ as input. If in this situation, a random variable having the ideal uniform $(0, \frac{1}{2})$ distribution is generated and a random sign attached later, the probability of an overflow would be acceptably small and the tails would be generated well.

Finally, some simulation studies do not use a mean as an estimator, especially for queuing systems. Consequently, the analysis given here based on the law of large numbers and the central limit theorem do not apply, although the recommendations given here should prove just as useful.

Department of Statistics
North Carolina State University
Raleigh, North Carolina 27695-8203

1. G. DAHLQUIST & A. BJÖRCK, *Numerical Methods* (N. Anderson, transl.), Prentice-Hall, Englewood Cliffs, N. J., 1974.

2. P. J. DAVIS & P. RABINOWITZ, *Numerical Integration*, Academic Press, New York, 1975.

3. L. DEVROYE, "A note on approximations in random variate generation," *J. Statist. Comput. Simulation*, v. 14, 1982, pp. 149–158.

4. U. DIETER & J. H. AHRENS, "A combinatorial method for the generation of normally distributed random numbers," *Computing*, v. 11, 1973, pp. 137–146.

5. G. FORSYTHE, "Von Neumann's comparison method for random sampling from the normal and other distributions," *Math. Comp.*, v. 26, 1972, pp. 817–826.

6. J. E. GENTLE, "Portability considerations for random number generation," *Comp. Science and Statist.: Proc. 13th Sympos. Interface* (W. F. Eddy, ed.), Springer-Verlag, Berlin and New York, 1981, pp. 158–164.

7. A. J. KINDERMAN & J. F. MONAHAN, "Computer generation of random variables using the ratio of uniform deviates," *ACM Trans. Math. Software*, v. 3, 1977, pp. 257–260.

8. D. E. KNUTH & A. C. YAO, "The complexity of nonuniform random number generation," in *Algorithms and Complexity, New Directions and Recent Results* (J. F. Traub, ed.), Academic Press, New York, 1976, pp. 357–428.

9. P. A. W. LEWIS, A. S. GOODMAN & J. M. MILLER, "A pseudorandom number generator for the System/360," *IBM Systems J.*, v. 8, 1969, pp. 136–146.

10. M. LOEVE, *Probability Theory*, 4th ed., Springer-Verlag, Berlin and New York, 1977.

11. J. F. MONAHAN, "Extensions of Von Neumann's method for generating random variables," *Math. Comp.*, v. 33, 1979, pp. 1065–1069.

12. R. E. ODEH & J. O. EVANS, "Algorithm AS 70: Percentage points of the normal distributions," *Appl. Statist.*, v. 23, 1974, pp. 96–97.

13. L. SCHRAGE, "A more portable Fortran random number generator," *ACM Trans. Math. Software*, v. 5, 1979, pp. 132–138.

14. J. VON NEUMANN, "Various techniques used in connection with random digits," in *Monte Carlo Method*, Nat. Bur. Standards Appl. Math. Series, v. 12, 1951, pp. 36–38.