MATHEMATICS OF COMPUTATION, VOLUME 33, NUMBER 147
JULY 1979, PAGES 1065–1069

Extensions of von Neumann's Method for Generating Random Variables

By John F. Monahan*

Abstract. Von Neumann's method of generating random variables with the exponential distribution and Forsythe's method for obtaining distributions with densities of the form $e^{-G(x)}$ are generalized to apply to certain power series representations. The flexibility of the power series methods is illustrated by algorithms for the Cauchy and geometric distributions.

1. Introduction. Von Neumann's [7] ingenious algorithm for generating deviates from the exponential distribution using uniformly distributed deviates was extended by Forsythe [5]. The novelty of both of these methods is that apparently "transcendental" distributions can be generated via purely arithmetic operations. The efficiency of these methods, due to their avoidance of logarithm computations, has been demonstrated in implementations for several important distributions [1]–[4].

In Section 2, a generalization of the von Neumann algorithm to a family of distributions with a power series representation is presented. The method is not restricted to absolutely continuous distributions; discrete (or any mixture) distributions are achievable. In Section 3, the power series generalization is applied to the Forsythe algorithm and illustrated in algorithms for the Cauchy and geometric distributions in Section 4.

2. Generalization of von Neumann's Algorithm. Let $Y_1, Y_2, \ldots$ be iid random variables from an arbitrary distribution with *df* F, i.e. $Pr(Y_j \leqslant y) = F(y)$. Let $Z_1 \equiv 1$ and $Z_2, Z_3, \ldots$ be independent Bernoulli deviates with $Pr(Z_i = 1) = a_i/a_{i-1} = p_i$ for $i > 1$ and given constants $1 = a_1 \geqslant a_2 \geqslant \cdots \geqslant 0$. Consider the following algorithm:

Algorithm V.

1. Generate Y_1, Z_1, $n \leftarrow 1$.
2. Generate Y_{n+1}, Z_{n+1}.
3. If $Y_{n+1} \leqslant Y_1$ and $Z_{n+1} = 1$, then $n \leftarrow n + 1$ and go to 2.
4. If n is even, go to 1.
5. Deliver $X \leftarrow Y_1$.

Let E_n be the event $\max(Y_1, \ldots, Y_n) = Y_1$ and $Z_1 = Z_2 = Z_3 = \cdots = Z_n = 1$.

Received October 10, 1977; revised March 24, 1978.
AMS (MOS) subject classifications (1970). Primary 65C10.
Key words and phrases. Random number generation, von Neumann's comparison method.

0025-5718/79/0000-0114/$02.25

Then $Pr(Y_1 \leqslant y, E_n) = a_n F(y)^n$ and since $E_{n+1} \subset E_n$,

$$Pr(Y_1 \leqslant y, E_n, E_{n+1}^c) = Pr(Y_1 \leqslant y, E_n) - Pr(Y_1 \leqslant y, E_{n+1})$$
$$= a_n F(y)^n - a_{n+1} F(y)^{n+1}.$$

Hence, $Pr(\text{Accept } Y_1) \equiv p_A = Pr(n \text{ is odd}) = \Sigma_{n=1}^{\infty} a_n (-1)^{n+1}$ and X has df

$$F_X(X) = Pr(X \leqslant x) = \sum_{n=1}^{\infty} a_n F(x)^n (-1)^{n+1} / p_A = h(-F(x))/h(-1),$$

where $h(w) = \Sigma_{n=1}^{\infty} a_n w^n$. The expected number of Y's needed to produce a single X, denoted by EN_F, can be computed

$$EN_F = p_A^{-1} \left[\sum_{n=1}^{\infty} Pr(E_n, E_{n+1}^c)(n+1) \right]$$
$$= p_A^{-1} \left[\sum_{n=1}^{\infty} (n+1)(a_n - a_{n+1}) \right] = \left(1 + \sum_{n=1}^{\infty} a_n \right) \Big/ p_A$$
$$= [1 + h(1)] / [-h(-1)].$$

In the von Neumann algorithm, $h(w) = e^w - 1$, $F(x) = x$, hence $F_X(x) = (1 - e^{-x})/(1 - e^{-1})$, $p_A = 1 - e^{-1}$ and $EN_F = e/(e-1)$. Note that for iid random variables, the ranks and order statistics are independent; hence, the random variable Z_j's can be obtained from the Y_j's: $Z_j = 1$ if $Y_j < Y_{j-1} < \cdots < Y_1$ which occurs with probability $1/j$ given E_{j-1}, hence $a_j = 1/j!$. This is obviously peculiar to the exponential distribution. Note that if F has atoms, the ordering of the Y_j's may no longer be unique but this tie-breaking scheme TB can adjust and yield a deviate with $df (1 - e^{-F(x)})/(1 - e^{-1})$ for any F.

Algorithm TB.

1. If $Y_j < Y_{j-1}$, then $Z_j = 1$.
2. If $Y_j > Y_{j-1}$, then $Z_j = 0$.
3. If $Y_j = Y_{j-1} = \cdots = Y_{j-k+1}$, then with probability $1/k$: $Z_j = 1$
   with probability $(k-1)/k$: $Z_j = 0$.

If even n are accepted (step 4 of Algorithm V) instead of odd, a deviate with $df (F + h(-F))/(1 + h(-1))$ is obtained and $p_A = (h(-1) + 1)$ and $EN_F = (1 + h(1))/(h(-1) + 1)$. If a decreasing sequence of Y_j's is used (change $\leqslant$ to $>$ in step 3), then the output has df $1 - h(F(x) - 1)/h(-1)$, with p_A and EN_F unchanged.

To illustrate the method, let the desired df be $F_X(x) = 1 - \cos(\pi x/2)$ for $x \in (0, 1)$ and let

$$h(w) = w + \frac{\pi^2}{48} w^2 + \frac{\pi^4}{5760} w^3 + \cdots + \frac{\pi^{2i-2}}{2^{2i-3}(2i)!} w^i + \cdots.$$

Note $1 = a_1 \geqslant a_2 \geqslant \cdots$ and let $F(u) = u^2$ which is easily generated as the maximum

of two independent uniform deviates. Simple calculations yield $p_A = 8\pi^{-2}$ and $EN_F \approx 2.74$.

3. Generalizations of Forsythe's Method. Forsythe's [5] method can also be generalized for functions with certain power series expansions. With Y_j's and Z_j's as before, consider the algorithm

Algorithm F.

1. Generate X with df P, and let $Y_0 = G(X)$.
2. Generate $Y_1, Y_2, \ldots, Z_1, Z_2, \ldots$ as long as $\max(Y_1, \ldots, Y_n) \leqslant Y_0$ and $Z_1 = \cdots = Z_n = 1$.
3. If $Y_{n+1} > Y_1$ or $Z_{n+1} = 0$, stop.
4. If n is even, deliver X, else go to 1.

Note that p_1 need not be 1, and the sole condition on G is that it maps the support of P onto the support of F. Let E_n be as before and note the following calculations:

$$Pr(X \leqslant x,\ Y_1 \leqslant G(x), E_n) = \int_{-\infty}^{x} Pr(Y_1 \leqslant G(y), E_n) dP(y),$$

$$\begin{aligned} &Pr(X \leqslant x,\ Y_1 \leqslant G(X), E_n, E_{n+1}^c) \\ &\quad = \int_{-\infty}^{x} [Pr(Y_1 \leqslant G(y), E_n) - Pr(Y_1 \leqslant G(y), E_{n+1})]\ dP(y) \\ &\quad = \int_{-\infty}^{x} [a_n F(G(y))^n - a_{n+1} F(G(y))^{n+1}]\ dP(y). \end{aligned}$$

Again let $h(w) = \Sigma_{n=0}^{\infty} a_n w^n$ but with $1 = a_0 \geqslant a_1 \geqslant \cdots \geqslant 0$. Convergence is implied for $w \in [-1, 1)$ by $a_n \longrightarrow 0$. Summing the last displayed expression over even n (beginning with 0), X has the df

$$\int_{-\infty}^{x} h(-F(G(y)))\, dP(y) \Big/ \int_{-\infty}^{+\infty} h(-F(G(y)))\, dP(y).$$

The acceptance probability is the denominator of the df above, and the expected number of Y_j's is

$$EN_F = \frac{\int_{-\infty}^{+\infty} h(F(G(y)))\, dP(y)}{\int_{-\infty}^{+\infty} h(-F(G(y)))\, dP(y)}.$$

The expected number of samples from the distribution P, EN_P is the reciprocal of the acceptance probability. These results correspond precisely to those of Ahrens and Dieter [4, Lemma 2] for $h(w) = e^w$.

4. Examples. To illustrate the generalization of Forsythe's method, consider the following algorithm for the Cauchy distribution.

$$h(w) = (1 - w/2)^{-2}$$
$$= 1 + w + 3w^2/4 + \cdots + (n+1)w^n/2^n + \cdots,$$
$$F(u) = u, \quad u \in (0, 1),$$
$$P(x) = (1 + x)/2,$$
$$G(x) = 2[(1 + x^2)^{1/2} - 1].$$

Algorithm C.

0. (Executed only on first entry) Generate u.
1. $X \leftarrow 2u - 1$; $n \leftarrow 0$; $y_0 \leftarrow G(X)$.
2. Generate u_1, $y_{n+1} \leftarrow u_1$.
3. If $y_{n+1} > p_{n+1} y_0$, go to 5.
4. $n \leftarrow n + 1$; go to 2.
5. $u \leftarrow (y_{n+1} - p_{n+1} y_0)/(1 - p_{n+1} y_0)$.
6. If n is odd, go to 1.
7. $u \leftarrow 2u$; if $u < 1$, deliver X,
8. otherwise, $u \leftarrow u - 1$ and deliver $X \leftarrow 1/X$.

Here the words "Generate u (or u_1)" means to draw an independent uniform (0, 1) deviate. Note that the dual check of step 3 of Algorithm F can be done in a single step, #3 of Algorithm C. If $y_{n+1} > y_0$, then the sampling of y's ceases or if $z_{n+1} = 0$. Conditional on $y_{n+1} < y_0$ then $z_{n+1} = 1$ is equivalent to $y_{n+1} < p_{n+1} y_0$; hence, the algorithm continues only when $y_{n+1} < p_{n+1} y_0$. Also resulting from F being uniform is the "saving" or "reusing" in step 5 due to Brent [3] in his Forsythe-type algorithm for Gaussian deviates. Note also that the condition on G is satisfied

$$\underset{\text{support of } P}{G\colon [-1, +1]} \longrightarrow [0, 2(\sqrt{2} - 1)] \subseteq \underset{\text{support of } F.}{[0, 1]}$$

Some calculations yield $p_A = \pi/4$ and $EN_F \approx 1.88$; the "reusing" eliminates EN_p. Because of the square root computation in G, Algorithm C will be slower in most implementations than the synthetic tangent algorithm [6]. But it requires 1/4 fewer uniform deviates than the synthetic tangent's $8/\pi \approx 2.56$.

To illustrate the flexibility of G, consider the following algorithm for the geometric distribution, i.e., $F_X(x) = 1 - (1 - p)^{x+1}$, $x = 0, 1, 2, \ldots$. There, P is the *discrete* uniform distribution on the integers 0 through $n(p)$ and F is the *continuous* uniform distribution, $F(x) = x$, $x \in [0, 1]$, and $G(x) = -x \ln(1 - p)$. Let $b = -\ln(1 - p)$, then $n(p)$ must satisfy $0 < bn(p) \leqslant 1$ in order that G satisfy the requirement that the support of P, $\{0, 1, 2, \ldots, n(p)\}$, be mapped into the support of F, $[0, 1]$. Here $h(w) = e^w$ (Forsythe).

Algorithm G.

0. $X = 0$; $a = (1 - p)^{n(p)+1}$; generate u.
1. If $u > a$, go to 3.
2. $X \leftarrow X + n(p) + 1$, $u \leftarrow u/a$, go to 1.
3. Generate I with *df* P.

4. Generate $u_1, u_2, \ldots$ from F until $u_1 > Ib$ or $u_{n+1} > u_n$.
5. If n is odd, go to 3.
6. Deliver $X \leftarrow X + I$.

Another algorithm can be written using Algorithm V with

$$h(w) = (e^{bn(p)w} - 1)/(bn(p))$$

and $F(i) = i/n(p)$, which has nearly the same characteristics as Algorithm G. If $pn(p) \longrightarrow \gamma$ as $p \longrightarrow 0$ (the interesting case), EN_F decreases as γ decreases; but the number of checks in steps 1 and 2 above increases, hence a machine dependent tradeoff.

These elegant algorithms are denigrated by the simple method of obtaining a geometric deviate by taking the greatest integer in Z/b where Z is an exponential deviate. The new algorithms were competitive with $[Z/b]$ when implemented on a CDC 6600 in Fortran using the von Neumann exponential algorithm. But the simplicity of $[Z/b]$ should be preferred to the slight increase in speed.

5. Acknowledgement. An anonymous referee's strong criticism of an earlier version of this paper is greatly appreciated.

Department of Statistics
North Carolina State University at Raleigh
Raleigh, North Carolina 27650

1. J. H. AHRENS & U. DIETER, "Extensions of Forsythe's method for random sampling from the normal distribution," *Math. Comp.*, v. 27, 1973, pp. 927–937.

2. A. C. ATKINSON & M. C. PIERCE, "The computer generation of beta, gamma and normal random variables," *J. Roy. Statist. Soc. Ser. A*, v. 139, 1976, pp. 431–448.

3. R. P. BRENT, "Algorithm 488: A Gaussian pseudo-random number generator," *Comm. ACM*, v. 17, 1974, pp. 704–706.

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. A. J. KINDERMAN, J. F. MONAHAN & J. G. RAMAGE, "Computer methods for sampling from Student's t distribution," *Math. Comp.*, v. 31, 1977, pp. 1009–1018.

7. 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.