Comp. & Maths. with Appls. Vol. 6, pp. 305–315

0097-4943/80/0901-0305/$02.00/0

GENERATING THE MAXIMUM OF INDEPENDENT IDENTICALLY DISTRIBUTED RANDOM VARIABLES

LUC DEVROYE

School of Computer Science, McGill University, 805 Sherbrooke Street West, Montreal, Canada H3A 2K6

Communicated by Donald E. Knuth

(*Received November* 1978; *and in revised form July* 1979)

Abstract—Frequently the need arises for the computer generation of variates that are *exactly* distributed as $Z = \max(X_1, \ldots, X_n)$ where $X_1, \ldots, X_n$ form a sequence of independent identically distributed random variables. For large n the individual generation of the X_i's is unfeasible, and the inversion-of-a-beta-variate is potentially inaccurate.

In this paper, we discuss and compare the corrected inversion method, the $\log(n)/n$-tail method and the record time method. The latter two methods have an average complexity $0(\log(n))$, are very accurate and do not require the inversion of a distribution function.

The normal, exponential and gamma densities are treated in detail. The existence of fast and accurate inversion methods for the error function makes the corrected inversion method faster than the other ones for n large enough when the X_i's are normal random variables.

1. INTRODUCTION

We consider random number generators for $Z_n = \max(X_1, \ldots, X_n)$ where $X_1, \ldots, X_n$ are independent identically distributed random variables with a common density f (and corresponding distribution function F). Such generators are important in various Monte Carlo simulations and experiments involving extrema. In this paper we study the speed and the accuracy of several techniques such as

(i) the inversion method,
(ii) the "brute force" method,
(iii) the log n/n-tail method, and
(iv) the record time method.

The choice of a particular method depends upon n, F, the desired speed, the desired accuracy, the particular computer in use, and the nature of the application. We will briefly discuss

(v) approximate methods

and their accuracy for large n, but for obvious reasons, these methods should not be compared with the exact ones presented below.

2. INVERSION

It is easy to see that $F^{-1}(U^{1/n})$ is distributed as Z_n whenever U is a uniform (0,1) variate (or, in short: U is uniform). If one is going to follow this course, three situations can be distinguished:

(1) F is easy to invert (e.g. $F(x) = 1 - e^{-x}$, $x > 0$, or $F(x) = 1 - x^{-a}$, $x > 1$, $a > 0$).

(2) F^{-1} can be arbitrarily closely approximated through a series expansion or a class of functions.

(3) None of the above is possible: one has to solve the equation $U^{1/n} = F(x)$ for x.

In all three situations, there is an accuracy problem for large n because $U^{1/n}$ is likely to be close to 1. The problem remains even if we consider instead of $F^{-1}(U^{1/n})$,

$$G^{-1}(1 - U^{1/n}) \text{ or } G^{-1}(1 - (1 - U)^{1/n}) \text{ where } G(x) = 1 - F(x).$$

One possibility is to replace $1 - (1 - U)^{1/n}$ by its Taylor series expansion about 0,

$$n^{-1}(U + (1 - 1/n)U^2/2! + (1 - 1/n)(2 - 1/n)U^3/3! + \ldots),$$

but this series converges too slowly to be practical. We could however exploit the fact that e^{-E} is uniform whenever E is an exponential variate. Thus,

$$1 - U^{1/n} = 1 - e^{-E/n} = (E/n) - (E/n)^2/2! + (E/n)^3/3! - \ldots, \tag{1}$$

which converges quickly for large n. If we break off the series after r terms, then the error will not exceed $(E/n)^{r+1}/(r+1)!$. When E/n is small, the overall answer is close to E/n, which makes the relative error smaller than $(E/n)^r/(r+1)!$. Thus, rather than to generate $1 - U^{1/n}$, we propose the following:

(1) Generate an exponential variate E.

(2) Compute the series (1), and stop after the rth term if $(E/n)^r < (r+1)!d$ where d is the maximum relative error allowed (e.g. $d = 10^{-7}$ in single precision arithmetic on an IBM 360 computer), or if $(E/n)^{r+1} < (r+1)!d^*$ where d^* is the smallest number that can be stored.

The number of terms from (1) that are included is a random variable N. We have

$$P(N > k) = \exp(-n((k+1)!d)^{1/k}).$$

For $d = 10^{-7}$ $p(N > k)$ is smaller than $e^{-20} (< 10^{-7})$ for $n > 10^8$ $(k = 1)$, $n \geq 25820$ $(k = 2)$, $n \geq 1494$ $(k = 3)$, $n \geq 340$ $(k = 4)$, $n \geq 135$ $(k = 5)$, $n \geq 18$ $(k = 10)$, $n \geq 5$ $(k = 20)$. Since N decreases as n increases, the average running time of the inversion method does not increase with n when the average time needed for F^{-1} or G^{-1} is uniformly bounded from above on $(0, 1)$.

Since uniform spacings are distributed as independent exponential variates proportional to their sum (see, e.g. Pyke[1]), we can avoid the series computation in (1) altogether by replacing $1 - U^{1/n}$ by $E/(E + E_n)$ where E is an exponential variate and E_n is an independent gamma variate with parameter n. Fast methods for obtaining E_n in average time $0(1)$ as $n \to \infty$ are described by Marsaglia[2], Ahrens and Dieter[3], Vaduva[4], Tadikamalla[5], Cheng[6]. For the exponential variate E, see Ahrens and Dieter[7], Marsaglia[8] or Sibuya[9]. For a more detailed bibliography, we refer to Sowey[10].

Example. (The normal density.) Among the favorite methods for the inversion of the normal density figures Hasting's approximation[11] used by Cunningham[12] and Milton and Hotchkiss[13], and refined by Odeh and Evans[14]. Algorithm AS70 of Odeh and Evans is accurate to seven decimal places when its argument lies between 10^{-20} and $1 - 10^{-20}$. For higher accuracy, one can use one of many iterative algorithms that are based upon consecutive evaluations of the normal integral.

3. BRUTE FORCE METHOD

The algorithm that first produces all the X_i's and then takes the maximum has several drawbacks in the present context. First, its average complexity increases as kn where the constant k depends upon how hard it is to obtain one variate X_i. For n greater than a small threshold n_0, it will require more time on the average than with any of the other methods listed in the introduction. There is also a potential accuracy problem because standard random number generators are often not designed to deliver variates with great precision from the tails of the distribution.

4. LOG N/N-TAIL METHOD

We propose a new method here that is based on the assumption that a simple family of generators for the tail densities of f exists. It uses the principle of rejection, and does not require a computation of F or F^{-1} for each variate.

We define the tail density f_a by

$$f_a(x) = \begin{cases} f(x)/p & ,x > a, \\ 0 & ,x \leq a, \end{cases}$$

where $p = 1 - F(a)$. The threshold a and the corresponding tail probability p are carefully

picked *before* application of the algorithm. The number of X_i's that exceed a is binomially distributed with parameters n and p. Rather than to generate nX_i's, one could just as well generate a binomial (n,p) variate B, obtain B variates from the tail density f_a and find the maximum.

The algorithm

(1) Generate a binomial (n,p) variate B.

(2) If $B=0$, go to 4.

(3) Generate B variates $Y_1,\ldots,Y_B$ from f_a, and exit with $Z_n \leftarrow \max(Y_1,\ldots,Y_B)$.

(4) Generate variates $X_1,X_2,\ldots$ from f until exactly n of them satisfy $X_i < a$. Exit with $Z_n \leftarrow \max\{X_i : X_i < a\}$.

In order to analyze the average complexity of this algorithm we assume the following; using the symbols C_i for complexities that are random variables, c_i for constants, and E for expected value:

—The average complexity $E(C_1)$ to get one variate from f_a satisfies $\limsup_{a\to\infty} E(C_1) = c_1' < \infty$, and thus $\sup_{a>a_0} E(C_1) \le c_1 < \infty$ for some a_0.

—The average complexity $E(C_2)$ to obtain B is bounded from above by $c_2np + c_2'$.

—The average complexity $E(C_3)$ to obtain one variate from f is $c_3 < \infty$.

If c_4 and c_5 are positive constants, then the overall average complexity $E(C)$ satisfies

$$\begin{aligned} E(C) &\le E(B)E(C_1) + E(C_2) + P(B=0)c_3n/(1-p) + c_4 \\ &\le (c_1+c_2)np + c_5 + c_3n(1-p)^{n-1} \\ &\le (c_1+c_2)np + c_5 + c_3n\,\mathrm{e}^{-(n-1)p}. \end{aligned} \tag{2}$$

Here we used the fact that the average number of X_i's generated in Step 4 is $n/(1-p)$, a property of the negative binomial distribution. Minimizing (2) with respect to p gives a solution

$$p_0 = \log(c_3(n-1)/(c_1+c_2))/(n-1) \sim \log(n)/n \text{ as } n\to\infty.$$

For large n, the optimal choice of p is independent of f and the relative costs c_1,c_2,c_3. After resubstitution of either $p=p_0$ or $p=\log(n)/n$ into (2) we have

$$E(C) = 0(\log(n)). \tag{3}$$

Also, $P(B=0)n/(1-p) = 0(1)$, that is, the average complexity due to Step 4 is uniformly bounded for all n. This shows that to reduce $E(C)$ one has to carefully pick the methods to generate binomial variates and variates from f_a.

Note: Step 4 may be replaced by Step 4′ where f_a^* is the central portion of f (thus, $f = pf_a + (1-p)f_a^*$):

(4′) Generate n variates $X_1,X_2,\ldots,X_n$ from f_a^* and exit with $Z_n \leftarrow \max(X_1,\ldots,X_n)$.

The analysis of $E(C)$ given above remains valid here if we replace the term $P(B=0)c_3n/(1-p)$ by one of the form $P(B=0)c_6n$. Again, the best p satisfies $p_0 \sim \log(n)/n$.

Generation of B. Ahrens and Dieter[3] survey binomial random number generators such as

(i) the count-ones method (cost is proportional to n);

(ii) linear tree search (Knuth[15], the cost is proportional to $c+np$ but for large n the computation of factorials becomes cumbersone);

(iii) the beta method (requires $0(\log(n))$ computation on average for fixed p).

Here we propose a simple generator with average complexity $0(1+np)$ that uses the fact that if B is the first nonnegative integer for which

$$\sum_{i=1}^{B+1} V_i > n,$$

where $V_1, V_2, \ldots$ are independent geometrical (p) variates (that is, $P\{V_1 = n\} = p(1-p)^{n-1}$, all $n \geq 1$), then B is binomial (n,p). Clearly, the expected number of geometrical variates needed is

$$E(B) + 1 = np + 1.$$

Thus, if $p \sim \log(n)/n$, then this number is $0(\log(n))$.

To generate the V_i's we put

$$V_i \leftarrow \overline{\lfloor -E_i/\log(1-p) \rfloor}$$

where $E_1, E_2, \ldots$ are independent exponential variates. For the proof of this, see Knuth[15]. Exponential generators are discussed by Marsaglia[8], Sibuya[9] and Ahrens and Dieter[7]. We are using the program REXP developed by Marsaglia at McGill University as part of the "super-duper" random number generator package.

To obtain an accurate result for $-\log(1-p)$ where p is small, one can follow two courses:

(1) Compute $p + p^2/2 + p^3/3 + \ldots$ and stop after k terms where $p^{k+1}/(k+1) < pd$ where d is the achievable accuracy.

(2) Set $r = 1 - 2/p$ and compute $-(2/r)(1 + 1/(3r^2) + 1/(5r^4) + \ldots)$.

Thus, we summarize the procedure as follows:

(1) Compute $v = -\log(1-p)$; $S \leftarrow 0$; $B \leftarrow 0$.

(2) Generate an exponential variate E; $S \leftarrow S + \overline{\lfloor vE \rfloor}$

(3) If $S > n$, exit with B.

(4) $B \leftarrow B + 1$; return to 2.

Tail of the normal density. The principle of rejection (von Neumann[16]) can be used to obtain variates from the tail of f. When f is normal, rejection from the tail of the Rayleigh distribution is simple (Marsaglia[17]). When f is gamma, a shifted exponential seems well suited for the purpose, and when f has a polynomial tail (such as the Cauchy density), rejection from Paretovariates is indicated. In all cases, it is wise to choose a density whose tail decreases at the same rate as that of f. We will use the symbols g_a for the auxiliary density, U for a uniform variate, V for a variate from g_a, X for a variate from f_a (thus, X is the answer), and ρ for the efficiency of the rejection technique, that is, the probability of accepting the first couple (U,V) that is generated. The average number of such couples needed is $1/\rho$.

If $p = 1 - F(a)$, then

$$f_a(x) = (1/p\sqrt{(2\pi)})\, e^{-x^2/2}, x > a,$$

and

$$g_a(x) = x\, e^{(a^2-x^2)/2}, x > a,$$

in the normal case. Since g_a has distribution function $1 - e^{(a^2-x^2)/2}$, $x > a$, the random variable $V = \sqrt{(a^2 - 2\log(U_1))}$ has density g_a whenever U_1 is uniform (Marsaglia[17]). Equivalently, $V = \sqrt{(a^2 + 2E)}$ has density g_a whenever E is exponential. The multiplication constant in the rejection step

$$\rho f_a(V) < U g_a(V)$$

is

$$\rho = \inf_{x>a} g_a(x)/f_a(x) = pa/f(a).$$

From Gordon's inequality[18, 19],

$$\frac{a}{a^2+1} f(a) \leq p \leq \frac{1}{a} f(a)$$

we deduce that $\rho \geq a^2/(1 + a^2)$ which quickly tends to 1 as $a \to \infty$. Thus, the average complexity of this procedure tends to a constant as $a \to \infty$. We summarize:

(1) Generate a uniform variate U and an exponential variate E.

(2) Set $V \leftarrow \sqrt{(a^2+2E)}$.
(3) If $UV > a$, go back to 1.
(4) Exit with $X \leftarrow V$.

In the case $B > 0$, the square root in Step 2 can be avoided most of the time. Replace Steps 1–4 by

(1') Generate a uniform variate U and an independent exponential variate E.
(2') $V \leftarrow a^2 + 2E$.
(3') *If* $U^2 V > a^2$, return to 1'.
(4') Exit with $X \leftarrow V$.

The variates $Y_1, \ldots, Y_B$ so obtained give us Z_n through the formula $Z_n = (\max(Y_1, \ldots, Y_B))^{1/2}$. In other words, instead of $\log(n)$ square root operations per variate, we need only one square root operation.

Tail of the gamma density. We recall that the gamma density is defined by

$$f(x) = x^{\gamma-1} e^{-x}/\Gamma(\gamma), x > 0, \gamma > 0.$$

We will discuss rejection from the density

$$g_a(x) = b\, e^{b(a-x)}, \; x > a,$$

(see also Ahrens and Dieter[3] and Vaduva[4]) as opposed to rejection from densities with a polynomial tail (Cheng[6]). The parameter b is a function of γ and a. It is easy to see that $a + E/b$ has density g_a where E is an exponential variate. We will now consider the cases $\gamma < 1$ and $\gamma > 1$ separately.

If $p = 1 - F(a)$ and γ is defined as in the previous section, then for the *case* $\gamma < 1$ we have, with $b = 1$,

$$\begin{aligned}
\rho &= \inf_{x>a} g_a(x)/f_a(x) = p\Gamma(\gamma)/a^{\gamma-1} e^{-a} = p/f(a) \\
&= \int_a^\infty (x/a)^{\gamma-1} e^{a-x}\, dx = \int_0^\infty (1 + x/a)^{\gamma-1} e^{-x}\, dx \\
&\geq \int_0^\infty e^{x((\gamma-1)/a-1)}\, dx \\
&= (1 - (\gamma-1)/a)^{-1} \\
&\to 1 \text{ as } a \to \infty.
\end{aligned}$$

Once again, the asymptotic efficiency is established. Incorporating the value of ρ in the rejection step $\rho f_a(V) < U g_a(V)$ gives for all $a > 0$:

(1) Generate a uniform variate U and an exponential variate E.
(2) $V \leftarrow a + E$.
(3) If $V/a \geq 1/U^{1/(1-\gamma)}$ go back to 1.
(4) Exit with $X \leftarrow V$.

The squeeze principle can be used to avoid logarithmic/exponential computations in Step 3 most of the time (see for instance Marsaglia[2] and Cheng[6]) for discussions and examples).

In the *case* $\gamma > 1$, the inequality

$$(x/a)^{\gamma-1} \leq \exp((\gamma-1)(x/a - 1))$$

suggests the choice of $b = 1 - (\gamma-1)/a$. With this choice we have

$$\rho = \inf_{x>a} g_a(x)/f_a(x) = b \int_a^\infty u^{\gamma-1} e^{-u}\, du/a^{\gamma-1} e^{-a}$$

where we used the fact that for $x \geq a$, the function $x\,e^{-x/a}$ is maximal at $x = a$. Thus,

$$\rho = b\int_0^\infty (1 + x/a)^{\gamma-1} e^{-x}\,dx \to 1 \text{ as } a \to \infty.$$

Having established the uniform efficiency for all a large enough, we can summarize the algorithm as follows:

(1) Generate two exponential variates, E and E_1.
(2) Compute $V \leftarrow a + E/b$ (where $b = 1 - (\gamma - 1)/a$).
(3) If $V/a - 1 + \log(a/V) \geq E_1/(\gamma - 1)$, go back to 1.
(4) Exit with $X \leftarrow V$.

A squeeze-type of improvement is possible if we notice that $\log(V/a) = \log(1 + (V - a)/a) \geq 2(V - a)/(V + a) = 2E/(b(V + a))$. Here we used the inequality $\log(1 + y) \geq 2y/(y + 2)$ [19]. Thus, between Step 2 and Step 3, we may put,

(2') If $E^2/(b^2 a(V + a)) \leq E_1/(\gamma - 1)$, exit with $X \leftarrow V$.

All that is said here is only true when $a > \gamma - 1$.

Computation of a and p. To compute a, one can rely on $a = G^{-1}(p_0)$ after estimation of $c_3/(c_1 + c_2)$, or one can use $a = G^{-1}(\log(n)/n)$. Notice that for fixed n, the time spent on the inversion is well spent, because once a is found, the same a can be used to generate a sequence of Z_n's. Clearly, if only one or two variates Z_n are needed, one could just as well employ straightforward inversion.

Another possibility is to choose a good estimate a to the solution of $G(a) = p_0$ from theoretical considerations, and then compute $p = G(a)$ using a numerical integration subprogram. For example, when F is normal, then $G(x) \sim f(x)/x$ as $x \to \infty$ (Feller [20]). If $q = 1/p_0$, a first approximate solution of $f(x)/x = 1/q$ is $a = (2\log(q))^{1/2}$. This is unsatisfactory because $E(C) = 0(n)$ in that case. A second approximation is

$$a = (2\log(q))^{1/2} - \frac{\log(4\pi) + \log(\log(q))}{2(2\log(q))^{1/2}}$$

(Cramer [47]), and in that case $E(C) = 0(\log(n))$ as claimed. For the gamma density one can use the fact that $G(x) \sim f(x)$ as $x \to \infty$ because of the inequalities

$$f(x) \leq G(x) \leq f(x)/(1 - (\gamma - 1)/x)$$

when $\gamma > 1$ and $x > \gamma - 1$.

The computation of $G(a)$ is relatively easy for most distribution functions. For example, for the normal density, we refer to a sequence of papers that start with a modified version of a method described in Kendall and Stuart [21] (see Cooper [22], Hill [23], Hitchin [24]), the paper by Adams [25] and an improved version of it (Hill [26]). The latter method, algorithm AS66, is highly recommended because of its speed and accuracy. For the gamma density, a continued fraction expansion method (see Abramowitz and Stegun [27]) is efficiently programmed by Bhattacharjee [28] under the name: algorithm AS32.

5. THE RECORD TIME ALGORITHM

In some process simulations one needs a sequence $(Z_{n_1}, Z_{n_2}, \ldots, Z_{n_k})$ of maxima that correspond to *one* realization of the experiment, where $n_1 < n_2 < \ldots < n_k$. The inversion method will require k inversions. The method given below will, on average, require the generation of $\log(n_k)$ exponential variates and the computation of $\log(n_k)$ values of the distribution function. Clearly, for $\log(n_k) < ck$ (some constant c), its average complexity will be smaller.

The *record times* $L_1, L_2, \ldots$ and the *record values* $Y_1, Y_2, \ldots$ corresponding to a sequence $X_1, X_2, \ldots$ are defined by:

$$L_1 = 1,$$
$$L_{n+1} = \inf(j : X_j > X_{L_n},\ j > L_n),\ n > 0,$$
$$Y_n = X_{L_n},\ n \geq 1.$$

In the record time algorithm one essentially replaces the problem of the production of the X_i's by that of the generation of (L_n, Y_n), $n = 1,2,\ldots$.

The algorithm

(1) Generate Y from f, set $L \leftarrow 1$ and compute $p \leftarrow G(Y)$.
(2) Compute $v = -1/\log(1-p)$ using a method that is accurate for small p.
(3) Set $L \leftarrow L + \lfloor vE \rfloor$ where E is an exponential variate.
(4) If $L > n$, exit with $Z_n \leftarrow Y$.
(5) Generate a new Y from the tail density f_Y, set $p \leftarrow G(Y)$, go back to 2.

Record times have a few noteworthy properties which explain why the algorithm produces variates with the correct density, and what efficiency we can expect. First of all, given Y_i, the random variables $L_{i+1} - L_i$ and Y_{i+1} are independent with the followiing distributions: $L_{i+1} - L_i$ (the waiting time for the next record) is geometrically distributed with parameter $p = G(Y_i)$:

$$P(L_{i+1} - L_i = k) = p(1-p)^{k-1},\ k \geq 1.$$

Also, Y_{i+1} has the tail density f_{Y_i}. When F is continuous, the joint distribution of $L_1, L_2, \ldots,$ is independent of F. In particular, if N_n is the *number of record times* in the sequence $X_1, X_2, \ldots, X_n$, then as $n \to \infty$,

(i) $N_n \sim \log n$ with probability one (Renyi[29]),
(ii) $(N_n - \log(n))/\sqrt{\log(n)}$ tends in distribution to the normal law (Renyi[29]),

and

(iii) $E(N_n) = 1 + 1/2 + 1/3 + \ldots + 1/n = \log(n) + e^* + 0(1/n)$ where e^* is Euler's constant (0.5772...).

These and other results are surveyed by Resnick[30]. All of them contain information about how long-tailed the distribution of the record times is. It is also known that $\log(L_n) \sim n$ with probability one, and that $E(L_{n+1} - L_n) = \infty$ for all n. The latter somewhat surprising property is very easy to prove.

Step 5 almost solely determines the speed of the procedure. One needs a good but fast method for evaluating the integral G (see Section 4), unless G is easy to invert. In that case, Step 5 can be replaced by

(5') Generate a uniform variate U, set $p \leftarrow Up$, and $Y \leftarrow G^{-1}(p)$. Go back to 2.

One also needs efficient generators for the tail densities of f (that is, methods whose average complexity remains bounded as the tail gets smaller). Some of these are discussed in the previous section for well-known densities. The algorithm given above for the normal tail cannot be used when the cut-off point is near 0. One may then define a mixed strategy. Let c be a real number and define the following method to generate a variate from f_a:

(1) If $a < c$, go to 3.
(2) Generate a variate from f_a using a specific tail method (Section 4) and stop.
(3) Generate variates from f, and stop when one of them exceeds a.

The generation of normal variaties is treated in a series of papers with increasing sophistication[7, 16, 17, 32–38]. See the survey papers by Atkinson and Pearce[39] and Payne[40]. Gamma variates are discussed by Ahrens and Dieter[3], Chen[6] Marsaglia[2], Tadikamalla[5] and Vaduva[4], Dagpunar[41], Atkinson[42], and Fishman[43]. In each case, the point c has to be determined by the user because it depends upon the relative efficiencies of the generators for f and f_a.

We have no control in the record time algorithm over the number of times the loop 5–2–3–4 is executed: it is independent of f. Since all the operations in the different steps can be carried out with constant average complexity for most densities, the overall average complexity of the record time algorithm is $0(\log(n))$. Because of the repeated computation of $p \leftarrow G(Y)$ (Step 5) it is not competitive with the $\log(n)/n$-tail method in most raw simulations. It may have the edge in some continued simulations.

6. APPROXIMATE METHODS

The limiting distribution function of Z_n is often of the form $\exp(-e^{-x})$, in the sense that for large classes of distribution functions F, one can find sequences a_n and b_n such that

$$\lim_{n\to\infty} P(a_n(Z_n - b_n) \le x) = e^{-e^{-x}}. \tag{4}$$

To this class belong all the gamma distributions and normal distributions. For the sake of completeness, we recall von Mises' theorem (von Mises[44]): if $F(x) < 1$ for all x, if F is twice differentiable for all x greater than some x_0, and if

$$\frac{d}{dx}\left(\frac{1-F(x)}{f(x)}\right) \to 0 \text{ as } x \to \infty, \tag{5}$$

then (4) is true with $a_n = nf(b_n)$, and $1 - F(b_n) = 1/n$. (see also David[45]). Condition (5) holds in essence for all densities with an exponential tail. For example, if $1 - F(x) = \exp(-R(x))$, then (5) reduces to

$$R''(x)/(R'(x))^2 \to 0 \text{ as } x \to \infty. \tag{6}$$

(DeHaan[46]). For the exponential density, (4) holds with $a_n = 1$, $b_n = \log(n)$. For the normal density, in view of $G(x)/f(x) \sim 1/x$, (4) holds with $a_n = b_n = (2\log(n))^{1/2}$. A strictly better choice is

$$a_n = b_n = \sqrt{(2\log(n))} - \frac{\log(4\pi) + \log(\log(n))}{2\sqrt{(2\log(n))}}$$

(Cramer[47]). Fisher and Tipett[48], Dronkers[49], Gumbel[50] and Haldame and Jayakar[51] have results on the rather slow rate of convergence in (4) for normal densities. For normal densities, it is known that $Z_n - \sqrt{(2\log n)} \to 0$ almost surely so that, oddly enough, $\sqrt{(2\log n)}$ can be used as a first approximation to Z_n.

Gnedenko[59] has shown that

$$\lim_{n\to\infty} P(a_n(Z_n - b_n) \le x) = e^{-x^{-a}},\ x > 0, \tag{7}$$

for some $a > 0$, and some sequences a_n and b_n if and only if for every $y > 0$,

$$(1 - F(x))/(1 - F(yx)) \to y^a \text{ as } x \to \infty$$

(see also Barndorff-Neilsen[52] and David[45]). This holds in general when f has a tail that decreases at a polynomial rather than exponential rate.

The main use of approximate methods here is the rough analysis of the behavior of Z_n for large n. Indeed,

$$b_n - a_n^{-1}\log(\log(1/U)) \text{ or } b_n - a_n^{-1}\log(E)$$

is approximately distributed as Z_n for large n, when U is uniform and (4) holds.

7. EXPERIMENTAL RESULTS

Throughout the experiments we are using single precision arithmetic on an IBM 370/158 computer. The language is standard FORTRAN, and the random number generators needed are those from McGill University's "Super-Duper" package. To give the reader an idea of the relative speeds with which several functions are performed, we timed some basic functions (in

Table 1.

Function	Time (μs)	Program
Generate uniform variate	26	UNI (Super-Duper Package)
Generate exponential variate	48	REXP (Super-Duper Package)
Generate normal variate	46	RNOR (Super-Duper Package)
Square root	71	SQRT
Logarithm	113	ALOG
Accurate logarithm: $\log(1-x)$ x very small	133	ACLOG (available from author)
Integrate tail of of normal density	163	ALNORM (see [51])
Inverse of the normal integral	330	GAUINV (see [8])

microseconds) over a broad range of values for their arguments. The figures in Table 1 are averages over 10,000 runs.

Goodness-of-fit-test. We consider the Kolmogorov–Smirnov test with 1000 variates Z_n for each value of n. If K_n is the Kolmogorov–Smirnov statistic, then the number

$$J_n = \sum_{-\infty}^{+\infty} \exp(-2j^2K_n^2)$$

is computed. For each n the whole procedure is repeated three times. As an example, we picked for f the normal density. The algorithms considered here are the inversion method, the $\log(n)/n$-tail method and the brute force method. For practical reasons, the latter could only be tested for small n. The inversion method gives inaccurate results for n greater than 10^5 unless $1-U^{1/n}$ is replaced by the exponential series given in (1). The $\log(n)/n$-tail method gave satisfactory results over the whole range of values of n. Typical values for J_n are given for these algorithms when n varies from 10^2 to 10^8 (see Table 2). The inversion method without the series (1) is eliminated from consideration from now on in the experiments that follow.

A technical problem develops when one applies the Kolmogorov–Smirnov test: the computation of $F^n(Z_n)$ is not very accurate itself when carried out as such. Rather, we used the numbers $1-\exp(n\log(1-G(Z_n)))$ by first computing $\log(1-G(Z_n))$ accurately through a special program.

Timing. Again restricting ourselves to the normal case, most algorithms discussed above were timed using 1000 variates for each figure in Table 3.

Table 2.

Method	n =	10^2	10^3	10^4	10^5	10^6	10^7	10^8
Inversion (without series (1))		0.05	0.49	0.97	0.65	1.00	1.00	1.00
Inversion with series (1)		0.49	0.77	0.07	0.43	0.96	0.46	0.08
Log(n)/n-tail method		0.95	0.21	0.28	0.75	0.61	0.18	0.55

Table 3. All times milliseconds/sample

Method	$n = 10$	10^2	10^3	10^4	10^5	10^6	10^7	10^8	10^9
Log(n)/n-tail method	0.45	0.85	1.25	1.32	1.70	2.03	2.38	2.72	3.00
Inversion	0.73	0.63	0.57	0.60	0.43	0.43	0.43	0.43	0.43
Brute force	≈ 0.06025 n ms.								

The brute force method with its linear increase in complexity is only competitive for n smaller than 10. The inversion method has a constant complexity in the range shown in the table. The slight decrease in the timings for larger n is due to the faster convergence of the series (1). Omission of (1) for small n will not affect the accuracy and might be considered when speed is important. The $\log(n)/n$-tail method clearly shows a logarithmically increasing complexity as n increases. It outperformed the brute force and the inversion methods in the range $8 < n < 60$.

The inversion was done by the algorithm AS70 of Odeh and Evans[14]. Without a fast inversion subprogram, the $\log(n)/n$-tail method will be superior over a broader range of values for n when, as is often the case, good generators are available for the tail densities.

The $\log(n)/n$-tail method had $p = \log(0.22n)/n$ (the constant 0.22 was arrived at purely by guessing after consulting Table 1. No attempt was made to optimize this value). The constant $a = G^{-1}(p)$ was computed through algorithm AS70.

8. CONCLUDING REMARKS

We have in more or less detail treated the normal case. The exponential density is easy to invert and poses no problem either. From these cases, we can generalize as follows: if X_1 is distributed as $h(Y)$ where Y has a simple maxima-generation method associated with it, and where h is a strictly increasing and continuous function, then

$$\max(X_1, \ldots, X_n) = h(\max(Y_1, \ldots, Y_n))$$

where the Y_i's are distributed as Y. Thus, without any effort we have obtained exact methods for random variables that are monotone functions of normal or exponential random variables. The maximum of chi-square variates can be obtained with some care from the $\log(n)/n$-tail method for normal variates.

This study was motivated by the need for obtaining exact methods for generating extrema when n is large. For one thing, the variates Z might be used later on to study how quickly the convergence to the asymptotic distribution function $\exp(-\exp(-x))$ takes place. For the same reason, we also need exact methods for generating sample quantiles when n is large. For quantiles near the lower or upper end of the scale, the methods given in this paper can be extended without great difficulty. For the median and other mid-range quantiles, one first generates a beta variate and then applies an inversion method. The generation of an entire subset of the order statistics leads to yet different problems. See for instance[53], [54], [55], [56], [57], [58].

REFERENCES

1. R. Pyke, Spacings. *J. Roy. Stat. Soc. B* **27**, 395–436 (1965).
2. G. Marsaglia, The squeeze method for generating gamma variates. *Comput. Math. Applics* **3**, 321–325 (1977).
3. J. H. Ahrens and U. Dieter, Computer methods for sampling from gamma, beta, Poisson and binomial distributions. *Computing* **12**, 223–246 (1974).
4. I. Vaduva, On computer generation of gamma random variables. *Mathematische Operationsforschung und Statistik, Series Statistics* **8**, 545–576 (1977).
5. P. R. Tadikamalla, Computer generation of gamma random variables, II. *Comm. ACM* **21**, 925–928 (1978).
6. R. C. H. Cheng, The generation of gamma variables with non-integral shape parameter. *Appl. Stat.* **26**, 71–75 (1977).
7. J. H. Ahrens and U. Dieter, Computer methods for sampling from the exponential and normal distributions. *Commun. ACM* **15**, 873–882 (1972).
8. G. Marsaglia, Generating exponential random variables. *Ann. Math. Stat.* **32**, 899–902 (1961).
9. M. Sibuya, Exponential and other random variable generators. *Ann. Inst. Stat. Math.* **13**, 231–237 (1961).
10. E. R. Sowey, A second classified bibliography on random number generation and testing. *Int. Stat. Rev.* **46**, 89–102 (1978).
11. C. Hastings, *Approximations for Digital Computers*. Princeton University Press, Princeton, New Jersey (1955).
12. S. W. Cunningham, Algorithm AS24. Form normal integral to deviate. *Appl. Stat.* **18**, 290–293 (1969).
13. R. C. Milton and R. Hotchkiss, Computer evaluation of the normal and inverse normal distribution functions. *Technometrics* **11**, 817–822 (1969).
14. R. E. Odeh and J. O. Evans, Algorithm AS70. The percentage points of the normal distribution. *Appl. Stat.* **23**, 96–97 (1974).
15. D. Knuth, *The Art of Computer Programming*, Vol. 2. Addison-Wesley, Reading, Mass. (1969).
16. J. von Neumann, Various techniques in connection with random digits. Monte Carlo methods. *Nat. Bur. Stand. AMS* **12**, 36–38 (1951).
17. G. Marsaglia, Generating a variable from the tail of the normal distribution. *Technometrics* **6**, 101–102 (1964).

18. R. D. Gordon, Values of Mills' ratio of area to bounding ordinate and of the normal probability integral for large values of the argument. *Ann. Math. Stat.* **12**, 364–366 (1941).
19. D. S. Mitrinovic, *Analytic Inequalities*. Springer-Verlag, Berlin (1970).
20. W. Feller, *An Introduction to Probability Theory and its Applications*, Vol. 1, pp. 193–194. Wiley, New York (1968).
21. M. G. Kendall and A. Stuart, *The Advanced Theory of Statistics*, Vol. 1. Charles Griffin, London (1963).
22. B. E. Cooper, Algorithm AS2. The normal integral. *Appl. Stat.* **17**, 186–187 (1968).
23. I. D. Hill, Remark ASR2. A remark on algorithm AS2. *Appl. Stat.* **18**, 299–300 (1969).
24. D. Hitchin, Remark ASR8. A remark on algorithm AS4 and AS5. *Appl. Stat.* **22**, 428 (1973).
25. A. G. Adams, Algorithm 39. Areas under the normal curve. *Comput. J.* **12**, 197–198 (1969).
26. I. D. Hill, Algorithm AS66. The normal integral. *Appl. Stat.* **22**, 424–427 (1973).
27. M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. Dover, New York (1965).
28. G. P. Bhattacharjee, Algorithm AS32. The incomplete gamma integral. *Appl. Stat.* **19**, 285–287 (1970).
29. A. Renyi, Theorie des elements saillants d'une suite d'observations. *Colloquium on Combinatorial Methods in Probability Theory*, pp. 104–115. Mathematisk Institut, Aarhus University, Denmark (1962).
30. S. I. Resnick, Extremal processes and record value times. *J. Appl. Prob.* **10**, 864–868 (1973).
31. A. J. Kinderman and J. F. Monahan, Computer generation of random variables using the ration of uniform deviates. *ACM Trans. Math. Software* **3**, 257–260 (1977).
32. J. H. Ahrens and U. Dieter, Extensions of Forsythe's method for random sampling from the normal distribution. *Math. Comp.* **27**, 927–937 (1973).
33. G. E. Forsythe, Von Neumann's comparison method for random sampling from the normal and other distributions. *Math. Comp.* **26**, 817–826 (1972).
34. G. Marsaglia and T. A. Bray, A convenient method for generating normal variables, *SIAM Rev.* **6**, 260–264 (1964).
35. G. Marsaglia, M. D. Maclaren and T. A. Bray, A fast procedure for generating normal random variables. *Commun. ACM* **7**, 4–10 (1964).
36. A. J. Kinderman and J. G. Ramage, Computer generation of normal random variables. *J. Am. Stat. Assoc.* **71**, 893–896 (1976).
37. G. Marsaglia, K. Ananthanarayanan and N. J. Paul, Improvements on fast methods for generating normal random variables. *Inform. Proc. Lett.* **5**, 27–30 (1976).
38. H. Sakasegawa, On a generation of normal pseudo-random numbers. *Ann. Inst. Stat. Math.* **30**, 271–279 (1970).
39. A. C. Atkinson and M. C. Pearce, The computer generation of beta, gamma and normal random variables. *J. Roy. Stat. Soc.* **139**, 431–461 (1976).
40. W. H. Payne, Normal random numbers: using machine analysis to choose the best algorithm. *ACM Trans. Math. Software* **3**, 346–358 (1977).
41. J. S. Dagpunar, Sampling of variates from a truncated gamma distribution. *J. Stat. Comput. Sim.* **8**, 59–64 (1978).
42. A. C. Atkinson, An easily programmed algorithm for generating gamma random variables *J. Roy. Stat. Soc. A* **140**, 232–234 (1977).
43. G. S. Fishman, Sampling from the gamma distribution on a computer. *Commun. Assoc. on Comput. Mach.* **19**, 407–409 (1976).
44. R. von Mises, La distribution de la plus grande de n valeurs. *Selected Papers of Richard von Mises*, Vol. 2. pp. 271–294. American Mathematical Society, Providence, New Jersey (1964).
45. H. A. David, *Order Statistics*. Wiley, New York (1969).
46. L. Dehaan, *On Regular Variation and its Application to Weak Convergence of Sample Extremes* (MC Tract 32). Mathematich Centrum, Amsterdam (1970).
47. H. Cramer, *Mathematical Methods of Statistics*. Princeton University Press, Princeton, New Jersey (1964).
48. R. A. Fisher and L. H. C. Tippet, Limiting forms of the frequency distribution of the largest or smallest member of a sample. *Proc. Camb. Phil. Soc.* **24**, 180–190 (1928).
49. J. J. Dronkers, Appropriate formulae for the statistical distributions of extreme values. *Biometrika* **45**, 447–470 (1958).
50. E. J. Gumbel, *Statistics of Extremes*. Columbia University Press, New York (1958).
51. J. B. S. Haldane and S. D. Jayakar, The distribution of extremal and nearly extremal values in samples from a normal distribution. *Biometrika* **50**, 89–94 (1963).
52. O. Barndorff-Neilsen, On the limit behavior of extreme order statistics. *Ann. Math. Stat.* **34**, 992–1002 (1963).
53. D. Lurie and H. O. Hartley, Machine-generation of order statistics for Monte Carlo computations. *Am. Statist.* **26**, 26–27 (1972).
54. D. Lurie and R. L. Mason, Empirical investigation of several techniques for computer generation of order statistics. *Commun. Stat.* **2** 363–371 (1973).
55. M. Rabinowitz and M. L. Berenson, A comparison of various methods of obtaining random order statistics for Monte Carlo Computations. *Am. Statist.* **28**, 27–29 (1974).
56. J. S. Ramberg and P. R. Tadikamalla, On the generation of subsets of order statistics. *J. Stat. Comp. Sim.* **6**, 239–241 (1978).
57. B. Schmeiser, Generation of the maximum (minimum) value in digital computer simulation. *J. Stat. Comp. Sim.* **8**, 103–115 (1978).
58. W. R. Schucany, Order Statistics in simulation. *J. Stat. Comp. Sim.* **1**, 281–286 (1972).
59. B. Gnedenko, Sur la distribution limite du terme maximum d'une serie aleatoire. *Ann. Math.* **44**, 423–453 (1943).
60. A. J. Kinderman, J. F. Monahan and J. G. Ramage, Computer methods for sampling from Student's t-distribution. *Math. Comp.* **31**, 1009–1018 (1977).