ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XVII, 1 (1980)

I. DEÁK (Budapest)

THE ELLIPSOID METHOD FOR GENERATING NORMALLY DISTRIBUTED RANDOM VECTORS

1. Introduction. Many algorithms exist for generating one-dimensional normally distributed random numbers [1], but only two methods are offered for the generation of n-dimensional normally distributed vectors (see [3], [9] and [15]): the matrix diagonalization and the conditional decomposition methods. The first one could slightly be changed using the Crout factorization to decrease the number of multiplications. In these well-known methods at least $n(n+1)/2$ multiplications are to be performed to obtain one n-dimensional normally distributed random vector.

The ellipsoid method presented in this paper is based on the multivariate composition technique; the name indicates that mostly uniformly distributed random vectors are needed in n-dimensional hyperellipsoids. The density function $\varphi(x)$ of the multidimensional normal distribution is given, as the mixture of some density functions $e_i(x)$ $(i = 1, \ldots, k)$ and two correcting densities $r_1(x)$ and $r_2(x)$, by the formula

$$\varphi(x) = p_1 e_1(x) + \ldots + p_k e_k(x) + p_{k+1} r_1(x) + p_{k+2} r_2(x), \tag{1}$$

where $p_i \geqslant 0$, $\sum_{i=1}^{k+2} p_i = 1$, the functions $e_i(x)$ are constant in hyperellipsoids E_i, and

$$\varphi(x) = (2\pi)^{-n/2} |R|^{-1/2} \exp\left\{-\frac{1}{2} x' R^{-1} x\right\}, \tag{2}$$

where R is the correlation matrix. Thus random vectors with density $\varphi(x)$ can be produced by generating uniformly distributed vectors in the hyperellipsoid E_i. The detailed description of decomposition (1) is given in the next sections.

Let the ellipsoid E be given as

$$E = \{x \mid x' R^{-1} x \leqslant c^2\}$$

for some c, and let T be a lower triangular matrix for which $T'T = R$. The random vector $T\xi$ is uniformly distributed in E provided ξ is uniformly distributed in

$$S = \left\{x \middle| \sum_{i=1}^{n} x_i^2 \leqslant c^2\right\}.$$

The fastest method for generating uniformly distributed points in a hypersphere is that of Muller [14] (for a comparison of these methods see [6]) which works in the following way: Generate η_i $(i = 1, \ldots, n)$, i.e., n independent normally distributed random numbers; the vector

$$\left(\eta_1 \Big/ \left(\sum \eta_i^2\right)^{1/2}, \ldots, \eta_n \Big/ \left(\sum \eta_i^2\right)^{1/2}\right)$$

is uniformly distributed on the surface of

$$S_1 = \left\{x \middle| \sum_{i=1}^{n} x_i^2 \leqslant 1\right\}.$$

Multiplying this vector by a random radius length ($r = u^{1/n}$, where u is uniform in $[0, 1)$) we obtain the point uniform in S.

Thus, to achieve a fast sampling procedure for generating normally distributed vectors an approximate algorithm is proposed for generating points uniformly distributed in hyperellipsoids. This so-called chord method is presented in Section 4. Finally, in the last section the results of the computer runs are given together with the comparisons and possible generalizations of the method for other distributions.

The ellipsoid method was successfully used in evaluating multinormal probabilities by Monte Carlo when the probabilities to be computed were high [7].

2. The ellipsoid method for generating normal vectors. The composition technique for the generation of one-dimensional random variables was described first by Butler [5] as follows.

Assume that our task is to generate random numbers with density function $f(x)$ written in the form

$$f(x) = \int_{-\infty}^{+\infty} g_y(x)\,dH(y), \tag{3}$$

where g_y is a family of densities depending on the parameter y. Here y is also a random variable with distribution function H. First a value η is generated for y from the distribution H, and then a value ξ is sampled for x with density $g_\eta(x)$. This technique was used in [4], [12] and [2].

The generalization of equation (3) for the multivariate case is nothing particular and for the discrete distribution H it may be written in form (1), where x is an n-dimensional vector. The real difficulty lies in the good choice of the density functions $e_i(x)$ $(i = 1, \ldots, k)$, $r_1(x)$, $r_2(x)$ so as to achieve fast sampling with density $e_i(x)$ and to determine $p_1 + \ldots + p_k$ near to 1 without requiring many memory locations.

We have chosen the functions e_i $(i = 1, \ldots, k)$ for the ellipsoid method to be constant in hyperellipsoids. To put it in a more precise way, we introduce some notation. Let us split up the interval $[0, \varphi(0)]$ into $k+1$ parts by the numbers m_i $(i = 1, \ldots, k)$:

$$0 = m_0 < m_1 < \ldots < m_k < m_{k+1} = \varphi(0).$$

Consider the varieties E_i in the n-dimensional space determined by $\varphi(x) \geqslant m_i$, i.e.,

$$E_i = \{x \mid x' R^{-1} x \leqslant -2 \ln (m_i (2\pi)^{n/2} |R|^{1/2})\} \quad (i = 1, \ldots, k+1).$$

These varieties are centre-symmetric hyperellipsoids in R^n (E_1 is the greatest one and E_{k+1} is merely a point), since R is a positive definite symmetric matrix being the correlation matrix of the normal distribution. Let us define the multipliers μ_i $(i = 1, \ldots, k)$ by the aid of which we can write

$$E_i = \{x \mid x = \mu_i y, y \in E_1\} \quad (i = 1, \ldots, k).$$

Let us put

$$e_i(x) = \begin{cases} 1/V_i & \text{if } x \in E_i, \\ 0 & \text{if } x \notin E_i, \end{cases}$$

where

$$V_i = \int_{E_i} dx$$

is the volume of the hyperellipsoid E_i. Thus $e_i(x)$ is the density function of the uniform distribution in E_i. The probabilities p_i can be determined by the equations $p_i e_i(x) = m_i - m_{i-1}$ if $x \in E_i$ $(i = 1, \ldots, k)$. Hence and from the condition

$$\sum_{i=1}^{k+2} p_i = 1$$

we obtain

$$(4) \quad p_i = V_i(m_i - m_{i-1}) \ (i = 1, \ldots, k), \quad p_{k+1} + p_{k+2} = 1 - \sum_{i=1}^{k} p_i.$$

Of course, we intend to determine the division $m_1, \ldots, m_k$ in such a way as to make the sum $p_1 + \ldots + p_k$ as near to 1 as possible in order to gain a fast sampling procedure (see the next section).

To make equation (1) valid the functions r_1 and r_2 are defined as follows:

$$p_{k+1} r_1(x) = \begin{cases} \varphi(x) - \sum_{i=1}^{k} p_i e_i(x) & \text{if } x \in E_1, \\ 0 & \text{if } x \notin E_1, \end{cases}$$

$$p_{k+2} r_2(x) = \begin{cases} 0 & \text{if } x \in E_1, \\ \varphi(x) & \text{if } x \notin E_1. \end{cases}$$

Since r_2 has to be a density function, we calculate the constant

$$p_{k+2} = \int_{R^n - E_1} \varphi(x)\,dx = \int_{c_1}^{\infty} k_n(x)\,dx,$$

where $k_n(x)$ is the density function of the χ^2-distribution with n degrees of freedom, and c_1 is the constant of the hyperellipsoid E_1, i.e.,

$$x' R^{-1} x \leqslant c_1^2 \quad \text{if } x \in E_1.$$

The constants c_i $(i = 1, \ldots, k)$ can be computed by the formula

$$c_i = -2 \ln \left(m_i (2\pi)^{n/2} |R|^{1/2} \right).$$

Now the constant p_{k+1} can be evaluated from the second equation of (4). The constants p_{k+1} and p_{k+2} will be called the *wedge* and the *tail probabilities*, respectively, because of the definitions of the functions r_1 and r_2. Observe that

$$\mu_i = c_i / c_1 \quad (i = 1, \ldots, k).$$

The sampling procedure related to decomposition (1) consists of the following three main parts:

1. With probabilities $p_1, \ldots, p_k$ choose one of the multipliers μ_i, then generate a uniform point y in E_1 and deliver $x \leftarrow \mu_i y$ as a final sample.

2. With probability p_{k+1} generate a sample with density $r_1(x)$, e.g., by the rejection technique (sampling from the wedge).

3. With probability p_{k+2} generate a normally distributed random vector outside the hyperellipsoid E_1, e.g., by inverting the χ^2-distribution function.

For $p_1 + \ldots + p_k$ near to 1 almost all the time, the first step is repeated. Since a multiplier μ_i has been chosen by the aid of a base 2 Marsaglia table, the speed of the ellipsoid method depends essentially

on the speed of the generation of points uniformly distributed in hyperellipsoids.

The above-described decomposition can be illustrated geometrically in dimensions $n = 2$ as shown in Fig. 1. The volume between the surface

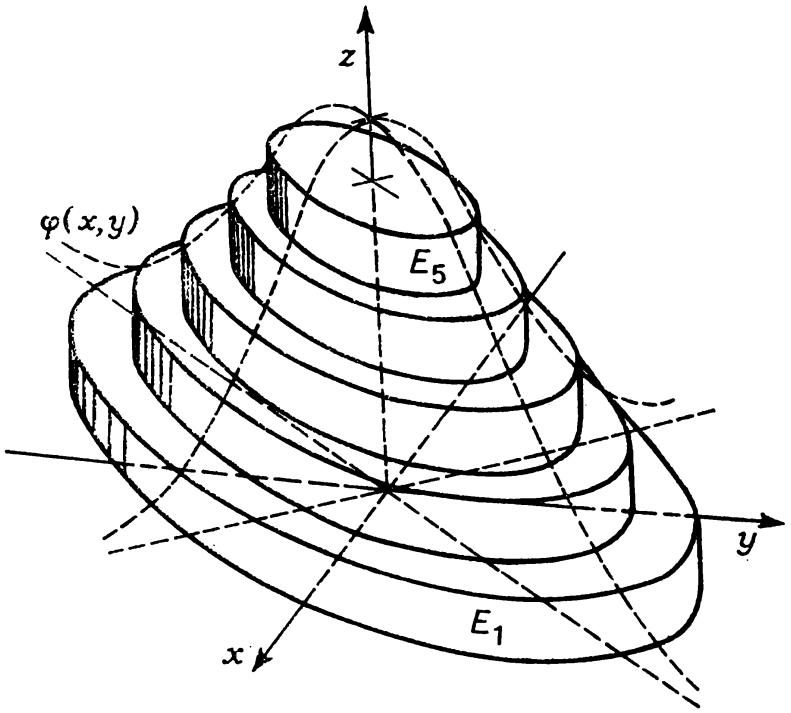

Fig. 1. The decomposition of the density function $z = \varphi(x, y)$, where the correlation coefficient ϱ is greater than 0, and the interval $[0, \varphi(0, 0)]$ is divided into 6 equal parts

$z = \varphi(x, y)$ and the plane $z = 0$ is filled out by prisms, one standing on the other. Their bases are the ellipses E_i $(i = 1, \ldots, 6)$ and their altitudes are $m_i - m_{i-1}$ long, each the same in this case. The function $p_{k+1} r_1(x, y)$ gives the remainder volume above the ellipses and the function $p_{k+2} r_2(x, y)$ is the tail.

3. Details of the sampling procedure. The main difficulty in preparing the program arose in connection with the division $m_1, \ldots, m_k$. We had the following three aims:

1. The sum $p_1 + \ldots + p_k$ should be near to 1 to allow fast sampling.

2. The number k should be possibly small because of the storage requirements.

3. The sampling from the wedge function $r_1(x)$ should not be too slow.

After some computer experiences the following scheme had been decided upon for determining the constants $m_1, \ldots, m_k$. Hyperellipsoids

are divided into classes, hyperellipsoids in the same class have the same distance, i.e., $m_i - m_{i-1} = m_j - m_{j-1}$, if E_i and E_j are in the same class.

Let us determine an initial division number k_{in} and two controlling constants $t_{cr} = 0.5/k_{in}$ and $w_{cr} = 0.3/k_{in}$. Divide the interval $[0, \varphi(\boldsymbol{0})]$ into k_{in} equal parts and compute the corresponding p_i $(i = 1, \ldots, k_{in})$ if

$$m_1 = \varphi(\boldsymbol{0})(1 - 1/k_{in}), \quad m_2 = \varphi(\boldsymbol{0})(1 - 2/k_{in}), \quad \ldots$$

(the division is going top-down). Let j be the index for which $p_i \leqslant w_{cr}$ if $i \leqslant j$ and $p_i > w_{cr}$ if $i > j$. Then the ellipsoids with indices $i = j+1, \ldots, k_{in}-1$ are deleted and the ellipsoids E_i $(i = 1, \ldots, j)$ form the first class. The remaining part of the interval $[0, \varphi(\boldsymbol{0})]$, i.e., $[0, \varphi(\boldsymbol{0})(k_{in} - j - 1)/k_{in}]$, is next divided into k_{in} equal parts and the whole procedure is repeated until the remaining tail has a probability less than t_{cr} (i.e., $t_{cr} \geqslant p_{k+2}$).

Let us introduce some notation. The length of the interval to be divided in the q-th class is denoted by h_q, the number of the ellipsoids in the q-th class is s_q, and the number of classes is nc. A special care should be paid in computing the values h_j $(j = 1, \ldots, nc)$, since the ratio h_1/h_{nc} is of range 10^6-10^{12} in cases $n \geqslant 10$. (The values h_j $(j = 1, \ldots, nc)$ and m_i $(i = 1, \ldots, k)$ are to be computed in double precision.)

Let V_0 be the volume of the unit hyperellipsoid $\{\boldsymbol{x} \mid \boldsymbol{x}' R^{-1} \boldsymbol{x} \leqslant 1\}$, i.e.,

$$V_0 = 2\gamma_1 \ldots \gamma_n (\pi)^{n/2} / \{n\Gamma(n/2)\},$$

where $\gamma_1, \ldots, \gamma_n$ are the eigenvalues of the correlation matrix; then

$$p_i = c_i^n V_0 (m_i - m_{i+1}).$$

The χ^2-distribution function with n degrees of freedom is denoted by F.

For $i = 1, \ldots, nc$ the following nc-dimensional vectors cp, cw, pw are to be stored:

$$cp(i) = p_1 + \ldots + p_{l_i}, \quad cw(i) = c_{l_i+1} \quad (l_i = s_1 + \ldots + s_i),$$

$$pw(i) = F\{cw^2(i)\} - cp(i).$$

This division procedure performs as shown in Tables 1 and 2. In the first case we chose $k_{i_n} = 50$, in the second one $- k_{i_n} = 100$.

In Tables 1 and 2 the following notation is used:

$$p_E = p_1 + \ldots + p_k, \quad p_w = pw(1) + \ldots + pw(nc) = p_{k+1}, \quad p_T = p_{k+2}.$$

The last columns show the maximum probability of rejecting a vector in the wedge sampling algorithm described in the sequel.

TABLE 1

n	p_E	p_w	p_T	nc	k	Maximum probability of rejecting
5	0.973	0.019	0.008	11	203	0.81
10	0.975	0.020	0.005	10	310	0.57
15	0.968	0.023	0.009	10	354	0.47
20	0.971	0.024	0.005	11	420	0.40

TABLE 2

n	p_E	p_w	p_T	nc	k	Maximum probability of rejecting
5	0.984	0.011	0.005	11	399	0.80
10	0.985	0.011	0.004	10	616	0.57
15	0.987	0.011	0.002	11	794	0.47
20	0.982	0.015	0.003	11	834	0.41

WEDGE SAMPLING ALGORITHM.

1. With probability $[pw(i)-(pw(1)+\ldots+pw(i-1))]/p_w$ select one of the classes, say the i-th one.

2. Generate a uniformly distributed random vector x in the hyperellipsoid with constant $cw(i)$. If x is inside the smallest hyperellipsoid of the i-th class (i.e., $x'R^{-1}x \leqslant (cw(i-1))^2$), then reject x and repeat this step from the beginning.

3. Generate a uniformly distributed number u in $[0, 1)$. If

$$\varphi(x) - \sum_{j=1}^{i} m_j \leqslant h_i u,$$

then go back to step 2, otherwise deliver x as a sample from the wedge with density $r_1(x)$.

For tail sampling the χ^2-distribution function F is inverted [8].

TAIL SAMPLING ALGORITHM.

1. Generate u uniformly in $[0, 1)$ and set $u \leftarrow u(1-p_T)+p_T$. Generate a uniformly distributed vector y in the greatest hyperellipsoid and set $s \leftarrow y'R^{-1}y$.

2. Set $r \leftarrow (F^{-1}(u)/s)^{1/2}$ and deliver $x \leftarrow ry$ as a sample from the tail with density $r_2(x)$.

4. The chord method for generating points uniformly distributed in hyperellipsoids. First some theoretical results are cited, next the sampling procedure is described and, finally, some considerations are given on the computer program.

Let us denote by S_r an n-dimensional hypersphere with radius r and with centre at the origin. Let Q_1 and Q_2 be two uniformly distributed random points on the surface of S_r.

THEOREM 1. *Let h be the length of the random chord Q_1Q_2. Then the density function of h is*

$$f(h) = \frac{\Gamma(n/2)}{2^{n-3}r^{2n-4}\Gamma\big((n-1)/2\big)\pi^{1/2}} h^{n-2}(4r^2-h^2)^{(n-3)/2}. \tag{5}$$

The result is well known in the theory of the geometrical probability (for a proof see [11], p. 53-54).

LEMMA. *The expected value of $\sigma_n = h^{4-n}$ is $h_0 = \mathrm{E}\{h^{4-n}\} = 4r^{4-n}/n$.*

The result can be obtained by straightforward integration.

The following theorem is the theoretical justification of the chord method:

THEOREM 2. *Consider a random chord in S_r with length h. On this chord generate uniformly distributed random points R_i, the number of which is a random variable $v = h^{4-n}/\lambda$, where λ is a suitable constant. Let S_ϱ be another n-dimensional hypersphere in S_r with the same centre and with radius ϱ, $\varrho \leqslant r$. Then the expected number of the points which lie in S_ϱ is proportional to ϱ^n, i.e.,*

$$\mathrm{E}\{v \mid R_i \in S_\varrho\} = \varrho^n C(n, r, \lambda),$$

where $C(n, r, \lambda)$ is independent of ϱ.

Proof. Let us make some preliminary remarks. A chord with length $h < h_\varrho$, where $h_\varrho = 2(r^2-\varrho^2)^{1/2}$, does not intersect the hypersphere S_ϱ. A chord with length $h \geqslant h_\varrho$ has an interval of length $(h^2-h_\varrho^2)^{1/2}$ in S_ϱ. Thus the expected number of points lying in S_ϱ equals to

$$I = \int_{h_\varrho}^{2r} f(h)\, \frac{h^{4-n}}{\lambda}\, \frac{(h^2-h_\varrho^2)^{1/2}}{h}\, dh,$$

where h^{4-n}/λ is the number of points to be generated on the random chord with length h and the last factor of the integrand is the portion of the points in S_ϱ.

Making use of the density function (5), introducing a new variable $y = (4r^2-h^2)^{1/2}$, then another variable α and putting $\sin\alpha = y/2\varrho$, we get

$$I = C_f \frac{1}{\lambda} \int_{h_\varrho}^{2r} h(4r^2-h^2)^{(n-3)/2}(h^2-h_\varrho^2)^{1/2}\, dh$$

$$= C_f \frac{1}{\lambda} \int_0^{2\varrho} y^{n-2}(4\varrho^2-y^2)^{1/2}\, dy = C_f \frac{(2\varrho)^n}{\lambda} \int_0^{\pi/2} \cos^2\alpha \sin^{n-2}\alpha\, d\alpha,$$

where C_f is the constant in the density function (5). This equation proves the theorem.

The chord method as described in the sequel is approximative because of two reasons. First, points generated on the same chord are correlated; the outcomes are not independent samples of the uniform distribution in the hypersphere. (But the commonly used multiplicative pseudo-random number generators produce points in a hypersphere which lie on hyperplanes, thus this argument is not so strong.) Second, the chords used in this procedure are not random because of making multiple use of the terminal points of previous chords.

Before the description of the chord method algorithm for the case of S_1, some preparations are needed.

Generate $\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(M)}$, the so-called basic points uniformly distributed on the surface of S_1, and initiate the following counters: $CP = 0$, $I = 1$, $ICH = 1$, $s = h_0 = 4/n$.

CHORD ALGORITHM FOR THE HYPERSPHERE S_1.

1. If $CP = 1$, then go to 4.

2. [Compute the (I, M)-th chord length.] Set $w_j \leftarrow y_j^{(I)} - y_j^{(M)}$ $(j = 1, \ldots, n)$ and $CP \leftarrow 1$.

3. Compute $SQ \leftarrow (w_1^2 + \ldots + w_n^2)^{(4-n)/2}$.

4. If $SQ < s$, then go to 6.

5. [Generate a "sure" point on the (I, M)-th chord.] Set $IG \leftarrow 1$, generate u uniformly in $[0, 1)$, set $x_j \leftarrow y_j^{(M)} + uw_j$ $(j = 1, \ldots, n)$, decrease $SQ \leftarrow SQ - s$ and go to 13.

6. Generate u and set $v \leftarrow su$.

7. [Test for an "additional" point.] If $SQ < v$, then set $IG \leftarrow 0$ and go to 9.

8. [Generate an "additional" point on the (I, M)-th chord.] Generate u, set $IG \leftarrow 1$ and $x_j \leftarrow y_j^{(M)} + uw_j$ $(j = 1, \ldots, n)$.

9. [Change the chord.] If $I \neq M-1$, then set $I \leftarrow I+1$, $CP \leftarrow 0$ and go to 12.

10. [Generate a new basic point, if necessary.] Generate a point $\boldsymbol{z}$ uniformly on the surface of S_1.

11. If $ICH = M-1$, then set $ICH \leftarrow 0$.

12. Set $ICH \leftarrow ICH+1$, $I \leftarrow 1$, $\boldsymbol{y}^{(ICH)} \leftarrow \boldsymbol{y}^{(M)}$, $\boldsymbol{y}^{(M)} \leftarrow \boldsymbol{z}$. If $IG = 0$, then go back to 2.

13. Deliver $\boldsymbol{x} = (x_1, \ldots, x_n)$ as a uniformly distributed point in S_1.

The algorithm generates points uniformly on the chords between the basic points 1 and M, then between basic points 2 and M, etc. After $M-1$ chords have been considered, one of the basic points is replaced

by $y^{(M)}$, and a new $y^{(M)}$ is generated. The number of the chosen basic points was equal to $M = 30$; the "scaling factor" $s = h_0$ (in Theorem 2 it was denoted by λ) ensures that one point is generated on one chord on the average. In computer runs the choice $s = h_0/4$ was also considered.

The algorithm shows significant instability for dimensions $n > 15$ in view of the following. The random variable h^{4-n} (where h is a random chord length) which controls the number of the points to be generated on one chord has infinite variance for $n > 4$. Chords with small length are rarely occurring but they require a great number of points to be generated on themselves. To illustrate this phenomenon, in Table 3 we give the probability of the event that a point generated by the chord algorithm originates from a chord on which more than k points are generated.

TABLE 3

k	$n = 5$	$n = 10$	$n = 15$
32	$2 \cdot 10^{-5}$	0.216	0.557
1024	$1 \cdot 10^{-9}$	0.044	0.286
$3.2 \cdot 10^4$	$1 \cdot 10^{-14}$	0.008	0.129
$1.0 \cdot 10^6$	–	$1 \cdot 10^{-3}$	0.054
$1.3 \cdot 10^8$	–	$1 \cdot 10^{-4}$	0.015
$8.5 \cdot 10^9$	–	$1 \cdot 10^{-5}$	0.005

Our proposal is to use the chord algorithm only if the sample size is greater than 10^5 in the dimension $n = 10$ and than 10^7 in the dimension $n = 15$.

The modification of the chord algorithm for the case of the hyperellipsoid is quite straightforward. Let us denote by T a triangular matrix for which $T'T = R$; then

$$E = \{\boldsymbol{x} \mid \boldsymbol{x}' R^{-1} \boldsymbol{x} \leqslant 1\} = \{\boldsymbol{x} \mid \boldsymbol{x} = T\boldsymbol{y}, \boldsymbol{y} \in S_1\}.$$

If $\boldsymbol{y}$ is a uniformly distributed point in S_1, then $T\boldsymbol{y}$ is uniformly distributed in E. Since the points generated by the chord algorithm are of the form $\boldsymbol{y} = a\boldsymbol{y}^{(I)} + (1-a)\boldsymbol{y}^{(M)}$, we have

$$T\boldsymbol{y} = aT\boldsymbol{y}^{(I)} + (1-a)T\boldsymbol{y}^{(M)}.$$

According to these equations it is sufficient to transform the basic points and there is no need of transforming the generated points. The following modifications should take place to gain an algorithm for the case of a hyperellipsoid.

As preparation – the points $\boldsymbol{y}^{(I)*} = T\boldsymbol{y}^{(I)}$ have to be computed and stored. Steps 2, 5, 8, 10 and 12 should be replaced by the corresponding modified versions:

2^*. Set $w_j \leftarrow y_j^{(I)} - y_j^{(M)}$, $w_j^* \leftarrow y_j^{(I)*} - y_j^{(M)*}$ $(j = 1, \ldots, n)$ and $CP \leftarrow 1$.

5^*. Generate u, set $x_j \leftarrow y_j^{(M)*} + uw_j^*$ $(j = 1, \ldots, n)$, $SQ \leftarrow SQ - s$ and go to 13.

8^*. Generate u, set $x_j \leftarrow y_j^{(M)*} + uw_j^*$ $(j = 1, \ldots, n)$.

10^*. Generate a point z uniformly on the surface of S_1 and compute $z^* \leftarrow Tz$.

12^*. Set $ICH \leftarrow ICH + 1$, $I \leftarrow 1$, $y^{(ICH)} \leftarrow y^{(M)}$, $y^{(ICH)*} \leftarrow y^{(M)*}$, $y^{(M)} \leftarrow z$, $y^{(M)*} \leftarrow z^*$. If $IG = 0$, then go back to 2.

5. Computer times and comparisons. The algorithms were written in FORTRAN and run on a Honeywell 66/60 computer. Two subroutines were written in assembly language (GMAP): a uniform random number generator which by two shifts and one addition generated numbers and the algorithm FL_5 of Ahrens and Dieter [2] for generating normally distributed numbers (one sample in 24 μsec and 60 μsec relatively).

In Table 4 we give the computer running times of generating one sample from a given distribution. Each time is given in μsec and is an average based on 10 000 samples. The algorithms considered are the following:

(a) *Chord, i, S.* It generates a uniformly distributed random vector inside the hypersphere S_1 as described in the preceding section ($\lambda = h_0 = \mathrm{E}\{h^{4-n}\}$).

(b) *Chord, s, S.* The same as in (a) but the resulting point is projected to the surface of S_1.

(c) *NO, s, S.* It generates a uniformly distributed point on the surface of S_1 by the normal approach of Muller [14] as described in Section 1.

(d) *NO, i, S.* It generates a point y by *NO, s, S* and a uniform random number u in $[0, 1)$, and then delivers $x = u^{1/n}y$ which is uniformly distributed in S_1.

(e) *Chord, i, E.* A point is generated inside the hyperellipsoid E by the chord algorithm using the modified steps with asterisks ($\lambda = h_0$).

(f) *NO, s, E.* It generates a point y by *NO, s, S* and delivers $x = Ty$, where $T'T = R$.

(g) *Crude.* This is the well-known technique for generating normally distributed random vectors with correlation matrix R: generate normally distributed, independent samples $x_1, \ldots, x_n$ and deliver $y = T(x_1, \ldots, x_n)$.

(h) *Ellipsoid*$_1$. It generates normally distributed random vectors by the ellipsoid & chord algorithm ($\lambda = h_0$, i.e., one point is generated on the average on each chord).

(i) *Ellipsoid*$_4$. The algorithm works as *Ellipsoid*$_1$ except one detail: the scaling factor $\lambda = h_0/4$ was chosen, thus on an average four points were generated on one chord.

TABLE 4

Algorithms	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 10$	$n = 15$
Chord, i, S	472	525	399	629	934	1413
Chord, s, S	622	701	603	872	1316	1859
NO, s, S	330	428	529	631	1130	1620
NO, i, S	584	679	779	883	1383	1865
Chord, i, E	501	575	473	717	1171	1188
NO, s, E	438	598	802	1032	2615	4763
Crude	287	450	637	849	2267	4267
Ellipsoid$_1$	697	802	704	998	1662	2760
Ellipsoid$_4$	403	461	467	585	897	1504

The first four rows show the chord algorithm to be competitive with the fastest exact method. According to the last three rows we recommend the ellipsoid & chord algorithm for the dimensions 5-15. When only a small sample size is required, one should use the crude algorithm, but in the case where a great sample size (100000, or more) is needed, the ellipsoid method may be preferred.

The additional programming effort due to the complexity of the ellipsoid & chord method, the storage requirements and the approximate character of the chord method are compensated by its speed.

The ellipsoid & chord method is applicable in a natural way to elliptically symmetric distributions which have density functions depending only on quadratic functions of the variable (see [10], p. 296, and [13]). The only change in the above-described algorithm concerns the constants c_i of the hyperellipsoids E_i. The most commonly known among those distributions is the multivariate t-distribution which has the density function

$$h(x) = \frac{\Gamma((\nu+n)/2)}{(\pi\nu)^{n/2}\Gamma(\nu/2)|R|^{1/2}}(1+\nu^{-1}x'R^{-1}x)^{-(\nu+n)/2},$$

where ν is the number of degrees of freedom of the corresponding χ-variable.

References

[1] J. H. Ahrens and U. Dieter, *Computer methods for sampling from the exponential and normal distributions*, Comm. ACM 15 (1972), p. 873-882.

[2] — *Extensions of Forsythe's method for random sampling from the normal distribution*, Math. Comp. 27 (1973), p. 927-937.

[3] D. R. Barr and N. L. Slezak, *A comparison of multivariate normal generators*, Comm. ACM 15 (1972), p. 1048-1049.

[4] J. C. Butcher, *Random sampling from the normal distribution*, Computer J. 3 (1961), p. 251-253.

[5] J. W. Butler, *Machine sampling from given probability distributions*, p. 249-264 in H. A. Meyer (editor), *Symposium on Monte Carlo methods*, New York 1956.
[6] I. Deák, *Comparison of methods for generating uniformly distributed random points in and on a hypersphere*, Probl. Contr. Inf. Theory 8 (1979), p. 105-113.
[7] — *Monte Carlo evaluation of the multidimensional normal distribution function by the ellipsoid method*, ibidem 7 (1978), p. 203-212.
[8] R. B. Goldstein, *Chi-square quantiles, Algorithm 451*, Comm. ACM 16 (1973), p. 483-485.
[9] R. L. Hurst and R. E. Knop, *Generation of random correlated normal variables*, ibidem 15 (1972), p. 355-357.
[10] N. L. Johnson and S. Kotz, *Distributions in statistics*, Vol. 4, New York 1972.
[11] M. G. Kendall and P. A. P. Moran, *Geometrical probability*, London 1963.
[12] G. Marsaglia, M. D. MacLaren and T. A. Bray, *A fast procedure for generating normal random variables*, Comm. ACM 7 (1964), p. 4-10.
[13] D. K. McGraw and J. F. Wagner, *Elliptically symmetric distributions*, IEEE Trans. Inform. Theory IT-14 (1968), p. 110-120.
[14] M. E. Muller, *A note on a method for generating points uniformly on n-dimensional spheres*, Comm. ACM 2 (1959), p. 19-20.
[15] E. M. Scheuer and D. S. Stoller, *On the generation of normal random vectors*, Technometrics 4 (1962), p. 278-281.

COMPUTER AND AUTOMATION INSTITUTE
HUNGARIAN ACADEMY OF SCIENCES
KENDE U. 13-17
H-1111 BUDAPEST, HUNGARY

Received on 19. 1. 1978

I. DEÁK (Budapest)

METODA ELIPSOID DLA GENEROWANIA WIELOWYMIAROWYCH ZMIENNYCH LOSOWYCH O ROZKŁADZIE NORMALNYM

STRESZCZENIE

Opisano metodę superpozycji dla generowania wielowymiarowych zmiennych losowych o rozkładzie normalnym. Algorytm oparty jest na rozłożeniu funkcji gęstości wielowymiarowego rozkładu normalnego i wykorzystuje generowanie zmiennych losowych o rozkładzie równomiernym w n-wymiarowych elipsoidach. Podana jest przybliżona metoda generowania takich punktów. Program komputerowy jest szybki przy założeniu, że może on zająć kilkaset słów roboczych pamięci. Do wygenerowania jednego wektora algorytm potrzebuje $2n$-$3n$ mnożeń i kilka mniejszych operacji. Z algorytmu można korzystać przy generowaniu wielowymiarowych zmiennych losowych, które mają symetryczne rozkłady eliptyczne (np. wielowymiarowy rozkład t).