Comp. & Maths. with Appls. Vol. 12B, No. 5/6, pp. 1187–1190, 1986
Printed in Great Britain.
0886-9561/86 $3.00 + 0.00

THE INCOMPLETE Γ FUNCTION AS A CONTINUOUS POISSON DISTRIBUTION

G. MARSAGLIA
Supercomputer Research Institute, Florida State University, Tallahassee, FL 32301-3033 U.S.A.

(*Received* 10 *April* 1985)

Abstract—Among the many contributions of Professor Luke to the theory of special functions, the most useful in computational statistics is probably that on the incomplete Γ function. This short paper points out that an incomplete Γ function routine is so important that it should be a standard part of any library of statistical subroutines. The paper goes on to give another example of use of the incomplete Γ function: as a means for computer generation of Poisson random variables, and, having urged wide use of the incomplete Γ function, proceeds with development of a Poisson generator whose principal aim is to avoid use of the very function is has previously lauded. Occasional use of an accurate incomplete Γ routine is essential however, in order that the composite method be exact.

INTRODUCTION

The incomplete Γ function is one of the most important of the special functions used in statistical computing. With suitable choice of arguments and division by the appropriate Γ value, it provides χ^2, Γ and Erlang distributions, as well as cumulative Poisson probabilities. Every library of scientific subroutines should have an accurate incomplete routine; unfortunately, many do not, as it is a difficult function over the full range of its two arguments. This paper provides yet another use for the incomplete Γ function in statistical computing: as a means to generate Poisson random variables in a computer. It is based on this version of the incomplete Γ function:

$$G(y, \lambda) = \int_{\lambda}^{\infty} t^{y-1} e^{-t}\, dt/\Gamma(y).$$

For fixed λ, G is a distribution function, increasing with y, with $G(0, \lambda) = 0$ and $G(\infty, \lambda) = 1$. It will be called a continuous Poisson distribution because, as integration by parts shows,

$$G(y+1, \lambda) - G(y, \lambda) = \lambda^y e^{-\lambda}/\Gamma(y+1).$$

Thus if Y is a continuous random variable with distribution $G(y, \lambda)$ then $\lfloor Y \rfloor$, the integer part of Y, has the discrete Poisson distribution,

$$\text{Prob}[\lfloor Y \rfloor = k] = G(k+1, \lambda) - G(k, \lambda) = \lambda^k e^{-\lambda}/k!.$$

GENERATION OF POISSON RANDOM VARIABLES

Is it feasible to generate a Poisson variate as the integer part of a continuous variate Y? A variety of methods for generating continuous variates have been developed, but the distribution function $G(y, \lambda)$ is formidable and its density function even more so. Thus, for example, squeeze methods based on the density function do not look promising.

Suppose we take an approach similar to that of the exact approximation method [1], in which a suitable approximation to the inverse distribution is used most of the time. Let $G^{-1}(u)$ be the solution to $G[G^{-1}(u), \lambda] = u$, i.e. G^{-1} is the inverse of the distribution function $G(y, \lambda)$, where, for economy, we suppress explicit mention of λ, assumed fixed. If U is a uniform random variable on (0,1) then $Y = G^{-1}(U)$ has distribution $G(y, \lambda)$, since G'ing both sides of $G^{-1}(U) < y$ gives

$$P[G^{-1}(U) < y] = P\{G[G^{-1}(U), \lambda] < G(y, \lambda)\} = P[U < G(y, \lambda)] = G(y, \lambda).$$

Thus to generate Y with distribution $G(y, \lambda)$ we need only put $Y = G^{-1}(U)$ for a uniform variate U. Unfortunately, G^{-1} is even more difficult than the notoriously difficult incomplete Γ function $G(y, \lambda)$. We will describe a method that avoids use of G^{-1}, using G instead and avoiding even the evaluation of G by using an easy-to-compute approximation to G^{-1} some 99% of the time.

Suppose we choose an easy-to-compute function $g(u)$ that is close to $G^{-1}(u)$. In fact, suppose we choose such a g so that

$$G^{-1}(u) < g(u) < G^{-1}(u) + \epsilon \quad \text{for} \quad 0 < u < 1,$$

with ϵ small. Since we only want the integer part of $G^{-1}(U)$, we may use the following algorithm, which avoids computing G^{-1}:

1. Generate a uniform variate U on (0, 1).
2. Put $Y = g(U)$ and $J = \lfloor Y \rfloor$.
3. If $Y - J > \epsilon$ return J.
4. If $U > G(J, \lambda)$ return J, else return $J - 1$.

If ϵ is small, say $\epsilon = 0.01$, then about 99% of the time the required discrete Poisson variate will be returned at Step 3, with only 1% of the cases calling for evaluation of $G(J, \lambda)$ in Step 4. Correctness of the algorithm follows from the assumption that $G^{-1}(u) < g(u) < G^{-1}(u) + \epsilon$ and the general result that $a < b < a + \epsilon$ and $b - \lfloor b \rfloor > \epsilon$ implies $\lfloor a \rfloor = \lfloor b \rfloor$.

For fixed λ, it is not difficult to develop easy-to-compute functions g for which $G^{-1}(u) < g(u) < G^{-1}(u) + \epsilon$ with ϵ small. But many simulations calling for Poisson variates have the parameter λ changing from call to call, so that the approximating function g must change with each new λ. To handle this more difficult case we exploit the near normality of Y for large λ, say $\lambda \geq 32$.

USING NORMAL RANDOM VARIABLES

We now change the problem. We want $\lfloor Y \rfloor$, the integer part of a variate Y with the continuous Poisson distribution $G(y, \lambda)$. If X is a standard normal variate, distribution $\Phi(x)$, then $\Phi(X)$ is uniform on (0, 1). If we can choose an easy-to-compute function $g(x)$ such that

$$G^{-1}[\Phi(x)] < g(x) < G^{-1}[\Phi(x)] + \epsilon,$$

then this algorithm will produce a discrete Poisson variate with parameter λ:

1. Generate a standard normal variate X.
2. Put $Y = g(X)$ and $J = Y = \lfloor Y \rfloor$.
3. If $Y - J > \epsilon$ return J.
4. If $\Phi(X) > G(J, \lambda)$ return J, else return $J - 1$.

The choice of g depends on λ, but g should be close to a linear or quadratic function, since Y is nearly normal for $\lambda \geq 32$. This g works fairly well:

$$g(x) = \lambda + 0.4076 + \lambda^{1/2}x + x^2/6 \quad \text{for} \quad |x| < 2.75.$$

Then $G^{-1}[\Phi(x)] < g(x) < G^{-1}[\Phi(x)] + 0.132$ for $|x| < 2.75$. Since a normal variate fails to satisfy $|x| < 2.75$ only six times a thousand, Step 3 in the above algorithm will provide the required Poisson variate some 87% of the time, requiring only a normal variate, a square root, two multiplications and three additions. The other 13% of the time either a more complex g (to handle $|X| > 2.75$) or computation of Φ and $G(J, \lambda)$ is required.

Reducing the need to compute Φ and $G(J, \lambda)$ to 13% may still make the average time for generating Poisson variates too long. Below are the results of numerical experimentation that provided a g for all $\lambda \geq 32$ with need to compute Φ and $G(J, \lambda)$ are reduced to less than 1%:

An Algorithm for Generating a Poisson Variate, Parameter $\lambda \geqslant 32$

Input: λ Output: the required Poisson variate

1. Generate a standard normal variate X.
2. If $|X| > 2.75$ go to Step 6.
3. Put $Y = c_0 + \lambda^{1/2}X + c_2X^2$ and $J = \lfloor Y \rfloor$, where, for $X \geqslant 0$.
   $c_0 = \lambda + 0.33615$, $c_2 = 1/6 - 0.04819/(\lambda^{1/2} + 0.7359)$, and for $X < 0$,
   $c_0 = \lambda + 0.33354$, $c_2 = 1/6 + 0.04819/(\lambda^{1/2} - 0.7857)$.
4. If $Y - J > 0.008$ return J.
5. If $\Phi(X) > G(J, \lambda)$ return J, else return $J - 1$.
6. With c_0, c_1, c_2, c_3 given below,
   put $Y = c_0 + c_1X + c_2X^2 + c_3X^3$, $J = \lfloor Y \rfloor$ and go to Step 4.

For $2.75 < x$ and $32 \leqslant \lambda \leqslant 100$,

$$c_0 = \lambda + 0.336755 - 0.17/(\lambda^{1/2} + 5.066),$$
$$c_1 = \lambda^{1/2} + 0.02875/(\lambda^{1/2} - 2.665),$$
$$c_2 = 1/6 - 0.0199/(\lambda^{1/2} - 2.365),$$
$$c_3 = -0.014/(\lambda^{1/2} + 5.066).$$

For $2.75 < x$ and $100 < \lambda < \infty$,

$$c_0 = \lambda + 0.334 - (7.99 + 4.836\lambda^{1/2} + 0.879\lambda)^{-1},$$
$$c_1 = \lambda^{1/2} + (425.1 - 75.7\lambda^{1/2} + 5.75\lambda)^{-1},$$
$$c_2 = 1/6 - (13.4 + 13.5\lambda^{1/2} + 2.455\lambda)^{-1},$$
$$c_3 = -(378 + 69\lambda^{1/2} + 0.07\lambda)^{-1}.$$

For $x < -2.75$ and $32 \leqslant \lambda \leqslant 100$, with $s = \lambda^{-1/2}$,

$$c_0 = \lambda + 0.7 - s[9.568 - s(83.43 - 269s)],$$
$$c_1 = \lambda^{1/2} + 0.3027 - s[7.953 - s(69.13 - 225.2s)],$$
$$c_2 = 0.24866 - s[2.15112 - s(18.73284 - 62.05328s)],$$
$$c_3 = 0.00736 - s[0.20752 - s(1.6664 - 5.7292s)].$$

For $x < -2.75$ and $100 < \lambda < \infty$,

$$c_0 = \lambda + 0.337 - (2.7 - 4.3\lambda^{1/2} + .81\lambda)^{-1},$$
$$c_1 = \lambda^{1/2} + (34.2 - 3.56\lambda^{1/2} - 0.36\lambda)^{-1},$$
$$c_2 = 1/6 - (9.54 - 12.43\lambda^{1/2} + 2.41\lambda)^{-1},$$
$$c_3 = (286 - 68.25\lambda^{1/2} - 0.09\lambda)^{-1}.$$

$$\Phi(x) = \int_{-\infty}^{x} e^{-t^2}\,dt/(2\pi)^{1/2}.$$

$$G(y, \lambda) = \int_{\lambda}^{\infty} t^{y-1}\,e^{-t}\,dt/\Gamma(y).$$

CONCLUDING NOTE

The method for generating Poisson variates via a continuous Poisson distribution was developed some 20 years ago, but has not been previously published in a research journal. The first edition (1969) of Donald Knuth's *The Art of Computer Programming*, Vol. 2 [2], includes a problem, No. 22, p. 135, with a difficulty rating of 41: "Can the exact Poisson distribution for large λ be obtained by generating an appropriate normal variate, converting it to an integer, and applying a (possibly complicated) correction a small percentage of the time?" (All problems in Professor Knuth's remarkable books are given a difficulty rating, ranging from 10 to 30, with occasional difficult research problems given higher ratings. Fermat's conjecture on $x^n + y^n = z^n$ is rated 50.)

Professor Knuth was not aware that I had considered that very problem; a brief description of the method developed above is in the second (1981) edition of Vol. 2, pp. 135 and 552. Several of my students have developed versions of the approximating function $g(x)$, notably Melvin Cohen at McGill and Gary Ngai at WSU [3].

REFERENCES

1. G. Marsaglia, The exact approximation method for generating random variables in a computer. *J. Am. statist. Ass.* **5**, No. 385, 218–221 (1984).
2. D. E. Knuth, *The Art of Computer Programming*; Vol. 2, *Seminumerical Algorithms*. Addison-Wesley, Reading, Mass. (1st edn, 1969; 2nd edn, 1981).
3. G. C.-Y. Ngai, Computer methods for efficient sampling from discrete distributions with variable parameters with emphasis on Poisson distributions Ph.D. Dissertation, Washington State Univ., Pullman, Wash. (1984).