Finally, the quick and dirty in-line generators `ranqd1` and `ranqd2` are very fast, but they are machine dependent, nonportable, and at best only as good as a 32-bit linear congruential generator ever is — in our view not good enough in many situations. We would use these only in very special cases, where speed is critical.

CITED REFERENCES AND FURTHER READING:

Park, S.K., and Miller, K.W. 1988, *Communications of the ACM*, vol. 31, pp. 1192–1201. [1]

Schrage, L. 1979, *ACM Transactions on Mathematical Software*, vol. 5, pp. 132–138. [2]

Bratley, P., Fox, B.L., and Schrage, E.L. 1983, *A Guide to Simulation* (New York: Springer-Verlag). [3]

Knuth, D.E. 1981, *Seminumerical Algorithms*, 2nd ed., vol. 2 of *The Art of Computer Programming* (Reading, MA: Addison-Wesley), §§3.2–3.3. [4]

Kahaner, D., Moler, C., and Nash, S. 1989, *Numerical Methods and Software* (Englewood Cliffs, NJ: Prentice Hall), Chapter 10. [5]

L'Ecuyer, P. 1988, *Communications of the ACM*, vol. 31, pp. 742–774. [6]

Forsythe, G.E., Malcolm, M.A., and Moler, C.B. 1977, *Computer Methods for Mathematical Computations* (Englewood Cliffs, NJ: Prentice-Hall), Chapter 10.

7.2 Transformation Method: Exponential and Normal Deviates

In the previous section, we learned how to generate random deviates with a uniform probability distribution, so that the probability of generating a number between x and $x + dx$, denoted $p(x)dx$, is given by

$$p(x)dx = \begin{cases} dx & 0 < x < 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.2.1}$$

The probability distribution $p(x)$ is of course normalized, so that

$$\int_{-\infty}^{\infty} p(x)dx = 1 \tag{7.2.2}$$

Now suppose that we generate a uniform deviate x and then take some prescribed function of it, $y(x)$. The probability distribution of y, denoted $p(y)dy$, is determined by the fundamental transformation law of probabilities, which is simply

$$|p(y)dy| = |p(x)dx| \tag{7.2.3}$$

or

$$p(y) = p(x)\left|\frac{dx}{dy}\right| \tag{7.2.4}$$

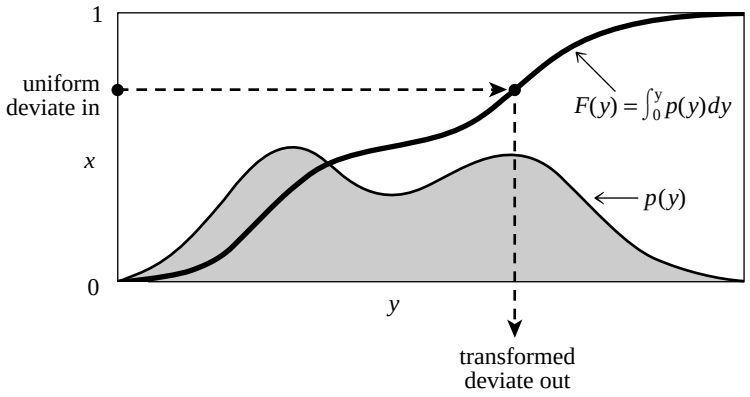

Figure 7.2.1. Transformation method for generating a random deviate y from a known probability distribution $p(y)$. The indefinite integral of $p(y)$ must be known and invertible. A uniform deviate x is chosen between 0 and 1. Its corresponding y on the definite-integral curve is the desired deviate.

Exponential Deviates

As an example, suppose that $y(x) \equiv -\ln(x)$, and that $p(x)$ is as given by equation (7.2.1) for a uniform deviate. Then

$$p(y)dy = \left|\frac{dx}{dy}\right| dy = e^{-y}dy \tag{7.2.5}$$

which is distributed exponentially. This exponential distribution occurs frequently in real problems, usually as the distribution of waiting times between independent Poisson-random events, for example the radioactive decay of nuclei. You can also easily see (from 7.2.4) that the quantity y/λ has the probability distribution $\lambda e^{-\lambda y}$.

So we have

```
      FUNCTION expdev(idum)
      INTEGER idum
      REAL expdev
C     USES ran1
        Returns an exponentially distributed, positive, random deviate of unit mean, using
        ran1(idum) as the source of uniform deviates.
      REAL dum,ran1
1     dum=ran1(idum)
      if(dum.eq.0.)goto 1
      expdev=-log(dum)
      return
      END
```

Let's see what is involved in using the above *transformation method* to generate some arbitrary desired distribution of y's, say one with $p(y) = f(y)$ for some positive function f whose integral is 1. (See Figure 7.2.1.) According to (7.2.4), we need to solve the differential equation

$$\frac{dx}{dy} = f(y) \tag{7.2.6}$$

But the solution of this is just $x = F(y)$, where $F(y)$ is the indefinite integral of $f(y)$. The desired transformation which takes a uniform deviate into one distributed as $f(y)$ is therefore

$$y(x) = F^{-1}(x) \tag{7.2.7}$$

where F^{-1} is the inverse function to F. Whether (7.2.7) is feasible to implement depends on whether the *inverse function of the integral of f(y)* is itself feasible to compute, either analytically or numerically. Sometimes it is, and sometimes it isn't.

Incidentally, (7.2.7) has an immediate geometric interpretation: Since $F(y)$ is the area under the probability curve to the left of y, (7.2.7) is just the prescription: choose a uniform random x, then find the value y that has that fraction x of probability area to its left, and return the value y.

Normal (Gaussian) Deviates

Transformation methods generalize to more than one dimension. If $x_1, x_2, \ldots$ are random deviates with a *joint* probability distribution $p(x_1, x_2, \ldots)\, dx_1 dx_2 \ldots$, and if $y_1, y_2, \ldots$ are each functions of all the x's (same number of y's as x's), then the joint probability distribution of the y's is

$$p(y_1, y_2, \ldots) dy_1 dy_2 \ldots = p(x_1, x_2, \ldots) \left| \frac{\partial(x_1, x_2, \ldots)}{\partial(y_1, y_2, \ldots)} \right| dy_1 dy_2 \ldots \tag{7.2.8}$$

where $|\partial(\quad)/\partial(\quad)|$ is the Jacobian determinant of the x's with respect to the y's (or reciprocal of the Jacobian determinant of the y's with respect to the x's).

An important example of the use of (7.2.8) is the *Box-Muller* method for generating random deviates with a normal (Gaussian) distribution,

$$p(y)dy = \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \tag{7.2.9}$$

Consider the transformation between two uniform deviates on (0,1), x_1, x_2, and two quantities y_1, y_2,

$$\begin{aligned} y_1 &= \sqrt{-2 \ln x_1} \cos 2\pi x_2 \\ y_2 &= \sqrt{-2 \ln x_1} \sin 2\pi x_2 \end{aligned} \tag{7.2.10}$$

Equivalently we can write

$$\begin{aligned} x_1 &= \exp\left[-\frac{1}{2}(y_1^2 + y_2^2) \right] \\ x_2 &= \frac{1}{2\pi} \arctan \frac{y_2}{y_1} \end{aligned} \tag{7.2.11}$$

Now the Jacobian determinant can readily be calculated (try it!):

$$\frac{\partial(x_1, x_2)}{\partial(y_1, y_2)} = \begin{vmatrix} \frac{\partial x_1}{\partial y_1} & \frac{\partial x_1}{\partial y_2} \\ \frac{\partial x_2}{\partial y_1} & \frac{\partial x_2}{\partial y_2} \end{vmatrix} = -\left[\frac{1}{\sqrt{2\pi}} e^{-y_1^2/2} \right] \left[\frac{1}{\sqrt{2\pi}} e^{-y_2^2/2} \right] \tag{7.2.12}$$

Since this is the product of a function of y_2 alone and a function of y_1 alone, we see that each y is independently distributed according to the normal distribution (7.2.9).

One further trick is useful in applying (7.2.10). Suppose that, instead of picking uniform deviates x_1 and x_2 in the unit square, we instead pick v_1 and v_2 as the ordinate and abscissa of a random point inside the unit circle around the origin. Then the sum of their squares, $R^2 \equiv v_1^2 + v_2^2$ is a uniform deviate, which can be used for x_1, while the angle that (v_1, v_2) defines with respect to the v_1 axis can serve as the random angle $2\pi x_2$. What's the advantage? It's that the cosine and sine in (7.2.10) can now be written as $v_1/\sqrt{R^2}$ and $v_2/\sqrt{R^2}$, obviating the trigonometric function calls!

We thus have

```
      FUNCTION gasdev(idum)
      INTEGER idum
      REAL gasdev
C     USES ran1
        Returns a normally distributed deviate with zero mean and unit variance, using ran1(idum)
        as the source of uniform deviates.
      INTEGER iset
      REAL fac,gset,rsq,v1,v2,ran1
      SAVE iset,gset
      DATA iset/0/
      if (idum.lt.0) iset=0                  Reinitialize.
      if (iset.eq.0) then                    We don't have an extra deviate handy, so
1       v1=2.*ran1(idum)-1.                  pick two uniform numbers in the square extend-
          ing from -1 to +1 in each direction,
        v2=2.*ran1(idum)-1.
        rsq=v1**2+v2**2                      see if they are in the unit circle,
        if(rsq.ge.1..or.rsq.eq.0.)goto 1     and if they are not, try again.
        fac=sqrt(-2.*log(rsq)/rsq)           Now make the Box-Muller transformation to get
          two normal deviates. Return one and save
          the other for next time.
        gset=v1*fac
        gasdev=v2*fac
        iset=1                               Set flag.
      else                                   We have an extra deviate handy,
        gasdev=gset                          so return it,
        iset=0                               and unset the flag.
      endif
      return
      END
```

See Devroye [1] and Bratley [2] for many additional algorithms.

CITED REFERENCES AND FURTHER READING:

Devroye, L. 1986, *Non-Uniform Random Variate Generation* (New York: Springer-Verlag), §9.1. [1]

Bratley, P., Fox, B.L., and Schrage, E.L. 1983, *A Guide to Simulation* (New York: Springer-Verlag). [2]

Knuth, D.E. 1981, *Seminumerical Algorithms*, 2nd ed., vol. 2 of *The Art of Computer Programming* (Reading, MA: Addison-Wesley), pp. 116ff.

7.3 Rejection Method: Gamma, Poisson, Binomial Deviates

The *rejection method* is a powerful, general technique for generating random deviates whose distribution function $p(x)dx$ (probability of a value occurring between x and $x + dx$) is known and computable. The rejection method does *not* require that the cumulative distribution function [indefinite integral of $p(x)$] be readily computable, much less the inverse of that function — which was required for the transformation method in the previous section.

The rejection method is based on a simple geometrical argument:

Draw a graph of the probability distribution $p(x)$ that you wish to generate, so that the area under the curve in any range of x corresponds to the desired probability of generating an x in that range. If we had some way of choosing a random point *in two dimensions*, with uniform probability in the *area* under your curve, then the x value of that random point would have the desired distribution.

Now, on the same graph, draw any other curve $f(x)$ which has finite (not infinite) area and lies everywhere *above* your original probability distribution. (This is always possible, because your original curve encloses only unit area, by definition of probability.) We will call this $f(x)$ the *comparison function*. Imagine now that you have some way of choosing a random point in two dimensions that is uniform in the area under the comparison function. Whenever that point lies outside the area under the original probability distribution, we will *reject* it and choose another random point. Whenever it lies inside the area under the original probability distribution, we will *accept* it. It should be obvious that the accepted points are uniform in the accepted area, so that their x values have the desired distribution. It should also be obvious that the fraction of points rejected just depends on the ratio of the area of the comparison function to the area of the probability distribution function, not on the details of shape of either function. For example, a comparison function whose area is less than 2 will reject fewer than half the points, even if it approximates the probability function very badly at some values of x, e.g., remains finite in some region where x is zero.

It remains only to suggest how to choose a uniform random point in two dimensions under the comparison function $f(x)$. A variant of the transformation method (§7.2) does nicely: Be sure to have chosen a comparison function whose indefinite integral is known analytically, and is also analytically invertible to give x as a function of "area under the comparison function to the left of x." Now pick a uniform deviate between 0 and A, where A is the total area under $f(x)$, and use it to get a corresponding x. Then pick a uniform deviate between 0 and $f(x)$ as the y value for the two-dimensional point. You should be able to convince yourself that the point (x, y) is uniformly distributed in the area under the comparison function $f(x)$.

An equivalent procedure is to pick the second uniform deviate between zero and one, and accept or reject according to whether it is respectively less than or greater than the ratio $p(x)/f(x)$.

So, to summarize, the rejection method for some given $p(x)$ requires that one find, once and for all, some reasonably good comparison function $f(x)$. Thereafter, each deviate generated requires two uniform random deviates, one evaluation of f (to get the coordinate y), and one evaluation of p (to decide whether to accept or reject