Contents

4

Non-uniform random variables

Uniform random variables are the basic building block for Monte Carlo methods. Often, the first thing we do with them is construct the non-uniform random variables that our problem requires. Most widely used mathematical computing environments include generators for a wide selection of non-uniform distributions. The ones with famous names (normal, Poisson, binomial, exponential gamma, etc.) might all be implemented for us. But from time to time we need to sample from a distribution that our environment does not include, perhaps because our application is the first to require it. Then we need to understand how non-uniform random numbers are generated in order to make a custom solution. Also, the methods for generating random vectors and processes as well as the way in which Markov chain Monte Carlo works, are based on the same ideas that we use to generate non-uniform scalar random variables. Therefore even when we have all the univariate distributions we need, it pays to understand how they are obtained.

This chapter shows how to get non-uniform random variables from uniform ones. We will look at general principles like inversion and acceptance-rejection sampling that apply to many settings. We will also include some very specific techniques, tricks almost, that work wonders on a few problems, but are not general purpose tools.

4.1 Inverting the CDF

The most direct way to convert uniform into non-uniform random variables is by inverting the cumulative distribution function. This is the gold standard method. It is available in principle for every distribution, and it allows very powerful variance reduction methods (see Chapter 8) to be applied, and so we

look at it first. Unfortunately, it is often very hard to do and so we also look at alternatives.

The distribution of a real valued random variable X can be completely specified through it's ***cumulative distribution function*** (CDF)

$$F(x) = \mathbb{P}(X \leqslant x). \tag{4.1}$$

For a proper distribution, $F(\infty) = 1$ and $F(-\infty) = 0$. A CDF is continuous from the right: $\lim_{x' \to x^+} F(x') = F(x)$.

There are two main kinds of real random variables, continuous and discrete. When X has a continuous distribution then it has a probability density function $f(x) \geqslant 0$ satisfying

$$\mathbb{P}(a \leqslant X \leqslant b) = \int_a^b f(x)\,\mathrm{d}x = F(b) - F(a),$$

for $-\infty \leqslant a \leqslant b \leqslant \infty$. In particular $\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = 1$.

When X has a discrete distribution then

$$\mathbb{P}(X = x_k) = p_k$$

for $0 \leqslant k < N$ where $p_k \geqslant 0$ and $\sum_k p_k = 1$. The values x_k are called atoms. Often x_k are integers or natural numbers. The number N of distinct values can be finite or countably infinite.

Sometimes we need distributions with both discrete and continuous parts. If F_d is a discrete CDF, F_c is a continuous one, and $0 < \lambda < 1$, then $\lambda F_d + (1-\lambda)F_c$ is the CDF of a distribution with atoms from F_d and a density which comes from F_c.

Inversion is simplest for continuous distributions. Suppose that the random variable X has PDF $f(x) > 0$ for all $x \in \mathbb{R}$. Then $F(x)$ is monotone and continuous and it has an inverse F^{-1}. If we generate $U \sim \mathbf{U}(0,1)$ and put $X = F^{-1}(U)$, then

$$\begin{aligned}\mathbb{P}(X \leqslant x) &= \mathbb{P}(F^{-1}(U) \leqslant x) = \mathbb{P}(F(F^{-1}(U)) \leqslant F(x)) \\ &= \mathbb{P}(U \leqslant F(x)) = F(x).\end{aligned}$$

Therefore $X \sim F$.

For some distributions, we run into problems with this recipe. For example suppose that $f(x) = 0$ for all x in the interval $[a, b]$ with $a < b$ and let $u = F(a)$. Then $F(x) = u$ for all $a \leqslant x \leqslant b$ and so F does not have a unique inverse at u. Perhaps worse, suppose that X is a discrete random variable with $F(x) > \lim_{\epsilon \to 0^+} F(x - \epsilon) \equiv F(x-)$. Then for u in the interval $[F(x-), F(x))$, there is no value x' with $F(x') = u$.

Both of these problems are solved by defining

$$F^{-1}(u) = \inf\{\, x \mid F(x) \geqslant u \,\}, \quad 0 < u < 1. \tag{4.2}$$

The set in (4.2) is never empty, and it always attains its infimum because F is continuous from the right.

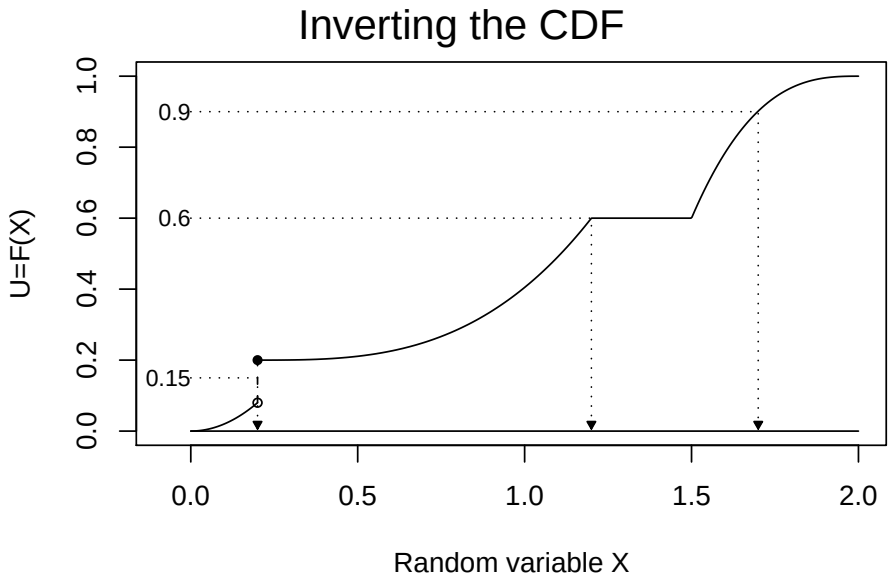

Figure 4.1: This figure illustrates inversion of a CDF by equation (4.2). The random variable X takes values in the interval $[0, 2]$. The CDF of X is shown as a solid curve. It has a jump at $x = 0.2$ and it places zero probability in the interval $[1.2, 1.5]$. Inversion of F is depicted with a dotted line for three points: $F^{-1}(0.15) = 0.2$, $F^{-1}(0.6) = 1.2$, and $F^{-1}(0.9) \doteq 1.7$.

Points $u = 0$ and 1 have zero probability of arising when u is an observed $\mathbf{U}(0, 1)$ value, but they could arise numerically. We extend equation (4.2) to these values via $F^{-1}(1) = \lim_{u\to 1^-} F^{-1}(u)$ and $F^{-1}(0) = \lim_{u\to 0^+} F^{-1}(u)$ obtaining $\pm\infty$ for unbounded distributions like $\mathcal{N}(0, 1)$.

Figure 4.1 illustrates the solution. When $F(x) = u$ for all $u \in [a, b]$ then $F^{-1}(u)$ is the left endpoint a of that horizontal interval. When $F(x) > F(x-)$ there is a vertical interval of height $F(x) - F(x-)$ such that u in this interval gives $F^{-1}(y) = x$. From the geometry in Figure 4.1 it is clear that equation (4.2) should handle these cases well, but the point is important enough to prove.

Theorem 4.1. *Let F be a cumulative distribution function and let F^{-1} be its inverse as given by* (4.2)*. If $U \sim \mathbf{U}(0, 1)$ and $X = F^{-1}(U)$ then $X \sim F$.*

Proof. It is enough to show that $F(x) \leqslant u$ if and only if $x \leqslant F^{-1}(u)$ for $0 < u < 1$. Both F and F^{-1} are nondecreasing. Because the infimum in equation (4.2) is attained, $F^{-1}(u) \in \{x \mid F(x) \geqslant u\}$ and so $F(F^{-1}(u)) \geqslant u$. Similarly, $F^{-1}(F(x)) \leqslant x$.

Now suppose that $F^{-1}(u) \leqslant x$. Reversing it, we get $x \geqslant F^{-1}(u)$, so $F(x) \geqslant F(F^{-1}(u)) \geqslant u$, from which $u \leqslant F(x)$. Conversely, suppose that $u \leqslant F(x)$. Then $F^{-1}(u) \leqslant F^{-1}(F(x)) \leqslant x$. □

Inversion allows very sharp investigation of the effects on a simulation of different input distributions. If there are three candidate distributions F, G, and H for a random variable X in a simulation, then generating X_i via $F^{-1}(U_i)$, $G^{-1}(U_i)$ and $H^{-1}(U_i)$ respectively allows us to make better comparisons of these three distributions than we would get simulating them separately. This is the method of common random numbers from §8.6. Inversion is also very convenient for antithetic sampling (§8.2) and stratification (§8.4).

When $U \sim \mathbf{U}(0,1)$ then $1-U \sim \mathbf{U}(0,1)$ too. It follows that $F^{-1}(1-U) \sim F$ as well. Sometimes this ***complementary inversion*** formula ends up being a little simpler to use. When it is important for large values of X to correspond to large values of U, for example when comparing two simulations, then we need the ordinary inversion method.

We defined inversion to operate on $\mathbf{U}(0,1)$ random variables, but the idea works more generally. If F is a continuous CDF and G is any CDF, then

$$Y = G^{-1}(F(X)) \tag{4.3}$$

produces a G distributed variable Y from an F distributed variable X, because $F(X) \sim \mathbf{U}(0,1)$. The function $G^{-1}(F(\cdot))$ is called the ***QQ transformation*** because it transforms quantiles of the distribution F into corresponding quantiles of G. Sometimes $G^{-1}(F(\cdot))$ has a simpler form than either G^{-1} or F so we can go directly from $X \sim F$ to $Y = (G^{-1} \circ F)(X) \sim G$ without first passing through $\mathbf{U}(0,1)$. The construction $Y = \mu + \sigma Z$, with $\sigma > 0$, for generating $Y \sim \mathcal{N}(\mu, \sigma^2)$ via $Z \sim \mathcal{N}(0,1)$ is an example of (4.3) that we use automatically without thinking of inversion.

If $X = F^{-1}(U)$ for $U \sim \mathbf{U}(0,1)$ where F is a discrete distribution, then $F(X)$ does not have the $\mathbf{U}(0,1)$ distribution and so (4.3) will not work in general for discrete F. See Exercise 4.3.

4.2 Examples of inversion

Many important univariate distributions can be sampled by inversion using simple closed form expressions. Some of the most useful ones are listed here.

Example 4.1 (Exponential distribution)**.** The standard exponential distribution has density $f(x) = e^{-x}$ on $x > 0$. If X has this distribution, then $\mathbb{E}(X) = 1$, and we write $X \sim \mathrm{Exp}(1)$. The cumulative distribution function is $F(x) = \mathbb{P}(X \leqslant x) = 1 - e^{-x}$, with $F^{-1}(u) = -\log(1-u)$. Therefore taking $X = -\log(1-U)$ for $U \sim \mathbf{U}(0,1)$, generates standard exponential random variables. Complementary inversion uses $X = -\log(U)$.

The exponential distribution with rate $\lambda > 0$ (and mean $\theta = 1/\lambda$) has PDF $\lambda \exp(-\lambda x)$ for $0 \leqslant x < \infty$. If X has this distribution, then we write $X \sim \mathrm{Exp}(1)/\lambda$ or equivalently $X \sim \theta\mathrm{Exp}(1)$, depending on whether the problem is more naturally formulated in terms of the rate λ or mean θ. We may generate X by taking $X = -\log(1-U)/\lambda$.

The exponential distribution has a memoryless property in that

$$\mathbb{P}(X \geqslant x + \Delta \mid X \geqslant x) = \frac{e^{-\lambda(x+\Delta)}}{e^{-\lambda x}} = e^{-\lambda\Delta}$$

does not depend on x. Exponential distributions are therefore unsuitable as a model for the lifetime of a product that wears out over time. The Weibull distribution of Example 4.7 provides some better alternatives.

Example 4.2 (Bernoulli distribution)**.** A Bernoulli random variable X takes the value 1 with probability p and 0 with probability $1-p$. We write $X \sim \mathrm{Bern}(p)$. To sample the Bernoulli distribution by inversion, take $X = \mathbb{1}_{1-U\leqslant p}$. Complementary inversion is simpler: $X = \mathbb{1}_{U\leqslant p}$.

Example 4.3 (Cauchy distribution)**.** The Cauchy distribution is used to model random variables with very heavy tails. The standard Cauchy distribution has probability density function $f(x) = (1/\pi)(1+x^2)^{-1}$ for $x \in \mathbb{R}$. This density decays so slowly that $\mathbb{E}(|X|) = \infty$. It has cumulative distribution function $F(x) = (1/\pi)\arctan(x) + 1/2$. Using inversion, we can sample the Cauchy distribution by taking $X = \tan(\pi(U - 1/2))$ where $U \sim \mathbf{U}(0,1)$. Geometrically, it is the tangent of a random angle uniformly distributed on $(-\pi/2, \pi/2)$. Cauchy variables can be used to stress test algorithms that are meant to handle mostly Gaussian random variables while being robust to the presence of some extremely large values.

Example 4.4 (Discrete uniform distribution)**.** For the discrete uniform distribution on $\{0, 1, \ldots, k-1\}$, take $X = \lfloor kU \rfloor$. It is safer to use $\min\{\lfloor kU \rfloor, k-1\}$ in case $U = 1$ is possible. If we prefer $\mathbf{U}\{1, \ldots, k\}$, then we take $X = \lceil kU \rceil$ or better, $\max\{1, \lceil kU \rceil\}$, in case $U = 0$ is possible. Notice that these discrete uniform variables are determined by the higher order bits of U. Those bits are usually more approximately random than the lower order bits.

Example 4.5 (Poisson distribution)**.** The Poisson distribution with mean $\lambda > 0$ has $\mathbb{P}(X = x) = e^{-\lambda}\lambda^x/x!$ for integers $x \geqslant 0$. Algorithm 4.1 describes how to sample this distribution by inversion. The test $U > q$ is made $X + 1$ times and so the number of iterations required is $\mathbb{E}(X+1) = \lambda + 1$. Therefore this method is slow if λ is very large. Careful coding has to account for the possibility that q can converge to a number below 1 (if p goes below the machine epsilon). This could cause the algorithm to loop indefinitely if U is larger than the limiting q.

Example 4.6 (Normal distribution)**.** If $U \sim \mathbf{U}(0,1)$ and $Z = \Phi^{-1}(U)$ then $Z \sim \mathcal{N}(0,1)$. Unfortunately, neither Φ nor Φ^{-1} is available in closed form. This is the most important distribution for which inversion can be hard to do. Very accurate numerical approximations to Φ^{-1} are now widely implemented and some are described in §4.3, so inversion is feasible for $\mathcal{N}(0,1)$. A convenient alternative to inversion is provided by the celebrated Box-Muller transformation, described in §4.6.

Algorithm 4.1 Sample from the Poisson distribution with mean λ

$X \leftarrow 0, \quad p \leftarrow q \leftarrow e^{-\lambda}, \quad U \sim \mathbf{U}(0,1)$
while $U > q$ **do**
  $X \leftarrow X + 1$
  $p \leftarrow p\lambda/X$
  $q \leftarrow q + p$
deliver X

This algorithm generates a Poisson(λ) random variable by inversion. It takes about $1 + \lambda$ steps on average. Faster algorithms are available for large λ. It is from Devroye (1986).

Example 4.7 (Weibull distribution)**.** The Weibull distribution generalizes the exponential. It has PDF

$$f(x;\sigma,k) = \frac{k}{\sigma}\Big(\frac{x}{\sigma}\Big)^{k-1} e^{-(x/\sigma)^k}, \quad 0 < x < \infty,$$

for parameters $\sigma > 0$ and $k > 0$. When $k = 1$, it reduces to the exponential distribution with mean σ (rate $1/\sigma$). The CDF of the Weibull distribution is $1 - \exp(-(x/\sigma)^k)$. To sample it by inversion, we take $X = \sigma(-\log(1-U))^{1/k}$ where $U \sim \mathbf{U}(0,1)$. The parameter σ simply rescales the distribution, while k changes its shape. The mean of a Weibull random variable is $\mu = \sigma\Gamma(1 + 1/k)$, where Γ is the Gamma function (equation 4.13 in §4.8) and the Weibull variance is $\sigma^2\Gamma(1 + 2/k) - \mu^2$. Exercise 4.18 investigates an elementary property (the hazard function) of the Weibull distribution.

Example 4.8 (Double exponential distribution)**.** The standard double exponential distribution has density $f(x) = \frac{1}{2}\exp(-|x|)$ for $x \in \mathbb{R}$. The CDF and inverse CDF of this distribution are

$$F(x) = \begin{cases} \frac{1}{2}\,e^{x}, & x < 0 \\ 1 - \frac{1}{2}\,e^{-x}, & x \geqslant 0 \end{cases}$$

and

$$F^{-1}(u) = \begin{cases} \log(2u), & 0 < u \leqslant 1/2 \\ -\log(2(1-u)), & 1/2 < u < 1, \end{cases}$$

respectively.

Example 4.9 (Gumbel distribution)**.** The Gumbel distribution with parameters $\mu \in \mathbb{R}$ and $\sigma > 0$ has CDF $F(x;\mu,\sigma) = \exp(-e^{-(x-\mu)/\sigma})$ for $x \in \mathbb{R}$. The standard version with $\mu = 0$ and $\sigma = 1$ has CDF $F(x) = \exp(-e^{-x})$ for $x \in \mathbb{R}$ and probability density function

$$f(x) = \exp(-x - e^{-x}).$$

Taking $X = -\log(-\log(U))$ for $U \sim \mathbf{U}(0,1)$ generates a sample from the standard Gumbel distribution. The general case has $X = \mu - \sigma\log(-\log(U))$. The parameters μ and σ shift and scale the random variable, but they are not the mean and standard deviation of X. The Gumbel distribution is also known as the extreme value distribution of type I. If Y_i are IID from a distribution G then $X = \max(Y_1, \dots, Y_n)$ has approximately a Gumbel distribution for large n and many (though not all) distributions G.

Example 4.10 (Truncated distributions)**.** Let F be a CDF and suppose that X has the distribution F restricted to the interval (a, b) where $-\infty \leqslant a < b \leqslant \infty$. To generate X, we want to sample from F conditionally on $a < X < b$. The problem only makes sense if the distribution F places positive probability on the interval (a, b), so we assume that it does. The distribution is said to be truncated to the interval (a, b).

Suppose first that F is continuous with density f. Then the random variable X has density

$$g(x) = \begin{cases} f(x)\big/\int_a^b f(x)\,\mathrm{d}x, & a < x < b \\ 0, & \text{else.} \end{cases}$$

The truncated random variable can be sampled by taking

$$X = F^{-1}\big(F_a + (F_b - F_a)U\big), \quad \text{where} \quad U \sim \mathbf{U}(0,1), \tag{4.4}$$

and $F_a = F(a) < F_b = F(b)$. In the continuous case, we get the same answer whether we truncate F to open, closed or half-open intervals.

The discrete case is a little trickier because of the possibility that F might have an atom at a or b or both. It turns out that formula (4.4) yields a random variable with the distribution of F truncated to the half-open interval $(a, b]$. If we want to include a then we simply take $F_a = F(a-) \equiv \lim_{x\to a^-} F(a)$. Similarly, to exclude b we take $F_b = F(b-)$.

Example 4.11 (Triangular and power densities)**.** The density $f(x) = 2x$ on $0 \leqslant x \leqslant 1$ is a triangular density. It has CDF $F(x) = x^2$ and so $F^{-1}(U) = \sqrt{U}$ has the triangular distribution when $U \sim \mathbf{U}(0,1)$. Similarly $f(x) = \alpha x^{\alpha-1}$ for $\alpha > 0$ may be sampled by taking $X = U^{1/\alpha}$ for $U \sim \mathbf{U}(0,1)$.

The density $2(1-x)$ on $0 \leqslant x \leqslant 1$ is also triangular and may be sampled via $X = 1 - \sqrt{1-U}$. Similarly, to sample from $f(x) = \beta(1-x)^{\beta-1}$ for $\beta > 0$ one takes $X = 1 - (1-U)^{1/\beta}$. We will look at a potentially faster generator of triangular random variables in §4.6. These triangular and power densities are also special cases of the Beta distribution of Example 4.29 of §4.8 which in turn is a special case of the Dirichlet distribution of §5.4.

4.3 Inversion for the normal distribution

We write

$$\varphi(z) = \frac{e^{-z^2/2}}{\sqrt{2\pi}}, \quad -\infty < z < \infty,$$

Algorithm 4.2 Approximate $\Phi^{-1}(u)$ by AS70 of Odeh and Evans (1974).

if $u > 1/2$ **then**
  return $-\Phi^{-1}(1-u)$
if $u < 10^{-20}$ **then**
  $u \leftarrow 10^{-20}$ // This algorithm is not intended for the extreme tails of $\mathcal{N}(0,1)$
$v \leftarrow \sqrt{-2\log(u)}$
$A \leftarrow (((a_4 \times v + a_3) \times v + a_2) \times v + a_1) \times v + a_0$
$B \leftarrow (((b_4 \times v + b_3) \times v + b_2) \times v + b_1) \times v + b_0$
return $-(v + A/B)$

This algorithm approximates Φ^{-1} by the method of Odeh and Evans (1974). It uses constants a_i and b_i from Table 4.1. The published algorithm returned 0 and signalled an error for $u \notin [10^{-20}, 1-10^{-20}]$. A later algorithm due to Wichura is more accurate, though less compactly written.

for the $\mathcal{N}(0,1)$ probability density function. The $\mathcal{N}(0,1)$ CDF is

$$\Phi(z) = \int_{-\infty}^{z} \varphi(x)\,\mathrm{d}x$$

and so we can, in principle, sample $\mathcal{N}(0,1)$ via $Z = \Phi^{-1}(U)$ where $U \sim \mathbf{U}(0,1)$. Many computing environments include Φ^{-1}, and then it is straightforward to use inversion for the normal distribution.

Neither Φ nor Φ^{-1} has a closed form expression. Sometimes we must obtain or write our own version of Φ^{-1}. For the widely used programming languages, the odds are good that a suitable version of Φ^{-1} is available on the Internet, either in that language or one that translates easily. It is safest to use a function that has been well tested and for which a published analysis is available.

The function Φ is closely related to $\mathrm{erf}(z) = 2\pi^{-1/2}\int_0^z e^{-t^2}\,\mathrm{d}t$, known as the ***error function***. The error function can be used to get Φ via $\Phi(z) = (1+\mathrm{erf}(z/\sqrt{2}))/2$ and similarly $\Phi^{-1}(u) = \sqrt{2}\,\mathrm{erf}^{-1}(2u-1)$. Thus, the many published algorithms and tables for the error function can be used for Φ.

Algorithm 4.2 presents a very simple approximation of Φ^{-1} due to Odeh and Evans (1974). It requires 10 rather arbitrary looking constants, most given to 12 significant figures. For this simple algorithm, the estimate $\hat{\Phi}^{-1}_{\mathrm{OE}}$ satisfies

$$|\hat{\Phi}^{-1}_{\mathrm{OE}}(u) - \Phi^{-1}(u)| \leqslant 1.5 \times 10^{-8}, \quad \text{for} \quad 10^{-20} \leqslant u \leqslant 1/2,$$

if computed in double precision. For $u > 1/2$ we may lose some accuracy if we have to explicitly compute $1-u$ as the algorithm indicates. For example, with u very near 1 a floating point computation of $1-u$ could produce 1 as a result.

Algorithm AS70 is hardly the last word in approximations to Φ^{-1}. Beasley and Springer (1977) published algorithm AS111 which they find is about twice as fast as AS70. (The algorithm numbering is chronological from the journal Applied Statistics.) Their error analysis has $\hat{\Phi}^{-1}_{\mathrm{BS}}(u) = \Phi^{-1}(\widetilde{u})$ for some $\widetilde{u}$ with $|\widetilde{u} - u| \leqslant 2^{-31}$, and assuming no roundoff error. The result is not particularly

i	a_i	b_i
0	−0.3222 3243 1088	0.0993 4862 6060
1	−1.0	0.5885 8157 0495
2	−0.3422 4208 8547	0.5311 0346 2366
3	−0.0204 2312 1024 5	0.1035 3775 2850
4	−0.0000 4536 4221 0148	0.0038 5607 0063 4

Table 4.1: These are the constants used in the Odeh and Evans approximation to Φ^{-1} given in Algorithm 4.2.

accurate in the tails of the distribution because changing u by 2^{-31} makes a big difference there.

Wichura (1988) published algorithm AS241 for the same problem. The double precision version yields a relative error

$$\frac{|\hat{\Phi}_W^{-1}(u) - \Phi^{-1}(u)|}{|\Phi^{-1}(u)|} < 10^{-15}, \quad \text{for} \quad \min(u, 1-u) > 10^{-316},$$

interpreting $0/0$ as 0 for $u = 1/2$. The relative error is close to the inherent limit of the double precision representation and so is the range of values u where this accuracy is obtained.

Algorithm AS241 is reliable much farther into the tails than either AS70 or AS111 is. It should be used instead of AS70 shown in Algorithm 4.2 when extreme values of u are required. The full algorithm includes dozens of high precision constants, so it is better obtained electronically than from the printed page. AS241 is easily found on the Internet.

One of the most ambitious approximations to Φ^{-1} is that of Blair et al. (1976). They approximate the inverse error function. They present a method that gives at least 22 digits of accuracy (relative error) for $\mathrm{erf}^{-1}(v)$ over $0 \leqslant v \leqslant 1 - 10^{-10,000}$ and at least 25 digits (absolute error) over $1 - 10^{-10,000} \leqslant v < 1$. Values for $-1 < v < 0$ are obtained via $\mathrm{erf}^{-1}(-v) = -\mathrm{erf}^{-1}(v)$. They break the range for v into four regions and use rational function approximations for each one. They supply about 20 different approximations for each interval, allowing the user to trade off speed versus accuracy.

4.4 Inversion for discrete random variables

Inversion is easy to use for discrete random variables taking finitely many values. We will suppose for now that those values are $1, \dots, N$ inclusive, but switch to different sets as needed. Let X be a random variable with $\mathbb{P}(X = k) = p_k \geqslant 0$ for $k = 1, \dots, N < \infty$. For $k = 1, \dots, N$, let $P_k = \sum_{i=1}^{k} p_i$ be the cumulative distribution and adjoin $P_0 = 0$. Then

$$F^{-1}(u) = k, \quad \text{where} \quad P_{k-1} < u \leqslant P_k. \tag{4.5}$$

Algorithm 4.3 Bisection-based inversion of a CDF on $\{1, \dots, N\}$.

Bisectinvert$(u, N, P_{0:N})$
if $u = 0$ **then**
  return $\min\{k \in \{1, 2, \dots, N\} \mid P_k > 0\}$
$L \leftarrow 0$, $R \leftarrow N$
while $L < R - 1$ **do**
  $k \leftarrow \lfloor (L + R)/2 \rfloor$
  if $u > P_k$ **then**
    $L \leftarrow k$
  else
    $R \leftarrow k$
return R

The algorithm is given a value $u \in [0, 1]$ and a CDF $P_k = F(k)$ for $k = 0, \dots, N$ with $P_0 = 0$. It returns $F^{-1}(u)$ as defined by (4.5). The check on $u = 0$ is necessary if $P_1 = 0$.

Algorithm 4.4 Guide table inversion of a CDF on $\{1, \dots, N\}$.

Guidedinvert$(u, N, P_{0:N}, M, G_{0:M})$
$k \leftarrow G_{\lfloor uM \rfloor}$
while $P_k < u$ **do**
  $k \leftarrow k + 1$
return k

The algorithm is given a value $u \in [0, 1]$, a CDF $P_k = F(k)$ for $k = 0, \dots, N$ with $P_0 = 0$, and a guide table $G_m = F^{-1}(m/M)$ for $m = 0, \dots, M$. It returns $F^{-1}(u)$ as defined by (4.5).

We can compute $F^{-1}(u)$ by constructing the table $\{(k, P_k) \mid 0 \leqslant k \leqslant N\}$ and then searching for the solution k of (4.5).

A sequential search through the table starting at $k = 1$ takes $\mathbb{E}(X)$ comparisons on average and this could grow proportionally to N. A bisection search takes $O(\log(N))$ time to invert F. Algorithm 4.3 is a bisection search specifically for inverting a discrete CDF. See Exercise 4.7 for more.

There are many more sophisticated ways to invert a discrete distribution. They typically require some precomputation and bookkeeping. One notable example is the guide table method of Chen and Asau (1974). The first step is to precompute a table $G_m = F^{-1}(m/M)$ for $m = 0, \dots, M - 1$. The end points are $G_0 = 1$ and $G_M = N$. The guide table G can be computed at the same time that the P_k are computed. Then one does a sequential scan starting at $k = G_{\lfloor Mu \rfloor}$ as in Algorithm 4.4. Fishman and Moore (1984) show that the expected number of comparisons made is at most 2 for the common special case where we choose M equal to N. Devroye (1986, Chapter 2.4) shows that the expected number of comparisons is at most $1 + 1/\alpha$ when $M = \alpha N$.

When the same discrete distribution is being used a large number n of times,

then guide tables are very effective. If instead, each new discrete random variable must be sampled using different p_k, usually because p_k are themselves randomly generated, then the setup costs dominate and guide tables don't help.

When the discrete random variable takes infinitely many values, then of course tabulation will not work. When there are infinitely many values, their description can only be a mathematical one and that may even mean a closed form expression. Sometimes this allows us to invert a continuous CDF and truncate the result to an integer, as for example with the geometric distribution.

Example 4.12 (Geometric distribution)**.** The geometric distribution with parameter $\theta \in (0, 1]$ has probability mass function

$$\mathbb{P}(X = k) \equiv p_k = \theta(1-\theta)^k, \quad k = 0, 1, \ldots.$$

It arises as the number X of failures before the first success in independent trials with success probability θ. It can be sampled by generating and truncating an exponential random variable: $X = \lfloor \log(U)/\log(1-\theta) \rfloor$ where $U \sim \mathbf{U}(0,1)$. Sometimes the geometric distribution is defined as the number of trials until the first success, including that success. Then it is $1 + X$ for the X given above and $\lfloor \cdot \rfloor$ can be replaced by $\lceil \cdot \rceil$.

For other discrete distributions with infinite tails, we might be able to use a numerical search without tabulation, if F is available in closed form. But sometimes numerical inversion is simply so expensive for discrete distributions that the cost erases the benefits of inversion.

4.5 Numerical inversion

In some problems we have no direct F^{-1} available but we might be able to invert F numerically. In the easiest case $F(x)$ is available in closed form. When F is continuous we can search numerically for the value x with $F(x) = u$. When F is not available in closed form but the density $f(x) = F'(x)$ is, then we can build up an approximation to F by numerical integration and then invert it. Sometimes, even f is not available, but the characteristic function $\phi(t) = \int_{-\infty}^{\infty} e^{itx} f(x)\,\mathrm{d}x$ is. Then we may be able to first get

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{itx} \overline{\phi(t)}\,\mathrm{d}t$$

(the bar denotes complex conjugate) before integrating to F and inverting to F^{-1}.

Clearly, the complexity of the numerical chore varies greatly depending on what we have to work with. There is also a natural progression from simple methods to more elaborate ideas that are faster or more robust but require much more bookkeeping. Here we look at a few ideas near the simpler end of the spectrum.

When we already have an algorithm to compute F, we can simply use bisection to search for x such that $F(x) = u$. Bisection is easy to code, but slow: each time it cuts the search interval in half which amounts to only one more correct bit for the estimate. By contrast, Newton's method is extremely fast, when started close to the solution. It requires computation of the PDF $F'(x) = f(x)$. If x_n is close to x, then by Taylor expansion $F(x) \doteq F(x_n) + f(x_n)(x - x_n)$. Equating the approximation to u, solving for x, and taking x_{n+1} to be the result, leads to the Newton step

$$x_{n+1} = x_n - \frac{F(x_n) - u}{f(x_n)}. \tag{4.6}$$

When $f'(\cdot)$ is continuous and positive at the solution x, and the search starts out close enough to x, then the error in Newton's iteration satisfies $|x_{n+1} - x| = O(|x_n - x|^2)$. The number of correct bits in the estimate essentially doubles at each iteration.

Newton's method is not very robust. When f is close to zero, a large step can occur and the method can fail to converge. Roughly: there is no benefit from doubling the number of correct bits, when that number is zero.

There are search algorithms that combine the reliability of bisection while converging to the root at a better rate. Such methods include Brent's method and modified regula falsi (the Illinois method). Several of them are described in Press et al. (2007, Chapter 9).

4.6 Other transformations

Inversion is a conceptually simple method for transforming a single $\mathbf{U}(0,1)$ random variable U into a single random variable X having distribution F. While conceptually simple, it can be numerically awkward. Sometimes we get a better result from more general transformations, usually consuming multiple $\mathbf{U}(0,1)$ variables. These other transformations typically depend on specialized relationships among distributions. When available they provide a particularly elegant solution to a specific problem. They may also be faster than inversion.

Here we consider some of the most important special cases. In a few cases, we give the derivations as examples, but in others we just give the result.

Monotone transformations

Given a distribution F with mean 0 and standard deviation 1 we may shift it to have mean μ and scale it to have standard deviation $\sigma > 0$. If $X \sim F$ then $Y = \mu + \sigma X$ has the desired mean and standard deviation. When X has a probability density function, then so does Y and

$$f_Y(y) = \frac{1}{\sigma} f_X\Big(\frac{y - \mu}{\sigma}\Big). \tag{4.7}$$

More generally, suppose $X \sim f_X$ on $\mathbb{R}$ and $Y = \tau(X)$ for some invertible increasing function τ. Then $\mathbb{P}(Y \leqslant y) = \mathbb{P}(\tau(X) \leqslant y) = \mathbb{P}(X \leqslant \tau^{-1}(y))$ and so Y has density function

$$f_Y(y) = \frac{d}{dy}\mathbb{P}(X \leqslant \tau^{-1}(y)) = f_X(\tau^{-1}(y))\frac{d}{dy}\tau^{-1}(y). \tag{4.8}$$

For example, let $\tau(x) = x^p$, for $p > 0$, and suppose that $X \sim \mathbf{U}(0,1)$. Then $Y = \tau(X) = X^p$ has density function

$$f_Y(y) = \frac{1}{p}y^{(1/p)-1}, \quad 0 < y < 1,$$

and taking $p = 1/2$ yields a triangular density function $f_Y(y) = 2y\mathbb{1}_{0<y<1}$. We will look at another way to sample the triangular density using order statistics, on page 16.

The log-normal distribution is defined by a monotone transformation. Let $X \sim \mathcal{N}(\mu, \sigma^2)$. Then $Y = \exp(X)$ has the log-normal distribution with parameters μ and σ. Its probability density function is

$$f_Y(y) = \frac{1}{y\sqrt{2\pi}\sigma}\exp\left(\frac{-(\log(y)-\mu)^2}{2\sigma^2}\right)$$

on $0 < y < \infty$.

An increasing transformation τ from $X \sim F_X$ (where F_X is continuous) to $Y = \tau(X) \sim F_Y$ implements the QQ transformation $F_Y^{-1}(F_X(\cdot))$.

Box-Muller

One of the best-known transformations is the Box-Muller method for generating two normally distributed random variables from two uniformly distributed ones. The Box-Muller prescription is to generate two independent $\mathcal{N}(0,1)$ variables, Z_1 and Z_2 via

$$\begin{aligned} Z_1 &= \sqrt{-2\log U_1}\cos(2\pi U_2) \\ Z_2 &= \sqrt{-2\log U_1}\sin(2\pi U_2) \end{aligned} \tag{4.9}$$

where $U_1, U_2 \sim \mathbf{U}(0,1)$ and are independent.

A short explanation is available, using properties of the normal distribution. Box-Muller works because points from the $\mathcal{N}(0, I_2)$ distribution, when written in polar coordinates, have an angle $\theta \sim \mathbf{U}[0, 2\pi)$ that is independent of the radius R. Taking $\theta = 2\pi U_2$ generates the angle. As for the radius, $R^2 = Z_1^2 + Z_2^2 \sim \chi^2_{(2)}$, which is the same as $2 \times \mathrm{Exp}(1)$, the exponential distribution with mean 2 (rate 1/2). Thus we may take $R = \sqrt{-2\log(U_1)}$ using (complementary) inversion of the exponential distribution.

Without knowing these facts about the normal distribution, we could prove the Box-Muller formula via the change of variable formula for probability density functions. The details are on page 35 of the chapter end notes.

The Box-Muller method is not the fastest way to generate $\mathcal{N}(0,1)$ random variables, and numerical computing environments don't always use it. There is some cost in computing cos, sin, log and sqrt that, with clever programming can be avoided. Box-Muller remains very popular because it is simple to use. The expression (4.9) for Z_1 is literally a one-liner, and so it is easy to debug and even very rudimentary computing environments have all the functions it needs. These advantages often outweigh the small speed disadvantage. We don't really need to use Z_2, though of course with a little extra programming complexity we can deliver Z_1 and save Z_2 for the next function call, if it comes, or arrange to generate all our $\mathcal{N}(0,1)$ random variables in pairs.

Box-Muller is also exact, if we assume that sin, cos, log and the square root are implemented in exact arithmetic. In floating point computations, a well implemented Φ^{-1} need not be less accurate than sin or cos or the others, so exactness of Box-Muller is not a compelling advantage.

While Box-Muller is simple to use for plain independent $\mathcal{N}(0,1)$ random variables, it can be awkward to incorporate into variance reduction methods. There is a lot to be said for inversion, $Z = \Phi^{-1}(U)$, which is also a one-liner when we have an implementation of Φ^{-1} at hand. With inversion, the number of uniform random variables consumed equals the number of $\mathcal{N}(0,1)$ variables produced and this helps to synchronize some simulations.

Maxima, minima and order statistics

Suppose that $Y = \max(X_1, \dots, X_r)$ where X_i are independent with CDF F. Then

$$\mathbb{P}(Y \leqslant y) = \mathbb{P}\Big(\max_{1\leqslant i\leqslant r} X_i \leqslant y\Big) = \mathbb{P}(X_1 \leqslant y)^r = F(y)^r.$$

To sample from the CDF, $G = F^r$, we may take the maximum of r independent samples from F. As a case in point, suppose that F is the uniform distribution on $(0,1)$ and that $r = 2$. That is $Y = \max(U_1, U_2)$ where $U_k \sim \mathbf{U}(0,1)$. Then $G(y) = y^2$, so Y has the triangular density $g(y) = 2y$ on $0 \leqslant y \leqslant 1$. In Example 4.11 we saw that the triangular distribution is easy to sample by inversion: $Y = \sqrt{U_1}$. Taking $\max(U_1, U_2)$ could be much faster than $\sqrt{U_1}$.

Similarly, if $Y = \min(X_1, \dots, X_r)$ for independent $X_k \sim F$, then the CDF G of Y satisfies $G(y) = 1 - (1 - F(y))^r$. As an example, $Y = \min(U_1, U_2)$ has the triangular distribution with density $g(y) = 2(1-y)$ for $0 \leqslant y \leqslant 1$, for independent $U_i \sim \mathbf{U}(0,1)$.

The minimum and maximum are special cases of order statistics. Let $X_1, \dots, X_n$ be independent identically distributed random variables. The ***order statistics*** of $X_1, \dots, X_n$ are the same n values arranged in increasing order and denoted by $X_{(1)} \leqslant X_{(2)} \leqslant \cdots \leqslant X_{(n)}$.

Order statistics are particularly simple for $\mathbf{U}(0,1)$ random variables. Suppose that $U_{(r)}$ is the r'th order statistic among n independent $\mathbf{U}(0,1)$ random variables. If $x \leqslant U_{(r)} < x + \mathrm{d}x$ then the three intervals $(0, x)$, $[x, x + \mathrm{d}x)$, and $(x + \mathrm{d}x, 1)$ contain $r - 1$, 1, and $n - r$ of the U_i respectively, since for small $\mathrm{d}x$

the probability of two or more U_i landing in $[x, x+\mathrm{d}x)$ is negligible. Thus

$$\begin{aligned}\mathbb{P}(x \leqslant U_{(r)} \leqslant x+\mathrm{d}x) &= \frac{n!}{(r-1)!1!(n-r)!}x^{r-1}(1-x)^{n-r}\mathrm{d}x \\ &= \frac{x^{r-1}(1-x)^{n-r}}{(n-r)!(r-1)!/n!}\mathrm{d}x \end{aligned} \tag{4.10}$$

which is the Beta distribution (Example 4.29 of §4.8) with parameters $\alpha = r$ and $\beta = n - r + 1$.

Equation (4.10) may be used in two ways. First it shows that for positive integers α and β we can sample from the Beta(α, β) distribution by generating $\alpha + \beta - 1$ independent $\mathbf{U}(0,1)$ random variables and selecting $U_{(\alpha)}$. If $\alpha + \beta$ is small this method is not too expensive.

The second way to use (4.10) is in reverse. If we want to sample the value of $Y_{(r)}$ where Y_i are IID with distribution function F, then we take $Y_{(r)} = F^{-1}(X)$ where $X \sim \text{Beta}(n, n-r+1)$. For large n this recipe can save considerable computation, but only if we are able to use inversion.

Example 4.13 (k out of n systems)**.** A k out of n system is made up of n identical parts. Each part is either functional or not. The system is functional if at least k of its parts are functional. If the system starts at time 0 with all n parts operating, and part i fails at time $Y_i > 0$ then the system fails at time $Y_{(n-k+1)}$. If $Y_i \sim F$ independently, then the system fails at time $F^{-1}(X)$ where $X \sim \text{Beta}(n-k+1, k)$. Exercise 4.19 presents an application to fault-tolerant memory systems and Exercise 4.20 considers two-level redundancy.

Sums

A good many of the rules in elementary probability theory give us the distribution for a sum $Y = X_1 + \cdots + X_n$ of IID random variables X_i in terms of the distribution of individual terms X_i. Often a simple way to sample the distribution of X is to generate and sum n independent X_i, even when Y did not arise as an explicit sum of real world quantities. Of course this approach slows down considerably when n is large.

Example 4.14 (Binomial distribution)**.** The random variable Y has the Binomial distribution with parameters $n \in \{1, 2, \dots\}$ and $p \in (0,1)$ if

$$\mathbb{P}(Y = y) = \binom{n}{x} p^y (1-p)^{n-y}, \quad y = 0, 1, \dots, n.$$

We write $Y \sim \text{Bin}(n, p)$. This distribution describes the number of successes in n independent trials each with probability p of success. It follows that we may write

$$Y = \sum_{i=1}^{n} X_i \tag{4.11}$$

where $X_i \sim \text{Bern}(p)$ are independent Bernoulli random variables (Example 4.2).

Example 4.15 ($\chi^2_{(n)}$ distribution)**.** Suppose that $X_1, \ldots, X_n$ are independent $\chi^2_{(\alpha)}$ random variables. Then

$$Y = \sum_{i=1}^{n} X_i \sim \chi^2_{(n\alpha)}.$$

For $\alpha = 1$ we may take $X_i = Z_i^2$ where $Z_i \sim \mathcal{N}(0,1)$. For $\alpha = 2$ we may take $X_i \sim \mathrm{Exp}(1/2)$, the exponential distribution with mean 2. The chi-squared distribution is a special case of the gamma distribution (§4.8) which also has a summation rule. Example 4.28 generalizes $\chi^2_{(\nu)}$ to arbitrary $\nu > 0$.

Example 4.16 (Noncentral distributions)**.** The noncentral χ^2 distribution with n degrees of freedom and noncentrality parameter $\lambda \geqslant 0$, denoted $\chi'^2_{(n)}(\lambda)$, is the distribution of $Y = \sum_{i=1}^n X_i^2$ where $X_i \sim \mathcal{N}(a_i, 1)$ and $\lambda = \sqrt{\sum_{i=1}^n a_i^2}$. It can be simulated directly from this recipe. When d is large, we may take advantage of the fact that this distribution depends on the a_i only through their sum of squares. As a result we may simulate $Y = X + (Z - \sqrt{\lambda})^2$ where $X \sim \chi^2_{(n-1)}$ independently of $Z \sim \mathcal{N}(0,1)$.

The noncentral F distribution with degrees of freedom n and d and noncentrality λ, is the distribution of $(Y_1/n)/(Y_2/d)$ where $Y_1 \sim \chi'^2_{(n)}(\lambda)$ independently of $Y_2 \sim \chi^2_{(d)}$. This distribution, denoted $F'_{n,d}(\lambda)$ is used in computations of statistical power. It is also used in sampling the Cox-Ingersoll-Ross model for interest rates, presented in §6.5. We consider a mixture sampling approach to the noncentral chi-squared in §4.9, which extends to non-integer n and even allows for $n = 0$.

One-liners

Luc Devroye has compiled a collection of one-line algorithms for generating random variables: see Devroye (1996). One-liners are easy to code and debug, but are only available for special distributions or sometimes for special parameter values of those distributions. But when they're available, they're really useful.

A ***one-liner*** is an expression involving some finite number of $\mathbf{U}(0,1)$ random variables U_i, the usual arithmetic operations ($+$, $-$, $\div$ $\times$), and some set of familiar functions log, exp, sin, cos, $\lfloor\cdot\rfloor$, inverse trigonometric functions and the like. The precise definition depends on the menu of familiar functions that are allowed.

A rule that reuses the same value of a random variable (e.g., $U_1 + U_1U_2$) would require storing and retrieving that value. Such rules are called ***extended one-liners***.

Many of the one-liners are familiar inversions and for a few the distribution itself was defined by its one-liner. For instance Tukey's λ distribution is defined by $X = (U^\lambda - (1-U)^\lambda)/\lambda$ for a parameter $\lambda \neq 0$, where $U \sim \mathbf{U}(0,1)$. Here are some one-liners for the t and Beta families.

Example 4.17 (Symmetric beta)**.** The Beta(α, α) distribution for $\alpha > 1/2$ can be sampled via

$$X = \frac{1}{2}\left(1 + \sqrt{1 - U_1^{\frac{2}{2\alpha-1}}}\cos(2\pi U_2)\right).$$

Example 4.18 (Symmetric beta again)**.** Another one-liner for Beta(α, α) is,

$$X = \frac{1}{2} + \frac{\mathbb{1}_{U_3 \leqslant 1/2}}{2\sqrt{1 + \dfrac{1}{(U_1^{-1/\alpha} - 1)\cos^2(2\pi U_2)}}}.$$

This one works for all $\alpha > 0$.

Example 4.19 (Half beta)**.** The Beta$(1/2, \beta)$ distribution for $\beta > 1/2$ can be sampled via

$$X = \left(1 - U_1^{\frac{2}{2\beta-1}}\right)\cos^2(2\pi U_2).$$

Example 4.20 (t distribution)**.** The $t_{(n)}$ distribution for $n > 0$ can be sampled via

$$X = \frac{\sqrt{n}(B - 1/2)}{2\sqrt{B(1-B)}}$$

where $B \sim \text{Beta}(n/2, n/2)$. If B is sampled by a one-liner, then this formula becomes an extended one-liner because B is reused.

Example 4.21 (t distribution again)**.** The $t_{(n)}$ distribution for $n > 0$ can be sampled via

$$X = \sqrt{n\left(U_1^{-2/n} - 1\right)}\cos(2\pi U_2).$$

Example 4.22 (Symmetric beta, yet again)**.** The Beta(α, α) distribution for $\alpha > 0$ can be sampled via

$$X = \frac{1}{2}\Big(1 + \frac{T_{2\alpha}}{\sqrt{2\alpha + T_{2\alpha}^2}}\Big)$$

where $T_{2\alpha}$ is a $t_{(2\alpha)}$ random variable.

Example 4.23 (Symmetric stable law distribution)**.** A stable law distribution F is closed under linear combinations. This means that if X_1, X_2 are independent from F and $a_1, a_2 > 0$, then $a_1X_1 + a_2X_2$ has the same distribution as $bX + c$ for some $b > 0$, $c \in \mathbb{R}$ and $X \sim F$. As a result the average $(X_1 + \cdots + X_n)/n$ of stable law random variables has the same distributional shape as the individual members. There is a four parameter family of stable laws. Only a few special cases have closed forms for their probability density function. An important three parameter subfamily of stable laws are those that are symmetric about their median. If X has a symmetric stable law distribution then X has representation $\mu + \sigma Z$ for $\mu \in \mathbb{R}$, $\sigma > 0$ and a random variable Z with characteristic

function $\mathbb{E}(\exp(\sqrt{-1}tZ)) = \exp(-|t|^\alpha)$ where $0 < \alpha \leqslant 2$. Chambers et al. (1976) provide an extended one-liner for the stable law. In the symmetric case it reduces to

$$Z = \begin{cases} \dfrac{\sin(\alpha V)}{\cos(V)^{1/\alpha}} \left(\dfrac{\cos((\alpha-1)V)}{W} \right)^{(1-\alpha)/\alpha}, & \alpha \neq 1 \\ \tan(V), & \alpha = 1, \end{cases}$$

where $V \sim \mathbf{U}(-\pi/2, \pi/2)$ independently of $W \sim \mathrm{Exp}(1)$. For more about stable laws, see Nolan (2013).

Discrete transformations

The transformation method can also be applied to discrete random variables. If $\mathbb{P}(Z = k) = q_k$ and $X = \tau(Z)$ is a transformation of X then $p_k = \mathbb{P}(X = k) = \sum_j q_j \mathbb{1}_{\tau(j)=k}$.

One particularly simple and useful method is based on the empirical distribution of data. We may not know the real distribution of some quantity but we might have some historical data in the form of sample values $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M$. We can sample one of these values by taking $\boldsymbol{X} = \boldsymbol{x}_I$ where $I \sim \mathbf{U}\{1, \ldots, M\}$. Resampling data is widely used in operations research simulations where we can re-play historical values for prices, demands or travel times. The distribution being sampled is the ***empirical distribution*** $\widehat{F}_M = \frac{1}{M}\sum_{i=1}^M \delta_{\boldsymbol{x}_i}$ where $\delta_{\boldsymbol{x}}$ is a distribution taking the value $\boldsymbol{x}$ with probability 1.

Example 4.24 (Bootstrap)**.** Bootstrap inferences are based on resampling a data set some large number, B, of times. Suppose that we have n observations $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n \overset{\text{iid}}{\sim} F$ and our data analysis includes the statistic $\widehat{T}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n) \in \mathbb{R}$ computed from those values. When $\widehat{T}$ is too complicated to analyze or when F has an unknown form, it is difficult to describe the variance of $\widehat{T}$. In a bootstrap analysis, we generate observations $\boldsymbol{X}_i^{*b} \overset{\text{iid}}{\sim} \widehat{F}_n$, for $i = 1, \ldots, n$ and $b = 1, \ldots, B$, where $\widehat{F}_n$ is the empirical distribution of $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$. We now have B resampled copies of our dataset and we compute B resampled versions of our statistic, $\widehat{T}^{*b} = T(\boldsymbol{X}_1^{*b}, \ldots, \boldsymbol{X}_n^{*b})$, for $b = 1, \ldots, B$.

It is a remarkable fact that the sampling fluctuations in $\widehat{T}^{*b}$ often provide a reasonable gauge of the sampling fluctuations of our single computed value $\widehat{T}$. The bootstrap estimate of $\mathrm{Var}(\widehat{T})$ is $(1/(B-1))\sum_{b=1}^B(\widehat{T}^{*b} - \bar{T})^2$, where $\bar{T} = (1/B)\sum_{b=1}^B \widehat{T}^{*b}$. The percentile method 95% confidence interval is from the 2.5'th to 97.5'th percentile of the resampled $\widehat{T}^{*b}$ values.

To obtain the bootstrap samples we draw $I(i,b) \sim \mathbf{U}\{1, \ldots, n\}$. Then $\boldsymbol{X}_i^{*b} = \boldsymbol{X}_{I(i,b)}$.

Discrete transformations based on lookup tables are particularly useful. Suppose that $p_k = n_k/N$ where $n_k \geqslant 0$ are integers summing to N. When we have adequate storage at our disposal then we may make an array A of length N containing n_k copies of k. Now we sample $Z \sim \mathbf{U}\{1, \ldots, N\}$ and deliver $X = A_Z$.

This ***array sampling*** method is a QQ transformation of Z when A is sorted, but in applications where p is changing through the simulation, A need not be sorted. We will use this method to simulate preferential attachment random graphs in §5.12.

Gumbel trick

Guide tables are cumbersome to set up and we might only have to draw one sample from a given discrete distribution before some other step in our algorithm changes the target distribution. Suppose that we are given $q_k > 0$ for $k = 1, \dots, N$ and we want to sample a random index k with probability $p_k = q_k / \sum_{j=1}^N q_j$. A very simple way to do this is to generate $E_j \overset{\text{iid}}{\sim} \text{Exp}(1)$ and then pick

$$k = \operatorname*{argmin}_j E_j / q_j.$$

It is convenient that we do not even need to form the normalized probabilities p_k. See Exercise 4.9.

The values of q_k may range over many orders of magnitude and then we might prefer to work with $t_k = \log(q_k)$. For that we simply return as k, the j that minimizes $\log(E_j) - t_j$. Equivalently we can find the j that maximizes

$$t_j + G_j, \quad \text{G}_j = -\log(E_j), \quad E_j \overset{\text{iid}}{\sim} \text{Exp}(1).$$

The random variables G_j have the standard ***Gumbel distribution***, and so this algorithm is sometimes called the Gumbel trick. The Gumbel distribution has CDF $\exp(-\exp(-x))$ for $x \in \mathbb{R}$.

4.7 Acceptance-Rejection

Sometimes we cannot find a computable transformation that takes a fixed number of $\mathbf{U}(0,1)$ random variables and produces a random variable X with the distribution F that we want. In these cases, we can often sample from another distribution G, and then by strategically rejecting some values from G while accepting the others, induce a sampling bias that works in our favor to produce F distributed values. The method is variously called rejection sampling, accept-reject and ***acceptance-rejection*** sampling.

We present the idea below, assuming that F and G are continuous distributions with probability density functions f and g respectively. But the method extends naturally to discrete distributions.

To use acceptance-rejection, we find a density $g(x)$ and a finite constant c satisfying three conditions: we can draw samples from g, we can compute the function f/g, and

$$f(x) \leqslant cg(x) \tag{4.12}$$

always holds. We repeatedly sample candidates $Y \sim g$. Given that $Y = y$, the candidate y is accepted with probability $A(y) = f(y)/(cg(y))$. Otherwise it is rejected and we try the next candidate. The first accepted value is delivered as the observed value x of X. There will never be proposals with $g(y) = 0$ but for completeness we specify $A(y) = 0$ when $g(y) = 0$. Then $g(y)A(y) = f(y)/c$ holds for all y.

Algorithm 4.5 contains pseudo-code for acceptance-rejection sampling. We will extend it in Theorem 4.5 below for densities that are not normalized. The condition (4.12) ensures that $A(y) \leqslant 1$. Notice that if $f(x)/g(x)$ is unbounded, then we cannot use g.

The intuition behind acceptance-rejection sampling is as follows. If candidates have density $g(y)$ and they are accepted with probability $A(y)$, then the accepted ones should have density proportional to $A(y)g(y)$. Choosing $A \propto f/g$ gives us $Ag \propto f$. The proof is in Theorem 4.2 below. We will use similar ideas in §11.4 to derive the Metropolis-Hastings acceptance rule for Markov chain Monte Carlo.

Theorem 4.2. *Let $f(x)$ and $g(x)$ be probability density functions with $f(x) \leqslant cg(x)$ for all $x \in \mathbb{R}$. Then X generated by Algorithm 4.5 has probability density function f.*

Proof. Let $Y \sim g$ and $U \sim \mathbf{U}(0,1)$ be independent. Given $Y = y$, the proposal is accepted if $U \leqslant A(y)$ where $A(y)$ satisfies $g(y)A(y) = f(y)/c$. The probability that Y is accepted is

$$\int_{-\infty}^{\infty} g(y)A(y)\,\mathrm{d}y = \int_{-\infty}^{\infty} f(y)/c\,\mathrm{d}y = \frac{1}{c}.$$

Now for any $x \in \mathbb{R}$,

$$\begin{aligned}\mathbb{P}(X \leqslant x) &= \int_{(-\infty,x]} g(y)A(y)\,\mathrm{d}y + \Bigl(1 - \frac{1}{c}\Bigr)\mathbb{P}(X \leqslant x)\\ &= \frac{1}{c}\int_{(-\infty,x]} f(y)\,\mathrm{d}y + \Bigl(1 - \frac{1}{c}\Bigr)\mathbb{P}(X \leqslant x),\end{aligned}$$

where the first term comes from acceptance of Y and the second (recursive) term comes from rejection of Y. Therefore $\mathbb{P}(X \leqslant x) = \int_{(-\infty,x]} f(y)\,\mathrm{d}y$ and so $X \sim f$. □

The proof of Theorem 4.2 uses $\int_{(-\infty,x]} f(y)\,\mathrm{d}y$ for $\int_{-\infty}^{x} f(y)\,\mathrm{d}y$. The argument generalizes to the d dimensional case that we study in Chapter 5 by replacing $(-\infty, x]$ with a rectangular region of the form $\prod_{j=1}^{d}(-\infty, x_j] \subset \mathbb{R}^d$.

From the proof of Theorem 4.2 we see that the probability of acceptance is $1/c$. Therefore the number N of proposals that we must make before one is accepted has a geometric distribution with $\mathbb{P}(N = k) = c^{-1}(1 - c^{-1})^{k-1}$. As a result $\mathbb{E}(N) = c$, which is intuitively obvious: if in the long run we accept $1/c$ of the candidates, then we must generate c times as many candidates as accepted variables.

Algorithm 4.5 Acceptance-rejection sampling.

given c with $f(x) \leqslant cg(x)$, $\forall x \in \mathbb{R}$
repeat
  $Y \sim g$
  $U \sim \mathbf{U}(0,1)$
until $U \leqslant f(Y)/(cg(Y))$
$X \leftarrow Y$
deliver X

Smaller values of c correspond to more efficient sampling schemes. The smallest allowable value is $c = \sup_x f(x)/g(x)$. The supremum is over x with $f(x) > 0$, that is we can work as if $f(x)/g(x) = 0$ when $f(x) = g(x) = 0$. We always have $c \geqslant 1$ because $1 = \int f(x)\,\mathrm{d}x \leqslant \int cg(x)\,\mathrm{d}x = c$.

Example 4.25. Let $f(x) = \varphi(x)$ be the $\mathcal{N}(0,1)$ density and $g(x) = \pi^{-1}(1 + x^2)^{-1}$ be the standard Cauchy distribution. We can sample $Y \sim g$ and accept $Y = y$ with probability $f(y)/(cg(y))$ where

$$c = \sup_x \sqrt{\frac{\pi}{2}}(1 + x^2)\exp(-x^2/2).$$

The supremum is attained at $x = \pm 1$ (Exercise 4.10) and has value $c \doteq 1.52$. As result, the acceptance probability is $1/c \doteq 0.658$. We cannot use acceptance-rejection with proposals from $\mathcal{N}(0,1)$ to sample from the Cauchy distribution because $g(x)/f(x)$ is unbounded.

There is a geometric interpretation to acceptance-rejection sampling. If we sample a point (X, Z) uniformly in the portion of the plane between the horizontal axis and the curve $f(x)$, then the marginal distribution of X is f. To get such a point (X, Z) we can sample uniformly from the region under the curve $cg(x)$ which lies above $f(x)$ and keep only the points below f. Figure 4.2 illustrates this geometry.

To establish the geometric interpretation, we introduce sets of the form

$$\mathcal{S}_M(h) = \{(x, z) \mid 0 \leqslant z \leqslant Mh(x),\ x \in \mathbb{R}\} \subset \mathbb{R}^2,$$

where h is a probability density function on $\mathbb{R}$, such as f or g and $M > 0$ could be either 1 or c as needed.

Theorem 4.3. *If $(X, Z) \sim \mathbf{U}(\mathcal{S}_M(h))$ for $M > 0$ and a probability density function h on $\mathbb{R}$, then $X \sim h$.*

Proof. Write $\mathcal{S} = \mathcal{S}_M(h)$, suppose that $(X, Z) \sim \mathbf{U}(\mathcal{S})$ and pick $x \in \mathbb{R}$. Then

$$\mathbb{P}(X \leqslant x) = \frac{\mathbf{vol}(\mathcal{S} \cap (-\infty, x] \times [0, \infty))}{\mathbf{vol}(\mathcal{S})} = \frac{\int_{(-\infty,x]} \int_0^{Mh(y)} 1\,\mathrm{d}z\,\mathrm{d}y}{\int_{\mathbb{R}} \int_0^{Mh(y)} 1\,\mathrm{d}z\,\mathrm{d}y}.$$

Now $\int_0^{Mh(y)} 1\,\mathrm{d}z = Mh(y)$ and so $\mathbb{P}(X \leqslant x) = \int_{(-\infty,x]} h(y)\,\mathrm{d}y$ as required. □

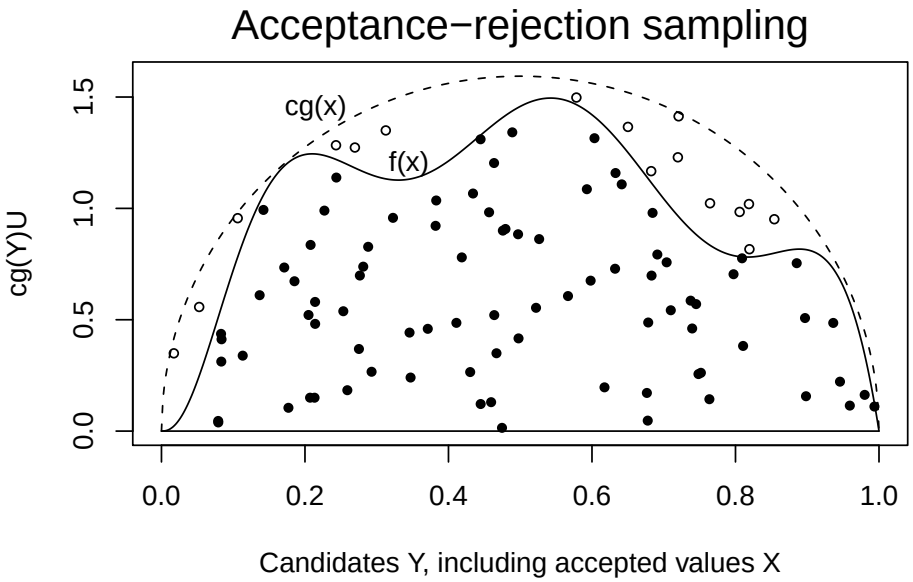

Figure 4.2: This figure illustrates acceptance-rejection sampling. The target density $f(x)$ is shown as a solid line. The proposal density is g and $cg(x)$ is shown as a dashed line. The open circles are rejected points. The solid circles are the accepted points. We sample uniformly under $cg(x)$, keeping only the points that are under $f(x)$, whose horizontal components then have density f.

Theorem 4.4. *Suppose that $X \sim h$ and that $Z = Mh(X)U$ where $U \sim \mathbf{U}(0,1)$ is independent of X and $M > 0$. Then $(X, Z) \sim \mathbf{U}(\mathcal{S}_M(h))$.*

Proof. This follows from Devroye (1986, Theorem 3.1). We sketch it here under the further assumption that h is Riemann integrable.

Let $R \equiv [x_0, x_1] \times [z_0, z_1]$ be an arbitrary rectangle. We need to show that $\mathbb{P}((X, Z) \in R) = \mathbf{vol}(R \cap \mathcal{S}_M(h))/M$. First assume that R is strictly inside $\mathcal{S} = \mathcal{S}_M(h)$. Then

$$\begin{aligned}
\mathbb{P}((X,Z) \in R) &= \int_{x_0}^{x_1} h(x)\mathbb{P}(z_0 \leqslant Z \leqslant z_1 \mid X = x)\,\mathrm{d}x \\
&= \int_{x_0}^{x_1} h(x)\mathbb{P}\Big(\frac{z_0}{Mh(x)} \leqslant U \leqslant \frac{z_1}{Mh(x)} \mid X = x\Big)\,\mathrm{d}x \\
&= \frac{1}{M}(x_1 - x_0)(z_1 - z_0) = \frac{\mathbf{vol}(R)}{M}.
\end{aligned}$$

Similarly, if $R \cap \mathcal{S}_M(h) = \varnothing$ then $\mathbb{P}((X, Z) \in R) = 0$. A general rectangle may intersect $\mathcal{S}_M(h)$ without being contained in it. We can write that rectangle as the union of n^2 congruent subrectangles with height and width $1/n$ times that of the original. The probability content of the rectangle is between $1/M$ times

the volume of subrectangles contained within R and $1/M$ times the volume of subrectangles that intersect R. Those are Riemman sums that converge to $\mathbf{vol}(R \cap \mathcal{S}_M(h))$ for Riemann integrable h. □

Algorithm 4.5 uses properly normalized density functions. In some applications we know f or g or both only up to a constant of proportionality. For instance f might be a well-known density restricted to a set A whose probability is unknown. The geometric viewpoint shows us that acceptance-rejection sampling can be used when one or both of f, g is unnormalized. As the following theorem shows, we can generate points uniformly under a curve proportional to $g(y)$, accepting only those points under a curve proportional to $f(y)$, and thereby sample from f.

Theorem 4.5. *Let $\widetilde{f}(x) \geqslant 0$ with $0 < \int_{-\infty}^{\infty} \widetilde{f}(x)\,\mathrm{d}x < \infty$ and $\widetilde{g}(x) \geqslant 0$ with $0 < \int_{-\infty}^{\infty} \widetilde{g}(x)\,\mathrm{d}x < \infty$. Suppose that $\widetilde{f}(x) \leqslant c\widetilde{g}(x)$ holds for all x and some $c < \infty$. Let candidates Y be sampled from the density proportional to $\widetilde{g}$. Let a candidate $Y = y$ be accepted with probability $\widetilde{f}(y)/(c\widetilde{g}(y))$. Then the accepted random variables have probability density proportional to $\widetilde{f}$.*

Proof. Let $U \sim \mathbf{U}(0,1)$ independently of $Y \sim g$ where $g(y) = \widetilde{g}(y)/c_g$ for $c_g = \int_{-\infty}^{\infty} \widetilde{g}(y)\,\mathrm{d}y$. Define $Z = cU\widetilde{g}(Y) = cc_gUg(Y)$. By Theorem 4.4 the candidates satisfy $(Y, Z) \sim \mathbf{U}(\mathcal{S}_{cc_g}(g))$. We may accept the points that satisfy $U \leqslant \widetilde{f}(Y)/(c\widetilde{g}(Y))$ in which case $Z \leqslant \widetilde{f}(Y) = c_f f(Y)$. An accepted point $(X, Z) \sim \mathbf{U}(\mathcal{S}_{cc_g}(g) \cap \mathcal{S}_{c_f}(f))$. Now $cc_g g(y) = c\widetilde{g}(y) \geqslant \widetilde{f}(y) = c_f f(y)$ so $\mathcal{S}_{c_f}(f) \subseteq \mathcal{S}_{cc_g}(g)$. Therefore the accepted points satisfy $(X, Z) \sim \mathbf{U}(\mathcal{S}_{c_f}(f))$ and so their first component has distribution f by Theorem 4.3. □

The acceptance probability for acceptance-rejection based on the unnormalized densities in Theorem 4.5 is

$$\frac{\mathbf{vol}(\mathcal{S}_{c_f}(f))}{\mathbf{vol}(\mathcal{S}_{cc_g}(g))} = \frac{c_f}{cc_g},$$

where c_f and c_g are the integrals of $\widetilde{f}$ and $\widetilde{g}$ respectively.

Example 4.26 (Tail of a normal distribution)**.** Here we sample from the tail of a normal distribution by shifting and scaling exponentially distributed proposals. Let $f(x) \propto \widetilde{f}(x) = \exp(-x^2/2)$ on $[A, \infty)$ where $A \geqslant 1$. Let $g(x) \propto \widetilde{g}(x) = \exp(-A(x-A))$ on $[A, \infty)$. We can sample from g by taking $Y \sim A + \mathrm{Exp}(1)/A$. Using complementary inversion, our proposal is $Y = A - \log(U)/A$. The function $\widetilde{f}(x)/\widetilde{g}(x) = \exp(A(x-A) - x^2/2)$ is decreasing on $[A, \infty)$ and so $\widetilde{f}(x)/\widetilde{g}(x) \leqslant \widetilde{f}(A)/\widetilde{g}(A) = \exp(-A^2/2)$. Therefore we can use $c = \exp(-A^2/2)$. In this case, we know the normalizing constants. We can use them to find that the acceptance probability is

$$\frac{1}{c}\frac{\int_A^{\infty} \exp(-x^2/2)\,\mathrm{d}x}{\int_A^{\infty} \exp(-A(x-A))\,\mathrm{d}x} = A\exp\Big(\frac{A^2}{2}\Big)\sqrt{2\pi}\Phi(-A).$$

Algorithm 4.6 Acceptance-rejection sampling with a squeeze.

given c with $f(x) \leqslant cg(x)$, and $L(\cdot)$ with $L(x) \leqslant f(x)/(cg(x))$, $\forall x \in \mathbb{R}$
repeat
  $Y \sim g$
  $U \sim \mathbf{U}(0,1)$
  if $U \leqslant L(Y)$ **then**
    break
until $U \leqslant f(Y)/(cg(Y))$
$X = Y$
deliver X

For instance, with $A = 2$ (or 5 or 8) the acceptance probability is 0.843 (respectively 0.964 or 0.985).

The running time for acceptance-rejection is proportional to c times the cost of generating and testing one proposal. Sometimes the time spent testing is the largest part, usually because f itself is expensive to compute. If we have an easily computed function $L(y) \leqslant A(y)$ then we can speed up the algorithm by accepting $X = y$ immediately if $u \leqslant L(y)$ without even computing $f(y)$. See Algorithm 4.6. Only on the hopefully rare occasions when $u > L(y)$ do we need to compute $A(y)$ to decide whether to accept or reject the proposal.

This early acceptance technique is called a ***squeeze***. A well chosen squeeze function L lies just barely below A. The probability that we can accept the proposal using L is the fraction of the area below $(x, A(x))$ that is also below $(x, L(x))$.

We can also squeeze from above with a function $H(x) > A(x)$. When $u > H(x)$ we know the point will be rejected without having to carry out the computation of A. For a well-chosen proposal function g there is more to gain by using a lower bound L than an upper bound H. The probability that a proposal lies below f is $1/c$ while a proportion $1 - 1/c$ of proposals lie above f. So when c is near 1 there is less room to improve from using H.

We could do both: after finding $u > L(y)$ we might then test $u > H(y)$ before computing $A(y)$. In principal, we could use a whole cascade of functions

$$L_1(x) \leqslant L_2(x) \leqslant \cdots \leqslant L_s(x) \leqslant A(x) \leqslant H_r(x) \leqslant \cdots \leqslant H_2(x) \leqslant H_1(x).$$

In practice however, one well chosen lower bound is often all that is used.

The computation being avoided is often just a call to a special function like log, exp or cos. These don't seem like onerous computations but they can cost much more than the plain arithmetic used in the rest of the algorithm.

Example 4.27 (Normal polar)**.** Acceptance-rejection sampling can be used to eliminate the need for the sine and cosine functions used in the Box-Muller method. Specifically, sample

$$V_j \sim \mathbf{U}(-1,1), \quad j = 1, 2, \quad \text{then put}$$

$$S = V_1^2 + V_2^2.$$

If $S < 1$, deliver two independent $\mathcal{N}(0,1)$ random variables

$$Z_1 = V_1\sqrt{-2\log(S)/S}, \quad \text{and}$$
$$Z_2 = V_2\sqrt{-2\log(S)/S}.$$

If $S \geqslant 1$, then try again.

The explanation is similar to the one for Box-Muller. The pair (V_1, V_2) is uniformly distributed in the square $(-1,1)^2$. The pairs that are accepted are uniformly distributed in the open unit disk $\{(V_1, V_2) \mid V_1^2 + V_2^2 < 1\}$. The polar coordinates of (V_1, V_2) have angle $\theta \sim \mathbf{U}[0, 2\pi)$ independently of the radius $R = \sqrt{S}$ which has a triangle density $2r$ on $0 < r < 1$. The transformation $\sqrt{-2\log(S)/S}$ is a QQ transformation from the distribution of S to the distribution of the square root of a $\chi^2_{(2)}$ random variable.

4.8 Gamma random variables

The gamma distribution, $\mathrm{Gam}(\alpha, \theta)$, has probability density function

$$f(x;\alpha,\theta) = \frac{x^{\alpha-1}e^{-x/\theta}}{\Gamma(\alpha)\theta^\alpha}, \quad 0 < x < \infty.$$

The parameter $\alpha > 0$ is the shape parameter and $\theta > 0$ is a scale parameter. If $X \sim \mathrm{Gam}(\alpha, \theta)$ then $\mathbb{E}(X) = \alpha\theta$ and $\mathrm{Var}(X) = \alpha\theta^2$.

Gamma distributions are used to model nonnegative quantities that are skewed to the right. They are often a good model for random times, weights, lengths and so on. The gamma distribution includes the χ^2 distributions as a special case, and also provides a route to simulating Beta, t, and F distributions in the examples below. If $X_i \sim \mathrm{Gam}(\alpha_i, \theta)$ independently for $i = 1, \dots, n$, then $S = \sum_{i=1}^n X_i \sim \mathrm{Gam}(\sum_{i=1}^n \alpha_i, \theta)$.

If $Y \sim \mathrm{Gam}(\alpha, 1)$ then $X = \theta Y \sim \mathrm{Gam}(\alpha, \theta)$. Therefore it is enough to sample from $\mathrm{Gam}(\alpha, 1)$, denoted $\mathrm{Gam}(\alpha)$ for short. Sometimes the Gamma distribution is parameterized by its shape and a rate parameter equal to $1/\theta$ which could be confusing. Writing $X \sim \theta\mathrm{Gam}(\alpha)$ lets us eliminate parametrization errors.

The denominator of $f(x;\alpha,\theta)$ contains the Gamma function

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1}e^{-x}\,\mathrm{d}x. \tag{4.13}$$

The Gamma function is not available in closed form, but is widely implemented. For positive integers n, we have $\Gamma(n) = (n-1)!$.

Inversion of the Gamma CDF F_α requires a numerical inverse $\gamma_\alpha^{-1}(u)$ of the (lower) incomplete gamma function

$$\gamma_\alpha(x) = \int_0^x t^{\alpha-1}e^{-t}\,\mathrm{d}t. \tag{4.14}$$

Algorithm 4.8 Gamma random variables via Marsaglia and Tsang (2000a).

```
rgam(α)
if α < 1 then
  U ∼ U(0,1)
  X ← rgam(α + 1)        // Handle α < 1 by recursion
  return U^{1/α} X
d ← α − 1/3
c ← 1/√(9d)
repeat
  repeat
    Z ∼ N(0,1)
    V ← 1 + cZ
  until V > 0            // Now V is truncated normal
  V ← V^3
  U ∼ U(0,1)
  if U < 1.0 − 0.0331 × Z^4 then
    return d × V
  if log(U) < 0.5 × Z^2 + d(1 − V + log(V)) then
    return d × V
until forever            // Algorithm returns from within the loop
```

Taking $U \sim \mathbf{U}(0,1)$ and $X = F_\alpha^{-1}(U) = \gamma_\alpha^{-1}(U\Gamma(\alpha))$ produces $X \sim \mathrm{Gam}(\alpha)$. Many computing environments provide F_α^{-1}. When inversion is either not available or is slow, then acceptance-rejection is the method of choice. No one-liners are known for sampling $\mathrm{Gam}(\alpha)$, where methods using γ_α^{-1} don't count as one-liners.

Acceptance-rejection requires a dominating density. The $\mathrm{Gam}(\alpha)$ density is unbounded if $\alpha < 1$, but not otherwise. As a result, efficient dominating densities look different for $\alpha \geqslant 1$ than for $\alpha < 1$.

For $\alpha < 1$, Marsaglia and Tsang (2000a) use the following result: If $U \sim \mathbf{U}(0,1)$ and $X \sim \mathrm{Gam}(\alpha+1)$ are independent, then $XU^{1/\alpha} \sim \mathrm{Gam}(\alpha)$. As a result, generators for $\alpha \geqslant 1$ can be extended to handle $\mathrm{Gam}(\alpha)$ for $0 < \alpha < 1$.

For large α, the $\mathrm{Gam}(\alpha)$ density is similar to $\mathcal{N}(\alpha, \alpha)$ in the center, but has a much heavier right tail. An early and very influential gamma generator of Ahrens and Dieter (1974) took proposals from a density that combines a Gaussian density in the center and an exponential density in the right tail.

Marsaglia and Tsang (2000a) present an acceptance-rejection algorithm for $\mathrm{Gam}(\alpha)$ with $\alpha \geqslant 1$. It employs several clever devices to get a fast yet simple generator. For instance, if Y has the density $h(y)^{\alpha-1}e^{-h(y)}h'(y)/\Gamma(\alpha)$ for monotone h, then $X = h(Y) \sim \mathrm{Gam}(\alpha)$ by the change of variable formula (4.8). After some analysis, they choose to work with $h(Y) = d(1+cY)^3$ for $d = \alpha - 1/3$ and $c = 1/\sqrt{9d}$ with Y from a truncated $\mathcal{N}(0,1)$ distribution, also obtained by acceptance-rejection.

Algorithm 4.8 presents their method for $\alpha \geqslant 1$, along with a recursion for

the $\alpha < 1$ case. The test for $U < 1 - 0.0331Z^4$ is a squeeze that prevents calls to the logarithm from being executed frequently. The arbitrary looking constant 0.0331 does not have to be given to high accuracy, because it only appears in an inequality. The final test is written in terms of the logarithm, instead of using a mathematically equivalent test that would involve the exponential. They find that this is faster.

Several important distributions can be sampled using gamma random variables.

Example 4.28 (General chi-squared distribution)**.** In Example 4.15 we considered the $\chi^2_{(m)}$, a sum of squared $\mathcal{N}(0,1)$ random variables. The $\chi^2_{(m)}$ distribution is the same as $2\mathrm{Gam}(m/2)$, and sampling via the gamma distribution lets us avoid generating and summing m random variables. This distribution exists for any $m > 0$, not just integers.

Example 4.29 (Beta distribution)**.** The $\mathrm{Beta}(\alpha, \beta)$ distribution has probability density function

$$f(x;\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}, \quad 0 < x < 1,$$

for $\alpha > 0$ and $\beta > 0$. The normalizing constant is $B(\alpha,\beta) = \Gamma(\alpha)\Gamma(\beta)/\Gamma(\alpha+\beta)$ where $\Gamma(\cdot)$ is given by (4.13). For $X \sim \mathrm{Beta}(\alpha,\beta)$, $\mathbb{E}(X) = \alpha/(\alpha+\beta)$ and $\mathrm{Var}(X) = \mathbb{E}(X)(1-\mathbb{E}(X))/(\alpha+\beta+1)$. If $X \sim \mathrm{Gam}(\alpha)$ and $Y \sim \mathrm{Gam}(\beta)$ (independently), then $V = X/(X+Y) \sim \mathrm{Beta}(\alpha,\beta)$. Furthermore V is independent of $X + Y \sim \mathrm{Gam}(\alpha+\beta)$.

Example 4.30 (t distribution)**.** Student's t distribution on $\nu > 0$ degrees of freedom, denoted $t_{(\nu)}$ has probability density function

$$f(x;\nu) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1 + \frac{x^2}{\nu}\right)^{-(\nu+1)/2}, \quad x \in \mathbb{R}.$$

For $\nu = 1$, Student's distribution $t_{(\nu)}$ is the standard Cauchy distribution, while as $\nu \to \infty$ it approaches $\mathcal{N}(0,1)$. It is used in statistical testing, where it arises as the distribution of $Z/\sqrt{V/\nu}$ for $Z \sim \mathcal{N}(0,1)$ independently of $V \sim \chi^2_{(\nu)}$. This rule lets us simulate $t_{(\nu)}$ random variables using $2\mathrm{Gam}(\nu/2)$ and $\mathcal{N}(0,1)$ random variables. The t distribution is sometimes used in simulations where we need a distribution that is similar to $\mathcal{N}(0,1)$ but has heavier tails.

Example 4.31 (F distribution)**.** The F distribution with n numerator and d denominator degrees of freedom is denoted $F_{n,d}$. It is the distribution of X/Y where $X \sim \chi^2_{(n)}/n$ independent of $Y \sim \chi^2_{(d)}/d$.

4.9 Mixtures and automatic generators

Many important distributions are defined as mixtures of other distributions. Usually $X \sim f(\cdot;\theta)$ where the parameter θ is itself random with distribution G.

Then we may sample X directly from its definition

$$\theta \sim G, \quad \text{followed by} \quad X \sim f(\cdot;\theta). \tag{4.15}$$

Example 4.32 (Beta-binomial distribution)**.** The random variable X has a beta-binomial distribution with parameters $\alpha > 0$, $\beta > 0$ and integer $n \geqslant 1$ if the distribution of X given $\theta \in (0,1)$ is $\mathrm{Bin}(n,\theta)$ where $\theta \sim \mathrm{Beta}(\alpha,\beta)$.

Example 4.33 (compound Poisson distribution)**.** A compound Poisson distribution is the distribution of a quantity $S = \sum_{i=1}^{N} X_i$ where $N \sim \mathrm{Poi}(\lambda)$ and X_i are IID from a distribution F. By convention $S = 0$ when $N = 0$. The parameters are $\lambda > 0$ and the distribution F. One use is to model the total of a random number of insurance claims.

Some distributions can be written as mixtures in a way that makes for easy simulation.

Example 4.34 (Negative binomial distribution)**.** The negative binomial distribution with parameters $r \in \{1,2,\dots\}$ and $p \in (0,1)$, denoted $\mathrm{Negbin}(r,p)$, has probability mass function

$$p_k = \mathbb{P}(X = k) = \binom{k+r-1}{r-1} p^r (1-p)^k, \quad k = 0,1,\dots.$$

It describes the number of failures before the r'th success in a sequence of independent Bernoulli trials with success probability p. For $r = 1$, it reduces to the geometric distribution. The negative binomial distribution has the following compound representation: $X \sim \mathrm{Poi}(\lambda)$ for $\lambda \sim \mathrm{Gam}(r) \times (1-p)/p$. We don't need r to be an integer. Writing

$$\binom{k+r-1}{r-1} = \frac{(k+r-1)!}{(r-1)!k!} = \frac{\Gamma(k+r)}{\Gamma(r)\Gamma(k+1)}$$

yields a valid distribution

$$p_k = \mathbb{P}(X = k) = \frac{\Gamma(k+r)}{\Gamma(r)\Gamma(k+1)} p^r (1-p)^k$$

for $k \in \{0,1,2,\dots\}$ for any real $r > 0$. The Poisson-gamma mixture representation also holds for $r > 0$. Using the compound representation we find (Exercise 4.16) that $\mathbb{E}(X) = r(1-p)/p$ and $\mathrm{Var}(X) = r(1-p)/p^2$.

Example 4.35 (Noncentral chi-squared)**.** The non-central chi-squared CDF is an infinite sum. The terms in that sum are chi-squared CDFs multiplied by Poisson probabilities leading to the representation

$$Y \sim \chi^2_{(\nu+2K)}, \quad \text{where} \quad K \sim \mathrm{Poi}(\lambda/2).$$

for $Y \sim \chi'^2_{(\nu)}(\lambda)$. When ν is not an integer, we may replace the $\chi^2_{(m)}$ distribution ($m = \nu + 2K$) by the $2\mathrm{Gam}(m/2)$ distribution. This Poisson mixture representation is available for non-integer real-valued $\nu > 0$, where the interpretation as

a sum of ν squared Gaussian random variables does not hold. The case $\nu = 0$ and $\lambda > 0$ is especially interesting. The sum of 0 squared Gaussians could not have a nonzero sum of squared means λ. But the mixture representation does have a meaningful interpretation for this case. The $\chi'^2_{(0)}(\lambda)$ distribution has an atom at zero, $\mathbb{P}(Y = 0) = e^{-\lambda/2}$, but when $\lambda > 0$, it also takes strictly positive values with the remaining $1 - e^{-\lambda/2}$ probability.

Mixture sampling methods are often used inside the random number generators for popular computing environments. These methods typically require lots of careful bookkeeping to produce. Most individual users of Monte Carlo methods would not program them but may still want to understand how they work. The task of organizing the bookkeeping is often also done by computer, yielding computer generated code for sampling random numbers.

Figure 4.3 shows the ***half-normal*** pdf $f(z) = 2\varphi(z)$ for $0 \leqslant z < \infty$, approximated by rectangles, wedges, and a tail. Given a sample X from the half-normal distribution, we obtain a $\mathcal{N}(0, 1)$ random variable as $\pm X$, with each sign having probability $1/2$ (independent of X).

In the ***rectangle-wedge-tail*** method we use mixture sampling (4.15) to approximate f by a mixture distribution whose components are the rectangles, wedges and one or two tails. In Figure 4.3, the half-normal pdf f is represented as a mixture of 10 uniform distributions (the rectangles), 10 nearly triangular distributions (the wedges) and one infinite tail. Having decomposed f this way, we need to set up the algorithm by computing the probabilities of the 21 relevant components. Then to sample from f we choose a component at random and sample from it.

If the chosen region is a rectangle over the interval $[a, b]$, then the sample generated is $\mathbf{U}[a, b]$. If the chosen region is a wedge over the interval $[a, b]$, then the sample is generated by sampling from the wedge shaped density

$$\frac{\varphi(z) - \varphi(b)}{\Phi(b) - \Phi(a) - (b - a)\varphi(b)}$$

on the interval $[a, b]$. If the chosen region is the tail, supported on $[B, \infty)$, then we can sample by acceptance-rejection sampling from the distribution $\varphi(z)/(1 - \Phi(B))$ on the region $[B, \infty)$. An exponential distribution shifted by B units, as in Example 4.26, is a suitable proposal distribution, especially for moderately large B, such as $B \geqslant 2$.

Library routines could use many more than 10 rectangles. That raises the fraction of samples which come from uniformly distributed mixture compnents and will ordinarily result in greater speed. Also, it is not necessary to use equal width intervals. For unimodal and asymmetric f, the region to the left of the mode is decomposed into rectangles, decreasing wedges and a tail to the left.

The ***ziggurat*** method makes a decomposition of f, but it uses horizontal rectangles. For the half-normal case, it generates a sample point (Z, Y) uniformly distributed in $\mathcal{S}_c(f) = \{(z, y) \mid 0 \leqslant y \leqslant \exp(-z^2/2),\ 0 \leqslant z < \infty\}$, where $c = \sqrt{\pi/2}$. The resulting Z coordinate has the half-normal distribution. A speed advantage arises from ignoring the $1/\sqrt{2\pi}$ normalizing factor. Figure 4.4

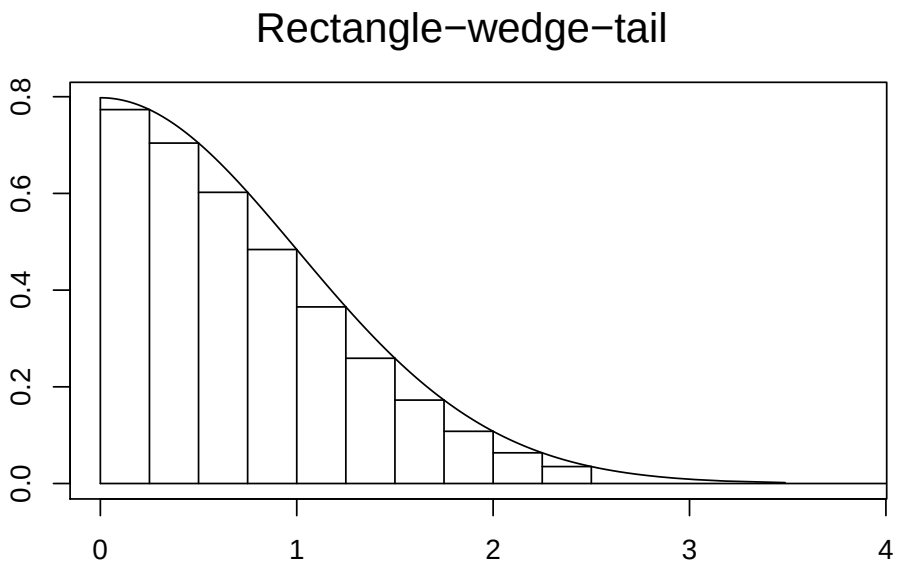

Figure 4.3: This figure illustrates a rectangle-tail-wedge decomposition of the half-normal probability density function. There are 10 rectangles, 10 nearly triangular wedges, and one tail.

sketches the ziggurat method when 8 horizontal regions are used. The top 7 regions are horizontal rectangles and they include some points outside of $\mathcal{S}_c(f)$. The bottom region is whatever is left over. In applications, the method might have 256 regions instead of the eight shown.

By making a very careful choice for the rectangles it is possible to make all of the regions equi-probable. Having equiprobable regions speeds up the first step of mixture sampling in which we select the mixture component. When the method selects a horizontal rectangle it then follows up by acceptance-rejection sampling within that rectangle. Most of the values can be immediately accepted but the regions at the right-hand side of the rectangle require acceptance rejection. When the bottom region is chosen, it can be sampled as a mixture of a horizontal subregion and a tail.

The ziggurat method is very fast. Setting up for it requires solving a complicated set of equations. The rectangles in Figure 4.4 are only approximate, obtained by mimicking Figure 1 of Marsaglia and Tsang (2000b) by eye. That reference describes how to search for the right split points and gives useful results for splitting the half-normal and exponential distributions into either 128 or 256 horizontal strips.

A third automated generation method is available for log concave density functions. If the function $\log f(x)$ is concave, then it lies below its tangent at any point x. If we then evaluate $\log f(x)$ at k points $x_1 < x_2 < \cdots < x_k$ we can find an upper bound for $\log(f)$ made up of $k+1$ linear segments. Figure 4.5

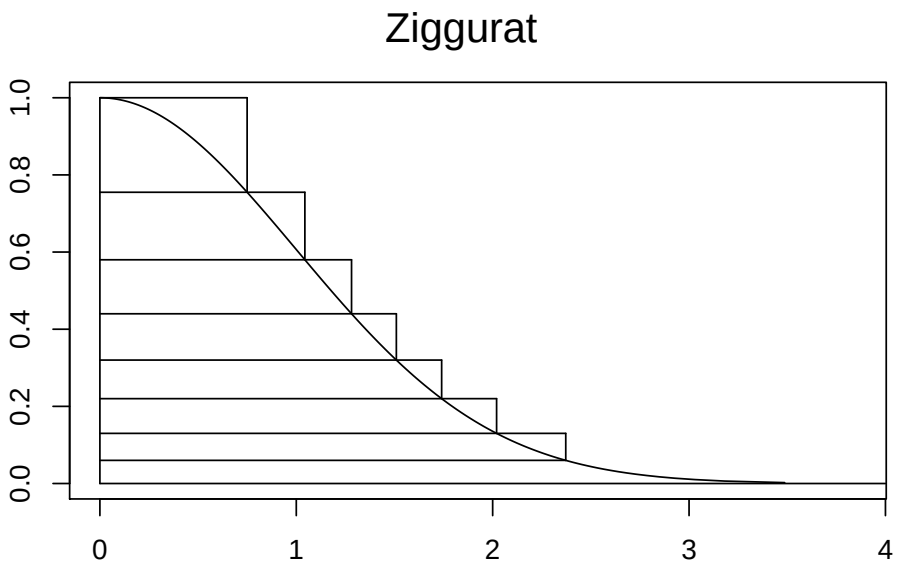

Figure 4.4: This figure illustrates a ziggurat decomposition of the half-normal probability density function $\varphi(z) \propto \exp(-z^2/2)$ on $0 \leqslant z < \infty$. The density is covered by 7 rectangular regions and one other (bottom) region.

shows such an upper bound for the Gam(4) distribution using $k = 3$ points. When the random variable X has no upper bound, then $\log f(x)$ must have negative slope at x_k or else the upper bound cannot be normalized. That is we require $f'(x_k) < 0$. Similarly, when X has no lower bound, we need $f'(x_1) > 0$. For unimodal f it is enough to have at least one point x_j on each side of the mode.

Since $\log g$ is piecewise linear, g itself is piecewise exponential. We can therefore normalize g and find the probability that it assigns to each of pieces on which it is exponential. If g is decreasing on a line segment of finite length, then we can sample within that segment by adding a truncated exponentially distributed quantity to the left endpoint. If g is increasing on a finite length interval, we can subtract a truncated exponential random variable from its right endpoint. For unbounded intervals, as might occur at the right or left endpoints of the range of f, we can add (respectively subtract) an exponentially distributed random variable to the finite endpoint.

In ***adaptive rejection sampling*** we update the piecewise linear approximation to $\log f$ as sampling proceeds. When a proposal $x \sim g$ is rejected we add a new tangent line to g at that point x. The function $\log g$ very quickly approaches $\log f$. The method can be sped up by using a squeeze function h such that $\log h$ is piecewise linear and lies under $\log f$. We can take $\log h$ to be piece-wise linear connecting the tangent points $(x_k, \log f(x_k))$ on the interval

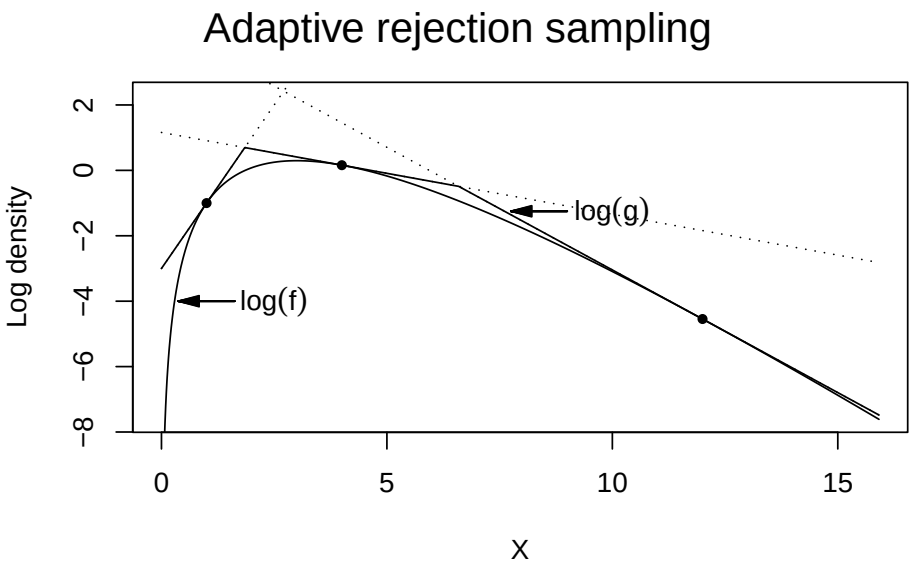

Figure 4.5: This figure illustrates adaptive rejection sampling. Here $\log f(x)$ is the logarithm of the Gam(4) density and $\log g(x)$ is a piecewise linear upper bound for $\log f$ based on the tangents to $\log f$ at points $x \in \{1, 4, 12\}$. Acceptance-rejection sampling can use g as a proposal distribution. In adaptive rejection sampling the locations of rejected points can be used to refine the piecewise linear function $\log g$ bringing it closer to $\log f$.

$[x_1, x_k]$ and let $\log h = -\infty$ outside this interval. The squeeze will also accept a greater fraction of generated points as the number of segments increases.

In practice we might stop updating g once the rejection rate has fallen below a threshold. It is also possible to precompute a piecewise linear approximation to $\log f$ given a target acceptance probability.

Chapter end notes

For an extensive discussion of non-uniform random number generation, see Devroye (1986). The truncation based algorithm for the geometric distribution in Example 4.12 is from page 500 of that book. Among the widely cited methods covered there (but not here) are the ratio of uniforms method, which is a form of acceptance-rejection sampling, and the alias method, which is an alternative to guide tables for sampling discrete random variables.

Von Neumann (1951) presents both inversion and acceptance-rejection from a uniform proposal distribution. The proof of Theorem 4.2 is expanded from that of Knuth (1998), who does it in just under three lines. Robert and Casella (2004) have called the geometric view of acceptance-rejection sampling (encom-

passing Theorems 4.3 and 4.4) the fundamental theorem of simulation.

The rectangle-wedge-tail method is due to Marsaglia et al. (1964). The ziggarut method is from Marsaglia and Tsang (1984, 2000b). Adaptive rejection sampling is due to Gilks and Wild (1992). Hörmann et al. (2004) emphasize automatic generation of random variables. They include generalizations of adaptive rejection sampling in which $T(f(x))$ is concave for more general functions T than the logarithm considered in §4.9.

There are a great many variants of the bootstrap idea. For more background on the bootstrap, see Efron and Tibshirani (1994) and Davison and Hinkley (1997).

The Box-Muller method for sampling normal random variables is due to Box and Muller (1958). Several methods for sampling normal random variables by inversion are given in §4.3. Moro (1995) gives an implementation of Φ^{-1} that is popular in the finance literature. It is based on repairing the problem that Beasley-Springer has in the tails of the distribution. The combination is sometimes known as Beasley-Springer-Moro. It is not more accurate than Wichura's method. A comprehensive survey (Thomas et al., 2007) of methods for sampling the normal distribution finds that the ziggurat method is generally fastest apart from one method which they report suffers from correlation problems.

The noncentral chi-squared distribution on 0 degrees of freedom is due to Siegel (1979).

Box-Muller in more detail

Here are the rest of the details for the Box-Muller transformation. The joint density of two normal random variables Z_1 and Z_2 is

$$f(z_1, z_2) = \frac{e^{-z_1^2/2}}{\sqrt{2\pi}} \frac{e^{-z_2^2/2}}{\sqrt{2\pi}} = \frac{1}{2\pi} e^{-(z_1^2+z_2^2)/2}.$$

To get the distribution of polar coordinates r and θ we replace z_1 and z_2 by equivalent expressions in r and θ and then multiply the result by the absolute value of the determinant of the Jacobian matrix

$$J = \frac{\partial(z_1, z_2)}{\partial(r, \theta)} = \begin{pmatrix} \dfrac{\partial z_1}{\partial r} & \dfrac{\partial z_1}{\partial \theta} \\ \dfrac{\partial z_2}{\partial r} & \dfrac{\partial z_2}{\partial \theta} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & r\sin(\theta) \\ \sin(\theta) & -r\cos(\theta) \end{pmatrix}.$$

We find that $|\det(J)| = r$ and with $z_1^2 + z_2^2 = r^2$ we arrive at the joint density

$$f_{r,\theta}(r, \theta) = \frac{1}{2\pi} r\, e^{-r^2/2}, \quad 0 \leqslant \theta < 2\pi, \quad 0 < r < \infty.$$

The density factors into a function of R times a (constant) function of θ, and so R and θ are independent. The distribution of θ is $\mathbf{U}[0, 2\pi)$.

The density of R is $f_r(r) = r\exp(-r^2/2)$ on $0 < r < \infty$. Now let $S = R^2$. Inverting $\tau(R) = R^2$ yields $\tau^{-1}(S) = S^{1/2}$ and so $(d/dS)\tau^{-1}(S) = S^{-1/2}/2$.

Equation (4.8) for the probability density function of a monotone transformation yields

$$f_s(s) = \frac{s^{-1/2}}{2} f_r(s^{1/2}) = \frac{s^{-1/2}}{2} s^{1/2} e^{-s/2} = \frac{1}{2} e^{-s/2},$$

and so $R^2 \sim 2 \times \mathrm{Exp}(1)$ as we claimed in §4.6.

We have in fact been able to dodge the worst issue in this transformation. Expressing θ as a function of Z_1 and Z_2 is somewhat awkward, because it involves multiple branches of the arctangent function. However θ did not appear in either the raw density or the Jacobian.

Exercises

4.1. Define a double Weibull distribution to generalize the Weibull distribution in the same way that the double exponential generalizes the exponential. The distribution should be symmetric. Give an expression for its probability density function, its cumulative distribution function and the inverse CDF.

4.2. Determine the values for $F^{-1}(0)$ and $F^{-1}(1)$ if equation (4.2) is used directly, instead of through a limit, for these cases:

a) F is the $\mathcal{N}(0,1)$ CDF,

b) F is the $\mathbf{U}(0,1)$ CDF.

Note: the usual convention is that $\inf \varnothing = \infty$.

4.3. The QQ transformation (4.3) does not necessarily work if F is a discrete random variable.

a) Find example distributions F and G where it fails to work, and show what the resulting distribution of $G^{-1}(F(X))$ is.

b) Does the QQ transformation ever work for a discrete distribution F? If so, find a pair of distributions F and G, with F discrete and $G \neq F$, for which it does work and prove that it works. If not, prove that it does not work for any pair of CDFs with F discrete and $G \neq F$.

4.4. Suppose that $F(x) = \prod_{j=1}^{J} F_j(x)$ where F_j are CDFs from which we can sample easily. Describe a way of sampling $X \sim F$.

4.5. Suppose that there is a 20% chance of rain on a given day. If it does rain then the amount of rain has an exponential distribution with mean 1/2. We could easily sample the amount of rain with a mixture, but we choose inversion because that will allow variance reduction methods from later chapters to be used. Write out the inverse CDF for the distribution of the amount of rain.

4.6. Let $a < b$ and $a \leqslant c \leqslant b$. The triangular density f on the interval $[a, b]$ with mode at $c \in [a, b]$ is continuous and piecewise linear. If $a < c < b$ then f is linear on $[a, c]$ and linear on $[c, b]$ with $f(a) = f(b) = 0$. If $a = c < b$, then f

is linear on $[a,b]$ with $f(b)=0$. If instead $a<c=b$, then f is linear on $[a,b]$ with $f(a)=0$. We write $X\sim\mathrm{Tri}(a,b,c)$ for this distribution. It is often used when we have only qualitative information, such as the possible values and most likely value, for an input variable to a simulation.

For the case with $a<c<b$, derive an expression for:

a) the height $f(c)$ of this density at the mode,

b) the PDF $f(x)$,

c) the CDF $F(x)$,

d) the inverse CDF F^{-1}.

4.7. Write a function `discrand`$(\boldsymbol{a},u)$ so that if $X=$ `discrand`$(\boldsymbol{a},U)$ for $U\sim\mathbf{U}(0,1)$, then $\mathbb{P}(X=j)\propto a_j$. Here $\boldsymbol{a}=(a_1,\dots,a_M)\in[0,\infty)^M$ with $\boldsymbol{a}\neq(0,\dots,0)$ and the a_j need not sum to 1. It should sample by inversion so that if $u\leqslant u'$ then `discrand`$(\boldsymbol{a},u)\leqslant$ `discrand`$(\boldsymbol{a},u')$. It can be based on either a sequential scan, or the faster bisection search from Algorithm 4.3. The function should work correctly if given $u=0$ or $u=1$. In particular, if $a_j=0$, then no $u\in[0,1]$ should give `discrand`$(\boldsymbol{a},u)=j$.

The function should reject its input if any of the following hold:

i) $M\leqslant 0$,
ii) $U<0$ or $U>1$,
iii) any $a_j<0$,
iv) any $a_j=\infty$, or,
v) all $a_j=0$.

Rejection can mean returning an error indicator, printing a warning message, stopping the computation or throwing an exception, depending on the language you are using. Rejection should be conspicuous to the user.

a) Test your function on examples of all the invalid input conditions above and verify that the errors are caught. Omit cases that are ruled out by your computing language. (For example, if your system does not allow for vectors with 0 components, then you need not test case i.)

b) Summarize the results of applying your code on the following vectors $\boldsymbol{a}$

$$(1,2,3,4)\qquad(0,1,2,3,4)\qquad(1,2,3,4,0)\qquad(0,1,2,3,4,0)$$

for each $u=(i-1/2)/1000$ for $i=1,\dots,1000$.

c) Repeat the previous test retaining the vectors $\boldsymbol{a}$ but using $u=i/10$ and $i=0,1,\dots,10$. Compare your results to the correct ones as defined by equation (4.5). They might mismatch due to roundoff errors in $i/10$. The flaw is only serious if it leads to X taking values that should have probability 0.

Notes: Make necessary adjustments if your computing environment has a different syntax for functions. Also, if you prefer to have $X\in\{0,1,\dots,M-1\}$ with $\boldsymbol{a}=(a_0,\dots,a_{M-1})$, then do that. If your computing environment cannot

determine the dimension M from the given vector $\boldsymbol{a}$, then pass M directly to the function.

If you prefer, write `discrand` so that it takes input $\boldsymbol{u} \in [0,1]^n$ and returns $\boldsymbol{x}$ with $x_j = \texttt{discrand}(\boldsymbol{a}, u_j)$.

In applications it may be more convenient to let `discrand` call the random number generator itself, but at least for this exercise, write it as described above.

4.8. Consider the function `discrand` from Exercise 4.7. Here we investigate what it does when $\max_{1\leqslant j\leqslant M} a_j < \infty$ but $\sum_{j=1}^M a_j$ overflows to ∞ in floating point.

a) Find a value $c < \infty$ such that $10 \times c$ overflows to $+\infty$ in your floating point system.

b) If the `discrand` algorithm starts by forming $\mathbb{P}(X = j) \equiv p_j = a_j / \sum_{k=1}^M a_k$ for $j = 1, \dots, M$ then it will get $p_j = 0$ and then fail. Find another way to implement `discrand` that does not fail this way. What values do you get for $\texttt{discrand}(\boldsymbol{a}, u)$ for $\boldsymbol{a} = (c, c, \dots, c) \in (0,\infty)^{10}$ and each of $u = 0$, $u = 1/3$ and $u = 1$? What values should you get?

4.9. Here we prove that to sample $X = k$ with probability proportional to $q_k > 0$ for $k = 1, \dots, N$ it suffices to choose the k that minimizes E_j/q_j over $1 \leqslant j \leqslant N$ for $E_j \overset{\text{iid}}{\sim} \text{Exp}(1)$.

a) For $X_j = E_j/q_j$, find $\mathbb{P}(\min_{2\leqslant j\leqslant N} X_j > t)$.

b) Now find $\mathbb{P}(X_1 < Y)$ where $Y = \min_{2\leqslant j\leqslant N} X_j$.

4.10. Here we follow up Example 4.25 on acceptance-rejection sampling for the $\mathcal{N}(0,1)$ distribution.

a) Show that the ratio $f(x)/g(x)$ of $\mathcal{N}(0,1)$ to standard Cauchy densities is maximized at $x = \pm 1$.

b) The double exponential distribution has probability density function $g(x) = \exp(-|x|)/2$. Show that proposals from g can be used in acceptance-rejection to sample from the $\mathcal{N}(0,1)$ distribution by proving that the threshold constant $c = \sup_x f(x)/g(x)$ is finite.

c) Which proposal distribution (Cauchy or double exponential) has a higher acceptance rate for sampling the $\mathcal{N}(0,1)$ distribution?

4.11. Here we consider sampling the $\mathcal{N}(0,1)$ distribution truncated to the interval $[a,\infty)$ where $a > 0$. Inverting the truncated CDF leads to the formula $X = g_1(U) \equiv \Phi^{-1}(\Phi(a) + (1-\Phi(a))U)$ where $U \sim \mathbf{U}(0,1)$.

a) Choose a computing environment with double precision floating point implementations of Φ and Φ^{-1}. Now find the smallest integer $a \geqslant 1$ for which $X = g_1(U)$ fails to work. For concreteness we can define failure as delivering at least one NaN or $\pm\infty$ when tested with $U \in \{(i-1/2)/1000 \mid$

$i = 1, 2, \ldots, 1000\}$. Provide the details of your envirnoment, naming the kind of computer, the software, and which specific functions you used for Φ and Φ^{-1}.

b) Show that $X = g_2(U) \equiv -\Phi^{-1}(\Phi(-a)(1-U))$ also generates X from the same distribution.

c) Find the smallest integer $a \geqslant 1$ at which g_2 fails using the same criterion as for g_1.

High quality implementations of Φ and Φ^{-1} will work for a corresponding to quite small tail probabilties, such as $\mathbb{P}(\mathcal{N}(0,1) > a)$ well below 10^{-100}. For even more extreme truncated normal random variables, acceptance-rejection sampling remains effective (Robert, 1995).

4.12. Determine a method to draw samples from the distribution with probability density function proportional to $\exp(-x^4/12)$ for $x \in \mathbb{R}$. (This distribution arises in the study of a one dimensional Ising model at the critical temperature.)

4.13. Give a formula to generate a sample from the exponential distribution with mean $\mu > 0$ truncated to the interval $[a, b]$ for $0 \leqslant a < b < \infty$, using one random variable $U \sim \mathbf{U}(0,1)$.

4.14. The Henyey-Greenstein distribution is often used to model the angle θ at which photons scatter. The photons might be passing through human tissue in a medical application or through the atmosphere in a climate modeling problem. Let $\eta = \cos(\theta)$. Then the probability density function of η is

$$f(\eta; g) = \frac{1}{2}\frac{1-g^2}{(1+g^2-2g\eta)^{3/2}}, \qquad -1 \leqslant \eta \leqslant 1$$

for a parameter $g \in (-1, 1)$. The parameter g is $\mathbb{E}(\eta)$, although that is not obvious at first sight. In applications g is typically in the range from 0.7 to almost 1.

a) Plot $f(\eta; g)$ versus η for $g = 0.97$.

b) Develop a way to sample the Henyey-Greenstein distribution for $g = 0.97$. Turn in a mathematical derivation of your method along with your code.

c) Estimate the median of η by Monte Carlo, when $g = 0.97$ and give a 99% confidence interval.

d) Estimate $\mathbb{E}(\theta)$ for $g = 0.97$ and give a 99% confidence interval.

e) Estimate $\mathrm{Var}(\theta)$ for $g = 0.97$ and give a 99% confidence interval.

4.15. Suppose that $X \in \mathbb{R}^{n\times n}$ is a random matrix with independent $\mathcal{N}(0,1)$ elements. Let ℓ_1 be the *smallest* eigenvalue of $S_n = X^{\mathsf{T}}X/n$ and let $Y = n\ell_1$. Edelman (1988) shows that as $n \to \infty$, the probability density function of Y approaches

$$f(y) = \frac{1+\sqrt{y}}{2\sqrt{y}}\, e^{-(y/2+\sqrt{y})}, \qquad 0 < y < \infty. \tag{4.16}$$

a) Develop a method to sample Y from the density f given in (4.16). Turn in your derivation and code.

b) Test your method by estimating $\mathbb{E}(\log(Y))$ by simple Monte Carlo, and giving a 99% confidence interval. Edelman (1988) found that the answer was roughly -1.68788.

c) Estimate $\mathbb{E}((\log(Y) + 1.68788)^2)$ by Monte Carlo. Give a confidence interval for this mean square.

4.16. Using the compound representation of the negative binomial distribution, show that when $X \sim \text{Negbin}(r, p)$ that $\mathbb{E}(X) = r(1-p)/p$ and $\text{Var}(X) = r(1-p)/p^2$.

4.17. An insurance company will face a random number N of claims of a given type in the coming year. The distribution of N is negative binomial with $\mathbb{E}(N) = \mu$ and $\text{Var}(N) = 50\mu$, so it has a heavier right tail than a Poisson distribution. If $N > 0$, then for $i = 1, \dots, N$, the customer with the i'th claim has lost an amount given by $X_i \sim \theta\text{Gam}(\alpha)$. The X_i are independent of each other and of N. The total loss is $L = \sum_{i=1}^{N} X_i$. Each customer has a deductable τ_0 and so only claims $Y_i = \max(0, X_i - \tau_0)$. The insurance policy imposes a limit τ_1 on the payment to any customer, and so the company pays $Z_i = \min(\tau_1, Y_i)$ to customer i. The total of claims made to the insurance company is $C = \sum_{i=1}^{N} Z_i$. The insurance company reinsures with a second company. The reinsurance company pays $R = \min(\rho_1, \max(0, C - \rho_0))$ to the insurance company. The customer claims will therefore cost the insurance company $C - R$. For the parameter values below, use Monte Carlo to simulate the year's claims 100,000 times. Report your estimate, and 99% confidence interval for each of the following:

a) The expected cost to the insurance company.

b) The expected cost to the reinsurance company.

c) The expected sum of unreimbursed costs to all customers.

d) The probability that the reinsurance company has to pay anything.

e) For which of these items is the simulation result least accurate, as measured by the width of the confidence interval divided by the value in the center?

The parameter values for this problem are $\mu = 1{,}000$, $\alpha = 0.5$, $\theta = 20{,}000$, $\tau_0 = 1{,}000$, $\tau_1 = 200{,}000$, $\rho_0 = 15{,}000{,}000$ and $\rho_1 = 100{,}000{,}000$. The compound representation of the negative binomial distribution be useful.

4.18. Let $X > 0$ be a random variable. Then the hazard function $h(x)$ is defined as

$$h(x) = \lim_{t \to 0^+} \frac{1}{t} \mathbb{P}(X \leqslant x + t \mid X \geqslant x).$$

The hazard function is the instantaneous probability of failure.

a) Find the hazard function for the exponential distribution with rate λ.

b) Find the hazard function for the Weibull distribution with scale parameter σ and shape k.

Fault tolerant systems

Fault-tolerant systems with redundant components are useful when it is expensive to replace failed components. Instead of replacing each component that fails we set up a bundle of components including some spares. When the number of working components falls below a threshold the whole bundle gets replaced.

Using the beta distribution it is possible to develop a one-dimensional representation for the failure time of a fault tolerant system. That representation makes possible some more powerful methods than simple Monte Carlo, from later chapters. For example, quadrature rules (Chapter 7) and stratified sampling (Chapters 8 and 10) would be more accurate.

4.19. A fault-tolerant memory bank is built with 5 memory units. These units have independent random failure times, each with a distribution F. The memory bank is operable as long as 4 or more units are still working, so it fails when the second unit fails. Let $G(\cdot;\alpha,\beta)$ be the CDF of the Beta(α,β) distribution and $G^{-1}(\cdot;\alpha,\beta)$ be its inverse. That is, $\mathbb{P}(X \leqslant G^{-1}(u;\alpha,\beta)) = u$ when $X \sim$ Beta(α,β) and $0 < u < 1$.

a) Express the failure time of the memory bank as a function of a single variable $U \sim \mathbf{U}(0,1)$ via $F(\cdot)$ and $G(\cdot;\alpha,\beta)$ and/or their inverses.

b) Suppose that F is the exponential distribution with mean 500,000 (hours). Sample 10,000 memory bank lifetimes using the expression from part **a**. Estimate the mean memory bank life time and give a 99% confidence interval for it.

c) If we didn't use a redundant system, we might instead use 4 memory units and the bank would fail when the first of those failed. Find the expected lifetime of such a system without redundancy and give a 99% confidence interval. The redundant system takes 1.25 times as many memory units. Does it last at least 1.25 times as long on average?

d) Compare the speed of sampling by inversion with that of generating 5 exponential variables, sorting them and picking out the second smallest. Be sure to document the computer hardware and software in which you made the timing comparison.

4.20. A memory system with two-level redundancy is built as follows. The two-level system contains 5 of the fault-tolerant memory banks from Exercise 4.19. One of those 5 banks is a spare. The two-level memory system functions as long as 4 or more of those fault-tolerant memory banks are operable.

a) Express the lifetime of the memory system in terms of F and the Beta(α,β) CDF $G(\cdot;\alpha,\beta)$ and/or their inverses and one random variable $U \sim \mathbf{U}(0,1)$.

b) Estimate the mean time to failure of the entire memory system with two-level redundancy, by simple Monte Carlo using one single $U \sim \mathbf{U}(0,1)$ for each simulated two-level system. For F, take the same distribution you used in Exercise **4.19b**.

A good way to test your one-dimensional representation is to repeat the simple Monte Carlo by direct sampling of all 25 exponential variables, verifying that the answers are close enough.

Balls and bins

Exercises 4.21 through 4.23 are based on some fascinating results for 'balls and bins' from Berenbring et al. (2006) and other sources cited by them. We'll place m balls in n bins, hoping than no single bin gets many more than m/n of the balls.

The motivation is a simplified version of a load balancing problem that arises in parallel computation. Suppose that we have a source of tasks that are to be allocated to n servers. The number of tasks being handled by server j is T_j. When a task arrives, we would like to assign it to whichever server j presently has the smallest T_j, in order to balance the loads. It is however quite inconvenient to interrupt all n servers in order to find out their present load. A simple strategy is to just assign each task to a server chosen uniformly at random.

A much better load balance can be obtained by the following two-choice strategy. When a task arrives, we pick two servers j and $k \neq j$ and find their loads T_j and T_k. If $T_j < T_k$ then server j gets the task, so $T_j \leftarrow T_j + 1$, while if $T_k < T_j$ then server k gets the task. If $T_j = T_k$, then the two sampled servers are equally likely to get the task.

In a real system, tasks arrive at random, take random amounts of time to complete, and then disappear. In a simplified setting, we start with $T_1(0) = \cdots = T_n(0) = 0$. After task t has arrived the loads are $T_j(t)$. We suppose here that m tasks arrive before any are completed. It is known that for the two-choice strategy, the maximum load is

$$T^*(m) = \max_{1 \leqslant j \leqslant n} T_j(m) = \frac{m}{n} + \frac{\log(\log(n))}{\log(2)} + O(1)$$

as $n \to \infty$ with high probability. By contrast, uniform random allocation gives a maximum load of $m/n + O\bigl(\sqrt{m \log(n)/\log(\log(n))}\bigr)$ with high probability.

The asymptotic relations hide a constant inside the $O(\cdot)$ terms, and we might also wonder whether a given n is large enough for the high probability to be high enough. Monte Carlo methods are well suited for filling in the details at specific m and n.

4.21 (Two-choice versus simple random sampling)**.** Simulate both the two-choice and uniform random strategies for the values of m and n listed, for each of the three cases below:

	Case I	Case II	Case III
Number of balls: m	10	100	100
Number of bins: n	1000	1000	100

Repeat each strategy 10,000 times, and compare the histograms of $T^*(m)$ for the two methods. For each method and each case, report the 99'th percentile of $T^*(m)$ from your simulation. Also report the estimated probability that $T^*(m)$ is less than or equal to its 99'th percentile. This does not always equal 0.99, because T^* has a discrete distribution.

4.22 (Two choice versus go-left)**.** The following 'go-left' version of two-choice allocation is defined for even numbers n of bins. It differs in two ways from the two-choice method: it stratifies the sampling of bins, and it uses a deterministic tie breaker. To place a ball in a bin, go-left samples $j \sim \mathbf{U}\{1,\dots,n/2\}$ independently of $k \sim \mathbf{U}\{n/2+1,\dots,n\}$. If $T_k < T_j$ then $T_k \leftarrow T_k + 1$. Otherwise $T_j \leftarrow T_j + 1$. It is called go-left because whenever $T_j = T_k$, bin j, on the left, gets the new ball. Simulate the go-left version of the algorithm for the cases listed in Exercise 4.21 using 10,000 repetitions, independent of any from the other exercises. Compare the go-left method to the two-choice method, by comparing their 99'th percentiles. If the result is a tie, devise and apply a reasonable method to determine which is better.
Note: In this problem, we use 99% as an empirical proxy for high probability. When there is a tie, more than one reasonable way exists to decide which method is better. The method used should however be sensitive to the upper end of the distribution of T^*.

4.23. This exercise looks at the effects of the two parts of the go-left strategy.

- **a)** In this part, we turn off the stratification feature in go-left. Now it samples j and k in the same way that two-choice does. But if $T_j = T_k$ then $T_{\min(j,k)} \leftarrow T_{\min(j,k)} + 1$. For the three cases above, compare go-left without stratification to go-left with stratification and also compare it to two-choice with random tie breaking. Once again, use 10,000 repetitions. Make your comparisons at the 99'th percentile, using a tie breaker as in Exercise 4.22, if the 99'th percentiles are equal.
- **b)** Devise a way to combine the stratification of go-left with the tie breaking strategy of two-choice. Compare the resulting method to the others.
- **c)** Summarize the results. Which of the algorithms gives the best load balancing for the three cases we considered? Is stratification important? tie breaking?

Bibliography

Ahrens, J. H. and Dieter, U. (1974). Computer methods for sampling from gamma, beta, Poisson and binomial distributions. *Computing*, 12(3):223–246.

Beasley, J. D. and Springer, S. G. (1977). The percentage points of the normal distribution. *Applied Statistics*, 26(1):118–121.

Berenbring, P., Czumaj, A., Steger, A., and Vöcking, B. (2006). Balanced allocations: the heavily loaded case. *SIAM Journal of Computation*, 35(6):1350–1385.

Blair, J. M., Edwards, C. A., and Johnson, J. H. (1976). Rational Chebyshev approximation for the inverse of the error function. *Mathematics of Computation*, 30(136):827–830.

Box, G. E. P. and Muller, M. E. (1958). A note on the generation of random normal deviates. *Annals of Mathematical Statistics*, 29(2):610–611.

Chambers, J. M., Mallows, C. L., and Stuck, B. W. (1976). A method for simulating stable random variables. *Journal of the American Statistical Association*, 71(354):340–344.

Chen, H.-C. and Asau, Y. (1974). On generating random variates from an empirical distribution. *AIEE transactions*, 6(2):153–166.

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap methods and their application*. Cambridge University Press, Cambridge.

Devroye, L. (1986). *Non-uniform Random Variate Generation*. Springer, New York.

Devroye, L. (1996). Random variate generation in one line of code. In *Proceedings of the 1996 Winter Simulation Conference*, pages 265–272.

Edelman, A. (1988). Eigenvalues and condition numbers of random matrices. *SIAM Journal of Matrix Analysis and Applications*, 9(4):543–560.

Efron, B. and Tibshirani, R. J. (1994). *An Introduction to the Bootstrap*. Chapman & Hall, Boca Raton, FL.

Fishman, G. S. and Moore, L. R. (1984). Sampling from a discrete distribution while preserving monotonicity. *The American Statistician*, 38(3):219–223.

Gilks, W. R. and Wild, P. (1992). Adaptive rejection sampling for Gibbs sampling. *Journal of the Royal Statistical Society, Series B*, 41(2):337–348.

Hörmann, W., Leydold, J., and Derflinger, G. (2004). *Automatic nonuniform random variate generation*. Springer, Berlin.

Knuth, D. E. (1998). *The Art of Computer Programming*, volume 2: Seminumerical algorithms. Addison-Wesley, Reading MA, 3rd edition.

Marsaglia, G., MacLaren, M. P., and Bray, T. A. (1964). A fast procedure for generating normal random variables. *Communications of the ACM*, 7(1):4–10.

Marsaglia, G. and Tsang, W. W. (1984). A fast, easily implemented method for sampling from decreasing or symmetric unimodal density functions. *SIAM Journal on Scientific and Statistical Computing*, 5(2):349–359.

Marsaglia, G. and Tsang, W. W. (2000a). A simple method for generating gamma variables. *ACM transactions on mathematical software*, 28(3):363–372.

Marsaglia, G. and Tsang, W. W. (2000b). The ziggurat method for generating random variables. *Journal of Statistical Software*, 5(8):1–7.

Moro, B. (1995). The full Monte. *Risk*, 8(2):57–58.

Nolan, J. P. (2013). *Stable Distributions: Models for Heavy-Tailed Data*. Birkhauser, Boston.

Odeh, R. E. and Evans, J. O. (1974). Algorithm AS 70: the percentage points of the normal distribution. *Applied Statistics*, 23(1):96–97.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes: the art of scientific computing*. Cambridge University Press, Cambridge, 3rd edition.

Robert, C. P. (1995). Simulation of truncated normal variables. *Statistics and computing*, 5(2):121–125.

Robert, C. P. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer, New York, 2nd edition.

Siegel, A. F. (1979). The noncentral chi-squared distribution with zero degrees of freedom and testing for uniformity. *Biometrika*, 66(2):381–386.

Thomas, D. B., Luk, W., Leong, P. H. W., and Villasenor, J. D. (2007). Gaussian random number generators. *ACM Computing Surveys*, 39(4):Article 11.

von Neumann, J. (1951). Various techniques used in connection with random digits. Monte Carlo method. *National Bureau of Standards Applied Math Series*, 12:36–38.

Wichura, M. J. (1988). Algorithm AS 241: the percentage points of the normal distribution. *Applied Statistics*, 37(3):477–483.