Modern statistics is dominated by computations made by simulation. There are many many clever simulation ideas; here we discuss only the basics. We imagine we are given random variables $X_1, \ldots, X_n$ whose joint distribution is somehow specified. We are interested in some statistic $T(X_1, \ldots, X_n)$ whose distribution we want.

Here is the basic Monte Carlo method to compute the survival function of T, that is, to compute $P(T > t)$:

1. Generate $X_1, \ldots, X_n$ from the density f.
2. Compute $T_1 = T(X_1, \ldots, X_n)$.
3. Repeat this process independently N times getting statistic values $T_1, \ldots, T_N$.
4. Estimate $p = P(T > t)$ by $\hat{p} = M/N$ where M is number of repetitions where $T_i > t$.
5. Estimate the accuracy of $\hat{p}$ using $\sqrt{\hat{p}(1-\hat{p})/N}$. In the jargon of later chapters this is the estimated standard error of $\hat{p}$.

Note: The accuracy of this computational method is inversely proportional to $\sqrt{N}$.

Next: we review some tricks to make the method more accurate.

Warning: The tricks only change the constant of proportionality — the standard error is still inversely proportional to $\sqrt{N}$.

0.0.1 Generating the Sample

Step 1 in the overall outline just presented calls for "generating" samples from the known distribution of $X_1, \ldots, X_n$. In this subsection I want to try to explain what is meant. The basic idea is to carry out an experiment which is like performing the original experiment, generating an outcome ω and calculating the value of the random variables. Instead of doing a real experiment we use a *pseudo-random number generator*, a computer program which is intended to mimic the behaviour of a real random process. This relies on a basic computing tool: pseudo uniform random numbers — variables U which have (approximately) a Uniform$[0, 1]$ distribution. I will not be discussing the algorithms used for such generators. Instead we take them as

a given, ignore any flaws and pretend that we have a way of generating a sequence of independent and identically distributed Uniform[0,1] variables.

0.0.2 Transformation

Other distributions are often then generated by transformation:

Example: **Exponential**: If U is Uniform[0,1] then $X = -\log U$ has an exponential distribution:

$$\begin{aligned} P(X > x) &= P(-\log(U) > x) \\ &= P(U \leq e^{-x}) = e^{-x} \end{aligned}$$

This generator has the following pitfall: random uniform variables generated on a computer sometimes have only 6 or 7 digits. As a consequence the tail of the generated distribution (using the transformation above) is grainy.

Here is a simplified explanation. Suppose the generated value of U is always a multiple of 10^{-6}. Then the largest possible value of X is $6\log(10)$ and the number of values larger than $3\log(10) = 6.91$ is 1000

Here is an improved algorithm

- Generate U a Uniform[0,1] variable.
- Pick a small ϵ like 10^{-3} say. If $U > \epsilon$ take $Y = -\log(U)$.
- If $U \leq \epsilon$ we make use of the fact that the conditional distribution of $Y - y$ given $Y > y$ is exponential. Generate an independent new uniform variable U'. Compute $Y' = -\log(U')$. Take $Y = Y' - \log(\epsilon)$.

The resulting Y has an exponential distribution. As an exercise you should check this assertion by computing $P(Y > y)$. The new Y has 1,000,000 possible values larger than $3\log(10)$ and the largest possible values is now $9\log(10)$. As a result the distribution is much less grainy.

0.0.3 General technique: inverse probability integral transform

One standard technique which is closely connected to our exponential generator is called the inverse probability integral transformation. If Y is to have cdf F we use the following general algorithm:

- Generate $U \sim Uniform[0,1]$.
- Take $Y = F^{-1}(U)$:

$$\begin{aligned} P(Y \leq y) &= P(F^{-1}(U) \leq y) \\ &= P(U \leq F(y)) = F(y) \end{aligned}$$

Jargon: $F^{-1}(U)$ is the inverse probability integral transform. In fact $U = F(Y)$ is called the probability integral transform of Y.

Example: Suppose X has a standard exponential distribution. Then $F(x) = 1 - e^{-x}$ and $F^{-1}(u) = -\log(1-u)$. Compare this generator to our previous method where we used U instead of $1-U$. Of course U and $1-U$ both have Uniform[0,1].

Example: **Normal**: $F = \Phi$ (this is common notation for the standard normal cumulative distribution function). There is no closed form for F^{-1}. One way to generate $N(0,1)$ is to use a numerical algorithm to compute F^{-1}.

An alternative method is the Box Müller generator:

- Generate U_1, U_2, two independent Uniform[0,1] variables.
- Define

$$Y_1 = \sqrt{-2\log(U_1)}\cos(2\pi U_2)$$

and

$$Y_2 = \sqrt{-2\log(U_1)}\sin(2\pi U_2)\,.$$

- As an exercise: use the change of variables technique to prove that Y_1 and Y_2 are independent $N(0,1)$ variables.

0.0.4 Acceptance Rejection

Suppose we can't calculate F^{-1} but know the density f. Find some density g and constant c such that

1. $f(x) \leq cg(x)$ for each x and
2. either G^{-1} is computable or we can generate observations $W_1, W_2, \ldots$ independently from g.

Then we use the following algorithm:

1. Generate W_1.
2. Compute $p = f(W_1)/(cg(W_1)) \leq 1$.
3. Generate a Uniform[0,1] random variable U_1 independent of all Ws.
4. Let $Y = W_1$ if $U_1 \leq p$.
5. Otherwise get new W, U; repeat until you find $U_i \leq f(W_i)/(cg(W_i))$.
6. Make Y be the last W generated.
7. This Y has density f.

0.0.5 Markov Chain Monte Carlo

Recently popular tactic, particularly for generating multivariate observations.

Theorem Suppose $W_1, W_2, \ldots$ is an (ergodic) Markov chain with stationary transitions and the stationary initial distribution of W has density f. Then starting the chain with *any* initial distribution

$$\frac{1}{n}\sum_{i=1}^{n} g(W_i) \to \int g(x)f(x)dx\,.$$

Estimate things like $\int_A f(x)dx$ by computing the fraction of the W_i which land in A.

Many versions of this technique including Gibbs Sampling and Metropolis-Hastings algorithm.

Technique invented in 1950s: Metropolis et al.

One of the authors was Edward Teller "father of the hydrogen bomb".

Importance Sampling

If you want to compute

$$\theta \equiv E(T(X)) = \int T(x)f(x)dx$$

you can generate observations from a different density g and then compute

$$\hat{\theta} = n^{-1}\sum T(X_i)f(X_i)/g(X_i)$$

Then

$$\begin{aligned} E(\hat{\theta}) &= n^{-1} \sum E\left\{T(X_i) f(X_i)/g(X_i)\right\} \\ &= \int \{T(x) f(x)/g(x)\} g(x) dx \\ &= \int T(x) f(x) dx \\ &= \theta \end{aligned}$$

Variance reduction

Example: In this example we simulate to estimate the distribution of the sample mean for a sample from the Cauchy distribution. The Cauchy density is

$$f(x) = \frac{1}{\pi(1+x^2)}$$

The basic algorithm is

1. Generate $U_1, \ldots, U_n$ uniforms.

   The basic algorithm is

2. Define $X_i = \tan^{-1}(\pi(U_i - 1/2))$.

3. Compute $T = \bar{X}$.

4. To estimate $p = P(T > t)$ use

   $$\hat{p} = \sum_{i=1}^{N} 1(T_i > t)/N$$

   after generating N samples of size n.

5. This estimate is unbiased.

6. Its standard error is $\sqrt{p(1-p)/N}$.

The algorithm can be improved by using *antithetic variables*. Note first that $-X_i$ also has a Cauchy distribution. Take $S_i = -T_i$. Remember that S_i has the same distribution as T_i. Try (for $t > 0$)

$$\tilde{p} = [\sum_{i=1}^{N} 1(T_i > t) + \sum_{i=1}^{N} 1(S_i > t)]/(2N)$$

which is the average of two estimates like $\hat{p}$. Then the variance of $\tilde{p}$ is

$$(4N)^{-1}\mathrm{Var}(1(T_i > t) + 1(S_i > t)) \\ = (4N)^{-1}\mathrm{Var}(1(|T| > t))$$

which is

$$\frac{2p(1-2p)}{4N} = \frac{p(1-2p)}{2N}$$

This variance has an extra 2 in the denominator and the numerator is also smaller – particularly for p near $1/2$. So we need only half the sample size to get the same accuracy.

0.0.6 Regression estimates

Suppose $Z \sim N(0,1)$. In this example we consider ways to compute

$$\theta = E(|Z|)\,.$$

To begin with we generate N iid $N(0,1)$ variables $Z_1, \ldots, Z_N$. Compute the basic estimate $\hat{\theta} = \sum |Z_i|/N$. But we know that $E(Z_i^2) = 1$. We also know that $\hat{\theta}$ is positively correlated with $\sum Z_i^2/N$. So we try

$$\tilde{\theta} = \hat{\theta} - c(\sum Z_i^2/N - 1)$$

Notice that $E(\tilde{\theta}) = \theta$ and

$$\mathrm{Var}(\tilde{\theta}) = \\ \mathrm{Var}(\hat{\theta}) - 2c\mathrm{Cov}(\hat{\theta}, \sum Z_i^2/N) \\ + c^2\mathrm{Var}(\sum Z_i^2/N)$$

The value of c which minimizes this is

$$c = \frac{\mathrm{Cov}(\hat{\theta}, \sum Z_i^2/N)}{\mathrm{Var}(\sum Z_i^2/N)}$$

We can estimate c by regressing $|Z_i|$ on Z_i^2! Notice that minimization is bound to produce a smaller variance than just using $c = 0$ which is the original estimate.