Generating Random Variables

Motivation

- We need to generate/simulate random variables in
  - simulating a queueing model if solving the model is too difficult
  - simulating traffic generation, user logon/logoff, component failures, etc., when simulating a networking environment
- The challenge is twofold.
  - to make sure the result is truly random
  - to make sure the result follows a designated distribution

Generating Uniform Random Variables

- Linear Congruential Method (by D. H. Hehmer, 1949)

$$X_{n+1} = (aX_n + c) \text{ mod } m, \quad n \geq 0$$

where $0 \leq X_0 < 0$ is called the *seed* (the starting value), m>0 the modulus, $0 \leq a < m$ the multiplier and $0 \leq c < m$ the increment.

- The selection of m, a, and c values is critical to the quality of produced random values.

- Example, m=10 and $X_0=a=c=2$ produce

$$2, 6, 4, 0, 2, 6, \ldots$$

- The X sequence has at most m values to draw from and must eventually repeat a cycle.
  - this will not be a problem if m is sufficiently large and all possible values are exhausted before repetition.

Theorem. The linear congruential sequence has period length m if and only if

1. c is relatively prime to m
2. a-1 is a multiple of p, for every prime p dividing m
3. a-1 is a multiple of 4, if m is a multiple of 4

Selecting Modulus

- Consider an n-bit computer.
- A natural choice of m is 2^n.
  - this effectively remove divisions from the computation of X division.
  - the problem is that the result X tends to be less random in right-hand digits
  - however, since right-hand digits are insignificant, the resultant random values can be good enough for many applications

- $m = 2^n \pm 1$
  - with these two m values, there are tricks in assembly programming to compute X without division
  - the results are random in all digits
- Set m to the largest prime number smaller than 2^n.
  - the results are random in all digits
  - however, the computation involves real divisions, which are expensive.

Discussions

- The X produced above is a discrete uniform random variable ranging from 0 to m-1.
- To produce a discrete random variable Y from 1 to 100: $Y = (X \mod 100) + 1$
  - for the range from a to b,

$$Y = (X \mod (b - a + 1)) + a$$

- To produce a continuous uniform random variable Z in [0,1): $Z = X/m$.
  - X and m must be converted to floating point before the division

The Inverse Transformation Method

- Let U be a continuous uniform random variable in (0,1).
- Let f(x) be the pdf of a continuous random variable X and

$$F(a) = P\{X \le a\} = \int_{-\infty}^{a} f(x)dx$$

  be the **cumulative distribution function**.
- We can compute X by setting $X = F^{-1}(U)$.
- We will justify this method by showing that the pdf of $F^{-1}(U)$ is also f.

Proof

- Since F is a monotone increasing function,

$$F^{-1}(U) \le a \text{ if and only if } U \le F(a).$$

- We have

$$P\{F^{-1}(U) \le a\} = P\{U \le F(a)\}$$
$$= F(a) \qquad \text{(because } U \text{ is uniform)}$$

- We are done.

CS 756 9

Producing Exponential Random Variables

- Since the pdf is $f(x) = e^{-\lambda x}$, we have $F(x) = 1 - e^{-\lambda x}$

- That is $X = F^{-1}(U) = -\dfrac{\log(1-U)}{\lambda}$

- Actually, since 1-U is also a uniform random variable, it is sufficient to compute

$$-\frac{\log(U)}{\lambda}$$

CS 756 10

Discussion

- The inverse transformation method can be applied to, in theory, all continuous distributions. In practice, however, figuring out $F^{-1}(U)$ can be difficult for some distributions (such as the normal distribution).

CS 756 11

The Rejection Method

- Suppose that we have a method of generating a random variable *Y* whose pdf is *g*(y).
- We wish to generate a random variable *X* with pdf *f*(x) that satisfies

$$\frac{f(x)}{g(x)} \le c, \quad \text{for all } x \ \ (c \text{ is a constant})$$

- We can generate *X* with the following loop:

  Step-1. Generate *Y* and a uniform random variable *U*.

  Step-2. If $U \le f(Y)/(cg(Y))$, set *X*=*Y*; otherwise return to Step-1.

CS 756 12

Proof

- We will show that the last *Y*, which is used to determine the value of *X*, will have the same distribution with *X*.
- Specifically, we want to show they have the same cumulative distribution functions, that is, $P\{Y \le \alpha \mid U \le f(Y)/cg(Y)\} = P\{X \le \alpha\}$

Let $K = P\{U \le f(Y)/cg(Y)\}$. Then,

$$
\begin{aligned}
& P\{Y \le \alpha \mid U \le f(Y)/cg(Y)\} \\
&= \frac{P\{Y \le \alpha, U \le f(Y)/cg(Y)\}}{K} \\
&= \frac{\int P\{Y \le \alpha, U \le f(Y)/cg(Y) \mid Y = y\} g(y)dy}{K} \\
&= \frac{\int_{-\infty}^{\alpha} (f(y)/cg(y))g(y)dy}{K} \\
&= \frac{\int_{-\infty}^{\alpha} f(y)dy}{Kc}
\end{aligned}
$$

- Notice that

$$1 = \lim_{\alpha\to\infty} P\{Y \le \alpha \mid U \le f(Y)/cg(Y)\} = \lim_{\alpha\to\infty} \frac{\int_{-\infty}^{x} f(y)dy}{Kc},$$

which implies *K*=1/c.

- Finally, we have

$$P\{Y \le \alpha \,|\, U \le f(Y)/cg(Y)\} = \int_{-\infty}^{\alpha} f(y)dy = P\{X \le \alpha\}$$

We are done.

- **Remark.** Since each iteration will result in an accepted *Y* with probability

$$P\{U \le f(Y)/cg(Y)\} = 1/c$$

- it follows that the number of iterations is geometric with mean *c*.

Generating Normal Random Variables

- We wish to produce a normal random variable *Z* with mean 0 and variance 1.
- We first focus on the absolute value of *Z*, *X*=|*Z*|, which has the pdf

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2}, \quad 0 < x < \infty$$

- Let *Y* be an exponential random variable with pdf $g(x) = e^{-x}, \quad 0 < x < \infty$.
- Note that

$$\frac{f(x)}{g(x)} = \sqrt{2e/\pi}\, e^{-(x-1)^2/2} \le \sqrt{2e/\pi}$$

- We can now generate X using the rejection method:

  Step-1. Generate Y and a uniform random variable U.

  Step-2. Set $X=Y$ if $U \le e^{-(Y-1)^2/2}$

  Otherwise return to Step-1.
- In average, we will need $\sqrt{2e/\pi} \approx 1.32$ iterations.
- Lastly, set $Z=X$ or $Z=-X$ with equal chances.

Remark. A preliminary version of this method is due to Dr. Von Neumann, the very inventor of stored-instruction computing systems.