

Random-number generators

It is important to be able to efficiently generate independent random variables from the uniform distribution on (0, 1), since:

- Random variables from all other distributions can be obtained by transforming uniform random variables;
- Simulations require many random numbers.

Most random-number generators are of the form:

Start with z_0 (seed) For $n = 1, 2, \ldots$ generate

$$z_n = f(z_{n-1})$$

and

$$u_n = g(z_n)$$

f is the pseudo-random generator g is the output function

 $\{u_0, u_1, \ldots\}$ is the sequence of uniform random numbers on the interval (0, 1).

A 'good' random-number generator should satisfy the following properties:

- Uniformity: The numbers generated appear to be distributed uniformly on (0, 1);
- **Independence:** The numbers generated show no correlation with each other;
- **Replication**: The numbers should be replicable (e.g., for debugging or comparison of different systems).
- Cycle length: It should take long before numbers start te repeat;
- **Speed:** The generator should be fast;
- **Memory usage:** The generator should not require a lot of storage.

Linear (or mixed) congruential generators

Most random-number generators in use today are *linear congruential generators*. They produce a sequence of integers between 0 and m - 1 according to

$$z_n = (az_{n-1} + c) \mod m, \qquad n = 1, 2, \dots$$

a is the multiplier, c the increment and m the modulus. To obtain uniform random numbers on (0,1) we take

$$u_n = z_n/m$$

A good choice of a, c and m is very important.

A linear congruential generator has full period (cycle length is m) if and only if the following conditions hold:

- The only positive integer that exactly divides both m and c is 1;
- If q is a prime number that divides m, then q divides a 1;
- If 4 divides m, then 4 divides a 1.

Multiplicative congruential generators

These generators produce a sequence of integers between 0 and m-1 according to

$$z_n = a z_{n-1} \mod m, \qquad n = 1, 2, \dots$$

So they are linear congruential generators with c = 0.

They cannot have full period, but it is possible to obtain period m - 1 (so each integer 1, ..., m - 1 is obtained exactly once in each cycle) if a and m are chosen carefully. For example, as a = 630360016 and $m = 2^{31} - 1$.

Additive congruential generators

These generators produce integers according to

$$z_n = (z_{n-1} + z_{n-k}) \mod m, \qquad n = 1, 2, \dots$$

where $k \geq 2$. Uniform random numbers can again be obtained from

$$u_n = z_n/m$$

These generators can have a long period upto m^k .

Disadvantage:

Consider the case k = 2 (the *Fibonacci* generator). If we take three consecutive numbers u_{n-2} , u_{n-1} and u_n , then it will never happen that

$$u_{n-2} < u_n < u_{n-1}$$
 or $u_{n-1} < u_n < u_{n-2}$

whereas for true uniform variables both of these orderings occurs with probability I/6.

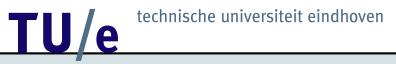
/ department of mathematics and computing science

(Pseudo) Random number generators:

- Linear (or mixed) congruential generators
- Multiplicative congruential generators
- Additive congruential generators

• ...

How random are pseudorandom numbers?



Testing random number generators

Try to test two main properties:

- Uniformity;
- Independence.

Uniformity or goodness-of-fit tests:

Let X_1, \ldots, X_n be n observations. A goodness-of-fit test can be used to test the hyphothesis:

 H_0 : The X_i 's are i.i.d. random variables with distribution function F.

Two goodness-of-fit tests:

- Kolmogorov-Smirnov test
- Chi-Square test



Kolmogorov-Smirnov test

Let $F_n(x)$ be the emperical distribution function, so

$$F_n(x) = \frac{\text{number of } X'_i s \le x}{n}$$

Then

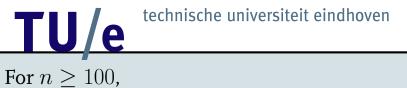
$$D_n = \sup_x |F_n(x) - F(x)|$$

has the Kolmogorov-Smirnov (K-S) distribution. Now we reject H_0 if

 $D_n > d_{n,1-\alpha}$

where $d_{n,1-\alpha}$ is the $1-\alpha$ quantile of the K-S distribution.

Here α is the *significance level* of the test: The probability of rejecting H_0 given that H_0 is true.



$$d_{n,0.95} \approx 1.3581/\sqrt{n}$$

In case of the uniform distribution we have

$$F(x) = x, \qquad 0 \le x \le 1.$$

Chi-Square test

Divide the range of F into k adjacent intervals

$$(a_0, a_1], (a_1, a_2], \dots, (a_{k-1}, a_k]$$

Let

$$N_j =$$
 number of X_i 's in $[a_{j-1}, a_j)$

and let p_j be the probability of an outcome in $(a_{j-1}, a_j]$, so

$$p_j = F(a_j) - F(a_{j-1})$$

Then the test statistic is

$$\chi^{2} = \sum_{j=1}^{k} \frac{(N_{j} - np_{j})^{2}}{np_{j}}$$

If H_0 is true, then np_j is the expected number of the $n X_i$'s that fall in the *j*-th interval, and so we expect χ^2 to be small.

/ department of mathematics and computing science

If H_0 is true, then the distribution of χ^2 converges to a chi-square distribution with k - 1 degrees of freedom as $n \to \infty$.

The chi-square distribution with k - 1 degrees of freedom is the same as the Gamma distribution with parameters (k - 1)/2 and 2.

Hence, we reject H_0 if

$$\chi^2 > \chi^2_{k-1,1-\alpha}$$

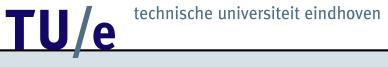
where $\chi^2_{k-1,1-\alpha}$ is the $1-\alpha$ quantile of the chi-square distribution with k-1 degrees of freedom.

Chi-square test for $U(0,1)\ {\rm random}\ {\rm variables}$

We divide (0, 1) into k subintervals of equal length and generate U_1, \ldots, U_n ; it is recommended to choose $k \ge 100$ and $n/k \ge 5$. Let N_j be the number of the $n U_i$'s in the j-th subinterval.

Then

$$\chi^2 = \frac{k}{n} \sum_{j=1}^k \left(N_j - \frac{n}{k} \right)^2$$



Example:

Consider the linear congruential generator

 $z_n = a z_{n-1} \mod m$

with a = 630360016, $m = 2^{31} - 1$ and seed

 $z_0 = 1973272912$

Generating $n = 2^{15} = 32768$ random numbers U_i and dividing (0, 1) in $k = 2^{12} = 4096$ subintervals yields

 $\chi^2 = 4141.0$

Since

 $\chi_{4095,0.9} \approx 4211.4$

we do not reject H_0 at level $\alpha = 0.1$.



Serial test

This is a 2-dimensional version of the chi-square test to test *independence* between successive observations.

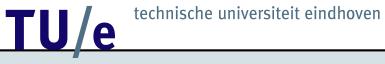
We generate U_1, \ldots, U_{2n} ; if the U_i 's are really i.i.d. U(0, 1), then the non-overlapping pairs

$$(U_1, U_2), (U_3, U_4), \dots, (U_{2n-1}, U_{2n})$$

are i.i.d. random vectors uniformly distributed in the square $(0, 1)^2$.

- Divide the square $(0,1)^2$ into n^2 subsquares;
- Count how many outcomes fall in each subsquare;
- Apply a chi-square test to these data.

This test can be generalized to higher dimensions.



Permutation test

Look at n successive d-tuples of outcomes

$$(U_0, \ldots, U_{d-1}), (U_d, \ldots, U_{2d-1}),$$

 $\ldots, (U_{(n-1)d}, \ldots, U_{nd-1});$

Among the d-tuples there are d! possible orderings and these orderings are equally likely.

- \bullet Determine the frequencies of the different orderings among the $n\ d$ -tuples;
- Apply a chi-square test to these data.



Runs-up test

Divide the sequence U_0, U_1, \ldots in blocks, where each block is a subsequence of *increasing* numbers followed by a number that is *smaller* than its predecessor.

Example: The realization 1,3,8,6,2,0,7,9,5 can be divided in the blocks (1,3,8,6), (2,0), (7,9,5).

A block consisting of j + 1 numbers is called a *run-up of length* j. It holds that

$$P(\text{run-up of length } j) = \frac{1}{j!} - \frac{1}{(j+1)!}$$

- Generate *n* run-ups;
- Count the number of run-ups of length $0, 1, 2, \ldots, k-1$ and $\geq k$;
- Apply a chi-square test to these data.

Correlation test

Generate U_0, U_1, \ldots, U_n and compute an estimate for the (serial) correlation

$$\hat{\rho}_1 = \frac{\sum_{i=1}^n (U_i - \bar{U}(n))(U_{i+1} - \bar{U}(n))}{\sum_{i=1}^n (U_i - \bar{U}(n))^2}$$

where $U_{n+1} = U_1$ and $\overline{U}(n)$ the sample mean.

If the U_i 's are really i.i.d. U(0, 1), then $\hat{\rho}_1$ should be close to zero. Hence we reject H_0 is $\hat{\rho}_1$ is too large.

If H_0 is true, then for large n,

 $P(-2/\sqrt{n} \le \hat{\rho}_1 \le 2/\sqrt{n}) \approx 0.95$

So we reject H_0 at the 5% level if

$$\hat{\rho}_1 \notin (-2/\sqrt{n}, 2/\sqrt{n})$$