Random Number Generation

Outline

1. Properties of Random Numbers
2. Generation of Pseudo-Random Numbers (PRN)
3. Techniques for Generating Random Numbers
4. Tests for Random Numbers

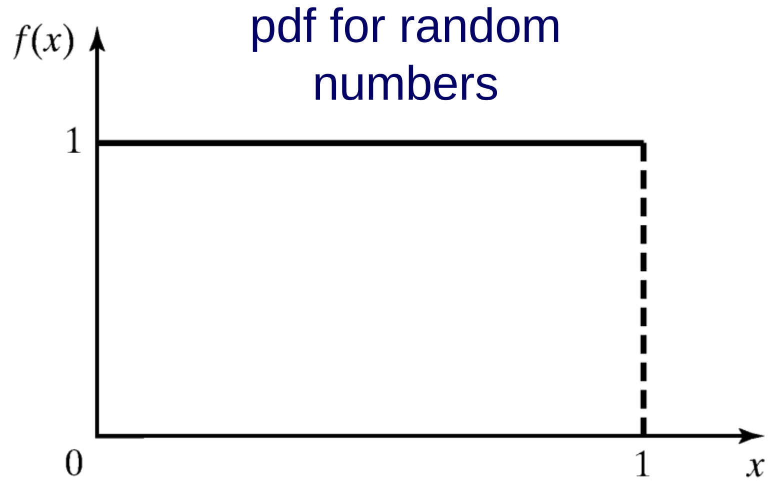
1. Properties of Random Numbers

A sequence of random numbers R_1, R_2, …, must have two important statistical properties:

- Uniformity
- Independence.

Random Number, R_i, must be independently drawn from a uniform distribution with pdf:

$$f(x) = \begin{cases} 1, & 0 \le x \le 1 \\ 0, & \text{otherwise} \end{cases}$$



$$E(R) = \int_0^1 x\,dx = \left.\frac{x^2}{2}\right|_0^1 = \frac{1}{2}$$

$$V(R) = \int_0^1 x^2 dx - [E(R)]^2 = \left.\frac{x^3}{3}\right|_0^1 - \left(\frac{1}{2}\right)^2 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}$$

1. Properties of Random Numbers

Uniformity: If the interval [0,1] is divided into n classes, or subintervals of equal length, the expected number of observations in each interval is N/n, where N is the total number of observations

Independence: The probability of observing a value in a particular interval is independent of the previous value drawn

2. Generation of Pseudo-Random Numbers (PRN)

"Pseudo", because generating numbers using a known method removes the potential for true randomness.

- If the method is known, the set of random numbers can be replicated!!

Goal: To produce a sequence of numbers in [*0,1*] that simulates, or imitates, the ideal properties of random numbers (RN) - *uniform distribution and independence*.

2. Generation of Pseudo-Random Numbers (PRN)

Problems that occur in generation of pseudo-random numbers (PRN)

- Generated numbers might not be uniformly distributed
- Generated numbers might be discrete-valued instead of continuous-valued
- Mean of the generated numbers might be too low or too high
- Variance of the generated numbers might be too low or too high
- There might be dependence (i.e., correlation)

2. Generation of Pseudo-Random Numbers (PRN)

Departure from uniformity and independence for a particular generation scheme can be tested.

If such departures are detected, the generation scheme should be dropped in favor of an acceptable one.

2. Generation of Pseudo-Random Numbers (PRN)

Important considerations in RN routines:

- *The routine should be fast*. Individual computations are inexpensive, but a simulation may require many millions of random numbers
- *Portable to different computers* – ideally to different programming languages. This ensures the program produces same results
- Have sufficiently *long cycle*. The *cycle length,* or *period* represents the length of random number sequence before previous numbers begin to repeat in an earlier order.
- *Replicable*. Given the starting point, it should be possible to generate the same set of random numbers, completely independent of the system that is being simulated
- *Closely approximate the ideal statistical properties* of uniformity and independence.

3. Techniques for Generating Random Numbers

3.1 Linear Congruential Method (LCM).

- Most widely used technique for generating random numbers

3.2 Combined Linear Congruential Generators (CLCG).

- Extension to yield longer period (or cycle)

3.3 Random-Number Streams.

3. Techniques for Generating Random Numbers: Linear Congruential Method

To produce a sequence of integers, X_1, X_2, … between 0 and $m-1$ by following a recursive relationship:

$$X_{i+1} = (aX_i + c) \bmod m, \quad i = 0,1,2,\ldots$$

The multiplier

The increment

The modulus

X_0 is called the *seed*

The selection of the values for a, c, m, and X_0 drastically affects the statistical properties and the cycle length.

If $c \neq 0$ then it is called *mixed congruential* method

When $c=0$ it is called *multiplicative congruential* method

3. Techniques for Generating Random Numbers: Linear Congruential Method

The random integers are being generated in the range [*0,m-1*], and to convert the integers to random numbers:

$$R_i = \frac{X_i}{m}, \quad i = 1,2,...$$

3. Techniques for Generating Random Numbers: Linear Congruential Method

EXAMPLE: Use $X_0 = 27$, $a = 17$, $c = 43$, and $m = 100$.

The X_i and R_i values are:

$X_1 = (17*27+43) \bmod 100 = 502 \bmod 100 = 2,$ $R_1 = 0.02;$

$X_2 = (17*2+43) \bmod 100 = 77 \bmod 100 = 77,$ $R_2 = 0.77;$

$X_3 = (17*77+43) \bmod 100 = 1352 \bmod 100 = 52$ $R_3 = 0.52;$

…

Notice that the numbers generated assume values only from the set $I = \{0, 1/m, 2/m, \ldots, (m-1)/m\}$ because each X_i is an integer in the set $\{0, 1, 2, \ldots, m-1\}$

Thus each Ri is discrete on I, instead of continuous on interval [0,1]

3. Techniques for Generating Random Numbers: Linear Congruential Method

Maximum Density

- Such that the values assumed by R_i, $i = 1,2,\ldots$, leave no large gaps on $[0,1]$
- Problem: Instead of continuous, each R_i is discrete
- Solution: a very large integer for modulus m (e.g., $2^{31}-1$, 2^{48})

Maximum Period

- To achieve maximum density and avoid cycling.
- Achieved by: proper choice of a, c, m, and X_0.

Most digital computers use a binary representation of numbers

- Speed and efficiency are aided by a modulus, m, to be (or close to) a power of 2.

3. Techniques for Generating Random Numbers: Linear Congruential Method

Maximum Period or Cycle Length:

For m a power of 2, say $m=2^b$, and $c \neq 0$, the longest possible period is $P=m=2^b$, which is achieved when c is relatively prime to m (greatest common divisor of c and m is 1) and $a=1+4k$, where k is an integer

For m a power of 2, say $m=2^b$, and $c=0$, the longest possible period is $P=m/4=2^{b-2}$, which is achieved if the seed X_0 is odd and if the multiplier a is given by $a=3+8k$ or $a=5+8k$ for some $k=0,1,\ldots$.

For m a prime number and $c=0$, the longest possible period is $P=m-1$, which is achieved whenever the multiplier a has the property that the smallest integer k such that a^k-1 is divisible by m is $k=m-1$

1. **When m is a power of 2 (e.g., $m = 2^b$) and $c \neq 0$:**
   - **Maximum Period: $P = m = 2^b$**
   - **Conditions for Maximum Period:**
     - c must be relatively prime to m (i.e., $\gcd(c, m) = 1$).
     - The multiplier a must be of the form $a = 1 + 4k$, where k is an integer.

2. **When m is a power of 2 (e.g., $m = 2^b$) and $c = 0$:**
   - **Maximum Period: $P = \frac{m}{4} = 2^{b-2}$**
   - **Conditions for Maximum Period:**
     - The seed X_0 must be odd.
     - The multiplier a must be of the form $a = 3 + 8k$ or $a = 5 + 8k$, where k is a non-negative integer (i.e., $k = 0, 1, 2, \ldots$).

3. **When m is a prime number and $c = 0$:**
   - **Maximum Period:** $P = m - 1$
   - **Conditions for Maximum Period:**
     - The multiplier a must be such that the smallest integer k for which $a^k - 1$ is divisible by m is $k = m - 1$.

These conditions ensure that the sequence generated by the LCM has the longest possible period before repeating, making the random number generation more effective and less predictable.

3. Techniques for Generating Random Numbers : Linear Congruential Method

Example: Using the multiplicative congruential method, find the period of the generator for $a=13$, $m=2^6=64$ and $X_0=1,2,3$ *and* 4

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Xi	1	13	41	21	17	29	57	37	33	45	9	53	49	61	25	5	1
Xi	2	26	18	42	34	58	50	10	2								
Xi	3	39	59	63	51	23	43	47	35	7	27	31	19	55	11	15	3
Xi	4	52	36	20	4												

$m=64$, $c=0$; Maximal period $P=m/4 = 16$ is achieved by using odd seeds X_0=1 and $X_0=3$ ($a=13$ is of the form $5+8k$ with $k=1$)

With $X_0=1$, the generated sequence {1,5,9,13,…,53,57,61} has large gaps

Not a viable generator !! Density insufficient, period too short

3. Techniques for Generating Random Numbers (1): Linear Congruential Method

Example: Speed and efficiency in using the generator on a digital computer is also a factor

Speed and efficiency are aided by using a modulus m either a power of 2 ($=2^b$)or close to it

After the ordinary arithmetic yields a value of aX_i+c, X_{i+1} can be obtained by dropping the leftmost binary digits and then using only the b rightmost digits

3. Techniques for Generating Random Numbers : Linear Congruential Method

Example: $c=0; a=7^5=16807; m=2^{31}-1=2,147,483,647$ *(prime #)*

Period $P=m-1$ (well over 2 billion)

Assume $X_0=123,457$

$X_1=7^5(123457)\mathrm{mod}(2^{31}-1)=2,074,941,799$

$R_1=X_1/2^{31}=0.9662$

$X_2=7^5(2,074,941,799)\ \mathrm{mod}(2^{31}-1)=559,872,160$

$R_2=X_2/2^{31}=0.2607$

$X_3=7^5(559,872,160)\ \mathrm{mod}(2^{31}-1)=1,645,535,613$

$R_3=X_3/2^{31}=0.7662$

……….

Note that the routine divides by $m+1$ instead of m. Effect is negligible for such large values of m.

3. Techniques for Generating Random Numbers: Combined Linear Congruential Generators.

With increased computing power, the complexity of simulated systems is increasing, requiring longer period generator.

- Examples: 1) highly reliable system simulation requiring hundreds of thousands of elementary events to observe a single failure event;
- 2) A computer network with large number of nodes, producing many packets

Approach: Combine two or more *multiplicative congruential generators* in such a way to produce a generator with good statistical properties

3. Techniques for Generating Random Numbers: Combined Linear Congruential Generators

L'Ecuyer suggests how this can be done:

- If $W_{i,1}, W_{i,2}, \ldots, W_{i,k}$ are any independent, discrete valued random variables (not necessarily identically distributed)
- If one of them, say $W_{i,1}$ is uniformly distributed on the integers from 0 to $m_1 - 2$, then

$$W_i = \left(\sum_{j=1}^{k} W_{i,j} \right) \bmod m_1 - 1$$

is uniformly distributed on the integers from 0 to $m_1 - 2$

3. Techniques for Generating Random Numbers: Combined Linear Congruential Generators

Let $X_{i,1}$, $X_{i,2}$, …, $X_{i,k}$, be the i^{th} output from k different multiplicative congruential generators.

- The j^{th} generator:
  - Has prime modulus m_j and multiplier a_j and period is m_j-1
  - Produced integers $X_{i,j}$ is approx ~ Uniform on integers in [1, m_j-1]
  - $W_{i,j} = X_{i,j}$ -1 is approx ~ Uniform on integers in [0, m_j-2]

3. Techniques for Generating Random Numbers: Combined Linear Congruential Generators

Suggested form:

$$X_i = \left(\sum_{j=1}^{k} (-1)^{j-1} X_{i,j} \right) \bmod m_1 - 1 \qquad \text{Hence, } R_i = \begin{cases} \dfrac{X_i}{m_1}, & X_i > 0 \\ \dfrac{m_1 - 1}{m_1}, & X_i = 0 \end{cases}$$

- The maximum possible period for such a generator is:

$$P = \frac{(m_1 - 1)(m_2 - 1)\ldots(m_k - 1)}{2^{k-1}}$$

3. Techniques for Generating Random Numbers: Combined Linear Congruential Generators

Example: For 32-bit computers, L'Ecuyer [1988] suggests combining $k = 2$ generators with $m_1 = 2,147,483,563$, $a_1 = 40,014$, $m_2 = 2,147,483,399$ and $a_2 = 40,692$. The algorithm becomes:

Step 1: Select seeds

– $X_{1,0}$ in the range $[1, 2,147,483,562]$ for the 1st generator

– $X_{2,0}$ in the range $[1, 2,147,483,398]$ for the 2nd generator.

Step 2: For each individual generator,

$$X_{1,j+1} = 40,014\, X_{1,j} \bmod 2,147,483,563$$
$$X_{2,j+1} = 40,692\, X_{1,j} \bmod 2,147,483,399.$$

Step 3: $X_{j+1} = (X_{1,j+1} - X_{2,j+1}) \bmod 2,147,483,562.$

Step 4: Return

$$R_{j+1} = \begin{cases} \dfrac{X_{j+1}}{2,147,483,563}, & X_{j+1} > 0 \\ \dfrac{2,147,483,562}{2,147,483,563}, & X_{j+1} = 0 \end{cases}$$

Step 5: Set $j = j+1$, go back to step 2.

- Combined generator has period: $(m_1 - 1)(m_2 - 1)/2 \sim 2 \times 10^{18}$

3. Techniques for Generating Random Numbers: Random-Numbers Streams

The *seed* for a linear congruential random-number generator:

- Is the integer value X_0 that initializes the random-number sequence.
- Any value in the sequence can be used to "seed" the generator.

A *random-number stream*:

- Refers to a starting seed taken from the sequence X_0, X_1, …, X_P.
- If the streams are b values apart, then stream i could defined by starting seed:

$$S_i = X_{b(i-1)} \text{ for } i = 1, 2, \cdots, \lfloor P/b \rfloor$$

- Older generators: $b = 10^5$; Newer generators: $b = 10^{37}$.

3. Techniques for Generating Random Numbers: Random-Numbers Streams

A single random-number generator with k streams can act like k distinct virtual random-number generators

To compare two or more alternative systems.

- Advantageous to dedicate portions of the pseudo-random number sequence to the same purpose in each of the simulated systems.

4. Tests for Random Numbers: Principles

Desirable properties of random numbers: *Uniformity* and *Independence*

Number of tests can be performed to check these properties been achieved or not

Two type of tests:

- *Frequency Test*: Uses the Kolmogorov-Smirnov or the Chi-square test to compare the distribution of the set of numbers generated to a uniform distribution
- *Autocorrelation test:* Tests the correlation between numbers and compares the sample correlation to the expected correlation.

4. Tests for Random Numbers: Principles

Two categories:

- Testing for uniformity. The hypotheses are:

$$H_0: \quad R_i \sim U[0,1]$$

$$H_1: \quad R_i \nsim U[0,1]$$

  - Failure to reject the null hypothesis, H_0, means that evidence of non-uniformity has not been detected.

- Testing for independence. The hypotheses are:

$$H_0: \quad R_i \nsim \text{independently distributed}$$

$$H_1: \quad R_i \sim \text{independently distributed}$$

  - Failure to reject the null hypothesis, H_0, means that evidence of dependence has not been detected.

4. Tests for Random Numbers: Principles

For each test, a *Level of significance* α must be stated.

The level α , is the probability of rejecting the null hypothesis H_0 when the null hypothesis is true:

$$\alpha = P(\text{reject } H_0 \mid H_0 \text{ is true})$$

The decision maker sets the value of α for any test

Frequently α is set to 0.01 or 0.05

4. Tests for Random Numbers: Principles

When to use these tests:

- If a well-known simulation languages or random-number generators is used, it is probably unnecessary to test
- If the generator is not explicitly known or documented, e.g., spreadsheet programs, symbolic/numerical calculators, tests should be applied to many sample numbers.

Types of tests:

- *Theoretical tests*: evaluate the choices of m, a, and c without actually generating any numbers
- *Empirical tests*: applied to actual sequences of numbers produced. *Our emphasis*.

4. Tests for Random Numbers: Frequency Tests

Test of uniformity

Two different methods:

- Kolmogorov-Smirnov test
- Chi-square test

Both these tests measure the degree of agreement between the distribution of a sample of generated random numbers and the theoretical uniform distribution

Both tests are based on null hypothesis of no significant difference between the sample distribution and the theoretical distribution

4. Tests for Random Numbers: Frequency Tests Kolmogorov-Smirnov Test

Compares the continuous cdf, $F(x)$, of the uniform distribution with the empirical cdf, $S_N(x)$, of the N sample observations.

- We know: $F(x) = x, \quad 0 \le x \le 1$
- If the sample from the RN generator is $R_1, R_2, \ldots, R_N$, then the empirical cdf, $S_N(x)$ is:

$$S_N(x) = \frac{\text{number of } R_1, R_2, \ldots, R_n \text{ which are} \le x}{N}$$

The cdf of an empirical distribution is a step function with jumps at each observed value.

4. Tests for Random Numbers: Frequency Tests Kolmogorov-Smirnov Test

Test is based on the largest absolute deviation statistic between $F(x)$ *and* $S_N(x)$ over the range of the random variable:

$$D = max \mid F(x) - S_N(x) \mid$$

The distribution of D is known and tabulated (A.8) as function of N

Steps:

1. Rank the data from smallest to largest. Let $R_{(i)}$ denote i^{th} smallest observation, so that $R_{(1)} \leq R_{(2)} \leq \ldots \leq R_{(N)}$
2. Compute $$D^{+} = \max_{1 \leq i \leq N} \left\{ \frac{i}{N} - R_{(i)} \right\}; \quad D^{-} = \max_{1 \leq i \leq N} \left\{ R_{(i)} - \frac{i-1}{N} \right\}$$
3. Compute $D = max(D^{+}, D^{-})$
4. Locate in Table A.8 the *critical value* $D\alpha$, for the specified significance level α and the sample size N
5. If the sample statistic D is greater than the critical value $D\alpha$, the null hypothesis is rejected. If $D \leq D\alpha$, conclude there is no difference

4. Tests for Random Numbers: Frequency Tests Kolmogorov-Smirnov Test

Example: Suppose *5* generated numbers are *0.44, 0.81, 0.14, 0.05, 0.93.*

Step 1:

$R_{(i)}$	0.05	0.14	0.44	0.81	0.93
i/N	0.20	0.40	0.60	0.80	1.00
$i/N - R_{(i)}$	0.15	0.26	0.16	-	0.07
$R_{(i)} - (i\text{-}1)/N$	0.05	-	0.04	0.21	0.13

Step 2: (rows 3–4)

Arrange $R_{(i)}$ from smallest to largest

$D^+ = max\ \{i/N - R_{(i)}\}$

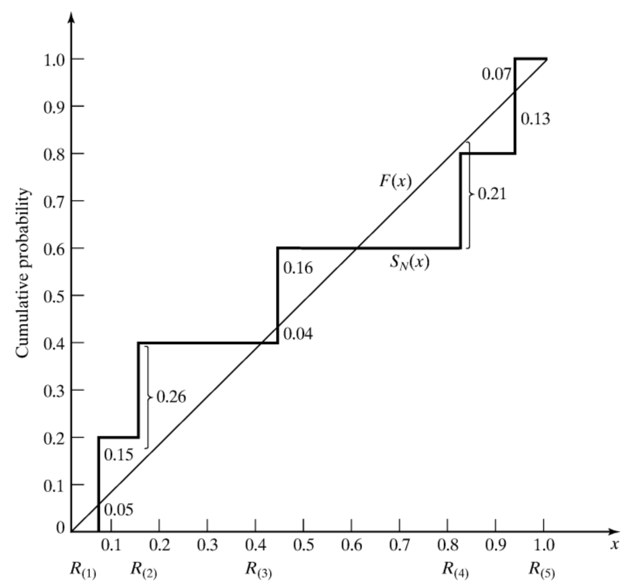
$D^- = max\ \{R_{(i)} - (i\text{-}1)/N\}$

Step 3: $D = max(D^+, D^-) = 0.26$

Step 4: For $\alpha = 0.05$,

$D_\alpha = 0.565 > D$

Hence, H_0 is not rejected.



4. Tests for Random Numbers: Frequency Tests - Chi-Square Test

Chi-square test uses the sample statistic:

$$\chi_0^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

n is the # of classes

E_i is the expected # in the i^{th} class

O_i is the observed # in the i^{th} class

- Approximately the chi-square distribution with *n-1* degrees of freedom (where the critical values are tabulated in Table A.6)
- For the uniform distribution, E_i, the expected number in the each class is:

$$E_i = \frac{N}{n}, \quad \text{where N is the total \# of observation}$$

Valid only for large samples, e.g. N >= 50

Reject H_0 if $\chi_0^2 > \chi_{\alpha,N-1}^2$

4. Tests for Random Numbers : Frequency Tests - Chi-Square Test

Example : Use Chi-square test for the data shown below with α=0.05. The test uses *n=10* intervals of equal length, namely [0,0.1),[0.1,0.2), …., [0.9,1.0)

0.34	0.90	0.25	0.89	0.87	0.44	0.12	0.21	0.46	0.67
0.83	0.76	0.79	0.64	0.70	0.81	0.94	0.74	0.22	0.74
0.96	0.99	0.77	0.67	0.56	0.41	0.52	0.73	0.99	0.02
0.47	0.30	0.17	0.82	0.56	0.05	0.45	0.31	0.78	0.05
0.79	0.71	0.23	0.19	0.82	0.93	0.65	0.37	0.39	0.42
0.99	0.17	0.99	0.46	0.05	0.66	0.10	0.42	0.18	0.49
0.37	0.51	0.54	0.01	0.81	0.28	0.69	0.34	0.75	0.49
0.72	0.43	0.56	0.97	0.30	0.94	0.96	0.58	0.73	0.05
0.06	0.39	0.84	0.24	0.40	0.64	0.40	0.19	0.79	0.62
0.18	0.26	0.97	0.88	0.64	0.47	0.60	0.11	0.29	0.78

4. Tests for Random Numbers: Frequency Tests Chi-Square Test

The value of χ_0^2=3.4; The critical value from table A.6 is $\chi_{0.05,9}^2$=16.9. Therefore the null hypothesis is not rejected

Table 7.3 Computations for Chi-Square Test

Interval	O_i	E_i	$O_i - E_i$	$(O_i - E_i)^2$	$\frac{(O_i - E_i)^2}{E_i}$
1	8	10	−2	4	0.4
2	8	10	−2	4	0.4
3	10	10	0	0	0.0
4	9	10	−1	1	0.1
5	12	10	2	4	0.4
6	8	10	−2	4	0.4
7	10	10	0	0	0.0
8	14	10	4	16	1.6
9	10	10	0	0	0.0
10	11	10	1	1	0.1
	100	100	0		3.4

4. Tests for Random Numbers: Tests for Autocorrelation

The test for autocorrelation are concerned with the dependence between numbers in a sequence.

Consider:

0.12	0.01	0.23	0.28	0.89	0.31	0.64	0.28	0.83	0.93
0.99	0.15	0.33	0.35	0.91	0.41	0.60	0.27	0.75	0.88
0.68	0.49	0.05	0.43	0.95	0.58	0.19	0.36	0.69	0.87

Though numbers seem to be random, every fifth number is a large number in that position.

This may be a small sample size, but the notion is that numbers in the sequence might be related

4. Tests for Random Numbers: Tests for Autocorrelation

Testing the autocorrelation between every m numbers (m is a.k.a. the lag), starting with the i^{th} number

- The autocorrelation ρ_{im} between numbers: R_i, R_{i+m}, R_{i+2m}, $R_{i+(M+1)m}$
- M is the largest integer such that $i+(M+1)m \leq N$

Hypothesis:

$$H_0: \quad \rho_{im} = 0, \quad \text{if numbers are independent}$$

$$H_1: \quad \rho_{im} \neq 0, \quad \text{if numbers are dependent}$$

If the values are uncorrelated:

- For large values of M, the distribution of the estimator of ρ_{im}, denoted $\hat{\rho}_{im}$ is approximately normal.

4. Tests for Random Numbers : Tests for Autocorrelation

Test statistics is: $$Z_0 = \frac{\hat{\rho}_{im}}{\hat{\sigma}_{\hat{\rho}_{im}}}$$

- Z_0 is distributed normally with mean = *0* and variance = *1*, and:

$$\hat{\rho}_{im} = \frac{1}{M+1}\left[\sum_{k=0}^{M} R_{i+km} R_{i+(k+1)m}\right] - 0.25$$

$$\hat{\sigma}_{\rho_{im}} = \frac{\sqrt{13M+7}}{12(M+1)}$$

If $\rho_{im} > 0$, the subsequence has positive autocorrelation

- High random numbers tend to be followed by high ones, and vice versa.

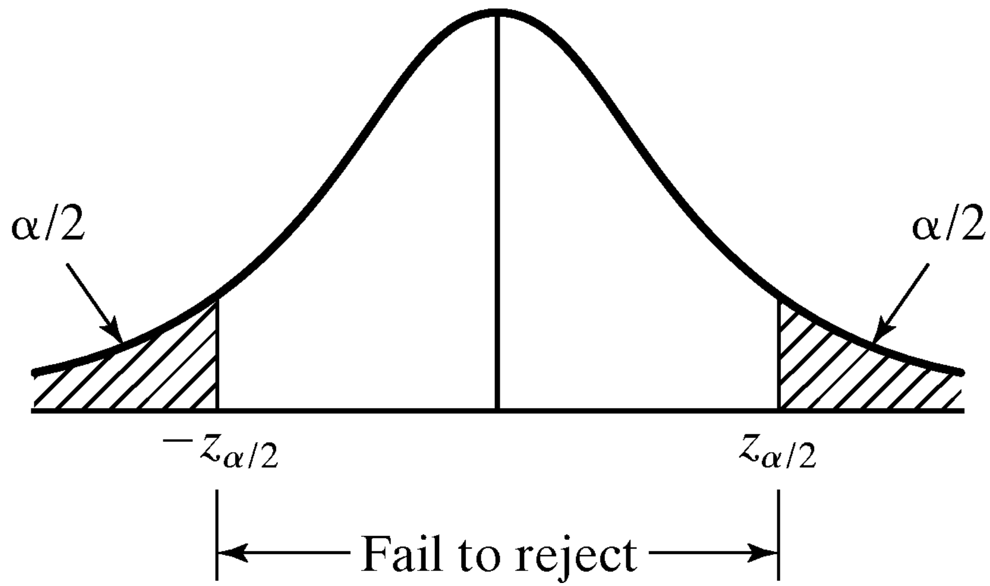
If $\rho_{im} < 0$, the subsequence has negative autocorrelation

- Low random numbers tend to be followed by high ones, and vice versa.

4. Tests for Random Numbers: Tests for Autocorrelation

After computing Z_0, do not reject the hypothesis of independence if $-z_{\alpha/2} \leq Z_0 \leq z_{\alpha/2}$

α is the level of significance and $z_{\alpha/2}$ is obtained from table A.3



4. Tests for Random Numbers: Tests for Autocorrelation

Example: Test whether the *3rd*, *8th*, *13th*, and so on, for the output on Slide 37 are auto-correlated or not.

- Hence, $\alpha = 0.05$, $i = 3$, $m = 5$, $N = 30$, and M = 4. M is the largest integer such that $3+(M+1)5 \leq 30$.

$$\hat{\rho}_{35} = \frac{1}{4+1}\begin{bmatrix}(0.23)(0.28)+(0.25)(0.33)+(0.33)(0.27) \\ +(0.28)(0.05)+(0.05)(0.36)\end{bmatrix} - 0.25$$

$$= -0.1945$$

$$\hat{\sigma}_{\rho_{35}} = \frac{\sqrt{13(4)+7}}{12(4+1)} = 0.128$$

$$Z_0 = -\frac{0.1945}{0.1280} = -1.516$$

- From Table A.3, $z_{0.025} = 1.96$. Hence, the hypothesis is not rejected.

4. Tests for Random Numbers: Tests for Autocorrelation

Shortcoming:

The test is not very sensitive for small values of M, particularly when the numbers being tested are on the low side.

Problem when "fishing" for autocorrelation by performing numerous tests:

- If $\alpha = 0.05$, there is a probability of 0.05 of rejecting a true hypothesis.
- If 10 independent sequences are examined,
  - The probability of finding no significant autocorrelation, by chance alone, is $0.95^{10} = 0.60$.
  - Hence, the probability of detecting significant autocorrelation when it does not exist = *40%*