Chapter Two
Random Number

Ermias Tesfaye (PhD)
Assistant Professor of Industrial Engineering
School of Mechanical and Industrial Engineering
Addis Ababa Institute of Technology
Addis Ababa University
Email: ermiastes@gmail.com

Outline

- ✓Introduction to System, Modeling and Simulation
- ✓Types of Models
- ✓Simulation
- ✓Advantage and Disadvantage of Simulation
- ✓Types of Simulations

Introduction

- Random numbers are a necessary basic ingredient in the simulation of almost all discrete systems.
- Most computer languages have a subroutine, object, or function that will generate a random number.
- Similarly simulation languages generate random numbers that are used to generate event limes and other random variables.

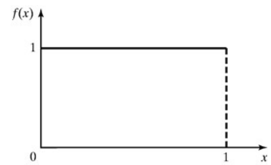
Introduction

- ***Properties of Random Numbers***
- A sequence of random numbers, W1, W2, .. , must have two important statistical properties,
  - Uniformity and independence
  - Should be able to reproduce a given sequence of random numbers
    - Helps program debugging
    - Helpful when comparing alternative system design
  - Should have provision to generate several streams of random numbers
  - Computationally efficient

Introduction

- ***Properties of Random Numbers***
- A sequence of random numbers, W1, W2, .. , must have two important statistical properties,
  - uniformity and independence.
- Each random number Ri, is an independent sample drawn from a continuous uniform distribution between zero and 1. That is, the pdf is given by:

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



Introduction

- ***Generation of Pseudo – Random Numbers***
- Pseudo means false, so false random numbers are being generated.
- The goal of any generation scheme, is to produce a sequence of numbers between 0 and 1 which simulates, or initiates, the ideal properties of uniform distribution and independence as closely as possible.
- When generating pseudo-random numbers, certain problems or errors can occur.
- These errors, or departures from ideal randomness, are all related to the properties stated previously.

Introduction

- ***Generation of Pseudo – Random Numbers***
- Some examples include the following:
  - ➢The generated numbers may not be uniformly distributed.
  - ➢The generated numbers may be discrete -valued instead continuous valued.
  - ➢The mean of the generated numbers may be too high or too low.
  - ➢The variance of the generated numbers may be too high or low.
  - ➢There may be dependence. The following are examples:
    - ✓Autocorrelation between numbers.
    - ✓Numbers successively higher or lower than adjacent numbers.
    - ✓Several numbers above the mean followed by several numbers below the mean.

Introduction

- ***Generation of Pseudo – Random Numbers***
- Usually, pseudo-random numbers are generated by a computer as part of the simulation. Numerous methods are available. In selecting a routine, there are a number of important considerations.
  - The routine should be fast.
  - The routine should be platform independent and portable between different programming languages.
  - The routine should have a sufficiently long cycle (much longer than the required number of samples).
  - The random numbers should be replicable. Useful for debugging and variance reduction techniques.
  - Most importantly, the routine should closely approximate the ideal statistical properties.

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***
- Produces a sequence of integers, X1, X2,... between zero and m — 1 according to the following recursive relationship:
  - $\mathbf{X_{i+1} = (a\,X_i + c) \bmod m}$ i = 0,1, 2,...
- The initial value X0 is called the seed, a is called the constant multiplier, c is the increment, and m is the modulus.
  - If $c \neq 0$ in above equation, the form is called the ***mixed congruential method***.
  - When c = 0, the form is known as the ***multiplicative congruential method***. The selection of the values for a, c, m and Xo drastically affects the statistical properties and the cycle length.
    - Eg 1: X_0=27, a=17, c=43 and m=100
- Here integer values will be between 0 and 99 because of the modulus m. Note that random integers are being generated rather than random numbers.

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***
- These integers should be uniformly distributed. Convert to numbers in [0,1] by normalizing with modulus m:
  - Ri=Xi/m
- Of primary importance is uniformity and statistical independence. Of secondary importance is maximum density and maximum period within the sequence:
  - R_i, i=1,2,....
- Note that, the sequence can only take values in:
  - {0,1/m,2/m,(m-1)/m,1}
- Thus R_i is discrete rather than continuous.
- This is easy to fix by choosing large modulus m.
- Values such as m=2^{31}-1 and m=2^{48} are in common use in generators appearing in many simulation languages).
- Maximum Density and Maximum period can be achieved by the proper choice of a, c, m and X_0.

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***

$$x_n = 5x_{n-1} + 1 \mod 16$$

- Starting with x0 = 5:

$$x_1 = 5(5) + 1 \mod 16 = 26 \mod 16 = 10$$

- The first 32 numbers obtained by the above procedure
  10, 3, 0, 1, 6, 15, 12, 13, 2, 11, 8, 9, 14, 7, 4, 5, 10, 3, 0, 1, 6, 15, 12, 13, 2, 11, 8, 9, 14, 7, 4, 5.
- By dividing x's by 16:
  0.6250, 0.1875, 0.0000, 0.0625, 0.3750, 0.9375, 0.7500, 0.8125,
  0.1250, 0.6875, 0.5000, 0.5625, 0.8750, 0.4375, 0.2500, 0.3125,
  0.6250, 0.1875, 0.0000, 0.0625, 0.3750, 0.9375, 0.7500, 0.8125,
  0.1250, 0.6875, 0.5000, 0.5625, 0.8750, 0.4375, 0.2500, 0.3125.

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***

Properties

- Can have at most m distinct integers in the sequence
  - As soon as any number in the sequence is repeated, the whole sequence is repeated
  - Period: number of distinct integers generated before repetition occurs
- Problem: Instead of continuous, the X_i's can only take on discrete values 0, 1/m, 2/m,..., (m-1)/m
  - Solution: m should be selected to be very large in order to achieve the effect of a continuous distribution (typically, m > 109)
  - Approximation appears to be of little consequence

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***

Characteristics of Good Generator

- Maximum Density
  - Such that the values assumed by x_i, i=1,2,… leave no large gaps on [0,1]
- Maximum Period
  - To achieve maximum density and avoid cycling
  - Achieve 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

Techniques for generating Random Number

- ***Linear Congruential Generator (LCG)***
- A currently popular multiplicative LCG is:

$$x_n = 7^5 x_{n-1} \bmod (2^{31} - 1)$$

- 2^{31}-1 is a prime number and 75 is a primitive root of it
  → Full period of 231-2.
- This generator has been extensively analyzed and shown to be good

See the following book for advanced RNGs:

Numerical Recipes: The Art of Scientific Computing
http://www.nr.com/

Techniques for generating Random Number

- ***Multiplicative Congruential Generator (LCG)***
- Basic Relationship:
  - $X_{i+1} = a\, X_i$ (mod m), where a ≥ 0 and m ≥ 0
- Most natural choice for m is one that equals to the capacity of a computer word. m = 2b (binary machine), where b is the number of bits in the computer word.
- m = 10d (decimal machine), where d is the number of digits in the computer word

Techniques for generating Random Number

- ***Multiplicative Congruential Generator (LCG)***
- **EXAMPLE 1:** Let m = 102 = 100, a = 19, c = 0, and X0 = 63, and generate a sequence c random
- integers using Equation
  - Xi+1 = (aXi + c) mod m, i = 0, 1, 2....
  - X0 = 63; X1 = (19)(63) mod 100 = 1197 mod 100 = 97
  - X2 = (19) (97) mod 100 = 1843 mod 100 = 43
  - X3 = (19) (43) mod 100 = 817 mod 100 = 17 . . . .
- When m is a power of 10, say m = 10b, the modulo operation is accomplished by saving the b rightmost (decimal) digits.

Techniques for generating Random Number

- ***Combined Linear Congruential Generators***
- As computing power has increased, the complexity of the systems that we are able to simulate has also increased.
- One fruitful approach is to combine two or more multiplicative congruential generators in such a way that the combined generator has good statistical properties and a longer period.
- The following result from L'Ecuyer [1988] suggests how this can be done:
  - If $W_{i,1}$, $W_{i,2}$,... , $W_{i,k}$ are any independent, discrete-valued random variables (not necessarily identically distributed), but one of them, say $W_{i,1}$, is uniformly distributed on the integers 0 to mi — 2, then

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

  - is uniformly distributed on the integers 0 to m_i — 2.

Techniques for generating Random Number

- ***Combined Linear Congruential Generators***
- To see how this result can be used to form combined generators,
- Let $X_{i,1}$, $X_{i,2}$,..., $X_{i,k}$ be the i th output from k different multiplicative congruential generators, where the j^{th} generator has prime modulus m_j, and the multiplier a_j is chosen so that the period is $m_j - 1$.
- Then the j'th generator is producing integers $X_{i,j}$ that are approximately uniformly distributed on 1 to m_j - 1, and $W_{i,j} = X_{i,j}$ — 1 is approximately uniformly distributed on 0 to m_j - 2. L'Ecuyer [1988] therefore suggests combined generators of the form

$$X_i = \left[\sum_{j=1}^{k}(-1)^{j-1}W_{i,j}\right] mod m_1 - 1$$

$$R_i = \begin{cases} \dfrac{X_i}{m_1}, X_i > 0 \\ \dfrac{m_1 - 1}{m_1}, X_i = 0 \end{cases}$$

Techniques for generating Random Number

- ***Combined Linear Congruential Generators***
- Notice that the "$(-1)^{j-1}$" coefficient implicitly performs the subtraction $X_{i,1}$-1; for example, if k=2, then
  - (-1)°(X i 1 - 1) - (- l) l (X i 2 - 1)=∑2j=1(-1)j-1 X i,j
- The maximum possible period for such a generator is

$$\frac{(m_1 - 1)(m_2 - 1) - - - (mk - 1)}{2^{k-1}}$$

which is achieved by the following generator:

Tests for Random Numbers

- To insure the desirable properties of random numbers (uniformity and independence) are achieved, a number of tests can be performed. (fortunately, the appropriate tests have already been conducted for most commercial simulation software).
- The tests can be placed in two categories according to the properties of interest. The first entry in the list below concerns testing for uniformity. The second through fifth entries concern testing for independence. The five types of tests are:

Tests for Random Numbers

1. **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.
2. **Runs test.** Tests the runs up and down or the runs above, and below the mean by comparing the actual values to expected values. The statistics for comparison is the chi-square.
3. **Autocorrelation test** Tests the correlation between numbers and compares the sample correlation to the expected correlation of zero.
4. **Gap test.** Counts the number of digits that appear between repetitions of particular digit and then uses the Kolmogorov-Smirnov test to compare with the expected size of gaps,
5. **Poker test** Treats numbers grouped together as a poker hand. Then the hands obtained are compared to what is expected using the chi-square test.

Tests for Random Numbers

Frequency Tests

- A basic test that should always be performed to validate a new generator is the test of uniformity.
- Two different methods of testing are available. They are the **Kolmogorov-Smirnov** and the **chi-square** test. Both of 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 the null hypothesis of no significant difference between the sample distribution and the theoretical distribution.

Tests for Random Numbers

1. The Kolmogorov-Smirnov test.

- This test compares the continuous cdf, F(X), of the uniform distribution to the empirical cdf, SN(x), of the sample of N observations. By definition,
  - $F(x) = x,\ 0 \le x \le 1$
- If the sample from the random-number generator is $R_1, R_2,\ldots,R_N$, then the empirical cdf, SN(x), is defined by
- Type equation here.

$$S_N(X) = \frac{\text{number of R1 R2, ,}\cdot\cdot\cdot\text{, Rn which are <= x}}{N}$$

- As N becomes larger, $S_N(X)$ should become a better approximation to F(X), provided that the null hypothesis is true.

Tests for Random Numbers

1. The Kolmogorov-Smirnov test.

- The Kolmogorov-Smirnov test is based on the largest absolute deviation between F(x) and $S_N(X)$ over the range of the random variable. That is it is based on the statistics
  - $D = \max \mid F(x) - S_N(x) \mid$
- For testing against a uniform cdf, the test procedure follows these steps:
- **Step 1.** Rank the data from smallest to largest. Let $R_{(i)}$ denote the i^{th} smallest observation, so that
  - $R_{(1)} \le R_{(2)} \le \cdot\cdot\cdot \le R_{(N)}$
- **Step 2.** Compute
  - $D^+ = \max \{ \frac{i}{N} - R_{(i)} \};\qquad D^- = \max \{R_{(i)} - \frac{i-1}{N}\}$

Tests for Random Numbers

1. The Kolmogorov-Smirnov test.

- **Step 3:** Compute D = max (D+, D-).
- **Step 4:** Determine the critical value, $D\alpha$, for the specified significance level α and the given sample size N.
- If the sample statistics D is greater than the critical value, $D\alpha$, the null hypothesis that the data are a sample from a uniform distribution is rejected.
- If $D \leq D\alpha$, conclude that no difference has been detected between the true distribution of $\{R_1, R_2, ,........, R_n\}$ and the uniform distribution.

Tests for Random Numbers

2. The Chi-Square test.

- The chi-square test uses the sample statistics
- $X_0^2 = \sum_{i=1}^{n} \frac{(O_i - Ei)^2}{E_i}$
- Where O_i is the observed number in the i^{th} class, E_i is the expected number in the i^{th} class, and n is the number of classes.

1. Determine Order Statistics
   $R_{(1)} <= R_{(2)} <= .... <= R_{(N)}$
2. Divided Range $R_{(N)}$-$R_{(1)}$ in n equidistant intervals $[a_i, b_i]$, such that each interval has at least 5 observations.
3. Calculate for i=1,…..,N.
   $O_i = N.\{S_N(b_i) - SN(ai)\}, Ei = N.\{F(b_i) - F(ai)\}$

Tests for Random Numbers

2. The Chi-Square test.

- Calculate
- $X_0^2 = \sum_{i=1}^{n} \frac{(O_i - Ei)^2}{E_i}$
- Where, O_i is observed number in the i^{th} class E_i is expected number in the i^{th} class,
- $Ei = \frac{N}{n}$, Where N- No. of Observation; n – No. of Classes
- Determine for significant level, , X_{n-1}^2
  - $\begin{cases} X_0^2 \leq X_{\alpha,n-1}^2 \; accept: No\ difference\ between\ SN(x)\ and\ F(x) \\ X_0^2 > X_{\alpha,n-1}^2 \; Reject: Difference\ between\ SN(x)\ and\ F(x) \end{cases}$

Tests for Random Numbers

3. Auto Correlation test.

- The tests for auto-correlation are concerned with the dependence between numbers in a sequence. The list of the 30 numbers appears to have the effect that every 5^{th} number has a very large value. If this is a regular pattern, we can't really say the sequence is random.
- The test computes the auto-correlation between every m numbers (m is also known as the lag) starting with the i^{th} number. Thus the autocorrelation ρ_{im} between the following numbers would be of interest.
  - Where, ρ_{im} is between no.s R_i, R_{i+m}, R_{i+2m},…, $R_{i+(M+1)m}$
  - M is the largest integer; $i+(M+1)m \leq N$

Tests for Random Numbers

3. Auto Correlation test.

- The test statistics $Zo = \frac{\rho_{im}}{\sigma_{\rho_{im}}}$, which is distributed normally with a mean of zero and variance of one.
- The actual formula for ρ_{im} and the standard deviation is
- $\rho_{im} = \frac{1}{M+1}\left[\sum_{k=0}^{M} R_{i+km} R_{i+(k+1)m}\right] - 0.25$

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

- After computing Zo, do not reject the null hypothesis of independence if
- $-Z_{\alpha/2} \le Z_0 \le Z_{\alpha/2}$ Where ∝ is the level of significance

Random Variate

(Pseudo-) Random Number Generation (RNG)

- A fundamental primitive required for simulations
- Goal: Uniform(0,1)
  - Uniformity
  - Independence
- Computational efficiency
- Long period
- Multiple streams
- Common approach: LCG
- Careful design and seeding
- Never generates 0.0 or 1.0

Random Variate Generator

- Builds upon Uniform(0,1)
- Goal: any distribution
  - Discrete distributions
  - Continuous distributions
- Independence (usually)
- Correlation (if desired)
- Computational efficiency
- Common approach: the inverse transform method
- Straightforward math (usually)
- Might generate 0.0 or 1.0

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**
- **ACCEPTANCE-REJECTION TECHNIQUE**

- All these techniques assume that a source of uniform (0,1) random numbers is available R_1, R_2….. Where each R_1 has probability density function and cumulative distribution function.

Note: The random variable may be either discrete or continuous.

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- The inverse transform technique can be used to sample from exponential, the uniform, the Weibull, and the triangular distributions and empirical distributions.
- Additionally, it is the underlying principle for sampling from a wide variety of discrete distributions. The technique will be explained in detail for the exponential distribution and then applied to other distributions.
- It is the most straightforward, but always the most efficient., technique computationally.

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- Suppose that we wish to generate a random variate X that is continuous and has distribution function F that is continuous and strictly increasing when $0<F(x)<1$. [This means that if $x1<x2$ and $0<F(x_1) \leq F(x_2)$, 1, then in fact $F(x_1)<F(x_2)$.]
- Let F^{-1} denote the inverse of the function F. Then an algorithm for
- generating a random variate X having distribution function F is as follows (recall that ~ is read "is distributed as"):
  1. Generate $U \sim U(0, 1)$.
  2. Return $X = F^{-1}(U)$.
- Note that $F^{-1}(U)$ will always be defined, since $0 \leq U \leq 1$ and the range of F is [0, 1].

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- To show that the value X returned by the above algorithm, called the general inverse-transform method, has the desired distribution F, we must show that for any real number x, $P(X \leq x) = F(x)$. Since F is invertible, we have
- $P(X \leq x) = P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x)$
- where the last equality follows since $U \sim U(0, 1)$ and $0 \leq F(x) \leq 1$.

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**
- **Advantages and Disadvantages**

- The first is to facilitate variance-reduction techniques that rely on inducing correlation between random variates;
- examples of such techniques are common random numbers and antithetic variates. If F_1 and F_2 are two distribution functions, then $X_1 = F_1^{-1}(U_1)$ and $X_2 = F_2^{-1}(U2)$ will be random variates with respective distribution functions F_1 and F_2, where U_1 and U_2 are random numbers.
- If U_1 and U_2 are independent, then of course X_1 and X_2 will be independent as well. However, if we let $U_2 = U_1$, then the correlation between X_1 and X_2 is made as positive as possible, and taking $U_2 = 1 - U_1$ (which, recall, is also distributed uniformly over [0, 1]) makes the correlation between X_1 and X_2 as negative as possible.

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**
- **Advantages and Disadvantages**

- Thus, the inverse-transform method induces the strongest correlation (of either sign) between the generated random variates, which we hope will propagate through the simulation model to induce the strongest possible correlation in the output, thereby contributing to the success of the variance-reduction technique.
- On a more pragmatic level, inverse transform eases application of variance-reduction techniques since we always need exactly one random number to produce one value of the desired X.

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**
- **Advantages and Disadvantages**

- The second advantage concerns ease of generating from truncated distributions. In the continuous case, suppose that we have a density f with corresponding distribution function F. For a , b (with the possibility that a = -∞ or b = +∞), we define the truncated density

$$f^*(x) = \begin{cases} \dfrac{f(x)}{F(b) - F(a)} & \text{if } a \le x \le b \\ 0 & \text{otherwise} \end{cases}$$

which has corresponding *truncated distribution function*

$$F^*(x) = \begin{cases} 0 & \text{if } x < a \\ \dfrac{F(x) - F(a)}{F(b) - F(a)} & \text{if } a \le x \le b \\ 1 & \text{if } b < x \end{cases}$$

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**
- **Advantages and Disadvantages**

- One possible impediment to use of this method in the continuous case is the need to evaluate $F^{-1}(U)$. Since we might not be able to write a formula for F^{-1} in closed form for the desired distribution (e.g., the normal and gamma distributions), simple use of the method, might not be possible.
- A second potential disadvantage is that for a given distribution the inverse-transform method may not be the fastest way to generate the corresponding random variate;

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- Example: Let X have the exponential distribution with mean β

$$F(x) = \begin{cases} 1 - e^{-x/\beta} & \text{if } x \ge 0 \\ 0 & \text{otherwise} \end{cases}$$

so to find F^{-1}, we set $u = F(x)$ and solve for x to obtain

$$F^{-1}(u) = -\beta \ln(1 - u)$$

- Thus, to generate the desired random variate, we first generate a U~U(0,1) and then let **X= -β ln (1-U).**

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- **Step 1.** Compute the cdf of the desired random variable X. For the exponential distribution, the cdf is F(x) = 1 — e , x > 0.
- **Step 2.** Set F(X) = R on the range of X. For the exponential distribution, it becomes $1 - e^{-\lambda X}$ = U on the range x >=0. Since X is a random variable (with the exponential distribution in this case), it follows that 1 - is also a random variable, here called U. As will be shown later, U has a uniform distribution over the interval (0,1).,
- **Step 3.** Solve the equation F(X) = U for X in terms of U. For the exponential distribution, the solution proceeds as follows:

$1 - e^{-\lambda x} = R$

$e^{-\lambda x} = 1 - R$

$-\lambda X = \ln(1 - R)$

$x = -1/\lambda \ln(1 - R)$

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

- Step 4. Generate (as needed) uniform random numbers R1, R2, R3,... and compute the desired random variates by

  $X_i = F^{-1}(R_i)$ (1)
- For the exponential case, F (R) = (-1/λ)ln(1- R) by Equation (1), so that

  $X_i = -1/\lambda \ln(1 - R_i)$ (2)
- for i = 1,2,3,.... One simplification that is usually employed in Equation (2) is to replace $1 - R_i$ by R_i to yield

  $X_i = -1/\lambda \ln R_i$ (3)
- which is justified since both Ri and 1- R_i are uniformly distributed on (0,1).

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

Uniform Distribution:

- Consider a random variable X that is uniformly distributed on the interval [a, b]. A reasonable guess for generating X is given by

  X = a + (b - a)R (5)
- [Recall that R is always a random number on (0,1). The pdf of X is given by

  f (x) = 1/ (b-a), a ≤ x ≤ b 0, otherwise
- The derivation of Equation (5) follows steps 1 through 3 of the previous example for exponential distribution:
- **Step 1.** The cdf is given by
  - F(x) = 0, x < a
  - (x – a) / (b –a), a ≤ x ≤ b 1, x > b
- **Step 2.** Set F(X) = (X - a)/(b -a) = R
- **Step 3.** Solving for X in terms of R yields X = a + (b — a)R, which agrees with Equation (5).

Random Variate Generation

- **INVERSE TRANSFORMATION TECHNIQUE**

Uniform Distribution:

- Consider a random variable X that is uniformly distributed on the interval [a, b]. A reasonable guess for generating X is given by

  X = a + (b - a)R (5)
- [Recall that R is always a random number on (0,1). The pdf of X is given by

  f (x) = 1/ (b-a), a ≤ x ≤ b 0, otherwise
- The derivation of Equation (5) follows steps 1 through 3 of the previous example for exponential distribution:
- **Step 1.** The cdf is given by
  - F(x) = 0, x < a
  - (x – a) / (b –a), a ≤ x ≤ b 1, x > b
- **Step 2.** Set F(X) = (X - a)/(b -a) = R
- **Step 3.** Solving for X in terms of R yields X = a + (b — a)R, which agrees with Equation (5).