Random Number Generation

CS1538: Introduction to simulations

Random Numbers

- Stochastic simulations require random data
  - True random data cannot come from an algorithm
    - We must obtain it from some process that itself has random behavior
    - Ex: thermal noise, atmospheric noise, radioactive decay
- What about all these different implementations of rand()?
- Pseudo-Random Number Generation:
  - Sampling from a true random source may just not be practical
  - Plus, there are advantages to being able to reuse the same "random" values
    - Debugging
    - Compare across systems

Pseudo-Random Numbers

- These numbers are generated deterministically (i.e. can be reproduced)

- However, we still want them to have most of the properties of true random numbers:
  - Numbers are distributed uniformly on [0,1]
    - Assuming a generator from [0,1), which is the most common
  - Numbers should show no correlation with each other
    - Must *appear* to be independent
    - There are no discernible patterns in the numbers

Pseudo-Random Numbers

- First, let's consider the following "generation" algorithm:
  - Suppose we generate $X_0 \ldots X_{99}$ in the following way:

    $X_0 = 0.00;$

    $X_i = X_{i-1} + 0.01$

Pseudo-Random Numbers

- First, let's consider the following "generation" algorithm:
  - Suppose we generate $X_0 \ldots X_{99}$ in the following way:

    $X_0 = 0.00;$

    $X_i = X_{i-1} + 0.01$

- Next idea:

  X_0 = seed value

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

  where the seed value, a, c, and m are values we choose

Pseudo-Random Numbers

- Next idea:

  X_0 = seed value

  $X_{i+1} = (aX_i + c) \bmod m$ for i = 1, 2, …

  where the seed value, a, c, and m are values we choose

- Suppose we want some uniformly distributed integers between 0 and 100. Which seems better?
  - Choice A: let $X_0 = 31$, a=11, c=43, and m=100
  - Choice B: let $X_0 = 27$, a=17, c=43, and m=100

- Is the "better" choice a good enough pseudo random number generator?

Linear Congruential Generators

- Standard Eq's:

  X_0 = seed value

  $X_{i+1} = (aX_i + c) \bmod m$ for i = 1, 2, …

  where a, c, and m are constants we choose;

  if c == 0 it is called a multiplicative congruential generator

  if c != 0 it is called a mixed congruential generator

- Easy and fairly efficient
  - Depending upon parameter choices
  - Simple to reproduce sequences
  - Give good results (for the most part, and when used properly)
- But we wanted random numbers in [0, 1)
  - How can we get that?

Linear Congruential Generators

- The density of the distribution
  - How many different values in the range can be generated?
  - Larger m → more densely populated [0,1)
- The period of the generator
  - How many numbers will be generated before the generator cycles?
    - Since the values are deterministic this will inevitably happen
    - Clearly, a large period is desirable, especially if a lot of numbers will be needed
    - A large period also implies a denser distribution
- The ease of calculation
  - Generate numbers quickly, with few complex operations

Multiplicative Linear Congruential Generators (c=0)

$$X_{i+1} = (aX_i) \bmod m$$

- If **m is prime**, longest period possible is (m–1)
  - Requirement: the smallest integer k such that $2^k - 1$ is divisible by m is k = m – 1.
- If **m = 2^b** for some b, longest period possible is m/4
  - Requirement: X_0 is odd; and a = 3 + 8k or a = 5 + 8k for some k = 0,1…

Mixed Linear Congruential Generators (c≠0)

$$X_{i+1} = (aX_i + c) \bmod m$$

- If **$m = 2^b$ for some b**, the longest possible period is 2^b
  - Requirement: c and m are relatively primes (greatest common factor of c and m is 1) and a = (1+4k) for some k

Some Commonly Used Parameters

- For more, see: http://en.wikipedia.org/wiki/Linear_congruential_generator

Source	m	a (multiplier)	c (increment)
Glibc (GCC) and ANSI C	2^32	1103515245	12345
Numerical Recipes	2^32	1664525	1013904223
MS Visual/Quick C/C++	2^32	214013	2531011
MMIX (by Knuth)	2^64	6364136223846793005	1442695040888963407

Quality of Linear Congruential Generators

- The previous criteria for **m, a** and **c** can guarantee a full period of m (or m-1 for multiplicative congruential generators)
  - This does not guarantee that the generator will be good
  - Still need to check for uniformity and independence in the values generated

Testing for uniformity

- Consider two options:
  - Kolmogorov-Smirnov Test
  - Chi-Square Test
- Both try to compare the generated numbers against the uniform distribution
  - NULL Hypothesis, H_0: The generated data *might* have been sampled from the Uniform Distribution
  - Hypothesis H_1: The generated data is very unlikely to have been sampled from the Uniform Distribution

Testing for uniformity

- NULL Hypothesis, H_0
  - The generated data *might* have been sampled from the Uniform Distribution
- Hypothesis H_1
  - The generated data is very unlikely to have been sampled from the Uniform Distribution
- Procedure:
  - Apply an appropriate statistical test to compute the likelihood of seeing observed data given that H_0 is true.
  - If the chance seems too small, we reject the NULL hypothesis.
    - We formalize "too small" by choosing α, a probability value representing Pr(reject H_0 | H_0 is true).

Kolmogorov-Smirnov Test

- Suppose we generate N values with our RNG. Consider the empirical distribution based on these N values.

- Let $S_N(x)$ be the % of N values that are ≤ x, for any value x
  - For example, consider the following 10 values:

    (0.275, 0.547, 0.171, 0.133, 0.865, 0.112, 0.806, 0.155, 0.572, 0.222)

    $S_N(0.25)$ =
    $S_N(0.5)$ =
    $S_N(0.75)$ =

Kolmogorov-Smirnov Test

- Suppose we generate N values with our RNG. Consider the empirical distribution based on these N values.

- Let $S_N(x)$ be the % of N values that are ≤ x, for any value x
  - For example, consider the following 10 values:

    (0.275, 0.547, 0.171, 0.133, 0.865, 0.112, 0.806, 0.155, 0.572, 0.222)

    $S_N(0.25)$ =
    $S_N(0.5)$ =
    $S_N(0.75)$ =

- The Kolmogorov-Smirnov Test checks to see what is the biggest deviation of $S_N(x)$ from F(x).
  - If max $|S_N(x) - F(x)|$ is very large, we reject the null hypothesis

Procedure of the Kolmogorov-Smirnov Test

- Sort the empirical data $R_{(1)} \leq R_{(2)} \leq \ldots \leq R_{(N)}$
- Compute $D^+ = \max_{1 \leq i \leq N} \{i/N - R_{(i)}\}$
- Compute $D^- = \max_{1 \leq i \leq N} \{R_{(i)} - (i-1)/N\}$
- Compute $D = \max(D^+, D^-)$
- Find D_α in a table for the desired level of α and sample size N
  - If $D > D_\alpha$, reject the null hypothesis
  - See table on next slide, or the course website

Kolmogorov-Smirnov Critical Values

Number of trials, n	Level of significance, α			
	0.10	0.05	0.02	0.01
1	0.95000	0.97500	0.99000	0.99500
2	0.77639	0.84189	0.90000	0.92929
3	0.63604	0.70760	0.78456	0.82900
4	0.56522	0.62394	0.68887	0.73424
5	0.50945	0.56328	0.62718	0.66853
6	0.46799	0.51926	0.57741	0.61661
7	0.43607	0.48342	0.53844	0.57581
8	0.40962	0.45427	0.50654	0.54179
9	0.38746	0.43001	0.47960	0.51332
10	0.36866	0.40925	0.45662	0.48893
11	0.35242	0.39122	0.43670	0.46770
12	0.33815	0.37543	0.41918	0.44905
13	0.32549	0.36143	0.40362	0.43247
14	0.31417	0.34890	0.38970	0.41762
15	0.30397	0.33760	0.37713	0.40420
16	0.29472	0.32733	0.36571	0.39201
17	0.28627	0.31796	0.35528	0.38086
18	0.27851	0.30936	0.34569	0.37062
19	0.27136	0.30143	0.33685	0.36117
20	0.26473	0.29408	0.32866	0.35241
21	0.25858	0.28724	0.32104	0.34427
22	0.25283	0.28087	0.31394	0.33666
23	0.24746	0.27490	0.30728	0.32954
24	0.24242	0.26931	0.30104	0.32286
40[b]	0.18913	0.21012	0.23494	0.25205

[a] Values of $d_\alpha(n)$ such that $p(\max)|F^n(x) - F(x)| d^{\alpha}(n) = \alpha$.

[b] $N > 40 \approx \frac{1.22}{N^{1/2}}, \frac{1.36}{N^{1/2}}, \frac{1.51}{N^{1/2}}$ and $\frac{1.63}{N^{1/2}}$ for the four levels of significance.

Appendix 3, Practical Reliability Engineering, Fifth Edition

Summary of Kolmogorov-Smirnov Test

- We saw the one sample test
  - Compares a sample with a reference (continuous) probability distribution
  - This is a general purpose test. The reference doesn't have to be uniform in general
- Compares the distance between the empirical distribution function of the sample against the CDF of the reference distribution
- Null hypothesis: the sample is from the reference distribution
- Reject null hypothesis if the observed distance is greater than critical value

Chi-Squared Test - introduction

- Another useful general-purpose statistics test
- Can be used for any distribution
- Compares a histogram of observed data (samples) with the expected theoretical values
- Null hypothesis: sample came from the reference theoretical distribution
  - If null hypothesis is true, the sample distribution of the test statistic (sum of squared differences between observed and theoretical frequencies) follows a probability distribution called the chi-squared distribution
- Alternate Hypothesis (H_1)?

Chi Square Test

- Let X be a r.v. that can take on possible values $x_1, x_2, \ldots, x_k$ with probabilities $p_1, p_2, \ldots, p_k$ (p_i's sum to 1)
- Suppose we perform n trials to assign values to X
  - The expected number of times x_i will come up: $E_i = np_i$
- Now suppose there is a r.v. Y that also takes on possible values $x_1, x_2, \ldots, x_k$ but we don't know their probability distribution. However, Y is thought to be the same as X.
- Is distribution Y really the same as X?
  - Check to see if the occurrences of x_i's under Y more or less match those of E_i according to X's distribution.
- NB: For this test to work, the number of trials has to be large enough so that each $E_i \geq 5$.

Chi Square Test

$$C = \sum_{i=1}^{k} \frac{(Y_i - E_i)^2}{E_i}$$

where Y_i is the observed number of occurrences of value x_i and E_i is the expected number of occurrences of value x_i

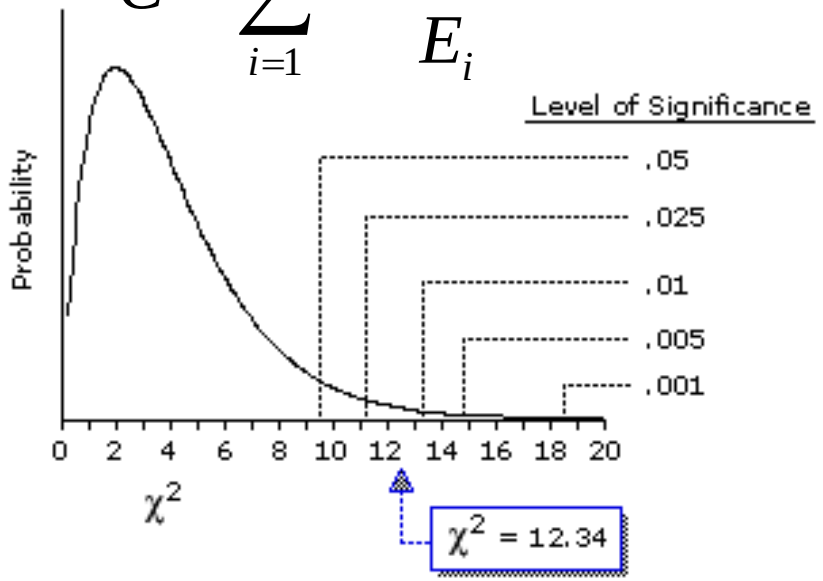
- NULL Hypothesis, H_0 :Y matches distribution of X
- Hypothesis H_1:Y does not match distribution of X
- If the value for C is "too large", we reject the null hypothesis.

- C is actually a random variable whose distribution approximates a χ^2 distribution with k-1 degrees of freedom.
- χ^2 distribution is another special case of Gamma Distribution, where r = (k-1)/2 and λ = 1/2

Chi-Square Test

- C is "too large" basically means the value for the cumulative distribution (area under the curve) is too large compared to what is considered acceptable according to the level of significance (that we specify).

$$C = \sum_{i=1}^{k} \frac{(Y_i - E_i)^2}{E_i}$$



Level of Significance (non-directional test)

df	.05	.025	.010	.005	.001
4	9.49	11.14	13.28	14.86	18.47

critical values of chi-square for **df** = 4

Image from: http://faculty.vassar.edu/lowry/ch8pt1.html

Chi Square Test for uniformity

- Chi Square Test is discrete, so we have to quantize the uniform distribution first.
  - Divide [0,1) into intervals that represent each discrete value.
  - Count how many generated values are in each interval
  - Perform the Chi Square test

- Example
  - Do the random numbers in the spreadsheet on the daily schedule come from U[0, 1)?

Example

- Do the random numbers in the spreadsheet on the daily schedule come from U[0, 1)?
  1. Write out null and alternative hypotheses
  2. Divide [0,1) into 10 intervals (so k-1 = 9)
  3. Bin the numbers from the RNG into their corresponding intervals
  4. Compute C
  5. If we want to be 95% confident about rejecting H_0, we would compare C against $\chi^2_{0.05,9}$

Chi-Squared Critical Value Table

df	Proportion in Critical Region				
	0.10	0.05	0.025	0.01	0.005
1	2.71	3.84	5.02	6.63	7.88
2	4.61	5.99	7.38	9.21	10.60
3	6.25	7.81	9.35	11.34	12.84
4	7.78	9.49	11.14	13.28	14.86
5	9.24	11.07	12.83	15.09	16.75
6	10.64	12.59	14.45	16.81	18.55
7	12.02	14.07	16.01	18.48	20.28
8	13.36	15.51	17.53	20.09	21.96
9	14.68	16.92	19.02	21.67	23.59
10	15.99	18.31	20.48	23.21	25.19

From: http://planetandepoch.com/?p=783

Kolmogorov-Smirnov test vs. Chi-square test

Kolmogorov-Smirnov	Chi-square
Small sample (i.e. small values of N)	Large sample
Reference is continuous distribution	Reference is discrete distribution (but we can quantize continuous reference, as done with Uniform distribution)
Difference between observed and expected CDFs	Difference between observed and expected PDFs
Uses each observed sample without grouping	Group observations (i.e. make histograms)

Returning to Random Number Generation

- Linear Congruential Generators
  - $X_{i+1} = (aX_i + c) \bmod m$ for i = 1, 2, …
    - X_0 = seed value = 31
    - a=11, c=43, and m=100

- Is this RNG good enough?
  - Qualities?

Example

- Was our "better" choice from earlier a good enough pseudo random number generator?

Testing for Independence

- Many Tests:
  - Runs Tests (Wald–Wolfowitz)
  - Auto-correlation (in textbook)
  - Gap Test
  - Poker Test
  - many others …
- A sequence of numbers may pass some tests while not others; therefore: run as many tests as practically possible
  - e.g. the Diehard test suite (http://en.wikipedia.org/wiki/Diehard_tests)
  - Knuth, The Art Of Computer Programming vol. 2 Seminumerical Algorithms
  - For links to more test suites, see: http://csrc.nist.gov/groups/ST/toolkit/rng/documents/nissc-paper.pdf

Runs Tests

- The Wald-Wolfowitz test is on a sequence consisting of two types of elements (say + or -).
  - A run is a contiguous segment of the sequence where the adjacent elements are all the same.
    - Example: In the sequence +++----++-----+-+++
      - 7 runs, with lengths (3,4,2, 5, 1,1,3)
  - In random data the number and length of runs should not be either too great or too small

- Test: let S_+ = number of +'s in the sequence, and S_- = number of –'s in the sequence and $S = S_+ + S_-$
  - The expected number of runs is $\mu = ((2\ S_+\ S_-)/S) + 1$
  - And the variance is $\sigma^2 = (\mu-1)(\mu-2)/(S-1)$

Applying the Runs Test

- To apply the Wold-Wolfowitz test on our sequence of N numbers between [0,1), we have to transform it.
  - Option 1: Define +/- to represent whether a number is above or below the (observed) mean
  - Option 2: Transform the sequence of N numbers into a sequence of N-1 symbols of +/- by taking the differences between adjacent numbers.
    - For this option, we use a different calculation of mean/variance:
      - The expected number of runs is $\mu = (2N-1)/3$
      - And the variance is $\sigma^2 = (16N - 29)/90$
    - If the observed number of runs is very different from (2 N-1)/3, we would reject the null hypothesis that the numbers came from U[0,1).

Hypothesis Testing Procedure

- We know the expected mean and variance, so we can use the normal distribution to check whether the observed mean is too far from the expected.
- Null hypothesis: the numbers are independent
- The test statistic:
  - Subtract observed number of runs from expected, divide by the expected standard deviation
  - $Z = (O-E)/\sigma$
- Say we want to test with a statistical significance of $1-\alpha$
  - Reject null hypothesis if $Z < Z_{\alpha/2}$ or if $Z > Z_{1-\alpha/2}$

Autocorrelation test

- We check for correlations between every l numbers, starting with the ith number.
  - We compute

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

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

where R_i is the value of the i^{th} number, and M is the maximum rounds of intervals we'd get (that is, $i+(M+1)\ell \leq N$))

  - Under the null hypothesis $\rho_{il} = 0$, so we'd reject it if the observed value is very different from 0