# Review of Statistical Terminology

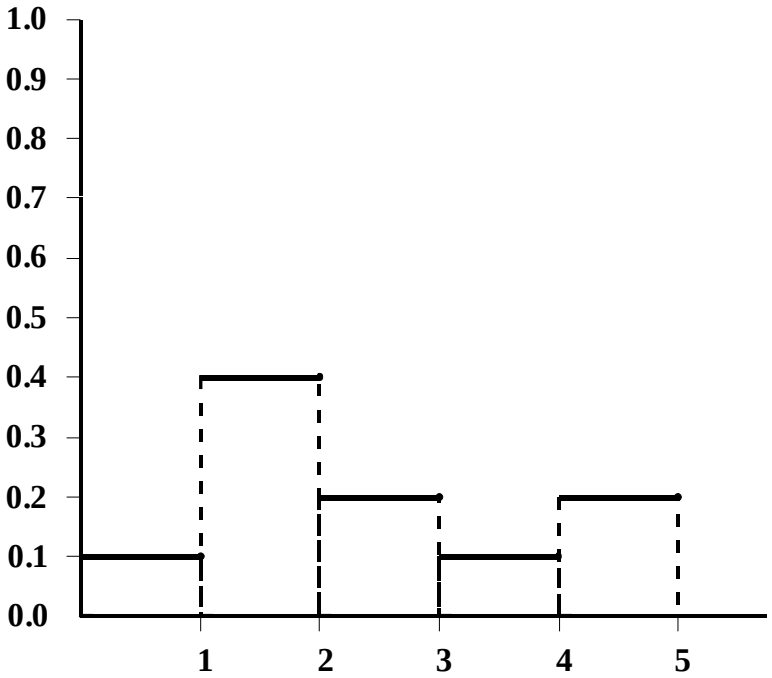
# Formal Terminology

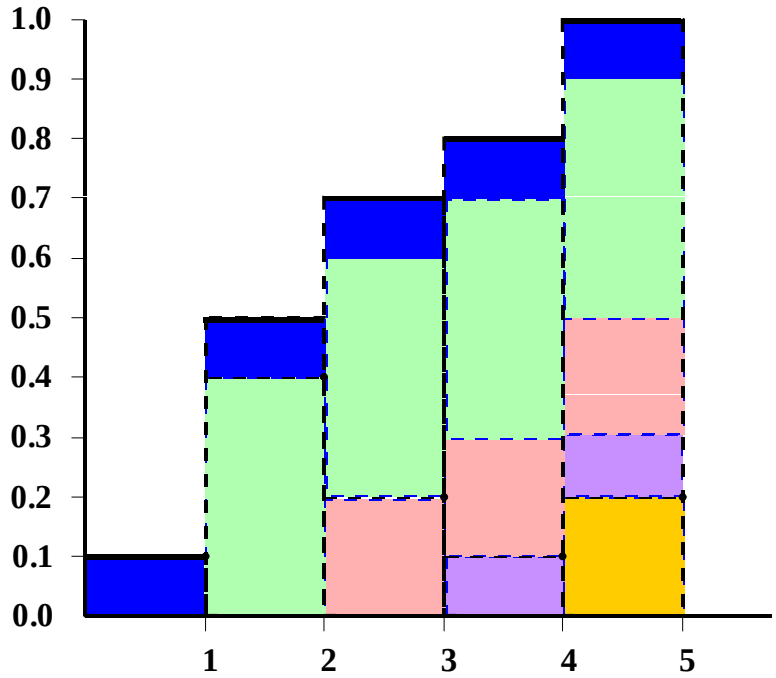
- An ***experiment*** is a process whose outcome is not known with certainty
- The experiment's ***sample space*** S is the set of all possible outcomes.
- A ***random variable*** is a function $X: S \rightarrow R$, so called because the experiment generates an outcome s not known with certainty that has the assigned value X(s), called a ***random variate***; in effect X represents the outcome of the experiment by a point on the real line
- In the finite case the possible outcomes for the random variable X can be delineated $x_1, x_2, \ldots, x_n$. The probability $p_i$ that $X = x_i$ has the property that $\sum p_i = 1$ and gives a function as $f(x_i) = p_i$, called the ***probability distribution*** for X. The cumulative ***distribution function*** F for X is given by $F(x_1) = p_1$, $F(x_2) = p_1+p_2$, $F(x_3) = p_1+p_2+p_3, \ldots,$ $F(x_n) = \sum p_i = 1$

UNF

# Probability Mass Function

- When $f$ is extended to a step function on $1, 2, \ldots$ it is called the probability mass function and $\sum$ becomes $\int$




UNF

# Probability Density Function

- In general an integrable function $f : R \to [0,1]$ such that $\int_R f(x)dx = 1$ is called a ***probability density function*** (pdf)
- If $f$ is a pdf and random variable $X$ has the property that for any $B \subseteq R$

$$\Pr(X \in B) = \int_B f(x)dx$$

then $f$ is called the ***probability distribution*** for $X$ and the ***distribution function*** for $X$ is given by

$$F(x) = \Pr(X \in [-\infty, x]) = \int_{-\infty}^{x} f(z)dz$$

corresponding to the cumulative distribution function for the discrete case

UNF

# Expected Value

- If f is the probability distribution for X, then if the integral

$$E(X) = \int_{-\infty}^{\infty} xf(x)dx$$

exists, it is called the ***expected value*** of X or the ***mean value*** $\mu$ of X.

- In the discrete case, E(X) is just the weighted sum of the probabilities
  - the average value when all items are equally probable
- In a simulation model which uses an empirical distribution, it is easy to calculate its expected value, capture the sample mean from simulation runs, and compare the results to the expected value as a simple model verification check

UNF

# Variance

- The expected value $E[(X-\mu)^2]$ is called the ***variance*** of X

$$E[(X-\mu)^2] = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)dx$$

- It is easy to see in the discrete case that it is just a measure of the mean difference between outcomes of X and the mean $\mu$
  - The square is used because it is algorithmically easier to work with than the absolute value.
- $\sigma^2$ denotes the variance
  - its square root $\sigma$ = ***the standard deviation***

UNF

$$E[(X-\mu)^2] = E(X^2) - \mu^2$$

$$E[(X-\mu)^2] = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)dx$$

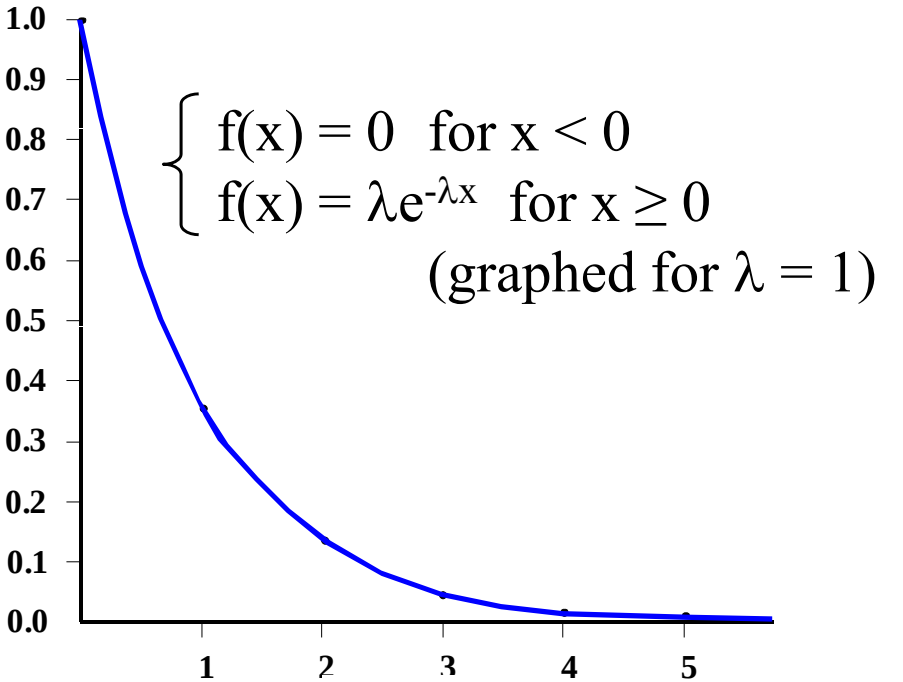
$$= \int_{-\infty}^{\infty} (x^2 - 2x\mu + \mu^2) f(x)dx$$

$$= E(X^2) - \underbrace{2\mu \int_{-\infty}^{\infty} x f(x)dx}_{-2\mu^2} + \mu^2 \underbrace{\int_{-\infty}^{\infty} f(x)dx}_{1} = E(X^2) - \mu^2$$

UNF

# Examples of Probability Distributions

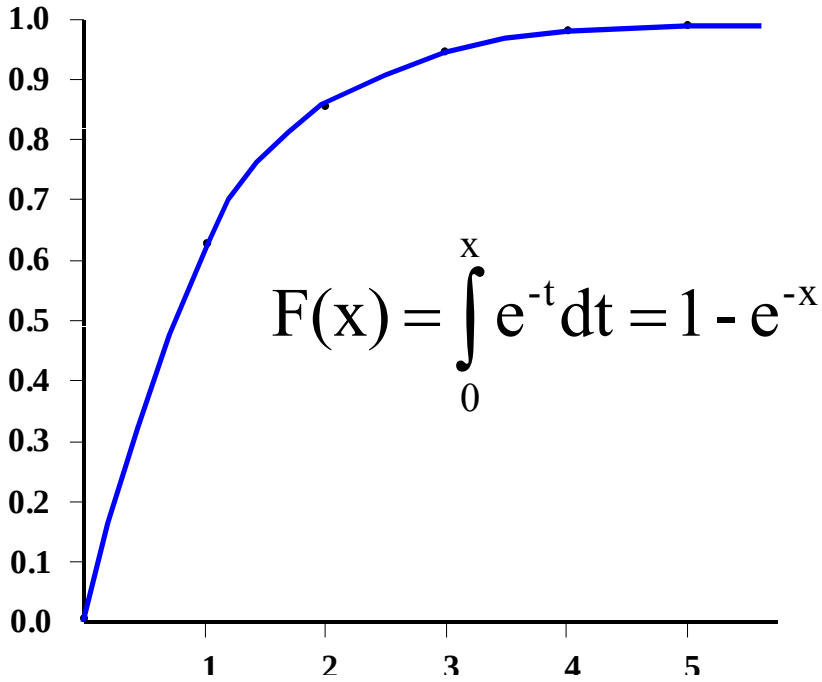
- Exponential distribution




$$\begin{cases} f(x) = 0 & \text{for } x < 0 \\ f(x) = \lambda e^{-\lambda x} & \text{for } x \geq 0 \end{cases}$$

(graphed for $\lambda = 1$)

(pdf)

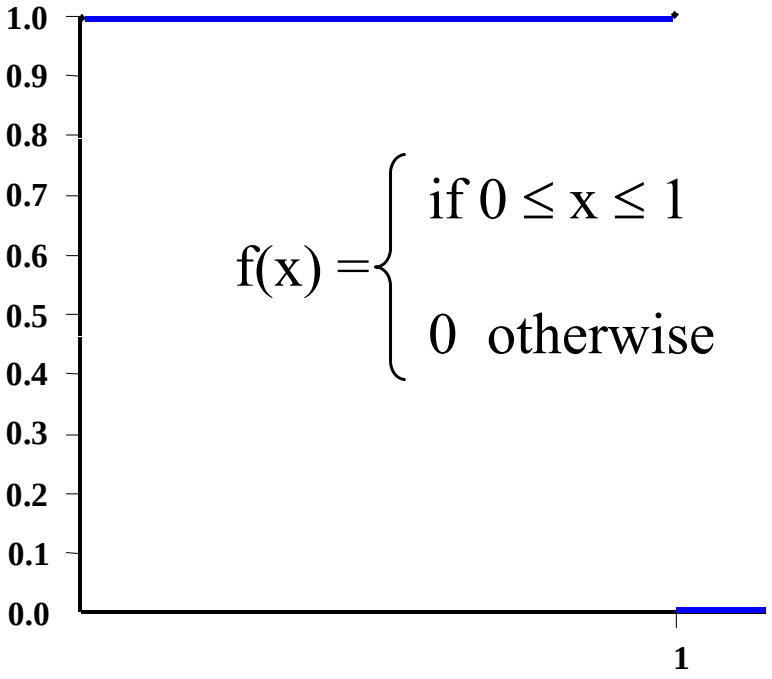



$$F(x) = \int_0^x e^{-t} dt = 1 - e^{-x}$$

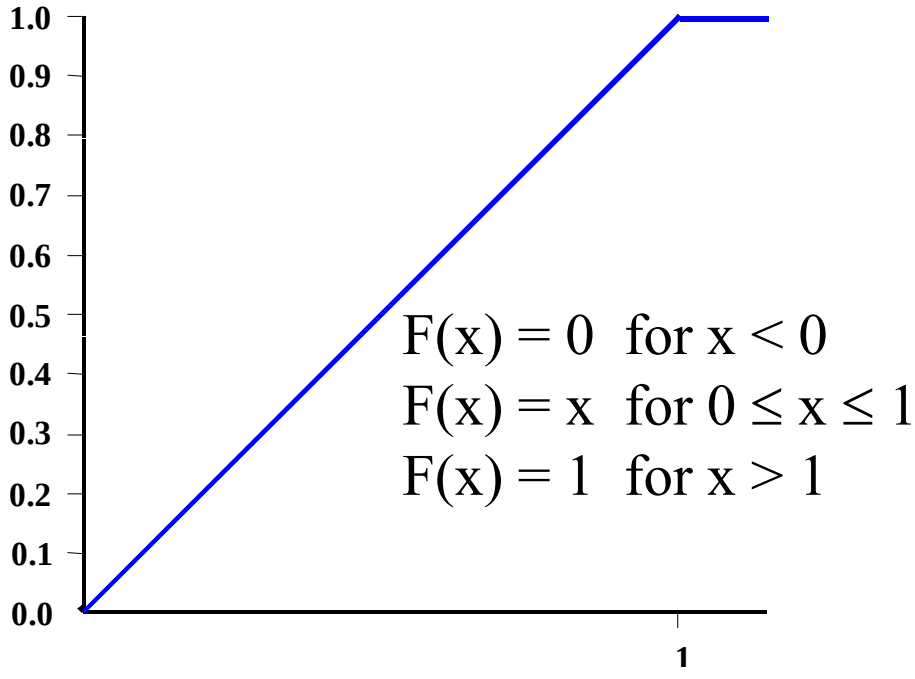
(distribution function)

# Examples of Probability Distributions (2)

- Uniform distribution




(pdf)




(distribution function)

UNF

# Terminology of Random Numbers

- The random variable $X$ on $[0,1]$ with probability density function $f$ (the uniform distribution) given by

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

  is called a ***uniform*** random variable.

- $\Pr(X \in [x, x+\Delta x]) = \int_x^{x+\Delta x} f(z)dz = \Delta x \quad (\text{within } [0,1])$

  so $X$ is equally likely to occur in any interval of length $\Delta x$.

- ***Random numbers*** are random variates of uniform random variables
  - A true random number generator can exist only in theory, since if the underlying process for a generator can be defined, its behavior can be predicted.
- A ***pseudo-random number generator*** (RNG) is a process that in multiple trials produces sets of values that are statistically consistent with statistical properties of "true" random numbers

UNF

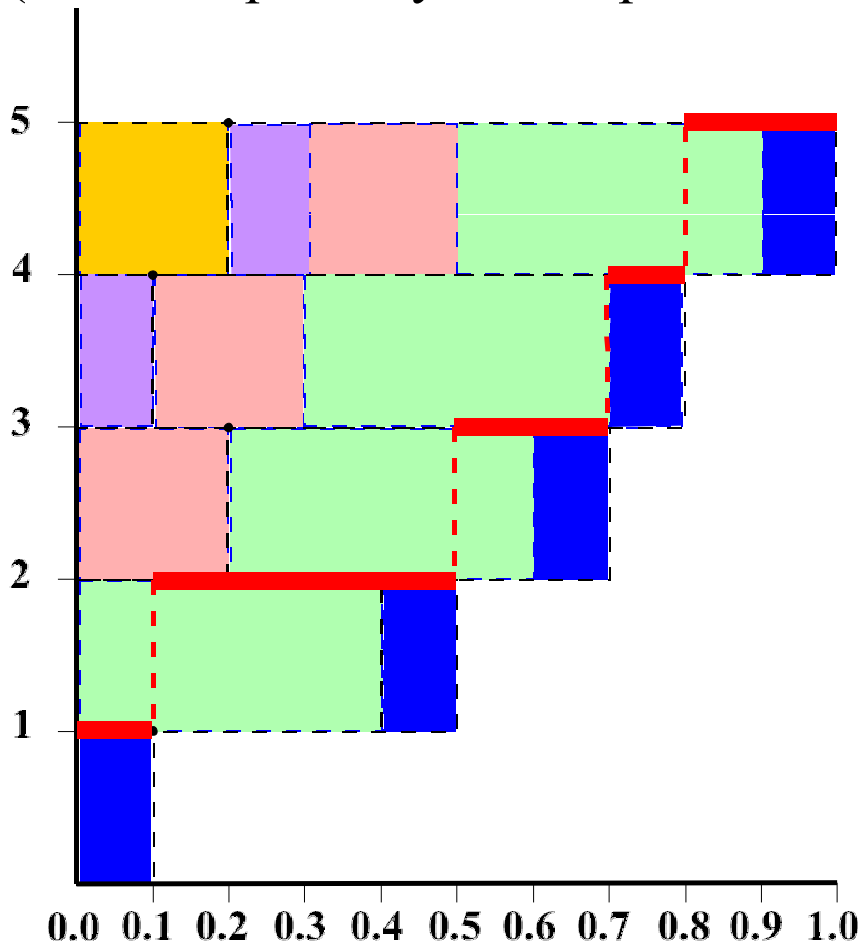
# Simulating a Population Sample

- Members of the population have some characteristic c that occurs in accord with a probability distribution f
  - Ideally, a sample set of random variates for a random variable X with distribution f would give c for the population sample
- In practice, a (pseudo) random number generator is used to produce a sample set of "random" values $r_1, r_2, ..., r_t$ from (0,1]
  - approximately uniformly distributed
- $r_1, r_2, ..., r_t$ are converted to random variate values of X by inverting the distribution function F, if computationally feasible.
- For more complex probability density functions there are other techniques for producing the t random variate values, such as "accept/reject" methods

UNF

# Step Sampling

- f is discrete (either empirically or computationally determined)



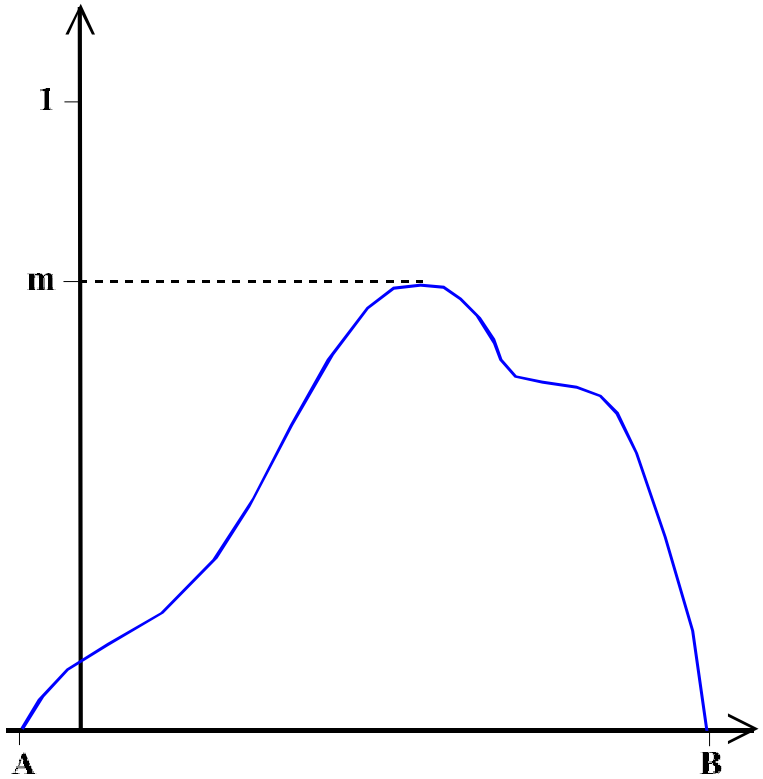

For each $r_k$ let $c_k = \text{step}(r_k) = x_i$, where $F(x_{i-1}) < r_k \leq F(x_i)$ and $F(x_i)$ is the distribution function of the probability mass function for random variate $x_i$

# Sample using $F^{-1}$

- If the distribution function F is invertible then let sample values $c_k = F^{-1}(r_k)$ for each k and random values $r_k$

# Accept/Reject Sampling Method

- Suppose f is given by the curve




- Select a random pair (x,y) where $x \in [A,B]$ and $y \in [0,m]$
  - If $y \leq f(x)$ (i.e., y is under the curve), then ***accept*** x
  - Otherwise, repeat. (***reject*** x and try again - note that this technique "consumes" 2 random numbers on each pass)

UNF

# Random Number Generation

- History
  - ***mechanical generation*** of "random" values has included
    - casting of dice
    - dealing of numbered cards
    - drawing balls from urns (lotteries still use this)
    - spinning disks
    - pulsating vacuum tubes
  - Mechanical techniques were used to produce large tables of random numbers for use in statistical sampling
  - Allows for a repeatable random number stream and statistical analysis of the "randomness" of the numbers in the table
  - Obviously limited due to the relatively small size of the tables produced

UNF

# Random Number Generation

- History
  - ***numerical/arithmetic generation*** of "random" values
    - The ***Linear Congruential Method*** (LCM) first proposed by Lehmer in 1951 provides a basis for forming relatively good RNGs of this sort)
    - There have been inadequate techniques, notably the "mid-square" technique proposed by von Neumann and Metropolis
      - Mid 1940's
      - square a 4 digit number, view the result as an 8 digit number, and use the middle 4 digits to generate the next value
      - unfortunately, the stream produced can easily degenerate

UNF

# Linear Congruential Generators (LCG's)

- Recursively define a sequence of integers $z_1, z_2, \ldots$ by

  $$z_i = (az_{i-1} + c) \bmod m \quad (m > 0)$$

  with basis $z_0$ being a "*seed*" value
  - $m$ is called the *modulus*
  - $a$ the *multiplier*
  - $c$ the *increment* for the LCM
  - $z_0$, $a$, and $c$ are assumed to be non-negative integers each $< m$
- *The "random" values returned by the LCG are given by $z_i/m$*
- Values of $a$, $c$, and $m$ have been established for which the sequence of random values produced by the LCG appear to be random variates of a uniform random variable under a number of statistical tests

UNF

# Choosing LCG Configuration Values *a*, *c*, *m*

- If *a* is a primitive element modulo *m*, then *c* can be taken to be 0 without sacrificing period length
  - This is called a *multiplicative* LCG
  - *a* is a primitive element modulo *m* means that *a* is a cyclic generator for the non-zero integers mod *m*
    - the powers of *a* generate the non-zero integers mod m
- If $m=2^{31}-1$(a Mersenne prime) then any *a* is primitive; in particular $a=16{,}807=7^5$ is primitive
  - The period of this RNG is $m - 1 = 2^{31} - 2$
  - Use of $a=16{,}807$ with this *m* first appeared in a paper by Lewis, Goodman, and Miller in the *IBM Systems Journal*, in 1969
    - $a=16{,}807$ has good statistical randomness properties and implements naturally without overflow if double precision arithmetic is used

UNF

# Research RE LCGs

- Fishman and Moore ("An exhaustive analysis of multiplicative congruential random number generators with modulus $2^{31}$-1"), *SIAM Journal on Scientific and Statistical Computing* (1986), examined all multipliers that are primitive module *m* (about 534 million), testing them according to a number of statistical criteria
  - best case for n-space uniformity uses $a = 62{,}089{,}911$
- Park and Miller [*Communications of the ACM* (1988)] recommend "Easily portable" versions compatible with Schrage's 32-bit method [*ACM Transactions on Mathematical Software* (1979), *A Guide to Simulation* (1983)] that avoid overflow by using $a = 48{,}271$ or $a = 69{,}621$

UNF

# Research RE LCGs

- Law and Kelton (*Simulation Modeling and Analysis*, McGraw-Hill) suggest using $a = 630{,}360{,}016$, also derived in 1969, since it has been widely used, is well-tested, and effective implementations have been published
  - Excel cannot compute the sequence for this value, perhaps because its 15 digit precision is exceeded (smaller values do not cause this kind of overflow
  - It is unclear why Law and Kelton suggest this value since they also indicate calculation takes longer than for smaller values
  - 630,360,016 is not among the best choices reported by Fishman and Moore (but then, neither is 16,807)
- *Numerical Recipes in C* advocates limiting the choice to $a = 16{,}807$ (the "Minimal Standard"), $a = 48{,}271$, and $a = 69{,}621$

UNF

# LCG Period

- An LCG cycles when a seed value repeats
- *m-1* is the maximum possible ***period*** for the LCG
  - With $m = 2^{31} - 1 = 2{,}147{,}483{,}647$ the random number stream of the LCG is easily exhausted at today's computer speeds
    - keep in mind that techniques such as accept/reject use multiple random numbers

UNF

# RANDU: A Notorious RNG (IBM, SYSTEM/360)

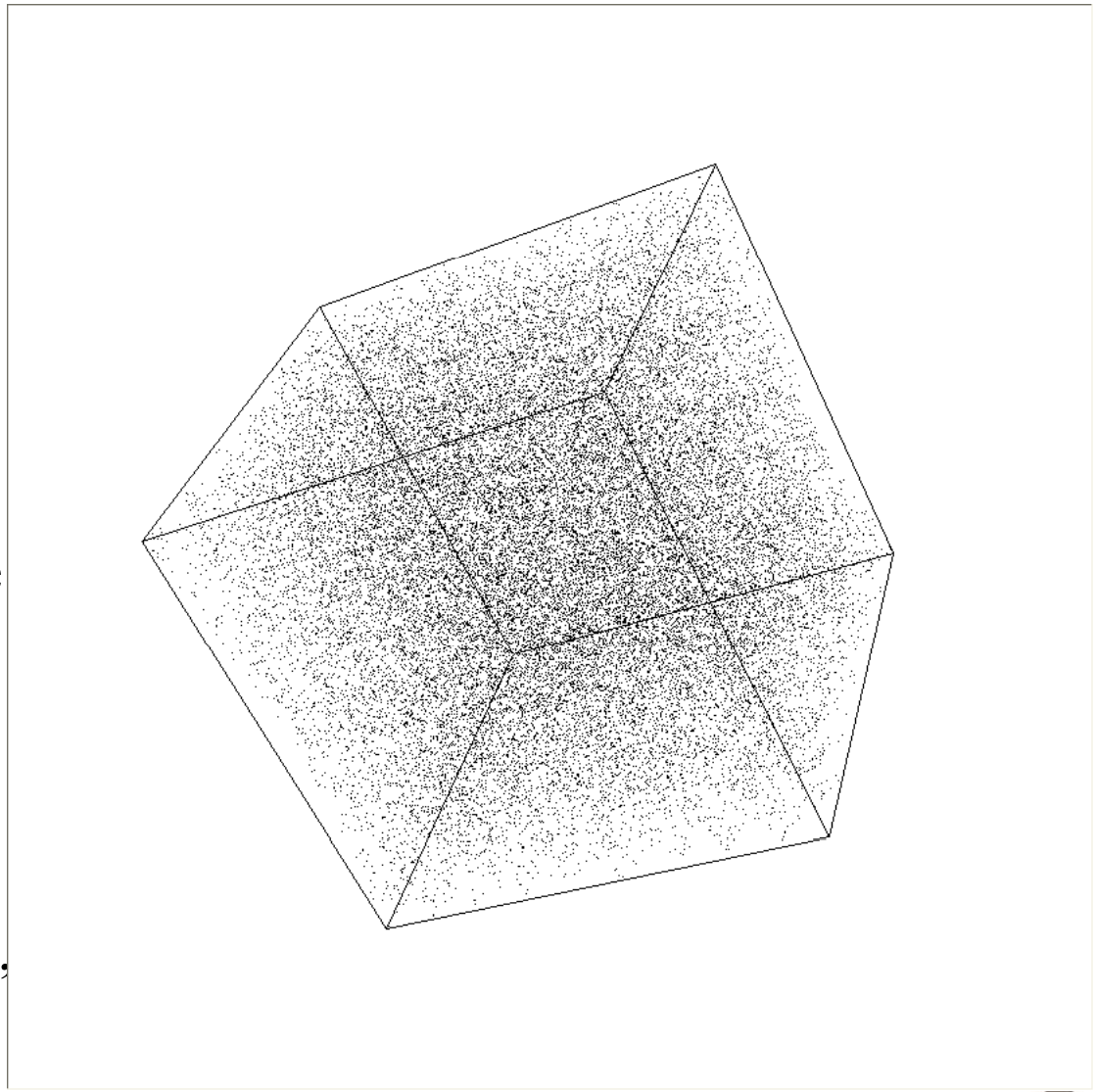
- RANDU, an LCG with $m = 2^{31}$, $a = 2^{16} + 3 = 65539$, $c = 0$ has a major failing which became apparent as per Marsaglia's (1968) observation regarding LCG random numbers that "*random numbers fall mainly in the planes*" [*Proceedings of the National Academy of Sciences*, 61, pp. 25-28 (1968)]
  - Marsaglia's main result states that n-tuples produced using successive values drawn from an LCG lie in at most (n!m)1/n parallel (n -1)-dimensional hyperplanes in n-space
  - For n=3, this implies that with m=$2^{31}$ the points will fall in at most 2,344 parallel (2-dimensional) planes
- For RANDU, a triple of successive values ($x_i$, $x_{i+1}$, $x_{i+2}$) has the property $9x_i - 6x_{i+1} + x_{i+2} = 0 \pmod{2^{31}}$
  - By Marsaglia's Theorem, each of the triples lies on one of the set of parallel planes defined by the equations 9x - 6y + z = 0, ±1, ±2, .... with at most 9+6+0=15 of these intersecting the $2^{31}$ x $2^{31}$ x $2^{31}$ cube containing the triples.

UNF

# More RANDU

- A plot of 32,000 triples of the form

  $(x_0, x_1, x_2), (x_3, x_4, x_5), \ldots$ taken from 96,000 consecutive RANDU values demonstrate that they "clump" along a mere 15 parallel planes!
  - Marsaglia's paper led to the development of tests for checking for *n-space uniformity* (or density) ; e.g., the *spectral test*



UNF

# What Makes a "Good" RNG?

- Numbers generated should appear to **distribute uniformly** on [0,1], exhibiting no correlation with each other
- **Fast** and with a relatively small, fixed memory need
- **Reproducibility** of a random number stream
  - Particularly important for simulation experiments
- The generator should have a period of sufficient length to support several **separate "streams"** of random numbers
  - Sampling from a distribution implicitly calls on the RNG, possibly multiple times
  - Using a separate stream helps to avoid a repeat generation of a population
- The generator should be **portable**
  - When implemented on other systems it should still produce the same results.
- The generator should be **easy to use** in actual implementation; ie., it should not require specialized knowledge for any of implementation, setup, and use.

UNF

# Why it Matters – The Monte Carlo Method

- The Monte Carlo method is used for problems having no probabilistic content as well as those that are inherently probabilistic
  - The method is applied to problems that involve n-tuples in m-space to obtain an approximate solution
  - The method came to the forefront due to its use during the World War II for work on the atomic bomb that involved simulation of random neutron diffusion in fissile material.
- In his 1968 paper, Marsaglia expressed his concern to the National Academy of Sciences regarding the widespread use of LCM generators for problems being addressed using the Monte Carlo method
  - The fact that Fortran is still in wide use in Physics and Engineering, and RANDU persists in Fortran libraries means that this problem persists to this day.

UNF