Lecture 6: Random Numbers 1

Sourendu Gupta

TIFR Graduate School

Computational Physics 1
February 26, 2010

Outline

What is random?

To the best of our present physical understanding, the universe is truly random. Apparent determinism arises through a chance cancellation of randomness (called quantum decoherence).
Take a single atom in a trap and switch on a uniform magnetic field. Then make measurements of the spin component in a fixed direction orthogonal to the magnetic field. If the spin of the atom is $S\hbar$, then the results of successive measurements are random numbers $k\hbar$ where $-S \leq k \leq S$. All values of k are equally likely, and successive values of k are completely independent of each other. Then

$$\langle k \rangle = 0, \qquad \langle k_i k_j \rangle = \frac{\delta_{ij}}{(2S+1)}.$$

Our operational definition of the randomness of such a sequence $\{k_i | 0 \leq i \leq N,\ -S \leq k_i \leq S\}$ is that knowing any N of the values k_i does not allow us to predict the value of the remaining one with a probability better than $1/(2S+1)$.

Uniform and other distributions

In the example that we took, the probability that a measurement yields a value k is

$$P(k) = \frac{1}{2S+1}, \qquad -S \leq k \leq S.$$

Successive measurements are **uncorrelated**, *i.e.*, the joint probability of two measurements giving values $k+1$ and k_2 is

$$P(k_1, k_2) = P(k_1)P(k_2) = \frac{1}{(2S+1)^2}.$$

The joint probability of N number of measurements is a product of the probabilities of each of them [Feller].
The probability that exactly r of the N measurements gives the value ℓ is the binomial distribution. If $P(\ell) = p$, then the above probability is

$$B(N, r) = \binom{N}{r} p^r (1-p)^{N-r}.$$

In the limit $N \to \infty$ this becomes a Gaussian in the variable $x = r/N$.

Continuous variables

In most applications it is useful to discuss random real numbers. Such distributions involve a probability density function, $P(x)$, where x is real number lying in some interval $\mathcal{D}$ (which can be the whole real line). The distribution is normalized, *i.e.*,

$$\int_{\mathcal{D}} P(x)dx = 1.$$

Then the probability that the variable lies in a small interval of width dx around a value x is $P(x)dx$.

> **Problem 1**: Assume that random numbers, x, are drawn from an uniform distribution over the interval $[0, 1]$. What is the distribution of the quantity $\log(1/x)$?

Wider definitions of randomness

Problem 2: Find the probability distribution of $m = k_1 + k_2$. How could we modify the operational definition of randomness in order to accommodate our intuition about the randomness of a sequence of measurements of m?

If the experiment consists of N successive measurement of k and the outcome is r, *i.e.*, the number of measurements where $k = \ell$, then how does not modify the operational definition of randomness in order to accommodate out intuition that the sequence of r's is random?

Given a set of uncorrelated k_i, define $s_0 = k_0$ and $s_i = rs_{i-1} + k_i$. Find the **correlation function** where r is a fixed number $(0 < r < 1)$

$$C(t) = \frac{\langle s_i s_{i+t} \rangle}{\langle s_i^2 \rangle \langle s_{i+t}^2 \rangle}.$$

How should one modify the operational definition of randomness in order to accommodate sequences such as these s_i?

Which sequence is random?

1. 1101011111 1100100100 1001011011 0110110111 0110110110 1101101111 1111111111 1111111
2. 1010110001 0100110100 1001100010 0011110001 0010100100 0001100010 1010001000 1110101

The first has 57 1's and 21 0's, the second has 32 0's and 46 1's. In a random random sequence one would expect $n(0) = n(1) = 39$. But there are **fluctuations** in these numbers.
Is the second sequence within the range of expected fluctuations? Define

$$\chi^2(N) = \frac{[n(0) - p_0 N]^2}{p_0 N} + \frac{[n(1) - p_1 N]^2}{p_1 N},$$

where $N = n(0) + n(1)$, and $p_0 = p_1 = 1/2$ are the expected probabilities of a bit being 0 or 1. The assumption is that fluctuations go as the square root of the expected number. The larger this quantity, the less likely are p_0 and p_1 correct. When $\chi^2(N) \simeq 1$ then the sequence is random [Knuth].

Definition of χ^2

In an experiment with M outcomes and probabilities of the i-th outcome being p_i, one defines

$$\chi^2(N, M-1) = \sum_{i=1}^{M} V_i^2 \qquad \text{where} \qquad V_i = \frac{(N_i - p_i N)^2}{p_i N},$$

where N_i is the observed number of times the outcome is i and N is the total number of trials ($N = \sum_i N_i$). Since the number of independent observations is $M-1$ of the N_i's, the **number of degrees of freedom** is said to be $M-1$. The probability of having a value of $\chi^2(N, M)$ can be computed given $\{p_i\}$.

Example: Suppose $M = 2$ and $p_0 = p$, $p_1 = 1-p$. There are 2^N sequences of length N. The probability of $n(0) = r$ and hence of $n(1) = N - r$ is the binomial distribution $p(N, r) = \binom{N}{r} p^r (1-p)^{N-r}$. The value of χ^2 is $N(1-2r/N)^2$ with the probability $p(N, r) + p(N, N-r)$ if $r \neq N/2$.

The distribution of χ^2 values

For every finite N the distribution of χ^2 values can be computed as in the above example. As $N \to \infty$, the V_i go to a continuous variable. Also, in this limit each V_i can be treated as a Gaussian random number. Then the distribution of the variable χ^2 is

$$P(\chi^2) = \int \prod_{i=1}^{M} \left(\frac{dV_i}{\sqrt{2\pi}} e^{-V_i^2/2} \right) \delta \left(\chi^2 - \sum_{i=1}^{M} V_i^2 \right).$$

Problem 3: Evaluate the distribution $P(\chi^2)$. The simplest way to do this is to take the Fourier transform of both sides. The integral on the right is then easily evaluated. $P(\chi^2)$ can be found by inverse Fourier transformation. Plot the result for different M. Check your results against tables of χ^2 values.

Checking the randomness of a sequence

Problem 4: **The frequency test**: If a sequence of N bits is random with equal probabilities of 0 and 1, then the expected number of each bit is $N/2$. Compute the expected numbers of pairs of bits, triplets of bits, *etc.*

Coupon collector's test: What is the probability that you will have to take exactly r bits in order to get both a 0 or 1? In other words, what is the probability that you will have a run of $r-1$ 0's or 1's? Ares successive values of r independent? [Knuth]

Write a general purpose code that takes as input a sequence of random numbers $\{r_i | 1 \leq i \leq N\}$, the number of different values of r_i, *i.e.*, M, and implements the frequency and coupon collector's tests for the sequence. The answer should be the probability from each test that the sequence is an uncorrelated sequence drawn from an uniform distribution.

Use this code to check whether either of the sequence of bits shown in page 7 is random.

Outline

Is this a random number?

0.318309886

What is a random number?

1. Rational numbers are a measure-zero set in the real numbers. Hence any rational number cannot be a random number.

What is a random number?

1. Rational numbers are a measure-zero set in the real numbers. Hence any rational number cannot be a random number.
2. Polynomials are a measure-zero subset of all functions, hence any algebraic number cannot be a random number.

What is a random number?

1. Rational numbers are a measure-zero set in the real numbers. Hence any rational number cannot be a random number.
2. Polynomials are a measure-zero subset of all functions, hence any algebraic number cannot be a random number.
3. The number of real numbers is $\aleph_0$. The probability of drawing any particular number is zero. Hence no number is random.

What is a random number?

1. Rational numbers are a measure-zero set in the real numbers. Hence any rational number cannot be a random number.
2. Polynomials are a measure-zero subset of all functions, hence any algebraic number cannot be a random number.
3. The number of real numbers is $\aleph_0$. The probability of drawing any particular number is zero. Hence no number is random.
4. Alternative argument based on Kolmogorov complexity: any number can be generated by an algorithmic process. If n bits of the number (for every value of n) cannot be generated by a program which is less than $\mathcal{O}(n)$ bits long, then the number is said to be random. [Knuth]

What is a random number?

1. Rational numbers are a measure-zero set in the real numbers. Hence any rational number cannot be a random number.
2. Polynomials are a measure-zero subset of all functions, hence any algebraic number cannot be a random number.
3. The number of real numbers is $\aleph_0$. The probability of drawing any particular number is zero. Hence no number is random.
4. Alternative argument based on Kolmogorov complexity: any number can be generated by an algorithmic process. If n bits of the number (for every value of n) cannot be generated by a program which is less than $\mathcal{O}(n)$ bits long, then the number is said to be random. [Knuth]
5. In other words, if the information content in n bits of the number can be **compressed** into less than $\mathcal{O}(n)$ bits, then the number is not random.

Information and compressibility

Problem 5: Write a program which counts the number times each ascii character appears in it. The probability of a character α_i in that file is $p_i = N_i/(\sum N_i)$. The entropy of the file is

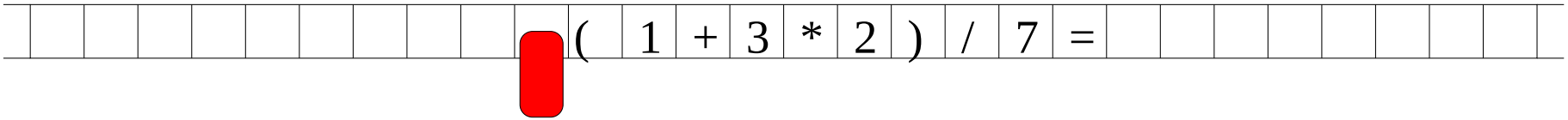
$$S = -\sum_i p_i \log p_i.$$

For what values of $\{p_i\}$ does S reach its minimum? The negative of $S/\log 2$ is called **Shannon information**. It is maximum when the entropy is minimum.

Take a large set of ascii files from many sources (the unix command `file` tells you whether a file is ascii). Find the entropy of each file, S_f, using your program. Find the length of each file in bytes, L_f. Then compress each file using the unix utility `gzip`, and find its new length in bytes L'_f. The compressibility of the file is $K_f = L_f/L'_f$. Plot the pairs (S_f, K_f). What is the functional relation between the two? Do you have a theory for it?

Which programming language?

The length of a program which specifies a number can be different in different programming languages. For a precise definition one uses a **Turing machine**.

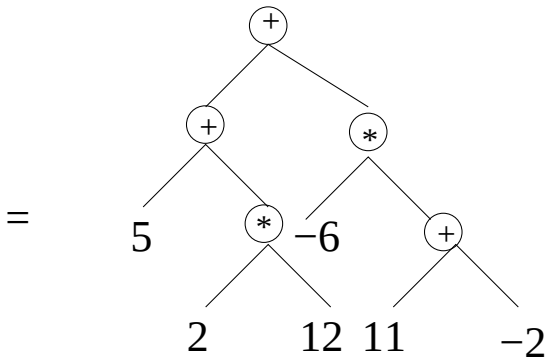


A Turing machine is a model of computation in which a machine reads a tape with data written on it. The machine has a finite number of internal states (colours) which may change according to what is read from the tape. The tape contains a sequence of cells which may be blank or marked up with a (finite) alphabet of symbols. The machine may cause the tape to move left or right. and it may write on the tape or erase the contents of a cell.

The **Church-Turing thesis** is a theorem which states that all possible computations can be carried out by an appropriate Turing machine. There exists an **universal Turing machine** which can simulate any other Turing machine, and hence can carry out all possible computations.

Evaluating an arithmetic expression

$5+2*12-6(11-2) = 5+2*12+(-6)*(11+(-2))$



$=++(5)*(2)(12)*(-6)+11(-2)$

$=+*+(11)(-2)(-6)+*(2)(12)5$

An arithmetic expression is a sequence of binary operations (addition, multiplication and their inverses) along with grouping through brackets. Every expression can be written as a tree: an operation on a every internal vertex and a number on every terminal vertex.

Every tree is evaluated by traversing it in a certain order. When the order of traversal is written out we have the expression in Polish notation (PN). Writing it out in the reverse order we have the reverse Polish notation (RPN). **Not quite correct**

Problem 6: Program a Turing machine which will take any arithmetic expression written in RPN and evaluate it. (Assume for simplicity that each number can be written in a single cell).

Codes

Problem 7: Natural languages when written out using their usual alphabets do not have the minimum possible entropy. The probabilities of all the letters are not equal; the probabilities of two-letter sequences cannot be obtained by assuming that successive letters are independent, and similarly for three, four or five letter sequences.

As a result, a **substitution cipher** (*i.e.*, an enciphering system in which each letter, or di-graph, tri-graph, *etc.*, is replaced by another can be broken by statistical analysis. (Deciphering may involve collecting a large body of cipher text).

Now construct the following **one-time pad**. Evolve any code for the letters of the alphabet using d symbols (for example, each letter of the Roman alphabet can be uniquely encoded using 5 bits, since that allows us to code 32 symbols). Then for every plain text of N symbols (*i.e.*, dN bits), generate a stream of dN random bits. The cipher text is obtained by adding the two binary streams bit by bit modulo 2. Which statistical properties of the plain text carry over into the cipher text? How can one recover the plain text from the cipher?

Outline

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$
2. 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, $\cdots$ generated by the formula $s_i = 3^{i-1} \mod 10$

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$
2. 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, $\cdots$ generated by the formula $s_i = 3^{i-1} \mod 10$
3. 1, 2, 3, 5, 7, 1, 3, 7, 9, 3, 9, 1, 7, 1, 3, $\cdots$ generated by $s_i = prime_i \mod 10$

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$
2. 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, $\cdots$ generated by the formula $s_i = 3^{i-1} \mod 10$
3. 1, 2, 3, 5, 7, 1, 3, 7, 9, 3, 9, 1, 7, 1, 3, $\cdots$ generated by $s_i = prime_i \mod 10$
4. 1, 2, 3, 5, 8, 3, 1, 4, 5, 9, 4, 3, 7, 0, 7, $\cdots$ generated using $s_{i+1} = s_i + s_{i-1} \mod 10$

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$
2. 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, $\cdots$ generated by the formula $s_i = 3^{i-1} \mod 10$
3. 1, 2, 3, 5, 7, 1, 3, 7, 9, 3, 9, 1, 7, 1, 3, $\cdots$ generated by $s_i = prime_i \mod 10$
4. 1, 2, 3, 5, 8, 3, 1, 4, 5, 9, 4, 3, 7, 0, 7, $\cdots$ generated using $s_{i+1} = s_i + s_{i-1} \mod 10$
5. 1, 2, 3, 6, 1, 0, 7, 8, 5, 0, 3, 8, 1, 2, 1, $\cdots$ from $s_{i+1} = s_i + s_{i-1} + s_{i-2} \mod 10$

Is this a random sequence of digits?

The following sequences of numbers are generated by some simple algorithm. Which can be distinguished from a truly random sequence?

1. 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9, $\cdots$ generated by the formula $s_{i+1} = s_i + 2 \mod 10$
2. 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, 7, 1, 3, 9, $\cdots$ generated by the formula $s_i = 3^{i-1} \mod 10$
3. 1, 2, 3, 5, 7, 1, 3, 7, 9, 3, 9, 1, 7, 1, 3, $\cdots$ generated by $s_i = prime_i \mod 10$
4. 1, 2, 3, 5, 8, 3, 1, 4, 5, 9, 4, 3, 7, 0, 7, $\cdots$ generated using $s_{i+1} = s_i + s_{i-1} \mod 10$
5. 1, 2, 3, 6, 1, 0, 7, 8, 5, 0, 3, 8, 1, 2, 1, $\cdots$ from $s_{i+1} = s_i + s_{i-1} + s_{i-2} \mod 10$
6. 7, 7, 4, 1, 5, 6, 4, 0, 4, 1, 5, 6, 1, 7, 8, 7, 5, 2, 7, 9, 6, 4, 6, 0, 6, 9, 5, 4, 1, $\cdots$ using $s_{i+1} = (s_i\%4 == 0)?s_i/4 : s_i + s_{i-1} \mod 10$

Linear congruential generators

Pseudo-random numbers are sometimes generated by an algorithm called a linear congruential generator (LCG):

$$s_{i+1} = (as_i + c) \mathrm{mod}\, m.$$

Random digits are generated when $m = 10$.

1. The sequence cannot have length greater than m.
2. If the sequence has length m for some s_0, then it has length m for all s_0.
3. There are pairs (a, c) for which the sequence has length less than m.

Problem 8: Write a program which runs over all possible values of (a, c) for fixed m and writes out those values of the pair for which the cycle length is m. Investigate all m between 4 and 20. Look for, and report, regularities in the results. Check your hypotheses with larger m. Can you prove some of the regularities?

Another representation of a sequence

Problem 9: The LCG $s_{i+1} = (7s_i + 3) \mathrm{mod}\, 10$ starting from any digit gives sequences of lengths 4 or 2. The sequences are 0341, 27, 5896, and their cyclic permutations. These can be expressed as the digit sequence of a rational number.

For a 2-digit recurrence we have—

$$0.ababababababab\cdots = \frac{a}{10} + \frac{b}{10^2} + \frac{a}{10^3} + \cdots = \frac{10a+b}{99}.$$

For a 4-digit recurrence one obtains —

$$0.abcdabcdabcd\cdots = \frac{d + 10(c + 10(b + 10a))}{9999}$$

What are the rational numbers which represent the digit sequences for various LCG?

Use a similar construction for any LCG specified by the triplet (a, c, m) to find the rational number r_n^{acm} representing the sequence with $s_0 = n$. When the period is maximum what special properties are enjoyed by r_0^{acm}?

Continued fractions

Problem 10: $r_0^{7,3,10} = 0.034103410341\cdots$. Using a simple recursive procedure we find

$$r_0^{7,3,10} = \frac{1}{1/r_0^{7,3,10}} = \frac{1}{29 + 0.3225806\cdots} = \frac{1}{29 + \frac{1}{3+\frac{1}{10}}}.$$

For the same generator we find

$$r_5^{7,3,10} = 0.58965896\cdots = \frac{1}{1 + \frac{1}{1+\frac{1}{2+\frac{1}{3+\frac{1}{2+\frac{1}{7+\frac{1}{3}}}}}}}.$$

Is there anything special about r_0^{acm} when the period is m?

Can continued fractions be used in any way to represent the Kolmogorov-Chaitin complexity of a real number?

Continued fractions were first used by Aryabhata, and subsequently analyzed by Euler.

Fibonacci generators

Random integers can be generated by the sequence $s_{i+1} = (s_i + s_{i-1}) \mathrm{mod} m$. The initial conditions (s_0, s_1) determine the sequence. Digits are generated when $m = 10$.

1. The sequence starting from $(0, 0)$ is clearly trivial.
2. The sequence cannot have length greater than $m^2 - 1$.
3. If the sequence has length $m^2 - 1$ for some (s_0, s_1), then it has the same length for all initial conditions.

Problem 11: Write a program which runs over all possible values of (s_0, s_1) for fixed m and writes out the cycle length for each initial condition. Investigate all m between 4 and 20. Look for, and report, regularities in the results. Check your hypotheses with larger m. Can you prove any of the regularities?

Variants of Fibonacci generators

A general r-term lagged Fibonacci generator (LFG) can be written as

$$s_{i+r} = \left(\sum_{j=0}^{r-1} a_j s_{i-j} \right) \mathrm{mod} m,$$

where a_j are fixed integers. The maximum possible cycle length is $m^r - 1$. The theory of the best choices of $\{a_j\}$ is incomplete: there are some theorems on how to maximize the cycle length, but there is little theory of the statistical properties of the sequences. As a result, numerical experiments are often important to understand the behaviour of these generators.

A **widely used generator** is

$$s_i = (s_{i-24} + s_{i-55}) \mathrm{mod}\, m$$

where $m = 2^e$ and at least one of $s_0, \cdots, s_{54}$ is odd. The period of this generator is $2^{e-1}(2^{55} - 1)$.

Testing Fibonacci sequences

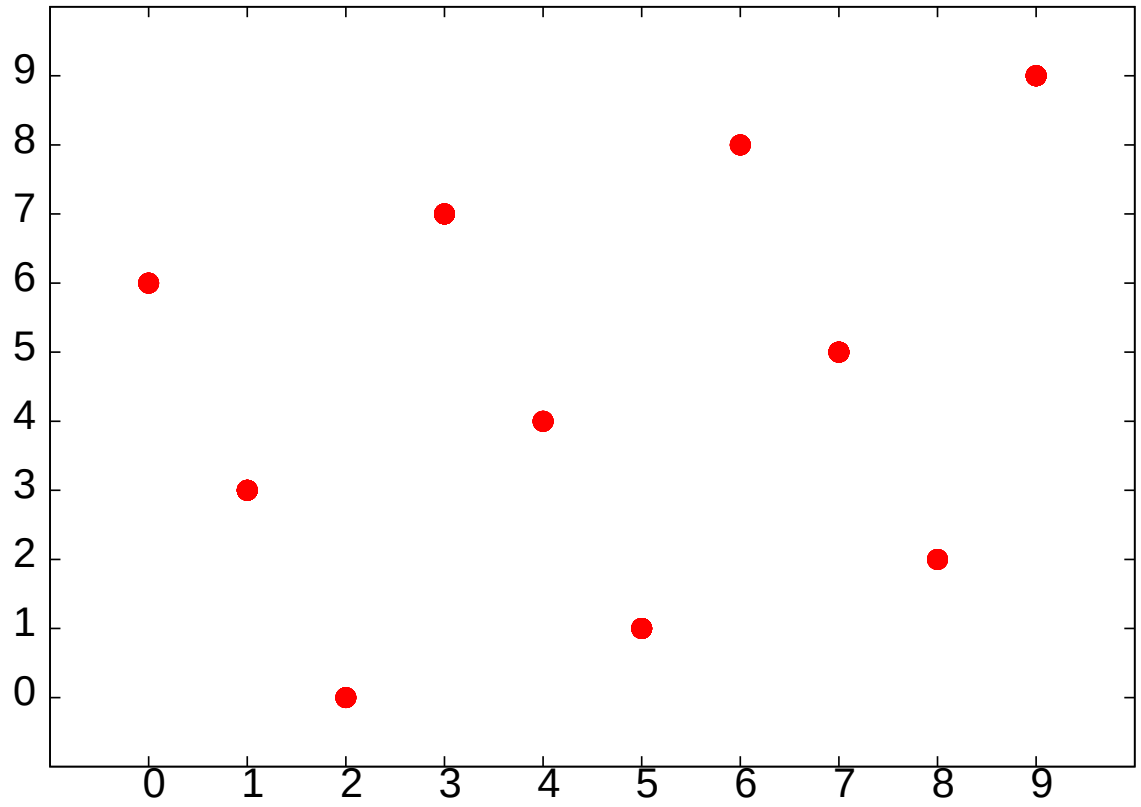
Problem 12: Some LFG's can be classified by the pair (l, k), where

$$s_i = (s_{i-l} + s_{i-k}) \mod 2^e.$$

Some useful pairs are $(24, 55)$, $(38, 89)$, $(37, 100)$, $(30, 127)$, each of which has period $2^{e-1}(2^k - 1)$. Generate large sequences of numbers from each of these generators and perform frequency and coupon collector's tests on them. Measure and plot the correlation functions for each of these sequences.

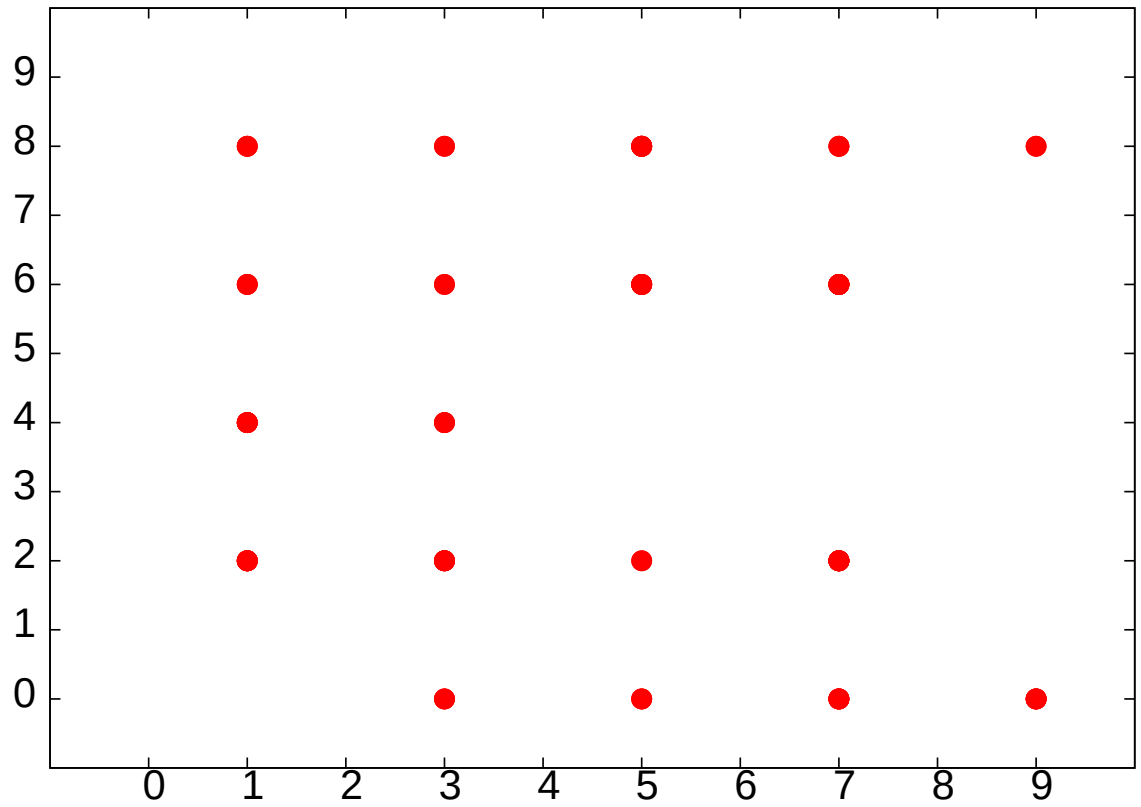
Problem 13: Take the LGF's in the problem above (and any others of your choice) with $m = 2$. Construct the rational numbers which represent the cycles of these generators. Is there any pattern? Construct the continued fraction representation of these numbers. Is there any pattern to these?

Hidden order: the Fibonacci sequence



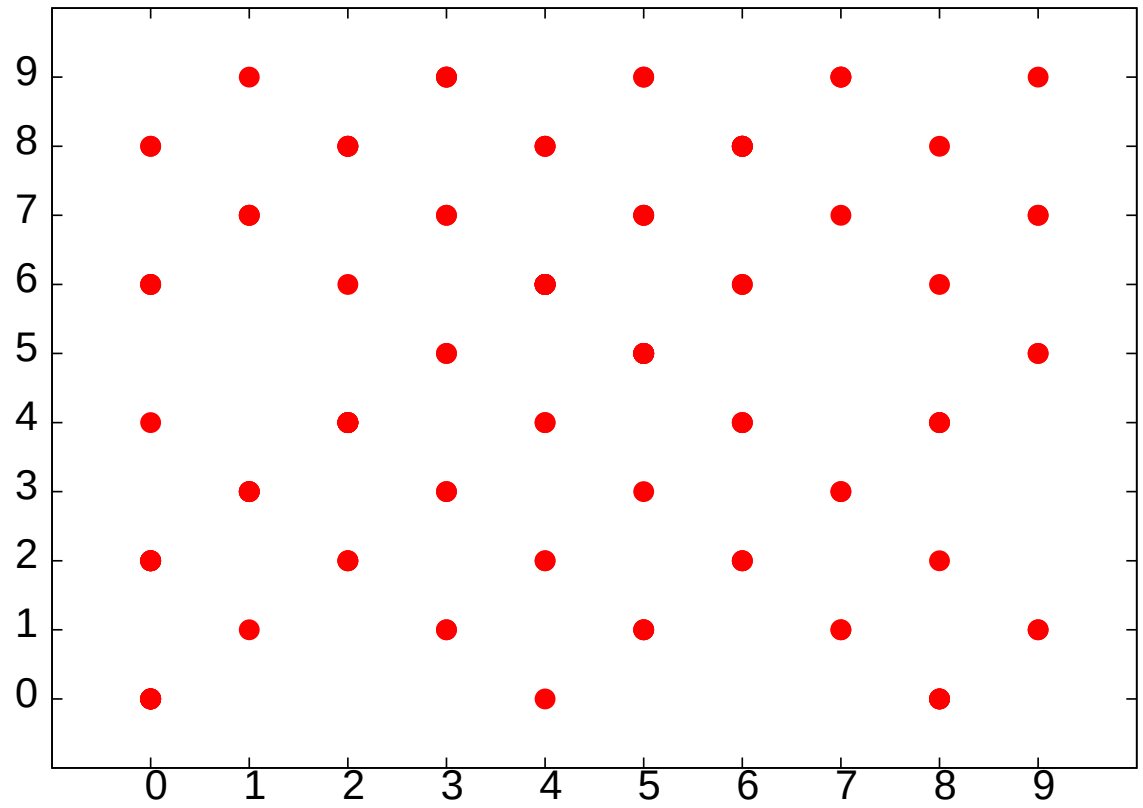
Group into quadruplets, treat as coordinates in 4d space. Plot 1st against 4th coordinates.

Disrupting order in the Fibonacci sequence



3-term recurrence. Group into quadruplets, treat as coordinates in 4d space. Plot 1st against 4th coordinates.

Disrupting order in the Fibonacci sequence



4-term recurrence. Group into octets, treat as coordinates in 8d space. Plot 1st against 6th coordinates.

Order in the LCG

Problem 14: The LCG's defined by the triplets (a, c, m) with values $(21, 1, 40)$, $(21, 1, 80)$, $(21, 1, 100)$, $(21, 1, 1000)$, $(21, 1, 10000)$, $(21, 1, 100000)$ all have the maximum possible cycle lengths, m. The **potency** of an LCG is defined to be the minimum value of s such that

$$(a-1)^s = 0 \pmod{m}.$$

What are the potencies of each of the above generators?

Construct bits $b_i = 0$ when $s_i < m/2$ and 1 otherwise. Examine these bits for randomness using the frequency and coupon collector tests. How are the results correlated with the potency?

Also perform the **equidistribution test**. Define 4-dimensional vectors $(s_i, s_{i+1}, s_{i+2}, s_{i+3})$ where $i \mod 4 = 1$. Check in two-dimensional subspaces whether the points lie in a small number of lines. How are the results correlated with potency?

How do the LFG's with maximum cycle lengths perform on the equidistribution test?

Outline

References and further reading

- "Introduction to Probability Theory and its Applications", by W. Feller, John Wiley, 1983. The comprehensive classic textbook on probability theory and its applications. Highly recommended reading, especially for the very well thought out examples.
- "The Art of Computer Programming: Seminumerical Algorithms", by Donald E. Knuth, Addison Wesley, 2000. This is the classic reference on random number generation and testing. If you can solve all the exercises then you probably don't need to take this course.
- "Gödel, Escher, Bach: an Eternal Golden Braid", by D. Hofstadter Jr. A popular account of complexity, Gödel's theorem, the Church-Turing Thesis, *etc.*. More entertaining than illuminating.
- "Introduction to Automata Theory, Languages and Computation". by J. Hopcroft and J. Ullman, Addison-Wesley, 1979.