Testing Random Number Generators

Dan Biebighauser
University of Minnesota - Twin Cities

REU Summer 2000

Testing Random Number Generators

Introduction

This paper is a summary of the research I conducted during six weeks at the REU Summer 2000 program at the University of Minnesota - Twin Cities. This research, mentored by Prof. Paul Garrett, was initially directed towards comparing different random number generators and presenting the results. This has certainly been done before (see the pLab project at *http://random.mat.sbg.ac.at* for extensive results), and in the course of reviewing others' results I became more interested in the actual tests used to compare different random number generators. A great source for explanations of these tests is the landmark work *The Art of Computer Programming* by D. E. Knuth (second edition, 1981). My project shifted towards studying this work (much of this paper is due to it) and preparing a summary of some of the tests used. This paper begins by introducing random numbers and random number generators, then explains some empirical tests (with an initial focus on the chi-square and Kolmogorov-Smirnov (KS) tests, the foundations for the empirical tests) and also the theoretical spectral test.

What is a Random Number?

We all probably have an intuitive feeling for what *randomness* is. For example, compare the following sequences of heads and tails generated by a fair coin ($P(\text{`heads'}) = 1/2$):

HTHTHTHTHTHTHTHTHTHT

HTHTTHHHTHTHHTTTTHTH

Most people would probably feel that the second sequence is random while the first is not, even though both sequences have the same $1/2^{20}$ probability of exactly occuring (to be honest, I did flip an actual coin for the second sequence and not the first).

Some reflection will show that this 'feeling' is very difficult to define in any mathematical sense. Definitions of randomness, over time, have included human ignorance of initial conditions or human indifference to what comes next in a sequence of events. This is *subjective randomness*, and implies that randomness is only present in human minds, not in the objective world [Bennett 154]. Other definitions rely on being unable to 'predict' future events based on past events (leaving the question of defining 'predictability'; as Prof. Garrett mentioned, in a sequence such as 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, ... it's possible that someone could be "too dumb" to predict the next number – does that make the sequence 'random'?)

Complexity theory sheds a different light on the subject of randomness. Suppose there is a finite set of objects X and a finite set of descriptions of these objects Y. Then let D be a function from Y to X, and for each object x in X, let there exist a description y in Y such that $D(y) = x$. Then each object has a description. Further, let each description be a finite string. The *descriptional complexity* of an object is the length of the string needed to describe it. This set-up avoids the well-known Richard-Berry paradox [Li 1]. It also gives another informal definition of randomness - a string of numbers is 'random' if its descriptional complexity is the same as the length of the string. For example, the string:

1000100010001000100010001000100010001000100010001000100010001000

can be described as '16 copies of 1000', while the string:

0101011111010001100101101000011000110000110011111010001000101010110

doesn't have an obvious description shorter than the string itself. With the above definition, this string could therefore be considered random. Most strings of a length n do not have a description shorter than n, and listing the string itself is the best possible description in these cases [Garrett 77]. If a sequence is generated by a linear feedback shift register (to be described later), then the *linear complexity* of the sequence can be determined, which is much shorter than the sequence itself for large sequences, by use of the Massey-Berlekamp algorithm (the research focus of another REU 2000 student, Erin Casey) [Garrett 212].

Knuth, in his book (section 3.5) lists five 'insufficient' definitions of an *infinite* random sequence before coming to a sixth definition that "surely meets all reasonable philosophical requirements for randomness" [Knuth 156]. This definition is complex, but it might be illustrative to examine an *insufficient* definition of randomness to see why it fails:

A sequence of real numbers U_0, U_1, U_2, ... *in the interval* [0,1) *is defined to be "random" if, whenever* P *is a property such that* $P([V_n])$ *holds with probability one for a sequence* V_0, V_1, V_2, ... *of independent samples of random variables from the uniform distribution, then* $P([U_n])$ *is true.* (In this paper, we let $[S_n]$ denote the entire sequence.)

This is a bad definition, and here's why: *no sequence satisfies this condition.* Let P be the property that no element of the sequence is equal to a fixed number x_0. P certainly has probability one, and so any sequence with x_0 is not random by this definition. Examine our sequence $[U_n]$, and let $x_0 = U_0$. Then our sequence is not random since $P([U_n])$ is false [Knuth 153].

Clearly, randomness is not an easy thing to define. For more of the history of attempts to define randomness, as well as other definitions, see Ch. 9 of *Randomness* by D. J. Bennett.

Random Number Generation

Now that we have a little bit of a handle on what random numbers are, we can look at ways to create these numbers. Although there are many methods of doing this, they all fall into two categories: deterministic and non-deterministic approaches. We'll look at the latter first.

Certainly, any deterministic method of creating sequences of numbers (a formula where the input completely determines the output) will not create random numbers [Garrett 212]. (It wouldn't be a paper on random numbers if it didn't include the oft-quoted statement from von Neumann: "Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin" [Knuth 1].) Our only chance to get *truly* random number sequences (we haven't defined the meaning of 'truly' here) is through a non-deterministic approach. Examples of these, which either have been used or are being used currently, with varying degrees of acceptability, include flipping coins, rolling dice, picking slips out of a well-mixed hat (balls from an urn, etc.), *keyboard latency* (the time elapsed between keyboard actions on a computer), atmospheric noise picked up by a radio receiver (this is done at *http://www.random.org*, and it's where the second sequence in the complexity theory discussion came from), and my personal favorite, the motions of lava lamps (see *http://lavarand.sgi.com*). The best source of random numbers seems to be radioactive materials, generated by hooking a computer up to a Geiger counter (this is being done at the Fourmilab (*http://www.fourmilab.ch/hotbits/*)).

While these sources are interesting in their own right, and the sequences they generate can be analyzed by the tests we'll discuss later, *pseudo-random number generators* (pRNG's), deterministic formulas to create sequences that look random, are good enough for most applications if used carefully and have more interesting mathematical properties. For the remainder of this section, we assume that a random number generator (RNG) is really pseudo-random. Later, in the discussion of testing RNG's, empirical tests will evaluate the randomness of any sequence, regardless of its source, and the (theoretical) spectral test works only for the linear congruential generator (LCG), a pRNG to be described shortly. There are many types of pRNG's; the purpose of this paper is not by any means to give a comprehensive list, but to list a few and some of their properties to give some perspective before describing the testing of sequences.

Linear congruential generators (LCG's) are defined by selecting the four integers:

m, the modulus, with $m > 0$
a, the multiplier, with $0 \leq a < m$
c, the increment, with $0 \leq c < m$
x_0, the seed, with $0 \leq x_0 < m$

and defining the sequence $[x_n]$ of 'random numbers' by the recursive

$$x_{i+1} = (ax_i + c) \ \% \ m$$

('%' in this sense means to reduce the result mod m, leaving the smallest positive remainder.)

For example, let $m = 8$, $a = c = 3$ and $x_0 = 2$. The sequence generated by these values is

$$2, 1, 6, 5, 2, 1, 6, 5, 2, ...$$

Since the definition is recursive, if a number reappears in the sequence (as one eventually must, since m is finite), then the entire sequence begins to repeat. The *period* of a sequence is a cycle of endlessly repeating numbers, and the length of the period is defined to be the smallest k such that $x_{i+k} = x_i$ for all indices i (we could force i to be greater than some i_0 to allow for some early non-repetition in a pRNG)[Garrett 213, 214]. The above sequence therefore has a period of length 4.

It is desirable to have the period of a pRNG be as long as possible (to make it more 'random'). There are ways to do this; without going into any of the details, we briefly present them here for LCG's (for much more on LCG's and proofs of the following details, see Knuth, p. 9-25, and Garrett, p. 213-216, 219-222):

(due to Carmichael) With $c = 0$ and m prime, the maximum possible period of $m - 1$ is achieved when x_0 is relatively prime to m and a is a primitive root modulo m (g is a *primitive root modulo m* if for every x relatively prime to m there is an integer l so that $g^l = x \bmod m$ [Garrett 68])[Knuth 19].

A LCG has period length m if and only if:
1) c is relatively prime to m
2) $a - 1$ is a multiple of p, for every prime p dividing m
3) $a - 1$ is a multiple of 4, if m is a multiple of 4 [Knuth 16]

Another consideration is the potency of a LCG with period m. The *potency* is defined as the smallest positive integer l such that

$$(a - 1)^l = 0 \bmod m$$

(l will exist if the above requirements for a period of length m are met)[Knuth 23]. Knuth recommends a potency of at least 4 and probably 5 or higher. This condition is necessary but not sufficient to have a 'good' LCG [Knuth 24].

While LCG's are sufficient for some applications, there are better pRNG's. Related to the LCG's are the *linear feedback shift registers* (LFSR's). The following discussion is based on Garrett's work (p. 216-217, 222-225):

With a size N and a modulus m, choose coefficients $(c_0, c_1, ..., c_{N-1})$ and a *seed* (like the LCG's x_0) $(s_0, s_1, ..., s_{N-1})$ (these will be the first numbers of the sequence). Then define

$$s_{n+1} = (c_0 s_n + c_1 s_{n-1} + c_2 s_{n-2} + ... + c_{N-1} s_{n-(N-1)}) \ \% \ m$$

for all $n + 1 \geq N$. (Note that if $N = 2$ and $c_0 = c_1 = s_0 = s_1 = 1$ then this is the Fibonacci sequence $1, 1, 2, 3, 5, 8, ...$) This process is more easily expressed in terms of matrices. As a simple example, let $N = 3$ (with coefficients (c_0, c_1, c_2)) and write

$$C = \begin{pmatrix} c_0 & c_1 & c_2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Then the recursive definition above can be written as

$$\begin{pmatrix} s_{n+1} \\ s_n \\ s_{n-1} \end{pmatrix} = C \begin{pmatrix} s_n \\ s_{n-1} \\ s_{n-2} \end{pmatrix} \ \% \ m$$

For instance, let $m = 2$, $(c_0, c_1, c_2) = (0, 1, 1)$ and $(s_0, s_1, s_2) = (0, 0, 1)$. Then the sequence (including the initial 0,0,1) is

$$0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, ...$$

Note that in a LFSR, if a pattern of N consecutive elements that occured earlier in the sequence recurs later, the entire sequence begins to repeat. Here the initial 0,0,1 recurs after 7 steps, and the LFSR has period 7. These ideas are similar to those of the LCG's, but it can be seen that it is easier to create longer periods with LFSR's since more than one element needs to recur for the sequence to begin repeating.

The period of a LFSR can also be examined. Let the *characteristic polynomial* of the matrix C be (as usual)

$$\det(xI_N - C)$$

We state without proof (see Garrett) that if the characteristic polynomial of C in $\mathbf{Z}/m[x]$ is primitive, then the associated LFSR has the maximum period $m^N - 1$ for *any* initial seed other than all zeros. (A polynomial P of degree N in $\mathbf{Z}/m[x]$ is *primitive* if P divides $x^N - 1$ but does not divide $x^t - 1$ for any integer t with $0 < t < N$). Note the difference in maximum period size for a LFSR with modulus m compared to a LCG with the same modulus.

Let's examine one more pRNG for comparison. The Blum-Blum-Shub generator is included here for three reasons. First, it's the pRNG used with the lava lamps mentioned earlier. Second, it's fun to say out loud. Third, and most important, it's a really good generator, *provably* secure assuming that it is *hard* (the algorithm doesn't run in polynomial time) to factor large numbers into primes (which is believed to be true) [Garrett 217]. This fact makes this pRNG much more useful for cryptographic applications than the previous two (which should not be used for any serious cryptography), but each bit in the sequence is more *expensive* (difficult) to compute, so the LCG's and LFSR's would be better for simpler applications when used with care. The Blum-Blum-Shub generator works as follows:

Select two very large primes p, q both congruent to 3 mod 4 and compute the product $m = pq$. This m is the modulus. Start with seed s_0 and compute the sequence $s_1, s_2, s_3, ...$ by the formula

$$s_{n+1} = s_n^2 \ \% \ m$$

From this sequence, create a sequence of bits by

$$b_n = s_n \ \% \ 2$$

This is the desired (pseudo)random sequence [Garrett 217].

Of course, there are many other pRNG's in use, many better than those described here (or at least better than the first two). The purpose of this section was to provide some insight as to how random numbers might be generated.

Applications for Random Numbers

We now know some methods for creating random numbers, but an unanswered question here is why we would want to do such a thing. This section will briefly list some applications of random numbers, and then we'll move on to the testing of random number sequences.

Here are some uses for random numbers (some uses are obviously more mathematical in nature than others):

1. *Simulation* Random numbers are almost always necessary to make a realisitic model of natural phenomena. These simulations include economic, traffic, nuclear physics and many other models. A benefit of some of the pRNG's is that the sequence can be set so that it starts at the same place in the sequence each time, allowing one to look at the effects of varying certain parameters while exposing it to the same 'random' sequence each time [Bennett 150, Knuth 1].

2. *Statistical Sampling* Obviously, samples are sometimes needed to study a larger collection of things, and a random sample would lend itself to the best possible accuracy of these statistical tests [Bennett 148, Knuth 1].

3. *Cryptography* Public key crytography systems make use of large amounts of random data. For example, RSA requires the use of large, random primes for its security. One-time pads require a long keystream of random integers (if these integers are generated by a periodic pRNG, the period needs to be longer than the message to be encoded, or the cipher effectively degenerates into a Vigenere cipher, which is vulnerable to attack (unlike a one-time pad)) [Bennett 149, Garrett 82, 213].

4. *Computer Programming* Many computer algorithms in use require a random number or random sequence, and random numbers can be used as input to test the effectiveness of an algorithm [Knuth 1].

5. *Numerical Analysis* Many problems that are too difficult to solve (or quickly solve) can be approximated by techniques relying on random numbers (Monte Carlo methods, for example) [Bennett 136, Knuth 1].

6. *Decision Making* In 'real life' decisions, computer algorithms (mentioned above) and game theory, randomness can play an important role [Knuth 2].

7. *Recreation* Finally, there are many fun applications of random numbers (not that the 'mathematical' ones aren't fun), including gambling and computer games [Knuth 2, *www.random.org*].

An important thing to remember when dealing with applications for random numbers: *no random number generator is good for every application* [Knuth 173]. As a simple example, a truly random infinite sequence of zeros and ones will contain a million zeros in a row (in fact, this will happen an infinite number of times). This behavior is necessary for an application requiring extremely large amounts of random numbers (otherwise the sequence couldn't be considered random), but it is devastating for an application that just needed, say, a thousand random digits and received a thousand zeros from the RNG (this ties into the distinction between *global* nonrandomness and *local* nonrandomness). Caution should always be used when choosing a RNG for a situation, and more than one RNG is sometimes needed for an application [Knuth 145].

Testing Random Number Generators

So far, we've seen that there are many different applications requiring random numbers, and that there are many different ways of acquiring them. We've mentioned above that not all sources of random numbers behave in the same way, and that some are better than others, at least for different applications. This begs the question: how can we tell if a random number generator is 'good' (or 'good enough')? We've already seen that there are ways to maximize the period of a LCG or a LFSR. Certainly, this is not the only requirement we would demand of a RNG. For example, it would be important for a RNG designed to produce a long sequence of zeros and ones to produce them in roughly equal quantities. It would be nice for there to be independence between elements or subsequences of a sequence produced by a RNG. In addition, a RNG should be fairly efficient in order to be of any real use.

Not surprisingly, there are a lot of different tests for RNG's and the sequences they produce. These tests can be divided into two distinct groups: empirical tests and theoretical tests. *Empirical* tests are conducted on a sequence generated by a RNG, and require no knowledge of how the RNG produces the sequence. *Theoretical* tests, which are better when they exist, are *a priori* tests in the sense that they require a knowledge of the structure of the RNG but the sequence does not necessarily need to be generated [Knuth 38, 75]. We will focus mainly on the empirical tests here. Before we look at these tests, there are two major tests that provide the foundations for the empirical tests, the *chi-square* test and the *Kolmogorov-Smirnov* test, and these will be discussed in detail. Following this, we'll list some of the empirical tests with brief descriptions. The *spectral* test is a theoretical test used for LCG's and will be briefly described as well. All of the following has its roots in Knuth's book, p. 38-113, and the interested reader is referred to this excellent work for additional details and proofs of the following material.

The Chi-Square Test

The chi-square (χ^2) test was initially published by Karl Pearson in 1900. Pearson's original notation in explaining the theory behind the test included use of the χ^2 symbol; hence the name. This test can be used in many situations and basically, when given an outcome of an experiment, can give an approximate probability as to how likely that outcome is [Knuth 39, 52, 54].

Suppose we had n independent observations (in our case, perhaps elements of a sequence generated by a possible RNG), each falling into one of k categories. Let Y_s be the number of observations falling into the sth category and p_s be the probability that an observation falls into category s. Then we would expect that

$$Y_s \approx np_s$$

for large values of n. We'd like a way to measure 'how far off' we are from these expected values, so define the reasonable statistic V_i (the i here stands for 'inadequate,' as we'll shortly see) as the following

$$V_i = (Y_1 - np_1)^2 + (Y_2 - np_2)^2 + ... + (Y_k - np_k)^2$$

(the squaring is to make each term positive; without this, discrepancies could 'balance out'). This will give some measure as to how close the actual results are to the expected.

Note that V_i gives equal weight to each category. If not every p_s is the same, then V_i can be misleading because it could 'overemphasize' some discrepancies and 'hide' others. We modify the statistic to the one that's actually used, V, which gives the correct weight to each term. V (called the *chi-square* statistic of Y_1,...,Y_k [Knuth 40]) is defined as

$$V = \frac{(Y_1 - np_1)^2}{np_1} + \frac{(Y_2 - np_2)^2}{np_2} + ... + \frac{(Y_k - np_k)^2}{np_k} = \sum_{1 \le s \le k} \frac{(Y_s - np_s)^2}{np_s}$$

This solves the weight problem (if only it were so easy for humans). V can be written another way, which is often easier to compute. We take advantage of two fairly obvious facts:

$$Y_1 + Y_2 + ... + Y_k = n$$

$$p_1 + p_2 + ... + p_k = 1$$

and write V as

$$\begin{aligned} V &= \sum_{1 \le s \le k} \frac{(Y_s - np_s)^2}{np_s} = \sum_{1 \le s \le k} \frac{Y_s^2 - 2np_sY_s + n^2p_s^2}{np_s} \\ &= \sum_{1 \le s \le k} \frac{Y_s^2}{np_s} - \sum_{1 \le s \le k} 2Y_s + \sum_{1 \le s \le k} np_s = \frac{1}{n} \sum_{1 \le s \le k} \frac{Y_s^2}{p_s} - 2n + n \\ &= \frac{1}{n} \sum_{1 \le s \le k} \frac{Y_s^2}{p_s} - n \end{aligned}$$

Now the important question: what's a reasonable value for V? We would expect it to be bigger than zero (Y_s probably doesn't *equal* np_s for large values of n), but it shouldn't be too large. We begin to answer this question by referring to a table such as the following [Knuth 41]:

TABLE 1
Selected Percentage Points Of The Chi-Square Distribution

	$p = .01$	$p = .05$	$p = .25$	$p = .50$	$p = .75$	$p = .95$	$p = .99$
$\nu = 1$	0.00016	0.00393	0.1015	0.4549	1.323	3.841	6.635
$\nu = 2$	0.02010	0.1026	0.5753	1.386	2.773	5.991	9.210
$\nu = 3$	0.1148	0.3518	1.213	2.366	4.108	7.815	11.34
$\nu = 4$	0.2971	0.7107	1.923	3.357	5.385	9.488	13.28
$\nu = 5$	0.5543	1.1455	2.675	4.351	6.626	11.07	15.09
$\nu = 6$	0.8720	1.635	3.455	5.348	7.841	12.59	16.81
$\nu = 7$	1.239	2.167	4.255	6.346	9.037	14.07	18.48
$\nu = 8$	1.646	2.733	5.071	7.344	10.22	15.51	20.09
$\nu = 9$	2.088	3.325	5.899	8.343	11.39	16.92	21.67
$\nu = 10$	2.558	3.940	6.737	9.342	12.55	18.31	23.21
$\nu = 11$	3.053	4.575	7.584	10.34	13.70	19.68	24.73
$\nu = 12$	3.571	5.226	8.438	11.34	14.84	21.03	26.22
$\nu = 15$	5.229	7.261	11.04	14.34	18.25	25.00	30.58
$\nu = 20$	8.260	10.85	15.45	19.34	23.83	31.41	37.57
$\nu = 30$	14.95	18.49	24.48	29.34	34.80	43.77	50.89
$\nu = 50$	29.71	34.76	42.94	49.33	56.33	67.50	76.15

To use this table, read the line with $\nu = k - 1$. Then compare V to the entries in that row. For example, if $k = 9$ ($\nu = k - 1 = 8$), then the fact that the 99 percent entry is 20.09 means that $V < 20.09$ around 99 percent of the time and we would expect $V > 20.09$ to occur only about 1 percent of the time (it should be repeated that the chi-square test gives approximate probabilities), and a V of 35 would be pretty suspicious.

One might ask why we look at the $(k-1)$th row. ν is called the number of *degrees of freedom* [Knuth 41]. This can be explained much more precisely (see Knuth, p. 53), but intuitively, the $Y_1, Y_2, ..., Y_k$ can be seen as independent Poisson random variables except for the fact that $Y_1 + Y_2 + ... + Y_k = n$, so if we know, say, $Y_1, Y_2, ..., Y_{k-1}$, we can compute Y_k, meaning that only $k-1$ of the Y_s's are 'independent' [Knuth 41].

Note that these numbers in the table appear to only be related to k, amazingly not the number of observations n or the probabilities of landing in each category p_s. This is not entirely correct, since the numbers in the table are approximations that are only valid for large values of n [Knuth 42]. Knuth recommends that n be big enough so that each expected value np_s is at least 5, although this seems to be a fairly arbitrary choice (according to *http://www.windsor.igs.net/* (full address in bibliography), it has recently been demonstrated that 5 in each category is not necessary as long as there's *something* in each category). The correct choice of n for each situation is not clear, since a really large n can better detect global nonrandomness but 'smooth out' local nonrandomness. Since the test really should be run more than once on a sequence anyway (recall the 'million zeros' discussion; just because something only happens 1% of the time doesn't mean it can't or shouldn't happen), these different tests could use different values for n, hopefully making the results more accurate.

How do we interpret the results of a chi-square test? Knuth recommends a fairly arbitrary method (this will be improved on later, although it's good enough for now). Consider

V less than the 1% entry or greater than the 99% entry to be "nonrandom"

V between the 1% and 5% entries or between the 95% and 99% entries to be "suspect"

V between the 5% and 10% entries (a more detailed table would be necessary here) or between the 90% and 95% entries to be "almost suspect"

One would test parts of the 'proposed' random sequence at least three times, probably with a different n each time, and if two or more tests are "suspect," the sequence is not suffciently random [Knuth 44]. Knuth tested a few LCG's with the chi-square test (see p. 44, 45); many more LCG's and other pRNG's have been tested at the pLab project (*http://random.mat.sbg.ac.at*).

One major question remains, and it won't be answered here. The reader might wonder how the above table is generated. The process is not trivial and is too complicated to include here, but an interested reader should see Knuth's book, p. 53, 54, for the details.

Again, the chi-square test is a foundation for many of the empirical tests we'll describe, and it is probably the most-used test for RNG's.

The Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov (KS) test has its origins in a 1933 paper by A. N. Kolmogorov, and N. V. Smirnov suggested some improvements in 1939, leading to the joint name of the test [Knuth 54]. The KS test is useful in areas where the chi-square test is not, and can also be used in conjunction with the chi-square test. Because of this, it too is a foundational test for many of the empirical tests to follow, and will be examined here.

We first introduce a common function in probability theory. Given a random variable X, the *cumulative distribution function (cdf)* $F_X(x)$, is defined as

$$F_X(x) = \text{probability that } (X \leq x)$$

Note that any $F_X(x)$ has a range of $[0,1]$ (sometimes asymptotically) and will always be increasing (or remain constant over some intervals) as x increases from $-\infty$ to $+\infty$ [Knuth 47].

Let X take on the values of the sequence generated by a RNG (we essentially did the same thing in our discussion of the chi-square test, except we used Y above). For the KS test, we require that $F_X(x)$ be continuous. This is the exact opposite of what we did above, where the $F_Y(y)$ would have been nothing but 'jumps' since Y was only allowed to take on certain discrete values. The KS test deals with a different situation, where the numbers generated by a RNG are allowed to take on any value within a certain interval, leading to a continuous cdf. This $F_X(x)$ is the theoretical distribution we would expect our RNG to have [Knuth 45, 51].

Suppose we make n independent observations of X, creating the values $X_1, X_2, ..., X_n$. We define the *empirical distribution function* $F_n(x)$ as

$$F_n(x) = \frac{\text{number of } X_1, X_2, ..., X_n \text{ that are} \leq x}{n}$$

The KS test compares $F_X(x)$ to $F_n(x)$ by measuring the difference between the two distribution functions. When n is sufficiently large, we would expect the two functions to be similar if the sequence we're examining is truly random. We therefore measure the difference between the two functions by forming the following statistics:

$$K_n^+ = \sqrt{n}\max(F_n(x) - F_X(x)) \ , \ -\infty < x < +\infty$$

$$K_n^- = \sqrt{n}\max(F_X(x) - F_n(x)) \ , \ -\infty < x < +\infty$$

In words, K_n^+ is the greatest deviation when F_n is greater than F_X, and K_n^- is the greatest deviation when F_n is less than F_X. (The $\sqrt{n}$ factor is present because, for fixed x, the standard deviation for $F_n(x)$ is proportional to $1/\sqrt{n}$, so multiplying by $\sqrt{n}$ allows K_n^+ and K_n^- to be independent of n) [Knuth 47]. We use these statistics by comparing them to the following table in a manner similar to the chi-square test [Knuth 48]:

TABLE 2
Selected Percentage Points Of The Distributions Of K_n^+ And K_n^-

	$p = .01$	$p = .05$	$p = .25$	$p = .50$	$p = .75$	$p = .95$	$p = .99$
$n = 1$	0.01000	0.05000	0.2500	0.5000	0.7500	0.9500	0.9900
$n = 2$	0.01400	0.06749	0.2929	0.5176	0.7071	1.0980	1.2728
$n = 3$	0.01699	0.07919	0.3112	0.5147	0.7539	1.1017	1.3589
$n = 4$	0.01943	0.08789	0.3202	0.5110	0.7642	1.1304	1.3777
$n = 5$	0.02152	0.09471	0.3249	0.5245	0.7674	1.1392	1.4024
$n = 6$	0.02336	0.1002	0.3272	0.5319	0.7703	1.1463	1.4144
$n = 7$	0.02501	0.1048	0.3280	0.5364	0.7755	1.1537	1.4246
$n = 8$	0.02650	0.1086	0.3280	0.5392	0.7797	1.1586	1.4327
$n = 9$	0.02786	0.1119	0.3274	0.5411	0.7825	1.1624	1.4388
$n = 10$	0.02912	0.1147	0.3297	0.5426	0.7845	1.1658	1.4440
$n = 11$	0.03028	0.1172	0.3330	0.5439	0.7863	1.1688	1.4484
$n = 12$	0.03137	0.1193	0.3357	0.5453	0.7880	1.1714	1.4521
$n = 15$	0.03424	0.1244	0.3412	0.5500	0.7926	1.1773	1.4606
$n = 20$	0.03807	0.1298	0.3461	0.5547	0.7975	1.1839	1.4698
$n = 30$	0.04354	0.1351	0.3509	0.5605	0.8036	1.1916	1.4801

We read the table as we did with the χ^2 test: the probability is 95 percent that K_6^+ will be less than 1.1463, and since the distribution is the same for both K_n^+ and K_n^-, we could make the same statement for K_6^-. Unlike the chi-square test, the numbers in the above table are not approximate values but are exact (within the rounding error) [Knuth 48].

In addition, like the chi-square test, we need to be careful about our selection of n. n should be big enough so that we detect if the distribution functions $F_n(x)$ and $F_X(x)$ are significantly different, and yet n being too large will usually 'smooth out' local nonrandomness. Knuth recommends n to be around 1000 (obviously requiring a more extensive table) [Knuth 49]. Knuth also describes a procedure where a fairly large number of K_{1000}^+ statistics are calculated and then a KS test is made on *these* observations, thereby detecting both global and local nonrandomness (see p. 49 for details).

We have already seen some of the differences between the chi-square and KS tests (namely that the chi-square uses discrete elements while the KS works with a continuous sample space, although certainly a continuous sample space can be 'broken up' into distinct intervals, allowing for a chi-square test), but it is worth noting that they can be used together [Knuth 50, 51]. In our discussion of the chi-square test, we came up with a fairly inadequate way of describing a sequence as "nonrandom," "suspect," "almost suspect" or supposedly "random." Now we have a better procedure. Make m independent χ^2 tests on different parts of a random sequence and record the values $V_1, V_2, ..., V_m$. Then apply a KS test to these V_i's (F_m here is described by the plotted values of each V_i, and F_V could be found from an extension of Table 1). This is certainly better than our comparatively arbitrary previous method [Knuth 50, 51].

It's that time again in the section where the interested reader might want to know the technical details of why the KS test works, and once again the reader is referred to Knuth's discussion of those details (see p. 54-56). For now, we will accept the validity of both the chi-square and KS tests, both individually and when used together, and use them as the building blocks of the empirical tests to follow.

Empirical Tests

Now that the chi-square and KS tests are in place, we can move to a discussion of empirical tests used to determine the 'randomness' of a sequence. Again, empirical tests are used on a sequence produced by a (proposed) RNG and don't require knowing exactly how the RNG operates. We will expand the sequence notation introduced earlier. $[U_n]$ refers to the sequence $U_0, U_1, U_2, ...$ of real numbers between zero and one, supposedly independently and uniformly distributed throughout that interval. $[Y_n]$ refers to the sequence $Y_0, Y_1, Y_2, ...$ of integers between 0 and $d-1$ (with d an integer), using the rule

$$Y_i = \lfloor dU_i \rfloor$$

and therefore preserving the same properties of being independently and uniformly distributed, only that here this distribution is over the integers between 0 and $d-1$ [Knuth 59].

Our purpose here is to briefly describe some of the empirical tests used. Knuth does this in his book (p. 59-73), and the following material is a summary of what he presents there. The reader should be aware that Knuth offers more detail and probably better descriptions of the following tests; however, this paper is a summary of my research over these past weeks and therefore includes the following material. Each test presented here will also include the pages where additional material can be found in Knuth's book. With this understanding, and the definitions above, we proceed to describe some of the empirical tests used to determine the 'randomness' of a sequence.

1. *Equidistribution or Frequency Test* [Knuth 59]

With an understanding of the material previously covered, this is the simplest of the empirical tests. Given a sequence $[U_n]$, we would require that its elements be uniformly distributed between zero and one. We can apply the KS test, with the obvious $F_X(x) = x$ for $0 \leq x \leq 1$ (as it would have to be for the sequence to have a uniform distribution; i.e., the probability that $U_i \leq 2/3$ had better be $2/3$). Alternatively, select any integer d and use $[Y_n]$. Now we can get categories to use in a chi-square test. For every integer r such that $0 \leq r < d$, count the number of times that $Y_i = r$ for $0 \leq i < n$. Then each such integer r defines a chi-square category, and we would use the chi-square test with $k = d$ (since there are d integers between 0 and $d-1$) and $p_r = 1/d$ (in order to be uniformly distributed). This amounts to saying that, for example, if we had a sequence of zeros and ones generated by a RNG, we would expect to get about as many zeros as we would ones if we were to take a sample of the sequence.

2. *Serial Test* [Knuth 60]

The equidistribution test examines whether an individual element in a sequence comes up more often than it should, but we would also like pairs of successive numbers to be uniformly and independently distributed (for example, in a binary sequence, (0,0), (0,1), (1,0) and (1,1) should all be equally likely). As Knuth eloquently puts it, "the sun comes up just about as often as it goes down, in the long run, but this doesn't make its motion random" [60]. To do this, count the number of times that the pair $(Y_{2j}, Y_{2j+1}) = (q, r)$ occurs for $0 \leq j < n$. This should be done for each pair of integers (q, r) with $0 \leq q, r < d$. We then have d^2 categories for a χ^2 test, with a probability of $1/d^2$ assigned to each category.

Instead of just pairs, this method can be generalized to triples, quadruples and so on. The choice of d has to be a careful one, however, to meet Knuth's 'five in each category' requirement for the chi-square test, and this gets more difficult as we move to larger subsequences. In practice, some of the following tests are used in place of a generalized serial test.

We also note in passing that $2n$ elements of a sequence are needed to make n observations. This is required; if we were to perform the serial test on the pairs $(Y_0, Y_1), (Y_1, Y_2), ..., (Y_{n-1}, Y_n)$, our observations are by no means independent, and independent observations are required for the χ^2 test.

3. *Gap Test* [Knuth 60, 61]

For each U_j in a certain range, this test examines the length of the 'gap' between this element and the next element to fall in that range, and hence the name of the test. So, if α and β are two real numbers such that $0 \leq \alpha < \beta \leq 1$, we're looking for the length of consecutive subsequences $U_j, U_{j+1}, ..., U_{j+r}, U_{j+(r+1)}$ such that U_j and $U_{j+(r+1)}$ are between α and β but the other elements in the subsequence are not (this is a gap of length r). We would then perform a χ^2 test on the results using the different lengths of the gaps as the categories, and the probabilities as follows:

$$p_0 = p,\ \ p_1 = p(1-p),\ \ p_2 = p(1-p)^2,\ ...\ ,\ \ p_k = p(1-p)^k,\ ...$$

Here $p = \beta - \alpha$, which is the probability that any element U_j is between α and β. Knuth gives the details of an algorithm to record the lengths of consecutive gaps.

4. *Poker Test* [Knuth 62]

As with the gap test, the name of the poker test suggests its description. We examine n groups of five consecutive integers $(Y_{5j}, Y_{5j+1}, Y_{5j+2}, Y_{5j+3}, Y_{5j+4})$ for $0 \leq j < n$ and put each of these groups into one of the following categories:

All different: *abcde*
One pair: *aabcd*
Two pairs: *aabbc*
Three of a kind: *aaabc*
Full house: *aaabb*
Four of a kind: *aaaab*
Five of a kind: *aaaaa*

A chi-square test is applied to these seven categories, although less likely categories are sometimes grouped together in order to meet the 'five in each category' requirement. Knuth derives the necessary probabilities in his book, and if categories were to be combined, their respective probabilities would simply be added together. Again, the groups of integers are not allowed to overlap in order to preserve the independence required by the χ^2 test.

5. *Coupon Collector's Test* [Knuth 62, 63]

Using the integer sequence $[Y_n]$, this test examines the lengths of segments $Y_{j+1}, Y_{j+2}, ..., Y_{j+r}$ needed to get the complete set of integers from 0 to $d-1$. The test is named because, as Knuth states, "we may think of a boy collecting d types of coupons, which are randomly distributed in his breakfast cereal boxes; he must keep eating more cereal until he has one coupon of each type" [63]. (I suppose he wouldn't have to *eat* the cereal, but at least open the box.) The lengths of these sequences make good categories for (guess what?) the chi-square test. Knuth describes the algorithm for implementing this test, as well as deriving the necessary probabilities.

6. *Permutation Test* [Knuth 64]

This test examines a $[U_n]$ sequence (that is, the elements are real numbers between zero and one). We take a portion of the sequence and divide it into n groups, each with t elements. We now have groups of the form $(U_{jt}, U_{jt+1}, ..., U_{jt+(t-1)})$ with $0 \leq j < n$. In these groups, we assume each element is distinct from the others (a valid assumption, since the probability of any two elements being equal is zero) and put each group into a category depending on its relative ordering. For example, with $t = 4$, the elements of the group could be such that $U_{4j} < U_{4j+1} < U_{4j+2} < U_{4j+3}$ or $U_{4j} < U_{4j+2} < U_{4j+3} < U_{4j+1}$ or any of the other possibilities. In this example, there are 4! possible relative orderings; in general, there are $t!$ possibilities. A chi-square test is then performed with $k = t!$ and a probability of $1/t!$ for each ordering (since each should be equally likely in a random sequence). Knuth gives an algorithm for this test as well.

7. *Run Test* [Knuth 65-68]

For this test, we need a preliminary definition. A *monotone* sequence is a sequence that has elements that are either all increasing or all decreasing. For example, $1, 3, 6, 8$ and $9, 6, 5, 4, 2, 0$ are both monotone sequences, while $1, 3, 7, 2, 9, 4$ is not (although it contains monotone subsequences). Here we test a sequence for its 'runs up' and 'runs down' (again, we see the descriptive test name); that is, we examine the lengths of the sequence's monotone subsequences.

As an example, we divide the sequence $1, 3, 8, 7, 5, 2, 6, 7, 1, 6$ into 'runs up' as follows:

$$1, 3, 8|7|5|2, 6, 7|1, 6$$

and we have a run of length 3, followed by two runs of length 1, another run of length 3 and a run of length 2.

One would initially think (especially after the pattern of the previous tests) that the lengths of these runs could be categories for a χ^2 test. This is not the case, however, since long runs tend to be followed by short runs, which conversely tend to be followed by long runs (we're working either in a finite interval of real numbers or a finite collection of integers in our sequences, and it would be unlikely to have consecutive long 'runs up,' for example). This means that consecutive observations are not independent from each other, which invalidates the use of a chi-square test. Instead, we need to use a more complicated statistic which is defined and derived in Knuth's book. We'll leave that discussion out here, but at least we understand the basic idea behind the test.

8. *Maximum-of-t Test* [Knuth 68]

This is the last empirical test we'll describe here. The idea is that we break a sequence up into subsequences of equal length, take the maximum value of each subsequence and apply the KS test (which finally returns) to those maximum values.

For $0 \leq j < n$, define V_j by the rule

$$V_j = \max(U_{tj}, U_{tj+1}, ..., U_{tj+(t-1)})$$

(here we've broken the original sequence $[U_n]$ into subsequences of length t). We now have a sequence $V_0, V_1, ..., V_{n-1}$. We apply a KS test to these values, using $F_X(x) = x^t$ with $0 \leq x \leq 1$ as our cdf for the comparison.

Let's verify that this is the valid cdf. We're looking for the cdf of each V_j. For a V_j, the probability that $V_j = \max(U_{tj}, U_{tj+1}, ..., U_{tj+(t-1)}) \leq x$ is the probability that $U_{tj} \leq x$ and $U_{tj+1} \leq x$ and ... and $U_{tj+(t-1)} \leq x$, which is the product of the individual probabilities. So

$$F_X(x) = xx...x = x^t$$

and our KS test is valid.

This is by no means an exhaustive list of the empirical tests used, and in fact Knuth himself includes a few more (p. 68-71), but it gives a feel for the different kinds of empirical tests used for RNG's. Most of these tests can also be used on subsequences of the original sequence instead of just the elements of the sequence [Knuth 71], adding further to our testing capabilities. And, as might be suspected, some of these tests are better than others. For example, the equidistribution and serial tests tend to be 'weaker' (since most 'random' sequences will pass them), while the run test tends to be 'stronger' in this sense [Knuth 72].

We leave the empirical tests for now and examine the theoretical spectral test.

The Spectral Test

As mentioned before, theoretical tests differ from empirical tests in that they can predict how good a RNG will be without actually using a sequence generated by the RNG. In addition, theoretical tests give us more insight about the RNG itself, so they are better than empirical tests when they exist. Obviously, however, these tests cannot be used for every RNG (unlike the empirical tests), they are usually hard to find, and they tend to be considerably more complicated than empirical tests [Knuth 75]. For these reasons, and others, this paper has so far focused on empirical tests and will avoid discussion of theoretical tests except for the unavoidable spectral test, "by far the most powerful test known" [Knuth 89]. A paper on the testing of RNG's would be glaringly incomplete without at least a brief overview of this test, and that's exactly what we'll provide here. Knuth, of course, goes into the full details and is worth reading (p. 89-113). As always, the following material comes from Knuth's book.

The spectral test has its roots in a 1959 paper by N. M. Korobov, and was developed and improved upon by many authors in multiple papers throughout the 1960's and 1970's [Knuth 110]. It is a powerful test because "all good generators pass this test [and] all generators now known to be bad actually fail it" [Knuth 89]. The test deals with LCG's. Recall that a LCG is a pRNG defined by the following:

Select the following four integers:
m, the modulus, with $m > 0$
a, the multiplier, with $0 \leq a < m$
c, the increment, with $0 \leq c < m$
x_0, the seed, with $0 \leq x_0 < m$
and define the sequence $[x_n]$ of 'random numbers' by the recursive

$$x_{i+1} = (ax_i + c) \ \% \ m$$

Recall also that the maximum possible period length of a LCG is m if $c \neq 0$ or $m - 1$ if m is prime and $c = 0$. We will assume our LCG has the maximum period length possible throughout this section.

The spectral test examines the entire period of the LCG (and other theoretical tests examine the entire periods of pRNG's in general). In fact, very few theoretical results have been proven about LCG's that deal with less than their full period (and even fewer of these results are actually useful; see Knuth (p. 109-110) for one of them) [Knuth 75]. Since this type of test examines the entire period, it can detect *global* nonrandomness, while the empirical tests examine parts of the sequence and therefore can detect *local* nonrandomness. Obviously, being able to do both is nice, but we have to remember that theoretical tests don't exist in every situation.

For the spectral test, we are working with consecutive elements of a $[U_n]$ sequence (real numbers between zero and one), but we assume here that the sequence is periodic with period m and can be described by a LCG. We want to look at the set of all m points in t-dimensional space

$$\{(U_n, U_{n+1}, ..., U_{n+(t-1)})\}$$

The fact that we're not working with integers may seem to conflict with our statement that we're working with LCG's, but note that if we truncate the value of each U_i by a constant amount (to a certain number of decimal places), we can use a LCG to express this set of points. We need this LCG either to have period m, or we must add the point (0,0,...,0) to the above set if our LCG has period $m - 1$ (this addition does not affect the results of the test in any noticable manner when m is large, as it should be). Given this, we can express the above set of points as

$$\left\{ \frac{1}{m}(x, s(x), s(s(x)), ..., s^{t-1}(x)) \mid 0 \leq x < m \right\}$$

where

$$s(x) = (ax + c) \ \% \ m$$

(The division by $1/m$ above is similar to our definition of $[Y_n]$, only we're 'going backwards.')

Again, we are concerned here with the set of points in t-dimensional space, and not the order in which they are generated. We plot this set of points in t-dimensional space, and of course this can be visualized in 2 and 3 dimensions (the reader is urged to see a visual representation of this process for one LCG in Knuth (p. 90) and for many different LCG's (most of which are actually used in practice) at the pLab project (*http://random.mat.sbg.ac.at*)). Once these points are plotted, we can analyze them with the spectral test.

Certainly, with any finite set of points in a finite area (the unit square, the unit cube, etc.), we can 'cover' all of the points using a finite collection of parallel lines in 2-dimensional space, or a finite collection of parallel planes in 3-dimensional space, or a finite collection of $(t-1)$-dimensional hyperplanes in t-dimensional space. Let $1/\nu_2$ be the maximum distance between lines taken over all sets of parallel lines that cover all of the points $\{(x/m, s(x)/m)\}$ in two dimensions. Similarly, let $1/\nu_3$ be the maximum distance between planes taken over all sets of parallel planes that cover all of the points $\{(x/m, s(x)/m, s(s(x))/m)\}$ in three dimensions. Extend this in an analogous manner for t-dimensional space; that is, $1/\nu_t$ is the maximum distance between $(t-1)$-dimensional hyperplanes over all sets of such hyperplanes that cover all of the points $\{(x/m, s(x)/m, ..., s^{t-1}(x)/m)\}$ in t dimensions. Then we call ν_2 the two-dimensional *accuracy* of the RNG, and, in general, ν_t is the t-dimensional accuracy of the RNG [Knuth 90, 91].

Now, the point of the above discussion is that the accuracy of truly random sequences is the same in all dimensions, while the accuracy of a periodic sequence decreases as the dimension increases. The spectral test, then, compares the values of the accuracy of a sequence in different dimensions to determine how 'random' the sequence is. (It seems that not much changes when $t \geq 10$, which is good from a computational standpoint) [Knuth 91]. The theory behind this (which really isn't too complicated, involving manipulations of vectors), in addition to an algorithm for computing such accuracies, is described in Knuth's book (p. 92-101), but will not be included here, except that we state without proof that

$$\nu_t = \min\left\{\sqrt{x_1^2 + ... + x_t^2} \mid x_1 + ax_2 + ... + a^{t-1}x_t = 0 \bmod m\right\}$$

with $2 \leq t$, $0 < a < m$ and a relatively prime to m [Knuth 98].

Again, this is a complicated test, but it's the most powerful test to use if the sequence meets the requirements of our assumptions. We are also able to examine the entire sequence without generating a full period, which allows us to search for global nonrandomness, in addition to local nonrandomness by means of empirical tests.

Conclusion

A lingering question for the reader might be: why are there so many tests? (If not, try writing a paper about testing RNG's and see if you don't eventually ask yourself that question.) There are two reasonable answers. First, a bad RNG can pass certain tests, so trying more than one test is essential for making sure that a bad RNG is detected as such. For example, the nonrandom sequence

1010101010101010101010101010101010101010101010101010101010

passes the equidistribution test with a perfect score, but miserably fails the serial test. Secondly, in a related sense, we can't consider a RNG to be 'good' unless it passes multiple tests (probably at least five or six). So, we need a number of different tests to sort out the bad RNG's from the good [Knuth 73].

The intent of this paper was to introduce the reader to the testing of random number generators. In so doing, we have discussed some definitions of randomness, different ways to generate random numbers, and applications for these numbers. We then introduced the reader to around a dozen tests, beginning with the foundational chi-square and Kolmogorov-Smirnov tests, then a variety of empirical tests, followed by a theoretical test, the most powerful of them all, the spectral test. It is hoped that the reader has developed an appreciation of this subject and has recognized the importance of many different aspects of mathematics involved in this area.

Finally, if you've read the entire paper, you know how important and fundamental Knuth's book is to this subject. The reader is once more strongly encouraged to read and study the third chapter of this work. The reader (as well as the author) would greatly benefit from doing so.

Personal

I currently attend Concordia College in Moorhead, MN, and am majoring in mathematics. My thanks to Concordia's Department of Mathematics and Computer Science for making me aware of this opportunity and especially to Dr. Forde and Dr. Tomhave for writing letters of recommendation. Thanks also to Dr. Garrett for his notes, lectures and help during these past weeks.

Bibliography

Bennett, Deborah J. *Randomness.* Cambridge: Harvard University Press, 1998.

Garrett, Paul. *Introduction to Cryptography.* Notes. 2000.

Haahr, Mads. *random.org.* 30 June 2000. <*http://www.random.org*>.

Hellekalek, Peter. *The pLab Project.* <*http://random.mat.sbg.ac.at*>.

Knuth, Donald E. *The Art of Computer Programming.* vol. 2. 2 ed. Reading, MA: Addison-Wesley, 1981.

Lavarand. Silicon Graphics, Inc. 1998. <*http://lavarand.sgi.com*>.

Li, Ming and Paul Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications.* New York: Springer-Verlag, 1993.

The Royal Windsor Society for Nursing Research. *Chi Square Research Analysis.* < *http* : //*www.windsor.igs.net*/˜*nhodgins*/*chi_square_research_analysis.html* >.

Walker, John. *HotBits.* 10 June 2000. <*http://www.fourmilab.ch/hotbits/*>.