On the Randomness of Pi and Other Decimal Expansions

George Marsaglia
Department of Statistics
Florida State University

Abstract

Tests of randomness much more rigorous than the usual frequency-of-digit counts are applied to the decimal expansions of π, e and $\sqrt{2}$, using the Diehard Battery of Tests adapted to base 10 rather than the original base 2. The first 10^9 digits of π, e and $\sqrt{2}$ seem to pass the Diehard tests very well. But so do the decimal expansions of most rationals k/p with large primes p. Over the entire set of tests, only the digits of $\sqrt{2}$ give a questionable result: the monkey test on 5-letter words. Its significance is discussed in the text.

Three specific k/p are used for comparison. The cycles in their decimal expansions are developed in reverse order by the multiply-with-carry (MWC) method. They do well in the Diehard tests, as do many fast and simple MWC RNGs that produce base-b 'digits' of the expansions of k/p for $b = 2^{32}$ or $b = 2^{32} - 1$. Choices of primes p for such MWC RNGs are discussed, along with comments on their implementation.

1 Introduction

This article may be considered to have two themes:

1. Assessing apparent randomness of familiar decimal expansions such as those of $\pi, e, \sqrt{2}$ as well as those of not so familiar expansions of rationals k/p for large primes p, using far more extensive testing than heretofore.

2. Describing simple and fast ways to get decimal or base-b expansions for such k/p because, along with π, e and $\sqrt{2}$, they seem to produce digits that serve very well for use as a set of independent, identically distributed random digits for Monte Carlo applications.

Methods for producing, in reverse order, digits of the expansions of k/p can lead to very fast and simple random number generators (RNGs), with immense periods and performances matching those for digits of π, e and $\sqrt{2}$, but more generally for other bases, particularly for bases 2^{32} or $2^{32}-1$, the latter chosen because a large prime p cannot have $b = 2^{32}$, a square, as a primitive root.

2 The randomness of pi

Versions of the long history of π, and dates for the record numbers of digits, (51.5×10^9 as of Jan. 2005) are available in many forms via web searches. A particularly popular one is π *through the ages* [7]. Many websites have taken all or parts of it, often without attribution, to comment on an apparent paucity of 7s among the digits of π published by Shanks[8] in 1853. The book *A Budget of Paradoxes* by De Morgan[2] is the primary source for the comments. A search of the web for the phrase "a curious statistical freak" will show at least 21 references containing all or part of this quotation from π *through the ages*[6]:

> "Very soon after Shanks' calculation a curious statistical freak was noticed by De Morgan, who found that in the last of 707 digits there was a suspicious shortage of 7s. He mentions this in his Budget of Paradoxes of 1872 and a curiosity it remained until 1945 when Ferguson discovered that Shanks had made an error in the 528th place after which all his digits were wrong. In 1949 a computer was used to calculate π to 2000 places."

Unfortunately, all 21 or more of those websites got things wrong. The 607 digits from Shanks[8], (actually 608 because the leading 3 was not counted), were based on the series for $\frac{\pi}{4} = \arctan(\frac{1}{5}) - 4\arctan(\frac{1}{239})$ and accumulated over a period of many years. Shanks [9] later amended his list and extended it to 707 digits. The first publication led to the observation by De Morgan that 7 occurred only 44 times in that list of 608. De Morgan, (page 65, second edition, V2) writes

> "It is 45 to 1 against the number of 7s being as distant from the probable value (say 61) as 44 on one side or 78 on the other. There must be a reason why the number 7 is thus deprived of its fair share in the structure."

So the observed 44 7s were from the 608 digits, not from Shanks' later list of 707+1. De Morgan apparently applied the binomial distribution, known from de Moivre 150 years before, to the observed number of 7s. For Shanks' list, the digit with the least count happened to be 7, and the most frequent was 3, leading to De Morgan's witty debunking of conclusions relating to $3\frac{1}{7}$ becoming the nearest simple-digit approximation to π, with allusions to number mysticisms of the Great Pyramid.

A more appropriate modern assessment of the frequencies in Shanks' 608 digits might be based on the point that it is unreasonable to single out 7, then apply the binomial distribution to the number of 7s. A better assessment, which is more difficult but no doubt available to De Morgan, is to assess the chances that the least-frequent digit will appear 44 or fewer times. There is a better than 10% chance that in 608 independent choices of a random digit, the count for the least-frequent digit will be 44 or less.

The paucity of 7s might have been proper cause for concern if based an a list of 608 digits, bumping De Morgan's 45 to 1 or the more modern 10% to more disturbing levels. If, in De Morgan's style, we merely apply the binomial distribution to counts for the number of 7s in a random list of 708 digits, then it is 2700 to 1 that the number of 7s will be as distant from the probable value (say 71) as 44 on one side or 98 on the other. Or, not singling out 7, it is about 500 to 1 against the least-frequent digit count being 44 or less when drawing 708 random digits.
But De Morgan's count of 44 7s was based on 608, not 708 digits, and those 21+ websites got it wrong.

Also contrary to some of the web histories of π, Ferguson[1] found Shanks' error by using a combination of hand and mechanical calculation, and not by means of an electronic computer.

When the list of digits of π is extended to the 51.5×10^9 digits available as of Jan 2005, the frequency of 7s, and of all the other digits, seems consistent with the assumption that they came from a sequence of id (identically distributed) variates taking values 0,1,2,3,4,5,6,7,8,9 with probabilities 1/10.

But is it reasonable to view those digits as the realization of a sequence of iid, **independent** identically distributed, random choices from {0,1,2,3,4,5,6,7,8,9}?. That is the question we address here, for $e, \sqrt{2}$ as well as π, and also for three specific, but more or less random, choices of decimal expansions for rationals k/p.

3 Source digits for the tests of randomness

Results mentioned in the next section, and detailed in an appendix, are based on six sources, each containing around 10^9 digits. Three files contain digits from the expansions of $\pi,\ e,\ \sqrt{2}$ and three from expansions of these rationals:

$$\frac{53480293019803\cdots}{44353 \times 10^{4608}+1},\quad \frac{3624360069}{7000000001},\quad \frac{123456789012}{1000000000061}.$$

The numerator in that first rational has over 4600 digits, too long to list. The digits for the $\pi, e, \sqrt{2}$ expansions were obtained by means of **PiFast**[3], a nice, easy-to-download-and-use program for getting many digits of important constants. The expansion for the three rationals were obtained by the reverse-order multiply-with-carry method described in Section 5. Each of the six files is too large to be included as attachments here, but could be reproduced by downloading and running the PiFast program, requiring 3-4 hours for each of $\pi, e, \sqrt{2}$, or less than a minute for C implementations of the methods of Section 5 for rationals k/p.

4 Test results

Results on thirteen tests of randomness are appended: (1-4) monkey tests for 3,5,6,7-letter words, (5-6) gcd and birthday spacings tests, (7) rank of mod 5 matrices, (8-9) placement of random points in squares and in cubes,

(10-11) overlapping permutation and runs tests, (12) a parking lot and (13) a squeeze test. In addition, frequent website references to the appearance of primes in π led to two more tests: for strings of 10^6 random digits, count the number of overlapping 3- and 4-tuples that represent primes.

Thus the appendix contains the results of 15 tests applied to the decimal expansions of

$$\pi,\ e,\ \sqrt{2},\ \frac{53480293019803\cdots}{44353\times 10^{4608}+1},\ \frac{3624360069}{7000000001},\ \text{and}\ \frac{123456789012}{1000000000061}.$$

Except for the two count-the-primes, the tests are based either on the Diehard Battery[4] or on some more recent tests[5], but applied to a sequence of digits rather than a sequence of bits. Those tests were selected because, at least in binary form, each of them has shown drastic departures from underlying probability theory for certain kinds of RNGs.

The idea behind each test is to see if the digits in question, whether viewed one at a time, or in segments or, more generally, as arguments in complicated functions, can be reasonably considered to have arisen as the output of those same functions applied to a sequence of iid random digits, iid meaning independent identically distributed.

If the output is designated X, and underlying probability theory provides the distribution function $\Pr(X < x) = F(x)$, then the random variable $p = F(X)$ should be uniformly distributed in [0,1), and the test returns that as a p-value. Often, this is repeated 10,20,50 or more times, and two functions of the resulting p-values—the Anderson Darling and Kolmogorov tests of uniformity—then provide a final pair of p-values, usually labeled ADp and Kp.

The tests and their results are pretty much self-explanatory—at least for those who understand the process of converting, by means of their distribution functions, the outputs of random experiments into uniform distributions in [0,1), and interpreting what, if anything, the results mean. For those who do not understand, I can only, as I do in responding to numerous e-mails seeking interpretation of results from Diehard, suggest that proper interpretation of tests of randomness probably requires more background in Probability and Statistics than the requestor has had.

This is not to say that, for those familiar with testing, there is agreement on either the tests to use or the interpretation of results. An example above illustrates this, for the simple case of finding 'only' 44 7s in Shanks' first 608 digits of π. De Morgan said chances were 45 to 1 against it. I would agree that it was a proper example of converting the binomial distribution, getting a p-value of about 45/46=.978, and would agree that, if that were the proper distribution to use, a p-value of .978 might lead one to suspect the uniformity of a process that produced the 608 random digits. And if the 44 7s came from a process that produced 708 random digits, I would say that the p-value of 2700/2701=.9996 is an even stronger reason for doubting the uniformity of the random process that produced them.

But I would argue that converting the lowest digit count to a p-value is more appropriate. In the above case for 608 digits, a lowest digit count of 44 would convert to a p-value of 0.891, not at all shaky. If 708 digits had been the proper total to use, p=.9977 would result from converting a lowest count of 44 to a uniform value, much stronger evidence for a lack of uniformity. In most cases, we would, as was done in the case of π, call for a more extensive examination of the process producing the digits, to see if early doubts on the uniformity were justified. Occasional p-values near 1 should be expected—indeed, required from myriad tests.

As I stated in the instructions for Diehard, keep in mind that "p happens".

Among the many hundreds of p-values reported in the appendix, there was only one case that might raise an eyebrow—the monkey test for 5-letter words on the digits of $\sqrt{2}$. In this test, we imagine the ($\sqrt{2}$) monkey is randomly striking a keyboard with ten keys and find, after a string of 10 million keystrokes, how many times each 5-letter word (5-digit number) appears.

If $Q_5 = \sum(\text{OBS}-\text{EXP})^2/\text{EXP}$ for 5-letter words, $Q_4 = \sum(\text{OBS}-\text{EXP})^2/\text{EXP}$ for 4-letter words, then $Q_5 - Q_4$ should have a chisquare distribution with $10^5 - 10^4 = 9000$ degrees of freedom, which is converted to a uniform distribution, a p-value. The test is repeated 50 times, then Anderson-Darling and Kolmogorov tests for uniformity are made on the 50 ps.

Individually, the 50 p-values from the digits of $\sqrt{2}$ seem OK, but collectively they give suspect p-values from tests for uniformity: ADp = 0.997296 for Anderson-Darling and Kp = 0.998832 for Kolmogorov. This illustrates the uncertainty in assessing randomness. If the digits of the expansion of $\sqrt{2}$ can serve as the realization of a sequence of iid random digits, then there must be occasional stretches where such a set of p-values will produce an ADp and Kp value around .998, and we happened to get one. Or perhaps there is some kind of regularity in the digits of $\sqrt{2}$ that will consistently show aberrant behavior for the frequency of 5-letter words (5-digit strings). I leave further

exploration to the reader, an obvious step being to take the second set of 10^9 digits from the expansion of $\sqrt{2}$ and see if it fares better or no-better in the monkey test for 5-letter words. I predict it will do better.

5 Decimal expansion of rationals

In 1863, two MA's from Cambridge, G. Suffield and J.R.Lunn, published the cycle of 7698 digits in the decimal expansion of 1000/7699. This is mentioned as one of numerous 'curious calculations' in the *Budget of Paradoxes* [2], the most significant being Shanks' arduous expansion of π mentioned above. Details of the 1000/7699 expansion are not supplied, other than to say that it was done to illustrate Suffield's method of 'synthetic division'.

The period of the expansion of k/p for a prime p is the order of 10 for p, and 10 happens to be a primitive root for $p = 7699$, so the expansion is periodic with period $p - 1 = 7698$. The first cycle of 7698 digits begins, then ends like this:

$$\frac{1000}{7699} = .129886998311469021950902714638264709702558773866 7\cdots$$
$$\cdots 937524353812183400441615794258994674633069229770100 0\ 12988\cdots$$

Examination of long-division procedures that we still learn in grammar school suggests that the procedure used by Suffield and Lunn might be reduced to repeated application of these simple rules, starting with an initial $k = 1000$:

$$\text{Form } d = \lfloor \frac{10k}{7699} \rfloor;\ \text{Form } k = 10k - 7699d;\ \text{Output } d;$$

The full cycle, $12988699831\cdots7701000$, would result from 7698 iterations, the first few of which are:

$$d = 1, k = 2301;\ d = 2, k = 7612;\ d = 9, k = 6829;\ d = 8, k = 6698;\ d = 8, k = 5388;\ d = 6, k = 7686;$$

Diligent application of those iterations—find the new d as the integer part of $10k/p$ and the new k as $10k \bmod p$—might produce the full 7698 digit expansion in a week or two, but divisions by p and reductions mod p are not as easy as divisions by 10 and reductions mod 10.

I found this example of De Morgan's 'curious calculations' all the more curious because I had, as a consequence of some curious calculations for a different purpose, devised a way to get the digits of a decimal expansion using only these more desirable operations: multiplication by 10 and reduction mod 10, as in the following development for 1000/7699:

Given initial $x=9, y=2, z=1$, and a 'carry' $c=13$, iterate, using temporary t:

$$t = 7(x + y) + c,\ x \leftarrow y,\ y \leftarrow z,\ c \leftarrow \lfloor (t/10 \rfloor,\ z \leftarrow t \bmod 10,\ \text{output } z,$$

This iteration will produce the digits in the expansion of 1000/7699, but in reverse order!

For example, suppose the digits x, y, z and carry c are represented in this form: $x, y,^c z$, with the carry c in elevated form. The iteration rule is: for each elevated c, take 7 times the sum of the two preceding digits, add that to c, then write the result in elevated form, except place the trailing digit unelevated;

$$9, 2,^{13}1,^{9}0,^{3}0,^{1}0,^{0}1,^{0}0,^{0}7,^{0}7,^{4}9,^{10}2,^{12}2,^{8}9,^{3}6,^{8}0,^{11}3,^{5}3,^{2}6,^{4}4, \cdots$$

Thus $7 * (9 + 2) + 13 = 90$, which is written $^{9}0$. Then $7 * (2 + 1) + 9 = 30$, leading to $^{3}0$, and $7 * (1 + 0) + 3$ leads to $^{1}0$, $7 * (0 + 0) + 1$ leads to $^{0}1$ and so on. Mental arithmetic and a little practice would provide this output of z's:

$$0001077922960336476499852497516144004381218353425739818508400739 4\cdots,$$

which is the end of the first cycle of 1000/7699, but in reverse order. At a rate of perhaps 10/minute, the full output of 7698 z's might take around 13 hours.

This is an example of what I call the *multiply-with-carry* method, which, along with the `add-with-carry` method[6] was developed in response to a request that I provide a random number generator for The Connection Machine. That was, in the 80's, a new supercomputer with 65536 processors working in parallel. But those

processors were simple, void of the registers and integer arithmetic instructions usually available when developing RNGs.

So the problem was to create long sequences of bits, each new bit the sum, mod 2, of a few previous ones. The arithmetic behind such a scheme was too simple to provide sequences with long periods, unless one used the 'carry' resulting from a previous addition to be used in forming the new bit. A little blackboard experimenting with methods for determining the periods of such sequences led to the surprising fact that I would be creating the binary expansions of certain integers—but in reverse order. I never did get to providing a RNG specifically for the Connection Machine, but experimenting with the add-with-carry operation using conventional 32-bit CPUs and expansions to base $b = 2^{32}$ rather than $b = 2$ led to a variety of simple generators with astonishingly long periods [6].

Use of integer multiplication rather than addition led to multiply-with-carry sequences which produced, in reverse order, the base b expansions for rationals of the form k/p, preferably with p a prime having b as a primitive root. In addition to the immense periods, the generating procedure led, as above, to production of 'digits' in reverse order by means of operations particularly well suited for computer implementation: divide by b, reduce mod b. This in contrast to the divide by p, reduce mod p needed to get the 'digits' of k/p in the conventional order.

6 Choice of primes for RNGs based on the expansion of k/p

This section concentrates on the selection of primes p whose base-b expansions of k/p lead to quality RNGs. The development in Section 5 shows how the base-b digits can be generated in reverse order by means of the operations: divide by b and reduce mod b, rather than divide by p, reduce mod p for the conventional order. The obvious choice for b is 2^{32} for the integer arithmetic available in most CPUs, and the least complicated RNGs of this kind are multiply-with-carry (MWC) sequences

$$x_n = (ax_{n-r}+c_{n-1}) \bmod b,\ c_n = \lfloor \frac{ax_{n-r}+c_{n-1}}{b} \rfloor.$$

Note how well-suited the two operations, $(ax + c) \bmod b$ and $\lfloor (ax + c)/b \rfloor$, are for modern CPUs when $b = 2^{32}$. If we form $t = ax + c$ in 64 bits, then $(ax + c) \bmod b$ is just the bottom, and $\lfloor (ac + c)/b \rfloor$ the top, 32-bit part of t.

Such MWC sequences are based on primes of the form $p = ab^r - 1$, and require an initial set of seed values $x_1, x_2, \ldots, x_r$ randomly selected in $0 \le x < b$ and an initial seed c_r randomly chosen in $0 \le c < a$. There are ab^r ways to chose the seed set, and for every choice of such an initial seed set, the resulting x's will be, in reverse order, the digits of the expansion of k/p for some $0 \le k \le p$. The two extremes, $0/p$ and p/p, arise from seed sets with all x's and c set to 0, or all x's set to $b - 1$ and c set to $a - 1$. For those seed sets, the MWC sequence has period 1.

The ideal case is to have b a primitive root of p, so that, except for the two forbidden sets, every choice of seeds leads to a sequence of period $p - 1$, and thus, for $m \le r$, the RNG can produce every possible string of m successive digits, a highly desirable feature for assessing the behavior of points produced by the RNG in higher dimensions. Note, for example, that many commonly used RNGs have periods around 2^{32} and can produce at most $1/2^{32}$ of the possible 2-tuples, $1/2^{64}$ of the possible 3-tuples, etc., the reason why such RNGs fail many of the tests in Diehard. But we will have RNGs here that can produce, for example, every possible 1000-tuple, or 4000-tuple or higher.

Unfortunately, $b = 2^{32}$ is a square and cannot be a primitive root for $p = ab^r - 1$. The best we can do is choose p a *safeprime*, that is, a prime p for which $(p-1)/2$ is also prime. Then the period of the sequence will be $(p-1)/2$ and we still will produce all possible m-tuples for m's up to $(p - 1)/2$. Numerous safeprimes of the form $ab^r - 1, b = 2^{32}$ will be given in a separate article. The largest such is $3686175744b^{1359} - 1$, so every possible sequence of 1058 successive 32-bit integers can be produced by the MWC RNG based on that prime. The period of the MWC RNG based on that safeprime exceeds 10^{13100}.

Problems with developing MWC RNGs from larger primes come from both the difficulty in finding the prime p itself and in finding the order of b for that prime. Unless p is a safeprime, factoring of $p-1$ is required. To make that feasible, we turn to primes of the form $ab^r + 1$, $b = 2^{32}$. Such primes provide what I call complimentary-multiply-with-carry (CMWC)RNGs, with generating procedure

$$x_n = (b-1) - [(ax_{n-r}+c_{n-1}) \bmod b],\ c_n = \lfloor \frac{ax_{n-r}+c_{n-1}}{b} \rfloor.$$

The seed sets for CMWC are the same as those for MWC: choose r $x's$ with $0 \le x < b$ and an initial seed $0 \le c < a$, for which there are ab^r possible choices. But unlike MWC, for which one must avoid those two period-1 seed sets, for CMWC every choice of seeds produces sequences with the same period, the order of b.

One of the largest CMWC primes for which I have been able to establish the order of $b = 2^{32}$ is $p = 2169967b^{25000}+1$, for which every choice of the ab^{25000} seed sets provides a sequence with period $(p-1)/2^6 > 10^{240833}$, and every possible sequence of 24994 successive 32-bit integers appears somewhere in that sequence. And note also, for cryptographic purposes, that many such CMWC RNGs can have output that contains every possible 24000-tuple, so that anyone having, say, 10000 successive elements from the sequence will have no easy way to determine succeeding ones. I will, in the separate article mentioned above, provide numerous MWC and CMWC RNGs that have this property, as well as suggest ways to choose many more, making such code-breaking extrapolations difficult.

With $b = 2^{32}$, 2 itself cannot be primitive for primes of the form $p = ab^r + 1$. If the order of 2 is maximal, $e \le (p-1)/2$, then the order of $b = 2^{32}$ is $e/\gcd(e, 32) = e/32$, so the largest possible order of $b = 2^{32}$ for primes of the form $p = ab^r + 1$ is $(p-1)/2^6$, that is, we 'lose' at least six 2s from the exponent in p. Numerous examples for which we lose from six to ten 2s will be listed in that forthcoming article. Resulting periods are much greater than the already astonishing 10^{240833} above, but, as with the climbing of Mt. Everest and worldwide interest in large primes themselves, they are goals we seek 'because they are there'. A current goal is to find the order of 2^{32} for $p = 28433 \times 2^{7830457}+1$, the largest known non-Mersenne prime as of 1/2005.

6.1 Choice of $b = 2^{32}-1$ for providing primitive roots

The most desirable MWC and CMWC, from the standpoint of elegance of theory, are those based on the primes p for which b is a primitive root, with $b = 2^{32}$ singled out because it makes implementing the multiply-with-carry operation so simple in today's CPU's. Since $b = 2^{32}$ cannot be a primitive root for large primes, we turn to another choice that can be primitive, but will still have the property that computer implementation of the multiply-with-carry operations is simple. An obvious choice is $b = 2^{32}-1$. Our MWC or CMWC sequences will then produce integers x in $0 \le x < 2^{32}-1$ rather than $0 \le x < 2^{32}$, and the 'divide by b' and 'reduce mod b' operations can be based on slight modifications of those for 2^{32}.

Recall that, with $b = 2^{32}$, if $t = ax + c$ is formed in 64 bits then $\lfloor (t/b) \rfloor$ is the top 32 bits of t, and $t \bmod b$ is the bottom 32. Put another way, if $t = 2^{32}v + w$, with $v, w < 2^{32}$, then v and w are the top and bottom 32 bits, but $t = (2^{32}-1)v + (v + w)$ shows that $\lfloor \frac{t}{2^{32}-1} \rfloor = v + \lfloor \frac{v+w}{2^{32}-1} \rfloor$ and $t \bmod (2^{32}-1) = (v + w) \bmod (2^{32}-1)$, both corrections easily made, depending on whether $v + w$ overflows when added as 32-bit integers.

The following C function provides a CMWC RNG based on $p = 18782b^{4096} + 1$, the largest prime for which I have been able show that $b = 2^{32}-1$ is a primitive root.

```
static unsigned long Q[4096],c=362436;
unsigned long CMWC4096(void){
  unsigned long long t, a=18782LL,b=4294967295LL;
  static unsigned long i=4095;
  unsigned long x,r=b-1;
    i=(i+1)&4095;
    t=a*Q[i]+c;
    c=(t>>32); t=(t&b)+c;  if(t>r) {c++; t=t-b;}
    return(Q[i]=r-t);        }
```

The prime is $p = 18782 \times (2^{32}-1)^{4096} + 1$ and the period of this CMWC RNG is $p - 1 > 10^{39460}$. It requires 4096 seeds $0 \le x_i < 2^{32}-1$ and an initial seed $0 \le c < 18782$, and for any such seed set, the RNG produces, in reverse order, the base $b = 2^{32}-1$ digits of the expansion of k/p for some $0 < k < p$. Of course, because b is primitive, any two such expansions are just circular rotations of one another. An initial seed $c < 18782$ and the 4096 elements in the static array Q[4096] should be assigned values before calls to the function `CMWC4096()`, otherwise the first few thousand returned values will be zeros. (That is consistent with the view that the choice of seeds merely chooses a random starting point in a huge circle of over 10^{39460} base-$(2^{32}-1)$ digits. Failing to initialize the Q[4096] array (set

by default to 0's) merely provides a starting point at a long string of zeros, which should be occasionally encountered in any random string.)

That C procedure, CMWC4096(), has one of the longest periods of RNGs I have developed—over 10^{33000} times as long as that of the Mersenne twister, which many websites cite as the longest period RNG.

But of course the speed, simplicity and immense periods of such MWC and CMWC RNGs are not as important as the property whose apparent truth is the main theme of this article:

> *The digits in the expansion of irrationals such as π, e and $\sqrt{2}$, as well as those of rationals k/p for large primes p, seem to behave as though they were the output of a sequence of independent identically distributed (iid) random variables.*

References

[1] D Ferguson, *Evaluation of pi. Are Shanks' Figures Correct?*, Mathematical Gazette **30**, 1946, 89–90.

[2] A De Morgan, *A Budget of Paradoxes*, Longmans, Green, London., 1872, Originally published as a collection of De Morgan's miscellaneous notes and reviews, edited by his wife after De Morgan's death in 1871. Then, "budget" meant a purse or other depository, and "paradox" was used in the sense of *not orthodox*, so the title might be interpreted today as "A collection of views contrary to universal opinion". Specifics here are taken from the more readily available second edition, published in two volumes in 1915 byThe Open Court Publishing Company, London and edited by David Eugene Smith.

[3] X Gourdon, *PiFast, an easy-to-use package for computing pi and other irrationals to large numbers of digits*, http://www.numbers.computation.free.fr/Constants/PiProgram/pifast.html. 1999.

[4] G Marsaglia, *The Marsaglia Random Number CDROM, with The Diehard Battery of Tests of Randomness,* produced at Florida State University under a grant from The National Science Foundation, available at http://www.cs.hku.hk/∼diehard/ or an earlier version at http://www.stat.fsu.edu/pub/diehard, 1985.

[5] G Marsaglia and WW Tsang, *Some difficult-to-pass tests of randomness*, Journal of Statistical Software **7, issue 3,** 2002.

[6] G Marsaglia and A Zaman, *A new class of random number generators*, Annals of Applied Probability **1**, 1991, 462–480.

[7] J O'Conner and EF Robertson, *π through the ages*, 2001, www-groups.dcs.st-and.ac.uk/∼history/HistTopics/Pi_through_the_ages.html.

[8] W Shanks, *Contributions to Mathematics Comprising Chiefly the Rectification of the Circle to 607 Places of Decimals*, G Bell, London, 1853.

[9] ———, *On the Extension of the Numerical Value of pi*, Proceedings of the Royal Society of London **21**, 1873, 315–319.

7 Appendix

This appendix gives results from 15 tests of randomness for $\pi, e, \sqrt{2}$ and three k/p. The first 13 are modifications from the Diehard Battery[4], or from the tough tests in [5], and have shown unsatisfactory results from many proposed RNGs.

Monkey tests for k-letter words. A monkey randomly strikes keys on a typewriter having ten different keys. For each possible k-letter word, we count how many times that word appears in a string of n keystrokes. If $Q_k = \sum(\text{OBS-EXP})^2/\text{EXP}$ for the counts for k-letter words, then $Q_k - Q_{k-1}$ is the quadratic form in the weak inverse of the covariance matrix of the joint mean-adjusted k-letter word counts, and is asymptotically chisquare distributed with $10^k - 10^{k-1}$ degrees of freedom.

Monkey at a 10-key typewriter, $Q_3 - Q_2$ for 10^5 3-letter words, 50 p's.

```
       3-letter-word Monkey Test on digits of pi:
.552878 .823542 .129536 .551312 .067591 .167376 .421748 .192401 .542534 .148380 .492628 .096875
.677799 .728303 .471852 .345121 .417602 .994201 .155789 .487214 .483530 .145260 .779921 .763417
.740147 .955455 .380004 .872453 .101420 .421932 .877049 .463885 .113810 .691983 .978671 .551107
.520321 .234914 .289098 .467913 .477520 .318558 .782897 .732787 .970860 .497160 .995828 .922848
.395496 .477970    ADp= .528741 , Kp= .422532
       3-letter-word Monkey Test on digits of e:
.293642 .594996 .794445 .234524 .256395 .021610 .369478 .318339 .638905 .042199 .167600 .568250
.655712 .099034 .260955 .238186 .228404 .585597 .004558 .120512 .139581 .198334 .929973 .173638
.283437 .749504 .060083 .874294 .823494 .803734 .341567 .047661 .692812 .075428 .882209 .213190
.338682 .303537 .520678 .341827 .407545 .089555 .497969 .526648 .037733 .994200 .447849 .726298
.593973 .407253    ADp= .975320 , Kp= .966113
       3-letter-word Monkey Test on digits of sqrt(2):
.790790 .088220 .598482 .988284 .799611 .598063 .892300 .588479 .217228 .469451 .255440 .720239
.080113 .797689 .618344 .486763 .077658 .724380 .794994 .554164 .223304 .267120 .623616 .905861
.133405 .756233 .954593 .255092 .746933 .196671 .248521 .960442 .230076 .010216 .846633 .367807
.789854 .170249 .549558 .017197 .872847 .825599 .829404 .980072 .739305 .157758 .415892 .912981
.809885 .942661    ADp= .841796 , Kp= .931831
       3-letter-word Monkey Test on digits of 53480293019803.../(44353*10^4608+1):
.067844 .732244 .958921 .768044 .729396 .939194 .783962 .701225 .403579 .133587 .896965 .896898
.731327 .209318 .307055 .725905 .422246 .870295 .975578 .788192 .387908 .804164 .381081 .913977
.250916 .881341 .779515 .983750 .253867 .087866 .181399 .542946 .119927 .582674 .056322 .040929
.276014 .626566 .420973 .205609 .185001 .568973 .940889 .604395 .168453 .139707 .544217 .679437
.233036 .412402    ADp= .291580 , Kp= .408170
       3-letter-word Monkey Test on digits of 3624360069/7000000001:
.269217 .428824 .879686 .616725 .772094 .641691 .363711 .029508 .479642 .600304 .971400 .939788
.905250 .056392 .975416 .331093 .417934 .487233 .151302 .795516 .726313 .222025 .682333 .731094
.901491 .260680 .084794 .480731 .637120 .169748 .267291 .245484 .472621 .692448 .071476 .147444
.409010 .331862 .606700 .525972 .451972 .399568 .204444 .500602 .334999 .813738 .934364 .601014
.200455 .072530    ADp= .079045 , Kp= .113874
       3-letter-word Monkey Test on digits of 123456789012/1000000000061:
.177743 .106103 .684917 .848212 .696941 .585377 .755597 .417014 .634323 .521128 .090036 .383617
.574449 .119654 .831832 .874586 .128672 .426124 .016902 .899537 .565117 .320496 .016979 .940505
.565543 .282273 .107264 .739198 .056312 .962418 .078691 .200442 .008994 .407088 .076785 .031877
.747264 .997366 .606772 .881481 .018749 .747626 .502125 .506074 .406338 .053019 .056477 .622470
.577053 .017231    ADp= .959387 , Kp= .955399
```

Monkey at a 10-key typewriter, $Q_5 - Q_4$ for 10^7 5-letter words, 50 p's.

```
            5-letter-word Monkey Test on digits of pi:
.372079 .580385 .490549 .875977 .329730 .409374 .234803 .896900 .715522 .879609 .595042 .540900
.783273 .707621 .030280 .990831 .030097 .741539 .386933 .501719 .473247 .403703 .446969 .085283
.503383 .352475 .218015 .258597 .940077 .298937 .946689 .377709 .994437 .063976 .254174 .901918
.905861 .570112 .502080 .391362 .103804 .522539 .167597 .327718 .853531 .003656 .505102 .514666
.523355 .126401   ADp = .319807 , Kp = .615995
            5-letter-word Monkey Test on digits of e:
.722274 .464361 .931290 .822275 .216001 .421731 .375407 .125941 .177748 .838363 .592638 .990553
.041359 .408810 .518995 .326927 .246005 .858227 .580711 .085746 .396669 .034698 .027947 .251017
.069772 .066966 .846613 .863321 .150172 .760726 .005565 .748886 .605361 .348911 .148212 .967198
.681314 .392309 .504135 .715312 .098075 .814971 .062877 .375881 .839378 .397496 .419761 .429955
.526946 ..441167  ADp = .598523 , Kp = .554014
            5-letter-word Monkey Test on digits of sqrt(2):
.096511 .034211 .131736 .036309 .825970 .164387 .171196 .562601 .030843 .527668 .152595 .328263
.142102 .109518 .240564 .134643 .078505 .996754 .930394 .479416 .134469 .249720 .632664 .029355
.804028 .551401 .131001 .199612 .898123 .102385 .252099 .156920 .741719 .734167 .551256 .750589
.930250 .395552 .307844 .074091 .072626 .559172 .005057 .975232 .341164 .946162 .189161 .608841
.924069 .202008   ADp = .997296 , Kp = .998832
            5-letter-word Monkey Test on digits of 53480293019803.../(44353*10^4608+1):
.767676 .988740 .953561 .824997 .864988 .637684 .603222 .188060 .560689 .976317 .030124 .478408
.262906 .456553 .242319 .170496 .607171 .221593 .175036 .090318 .643082 .722378 .782823 .234483
.516840 .793632 .179416 .126529 .062226 .912470 .688440 .861970 .260016 .603772 .235049 .962019
.186867 .059554 .215062 .176654 .879025 .054014 .027319 .060665 .749336 .435685 .782021 .929789
.511089 .864018   ADp = .534139 , Kp = .847862
            5-letter-word Monkey Test on digits of 123456789012/1000000000061:
.916817 .670463 .012329 .955258 .317235 .363552 .242206 .588723 .766097 .917976 .065060 .897567
.302766 .424239 .143845 .816340 .994433 .014688 .160104 .280093 .304046 .789716 .735453 .564628
.513790 .245576 .914993 .814688 .547404 .465419 .948477 .240743 .001844 .478601 .814954 .056032
.377757 .617705 .778261 .638282 .167022 .860717 .484746 .638660 .136957 .421423 .108750 .037093
.276031 .761975   ADp = .088273 , Kp = .017782
            5-letter-word Monkey Test on digits of 3624360069/7000000001:
.665915 .591862 .694823 .634431 .435433 .560176 .942535 .462728 .664686 .199567 .123420 .439826
.477214 .735785 .979145 .441576 .388600 .907961 .290529 .690142 .193544 .711771 .694615 .848761
.936787 .850164 .679049 .883409 .501367 .433928 .063014 .733981 .056689 .620116 .739461 .364671
.723262 .319233 .551399 .593139 .287775 .785130 .994758 .193937 .573213 .625490 .466127 .747525
.323995, .717313  ADp = .942655 , Kp = .959772
```

Monkey at a 10-key typewriter, $Q_6 - Q_5$ for 10^8 6-letter words, 10 p's.

```
        6-letter-word Monkey Test on digits of pi:
.55742 .98224 .25092 .29776 .67444 .13658 .35472.72908 .34168 .86568   ADp=.075105, Kp=.048306
        6-letter-word Monkey Test on digits of e:
.12131 .23031 .25774 .81497 .70648 .24832 .54908 .70101 .11026 .44073  ADp=.469382, Kp=.476322
        6-letter-word Monkey Test on digits of sqrt(2):
.05682 .64688 .37158 .63208 .37241 .26006 .94965 .44406 .08595 .48732  ADp=.329194, Kp=.531438
        6-letter-word Monkey Test on digits of 53480293019803.../(44353*10^4608+1):
.61629 .20691 .47563 .15624 .55689 .79736 .22933 .75683 .69366 .21198  ADp=.318255, Kp=.264949
        6-letter-word Monkey Test on digits of 3624360069/7000000001:
.93234 .16491 .91066 .86426 .61637 .97234 .50992 .50974 .95771 .05487  ADp=.926747, Kp=.892703
        6-letter-word Monkey Test on digits of 123456789012/1000000000061:
.85469 .34589 .93289 .57257 .87682 .99526 .88800 .82480 .30554 .36141  ADp=.968693, Kp=.962517
```

Monkey at a 10-key typewriter, 7-letter words.
Because there are too many 7-letter words to easily maintain word counts, this test uses the number of missing 7-letter words from a string of 10^8 keystrokes , which should be approximately normal with (exact) mean $453.999637678\cdots$, (the lack-of-memory approximation gives 453.999297624). The (simulation provided) standard deviation is 21.5, so a p-value results from $\Phi((\text{missing} - 454)/21.5)$. The files provide enough digits for ten p-values.

```
          7-letter-word Monkey Test on digits of pi:
.99908 .44451 .21456 .07467 .03858 .42620 .17613 .48145, .55549 .17613  ADp=.889048, Kp=.854452
          7-letter-word Monkey Test on digits of e:
.33775 .88673 .66225 .95737 .22838 .55549 .94823 .12246, .75731 .83565  ADp=.753828, Kp=.57549
          7-letter-word Monkey Test on digits of sqrt(2):
.53706 .81158 .05689 .03141 .44451 .20124 .21456 .69554, .53706 .91131  ADp=.164625, Kp=.178223
          7-letter-word Monkey Test on digits of 53480293019803.../(44353*10^4608+1):
.10459 .96859 .01002 .91854 .59195 .04702 .27271 .30446, .78544 .64509  ADp=.428261, Kp=.228474
          7-letter-word Monkey Test on digits of 3624360069/7000000001:
.75731 .27271 .71163 .37237 .57380 .79876 .71163 .02826, .67908 .22838  ADp=.292980, Kp=.257653
          7-letter-word Monkey Test on digits of 123456789012/1000000000061:
.86785 .01279 .04702 .96516 .67908 .10459 .90360 .40805, .67908 .74253  ADp=.617002, Kp=.649966
```

The gcd test, see[5].
For two succesive 9-digit integers u and v, get k, the number of steps needed to find $\gcd(u, v)$ by means of Euclid's algorithm. See if 10^6 such k's have the proper (simulation provided) distribution. Repeat, getting 50 p-values.

```
       The gcd Test on digits of pi:
.671667 .883772 .009811 .839441 .369754 .591650 .784751 .868265 .237410 .963933 .849468 .494493
.533792 .473864 .234884 .636397 .588939 .853354 .176993 .870112 .226929 .804280 .873724 .259657
.887194 .174462 .967615 .737803 .130399 .491017 .719674 .909086 .834324 .116745 .526080 .495457
.773433 .258881 .006449 .903006 .950138 .380904 .754911 .778149 .639503 .149432 .045958 .092331
.478044 .725147     ADp= .849312, Kp= .860114
       The gcd Test on digits of e:
.364784 .534584 .842788 .559476 .768062 .162040 .758693 .626092 .872635 .465484 .943743 .502491
.115033 .704692 .911332 .007527 .307169 .713325 .667968 .215541 .398160 .317481 .069268 .192768
.140008 .526837 .464967 .880475 .698219 .938419 .471363 .702865 .815406 .555057 .141919 .884861
.096316 .089996 .004562 .363364 .673914 .743025 .437455 .789266 .883723 .944231 .800914 .214991
.019041 .977924     ADp= .313874, Kp= .432467
       The gcd Test on digits of sqrt(2):
.999282 .296828 .135840 .524178 .359828 .859467 .443873 .798141 .729806 .336649 .876883 .500228
.369207 .941833 .608697 .640314 .925101 .753456 .287965 .125629 .402413 .217178 .747438 .577739
.391712 .729567 .639658 .997947 .626816 .239069 .581269 .584027 .737899 .934103 .462589 .852528
.890138 .992526 .771883 .147292 .011158 .642617 .776497 .931298 .619402 .770106 .295209 .566845
.307424 .651485     ADp= .971043, Kp= .946891
       The gcd Test on digits of 53480293019803.../(44353*10^4608+1):
.845923 .796825 .996363 .140542 .127042 .329533 .363498 .703565 .610829 .798463 .831095 .147745
.836567 .990148 .624513 .126197 .526628 .044942 .116213 .684613 .859655 .207504 .997225 .442651
.384927 .603911 .995462 .993390 .403834 .290100 .947190 .996798 .885665 .880252 .048373 .815011
.290014 .552070 .636909 .917952 .445767 .760912 .042772 .790244 .734967 .070527 .966859 .553830
.167275 .507069     ADp= .983674, Kp= .902253
       The gcd Test on digits of 3624360069/7000000001:
.937022 .349717 .855292 .597600 .933463 .351986 .394742 .467830 .894119 .554223 .866630 .993023
.645313 .164540 .542974 .785757 .836442 .848460 .952434 .742094 .922251 .030468 .922152 .276382
.947681 .643784 .661742 .390999 .739609 .980587 .807090 .505092 .732283 .094249 .305760 .983442
.208305 .530316 .121309 .501365 .257151 .606444 .239208 .364094 .893960 .073256 .486915 .941823
.963072 .149828     ADp= .985834, Kp= .921637
```

```
        The gcd Test on digits of 123456789012/1000000000061:
.038566 .841213 .465589 .596742 .437562 .947480 .655427 .670574 .558254 .517118 .467300 .209585
.309187 .084061 .759393 .211197 .484732 .780381 .631271 .760499 .809770 .071113 .878222 .482427
.940250 .622305 .631284 .876845 .329044 .292128 .986103 .592577 .738636 .971134 .983040 .142789
.850048 .174020 .174307 .133373 .364675 .819730 .765375 .759415 .387761 .603292 .132554 .423091
.860927 .050215     ADp= .501575, Kp= .485551
```

The Birthday Spacings Test

Choose $m = 4000$ birthdays from a year of $n = 10^9$ days. Sort the birthdays into increasing order then make a list of the spacings between the birthdays. If k is the number of distinct values in that list of m spacings, then $y = m-k$ should have nearly a Poisson distribution with mean $m^3/(4n) = 16$. The standard $\sum(O-E)^2/E$ value for 10000 such y's, counting $y <= 5, y = 6, \ldots, y = 29, y >= 30$, should provide a chisquare variate with 24 degrees of freedom, leading to a p value. Digits available for pi,e,$\sqrt{2}$ permit two applications.

```
        Birthday Spacings Test on digits of pi:
chisq24 = 29.176961, p = .786458 chisq24 = 42.531531, p = .988756
        Birthday Spacings Test on digits of e:
chisq24 = 27.647512, p = .724812 chisq24 = 41.381710, p = .984880
        Birthday Spacings Test on digits of sqrt(2):
chisq24 = 11.808707, p = .018022 chisq24 = 33.344739, p = .902994
        Birthday Spacings Test on digits of 53480293019803.../(44353*10^4608+1):
chisq24 = 2 .340080, p = .322719 chisq24 = 33.687752, p = .909627
        Birthday Spacings Test on digits of 3624360069/7000000001:
chisq24 = 18.744330, p = .234517 chisq24 = 4 .628181, p = .981709
        Birthday Spacings Test on digits of 123456789012/1000000000061:
chisq24 = 26.867268, p = .689322 chisq24 = 39.217804, p = .974094
```

Rank test for random 4x4 matrices, elements mod 5.

Sixteen successive random digits form a 4x4 matrix. Its rank (over the field mod 5) will be $4, 3, \geq 2$ with probabilities .76063703, .23731875, .002044216. From 10^6 such random matrices, form $s = \sum(O_4-E_4)^2/E_4 + \sum(O_3-E_3)^2/E_3 + \sum(O_2-E_2)^2/E_2$. Then $p = 1-e^{-s/2}$ should be uniform in [0,1). AD,K tests for 60 ps.

```
         Rank Test on digits of pi:
.61236 .32774 .24743 .62036 .47920 .39825 .83451 .33955 .21810 .02846 .38497 .55130 .91112
.23673 .48132 .20022 .62314 .16924 .41914 .11120 .96091 .92289 .17036 .87940 .92530 .81269
.93233 .61074 .92016 .85609 .21022 .37278 .19652 .84700 .66664 .62260 .27112 .96348 .28388
.90680 .93989 .45602 .15134 .74659 .78556 .83395 .03744 .89305 .98828 .43715 .26146 .01031
.55079 .89585 .07997 .43159 .62127 .68497 .03057 .41909      ADp = .536879 , Kp = .646582
         Rank Test on digits of e:
.12951 .92527 .88829 .83371 .20468 .00878 .71883 .07410 .51399 .70295 .91429 .16873 .12677
.61347 .79059 .97036 .09684 .23057 .75906 .78590 .90164 .04031 .80715 .14913 .16800 .44514
.00025 .55171 .43761 .51266 .53639 .30696 .23939 .29681 .72281 .41517 .90074 .87909 .41690
.03010 .66011 .34184 .25025 .89763 .62858 .40861 .26139 .64248 .70410 .43540 .66902 .65901
.51060 .89945 .22753 .26508 .10193 .95007 .22256 .44659      ADp = .069178 , Kp = .075480
         Rank Test on digits of sqrt(2):
.09128 .45698 .83961 .53255 .79063 .86799 .53850 .67548 .60443 .59196 .15325 .99432 .16537
.39950 .58389 .67670 .76330 .02088 .04918 .01534 .05008 .20734 .07539 .21107 .62461 .59035
.24468 .68984 .82302 .41968 .31665 .15397 .56181 .96717 .79632 .53108 .45995 .10532 .44436
.31481 .28582 .80010 .85732 .40758 .16248 .35456 .76690 .69930 .80077 .71324 .82052 .52464
.03549 .71394 .75174 .82363 .11021 .00583 .91421 .47826      ADp = .307258 , Kp = .216126
         Rank Test on digits of 53480293019803.../(44353*10^4608+1):
.06035 .86008 .69381 .04901 .17196 .43270 .69836 .48883 .12735 .35862 .64930 .03218 .18508
.09487 .19441 .69731 .02436 .67054 .42110 .81008 .01807 .46099 .19884 .41108 .04983 .14816
.67551 .69229 .45123 .62256 .28227 .36480 .48864 .34517 .30020 .83924 .84685 .91042 .83713
```

```
.94957 .09067 .04419 .59719 .40424 .89413 .41900 .57339 .68666 .16281 .92216 .73189 .00692
.93960 .02133 .63535 .01682 .67704 .51112 .80560 .42531      ADp = .849494 , Kp = .652042
        Rank Test on digits of 123456789012/1000000000061:
.50457 .76202 .46063 .41079 .82090 .58449 .18265 .17822 .76772 .00657 .61762 .21230 .33668
.75466 .82648 .02318 .87729 .79113 .10266 .44187 .18629 .12213 .69347 .00993 .03970 .08674
.97598 .40317 .93261 .51054 .17408 .83883 .78718 .52989 .45490 .01319 .15533 .08037 .78510
.95329 .91542 .19989 .47248 .05703 .29830 .47725 .65899 .10816 .54400 .50611 .18640 .19026
.56005 .50394 .93620 .31934 .23941 .39870 .04917 .39145      ADp = .857473 , Kp = .812906
        Rank Test on digits of 3624360069/7000000001:
.50548 .40407 .45985 .21544 .71676 .07310 .39731 .84733 .12748 .66914 .77494 .94007 .04601
.08503 .96720 .36105 .10843 .36816 .95347 .13008 .91868 .48640 .20395 .23753 .76300 .78650
.43418 .79891 .16382 .67890 .33172 .67367 .79725 .02915 .43167 .38174 .99944 .36345 .57114
.71157 .49052 .09132 .36349 .42277 .98883 .94481 .60543 .71442 .74579 .33952 .27278 .30311
.89085 .20119 .90128 .97997 .84571 .10466 .65603 .26555      ADp = .235736 , Kp = .113819
```

The Minimum Distance Test

Do this 50 times: choose $n = 8000$ random points in a square of side 1000. Find d, the minimum distance between the $\binom{n}{2}$ pairs of points. If the points are truly independent uniform, then d^2, the square of the minimum distance, should be (very close to) exponentially distributed with mean .995 . Thus $p = 1-e^{-d^2/.995}$ should behave as uniform, so that AD and K tests on the 50 resulting p-values serve as tests for independence and uniformity of the random points in the square.

```
        Minimum Distance Test on digits of pi:
 .4120 .9175 .2556 .0988 .5167 .0699 .4526 .7175 .2981 .5344 .9872 .5052 .0365 .7453
 .4265 .7265 .7700 .1455 .0446 .3800 .2094 .2827 .3144 .3004 .2808 .2852 .8862 .2590
 .8913 .2264 .1235 .3112 .3297 .7302 .3860 .4436 .7775 .7097 .4254 .5360 .2667 .1402
 .8451 .8067 .2958 .9455 .0285 .4041 .7998 .6931     ADp=.63717, Kp=.79394
       Minimum Distance Test on digits of e:
 .1357 .8348 .8902 .2946 .4938 .9401 .6976 .2712 .0378 .3852 .9216 .5803 .7594 .9491
 .1890 .9076 .0673 .3586 .5735 .3909 .7321 .6515 .8213 .9362 .2354 .3881 .8479 .0197
 .8315 .8686 .4735 .5822 .6445 .6147 .7286 .6512 .1989 .7446 .5623 .7921 .5824 .8493
 .8761 .1892 .3659 .6158 .0636 .0334 .5828 .2407     ADp=.66537, Kp=.87180
       Minimum Distance Test on digits of sqrt(2):
 .0077 .4407 .4620 .7883 .0799 .6719 .1602 .4676 .5426 .3991 .9066 .2523 .3823 .3534
 .7530 .6412 .1529 .5894 .8086 .7753 .4417 .0348 .0458 .8484 .3757 .5262 .9345 .9771
 .6694 .7444 .3441 .8183 .5741 .0056 .9611 .3099 .5381 .4850 .2280 .9116 .4221 .2799
 .6936 .4582 .9657 .7950 .2678 .1827 .9586 .4173     ADp=.14338, Kp=.15792
       Minimum Distance Test on digits of 53480293019803.../(44353*10^4608+1):
 .5161 .8938 .2474 .4979 .3588 .9394 .9412 .7104 .8043 .8729 .1297 .4149 .2219 .9594
 .6943 .2905 .0368 .7528 .0974 .7670 .5444 .6224 .5252 .3824 .4843 .3665 .7193 .3829
 .3083 .8355 .4241 .8762 .7788 .3617 .2047 .9318 .1383 .0178 .4023 .1077 .6065 .4083
 .7635 .7498 .8317 .2259 .0243 .0631 .1460 .4909     ADp=.00907, Kp=.00880
       Minimum Distance Test on digits of 3624360069/7000000001:
 .8357 .0835 .6336 .8672 .3824 .0020 .4191 .2951 .7993 .4287 .7920 .1861 .7822 .5226
 .4637 .2042 .1283 .0340 .9325 .4848 .0200 .9879 .8050 .2330 .4975 .2646 .1856 .2523
 .8477 .2135 .1312 .1995 .5809 .6799 .3381 .0175 .6803 .2068 .7132 .5502 .5180 .8787
 .7058 .5184 .9021 .0617 .3266 .2121 .6527 .6818     ADp=.39568, Kp=.28278
       Minimum Distance Test on digits of 123456789012/1000000000061:
 .4595 .3482 .9125 .0510 .3347 .3996 .2245 .3798 .5560 .3698 .0870 .4287 .2176 .6243
 .2885 .6139 .3978 .0347 .3132 .6388 .9161 .9112 .7615 .5676 .2115 .6418 .3012 .1733
 .0105 .7873 .9384 .2148 .0177 .1956 .1304 .5768 .6472 .1301 .6133 .2343 .8092 .0867
 .8041 .4195 .0603 .8834 .4647 .8749 .8749 .1865     ADp=.73134 Kp=.708020
```

The Random Spheres Test

Choose 4000 random points in a cube of edge 1000. At each point, center a sphere large enough to reach the next closest point. Then the volume of the smallest such sphere is (very close to) exponentially distributed with mean $120\pi/3$. Thus the radius cubed should behave as exponential with mean 3. The test generates 4000 points in twenty edge-1000 cubes. Each minimal radius r leads to a uniform p-value by means of $p = 1-e^{-r^3/3}$. Then AD and K tests are done on 20 p-values.

```
        Random Spheres Test on digits of pi:
.13324 .14133 .06849 .26272 .17096 .72980 .07004 .44462 .45081 .69426 .16412 .35971 .63095
.44411 .57254 .15747 .21987 .55526 .50223 .41768    ADp=.967585, Kp=.925277
        Random Spheres Test on digits of e:
.86788 .36776 .77080 .43359 .49372 .60749 .56377 .84927 .61154 .72559 .59682 .34150 .89111
.81615 .62711 .14515 .27623 .09146 .77067 .82867    ADp=.794671, Kp=.721647
        Random Spheres Test on digits of sqrt(2):
.83747 .80577 .98976 .25698 .13862 .72333 .07953 .31667 .65774 .50034 .33809 .77751 .04578
.23542 .12255 .01773 .92521 .74169 .27930 .29777    ADp=.387178. Kp=.712280
        Random Spheres Test on digits of 53480293019803.../(44353*10^4608+1)
.55927 .14516 .62104 .76105 .55530 .91638 .38106 .97330 .00192 .75081 .54258 .44357 .72508
.09787 .93296 .84232 .11626 .09494 .46035 .46473    ADp=.174649, Kp=.247757
        Random Spheres Test on digits of 3624360069/7000000001:
.61933 .15648 .18816 .62217 .31516 .30767 .78522 .02561 .95307 .41675 .36260 .55015 .15653
.61953 .14190 .47992 .26477 .59653 .26854 .65864    ADp=.710045, Kp=.835487
        Random Spheres Test on digits of 123456789012/1000000000061:
.61640 .79071 .20031 .15108 .03438 .93839 .31778 .01145 .48771 .57387 .30419 .39804 .38313
.76003 .92604 .71981 .47910 .43223 .14415 .59400    ADp=.121779, Kp=.177810
```

The Overlapping 5-Permutation Test

This is the OPERM5 test. It looks at a sequence of ten million 9-digit integers. Each set of five consecutive integers can be in one of 120 states, for the 5! possible orderings that would result if the five were sorted, (a rare $i = j$ is counted as $i < j$). Thus the 5th, 6th,... integers each lead to a state, (but this is not a Markov process). As many thousands of state transitions are observed, cumulative counts are made of the number of occurences of each state. A quadratic form in C^-, a weak inverse of the 120x120 covariance matrix C, should have a chisquare distribution with degrees of freedom the rank of C, leading to a p-value for testing that the counts came from the specified (asymptotically jointly normal) distribution with the specified 120x120 covariance matrix C. This version uses 10,000,000 integers, ten times.

```
       OPERM5 Test on digits of pi:
.9971 .1165 .3786 .8458 .3110 .2483 .3922 .5333 .7244 .7192   ADp=.198201, Kp=.042075
       OPERM5 Test on digits of e:
.1369 .7283 .6581 .7493 .7363 .5089 .7026 .5570 .7394 .5000   ADp=.884353, Kp=.941002
       OPERM5 Test on digits of sqrt(2):
.3268 .6270 .0273 .5704 .6183 .7432 .7781 .1704 .4118 .1665   ADp=.272362, Kp=.367715
       OPERM5 Test on digits of 53480293019803.../(44353*10^4608+1):
.7807 .6140 .6171 .6926 .3044 .0377 .8617 .4061 .5980 .4939   ADp=.394569, Kp=.283185
       OPERM5 Test on digits of 3624360069/7000000001:
.2640 .4178 .3434 .2756 .8069 .3949 .6555 .9673 .1909 .3971   ADp=.457414, Kp=.662883
       OPERM5 Test on digits of 123456789012/1000000000061:
.4467 .3650 .5587 .2181 .4417 .9622 .5630 .3778 .0976 .4238   ADp=.622551, Kp=.837349
```

A Runs Test

Count runs up, and runs down, in a sequence of uniform [0,1) variables, obtained by floating the 9-digit integers in the specified file. Example of how runs are counted: .123, .357, .789, .425, .224, .416, .95 contains an up-run of length 3, a down-run of length 2 and an up-run of (at least) 2, depending on the next values. Count upruns and downruns until there are 100 million of each. (For $k \geq 2$, the prob. of a run of length k is $2k/(k+1)!$)

Runs Test on digits of pi:

Length	Expected	UpRuns	(O-E)^2/E	DownRuns	(O-E)^2/E
2	666666.7	667054	.23	666664	.00
3	25000 .0	249379	1.54	250317	.40
4	66666.7	66747	.10	66505	.39
5	13888.9	14050	1.87	13703	2.49
6	238 .9	2374	.02	2446	1.78
7	347.2	342	.08	312	3.57
8	44.2	47	.18	45	.01
9	5.0	4	.19	7	.84
		p= .06184		p= .51293	

Runs Test on digits of e:

Length	Expected	UpRuns	(O-E)^2/E	DownRuns	(O-E)^2/E
2	666666.7	665943	.79	666332	.17
3	25000 .0	250229	.21	250131	.07
4	66666.7	66904	.84	66948	1.19
5	13888.9	14094	3.03	13833	.22
6	238 .9	2454	2.24	2343	.60
7	347.2	330	.85	374	2.07
8	44.2	43	.03	34	2.35
9	5.0	1	3.16	4	.19
		p= .65453		p= .26158	

Runs Test on digits of sqrt(2):

Length	Expected	UpRuns	(O-E)^2/E	DownRuns	(O-E)^2/E
2	666666.7	667194	.42	666515	.03
3	25000 .0	249506	.98	250027	.00
4	66666.7	66453	.69	66825	.38
5	13888.9	14040	1.64	13768	1.05
6	238 .9	2378	.00	2492	5.18
7	347.2	373	1.92	326	1.29
8	44.2	53	1.75	42	.11
9	5.0	2	1.77	3	.77
		p= .48314		p= .45109	

Runs Test on digits of 53480293019803.../(44353*10^4608+1):

Length	Expected	UpRuns	(O-E)^2/E	DownRuns	(O-E)^2/E
2	666666.7	667447	.91	666891	.08
3	25000 .0	249739	.27	250114	.05
4	66666.7	66236	2.78	66479	.53
5	13888.9	13795	.63	13729	1.84
6	238 .9	2404	.22	2364	.12
7	347.2	326	1.29	365	.91
8	44.2	39	.61	49	.52
9	5.0	11	7.36	7	.84
		p= .83097		p= .10162	

```
          Runs Test on digits of 123456789012/1000000000061:
 Length  Expected        UpRuns   (O-E)^2/E  DownRuns (O-E)^2/E
    2     666666.7        666247     .26      667031    .20
    3     25000 .0        250512    1.05      249445   1.23
    4      66666.7         66421     .91       66851    .51
    5      13888.9         13926     .10       13912    .04
    6       238 .9          2456    2.37        2353    .33
    7        347.2           384    3.90         350    .02
    8         44.2            45     .01          49    .52
    9          5.0             8    1.86           8   1.86
                       p= .59900           p= .09057
          Runs Test on digits of 3624360069/7000000001:
 Length  Expected        UpRuns   (O-E)^2/E  DownRuns (O-E)^2/E
    2     666666.7        665869     .95      666392    .11
    3     25000 .0        251165    5.43      250505   1.02
    4      66666.7         66249    2.62       66445    .74
    5      13888.9         13959     .35       13890    .00
    6       238 .9          2358     .22        2366    .09
    7        347.2           350     .02         356    .22
    8         44.2            41     .23          40    .40
    9          5.0             5     .00           5    .00
                       p= .54448           p= .01045
```

The Parking Lot Test

In a square of side 100, randomly 'park' a car: a disc with radius 1. Then try to park a 2nd, a 3rd, and so on, each time parking 'by ear'—that is, if an attempt to park a car causes a crash with one already parked, try again at a new random location. (To avoid path problems, consider parking helicopters rather than cars.) Each attempt leads to either a crash or a success, the latter followed by an increment to the list of cars already parked. If we plot n, the number of attempts, versus k, the number successfully parked, we get a curve that should be similar to those provided by a perfect random number generator. A simple characterization of such an extensive experiment is used: k, the number of cars successfully parked after $n = 12000$ attempts. Simulation shows that k should average 3523 with sigma 21.9, and it seems close to normally distributed. Thus $\Phi((k - 3523)/21.9)$ should provide a p-value, and each file has enough digits to provide twenty such ps.

```
        Parking Lot Test on digits of pi:
.92154 .13656 .03747 .92154 .46362 .99399 .02480 .44552 .81944 .98407 .65945 .64255
.03389 .46362 .51821 .90728 .53638 .55448 .92154 .83120    ADp=.910269, Kp=.848908
         Test on digits of e:
.73867 .53638 .00353 .40970 .863444 .53638 .89947 .64255 .05500 .85319 .48179 .81944
.32397 .35744 .554480 .32397 .64255 .75331 .93973 .62538   ADp=.652575, Kp=.769028
        Parking Lot Test on digits of sqrt(2):
.73868 .26132 .60795 .98737 .60795 .07198 .35744 .29186 .78120 .76749 .35744 .46361
.93407 .93407 .67603 .99647 .01263 .75330 .40970 .44552    ADp=.664483, Kp=.381541
        Parking Lot Test on digits of 53480293019803.../(44353*10^4608+1):
.59030 .27639 .55448 .59030 .02226 .51821 .30773 .87318 .34055 .42754 .11757 .94500
.89119 .64255 .60795 .92154 .89947 .59030 .70813 .81944    ADp=.701556, Kp=.743065
        Parking Lot Test on digits of 3624360069/7000000001:
.57246 .53638 .84245 .79444 .83120 .29186 .39205 .85319 .48179 .53638 .87318 .75331
.02478 .37462 .80719 .76749 .37462 .32397 .67603 .73868    ADp=.876019  Kp=.837159
        Parking Lot Test on digits of 123456789012/1000000000061:
.70813 .69227 .32397 .35744 .59030 .81944 .13656 .70813 .62538 .05500 .55448 .50000
.92802 .24669 .87318 .69227 .62538 .92154 .78120 .88243    ADp=.839585, Kp=.874860
```

The Squeeze Test

Random 9-digit integers are floated to get uniforms in [0,1). Initializing $k = 2^{31}$, the test finds j, the number of iterations necessary to reduce k to 1, using the reduction $k \leftarrow \lceil kU \rceil$, with U provided by floating integers from the file being tested. Such j's are found 100,000 times, then counts for the number of times j was $\leq 6, 7, \ldots, 47, \geq 48$ are used to provide a χ^2_{42} test for cell frequencies, repeated ten times.

```
  Squeeze Test on digits of pi:
Chisquare_42= 36.88, z= -.5588, p= .28815   Chisquare_42= 44.09, z=  .2281, p= .59021
Chisquare_42= 45.01, z=  .3287, p= .62883   Chisquare_42= 45.37, z=  .3678, p= .64349
Chisquare_42= 43.94, z=  .2119, p= .58391   Chisquare_42= 46.47, z=  .4874, p= .68701
Chisquare_42= 36.01, z= -.6534, p= .25675   Chisquare_42= 23.91, z= -1.973, p= .02420
Chisquare_42= 39.27, z= -.2981, p= .38281   Chisquare_42= 34.83, z= -.7828, p= .21687
        AD and K tests on the above ten p-values: .586129 .771924
  Squeeze Test on digits of e:
Chisquare_42= 49.24, z=  .7899, p= .78520   Chisquare_42= 46.33, z=  .4729, p= .68187
Chisquare_42= 36.24, z= -.6289, p= .26469   Chisquare_42= 4 .21, z= -.1953, p= .42259
Chisquare_42= 49.11, z=  .7763, p= .78120   Chisquare_42= 26.67, z= -1.672, p= .04720
Chisquare_42= 37.90, z= -.4469, p= .32748   Chisquare_42= 46.27, z=  .4654, p= .67919
Chisquare_42= 53.40, z=  1.243, p= .89322   Chisquare_42= 75.82, z=  3.689, p= .99989
        AD and K tests on the above ten p-values: .704968 .650456
  Squeeze Test on digits of sqrt(2):
Chisquare_42= 51.05, z=  .9878, p= .83839   Chisquare_42= 41.01, z= -.1075, p= .45721
Chisquare_42= 43.82, z=  .1985, p= .57866   Chisquare_42= 56.15, z=  1.544, p= .93871
Chisquare_42= 36.62, z= -.5869, p= .27865   Chisquare_42= 34.62, z= -.8053, p= .21032
Chisquare_42= 38.93, z= -.3347, p= .36893   Chisquare_42= 29.07, z= -1.411, p= .07918
Chisquare_42= 44.53, z=  .2758, p= .60866   Chisquare_42= 49.21, z=  .7870, p= .78437
        AD and K tests on the above ten p-values: .000540 .001727
  Squeeze Test on digits of 3650075.../(105994*10^105994+1):
Chisquare_42= 33.63, z= -.9131, p= .18058   Chisquare_42= 46.26, z=  .4644, p= .67881
Chisquare_42= 44.78, z=  .3038, p= .61936   Chisquare_42= 38.53, z= -.3782, p= .35264
Chisquare_42= 51.93, z=  1.084, p= .86079   Chisquare_42= 43.27, z=  .1390, p= .55529
Chisquare_42= 52.45, z=  1.141, p= .87299   Chisquare_42= 5 .55, z=  .9334, p= .82470
Chisquare_42= 33.06, z= -.9756, p= .16464   Chisquare_42= 47.80, z=  .6326, p= .73651
        AD and K tests on the above ten p-values: .490230 .542116
  Squeeze Test on digits of 3624360069/7000000001:
Chisquare_42= 27.47, z= -1.585, p= .05649   Chisquare_42= 68.21, z=  2.860, p= .99788
Chisquare_42= 44.06, z=  .2251, p= .58905   Chisquare_42= 47.53, z=  .6038, p= .72702
Chisquare_42= 42.57, z=  .0618, p= .52463   Chisquare_42= 43.82, z=  .1988, p= .57879
Chisquare_42= 69.65, z=  3.017, p= .99872   Chisquare_42= 65.14, z=  2.525, p= .99421
Chisquare_42= 43.82, z=  .1985, p= .57867   Chisquare_42= 58.73, z=  1.825, p= .96600
         AD and K tests on the above ten p-values: .998123 .962399
  Squeeze Test on digits of 123456789012/1000000000061:
Chisquare_42= 33.49, z= -.9289, p= .17647  Chisquare_42= 37.69, z= -.4704, p= .31903
Chisquare_42= 32.08, z= -1.082, p= .13955  Chisquare_42= 4 .67, z= -.1453, p= .44225
Chisquare_42= 38.85, z= -.3437, p= .36553  Chisquare_42= 28.09, z= -1.517, p= .06459
Chisquare_42= 65.65, z=  2.580, p= .99506  Chisquare_42= 39.35, z= -.2894, p= .38615
Chisquare_42= 41.39, z= -.0661, p= .47366  Chisquare_42= 41.60, z= -.0439, p= .48249
         AD and K tests on the above ten p-values: .819619 .957027
```

I added the following two tests after seeing numerous website references to the appearance of primes among the digits of π:

Count the 3-Digit Primes Test

With $n = 10^6$, count K, the number of 3-digit primes in a string of $n+2$ random digits. Use overlapping 3-tuples; thus the string 10197 contains primes 101, 019 and 197. The mean and variance of K are $\mu = 168n/10^3$ and $\sigma^2 = (138640n + 3224)/10^6$. Thus $p = \Phi((K - \mu)/\sigma)$ should be very close to uniform with $\mu = 168000, \sigma = 372.34$. (Note that K is not binomial with $n, p = 168/10^3, npq = 139776n/10^6$.) Repeat 999 times, do tests of uniformity (Anderson-Darling and Kolmogorov) on those 999 ps.

```
3-digit Primes Test on digits of pi:
          ADp= .575829 , Kp= .612172
3-digit Primes Test on digits of e:
          ADp= .883840 , Kp= .967424
3-digit Primes Test on digits of sqrt(2):
          ADp= .991416 , Kp= .975246
3-digit Primes Test on digits of 53480293019803.../(44353*10^4608+1)
          ADp= .679134 , Kp= .509247
3-digit Primes Test on digits of 3624360069/7000000001:
          ADp= .901572 , Kp= .919128
3-digit Primes Test on digits of 123456789012/1000000000061:
          ADp= .628301 , Kp= .728232
```

Count the 4-Digit Primes Test

With $n = 10^6$, count K, the number of 4-digit primes in a string of n+3 random digits. Use overlapping 4-tuples; thus the string 10197 contains primes 1019 and 0197. The mean and variance of K are $\mu = 1229n/10^4$ and $\sigma^2 = (10619573n+525112)/10^8$. Thus $p = \Phi((K - \mu)/\sigma)$ should be very close to uniform with $\mu = 122900, \sigma = 325.88$. (Note that K is not binomial $n, p = 1229/10^4$,$npq = 10779559n/10^8$.) Repeat 999 times, then do Anderson-Darling and Kolmogorov tests on the resulting 999 ps.

```
4-digit Primes Test on digits of pi:
          ADp= .588839 , Kp= .854419
4-digit Primes Test on digits of e:
          ADp= .821890 , Kp= .578593
4-digit Primes Test on digits of sqrt(2):
          ADp= .987949 , Kp= .996103
4-digit Primes Test on digits of 53480293019803.../(44353*10^4608+1):
          ADp= .299561 , Kp= .238585
4-digit Primes Test on digits of 3624360069/7000000001:
          ADp= .477074 , Kp= .405052
4-digit Primes Test on digits of 123456789012/1000000000061:
          ADp= .280296 , Kp= .440059
```