Randomness

1. True randomness

True intrinsic randomness is found in Quantum Mechanics.

RNG's based on Quantum Mechanics exist, but require an external device attached to the computer and have many other disadvantages.

A different kind of randomness is found in Classical Mechanics. Poincaré already knew that classical mechanical systems of three or more bodies were intrinsically unstable and would eventually become unpredictable.

The Russian school of mathematicians developed a full theory of randomness in non-integrable classical mechanical Hamiltonian systems. But for many years, no one was able to make a workable random number generator based on this kind of classical randomness.

Instead, the urgent need for random numbers in Monte Carlo calculations caused computer users to make use of the mysterious phenomenon called pseudo randomness.

2. Pseudorandomness.

In the 1940's, John von Neumann and friends were amazed to discover that simple arithmetic operations like integer multiplication could produce sequences of numbers that appeared to be random. They called them pseudorandom. Nobody understood why they appeared to be random.

Around 1960 the first computers became available in universities.
They came with a Fortran compiler and a small library of subroutines like sine, cosine, logarithms, and ... a Random Number Generator.
Everyone assumed that all these things made by IBM worked correctly.

But our MC calculations sometimes gave funny results: lots of regular peaks in a distribution that should be smooth.
We complained, IBM sent us a new library, and everything seemed OK.

In 1968, George Marsaglia published his famous paper explaining that the problem we saw was inherent in all multiplicative congruential generators (the kind everybody was using).

Pseudorandomness: "La fuite en avant": [If you don't know what you're doing, do it faster!]

The following years produced progress in finding defects in existing generators, and finding faster generators with longer periods, but

no progress in explaining why the numbers were (apparently) random.

Donald Knuth, The Art of Computer Programming, Vol. 2, 3rd edition, 1994. Over 600 pages on pseudorandom number generators, but no hint about how they make randomness. A lot about how hard it is to define.

The long-period generators were based on an array of numbers in memory. The next random number was formed by combining two numbers in the array, with the new number replacing one of the two in the array.

Around 1990, the most popular generators were:
1. RCARRY (or AWC or SWB) of Marsaglia and Zaman, is based on an array of 24 words in memory, and has period$= 2^{23\times 24} \approx 5 \times 10^{171}$
2. Mersenne Twister (from Japan), with an array of 624 words in memory, is even faster and has a much longer period.

Toward a Theory of Pseudorandomness

Pseudorandomness has been studied extensively from the point of view of Kolmogorov complexity, which defines the randomness of a sequence in terms of the shortest algorithm that could produce that sequence. Clearly, if a given sequence cannot be produced by any algorithm shorter than the sequence itself, it must be random because it has no structure.

However, Kolmogorov complexity is not a good candidate concept to help in making good RNG's, because it could at best only produce a good RNG which is practically impossible to use because it is too slow.

Better candidates are: unpredictability and independence. These are in principle the same:

- A sequence is unpredictable if the probability of predicting the next number, given the knowledge of the previous numbers, is no greater than would be the case for a truly random sequence.
- The numbers in a sequence are independent if the probability of occurrence of a given number is independent of the previous numbers.

A different approach from the Russian school

The study of classical mechanical systems developed differently in the West (only interested in solvable systems) and in the Soviet Union (interested mainly in non-integrable systems because they are candidates for chaos.).

Some physicists from Armenia led by George Savvidy thought that the right way to generate independent sequences was to make use of Kolmogorov mixing (not related to Kolmogorov complexity).

Representing the state of the dynamical system at the i^{th} time step by an array $\mathbf{X}_i$, the system evolves via linear transformations

$$\mathbf{X}_{i+1} = \mathcal{A}\, \mathbf{X}_i \ .$$

To obtain Kolmogorov mixing, the matrix $\mathcal{A}$ had to have

- Determinant $= 1$
- All eigenvalues complex and distinct
- No eigenvalue has modulus 1

Unfortunately, the Matrix RNG was too slow, and its period unknown.

The Crisis in Ising Model Simulations

Meanwhile a crisis was developing in Condensed Matter Physics where the "best" generators were giving wrong results. The failures were being revealed by a new algorithm invented by Ulli Wolff to overcome the critical slowing down that had made it nearly impossible to simulate across phase transitions.
[Ferrenberg, Landau and Wong, MC Simulations: Hidden errors from "good" random number generators, PRL 69, 3382-3384 (1992)]

Ulli Wolff's algorithm worked by flipping many spins at once, instead of one at a time, so he thought it would be more sensitive to long-term correlations in the random number generator than other algorithms.
He also said there was a theoretician at DESY, Martin Lüscher, who had found a way to make a better generator, but it was coded only for the special-purpose computer APE designed for Lattice Gauge calculations.

Lüscher's algorithm

To my great surprise, Martin Lüscher had decided to base his new generator on the RCARRY algorithm. This was surprising because RCARRY was already known to be defective, but Martin knew exactly what he was doing.

If you consider the RCARRY algorithm as a sequence of points in 24-dimensional space, then it is of the form

$$\mathbf{X}_{i+1} = \mathcal{A}\, \mathbf{X}_i$$

where $\mathbf{X}_i$ is the array of 24 24-bit numbers that determines (along with some indices) the state of the generator.

The algorithm is then defined by the matrix $\mathcal{A}$, which you can readily see is a matrix with mostly zeroes, with 24 values "$+1$" along the diagonal and 24 "-1" at the positions corresponding to the values of r and s. The determinant is 1, all of its eigenvalues are different and none of them have modulus 1.

RCARRY has Kolmogorov Mixing

He had seen that the RCARRY algorithm was in fact the discrete analog of a continuous classical dynamical system with Kolmogorov mixing. And he knew what that meant:

1. Zero-mixing is the same as ergodic motion. It means that the system will asymptotically sweep out all the available states.
2. One-mixing means that the coverage will be uniform, in the sense that asymptotically, the probability of finding the system in any given volume of state space is proportional to the volume.
3. Two-mixing means that the probability of the system being in one volume of state space at one time, and another volume at another time, is proportional to the product of the two volumes.
4. N-mixing means that the probability of finding the system in N different volumes at N different times is proportional to the product of the N volumes.
5. k-mixing is N-mixing for arbitrarily large N.

Mixing in Sinai's billiards

Another way of looking at chaos in classical dynamical systems is the trajectory of a ball on a frictionless billiard table, studied extensively by Ya. G. Sinai.

On a standard rectangular table, it can be shown that the motion is almost always ergodic (almost all trajectories will cover the whole table asymptotically), but there is no mixing, so these systems are not k-systems.

If the edges of the table are curved outward, there is a focussing effect, and the motion is not even sure to be ergodic. On an elliptical table, for example, the system is integrable, and not chaotic.

But if the edges of the table are curved inward, they defocus nearby trajectories, leading usually to Kolmogorov mixing.

Lüscher's algorithm: The Lyapunov Exponent

Now the big question: Why does RCARRY fail the tests of randomness?

Answer: Because Kolmogorov mixing is only an asymptotic property .
Such systems forget their past history, but only after a certain time.

K-systems diverge exponentially, in the following sense:
If a k-system is initiated from two different but arbitrarily close initial states, the two trajectories will diverge exponentially. The exponent describing the speed of divergence is called the Lyapunov exponent.

For the matrix $\mathcal{A}$, there are 4 eigenvalues with maximum absolute value $|\lambda|_{max} = 1.04299..$, and the Lyapunov exponent is $\ln |\lambda|_{max} = 1.01027..$

For this system, that means that full mixing (where the deviation between the trajectories exceeds $1/2$) is attained only after 16 applications of $\mathcal{A}$.

Testing Random Number Generators

Since there has been (until recently) no theory of randomness that could be applied to RNG, the only way to have some evidence for randomness was by statistical testing of the results.

A test of randomness is any function of the output of a RNG for which the expectation and variance are known (under the hypothesis of complete randomness of course). There are an uncountably infinite number of such tests possible. No test can prove a RNG to be good, but if a test is failed, that is proof that it is not good. Since these are Goodness-of-Fit tests, the power of the test is undefined, and no test is necessarily better than any other, but with experience, we can have some idea.

The DIEHARD test suite of Marsaglia was used for many years, but it is not extendable for testing long-period generators. The current standard is TestU01 by L'Ecuyer and Simard of the University of Montreal. The heart if this excellent software is the test affectionately known as Big Crush.

The situation concerning RNG's from 1995 to 2015

There has been little or no significant progress in traditional RNG's (not based on Kolmogorov-Anisov Mixing), which is not surprising since the people working on these generators are working in the dark, with no theory to guide them. By chance they managed to produce a k-system (RCARRY), but didn't know how to use it.

Martin Lüscher's RANLUX, which requires considerable decimation, slowing it down by about a factor of ten for the double-precision (48-bit) version, has been the only popular RNG which has a solid theoretical base, passes all the current tests, and with which we have considerable experience.

Its principal rival, the Mersenne Twister, is much faster but is now known to have serious defects and fails some Big Crush tests.

But the biggest progress has been MIXMAX.

The new MIXMAX

The MIXMAX project is considerably more ambitious than RANLUX. Instead of using an existing generator and showing it was a k-system, George Savvidy set out to find the optimal matrix, a k-system with eigenvalues as far as possible from the unit circle.
George made some progress, reducing the computation time to produce N random numbers from $O(N^2)$ to $O(N \ln N)$, but some very serious mathematical problems remained, and MIXMAX was still too slow.

Enter Konstantin Savvidy, also a theoretical physicist specialist in Quantum Field Theory, who was able to reduce the computation time to $O(N)$ and solve the Herculean problem of calculating the (astronomical) period of the new generator.