13. Various Techniques Used in Connection With Random Digits

By John von Neumann

Summary written by George E. Forsythe

In manual computing methods today random numbers are probably being satisfactorily obtained from tables. When random numbers are to be used in fast machines, numbers will usually be needed faster. More significant is the fact that, because longer sequences will be used, one is likely to have more elaborate requirements about what constitutes a satisfactory sequence of random numbers. There are two classes of questions connected with high-speed computation about which I should like to make some remarks: (A) How can one produce a sequence of random decimal digits—a sequence where each digit appears with probability one-tenth and where consecutive ones are independent of each other in all combinations? (B) How can one produce random real numbers according to an assigned probability distribution law?

On problem (A), I shall add very little to what has been said earlier in this symposium. Two quantitatively different methods for the production of random digits have been discussed: physical processes and arithmetical processes. The main characteristics of physical processes have been pointed out by Dr. Brown. There are nuclear accidents, for example, which are the ideal of randomness, and up to a certain accuracy you can count them. One difficulty is that one is never quite sure what is the probability of occurrence of the nuclear accident. This difficulty has been overcome by taking larger counts than one in testing for either even or odd. To cite a human example, for simplicity, in tossing a coin it is probably easier to make two consecutive tosses independent than to toss heads with probability exactly one-half. If independence of successive tosses is assumed, we can reconstruct a 50–50 chance out of even a badly biased coin by tossing twice. If we get heads-heads or tails-tails, we reject the tosses and try again. If we get heads-tails (or tails-heads), we accept the result as heads (or tails). The resulting process is rigorously unbiased, although the amended process is at most 25 percent as efficient as ordinary coin-tossing.

We see then that we could build a physical instrument to feed random digits directly into a high-speed computing machine and could have the control call for these numbers as needed. The real objection to this procedure is the practical need for checking computations. If we suspect that a calculation is wrong, almost any reasonable check involves repeating something done before. At that point the introduction of new random numbers would be intolerable. I think that the direct use of a physical supply of random digits is absolutely inacceptable for this reason and for this reason alone. The next best thing would be to produce random digits by some physical mechanism and record them, letting the machine read them as needed. At this point we have maneuvered ourselves into using the weakest portion of presently designed machines—the reading organ. Whether or not this difficulty is an absolute one will depend on how clumsy the competing processes turn out to be.

Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin. For, as has been pointed out several times, there is no such thing as a random number—there are only methods to produce random numbers, and a strict arithmetic procedure of course is not such a method. (It is true that a problem that we suspect of being solvable by random methods may be solvable by some rigorously defined sequence, but this is a deeper mathematical question than we can now go into.) We are here dealing with mere "cooking recipes" for making digits; probably they can not be justified, but should merely be judged by their results. Some statistical study of the digits generated by a given recipe should be made, but exhaustive tests are impractical. If the digits work well on one problem, they seem usually to be successful with others of the same type.

If one nevertheless considers certain arithmetic methods in detail, it is quickly found that the critical thing about them is the very obscure, very imperfectly understood behavior of round-off errors in mathematics. In obtaining y as the middle ten digits in the square of a ten-digit number x, we are really mapping x onto y by a certain saw-toothed discontinuous curve $y=f(x)$, for $0\leq x\leq 1$, $0\leq y\leq 1$. When we take $x_{i+1}=f(x_i)$ for $i=1, 2, 3, \ldots,$ this curve will gradually scramble the digits of x_1 and produce something fairly pseudo-random. A simpler process suggested by Dr. Ulam is to use the mapping function $f(x)=4x(1-x)$. If one produces a sequence $\{x_i\}$ in this manner, x_{i+1} is completely determined by x_i, so that independence is lacking. It is, however, quite instructive to analyze the nature of randomness that exists in this sequence. One can, by an incomplete argumentation, apparently establish one kind, and then see that in reality a very different kind holds. First, let the relations $x_i=\sin^2 \pi\alpha_i$ define the sequence $\{\alpha_i\}$ (each modulo 1). Since $x_{i+1}=4x_i(1-x_i)$, one sees that $\alpha_{i+1}=2\alpha_i$ (modulo 1). The sequence $\{\alpha_i\}$ is thus obtained in binary representation by shifting the binary number $\alpha_1=\cdot\beta_1\beta_2\beta_3\beta_4 \ \ldots$ as follows: $\alpha_2=\cdot\beta_2\beta_3\beta_4 \ \ldots, \ \ldots, \ \alpha_i=\cdot\beta_i\beta_{i+1}\beta_{i+2} \ \ldots, \ \ldots$. It follows from the theorem of Borel about the randomness of the digits of real numbers that, for all numbers α_1 except those in a set of Lebesgue measure zero, the numbers α_i are uniformly distributed on the interval (0,1). Hence, the x_i are distributed like the numbers $x=\sin^2\pi y$, with equidistributed y, i. e., with the probability distribution $(2\pi)^{-1}[x(1-x)]^{-\frac{1}{2}}dx$.

However, any physically existing machine has a certain limit of precision. Since the average value of $|f'(x)|$ on (0, 1) is 2, in each transformation from x_i to x_{i+1} any error will be amplified on the average by approximately two. In about 33 steps the first round-off error will have grown to about 10^{10}. No matter how random the sequence $\{x_i\}$ is in theory, after about 33 steps one is really only testing the random properties of the round-off error. Then one might as well admit that one can prove nothing, because the amount of theoretical information about the statistical properties of the round-off mechanism is nil.

As Dr. Hammer told us, sequences $\{x_i\}$ obtained from the squaring routine have been successfully used for various calculations. By the very assumption of randomness, however, one is exposed to the systematic danger of the appearance of self-perpetuating zeros at the ends of the numbers x_i. Dr. Forsythe's remark that in some cases the zero mechanism is the major mechanism destroying the sequences is encouraging, because one always fears the appearance of undetected short cycles. I fear, however, that if we used one of Dr. Brown's ingenious tricks to overcome the zero mechanism, we might just bring out the next most disastrous mechanism. In any case, I think nobody who is practically concerned will want to use a sequence produced by any method without testing it statistically, and it has been the uniform experience with those sequences that it is more trouble to test them than to manufacture them. Hence the degree of complication of the method by which you make them is not terribly important; what is important is to carry out a relatively quick and efficient test. Personally I suspect that it might be just as well to use some cooking recipe like squaring and taking the middle digits, perhaps with more digits than ten.

Let me now consider questions of the class (B). If one wants to get random real numbers on (0, 1) satisfying a uniform distribution, it is clearly sufficient to juxtapose enough random binary digits. To avoid any bias it is probably advisable always to force the last digit to be a 1. If, however, the numbers are to satisfy some nonuniform probability distribution $f(x)dx$ on (0, 1), some tricks are possible and advisable. Suppose one lets $t=F(x)=\int_0^x f(u)du$, and lets $x=\Phi(t)$ represent the transformation inverse to F. If the random variable T is uniformly distributed on (0, 1), then the random variable $X=\Phi\ (T)$ obeys the distribution law $F(x)$. Now the human computer can obtain the inverse function reasonably efficiently by scanning, with or without the aid of the rotating drum mentioned in Dr. Wilson's paper earlier in this symposium. A machine scans poorly, but might be instructed to calculate each X from T. This calculation is, however, likely to be quite cumbersome in practice, and I should like to mention some other methods that are often more efficient when they are applicable.

One trick is to pick a scaling factor a such that $af(x)\leq 1$, and then to produce two random numbers, X and Y, from a uniform distribution on (0, 1). If $Y>af(X)$, we reject the pair and call for a new pair. If $Y\leq af(X)$, we accept X. Since the acceptance ratio is proportional to $f(X)$, the accepted numbers X will have the probability distribution $f(x)dx$. One can see that the efficiency of the method depends on the ratio of the average value of $f(x)$ to its maximum value; the smaller the ratio, the more difficult it will be to use the method. One characteristic of this method is that one can often make the test "is $Y>af(X)$?" implicitly, without having to calculate $f(X)$ explicitly. For example, if

$$af(x)=(1-x^2)^{\frac{1}{2}},$$

one may make the test "is $X^2+Y^2>1$?" and only elementary operations are needed.

Suppose one wants to produce random numbers Θ in $(-1, 1)$ according to the distribution

$$\pi^{-1}(1-\theta^2)^{-\frac{1}{2}}d\theta.$$

The obvious procedure is to take a uniformly distributed random variable T in $(-1, 1)$ and calculate

$$\Theta=\sin\pi T.$$

I have a feeling, however, that it is somehow silly to take a random number and put it elaborately into a power series, and in this case one can employ a trick related to the last one. Select two random numbers X, Y from a uniform distribution on (0, 1); the point (X, Y) lies in the unit square. To make sure that its angle,

$$\phi=\text{arc tan }(X/Y),$$

is uniformly distributed on $(0, \pi/2)$, we first reject the pair if $X^2+Y^2>1$. If $X^2+Y^2\leq 1$, we accept tbe pair and form $\pm X/(X^2+Y^2)^{1/2}$ ($\pm$random), which will have the desired distribution. The efficiency of the process is clearly $\pi/4$. The only disagreeable feature is the square root, and even it may be eliminated by forming Θ not as

$$\pm\sin\phi=\pm X/(X^2+Y^2)^{1/2},$$

but as

$$\sin(2\phi-\pi/2)=-\cos 2\phi=(X^2-Y^2)/(X^2+Y^2),$$

which has the same distribution.

Let me give one final example, the generation of nonnegative random numbers X with the distribution $e^{-x}\,dx$. As you know, such numbers X represent free paths in the slab problems discussed in this symposium. The normal procedure is to pick T from a uniform distribution on $(0, 1)$ and compute $X=-\log T$, but, as before, it seems objectionable to compute a transcendental function of a random number. Another method for getting X resembles a well known game of chance—Twenty-one, or Black Jack. We select numbers Y_i from a uniform distribution on $(0, 1)$ as follows. Pick Y_1 and Y_2. If $Y_1\leq Y_2$, we stop. If $Y_1>Y_2$, we select Y_3. If $Y_2\leq Y_3$, we stop. Otherwise $Y_1>Y_2>Y_3$, and we select Y_4, etc. With probability one there will be a least n for which

$$Y_1>Y_2>\ldots>Y_n, \text{ but } Y_n\leq Y_{n+1}.$$

Now if n is odd, we accept Y_1 as X, but if n is even, we reject all the Y_i. To analyze this process, let E_n represent the event

$$Y_1>Y_2>\ldots>Y_n.$$

It is easily shown by induction that

$$\text{Prob}\,(E_n)=(n!)^{-1},$$

while

$$\text{Prob}\,\{x<Y_1<x+dx \mid \text{given } E_n\}=nx^{n-1}dx.$$

Hence the probability that we simultaneously have

$$x<Y_1<x+dx,\ E_n, \text{ and not-}E_{n+1}$$

is

$$[x^{n-1}/(n-1)!-x^n/n!]\,dx.$$

Summing over all odd n, we see that

$$\text{Prob}\{x<Y_1<x+dx\}=e^{-x}dx.$$

We have therefore generated the portion $0\leq x\leq 1$ of the desired distribution; to get the complete distribution we must do something with the rejected fraction (e^{-1}) of the trials. Since

$$e^{-x}dx=e^{-1}e^{-x+1}dx,$$

what we do is repeat the rejected trial but interpret the new Y_1 differently. If Y_1 from the retrial is accepted, take $X=Y_1+1$. If the retrial is rejected, but the Y_1 from a second retrial is accepted, we take $X=Y_1+2$. Each time a trial is rejected, the range of values of X is increased by unity. A calculation shows that one may expect to choose about

$$(1+e)(1-e^{-1})^{-1}\approx 6$$

values of Y for each X_i selected; the process efficiency is thus about one-sixth. The machine has in effect computed a logarithm by performing only discriminations on the relative magnitude of numbers in $(0, 1)$. It is a sad fact of life, however, that under the particular conditions of the Eniac it was slightly quicker to use a truncated power series for $\log(1-T)$ than to carry out all the discriminations. In conclusion, I should like to mention that the above method can be modified to yield a distribution satisfying any first-order differential equation.