▶ **24.** [*40*] Experiment with various probability distributions (p, q, r) on three categories, where $p+q+r=1$, by computing the exact distribution of the chi-square statistic V for various n, thereby determining how accurate an approximation the chi-square distribution with two degrees of freedom really is.

3.3.2. Empirical Tests

In this section we shall discuss ten kinds of specific tests that have been applied to sequences in order to investigate their randomness. The discussion of each test has two parts: (a) a "plug-in" description of how to perform the test; and (b) a study of the theoretical basis for the test. (Readers lacking mathematical training may wish to skip over the theoretical discussions. Conversely, mathematically-inclined readers may find the associated theory quite interesting, even if they never intend to test random number generators, since some instructive combinatorial questions are involved here.)

Each test is applied to a sequence

$$\langle U_n\rangle = U_0, U_1, U_2, \ldots \tag{1}$$

of real numbers, which purports to be independently and uniformly distributed between zero and one. Some of the tests are designed primarily for integer-valued sequences, instead of the real-valued sequence (1). In this case, the auxiliary sequence

$$\langle Y_n\rangle = Y_0, Y_1, Y_2, \ldots, \tag{2}$$

which is defined by the rule

$$Y_n = \lfloor dU_n \rfloor, \tag{3}$$

is used instead. This is a sequence of integers that purports to be independently and uniformly distributed between 0 and $d-1$. The number d is chosen for convenience; for example, we might have $d = 64 = 2^6$ on a binary computer, so that Y_n represents the six most significant bits of the binary representation of U_n. The value of d should be large enough so that the test is meaningful, but not so large that the test becomes impracticably difficult to carry out.

The quantities U_n, Y_n, and d will have the above significance throughout this section, although the value of d will probably be different in different tests.

A. Equidistribution test (Frequency test). The first requirement that sequence (1) must meet is that its numbers are, in fact, uniformly distributed between zero and one. There are two ways to make this test: (a) Use the Kolmogorov–Smirnov test, with $F(x) = x$ for $0 \le x \le 1$. (b) Let d be a convenient number, e.g., 100 on a decimal computer, 64 or 128 on a binary computer, and use the sequence (2) instead of (1). For each integer r, $0 \le r < d$, count the number of times that $Y_j = r$ for $0 \le j < n$, and then apply the chi-square test using $k = d$ and probability $p_s = 1/d$ for each category.

The theory behind this test has been covered in Section 3.3.1.

> *The equanimity of your average tosser of coins*
> *depends upon a law . . . which ensures that*
> *he will not upset himself by losing too much*
> *nor upset his opponent by winning too often.*
>
> —TOM STOPPARD, *Rosencrantz & Guildenstern are Dead* (1966)

B. Serial test. More generally, we want pairs of successive numbers to be uniformly distributed in an independent manner. The sun comes up just about as often as it goes down, in the long run, but this doesn't make its motion random.

To carry out the serial test, we simply count the number of times that the pair $(Y_{2j}, Y_{2j+1}) = (q, r)$ occurs, for $0 \le j < n$; these counts are to be made for each pair of integers (q, r) with $0 \le q, r < d$, and the chi-square test is applied to these $k = d^2$ categories with probability $1/d^2$ in each category. As with the equidistribution test, d may be any coñvenient number, but it will be somewhat smaller than the values suggested above since a valid chi-square test should have n large compared to k (say $n \ge 5d^2$ at least).

Clearly we can generalize this test to triples, quadruples, etc., instead of pairs (see exercise 2); however, the value of d must then be severely reduced in order to avoid having too many categories. When quadruples and larger numbers of adjacent elements are considered, we therefore make use of less exact tests such as the poker test or the maximum test described below.

Note that $2n$ numbers of the sequence (2) are used in this test in order to make n observations. It would be a mistake to perform the serial test on the pairs (Y_0, Y_1), (Y_1, Y_2), $\ldots$, (Y_{n-1}, Y_n); can the reader see why? We might perform another serial test on the pairs (Y_{2j+1}, Y_{2j+2}), and expect the sequence to pass both tests. Alternatively, I. J. Good has proved that if d is prime, and if the pairs (Y_0, Y_1), (Y_1, Y_2), $\ldots$, (Y_{n-1}, Y_n) are used, and if we use the usual chi-square method to compute both the statistics V_2 for the serial test and V_1 for the frequency test on $Y_0, \ldots, Y_{n-1}$ with the same value of d, then $V_2 - 2V_1$ should have the chi-square distribution with $(d-1)^2$ degrees of freedom when n is large. (See *Proc. Cambridge Phil. Soc.* **49** (1953), 276–284; *Annals Math. Stat.* **28** (1957), 262–264.)

C. Gap test. Another test is used to examine the length of "gaps" between occurrences of U_j in a certain range. If α and β are two real numbers with $0 \le \alpha < \beta \le 1$, we want to consider the lengths of consecutive subsequences U_j, U_{j+1}, $\ldots$, U_{j+r} in which U_{j+r} lies between α and β but the other U's do not. (This subsequence of $r+1$ numbers represents a gap of length r.)

Algorithm G (*Data for gap test*). The following algorithm, applied to the sequence (1) for any given values of α and β, counts the number of gaps of lengths 0, 1, $\ldots$, $t-1$ and the number of gaps of length $\ge t$, until n gaps have been tabulated.

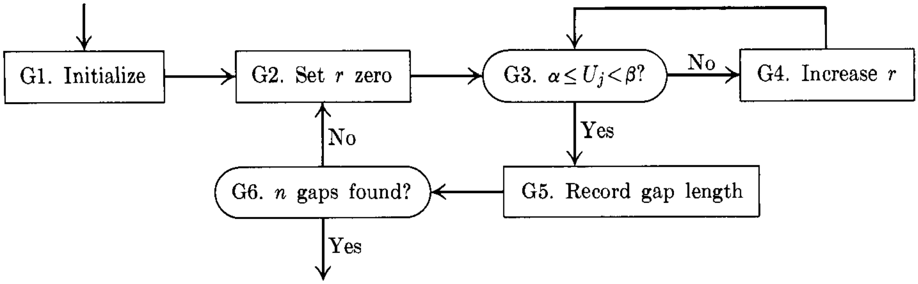

Fig. 6. Gathering data for the gap test. (Algorithms for the "coupon-collector's test" and the "run test" are similar.)

G1. [Initialize.] Set $j \leftarrow -1$, $s \leftarrow 0$, and set COUNT[r] $\leftarrow 0$ for $0 \le r \le t$.

G2. [Set r zero.] Set $r \leftarrow 0$.

G3. [$\alpha \le U_j < \beta$?] Increase j by 1. If $U_j \ge \alpha$ and $U_j < \beta$, go to step G5.

G4. [Increase r.] Increase r by one, and return to step G3.

G5. [Record gap length.] (A gap of length r has now been found.) If $r \ge t$, increase COUNT[t] by one, otherwise increase COUNT[r] by one.

G6. [n gaps found?] Increase s by one. If $s < n$, return to step G2. ▮

After this algorithm has been performed, the chi-square test is applied to the $k = t + 1$ values of COUNT[0], COUNT[1], ..., COUNT[t], using the following probabilities:

$$\begin{aligned} &p_0 = p, \qquad p_1 = p(1-p), \qquad p_2 = p(1-p)^2, \qquad \ldots, \\ &\qquad\qquad p_{t-1} = p(1-p)^{t-1}, \qquad p_t = (1-p)^t. \end{aligned} \tag{4}$$

Here $p = \beta - \alpha$, the probability that $\alpha \le U_j < \beta$. The values of n and t are to be chosen, as usual, so that each of the values of COUNT[r] is expected to be 5 or more, preferably more.

The gap test is often applied with $\alpha = 0$ or $\beta = 1$ in order to omit one of the comparisons in step G3. The special cases $(\alpha, \beta) = (0, \frac{1}{2})$ or $(\frac{1}{2}, 1)$ give rise to tests that are sometimes called "runs above the mean" and "runs below the mean," respectively.

The probabilities in Eq. (4) are easily deduced, so this derivation is left to the reader. Note that the gap test as described above observes the lengths of n *gaps*; it does not observe the gap lengths among *n numbers*. If the sequence $\langle U_n \rangle$ is sufficiently nonrandom, Algorithm G may not terminate. Other gap tests that examine a fixed number of U's have also been proposed (see exercise 5).

D. Poker test (Partition test). The "classical" poker test considers n groups of five successive integers, $(Y_{5j}, Y_{5j+1}, \ldots, Y_{5j+4})$ for $0 \le j < n$, and observes which of the following seven patterns is matched by each quintuple:

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

A chi-square test is based on the number of quintuples in each category.

It is reasonable to ask for a somewhat simpler version of this test, to facilitate the programming involved. A good compromise would simply be to count the number of *distinct* values in the set of five. We would then have five categories:

5 different = all different;
4 different = one pair;
3 different = two pairs, or three of a kind;
2 different = full house, or four of a kind;
1 different = five of a kind.

This breakdown is easier to determine systematically, and the test is nearly as good.

In general we can consider n groups of k successive numbers, and we can count the number of k-tuples with r different values. A chi-square test is then made, using the probability

$$p_r = \frac{d(d-1)\ldots(d-r+1)}{d^k}\left\{ {k \atop r} \right\} \tag{5}$$

that there are r different. (The Stirling numbers $\left\{ {k \atop r} \right\}$ are defined in Section 1.2.6, and they can readily be computed using the formulas given there.) Since the probability p_r is very small when $r = 1$ or 2, we generally lump a few categories of low probability together before the chi-square test is applied.

To derive the proper formula for p_r, we must count how many of the d^k k-tuples of numbers between 0 and $d-1$ have exactly r different elements, and divide the total by d^k. Since $d(d-1)\ldots(d-r+1)$ is the number of ordered choices of r things from a set of d objects, we need only show that $\left\{ {k \atop r} \right\}$ is the number of ways to partition a set of k elements into exactly r parts. Therefore exercise 1.2.6–64 completes the derivation of Eq. (5).

E. Coupon collector's test. This test is related to the poker test somewhat as the gap test is related to the frequency test. The sequence Y_0, Y_1, … is used, and we observe the lengths of segments Y_{j+1}, Y_{j+2}, …, Y_{j+r} required to get a "complete set" of integers from 0 to $d-1$. Algorithm C describes this precisely:

Algorithm C (*Data for coupon collector's test*). Given a sequence of integers Y_0, Y_1, ..., with $0 \le Y_j < d$, this algorithm counts the lengths of n consecutive "coupon collector" segments. At the conclusion of the algorithm, $\mathtt{COUNT}[r]$ is the number of segments with length r, for $d \le r < t$, and $\mathtt{COUNT}[t]$ is the number of segments with length $\ge t$.

```
C1. [Initialize.] Set $j \leftarrow -1$, $s \leftarrow 0$, and set $\mathtt{COUNT}[r] \leftarrow 0$ for $d \le r \le t$.

C2. [Set $q, r$ zero.] Set $q \leftarrow r \leftarrow 0$, and set $\mathtt{OCCURS}[k] \leftarrow 0$ for $0 \le k < d$.

C3. [Next observation.] Increase $r$ and $j$ by 1. If $\mathtt{OCCURS}[Y_j] \ne 0$, repeat this
    step.

C4. [Complete set?] Set $\mathtt{OCCURS}[Y_j] \leftarrow 1$ and $q \leftarrow q + 1$. (The subsequence
    observed so far contains $q$ distinct values; if $q = d$, we therefore have a
    complete set.) If $q < d$, return to step C3.

C5. [Record the length.] If $r \ge t$, increase $\mathtt{COUNT}[t]$ by one, otherwise increase
    $\mathtt{COUNT}[r]$ by one.

C6. [$n$ found?] Increase $s$ by one. If $s < n$, return to step C2. ▮
```

For an example of this algorithm, see exercise 7. 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.

A chi-square test is to be applied to $\mathtt{COUNT}[d]$, $\mathtt{COUNT}[d+1]$, ..., $\mathtt{COUNT}[t]$, with $k = t-d+1$, after Algorithm C has counted n lengths. The corresponding probabilities are

$$p_r = \frac{d!}{d^r}\left\{ {r-1 \atop d-1} \right\}, \qquad d \le r < t; \qquad p_t = 1 - \frac{d!}{d^{t-1}}\left\{ {t-1 \atop d} \right\}. \tag{6}$$

To derive these probabilities, we simply note that if q_r denotes the probability that a subsequence of length r is *incomplete*, then

$$q_r = 1 - \frac{d!}{d^r}\left\{ {r \atop d} \right\}$$

by Eq. (5); for this means we have an r-tuple of elements that do not have all d different values. Then (6) follows from the relations $p_r = q_{r-1} - q_r$ for $d \le r < t$; $p_t = q_{t-1}$.

For formulas that arise in connection with *generalizations* of the coupon collector's test, see exercises 9 and 10 and also the paper by Hermann von Schelling, *AMM* **61** (1954), 306–311.

F. Permutation test. Divide the input sequence into n groups of t elements each, that is, $(U_{jt}, U_{jt+1}, \ldots, U_{jt+t-1})$ for $0 \le j < n$. The elements in each group can have $t!$ possible relative orderings; the number of times each ordering appears is counted, and a chi-square test is applied with $k = t!$ and with probability $1/t!$ for each ordering.

For example, if $t = 3$ we would have six possible categories, according to whether $U_{3j} < U_{3j+1} < U_{3j+2}$ or $U_{3j} < U_{3j+2} < U_{3j+1}$ or $\cdots$ or $U_{3j+2} < U_{3j+1} < U_{3j}$. We assume in this test that equality between U's does not occur; such an assumption is justified, for the probability that two U's are equal is zero.

A convenient way to perform the permutation test on a computer makes use of the following algorithm, which is of interest in itself:

Algorithm P (*Analyze a permutation*). Given a sequence of distinct elements $(U_1, \ldots, U_t)$, we compute an integer $f(U_1, \ldots, U_t)$ such that

$$0 \le f(U_1, \ldots, U_t) < t!,$$

and $f(U_1, \ldots, U_t) = f(V_1, \ldots, V_t)$ if and only if $(U_1, \ldots, U_t)$ and $(V_1, \ldots, V_t)$ have the same relative ordering.

```
P1. [Initialize.] Set $r \leftarrow t$, $f \leftarrow 0$. (During this algorithm we will have $0 \le f < t!/r!$.)
P2. [Find maximum.] Find the maximum of $\{U_1, \ldots, U_r\}$, and suppose that $U_s$ is the maximum. Set $f \leftarrow r \cdot f + s - 1$.
P3. [Exchange.] Exchange $U_r \leftrightarrow U_s$.
P4. [Decrease r.] Decrease $r$ by one. If $r > 1$, return to step P2. ▮
```

Note that the sequence $(U_1, \ldots, U_t)$ will have been sorted into ascending order when this algorithm stops. To prove that the result f uniquely characterizes the *initial* order of $(U_1, \ldots, U_t)$, we note that Algorithm P can be run backwards: For $r = 2, 3, \ldots, t$, set $s \leftarrow f \bmod r$, $f \leftarrow \lfloor f/r \rfloor$, and exchange U_r U_s. It is easy to see that this will undo the effects of steps P2–P4; hence no two permutations can yield the same value of f, and Algorithm P performs as advertised.

The essential idea that underlies Algorithm P is a mixed-radix representation called the "factorial number system": Every integer in the range $0 \le f < t!$ can be uniquely written in the form

$$\begin{aligned} f &= (\ldots(c_{t-1} \times (t-1) + c_{t-2}) \times (t-2) + \cdots + c_2) \times 2 + c_1 \\ &= (t-1)!\,c_{t-1} + (t-2)!\,c_{t-2} + \cdots + 2!\,c_2 + 1!\,c_1 \end{aligned} \tag{7}$$

where the "digits" c_j are integers satisfying

$$0 \le c_j \le j, \qquad \text{for } 1 \le j < t. \tag{8}$$

In Algorithm P, $c_{r-1} = s - 1$ when step P2 is performed for a given value of r.

G. Run test. A sequence may also be tested for "runs up" and "runs down." This means we examine the length of *monotone* subsequences of the original sequence, i.e., segments that are increasing or decreasing.

As an example of the precise definition of a run, consider the sequence of ten numbers "1298536704"; putting a vertical line at the left and right and between X_j and X_{j+1} whenever $X_j > X_{j+1}$, we obtain

$$|1\ \ 2\ \ 9\,|\,8\,|\,5\,|\,3\ \ 6\ \ 7\,|\,0\ \ 4\,|, \tag{9}$$

which displays the "runs up": there is a run of length 3, followed by two runs of length 1, followed by another run of length 3, followed by a run of length 2. The algorithm of exercise 12 shows how to tabulate the length of "runs up."

Unlike the gap test and the coupon collector's test (which are in many other respects similar to this test), *we should not apply a chi-square test to the above data,* since adjacent runs are *not* independent. A long run will tend to be followed by a short run, and conversely. This lack of independence is enough to invalidate a straightforward chi-square test. Instead, the following statistic may be computed, when the run lengths have been determined as in exercise 12:

$$V = \frac{1}{n} \sum_{1\le i,j\le 6} (\texttt{COUNT}[i] - nb_i)(\texttt{COUNT}[j] - nb_j)a_{ij}, \tag{10}$$

where n is the length of the sequence, and the coefficients a_{ij} and b_i are equal to

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{pmatrix} = \begin{pmatrix} 4529.4 & 9044.9 & 13568 & 18091 & 22615 & 27892 \\ 9044.9 & 18097 & 27139 & 36187 & 45234 & 55789 \\ 13568 & 27139 & 40721 & 54281 & 67852 & 83685 \\ 18091 & 36187 & 54281 & 72414 & 90470 & 111580 \\ 22615 & 45234 & 67852 & 90470 & 113262 & 139476 \\ 27892 & 55789 & 83685 & 111580 & 139476 & 172860 \end{pmatrix}$$

$$(b_1\ \ b_2\ \ b_3\ \ b_4\ \ b_5\ \ b_6) = \left(\tfrac{1}{6}\ \ \tfrac{5}{24}\ \ \tfrac{11}{120}\ \ \tfrac{19}{720}\ \ \tfrac{29}{5040}\ \ \tfrac{1}{840}\right). \tag{11}$$

(The values of a_{ij} shown here are approximate only; exact values may be obtained by using formulas derived below.) *The statistic V in* (10) *should have the chi-square distribution with six* (not five) *degrees of freedom,* when n is large. The value of n should be, say, 4000 or more. The same test can be applied to "runs down."

A vastly simpler and more practical run test appears in exercise 14, so a reader who is interested only in testing random number generators should skip the next few pages and go on to the "maximum-of-t test" after looking at exercise 14. On the other hand it is instructive from a mathematical standpoint to see how a complicated run test with interdependent runs can be treated, so we shall now digress for a moment.

Given any permutation on n elements, let $Z_{pi} = 1$ if position i is the beginning of an ascending run of length p or more, and let $Z_{pi} = 0$ otherwise. For example, consider the permutation (9) with $n = 10$; we have

$$Z_{11} = Z_{21} = Z_{31} = Z_{14} = Z_{15} = Z_{16} = Z_{26} = Z_{36} = Z_{19} = Z_{29} = 1,$$

and all other Z's are zero. With this notation,

$$R'_p = Z_{p1} + Z_{p2} + \cdots + Z_{pn} \tag{12}$$

is the number of runs of length $\geq p$, and

$$R_p = R'_p - R'_{p+1} \tag{13}$$

is the number of runs of length p exactly. Our goal is to compute the mean value of R_p, and also the *covariance*

$$\operatorname{covar}(R_p, R_q) = \operatorname{mean}\bigl((R_p - \operatorname{mean}(R_p))(R_q - \operatorname{mean}(R_q))\bigr),$$

which measures the interdependence of R_p and R_q. These mean values are to be computed as the average over the set of all $n!$ permutations.

Equations (12) and (13) show that the answers can be expressed in terms of the mean values of Z_{pi} and of $Z_{pi}Z_{qj}$, so as the first step of the derivation we obtain the following results (assuming that $i < j$):

$$\begin{aligned}
\frac{1}{n!}\sum Z_{pi} &= \begin{cases}(p+\delta_{i1})/(p+1)!, & \text{if } i \leq n-p+1;\\ 0, & \text{otherwise.}\end{cases}\\
\frac{1}{n!}\sum Z_{pi}Z_{pj} &= \begin{cases}(p+\delta_{i1})q/(p+1)!(q+1)!, & \\ & \text{if } i+p < j \leq n-q+1;\\ (p+\delta_{i1})/(p+1)!q! - (p+q+\delta_{i1})/(p+q+1)!, & \\ & \text{if } i+p = j \leq n-q+1;\\ 0, & \text{otherwise.}\end{cases}
\end{aligned}\tag{14}$$

The $\sum$-signs stand for a summation over all possible permutations. To illustrate the calculations involved here, we will work the most difficult case, when $i+p = j \leq n-q+1$, and when $i > 1$. Note that $Z_{pi}Z_{qj}$ is either zero or one, so the summation consists of counting all permutations $U_1U_2\ldots U_n$ for which $Z_{pi} = Z_{qj} = 1$, that is, all permutations such that

$$U_{i-1} > U_i < \cdots < U_{i+p-1} > U_{i+p} < \cdots < U_{i+p+q-1}. \tag{15}$$

The number of such permutations may be enumerated as follows: there are $\binom{n}{p+q+1}$ ways to choose the elements for the positions indicated in (15); there are

$$(p+q+1)\binom{p+q}{p}-\binom{p+q+1}{p+1}-\binom{p+q+1}{1}+1 \tag{16}$$

ways to arrange them in the order (15), as shown in exercise 13; and there are $(n-p-q-1)!$ ways to arrange the remaining elements. Thus there are $\binom{n}{p+q+1}(n-p-q-1)!$ times (16) ways in all, and dividing by $n!$ we get the desired formula.

From relations (14) a rather lengthy calculation leads to

$$\begin{aligned}\operatorname{mean}(R'_p) &= \operatorname{mean}(Z_{p1}+\cdots+Z_{pn})\\ &= (n+1)p/(p+1)!-(p-1)/p!, \qquad 1\le p\le n;\end{aligned} \tag{17}$$

$$\begin{aligned}\operatorname{covar}(R'_p,R'_q) &= \operatorname{mean}(R'_p\,R'_q)-\operatorname{mean}(R'_p)\operatorname{mean}(R'_q)\\ &= \sum_{1\le i,j\le n}\frac{1}{n!}\sum Z_{pi}Z_{pj}-\operatorname{mean}(R'_p)\operatorname{mean}(R'_q)\\ &= \begin{cases}\operatorname{mean}(R'_t)+f(p,q,n), & \text{if } p+q\le n,\\ \operatorname{mean}(R'_t)-\operatorname{mean}(R'_p)\operatorname{mean}(R'_q), & \text{if } p+q>n,\end{cases}\end{aligned} \tag{18}$$

where $t=\max(p,q)$, $s=p+q$, and

$$\begin{aligned}f(p,q,n) &= (n+1)\biggl(\frac{s(1-pq)+pq}{(p+1)!\,(q+1)!}-\frac{2s}{(s+1)!}\biggr)+2\biggl(\frac{s-1}{s!}\biggr)\\ &\quad+\frac{(s^2-s-2)pq-s^2-p^2q^2+1}{(p+1)!\,(q+1)!}.\end{aligned} \tag{19}$$

This expression for the covariance is unfortunately quite complicated, but it is necessary for a successful run test as described above. From these formulas it is easy to compute

$$\begin{aligned}\operatorname{mean}(R_p) &= \operatorname{mean}(R'_p)-\operatorname{mean}(R'_{p+1}),\\ \operatorname{covar}(R_p,R'_q) &= \operatorname{covar}(R'_p,R'_q)-\operatorname{covar}(R'_{p+1},R'_q),\\ \operatorname{covar}(R_p,R_q) &= \operatorname{covar}(R_p,R'_q)-\operatorname{covar}(R_p,R'_{q+1}).\end{aligned} \tag{20}$$

In *Annals Math. Stat.* **15** (1944), 163–165, J. Wolfowitz proved that the quantities $R_1, R_2, \ldots, R_{t-1}, R'_t$ become normally distributed as $n\to\infty$, subject to the mean and covariance expressed above; this implies that the following test for runs is valid: Given a sequence of n random numbers, compute the number of runs R_p of length p for $1\le p<t$, and also the number of runs R'_t of length t or more. Let

$$\begin{gathered}Q_1=R_1-\operatorname{mean}(R_1),\quad \ldots,\quad Q_{t-1}=R_{t-1}-\operatorname{mean}(R_{t-1}),\\ Q_t=R'_t-\operatorname{mean}(R'_t).\end{gathered} \tag{21}$$

Form the matrix C of the covariances of the R's; for example, $C_{13} = \operatorname{covar}(R_1, R_3)$, while $C_{1t} = \operatorname{covar}(R_1, R'_t)$. When $t = 6$, we have

$$C = nC_1 + C_2$$

$$= n\begin{pmatrix} \frac{23}{180} & \frac{-7}{360} & \frac{-5}{336} & \frac{-433}{60480} & \frac{-13}{5670} & \frac{-121}{181440} \\ \frac{-7}{360} & \frac{2843}{20160} & \frac{-989}{20160} & \frac{-7159}{362880} & \frac{-10019}{1814400} & \frac{-1303}{907200} \\ \frac{-5}{336} & \frac{-989}{20160} & \frac{54563}{907200} & \frac{-21311}{1814400} & \frac{-62369}{19958400} & \frac{-7783}{9979200} \\ \frac{-433}{60480} & \frac{-7159}{362880} & \frac{-21311}{1814400} & \frac{886657}{39916800} & \frac{-257699}{239500800} & \frac{-62611}{239500800} \\ \frac{-13}{5670} & \frac{-10019}{1814400} & \frac{-62369}{19958400} & \frac{-257699}{239500800} & \frac{29874811}{5448643200} & \frac{-1407179}{21794572800} \\ \frac{-121}{181440} & \frac{-1303}{907200} & \frac{-7783}{9979200} & \frac{-62611}{239500800} & \frac{-1407179}{21794572800} & \frac{2134697}{1816214400} \end{pmatrix}$$

$$+\begin{pmatrix} \frac{83}{180} & \frac{-29}{180} & \frac{-11}{210} & \frac{-41}{12096} & \frac{91}{25920} & \frac{41}{18144} \\ \frac{-29}{180} & \frac{-305}{4032} & \frac{319}{20160} & \frac{2557}{72576} & \frac{10177}{604800} & \frac{413}{64800} \\ \frac{-11}{210} & \frac{319}{20160} & \frac{-58747}{907200} & \frac{19703}{604800} & \frac{239471}{19958400} & \frac{39517}{9979200} \\ \frac{-41}{12096} & \frac{2557}{72576} & \frac{19703}{604800} & \frac{-220837}{4435200} & \frac{1196401}{239500800} & \frac{360989}{239500800} \\ \frac{91}{25920} & \frac{10177}{604800} & \frac{239471}{19958400} & \frac{1196401}{239500800} & \frac{-139126639}{7264857600} & \frac{4577641}{10897286400} \\ \frac{41}{18144} & \frac{413}{64800} & \frac{39517}{9979200} & \frac{360989}{239500800} & \frac{4577641}{10897286400} & \frac{-122953057}{21794572800} \end{pmatrix} \tag{22}$$

if $n \geq 12$. Now form $A = (a_{ij})$, the inverse of the matrix C, and compute $\sum_{1 \leq i,j \leq t} Q_i Q_j a_{ij}$. The result for large n should have approximately the chi-square distribution with t degrees of freedom.

The matrix (11) given earlier is the inverse of C_1 to five significant figures. When n is large, A will be approximately $(1/n)C_1^{-1}$; a test with $n = 100$ showed that the entries a_{ij} in (11) were each about 1 percent lower than the true values obtained by inverting (22).

H. Maximum-of-t test. For $0 \leq j < n$, let $V_j = \max(U_{tj}, U_{tj+1}, \ldots, U_{tj+t-1})$. Now apply the Kolmogorov–Smirnov test to the sequence V_0, V_1, $\ldots$, V_{n-1}, with the distribution function $F(x) = x^t$, $0 \leq x \leq 1$. Alternatively, apply the equidistribution test to the sequence V_0^t, V_1^t, $\ldots$, V_{n-1}^t.

To verify this test, we must show that the distribution function for the V_j is $F(x) = x^t$. The probability that $\max(U_1, U_2, \ldots, U_t) \leq x$ is the probability that $U_1 \leq x$ and $U_2 \leq x$ and $\ldots$ and $U_t \leq x$, which is the product of the individual probabilities, namely $xx \ldots x = x^t$.

I. Collision test. Chi-square tests can be made only when there is a nontrivial number of items expected in each category. But there is another kind of test that can be used when the number of categories is much larger than the number of observations; this test is related to "hashing," an important method for information retrieval that we shall study in Chapter 6.

Suppose we have m urns and we throw n balls at random into those urns, where m is much greater than n. Most of the balls will land in urns that were previously empty, but if a ball falls into an urn that already contains at least one ball we say that a "collision" has occurred. The collision test counts the number of collisions, and a generator passes this test if it doesn't induce too many or too few collisions.

To fix the ideas, suppose $m = 2^{20}$ and $n = 2^{14}$. Then each urn will receive only one 64th of a ball, on the average. The probability that a given urn will contain exactly k balls is $p_k = \binom{n}{k} m^{-k}(1-m^{-1})^{n-k}$, so the expected number of collisions per urn is $\sum_{k\geq 1}(k-1)p_k = \sum_{k\geq 0} kp_k - \sum_{k\geq 1} p_k = n/m - 1 + p_0$. Since $p_0 = (1-m^{-1})^n = 1 - n/m + \binom{n}{2}m^{-2}$ + smaller terms, we find that the average total number of collisions taken over all m urns is very slightly less than $n^2/2m = 128$.

We can use the collision test to rate a random number generator in a large number of dimensions. For example, when $m = 2^{20}$ and $n = 2^{14}$ we can test the 20-dimensional randomness of a number generator by letting $d = 2$ and forming 20-dimensional vectors $V_j = (Y_{20j}, Y_{20j+1}, \ldots, Y_{20j+19})$ for $0 \leq j < n$. It suffices to keep a table of $m = 2^{20}$ bits to determine collisions, one bit for each possible value of the vector V_j; on a computer with 32 bits per word, this amounts to 2^{15} words. Initially all 2^{20} bits of this table are cleared to zero; then for each V_j, if the corresponding bit is already 1 we record a collision, otherwise we set the bit to 1. This test can also be used in 10 dimensions with $d = 4$, and so on.

To decide if the test is passed, we can use the following table of percentage points when $m = 2^{20}$ and $n = 2^{14}$:

collisions $\leq$	101	108	119	126	134	145	153
with probability	.009	.043	.244	.476	.742	.946	.989

The theory underlying these probabilities is the same we used in the poker test, Eq. (5); the probability that c collisions occur is the probability that $n - c$ urns are occupied, namely

$$\frac{m(m-1)\ldots(m-n+c+1)}{m^n}\left\{ {n \atop n-c} \right\}.$$

Although m and n are very large, it is not difficult to compute these probabilities using the following method:

Algorithm S (*Percentage points for collision test*). Given m and n, this algorithm determines the distribution of the number of collisions that occur when n balls are scattered into m urns. An auxiliary array $A[0]$, $A[1]$, $\ldots$, $A[n]$ of floating point numbers is used for the computation; actually $A[j]$ will be nonzero only for $j_0 \leq j \leq j_1$, and $j_1 - j_0$ will be at most of order $\log n$, so it would be possible to get by with considerably less storage.

S1. [Initialize.] Set $A[j] \leftarrow 0$ for $0 \le j \le n$; then set $A[1] \leftarrow 1$ and $j_0 \leftarrow j_1 \leftarrow 1$. Then do step S2 exactly $n-1$ times and go on to step S3.

S2. [Update probabilities.] (Each time we do this step it corresponds to tossing a ball into an urn; $A[j]$ represents the probability that exactly j of the urns are occupied.) Set $j_1 \leftarrow j_1 + 1$. Then for $j \leftarrow j_1, j_1 - 1, \ldots, j_0$ (in this order), set $A[j] \leftarrow (j/m)A[j] + \big((1+1/m) - (j/m)\big)A[j-1]$. If $A[j]$ has become very small as a result of this calculation, say $A[j] < 10^{-20}$, set $A[j] \leftarrow 0$; and in such a case, if $j = j_1$ decrease j_1 by 1, or if $j = j_0$ increase j_0 by 1.

S3. [Compute the answers.] In this step we make use of an auxiliary table $(T_1, T_2, \ldots, T_{\text{tmax}}) = (.01, .05, .25, .50, .75, .95, .99, 1.00)$ containing the specified percentage points of interest. Set $p \leftarrow 0$, $t \leftarrow 1$, and $j \leftarrow j_0 - 1$. Do the following iteration until $t = \text{tmax}$: Increase j by 1, and set $p \leftarrow p + A[j]$; then if $p > T_t$, output $n - j - 1$ and $1 - p$ (meaning that with probability $1 - p$ there are at most $n - j - 1$ collisions) and repeatedly increase t by 1 until $p \le T_t$. ▌

J. Serial correlation test. We may also compute the following statistic:

$$C = \frac{\begin{array}{r} n(U_0U_1 + U_1U_2 + \cdots + U_{n-2}U_{n-1} + U_{n-1}U_0) \\ -(U_0 + U_1 + \cdots + U_{n-1})^2 \end{array}}{n(U_0^2 + U_1^2 + \cdots + U_{n-1}^2) - (U_0 + U_1 + \cdots + U_{n-1})^2}. \qquad (23)$$

This is the "serial correlation coefficient," which is a measure of the amount U_{j+1} depends on U_j.

Correlation coefficients appear frequently in statistics; if we have n quantities $U_0, U_1, \ldots, U_{n-1}$ and n others $V_0, V_1, \ldots, V_{n-1}$, the correlation coefficient between them is defined to be

$$C = \frac{n\sum(U_jV_j) - (\sum U_j)(\sum V_j)}{\sqrt{\left(n\sum U_j^2 - (\sum U_j)^2\right)\left(n\sum V_j^2 - (\sum V_j)^2\right)}}. \qquad (24)$$

All summations in this formula are to be taken over the range $0 \le j < n$; Eq. (23) is the special case $V_j = U_{(j+1) \bmod n}$. (*Note:* The denominator of (24) is zero when $U_0 = U_1 = \cdots = U_{n-1}$ or $V_0 = V_1 = \cdots = V_{n-1}$; we exclude this case from discussion.)

A correlation coefficient always lies between -1 and $+1$. When it is zero or very small, it indicates that the quantities U_j and V_j are (relatively speaking) independent of each other, but when the correlation coefficient is ± 1 it indicates total linear dependence; in fact $V_j = \alpha \pm \beta U_j$ for all j in such a case, for some constants α and β. (See exercise 17.)

Therefore it is desirable to have C in Eq. (23) close to zero. In actual fact, since U_0U_1 is not completely independent of U_1U_2, the serial correlation coefficient is not expected to be *exactly* zero. (See exercise 18.) A "good" value of C will be between $\mu_n - 2\sigma_n$ and $\mu_n + 2\sigma_n$, where

$$\mu_n = \frac{-1}{n-1}, \qquad \sigma_n = \frac{1}{n-1}\sqrt{\frac{n(n-3)}{n+1}}, \qquad n > 2. \tag{25}$$

We expect C to be between these limits about 95 percent of the time.

Equations (25) are only conjectured at this time, since the exact distribution of C is not known when the U's are uniformly distributed. For the theory when the U's have the *normal* distribution, see the paper by Wilfrid J. Dixon, *Annals Math. Stat.* **15** (1944), 119–144. Empirical evidence suggests that we may safely use the formulas for the mean and standard deviation that have been derived from the assumption of the normal distribution, without much error; these are the values that have been listed in (25). It is known that $\lim_{n\to\infty} \sqrt{n}\sigma_n = 1$; cf. the article by Anderson and Walker, *Annals Math. Stat.* **35** (1964), 1296–1303, where more general results about serial correlations of *dependent* sequences are derived.

Instead of simply computing the correlation coefficient between the observations $(U_0, U_1, \ldots, U_{n-1})$ and their immediate successors $(U_1, \ldots, U_{n-1}, U_0)$, we can also compute it between $(U_0, U_1, \ldots, U_{n-1})$ and any cyclically shifted sequence $(U_q, \ldots, U_{n-1}, U_0, \ldots, U_{q-1})$; the cyclic correlations should be small for $0 < q < n$. A straightforward computation of Eq. (24) for all q would require about n^2 multiplications, but it is actually possible to compute all the correlations in only $O(n \log n)$ steps by using "fast Fourier transforms." (See Section 4.6.4; cf. also L. P. Schmid, *CACM* **8** (1965), 115.)

K. Tests on subsequences. It frequently happens that the external program using our random sequence will call for numbers in batches. For example, if the program works with three random variables X, Y, and Z, it may consistently invoke the generation of three random numbers at a time. In such applications it is important that the subsequences consisting of every *third* term of the original sequence be random. If the program requires q numbers at a time, the sequences

$$U_0, U_q, U_{2q}, \ldots; \quad U_1, U_{q+1}, U_{2q+1}, \ldots; \quad \ldots; \quad U_{q-1}, U_{2q-1}, U_{3q-1}, \ldots$$

can each be put through the tests described above for the original sequence U_0, U_1, U_2, ...

Experience with linear congruential sequences has shown that these derived sequences rarely if ever behave less randomly than the original sequence, unless q has a large factor in common with the period length. On a binary computer with m equal to the word size, for example, a test of the subsequences for $q = 8$ will tend to give the poorest randomness for all $q < 16$; and on a decimal computer, $q = 10$ yields the subsequences most likely to be unsatisfactory. (This can be explained somewhat on the grounds of potency, since such values of q will tend to lower the potency.)