Mixed Congruential Random Number Generators for Decimal Machines*

J. L. Allard, A. R. Dobell and T. E. Hull

University of British Columbia, Vancouver, B.C., Canada

Abstract. Random number generators of the mixed congruential type have recently been proposed. They appear to have some advantages over those of the multiplicative congruential type, but they have not been thoroughly tested. This paper summarizes the results of extensive testing of these generators which has been carried out on a decimal machine. Most results are for word length 10, and special attention is given to simple multipliers which give fast generators. But other word lengths and many other multipliers are considered. A variety of additive constants is also used. It turns out that these mixed generators, in contrast to the multiplicative ones, are not consistently good from a statistical point of view. The cases which are bad seem to belong to a well-defined class which, unfortunately, includes most of the generators associated with the simple multipliers. However, a surprise result is that all generators associated with one of the simplest and fastest multipliers, namely 101, turn out to be consistently good for word lengths greater than seven digits. A final section of the paper suggests a simple theoretical explanation of these experimental results.

1. *Generators To Be Tested*

Almost all random number generators that are used in practice can be obtained as special cases of the following procedure. One begins with the non-negative integers x_0, a, c, and m, where m is the largest. Then one defines $x_1, x_2, \cdots$ to be the non-negative integers less than m generated by

$$x_{i+1} \equiv ax_i + c \pmod{m} \qquad i = 1, 2, \cdots. \quad (1)$$

Finally, the sequence $x_0/m, x_1/m, \cdots$ is taken to be the sequence of random numbers. The hope is that the parameters x_0, a, c, and m have been chosen so that the resulting sequence appears to be drawn at random from the uniform distribution on [0, 1]. Such a sequence is often called "pseudo-random." A general treatment of these generators is given in [6], along with an extensive bibliography.

In practice it is especially convenient to choose m according to the particular computer being used, e.g. 10^{10} or 2^{35}, and then the problem is to choose the remaining parameters x_0, a, and c so that the period of the resulting sequence is as great as possible. Finally, one tests subsequences to see if they appear to be random.

Until recently the only cases considered were those associated with choosing $c = 0$. Corresponding generators are called "multiplicative" congruential generators. By choosing x_0 and a according to certain specifications [6], one can ensure that the resulting sequences are as long as is possible for this case, $c = 0$. For example, the maximal periods are 5×10^8 when $m = 10^{10}$, and 2^{33} when $m = 2^{35}$. Using $m = 10^L$ on a decimal machine, the fastest generators yielding maximal

* Received September, 1962.

period have $a = 10^S + 3$ or $a = 10^S + 11$ for $S \geqq 2$, because then the necessary multiplications are easily effected by shift-and-add instructions.

The multiplicative generators have generally demonstrated good statistical behavior. They have consistently passed the statistical tests which have been designed to test their randomness.

In several recent papers by Coveyou [1], Greenberger [4, 5], and Rotenberg [9], various "mixed" congruential generators ($c \neq 0$) have been proposed and discussed. See also [6, 7, 8, 10].

Theoretically these generators have a number of small advantages over the multiplicative generators. They can have longer periods, they may be used with any starting value, and, on most machines, they can be faster than the fastest multiplicative generators. It is easy to ensure that their periods are m. On a decimal machine one has only to require that $a \equiv 1 \pmod{20}$, and that c be not divisible by either 2 or 5. The fastest such generators have $a = 10^S + 1$ for $S > 1$, because these generators are easily effected by shift-and-add instructions.

Most tests of mixed generators that have been reported so far are quite limited [7, 8, 9]. They suggest that the statistical behavior of these generators is as good as that of the multiplicative generators. However, we will see that more extensive testing soon uncovers examples of mixed generators which are completely unacceptable. Fortunately, most examples turn out to be acceptable, including one of the simplest and fastest.

2. *Statistical Tests*

Many different statistical tests have been suggested for the testing of random number generators. References are given in [6]. We decided to use only the two best known ones throughout most of our investigation, because we first wanted to get an overall picture by examining a large number of cases. We used the frequency test and a serial correlation test. Our first step was to apply these tests to blocks of 1000 numbers each.

For the frequency test we let $f_i \quad (i = 1, 2, \cdots, 10)$ be the number of numbers u satisfying $(i - 1)/10 \leqq u < i/10$ and then we computed

$$\chi_1^2 = \tfrac{1}{100} \sum_{i=1}^{10} (f_i - 100)^2$$

which, for a truly random sequence, is distributed asymptotically as χ^2 with 9 degrees of freedom. This statistic was then computed for each of 100 consecutive blocks. Next we let F_i be the number of the resulting 100 values of χ_1^2 which were between the $(i - 1)$-th and the ith deciles for the χ^2-distribution with 9 degrees of freedom, and we then computed

$$\chi_F^2 = \tfrac{1}{10} \sum_{i=1}^{10} (F_i - 10)^2.$$

For the serial test we let $f_{ij} \quad (i, j = 1, 2, \cdots, 10)$ be the number of numbers

u satisfying $(i-1)/10 \leqq u < i/10$ and followed by a number v satisfying $(j-1)/10 \leqq v < j/10$. We then computed

$$\chi_2^2 = \tfrac{1}{10} \sum_{i,j=1}^{10} (f_{ij} - 10)^2.$$

It is known [2, 3] that $\chi_2^2 - \chi_1^2$ is, for a truly random sequence, distributed asymptotically as χ^2 with 90 degrees of freedom. We calculated $\chi_2^2 - \chi_1^2$ for each of 100 consecutive blocks. Then we let S_i be the number of the 100 values so found which were between the $(i-1)$-th and the ith deciles for χ^2 with 90 degrees of freedom. Finally, we computed

$$\chi_S^2 = \tfrac{1}{10} \sum_{i=1}^{10} (S_i - 10)^2.$$

Thus, the two test statistics χ_F^2 and χ_S^2 were computed for samples of 100,000 elements drawn from the sequence $\{x_n/m\}$ produced by each generator (1). Generally, for each sample only these two values were printed by the program.

In what follows, it will be convenient to refer to a particular generator as being "acceptable" or "good" at a given significance level (.01 for our tests) if the values of χ_F^2 and χ_S^2 are not inconsistent with the hypothesis that they are drawn at random from the χ^2-distribution with 9 degrees of freedom.

If for a particular generator the observed values of χ_F^2 or χ_S^2 exceed 21.7 (the 99 percent level for χ^2 with 9 degrees of freedom), this may be due either to ordinary sampling variability or to regularities in the sequence sufficiently pronounced that the sequence cannot be considered a result of random drawing from the uniform distribution.

From the first consideration alone, one expects that about 1 percent of all generators tested would be "rejected" using the 99 percent level. On the other hand, as a result of the second factor one may observe extremely high values of the test statistics, far in excess of the 99.9 percent, or even the 99.99 percent, levels. The occurrence of these extremely high values we took as an indication that the sequence sampled was too regular to be suitable for use as a pseudo-random sequence.

A multiplier a may be considered as defining a class of generators. Before a single multiplier can be considered satisfactory for use, many generators within the class ought to be tested—that is, the multiplier ought to be tried with many different values of x_0 and c.

We might define an acceptable *multiplier* by requiring that each generator within the associated class be acceptable, but this would not allow for ordinary sampling variability. Therefore it will be convenient to refer to a multiplier as acceptable if, when used with different values of x_0 and c, it yields no extremely high values of χ_F^2 or χ_S^2, and no more than about 1 percent in excess of 21.7.

3. *General Results for Word Length 10*

For machines with word length 10 it is natural to choose $m = 10^{10}$. We decided to begin our tests with an overall survey using this value of m.

Preliminary tests had suggested that the statistical behavior of a sequence depended primarily on the multiplier a. We therefore hoped to obtain a good cross-section of the different possibilities by first fixing x_0 and c, and then letting a vary systematically over representative samples of the values yielding maximal period. We chose $x_0 = 0$ and $c = 1$, and we concentrated on values of a which were either relatively small or were congruent to 1 modulo fairly large numbers. We felt that by thus choosing the parameters to be relatively simple we would have the best chance of uncovering unacceptable cases, if any existed.

Since the choice of x_0 and c generally has not a great effect on the acceptability of a generator, in this section we pronounce judgment on some multipliers on the basis of these tests of a single representative generator. Fuller testing of multipliers of special interest is reported in the next section.

We tested more than 1000 different multipliers under these circumstances. On the basis of this sample, we have come to the conclusion that less than 1 percent of all possible multipliers giving maximum period are unacceptable. Moreover, with the single exception of $a = 21$, all the unacceptable multipliers are included in the 4 percent of all possible multipliers which are congruent to 1 (mod 500). These conclusions were at first based primarily on results for the multipliers $a = 1001(20)1501(100)18001(500)44001(4000)230001(10000)540001$.

In searching for a pattern which might characterize the unacceptable multipliers, we also tried other runs, such as $a = 1000501(500)1101501$, where the emphasis was on those which appeared to be bad. We noticed that the unacceptable multipliers constituted just over half of those which were congruent to 1 (mod 2500), most of those congruent to 1 (mod 4000), and all of those congruent to 1 (mod 5000). It is in this sense that our results tend to characterize bad multipliers as those which are congruent to 1 modulo fairly large numbers. Table 1 illustrates these results.

Later we wanted to test more carefully the idea that unacceptable multipliers *must* be congruent to 1 (mod 500), and we tried $a = (10^S + 1)(20)\ (10^S + 501)$ for $S = 4, 5, \cdots, 9$. Table 2 shows some of the results for the case $S = 5$, which are typical. Incidentally, the case $S = 5$ is of special interest because it has been stated elsewhere [1, 4, 5] that multipliers near to $\sqrt{m}$ should have small serial correlations. The statements were based on a theoretical estimate of the serial correlation for a sequence covering the full period, and therefore need not be applicable to the shorter sequences such as we are testing.

We also tried $a = 21(20)981$. It was then that we found $a = 21$ to be unacceptable. It gave $\chi_r^2 = 10.0$, but $\chi_s^2 = 258.6$. In passing, we remark that $a = 21$ can also be used as the multiplier for a multiplicative congruential generator and will give the maximum period for such generators of 5×10^8. But it is not acceptable there either. Using $x_0 = 1$, we obtained $\chi_r^2 = 9.8$, but $\chi_s^2 = 279.8$.

Later we also remark that $a = 101$ yields an acceptable generator, giving $\chi_r^2 = 20.6$ and $\chi_s^2 = 3.8$. In connection with this multiplier it should be pointed out that it is the only one of the simple, and therefore fast, multipliers of the form $10^S + 1$ which is not congruent to 1 (mod 500).

TABLE 1. Typical results for multipliers which satisfy the condition $a \equiv 1 \pmod{500}$. Except for $a = 21$, all unacceptable multipliers satisfy this condition. The results are for $x_0 = 0$ and $c = 1$. The 99 percent level is 21.7.

a	χ_F^2	χ_S^2
501	5.4	7.4
1001	6.0	9.6
1501	10.8	5.8
2001	10.6	17.8
2501	96.6	25.8
3001	5.2	5.6
3501	11.8	5.6
4001	88.4	12.2
4501	6.8	11.4
5001	193.2	131.8
5501	14.2	6.8
6001	10.2	4.4
6501	6.4	9.2
7001	9.2	6.6
7501	45.0	13.6
8001	39.6	8.4
8501	10.2	7.8
9001	4.4	9.6
9501	11.2	7.4
10001	238.2	124.8
10501	4.4	14.0
11001	4.2	11.0
11501	5.2	24.0
12001	41.4	22.8
⋮	⋮	⋮

TABLE 2. Typical results for multipliers which are congruent to 1 (mod 20). Only the first one is not acceptable. The results are for $x_0 = 0$ and $c = 1$.

a	χ_F^2	χ_S^2
100001	520.8	900.0
100021	4.4	14.8
100041	6.2	2.6
100061	10.8	12.8
100081	11.6	14.6
100101	9.8	3.2
100121	6.2	6.4
100141	7.0	10.0
100161	8.6	5.4
100181	20.6	16.2
100201	12.2	7.8
100221	5.0	5.6
⋮	⋮	⋮
100481	10.4	14.4
100501	4.2	18.0
⋮	⋮	⋮

We next consider briefly the effect of changing either x_0 or c.

The value of x_0 was changed in some of the cases of this section, as well as in all cases of the next section. In general, these changes did not noticeably affect the acceptability of the results.

The effect of changing c was more noticeable. When a multiplier was very bad with $c = 1$, more complicated values of c would almost always cause the results to improve, but not enough to make them acceptable. For an example, see Table 3. We found no further relationship between the complexity of c and the degree of improvement. When results were only moderately bad with $c = 1$, more complicated values of c would often make them acceptable. For example, see Table 4.

Unacceptable results were usually caused by values of the individual χ_1^2 and χ_2^2 being too large. It occasionally happened, however, that χ_F^2 was unacceptable, not because of too many large values of χ_1^2, but because the values of χ_1^2 were too nearly the same, and similarly with χ_S^2. This happened only rarely, but

it does illustrate the possibility of having good local behavior along with bad global behavior. One usually worries about the possibility of bad local behavior even though the global behavior might be good.

4. *Effect of Changing the Word Length*

We now consider word lengths other than 10. One reason for doing so is that there are machines with word lengths 6 and 12. Moreover, word length 8 might be useful for an algebraic compiler on a machine with word length 10, when 8 digits are used for the "mantissa" of a floating-point number. Also, with variable-word-length machines it would be useful to have generators with as small a modulus as possible in order to take advantage of the shorter generating times that would result.

For these experiments we concentrated our attention on multipliers of the form $10^S + 1$, but we considered many different values of c for each multiplier and three values of x_0 for each generator. Specifically, for each word length $L = 6, 7, \cdots, 12$, we tried most of the values $S = 2, 3, \cdots, L-1$. For each L and S we took $c = 1, 3, 7, 9, 11, 33, 77, 99, 111, \cdots, m-1$, and for each generator we took three sequences of 100,000 each from the first 300,000 numbers beginning with $x_0 = 0$.

The results were remarkably consistent. In general, the larger the value of S the worse the multiplier, and the smaller the value of L the worse the multiplier. Specifically, the multiplier for $S = 2$ was good for all x_0 and c, and for all values of L with the exception of $L = 6$ and a few cases for $L = 7$. The multiplier for $S = 3$ was good only for $L = 11, 12$, although it was only marginally bad for $L = 10$. The other multipliers for $S = 4, 5, \cdots, L-1$ were all bad. Table 5 summarizes the results.

For the sake of completeness we consider one further advantage that mixed congruential methods might have over multiplicative ones. It has been claimed that adding a complicated c in a mixed method might scramble the least significant digits more than they are scrambled with multiplicative methods. The advantage would be that several parts of one number might then be used separately to give several random numbers at once. We have not tested this idea

TABLE 3. Examples which illustrate how more complicated values of c can cause the result for a very bad multiplier to improve. The multiplier is $a = 100001$, while $x_0 = 0$.

c	x_{P^2}	x_{S^2}
1	520.8	900.0
111	39.8	786.2
7777	16.4	14.0
87291	15.2	159.8
72911267	12.2	85.6

TABLE 4. Examples which illustrate how more complicated values of c can cause the result for a moderately bad multiplier to become acceptable. The multiplier is $a = 108001$, while $x_0 = 0$.

c	x_{P^2}	x_{S^2}
1	7.4	32.0
7777	6.2	6.6
87291	8.2	12.0

TABLE 5. Condensed results for multipliers of the form $10^S + 1$. Entries are percentages of χ_P^2 or χ_S^2 which exceeded the 99 percent level of 21.7. Brackets indicate entries for which the number of cases was less than indicated at the top of the column.

Word length L		6	7	8	9	10	11	12
Number of cases per multiplier		144	168	192	216	240	264	288
Multiplier a	$10^2 + 1$	64	4	1	2	3*	2	1
	$10^3 + 1$	100	84	73	45	7	1	1
	$10^4 + 1$	100	100	88	76	90	(60)	26
	$10^5 + 1$	(100)	100	100	89	94		(100)
	$10^6 + 1$		100	100	100	91		
	$10^7 + 1$			100	100	100		
	$10^8 + 1$				(100)	100		
	$10^9 + 1$					100		

* Rounded from 2.5.

thoroughly, but the results of this section are applicable. For example, the results for word length 8 are exactly the results one would get if one tested only the third digits of the numbers of word length 10. From the results of this section we therefore conclude that only the first 3 digits of the 10-digit numbers are reliable, and this is probably no better than would be obtained with multiplicative methods. In any event we have not tested for any correlation between the different digits of each number.

5. *The Multiplier 101*

The only one of the simple shift-and-add multipliers that has been consistently good so far has been the one corresponding to $S = 2$, namely 101. It appears to be reliable for all word lengths $L \geqq 8$.

As a final test of this multiplier we decided to carry out the serial test using only every second number in each sequence of 200,000 numbers and then using only every third number in each sequence of 300,000 numbers. These tests for serial correlations of lag 2 and lag 3 would be of interest in various applications—for example, if one wanted to generate points in the plane or in space. These tests were performed only for word length 10. But we tried $c = 1, 3, 7, 9, 11, 33, 77, 999, 111, \cdots, 10^{10}-1$, and for each value of c we used the three sequences following each other, beginning with $x_0 = 0$.

Table 6 shows some typical results. The multiplier 101 has once again turned out to be acceptable. (By contrast, 1001 was unacceptable, being particularly so for lag 2.) In these and the earlier experiments, we have deliberately chosen special values of c which might be expected to lead to patterns in the generated sequences, if any choices are going to lead to such patterns. Under these circumstances the results definitely lead to the conclusion that we can rely on the multiplier 101 as long as the word length is at least 8.

TABLE 6. Typical results of the serial test for lags 1, 2 and 3, using different values of c with the multiplier 101.

c	lag 1	lag 2	lag 3
1	3.8	6.4	15.0
33	8.4	7.2	4.6
7777	12.0	5.4	11.6
999999	5.4	8.0	3.0

A final remark should be made concerning the speed of the shift-and-add multipliers. Generators associated with the multiplier 101 will in general be the fastest of all the possible mixed or multiplicative congruential generators. Such a generator will ordinarily require the following sequence of instructions: FETCH x, SHIFT 2 PLACES, ADD x, ADD c, STORE x. The fastest multiplicative generators are associated with the multipliers 10^s+3 and 10^s+11, which would ordinarily require an extra instruction. However, the relative speeds of different generators will depend also on the machine itself. For example, on an IBM 1620 with its variable word length and its two-address instructions a multiplicative generator with $a = 10^6 + 11$ is actually faster (requiring about 2 msec) than a mixed generator with $a = 101$ (requiring about 2.5 msec). By comparison, the generation time would be about 20 msec if a shift-and-add multiplier were not used.

6. *An Analytical Interpretation*

In this section we present a brief analysis which helps to explain the results outlined above and which supports the conclusions drawn there.

Repeated application of the defining relation (1) yields

$$x_n \equiv a^n x_0 + \frac{(a^n - 1)c}{a - 1} \pmod{m}. \tag{2}$$

We assume throughout that the multiplier a has the form

$$a = 10k + 1, \quad k \text{ an even integer}; \tag{3}$$

with m a power of 10 and c relatively prime to m, this condition is sufficient to guarantee that the generator (1) has full period m [6]. Then x_0 determines only the starting point in the full sequence characterized by a, c, and m, and we may without loss of generality suppose $x_0 = 0$. We take $m = 10^L$, and for the moment assume $c = 1$.

Then the expression for x_n becomes

$$x_n \equiv \sum_{j=0}^{L-1} \binom{n}{j+1} k^j 10^j \pmod{10^L} \tag{4}$$

when it is observed that no terms having a factor 10^i, $i \geqq L$, will appear in a number of L digits.

The three principal conclusions which evolved from the results described in preceding sections were:

(i) sequences associated with multipliers of the form

$$a = 10^S + 1 \tag{5}$$

display suitable statistical properties only if S is small relative to the word length L (see Table 5);

(ii) other sequences displaying unsatisfactory statistical behavior are associated with multipliers falling in fairly well-defined residue classes;

(iii) the statistical properties of a sequence may be improved somewhat by a suitable choice of c (see Tables 3 and 4).

Relation (4) suggests an explanation of the results summarized above. Suppose that the multiplier a has the form (5). Then (4) becomes

$$x_n \equiv \sum_j \binom{n}{j+1} 10^{Sj} \pmod{10^L}, \tag{6}$$

and the index j will run only from zero to the largest integer not exceeding $(L-1)/S$. Thus if $S \geqq L/2$, at most two terms will appear in sum (6). That is,

$$x_n \equiv n + \binom{n}{2} \times 10^S \pmod{10^L} \tag{7}$$

so that

$$x_{n+1} \equiv x_n + 10^S \times n + 1 \pmod{10^L}. \tag{8}$$

It is not surprising that such a generator is found to have unsatisfactory statistical properties.

As L/S grows, additional terms enter sum (6), and satisfactory statistical behavior becomes possible. Table 5 suggests that at least four, and probably five terms of the sum are necessary before even our (minimal) statistical requirements are met.

Rather picturesque considerations suggest that not only the number of terms in the expansion (4), but also their size and the amount of "overlap" of terms will influence the statistical properties of the generator. Introducing multipliers of the more general form (3) may have one of two effects.

If, in expression (3), k has factors relatively prime to 10, then these factors and their powers remain to "enlarge" each term in expansion (4), thereby creating "overlap" of terms, and thus more effectively "scrambling" the digits of successive elements of $\{x_n\}$. Exactly the same argument may be offered to explain why more complicated values of c tend to improve results somewhat. For if $c \neq 1$, expression (4) reads

$$x_n \equiv c \sum_j \binom{n}{j+1} k^j 10^j \pmod{10^L}, \tag{9}$$

and c—chosen always relatively prime to 10—remains as a common factor in each term of the expansion.

If, on the other hand, k contains factors which are divisors of 10, these may

regularly combine with the binomial coefficients to create powers of 10—simple shifts of terms leftwards—which simplify, and reduce the number of, terms in expansion (4).

To illustrate these considerations, compare the generators associated with multipliers $a = 10^3 + 1$, $a = 4.10^3 + 1$, $a = 7.10^3 + 1$, $a = 10^4 + 1$. The corresponding expansions are

$$x_n \equiv \sum \binom{n}{j+1} 10^{3j} \pmod{10^L} \tag{10}$$

$$x_n \equiv \sum \binom{n}{j+1} 4^j\, 10^{3j} \pmod{10^L} \tag{11}$$

$$x_n \equiv \sum \binom{n}{j+1} 7^j\, 10^{3j} \pmod{10^L} \tag{12}$$

$$x_n \equiv \sum \binom{n}{j+1} 10^{4j} \pmod{10^L}. \tag{13}$$

For $L = 10$, expansion (13) contains at most three terms; the corresponding generator was unacceptable in our tests. (See Table 1.) Expansion (10) contains four terms and appears to yield (with $c = 1$) an acceptable sequence. Generators with $a = 4001$ were found to be much worse. A possible explanation follows.

Expansion (11) may have four terms, unless $\binom{n}{j+1}$ has a factor 5. But the binomial coefficient $\binom{n}{j+1}$ has a factor 5 whenever $n \not\equiv 4 \pmod 5$, and thus the sequence generated by (11) has runs of four consecutive elements out of five determined by only three terms.

Expression (12), on the other hand, has four terms throughout, each magnified by a factor of 7^j. It displays satisfactory statistical properties.

This argument, then, leads to the conclusion that multipliers for which $(a - 1)/20$ is relatively prime to 10 may be most suitable. The discussion is not completely precise, but it does focus on two obvious considerations affecting the properties of a sequence—namely, the number and nature of the terms in expansion (4)—which provide a simple and consistent interpretation of the experimental results. It demonstrates convincingly that if $m = 10^L$, multipliers of the form $a = 10^s + 1$ will generally give poor results unless a is much smaller than $10^{L/2}$. It suggests that more suitable multipliers may be found among those for which $(a - 1)$ has factors which do not divide 10, along with the fast and simple multiplier $a = 101$.

Our tests were designed to isolate generators which did not meet even plausible minimum requirements. Though we have spoken of many multipliers as "good" or "acceptable," it remains of course to test any of these more exactingly to determine their suitability in particular uses.

ACKNOWLEDGMENT

The results described in this paper were obtained on an IBM 1620 in the Computing Centre at the University of British Columbia.

REFERENCES

1. COVEYOU, R. R. Serial correlation in the generation of pseudo-random numbers. *J.ACM 7* (1960), 72–74.
2. GOOD, I. J. The serial test for sampling numbers and other tests of randomness. *Proc. Camb. Phil. Soc. 49* (1953), 276–284.
3. GOOD, I. J. On the serial test for random sequences. *Ann. Math. Stat. 28* (1957), 262–264.
4. GREENBERGER, MARTIN. Notes on a new pseudo-random number generator. *J.ACM 8* (1961), 163–167.
5. GREENBERGER, MARTIN. An a priori determination of serial correlation in computer generated random numbers. *Math. Comput. 15* (1961), 383–389. See also: corrigenda, *Math. Comput. 16* (1962), 126 and 406.
6. HULL, T. E.; DOBELL, A. R. Random number generators. *SIAM Rev. 4* (1962).
7. KUEHN, HEIDI G. A 48-bit pseudo-random number generator. *Comm. ACM 4* (1961), 350–352.
8. PEACH, PAUL. Bias in pseudo-random numbers. *J. Am. Stat. Assoc. 56* (1961), 610–618.
9. ROTENBERG, A. A new pseudo-random number generator. *J.ACM 7* (1960), 75–77.
10. THOMSON, W. E. A modified congruence method of generating pseudo-random numbers. *Comput. J. 1* (1958), 83 and 86.