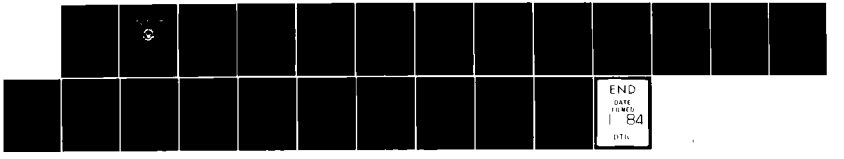
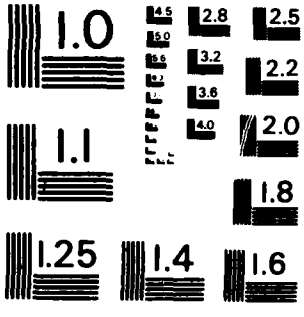
AD-A135 122 EMPIRICAL TESTS OF MULTIPLIERS FOR THE PRIME-MODULUS RANDOM NUMBER GENERA..(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA G P LEARMONTH JUN 77 NPS55-77-30 1/1

UNCLASSIFIED F/G 12/1 NL

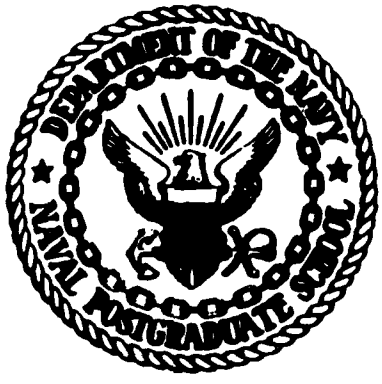




NPS55-77-30

NAVAL POSTGRADUATE SCHOOL

Monterey, California



EMPIRICAL TESTS OF MULTIPLIERS FOR THE PRIME-MODULUS RANDOM NUMBER GENERATOR

$X_{i+1} \equiv AX_i \bmod 2^{31}-1$

by

Gerard P. Learmonth

June 1977

Approved for public release; distribution unlimited.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral Isham Linder
Superintendent

Jack R. Borsting
Provost

Reproduction of all or part of this report is authorized.

This report was prepared by:

Gerard P. Learmonth /ps
Gerard P. Learmonth
University of Michigan

Reviewed by

Michael G. Sovereign, Chairman
Department of Operations Research

Robert Fossum
Dean of Research

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A135122	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Empirical tests of multipliers for the prime-modulus random number generator $x_{i+1} \equiv Ax_i \bmod 2^{31}-1$		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) ~~Gerard~~ P. Learmonth		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, Ca. 93940		12. REPORT DATE June 1977
		13. NUMBER OF PAGES 19
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

Approved for public release; distribution unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Random number generation; Prime-modulus random number generator; Primitive root multipliers; Runs Tests; Serial tests; Self-shuffled generators.

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

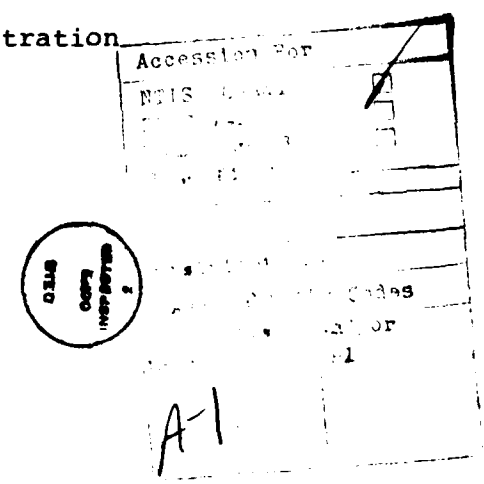
Five primitive root multipliers for the prime-modulus random number generator ~~$x_{i+1} = Ax_i \bmod 2^{31}-1$ have been~~ were subjected to a battery of runs tests and serial tests for pairs and triples. Recommendations regarding these multipliers are made. Interesting results regarding the relative timings of the multipliers are presented. We also give results for the generators with these multipliers after they have been self-shuffled. A case where self-shuffling produced adverse results is also presented.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

EMPIRICAL TESTS OF MULTIPLIERS FOR THE PRIME-MODULUS RANDOM NUMBER GENERATOR $X_{i+1} \equiv AX_1 \bmod 2^{31} - 1$

Gerard P. Learmonth*
University of Michigan
Graduate School of Business Administration
Ann Arbor, Michigan



*This work was done while the author was at the Naval Postgraduate School in consultation with Professor P. A. W. Lewis who was supported by the Office of Naval Research under Grant NR-042-374 and the National Science Foundation under Grant AG-476. This is a revised versio of a paper which appeared in the Proceedings of the Ninth Interface Symposium on Computer Science and Statistics.

ABSTRACT

Five primitive root multipliers for the prime-modulus random number generator $X_{i+1} \equiv AX_i \bmod 2^{31}-1$ have been subjected to a battery of runs tests and serial tests for pairs and triples. Recommendations regarding these multipliers are made. Interesting results regarding the relative timings of the multipliers are presented. We also give results for the generators with these multipliers after they have been self-shuffled. A case where self-shuffling produced adverse results is also presented.

1. Introduction

The purpose of this report is to summarize extensive testing performed on several primitive root multipliers for the prime-modulus random number generator $X_{i+1} \equiv AX_i \bmod 2^{31}-1$. Learmonth and Lewis [4] reported similar results for several different types of generators, including a prime-modulus generator, designed for use on 32-bit computers. The aim here is to attempt to discriminate between different multipliers for a specific type of generator using empirical test results.

The multipliers tested were six primitive roots of $2^{31}-1$ (supplied by Hoaglin and Sande [1]) and three others of interest: 16807, the multiplier used in LLRANDOM and IMSL's GGU3; 46325, a primitive root close to the square root of $2^{31}-1$; and $14^{29} \equiv 630360016$, used by Payne et al. [6]. Of the six proposed by Hoaglin and Sande, four were eliminated in preliminary testing. These appeared adequate statistically, however, their large magnitude made them rather slow when implemented. This point will be touched upon later.

Results are also given for these generators with the additional feature of self-shuffling.

2. The Generator

The purpose here was to examine several different primitive roots as potential multipliers for a specific generator, namely, the Lehmer congruential generator with prime modulus. For 32-bit computers an appropriate choice of modulus is $2^{31}-1$ since this is conveniently the largest prime

expressible in a 32-bit register. The multiplier must be a primitive root of $2^{31} - 1$ if the generator is to achieve a full period of $2^{31} - 2$ without the requirement of an additive constant. Of course not every primitive root of $2^{31} - 1$ will provide a generator whose sequence is adequate statistically. To date, there is no reliable theoretical basis for choosing a primitive root multiplier for any prime modulus which will guarantee good statistical performance.

A variation of the LLRANDOM package was put together for the testing. Since only the multiplier changed from generator to generator, the basic code could remain unchanged with the multiplier and starting value (seed) being arguments to the generator when called. Four entry points were incorporated into the IBM 360 Assembler code. Two entry points returned integer deviates on the interval $[1, 2^{31} - 2]$. One returned the sequence directly, while the other entry point returned the deviates after being shuffled. The other two entry points returned uniform random deviates on the interval (0.0, 1.0); again one was the direct sequence, while the other shuffled the deviates before returning them. The calling sequence to the code required the multiplier to be supplied as well as the starting value.

The shuffling scheme employed was as in the LLRANDOM and GGU3. A table of 128 integer values is maintained within the generator. These values were obtained by running LLRANDOM

with multiplier 16807 and staring value 1 and retaining every millionth integer for the table. When an integer is produced during a call to the generator the low-order seven bits are removed and are used to provide an index to the table. The value at that location in the table is then returned with the newly generated integer replacing the table value. It is felt that this scheme adequately breaks up the serial correlation inherent in linear congruential generators. It uses the idea that the low order bits in a prime modulus congruential generator with a positive primitive root of the modulus as multiplier, the lower bits are fairly random. Knuth (3, p. 12) has noted this but it does not seem to be a well understood phenomenon.

An interesting result came as a by-product of the testing performed here. A generally overlooked fact of the IBM System 360 and 370 computers, and presumably others, is that the integer (fixed-point) arithmetic instruction timings reported are average times. If speed is of importance in the generator, the multiplier should also have relatively few non-zero bits in its binary representation.

Some very early proposals for random number generators used multipliers which were some power of 2 with a small additive factor. Rather than a lengthy multiplication, the generator reduced to a shift-and-add type of operation. With the advances in computer hardware and the availability of the division simulation algorithm used in LLRANDOM, this consideration has been generally abandoned.

Table 1 summarizies the timing results as experienced on a System/360 Model 67. The times reported are the elapsed

CPU times for the runs test and serial test for pairs and triples. These runs were made in a multiprogramming environment (MVT) and do suffer from some contention among tasks running concurrently. Although the absolute times themselves are irrelvant, the relative rankings are a result of the number of bits set to one and their position in each multiplier. Each test run consisted of generating 100 samples of 65536 (2^{16}) deviates and then computing the appropriate test statistics.

The conclusion to be reached from these results is that among statistically satisfactory multipliers, one should opt for the one with the fewest high-order bits set to one in its binary respresentation. This is reinforced by the fact that, since the self-shuffling scheme used in LLRANDOM is very fast and shuffling generally improves the generator (Lewis and Learmonth, 4), one could shuffle with multiplier 16807 and obtain a faster and probably better generator than the straightforward generator with multiplier 2027812802.

Multiplier	No. of one bits	Runs test (seconds)	Serial - 2 (seconds)	Serial - 3 (seconds)
16807	7	304	1179	1399
46325	10	305	1186	1502
397204094	19	551	1268	1506
630360016	13	616	1476	1681
764261123	15	587	1391	1671
1078318381	13	621	1500	1900
1203248318	17	646	1757	1790
1323257245	20	689	1532	1940
2027812802	20	1281	1799	2285

TABLE 1: RELATIVE TIMINGS

3. The Runs Test

The first of the empirical tests applied here is the runs test. This particular test has been one of the most frequently cited tests in the literature on the testing of sequences produced by random number generators.

The development of the runs test antedates the development of digital computers. It was first proposed as a non-parametric test for serial dependence in time series. As a test for randomness, the runs test is the most powerful test against the alternative of first-order Markov dependence in a binary sequence. No other analytic results are known and Lewis and Learmonth [4] concluded that it is a poor test of randomness for random number generators.

For a discussion of the runs test, see Knuth [3]. Levene and Wolfowitz [5] have shown that for the observed number of runs of length d, say N_d, the statistic

$$\frac{N_d - E[N_d]}{(\text{var}[N_d])^{1/2}}$$

is asymptotically normally distributed with mean 0 and variance 1. By counting observed runs of lengths $d = 1,2,\ldots, 7$ and runs of length 8 or greater, these statistics may be combined to form a runs test statistic which has been assumed to be distributed as χ^2_7. This distributional assumption for the runs test statistic has been shown to be rather weak; a simulation result of the true distribution has been given in Lewis and Learmonth [4].

This, rather than compare runs test statistics to a chi-square distribution, empirical estimates of the distribution of the runs test statistic were obtained for each of the five multipliers. These distributional estimates were then compared pairwise by means of two-sample Kolmogorov-Smirnov tests and two-sample Anderson-Darling tests. Under the null hypothesis that the multipliers all produced "random" sequences, these two-sample tests would test this hypothesis on the distribution of the runs test statistics.

As a further comparison, these empirical runs distributions were also compared to the 500-point empirical distribution obtained in [4]. This 500-point distribution estimate was obtained from the results of runs tests performed on five different generators whose sequences were shuffled.

3.1. Results of the runs test

For each multiplier, samples of 2^{16} (65536) were generated, and a runs test statistic was computed. This process was repeated 100 times with independent starting values. Sorting these 100 runs test statistics provided an estimate of the distribution of the statistic for each multiplier. The same procedure was repeated for each multiplier with the sequence shuffled.

The distribution of the two-sample Kolmogorov-Smirnov test generally used is an asymptotic result which is generally felt to hold when both samples are of size 100 or greater. Since the samples here are of size 100, a small-sample version of

the Kolmogorov-Smirnov test due to Kim and Jennrich [2] was used. The values to be reported are the same test statistic, $C/100^2$, and the probability of exceeding that value under the null hypothesis,

$$\Pr\{D_{100,100} > C/100^2\} \quad .$$

For comparisons to the 500-point reference distribution, the asymptotic distribution of the K-S test was employed.

It is known that the K-S test is not sensitive to departures in the tails of the sample distributions. Therefore the Anderson-Darling test was also applied to provide more power in testing the tails. Until quite recently, there was no two-sample Anderson-Darling test. A paper by Pettitt [8] provides an algorithm for such a test and includes small-sample as well as asymptotic percentage points for the test.

In the tables to follow, the first line provides the value of the K-S criterion for the specific pair, followed in parentheses by the probability of exceeding that value under the null hypothesis. On the line immediately below is the Anderson-Darling test statistic. The critical values for this test statistics are 10% (1.933), 5% (2.492), and 1% (3.857). Table values significant at 10% are marked with a single asterisk, and those significant at 5% are marked by double asterisks. The last column presents the two-sample K-S test results against the 500-point reference sample.

From the results in Table 2 it can be surmised that all five multipliers produce distributionally commensurate runs test statistics. There are no grounds to conclude that any of the five differ from one another nor do they differ significantly from the 500-point reference distribution. Again the results on the runs test from [4] which were referred to above must be borne in mind.

Table 3 presents results applied to the five multipliers when their sequences were shuffled before computing the runs test statistics. The intention of shuffling is to break up the natural sequence produced by a generator. In [4] it was demonstrated that the shuffling scheme implemented here was effective in improving the quality of generators known beforehand to be poor. In Table 3 it is apparent that the shuffling has adversely affected one sequence. While this result is somewhat distressing it is intuitively plausible that shuffling can alter a satisfactory sequence into an unsatisfactory sequence. With the exception of the Lehmer multiplier, $14^{29} \equiv 630360016$, the results of shuffling have not changed the conclusions concerning the runs test results for the other multipliers. It is clear that it would be very interesting to investigate why shuffling affected the generator with multiplier 14^{29}, but this has not been undertaken.

4. The Serial Test

Further empirical testing of these five multipliers was performed using the serial test for pairs and triples. For many simulation applications, the k-dimensional uniformity of the generated sequences is an important consideration. The standard implementation of the serial test divides the k-dimensional unit hypercube into r^k smaller hypercubes where r is some power of 2. In this form, the test is essentially testing the k-dimensional uniformity of the first r bits of the generated numbers. By tabulating nonoverlapping k-tuples of the sequences, a contingency table is formed and the ordinary chi-square test is then performed on the table.

For the testing here, overlapped k-tuples were tabulated. For many applications requiring k random numbers, the serial dependence within pairs or triples is vital to the simulation as well as the serial dependence between tests or triples. With overlapping the chi-square distribution theory for the distribution of the test statistic does not hold. Hence the statistic generated from the contingency table does not possess a known distribution; in particular it is not chi-square. Rosenblatt [9] has investigated analytically the serial test for congruential generators with shuffling.

As with the runs tests, empirical estimates of the distribution of this statistic have been generated for each generator tested. For purposes of comparing the multipliers, the empirical serial test distributions were compared with one another using the two-sample Kolmogorov-Smirnov test and the two-sample Anderson-Darling test. Additionally, each

was compared to a 500-point reference distributional estimate, obtained in [4].

Each sample in the empirical estimate consisted of 2^{16} pairs or triples respectively. The pairs were tabulated into a 16 by 16 table, while the triples were tabulated into a 16 by 16 by 16 cube. The first four bits of each number were therefore being tested. (See [4] for more details of the test.) Under the hypothesis of multidimensional uniformity, the expected value for each subcell is known, and a "chi-square" type statistic was computed. To form an estimate of the distribution of this statistic, 100 such samples were computed and sorted for each multiplier.

4.1. Results of the serial tests

Table 4 presents the results of the serial test for pairs when the sequences are not shuffled. Table 5 presents the results for the shuffled sequences. As with the runs test results, the first line presents the sample K-S statistic followed by the probability of exceeding that value under the null hypothesis. The second line presents the Anderson-Darling test statistic. The last column presents the two-sample K-S test results against the 500-point reference distribution.

The results in these two tables are quite as expected. All of the multipliers are distributionally indistinguishable. One slightly suspect pair is marked significant, and this is most likely due to the fact that the multiplier 16807 is known to be

	46325	397204094	630360016	764261123	Reference Distribution
16807	.10 (.5830) .44	.11 (.4695) .43	.09 (.7021) .34	.14 (.2112) 1.00	.9311 (.3512)
46325		.09 (.7021) .85	.09 (.7021) .41	.12 (.3682) 1.11	.7486 (.6296)
397204094			.09 (.7021) .65	.11 (.4695) .57	1.0954 (.1813)
630360016				.13 (.2820) 1.28	.5295 (.9419)
764261123					1.1320 (.1541)

TABLE 2: PAIRWISE COMPARISON OF THE DISTRIBUTION OF RUNS TEST STATISTICS (NOT SHUFFLED).

	46325	.397204094	630360016	764261123	Reference Distribution
16807	.12 (.3682) .71	.07 (.9084) .34	.14 (.2212) 2.13*	.09 (.7921) .34	.4564 (.9853)
46325		.15 (.1549)	.20 (.0241**) 3.17**	.11 (.4695) .31	.8398 (.4809)
397204094			.14 (.2122) 1.55	.10 (.5830) .45	.6938 (.7216)
630360016				.13 (.2829) 2.24*	1.5701 (.0144**)
764261123					.8033 (.5387)

TABLE 3: PAIRWISE COMPAIRSON OF THE DISTRIBUTION OF RUNS TEST STATISTICS (SHUFFLED)

	46325	397204094	630360016	764261123	Reference Distribution
16807	.11 (.4695) .49	.17 (.0783*) 1.67	.12 (.3682) .63	.11 (.4695) 1.59	1.0042 (.2656)
46325		.12 (.3682) .77	.06 (.9684) .23	.11 (.4695) 1.23	.5112 (.9563)
397204094			.10 (.5830) .98	.12 (.3682) 1.30	.9859 (.2854)
630360016				.10 (.5830) 1.22	.4747 (.9779)
764261123					.7303 (.6604)

TABLE 4: PAIRWISE COMPARISON OF THE DISTRIBUTION OF SERIAL TEST STATISTICS FOR OVERLAPPED PAIRS (NOT SHUFFLED)

	46325	397204094	630360016	764261123	Reference Distribution
16807	.08 (.8154) .25	.08 (.8154) .26	.10 (.583) .53	.13 (.2820) 1.40	.4564 (.9853)
46325		.07 (.9084) .35	.08 (.8154) .28	.10 (.5830) .84	.4564 (.9853)
397204094			.09 (.7021) .52	.15 (.1549) 1.53	.5295 (.9419)
630360016				.11 (.4695) 1.06	.6573 (.7807)
764261123					1.1320 (.1541)

TABLE 5: PAIRWISE COMPARISON OF THE DISTRIBUTION OF SERIAL TEST STATISTICS FOR OVERLAPPED PAIRS (SHUFFLED)

weak with respect to pairs (see Table 8a in Lewis and Learmonth [4]). It can be concluded that all of the multipliers are satisfactory in distribution of pairs. It should also be noted that the shuffling has not affect the Lehmer multiplier, $14^{29} \equiv 630360016$, as it did in the runs test. This is bewildering since in the results for triples to follow, the shuffling does have a marked effect on this multiplier.

Tables 6 and 7 present the results of the serial test for consecutive overlapped triples. Without shuffling, the only suspect multiplier is 46325 which is a primitive root close to the square root of $2^{31} - 1$. This multiplier was chosen since it was conjectured that it would perform well in the runs test and serial test for pairs. The evidence here is that the multiplier is weak for triples. When shuffling is applied to the sequences, the results appear satisfactory for all multipliers except $14^{29} \equiv 630360016$. Here again, as in the runs test, the sequence produced by this multiplier performs well without shuffling but becomes very poor when the sequence is shuffled.

	46325	397204094	630360016	764261123	Reference Distribution
16807	.17 (.0783*) 2.53	.06 (.9884) .17	.12 (.3682) .75	.15 (.1549) 1.65	.6390 (.8088)
46325		.15 (.1549) 1.90	.11 (.4695) .72	.09 (.7021) .39	1.3693 (.0470**)
397294094			.09 (.7021) .38	.11 (.4695) 1.21	.4917 (.9970)
630360016				.12 (.3682) .64	.6573 (.7807)
764261123					.9311 (.3512)

TABLE 6: PAIRWISE COMPARISON OF THE DISTRIBUTION OF SERIAL TEST STATISTICS FOR OVERLAPPED TRIPLES (NOT SHUFFLED)

	46325	397204094	630360016	764261123	Reference Distribution
16807	.11 (.4695) .84	.11 (.4695) 1.00	.18 (.0539*) 2.16	.08 (.8154) .37	.7668 (.5990)
46325		.13 (.2820) .83	.25 (.0023**) 3.92**	.08 (.8154) .34	.8933 (.5387)
397204094			.15 (.1549) 1.80	.10 (.5830) .49	.7303 (.6604)
630360016				.22 (.0099**) 2.61	1.7162 (.0055**)
764261123					.4917 (.9970)

TABLE 7: PAIRWISE COMPARISON OF THE DISTRIBUTION OF SERIAL TEST STATISTICS FOR OVERLAPPED TRIPLES (SHUFFLED)

5. CONCLUSIONS

It is generally known that constructing Lehmer congruential generators for 32-bit computers requires considerable care due to the small word size. Prime-modulus generators using $2^{31} - 1$ as modulus provide an important class of generators for these systems. By using primitive-root multipliers and a division simulation algorithm, fast full-period generators can be easily constructed. The question arises then as to the statistical quality of these sequences.

The intent here has been to present results of statistical tests applied to several proposals for primitive root multipliers. The results may be summarized as follows. All of the multipliers tested appear to be quite adequate statistically with the possible exception of 46325. None of the multipliers produced test statistics which differed significantly among themselves nor against an independent reference distribution. The multiplier 46325 did perform poorly with regard to the distribution of triples and is therefore not recommended for use.

As a by-product of this testing, a question about the effect of shuflling has been raised and in particular about the self-shuffling scheme used in LLRANDOM. Specifically, for the Lehmer multiplier, 14^{29}, shuffling has adversely affected the statistical quality of the sequence produced on the runs test and the serial test for triples. Shuffling has been proposed by Marsaglia and Bray [6] and has been implemented in several generator schemes. For certain poor generators, shuffling has provided a simple method for improving the statistical quality of these generators. However, it is now

clear that shuffling is not a panacea for all situations although it does improve most generators [4]. Statistical tests must still be applied to shuffled sequences to ensure that the desired goal has been achieved.

Lastly, the desirability of fast generators, especially for large-scale simulations, requires that primitive-root multipliers be chosen with relatively few nonzero bits in their binary representation subject, of course, to theoretical and statistical validation.

ACKNOWLEDGMENTS

The author would like to express his sincere appreciation to Professor P. A. W. Lewis of the Naval Postgraduate School for his interest and encouragement in this work. Professor Michael Stephans of Stanford University kindly provided the paper by Pettitt on the two-sample Anderson-Darling test. Lastly, the W. R. Church Computer Center of the Naval Postgraduate School provided copious computer time to carry out the test in this report.

REFERENCES

[1] D. C. Hoaglin and G. T. Sande, "A study of multipliers for pseudo-random number generators with modulus $2^{31}-1$," presented at the Annual Meeting of the American Statistical Association, 1974.

[2] P. J. Kim and R. I. Jennrich, "Tables of the exact sampling distribution of the two-sample Kolmogorov-Smirnov criterion $D_{m,n}(m < n)$," in Selected Tables in Mathematical Statistics, Vol. 1, H. L. Harter and D. B. Owens, Eds., Chicago: Markham Publishing Co., 1970.

[3] D. E. Knuth, The Art of Computer Programming: Seminumerical Algorithms, Vol. 2, Reading, Mass.: Addison-Wesley, 1969.

[4] G. P. Learmonth and P. A. W. Lewis, "Statistical tests of some widely used and recently proposed random number generators," in Proceedings of Computer Science and Statistics: 7th Annual Symposium on the Interface, W. J. Kennedy, Ed., Ames, Iowa: Iowa State University Press, 1973.

[5] H. Levene and J. Wolfowitz, "The covariance matrix of runs up and down," Annals of Mathematical Statistics, Vol. 15, 1944.

[6] G. Marsaglia and T. A. Bray, "One-line random number generators and their use in combinations," CACM, Vol. 11, 1968.

[7] W. H. Payne, J. R. Rabung, and T. P. Bogyo, "Coding the Lehmer pseudo-random number generator," CACM, Vol. 12, 1969.

[8] A. N. Pettitt, "A two-sample Anderson-Darling rank statistics," to be published.

[9] M. Rosenblatt, "Multiply schemes and shuffling," Mathematics of Computation, 29, 1975, p. 929.

END

DATE
FILMED

L 84

DTIC