Portable Random Number Generators

Gerald P. Dwyer, Jr.* [a]
K. B. Williams[b]
a. Research Department, Federal Reserve Bank of Atlanta, 104 Marietta St., N.W., Atlanta GA 30303 USA
b. Melbourne Beach, Florida

ABSTRACT

We present a random number generator that is useful for serious computations and can be implemented easily in any language that has 32-bit signed integers, for example C, C^{++} and FORTRAN. This combination generator has a cycle length that would take two millennia to compute on widely used desktop computers. Based on an extensive search, we provide parameter values better than those previously available.

JEL Classifications: C63; C88

Keywords: pseudorandom numbers; pseudorandom number generators; congruential generators; random numbers

September 2000

Acknowledgment.
An earlier version of this paper was presented at the Society for Computation Economics meeting in 1997 and we thank the participants for helpful comments. Bruce McCullough, Daniel Waggoner, Peter Zadrozny and the referees provided helpful comments. Stephen L. Moshier kindly provided many underlying routines used in our programs in addition to helpful comments. Our respective e-mail addresses are gdwyer@dwyerecon.com and kbwms@aol.com. The computer code and detailed tables are available from either author or at http://www.dwyerecon.com, as is a related expositional paper. Any opinions are those of the authors and not necessarily those of the Federal Reserve Bank of Atlanta or of the Board of Governors of the Federal Reserve System.

* Corresponding author. E-mail: gdwyer@dwyerecon.com

I. INTRODUCTION

Economists use computer-generated random numbers in applications that range from the commonplace—simulation—to relatively novel ones—optimization and estimation (Robert and Casella 1999.) In this paper, we examine a generator that is easy to program in virtually any environment and is a generalization of one often used. We provide better parameter values than those previously available.[1]

II. CONGRUENTIAL GENERATORS

There are a large number of pseudorandom number generators available (Niederreiter 1992, Chs. 7-10; Knuth 1998, Ch. 3.) We focus on multiplicative congruential generators in this paper. While generators such as those proposed by Marsaglia and Zaman (1991) and related ones have received a great deal of attention in recent years, not all of this attention is complimentary (L'Ecuyer 1997) and the properties of congruential generators are well understood.

A multiplicative congruential generator is

$$x_i = ax_{i-1} \bmod m. \tag{1}$$

where x_i is the i'th member of the sequence of pseudorandom numbers, a is a multiplier, m is the nonzero modulus and the mod operator means that ax_{i-1} mod m is the least nonnegative remainder from dividing ax_{i-1} by m. Generators such as (1) are used to produce nonnegative integers because arithmetic in integers can be exact. The integers can be transformed to decimal numbers (Monahan 1985.)

Combination Generators

Combining congruential generators provides a powerful generalization of the multiplicative generator. Consider two multiplicative generators used to generate underlying sequences $\{y_i\}$ and $\{z_i\}$ with moduli m_y and m_z where $m_y > m_z$ without loss of generality. The sequences can be added or subtracted but subtraction makes it is easier to avoid overflow. The generator of the combined sequence $\{x_i\}$ is

$$x_i = (y_i - z_i) \bmod m_y. \tag{2}$$

The final mod operation on the difference keeps the sequence of pseudorandom numbers on $[1, m_y - 1]$.

L'Ecuyer and Tezuka (1991) show that the generator (1) is approximately equivalent in important respects to a multiplicative congruential generator with much larger multiplier and modulus. For example, if m_y is 2^{31}-1 and m_z is 2^{31}-19, each about $2.15 \cdot 10^9$, the combination generator is approximately equivalent to a generator with a modulus of about $4.61 \cdot 10^{18}$.

Length of Full Cycle

The length of a full cycle, or the period, of a congruential generator is a mathematical property that can be determined analytically. For multiplicative congruential generators, the best possible full cycle of the difference equation equals the modulus less one, *m-1*, and the values are on $[1, m-1]$ (Knuth 1998, pp. 10-23.) Most combinations of values of the multiplier *a* and the modulus *m* do not generate sequences with the maximum possible period. Prime moduli and some multipliers can produce full cycles.

One common modulus is 2^{31}-1, the largest signed integer representable in a register on many machines and in many languages. The maximum possible period of a multiplicative generator with this modulus is 2^{31}-2, or about 2.15 billion. A couple of billion pseudorandom numbers is not adequate for many applications in economics and finance, a deficiency only worsened if one agrees with Knuth (1998, p. 185) who suggests using no more one-thousandth of a full cycle. Uses of pseudorandom numbers are likely to become increasingly demanding, and indeed, one recent study of stochastic volatilities (Kim et al., 1998) uses almost a full cycle of a congruential generator. It is easy to generate a full cycle: It takes about 1.05 minutes on a Pentium 800 to generate a full cycle of a multiplicative generator with a modulus of 2^{31}-1.

A combined generator can have a dramatically longer period than either of the constituent multiplicative generators. The period of a combination generator based on two generators with prime moduli on the order of 2^{31} can have a period of about $2.31 \cdot 10^{18}$ (L'Ecuyer 1988, p. 744.) If it takes one minute to compute 2^{31}-2 pseudorandom numbers, it would take roughly 2,000 years to generate this cycle.

The Lattice Structure of Congruential Generators

No matter how long or short their periods, congruential generators are deterministic difference equations and phase diagrams can be used to examine their behavior. The points produced by a congruential generator in two or more dimensions lie on hyperplanes. The distance between these hyperplanes varies with the multiplier, which means that some multipliers are better than others.

These insights are used in the spectral test for congruential generators (Knuth 1998, pp. 93-118; Fishman 1996, pp. 611-28; Dwyer and Williams 2000), which finds the maximum distance in any direction between the hyperplanes for a given multiplier. This distance is summarized in a test value that indicates closer hyperplanes when the test value is higher. We have run spectral tests to determine good multipliers for the combined generator. We require that multipliers be approximately factorable (Schrage 1979) for computational reasons, which limits the multipliers considered.

Given two moduli, we performed a *random* search over multipliers. There are too many possible combination multipliers for an exhaustive search given available computational power and there is no regularity in the values from the spectral test. The upper part of Table 1 shows the moduli examined and the number of spectral tests for the alternative combinations of multipliers. The lack of regularity in the test values suggests sampling moduli uniformly, but we did run spectral tests for some moduli than others. We present the results of all spectral tests rather than throw away some results.

The lower part of Table 1 presents the spectral test results for the ten best combination generators. The table presents the multipliers and their associated moduli. The test results are the values of the spectral test and the dimension at which the test attains that value. The dimension is informative because the spectral value is the lowest value attained in an examination of several dimensions, in our case eight, and the dimension is the dimension at which that spectral test value is attained.

Our best generator is significantly better than combination generators previously available. The spectral test result for the combination generator in L'Ecuyer (1988) is 0.39, in L'Ecuyer (1997) is 0.70 and in Knuth (1998, Table 1, line 24 and p. 108) is 0.27.

Properties of Subsets

It is important to test subsets of pseudorandom numbers for apparent deviations from the desired distribution. We ran the set of tests from Knuth (1998) as implemented in Dwyer and Williams (1996) as well as the set of Diehard tests (McCullough 1999.) Our tests include tests for the consistency of the pseudorandom numbers with the underlying distribution, tests for serial correlation of normally-distributed pseudorandom numbers, runs tests and more specialized tests. The best combined generators based on the spectral test easily pass these tests on subsets of various lengths.

Not all readily available generators are adequate. For example, the generators included in the libraries with the Microsoft C^{++} version 4.2 and Borland C^{++} version 4.5 compilers do not pass the tests on subsets of numbers. We conclude that these readily available generators have serious deficiencies.[2] The generator in Gauss version 3.2.38 is a multiplicative congruential generator with a modulus of 2^{31}-1, which has a maximum cycle length of only 2^{31}-2.

III. CONCLUSION

We conclude that the combination generator with our best multipliers is useful for serious computations. The computer code available with this paper will work in any environment that has 32-bit signed integers, and a full cycle from the portable combination generator is orders of magnitude longer than simple congruential generators' cycle. We use the spectral test of the entire sequence of pseudorandom numbers from a combination generator to pick combination generators. In applications, pseudorandom generators produce subsets of these full sequences that are used as if they were draws from some distribution function. Tests on subsets of the pseudorandom numbers do not turn up any problems with the best combination generators. As a bonus, other work shows that the suggested algorithm is reasonably quick relative to alternatives (Dwyer and Williams 2000.)

Table 1

SPECTRAL TESTS AND RESULTS

Moduli Examined and Frequency

First Modulus		Second Modulus		Number of Trials	Fraction of Trials
2^{31}-1	2147483647	2^{31}-1	2147483629	26048975	0.164
		2^{31}-61	2147483587	16004883	0.101
		2^{31}-69	2147483587	4388455	0.028
		2^{31}-85	2147483563	4140000	0.026
		2^{31}-99	2147483549	5039152	0.032
		2^{31}-105	2147483543	39075500	0.246
2^{31}-19	2147483629	2^{31}-61	2147483587	3000000	0.019
		2^{31}-69	2147483587	3000000	0.019
		2^{31}-85	2147483563	3000000	0.019
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	6000000	0.038
2^{31}-61	2147483587	2^{31}-69	2147483587	13172284	0.083
		2^{31}-85	2147483563	4473228	0.028
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	3000000	0.019
2^{31}-69	2147483587	2^{31}-85	2147483563	3000000	0.019
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	3000000	0.019
2^{31}-85	2147483563	2^{31}-99	2147483549	6000000	0.038
		2^{31}-105	2147483543	3000000	0.019
2^{31}-99	2147483549	2^{31}-105	2147483543	4222502	0.027

Best Combination Multipliers

First		Second		Spectral Test	
Multiplier	Modulus	Multiplier	Modulus	Value	Dimension
65670	2147483647	44095	2147483587	0.7616092	8
28078	2147483543	2568	2147483629	0.7587240	6
67142	2147483579	78375	2147483563	0.7548043	7
75756	2147483647	104165	2147483629	0.7536803	5
19391	2147483647	15514	2147483629	0.7513183	8
17916	2147483587	342720	2147483549	0.7509227	6
19995	2147483647	172074	2147483543	0.7507221	8
7332	2147483647	5557	2147483587	0.7503238	7
164130	2147483587	44888	2147483579	0.7500295	7
56599	2147483647	75939	2147483543	0.7491809	4

REFERENCES

Dwyer, Jr., Gerald P., and K. B. Williams. 1996. Testing Random Number Generators. C/C^{++} Users Journal 14, 39-48.

_____, and _____. 2000. Portable Random Number Generators: An Exposition.Unpublished paper, available at http://www.dwyerecon.com/programming/.

Fishman, George S. Monte Carlo. 1996. New York: Springer.

Kim, Sangjoom, Neil Shephard and Siddhartha Chib. 1998. Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models. Review of Economic Studies 65 (July), 361-93.

Knuth, Donald E. 1998. The Art of Computer Programming. Volume 2, Seminumerical Algorithms. Third edition. Reading, Massachusetts: Addison-Wesley Publishing Company.

L'Ecuyer, Pierre. 1988. Efficient and Portable Combined Random Number Generators. Communications of the ACM. 31 (June), 742-49, 774.

_____. 1997, Bad Lattice Structures for Vectors of Non-Successive Values Produced by Some Linear Recurrences. INFORMS Journal on Computing. 9 (Winter), 57-60.

_____, and Tezuka. 1991. Structural Properties for Two Classes of Combined Random Number Generators. Mathematics of Computation 57 (October 1991), 735-46.

Marsaglia, George, and Arif Zaman. 1991. A New Class of Random Number Generators. Annals of Applied Probability 1 (3), 462-80.

McCullough, Bruce. 1999. Econometric Software Reliability: EViews, LIMDEP, SHAZAM and TSP. Journal of Applied Econometrics. 14 (March-April), 191-202 .

Monahan, John F. 1985. Accuracy in Random Number Generation. Mathematics of Computation 45 (October), 559-68.

Niederreiter, Harald. 1992. Random Number Generation and Quasi-Monte Carlo Methods. CBMS-NSF Regional Conference Papers in Applied Mathematics, volume 63. Philadelphia: Society for Industrial and Applied Mathematics.

Press, William H., Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery. 1992. Numerical Recipes in C. Second edition. Cambridge: Cambridge University Press, 1992.

Robert, Christian P., and George Casella. 1999. Monte Carlo Statistical Methods. New York: Springer-Verlag.

Schrage, Linus. 1979. A More Portable Fortran Random Number Generator. ACM Transactions on Mathematical Software 5 (June), 132-38.

ENDNOTES

1. Dwyer and Williams (2000) provide more details about congruential generators and computer code for faster generators than those suggested by Press *et al*. (1992, pp. 274-85), Knuth (1998) and others.

2. We wanted to test the generators in MatLab version 5 but the documentation does not provide sufficient detail to reproduce the generator without substantial, possibly unsuccessful reverse engineering. We are inclined not to use a generator which we cannot reproduce. Knowing the generator and being able to program it is necessary to have reproducible results.

Table 1

SPECTRAL TESTS AND RESULTS

Moduli Examined and Frequency

First Modulus		Second Modulus		Number of Trials	Fraction of Trials
2^{31}-1	2147483647	2^{31}-1	2147483629	26048975	0.164
		2^{31}-61	2147483587	16004883	0.101
		2^{31}-69	2147483587	4388455	0.028
		2^{31}-85	2147483563	4140000	0.026
		2^{31}-99	2147483549	5039152	0.032
		2^{31}-105	2147483543	39075500	0.246
2^{31}-19	2147483629	2^{31}-61	2147483587	3000000	0.019
		2^{31}-69	2147483587	3000000	0.019
		2^{31}-85	2147483563	3000000	0.019
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	6000000	0.038
2^{31}-61	2147483587	2^{31}-69	2147483587	13172284	0.083
		2^{31}-85	2147483563	4473228	0.028
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	3000000	0.019
2^{31}-69	2147483587	2^{31}-85	2147483563	3000000	0.019
		2^{31}-99	2147483549	3000000	0.019
		2^{31}-105	2147483543	3000000	0.019
2^{31}-85	2147483563	2^{31}-99	2147483549	6000000	0.038
		2^{31}-105	2147483543	3000000	0.019
2^{31}-99	2147483549	2^{31}-105	2147483543	4222502	0.027

Best Combination Multipliers

First		Second		Spectral Test	
Multiplier	**Modulus**	**Multiplier**	**Modulus**	**Value**	**Dimension**
65670	2147483647	44095	2147483587	0.7616092	8
28078	2147483543	2568	2147483629	0.7587240	6
67142	2147483579	78375	2147483563	0.7548043	7
75756	2147483647	104165	2147483629	0.7536803	5
19391	2147483647	15514	2147483629	0.7513183	8
17916	2147483587	342720	2147483549	0.7509227	6
19995	2147483647	172074	2147483543	0.7507221	8
7332	2147483647	5557	2147483587	0.7503238	7
164130	2147483587	44888	2147483579	0.7500295	7
56599	2147483647	75939	2147483543	0.7491809	4