Journal of Economic Dynamics & Control
27 (2003) 645–650

JOURNAL OF
Economic
Dynamics
& Control

www.elsevier.com/locate/econbase

Portable random number generators

Gerald P. Dwyer Jr.[a,*], K.B. Williams[b]

[a]*Research Department, Federal Reserve Bank of Atlanta, 1000 Peachtree Str., N.E., Atlanta, GA 30309, USA*
[b]*Melbourne Beach, FL, USA*

Received 11 December 2000; accepted 15 August 2001

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 for this generator. © 2002 Elsevier Science B.V. All rights reserved.

JEL classification: C63; C88

Keywords: Pseudorandom numbers; Pseudorandom number generators; Congruential generators; Random numbers

1. 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

* Corresponding author. Tel.: +1-404-498-7095; fax: +1-404-498-8810.
E-mail address: gdwyer@dwyerecon.com (G.P. Dwyer).

is a generalization of the one often used. We provide better parameter values than those previously available.[1]

2. Congruential generators

There are a large number of pseudorandom number generators available (Niederreiter, 1992, Chapters 7–10; Knuth, 1998, Chapter 3; Gentle, 1998, Chapter 1). 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. Multiplicative congruential generators with distributed lags of past values also have been proposed (L'Ecuyer, 1996), but the extra computation is not always necessary.

A multiplicative congruential generator is

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

where x_i is the ith member of the sequence of pseudorandom numbers, a is a multiplier, m is the nonzero modulus and the mod operator means that $ax_{i-1} \bmod m$ 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).

2.1. 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 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 a much larger multiplier and modulus. For example, if m_y is $2^{31} - 1$ and m_z

[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. 274–285), Knuth (1998) and others.

is $2^{31}-19$, each about 2.15×10^9, the combination generator is approximately equivalent to a generator with a modulus of about 4.61×10^{18}.

2.2. 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 L'Ecuyer and Hellekalek (1998), who suggest using sequences no more than the square root 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 1.03 min 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 2.31×10^{18} (L'Ecuyer, 1988, p. 744). If necessary, a generator with an even longer period is a combined congruential generator with more than one lag in the basic difference equation (1) (L'Ecuyer, 1996).

2.3. 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. 611–628; 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

Table 1

Best combination generators

First		Second		Spectral test	
Multiplier	Modulus	Multiplier	Modulus	Value	Dimension
10064	2147483543	64155	2147483629	0.77742	8
43049	2147483629	16493	2147483563	0.77201	4
204893	2147483579	19206	2147483563	0.76941	7
54863	2147483543	46772	2147483629	0.76473	7
65670	2147483647	44095	2147483587	0.76161	8
44241	2147483647	243976	2147483579	0.76136	6
30036	2147483563	98072	2147483549	0.76126	8
29465	2147483629	17107	2147483549	0.76106	8
2645	2147483647	61160	2147483549	0.76096	4
29155	2147483629	41284	2147483579	0.75903	8

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 full-period multipliers. We examined moduli that are the seven greatest prime numbers less than 2^{31}. There are too many possible combination multipliers for an exhaustive search given available computational power, and there is no regularity in the relationship between spectral-test values and the multipliers. We sampled 20 million or more combinations of multipliers for each set of moduli.

Table 1 presents the spectral test results for the 10 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 generators are better, at least in up to eight dimensions, than combination generators previously available. Our spectral test results for the parameter values suggested 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.[2]

[2] L'Ecuyer (1999a) calculates spectral values for multiplicative congruential generators in 8, 16 and 32 dimensions. L'Ecuyer (1999b) calculates spectral values for combined multiplicative congruential generators with more than one lag in these same dimensions.

2.4. 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.[3] 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 $2^{31} - 2$.

3. 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 drawn 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).

Acknowledgements

An earlier version of this paper was presented at the Society for Computation Economics meeting and we thank the participants for helpful

[3] 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.

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 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.

References

Dwyer Jr., G.P., Williams, K.B., 1996. Testing random number generators. C/C^{++} Users Journal 14, 39–48.

DwyerJr., G.P., Williams, K.B., 2000. Portable random number generators: an exposition. Unpublished paper, available at http://www.dwyerecon.com/programming/.

Fishman, G.S., 1996. Monte Carlo. Springer, New York.

Gentle, J.E., 1998. Random Number Generation and Monte Carlo Methods. Springer, New York.

Kim, S., Shepard, N., Chib, S., 1998. Stochastic volatility: likelihood inference and comparison with ARCH models. Review of Economic Studies 65, 361–393.

Knuth, D.E., 1998. The art of computer programming, In: Seminumerical Algorithms, Vol. 2. 3rd Edition. Addison-Wesley, Reading, MA.

L'Ecuyer, P., 1988. Efficient and portable combined random number generators. Communications of the ACM 31, 742–749, 774.

L'Ecuyer, P., 1996. Combined multiple recursive random number generators. Operations Research 44, 816–822.

L'Ecuyer, P., 1997. Bad lattice structures for vectors of non-successive values produced by some linear recurrences. INFORMS Journal on Computing 9, 57–60.

L'Ecuyer, P., 1999a. Tales of linear congruential generators of different sizes and good lattice structure. Mathematics of Computation 68, 249–260.

L'Ecuyer, P., 1999b. Good parameters and implementations for combined multiple recursive random number generators. Operations Research 47, 159–164.

L'Ecuyer, P., Hellekalek, P., 1998. Random number generators: selection criteria and testing. In: Hellekalek, P., Larcher, G. (Eds.), Random and Quasi-Random Point Sets, Vol. 138, Lecture Notes in Statistics. Springer, New York, pp. 223–265.

L'Ecuyer, P., Tezuka, S., 1991. Structural properties for two classes of combined random number generators. Mathematics of Computation 57, 735–746.

Marsaglia, G., Zaman, A., 1991. A new class of random number generators. Annals of Applied Probability 1 (3), 462–480.

McCullough, B., 1999. Econometric software reliability: EViews, LIMDEP, SHAZAM and TSP. Journal of Applied Econometrics 14, 191–202.

Monahan, J.F., 1985. Accuracy in random number generation. Mathematics of Computation 45, 559–568.

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

Press, W.H., Teukolsky, S., Vetterling, W.T., Flannery, B.P., 1992. Numerical Recipes in C. 2nd Edition. Cambridge University Press, Cambridge.

Robert, C.P., Casella, G., 1999. Monte Carlo Statistical Methods. Springer, New York.

Schrage, L., 1979. A more portable fortran random number generator. ACM Transactions on Mathematical Software 5, 132–138.