Geometric random inner products: A family of tests for random number generators

Shu-Ju Tu* and Ephraim Fischbach†

Department of Physics, Purdue University, West Lafayette, Indiana 47907-1396

(Received 4 October 2002; published 28 January 2003)

We present a computational scheme, GRIP (geometric random inner products), for testing the quality of random number generators. The GRIP formalism utilizes geometric probability techniques to calculate the average scalar products of random vectors distributed in geometric objects, such as circles and spheres. We show that these average scalar products define a family of geometric constants which can be used to evaluate the quality of random number generators. We explicitly apply the GRIP tests to several random number generators frequently used in Monte Carlo simulations, and demonstrate a statistical property for good random number generators.

DOI: 10.1103/PhysRevE.67.016113 PACS number(s): 02.50.Ng

I. INTRODUCTION

Monte Carlo methods are among the most widely used numerical algorithms in computational science and engineering [1]. The key element in a Monte Carlo calculation is the generation of random numbers. Although a truly random number sequence produced by either a physical process such as nuclear decay, an electronic device etc., or by a computer algorithm, may not actually exist, a new and computationally easy-to-implement scheme to investigate random number generators is always highly desirable.

There have been many proposed schemes for the quality measure of random number generators [2–12]. These computational tests are based either on probability theory and statistical methods (for example, the χ^2 test, the Smirnov-Kolmogorov test, the correlation test, the spectral test, and the DieHard battery of randomness tests), or on mathematical modeling and simulation for physical systems (for example: random walks and Ising model simulations). These methods also open the door to studying the properties of random number sequences such as randomness and complexity [13]. Some important attempts at an operational definition of randomness were previously developed by Kolmogorov and Chaitin (algorithmic informational theory) [14–17] and by Pincus (approximate entropy) [18].

In this paper, we study a method to measure n-dimensional randomness which we denote by GRIP (geometric random inner products). One of our main purposes in formulating the GRIP tests is to allow the characterization of geometric correlations which may cause unexpected errors in Monte Carlo simulations. The GRIP family of tests is based on the observation that the average scalar products of random vectors produced in geometric objects (e.g., circles and spheres), define geometric constants which can be used to evaluate the quality of random number generators. After presenting an example of a GRIP test, we exhibit a computational method for implementing GRIP, which is then used to analyze a number of random number generators. We then discuss the GRIP formalism in detail and show how a random number sequence, when converted to random points in a space defined by a geometric object, can produce a series of known geometric constants. Later we introduce additional members within the GRIP family. We then present results for configurations of four, six, and eight random points in an n ball. Finally, we conclude by discussing how the GRIP test quantifies random number generators by explicitly adding a new geometric property of truly random number sequences along with other known properties studied by previously proposed schemes [2–13].

II. GENERAL DESCRIPTION OF THE GRIP FORMALISM

The GRIP scheme is derived from the theory of random distance distribution for spherical objects, and can be generalized to other geometric objects with arbitrary densities [19,20]. First, three random points ($\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_3$) are independently produced from the sample space defined by a geometric object. We then evaluate the average inner product $\langle \vec{r}_{12}\cdot\vec{r}_{23}\rangle$ constructed from two associated random vectors, $\vec{r}_{12}=\vec{r}_2-\vec{r}_1$ and $\vec{r}_{23}=\vec{r}_3-\vec{r}_2$. For a geometric object such as an n-ball of uniform density with a radius R, the analytical result is a geometric constant which can be expressed in terms of the dimensionality n of the space [19,20]:

$$\langle \vec{r}_{12}\cdot\vec{r}_{23}\rangle_n=-\frac{n}{n+2}R^2. \tag{1}$$

A simple derivation of Eq. (1) can be found in the Appendix.

The following procedures are the numerical implementation of our testing programs. A random number generator is used to produce a series of random points $\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_3$ (n coordinates in the range of $-R$ and R for each point) such that these points are distributed in an n-dimensional spherical ball $\mathbf{B}$ of radius R, where

$$\mathbf{B}=\{(x_1,x_2,\ldots,x_n):x_1^2+x_2^2+\cdots+x_n^2\leqslant R^2\}. \tag{2}$$

Note that the points are accepted only if the condition (2) is satisfied, and rejected otherwise. We then compute a series of values for $\vec{r}_{12}\cdot\vec{r}_{23}$. If $\vec{r}_{12}\cdot\vec{r}_{23}$ is evaluated N times (Monte Carlo steps), then statistically we expect

*Electronic address: sjtu@physics.purdue.edu

†Electronic address: ephraim@physics.purdue.edu

$$\lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N}(\vec{r}_{12}\cdot\vec{r}_{23})_i=-\frac{n}{n+2}R^2, \tag{3}$$

as predicted by Eq. (1).

III. RANDOM NUMBER GENERATORS

The following random number generators are used in the GRIP test.

(1) LCG1—a 32-bit (multiplicative) linear congruential generator [2,3] using

$$x_n=a\times x_{n-1}+c \mod m, \tag{4}$$

where $a=16\,807$, $c=0$, and $m=2^{31}-1$.

(2) LCG2—a 48-bit (multiplicative) linear congruential generator [21] with $a=689\,096\,024\,602\,61$, $c=0$, and $m=2^{48}$.

(3) LCG3—a 48-bit (multiplicative) linear congruential generator [21] with $a=252\,149\,039\,17$, $c=11$, and $m=2^{48}$. We note that LCG3 and drand48, a standard library function in Unix systems, use the same algorithm.

(4) F55a—a lagged Fibonacci generator [2,3] using

$$x_n=(x_{n-p}\odot x_{n-q}) \mod m, \tag{5}$$

where $p=55$, $q=24$, $\odot=+$, and $m=2^{31}$.

(5) F55b—a lagged Fibonacci generator with $p=55$, $q=24$, $\odot=-$, and $m=2^{31}$.

(6) F100—a lagged Fibonacci generator [2] with $p=100$, $q=37$, $\odot=-$, and $m=2^{30}$.

(7) F378—a lagged Fibonacci generator with $p=378$, $q=107$, $\odot=+$, and $m=2^{31}$.

(8) F23209—a lagged Fibonacci generator with $p=23\,209$, $q=9739$, $\odot=+$, and $m=2^{31}$.

(9) R31—a generalized feedback shift register (GFSR) generator [2,8–12] using

$$x_n=x_{n-p}\oplus x_{n-q}, \tag{6}$$

where $p=31$, $q=3$, and $\oplus$ is the bitwise exclusive OR operator.

(10) R250—a GFSR generator with $p=250$ and $q=103$.

(11) R9689—a GFSR generator with $p=9689$ and $q=4187$.

(12) R44497—a GFSR generator with $p=44\,497$ and $q=21\,034$.

(13) R132049—a GFSR generator with $p=132\,049$ and $q=54\,454$.

(14) PENTA31—a four-tap shift-register-sequence random-number generator [9–12,22,23] using

$$x_n=x_{n-p}\oplus x_{n-q_1}\oplus x_{n-q_2}\oplus x_{n-q_3}, \tag{7}$$

where $p=31$, $q_1=23$, $q_2=11$, $q_3=9$, and $\oplus$ is the bitwise exclusive OR operator.

(15) PENTA89—a four-tap shift-register-sequence random-number generator with $p=89$, $q_1=69$, $q_2=40$, and $q_3=20$.

(16) Ziff31—a four-tap shift-register-sequence random-number generator with $p=31$, $q_1=13$, $q_2=8$, and $q_3=3$ [22,23].

(17) Ziff89—a four-tap shift-register-sequence random-number generator with $p=89$, $q_1=61$, $q_2=38$, and $q_3=33$.

(18) Ziff9689—a four-tap shift-register-sequence random-number generator with $p=9689$, $q_1=471$, $q_2=314$, and $q_3=157$.

(19) durxor—a generator selected from the IBM ESSL (Engineering and Scientific Subroutine Library) [24].

(20) durand—a generator selected from the IBM ESSL (Engineering and Scientific Subroutine Library) and the sequence period of durand is shorter than durxor [24].

(21) ran_gen—one of the subroutines in IMSL libraries from Visual Numeric [25].

(22) Random—a Fortran 90/95 standard intrinsic random number generator [26].

(23) Weyl—a Weyl sequence generator [27,28],

$$x_n=\{n\alpha\}, \tag{8}$$

where $\{x\}$ is the fractional part of x, and α is an irrational number such as $\sqrt{2}$.

(24) NWS—a nested Weyl sequence generator [27,28],

$$x_n=\{n\{n\alpha\}\}. \tag{9}$$

(25) SNWS—a shuffled nested Weyl sequence generator [27,28],

$$s_n=M\{n\{n\alpha\}\}+\tfrac{1}{2}, \tag{10}$$

$$x_n=\{s_n\{s_n\alpha\}\}, \tag{11}$$

where M is a large positive integer.

IV. OTHER MEMBERS OF THE GRIP FAMILY

For practical computational purposes, we may wish to transform a random number sequence from a uniform density distribution to one which is nonuniform. One of the most important nonuniform density distributions is the Gaussian (normal) distribution $P(r)$ with mean zero and standard deviation σ,

$$P_n(r)=\frac{1}{(2\pi)^{n/2}\sigma^n}e^{-(1/2)(r^2/\sigma^2)}. \tag{12}$$

Here $\int_0^\infty P_n(r)dr=1$, $r=(x_1^2+\cdots+x_n^2)^{1/2}$, and n is the space dimensionality. One can use either the Box-Muller transformation method to generate a random number sequence with a Gaussian density distribution, or use available subroutines from major computational scientific libraries such as IBM ESSL and IMSL [24,25]. By applying the probability density function of the random distance distribution as discussed in Ref. [20], one can add a new GRIP member to investigate the quality of a Gaussian random number generator, and this new GRIP test can be expressed as

$$\langle\vec{r}_{12}\cdot\vec{r}_{23}\rangle_n=-n\sigma^2. \tag{13}$$

TABLE I. Computed results for $\langle \vec{r}_{12} \cdot \vec{r}_{23} \rangle_n$, where ''Expected'' is the exact result obtained from Eq. (1). For each entry in the table, $N=10^8$ was used. The results have been rounded off to 10 significant digits. See text for additional details.

RNG	$n=3$	Error	Rating	$n=9$	Error	Rating
LCG1	−0.600 023 482 4	0.38574σ	√	−0.818 191 721 0	0.21441σ	√
LCG2	−0.600 063 774 9	1.04753σ	√	−0.818 196 741 8	0.32317σ	√
LCG3	−0.599 965 787 7	0.56203σ	√	−0.818 154 896 6	0.58295σ	√
F55a	−0.600 026 485 5	0.43511σ	√	−0.818 001 573 0	3.90347σ	
F55b	−0.599 962 385 5	0.61798σ	√	−0.818 377 153 0	4.22972σ	
F100	−0.599 922 968 6	1.26561σ	√	−0.818 144 811 9	0.80136σ	√
F378	−0.600 037 616 8	0.61796σ	√	−0.818 241 341 4	1.28886σ	√
F23209	−0.599 950 821 5	0.80803σ	√	−0.818 182 126 6	0.00668σ	√
R31	−0.600 036 514 6	0.59991σ	√	−0.818 138 813 5	0.93172σ	√
R250	−0.599 925 280 4	1.22755σ	√	−0.818 202 857 5	0.45560σ	√
R9689	−0.599 889 642 5	1.81311σ	√	−0.818 139 299 2	0.92077σ	√
R44497	−0.599 829 528 0	2.80110σ	√	−0.818 203 037 1	0.45949σ	√
R132049	−0.599 971 014 7	0.47621σ	√	−0.818 204 495 5	0.49106σ	√
PENTA31	−0.599 872 086 7	2.10175σ	√	−0.818 162 743 0	0.41304σ	√
PENTA89	−0.600 130 419 7	2.14242σ	√	−0.818 172 044 3	0.21164σ	√
Ziff31	−0.599 849 912 2	2.46602σ	√	−0.818 301 441 0	2.59027σ	√
Ziff89	−0.599 972 497 7	0.45181σ	√	−0.818 203 546 6	0.47050σ	√
Ziff9689	−0.599 932 334 3	1.11162σ	√	−0.818 176 395 9	0.11741σ	√
durxor	−0.599 914 506 2	1.40452σ	√	−0.818 119 610 8	1.34706σ	√
durand	−0.599 920 364 2	1.30827σ	√	−0.818 218 547 4	0.79528σ	√
ran_gen	−0.599 838 783 2	2.64900σ	√	−0.818 207 066 1	0.54667σ	√
Random	−0.599 929 863 4	1.15224σ	√	−0.818 250 867 8	1.49512σ	√
NWS	−0.629 874 106 5	463.606σ		−0.825 614 262 9	161.111σ	
SNWS	−0.599 694 521 4	5.01971σ		−0.817 973 418 9	4.51291σ	
Expected	−0.600 000 000 0			−0.818 181 818 1		

A very common situation arises when one has to produce random points uniformly distributed on the surface of an n sphere of radius R. Some general computational techniques for doing this are summarized in Refs. [2,19]. We can then use

$$\langle \vec{r}_{12} \cdot \vec{r}_{23} \rangle_n = -R^2, \tag{14}$$

to examine the quality of such transformed random number generators as discussed in Ref. [29].

Another application of the GRIP formalism is in stochastic geometry. We can design a test scheme for a configuration utilizing any number of random points [29], and these tests can be included in the GRIP family. Among the tests are the following.

(1) Four uniform random points configuration for an n ball of radius R,

$$\langle (\vec{r}_{12} \cdot \vec{r}_{23})(\vec{r}_{34} \cdot \vec{r}_{41}) \rangle_n = \frac{n(n+1)}{(n+2)^2} R^4, \tag{15}$$

$$\langle (\vec{r}_{12} \cdot \vec{r}_{34})(\vec{r}_{23} \cdot \vec{r}_{41}) \rangle_n = \frac{2n}{(n+2)^2} R^4, \tag{16}$$

$$\langle \vec{r}_{13} \cdot \vec{r}_{24} \rangle_n = 0. \tag{17}$$

(2) $2m$ uniform random points configuration for an n ball of radius R,

$$\langle (\vec{r}_{12} \cdot \vec{r}_{23}) \cdots (\vec{r}_{2m-1\ 2m} \cdot \vec{r}_{2m\ 1}) \rangle_n = (-1)^m \frac{n(n^{m-1}+1)}{(n+2)^m} R^{2m}, \tag{18}$$

where $2m$ ($m=2,3,4$, etc.) is a positive even number.

A derivation of Eq. (15) can be found in the Appendix. Equations (16)–(18) can be derived in a similar manner.

V. RESULTS

We summarize the computational results using Eq. (3) when $n=3$ and 9 in Table I. The results obtained from Eq. (18) when $m=2,3,4$ and $n=3$ and 9 are presented in Tables II, III, and IV. Note that in Tables I–IV, RNG denotes the specific random number generator defined in the text, ''Error'' is measured in terms of how many standard derivations σ [8–12] the result differs from the theoretical number in absolute value, and the check marks (√) designate those RNG's where the errors are less than 3σ. We consider those RNG's whose errors are larger than 3σ unacceptable, as they may contain subtle n-dimensional nonrandom patterns hidden in random number sequences produced by those RNG's.

TABLE II. Computed results for $\langle(\vec{r}_{12}\cdot\vec{r}_{23})(\vec{r}_{34}\cdot\vec{r}_{41})\rangle_n$, where "Expected" is the exact result obtained from Eq. (24). For each entry in the table, $N=10^6$ was used. The results have been rounded off to 10 significant digits. See text for additional details.

RNG	$n=3$	Error	Rating	$n=9$	Error	Rating
LCG1	0.480 326 220 7	0.38729σ	√	0.745 160 858 6	1.82074σ	√
LCG2	0.479 312 580 5	0.81684σ	√	0.747 142 341 9	4.47138σ	
LCG3	0.479 800 297 3	0.23710σ	√	0.746 097 024 7	3.07191σ	
F55a	0.479 472 279 1	0.62583σ	√	0.745 044 686 3	1.66400σ	√
F55b	0.480 251 112 9	0.29780σ	√	0.746 028 097 1	2.98178σ	√
F100	0.479 333 365 5	0.79406σ	√	0.747 173 425 4	4.51577σ	
F378	0.479 240 077 4	0.90314σ	√	0.747 946 360 4	5.53490σ	
F23209	0.478 891 696 2	1.31638σ	√	0.745 852 738 1	2.74513σ	√
R31	0.481 674 398 6	1.98036σ	√	0.743 751 213 1	0.06789σ	√
R250	0.479 246 168 2	0.89487σ	√	0.745 934 636 1	2.85605σ	√
R9689	0.480 480 236 5	0.56857σ	√	0.746 449 920 8	3.54730σ	
R44497	0.480 585 752 8	0.69443σ	√	0.745 035 278 8	1.65054σ	√
R132049	0.479 110 831 7	1.05649σ	√	0.746 629 847 5	3.78132σ	
PENTA31	0.479 709 543 3	0.34451σ	√	0.747 164 383 2	4.48989σ	
PENTA89	0.478 951 282 9	1.24614σ	√	0.746 251 912 0	3.27910σ	
Ziff31	0.479 977 626 4	0.02652σ	√	0.745 160 015 7	1.82056σ	√
Ziff89	0.480 608 958 3	0.72163σ	√	0.745 540 987 5	2.33093σ	√
Ziff9689	0.480 253 105 1	0.30062σ	√	0.745 934 238 7	2.85333σ	√
durxor	0.479 836 356 8	0.19433σ	√	0.746 112 464 7	3.09483σ	
durand	0.479 963 554 4	0.04317σ	√	0.746 827 102 8	4.05046σ	
ran_gen	0.479 737 323 6	0.31143σ	√	0.746 557 542 4	3.68475σ	
Random	0.482 301 244 0	2.72522σ	√	0.744 845 091 0	1.40049σ	√
NWS	0.565 091 874 9	86.8004σ		0.729 359 623 4	19.5459σ	
SNWS	0.478 633 590 1	1.62413σ	√	0.746 113 925 1	3.09980σ	
Expected	0.480 000 000 0			0.743 801 652 8		

Hence caution should be exercised when these generators are put into use.

We observe that NWS and Weyl (results not shown) perform poorly in $n=3$ and 9 on all GRIP tests, and hence these are not recommended for any serious Monte Carlo simulation. We also note from the $n=9$ results in Table II that these results are clearly biased to larger values (except R31 and NWS) compared to the expected value, and reveal much larger errors than the other cases. One interpretation may be that $\langle(\vec{r}_{12}\cdot\vec{r}_{23})(\vec{r}_{34}\cdot\vec{r}_{41})\rangle_9$ is a more sensitive and dedicated computational test for investigating random number generators than other GRIP tests. For RNG's such as LCG1, F23209, R250, R44497, Ziff31, Ziff89, Ziff9689, and Random whose errors are less than 3σ in all the GRIP tests, we quantify these RNG's as high quality, although additional tests for different geometric configurations in other dimensions should be further investigated.

Reference [29] contains additional results for random number generators based on modern algorithms such as the data encryption standard (DES) [2,3], and on turbulent electroconvection [30], along with the computed results from Eqs. (13) and (14), and results from other geometric objects such as an n cube.

VI. GRIP ANALYSIS

In the following, we analyze the relationship between GRIP and a random number sequence, and show how a good random number sequence, when converted to random points in a space defined by a geometric object, can produce a series of known n-dimensional geometric constants. A random number sequence generated from a random number generator can be written as

$$a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}\dots, \tag{19}$$

where each number $a_1,a_2,\dots$ has been computed to 16 significant digits in the present work. When the sequence is converted to represent random points in a two-dimensional geometric object, the random numbers in Eq. (19) can then be grouped in pairs as

$$(a_1a_2)(a_3a_4)(a_5a_6)(a_7a_8)(a_9a_{10})\dots, \tag{20}$$

where Cartesian coordinates are used. The first set of random points $\{\vec{r}_1,\vec{r}_2,\vec{r}_3\}$ can thus be identified as

$$\vec{r}_1=(a_1,a_2),\quad \vec{r}_2=(a_3,a_4),\quad \vec{r}_3=(a_5,a_6). \tag{21}$$

GRIP then uses $\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_3$ to evaluate the average scalar product, which can be computed by rewriting

TABLE III. Computed results for $\langle(\vec{r}_{12}\cdot\vec{r}_{23})(\vec{r}_{34}\cdot\vec{r}_{45})(\vec{r}_{56}\cdot\vec{r}_{61})\rangle_n$, where "Expected" is the exact result obtained from Eq. (24). For each entry in the table, $N=10^6$ was used. The results have been rounded off to 10 significant digits. See text for additional details.

RNG	$n=3$	Error	Rating	$n=9$	Error	Rating
LCG1	−0.240 047 538 8	0.06410σ	√	−0.555 872 308 5	1.93763σ	√
LCG2	−0.240 168 887 1	0.22627σ	√	−0.555 599 239 8	1.55591σ	√
LCG3	−0.239 831 802 3	0.22516σ	√	−0.556 386 685 9	2.63984σ	√
F55a	−0.240 281 321 8	0.37700σ	√	−0.554 215 639 8	0.35281σ	√
F55b	−0.240 681 256 5	0.90816σ	√	−0.555 189 014 3	0.99287σ	√
F100	−0.239 614 517 8	0.51725σ	√	−0.555 725 431 1	1.73039σ	√
F378	−0.241 330 520 1	1.77206σ	√	−0.555 505 033 9	1.42599σ	√
F23209	−0.239 049 868 6	1.28062σ	√	−0.554 812 146 3	0.47186σ	√
R31	−0.240 589 812 6	0.78523σ	√	−0.552 606 828 6	2.59612σ	√
R250	−0.239 661 783 7	0.45397σ	√	−0.554 965 423 6	0.68392σ	√
R9689	−0.239 287 902 2	0.95750σ	√	−0.555 024 278 3	0.76382σ	√
R44497	−0.239 187 102 4	1.09028σ	√	−0.554 233 051 3	0.32730σ	√
R132049	−0.239 127 336 7	1.17125σ	√	−0.555 329 893 5	1.18620σ	√
PENTA31	−0.240 104 407 4	0.13988σ	√	−0.556 116 076 9	2.26738σ	√
PENTA89	−0.240 516 074 8	0.68886σ	√	−0.555 130 919 7	0.91090σ	√
Ziff31	−0.239 123 375 9	1.17676σ	√	−0.555 829 828 6	1.86798σ	√
Ziff89	−0.239 424 598 6	0.77000σ	√	−0.555 491 103 1	1.41120σ	√
Ziff9689	−0.239 216 682 3	1.04965σ	√	−0.554 371 687 6	0.13627σ	√
durxor	−0.239 336 568 5	0.88983σ	√	−0.554 875 968 3	0.55897σ	√
durand	−0.239 676 774 6	0.43155σ	√	−0.555 594 143 1	1.54743σ	√
ran_gen	−0.239 893 600 6	0.14276σ	√	−0.555 514 082 2	1.43941σ	√
Random	−0.242 062 399 1	2.74386σ	√	−0.555 326 364 5	1.18388σ	√
NWS	−0.306 122 274 9	73.1801σ		−0.543 001 338 6	16.2303σ	
SNWS	−0.239 966 640 7	0.04487σ	√	−0.555 872 308 5	1.93763σ	√
Expected	−0.240 000 000 0			−0.554 470 323 0		

$$\langle\vec{r}_{12}\cdot\vec{r}_{23}\rangle=\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{2}(a_{6i-4+j}-a_{6i-6+j})\times(a_{6i-2+j}-a_{6i-4+j}),\tag{22}$$

where N is a large positive integer. When the geometric object is a circle of radius R and uniform density, we expect $\langle\vec{r}_{12}\cdot\vec{r}_{23}\rangle\approx-0.5R^2$ as predicted by Eq. (1).

The analysis for two-dimensional GRIP can be immediately generalized to the n-dimensional case. When the sequence in Eq. (19) is used to generate random points in an n-dimensional spherical object, we can regroup Eq. (19) as follows:

$$(a_1\cdots a_n)(a_{n+1}\cdots a_{2n})(a_{2n+1}\cdots a_{3n})(\cdots)(\cdots)(\cdots)\ldots.\tag{23}$$

The average scalar product of $\vec{r}_{12}\cdot\vec{r}_{23}$ can then be expressed as

$$\langle\vec{r}_{12}\cdot\vec{r}_{23}\rangle=\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n}(a_{3in-2n+j}-a_{3in-3n+j})\times(a_{3in-n+j}-a_{3in-2n+j}).\tag{24}$$

When the geometric object is an n ball with a radius $R=1$ and a uniform density, we expect from Eq. (1) that the result of Eq. (24) should be a geometric constant, $-n/(n+2)$.

VII. CONCLUSIONS

We have presented a computational paradigm, GRIP, for evaluating the quality of random number generators in multiple (n-dimensional) levels. We then demonstrate that GRIP gives rise to a geometric property characterizing truly random number generators. We have shown how a random number sequence, when converted to random points in a space defined by a geometric object, can produce a series of known geometric constants. Several random number generators were selected to run our GRIP tests, and they are graded based on the 3σ error criterion. Finally, we note that one implication of our work is that computational scientists should test the random number generators they use in their simulations, and verify that their random number generators pass as many of the proposed tests as possible.

ACKNOWLEDGMENTS

The authors wish to thank R. Ziff for correspondence and for providing references prior publication, T. K. Kuo and Dave Seaman for helpful discussions, and the Purdue Uni-

TABLE IV. Computed results for $\langle(\vec{r}_{12}\cdot\vec{r}_{23})(\vec{r}_{34}\cdot\vec{r}_{45})(\vec{r}_{56}\cdot\vec{r}_{67})(\vec{r}_{78}\cdot\vec{r}_{81})\rangle_n$, where "Expected" is the exact result obtained from Eq. (24). For each entry in the table, $N=10^6$ was used. See text for additional details.

RNG	$n=3$	Error	Rating	$n=9$	Error	Rating
LCG1	0.135 140 448 36	1.14563σ	√	0.448 409 903 3	0.45119σ	√
LCG2	0.134 603 480 25	0.31160σ	√	0.447 900 972 0	1.14776σ	√
LCG3	0.134 594 842 74	0.30013σ	√	0.450 139 109 2	1.90643σ	√
F55a	0.134 793 830 40	0.60703σ	√	0.447 299 025 3	1.97972σ	√
F55b	0.134 809 559 91	0.62657σ	√	0.449 843 729 7	1.50773σ	√
F100	0.135 268 986 61	1.34118σ	√	0.449 237 545 6	0.68210σ	√
F378	0.133 742 948 67	1.02105σ	√	0.449 082 210 0	0.46735σ	√
F23209	0.133 646 768 54	1.17825σ	√	0.448 361 681 9	0.51580σ	√
R31	0.137 021 008 35	3.94609σ		0.446 342 175 7	3.31142σ	
R250	0.134 120 025 93	0.43419σ	√	0.448 392 677 3	0.47485σ	√
R9689	0.135 421 421 80	1.56498σ	√	0.449 432 650 9	0.94622σ	√
R44497	0.135 124 568 49	1.12330σ	√	0.447 998 707 9	1.01310σ	√
R132049	0.134 205 319 96	0.30080σ	√	0.448 278 512 4	0.63130σ	√
PENTA31	0.133 793 533 03	0.93659σ	√	0.449 325 134 7	0.80024σ	√
PENTA89	0.133 268 761 04	1.76752σ	√	0.447 289 046 0	1.99025σ	√
Ziff31	0.135 303 988 59	1.39176σ	√	0.449 324 341 7	0.79657σ	√
Ziff89	0.134 159 909 60	0.36759σ	√	0.449 596 479 4	1.16867σ	√
Ziff9689	0.133 126 247 67	1.99475σ	√	0.448 318 538 9	0.57655σ	√
durxor	0.134 755 849 92	0.54729σ	√	0.448 671 385 1	0.09328σ	√
durand	0.135 199 790 76	1.24225σ	√	0.448 255 871 0	0.66290σ	√
ran_gen	0.135 356 217 59	1.46385σ	√	0.447 975 021 3	1.04562σ	√
Random	0.135 171 001 52	1.19057σ	√	0.448 359 930 6	0.51967σ	√
NWS	0.190 588 119 06	65.9349σ		0.434 860 321 0	19.6506σ	
SNWS	0.134 169 611 46	0.35408σ	√	0.448 844 043 1	0.14295σ	√
Expected	0.134 400 000 00			0.448 739 840 1		

versity Computing Center for computing support. This work was supported in part by the U.S. Department of Energy, Contract No. DE-AC02-76ER1428.

APPENDIX: DERIVATION OF $\langle(\vec{r}_{12}\cdot\vec{r}_{23})\rangle_n$ AND $\langle(\vec{r}_{12}\cdot\vec{r}_{23})\times(\vec{r}_{34}\cdot\vec{r}_{41})\rangle_n$

We derive the analytical result of Eq. (1) for a circle ($n=2$) of radius R and uniform density. The same derivation can be applied to the case of n dimensions where $n\geqslant 3$. We label three independent random points as 1, 2, and 3 in Fig. 1, and then calculate

$$\vec{r}_{12}\cdot\vec{r}_{23}=r_{12}r_{23}\cos\theta=-r_{12}r_{23}\cos\alpha, \tag{A1}$$

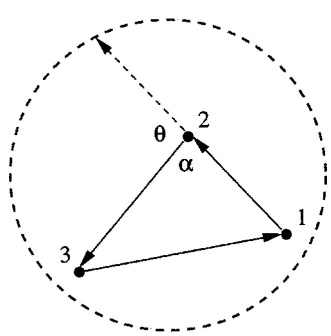

FIG. 1. Three random points configuration in a circle.

where $\alpha+\theta=\pi$. From the triangle formed by the random points, we then have

$$r_{31}^2=r_{12}^2+r_{23}^2-2r_{12}r_{23}\cos\alpha. \tag{A2}$$

Extending this two-dimensional case to the n-dimensional case, and combining Eqs. (A1) and (A2), we then evaluate

$$\langle\vec{r}_{12}\cdot\vec{r}_{23}\rangle_n=-\frac{1}{2}\langle r_{12}^2+r_{23}^2-r_{31}^2\rangle_n=-\frac{1}{2}\langle r_{12}^2\rangle_n$$

$$=-\frac{1}{2}\int_0^{2R}P_n(r)r^2dr=-\frac{n}{n+2}R^2, \tag{A3}$$

where $r\equiv r_{12}$ and we have utilized the fact that $\vec{r}_{12}$, $\vec{r}_{23}$, and $\vec{r}_{31}$ are three independent random vectors. The functions $P_n(r)$ in Eq. (A3), which can be found in Refs. [19,20,31–36], are the probability density functions for the random distance r between two random points in an n-dimensional spherical ball of radius R and uniform density.

We consider next the analytical result in Eq. (15) for a circle ($n=2$) of radius R and uniform density. A similar derivation can lead to Eqs. (16), (17), and (18), as well as to the case of n dimensions where $n\geqslant 3$. We begin by expressing four random points $\vec{r}_1$, $\vec{r}_2$, $\vec{r}_3$, and $\vec{r}_4$ in Cartesian coordinates, where $\vec{r}_i=(x_i, y_i)$. The expression in Eq. (15) can then be evaluated by writing

$$\langle(\vec{r}_{12}\cdot\vec{r}_{23})(\vec{r}_{34}\cdot\vec{r}_{41})\rangle_2=\frac{\int_{-R}^{R}dx_1\int_{-\sqrt{R^2-x_1^2}}^{\sqrt{R^2-x_1^2}}dy_1\cdots\int_{-R}^{R}dx_4\int_{-\sqrt{R^2-x_4^2}}^{\sqrt{R^2-x_4^2}}f_1\times f_2dy_4}{\int_{-R}^{R}dx_1\int_{-\sqrt{R^2-x_1^2}}^{\sqrt{R^2-x_1^2}}dy_1\cdots\int_{-R}^{R}dx_4\int_{-\sqrt{R^2-x_4^2}}^{\sqrt{R^2-x_4^2}}dy_4}=\frac{3}{8}R^4, \tag{A4}$$

where

$$f_1=(x_2-x_1)(x_3-x_2)+(y_2-y_1)(y_3-y_2),$$

$$f_2=(x_4-x_3)(x_1-x_4)+(y_4-y_3)(y_1-y_4).$$

A derivation of the general result using the probability density functions $P_n(r)$ in Eq. (A3) can be found in Ref. [29].

[1] J. Dongarra and F. Sullivan, Comput. Sci. Eng. **2**, 22 (2000).

[2] D. E. Knuth, *The Art of Computer Programming*, 3rd ed. (Addison-Wesley, Reading, MA, 1998), Vol. 2.

[3] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, 2nd ed. (Cambridge University Press, New York, 1992).

[4] G. Marsaglia, in *Computer Science and Statistics: Proceedings of the 16th Symposium on the Interface*, edited by L. Billard (Elsevier, Amsterdam, 1985), pp. 3–10.

[5] A. Garcia, *Numerical Methods for Physics*, 2nd ed. (Prentice-Hall, Upper Saddle River, NJ, 2000).

[6] N. Giordano, *Computational Physics* (Prentice-Hall, Upper Saddle River, NJ, 1997).

[7] J. Gentle, *Random Number Generation and Monte Carlo Methods*, 2nd ed. (Springer-Verlag, New York, 1998).

[8] A. M. Ferrenberg, D. P. Landau, and Y. J. Wong, Phys. Rev. Lett. **69**, 3382 (1992).

[9] I. Vattulainen, T. Ala-Nissila, and K. Kankaala, Phys. Rev. Lett. **73**, 2513 (1994).

[10] I. Vattulainen, K. Kankaala, J. Saarinen, and T. Ala-Nissila, Comput. Phys. Commun. **86**, 209 (1995).

[11] I. Vattulainen, T. Ala-Nissila, and K. Kankaala, Phys. Rev. E **52**, 3205 (1995).

[12] I. Vattulainen and T. Ala-Nissila, Comput. Phys. **9**, 500 (1995).

[13] A. N. Kolmogorov and V. A. Uspenskii, Theor. Probab. Appl. **32**, 389 (1987).

[14] A. N. Kolmogorov, Theor. Comput. Sci. **207**, 387 (1998).

[15] G. Chaitin, *Algorithmic Information Theory* (Cambridge University Press, New York, 1987).

[16] G. Chaitin, *Information, Randomness and Incompleteness: Papers on Algorithmic Information Theory*, 2nd ed. (World Scientific, Teaneck, NJ, 1990).

[17] G. Chaitin, *Exploring Randomness* (Springer, New York, 2001).

[18] S. M. Pincus, Proc. Natl. Acad. Sci. U.S.A. **88**, 2297 (1991).

[19] S. J. Tu, Ph.D. thesis, Purdue University, West Lafayette, IN, 2001.

[20] S. J. Tu and E. Fischbach, J. Phys. A **35**, 6557 (2002).

[21] G. Fishman, Math. Comput. **54**, 331 (1990).

[22] R. Ziff, Phys. Rev. Lett. **69**, 2670 (1992).

[23] R. Ziff, Comput. Phys. **12**, 385 (1998).

[24] http://www.ibm.com.

[25] http://www.vni.com.

[26] M. Metcalf and J. Reid, *Fortran 90/95 Explained*, 2nd ed. (Oxford, Midsomer Norton, Avon, 1999).

[27] B. L. Holian, O. E. Percus, T. T. Warnock, and P. A. Whitlock, Phys. Rev. E **50**, 1607 (1994).

[28] K. V. Tretiakov and K. W. Wojciechowski, Phys. Rev. E **60**, 7626 (1999).

[29] S. J. Tu and E. Fischbach (unpublished).

[30] J. Gleeson, Appl. Phys. Lett. **81**, 1949 (2002).

[31] M. G. Kendall and P. A. P. Moran, *Geometrical Probability* (Hafner, New York, 1963).

[32] L. A. Santaló, *Integral Geometry and Geometric Probability* (Addison-Wesley, Reading, MA, 1976).

[33] H. Solomon, *Geometrical Probability* (SIAM, Philadelphia, 1978).

[34] R. V. Ambartzumian, *Factorization Calculus and Geometric Probability* (Cambridge University Press, New York, 1990).

[35] D. Klain and G. C. Rota, *Introduction to Geometrical Probability* (Cambridge University Press, New York, 1997).

[36] E. Fischbach, Ann. Phys. (N.Y.) **247**, 213 (1996).