Statistical Tests of Randomness for Random Number Generators

Abhishek Ray and Tathagata Saha

13th April 2017

1 Abstract

Simulation is a core part of statistics and to simulate from any distribution it is pivotal to understand the process of simulating from Uniform distribution (U(0,1)).For this purpose Pseudo Random Number Generators (PRNGs) are being used for a considerable amount of time.They are very popular because of their high periodicity and easy implementaion in higher level languages.Now, in this project we will subject implementation of several algorithms in R which are used to generate array of random numbers to a battery of tests and report the outcome of these tests in a tabular format.

2 List of Random Number Generators:

- Wichmann-Hill
- Marsaglia-Multicarry
- Super-Duper
- Mersenne-Twister
- Knuth-TAOCP-2002
- Knuth-TAOCP
- L'Ecuyer-CMRG

3 Statistical Tests Of Randomness

- Turning Point Test
- Wald-Wolfowitz Runs Test
- Man-Kendall Rank Test

- Difference Sign Test
- Cox-Stuart Test
- Bartel's Rank Test

4 Wichmann-Hill

- This function returns a pseudo-random number uniformly distributed Between 0 and 1.
- The cycle length is 6.95 x 10^{12}, not as claimed in the original article.

Author:

- Brian Wichmann
- David Hill
- Modifications by John Burkardt.

4.1 Parameters

1. Input/output, integer S1, S2, S3, three values used as the Seed for the sequence. These values should be positive Integers between 1 and 30,000.

2. Output, real R4_RANDOM, the next value in the sequence.

4.2 Algorithm

```
Function r4_uni (s1, s2, s3)
    Integer s1
    Integer s2
    Integer s3
    Real r4_random
    s1 = mod (171 * s1, 30269)
    s2 = mod (172 * s2, 30307)
    s3 = mod (170 * s3, 30323)
    r4_random = mod (real (s1) / 30269 + real (s2) / 30307 + real (s3) / 30323,
1.0)
return r4_random
```

4.3 Tests of Randomness

4.3.1 Bartel's Rank Test

```
> library(randtests)
> n=1000
> RNGkind(kind = "Wichmann-Hill")
> set.seed(2)
> x1=runif(n)
> y11=bartels.rank.test(x1, "two.sided", pvalue="normal")
> y11

        Bartels Ratio Test

data:  x1
statistic = -1.4782, n = 1000, p-value = 0.1394
alternative hypothesis: nonrandomness
```

4.3.2 Cox-Stuart Test

```
> y12=cox.stuart.test(x1, "two.sided")
> y12

        Cox Stuart test

data:  x1
statistic = 258, n = 500, p-value = 0.5024
alternative hypothesis: non randomness
```

4.3.3 Difference-sign test of randomness

```
> y13=difference.sign.test(x1, "two.sided")
> y13

        Difference Sign Test

data:  x1
statistic = 0.82117, n = 1000, p-value = 0.4115
alternative hypothesis: nonrandomness
```

4.3.4 Mann-Kendall rank test of randomness

```
> y14=rank.test(x1, "two.sided")
> y14

        Mann-Kendall Rank Test

data:  x1
statistic = 0.067117, n = 1000, p-value = 0.9465
alternative hypothesis: trend
```

4.3.5 Wald-Wolfowitz runs test of randomness

```
> y15=runs.test(x1, "two.sided")
> y15

        Runs Test

data:  x1
statistic = -1.5187, runs = 477, n1 = 500, n2 = 500, n = 1000, p-value
= 0.1288
alternative hypothesis: nonrandomness
```

4.3.6 Turning Point test

```
> y16=turning.point.test(x1, "two.sided")
> y16

        Turning Point Test

data:  x1
statistic = -2.5023, n = 1000, p-value = 0.01234
alternative hypothesis: non randomness
```

5 Marsaglia-Multicarry

- In computer science, multiply-with-carry (MWC) is a method invented by George Marsaglia for generating sequences of random integers based on an initial set from two to many thousands of randomly chosen seed values.
- The main advantages of the MWC method are that it invokes simple computer integer arithmetic and leads to very fast generation of sequences of random numbers with immense periods, ranging from around 2^{60} to $2^{2000000}$. As with all pseudorandom number generators, the resulting sequences are functions of the supplied seed values.

5.1 Algorithm

- In its most common form, a lag-r MWC generator requires a base b, a multiplier a, and a set of $r+1$ random seed values, consisting of r residues of b, x_0,x_1, x_2, x_{r-1}, and an initial carry $c_{r-1} < a$.
- The lag-r MWC sequence is then a sequence of pairs x_n, c_n determined by $x_n = (ax_{n-r} + c_{n-1})\ mod\ b$, $c_n = \lfloor \frac{ax_{n-r}+c_{n-1}}{b} \rfloor$, $n \geq r$, and the MWC generator output is the sequence of $x's$, x_r, x_{r+1} , x_{r+2},...

5.2 Properties

- The period of a lag-r MWC generator is the order of b in the multiplicative group of numbers modulo $ab^r - 1$.
- In R, the seed is two integers (all values allowed).
- It has a period of more than 2^{60} and has passed all tests (according to Marsaglia).

5.3 Tests of Randomness

5.3.1 Bartel's Rank Test

```
> RNGkind(kind = "Marsaglia-Multicarry")
> set.seed(2)
> x2=runif(n)
> y21=bartels.rank.test(x2, "two.sided", pvalue="normal")
> y21

        Bartels Ratio Test

data:  x2
statistic = -0.62754, n = 1000, p-value = 0.5303
alternative hypothesis: nonrandomness
```

5.3.2 Cox-Stuart Test

```
> y22=cox.stuart.test(x2, "two.sided")
> y22

        Cox Stuart test

data:  x2
statistic = 251, n = 500, p-value = 0.9643
alternative hypothesis: non randomness
```

5.3.3 Difference-sign test of randomness

```
> y23=difference.sign.test(x2, "two.sided")
> y23

        Difference Sign Test

data:  x2
statistic = 1.2591, n = 1000, p-value = 0.208
alternative hypothesis: nonrandomness
```

5.3.4 Mann-Kendall rank test of randomness

```
> y24=rank.test(x2, "two.sided")
> y24

        Mann-Kendall Rank Test

data:  x2
statistic = 1.2682, n = 1000, p-value = 0.2047
alternative hypothesis: trend
```

5.3.5 Wald-Wolfowitz runs test of randomness

```
> y25=runs.test(x2, "two.sided")
> y25

        Runs Test

data:  x2
statistic = -0.18983, runs = 498, n1 = 500, n2 = 500, n = 1000, p-value
= 0.8494
alternative hypothesis: nonrandomness
```

5.3.6 Turning Point test

```
> y26=turning.point.test(x2, "two.sided")
> y26

        Turning Point Test

data:  x2
statistic = -1.3012, n = 1000, p-value = 0.1932
alternative hypothesis: non randomness
```

6 Super-Duper

- Super Duper developed by G. Marsaglia, combines the binary form of the output from the multiplicative congruential generator with multiplier $a = 69,069$ and modulus m=2^{32}. With the output of a Tausworthe generator using a left-shift of 17 and a right shift of 15.

- Tausworthe generator: Tausworthe Generator is a kind of multiplicative recursive which produces random bits. It has the following form:

  $x_{n+1} = (A_1 x_n + A_2 x_{n-1} + ... + A_k x_{n-k+1})\ mod\ 2$

  where $x_i,\ A_i \epsilon \{0, 1\} \forall i$.

6.1 Properties

- It has a period of about $4.6 * 10^{18}$ for most initial seeds. The seed is two integers (all values allowed for the first seed; the second must be odd).
- We use the implementation by Reeds et al (1982–84).
- The two seeds are the Tausworthe and congruence long integers, respectively.

6.2 Tests of Randomness

6.2.1 Bartel's Rank Test

```
> RNGkind(kind = "Super-Duper")
> set.seed(2)
> x3=runif(n)
> y31=bartels.rank.test(x3, "two.sided", pvalue="normal")
> y31

        Bartels Ratio Test

data:  x3
statistic = -0.46193, n = 1000, p-value = 0.6441
alternative hypothesis: nonrandomness
```

6.2.2 Cox-Stuart Test

```
> y32=cox.stuart.test(x3, "two.sided")
> y32

        Cox Stuart test

data:  x3
statistic = 253, n = 500, p-value = 0.8231
alternative hypothesis: non randomness
```

6.2.3 Difference-sign test of randomness

```
> y33=difference.sign.test(x3, "two.sided")
> y33

        Difference Sign Test

data:  x3
statistic = -0.38321, n = 1000, p-value = 0.7016
alternative hypothesis: nonrandomness
```

6.2.4 Mann-Kendall rank test of randomness

```
> y34=rank.test(x3, "two.sided")
> y34

        Mann-Kendall Rank Test

data:  x3
statistic = 0.87574, n = 1000, p-value = 0.3812
alternative hypothesis: trend
```

6.2.5 Wald-Wolfowitz runs test of randomness

```
> y35=runs.test(x3, "two.sided")
> y35

        Runs Test

data:  x3
statistic = -0.69605, runs = 490, n1 = 500, n2 = 500, n = 1000, p-value
= 0.4864
alternative hypothesis: nonrandomness
```

6.2.6 Turning Point test

```
> y36=turning.point.test(x3, "two.sided")
> y36

        Turning Point Test

data:  x3
statistic = -0.40036, n = 1000, p-value = 0.6889
alternative hypothesis: non randomness
```

7 Mersene-Twister

- The Mersenne Twister is a pseudorandom number generator (PRNG).
- It is by far the most widely used general-purpose PRNG.Its name derives from the fact that its period length is chosen to be a Mersenne prime.
- The Mersenne Twister was developed in 1997 by Makoto Matsumoto and Takuji Nishimura.
- It was designed specifically to rectify most of the flaws found in older PRNGs. It was the first PRNG to provide fast generation of high-quality pseudorandom integers.

- The most commonly used version of the Mersenne Twister algorithm is based on the Mersenne prime $2^{19937} - 1$.
- The standard implementation of that, MT19937, uses a 32-bit word length. Matsumoto & Nishimura (1998) work on the finite set $N_2 = \{0, 1\}$, so a variable x is represented by a vectors of ω bits (e.g. 32 bits).

7.1 Algorithm

- They use the following linear recurrence for the $n + i^{th}$ term: $x_{i+n} = x_{i+m} \bigoplus (x_i^{upp}|x_{i+1}^{low})A$,

  where $n > m$ are constant integers, x_i^{upp}(respectively x_i^{low}) means the upper (lower) $\omega - r$ (r) bits of x_i and A (a $\omega \times \omega$ matrix of N_2). $|$ is the operator of concatenation, so $x_i^{upp}|x_{i+1}^{low}$ appends the upper $\omega - r$ bits of x_i with the lower r bits of x_{i+1}.
- After a right multiplication with the matrix A , $\oplus$ adds the result with x_{i+m} bit to bit modulo two (i.e. $\oplus$ denotes the exclusive-or called xor).
- Once provided an initial seed $x_0, ..., x_{n-1}$, Mersenne Twister produces random integers in $0, ..., 2^{\omega} - 1$.

7.2 Properties

The commonly used version of Mersenne Twister, MT19937, which produces a sequence of 32-bit integers, has the following desirable properties:

1. It has a very long period of $2^{19937} - 1$. While a long period is not a guarantee of quality in a random number generator, short periods (such as the 232 common in many older software packages) can be problematic.
2. It is k-distributed to 32-bit accuracy for every $1 \leq k \leq 623$.
3. It passes numerous tests for statistical randomness, including the Diehard tests.
4. All operations used in the recurrence are bitwise operations, thus it is a very fast computation compared to modulus operations used in previous algorithms.

7.3 Tests of Randomness

7.3.1 Bartel's Rank Test

```
> RNGkind(kind = "Mersenne-Twister")
> set.seed(2)
> x4=runif(n)
> y41=bartels.rank.test(x4, "two.sided", pvalue="normal")
> y41
```

```
        Bartels Ratio Test

data:  x4
statistic = 0.50787, n = 1000, p-value = 0.6115
alternative hypothesis: nonrandomness
```

7.3.2 Cox-Stuart Test

```
> y42=cox.stuart.test(x4, "two.sided")
> y42

        Cox Stuart test

data:  x4
statistic = 251, n = 500, p-value = 0.9643
alternative hypothesis: non randomness
```

7.3.3 Difference-sign test of randomness

```
> y43=difference.sign.test(x4, "two.sided")
> y43

        Difference Sign Test

data:  x4
statistic = 0.27372, n = 1000, p-value = 0.7843
alternative hypothesis: nonrandomness
```

7.3.4 Mann-Kendall rank test of randomness

```
> y44=rank.test(x4, "two.sided")
> y44

        Mann-Kendall Rank Test

data:  x4
statistic = 0.26941, n = 1000, p-value = 0.7876
alternative hypothesis: trend
```

7.3.5 Wald-Wolfowitz runs test of randomness

```
> y45=runs.test(x4, "two.sided")
> y45

        Runs Test

data:  x4
statistic = 0.25311, runs = 505, n1 = 500, n2 = 500, n = 1000, p-value
```

```
= 0.8002
alternative hypothesis: nonrandomness
```

7.3.6 Turning Point test

```
> y46=turning.point.test(x4, "two.sided")
> y46

        Turning Point Test

data:  x4
statistic = 2.5273, n = 1000, p-value = 0.01149
alternative hypothesis: non randomness
```

8 Knuth-TAOCP-2002

- A particular case of this type of generators is when

  $X_n = (X_{n-37} + X_{n-100})\ mod\ 2^{30}$;

  which is a Fibonacci-lagged generator and the 'seed' is the set of the 100 last numbers (actually recorded as 101 numbers, the last being a cyclic shift of the buffer).

- The period is around 2^{129}. This generator has been invented by Knuth (2002) and is generally called "Knuth-TAOCP-2002" or simply "Knuth-TAOCP".

8.1 Tests of Randomness

8.1.1 Bartel's Rank Test

```
> RNGkind(kind = "Knuth-TAOCP-2002")
> set.seed(2)
> x5=runif(n)
> y51=bartels.rank.test(x5, "two.sided", pvalue="normal")
> y51

        Bartels Ratio Test

data:  x5
statistic = -1.7476, n = 1000, p-value = 0.08054
alternative hypothesis: nonrandomness
```

8.1.2 Cox-Stuart Test

```
> y52=cox.stuart.test(x5, "two.sided")
> y52
```

```
        Cox Stuart test

data:  x5
statistic = 262, n = 500, p-value = 0.3037
alternative hypothesis: non randomness
```

8.1.3 Difference-sign test of randomness

```
> y53=difference.sign.test(x5, "two.sided")
> y53

        Difference Sign Test

data:  x5
statistic = -0.93066, n = 1000, p-value = 0.352
alternative hypothesis: nonrandomness
```

8.1.4 Mann-Kendall rank test of randomness

```
> y54=rank.test(x5, "two.sided")
> y54

        Mann-Kendall Rank Test

data:  x5
statistic = 0.57106, n = 1000, p-value = 0.568
alternative hypothesis: trend
```

8.1.5 Wald-Wolfowitz runs test of randomness

```
> y55=runs.test(x5, "two.sided")
> y55

        Runs Test

data:  x5
statistic = -1.3921, runs = 479, n1 = 500, n2 = 500, n = 1000, p-value
= 0.1639
alternative hypothesis: nonrandomness
```

8.1.6 Turning Point test

```
> y56=turning.point.test(x5, "two.sided")
> y56

        Turning Point Test

data:  x5
```

```
statistic = 0.50045, n = 1000, p-value = 0.6168
alternative hypothesis: non randomness
```

9 Knuth-TAOCP

- An earlier version from Knuth (1997). The 2002 version was not backwards compatible with the earlier version: the initialization of the GFSR from the seed was altered. R did not allow you to choose consecutive seeds, the reported 'weakness', and already scrambled the seeds.
- Initialization of this generator is done in interpreted R code and so takes a short but noticeable time.

9.1 Tests of Randomness

9.1.1 Bartel's Rank Test

```
> RNGkind(kind = "Knuth-TAOCP")
> set.seed(2)
> x6=runif(n)
> y61=bartels.rank.test(x6, "two.sided", pvalue="normal")
> y61

        Bartels Ratio Test

data:  x6
statistic = -1.0752, n = 1000, p-value = 0.2823
alternative hypothesis: nonrandomness
```

9.1.2 Cox-Stuart Test

```
> y62=cox.stuart.test(x6, "two.sided")
> y62

        Cox Stuart test

data:  x6
statistic = 251, n = 500, p-value = 0.9643
alternative hypothesis: non randomness
```

9.1.3 Difference-sign test of randomness

```
> y63=difference.sign.test(x6, "two.sided")
> y63

        Difference Sign Test
```

```
data:  x6
statistic = -0.38321, n = 1000, p-value = 0.7016
alternative hypothesis: nonrandomness
```

9.1.4 Mann-Kendall rank test of randomness

```
> y64=rank.test(x6, "two.sided")
> y64

        Mann-Kendall Rank Test

data:  x6
statistic = -1.2817, n = 1000, p-value = 0.2
alternative hypothesis: trend
```

9.1.5 Wald-Wolfowitz runs test of randomness

```
> y65=runs.test(x6, "two.sided")
> y65

        Runs Test

data:  x6
statistic = -0.69605, runs = 490, n1 = 500, n2 = 500, n = 1000, p-value
= 0.4864
alternative hypothesis: nonrandomness
```

9.1.6 Turning Point test

```
> y66=turning.point.test(x6, "two.sided")
> y66

        Turning Point Test

data:  x6
statistic = 1.1761, n = 1000, p-value = 0.2396
alternative hypothesis: non randomness
```

10 L'Ecuyer-CMRG

- This is a 'combined multiple-recursive generator' from L'Ecuyer (1999), each element of which is a feedback multiplicative generator with three integer elements: thus the seed is a (signed) integer vector of length 6. It is given by:
- $z_n = (x_n - y_n) \bmod m_1$ where the two underlying generators x_n and y_n are,

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + a_3 x_{n-3}) \bmod m_1$$
$$y_n = (b_1 y_{n-1} + b_2 y_{n-2} + b_3 y_{n-3}) \bmod m_2$$

with coefficients a_1 = 0, a_2 = 63308, a_3 = -183326, b_1 = 86098, b_2 = 0, b_3 = -539608, and moduli m_1 = 2ˆ31 - 1 = 2147483647 and m_2 = 2145483479.

10.1 Properties

- The period of this generator is $lcm(m_1^3 - 1, m_2^3 - 1)$, which is approximately 2^{185} (about 10^{56}).

- This is not particularly interesting of itself, but provides the basis for the multiple streams used in package parallel.

10.2 Tests of Randomness

10.2.1 Bartel's Rank Test

```
> RNGkind(kind = "L'Ecuyer-CMRG")
> set.seed(2)
> x7=runif(n)
> y71=bartels.rank.test(x7, "two.sided", pvalue="normal")
> y71

        Bartels Ratio Test

data:  x7
statistic = 0.71466, n = 1000, p-value = 0.4748
alternative hypothesis: nonrandomness
```

10.2.2 Cox-Stuart Test

```
> y72=cox.stuart.test(x7, "two.sided")
> y72

        Cox Stuart test

data:  x7
statistic = 234, n = 500, p-value = 0.1656
alternative hypothesis: non randomness
```

10.2.3 Difference-sign test of randomness

```
> y73=difference.sign.test(x7, "two.sided")
> y73
```

```
        Difference Sign Test

data:  x7
statistic = -0.27372, n = 1000, p-value = 0.7843
alternative hypothesis: nonrandomness
```

10.2.4 Mann-Kendall rank test of randomness

```
> y74=rank.test(x7, "two.sided")
> y74

        Mann-Kendall Rank Test

data:  x7
statistic = -0.55438, n = 1000, p-value = 0.5793
alternative hypothesis: trend
```

10.2.5 Wald-Wolfowitz runs test of randomness

```
> y75=runs.test(x7, "two.sided")
> y75

        Runs Test

data:  x7
statistic = 1.5819, runs = 526, n1 = 500, n2 = 500, n = 1000, p-value =
0.1137
alternative hypothesis: nonrandomness
```

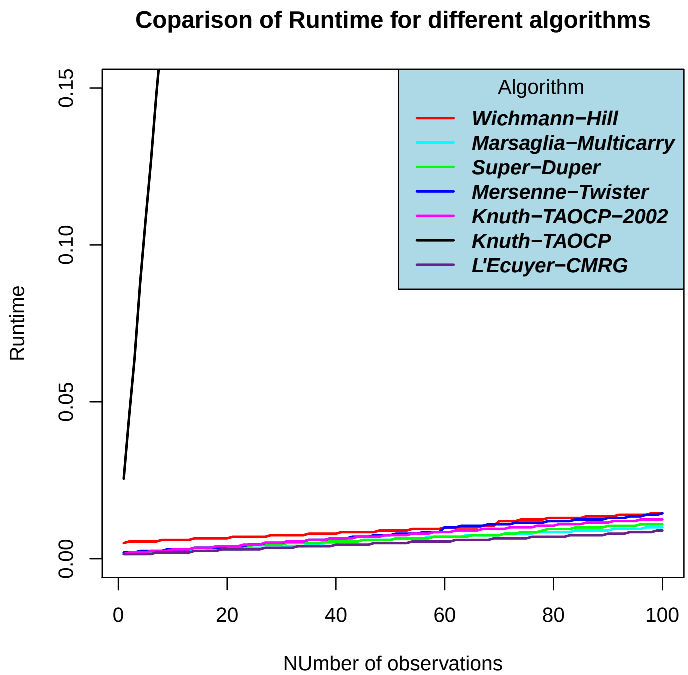
10.2.6 Turning Point test

```
> y76=turning.point.test(x7, "two.sided")
> y76

        Turning Point Test

data:  x7
statistic = 1.4763, n = 1000, p-value = 0.1399
alternative hypothesis: non randomness
```

11 Comparison Of Run-times



12 Summary of Tests for different Algorithms

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	0.14	0.96	0.70	0.79	0.16	0.24
2	Marsaglia-Multicarry	0.50	0.21	0.38	0.80	0.62	0.47
3	Super-Duper	0.41	0.20	0.49	0.01	0.28	0.17
4	Mersenne-Twister	0.95	0.85	0.69	0.08	0.96	0.78
5	Knuth-TAOCP-2002	0.13	0.19	0.61	0.30	0.70	0.58
6	Knuth-TAOCP	0.01	0.64	0.96	0.35	0.20	0.11
7	L'Ecuyer-CMRG	0.53	0.82	0.78	0.57	0.49	0.14

Table 1: Table of p-values for different algorithms for different tests for n=1000

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	PASS	PASS	PASS	PASS	PASS	PASS
2	Marsaglia-Multicarry	PASS	PASS	PASS	PASS	PASS	PASS
3	Super-Duper	PASS	PASS	PASS	FAIL	PASS	PASS
4	Mersenne-Twister	PASS	PASS	PASS	FAIL	PASS	PASS
5	Knuth-TAOCP-2002	PASS	PASS	PASS	PASS	PASS	PASS
6	Knuth-TAOCP	FAIL	PASS	PASS	PASS	PASS	PASS
7	L'Ecuyer-CMRG	PASS	PASS	PASS	PASS	PASS	PASS

Table 2: Table showing result of different tests for different algorithms at 0.1 level of significance for n=1000

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	0.04	0.73	0.56	0.95	0.59	0.60
2	Marsaglia-Multicarry	0.51	0.33	0.42	0.96	0.56	0.70
3	Super-Duper	0.67	0.36	0.45	0.09	0.47	0.97
4	Mersenne-Twister	0.20	0.37	0.40	0.11	0.87	0.56
5	Knuth-TAOCP-2002	0.04	0.79	0.73	0.97	0.05	0.72
6	Knuth-TAOCP	0.01	0.31	0.39	0.10	0.63	0.45
7	L'Ecuyer-CMRG	0.51	0.97	0.23	0.55	0.72	0.83

Table 3: Table of p-values for different algorithms for different tests for n=2000

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	FAIL	PASS	PASS	PASS	PASS	PASS
2	Marsaglia-Multicarry	PASS	PASS	PASS	PASS	PASS	PASS
3	Super-Duper	PASS	PASS	PASS	FAIL	PASS	PASS
4	Mersenne-Twister	PASS	PASS	PASS	PASS	PASS	PASS
5	Knuth-TAOCP-2002	FAIL	PASS	PASS	PASS	FAIL	PASS
6	Knuth-TAOCP	FAIL	PASS	PASS	FAIL	PASS	PASS
7	L'Ecuyer-CMRG	PASS	PASS	PASS	PASS	PASS	PASS

Table 4: Table showing result of different tests for different algorithms at 0.1 level of significance for n=2000

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	0.15	0.83	0.54	0.73	0.61	0.27
2	Marsaglia-Multicarry	0.79	0.61	1.00	0.41	0.84	0.87
3	Super-Duper	0.98	0.98	0.57	0.61	0.91	0.83
4	Mersenne-Twister	0.78	0.18	0.40	0.18	0.27	0.03
5	Knuth-TAOCP-2002	0.10	0.05	0.81	0.65	0.61	0.97
6	Knuth-TAOCP	0.02	0.65	0.83	0.06	0.35	0.12
7	L'Ecuyer-CMRG	0.21	0.67	0.19	0.83	0.87	0.23

Table 5: Table of p-values for different algorithms for different tests for n=2000

		Bartel's Rank Test	Cox-Stuart Test	Difference Sign Test	Man-Kendall Rank Test	Wald-Wolfowitz Runs Test	Turning Point Test
1	Wichmann-Hill	PASS	PASS	PASS	PASS	PASS	PASS
2	Marsaglia-Multicarry	PASS	PASS	PASS	PASS	PASS	PASS
3	Super-Duper	PASS	PASS	PASS	PASS	PASS	PASS
4	Mersenne-Twister	PASS	PASS	PASS	PASS	PASS	FAIL
5	Knuth-TAOCP-2002	PASS	FAIL	PASS	PASS	PASS	PASS
6	Knuth-TAOCP	FAIL	PASS	PASS	FAIL	PASS	PASS
7	L'Ecuyer-CMRG	PASS	PASS	PASS	PASS	PASS	PASS

Table 6: Table showing result of different tests for different algorithms at 0.1 level of significance for n=2000

13 Appendix

13.1 Statistical Tests Of Randomness

- Turning Point Test
- Wald-Wolfowitz Runs Test
- Man-Kendall Rank Test
- Difference Sign Test
- Cox-Stuart Test
- Bartel's Rank Test

13.2 Turning Point Test

- In statistical hypothesis testing, a turning point test is a statistical test of the independence of a series of random variables.
- Maurice Kendall and Alan Stuart describe the test as reasonable for a test against cyclicity but poor as a test against trend.
- We say i is a turning point if the vector $X_1, X_2, ..., X_i, ..., X_n$ is not monotonic at index i. The number of turning points is the number of maxima and minima in the series.

Let T be the number of turning points then for large n,

$$Z = \frac{T - \frac{2n-4}{3}}{\sqrt{\frac{16n-29}{90}}} \sim N(0,1) \tag{1}$$

R function:-
turning.point.test(x,alternative)

13.3 Wald-Wolfowitz Runs Test

- It is the simple runs test.
- Under the null hypothesis, the number of runs in a sequence of N elements is a random variable whose conditional distribution given the observation of N_+ positive values and N_- negative values ($N = N_+ + N_-$) is approximately normal with *mean* $\mu = \frac{2N_+N_-}{N} + 1$ and variance $\sigma^2 = \frac{2N_+N_-(2N_+N_- - N)}{N^2(N-1)} = \frac{(\mu-1)(\mu-2)}{N-1}$.
- R function:- runs.test(x,alternative,threshold,pvalue,plot).

13.4 Mann-Kendall Rank Test

- Originally, Kendall's tau statistic is used as a measure of association in a bivariate population (X, Y). If we treat the time, $\{1, 2, ...n\}$, of an observed sequence as X and the set of time-ordered observations, $\{Y_1, Y_2, .., Y_n\}$, as Y; then the association between X and Y can be considered as an indication of trend.
- The test statistic is $T = \sum_{i=2}^{n}\sum_{j=1}^{i-1} sign(Y_i - Y_j)$ which converges to a normal random variable under the null hypothesis of randomness: $T \sim N(0, \sigma_3^2)$; where $\sigma_3^2 = \frac{n(n-1)(2n+5)}{18}$.
- R function:- rank.test(x,alternative).

13.5 Difference Sign Test

- The sequence is $y_1, y_2, ..., y_n$. For this test we count the number S of values of i such that $y_i > y_{i-1}$, $i = 2, ..., n$ or equivalently the number of times the differenced series $(y_i - y_{i-1})$ is positive.
- For an iid sequence it is clear that $\mu_S = ES = 1/2(n-1)$. It can also be shown under the same assumption that $\sigma_S^2 = Var(S) = \frac{(n+1)}{12}$.
- For large n, $S \sim N(\mu_S, \sigma_S^2)$. A large positive(or negative) value of $(S - \mu_S)$ indicates the presence of increasing(or decreasing) trend in the data. We therefore reject the assumption of trend in the data if $\frac{|S-\mu_S|}{\sigma_S} > \Phi_{1-\alpha/2}$.
- R function:- difference.sign.test(x,alternative)

13.6 Cox- Stuart Test

- Given a set of ordered observations $X_1, X_2, ..., X_n$, let
c = n/2 if n even
= (n+1)/2 if n is odd

  Then pair the data as $X_1, X_{1+c}, X_2, X_{2+c}, ..., X_{n-c}, X_n$. The Cox-Stuart test is then simply a sign test applied to these paired data.

- R function:- cox.stuart.test(x; alternative)

13.7 Bartel's Rank Test

- This is the rank version of von Neumann's Ratio Test for Randomness.

- The test statistic is $RVN = \frac{\sum_{i=1}^{n-1}(R_i - R_{i+1})^2}{\sum_{i=1}^{n}(R_i - {(n+1)}/{2})^2}$ where $R_i = Rank(X_i)$. ${RVN-2}/{\sigma}$ is asymptotically standard normal , with $\sigma^2 = \frac{4(n-2)(5n^2-2n-9)}{5n(n+1)(n-1)^2}$.

- R function:- bartels.rank.test(x,alternative,pvalue=”normal”)

References

[1] R Core Team, R Foundation for Statistical Computing, *R: A Language and Environment for Statistical Computing*, 2016. Available at `https://www.R-project.org/`.

[2] Frederico Caeiro and Ayana Mateus, *randtests: Testing randomness in R* ; 2014. Available at `https://CRAN.R-project.org/package=randtests`.

[3] Christophe Dutang and Diethelm Wuertz, *A note on random number generation*; September 2009.

[4] Makoto Matsumoto and Takuji Nishimura, *Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator*; Keio University/ Max-Planck-Institut fur Mathematik