Quality of random numbers – Quality of random number generators

Thomas Risse
Institute of Informatics & Automation, IIA
Faculty E & I, Hochschule Bremen
University of Applied Sciences
risse@hs-bremen.de

SIP31, 28.-29.10.2013
Hochschule Bremen, University of Applied Sciences
Introduction

Deterministic RNGs

MATLABs random number generators

SAGEs random number generators

Criteria for pseudo random number generators

Quality Criteria for PRNGs

Comparison and Results

Resumé

The generation of random numbers is too important to be left to chance – Robert R. Coveyou
Motivation
by Some Recent Examples

ConsensusSecurityVulnerabilityAlert@sans.org 13/08/15
Flaw in Android random number generator leaves Bitcoin wallets open to theft. Users of such apps are encouraged to migrate away from insecure wallets through any feasible mechanism as soon as possible.

sans.org 13/08/31  [5] 'Security Analysis of Pseudo-Random Number Generators with Input /dev/random is not Robust.'

c’t 22/2013 S.44, wired  RSA Tells Its Developer Customers: Stop Using NSA-Linked Random Bit Generator
http://csrc.nist.gov/publications/PubsDrafts.html#SP-800-90-ARev1BandC
Introduction

Why random numbers, RNs?
Why random number generators, RNGs?
z.B. block ciphers, RSA-moduli, keystream generation, zero knowledge authentication, [10], [4], [24], PQC [22] etc per [8]

- Deterministic/Pseudo-RNGs, DRNG/PRNG
- Physical/True RNGs, TRNG, PTRNG

Quality of random numbers = Quality of Generators
Criteria specified by diverse bodies, e.g.

- NIST [24] \(\leadsto\) FIPS

disclaimer

Here, we in more detail examine the generators of uniformly distributed random numbers of MATLAB and SAGE only!
Deterministic RNGs

Typical is some initialisation (seed) and computation of the next random number based on the last random number(s).

- John von Neumann’s middle-square method from 1946: take any number (seed), square it, remove the middle digits of the resulting number as the ‘random number’, then use that number as the seed for the next iteration.

- Linear congruential generators [12]: choose seed $x_0$ and compute $x_{k+1} = a x_k + c \mod m$ for $k = 0, 1, \ldots$ and suitable parameters $a$, $c$ and $m$

- Linear Feedback Shift Registers, LFSRs: choose some initial state and run hardware, for example

$$p(x) = x^{16} + x^{14} + x^{13} + x^{11} + 1 \in GF_2$$
Deterministic RNGs’

- Mersenne\(^1\) twister algorithm [16]:
  choose \( N = 624 \) seeds, \( Y_1, \ldots, Y_N \). Compute \( Y_i \) for \( i > N \)
  by \( h := Y_{i-N} - Y_{i-N} \mod 2^{31} + Y_{i-N+1} \mod 2^{31} \)
  \( Y_i := Y_{i-227} \oplus \lfloor h/2 \rfloor \oplus ((h \mod 2) \cdot 9908b0df_{hex}) \)
  Postprocessing then guarantees uniform distribution:
  \( x := Y_i \oplus \lfloor Y_i/2^{11} \rfloor; \ y := x \oplus ((x \cdot 2^7) \land 9d2c5680_{hex}); \)
  \( z := y \oplus ((y \cdot 2^{15}) \land \text{efc60000}_{hex}); \ Z_i := z \oplus \lfloor z/2^{18} \rfloor \)
  Period is \( 2^{19937} - 1 \approx 4,3 \cdot 10^{6001} \). Improvements see [20]

- Blum-Blum-Shub generator [1]:
  choose \( n = pq \) for suitable primes \( p \) and \( q \), seed \( s \) with
  \( \gcd(s, n) = 1 \), compute \( s_0 = s^2 \mod n \) and for \( i = 0, 1, \ldots \)
  \( s_{i+1} = s_i^2 \mod n \)
  If desired, get random bits by \( b_i = s_i \mod 2 \) or similar.

---

\(^1\)M. Mersenne (1588-1648) [www-history.mcs.st-andrews.ac.uk/Biographies/Mersenne.html](http://www-history.mcs.st-andrews.ac.uk/Biographies/Mersenne.html) was the first to examine integers of the form \( 2^n - 1 \). Some are prime! see [GIMPS](https://www.mersenne.org/).
MATLABs PRNGs

< 1995  mcg16807, Park & Miller’s PNRG [21]
according to [19]:
uses linear congruential generator with
\( a = 7^5 = 16807, \ c = 0, \)
\( m = 2^{31} - 1 = 2147483647 \) when divided by \( m \)
produces random numbers \( \in [0, 1) \).

1995–2008  Marsaglia’s PNRG [13][14] according to [19]:
initialize internal state or cache of 32 floats
\( z_0, \ldots, z_{31} \in (0, 1) \), an index \( N_o \ni i \in [0, 31] \),
a random integer \( j \) and a borrow ‘flag’ \( b \).

\[
Z_i \mod 32 := Z_{(i+20)} \mod 32 - Z_{(i+5)} \mod 32 - b
\]
then \( b := \begin{cases} 
0 & \text{if } z_i \geq 0 \\
2^{-53} & \text{otherwise}
\end{cases} \)
\( z_i := z_i + 1 \) if \( z_i < 0 \).
To be uniformly distributed, the floating-point
fraction of each \( z_i \) is XORed with \( j \), a separately
generated 53bit integer random number.
MATLABs PRNGs’

since 2007  Mersenne twister is the default PRNG according to [19] cp rng(’type’) with type

’twister’       Mersenne Twister
’combRecursive’ Combined Multiple Recursive
’multFibonacci’ Multiplicative Lagged Fibonacci
’v5uniform’    Legacy MATLAB 5.0 uniform generator
’v5normal’     Legacy MATLAB 5.0 normal generator
’v4’           Legacy MATLAB 4.0 generator

• Combined Multiple Recursive [6]: for example compute
  \[x_{1,n} = (1403580 \cdot x_{1,n-2} - 810728 \cdot x_{1,n-3}) \mod m_1\]
  and
  \[x_{2,n} = (527612 \cdot x_{2,n-1} - 1370589 \cdot x_{2,n-3}) \mod m_2\]
  where \(m_1 = 2^{32} - 209\) and \(m_2 = 2^{32} - 22853\). Combine \(x_{1,n}\)
  and \(x_{2,n}\) to \(z_n = (x_{1,n} - x_{2,n}) \mod m_1\) and output
  \[u_n = \frac{z_n}{m_1 + 1}\] if \(z_n > 0\) and \(u_n = \frac{m_1}{m_1 + 1}\) if \(z_n = 0\).

• Multiplicative Lagged Fibonacci, cp wikipedia:
  \[S_n \equiv S_{n-j} \ast S_{n-k} \mod m\] with \(0 < j < k\) where \(m = 2^{32}\)
  or \(m = 2^{64}\), and \(\ast\) denotes multiplication or XOR.
SAGEs PRNGs

- SAGE comprises several CASs. Thus, not surprisingly SAGE offers a variety of built-in random number generators.

- SAGE is object oriented. Hence, quite a lot of built-in objects offer `random_element` methods, e.g.
  - **ZZ** random, uniformly distributed integers in a given interval
  - **QQ** random, uniformly distributed rational numbers with numerators and denominators in a given range
  - **RR** random, uniformly distributed floating point numbers with specified 'precision' (number of mantissa bits) in a given interval

which can be generalized to random elements of polynomial rings, to random elements of quotient polynomial rings etc, of permutation groups
SAGEs vs MATLABs PRNGs’

But in SAGE there are also – to name a few as examples

- random matrices over given fields, i.e. any finite field as well as $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$
- random elements of partitions
- random elements of permutation groups
- random elements of (certain type of) graphs
- random elements of paths in a graph

In contrast, MATLAB (version 7.13) in addition only provides

- `randi` integer random numbers
- `sprnd` sparse random matrices
- `randperm` random permutations of the integers 1, 2, ... , $n$
Criteria for pseudo RNGs

BSI [2] offers (critique see [23])

1. Monobit Test
2. Poker Test
3. Run Tests
4. Longrun Test
5. Autocorrelation Tests
6. Uniform Distribution Test
7. Homogeneity Tests
8. Entropy Test

In addition, NIST [24] offers

- binary matrix rank
- DFT
- template matching
- linear complexity
- random walks
Quality Criteria for PRNGs

Monobit Test: FIPS, HAC [18], NIST [24], BSI [9],[10] pp44

Let \( n = 20000 \), \( T_1 = \sum_{i=1}^{n} b_i \). Bit sequence \( (b_i)_{i=1}^{n} \) passes the Monobit Test if \( 9654 < T_1 < 10346 \).

\( T_1 \) is – independence assumed! – binomially distributed:
\[ E(T_1) = np \text{ and } \text{Var}(T_1) = np(1-p) \text{ for } p = P(b = 1). \]

For \( p = \frac{1}{2} \) SAGE [25] gives with any precision
\[ P(9654 < T_1 < 10346) \approx 0.999999078354697. \]

\( T_1 \) is approximately \( N(np, npq) \)-distributed. SAGE
\[ P(9654 < T_1 < 10346) \approx P(|U| < 4.89317892581091) \approx 0.999999503899380 \text{ for } N(0, 1) \text{-distributed } U, \]
where \( \Phi(u) = \frac{1}{2} \left( 1 + \text{erf}\left( \frac{u}{\sqrt{2}} \right) \right) \) is the distribution function of \( U \).

\Rightarrow \text{BSI-error probability } \leq 1 - 0.999999 = 10^{-6} \]

NB [18],[9],[10] examine (also) the approximately \( \chi^2 \)
distributed test statistic \( T_1' = \frac{1}{n}(T_1 - (n - T_1))^2 \) with \( df = 1 \).
Quality criteria for PRNGs


Let \( n = 20000 \). Every 4 bits give a nibble.

Let \( h_i := |\{ j : 8b_{4j-3} + 4b_{4j-2} + 2b_{4j-1} + b_{4j} = i \}| \)

and \( T_2 = \frac{16}{5000} \sum_{i=0}^{15} h_i^2 - 5000 \). Bit sequence \((b_i)_{i=1}^{n}\) passes the Poker Test, if \( 1.03 < T_2 < 57.4 \).

\[
T_2 = \frac{16}{n_4} \sum_{i=0}^{15} h_i^2 - n_4 = \sum_{i=0}^{15} \frac{(h_i - n_4/16)^2}{n_4/16} \geq 0 \text{ with } n_4 = \frac{n}{4}
\]
is \( \chi^2 \)-distributed with \( df = 15 \). NB BSI \( P(\chi^2 \geq 56.49) = 10^{-6} \)

SAGE: \( P(1.03 < T_2 < 57.4) \approx F_{15}(57.4) - F_{15}(1.03) \approx 0.999998985794408 \approx 1 - 10^{-6} \).

NB: lopsided!

We have \( F_{df}(x) = P(df, x) \) and for odd \( df \)

\[
P(df, x) = \text{erf}(\sqrt{x/2}) - e^{-x/2} \sum_{k=0}^{[df/2]-1} \frac{1}{\Gamma(k+3/2)} \left( \frac{x}{2} \right)^{k+1/2}
\]

SAGE: \( P(1.03 < T_2 < 57.4) \approx F_{15}(57.4) - F_{15}(1.03) \approx 0.999998985794408 \approx 1 - 10^{-6} \).

NB: lopsided!
Quality Criteria for PRNGs


Let $n = 20000$ and $k_\ell$ the number of runs of length $\ell$. The bit sequence $(b_i)_{i=1}^n$ passes the Run Tests, if

$k_1 \in [2267, 2733], k_2 \in [1079, 1421], k_3 \in [502, 748], k_4 \in [233, 402], k_5 \in [90, 223]$ and $k \geq 6 \in [90, 2^{2/3}]$

0-runs or 1-runs of length $\ell$ occur with $p_{run} = \frac{1}{2^{\ell+2}}$ in any of the $n - \ell - 1$ places and in each of the two boundaries with $p'_{run} = 2p_{run}$. \( \Rightarrow E(K_\ell) = \frac{n-\ell-1+2+2}{2^{\ell+2}} = \frac{n-\ell+3}{2^{\ell+2}} \).

\[ T_{3\ell} = \sum_{b=0}^{1} \sum_{i=1}^{\ell} \left( \frac{k_i^{(b)} - E(K_i)}{E(K_i)} \right)^2, \quad k_i = k_i^{(0)} + k_i^{(1)} \approx \chi^2 \text{-distributed}, \]

\[ df = 2\ell - 1 = \text{#observ} - \text{#params}. \]

NB [18] = $2\ell - 2$, NB [17] = $2\ell$

SAGE's find_root gives: $P(T_{31} < 10^{-12}) \approx 0.5 \cdot 10^{-6}$ and $P(T_{31} > 25.263820726226815) \approx 0.5 \cdot 10^{-6}$, (BSI onesided $P(T_{31} < 23.9281269768) = 1 - 10^{-6}$) with $E(K_1) = \frac{n+2}{8} = 2500.25$ implying NB $k_1 \in [2322, 2677]$ ? resp. NB $k_1 \in [2327, 2673]$ ? Zählweise?
Introduction

DRNGs

MATLAB

SAGE

PRNG-criteria

Results

Resumé

Quality Criteria for PRNGs


Let $n = 20000$. The bit sequence $(b_i)_{i=1}^n$ passes the Longrun Test, if $k_\ell = 0$ for all $\ell \geq 34$.

\[ P(k_\ell = 0) = \frac{F_{n+2}(\ell)}{2^n} \]

with the Fibonacci $\ell$-step numbers [27]

\[ F_k^{(\ell)} = \sum_{i=1}^{\ell} F_{k-i}^{(\ell)} \quad \text{with} \quad F_k^{(\ell)} = 0 \quad \text{for} \quad k \leq 0 \quad \text{and} \quad F_1^{(\ell)} = F_2^{(\ell)} = 1. \]

**SAGE**: $P(k_{34} = 0) \approx 0.999999418854882 \approx 1 - 10^{-6}$

and $P(k_\ell = 0) \uparrow$ for $\ell \uparrow$

By the way, roughly estimating SAGE gives

\[ P(k_\ell > 0 \text{ for at least one } \ell \geq 34) \approx (n - 34)2^{-34} \approx 1.16217415779829 \cdot 10^{-6} \]
Quality Criteria for PRNGs

**Autocorrelation Test:** HAC [18], NIST [24], BSI [9],[10] pp49

Let \( n = 20000 \) and \( T_{5\tau} = \sum_{j=1}^{n/4} b_j \oplus b_{j+\tau} \) for \( \tau \in \{1, 2, \ldots, \frac{n}{4}\} \). The bit sequence \( (b_i)_{i=1}^{n} \) passes the Autocorrelation Test, if \( |T_{5\tau} - \frac{n}{8}| < 174 \) for all \( \tau \).

**NB:** only the first half of the \( (b_i) \) is relevant!?!?

\( T_{5\tau} \) is approximately \( N\left(\frac{n}{4}, \frac{n}{4}, \frac{1}{2}, \frac{1}{2}\right) \)-distributed. With SAGE \( u = \frac{174}{\sqrt{n/16}} = \frac{1.7444}{\sqrt{2}} \approx 4.92146319705837 \) we get

\[
P\left(|T_{5\tau} - \frac{n}{8}| < 174\right) = P\left(|U| < u\right) = 2\Phi(u) - 1 \approx 0.999999914100 \approx 1 - 10^{-6} \text{ for } N(0, 1) \text{-distributed } U.
\]
Quality Criteria for PRNGs

Uniform distribution Test: HAC 2bit, NIST \textsuperscript{template}\textsuperscript{matching}, BSI [9],[10] pp50

Generate \( w_j \in \{0, 1\}^k \) from \( (b_i)_{i=1}^{nk} \); \( T_{6x} := \frac{|\{j: w_j = x\}|}{n} \) is the relative frequency of \( x \). The bit sequence \( (b_i)_{i=1}^{nk} \) passes the Uniform distribution Test for parameters \( k, n \) and \( \alpha \), if \( |T_{6x} - 2^{-k}| < \alpha \) for all \( x \in \{0, 1\}^k \).

Uniform distribution tests generalize Monobit Tests!

BSI [9],[10] p55 Test Procedure B: \( T_6 \) with NB \( k = 1 \), \( n = 10^5 \) and \( \alpha = 0.025 \). Explicitly p51: \( (b_i)_{i=1}^{n} \) passes if \( |T_{6o} - \frac{1}{2}| < \alpha \) ? \( T_6 \) ? NB only for 'PTRNG'.

let \( b \in \{0, 1\} \) and \( h_b = \# b \) in \( (b_i)_{i=1}^{n} \). independent!

\( \chi^2 \)-adaption test: \( T_6 = \sum_{b=0}^{1} \frac{(h_b - n/2)^2}{n/2} \) is \( \chi^2 \)-distributed with \( df = 1 \). BSI condition \( |\frac{h_b}{n} - \frac{1}{2}| \leq \alpha \) for \( b \in \{0, 1\} \Rightarrow \left( h_b - \frac{n}{2} \right)^2 \leq \alpha^2 n^2 \Rightarrow T_6 \leq 250 \). SAGE \( P(T_6 \leq 250) = 1 \), NB while SAGE \( P(T_6 \leq \approx 23.9) \approx 1 - 10^{-6} \) ?
Quality Criteria for PRNGs

Multinomial/Homogeneity Test: BSI [9],[10] pp51

Generate $w_{i,j} \in \{0, 1, ..., s-1\}$ for $i = 1, ..., h$ of $(b_k)_k$, i.e. $h$ independent repetitions of the $j$-th experiment. Let $f_i(t) = |\{j : w_{ij} = t\}|$ and $p_t = \frac{1}{hn} \sum_{i=1}^{h} f_i(t)$. The bit sequence $(b_i)_i$ passes the Multinomial Test for $h$, $s$, $n$ and $\alpha$ if $T_7 \leq \chi^2(\alpha, (h-1)(s-1))$ where

$$T_7 = \sum_{i=1}^{h} \sum_{t=0}^{s-1} \frac{(f_i(t)-np_t)^2}{np_t}$$

No longer up to date: BSI-example for $h = s = 2$, i.e. $i = 1, 2$ and template $t = 0, 1$ – adapted from [7], Test 76.

Two samples with $n$ elements each $w_{i,1}, w_{i,2}, \ldots, w_{i,n}$ for $i = 1, 2$ of $n$ bits each. Determine

absolute frequency $f_i(t) = |\{j : w_{ij} = t\}|$ of $t$ in sample

relative frequency $p_t = \frac{f_1(t)+f_2(t)}{2n}$ von $t$ in both samples

$T_7 = \sum_{i=1}^{h} \sum_{t=0}^{s-1} \frac{(f_i(t)-np_t)^2}{np_t}$ is $\chi^2$-distributed, $df = (h-1)(s-1) = 1$ and according to BSI p37, Tabelle [9],[10] p46 $P(T_7 \geq 15.13) = \alpha = 0.0001$ ?
Quality Criteria for PRNGs

BSI [9],[10] pp55 Test Procedure B demands (typos) three tests – with NB three different representations \( \uparrow \) pp51, p56, pp58 NB \( \uparrow \) only for ’PTRNG’ in spite of pp58 (typos)

step 2 bits \( \uparrow \)?

Extract \( TF_r = \{(b_{2j+1}, b_{2j+2}) : b_{2j+1} = r\} \) with \(|TF_0| = |TF_1| = n_1 = 10^5\) from sequence. Determine \( v_r(i) = \frac{|\{j : (b_{2j+1}, i) \in TF_r\}|}{n_1} \).

Sequence passes T7 if \(|v_0(1) + v_1(0) - 1| < \alpha_1 = 0.02? v_0(0)\? v_1(1)\? \)

step 3 bits \( \uparrow \) supposedly with \( h = 2? \) or \( h = 4? , s = 2 \)

Extract \( TF_{rs} = \{(b_{3j+1}, \ldots, b_{3j+3}) : (b_{3j+1}, b_{3j+2}) = (rs)\} \) with

\(|TF_{oo}| = |TF_{o1}| = |TF_{1o}| = |TF_{11}| = n_2 = 10^5\) from sequence.

Determine \( v_{rs}(i) := \frac{|\{j : (b_{3j+1}, \ldots, b_{3j+3}) = (rsi)\}|}{n_2} \).

’for each \( s \in \{0, 1\}\) compare \( v_{0s} \) and \( v_{1s} \) with \( \uparrow \) at \( \alpha_2 = 0.0001’ \)

step 4 bits \( \uparrow \) supposedly with \( h = 3? \) or \( h = 8? , s = 2 \)

Extract \( TF_{rst} = \{(b_{4j+1}, \ldots, b_{4j+4}) : (b_{4j+1}, \ldots, b_{4j+3}) = (rst)\} \) with

\(|TF_{ooo}| = |TF_{oo1}| = \ldots = |TF_{111}| = n_3 = 10^5\) from sequence.

Determine \( v_{rst}(i) := \frac{|\{j : (b_{4j+1}, \ldots, b_{4j+3}) = (rsti)\}|}{n_3} \).

’for each \((s, t) \in \{0, 1\}^2\) compare \( v_{0st} \) and \( v_{1st} \) with \( \uparrow \) at \( \alpha_3 = 0.0001’ \)
Quality Criteria for PRNGs


Generate $w_n \in \{0,1\}^L$ aus $(b_i)^{(Q+K)L}_{i=1}$. Let $A_n$ be the distance of $w_n$ to some identical predecessor,

i.e. $A_n = \begin{cases} n & \text{if there is no } i \geq 1 \text{ with } w_n = w_{n-i} \\
\min\{i \geq 1 : w_n = w_{n-i}\} & \text{else}
\end{cases}$

Let $T_8 = \frac{1}{K} \sum_{n=Q+1}^{Q+K} g(A_n)$ with $g(i) = \frac{1}{\log 2} \sum_{k=1}^{i-1} \frac{1}{k}$

$\approx \frac{\log i + \gamma + \frac{1}{2i} + \frac{1}{12i^2}}{\log 2} + \mathcal{O}(\frac{1}{i^4})$ with $\gamma \approx 0.577216$ Euler.

The bit sequence $(b_i)^{(Q+K)L}_{i=1}$ passes the Entropy Test, if $T_8$ approximately $N(\mu, \sigma^2)$-distributed with 'tabulated' $\mu = \mu(L, K)$ and $\sigma = \sigma(L, K)$.

BSI [9],[10] pp55 Test Procedure B: $(b_i)^n_{i=1}$ passes

with $L = 8$, $Q = 10 \cdot 2^L = 2560$, 
$K = 1000 \cdot 2^L = 256000$, $\mu = L$, 
$\sigma = c(L, K) \sqrt{\text{Var}(g(A_n))}/K$ if $T_8 > 7.976$

NB onesided? NB only for 'PTRNG'.
Quality Criteria for PRNGs

**Genesis: Maurer’s universal test** $\sim$ Coron $\neq$ BSI

Maurer [17]:
\[
f_{TU} = \frac{1}{K} \sum_{n=Q+1}^{Q+K} \log_2(A_n)
\]
with independent of $n$
\[
E(f_{TU}) = E(\log_2(A_n)) = 2^{-L} \sum_{i=1}^{\infty} (1 - 2^{-L})^{i-1} \log_2(i)
\]
and approximately
\[
\text{Var}(f_{TU}) = c^2(L, K) \text{Var}(\log_2(A_n))/K
\]
with
\[
c(L, K) \approx 0.7 - \frac{0.8}{L} + \left(4 + \frac{32}{L}\right) \frac{K^{-3/L}}{15}
\]
for $L \ll Q \ll K$
\[
\text{Var}(\log_2(A_n)) = 2^{-L} \sum_{i=1}^{\infty} (1 - 2^{-L})^{i-1} \log_2(i) - E^2(f_{TU}).
\]

Coron & Naccache [3],[4] generalize/correct Maurer to
\[
f^g_{TU} = \frac{1}{K} \sum_{n=Q+1}^{Q+K} g(A_n),
\]
which for $g(i) = \frac{1}{\log_2} \sum_{k=1}^{i-1} \frac{1}{k}$ gives
\[
E(f^g_{TU}) = L \text{ bit = Entropy of } L\text{-bit blocks of an ergodic stationary source as well as an exact representation and thus a better approximation of } c(L, K).
\]

**NB**
Table 1 for $\text{Var}(\log_2(A_n))$, $d(L)$ and $e(L)$ in
$c^2(L, K) = d(L) + e(L) \cdot 2^L/K$ in [3] for $\log_2$, in [4] for said $g$

BSI [9],[10] with said $g$, typo also in [11] SAGE $\sigma \approx 0.002$ vs
BSI $\sigma = 0.0014$ and $P(T_8 > 7.976) = P(U > -10.64) \approx 1$ ?

**NB** onesided? contrary to [17],[3],[4]
Comparison and Results

Here, in each version every random number generator it offers was tested except respectively for the legacy ones.

<table>
<thead>
<tr>
<th>BSI-Tests</th>
<th>MATLAB 7.0</th>
<th>MATLAB 7.3</th>
<th>MATLAB 7.11</th>
<th>MATLAB 7.13</th>
<th>MATLAB 8.1</th>
<th>SAGE 5.4.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monobit Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Poker Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Run Tests</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Longrun Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Autocorrelation Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Uniform Distribution Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Homogeneity Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Entropy Test</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

\(^2\)using ZZ.random_element(x=0, y=2, distribution="uniform")
\(^2\)with some interpretation of very problematic [10] pp50/51 and pp55
Resumé

MATLAB

- ease of use combined with optional selection of special generators
- randomization? usually by seeding using the system time

SAGE

- Here, only the generation of random integers is visible.

But weak legacy random number generators pass every BSI-test! Tests without discriminating power!

RNGs criteria for inclusion/exclusion of tests?
[24], [8] u.a.: binary matrix rank, DFT, template matching, linear complexity, random walks . . . ? error probabilities?  **not only for PRNGs?**

PRNG What if PRNG produces always the same \( n \) random numbers which passed all tests?
References


References’


[8] Charmaine Kenny: Random Number Generators – An Evaluation and Comparison of Random.org and Some Commonly Used Generators; Trinity College Dublin, April 2005,


References


References


www.weblearn.hs-bremen.de/risse/papers/MathEng11


[26] Werner Schindler: Functionality Classes and Evaluation Methodology for Deterministic Random Number Generators; BSI, version 2.0, 1999
AIS_20_Functionality_Classes_Evaluation_Methodology_DRNG_e.pdf

[27] Eric W. Weisstein: Run; MathWorld – A Wolfram Web Resource
http://mathworld.wolfram.com/Run.html