PRNG - Statistical testing

Maciej Szczutko

2024-12-17

Statistical test description

Testing binary expansion of constans

In this section we perform frequency monobit test for number $\pi, e, \sqrt{2}$. More formally we will use their binary expansion as the random bit sequence. We use provided files with binary expansions. For inference we will follow instruction from official **NIST** report.

Some important notes from report about most basic test.

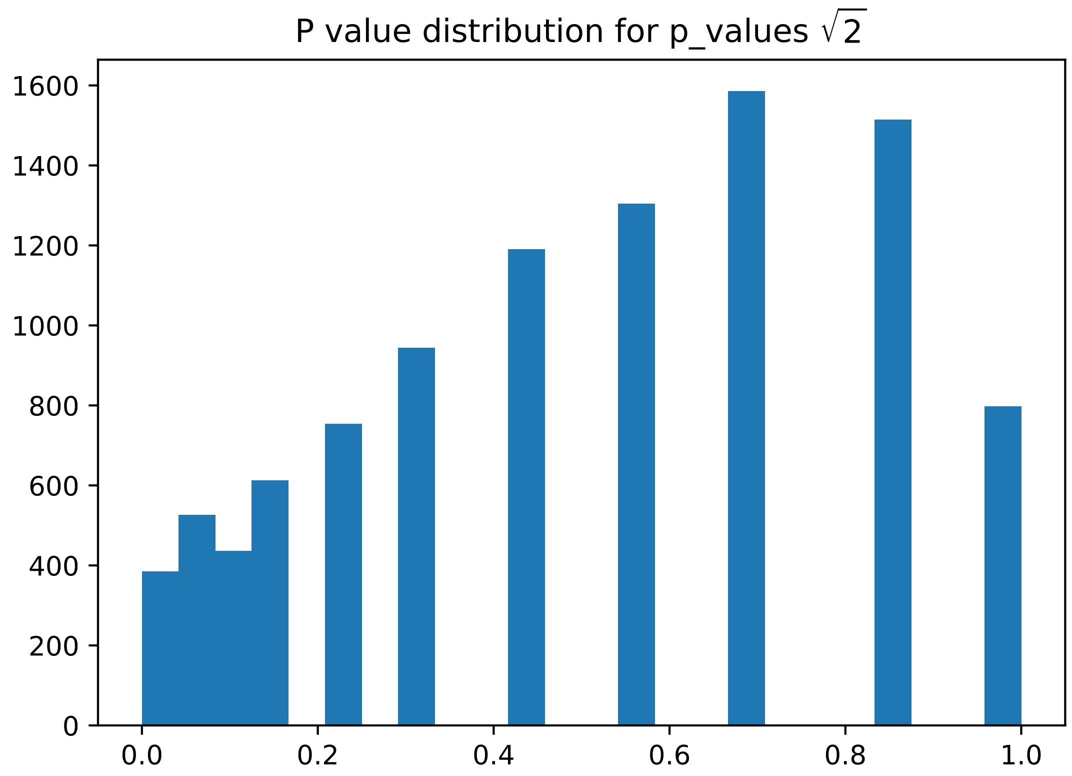
> 2.1.5 Decision Rule (at the 1% Level)
>
> If the computed P-value is < 0.01, then conclude that the sequence is non-random. Otherwise, conclude that the sequence is random.
>
> 2.1.7 Input Size Recommendation
>
> It is recommended that each sequence to be tested consist of a minimum of 100 bits (i.e., $n \geq 100$).

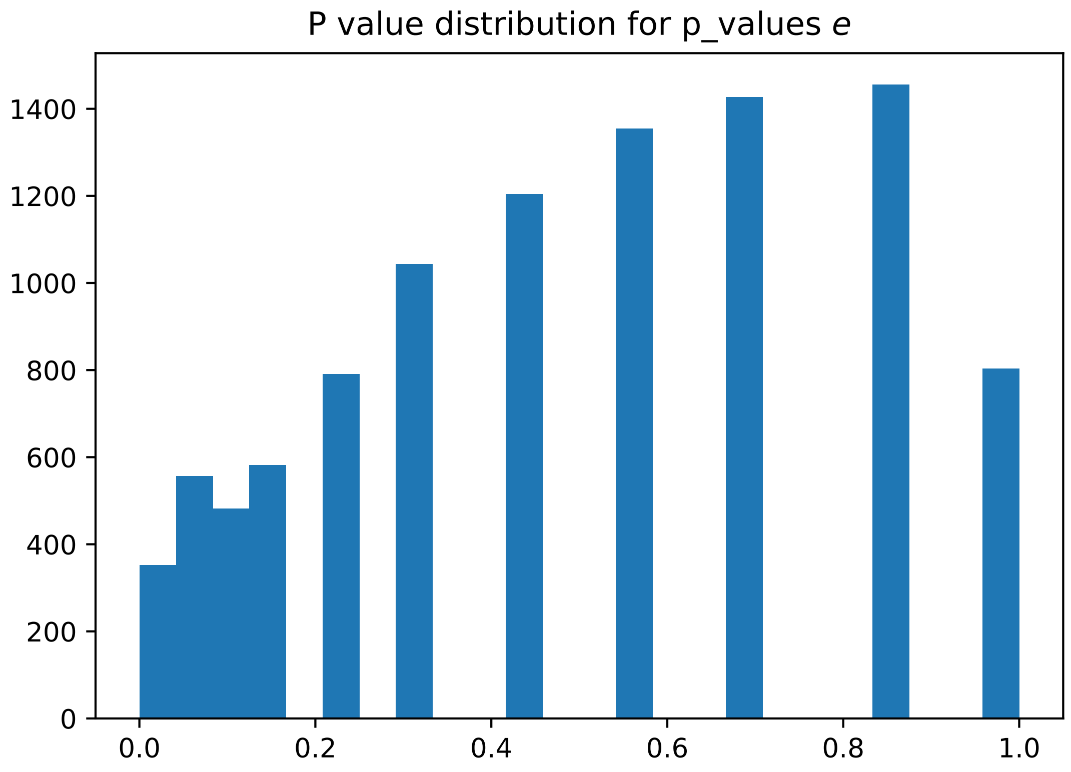
Constant name	p_{value}	Input Size
π	0.612315825298478	1004858
e	0.928460306674579	1004858
$\sqrt{2}$	0.817749242838411	1004859

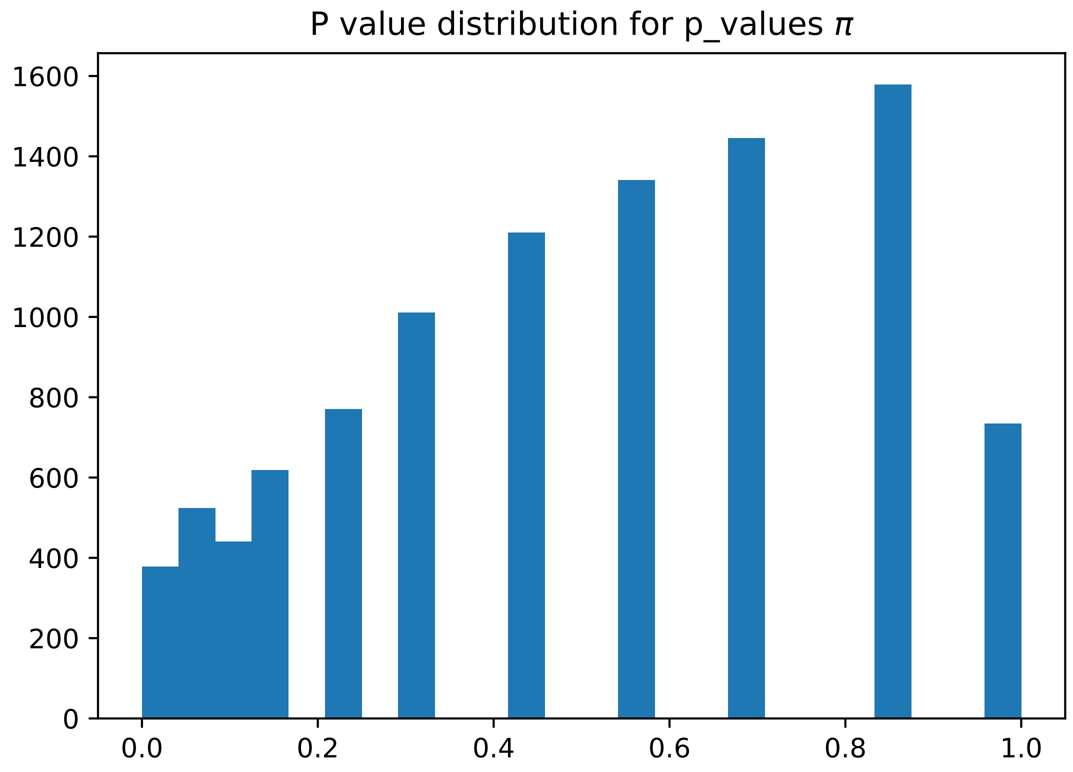
The size of out data is aligned with **NIST** recommendations. Authors recommend 0.01 as significance level for PRNG testing. From above table we conclude that binary expansion of each mentioned constants could be considered as random bit sequence.

Second level testing for bits

Because the size of sample are quite big we can try another approach. We split the sequence into a lot of smaller samples. We use the smallest recommended $n = 100$ for this test. The split method will be very straightforward. We simply take first 100 bits for first sample, another 100 for second sample and so on.


P value distribution for p_values $\sqrt{2}$
1600
1400
1200
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0


P value distribution for p_values e
0
200
400
600
800
1000
1200
1400
0.0
0.2
0.4
0.6
0.8
1.0



We can take a look at the histogram of p values. For me it doesn't look like uniformly distribute.

In below table we can see what fraction of samples pass the test.

Constant name	Fraction that pass
π	0.9867635
e	0.9876592
$\sqrt{2}$	0.9874602

RC(32)

The RC(32) random generator is a computational tool designed to produce sequences of random or pseudo-random numbers within a defined range. It typically employs algorithms optimized for speed and uniform distribution, making it suitable for simulations, cryptography, and data sampling. With its 32-bit architecture, the generator can provide high precision and a large range of outputs, ensuring versatility for various applications. Modern implementations often focus on enhancing randomness quality, reducing predictability, and adhering to statistical randomness tests like DIEHARD or TestU01. The RC(32) is favored in environments where reliable randomness is critical, such as in secure key generation or randomized algorithms. Additionally, its efficiency ensures minimal computational overhead, making it well-suited for embedded systems and high-performance computing tasks.

I provide some naive python implementation in *PRGA* and *KSA* modules. I will use my birthday (from year, month and day) as key (mod 32). So the key is [15, 8, 29].

```
## Elapsed generation time of sample size = 1000000 is 3.054799795150757s
```

We can check how the sequence begin 25, 15, 1, 22, 14, 4, 11, 26, 31, 1, 7, 28, 14, 15, 20, 3, 16, 21, 16, 16, 0, 21, 31, 19, 12, 15, 22, 4, 28, 1, 25, 9, 29, 20, 9, 21, 17, 17, 24, 21, 19, 22, 15, 31, 1, 20, 8, 31, 31, 12…

LCG$(13, 1, 5)$ used seed $= 42$

We can check how the sequence begin 8, 0, 5, 10, 2, 7, 12, 4, 9, 1, 6, 11, 3, 8, 0, 5, 10, 2, 7, 12, 4, 9, 1, 6, 11, 3, 8, 0, 5, 10, 2, 7, 12, 4, 9, 1, 6, 11, 3, 8, 0, 5, 10, 2, 7, 12, 4, 9, 1, 6…

For this example we can observe period of generator due relative small parameters. TODO determine longest possible cycle

LCG$(2^{10}, 3, 7)$ used seed $= 42$

We can check how the sequence begin 133, 406, 201, 610, 813, 398, 177, 538, 597, 774, 281, 850, 509, 510, 513, 522, 549, 630, 873, 578, 717, 110, 337, 1018, 1013, 998, 953, 818, 413, 222, 673, 1002, 965, 854, 521, 546, 621, 846, 497, 474, 405, 198, 601, 786, 317, 958, 833, 458, 357, 54…

Comparing with previous example we can't determine the period by manual inspection now.

GLCG$(2^{10}, 3, 7, 68)$

We can check how the sequence begin 78, 194, 429, 44, 433, 319, 279, 1000, 254, 533, 62, 517, 325, 305, 1014, 382, 197, 680, 637, 539, 143, 48, 762, 357, 58, 537, 105, 833, 622, 314, 557, 980, 825, 119, 423, 328, 22, 965, 198, 829, 237, 1009, 54, 182, 421, 368, 869, 467, 31, 816…

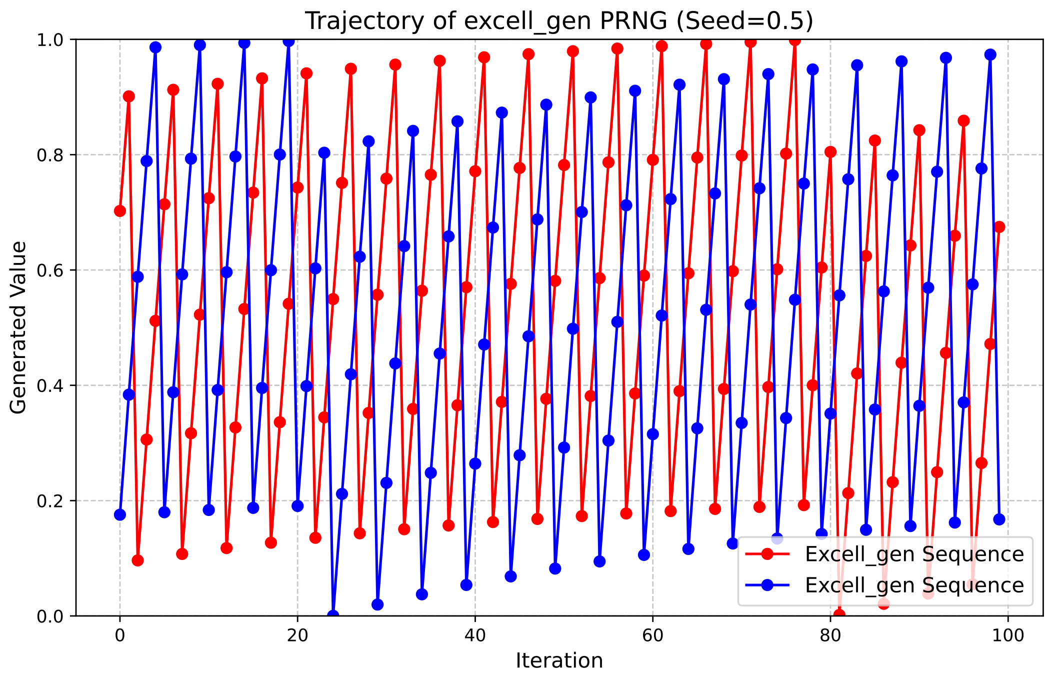
Excell

This PRNG is defined as

$$u_i = (0.9821 u_{i-1} + 0.211327) \mod 1$$

Actually is the ordinary $LCG(0.9821, 0.211327, 1)$. Probably this was used in Excell program till 2003 version.

```
## (0.0, 1.0)
```



Plotting the trajectory of beginning give us overview of quality of such generator. Maybe it simple and fast but not safe. We can observe pattern easily for difference seeds.