Pseudo-Random Number Generators

February 24, 2020

Yida Wang

Assuming that all random values in (0,1) are equally likely to be generated, the theoretical mean is obviously 0.500. Derive the theoretical value of the standard deviation.

The mean or expected value of the discrete variable X, denoted as $\mu$ or $E(X)$, is

$$\mu = E(X) = \frac{1}{b-a}\int_a^b x\,dx = \frac{a+b}{2}$$

The variance of X, demoted as $\sigma^2$ or V(X), is

$$\sigma^2 = V(X) = E(X-\mu)^2 = E(X^2) - E(X)^2$$

$$E(X^2) = \frac{1}{b-a}\int_a^2 x^2\,dx = \frac{b^3-a^3}{3(b-a)} = \frac{a^2+ab+b^2}{3}$$

$$E(X)^2 = \left(\frac{a+b}{2}\right)^2 = \frac{a^2+2ab+b^2}{4}$$

$$\sigma^2 = \frac{a^2+ab+b^2}{3} - \frac{a^2+2ab+b^2}{4} = \frac{(b-a)^2}{12}$$

The standard deviation is $\sigma = \sqrt{\sigma^2}$

$$\sigma = \sqrt{\frac{(1-0)^2}{12}} = \frac{1}{\sqrt{12}} = 0.288675134595$$

PRNG Program Output:

First 25 random values generated

| C++ Build-In Function | LFSR | LCG |
|---|---|---|
| 0.00125125888851588488 | 0.999999880790710449 | 0.774822711944580078 |
| 0.563585314493240119 | 0.999999761581420898 | 0.209692716598510742 |
| 0.193304239020966218 | 0.999999523162841797 | 0.200248658657073975 |
| 0.808740501113925592 | 0.999999046325683594 | 0.603763937950134277 |
| 0.585009308145390206 | 0.999998092651367188 | 0.455168724060058594 |
| 0.479873043000579869 | 0.999996185302734375 | 0.16328352689743042 |
| 0.350291451765495754 | 0.99999237060546875 | 0.866270661354064941 |
| 0.895962401196325531 | 0.9999847412109375 | 0.462814569473266602 |
| 0.822840052491836338 | 0.999969482421875 | 0.00484794378280639648 |
| 0.746604815820795298 | 0.99993896484375 | 0.998071551322937012 |
| 0.174108096560563974 | 0.9998779296875 | 0.347929716110229492 |
| 0.858943449201940989 | 0.999755859375 | 0.715225160121917725 |

| | | |
|---|---|---|
| 0.710501419110690646 | 0.99951171875 | 0.269772052764892578 |
| 0.513534958952604703 | 0.9990234375 | 0.802969634532928467 |
| 0.303994872890408052 | 0.998046875 | 0.196724176406860352 |
| 0.0149845881527146223 | 0.99609375 | 0.201556622982025146 |
| 0.0914029358806115882 | 0.9921875 | 0.212633013725280762 |
| 0.364452040162358493 | 0.984375059604644775 | 0.290182650089263916 |
| 0.147312845240638451 | 0.968750178813934326 | 0.162697315216064453 |
| 0.165898617511520741 | 0.937500417232513428 | 0.917162597179412842 |
| 0.988525040437025049 | 0.875000894069671631 | 0.475239038467407227 |
| 0.445692312387462986 | 0.750001847743988037 | 0.502730846405029297 |
| 0.119083223975341046 | 0.500003695487976074 | 0.888273000717163086 |
| 0.00466933194982757042 | 7.4505805969238281e-06 | 0.577593803405761719 |
| 0.00891140476699118014 | 1.4901161193847656e-05 | 0.22178959846496582 |

Summary Table

| # of values generated | Build-In Function | | LFSR | | LCG | |
|---|---|---|---|---|---|---|
| | Mean | Std dev | Mean | Std dev | Mean | Std dev |
| 25 | 0.414379 | 0.313594 | 0.880001 | 0.280948 | 0.460859 | 0.283604 |
| 50 | 0.47378 | 0.290865 | 0.560156 | 0.447441 | 0.488098 | 0.268389 |
| 100 | 0.494811 | 0.286705 | 0.579965 | 0.397113 | 0.497986 | 0.289465 |
| 500 | 0.497494 | 0.288725 | 0.528364 | 0.31421 | 0.497517 | 0.283768 |
| 1000 | 0.500417 | 0.288156 | 0.508304 | 0.30244 | 0.499273 | 0.285078 |
| 5000 | 0.504223 | 0.288094 | 0.510107 | 0.290091 | 0.498529 | 0.288348 |
| 10000 | 0.503787 | 0.288871 | 0.510144 | 0.288113 | 0.49813 | 0.288651 |
| Theoretical | 0.5 | 0.288675 | | | | |

Summary:

These three options for generating random numbers are not essentially equivalent. For the build-in function and LCG random numbers, we can see the mean and standard deviation are much closer to the theoretical mean and standard deviation compare to LFSR.

The build-in function random number is generated by an algorithm that returns some non-related numbers. This algorithm is using a seed to generate the series, which it initialized to some distinctive value by using function srand.

LFSR start from numbers closer to 1 because when it logical shift left 1 bit, the number does not change a lot initially since the most significant bit dominates. Compare to LCG, the number decrease significantly when it is using modulus operator.