Constructing a Suitable Pseudo-random Number Generator Using the *RND(X)* Function in PET BASIC

Your approach to this paper is unique. This would've been a publishable paper in compute magazine in 1980. Interestingly, I never saw this technique used in any code for this environment.

Aditya Singhvi

ATCS: Programming Languages

Period 3

Oct 28, 2020

The behavior of the random number generator function $RND(X)$ in PET BASIC was investigated, yielding flawed results with all primitive single-seed configurations. A suitable random number generator was eventually constructed by alternating calls to $RND(X)$ with a zero-seed input and a positive-seed input, storing only the results of the positive-seeded calls. The suitability of a given randomly-generated distribution was contingent on meeting the criteria of not displaying magnitude bias, extremity bias, value bias, or consistent bias.

1 Introduction

The problem of generating pseudo-random numbers with a deterministic algorithm has proved relevant since the invention of the first computers. With any programming language and device, having a passable pseudo-random number generator is essential for coding applications such as card games, simulations of random events, and even certain randomized algorithms that can outperform deterministic ones. PET BASIC, used beginning in 1997 on the Commodore PET, uses a built-in random number generator function, $RND(X)$ that theoretically generates a random real from 0 (inclusive) to 1 (non-inclusive) when given a seed value X [2][3]. However, the $RND(X)$ function is severely flawed, generating preset sequences that are often biased toward certain numbers, as detailed below. This report explores the construction of a plausible pseudo-random number generator in PET BASIC through manipulation of the $RND(X)$ function.

2 Procedure and Theory

To initially test the native random number generator, code (see Section 4) was written to loop through $100,000$ values of the $RND(X)$ function, with each set of $100,000$ running with preset seeds of $-999; -1; 0; 0.5; 1; 1.5; 100; 1,000; 100,000$; and $924,793$, as well as the constantly changing system time (ti). The value 924793 was chosen as it is a somewhat large prime number. The $RND(X)$ function was mapped into 100 discretized bins by scaling the output values from 0 to 1 into integer values from 0 to 99, ultimately generating a count of how many of the 100,000 values fell within each bin for each seed. The mean and standard deviation of the distribution were calculated within the code, while the final standard deviation of the counts (the bin values themselves) was calculated outside through a spreadsheet. The output from the emulator was copied and scraped through a simple Python script to prepare it into an analysis-ready format. After each code run, the VICE PET Emulator was reset to its startup configuration to ensure consistent results.

After running this simple configuration of the $RND(X)$ function, more complex configurations were attempted based on the initial results in search for a suitable pseudo-random function. The suitability of a certain configuration was based on four factors:

- **Magnitude Bias:** Bias toward high or low values
- **Extremity Bias:** Bias toward/against extremities
- **Value Bias:** Bias toward certain values
- **Consistent Bias:** Patterns across trials (code runs) favoring certain values

For a given random number generator to be suitable, it should eliminate all four categories of bias. The first three factors were determined quantitatively by comparing the mean of the distribution, the standard deviation of the distribution, and the standard deviation of the count of values in each bin to the respective theoretical values. The fourth factor was determined visually by graphing different runs of the same configuration and noting any patterns.

The mean of the distribution (referred to as the distribution mean henceforth) is theoretically the expected value $E[X]$ of a discretized uniform random variable X that takes on integer values from 0 to 99 inclusive each with a probability $p = 0.01$. The formula [1] for $E[X]$ is given by

$$E[X] = \frac{a+b}{2}, \tag{1}$$

where a and b represent the low and high values of the distribution. Substituting values of 0 and 99 respectively,

$$E[X] = 49.500, \tag{2}$$

yielding the theoretical distribution mean, used to identify magnitude bias. The standard deviation of the distribution (henceforth referred to as the distribution standard deviation) is modeled by the square root of the variance of X, given by the equation

$$\sigma[X] = \sqrt{\frac{(b-a-1)^2-1}{12}}. \tag{3}$$

Substituting for a, b yields

$$\sigma[X] = 28.866 \tag{4}$$

as the distribution standard deviation, used to determine extremity bias. Finally, the theoretical standard deviation of the counts across the bins (henceforth referred to as the bin standard deviation) can be modeled by a binomial random variable B with the number of trials $n = 100,000$ and probability of a number being sorted into a certain bin $p = 0.01$. The formula for the standard deviation of B is given by

$$\sigma[B] = \sqrt{np(1-p)}. \tag{5}$$

Substituting in values for n and p yields

$$\sigma[B] = 31.464 \tag{6}$$

as the theoretical bin standard deviation, used to determine value bias. Thus, for a certain number-generating configuration to be suitable, the first three metrics above should align closely with their respective theoretical values, and there should be no consistent pattern across separate trial runs of the same configuration.

3 Findings

3.1 Single-seed Configurations

Primitive single-seed configurations are defined as running the same seed for $100,000$ iterations in a row. These configurations yielded three categories of results as summarized in Table 1, dependent on the seed value: positive, negative, and zero. Positive seeds, independent of value, yielded the exact same reproducible sequence. Negative seeds produced only one specific value every iteration, independent of seed value. Although zero seeds did produce unique sequences with every trial, the results showed a consistent value bias. None of the three single-seed configuration categories satisfied all four criteria for a suitable random sequence.

Table 1: Summary Statistics for Distribution of single-seed configuration of $RND(X)$

Seed	Trials	Distribution Mean	Distribution Standard Deviation	Bin Standard Deviation
0	3	49.14 ± 0.040	28.96 ± 0.018	61 ± 2.0
Positive	–	49.58	28.97	40.77
Negative	–	0	0	10000
Theoretical	–	49.50	28.87	31.46

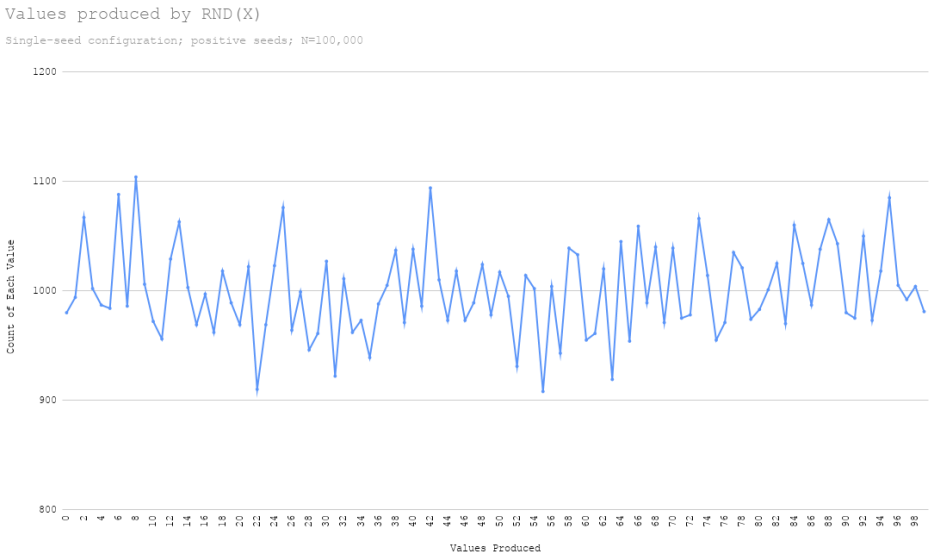

Figure 1: The distribution for a single-seed configuration with positive seeds, which yield the same sequence each time.

All positive seeds, independent of specific value, were found to produce the exact same number sequence with each trial, with constant positive seeds of $0.5, 1, 100, 1000, 100000$, and 924793, as well as the changing positive seed of the system time (ti) being tested. As shown in Table 1, both the distribution mean and the distribution standard deviation of the 100,000 values generated by positive seeds skewed slightly high, indicating a minimal high-magnitude bias favored toward extreme values. Furthermore, the bin standard deviation for the positive sequence was 40.77, much higher than the theoretical value of 31.46, indicating bias toward certain values. As the distribution was completely consistent across trials, as shown in Figure 1, the positive-seed distribution did not satisfy any of the four metrics for a suitable random sequence.

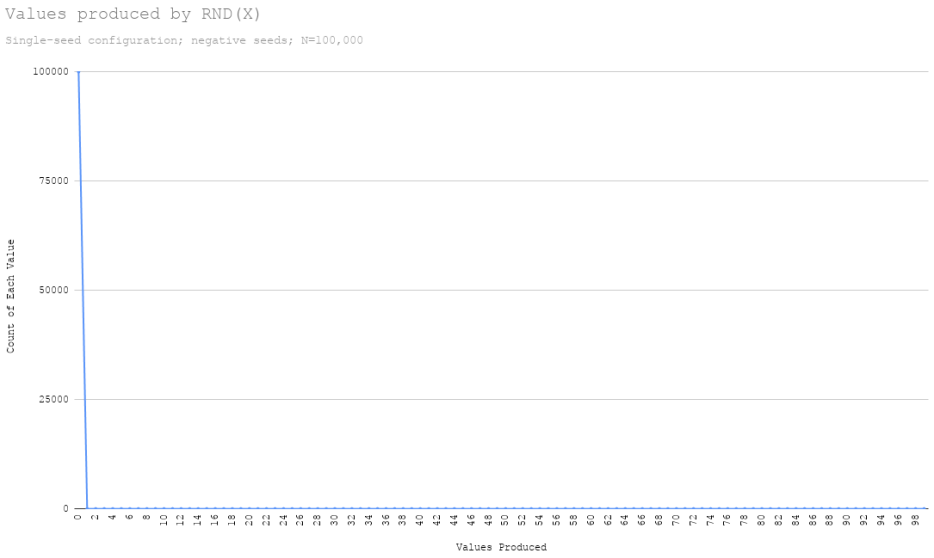

Figure 2: The distribution for a single-seed configuration with a negative seed, which returned the same constant every time.

When writing a paper do not use an * for multiplication. That is the symbol used in computer code. Use the appropriate symbol x for your scientific notation.

Negative seeds — no matter their value — did not produce a random sequence at all, instead just yielding the constant value $2.99196472 * 10^{-8}$ with every call to $RND(X)$, as demonstrated in Figure 2. Evidently, this is not a random sequence, so the negative-seed configuration also failed to satisfy any of the four metrics.

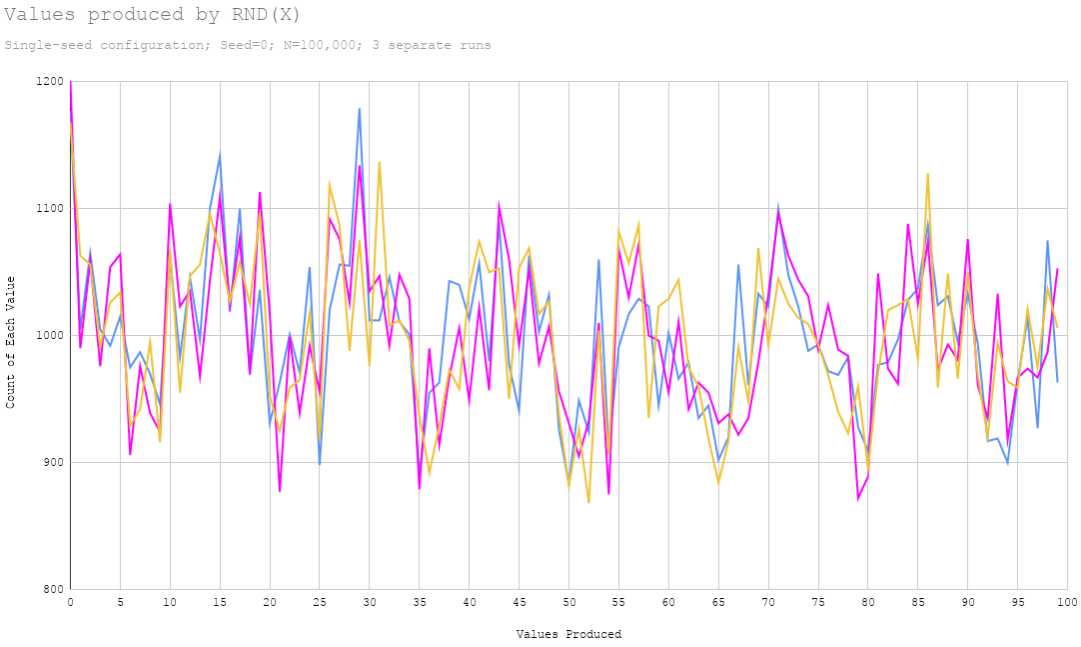

Figure 3: The distribution for three trials of a single-seed configuration with a zero-seed. This configuration produced different results each time, although the three trials aligned closely.

The zero-seed configuration produced a unique random sequence with each trial run, making it far superior to either of the positive or negative-seed configurations. However, as shown in Table 1, neither the distribution mean nor the distribution standard deviation of this configuration overlap with the theoretical values. The bin standard deviation of 61 ± 2.0 is even further above the 40.77 of the positive-seed configuration, suggesting a high degree of value bias. Indeed, Figure 3 confirms that notion, as the spikes on the graph are more extreme than that of Figure 1. Furthermore, although the actual values differed across three separate trials, the similarity of the three line charts overlayed in Figure 3 indicates a consistent bias favoring certain values. Thus, the primitive zero-seed configuration fails to satisfy any of the metrics for a suitable random number generator.

3.2 Zero-seeds Following Initial Non-Zero Seed

After single-seed configurations yielded disappointing results, configurations were tested that involved calling $RND(X)$ with a non-zero seed to prime the random number generator before running 100,000 consecutive, stored iterations of $RND(0)$. The seed value of 0 was chosen for the 100,000 iterations because of its non-deterministic nature as described in Section 3.1. It was determined that although this configuration produces a non-reproducible distribution, the general shape of the distribution is consistent across trials independent of seed value. As shown in Table 2, seed values of $-1, 1$, and 924793 were tested.

Table 2: Summary Statistics for Distribution of zero-seed following non-zero seed configuration of $RND(X)$

Initial Seed	Trials	Distribution Mean	Distribution Standard Deviation	Bin Standard Deviation
0	3	49.14 ± 0.040	28.96 ± 0.018	61 ± 2.0
1	1	49.48	28.66	75.55
-1	1	49.44	28.69	76.20
924793	1	49.35	28.65	80.71
Any non-zero	3	49.42 ± 0.067	28.67 ± 0.019	77 ± 2.8
Theoretical	–	49.50	28.87	31.46

All non-zero initial seeds — independent of value — produced similar distributions for the $RND(0)$ function. The distribution mean and distribution standard deviation for this distribution were both slightly lower than the corresponding theoretical values, as seen in Table 2, suggesting a thin bias toward low, non-extreme values. Furthermore, the bin standard deviation of 77 ± 2.8 deviates significantly from the theoretical value of 31.46; this suggests a high degree of bias favoring certain values, which corresponds to the extreme spikes displayed in the blue-shaded line charts in Figure 4. Furthermore, the chart clearly shows that all three blue-shaded lines — each from a different non-zero initial seed — follow the same general pattern, indicating consistent bias across trials. This consistent bias does not align with the bias for the single-zero-seed configuration, as shown by the pink line chart in Figure 4; however, this non-zero initial seed distribution still fails to meet the four criteria for a suitable random number generator.

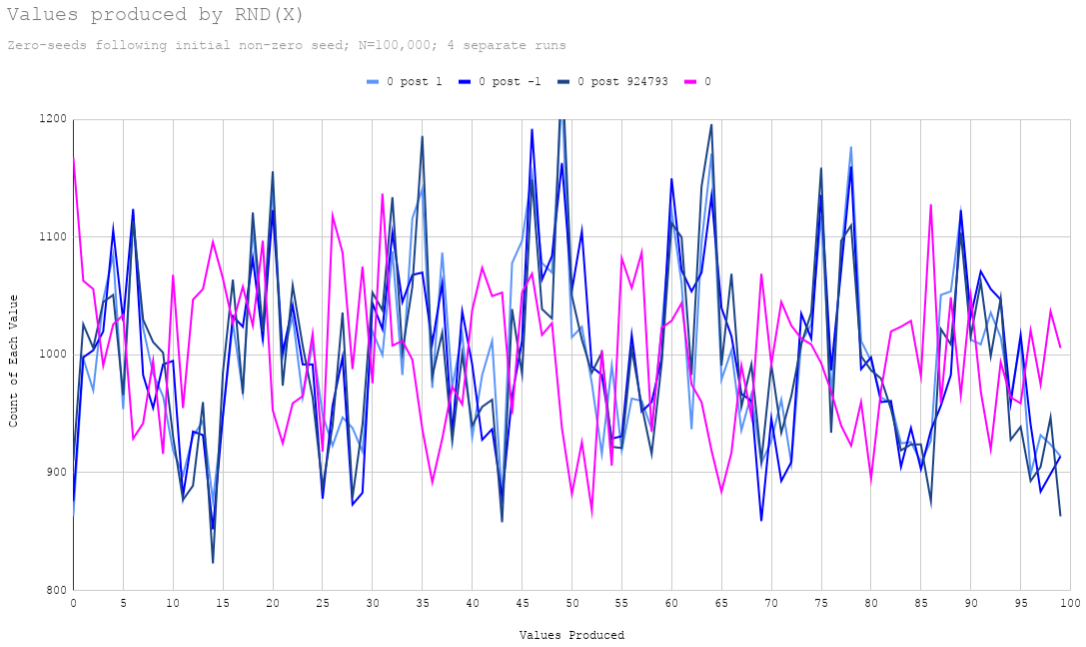

Figure 4: The distribution for three trials of a zero-seed configuration after being initially primed with a non-zero value, shown in shades of blue. The pink line represents a single-seed configuration with a zero-seed value for reference.

3.3 Alternating Zero and Positive seeds

Following the disappointing results of priming the $RND(0)$ function with a non-zero initial seed in Section 3.2, a configuration of calling $RND(X)$ with alternating zero and positive seeds was implemented. Two variations of this configuration were tried. First, only zero-seeded outputs of $RND(X)$ were stored, ignoring alternate positive-seeded outputs (explained in Section 3.3.1). Second, only outputs of the positive-seeded $RND(X)$ function were stored, ignoring the $RND(0)$ results in between (explained in Section 3.3.2).

3.3.1 Storing only Zero-Seed Results

Initially, only results from calling $RND(0)$ were stored, essentially using the calls to a positive-seeded $RND(X)$ as primers for each call of $RND(0)$. This decision was founded on results from Section 3.1 that show that calling $RND(X)$ with a zero-seed creates non-reproducible distributions, as opposed to the pre-set single-seed distributions of a positive seed. The $RND(X)$ function was thus called 200,000 times in total, with alternating output values being stored to produce a distribution of 100,000 values. Only trials with the alternate seed of 924,793 were tried as the configuration was evidently not yielding results satisfying all four criteria, as shown in Table 3.

Table 3: Distribution of alternating-seed configuration of $RND(X)$, storing only zero-seeds

Alternate Seed	Trials	Distribution Mean	Distribution Standard Deviation	Bin Standard Deviation
924793	3	49.52 ± 0.075	28.87 ± 0.057	38 ± 1.3
Theoretical	–	49.50	28.87	31.46

This configuration did not display any magnitude or extremity bias, as it produced distribution mean and distribution standard deviation values that matched the respective theoretical values (see Table 3). Furthermore, there was no consistent bias across trials, clearly visible in comparing Figure 5 to Figure 3. In Figure 5, the three trials, each run with the same alternate seed of 924,793, do not share a consistent pattern;

contrast this with Figure 3, where the three trials clearly match peaks and valleys. However, the configuration did produce some value bias, as shown by the bin standard deviation of 38 ± 1.3, differing significantly from the theoretical value of 31.46. Thus, although this alternate-seed configuration only storing zero-seeded outputs markedly improves over previous distributions, it satisfies only three out of the four criteria for a suitable random distribution.

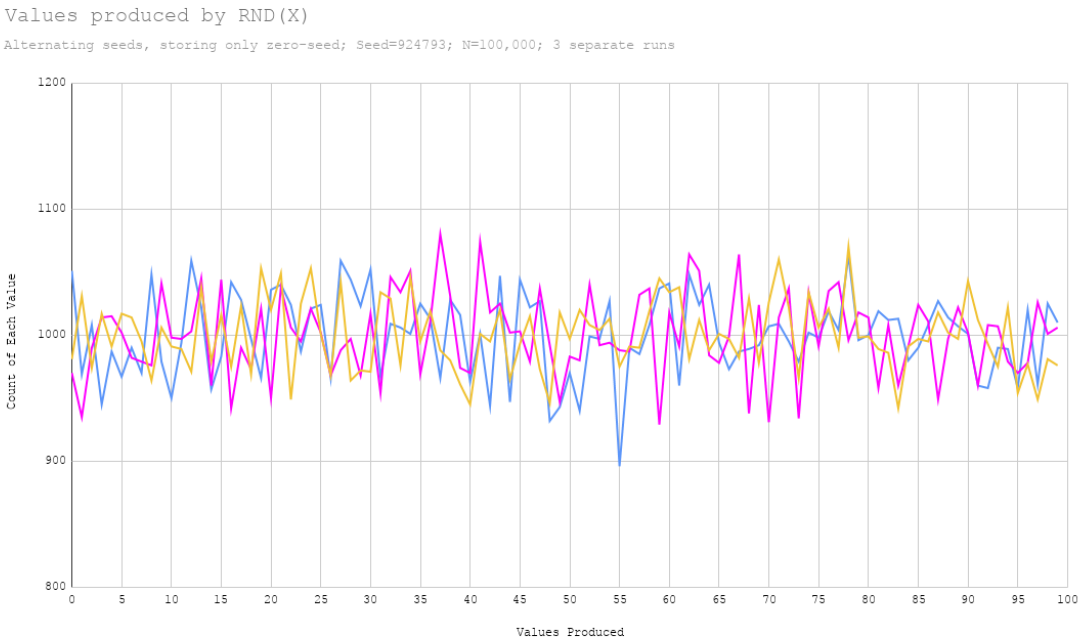

Figure 5: The distribution for three trials of an alternating zero and positive-seed configuration, storing only zero-seeded results.

3.3.2 Storing only Positive-Seed Results

After the narrow failure of Section 3.3.1, the converse configuration was implemented, alternating zero and positive seeds while only storing the results yielded from calling $RND(X)$ with positive seeds, thus using $RND(0)$ as a primer. Similarly to the previous configuration, $RND(X)$ was thus called 200,000 times, with alternating output values being stored to produce a distribution of 100,000 values. Alternate values of $1; 100; 924,793$; and ti were tested. As shown in Table 4 and Figure 6, this configuration did satisfy all four requirements for a suitable random number generator outlined in Section 2, yielding no magnitude, extremity, value, or consistent bias.

Table 4: Distribution of alternating-seed configuration of $RND(X)$, storing only positive-seeds

Alternate Seed	Trials	Distribution Mean	Distribution Standard Deviation	Bin Standard Deviation
0	3	49.14 ± 0.040	28.96 ± 0.018	61 ± 2.0
924793	3	49.52 ± 0.021	28.89 ± 0.022	31.0 ± 0.35
1	1	49.55	28.89	30.90
100	1	49.44	28.85	33.61
ti	1	49.51	28.86	29.95
Any Positive	6	49.51 ± 0.039	28.88 ± 0.025	31 ± 1.3
Theoretical	–	49.50	28.87	31.46

This configuration did not display any magnitude bias or extremity bias, as its distribution mean of 49.51 ± 0.039 and distribution standard deviation of 28.88 ± 0.025 match the respective theoretical values of 49.50 and 28.87, as shown in Table 4. Furthermore, this configuration yielded no discernible value bias either, with its bin standard deviation of 31 ± 1.3 matching the theoretical value of 31.46. This aligns with the relatively low spikes of the line graphs in Figure 6. Finally, as shown in Figure 6, no consistent bias emerges either as no clear pattern exists across trials. This alternate-seed configuration storing only positive-seeded values therefore satisfies all four criteria of a suitable random number generator. Thus, although running a single-seed configuration with positive-seed values yields a predetermined, reproducible sequence, priming the $RND(X)$ function by invoking $RND(0)$ prior to each call generates a reasonably random distribution.

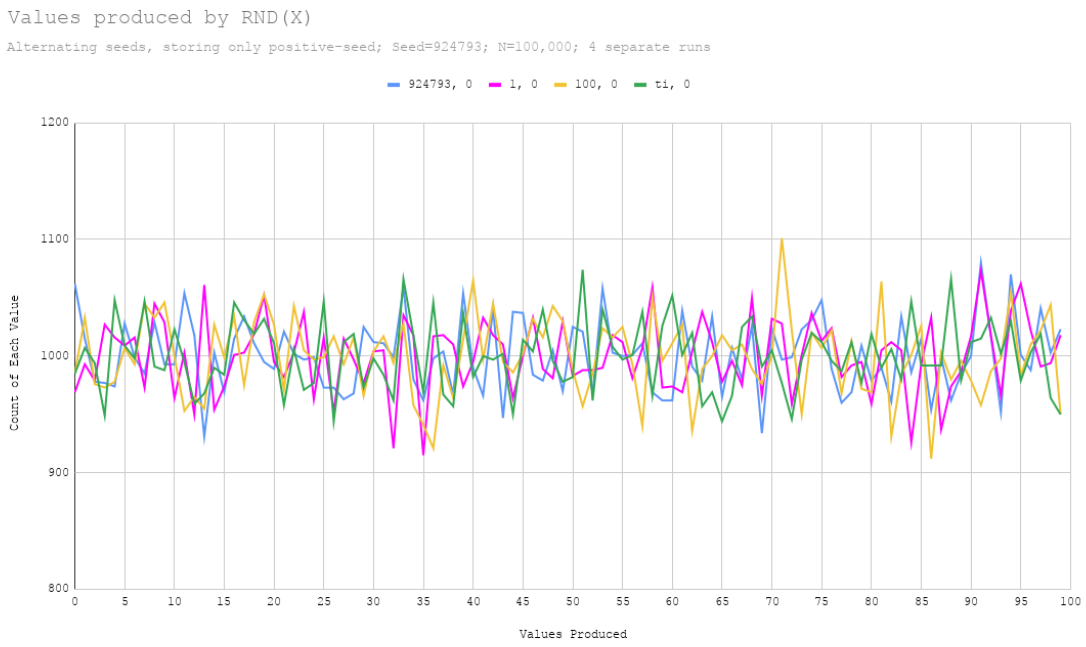

Figure 6: The distribution for four trials of an alternating zero and positive-seed configuration, storing only positive-seeded results. The four displayed trials were all run with different seeds.

4 Code

4.1 Single-seed Configuration

```
10 s = 0          :rem seed value
20 n = 100000     :rem number of trials
30 k = 100        :rem number of bins

35 sm = 0
40 dim v(k)

45 be = ti

50 for i = 1 to n
60 r = int(k * rnd(s))
70 v(r) = v(r) + 1
80 sm = sm + r
90 next i

95 print "Jiffies Elapsed:"; ti - be ; "for" ; n

100 mn = sm/n
110 sd = 0

120 for j = 0 to k - 1
130 print(v(j));
135 if int(j/10) = j/10 then print("")
140 sd = sd + (v(j) * (j - mn)^ 2)
150 next j

160 sd = sqr(sd/(n-1))

175 print ""
180 print "Mean:"; mn
190 print "StDev:"; sd
195 print "Seed Value:"; s

200 end
```

To configure for a changing *ti* seed, replace line 60 with:

```
60 r = int(k * rnd(ti))
```

4.2 Zero-seeds Following Initial Non-Zero Seed

Replace the appropriate lines from Section 4.1:

```
47 du = rnd(s)
60 r = int(k * rnd(0))
```

4.3 Alternating Zero and Positive-seeds

4.3.1 Store only Zero Seeds

Replace the appropriate lines from Section 4.1:

```
60 r = int(k * rnd(0))
85 du = rnd(s)
```

4.3.2 Store only Positive Seeds

Replace the appropriate lines from Section 4.1:

```
55 du = rnd(0)
```

References

[1] *Discrete Random Variables and Probability Distributions*. URL: https://homepage.divms.uiowa.edu/~rdecook/stat2020/notes/ch3_pt3.pdf.

[2] *Microsoft BASIC Timeline*. URL: http://www.weihenstephan.org/~michaste/pagetable/wait6502/msbasic_timeline.pdf.

[3] *User's Reference Manual: Commodore BASIC Version 4.0*. URL: http://www.zimmers.net/anonftp/pub/cbm/manuals/pet/BASIC_v4_Users_Reference_Manual.pdf.