1. Introduction

A pseudo-random number generator yields a set of numbers, \( x_1, x_2, \ldots, x_n \), which represent the values of a random sample from a uniform (or rectangular) distribution of the interval [0,1]. Such generators have been used extensively for simulations in statistical applications. They can also be exploited in illustrating distribution theory in statistics.

We shall give two limitations of a type of pseudo-random number generator which has often been used in the past. In developing a different generator, we shall illustrate some statistical distribution theory.

2. Congruential Generators

One of the most popular pseudo-random number generators is that based on the multiplicative congruential (or power residue) method. For our IBM 1130 computer, we have used the recursive relationship

\[
s_{i+1} = s_i \lambda \pmod{2^{15}},
\]

where \( s_o \) is an odd integer and \( \lambda \equiv 3 \pmod{8} \) or \( \lambda \equiv 5 \pmod{8} \).

---

1 Now at the University of Iowa.
If \( x_i = s_i / 32767 \), then \( x_1, x_2, \ldots, x_n \) represent the values of a random sample drawn from the uniform distribution on the interval \([0,1]\).

The multiplicative congruential method for generating pseudo-random numbers is quite efficient. Furthermore the numbers generated possess rather good statistical properties. One limitation of this method is that the maximum period is \( 2^{15}/4 = 8192 \). [1] Another limitation was pointed out by Marsaglia in an article entitled "Random numbers fall mainly in the planes." [2] Applying the conclusions of this article to \( E^2 \), the Euclidean plane, successive pairs of random numbers will all lie on at most 256 parallel lines.

To illustrate this second limitation, consider the congruential generator with \( s_0 = 71 \) and \( \lambda = 515 \). This generator will yield 8192 different numbers, \( x_1, x_2, x_3, \ldots, x_{8192} \) or 4096 pairs, \( (x_1, x_2), (x_3, x_4), \ldots, (x_{8191}, x_{8192}) \). Those pairs for which the first component is between .5 and .6 and the second component is between .3 and .4 are plotted in figure 1.

The congruential generator with \( s_0 = 71, \lambda = 699 \), yielded the pairs of numbers which are plotted in figure 2.

Because of the above limitations, we decided to develop a pseudo-random number generator which is based on the sum of the two congruential generators which have just been described.

3. The Sum of Uniform Random Variables

In this section we shall give the theory upon which our
pseudo-random number generator is based.

Let $U$ and $V$ be independent, identically distributed random variables which have uniform distributions on $[0,1]$, and let $W = U + V$. It is easy to show that the probability density function of $W$ is

$$
g(w) = \begin{cases} 
  w, & 0 \leq w \leq 1, \\
  2 - w, & 1 \leq w \leq 2, \\
  0 & \text{elsewhere}.
\end{cases}
$$

zero elsewhere. [3] (In order to give students a feeling for what this means, let them generate 2000 sums of pseudo-random numbers and draw a relative frequency histogram.)

Because our aim was to simulate a random sample from the uniform distribution on $[0,1]$ rather than the triangular distribution, we realized that another transformation was necessary. In particular we used the fact that $X = G(W)$ has a uniform distribution on $[0,1]$ where $G$ is the distribution function of $W$.

[4] For the triangular distribution, $G$ is defined by

$$
G(w) = \begin{cases} 
  0, & w < 0, \\
  w^2/2, & 0 \leq w < 1, \\
  1 - (1/2 - w)^2, & 1 \leq w < 2, \\
  1, & 2 \leq w.
\end{cases}
$$

4. A Pseudo-random Number Generator

Let

$$s_{i+1} = 515s_i \pmod{2^{15}},$$

$$t_{i+1} = 699t_i \pmod{2^{15}},$$

and $s_0 = t_0 = 71$. Let $u_i = s_i/32767$ and $v_i = t_i/32767$. 
Then \( u_1, u_2, u_3, \ldots \), and \( v_1, v_2, v_3, \ldots \), represent values of random samples from independent uniform distributions on the interval \([0,1]\). Let \( w_i = u_i + v_i \). We would expect \( w_1, w_2, w_3, \ldots \) to represent the values of a pseudo-random sample from the triangular distribution given in the last section.

Let

\[
x_i = \begin{cases} 
\frac{w_i^2}{2}, & w_i \leq 1, \\
1 - (1/2X2 - w_i)^2, & 1 < w_i. 
\end{cases}
\]

We would expect the sequence \( x_1, x_2, x_3, \ldots \) to represent the values of a pseudo-random sample from the uniform distribution on \([0,1]\).

In order to lengthen the period of this generator and also to permit n'tuples to lie out of the hyperplanes pointed out by Marsaglia [2], we decided on a "random" pairing of the \( u \) and \( v \) sequences.

In particular let \( k_1, k_2, k_3, \ldots \), be a sequence of integers, \( 1 \leq k_i \leq 2047 \). We let

\[
w_i = u_i + v_i, \quad i = 1, 2, \ldots, k_1;
\]

\[
= u_i + v_{k_1} + 1 + j, \quad i = k_1 + 1, \ldots, k_1 + k_2, \quad j = 1, \ldots, k_2;
\]

\[
= u_i + v_{k_1 + k_2} + 2 + j, \quad i = k_1 + k_2 + 1, \ldots, k_1 + k_2 + k_3,
\]

\[
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \q
\[ k_1 = 2047 - [2048 \, v_1], \]
\[ k_2 = 2047 - [2048 \, v_{k_1 + 1}], \]
\[ k_3 = 2047 - [2048 \, v_{k_1 + k_2 + 2}], \text{ etc., where } [t] \text{ is the} \]
greatest integer function.

5. Some Tests

The pseudo-random number generator described in section 4
no longer has a period of 8192. To determine whether pairs of
successively generated random numbers fall on parallel lines,
we generated 20,000 pairs of numbers. Of the 20,000 pairs, 211
fall in the square [.5,.6] x [.3,.4]. These points are plotted
in figure 3.

There are many tests which can be run to determine whether
\( x_1, x_2, x_3, \ldots \), do represent the values of a random sample
from the uniform distribution on [0,1]. We shall give three
rather simple chi-square goodness of fit tests.

For notation, partition the sample space into \( m \) cells. Let
\( p_i \) equal the probability of the \( i \)'th cell when the hypothesized
distribution is true. If \( n \) observations are taken and \( o_i \) is the
observed number of observations in the \( i \)'th cell, then

\[
\chi^2 = \sum_{i=1}^{m} \frac{(o_i - np_i)^2}{np_i}
\]

has an approximate chi-square distribution with \( v = m - 1 \) degrees
of freedom.

**Uniformity** The unit interval [0,1] was partitioned into
m = 100 subintervals, each of length .01. A sample of size
n = 40,000 pseudo-random numbers was generated. If these num-
bers do represent a random sample from a uniform distribution
on [0,1], p_i = .01 and np_i = 400, i = 1, 2, ..., 100. For this
set of data, \( \chi^2 = 74.415 \) which is approximately the 4th percent-
tile for a chi-square distribution with \( \nu = 99 \) degrees of free-
dom.

Pairs The unit square, [0,1] x [0,1] was partitioned into
100 squares, each square having sides of length .1. If succes-
sive pairs of numbers represent a random sample from a uniform
distribution on the unit square, then p_i = .01, i = 1, 2, ..., 100.
A sample of n = 20,000 pairs of numbers was generated. For this
random sample, \( \chi^2 = 116.780 \), which is approximately the 89th percentile for a chi-square distribution with \( \nu = 99 \) degrees of freedom.

Triples The unit cube, [0,1] x [0,1] x [0,1] was partitioned
into 1000 cubes, each cube having edges of length .1. If suc-
cessive triples of numbers represent a random sample from a uni-
form distribution on the unit cube, then p_i = .001, i = 1, 2, ..., 1000. A sample of n = 6000 triples of numbers was generated. For this random sample, \( \chi^2 = 989.312 \) which is approximately the 41st percentile for a chi-square distribution with \( \nu = 999 \) degrees of freedom.

6. Order Statistics

This section will illustrate how the pseudo-random number
generator can be used to illustrate the meaning of order statistics and at the same time test the pseudo-random number generator.

Let \( Y_1, Y_2, Y_3, Y_4 \) be the order statistics of a random sample of size 4 from the uniform distribution on \([0,1]\). I.e., if \( x_1, x_2, x_3, x_4 \) are the observed sample values, then
\[
y_1 = \min \{x_1, x_2, x_3, x_4\}, \ldots, y_4 = \max \{x_1, x_2, x_3, x_4\}.
\]
The probability density functions of these 4 order statistics are given by
\[
\begin{align*}
g_1(y_1) &= 4(1-y_1)^3, \quad 0 \leq y_1 \leq 1, \\
g_2(y_2) &= 12y_2(1-y_2)^2, \quad 0 \leq y_2 \leq 1, \\
g_3(y_3) &= 12y_3^2(1-y_3), \quad 0 \leq y_3 \leq 1, \\
g_4(y_4) &= 4y_4^3, \quad 0 \leq y_4 \leq 1.
\end{align*}
\]
Each of these functions is equal to zero elsewhere.

In addition, the probability density function of the range, \( R = Y_4 - Y_1 \), is given by
\[
h(r) = 12r^2(1-r), \quad 0 \leq r \leq 1, \quad \text{zero elsewhere.} \ [4]
\]
In order to give students some feeling for what is meant by the order statistics, 5000 sets of 4 numbers were generated. Each set of numbers was ordered and the range was calculated, yielding 5 sets of 5000 numbers. The interval \([0,1]\) was partitioned into 20 subintervals of length .05. The set of 5000 smallest numbers was grouped, etc. The relative frequency histograms and the respective probability density functions for \( Y_1, Y_2, Y_3, Y_4 \), and \( R \) are plotted in figures 4, 5, 6, 7, and 8, respectively. In doing this students should obtain a better understanding of
order statistics.

To test the pseudo-random number generator, a chi-square goodness of fit test was run for the above grouped data. It was necessary to combine some of the subintervals in order to make the expected number of outcomes for each sub-interval greater than 5. The results of the five chi-square goodness of fit tests are given in table 1. In addition the sample means are given. Recall that \( E(Y_i) = i/5, \ i = 1, 2, 3, 4 \) and \( E(R) = .6 \).

<table>
<thead>
<tr>
<th>Statistic</th>
<th>( \chi^2 )</th>
<th>( v )</th>
<th>Percentile</th>
<th>Sample Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y_1 )</td>
<td>10.495</td>
<td>16</td>
<td>16</td>
<td>.202</td>
</tr>
<tr>
<td>( Y_2 )</td>
<td>25.662</td>
<td>18</td>
<td>89</td>
<td>.399</td>
</tr>
<tr>
<td>( Y_3 )</td>
<td>13.595</td>
<td>18</td>
<td>24</td>
<td>.599</td>
</tr>
<tr>
<td>( Y_4 )</td>
<td>11.332</td>
<td>16</td>
<td>21</td>
<td>.799</td>
</tr>
<tr>
<td>( R )</td>
<td>10.262</td>
<td>18</td>
<td>8</td>
<td>.597</td>
</tr>
</tbody>
</table>

Table 1

7. The Standard Normal Distribution

To simulate random samples from a normal distribution, a sum of 12 pseudo-random numbers is often used. There is another rather interesting way to simulate normal random samples and at the same time illustrate change of variables for joint distributions.

Let \( U_1, U_2 \) be a random sample of size 2 from a uniform distribution on \([0,1] \).
Then
\[ X_1 = \sqrt{-2\ln U_1} \cdot \cos(2\pi U_2), \]
\[ X_2 = \sqrt{-2\ln U_1} \cdot \sin(2\pi U_2) \]
are independent normally distributed random variables with means, 0, and variances, 1. [5]

To illustrate this fact empirically, a random sample of size \( n = 2000 \) from a standard normal distribution was generated using this method. The data was grouped into 19 subintervals, \((-\infty, -2.55], (-2.55, -2.25], \ldots, (2.25, 2.55], (2.55, \infty)\). A chi-square goodness of fit test yielded \( \chi^2 = 16.0221 \) which is the 41st percentile for a chi-square distribution with \( v = 18 \) degrees of freedom. The histogram

The relative frequency for this data and the probability density function for the standard normal distribution are plotted in figure 9.

8. Conclusions

From the tests that have been run, the pseudo-random number generator that has been described seems to behave quite well. More importantly we see that statistical distribution theory can be illustrated empirically using a pseudo-random number generator.
9. Acknowledgement

The authors wish to thank NSF for a COSIP grant to Hope College which provides for summer student research grants, to the Research Corporation for faculty support, and to the Hope College Computer Center for their cooperation and generous computer time.

The figures were drawn on a CALCOMP Plotter, the purchase of which was made possible by grants from the Research Corporation, New York and the General Electric Foundation.
REFERENCES


