Physics 115/242
Randu: a bad random number generator

Peter Young

(February 5, 2001)

We are not going to do into the details of the various techniques used to generate pseudo-random number generators but here we will briefly describe one routine, which turned out to be very poor, to illustrate the pitfalls that can one can fall into.

Many random number generators are what is called *linear congruential generators* which generate a sequence of integers, $I_1, I_2, I_3, \cdots$, each between 0 and $m-1$ by the recurrence relation

$$I_{j+1} = aI_j + c \quad (mod\ m). \tag{1}$$

Here m is called the *modulus* and a and c are positive integers. Usually the I_j are converted to a real number between 0 and 1 as I_j/m. The routine is usually called as a function, say `ran(`*iseed*`)` where *iseed* is set as an initial "seed". On entry, the routine sets I_j to equal *iseed* and, on exit, the value of *iseed* is I_{j+1}, so *iseed* can be used on the next call to generate I_{j+2} etc.

IBM mainframes had such a random number generator, called `randu`, with $c = 0, a = 65539$ and $m = 2^{31}$. (Remember that on 32 bit machines like the mainframes, the largest positive integer is $2^{31} - 1$.) Here is a few simple lines of fortran code that implement it and which can easily be converted to C:

```
function randu(iseed)
    parameter (IMAX = 2147483647, XMAX_INV = 1./IMAX)
    iseed = iseed * 65539
    if (iseed < 0) iseed = iseed + IMAX + 1
    randu = iseed * XMAX_INV
end
```

The "mod m" operation is taken care of by adding 2^{31} if the left hand bit (the sign bit) is 1. Note that 2^{31} does not exist on 32 bit machines so we have to add $2^{31} - 1$ $(= IMAX)$ and then add 1.

If one generates a table of number and computes moments of the sample, the results look quite good with $\langle x^k \rangle \approx 1/(k+1)$. However, this is not sufficient; the numbers must be free of correlations with each other and randu fails badly in this regard.

Let us generate triplets of numbers x, y, z and plot them in three-dimensional space. Ideally they should fill a cube of unit side uniformly. However, *all* the triplets of random numbers generated by randu lie on only 15 planes. In fact the combination $9x - 6y + z$ is an *integer*(!) taking a value between -5 and 9. I demonstrate this in the next figure in which there are 1000 triplets of numbers. The result that all the triplets lie on these 15 planes doesn't seem to depend on the choice of initial seed. Here are the first three triplets starting with the *iseed* equal to 314159:

```
   x          y          z       9x-6y+z
0.58781    0.52555    0.86299    3.00000
0.44801    0.92114    0.49477   -1.00000
0.67837    0.61729    0.59842    3.00000
```

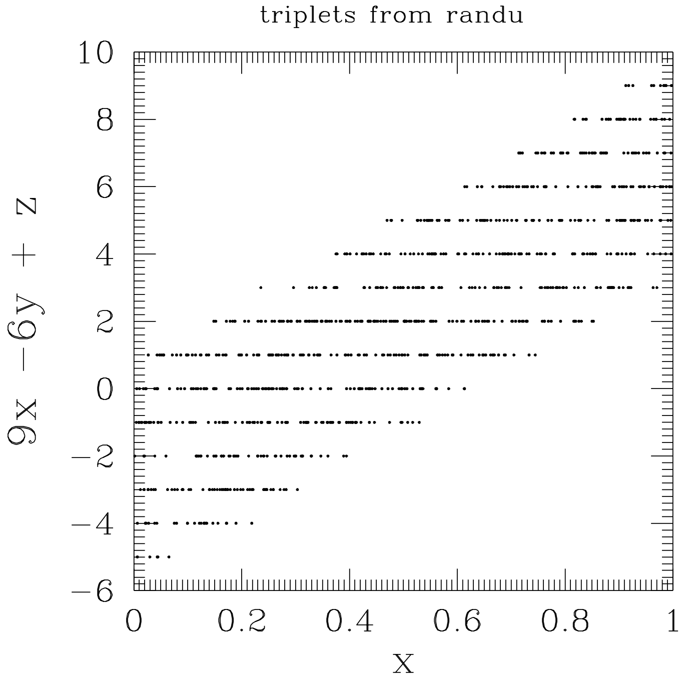

FIG. 1. Plot showing that all triplets of points generated by randu line on one of the 15 planes $9x - 6y + z = m$ where $m = -5, -4, \cdots, 8, 9$.

If one uses a decent random number generator, however, the points fill space in a random-looking manner, as shown in the next figure, which used ran0 from numerical recipes.

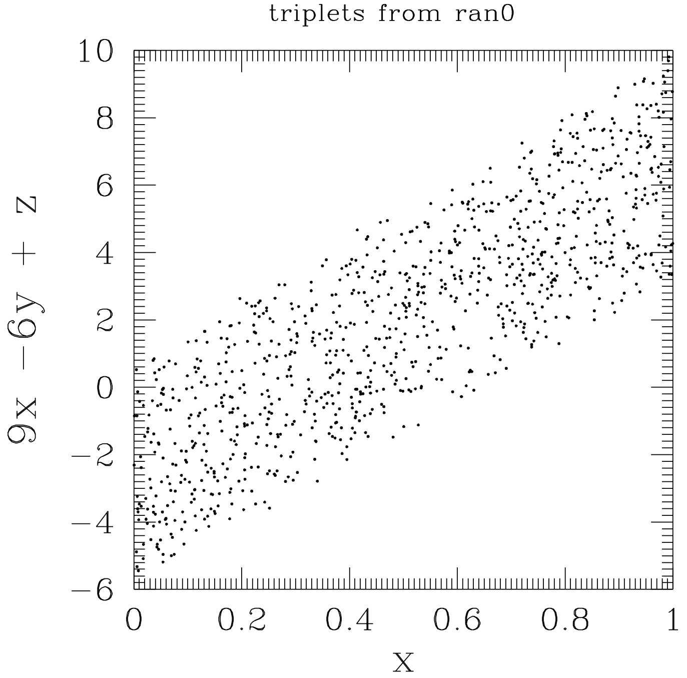

FIG. 2. Similar to Fig. 1 but for a "respectable" random number generator, ran0 in numerical recipes. The points now fill space in a random looking manner.