Random Number Generator of STMC

Bernd Berg

FSU MCMC Course ISC5228, August 28, 2007
Random Numbers and Fortran Code

According to Marsaglia and collaborators a list of desirable properties for random number generators is:

(i) **Randomness.** The generator should pass stringent tests for randomness.

(ii) **Long period.**

(iii) **Computational efficiency.**

(iv) **Repeatability.** Initial conditions (seed values) completely determine the resulting sequence of random variables.

(v) **Portability.** Identical sequences of random variables may be produced on a wide variety of computers (for given seed values).
(vi) Homogeneity. All subsets of bits of the numbers are random.

Physicists have added a number of their applications as new tests. In particular the exact solution of the $2d$ Ising model is used.

Frequently used is the congruential random number generator. It is based on the recursion

$$I_n = (a I_{n-1} + b) \mod(m)$$  \hspace{1cm} (1)

where $I_n$, $a$, $b$ and $m$ are integers. Uniformly distributed random numbers are then defined by the real numbers

$$x_n = I_n / m .$$  \hspace{1cm} (2)

Theorems (see the book by Knuth) state that good choices of the integer constants $a$, $b$ and $m$ exists, so that the series of random numbers becomes a permutation

$$\pi_0, \pi_1, \ldots \pi_{m-2}, \pi_{m-1}$$  \hspace{1cm} (3)

of $0, 1, \ldots m - 2, m - 1$. 

In STMC the random number generator by Marsaglia and collaborators is provided. It has a period $2^{144}$ and fulfills also the other desirable properties well. It relies on a combination of two generators:

$x_n$ from a lagged Fibonacci series $I_n = I_{n-r} - I_{n-s} \mod 2^{24}$, $r = 97$, $s = 33$.

$y_n$ from the arithmetic series $I - k$, $I - 2k$, $I - 3k$, $\ldots$, $\mod [2^{24} - 3]$.

For most applications this generator is a good compromise. Our Fortran code which implements Marsaglia random numbers consists of three subroutines:

- `rmaset.f` to set the initial state of the random number generator.
- `ranmar.f` which provides one random number per call.
- `rmasave.f` to save the final state of the generator.
The subroutine \texttt{rmaset.f} continues a saved state or initializes the generator to independent sequences of random numbers defined by distinct pairs of seeds:

\[-1801 \leq \text{iseed}_1 \leq 29527 \quad \text{and} \quad -9373 \leq \text{iseed}_2 \leq 20708. \quad (4)\]

This property makes the generator quite useful for parallel processing.

Table 1: Illustration of a start and a continuations run of the Marsaglia random number generator using the program \texttt{mar.f} with the default seeds (a0102-02).

<table>
<thead>
<tr>
<th>RANMAR INITIALIZED.</th>
<th>MARSAGLIA CONTINUATION.</th>
</tr>
</thead>
<tbody>
<tr>
<td>idat, $x_r = 1$</td>
<td>0.116391063</td>
</tr>
<tr>
<td>idat, $x_r = 2$</td>
<td>0.96484679</td>
</tr>
<tr>
<td>idat, $x_r = 3$</td>
<td>0.882970393</td>
</tr>
<tr>
<td>idat, $x_r = 4$</td>
<td>0.420486867</td>
</tr>
<tr>
<td>extra $x_r =$</td>
<td>0.495856345</td>
</tr>
<tr>
<td>idat, $x_r = 1$</td>
<td>0.495856345</td>
</tr>
<tr>
<td>idat, $x_r = 2$</td>
<td>0.577386141</td>
</tr>
<tr>
<td>idat, $x_r = 3$</td>
<td>0.942340136</td>
</tr>
<tr>
<td>idat, $x_r = 4$</td>
<td>0.243162394</td>
</tr>
<tr>
<td>extra $x_r =$</td>
<td>0.550126791</td>
</tr>
</tbody>
</table>
How to get and run the FORTRAN code?

Figure 1: The Fortran routines are provided and prepared to run in the a tree structure of folders depicted in this figure. This tree of directories unfolds from the downloaded file.

To **download** the Fortran code book visit the book website and follow the instructions given there.
The code is provided in the directories ForLib, ForProg and ForProc. ForLib contains a library of functions and subroutines which is closed in the sense that no reference to non-standard functions or subroutines outside the library is ever made. Fortran programs are contained in the folder ForProg and procedures for interactive use in ForProc.

**Assignment:** Marsaglia random numbers. See coursework website. Understand how to re-start the random number generator as well as how to perform different starts when the continuation data file ranmar.d does not exist.

Note: To compile properly, main programs have to be located two levels down from a root directory STMC.

**The hyperstructure of program dependencies introduced between the levels of the STMC directory tree should be kept intact!**

(Unless you really know better.)