APPENDIX 3

GAUSSIAN RANDOM NUMBER GENERATOR

The Algorithm for generating Gaussian random numbers is taken from Sophocles J. Orfanidis (1988).

It consists of two steps:

(a) Generation of uniform random numbers
(b) Generation of Gaussian random numbers from uniform random numbers

A3.1 UNIFORM RANDOM NUMBER GENERATOR

The Linear Congruential Generator (LCG) algorithm is used to generate uniform random numbers. The algorithm is defined as follows:

1. Given integers $i_1$ (the multiplier), $i_2$ (the displacement), $i_3$ (the modulus), and an initial integer seed $I_0$ in the range $0 \leq I_0 \leq i_3 - 1$, the algorithm computes the $n^{th}$ seed $I_n$ and corresponding uniformly distributed random number $b_n$ by the iteration:

2. $I_n = (i_1 I_{n-1} + i_2) \mod (i_3), \quad b_n = \frac{I_n}{i_3}$
The modulo \(- i_3\) operation restricts the range of the seeds \(I_n\) to the interval \(0 \leq I_n \leq i_3 - 1\), and therefore, the resulting uniform random number is, strictly speaking, in the interval \((0,1)\). The maximum possible length of such sequence is \(i_3\). However, not every choice of \(\{i_1, i_2, i_3\}\) will result in a maximal length sequences. There are certain guidelines for choosing \(\{i_1, i_2, i_3\}\), discussed in Sophocles J. Orfanidis (1988) that will ensure maximal length sequences. In our work, the values selected are \(i_1 = 25173, i_2 = 13849\) and \(i_3 = 65536\).

A Computer program named ‘ran’ is written to generate the uniform random numbers.

A 3.2 GAUSSIAN RANDOM NUMBER GENERATOR

It is remarkable that from uniform random numbers one can generate random numbers distributed according to any desired distribution. There exist several methods for generating Gaussian random numbers, such as the central limit theorem method, Box-Muller method, etc. The first method is based on the central limit theorem which states that the sum of a large number of independent random numbers

\[
b = b_1 + b_2 + b_3 + \ldots
\]

is essentially Gaussian, regardless of the distribution of the \(b_i\). Actually, keeping only 12 terms in this sum gives a very good approximation to a Gaussian random variable:

\[
b = b_1 + b_2 + b_3 + \ldots + b_{12}
\]

If \(b_i\) is uniformly distributed over \((0,1)\), then it has mean and variance \(\text{E}[b_i] = 0.5\), and \(\text{var}(b_i) = \frac{1}{2}\). The mean and variance (var) of \(b\) will be
$E[b] = E[b_1] + E[b_2] + \ldots + E[b_{12}] = 0.5 + 0.5 + \ldots + 0.5 = 6$

$\text{var}(b) = \text{var}(b_1) + \text{var}(b_2) + \ldots + \text{var}(b_{12})$

$= \frac{1}{12} + \frac{1}{12} + \ldots + \frac{1}{12} = 1$

$E$ represents the Expected value.

Thus, it is approximately Gaussian with variance 1. The range of values of $b$ is $0 < b < 12$ with mean at 6. It is an adequate approximation to a Gaussian distribution because there are $\pm 6$ SD on either side of the mean, and we know that for Gaussian random variables more than 99.99% of the values fall within $\pm 4$ SD. SD refers to the Standard Deviation. By shifting and scaling,

$x^1 = \text{SD} \ (b - 6) + i_3$

we obtain a Gaussian random number $x^1$ of mean $i_3$ and variance $\text{SD}^2$. A separate C - program, 'gran' is written to implement the Gaussian random number generator. It uses the uniform random number generator 'ran'. Starting with an integer seed, it produces a Gaussian- distributed random number of prescribed mean and variance, and updates the seed. We need to generate a block of independent and Gaussian-distributed numbers of prescribed mean and variance. Another computer program known as 'gauss' is written in C language to do this function by starting at an initial seed and making successive calls to 'gran'.