REFERENCES

Abramovitz, M., and Stegun, I. Handbook of mathematical functions with formulas, graphs, and mathematical tables. New York: Nover, 1965.

Amyx, J.W., Bass, D.M., Jr., and Whiting, R.L. Petroleum reservoir engineering: Physical properties. USA: McGraw-Hill book company, 1960.

Banks, J., and Carson, J.S. Discrete-event system simulation. New Jersey: Prentice-hall Inc., 1984.

Dickey, P.A. Petroleum development geology (2nd ed.). Oklahoma: Pennwell Publishing Company, 1981.

Forsythe, G.E., Malcolm, M.A., and Noler, C.B. Computer methods for mathematical computations. New Jersey: Prentice-Hall, Inc, 1977.

Hammersley, J.M., and Handscomb, D.C. Monte Carlo methods. London: Methuen, 1964.

Horn, M.E. Geostatistics and petroleum geology. New York: Van Nostrand reinhold, 1988.

Journel, A.B., and Huijbregts, Ch., J. Mining geostatistics. New York: Academic press, 1978.

Knuth, D.E. Seminumerical algorithms: The art of computer programming (Vol. 2, 2nd ed.). Canada: addison-wesley publishing company, 1981.

Krajewski, W.F., and Duffy, C.J. Estimation of correlation structure for a homogeneous isotropic random field: a simulation study. Computers & Geosciences 14 (1988): 113-122.

Mantoglou, A., and Wilson, J.L. The turning bands method for simulation of random fields using line generation by a spectral method. Water Resources Research 18 (October 1982): 1379-1394.

__________, Digital simulation of multivariate two- and three-dimensional stochastic process with a spectral turning bands method. Mathematical Geology 19 (1987): 129-149.

McCray, W. Petroleum evaluations and economic decisions. New Jersey: Prentice-Hall, Inc., 1975.

Mejia, J.M., and Rodriguez-Iturbe, I. On the synthesis of random field sampling from the spectrum: an application to the generation of hydrologic spatial process. Water Resources Research 10 (August 1974): 705-710.

Nind, T.E.W. Principles of oil well production. USA: McGraw-Hill book company, 1964.

Pritchard, K.C.G. Use of uncertainty analysis in evaluating hydrocarbon pore volumn in the Rainbow-Zama area. Journal of Petroleum Technology (November 1970): 1357-1366.

Shinozuka, M., and Jan, C.M. Digital simulation of random process and its applications. Journal of Sound and Vibration 25 (1972): 111-128.

Thomson, A. F.B., Ababou, R., and Gelhar, L.W. Implementation of three-dimensional turning bands random field generator. Water Resources Research 25 (October 1989): 2227-2243.

Walstrom, J.E., Mueller, T.D., and McFarlane, R.C. Evaluating uncertainty in engineering calculations. Journal of Petroleum Technology (December 1967): 1595-1603.

APPENDIX A

The following discussion follows closely the work of Banks and Carson, 1984.

<u>Random Number.</u>

The random number is an important ingredient in both the Monte Carlo simulation that is used to solve probabilistic problems and the spectral turning bands method for generating realizations with spatial correlation in this study.

The uniform random number is a sequence of numbers, R_1, R_2,..., which is in range between 0 and 1. They must have two important statistical properties, uniformity and independence. Each random number R_i is an independent sample drawn from a continuous uniform distribution between 0 and 1. That is, the probability density function is given by

$$f(x) = \begin{cases} 1, & 0 \leqslant x \leqslant 1 \\ 0, & \text{otherwise.} \end{cases} \tag{a.1}$$

The probability density function of random numbers is shown in Figure A.1.

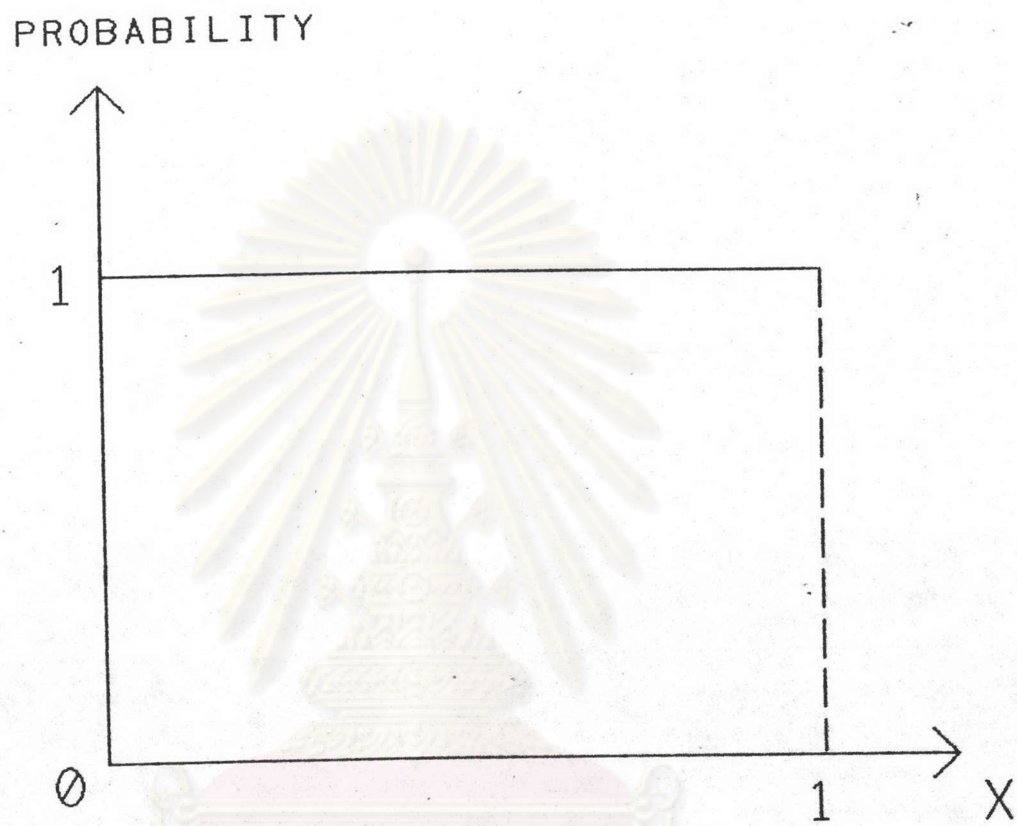

Figure A.1. Probability density function for random numbers.

The expected value of each R_i is given by

$$E(R) = \int_0^1 x\,dx = x^2/2\Big|_0^1 = 1/2, \qquad (a.2)$$

and, the variance is given by

$$V(R) = \int_0^1 x^2\,dx - [E(R)]^2 = x^3/3\Big|_0^1 - (1/2)^2 = 1/12. \qquad (a.3)$$

There are a number of important considerations should be mentioned for selecting method which can be used to generate random numbers as follows:

1. The method should be fast,
2. The method should not require a lot of core storage,
3. The method should have a sufficiently long cycle,
4. The random numbers should be replicable (to compare the result between systems),
5. Most important, and as indicated previously, the generated random numbers should closely approximate the ideal statistical preperties of uniformity and independence.

There are three classes of methods for producing random numbers:

1. Generating by using a random physical process. This method cannot be reproduced,
2. Computing the random numbers off line and store them in a

disk or tape file. This method will consume vary large storage memory,

3. The third and commonest method is to use an algorithm on line with computer program, to generate random numbers. These are available when they are needed and are perfectly reproducible for program checkout. Since such numbers are generated by an algorithm, they are not random at all and should be called pseudorandom.

Statistical tests for random numbers.

1. The Kolmogorov-Smirnov test.

This test compares the continuous cumulative distribution function (cdf), $F(x)$, of the uniform distribution to the empirical cdf, $S_N(x)$, of the sample of N observations. By definition,

$$F(x) = x, \qquad 0 \leqslant x \leqslant 1. \qquad (a.4)$$

If the sample from the random number generator is $R_1, R_2, \ldots, R_n$, the empirical cdf, $S_N(x)$, is defined by

$$S_N(x) = \frac{\text{number of } R_1, R_2, \ldots, R_n \text{ which are} \leqslant x}{N}. \qquad (a.5)$$

As N becomes larger, $S_N(x)$ should become a better approximation to $F(x)$, provided that the null hypothesis is true.

The cdf of an empirical distribution is a step function with jumps at each observed value. A bad source of random numbers will give empirical distribution functions that do not approximate F(x) sufficiently well.

The Kolmogorov-Smirnov test is based on the largest absolute deviation between F(x) and $S_N(x)$ over the range of the random variable. That is, it is based on the statistic

$$D = \max \left| F(x) - S_N(x) \right| . \tag{a.6}$$

The sampling distribution of D is known and is tabulated as a function of N in Table A.1. If $D \leqslant D_{crit}$ the hypothesis of no difference between the distribtuion of the generated numbers and the uniform distribution is not rejected.

Table A.1. Kolmogorov-Smirnov critical values, D_{crit}.

Degrees of Freedom (N)	$D_{0.10}$	$D_{0.05}$	$D_{0.01}$
1	0.950	0.975	0.995
2	0.776	0.842	0.929
3	0.642	0.708	0.828
4	0.564	0.624	0.733
5	0.510	0.565	0.669
6	0.470	0.521	0.618
7	0.438	0.486	0.577
8	0.411	0.457	0.543
9	0.388	0.432	0.514
10	0.368	0.410	0.490
11	0.352	0.391	0.468
12	0.338	0.375	0.450
13	0.325	0.361	0.433
14	0.314	0.349	0.418
15	0.304	0.338	0.404
16	0.295	0.328	0.392

Table A.1 (continued). Kolmogorov-Smirnov critical values, D_{crit}.

Degrees of Freedom (N)	$D_{0.10}$	$D_{0.05}$	$D_{0.01}$
17	0.286	0.318	0.381
18	0.278	0.309	0.371
19	0.272	0.301	0.363
20	0.264	0.294	0.356
25	0.240	0.270	0.320
30	0.220	0.240	0.290
35	0.210	0.230	0.270
over 35	$1.22/\sqrt{N}$	$1.36/\sqrt{N}$	$1.63/\sqrt{N}$

2. The chi-square test.

The chi-square test uses the sample statistic

$$\chi_o^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} . \qquad (a.7)$$

Where O_i is the observed number in the ith class, E_i is the expected number in the ith class, and n is the number of classes. For the uniform distribution, E_i, the expected number in each class, is given by

$$E_i = N/n. \qquad (a.8)$$

For equally spaced classes, where N is the total number of observations. It can be shown that the sampling distribution of X_o^2 is approximately the chi-square distribution with n-1 degrees of freedom.

3. Runs Tests.

a. Runs up and runs down.

The runs test examines the arrangement of numbers in a sequence to test the hypothesis of independence. Consider the following sequence of 15 numbers:

0.87	-0.15	+0.23	+0.45	+0.69	-0.32	-0.30	-0.19
+0.24	-0.18	+0.65	+0.82	+0.93	-0.22	+0.81	

The numbers are given a "+" or "-" depending on whether they are followed by a larger number or a smaller number. The sequence of 14 +'s and -'s is as follows:

– + + + – – – + – + + + – +

Each succession of +'s and –'s form a run. There are eight runs. The first run is of length one, the second and third are of length three, and so on. Further, there are four runs up and four runs down.

If a is the total number of runs in a sequence, the mean and variance of a is given by

$$\mu_a = (2N-1)/3 \qquad (a.9)$$

and

$$\sigma_a^2 = (16N-29)/90. \qquad (a.10)$$

For N > 20, the distribution of a is reasonably approximated by a normal distribution $N(\mu_a, \sigma_a^2)$. This approximation could be used to test the independence of numbers from a generator. In that case the standardized normal test statistic is developed by subtracting the mean from the observed number of runs, a, and dividing by the standard deviation. That is, the test statistic is

$$Z_o = (a - \mu_a) / \sigma_a \qquad (a.11)$$

and

$$Z_o = \frac{a - [(2N-1)/3]}{\sqrt{(16N-29)/90}} . \qquad (a.12)$$

Where $Z_o \sim N(0,1)$. Failure to reject the hypothesis of independence occurs when $-Z_{\alpha/2} \leqslant Z_o \leqslant Z_{\alpha/2}$, where α is the level of significance. The critical values and rejection region are shown in Figure A.2.

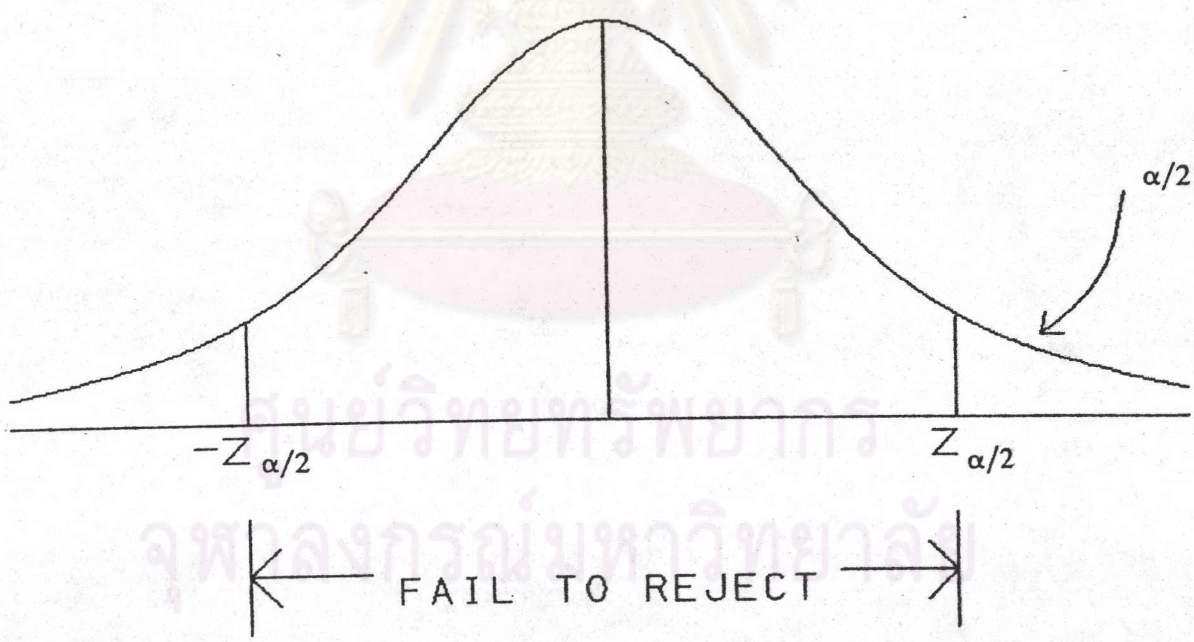

Figure A.2. Failure to reject hypothesis.

b. Runs above and below the mean.

The test for runs up and runs down is not completely adequate to assess the independence of a group of numbers. Consider the following sequence of 20 two-digit random numbers:

0.40	0.84	0.75	0.18	0.13	0.92	0.57	0.77	0.30	0.71
0.42	0.05	0.78	0.74	0.68	0.03	0.18	0.51	0.10	0.37

The pluses and minuses representing sample values above and below the mean are as follows:

− + + − − + + + − + − − + + + − − + − −

There is a run of length one below the mean followed by a run of length two above the mean, and so on. In all, there are 11 runs, five of which are above the mean, and six of which are below the mean. Let n_1 and n_2 be the number of individual observations above and below the mean and let b be the total number of runs. Note that the maximum number of runs is $N = n_1 + n_2$, and the minimum number of runs is one. Given n_1 and n_2, the mean, with a continuity correction suggested by Swed and Eisenhart (1943) (Banks and Carson, 1984), and the variance of b are given by

$$\mu_b = 2n_1n_2/N + 0.5 \qquad (a.13)$$

and

$$\sigma_b^2 = \frac{2n_1n_2(2n_1n_2 - N)}{N^2(N-1)}. \qquad (a.14)$$

For either n_1 or n_2 greather than 20, b is approximately normally distributed. The test statistic can be formed by subtracting the mean from the number of runs and dividing by the standard deviation, or

$$Z_o = \frac{b - \mu_b}{[\sigma_b^2]^{1/2}}. \qquad (a.15)$$

Failure to reject the hypothesis of independence occurs when $-Z_{\alpha/2} \leqslant Z \leqslant Z_{\alpha/2}$, where α is the level of significance.

c. length of runs.

Yet another concern is the length of runs. As an example of what might occur, consider the following sequence of numbers:

0.16,0.27,0.58,0.63,0.45,0.21,0.72,0.87,0.27,0.15,0.92,0.85,...

Assume that this sequence continues in a like fashion: two numbers below the mean followed by two numbers above the mean. A test of runs above and below the mean would detect no departure from independence. However, it is to be expected that runs other than of length two should occur.

Let Y_i be the number of runs of length i in a sequence of N numbers. For an independent sequence, the expected value of Y_i for runs up and down is given by

$$E(Y_i) = \frac{2}{(i+3)!}[N(i^2 + 3i + 1) - (i^3 + 3i^2 - i - 4)], \quad i \leqslant N - 2, \quad (a.16)$$

$$E(Y_i) = 2/N!, \qquad i = N - 1. \quad (a.17)$$

For runs above and below the mean, the expected value of Y_i is approximately given by

$$E(Y_i) = Nw_i/E(I), \qquad N>20. \qquad (a.18)$$

where w_i, the approximate probability that a run has length i, is given by

$$w_i = (n_1/N)^i(n_2/N) + (n_1/N)(n_2/N)^i, \qquad N>20. \qquad (a.19)$$

And where E(I), the approximate expected length of a run, is given by

$$E(I) = n_1/n_2 + n_2/n_1, \qquad N>20. \qquad (a.20)$$

The approximate expected total number of runs (of all lengths) in a sequence of length N, E(A), is given by

$$E(A) \quad = \quad N/E(I), \qquad\qquad N>20. \qquad (a.21)$$

The appropriate test is the chi-square test with O_i being the observed number of runs of length i. Then the test statistic is

$$\chi^2_o \quad = \quad \sum_{i=1}^{L} \frac{[O_i - E(Y_i)^2]}{E(Y_i)}. \qquad (a.22)$$

Where L = N - 1 for runs up and down and L = N for runs above and below the mean. If the null hypothesis of independence is true, then χ^2_o is approximately chi-square distributed with L-1 degrees of freedom. Note that, this test can be separated to 2 methods, length of runs up and down and length of runs above and below the mean.

5. Tests for Autocorrelation.

The tests for autocorrelation are concerned with the dependence between numbers in a sequence. As an example, consider the following sequence of numbers:

0.12 0.01 0.23 0.28 0.89 0.31 0.64 0.28 0.83 0.93

0.99 0.15 0.33 0.35 0.91 0.41 0.60 0.27 0.75 0.88

0.68 0.49 0.05 0.43 0.95 0.58 0.19 0.36 0.69 0.87

From a visual inspection, these numbers appear random, and they would probably pass all the tests presented to this point. However, an examination of the 5th, 10th, 15th (every five numbers beginning with the fifth), and so on, indicates a very large number in that position. Now, 30 numbers is a rather small sample size to reject a random number generator, but the notion is that numbers in the sequence might be related. In this particular section, a method for determining whether such a relationship exists is described. The relationship would not have to be all high numbers. It is possible to have all low numbers in the locations being examined, or the numbers may alternately shift from very high to very low.

The test to be described below requires the computation of the autocorrelation between every m numbers (m is also known as the lag) starting with the ith number. Thus, the autocorrelation ρ_{im} between the following numbers would be of interest: $R_i, R_{i+m}, R_{i+2m}, \ldots, R_{i+(M+1)m}$. The value M is the largest integer such that $i + (M+1)m \leqslant N$, where N is the total number of values in the sequence. (Thus, a subsequence of length M+2 is being tested.)

Since a nonzero autocorrelation implies a lack of independence, the following two-tailed test is appropriate:

$$H_o : \rho_{1m} = 0 \quad (a.23)$$

$$H_1 : \rho_{1m} \neq 0 \quad (a.24)$$

For large values of M, the distribution of the estimator of ρ_{1m}, or $\hat{\rho}_{1m}$, is approximately normal if the values R_1, R_{1+m}, R_{1+2m}, ..., $R_{1+(M+1)m}$ are uncorrelated. Then the test statistic can be formed as follows:

$$Z_o = \hat{\rho}_{1m} / \sigma_{\hat{\rho}1m} \quad (a.25)$$

Which is distributed normally with a mean of zero and a variance of 1, under the assumption of independence, for large M.

The formula for $\hat{\rho}_{1m}$, in a slightly different form, and the standard deviation of the estimator, $\sigma_{\hat{\rho}_{1m}}$, are given by Schmidt and Taylor (1970) (Banks and Carson, 1984) as follows:

$$\hat{\rho}_{1m} = [1/(M+1)] [\sum_{k=0}^{M} R_{1+km} R_{1+(k+1)m}] - 0.25 \quad (a.26)$$

and

$$\sigma_{\hat{\rho}_{1m}} = \frac{\sqrt{13M + 7}}{12(M + 1)}. \quad (a.27)$$

After computing Z_o, do not reject the null hypothesis of independence if $-Z_{\alpha/2} \leqslant Z_o \leqslant Z_{\alpha/2}$, where α is the level of significance. Figure A.2, presented earlier, illustrates this test.

If $\rho_{1m} > 0$, the subsequence is said to exhibit positive autocorrelation. In this case, successive values at lag m have a higher probability than expected of being close in value (i.e., high random numbers in the subsequence followed by high, and low followed by low). On the other hand, if $\rho_{1m} < 0$, the subsequence is exhibiting negative autocorrelation, which means that low random numbers tend to be followed by high one, and vice versa. The desired property of independence, which implies zero autocorrelation, means that there is no discernible relationship of the nature discussed here between successive random numbers at lag m.

4. Gap test.

The gap test is used to determine the significance of the interval between the recurrence of the same digit. A gap of length x occurs between the recurrence of some digit. The following example illustrates the length of gaps associated with the digit 3:

4,1,3,5,1,7,2,8,2,0,7,9,1,3,5,2,7,9,4,1,6,3

3,9,6,3,4,8,2,3,1,9,4,4,6,8,4,1,3,8,9,5,5,7

3,9,5,9,8,5,3,2,2,3,7,4,7,0,3,6,3,5,9,9,5,5

5,0,4,6,8,0,4,7,0,3,3,0,9,5,7,9,5,1,6,6,3,8

8,8,9,2,9,1,8,5,4,4,5,0,2,3,9,7,1,2,0,3,6,3

To facilitate the analysis, the digit 3 has been underlined. There are eighteen 3's in the list. Thus, only 17 gaps can occur. The first gap is of length 10, the second gap is of length 7, and so on. The frequency of the gaps is of interest. The probability of the first gap is determined as follows:

$$P(\text{gap of } 10) = \overbrace{P(\text{no } 3)\ldots P(\text{no } 3)}^{\text{there are 10 of these}} P(3)$$

$$= (0.9)^{10}(0.1). \qquad (a.28)$$

Since the probability that any digit is not a 3 is 0.9, and the probability that any digit is a 3 is 0.1. In general,

$$P(t \text{ followed by exactly } x \text{ non-}t \text{ digits}) = (0.9)^{x}(0.1)\ , \ x = 1,2,\ldots \qquad (a.29)$$

In the example above, only the digit 3 was examined. However, to fully analyze a set of numbers for independence using the gap test, every digit, 0,1,2,...,9, must be analyzed. The observed frequencies for all the digits are recorded and this is compared to the theoretical frequency using the Kolmogorov-Smirnov test for discretized data.

The theoretical frequency distribution for randomly ordered digits is given by

$$F(x) = 0.1 \sum_{n=0}^{x} (0.9)^{n} = 1-0.9^{x+1}. \qquad (a.30)$$

The procedure for the test follows the steps below:

Step 1. Specify the cdf for the theoretical frequency distribution given by equation (a.30) based on the selected class interval width.

Step 2. Arrange the observed sample of gaps in a cumulative distribution with these same classes.

Step 3. Find D, the maximum deviation between F(x) and $S_N(x)$,

Step 4. Determine the critical value, D_α, from Table A.1 for the specified value of α and the sample size N.

Step 5. If the calculated value of D is greater than the tabulated value of D_α, the null hypothesis of independence is rejected.

5. Poker test.

Treats numbers grouped together as a poker hand. Then the hands obtained are compared to what is expected using the chi-square test.

The poker test for independence is based on the frequency with which certain digits are repeated in a series of numbers. The following example shows an unusual amount of repetition:

0.255, 0.577, 0.331, 0.414, 0.828, 0.909, 0.303, 0.001, ...

In each case, a pair of like digits appears in the number that was generated. In three-digit numbers there are only three possibilities, as follows:

a. The individual numbers can all be different,

b. The individual numbers can all be the same,

c. There can be one pair of like digits.

The probability associated with each of these possibilities is given by the following:

$$\begin{aligned} P(\text{three different digits}) &= P(\text{second different from the first}) \\ &\quad \times P(\text{third different from the first and second}) \\ &= (0.9)(0.8) = 0.72 \end{aligned} \qquad (a.31)$$

P(three like digits) = P(second digit same as the first)

x P(third digit same as the first)

= (0.1)(0.1) = 0.01 (a.32)

P(exactly one pair) = 1 - 0.72 - 0.01 = 0.27 (a.33)

Random Variate Generation.

Procedures for sampling from a various distributions concerned in this study are discribed here. Four types of distribution are used as follows:

1. Uniform Distribution,
2. Triangular Distribution,
3. Normal Distribution,
4. Log-normal Distribution.

1. Uniform Distribution.

The uniform distribution is used when the upper and lower limits of the range of a variable can be specified and when any of the values between these limits is as likely to occur as another value. The way in which the distribution is used is illustrated in Figure A.3. A solution is obtained by taking a one-to-one correspondence between the uniform (rectangular) distribution of the random number and the cumulative probability of the variable X.

The cumulative probability of X is given by

$$F(X) = \frac{X - X_L}{X_H - X_L}. \qquad (a.34)$$

Replacing F(X) with R_N, the uniformly distributed random number, and solving for X gives

$$X = X_L + R_N(X_H - X_L). \qquad (a.35)$$

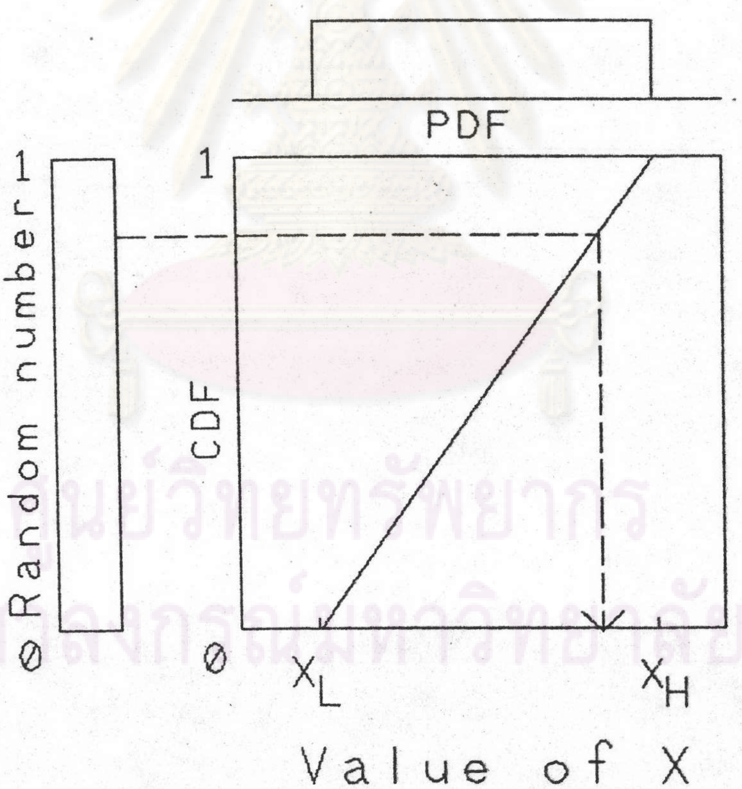

Figure A.3. Selecting random values from a uniform distribution.

2. Triangular Distribution.

The triangular distribution is used when the upper and lower limits as well as a most likely value can be specified. It is also apparent that the probability of an outcome occurring close to the limits of the range generally becomes progressively smaller, unless the distribution should happen to be highly skewed to one side. The manner of using the triangular distribution is illustrated in Figure A.4.

Again, a one-to-one correspondence is taken between cumulative probability and the uniformly distributed random number. In this case the equation used depends on whether the value of X will be greater or smaller than X_M, the mode.

When $X_L \leqslant X \leqslant X_M$, the cumulative probability of X is given by

$$F(X) = \left[\frac{X - X_L}{X_M - X_L}\right]^2 \left[\frac{X_M - X_L}{X_H - X_L}\right]. \qquad (a.36)$$

When $X_M \leqslant X \leqslant X_H$, the cumulative probability of X is given by

$$F(X) = 1 - \left[\frac{X_H - X}{X_H - X_M}\right]^2 \left[\frac{X_H - X_M}{X_H - X_L}\right]. \qquad (a.37)$$

Replacing F(X) with R_N, one has

if $R_N \leqslant (X_M - X_L)/(X_H - X_L)$:

$$X = X_L + \sqrt{(X_M - X_L)(X_H - X_L)R_N}\ , \qquad \text{(a.38)}$$

if $R_N \geqslant (X_M - X_L)/(X_H - X_L)$:

$$X = X_H - \sqrt{(X_H - X_M)(X_H - X_L)(1 - R_N)}. \qquad \text{(a.39)}$$

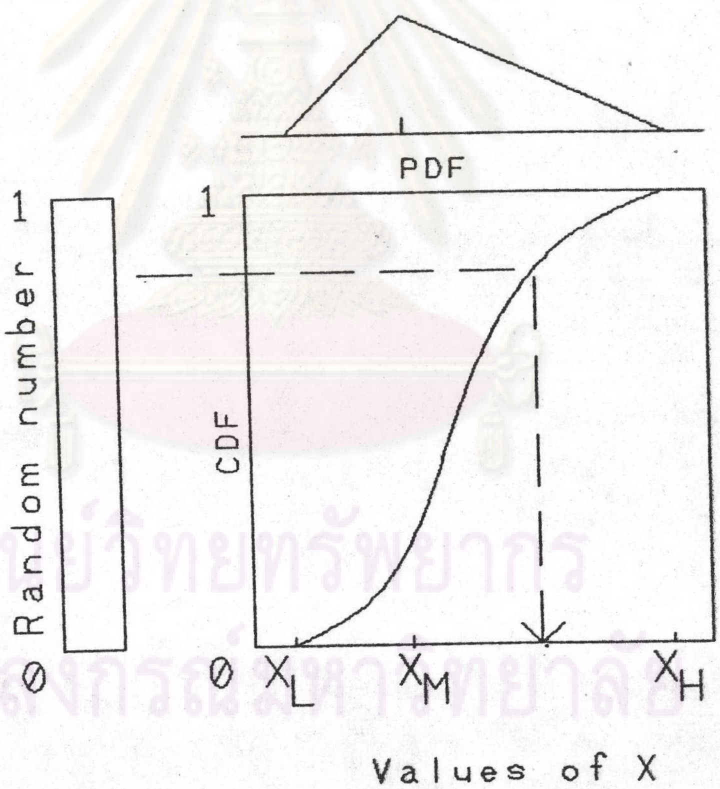

Figure A.4. Selecting random values from a triangular distribution.

3. Normal Distribution.

Many methods have been developed for generating normally distributed random variates. The standard normal cumulative distribution function is given by

$$\Phi(X) = \int_{-\infty}^{X} \frac{1}{\sqrt{2\pi}} e^{-t^2/2}\, dt \quad , -\infty < X < \infty . \qquad \text{(a.40)}$$

The normal probability distribution function is shown in Figure A.5.

A direct transformation that produces an independent pair of standard normal variates with mean zero and variance 1 is described here. The method is due to Box and Muller (1958) (Banks and Carson, 1984). Although not as efficent as many more modern techniques, it is easy to program in a scientific language such as FORTRAN.

Consider two standard normal random veriables, Z_1 and Z_2, plotted as a point in the plane as shown in Figure A.6 and represented in polar coordinates as

$$Z_1 = B \cos \theta, \qquad \text{(a.41)}$$

and

$$Z_2 = B \sin \theta. \qquad \text{(a.42)}$$

It is known that $B^2 = Z_1^2 + Z_2^2$ has the chi-square distribution with 2 degrees of freedom, which is equivalent to an exponential disribution with mean 2. Thus, the radius, B, can be generated by use of following equation:

$$B = (-2\ln R)^{1/2}. \qquad (a.43)$$

By the symmetry of the normal distribution, it seems reasonable to suppose, and indeed it is the case, that the angle θ is uniformly distributed between 0 and 2π radians. In addition, the radius, B, and the angle, θ, are mutually independent. Combining three equations gives a direct method for generating two independent standard normal variates, Z_1 and Z_2, from two independent random numbers R_1 and R_2.

$$Z_1 = (-2 \ln R_1)^{1/2} \cos(2\pi R_2) \qquad (a.44)$$

$$Z_2 = (-2 \ln R_1)^{1/2} \sin(2\pi R_2) \qquad (a.45)$$

The algorithms for generated random variable with 0 mean and 1 standard deviation suggested by James Bell of Stanford (Forsythe et al., 1977) are

Step 1. Form two uniform deviates U_1 ,U_2 on [0,1)

Step 2. Form $V_1 = 2U_1 - 1$ and $V_2 = 2U_2 - 1$ to get two uniform deviates on [-1,1)

Step 3. Form $S = V_1^2 + V_2^2$. If $S > 1$, discard V_1, V_2 and go to step 1. (We lose 22 % of our efficiency here.) If $S \leq 1$, then we have a random point (V_1, V_2) in the unit circle

Step 4. Form

$$Z_1 = V_1 \sqrt{\frac{-2 \ln S}{S}} , \qquad Z_2 = V_2 \sqrt{\frac{-2 \ln S}{S}} \qquad \text{(a.46)}$$

Step 5. Then the random samples are equal

$$X_1 = \text{mean} + Z_1 \times \text{standard deviation} \qquad \text{(a.47)}$$

$$X_2 = \text{mean} + Z_2 \times \text{standard deviation.} \qquad \text{(a.48)}$$

4. Log-Normal Distribution.

The probability density of log-normal distribution is shown in Figure A.5. The procedure for generating realization having log-normal distribution is same manner as the normal distribution but the input parameter, mean and standard deviation in this case is in log-term and the value obtained from the program will be transformed using anti-log function.

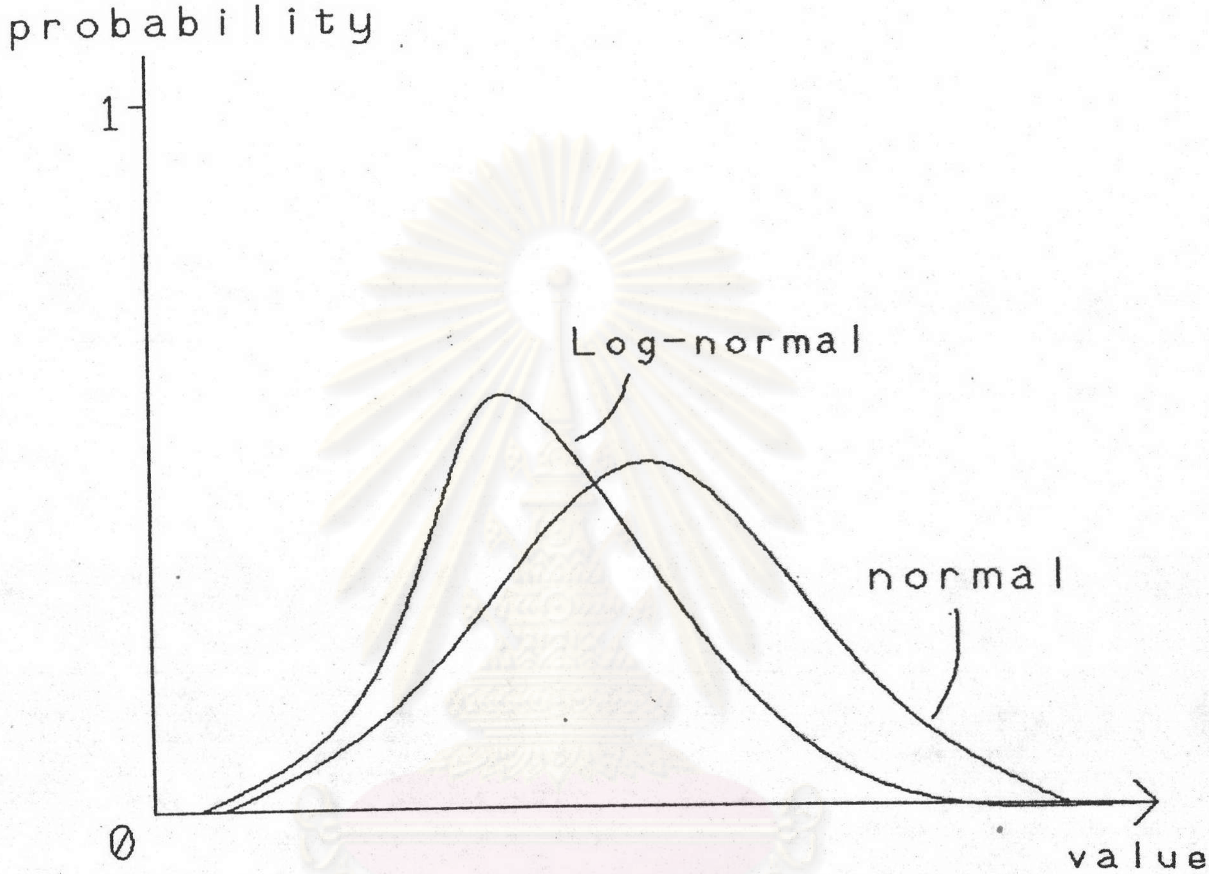

Figure A.5. Probability density function of the normal distribution and Log-normal distribution.

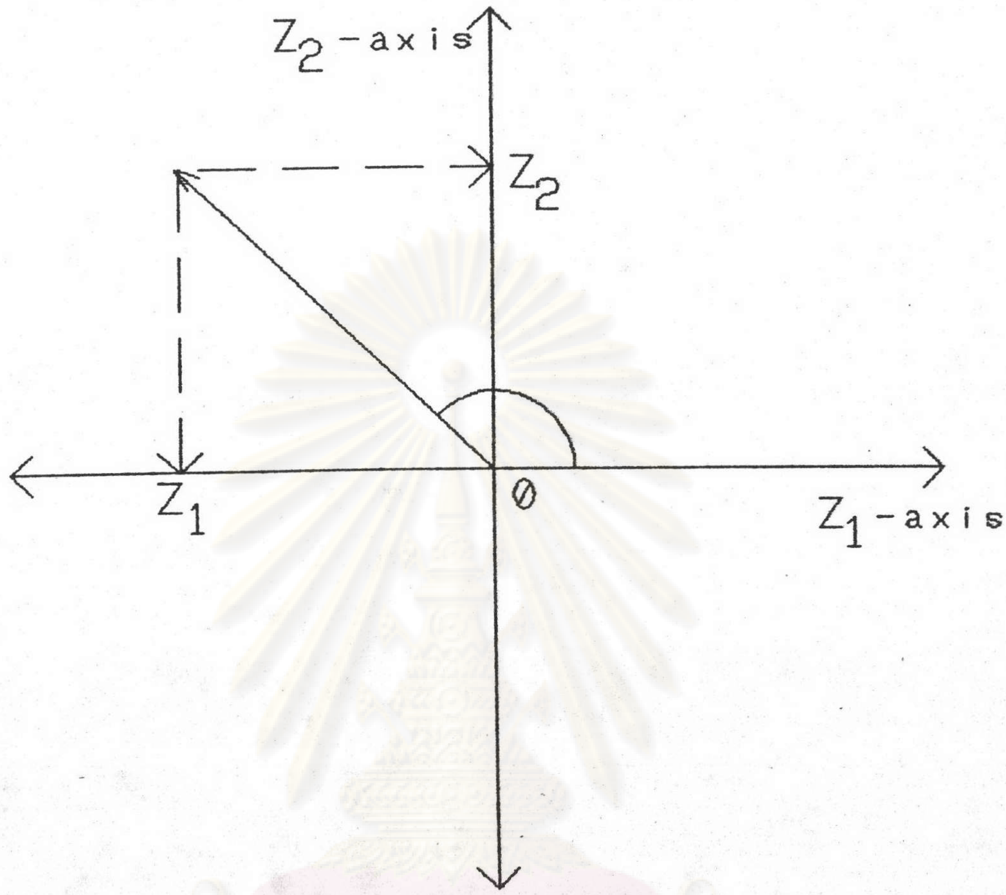

Figure A.6. Polar representation of pair of standard normal variables.

Results of random numbers tests.

Table A.2. Result of Komogorov-Simirnov test.

RESULT

Komogorov-Smirnov test: Test of uniformity

for random numbers generated by

Linear Congruential method

seed	= 50015
number of random numbers	= 600
then critical value	= 0.0555
maximum departure	= 0.0413

MAXIMUM DEPARTURE < CRITICAL VALUE

No difference between the true distribtuion of random numbers and the uniform distribution was detected.

Table A.3. Result of Chi-Square test.

class interval	Oi frequency	Ei expected value	(Oi-Ei)^2/Ei departure
0.00-0.10	71	60	2.017
0.10-0.20	60	60	0.000
0.20-0.30	58	60	0.067
0.30-0.40	68	60	1.067
0.40-0.50	58	60	0.067
0.50-0.60	64	60	0.267
0.60-0.70	54	60	0.600
0.70-0.80	56	60	0.267
0.80-0.90	53	60	0.817
0.90-1.00	58	60	0.067
		total departure =	5.233

Chi-Square test : Test of uniformity

critical value = 16.900

tested value = 5.233

TESTED VALUE < CRITICAL VALUE

The null hypothesis of no difference between the sample distribution and the uniform distribution is not rejected.

Table A.4. Result of Runs up and Runs down.

RESULT

Independence test

Runs up and Runs down

seed	=	50015
number of random numbers	=	600
total number of runs, A	=	392
mean of A	=	399.667
variance of A	=	106.3435
Z(0.025)= critical value	=	1.9600
tested value	=	-0.7434417

-CRITICAL VALUE < TESTED VALUE < CRITICAL VALUE

The hypothesis of independence cannot be rejected on the basis of this test.

Table A.5. Result of Runs above and Runs below the mean.

RESULT

Independence test

Runs above and Runs below the mean

seed	=	50015
number of random numbers	=	600

number of individual observation

above the mean	=	285
below the mean	=	315
total number of runs, B	=	295
mean of B	=	299.75
variance of B	=	149.00
Z(0.025)= critical value	=	1.9600
tested value	=	-0.3891

- CRITICAL VALUE < TESTED VALUE < CRITICAL VALUE

The hypothesis of independence cannot be rejected on the basis of this test.

Table A.6. Result of length of runs (up and down).

Independence test

Runs test: length of runs (up and down)

seed	= 50015
number of random numbers	= 600

Runs length	1	2	over 3
Observed Runs,	240.	109.	43.

number of runs of lengths one

E1 = 250.0833

number of runs of lengths two

E2 = 109.7666

number of runs of lengths = or > three

E345 = 39.8167

mean total number of runs (up and down)

AMEAN = 399.6666

Xsquare(0.05,1)= critical value = 3.8400

tested value = 0.4456015

TESTED VALUE < CRITCAL VALUE

The hypothesis of independence cannot be rejected on the basis of this test.

Table A.7. Result of length of runs (above and below the mean).

Independence test

Runs test: length of runs (above and below the mean)

SEED = 50015

number of random numbers = 600

Run Length	1	2	3	over 3
Observed Runs	140	78	36	41

Xsquare(0.05,1)= critical value = 5.9900

tested value = 0.7436030

TESTED VALUE < CRITCAL VALUE

The hypothesis of independence cannot be rejected

on the basis of this test.

Table A.8. Result of test for Autocorrelation.

Independence test

Test for Autocorrelation

SEED = 50015

number of random numbers = 600

Z(0.025)= critical value = 1.9600

- CRITICAL VALUE < TESTED VALUE < CRITICAL VALUE

The hypothesis of independence cannot be rejected

on the basis of this test.

Table A.9. Result of Gap test.

Independence test

Gap test

SEED	=	50015
number or random numbers	=	610

Digit	Number of gaps
0	56
1	60
2	57
3	67
4	58
5	65
6	53
7	57
8	52
9	58

Table A.9 (continued). Result of Gap test.

Gap length	Freq	Cum Re Freq	F(x)	departure
0- 3	201	0.33	0.3439	0.0089
4- 7	142	0.57	0.5695	0.0021
8-11	92	0.72	0.7176	0.0074
12-15	51	0.81	0.8147	0.0047
16-19	47	0.89	0.8784	0.0099
20-23	22	0.92	0.9202	0.0048
24-27	18	0.95	0.9477	0.0073
28-31	7	0.97	0.9657	0.0010
32-35	8	0.98	0.9775	0.0025
36-39	6	0.99	0.9852	0.0048
40-43	2	0.99	0.9903	0.0030
44-47	3	1.00	0.9936	0.0047
48-51	0	1.00	0.9958	0.0025
52-55	0	1.00	0.9973	0.0011
56-59	1	1.00	0.9982	0.0018

critical value (alpha = 0.05) = 0.0555218

tested value = 0.0099100

TESTED VALUE < CRITCAL VALUE. The hypothesis of independence cannot be rejected on the basis of this test.

Table A.10. Result of Poker test.

Independence test

Poker test

SEED = 50015

number or random numbers = 600

Xsquare(0.05,1)= critical value = 5.9900

Combination i	Observed Qi	Expected Ei	Departure (Qi-Ei)^2/Ei
Three diff	432	432.0	0.00
Three like	3	6.0	1.50
One pair	165	162.0	0.06
		Total =	1.56

TESTED VALUE < CRITCAL VALUE

The hypothesis of independence cannot be rejected on the basis of this test.

APPENDIX B

DEVELOPED COMPUTER PROGRAMS

1. Random number generator subroutine.

```
C     ************************************************
C     *              RANDOM NUMBER GENERATOR         *
C     *           Linear congruential method         *
C     *          Y(i+1) = (A * Y(i) + C) mod m       *
C     ************************************************
C     *    FILE'S NAME    : RAN1.F77                 *
C     *    DATE           : MARCH, 5, 1991           *
C     ************************************************
      DIMENSION RND(2Ø)
      COMMON RN,I,IY,N
C     ------------------------------------------------
C     RND,RN  = random numbers
C     IY,IIY  = seed
C     N       = number of random numbers
C     ------------------------------------------------
C     read seed and number of random numbers
      OPEN(5,FILE='RAN1.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,1Ø)IY,N
   1Ø FORMAT(////,I6,//,I5)
      CLOSE(5)
      OPEN(6,FILE='RAN1.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
C *** call subroutine for generating random numbers
      DO 2Ø I = 1,N
      CALL RAN
      RND(I) = RN
   2Ø CONTINUE
C *** print out set of random numbers
      DO 3Ø J = 1,N
      WRITE(6,4Ø)J,RND(J)
   3Ø CONTINUE
   4Ø FORMAT(2ØX,'RND (',I2,') = ',F11.9)
```

```
      CLOSE(6)
      STOP
      END
C *** ---------------------------------------------
C *** SUBROUTINE FOR GENERATING THE RANDOM NUMBERS
C *** ---------------------------------------------
      SUBROUTINE RAN
      COMMON RN,I,IY,N
      DOUBLE PRECISION HALFM
      DOUBLE PRECISION DATAN,DSQRT
      DATA M2/Ø/,ITWO/2/
      IF (M2.NE.Ø) GO TO 3Ø
      M      = 1
   1Ø M2     = M
      M      = ITWO*M2
      IF (M.GT.M2) GO TO 1Ø
      HALFM = M2
      IA     = 8*IDINT(HALFM*DATAN(1.DØ)/8.DØ) + 5
      IC     = 2*IDINT(HALFM*(Ø.5DØ-DSQRT(3.DØ)/6.DØ)) + 1
      MIC    = (M2-IC) + M2
      S      = Ø.5/HALFM
          WRITE(6,2Ø) IY,HALFM,IA,IC,MIC,S,N
   2Ø FORMAT(1ØX,'-----------------------------------------',/,
     *       1Øx,'       Random number generator          ',/,
     *       1ØX,'      Linear congruential method        ',/,
     *       1ØX,'-----------------------------------------',/,
     *       1ØX,'           SEED :     IY = ',I6          ,/,
     *       1ØX,'MAXIMUM NUMBER : HALFM = ',D12.5         ,/,
     *       1ØX,'                      IA = ',I1Ø         ,/,
     *       1ØX,'                      IC = ',I1Ø         ,/,
     *       1ØX,'                     MIC = ',I1Ø         ,/,
     *       1ØX,'                       S = ',F14.12      ,/,
     *       1ØX,'       NUMBER OF RNDS = ',I5             ,/,
     *       1ØX,'-----------------------------------------')
   3Ø IY = IY*IA
          IF (IY.GT.MIC)  IY = (IY-M2)-M2
       IY = IY+IC
          IF (IY/2.GT.M2) IY = (IY-M2)-M2
          IF (IY.LT.Ø)    IY = (IY+M2)+M2
       RN = FLOAT(IY)*S
       RETURN
       END
```

2. The Kolmogorov-Smirnov test.

```
C     ***********************************************
C     *        FREQUENCY TEST (TEST OF UNIFORMITY)       *
C     *            The Kolmogorov-Smirnov test           *
C     *         for random numbers generated by          *
C     *            Linear Congruential method            *
C     ***********************************************
C     *   FILE'S NAME   : TEST1.F77                      *
C     *   DATE          : MAY, 13, 1991                  *
C     ***********************************************
C     * This program can be used for ALPHA = 0.05 ,N > 35 *
C     ***********************************************
      DIMENSION RND(10000)
      COMMON /L1/RND,N /L2/IIY /L3/RN,IY
C     ---------------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST1.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10)IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
      OPEN(6,FILE='TEST1.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
   20 RND(I) = RN
C
C *** sort the random numbers
C
      CALL SORT
C
C *** calculate maximum departure from uniform distribution
C
      CALL KOLM
      CLOSE(6)
      STOP
      END
```

```
C *** ------------------------------------
C *** SUBROUTINE FOR SORTING RANDOM NUMBERS
C *** ------------------------------------
      SUBROUTINE SORT
      COMMON /L1/RND,N
      DIMENSION RND(10000)
      DO 20  I = 1, N - 1
             SMALL = RND(I)
             K = I
             DO 10 J = I+1,N
                  IF(RND(J).LT.SMALL) THEN
                  SMALL = RND(J)
                  K=J
             END IF
   10 CONTINUE
             RND(K) = RND(I)
             RND(I) = SMALL
   20 CONTINUE
      RETURN
      END
C *** -----------------------------------------------------------
C *** SUBROUTINE FOR
C *** COMPARING CDF OF RND TO CDF OF UNIFORM DISTRIBUTION
C *** ALPHA = 0.05 ,N > 35  ---> CRITICAL VALUE = 1.36/SQRT(N)
C *** -----------------------------------------------------------
      SUBROUTINE KOLM
      COMMON /L1/RND,N /L2/IIY
      DIMENSION RND(10000)
      AN = N
      D = 1.36/SQRT(AN)
      DMAX = 0.
      DO 10 I = 1,N
      AI = I
           D1 = AI/AN - RND(I)
           D2 = RND(I) - (AI - 1)/AN
           IF(DMAX.LT.D1)  DMAX = D1
           IF(DMAX.LT.D2)  DMAX = D2
   10 CONTINUE
           WRITE(6,20)IIY,N,D,DMAX
      IF(D.GE.DMAX)  THEN
           WRITE(6,30)
      ELSE
```

```
          WRITE(6,40)
        END IF
   20 FORMAT(10X,'---------------------------------------------',/,
     *        10X,'                    RESULT                   ',/,
     *        10X,'---------------------------------------------',/,
     *        10X,' Komogorov-Smirnov test : Test of uniformity ',/,
     *        10X,'       for random number generated by        ',/,
     *        10X,'         Linear Congruential method          ',/,
     *        10X,'---------------------------------------------',/,
     *        10X,'                    SEED  = ',I5             ,/,
     *        10X,' number of random number  = ',I5             ,/,
     *        10X,' then critical value : D  = ',F6.4           ,/,
     *        10X,' maximum departure : DMAX = ',F6.4           ,/,
     *        10X,'---------------------------------------------')
   30 FORMAT(10X,'        MAXIMUM DEPARTURE < CRITICAL VALUE    ',/,
     *        10X,'There are no difference has been detected    ',/,
     *        10X,'between the true distribution of random number',/,
     *        10x,'and the uniform distribution')
   40 FORMAT(10X,'        MAXIMUM DEPARTURE > CRITICAL VALUE    ',/,
     *        10X,'The random number generator is rejected !')
        RETURN
        END
```

3. The Chi-Square test.

```
C     ***********************************************
C     *        FREQUENCY TEST (TEST OF UNIFORMITY)   *
C     *              The Chi-Square test             *
C     *        for random numbers generated by       *
C     *          Linear congruential method          *
C     ***********************************************
C     *   FILE'S NAME    : TEST2.F77                  *
C     *   DATE           : MAY, 14, 1991              *
C     ***********************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     -----------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST2.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10)IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 20  I = 1, N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C *** call the subroutine of chi-square test
C
      CALL CHI
      STOP
      END
C *** -------------------------
C *** SUBROUTINE CHI-SQUARE TEST
C *** -------------------------
      SUBROUTINE CHI
      COMMON /L2/RND,N,IIY
      DIMENSION RND(10000),IRND(10000),AO(10),IO(10),T(10)
C     convert real rnd to 4-digit integer (multiple by 10000)
      DO 200  I = 1,N
```

```
      IRND(I) = (RND(I)+0.0000005)*10000
  200 CONTINUE
C
C     number of interval = 10 ---> degree of freedom = 10 - 1 = 9
C     alpha = 0.05  ,So the critical value = 16.9
      CRIT = 16.9
      MM1 = -1000
      MM2 = 0
      TT = 0.0
      AN = N
C     E is the expect value in each class of interval
      E = AN/10.
      IE = E
C     classification
      DO 220 J = 1,10
         IO(J) = 0
           MM1 = MM1 + 1000
           MM2 = MM2 + 1000
                 DO 210 K = 1,N
       IF((IRND(K).GT.MM1).AND.(IRND(K).LE.MM2)) IO(J) = IO(J) + 1
  210            CONTINUE
         AO(J) = IO(J)
          T(J) = ((AO(J)-E)**2)/E
            TT = TT + T(J)
  220 CONTINUE
C *** ------------------------------------------------------------------
C     report the results
      OPEN(6,FILE='TEST2.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
           WRITE(6,240)IIY,N
  240 FORMAT(10X,'SEED = ',I5,/,10X,'NUMBER OF RNDS = ',I5,/,
     +       10X,'----------------------------------------------------',/,
     +       10X,'  class        Oi           Ei        (Oi-Ei)^2/Ei',/,
     *       10X,' interval  frequency  expected value  departure ',/,
     *       10X,'----------------------------------------------------')
      ALO = 0.
      AUP = 0.1
      DO 250 I = 1,10
      WRITE(6,260)ALO,AUP,IO(I),IE,T(I)
      ALO = ALO + 0.1
      AUP = AUP + 0.1
  250 CONTINUE
  260 FORMAT(13X,F4.2,'-',F4.2,T25,I5,T38,I5,T51,F7.3)
```

```
      WRITE(6,270)TT
270 FORMAT(10X,'--------------------------------------------------',/,
   *       10X,'                      total departure =      ',F7.3        )
          WRITE(6,280)CRIT,TT
      IF(TT.LE.CRIT) THEN
          WRITE(6,290)
      ELSE
          WRITE(6,300)
      ENDIF
280 FORMAT(10X,'--------------------------------------------------',/,
   *       10X,'    Chi-Square test : Test of uniformity      ',/,
   *       10X,'     critical value        = ',F7.3            ,/,
   *       10X,'        tested value       = ',F7.3)
290 FORMAT(10X,'--------------------------------------------------',/,
   *       10X,'        TESTED VALUE < CRITICAL VALUE            ',/,
   *       10X,'The null hypothesis of no difference between  ',/,
   *       10X,'the sample distribution and the uniform        ',/,
   *       10X,'distribution is not rejected.')
300 FORMAT(10X,'--------------------------------------------------',/,
   *       10X,'        TESTED VALUE > CRITICAL VALUE            ',/,
   *       10X,'The random number generator is rejected !')
      CLOSE(6)
      RETURN
      END
```

4. Runs up and Runs down test.

```
C     ***********************************************
C     *              INDEPENDENCE TEST              *
C     *            Runs up and Runs down            *
C     *      for random numbers generated by        *
C     *         Linear congruential method          *
C     ***********************************************
C     *   FILE'S NAME      : TEST3.F77              *
C     *   DATE             : MAY, 17, 1991          *
C     ***********************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     -----------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST3.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,1)IY,N
    1 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 2 I = 1,N
      CALL RAN
      RND(I) = RN
    2 CONTINUE
      OPEN(6,FILE='TEST3.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
C
C *** call subroutine runs up and runs down
C
      CALL RUNS
      CLOSE(6)
      STOP
      END
C *** --------------------------------
C *** SUBROUTINE RUNS UP AND RUNS DOWN
C *** --------------------------------
      SUBROUTINE RUNS
      DIMENSION RND(10000)
      INTEGER SIGN(10000)
```

```
      COMMON /L2/RND,N,IIY
C *** two-sided hypothesis testing at alpha = 0.05
C *** critical value Z(0.025) = 1.96
      CRIT = 1.96
      DO 10 I = 1, N-1
         IF(RND(I+1).GT.RND(I)) THEN
            SIGN(I) = 1
         ELSE
            SIGN(I) = 2
         END IF
   10 CONTINUE
      A = 1.
      DO 20 J = 1, N-2
         IF(SIGN(J).NE.SIGN(J+1)) A = A + 1.
   20 CONTINUE
      AN = N
      Z = (A-(2.*AN-1.)/3.)/SQRT((16.*AN-29.)/90.)
          WRITE(6,30)IIY,N,CRIT,Z
                    IF(ABS(Z).LE.CRIT) THEN
          WRITE(6,40)
      ELSE
          WRITE(6,50)
      ENDIF
   30 FORMAT(10X,'---------------------------------------------',/,
     *      10X,'                   RESULT                    ',/,
     *      10X,'---------------------------------------------',/,
     *      10X,'              Independence test              ',/,
     *      10X,'            Runs up and Runs down            ',/,
     *      10X,'       for Linear congruential method        ',/,
     *      10X,'---------------------------------------------',/,
     *      10X,'                  SEED    = ',I5              ,/,
     *      10X,' number of random number = ',I5              ,/,
     *      10X,'Z(0.025)= critical value = ',F7.4            ,/,
     *      10X,'         tested value     = ',F10.7          ,/,
     *      10X,'---------------------------------------------')
   40 FORMAT(10X,'          TESTED VALUE<CRITICAL VALUE          ',/,
     *      10X,'The hypothesis of independence cannot be rejected',/,
     *      10X,'on the basis of this test.')
   50 FORMAT(10X,'          TESTED VALUE>CRITICAL VALUE          ',/,
     *      10X,'The random number generator is rejected !')
      RETURN
      END
```

5. Runs above and Runs below the mean.

```
C     **************************************************
C     *               INDEPENDENCE TEST                *
C     *        Runs above and Runs below the mean      *
C     *        for random numbers generated by         *
C     *           Linear Congruential method           *
C     **************************************************
C     *    FILE'S NAME   : TEST4.F77                   *
C     *    DATE          : MAY, 16, 1991               *
C     **************************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     --------------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST4.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C *** call the subroutine of runs above and below the mean
C
      CALL RUNS
      STOP
      END
C *** ---------------------------------------------
C *** SUBROUTINE FOR RUNS ABOVE AND BELOW THE MEAN
C *** ---------------------------------------------
      SUBROUTINE RUNS
      COMMON /L2/RND,N,IIY
      DIMENSION RND(10000),IA(10000)
C
C     number of RND that is above the mean = n1
```

```
C     number of RND that is below the mean = n2  :  n1 + n2 = N
C
      N1 = Ø
      N2 = Ø
      DO 1Ø J = 1,N
           IF(RND(J).GE.Ø.5) THEN
               N1 = N1 + 1
            IA(J) = 1
           ELSE
               N2 = N2 + 1
            IA(J) = 2
           END IF
   1Ø CONTINUE
C
C *** count number of runs
C
      B = 1.
      DO 2Ø K = 1, N-1
         IF(IA(K+1).NE.IA(K)) B = B + 1.
   2Ø CONTINUE
       AN1 = N1
       AN2 = N2
        AN = N
         U = 2.*AN1*AN2/AN+Ø.5
         D = 2.*AN1*AN2*(2.*AN1*AN2-AN)/AN**2/(AN-1)
         Z = (B-U)/D**Ø.5
      CRIT = 1.96
C
C *** save the result in file test4.out
C
      OPEN(6,FILE='TEST4.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
           WRITE(6,3Ø)IIY,N1,N2,B,N,CRIT,Z
                IF(ABS(Z).LE.CRIT) THEN
           WRITE(6,4Ø)
      ELSE
           WRITE(6,5Ø)
      ENDIF
   3Ø FORMAT(1ØX,'---------------------------------------------',/,
     *       1ØX,'                  RESULT                     ',/,
     *       1ØX,'---------------------------------------------',/,
     *       1ØX,'             Independence test               ',/,
     *       1ØX,'     Runs above and Runs below the mean      ',/,
```

```
*       10X,'                    SEED       = ',I5                    ,/,
*       10X,'                      N1       = ',I7                    ,/,
*       10X,'                      N2       = ',I7                    ,/,
*       10X,'                       B       = ',F8.0                  ,/,
*       10X,'                       N       = ',I7                    ,/,
*       10X,'Z(0.025)= critical value = ',F7.4                        ,/,
*       10X,'          tested value     = ',F7.4                      ,/,
*       10X,'-----------------------------------------------------------')
40 FORMAT(10X,'          TESTED VALUE<CRITICAL VALUE              ',/,
*       10X,'The hypothesis of independence cannot be rejected',/,
*       10X,'on the basis of this test.')
50 FORMAT(10X,'          TESTED VALUE>CRITICAL VALUE              ',/,
*       10X,'The random number generator is rejected !')
   CLOSE(6)
   RETURN
   END
```

6. Runs test : length of runs.

```
C     ***********************************************
C     *               INDEPENDENCE TEST             *
C     *          Runs test : length of runs         *
C     *       for random numbers generated by       *
C     *          Linear Congruential method         *
C     ***********************************************
C     *    FILE'S NAME    : TEST5.F77               *
C     *    DATE           : MAY, 20, 1991           *
C     ***********************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     -----------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST5.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C *** call the subroutine for testing the length of runs
C
      CALL LENGTH
      STOP
      END
C *** ------------------------------------
C *** SUBROUTINE FOR TEST THE LENGTH OF RUN
C *** ------------------------------------
      SUBROUTINE LENGTH
      COMMON /L2/RND,N,IIY
      DIMENSION RND(10000),IRND(10000),IAA(10000)
      DO 9 J = 1,N
    9 IRND(J) = (RND(J)+0.000005)*100000
```

```
      DO 10 J = 1,N-1
           IF(IRND(J+1).GT.IRND(J)) THEN
              IAA(J) = 1
           ELSE
              IAA(J) = 2
           END IF
   10 CONTINUE
      O1 = 0.
      O2 = 0.
      O345 = 0.
C     ---------------------------------------
C     count number of run of range 1
      IF(IAA(1).NE.IAA(2)) O1 = 1.
      DO 20 I = 2,N-2
          IF(IAA(I).EQ.IAA(I-1)) GO TO 20
          IF(IAA(I).NE.IAA(I+1)) O1 = O1 + 1.
   20 CONTINUE
      IF(IAA(N-1).NE.IAA(N-2)) O1 = O1 + 1.
C     ---------------------------------------
C     count number of run of range 2
      IF (IAA(1).NE.IAA(2)) GO TO 30
      IF (IAA(1).NE.IAA(3)) O2 = 1.
   30 DO 40 I = 2,N-3
      IF (IAA(I).EQ.IAA(I-1)) GO TO 40
      IF (IAA(I).NE.IAA(I+1)) GO TO 40
      IF (IAA(I).NE.IAA(I+2)) O2 = O2 + 1.
   40 CONTINUE
      IF (IAA(N-2).EQ.IAA(N-3)) GO TO 50
      IF (IAA(N-2).EQ.IAA(N-1)) O2 = O2 + 1.
C     ---------------------------------------
C     count number of run of range more than 2
   50 IF(IAA(1).NE.IAA(2)) GO TO 60
      IF(IAA(1).EQ.IAA(3)) O345 = 1.
   60 DO 70 I = 2,N-3
      IF(IAA(I).EQ.IAA(I-1)) GO TO 70
      IF(IAA(I).NE.IAA(I+1)) GO TO 70
      IF(IAA(I).EQ.IAA(I+2)) O345 = O345 + 1.
   70 CONTINUE
C     ---------------------------------------
C     calculate all essential values
      AN = N
      AMEAN = (2.*AN-1.)/3.
```

```
      E1 = (1./4./3.)*(5.*AN+1.)
      E2 = (1./5./4./3.)*(11.*AN-14.)
      E345 = AMEAN - E1 - E2
      O2345 = O2 + O345
      E2345 = E2 + E345
      T1 = (O1 - E1)**2/E1
      T2345 = (O2345-E2345)**2/E2345
      TT =  T1 + T2345
C     -------------------------------------
C     critical value Xsquare(Ø.Ø5,1) = 3.84
      CRIT = 3.84
C
C *** save the result in file test5.out
C
      OPEN(6,FILE='TEST5.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
           WRITE(6,8Ø)O1,O2,O345,IIY,N,CRIT,TT
      IF(TT.LE.CRIT) THEN
           WRITE(6,9Ø)
      ELSE
           WRITE(6,1ØØ)
      ENDIF
   8Ø FORMAT(1ØX,'-----------------------------------------------',/,
     *        1ØX,'                Independence test              ',/,
     *        1ØX,'   Runs test : length of runs (up and down)    ',/,
     *        1ØX,'-----------------------------------------------',//,
     *        1ØX,' Runs length ',T3Ø,'1',T37,'2',T44,'over 3'    ,/,
     *        1ØX,' Observed Runs,'T3Ø,F5.Ø,T37,F5.Ø,T44,F5.Ø     ,/,
     *        1ØX,'                       SEED        = ',I5       ,/,
     *        1ØX,'               number or rnds      = ',I5       ,/,
     *        1ØX,'Xsquare(Ø.Ø5,1)= critical value = ',F7.4       ,/,
     *        1ØX,'                tested value       = ',F1Ø.7    ,/,
     *        1ØX,'-----------------------------------------------')
   9Ø FORMAT(1ØX,'          TESTED VALUE < CRITCAL VALUE            ',/,
     *        1ØX,'The hypothesis of independence cannot be rejected',/,
     *        1ØX,'on the basis of this test.')
  1ØØ FORMAT(1ØX,'          TESTED VALUE > CRITCAL VALUE            ',/,
     *        1ØX,'The random number generator is rejected !')
      CLOSE(6)
      STOP
      END
```

7. Runs test : length of runs (above and below the mean).

```
C     ******************************************************
C     *                 INDEPENDENCE TEST                  *
C     * Runs test : length of runs (above and below the mean) *
C     *          for random numbers generated by           *
C     *             Linear Congruential method             *
C     ******************************************************
C     *       FILE'S NAME    : TEST6.F77                   *
C     *       DATE           : JUNE, 12, 1991              *
C     ******************************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST6.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C *** call the subroutine for testing the length of runs
C
      CALL LENGTH
      CLOSE(6)
      STOP
      END
C *** ---------------------------------------------------------------
C *** SUBROUTINE COMPUTE THE LENGTH OF RUN ABOVE AND BELOW THE MEAN
C *** ---------------------------------------------------------------
      SUBROUTINE LENGTH
      COMMON /L2/ RND,N,IIY
      DIMENSION RND(10000),IRND(10000),IAA(10000)
C
```

```
C     AM is number of random numbers above the mean
C     BM is number of random numbers below the mean
C
      AM = Ø.
      BM = Ø.
      DO 1ØØ J = 1,N
      IRND(J) = (RND(J)+Ø.ØØØØØ5)*1ØØØØ
            IF(IRND(J).GT.5ØØØ) THEN
               IAA(J) = 1
               AM = AM + 1.
            ELSE
               IAA(J) = 2
               BM = BM + 1.
            END IF
  1ØØ CONTINUE
C     --------------------------------------
      O1 = Ø.
      O2 = Ø.
      O3 = Ø.
      O456 = Ø.
C
C     count number of run of range 1
C
            IF(IAA(1).NE.IAA(2))   O1 = 1.
      DO 11Ø J = 2,N-1
            IF(IAA(J).EQ.IAA(J-1)) GO TO 11Ø
            IF(IAA(J).NE.IAA(J+1)) O1 = O1 + 1.
  11Ø CONTINUE
            IF(IAA(N).NE.IAA(N-1)) O1 = O1 + 1.
C     --------------------------------------
C     count number of run of range 2
            IF (IAA(1).NE.IAA(2))      GO TO 2ØØ
            IF (IAA(1).NE.IAA(3))      O2 = 1.
  2ØØ DO 21Ø I = 2,N-2
            IF (IAA(I).EQ.IAA(I-1))    GO TO 21Ø
            IF (IAA(I).NE.IAA(I+1))    GO TO 21Ø
            IF (IAA(I).NE.IAA(I+2))    O2 = O2 + 1.
  21Ø CONTINUE
            IF (IAA(N-1).EQ.IAA(N-2)) GO TO 22Ø
            IF (IAA(N-1).EQ.IAA(N))    O2 = O2 + 1.
C     --------------------------------------
C     count number of run of range 3
```

```
220     IF (IAA(1).NE.IAA(2))       GO TO 222
        IF (IAA(1).NE.IAA(3))       GO TO 222
        IF (IAA(1).NE.IAA(4))       O3 = 1.
222 DO 224 K = 2,N-3
        IF (IAA(K).EQ.IAA(K-1))     GO TO 224
        IF (IAA(K).NE.IAA(K+1))     GO TO 224
        IF (IAA(K).NE.IAA(K+2))     GO TO 224
        IF (IAA(K).NE.IAA(K+3))     O3 = O3 + 1.
224 CONTINUE
        IF (IAA(N-2).EQ.IAA(N-3)) GO TO 226
        IF (IAA(N-2).NE.IAA(N-1)) GO TO 226
        IF (IAA(N-2).EQ.IAA(N))     O3 = O3 + 1.
C     --------------------------------------
C     count number of run of range more than 3
226     IF(IAA(1).NE.IAA(2))     GO TO 230
        IF(IAA(1).NE.IAA(3))     GO TO 230
        IF(IAA(1).EQ.IAA(4))     O456 = 1.
230 DO 240 K = 2,N-3
        IF(IAA(K).EQ.IAA(K-1)) GO TO 240
        IF(IAA(K).NE.IAA(K+1)) GO TO 240
        IF(IAA(K).NE.IAA(K+2)) GO TO 240
        IF(IAA(K).EQ.IAA(K+3)) O456 = O456 + 1.
240 CONTINUE
C     -------------------------------------------
C     calculate essential values for chi-square test
C
      IO1 = O1
      IO2 = O2
      IO3 = O3
      IO456 = O456
      AN = N
      W1 = 2*AM/AN*BM/AN
      W2 = (AM/AN)**2*BM/AN + (BM/AN)**2*AM/AN
      EI = AM/BM + BM/AM
      EA = AN/EI
      E1 = AN*W1/EI
      E2 = AN*W2/EI
      E3456 = EA - E1 - E2
      O3456 = O3 + O456
      T1 = (O1 - E1)**2/E1
      T2 = (O2 - E2)**2/E2
      T3456 = (O3456-E3456)**2/E3456
```

```
      TT =  T1 + T2 + T3456
C     ------------------------------------
C     report the results
C
      OPEN(6,FILE='TEST6.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
C     degrees of freedom equals the number of class intervals minus one.
C     X^2(Ø.Ø5,2) = 5.99
      CRIT = 5.99
          WRITE(6,32Ø)IIY,N,IO1,IO2,IO3,IO456,CRIT,TT
                 IF(TT.LE.CRIT) THEN
          WRITE(6,33Ø)
      ELSE
          WRITE(6,34Ø)
      ENDIF
  32Ø FORMAT(
    *7X,'-----------------------------------------------------',/,
    *7X,'                Independence test                    ',/,
    *7X,'Runs test : length of runs (above and below the mean)',/,
    *7X,'                   SEED       = ',I6                  ,/,
    *7X,'           number of rnds     = ',I6                  ,/,
    *7X,'-----------------------------------------------------',/,
    *7X,'  Run Length          1          2          3       over 3',/,
    *7X,'  Observed Runs',T26,I4,T36,I4,T47,I4,T59,I4,/,
    *7X,'-----------------------------------------------------',/,
    *7X,'Xsquare(Ø.Ø5,1)= critical value = ',F7.4             ,/,
    *7X,'            tested value      = ',F1Ø.7              ,/,
    *7X,'-----------------------------------------------------',/)
  33Ø FORMAT(1ØX,'        TESTED VALUE < CRITCAL VALUE         ',/,
    *       1ØX,'The hypothesis of independence cannot be rejected',/,
    *       1ØX,'on the basis of this test.',/)
  34Ø FORMAT(1ØX,'        TESTED VALUE > CRITCAL VALUE         ',/,
    *       1ØX,'The random number generator is rejected !',/)
      RETURN
      END
```

8. Test for Autocorrelation.

```
C     ************************************************
C     *              INDEPENDENCE TEST                *
C     *          Test for Autocorrelation             *
C     *      for random numbers generated by          *
C     *         Linear congruential method            *
C     ************************************************
C     *    FILE'S NAME   : TEST7.F77                  *
C     *    DATE          : MAY, 21, 1991              *
C     ************************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     ------------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST7.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C     call the subroutine for generating random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C     call the subroutine for testing the Autocorrelation
C
      CALL AUTO
      STOP
      END
C *** -----------------------------------------
C     SUBROUTINE FOR TESTING THE AUTOCORRELATION
C *** -----------------------------------------
      SUBROUTINE AUTO
      COMMON /L2/ RND,N,IIY
      DIMENSION RND(10000)
C
C     two-sided hypothesis testing at alpha = 0.05
```

```
C     critical value Z(0.025) = 1.96
C
      CRIT = 1.96
      OPEN(6,FILE='TEST7.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      WRITE(6,60)IIY,N,CRIT
C
C     I = number of the beginning
C     M = frequency of the correlation
C
      DO 40 I = 1,20
           DO 30 M = 2,10
           IBIGM = (N-I)/M-1
           SUM = 0.
C
C                calculate rho(im)=RHO
C
                 DO 20 K = 0 ,IBIGM
                 ISUB1 = I + K*M
                 ISUB2 = I + (K+1)*M
                 SUM = SUM + RND(ISUB1)*RND(ISUB2)
   20            CONTINUE
           ABIGM = IBIGM
           RHO = (1./(ABIGM+1.))*SUM - 0.25
C
C          calculate standard deviation
C
           DEL = SQRT(13.*ABIGM+7.)/(12.*(ABIGM+1.))
           Z0 = RHO/DEL
C
C     print out the results
C
          WRITE(6,70)I,M,Z0
      IF(ABS(Z0).GT.CRIT) THEN
          IDUMMY = 1
      ELSE
      END IF
   30     CONTINUE
   40 CONTINUE
      IF(IDUMMY.NE.1) THEN
          WRITE(6,80)
      ELSE
          WRITE(6,90)
```

```
      END IF
60 FORMAT(10X,'--------------------------------------------',/,
  *       10X,'               Independence test          ',/,
  *       10X,'           Test for Autocorrelation       ',/,
  *       10X,'--------------------------------------------',/,
  *       10X,'                         SEED = ',I5      ,/,
  *       10X,'number of random numbers = ',I5         ,/,
  *       10X,'Z(0.025)= critical value = ',F7.4       ,/,
  *     6X,'-------------------------------------------------')
70 FORMAT(
  *5X,' begin at no. ',I3,' frequency =',I3,' tested value =',F7.4)
80 FORMAT(10X,'        TESTED VALUE < CRITICAL VALUE        ',/,
  *       10X,'The hypothesis of independence cannot be rejected',/,
  *       10X,'on the basis of this test.')
90 FORMAT(
  *   6X,'-------------------------------------------------',/,
  *       10X,'        TESTED VALUE > CRITICAL VALUE        ',/,
  *       10X,'The random number generator is rejected !')
      CLOSE(6)
      RETURN
      END
```

9. Gap test.

```
C     ***********************************************
C     *                INDEPENDENCE TEST            *
C     *                    Gap test                 *
C     *        for random numbers generated by      *
C     *          Linear Congruential method         *
C     ***********************************************
C     *   FILE'S NAME    : TEST8.F77                *
C     *   DATE           : JUNE, 13, 1991           *
C     ***********************************************
      DIMENSION RND(10010)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C     -----------------------------------------------
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST8.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
      N = N + 10
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C     call the subroutine gap test
C
      CALL GAP
C
      STOP
      END
C *** --------------------
C *** SUBROUTINE GAP TEST
C *** --------------------
      SUBROUTINE GAP
      COMMON /L2/RND,N,IIY
```

```
      DIMENSION RND(10010),IR(10010),IG(10010),NGAP(10000),
     *DEP(15),IFR(15),C(15),F(15),CF(15)
C
C
C     save the output in test8.out
      OPEN(6,FILE='TEST8.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
C
C     find the first digit of rnds; IR(I)
C
      DO 100 I = 1,N
  100 IR(I) = (RND(I)+0.000005)*10
C
C     ngap(i) are the number of gap of digit i (SUM ngap(i) = N - 10)
C
      DO 110 I = 0, 9
  110 NGAP(I) = -1
C
      DO 120 I = 0, 9
            DO 115 J = 1,N
            IF(IR(J).EQ.I) NGAP(I) = NGAP(I) + 1
  115       CONTINUE
  120 CONTINUE
C
C     find gap length of each gap
C
      L = 0
      DO 200 I = 0 ,9
C
C               find the position of the frist digit
                DO 170 J = 1, N
                     IF(IR(J).EQ.I) THEN
                          IP = J
                          GO TO 180
                     ELSE
                     ENDIF
  170           CONTINUE
C
C               count the length of the gap(i)
  180           DO 190 K = IP + 1 ,N
                     IF(IR(IP).EQ.IR(K)) THEN
                          L = L+1
                          IG(L) = K - IP - 1
```

```
                              IP = K
                              GO TO 195
                         ELSE
                         ENDIF
  190               CONTINUE
  195 IF(K.LT.N) GO TO 180
  200 CONTINUE
C
C     classify the gap lengths
C
C     There are 15 classes.
C     G(i) are gap that be classed by their length
C     LE = upper limit of each class
C     IFR(i) = frequency in class i
C     C(i) = cumulative frequency in class i
      LOWER = -4
      DO 205 I = 1,15
      IFR(I) = 0
      LOWER  = LOWER + 4
      IUPPER = LOWER + 3
            DO 204 J = 1,N - 10
      IF((IG(J).GE.LOWER).AND.(IG(J).LE.IUPPER)) IFR(I) = IFR(I) + 1
  204 CONTINUE
  205 CONTINUE
      LE = 3
      ANG = N-10
      DO 220 I = 1, 15
      C(I) = 0.
           DO 210 J = 1, N-10
           IF(IG(J).LE.LE) C(I) = C(I) + 1.
  210      CONTINUE
C     F(i)  = the theoretical frequency distribution
      ALE   = LE
      F(I)  = 1. - 0.9**(ALE+1.)
C     CF(i) = cumulative relative frequency
      CF(I) = C(I) / ANG
      LE = LE + 4
  220 CONTINUE
      DMAX = 0.
      DO 230 I = 1,15
           DEP(I) = ABS(F(I) - CF(I))
           IF(DMAX.LE.DEP(I)) DMAX = DEP(I)
```

```
  230 CONTINUE
C     ---------------------------------------
C     calculate the critical value
C     Kolmogorov-smirnov test at alpha = 0.05
      CRIT = 1.36/SQRT(ANG)
C
C     ---------------------------------------
C     print out the results
          WRITE(6,250)IIY,N,CRIT,DMAX
      IF(DMAX.LE.CRIT) THEN
          WRITE(6,260)
      ELSE
          WRITE(6,270)
      ENDIF
  250 FORMAT(10X,'--------------------------------------------',/,
     *       10X,'                Independence test           ',/,
     *       10X,'                    Gap test                ',/,
     *       10X,'--------------------------------------------',/,
     *       10X,'                        SEED      = ',I5      ,/,
     *       10X,'             number or rnds       = ',I5      ,/,
     *       10X,'  critical value (alpha = 0.05)  = ',F10.7   ,/,
     *       10X,'             tested value         = ',F10.7   ,/,
     *       10X,'--------------------------------------------')
  260 FORMAT(10X,'        TESTED VALUE < CRITCAL VALUE        ',/,
     *       10X,'The hypothesis of independence cannot be rejected',/,
     *       10X,'on the basis of this test.',/)
  270 FORMAT(10X,'        TESTED VALUE > CRITCAL VALUE        ',/,
     *       10X,'The random number generator is rejected !',/)
      WRITE(6,300)
      DO 280 I = 0,9
      WRITE(6,310) I,NGAP(I)
  280 CONTINUE
      WRITE(6,320)
      WRITE(6,330)
      LO = -4
      DO 290 I = 1,15
          LO = LO + 4
          IUP = LO + 3
          WRITE(6,340)LO,IUP,IFR(I),CF(I),F(I),DEP(I)
  290 CONTINUE
  300 FORMAT(15X,'-----------------------------------',/,
     *       15X,'     Digit       Number of gaps    ',/,
```

```
     *        15X,'-----------------------------------')
 310 FORMAT(22X,I1,T36,I5)
 320 FORMAT(15X,'-----------------------------------',/)
 330 FORMAT(7X,'------------------------------------------------',/,
     *        7X,'Gap length  Freq  Cum Re Freq    F(x)   departure',/,
     *        7X,'------------------------------------------------')
 340 FORMAT(8X,I2,'-',I2,T23,I4,T32,F4.2,T41,F6.4,T51,F6.4)
     WRITE(6,350)
 350 FORMAT(7X,'------------------------------------------------')
     CLOSE(6)
     RETURN
     END
```

10. Poker test.

```
C     ************************************************
C     *                INDEPENDENCE TEST            *
C     *                   Poker test                *
C     *     for random numbers generated by         *
C     *        Linear Congruential method           *
C     ************************************************
C     *    FILE'S NAME    : TEST9.F77               *
C     *    DATE           : JUNE, 13, 1991          *
C     ************************************************
      DIMENSION RND(10000)
      COMMON /L1/RN,IY /L2/RND,N,IIY
C
C *** read seed and number of random numbers
C
      OPEN(5,FILE='TEST9.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10) IY,N
   10 FORMAT(////,I5,//,I5)
      CLOSE(5)
      IIY = IY
C
C *** generate random numbers
C
      DO 20 I = 1,N
      CALL RAN
      RND(I) = RN
   20 CONTINUE
C
C     call the subroutine poker test
C
      CALL POKER
      STOP
      END
C *** ---------------------
C *** SUBROUTINE POKER TEST
C *** ---------------------
      SUBROUTINE POKER
      COMMON /L2/ RND,N,IIY
      DIMENSION RND(10000),IR1(1000),IR2(10000),IR3(10000)
C
C     find the first three digit of rnds
```

```
C
      DO 11Ø I = 1, N
      IR1(I) = (RND(I) + Ø.ØØØØØ5)*1Ø
      IR2(I) = (RND(I) + Ø.ØØØØØ5)*1ØØ
      IR3(I) = (RND(I) + Ø.ØØØØØ5)*1ØØØ
      IR3(I) = IR3(I)-IR2(I)*1Ø
      IR2(I) = IR2(I)-IR1(I)*1Ø
  11Ø CONTINUE
C
C     find the number of rnds the has same digit in the first
C     three digit of them
C     Let  Q1 =  number of rnds that has all different three digit
C     Let  Q2 =  number of rnds that has all same digit
C     Let  Q3 =  number of rnds that has one pair same digit
C
      Q1 = Ø.
      Q2 = Ø.
      Q3 = Ø.
      DO 2ØØ I = 1,N
      IF((IR1(I).NE.IR2(I)).AND.(IR1(I).NE.IR3(I)).AND.
     *(IR2(I).NE.IR3(I)))THEN
             Q1 = Q1 + 1.
      ELSE
             IF((IR1(I).EQ.IR2(I)).AND.(IR1(I).EQ.IR3(I))) THEN
             Q2 = Q2 + 1.
      ELSE
             Q3 = Q3 + 1.
      ENDIF
      ENDIF
  2ØØ CONTINUE
C
C     calculate essential values
C
C     The degrees of freedom equals the number of class intervals minus
C     one. X^2(Ø.Ø5,2) = 5.99
      CRIT = 5.99
      AN = N
      E1 = Ø.72 * AN
      E2 = Ø.Ø1 * AN
      E3 = Ø.27 * AN
      DEP1 = (Q1-E1)**2/E1
      DEP2 = (Q2-E2)**2/E2
```

```
      DEP3 = (Q3-E3)**2/E3
      DEP = DEP1 + DEP2 + DEP3
C
C *** save the result in file test9.out
C
      OPEN(6,FILE='TEST9.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      IQ1 = Q1
      IQ2 = Q2
      IQ3 = Q3
      WRITE(6,280)IIY,N,CRIT,IQ1,E1,DEP1,IQ2,E2,DEP2,IQ3,E3,DEP3,DEP
      IF(DEP.LE.CRIT) THEN
          WRITE(6,290)
      ELSE
          WRITE(6,300)
      ENDIF
  280 FORMAT(10X,'-----------------------------------------------',/,
     *       10X,'              Independence test                ',/,
     *       10X,'                 Poker test                    ',/,
     *       10X,'-----------------------------------------------',/,
     *       10X,'                    SEED        = ',I5         ,/,
     *       10X,'            number or rnds      = ',I5         ,/,
     *       10X,'Xsquare(0.05,1)= critical value = ',F7.4       ,//,
     *       10X,'Combination  Observed    Expected   Departure',/,
     *       10X,'     i          Qi          Ei    (Qi-Ei)^2/Ei',/,
     *       10X,'-----------------------------------------------',/,
     *       10X,'Three diff',T24,I4,T37,F6.1,T50,F6.2,/,
     *       10X,'Three like',T24,I4,T37,F6.1,T50,F6.2,/,
     *       10X,'One pair  ',T24,I4,T37,F6.1,T50,F6.2,/,
     *       10X,'-----------------------------------------------',/,
     *       10X,'                           Total = ',T50,F6.2,/,
     *       10X,'-----------------------------------------------')
  290 FORMAT(10X,'      TESTED VALUE < CRITCAL VALUE          ',/,
     *       10X,'The hypothesis of independence cannot be rejected',/,
     *       10X,'on the basis of this test.')
  300 FORMAT(10X,'      TESTED VALUE > CRITCAL VALUE          ',/,
     *       10X,'The random number generator is rejected !')
      CLOSE(6)
      RETURN
      END
```

11. The turning band method with covariance function calculation.

```
C ******************************************************************
C *  Program for simulating random variable in 2-dimensional space *
C *          with spatial correlation (exponential scheme)         *
C *              using spectral turning band method.               *
C ******************************************************************
      COMMON
     */L1/RN
     */L2/M2,IY,IA,IC,MIC,SRAN
     */L3/DC,DL,NCO,NLI
     */L4/AOMEGA,HARNO,DIV
      INTEGER HARNO
      REAL LENGTH
      DOUBLE PRECISION AOMEGA,DIV
      OPEN(5,FILE='TURN.DAT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10)AOMEGA,HARNO,DIV,LENGTH,DC,DL,NCO,NLI
   10 FORMAT(D12.5,/,I5,/,D12.5,/,F12.5,/,F12.5,/,F12.5,/,I5,/,I5)
      CLOSE(5)
      OPEN(6,FILE='TURN.OUT',FORM='FORMATTED',ACCESS='SEQUENTIAL')
      IY = 50015
      M2 = 0
      WRITE(6,20)AOMEGA,HARNO
   20 FORMAT(10X,'      AOMEGA = ',D12.5,/,
     *       10X,'       HARNO = ',I7,/)
C ***
      CALL TURN(LENGTH)
      CLOSE(6)
      STOP
      END
C     ----------------------------------------------------------
C            SUBROUTINE 2-DIMENSIONAL TURNING BAND METHOD
C     ----------------------------------------------------------
      SUBROUTINE TURN(LENGTH)
      COMMON
     */L1/RN
     */L2/M2,IY,IA,IC,MIC,SRAN
     */L3/DC,DL,NCO,NLI
     */L4/AOMEGA,HARNO,DIV
      INTEGER HARNO
      REAL LENGTH
      DIMENSION Y(70)
```

```
      DOUBLE PRECISION W(15Ø),S1(15Ø),WKP(15Ø),PHI(15Ø)
      DOUBLE PRECISION XX(2),G(28),ZI(16,8ØØ), SDW
      DOUBLE PRECISION PI, SX(16,2), DW, DWP, B, AOMEGA
      DOUBLE PRECISION UN,VARIAN,DIST,ANG,ANG1,DIV
C ***
      PI = 3.141592654DØ
      ANG = Ø.DØ
      ANG1 = PI/16.DØ
      DO 1Ø I = 1, 16
      SX(I,1) = SIN(ANG)
      SX(I,2) = COS(ANG)
   1Ø ANG = ANG + ANG1
C ***
      UN = AMIN1(Ø.9*DC,Ø.9*DL,Ø.Ø2*LENGTH)
      DL = DL/UN
      DC = DC/UN
      NGMAX = SQRT(NLI*NLI*DL*DL+NCO*NCO*DC*DC) + 4
      WRITE(*,*)'NGMAX',NGMAX
      DO 2Ø I = 1, 28
   2Ø G(I) = Ø.
      AOMEGA = AOMEGA/LENGTH
      DW = AOMEGA/FLOAT(HARNO)
      DWP = DW/DIV
      VARIAN = 1.DØ
      B = 1.DØ/LENGTH
      DDW = DW*VARIAN/2.DØ
      DO 3Ø K = 1, HARNO
      W(K) = (FLOAT(K)-Ø.5DØ)*DW
   3Ø S1(K) = SQRT(DDW*W(K)/B/B/
     *((1.DØ+(W(K)/B)**2.DØ)**(3.DØ/2.DØ)))
C
C *** uni-dimensional simulation
C
      CALL RAN
      SDW = RN*DWP + (-DWP/2.DØ)
          DO 6Ø K = 1, HARNO
                    WKP(K) = W(K) + SDW
   6Ø CONTINUE
      DO 9Ø ID = 1, 16
          DO 7Ø K = 1, HARNO
                    CALL RAN
                    PHI(K) = RN * 2.DØ * PI
```

```
   70 CONTINUE
      DO 90 J = 1, NGMAX
      DIST = FLOAT(J)*UN
      ZI(ID,J) = 0.D0
          DO 80 K = 1, HARNO
   80 ZI(ID,J) = ZI(ID,J)+ S1(K)*COS(WKP(K)*DIST + PHI(K))
      ZI(ID,J) = ZI(ID,J)*2.D0
   90 CONTINUE
C
C *** two-dimensional simulation
C
      N1 = (NLI+1)/2
      N2 = (NCO+1)/2
      NG = NGMAX/2
      DO 260 I = 1, NCO
      XX(2) = -0.5+(I-N2)*DC
      DO 140 J = 1, NLI
      XX(1) = -0.5+(J-N1)*DL
      Y(J) = 0.
      DO 130 II = 1, 16
      XI = 0.
      DO 120 JJ = 1, 2
  120 XI = XI + XX(JJ)*SX(II,JJ)
      XI = XI + 0.5
      LK = XI
      IF(XI.LT.0.) LK = LK - 1
      LK = LK + NG
      Y(J) = Y(J) + ZI(II,LK)
  130 CONTINUE
      Y(J) = Y(J)/4.D0
  140 CONTINUE
C ***
      KV = 0
      DO 250 IN = 1, 2
      GO TO (210,220),IN
  210 I1 = 1
      I2 = 19
      IP = 1
      GO TO 230
  220 I1 = 20
      I2 = 100
      IP = 10
```

```
230 DO 250 K = I1, I2, IP
    KL = NLI - K
    IF(KL.LE.0) GO TO 260
    KV = KV + 1
    DO 240 II = 1, KL
    T = Y(II) - Y(II+K)
240 G(KV) = G(KV) + T*T
250 CONTINUE
260 CONTINUE
    WRITE(6,280)
    DO 270 K = 1, KV
    NC = NCO*(NLI-K)
    G(K) =  VARIAN - 0.5*G(K)/NC
    D = K * UN * DL
    IF(K.GT.20) D = UN*DL*(10*K-180)
    WRITE(6,290)K,D,NC,G(K)
270 CONTINUE
C
280 FORMAT(//1H ,'COVARIANCE FUNCTION ALONG COLUMNS'
   1/1H ,'---------- -------- ----- -------'/1H ,'LAG ',
   2'DISTANCE N.PAIRS  ')
290 FORMAT(1H ,I3,1X,F7.2,3X,I6,2X,F6.3)
    RETURN
    END
```

12. Computer program for oil reserve calculation using Monte Carlo simulation.

```
C ***************************************************************
C *                                                             *
C *      Oil reserve calculation using Monte Carlo simulation   *
C *                                                             *
C ***************************************************************
      CHARACTER TYPE(8)*4,NAME(8)*21,SPCOR(8)*2,MUCOR*2,IP*4,OP*4
      REAL MAX(8),MIN(8),MODE(8),MEAN(8),STDDEV(8),DETEM(8),LENGTH(8)
      DOUBLE PRECISION PI,SX(16,2),DWP,S1(100),W(100)
      DIMENSION VAR(8,60,60),AVAR(8),TRES(1000),
     *          Y(60,60),AMU(3)
      COMMON
     */L1/RN
     */L2/M2,IY,IA,IC,MIC,SRAN
     */L3/INOR,RN1,RN2
     */L4/DC,DL,NCO,NLI
     */L5/Y,PI,SX,DWP,S1,W
C *** ---------------------------------------------------------------
C                 APB = size of area per block (acre)
C                  DC = length of area per block (ft.)
C                  DL = width of area per block (ft.)
C                  NB = number of blocks (NB = NLI*NCO)
C                 NCO = number of blocks in a column
C                 NLI = number of blocks in a line
C              NOLOOP = number of simulations
C               RATIO = ratio of length to width of area
C              LENGTH = correlation length (ft.)
C                TRES = total oil reserve (barrel)
C        VAR(8,NCO,NLI) = realizations in each block
C *** ---------------------------------------------------------------
C ***                      DATA INPUT SECTION
C *** ---------------------------------------------------------------
      WRITE(*,*)'INPUT FILE ,(****)'
      READ(*,*)IP
      WRITE(*,*)'OUTPUT FILE ,(****)'
      READ(*,*)OP
      OPEN(5,FILE=IP,FORM='FORMATTED',ACCESS='SEQUENTIAL')
      READ(5,10)NOLOOP,NCO,NLI,RATIO
   10 FORMAT(/////////////////////////////////////////,
     *T7,I4,///,T7,I2,///,T7,I2,///,T7,F12.5,///)
```

```
      DO 20 I = 1, 8
         READ(5,40)NAME(I),TYPE(I)
         IF (TYPE(I).EQ.'TRIA') READ(5,50)MAX(I),MIN(I),MODE(I)
         IF (TYPE(I).EQ.'NORM') READ(5,60)MEAN(I),STDDEV(I)
         IF (TYPE(I).EQ.'LOGN') READ(5,60)MEAN(I),STDDEV(I)
         IF (TYPE(I).EQ.'UNIF') READ(5,60)MAX(I),MIN(I)
         IF (TYPE(I).EQ.'DETM') READ(5,70)DETEM(I)
         IF ((I.EQ.1).OR.(I.EQ.2)) GOTO 20
         READ(5,80) SPCOR(I), LENGTH(I)
   20 CONTINUE
      READ(5,90) MUCOR
      IF(MUCOR.EQ.'NO') GOTO 105
      DO 30 I = 1, 3
   30    READ(5,100) AMU(I)
      READ(5,103) RNDP
   40 FORMAT(/,T7,A21,/,T7,A4)
   50 FORMAT(/,T7,F12.5,T20,F12.5,T35,F12.5)
   60 FORMAT(/,T7,F12.5,T20,F12.5)
   70 FORMAT(/,T7,F12.5)
   80 FORMAT(/,T7,A2,T20,F12.5)
   90 FORMAT(//,T7,A2,///)
  100 FORMAT(T20,F12.5)
  103 FORMAT(T23,F7.3)
  105 CLOSE(5)
      OPEN(6,FILE=OP,FORM='FORMATTED',ACCESS='SEQUENTIAL')
      WRITE(6,*)'o/p file = ',OP
C     ------------------------------------------------------------
C                          INITIALIZATION
C     ------------------------------------------------------------
C
C     assign parameters for random number generator
C
      IY = 50015
      M2 = 0
C
C     assign inor for normal variate generation subroutine
C
      INOR = 0
C
C     calculate number of blocks
C
      NB = NCO*NLI
```

```
      ANCO = NCO
      ANLI = NLI
      ANB = NB
C
C     ------------------------------------------------------------------
C                         DETERMINE REALIZATIONS
C     ------------------------------------------------------------------
C
C     calculate deterministic values
C
      DO 11Ø I = 1, 8
          IF((I.EQ.8).AND.(MUCOR.NE.'NO')) GOTO 11Ø
          IF(TYPE(I).NE.'DETM') GOTO 11Ø
             AVAR(I) = DETEM(I)
          IF(I.EQ.6) AVAR(I) = 1./AVAR(I)
          IF(I.EQ.8) AVAR(I) = 1.-AVAR(I)
  11Ø CONTINUE
C
C
C
      DO 28Ø II = 1, NOLOOP
         TRES(II) = Ø.
C
C     calculate value of area and recovery factor
C
          DO 12Ø I = 1, 2
      IF (TYPE(I).EQ.'DETM') GOTO 12Ø
      IF((TYPE(I).EQ.'NORM').OR.(TYPE(I).EQ.'LOGN')) GOTO 115
         CALL RAN
      IF (TYPE(I).EQ.'TRIA') CALL TRI(MAX(I),MIN(I),MODE(I),AVAR(I))
      IF (TYPE(I).EQ.'UNIF') CALL UNI(MAX(I),MIN(I),AVAR(I))
         GOTO 12Ø
  115 CALL NOR(MEAN(I),STDDEV(I),AVAR(I))
      IF (TYPE(I).EQ.'LOGN') AVAR(I) = EXP(AVAR(I))
  12Ø CONTINUE
C
C     calculate area per block ,  APB (acre)
C     calculate width per block ,  DL (ft.)
C     calculate length per block , DC (ft.)
C
        APB = AVAR(1)/ANB
        AFOOT = AVAR(1)*4356Ø.
```

```
      DC = SQRT(RATIO*AFOOT/ANCO/ANCO)
      DL = AFOOT/ANLI/DC/ANCO
C
C
C
      DO 190 I = 3, 8
      IF((I.EQ.8).AND.(MUCOR.NE.'NO')) GOTO 190
      IF(SPCOR(I).EQ.'NO') GOTO 190
      II1 = II
C
          CALL TURN(LENGTH(I),II1)
C
      IF(TYPE(I).EQ.'NORM') THEN
              DO 130 J = 1, NCO
              DO 130 K = 1, NLI
 130  VAR(I,J,K) = MEAN(I) + STDDEV(I)*Y(J,K)
              GOTO 160
      ELSE
      ENDIF
C
      IF(TYPE(I).EQ.'LOGN') THEN
              DO 140 J = 1, NCO
              DO 140 K = 1, NLI
 140  VAR(I,J,K) = EXP(MEAN(I) + STDDEV(I)*Y(J,K))
              GOTO 160
      ELSE
      ENDIF
              DO 150 J = 1, NCO
              DO 150 K = 1, NLI
      YY = ABS(Y(J,K))
      IF(YY.LE.1.4) RN = 0.5D0+0.5D0*(1.D0-EXP(-2.D0*YY**2.D0/PI)
     '-2.D0*(PI-3.D0)/(3.D0*PI**2.D0)*YY**4.D0*EXP(-YY**
     '2.D0/2.D0))**0.5D0
      IF((YY.GT.1.4).AND.(YY.LE.2.2)) RN = 1.D0 - ((4.D0+YY**
     '2.D0)**0.5D0-YY)/2.D0*(2.D0*PI)**(-0.5D0)*EXP(-YY**2.D0/2.D0)
      IF(YY.GT.2.2) RN = 1.D0 - 1.D0/YY*(2.D0*PI)**(-0.5D0)*
     'EXP(-YY**2.D0/2.D0)
C
      IF(Y(J,K).LT.0.) RN = 1.- RN
C
      IF(TYPE(I).EQ.'TRIA') CALL TRI(MAX(I),MIN(I),MODE(I),VAR(I,J,K))
      IF(TYPE(I).EQ.'UNIF') CALL UNI(MAX(I),MIN(I),VAR(I,J,K))
```

```
  150 CONTINUE
C
  160 IF(I.EQ.6) THEN
        DO 170 J = 1, NCO
        DO 170 K = 1, NLI
  170         VAR(I,J,K) = 1./VAR(I,J,K)
      ELSE
      ENDIF
C
C
C
      IF(I.EQ.8) THEN
        DO 180 J = 1, NCO
        DO 180 K = 1, NLI
  180         VAR(I,J,K) = 1.-VAR(I,J,K)
      ELSE
      ENDIF
  190 CONTINUE
C
C     generate realizations of random variables
C
      DO 230 I = 3, 8
        IF((I.EQ.8) .AND. (MUCOR.NE.'NO'))                 GOTO 230
        IF((TYPE(I).EQ.'DETM').OR.(SPCOR(I).NE.'NO')) GOTO 230
          DO 200 J = 1, NCO
          DO 200 K = 1, NLI
      IF((TYPE(I).EQ.'NORM').OR.(TYPE(I).EQ.'LOGN')) GOTO 195
      CALL RAN
      IF(TYPE(I).EQ.'TRIA') CALL TRI(MAX(I),MIN(I),MODE(I),VAR(I,J,K))
      IF(TYPE(I).EQ.'UNIF') CALL UNI(MAX(I),MIN(I),VAR(I,J,K))
      GOTO 200
  195 CALL NOR(MEAN(I),STDDEV(I),VAR(I,J,K))
      IF(TYPE(I).EQ.'LOGN') VAR(I,J,K) = EXP(VAR(I,J,K))
  200 CONTINUE
C
      IF(I.EQ.6) THEN
        DO 210 J = 1, NCO
        DO 210 K = 1, NLI
  210         VAR(I,J,K) = 1./VAR(I,J,K)
      ELSE
      ENDIF
C
```

```
      IF(I.EQ.8) THEN
        DO 220 J = 1, NCO
        DO 220 K = 1, NLI
220         VAR(I,J,K) = 1.-VAR(I,J,K)
      ELSE
      ENDIF
230   CONTINUE
C
C     assign water saturation in case of having the relationship
C     between porosity and water saturation
C
      IF((TYPE(7).NE.'DETM').AND.(MUCOR.NE.'NO')) THEN
        DO 240 J = 1, NCO
        DO 240 K = 1, NLI
         CALL NOR(Ø.,1.,RNN)
         VAR(8,J,K) = AMU(1)*VAR(7,J,K)**2.
     *               + AMU(2)*VAR(7,J,K) + AMU(3)
     *               + RNN*RNDP
         IF(VAR(8,J,K).GT.1.) VAR(8,J,K) = 1.
         IF(VAR(8,J,K).LT.Ø.) VAR(8,J,K) = Ø.
         VAR(8,J,K) = 1. - VAR(8,J,K)
240      CONTINUE
      ELSE
      ENDIF
C
      IF((TYPE(7).EQ.'DETM').AND.(MUCOR.NE.'NO')) THEN
         CALL NOR(Ø.,1.,RNN)
         AVAR(8) = AMU(1)*AVAR(7)**2.
     *            + AMU(2)*AVAR(7) + AMU(3)
     *            + RNN*RNDP
         IF(AVAR(8).GT.1.) AVAR(8) = 1.
         IF(AVAR(8).LT.Ø.) AVAR(8) = Ø.
         AVAR(8) = 1. - AVAR(8)
      ELSE
      ENDIF
C
C
C
         DO 260 J = 1, NCO
         DO 260 K = 1, NLI
         RES = 1.Ø
C
```

```
      DO 250 I = 3, 8
      IF((I.EQ.8).AND.(TYPE(7).EQ.'DETM').AND.(MUCOR.NE.'NO'))
     *      GOTO 250
      IF((I.EQ.8).AND.(TYPE(7).NE.'DETM').AND.(MUCOR.NE.'NO'))
     *      GOTO 245
         IF(TYPE(I).EQ.'DETM') GOTO 250
245   RES = RES*VAR(I,J,K)
250 CONTINUE
         TRES(II) = TRES(II) + RES
260 CONTINUE
C
C
    DO 270 I = 3, 8
        IF((I.EQ.8).AND.(TYPE(7).NE.'DETM').AND.(MUCOR.NE.'NO'))
     *      GOTO 270
        IF((I.EQ.8).AND.(TYPE(7).EQ.'DETM').AND.(MUCOR.NE.'NO'))
     *      GOTO 265
        IF(TYPE(I).NE.'DETM') GOTO 270
265     TRES(II) = TRES(II)*AVAR(I)
270 CONTINUE
C
C
         TRES(II) = 7758.*TRES(II)*APB*AVAR(2)
C
C
280 CONTINUE
C *** ---------------------------------------------------------------
C ***                  DATA OUTPUT SECTION
C *** ---------------------------------------------------------------
    WRITE(6,1010)NOLOOP,NCO,NLI,RATIO
1010 FORMAT(10X,'----------------------------------------------',/,
     *      10X,'    Oil reserve calculation using            ',/,
     *      10X,'        Monte Carlo simulation               ',/,
     *      10X,'----------------------------------------------',/,
     *      10X,'Data input                                   ',/,
     *      10X,'----------------------------------------------',/,
     *      10X,'number of simulations        =',T40,I5       ,/,
     *      10X,'number of blocks in column =',T40,I4         ,/,
     *      10X,'number of blocks in line     =',T40,I4       ,/,
     *      10X,'ratio of length to width     =',T40,F8.4     ,/)
      DO 1020 I= 1, 8
      WRITE(6,1040)NAME(I)
```

```
      IF (TYPE(I).EQ.'TRIA') WRITE(6,1050)MAX(I),MIN(I),MODE(I)
      IF (TYPE(I).EQ.'NORM') WRITE(6,1060)MEAN(I),STDDEV(I)
      IF (TYPE(I).EQ.'LOGN') WRITE(6,1070)MEAN(I),STDDEV(I)
      IF (TYPE(I).EQ.'UNIF') WRITE(6,1080)MAX(I),MIN(I)
      IF (TYPE(I).EQ.'DETM') WRITE(6,1090)DETEM(I)
      IF ((I.EQ.1).OR.(I.EQ.2)) GOTO 1020
      IF (SPCOR(I).NE.'NO') WRITE(6,1100) LENGTH(I)
1020  CONTINUE
      IF (MUCOR.NE.'NO') THEN
      WRITE(6,1110)
         DO 1030 I = 1, 3
1030        WRITE(6,1120) I, AMU(I)
      WRITE(6,1130) RNDP
      ELSE
      ENDIF
1040 FORMAT(T10,'random variable          : ',A21)
1050 FORMAT(T10,'type of distribution     : triangular distribution',/,
   *        T10,'              maximum    = ',F12.5,/,
   *        T10,'              minimum    = ',F12.5,/,
   *        T10,'                 mode    = ',F12.5,/)
1060 FORMAT(T10,'type of distribution     : normal distribution',/,
   *        T10,'                 mean    = ',F12.5,/,
   *        T10,'standard deviation       = ',F12.5,/)
1070 FORMAT(T10,'type of distribution     : log-normal distribution',/,
   *        T10,'                 mean    = ',F12.5,/,
   *        T10,'standard deviation       = ',F12.5,/)
1080 FORMAT(T10,'type of distribution     : uniform distribution',/,
   *        T10,'              maximum    = ',F12.5,/,
   *        T10,'              minimum    = ',F12.5,/)
1090 FORMAT(T10,'type of distribution     : deterministic value',/,
   *        T10,'                value    = ',F12.5,/)
1100 FORMAT(T10,/,'SPATIAL CORRELATIONS',/,
   *        T10,'EXPONENTIAL SCHEME    LENGTH = ',F12.6,/)
1110 FORMAT(T10,'CORRELATION BETWEEN POROSITY AND WATER SATURATION',/)
1120 FORMAT(T10,'AMU(',I1,') = ',F12.5)
1130 FORMAT(T10,'RANDOM PART = ',F12.5)
      CALL PROB(TRES,NOLOOP)
      CLOSE(6)
      STOP
      END
C
C
```

```
C ***********************************************************************
C     -------------------------------------------------------------
C             SUBROUTINE 2-DIMENSIONAL TURNING BAND METHOD
C     -------------------------------------------------------------
      SUBROUTINE TURN(LENGTH,II1)
      DOUBLE PRECISION W(100), S1(100), WKP(100), PHI(100)
      DOUBLE PRECISION XX(2), ZI(16,400), SDW
      DOUBLE PRECISION PI, SX(16,2), DW, DWP
      DOUBLE PRECISION UN, DIST, ANG, ANG1
      DIMENSION Y(60,60)
      REAL LENGTH
      COMMON
     */L1/RN
     */L2/M2,IY,IA,IC,MIC,SRAN
     */L4/DC,DL,NCO,NLI
     */L5/Y,PI,SX,DWP,S1,W
C
C *** compute rotation matrix SX of the 16 bands
C
      IF (II1.NE.1) GOTO 15
      PI = 3.141592654D0
      ANG = 0.D0
      ANG1 = PI/16.D0
      DO 10 IT = 1, 16
      SX(IT,1) = SIN(ANG)
      SX(IT,2) = COS(ANG)
   10 ANG = ANG + ANG1
C
C *** omega = 20/length    harmonic number = 100
C
   15 DW = 0.2D0/LENGTH
      DWP = DW/20.D0
      DDW = DW/2.D0
C
C *** exponential spectral density function
C
      DO 30 KT = 1, 100
      W(KT) = (FLOAT(KT)-0.5D0)*DW
   30 S1(KT) = SQRT(DDW*W(KT)*LENGTH*LENGTH/
     *((1.D0+(W(KT)*LENGTH)**2.D0)**(3.D0/2.D0)))
C
C
```

```
C
C *** uni-dimensional simulation
C
      UN = AMIN1(Ø.5*DC,Ø.5*DL)
      DL = DL/UN
      DC = DC/UN
      NGMAX = SQRT(NLI*NLI*DL*DL+NCO*NCO*DC*DC) + 4
C
      IF(NGMAX.GT.4ØØ) THEN
      WRITE(6,*)'NGMAX GREATER THAN 4ØØ : PROGRAM CANNOT WORK'
      STOP
      ELSE
      ENDIF
C
      CALL RAN
      SDW = DWP * (RN - Ø.5DØ)
          DO 4Ø KT = 1, 1ØØ
                   WKP(KT) = W(KT) + SDW
   4Ø CONTINUE
      DO 7Ø ID = 1, 16
          DO 5Ø KT = 1, 1ØØ
                   CALL RAN
                   PHI(KT) = RN * 2.DØ * PI
   5Ø CONTINUE
      DO 7Ø JT = 1, NGMAX
      DIST = FLOAT(JT)*UN
      ZI(ID,JT) = Ø.DØ
          DO 6Ø KT = 1, 1ØØ
   6Ø ZI(ID,JT) = ZI(ID,JT)+ S1(KT)*COS(WKP(KT)*DIST + PHI(KT))
      ZI(ID,JT) = ZI(ID,JT)*2.DØ
   7Ø CONTINUE
C
C *** two-dimensional simulation on NCO columns and NLI lines
C
      N1 = (NLI+1)/2
      N2 = (NCO+1)/2
      NG = NGMAX/2
      DO 1ØØ IT = 1, NCO
      XX(2) = -Ø.5+(IT-N2)*DC
      DO 1ØØ JT = 1, NLI
      XX(1) = -Ø.5+(JT-N1)*DL
      Y(IT,JT) = Ø.
```

```
      DO 90 IIT = 1, 16
      XI = 0.
      DO 80 JJT = 1, 2
   80 XI = XI + XX(JJT)*SX(IIT,JJT)
      XI = XI + 0.5
      LK = XI
      IF(XI.LT.0.) LK = LK - 1
      LK = LK + NG
      Y(IT,JT) = Y(IT,JT) + ZI(IIT,LK)
   90 CONTINUE
      Y(IT,JT) = Y(IT,JT)/4.D0
  100 CONTINUE
      DL = DL*UN
      DC = DC*UN
      RETURN
      END
C     ----------------------------------------------------------------
C                  SUBROUTINE TRIANGULAR DISTRIBUTION
C     ----------------------------------------------------------------
      SUBROUTINE TRI(MAX,MIN,MODE,VARI)
      REAL MAX,MIN,MODE
      COMMON /L1/RN
      AAA = (MODE - MIN)/(MAX - MIN)
      IF(RN.LE.AAA) THEN
          VARI = MIN + SQRT((MODE-MIN)*(MAX-MIN)*RN)
      ELSE
          VARI = MAX - SQRT((MAX-MODE)*(MAX-MIN)*(1.-RN))
      ENDIF
      RETURN
      END
C     ----------------------------------------------------------------
C                  SUBROUTINE NORMAL DISTRIBUTION
C     ----------------------------------------------------------------
      SUBROUTINE NOR(AME,STD,VARI)
      COMMON /L1/RN /L2/M2,IY,IA,IC,MIC,SRAN /L3/INOR,RN1,RN2
      IF(INOR.NE.0) GO TO 20
   10 CALL RAN
      V1 = 2.*RN - 1.
      CALL RAN
      V2 = 2.*RN - 1.
      SS = V1**2 + V2**2
      IF(SS.GT.1.) GOTO 10
```

```
      RN1 = V1*SQRT(-2.*ALOG(SS)/SS)
      RN2 = V2*SQRT(-2.*ALOG(SS)/SS)
      VARI = AME + RN1 * STD
      INOR = 1
      GO TO 3Ø
   2Ø VARI = AME + RN2 * STD
      INOR = Ø
   3Ø RETURN
      END
C     ----------------------------------------------------------------
C                SUBROUTINE UNIFORM DISTRIBUTION
C     ----------------------------------------------------------------
      SUBROUTINE UNI(MAX,MIN,VARI)
      REAL MAX,MIN
      COMMON /L1/RN
      VARI = MIN + RN*(MAX-MIN)
      RETURN
      END
C *** ----------------------------------------------------------
C ***           SUBROUTINE FOR HISTOGRAM PLOTING
C *** ----------------------------------------------------------
      SUBROUTINE PROB(X,N)
      REAL   X(1ØØØ), POP(15), NCU(15),
     *       APOP(15),ANCU(15),LOLIM, LOBND, MIDPT
      INTEGER IPOP(15),INCU(15)
      CHARACTER PRIT*3
C     ----------------------------------------------------------
C     X        = set of random variables
C     N        = numbers of random variables
C     XMIN     = minimum value
C     XMAX     = maximum value
C     XMED     = median
C     XBAR     = mean
C     XMOD     = mode
C     VARIAN   = variance
C     STD      = standard deviation
C     FREQ(I)  = frequency in class i
C     CU(I)    = cumulative frequency in class i
C     POP(I)   = probability in class i
C     NCU(I)   = negative cumulative probability in class i
C     IPOP(I)  = numbers of signs (]) for ploting
C                for probability distribution
```

```
C     INCU(I) = numbers of signs (*) for ploting
C               for negative cumulative probability
C     LOLIM   = lower limit
C     UPLIM   = upper limit
C     LOBND   = lower bound
C     UPBND   = upper bound
C     WDTH    = width of each class
C     -----------------------------------------------------
C
C     SORTING X(N)
C
      NN = N - 1
      DO 2Ø J = 1, NN
         L = J
        JJ = J + 1
           DO 1Ø I = JJ, N
              IF(X(L).LE.X(I)) GOTO 1Ø
              L = I
   1Ø CONTINUE
            T = X(L)
         X(L) = X(J)
         X(J) = T
   2Ø CONTINUE
C
C *** calculate statistical values
C
      XMIN = X(1)
      XMAX = X(N)
      MED1 = N/2
      MED2 = (N+1)/2
      XMED = X(MED2)
      IF(MED1.EQ.MED2) XMED = (XMED + X(MED2+1))/2.
      XBAR = Ø.
      VARIAN = Ø.
      DO 3Ø I = 1, N
          XBAR = XBAR + X(I)
   3Ø CONTINUE
      XBAR = XBAR/FLOAT(N)
      DO 35 I = 1, N
          VARIAN = VARIAN + (XBAR-X(I))**2
   35 CONTINUE
      VARIAN = VARIAN/FLOAT(N-1)
```

```
      STD = SQRT(VARIAN)
C
C     plot graph and find mode
C
      WRITE(*,*)'SELECT MODE OF PRINTING , MANUAL OR AUTO'
      READ(*,*)PRIT
      IF(PRIT.EQ.'MAN') THEN
          WRITE(*,*) 'INPUT LOWER BOUND'
          READ(*,*)LOBND
          WRITE(*,*) 'INPUT UPPER BOUND'
          READ(*,*)UPBND
          UPBND = UPBND + LOBND/1Ø.E15
          LOBND = LOBND - LOBND/1Ø.E15
      ELSE
          LOBND = XMIN - XMIN/1Ø.E15
          UPBND = XMAX + XMIN/1Ø.E15
      ENDIF
C
      WDTH = (UPBND - LOBND)/15.
      UPLIM = LOBND + WDTH
      LOLIM = LOBND
      AN = N
C
C     classification
C
      POPMAX = -1.
      DO 5Ø I = 1, 15
             POP(I) = Ø.
             NCU(I) = Ø.
            DO 4Ø J = 1,N
                 IF((X(J).GT.LOLIM).AND.(X(J).LE.UPLIM))
     *                              POP(I) = POP(I) + 1.
                 IF(X(J).LE.UPLIM) NCU(I) = NCU(I) + 1.
 4Ø         CONTINUE
            LOLIM = UPLIM
            UPLIM = UPLIM + WDTH
            POP(I) = POP(I)/AN
            NCU(I) = 1.- (NCU(I)/AN)
            IF(POP(I).GT.POPMAX) THEN
               POPMAX = POP(I)
               ISM = I
            ELSE
```

```
        ENDIF
  50 CONTINUE
        XMOD = LOBND + (FLOAT(ISM)-0.5)*WDTH
     WRITE(6,60)
  60 FORMAT(/,5X,'-------------------------------------------------',/,
    *      5X,'Results : Oil reserve calculation  (bbl.)        ',/,
    *      5X,'-------------------------------------------------')
     WRITE(6,*)'                  Minimum = ',XMIN
     WRITE(6,*)'                  Maximum = ',XMAX
     WRITE(6,*)'                     Mean = ',XBAR
     WRITE(6,*)'                   Median = ',XMED
     WRITE(6,*)'                     Mode = ',XMOD
     WRITE(6,*)'                 Variance = ',VARIAN
     WRITE(6,*)'       Standard deviation = ',STD
     WRITE(6,*)'   ---------------------------------------------'
C
C *** print out probability distribution
C
     WRITE(6,70)
  70 FORMAT(////,
    *'---------------------------------------------------',/,
    *'data output for probability                        ',/,
    *'---------------------------------------------------',/,
    *'Probability of oil reserve',/)
        MIDPT = LOBND + WDTH/2.D0
     DO 80 I = 1, 15
        WRITE(6,*)MIDPT, POP(I)
        MIDPT = MIDPT + WDTH
  80 CONTINUE
     IF(POPMAX.LE.0.5) THEN
        XX = 2.
        WRITE(6,120)
     ELSE
        XX = 1.
        WRITE(6,130)
     ENDIF
     DO 110 I = 1, 15
        APOP(I) = POP(I)*50.*XX +.5
        ANCU(I) = NCU(I)*50. +.5
        IPOP(I) = APOP(I)
        INCU(I) = ANCU(I)
 110 CONTINUE
```

```
120 FORMAT(///,
   *16X,'                    Probability                    ',/,
   *16X,'Ø.Ø       Ø.1       Ø.2       Ø.3       Ø.4       Ø.5',/,
   *16X,' +----+----+----+----+----+----+----+----+----+----+ ')
130 FORMAT(///,
   *16X,'                    Probability                    ',/,
   *16X,'Ø.Ø  Ø.1  Ø.2  Ø.3  Ø.4  Ø.5  Ø.6  Ø.7  Ø.8  Ø.9  1.Ø',/,
   *16X,' +----+----+----+----+----+----+----+----+----+----+ ')
        MIDPT = LOBND + WDTH/2.DØ
    DO 14Ø I = 1, 15
        IF(IPOP(I).EQ.Ø) THEN
            WRITE (6,*) MIDPT,' +'
            WRITE (6,*)'              +'
            MIDPT = MIDPT + WDTH
        ELSE
            WRITE (6,*) MIDPT,' +',(']',J=1,IPOP(I))
            WRITE (6,*)'              +'
            MIDPT = MIDPT + WDTH
        ENDIF
  14Ø CONTINUE
C
C *** print out negative cumulative probability
C
      WRITE(6,15Ø)
  15Ø FORMAT(////,
     *'----------------------------------------------',/,
     *'data output for negative cumulative probability ',/,
     *'----------------------------------------------',//,
     *'Probability of value grether than ')
      P = 1.Ø
      WRITE(6,17Ø)LOBND,P
      UPLIM = LOBND + WDTH
      DO 16Ø I = 1, 15
          WRITE(6,17Ø)UPLIM,NCU(I)
              UPLIM = UPLIM + WDTH
  16Ø CONTINUE
  17Ø FORMAT(F2Ø.Ø,'      =       ',F6.4)
      WRITE(6,18Ø)
  18Ø FORMAT(///,
     *16X,'            Negative cumulative probability         ',/,
     *16X,'Ø.Ø  Ø.1  Ø.2  Ø.3  Ø.4  Ø.5  Ø.6  Ø.7  Ø.8  Ø.9  1.Ø',/,
     *16X,' +----+----+----+----+----+----+----+----+----+----+ ')
```

```
      WRITE(6,*)LOBND,' +',(' ',J=1,49),'*'
      UPLIM = LOBND + WDTH
      DO 190 I = 1, 15
         IF(INCU(I).EQ.0) THEN
            WRITE(6,*)'                  +'
            WRITE (6,*) UPLIM,' +'
            UPLIM = UPLIM + WDTH
         ELSE
            WRITE(6,*)'                  +'
            WRITE (6,*) UPLIM,' +',(' ',J=1,INCU(I)-1),'*'
            UPLIM = UPLIM + WDTH
         ENDIF
  190 CONTINUE
      RETURN
      END
C     ------------------------------------------------------------
```

13. Data input file for oil reserve calculation program.

```
******************************************************
*          DATA FILE FOR OIL RESERVE CALCULATION        *
*               USING MONTE CARLO SIMULATION            *
*                (2-DIMENSIONAL SIMULATION)             *
******************************************************

DATA REQUIRED
     1. NUMBER OF SIMULATIONS
     2. NUMBER OF BLOCK IN A COLUMN
     3. NUMBER OF BLOCK IN A LINE
     4. RATIO OF LENGTH TO WIDTH OF AREA
     5. TYPE OF DISTRIBUTION OF VARIABLES
     6. PARAMETERS OF THEIR DISTRIBUTIONS
     7. CORRELATION LENGTH OF THE EXPONENTIAL SPATIAL CORRELATION
     8. PARAMETER OF FUNCTION BETWEEN POROSITY AND WATER SATURATION

THERE ARE 8 VARIABLES :
     1. AREA
     2. RECOVERY FACTOR
     3. GLOSS THICKNESS
     4. NET TO GLOSS RATIO
     5. OIL SAND TO NET SAND RATIO
     6. INITIAL OIL FORMATION VOLUMN FACTOR
     7. POROSITY
     8. WATER SATURATION

TYPE OF DISTRIBUTION OF VARIABLES
     1. DETERMINISTIC VALUE      ;'DETM' : NEED CONSTANT
     2. UNIFORM DISTRIBUTION     ;'UNIF' : NEED MAX,MIN
     3. TRIANGULAR DISTRIBUTION  ;'TRIA' : NEED MAX,MIN,MODE
     4. NORMAL DISTRIBUTION      ;'NORM' : NEED MEAN,STDDEV
     5. LOG-NORMAL DISTRIBUTION  ;'LOGN' : NEED MEAN,STDDEV
                                                (IN LOG-TERM)
-----------------------------------------------------------------


--------------------------
   ENTER THE PARAMETERS
--------------------------
NUMBER OF SIMULATIONS (NOLOOP <= 1000)
>600
```

```
NUMBER OF BLOCK IN X-AXIS (NCO <= 50)
>1

NUMBER OF BLOCK IN Y-AXIS (NLI <= 50)
>1

RATIO OF LENGTH TO WIDTH OF AREA (RATIO > 0.)
>1.

-----------------------------------------
ENTER THE DISTRIBUTION OF VARIABLES
-----------------------------------------
1) AREA
 'DETM'
  PARAMETERS OF DISTRIBUTION
 >1.0
---------------------------------------------------------------
2) RECOVERY FACTOR
 'DETM'
  PARAMETERS OF DISTRIBUTION
 >0.2
---------------------------------------------------------------
3) GLOSS THICKNESS
 'DETM'
  PARAMETERS OF DISTRIBUTION
 >1.0
  IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
 'NO' LENGTH = 0.0
---------------------------------------------------------------
4) NET TO GLOSS RATIO
 'DETM'
  PARAMETERS OF DISTRIBUTION
 >1.0
  IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
 'NO' LENGTH = 0.0
---------------------------------------------------------------
5) OIL SAND TO NET SAND RATIO
 'DETM'
  PARAMETERS OF DISTRIBUTION
 >1.0
  IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
```

```
    'NO' LENGTH = 0.0
------------------------------------------------------------
  6) INIT OIL FORM VOL FAC
    'DETM'
     PARAMETERS OF DISTRIBUTION
    >1.2
     IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
    'NO' LENGTH = 0.0
------------------------------------------------------------
  7) POROSITY
    'TRIA'
     PARAMETERS OF DISTRIBUTION
    >0.34          0.24           0.28
     IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
    'NO' LENGTH = 0.0
------------------------------------------------------------
  8) WATER SATURATION
    'TRIA'
     PARAMETERS OF DISTRIBUTION
    >0.55          0.27           0.45
     IS THERE IT'S SPATIAL CORRELATION ? (YES/NO)
    'NO' LENGTH = 0.0
-----------------------------------------------------------------------------
IS THERE THE CORRELATION BETWEEN POROSITY AND WATER SATURATION ? (YES/NO)
    'NO'
FUNCTION BETWEEN POROSITY AND WATER SATURATION
     SW = A1*PO^2 + A2*PO + A3

     A1 =          0.0
     A2 =          0.0
     A3 =          0.0
RANDOM PART AT 1STD = 0.0
*************************************************************************
```

VITAE

Anuntasak Suksasilp was born on October, 29, 1966 in Bangkok, Thailand. He received his B.Sc. in Industrial Chemistry from the Faculty of Science, King Mongkut's Institute of Technology, Ladkrabang in 1989. After graduating, he has been a graduate student in the Petro - Polymer College Project, Chulalongkorn University.