CHAPTER 1

Numerical Computation

INTRODUCTION

Engineers, technologists, and scientists have employed numerical methods of analysis to solve a wide range of steady and transient problems. The fundamentals are essential in the basic operations of curve fitting, approximation, interpolation, numerical solutions of simultaneous linear and nonlinear equations, numerical differentiation and integration. These requirements are greater when new processes are designed. Engineers also need theoretical information and data from published works to construct mathematical models that simulate new processes. Developing mathematical models with personal computers sometimes involves experimental programs to obtain the required information for the models. Developing an experimental program is strongly dependent on the knowledge of the process with theory, where the whole modification can be produced by some form of mathematical models or regression analyses. Figure 1-1 shows the relationship between mathematical modeling and regression analysis.

Texts [1-5] with computer programs and sometimes with supplied software are now available for scientists and engineers. They must fit a function or functions to measure data that fluctuate, which result from random error of measurement. If the number of data points equals the order of the polynomial plus one, we can exactly fit a polynomial to the data points. Fitting a function to a set of data requires more data than the order of a polynomial. The accuracy of the fitted curve depends on the large number of experimental data. In this chapter, we will develop

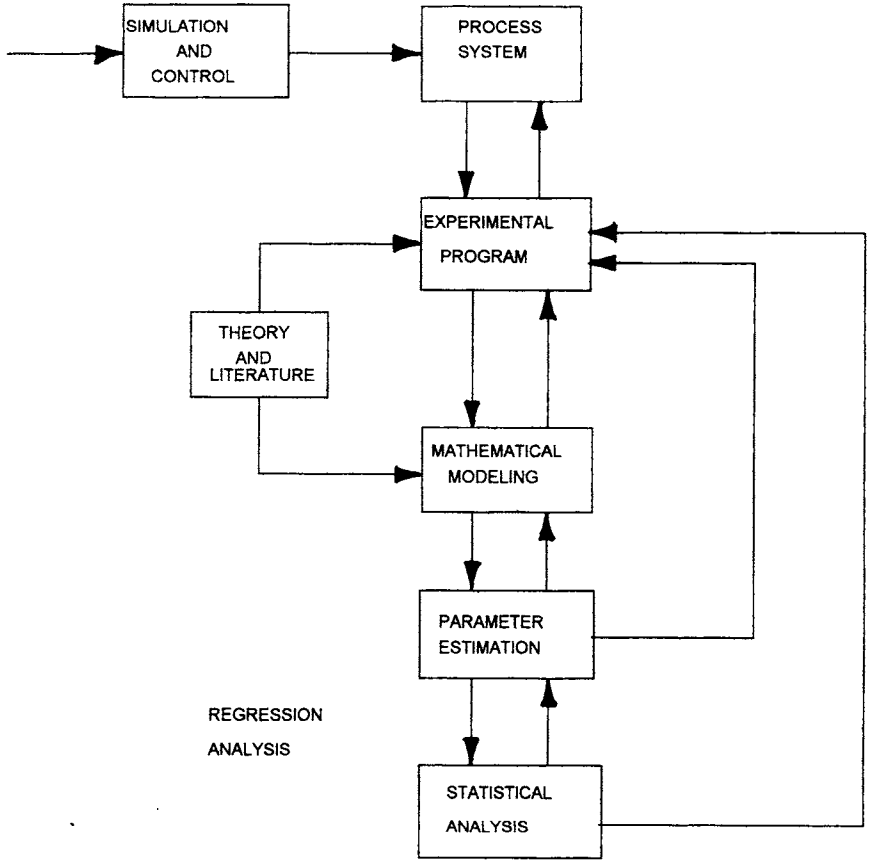


Figure 1-1. Mathematical modeling and regression analysis. By permission, A. Constantinides, *Applied Numerical Methods With Personal Computers*, McGraw-Hill Book Co., 1987.

least-squares curve fitting programs. Also, we will use linear regression analyses to develop statistical relationships involving two or more variables. The most common type of relation is a linear function. By transforming nonlinear functions, many functional relations are made linear.

For the program, we will correlate X-Y data for the following equations:

$$Y = a + bX \tag{1-1}$$

$$Y = a + bX^2 \tag{1-2}$$

$$Y = a + b/X \tag{1-3}$$

$$Y = a + bX^{0.5} \tag{1-4}$$

$$Y = aX^{b} \tag{1-5}$$

$$Y = ae^{bX} \tag{1-6}$$

$$Y = a + b \log X \tag{1-7}$$

$$Y = a + be^{x} \tag{1-8}$$

We can transform the nonlinear equations (1-5, 1-6, and 1-8) by linearizing as follows:

$$Y = aX^{b} \qquad \ln Y = \ln a + b \ln X \tag{1-9}$$

$$Y = ae^{bX} \qquad \ln Y = \ln a + bX \tag{1-10}$$

$$Y = a + be^{x} \qquad \ln Y = \ln a + (\ln b)X \tag{1-11}$$

LINEAR REGRESSION ANALYSIS

Regression analysis uses statistical and mathematical methods to analyze experimental data and to fit mathematical models to these data. We can solve for the unknown parameters after fitting the model to the data. Suppose we want to find a linear function that involves paired observations on two variables, X the independent variable and Y the dependent variable. Where

n = the number of observations
X_i = the i^{th} observation of the independent variable
Y_i = the i^{th} observation of the dependent variable

We can develop a linear regression model that expresses Y as a function of X. We can further derive formulae to determine the values of a and b that give the best fit of the equations. For each experimental point corresponding to an X-Y pair, there will be an element that represents the difference between the corresponding calculated value $\hat{Y}$ and the original value of Y. This is expressed as:

$$r_i = \hat{Y}_i - Y \tag{1-12}$$

r_i may be either positive or negative depending on the side of the fitted curve of X-Y points. Here, we can minimize the sum of the squares of the residuals by the following expression:

$$SRS = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} \zeta_i^2 = \text{minimum} \tag{1-13}$$

where

n is the number of observations of X-Y points

$$\hat{Y}_i = a + bX_i \tag{1-14}$$

$$r_i = a + bX_i - Y_i \tag{1-15}$$

and

$$SRS = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (a + bX_i - Y_i)^2 \tag{1-16}$$

The problem is reduced to finding the values of a and b so that the summation of Equation 1-16 is minimized. We can obtain this by taking the partial derivative of Equation 1-16 with respect to each variable a and b and set the result to zero.

$$\frac{\partial \sum r_i^2}{\partial a} = 0 \quad \text{and} \quad \frac{\partial \sum r_i^2}{\partial b} = 0 \tag{1-17}$$

Substituting Equation 1-16 into Equation 1-17, we obtain

$$\frac{\partial \sum (a + bX_i - Y_i)^2}{\partial a} = 0 \tag{1-18}$$

and

$$\frac{\partial \sum (a + bX_i - Y_i)^2}{\partial b} = 0 \tag{1-19}$$

This is equivalent to

$$\frac{2 \sum (a + bX_i - Y_i) \partial \sum (a + bX_i - Y_i)}{\partial a} = 0 \tag{1-20}$$

and

(1-21)

Since b, X, and Y are not functions of a, and the partial derivative of a with respect to itself is unity, Equation 1-20 reduces to

$$\sum a + \sum bX_i = \sum Y_i \tag{1-22}$$

Similarly, a, X, and Y are not functions of b. Therefore, Equation 1-21 becomes

$$\sum aX_i + \sum bX_i^2 = \sum X_iY_i \tag{1-23}$$

where a and b are constants. Equations 1-18 and 1-19 are expressed as:

$$an + b\sum X_i = \sum Y_i \tag{1-24}$$

and

$$a\sum X_i + b\sum X_i^2 = \sum X_iY_i \tag{1-25}$$

We have now reduced the problem of finding a straight line through a set of X – Y data points to one of solving two simultaneous equations 1-24 and 1-25. Both equations are linear in X, Y, and n, and the unknowns a and b. Using Cramer's rule for the simultaneous equations, we have

$$a = \frac{\begin{vmatrix} \sum Y_i & \sum X_i \\ \sum X_iY_i & \sum X_i^2 \end{vmatrix}}{\begin{vmatrix} n & \sum X_i \\ \sum X_i & \sum X_i^2 \end{vmatrix}} \tag{1-26}$$

and

$$b = \frac{\begin{vmatrix} n & \sum Y_i \\ \sum X_i & \sum X_iY_i \end{vmatrix}}{\begin{vmatrix} n & \sum X_i \\ \sum X_i & \sum X_i^2 \end{vmatrix}} \tag{1-27}$$

Solving Equations 1-26 and 1-27 gives

$$a = \frac{\sum X_i^2 \sum Y_i - \sum X_i \sum X_i Y_i}{n\sum X_i^2 - \sum X_i \sum X_i} \tag{1-28}$$

and

$$b = \frac{n\sum X_i Y_i - \sum X_i \sum Y_i}{n\sum X_i^2 - \sum X_i \sum X_i} \tag{1-29}$$

respectively, where all the sums are taken over all experimental observations.

Also, we can construct a table with columns headed X_i, Y_i, X_i^2, $X_i Y_i$ as an alternative to obtain the values of a and b. The sums of the columns will then give all the values required to Equations 1-28 and 1-29.

METHODS FOR MEASURING REGRESSION

Linear Regression

From the values of a, b, and X_i, the independent variable, we can now calculate the corresponding estimated value of Y, designated as $\hat{Y}_i$.

$$\hat{Y}_i = a + b.X_i \tag{1-30}$$

Figure 1-2 illustrates this equation known as the regression line. The variation of the observed Y's about the mean of the observed Y's is the total sum of squares, and can be expressed as:

$$SST = \sum (Y_i - \overline{Y})^2 \tag{1-31}$$

where

$$\overline{Y} = \frac{\sum Y_i}{N} \tag{1-32}$$

Figure 1-3 shows the differences between Y and $\overline{Y}$, which is the total sum of squares. This is the sum of these squared differences. The total sum of squares can be expressed as:

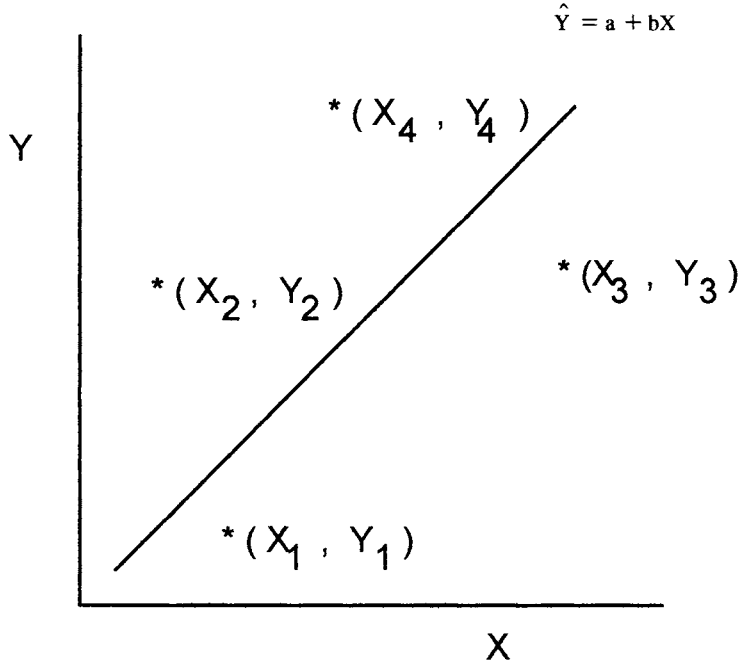


Figure 1-2. Regression line.

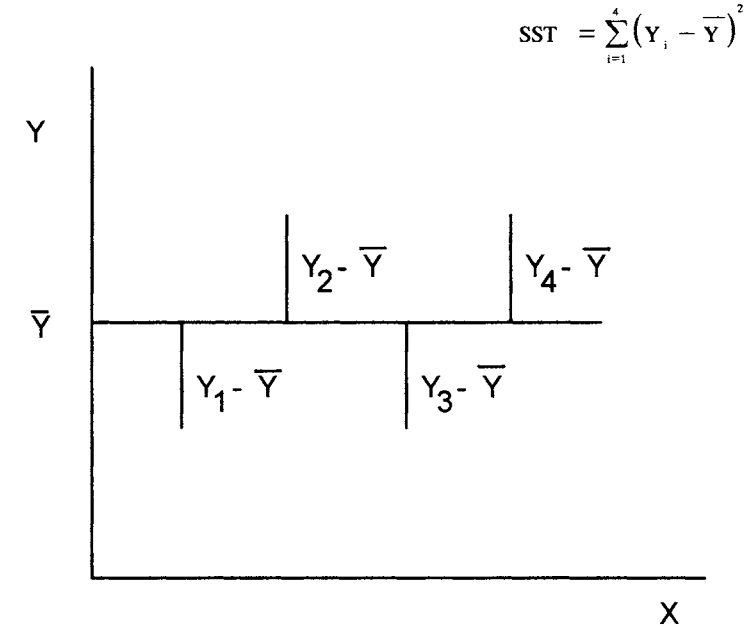


Figure 1-3. Total sum of squares.

$$SST = \sum (Y_i - \overline{Y})^2 \tag{1-33}$$

$$= \sum \left[(Y_i - \hat{Y}_i) + (\hat{Y}_i - \overline{Y}) \right]^2 \tag{1-34}$$

$$= \sum (Y_i - \hat{Y}_i)^2 + 2 \sum (Y_i - \hat{Y}_i)(\hat{Y}_i - \overline{Y}) + \sum (\hat{Y}_i - \overline{Y})^2 \tag{1-35}$$

and since

$$2 \sum (Y_i - \hat{Y}_i)(\hat{Y}_i - \overline{Y}) = 0$$

then

$$SST = \sum (Y_i - \hat{Y}_i)^2 + \sum (\hat{Y}_i - \overline{Y})^2 \tag{1-36}$$

$$= SSE + SSR$$

where

$$SSE = \sum (Y_i - \hat{Y}_i)^2 \tag{1-37}$$

and

$$SSR = \sum (\hat{Y}_i - \overline{Y})^2 \tag{1-38}$$

SSE is the error (or residual) sum of squares. This is the quantity represented by Equation 1-13, that is, the equation we want to minimize. It is the sum of the differences squared between the observed Y's and the estimated (or computed) $\hat{Y}$'s. Figure 1-4 shows the differences between the Y's and $\hat{Y}$'s. SSR is the regression sum of squares and measures the variation of the estimated values, $\hat{Y}$, about the mean of the observed Y's and $\overline{Y}$. Figure 1-5 shows the differences between Y's and $\overline{Y}$. The regression sum of squares is the sum of the squared differences.

The ratio of the regression sum of squares to the total sum of squares is used in calculating the coefficient of determination. This shows how well a regression line fits the observed data. The coefficient of determination is:

$$SSE = \sum_{i=1}^{4} \left(Y_i - \hat{Y}_i \right)^2$$

$$\hat{Y} = a + bX$$

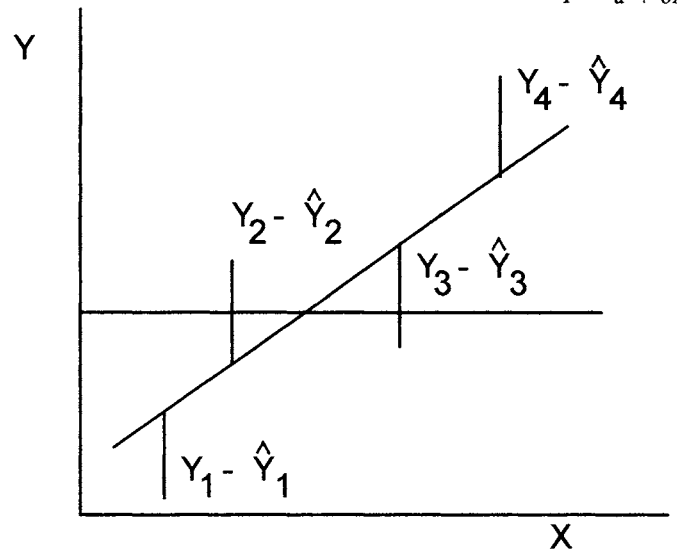


Figure 1-4. Error sum of squares.

$$SSR = \sum_{i=1}^{4} \left(\hat{Y}_i - \overline{Y} \right)^2 \qquad \hat{Y} = a + bX$$

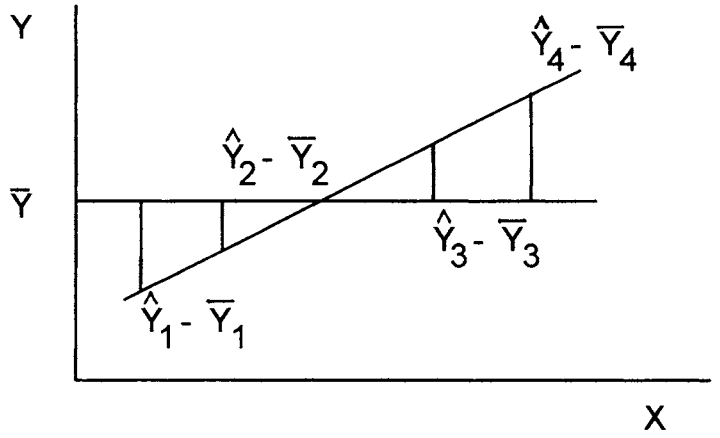


Figure 1-5. Regression sum of squares.

$$r^2 = \frac{SSR}{SST} \tag{1-39}$$

and since SSR = SST – SSE

$$r^2 = 1 - \frac{SSE}{SST} \tag{1-40}$$

The coefficient of determination has the following properties:

1. $0 \leq r^2 \leq 1$
2. If $r^2 = 1$ all ξ_i are zero. The observed Y's and the estimated $\hat{Y}$'s are the same and this indicates a perfect fit.
3. If $r^2 = 0$, no linear functional relationship exists.
4. As r^2 approaches one, the better the fit. The closer r^2 approaches zero, the worse the fit. The correlation coefficient r is:

$$r = \left(1 - \frac{SSE}{SST}\right)^{0.5} \tag{1-41}$$

Although the correlation coefficient gives a measure of the accuracy of fit, we should treat this method of analysis with great caution. Because r is close to one does not always mean that the fit is necessarily good. It is possible to obtain a high value of r when the underlying relationship between X and Y is not even linear. Draper and Smith [6] provide an excellent guidance of assessing results of linear regression.

THE ANALYSIS OF VARIANCE TABLE FOR LINEAR REGRESSION

Each sum of squares SSR, SSE, and SST corresponds to many degrees of freedom. This is the number of observations, n minus the number of independent quantities. The error sum of squares SSE has N-2 degrees of freedom since

$$SSE = \sum (Y_i - \hat{Y}_i)^2$$

and

$$\hat{Y} = a + b.X$$

then

$$SSE = \sum [Y_i - (a + bX_i)]^2 \tag{1-42}$$

where a and b represent two independent quantities calculated from a set of N observations giving the error sum of squares N-2 degrees of freedom.

The error mean squares, MSE, is an estimate of the variance of the error of observation, ξ, and is:

$$MSE = \frac{SSE}{N - 2} \tag{1-43}$$

The number of degrees of freedom for the total sums of squares SST is N-1 since

$$SST = \sum (Y_i - \overline{Y}_i)^2$$

$\overline{Y}$ represents a quantity from a set of N observations. This gives the total sum of squares N-1 degrees of freedom. The total mean squares, MST, is an estimate of the variance of the dependent variable Y and is:

$$MST = \frac{SST}{N - 1} \tag{1-44}$$

The regression sum of squares, SSR, has one degree of freedom. Since

$$\begin{aligned} SSR &= SST - SSE \\ &= (N - 1) - (N - 2) \\ &= 1 \end{aligned}$$

The regression mean squares, MSR, is an estimate of the variance of the estimates, $\hat{Y}$, and since the regression sum of squares has one degree of freedom, the regression mean squares is

$$MSR = SSR \tag{1-45}$$

Table 1-1 shows the variance table for linear regression.

The test statistics for linear regression is:

$$F = \frac{MSR}{MSE} \tag{1-46}$$

Table 1-1
Variance Table for Linear Regression

Source of Variation	Degrees of Freedom	Sum of Squares	Mean Squares
Total	$N-1$	$SST = \sum (Y_i - \overline{Y})^2$	$MST = \frac{SST}{N-1}$
Regression	1	$SSR = \sum (\hat{Y} - \overline{Y})^2$	$MSR = SSR$
Error	$N-2$	$SSE = \sum (Y_i - \hat{Y}_i)^2$	$MSE = \frac{SSE}{N-2}$

The F-distribution has one degree of freedom in the numerator and N-2 degree of freedom in the denominator.

MULTIPLE REGRESSION ANALYSIS

Inadequate results are sometimes obtained with a single independent variable. This shows that one independent variable does not provide enough information to predict the corresponding value of the dependent variable. We can approach this problem, if we use additional independent variables and develop a multiple regression analysis to achieve a meaningful relationship. Here, we can employ a linear regression model in cases where the dependent variable is affected by two or more controlled variables.

The linear multiple regression equation is expressed as:

$$Y = C_0 + C_1X_1 + C_2X_2 + \ldots + C_KX_K \qquad (1\text{-}47)$$

where Y = the dependent variable
$X_1, X_2, \ldots X_K$ = the independent variables
K = the number of independent variables
$C_0, C_1, C_2, \ldots C_K$ = the unknown regression coefficients

The unknown coefficients are estimated based on n observation for the dependent variable Y, and for each of the independent variables X_i's where $i = 1,2,3, \ldots K$.

These observations are of the form:

$$Y_j = C_0 = C_1X_{1j} + C_2X_{2j} + \ldots + C_KX_{Kj} + \xi_j \qquad (1\text{-}48)$$

for j = 1, 2, . . . N

where Y_j = the j_{th} observation of the dependent variable
$X_{1j}, \ldots X_{Kj}$ = the j_{th} observation of the X_1, X_2, . . . X_K independent variables

We can use a least squares technique to calculate estimates of $\hat{C}_0, \hat{C}_1, \ldots \hat{C}_K$ of the coefficients $C_0, C_1, \ldots C_K$ by minimizing the following equation:

$$S = \sum_{j=1}^{N} \left[Y_j - (\hat{C}_0 + \hat{C}_1X_{1j} + \ldots + \hat{C}_KX_{Kj}) \right]^2 = \sum_{j=1}^{N} \xi_j^2 \qquad (1\text{-}49)$$

Taking the partial derivatives of S with respect to $\hat{C}_0, \hat{C}_1, \ldots \hat{C}_K$, that is $\partial s/\partial \hat{C}_0$, $\partial s/\partial \hat{C}_1, \ldots \partial s/\partial \hat{C}_K$ and setting them equal to zero, we obtain the following set of equations:

$$N\hat{C}_0 + \left(\sum X_{1j}\right)\hat{C}_1 + \ldots + \left(\sum X_{Kj}\right)\hat{C}_K = \sum Y_j$$

$$\left(\sum X_{1j}\right)\hat{C}_0 + \left(\sum X_{1j}^2\right)\hat{C}_1 + \ldots + \left(\sum X_{1j}X_{Kj}\right)\hat{C}_K = \sum X_{1j}Y_j$$

$$\left(\sum X_{2j}\right)\hat{C}_0 + \left(\sum X_{1j}X_{2j}\right)\hat{C}_1 + \ldots + \left(\sum X_{2j}X_{Kj}\right)\hat{C}_K = \sum X_{2j}Y_j$$

$$\vdots$$

$$\vdots$$

$$\vdots$$

$$\vdots$$

$$\left(\sum X_{Kj}\right)\hat{C}_0 + \left(\sum X_{1j}X_{Kj}\right)\hat{C}_1 + \ldots + \left(\sum X_{Kj}^2\right)\hat{C}_K = \sum X_{Kj}Y_j \qquad (1\text{-}50)$$

Equation 1-50 can be expressed in matrix form as:

$$U\hat{C} = V$$

where

$$U = \begin{bmatrix} N & \sum X_{1j} & \cdots & \cdots & \sum X_{Kj} \\ \sum X_{1j} & \sum X_{1j}^2 & \cdots & \cdots & \sum X_{1j}X_{Kj} \\ \vdots & & & & \\ \vdots & & & & \\ \sum X_{Kj} & \sum X_{1j}X_{Kj} & & & \sum X_{Kj}^2 \end{bmatrix}$$

$$\hat{C} = \begin{bmatrix} \hat{C}_0 \\ \hat{C}_1 \\ \vdots \\ \vdots \\ \hat{C}_K \end{bmatrix} \qquad V = \begin{bmatrix} \sum Y_j \\ \sum X_{1j}Y_j \\ \vdots \\ \vdots \\ \sum X_{Kj}Y_j \end{bmatrix}$$

U is a symmetric matrix.

We can obtain estimates for the coefficients $\hat{C}_0, \hat{C}_1, \ldots \hat{C}_K$ by successive elimination or by solving for the inverse of U. That is

$$\hat{C} = U^{-1}V$$

where

U^{-1} = the inverse of U

After solving for $\hat{C}_0, \hat{C}_1, \ldots \hat{C}_K$, the estimates of the dependent variable observations $\hat{Y}_j$ can be obtained as follows:

$$\hat{Y}_j = \hat{C}_0 = \hat{C}_1X_{1j} + \ldots + \hat{C}_KX_{Kj} \tag{1-51}$$

The power equations have often been derived to calculate the parameters of experimental data. Such an equation can be expressed in the form:

$$Y = C_0 . X_1^{C_1} . X_2^{C_2} \ldots X_K^{C_K} \tag{1-52}$$

We can calculate the coefficients of the independent variables, if Equation 1-52 is linearized by taking its natural logarithm to give

$$\ln Y = \ln C_0 + C_1 \ln X_1 + C_2 \ln X_2 + \ldots + C_K \ln X_K \tag{1-53}$$

The coefficients $C_0, C_1, C_2, \ldots C_K$ can then be obtained by Gaussian elimination. Table 1-2 shows the variance table for linear multiple regression.

The coefficient of determination is

$$r^2 = 1 - \frac{SSE}{SST} \tag{1-54}$$

and the correlation coefficient is

$$r = \left(1 - \frac{SSE}{SST}\right)^{0.5} \tag{1-55}$$

The test statistic is the F - ratio, which we can define as:

$$F = \frac{MSR}{MSE} \tag{1-56}$$

POLYNOMIAL REGRESSION

Some engineering data are often poorly represented by a linear regression. If we know how Y depends on X, then we can develop some form of nonlinear regression [7], although total convergence of this iterative regression procedure can not be guaranteed. However, if the form of dependence is unknown, then we can treat Y as a general function of X by trigonometric terms (Fourier analysis) or polynomial function. The

Table 1-2
Analysis of Variance Table for Linear Multiple Regression

Source of Variance	Degree of Freedom	Sum of Squares	Mean Squares
Total	$N - 1$	$SST = \sum (Y_j - \overline{Y})^2$	$MST = \frac{SST}{N-1}$
Regression	K	$SSR = \sum (\hat{Y}_j - \overline{Y})^2$	$MSR = \frac{SSR}{K}$
Error	$N - K - 1$	$SSE = \sum (Y_j - \hat{Y}_j)^2$	$MSE = \frac{SSE}{N-K-1}$

least squares procedure can be readily extended to fit the data to an nth-degree polynomial:

$$Y = C_0 + C_1X + C_2X^2 + \ldots C_nX^n \tag{1-57}$$

where C_0, C_1, C_2, . . . C_n are constants.

For this case, the sum of the squares of the residuals is minimized:

$$S = \sum_{j=1}^{N} \left[Y_j - C_0 - C_1X_j - C_2X_j^2 - \ldots - C_nX_j^n \right]^2 \tag{1-58}$$

At the minimum, all the partial derivatives with respect to the chosen constants are zero; i.e.,

$$\frac{\partial S}{\partial C_0}, \frac{\partial S}{\partial C_1}, \ldots \frac{\partial S}{\partial C_n} - 0$$

This gives a system of (n+1) linear equations in (n+1) unknowns, C_0, C_1, . . . C_n

$$\frac{\partial S}{\partial C_0} = 0 = \sum_{j=1}^{N} 2(Y_j - C_0 - C_1X_j - C_2X_j^2 - \ldots - C_nX_j^n)(-1)$$

$$\frac{\partial S}{\partial C_1} = 0 = \sum_{j=1}^{N} 2(Y_j - C_0 - C_1X_j - C_2X_j^2 - \ldots - C_nX_j^n)(-X_j)$$

$$\frac{\partial S}{\partial C_2} = 0 = \sum_{j=1}^{N} 2(Y_j - C_0 - C_1X_j - C_2X_j^2 - \ldots - C_nX_j^n)(-X_j^2)$$

$$\vdots$$

$$\vdots$$

$$\vdots$$

$$\frac{\partial S}{\partial C_n} = 0 = \sum_{j=1}^{N} 2(Y_j - C_0 - C_1X_j - C_2X_j^2 - \ldots - C_nX_j^n)(-X_j^n)$$

(1-59)

The above equations are set to equal zero and can be rearranged in the following set of normal equations.

$$
\begin{aligned}
&C_0 N + C_1 \sum X_j + C_2 \sum X_j^2 + \ldots + C_n \sum X_j^n = \sum Y_j \\
&C_0 \sum X_j + C_1 \sum X_j^2 + C_2 \sum X_j^3 + \ldots + C_n \sum X_j^{n+1} = \sum X_j Y_j \\
&C_0 \sum X_j^2 + C_1 \sum X_j^3 + C_2 \sum X_j^4 + \ldots + C_n \sum X_j^{n+2} = \sum X_j^2 Y_j \\
&\vdots \\
&\vdots \\
&\vdots \\
&C_0 \sum X_j^n + C_1 \sum X_j^{n+1} + C_2 \sum X_j^{n+2} + \ldots + C_n \sum X_j^{2n} = \sum X_j^n Y_j
\end{aligned}
\tag{1-60}
$$

Equation 1-60 in matrix form, becomes

$$UC=V$$

$$
U = \begin{bmatrix}
N & \sum X_j & \sum X_j^2 & \cdots & \sum X_j^n \\
\sum X_j & \sum X_j^2 & \sum X_j^3 & \cdots & \sum X_j^{n+1} \\
\sum X_j^2 & \sum X_j^3 & \sum X_j^4 & \cdots & \sum X_j^{n+2} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\sum X_j^n & \sum X_j^{n+1} & \sum X_j^{n+2} & \cdots & \sum X_j^{2n}
\end{bmatrix}
$$

$$
C = \begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix}
\qquad
V = \begin{bmatrix} \sum Y_j \\ \sum X_j Y_j \\ \sum X_j^2 Y_j \\ \vdots \\ \sum X_j^n Y_j \end{bmatrix}
$$

Linear equations generated by polynomial regression can be ill-conditioned when the coefficients have very small and very large

numbers. This results in smooth curves that fit poorly. Ill-conditioning often happens if the degree of the polynomial is large and if the Y values cover a wide range. We can determine the error of polynomial regression by a standard error of the estimate as

$$\sigma^2 = \frac{\sum SSE}{N - n - 1} \tag{1-61}$$

where
n = the degree of polynomial
N = the number of data pairs

The coefficient of determination can be expressed as:

$$r^2 = 1 - \frac{SSE}{SST}$$

$$= 1 - \frac{\sum_{j=1}^{N} (Y_j - \hat{Y}_j)^2}{\sum_{j=1}^{N} (Y_j - \overline{Y}_j)^2} \tag{1-62}$$

The correlation coefficient is given by

$$r = \left(1 - \frac{SSE}{SST}\right)^{0.5} \tag{1-63}$$

The numerator of Equation 1-61 should continually decrease as the degree of the polynomial is raised. Alternatively, the denominator of Equation 1-61 causes σ^2 to increase as we move away from the optimum degree.

SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS

Analyses of physiochemical systems often give us a set of linear algebraic equations. Also, methods of solution of differential equations and nonlinear equations use the technique of linearizing the models. This requires repetitive solutions of sets of linear algebraic equations. Linear equations can vary from a set of two to a set having 100 or more equations. In most cases, we can employ Cramer's rule to solve a set of two or three linear algebraic equations. However, for systems of many linear

equations, the algebraic computation becomes too complex and may require other methods of analysis.

We will analyze two methods of solving a set of linear algebraic equations, namely Gauss elimination and Gauss-Seidel iteration methods. Gauss elimination is most widely used to solve a set of linear algebraic equations. Other methods of solving linear equations are Gauss-Jordan and LU decomposition. Table 1-3 illustrates the main advantages and disadvantages of using Gauss, Gauss-Jordan and LU decomposition.

Gaussian elimination method is based upon the principle of converting a set of N equations of N unknowns represented by

$$a_{1,1}X_1 + a_{1,2}X_2 + a_{1,3}X_3 + \ldots + a_{1,N}X_N = Y_1$$

$$a_{2,1}X_1 + a_{2,2}X_2 + a_{2,3}X_3 + \ldots + a_{2,N}X_N = Y_2$$

$$\vdots$$

$$\vdots$$

$$a_{N,1}X_1 + a_{N,2}X_2 + a_{N,3}X_3 + \ldots + a_{N,N}X_N = Y_N \qquad (1\text{-}64)$$

Table 1-3
Comparison of Three Methods of Linear Equations

Method	Advantages	Disadvantages
Gauss elimination	The most fundamental solution algorithm.	Solution of one set of linear equations at a time.
Gauss-Jordan	Basis for computing inverse; can solve multiple sets of equations.	Less efficient for a single set of equations.
LU decomposition	Efficient if one set of linear equations is repeatedly solved with different inhomogeneous terms (e.g., in the inverse power method.)	Less efficient and more cumbersome than Gauss elimination if used only once.

Source: S. Nakamura, *Applied Numercial Methods With Software,* Prentice-Hall Int. Ed., N.J., 1991. [2]

to a triangular set of the form

$$\begin{aligned} a_{1,1}X_1 + a_{1,2}X_2 + a_{1,3}X_3 + \ldots + a_{1,N}X_N &= Y_1 \\ a'_{2,2}X_2 + a'_{2,3}X_3 + \ldots + a'_{2,N}X_N &= Y'_2 \\ a''_{3,3}X_3 + \ldots + a''_{3,N}X_N &= Y''_3 \\ \vdots \\ a^{(N-1)}_{N,N}X_N &= Y^{(N-1)}_N \end{aligned} \tag{1-65}$$

The process involves converting the general form

$$aX = Y$$

to the triangular form

$$UX = Y'$$

where U is the upper triangular matrix. After completing the forward elimination process, we then employ the back substitution method starting with the last equation to obtain the solution of the set of the remaining equations. The solution of X_N is obtained from the last equation

$$X_N = \frac{Y^{(N-1)}_N}{a^{(N-1)}_{N,N}}$$

subsequent solutions are

$$X_{N-1} = \frac{\left[Y^{(N-2)}_{N-1} - a^{(N-1)}_{N-1,N}X_N\right]}{a^{(N-2)}_{N-1,N-1}}$$

$$\vdots$$

$$X_1 = \frac{\left[Y_1 - \sum_{j=2}^{N} a_{1,j}X_j\right]}{a_{1,1}} \tag{1-66}$$

This procedure completes the Gauss elimination. We can carry out the elimination process by writing only the coefficients and the matrix vector in an array as

$$\begin{matrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,N-1} & a_{1,N} & Y_1 \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,N-1} & a_{2,N} & Y_2 \\ \vdots & & & & & & \vdots \\ a_{N,1} & a_{N,2} & a_{N,3} & \cdots & a_{N,N-1} & a_{N,N} & Y_N \end{matrix} \tag{1-67}$$

The array after the forward elimination becomes

$$\begin{matrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,N-1} & a_{1,N} & Y_1 \\ 0 & a'_{2,2} & a'_{2,3} & \cdots & a'_{2,N-1} & a'_{2,N} & Y'_2 \\ 0 & 0 & a''_{3,3} & \cdots & a''_{3,N-1} & a''_{3,N} & Y''_3 \\ \vdots & & & & & & \\ 0 & 0 & 0 & \cdots & a^{(N-2)}_{N-1,N-1} & a^{(N-2)}_{N-1,N} & Y^{(N-2)}_{N-1} \\ 0 & 0 & 0 & \cdots & 0 & a^{(N-1)}_{N,N} & Y^{(N-1)}_{N} \end{matrix} \tag{1-68}$$

We can summarize the operations of Gauss elimination in a form suitable for a computer program as follows:

1. Augment the N × N coefficient matrix with the vector of right hand sides to form a N × (N–1) matrix.
2. Interchange the rows if required such that a_{11} is the largest magnitude of any coefficient in the first column.
3. Create zeros in the second through N^{th} rows in the first column by subtracting a_{i1}/a_{11} times the first row from the /ith row. Store the a_{i1}/a_{11} in a_{i1}, i=2, 3, . . . N.
4. Repeat Steps 2 and 3 for the second through the $(N-1)^{st}$ rows, putting the largest-magnitude coefficient on the diagonal by interchanging rows (considering only rows j to N). Then subtract a_{ij}/a_{jj} times the j^{th} row from the i^{th} row to create zeros in all the positions of the j^{th} column below the diagonal. Store the a_{ij}/a_{jj} in a_{ij}, i=j+1 . . . N. At the end of this step, the procedure is an upper triangular.
5. Solve for X_N from the N^{th} equation by

$$X_N = \frac{a_{N,N+1}}{a_{N,N}}$$

6. Solve for X_{N-1}, X_{N-2}, . . . ,X_1 from the $(N-1)^{st}$ through the first equation in turn by

$$X_i = \frac{a_{i,N+1} - \sum_{j=i+1}^{N} a_{ij} \cdot X_j}{a_{ii}}$$

The Guass-Seidel Iterative Method

Some engineering problems give sets of simultaneous linear equations that are diagonally dominant. This involves the computation of finite difference equations, derived from the approximation of partial differential equations. A diagonally dominant system of linear equations has coefficients on the diagonal that are larger in absolute value than the sum of the absolute values of the other coefficients.

Solving a system of N linear equations using the Gauss-Seidel iterative method, we can rearrange the rows so that the diagonal elements have larger absolute value than the sum of the absolute values of the other coefficients in the same row. This is defined as

$$AX=Y$$

We start with an initial approximation to the solution vector, $X^{(1)}$, and then calculate each component of $X^{(n+1)}$, for i = 1, 2, 3, . . . N by

$$X_i^{(n+1)} = \frac{Y_i}{a_{II}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} X_j^{(n+1)} - \sum_{j=i+1}^{N} \frac{a_{ij}}{a_{ii}} X_j^{n}, \quad n = 1, 2... \qquad (1\text{-}69)$$

A necessary condition for convergence is that

$$|a_{ii}| > \sum_{\substack{i=1, \\ j\neq 1}}^{N} |a_{ij}| \qquad i = 1, 2, \ldots N$$

When this condition exists, X^N will converge to the correct solution no matter what initial vector is used.

The Gauss-Seidel iterative method requires an initial approximation of the values of the unknowns X_1 to X_N. We use these values in Equation 1-69 to start calculation of new estimates of X's. Each newly calculated X_i replaces its previous value in subsequent calculations. The iteration continues until all the newly calculated X's converge to a

convergence criterion, ε, of the previous values. The Gauss-Seidel method is very efficient because of this faster convergence and should be used when the system is diagonally dominant. This method is widely used in the solution of engineering problems. However, the iterative method will not converge for all sets of equations or for all possible rearrangements of the equations. Assume that we are given a set of N simultaneous linear equations:

$$\begin{aligned} &a_{1,1}X_1 + a_{1,2}X_2 + \ldots + a_{1,N}X_N = a_{1,N+1} \\ &a_{2,1}X_1 + a_{2,2}X_2 + \ldots + a_{2,N}X_N = a_{2,N+1} \\ &\vdots \\ &a_{N,1}X_1 + a_{N,2}X_2 + \ldots + a_{N,N}X_N = a_{N,N+1} \end{aligned} \tag{1-70}$$

in which a_{ij}'s are constants.

Method of Solution

We first normalize the coefficients of Equation 1-70 to reduce the number of divisions required in the calculations. This is achieved by dividing all elements in row i by a_{ij}, i =1, 2, . . . N to produce an augmented coefficient matrix given by

$$\begin{bmatrix} 1 & a'_{1,2} & a'_{1,3} & \ldots & a'_{1,N} & a'_{1,N+1} \\ a'_{2,1} & 1 & a'_{2,3} & \ldots & a'_{2,N} & a'_{2,N+1} \\ \vdots & & & & & \\ a'_{N,1} & a'_{N,2} & a'_{N,3} & \ldots & 1 & a'_{N,N+1} \end{bmatrix} \tag{1-71}$$

where $a'_{i,j} = \dfrac{a_{i,j}}{a_{i,i}}$

The approximation to the solution vector after the k^{th} iteration is

$$X_k = [X_{1,k}, X_{2,k} \ldots X_{N,K}]$$

This is modified by the algorithm

$$X_{i,k+1} = a'_{i,N+1} - \sum_{j=1}^{i-1} a'_{ij} . X_{j,K+1} - \sum_{j=i+1}^{N} a'_{ij} . X_{jk} \tag{1-71}$$

where i = 1, 2, . . . N

The next approximation yields

$$X_{K+1} = [X_{1,K+1}, X_{2,K+1} \ldots X_{N,K+1}]$$

The iteration subscript K can be omitted because the new values $X_{i,K+1}$ replace the old values during computation and Equation 1-71 becomes

$$X_i = a'_{i,N+1} - \sum_{\substack{i=1 \\ j \neq 1}}^{N} a'_{ij} . X_j, \qquad i = 1, 2, \ldots N \tag{1-72}$$

The X_i values calculated by iterating with Equation 1-71 will converge to the solution of Equation 1-72, if the convergence criterion is

$$\left| X_{i,K+1} - X_i \right| < \varepsilon, \qquad i = 1, 2, \ldots N \tag{1-73}$$

provided that no element of the solution vector may have its magnitude changed by an amount greater than ε as a result of one Gauss-Seidel iteration. We can introduce an upper limit on the number of iterations, K_{max}, to terminate the iterative sequence when convergence does not occur.

SOLUTION OF NONLINEAR EQUATIONS

Solving problems in chemical engineering and science often requires finding the real root of a single nonlinear equation. Examples of such computations are in fluid flow, where pressure loss of an incompressible turbulent fluid is evaluated. The Colebrook [8] implicit equation for the Darcy friction factor, f_D, for turbulent flow is expressed

$$\frac{1}{f_D^{0.5}} = -2 \log \left\{ \frac{\varepsilon / D}{3.7} + \frac{2.51}{N_{Re} f_D^{0.5}} \right\} \tag{1-74}$$

where ε/D is the pipe roughness in feet, f_D is the Darcy friction factor, which is four times the Fanning friction factor, and N_{Re} is the Reynolds number. Equation 1-74 is nonlinear and involves trial and error solutions to achieve a solution for f_D; i.e., the root of the equation.

Equation 1-74 can be further expressed as:

$$F(f_D) = \frac{1}{f_D^{0.5}} + 2 \log \left\{ \frac{\varepsilon / D}{3.7} + \frac{2.51}{N_{Re} f_D^{0.5}} \right\} \tag{1-75}$$

In the form of natural logarithm, Equation 1-75 is

$$F(f_D) = \frac{1}{f_D^{0.5}} + 0.86858 \ln\left\{\frac{\varepsilon / D}{3.7} + \frac{2.51}{N_{Re} f_D^{0.5}}\right\} \tag{1-76}$$

The derivative of Equation 1-76 is

$$F'(f_D) = -\frac{0.5}{f_D^{1.5}} - \frac{4.03329}{\left[N_{Re} f_D^{1.5} \frac{\varepsilon}{D} + 9.287 f_D\right]} \tag{1-77}$$

In thermodynamics, the pressure-volume temperature relationships of real gases are described by the equation of state. The compressibility factor from the reduced pressure and temperature can be rearranged from Redlich and Kwong [9] to two constant state equations in the form

$$Z = 1 + A - \frac{AB}{A + Z} + \frac{A^2 B}{AZ + Z^2} \tag{1-78}$$

where

$$A = \frac{0.0867 p_r}{T_r} \quad \text{and} \quad B = \frac{4.934}{T_r^{1.5}}$$

Equation 1-78 is also nonlinear and can be expressed as:

$$F(Z) = Z - 1 - A + \frac{AB}{A + Z} - \frac{A^2 B}{AZ + Z^2} \tag{1-79}$$

The derivative of Equation 1-79 is

$$F'(Z) = 1 - \frac{AB}{(A + Z)^2} + \frac{A^2 B(A + 2Z)}{(AZ + Z^2)^2} \tag{1-80}$$

For multicomponent separations, it is often necessary to estimate the minimum reflux ratio of a fractionating column. A method developed for this purpose by Underwood [10] requires the solution of the equation

$$f(\theta) = \sum_{i=1}^{N} \frac{\alpha_i x_i}{(\alpha_i - \theta)} - 1 + q = 0 \tag{1-81}$$

where i = 1, 2, . . . N

and

N = the number of components in the feed
x_i = the mole fraction of component i
α_i = the relative volatility of component i
q = the thermal condition of the feed

Equation 1-81 is highly nonlinear and must be solved for θ, the Underwood parameter, or the root of the equation. It has a singularity at each value of $\theta = \alpha_i$. If we can determine with some precision the degree of vaporization of the feed and the distillation composition, then we can also obtain a single value of θ by an iterative solution. The value of θ is generally bounded between the relative volatilites of the light and heavy keys.

That is

$$\theta_{LK} < \theta < \theta_{HK}$$

Gjumbir and Olujic [11], Tao [12], and Sargent [13] have published articles for solving nonlinear problems. We will discuss some of the methods used in solving nonlinear equations, and employ one of the methods to solve Equation 1-66. Equation 1-81 uses the bisection method to solve for θ in Chapter 7.

The Bisection Method

The method assumes that f(x) is continuous over the interval (x_1,x_2) with $f(x_1)$ and $f(x_2)$ having opposite signs; i.e., $f(x_1).f(x_2) < 0$. There is at least one root of f(x) = 0 in this interval. If $f(x_1) < 0$, then $f(x_2) > 0$. Therefore the interval between these two points can be bisected. If I_1 represents the subinterval (x_2,x_3) at whose end points f(x) takes opposite signs and $x_3 = (x_1 + x_2)/2$, I_1 can be bisected to give an interval I_2, at whose endpoints f(x) still has opposite signs. We can continue with this process until a root of the equation is found.

The midpoint of interval I_m, is

$$x_{m+1} = (x_{m-1)} + x_m)/2 \qquad (1\text{-}82)$$

which is selected as an approximation to the root. The advantages are its simplicity and the number of iterations that can be predicted, because the chosen interval is halved during each iteration.

$$i = \log\left(\frac{x_2 - x_1}{\varepsilon}\right) \Big/ \log 2 \tag{1-83}$$

where i is the number of iterations and ε is the tolerance.

Algorithm for the Bisection Method

Determining a root of $f(x) = 0$, accurate within a specified tolerance value, requires given values of x_1 and x_2 such that $f(x_1)$ and $f(x_2)$ are of opposite signs. i is the iteration index, imax is the maximum number of iterations, ε is the tolerance, and x_3 is the subinterval midpoint.

Step 1. Set $i = 1$

Step 2. (i) Set $x_3 = x_1 + (x_2 - x_1)/2$
(ii) If $|x_3 - x_1| < \varepsilon$ then go to Step 3
(iii) If $f(x_1)f(x_3) > 0$, then $x_1 = x_3$, else $x_2 = x_3$ (set new interval bounds)
(iv) $i = i + 1$
(v) If $i >$ imax, then quit, else repeat Step 2i – v

Step 3. Output $x_3, f(x_3)$

The final value of x_3 approximates the root. The method may give a false root if f(x) is discontinuous on (x_1, x_2).

Method of Linear Interpolation (Regula-Falsi Method)

An initial guess, x_1, is made and function, f_1, is calculated. We then evaluate the function f_2 at another point, x_2, such that the two functions have opposite signs; i.e., $f_1 f_2 < 0$. We can calculate a new interval value, x_3, by linear interpolation, and then evaluate the function f_3.

From similar triangles,

$$\frac{x_2 - x_3}{x_2 - x_1} = \frac{f_2}{f_2 - f_1} \tag{1-84}$$

rearranging, we have

$$x_3 = x_2 - \frac{f_2}{f_2 - f_1}(x_2 - x_1) \tag{1-85}$$

Generalizing, we obtain the recursive algorithm

$$x_{n+1} = x_n - \frac{f_n}{f_n - f_{n-1}}(x_n - x_{n-1}) \tag{1-86}$$

where n = 1, 2, 3, . . . N

If $f_n f_{n+1} < 0$, then the righthand interval contains the root, and x_{n+1} becomes the new x_{n-1} for the next iteration. However, if $f_{n-1} f_{n+1} < 0$, the true root of the equation lies between x_{n-1} and x_{n+1}. x_{n+1} then becomes the new x_n for the next iteration.

Algorithm for the Modified Linear Interpolation Method

Step 1. Set SAVE = $f(x_1)$; set $F_1 = f(x_1)$ and $F_2 = f(x_2)$

Step 2. DO WHILE $|x_1 - x_2| \geq$ tolerance value 1, or

$|f(x_3)| \geq$ tolerance value 2

Step 3. $x_3 = x_2 - F_2(x_2 - x_1) / (F_2 - F_1)$
If $f(x_3)$ of opposite sign to F_1, then
$x_2 = x_3$, $F_2 = F_3$,
If $f(x_3)$ of same sign as SAVE; then
$F_1 = F_2/2$
ENDIF
ELSE
$x_1 = x_3$, $F_1 = f(x_3)$
If $f(x_3)$ of same sign as SAVE, then
$F_2 = F_2/2$
ENDIF
ENDIF
SAVE = $f(x_3)$

The Newton-Raphson Method

The Newton-Raphson method is widely used in finding the root of nonlinear equations. This method uses the derivative of f(x) at x to estimate a new value of the root. The tangent at x is then extended to intersect the x-axis, and the value of x at this intersection is the new estimate of the root. The desired precision is reached by iteration.

Suppose that f(x), f'(x) and f''(x) are continuous and that f(x) and f'(x) do not vanish for the same value of x, if x^r is an approximation to the root of f(x) = 0.

The second mean value theorem can be expressed as:

$$0 = f(x^r + h) = f(x^r) + hf'(x^r) + \frac{h^2}{2} f''(x^r + O_2 h) \tag{1-87}$$

Neglecting higher order terms, we have

$$h = -\frac{f(x^r)}{f'(x^r)} \tag{1-88}$$

Therefore, the next approximation to the root is given by

$$x^{r+1} = x^r - \frac{f(x^r)}{f'(x^r)} \tag{1-89}$$

The procedure is repeated until $|x^r| < \varepsilon$, where ε is a preset tolerance.

This method converges much faster than the bisection technique. However, its rate of convergence does not necessarily correlate with the nearness of the initial starting of the root. The main advantages of this technique are that it converges rapidly (i.e., quadratic convergence) and needs only a single starting point. Its disadvantages are sensitivity to local maxima and minima and a potentially narrow convergence range.

Algorithm for the Newton-Raphson Method

Step 1. Set i = 1

Step 2. $x_i = -f(x_i)/f'(x_i)$

Step 3. If $|x_{i+1} - x_i| < \varepsilon$, then print root x_{i+1}, quit

Step 4. i = i+1

If i > imax, then quit, else repeat Steps 2-4

The Secant Method

In cases where the first derivative cannot easily be determined by the Newton-Raphson method, a simple method to approximate the first derivative can be used. The secant method can be determined from two prior estimates.

$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \tag{1-90}$$

and the iteration process becomes

$$x_{i+1} = x_i + f(x_{i+1}) \frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})} \tag{1-91}$$

The advantages of this method are that it offers rapid convergence without requiring the first derivative. Convergence is between linear and quadratic, i.e., a power ranging from 1 to 2. The value is an approximation of the tangent of the secant. The value depends upon the steepness of the curve. Because the denominator always approaches zero near the root, this method is prone to instability. It also requires two initial estimates to start.

INTERPOLATIONS

Experimental and physical property data sometimes require values of their unknown functions that correspond to certain values of their independent variables. In certain cases, we may want to determine the behavior of the function. Alternatively, we may want to approximate other values of the function at values of the independent variables that are not tabulated. We can achieve these objectives either by interpolation or extrapolation of a polynomial that fits a selected set of points of both variables (x_i, $f(x_i)$). Also, we can assume after finding a polynomial that fits a selected set of points (x_i, $f(x_i)$ that the polynomial and the function behave over a given interval in question. The values of the polynomial will be estimates of the values of the unknown function.

Generally, the experimental data are approximated by a polynomial, the degree of which can often be calculated by constructing a difference table. The difference column that gives approximate constant value shows the degree of the polynomial that can be fitted to the data. When the polynomial is of the first degree, we have a linear interpolation. For polynomials of higher degrees, we can approximate functions if we construct a table with wider spacing. Such a table is known as a difference table.

A Difference Table

We can obtain a table of values, if the dependent variable f(x) is a function of the independent variable X. If we let h be the uniform difference in the x-values, $h = \Delta x$, we can define the first differences of the function as:

$$\begin{aligned}
\Delta f(x_1) &= f(x_2) - f(x_1) \\
\Delta f(x_2) &= f(x_3) - f(x_2) \\
&\vdots \\
\Delta f(x_i) &= f(x_{i+1}) - f(x_i)
\end{aligned} \tag{1-92}$$

or

$$\Delta f_i = f_{i+1} - f_i \tag{1-93}$$

The second difference is the difference of the first differences and can be expressed as:

$$\Delta^2 f(x) = \Delta\big(\Delta f(x_1)\big) \tag{1-94}$$

$$\begin{aligned}
&= \Delta\big(\Delta f(x_2) - \Delta f(x_1)\big) \\
&= \big(f(x_3) - f(x_2)\big) - \big(f(x_2) - f(x_1)\big)
\end{aligned} \tag{1-95}$$

or

$$\begin{aligned}
\Delta^2 f_1 &= (f_3 - f_2) - (f_2 - f_1) \\
&= f_3 - 2f_2 + f_1
\end{aligned} \tag{1-96}$$

$$\Delta^2 f_i = f_{i+2} - 2f_{i+1} + f_i \tag{1-97}$$

$$\begin{aligned}
\Delta^3 f_1 &= \Delta(\Delta^2 f_2 - \Delta^2 f_1) \\
&= (f_4 - 2f_3 + f_2) - (f_3 - 2f_2 + f_1) \\
&= f_4 - 2f_3 + f_2 - f_3 + 2f_2 - f_1
\end{aligned} \tag{1-98}$$

$$\Delta^3 f_1 = f_4 - 3f_3 + 3f_2 - f_1 \tag{1-99}$$

$$\Delta^3 f_i = f_{i+3} - 3f_{i+2} + 3f_{i+1} - f_i \tag{1-100}$$

We then have a general formula given by

$$\Delta^n f_i = f_{i+1} - \binom{n}{1} f_{i+n-1} + \binom{n}{2} f_{i+n-2} - \binom{n}{3} f_{i+n-3} + \ldots$$

$$= f_{i+1} - n f_{i+n-1} + \frac{n!}{2!(n-2)!} f_{i+n-2} - \frac{n!}{3!(n-3)!} f_{i+n-3} + \ldots$$

$$= f_{i+1} - n f_{i+n-1} + \frac{n(n-1)}{2!} f_{i+n-2} - \frac{n(n-1)(n-2)}{3!} f_{i+n-3} + \ldots \qquad (1\text{-}101)$$

Equation 1-101 is the array of coefficients in the binomial expansion.

Table 1-4 shows how the differences in the dependent variables are arranged.

Interpolating Polynomials

When the tabulated function resembles a polynomial of a constant value in the n^{th} order differences, we can approximate the function by constructing a polynomial. Various formulas have been expressed for the n^{th} degree polynomial that passes through n + 1 pairs of points (x_i,f_i), where i = 1, 2 . . . n + 1.

When the data points are available at equal intervals of the independent variable, we can use the Newton-Gregory forward polynomial.

Table 1-4
Diagonal Difference Table

x	f(x)	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x_0	f_0				
		Δf_0			
x_1	f_1		$\Delta^2 f_0$		
		Δf_1		$\Delta^3 f_0$	
x_2	f_2		$\Delta^2 f_1$		$\Delta^4 f_0$
		Δf_2		$\Delta^3 f_1$	
x_3	f_3		$\Delta^2 f_2$		
		Δf_3			
x_4	f_4				

$$P_n(x_s) = f_0 + \binom{s}{1}\Delta f_0 + \binom{s}{2}\Delta^2 f_0 + \binom{s}{3}\Delta^3 f_0$$

$$+ \binom{s}{4}\Delta^4 f_0 + \ldots + \binom{s}{n}\Delta^n f_0$$

$$= f_0 + s\Delta f_0 + \frac{s(s-1)}{2!}\Delta^2 f_0 + \frac{s(s-1)(s-2)}{3!}\Delta^3 f_0$$

$$+ \frac{s(s-1)(s-2)(s-3)}{4!}\Delta^4 f_0$$

$$+ \ldots + \frac{1}{n!}s(s-1)(s-2)\ldots(s-n+1) \qquad (1\text{-}102)$$

where $x = x_0 + sh$ and $h = x_{i+1} - x_i = x_2 - x_1$

The Newton-Gregory backward polynomial can be expressed as:

$$P_n(x_s) = f_0 + \binom{s}{1}\nabla f_0 + \binom{s+1}{2}\nabla^2 f_0 + \binom{s+2}{3}\nabla^3 f_0$$

$$+ \binom{s+3}{4}\nabla^4 f_0 + \ldots + \binom{s+n-1}{n}\nabla^n f_i$$

$$= f_0 + s\nabla f_0 + \frac{s(s+1)}{2!}\nabla^2 f_0 + \frac{s(s+1)(s+2)}{3!}\nabla^3 f_0$$

$$+ \frac{s(s+1)(s+2)(s+3)}{4!}\Delta^4 f_0 + \ldots$$

$$+ \frac{1}{n!}(s+n-1)(s+n-2)\ldots s(s+1)\nabla^n f_i \qquad (1\text{-}103)$$

where s is a local coordinate defined by

$$s = \frac{x - x_i}{h}$$

$\binom{s+n-1}{n}$ is a binomial coefficient and $\nabla^n f_i$ is the backward difference.

Stirling central differences represent a horizontal line using averages of the differences; i.e.,

$$\delta f_i = f_{i+1/2} - f_{i-1/2}$$

or

$$\delta f_{i+1/2} = f_{i+1} - f_i$$

where

$$f_{i+1/2} = f\left(x_i + \frac{h}{2}\right)$$

$$p_n(x) = f_0 + \binom{s}{1}\frac{\Delta f_{-1} + \Delta f_0}{2} + \frac{\binom{s+1}{2} + \binom{s}{2}\Delta^2 f_{-1}}{2}$$

$$+ \binom{s+1}{3}\frac{\Delta^3 f_{-2f} + \Delta^3 f_{-1}}{2} + \frac{\binom{s+2}{4} + \binom{s+1}{4}}{2}\Delta^4 f_{-2} + \quad (1\text{-}104)$$

The forward and the backward difference approximations give the derivatives of the Newton interpolation polynomial at the edges of the interpolation range. However, the central difference is derived from the Newton interpolation at the center of the range of interpolation. Accuracy of an interpolation formula based on equispaced points is highest at the center of the interpolation range. Therefore, the central difference interpolation formula is always more accurate than the forward or backward difference approximations.

SOLUTION OF INTEGRATION

We frequently use numerical techniques to integrate a function given in both analytical and tabular forms. For instance, we can use an integral method to determine the volumetric rate of a gas through a duct from the linear velocity distribution. In fluid mixing with residence time distribution theory, Danckwerts [14] showed that the fraction of material in the outlet stream that has been in the system for a period between t and t+dt is equal to Edt. E is a function of t, and E(t) is the residence time distribution function. We can express E(t) in integral form as:

$$\int_0^\infty E(t)dt = 1 \tag{1-105}$$

The average time spent by material flowing at a rate, q, through a volume, V, equals V/q.

$$\text{The mean residence time, } \bar{t} = \frac{V}{q}$$

We can also express Equation 1-105 in the form of dimensionless time where $\theta = tq/V$, and this becomes:

$$\int_0^\infty E(\theta)d\theta = 1 \tag{1-106}$$

Figure 1-6 shows a residence time distribution from a tracer experiment studying the mixing characteristics of a nozzle-type reactor [15] that behaves nonideally.

Numerical integral methods can be used to calculate areas under this or similar distribution functions.

The Trapezoidal Rule

This is a numerical integration method derived by integrating the linear interpolation formula. It is expressed as:

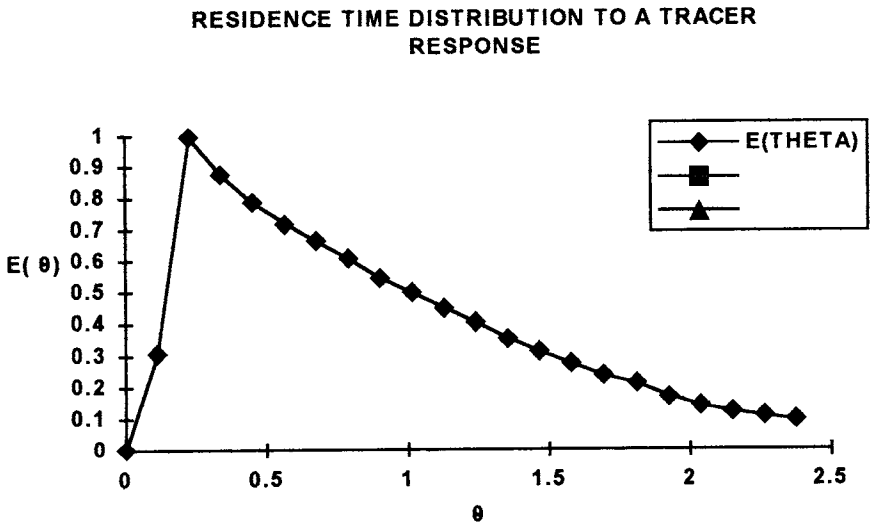


Figure 1-6. Residence time distribution in a nozzle type reactor. Source: A. K. Coker, *A Study of Fast Reactions in Nozzle Type Reactors,* Ph.D. thesis, Univ. of Aston, Birmingham, U.K., 1985.

$$I = \int_{x_i}^{x_{i+1}} f(x)dx = \frac{f(x_i) + f(x_{i+1})}{2}(\Delta x) \qquad (1\text{-}106)$$

$$= \frac{h}{2}(f_i + f_{i+1})$$

For (a, b) subdivided into sub-intervals of size h, we can express the area as:

$$I = \int_a^b f(x)dx = \sum_{i=1}^{n} \frac{h}{2}(f_i + f_{i+1})$$

$$= \frac{h}{2}(f_1 + 2f_2 + 2f_3 + \ldots + 2f_n + f_{n+1}) + E \qquad (1\text{-}107)$$

$$= \text{width} \times \text{average height}$$

E represents the local error, $E \cong -\left(\frac{b-a}{12}\right)h^2 f''(\varepsilon)$ where $a \le \varepsilon \le b$

Simpson's 1/3 Rule

Simpson's 1/3 rule is based on quadratic polynomial interpolation. For a quadratic integrated over two Δx intervals that are of uniform width or panels, we can express the area as:

$$I = \int_a^b f(x)dx = \frac{h}{3}(f_1 + 4f_2 + 2f_3 + 4f_4 +$$

$$2f_5 + \ldots + 2f_{n-1} + 4f_n + f_{n+1}) + E \qquad (1\text{-}108)$$

$$= \text{width} \times \text{average height}$$

$$E \cong -\frac{(b-a)}{180}h^4 f^{iv}(\varepsilon), \quad a \le \varepsilon \le b$$

Simpson's 3/8 Rule

Simpson's 3/8 rule is derived by integrating a third-order polynomial interpolation formula. For a domain (a,b) divided into three intervals, it is expressed as:

$$I = \int_a^b f(x)dx = \frac{3h}{8}(f_1 + 3f_2 + 3f_3 + 2f_4 + 3f_5 + 3f_6$$

$$+ \ldots + 2f_{n-2} + 3f_{n-1} + 3f_n + f_{n+1}) + E \qquad (1\text{-}109)$$

$$= \text{width} \times \text{average height}$$

$$E \cong -\frac{(b-a)}{80} h^4 f^{iv}(\varepsilon), \quad a \le \varepsilon \le b$$

Gaussian Quadrature

Gauss quadratures are numerical integration methods that employ Legendre points. Gauss quadrature cannot integrate a function given in a tabular form with equispaced intervals. It is expressed as:

$$I = \int_{-1}^{1} f(x)dx = af(x_1) + bf(x_2) + E \qquad (1\text{-}110)$$

where the limits of integration are a to b. To use the tabulated Gaussian quadrature parameters, we must change the interval of integration to (–1,1).

If we let

$$t = \frac{(b-a)x + b + a}{2} \quad \text{and} \quad dt = \left(\frac{b-a}{2}\right)dx$$

then

$$\int_a^b f(t)dt = \int_{-1}^{1} f(t)\left(\frac{dt}{dx}\right)dx$$

$$= \left(\frac{b-a}{2}\right)\int_{-1}^{1} f\left\{\frac{(b-a)x + b + a}{2}\right\}dx \qquad (1\text{-}111)$$

Table 1-5 summarizes the advantages and disadvantages of numerical integration techniques.

SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS

In certain cases, we may need to find out the behavior of many dynamic and physical processes. This is often expressed mathematically

Table 1-5
Advantages and Disadvantages of Numerical Integration Techniques

Method	Advantages	Disadvantages
Trapezoidal rule	Simplicity. Optimal for improper integrals.	Needs a large number of subintervals for good accuracy.
Simpson's 1/3 rule	Simplicity. Higher accuracy than the trapezoidal rule.	Even number of intervals only.
Simpson's 3/8 rule	Same order of accuracy as the 1/3 rule.	Intervals or panels in multiples of three only.
Gaussian quadrature	Functional data at two end points are not used.	Data points are not equispaced.

by ordinary differential equations. The solutions of these equations are of great value to engineers and scientists. Many differential equations can be solved by well-known analytical methods, although many physically differential equations are impossible to solve analytically. Numerical techniques can be developed to solve these equations. We shall consider the numerical techniques for solving differential equations.

N^{th} Order Ordinary Differential Equations

We consider the solutions of the Nth order differential equations of the form:

$$F\left(x, y, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \frac{d^3y}{dx^3}, \ldots \frac{d^{n-1}y}{dx^{n-1}}, \frac{d^ny}{dx^n}\right) = 0 \qquad (1\text{-}112)$$

Equation 1-112 has the highest derivative of the order n, and is ordinary because there is only one independent variable, x. We can obtain a unique solution when some additional information such as values of y(x) and its derivatives at some specific values of x are known. Therefore, for an N^{th} order equation, we require such N conditions to arrive at the unique solution y(x). If all N conditions are known at the same value of x, we can classify the problem as an initial value problem. However, when more than one value of x is involved, the problem is classified as a boundary value problem.

Solution of First Order Ordinary Differential Equations

We express a first order equation as:

$$\frac{dy}{dx} = f(x, y) \tag{1-113}$$

and we require a solution y(x) that satisfies Equation 1-113 and one initial condition. Here, we can subdivide the interval in the independent variable in x into steps over which a solution is required (a,b). The value of the exact solution y(x) is then approximated at N+1 evenly spaced values of x; i.e., $(x_0, x_1, \ldots x_{n-1}, x_n)$. The step size, h, is expressed as:

and

$$x_i = x_0 + ih, \quad i = 0, 1, \ldots n$$

If we let the true solution y(x) be $y(x_i)$, and the computed approximations of y(x) at these same points be y_i, so that

$$y_i = y(x_i)$$

then, the exact derivative dy/dx can be approximated by $f(x_i,y_i)$ and represented as f_i such that

$$f_i = f(x_i, y_i) = f\big(x_i, y(x_i)\big) \tag{1-114}$$

The difference between the computed value y_i and the true value $y(x_i)$ is ε_i and can be expressed as:

$$\varepsilon_i = y_i - y(x_i) \tag{1-115}$$

ε_i is the local truncation error.

Taylor Series Expansion

We can develop a relation between x and y by finding the coefficients of the Taylor series. Expanding y about the point $x = x_0$, we obtain

$$y(x) = y(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)(x - x_0)^2}{2!} + \frac{f'''(x_0)(x - x_0)^3}{3!} + \ldots \tag{1-116}$$

where $f'(x)$ represents $\frac{d}{dx} f(x)$

If we let $x - x_0 = h$

We can express the series as:

$$y(x) = y(x_0) + hf'(x_0) + \frac{f''(x_0)h^2}{2!} + \frac{f'''(x_0)h^3}{3!} + \ldots \qquad (1\text{-}117)$$

Since $y(x_0)$ is our initial condition, the first term is known from the initial condition $y(0) = 1$. The error term of the Taylor series after the h^4 term is

$$E = \frac{y^v \varepsilon h^5}{5!} \qquad 0 < \varepsilon < h$$

Euler and Modified Euler Methods

Using the Taylor's series,

$$y(x_0 + h) = y(x_0) + hf'(x_0) + \frac{f''(\varepsilon)h^2}{2} \qquad (1\text{-}118)$$

$$x_0 < \varepsilon < x_0 + h$$

The value of $y(x_0)$ is given by the initial condition and $f'(x_0)$ is evaluated from $f(x_0,y_0)$, given by the differential equation

$$\frac{dy}{dx} = f(x, y)$$

The Euler method can be expressed as:

$$y_{n+1} = y_n + hf'_n + O(h^2) \text{ error} \qquad (1\text{-}119)$$

For the modified Euler method, we expand the Taylor series as

$$y_{n+1} = y_n + f'_n h + \frac{f''_n}{2} h^2 + \frac{f'''_n}{6} h^3, \qquad x_n < \varepsilon < x_{n+h} \qquad (1\text{-}120)$$

Replacing the second derivative by the forward difference approximation for f" that is

$$f'' = \frac{f'_{n+1} - f'_n}{h}$$

having an error of O(h), we have

$$y_{n+1} = y_n + h\left\{f'_n + \frac{1}{2}\left[\frac{f'_{n+1} - f'_n}{h} + O(h)\right]h\right\} + O(h^3) \qquad (1\text{-}121)$$

$$y_{n+1} = y_n + h\left\{f'_n + \frac{1}{2}f'_{n+1} - \frac{1}{2}f'_n\right\} + O(h^3)$$

$$= y_n + \frac{h(f'_n + f'_{n+1})}{2} + O(h^3) \qquad (1\text{-}122)$$

Runge-Kutta Methods

The solution of a differential equation by direct Taylor's expansion cannot be easily obtained, if we retain the derivatives of higher order. We can develop one-step procedures that involve only first-order derivative evaluations and produce results equivalent in accuracy to higher order Taylor formulas. These algorithms are named the Runge-Kutta methods after the German mathematicians Runge and Kutta. The second-order Runge-Kutta algorithm for the first-order differential equation can be expressed as:

$$\frac{dy}{dx} = f(x, y)$$

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + h, y_i + k_i)$$

$$y_{n+1} = y_n + \frac{1}{2}(k_1 + k_2) + O(h^3) \qquad (1\text{-}123)$$

The third-order Runge-Kutta method is

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right)$$

$$k_3 = hf(x_i + h, y_i + 2k_2 - k_1)$$

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 4k_2 + k_3) + O(h^4) \qquad (1\text{-}124)$$

The fourth-order Runge-Kutta method is widely used in computer solutions of differential equations.

$$
\begin{aligned}
k_1 &= hf(x_i, y_i) \\
k_2 &= hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) \\
k_3 &= hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right) \\
k_4 &= hf(x_i + h, y_i + k_3) \\
y_{n+1} &= y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) + O(h^5)
\end{aligned}
\tag{1-125}
$$

k_1, k_2, k_3, k_4 are approximate derivative values computed on the interval $x_i \le x \le x_{i+1}$ and h is the step size.

$$x_{i+1} = x_i + h$$

The local error term for the fourth-order Runge-Kutta is $O(h^5)$.

Runge-Kutta-Gill Method

Runge-Kutta-Gill method is the most widely used single-step method for solving ordinary differential equations.

$$
\begin{aligned}
k_1 &= hf(x_i, y_i) \\
k_2 &= hf\left(x_i + \frac{h}{2}, y_i + \frac{1}{2}k_1\right) \\
k_3 &= hf\left(x_i + \frac{h}{2}, y_i + \left[\frac{-1}{2} + \frac{1}{\sqrt{2}}\right]k_1 + \left[1 - \frac{1}{\sqrt{2}}\right]k_2\right) \\
k_4 &= hf\left(x_i + h, y_i - \frac{1}{\sqrt{2}}k_2 + \left[1 + \frac{1}{\sqrt{2}}\right]k_3\right) \\
y_{n+1} &= y_n + \frac{1}{6}\left(k_1 + 2\left[1 - \frac{1}{\sqrt{2}}\right]k_2 + 2\left[1 + \frac{1}{\sqrt{2}}\right]k_3 + k_4\right) + O(h^5)
\end{aligned}
\tag{1-126}
$$

Runge-Kutta-Gill method provides an efficient algorithm for solving a system of first-order differential equations and makes use of much less computer memory when compared with other numerical methods.

Runge-Kutta-Merson Method

Runge-Kutta-Merson outlines a process for deciding the step size for a better predetermined accuracy. For this method, five functions are evaluated at every step. The algorithm is

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf\left(x_i + \frac{h}{3}, y_i + \frac{k_1}{3}\right)$$

$$k_3 = hf\left(x_i + \frac{h}{3}, y_i + \frac{k_1}{6} + \frac{k_2}{6}\right)$$

$$k_4 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{8} + \frac{3k_3}{8}\right)$$

$$k_5 = hf\left(x_i + h, y_i + \frac{k_1}{2} - \frac{3k_3}{2} + 2k_4\right)$$

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 4k_4 + k_5) + O(h^5) \qquad (1\text{-}127)$$

We can estimate the local error from a weighted sum of the individual estimate.

$$E = \frac{1}{30}(2k_1 - 9k_3 + 8k_4 - k_5) \qquad (1\text{-}128)$$

Multistep Methods

The multistep methods use past values of y and $y' = f(x,y)$ to construct a polynomial that approximates the derivative function and extrapolates this into the next interval. We can achieve an accurate estimate of y_{n+1} based on a polynomial that fits past values of the gradient. Also, we end with a slope equal to that at the predicted point. This forms the basis of predictor-corrector formulae.

Fourth Order Milne's Method

Values of y_n, y_{n-1}, y_{n-2} and y_{n-3} are required to calculate y_{n+1}. Milne's method uses Newton-Cotes formula for the predictor and Simpson's rule for the corrector.

Predictor

$$y^p_{n+1} = y_{n-3} + \frac{4h}{3}\left[2f(x_n, y_n) - f(x_{n-1}, y_{n-1}) + 2f(x_{n-2}, y_{n-2})\right] + O(h^5) \quad (1\text{-}129)$$

Corrector

$$y_{n+1} = y_{n-1} + \frac{h}{3}\left[f(x_{n+1}, y^p_{n+1}) + 4f(x_n, y_n) + f(x_{n-1}, y_{n-1})\right] + O(h^5) \quad (1\text{-}130)$$

Local truncation error

$$O(h^5) = -\frac{h^5}{90} f^v(\varepsilon), \qquad x_{n-1} < \varepsilon < x_{n+1}$$

Adams-Moulton Fourth-Step Method

Predictor

$$y^p_{n+1} = y_n + \frac{h}{24}\left[55f(x_n, y_n) - 59f(x_{n-1}, y_{n-1}) + 37f(x_{n-2}, y_{n-2}) - 9f(x_{n-3}, y_{n-3})\right] + O(h^5) \quad (1\text{-}131)$$

$$n = 3, 4, \ldots m - 1$$

Corrector

$$y_{n+1} = y_n + \frac{h}{24}\left[9f(x_{n+1}, y^p_{n+1}) + 19f(x_n, y_n) - 5f(x_{n-1}, y_{n-1}) + f(x_{n-2}, y_{n-2})\right] + O(h^5) \qquad n = 2, 3, \ldots m - 1 \quad (1\text{-}132)$$

Local truncation error

$$O(h^5) = -\frac{19}{720} h^5 f^v(\varepsilon), \qquad x_{n-2} < \varepsilon < x_{n+1}$$

The predictor calls for four previous values in Adams-Moulton and Milne's algorithms. We obtain these by the fourth-order Runge-Kutta method. Also, we can reduce the step size to improve the accuracy of these methods. Milne's method is unstable in certain cases because the errors do not approach zero as we reduce the step size, h. Because of this instability, the method of Adams-Moulton is more widely used.

The primary advantage of the single-step methods is that they are self starting. We can also vary the step sizes. In contrast, the multistep methods require a single-step formula to start the calculations. Step size variation is difficult. However, the efficiency of both the Milne's and Adams-Moulton methods is about twice that of the single-steps methods. We need two function evaluations per step in the former while four or five are required with the single step.

PROBLEMS AND SOLUTIONS

Problem 1-1

An experimental result in a liquid mixing experiment for the power correlation using a pitched-blade turbine shows that in the viscous regime, the power number is related to the Reynolds number by the following data:

Table P1-1

Reynolds number, N_{Re} $\frac{D^2N\rho}{\mu}$	Power number, P_o $\frac{Pg_c}{\rho N^3 D^5}$
1.0	50.0
3.0	18.0
5.0	11.0
7.0	7.9
9.0	6.2
13.0	5.0

Use the regression analysis to determine the best fit for the data.

Solution

Program PROG11.FOR determines the slope, intercept, and correlation coefficient for the above experimental data. The results show that Equation **Y=A+B*1/X** gives the best fit with the slope B = 49.01 and a correlation coefficient r = 0.9998. Figure 1-7 gives a plot of Reynolds number against power number. The figure shows that Equation 1-3 gives a perfect fit of the experimental data.

Problem 1-2

In a fluid flow experiment, the volumetric rate of fluid through a pipe is dependent on the pipe diameter and slope by the following equation.

$$Q = C_0 D^{C_1} D^{C_2} \tag{1-133}$$

where Q = flow rate, ft^3/sec
D = pipe diameter, ft
S = slope, ft/ft

Determine the flow rate of fluid for a pipe with a diameter of 3.25 ft and a slope of 0.025 ft.

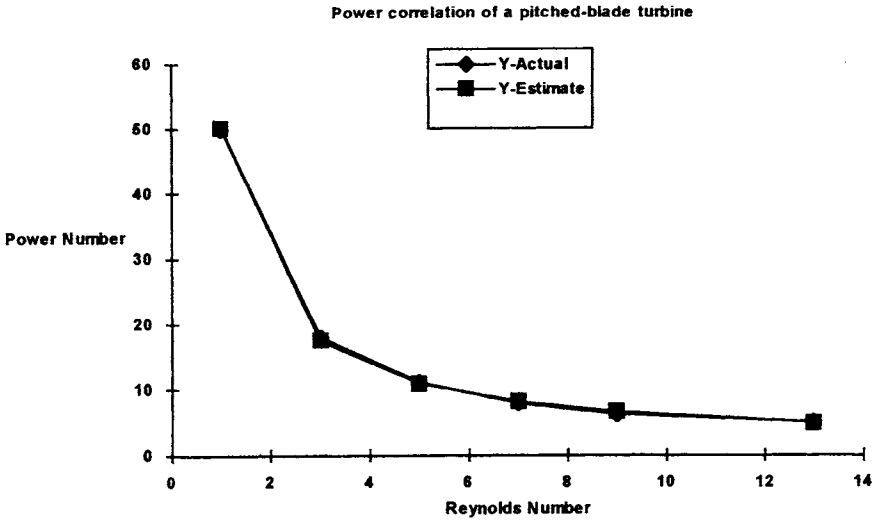


Figure 1-7. A characteristic impeller curve of power number as a function of Reynolds number.

Table P1-2

Diameter ft X_1	Slope, ft/ft X_2	Flowrate, ft^3/s Y
1.0	0.001	1.5
2.5	0.005	9.0
3.0	0.010	25.0
4.0	0.010	5.0
1.0	0.050	30.0
3.5	0.050	100.0

Solution

Program PROG12 determines the values of the coefficients C_0, C_1, and C_2 and the correlation coefficient. The coefficients are:

$$C_0 = 563.24$$
$$C_1 = 0.2725$$
$$C_2 = 0.8696$$

The correlation coefficient r = 0.9045.

The predicted flow for a pipe with a diameter of 3.25 ft and a slope of 0.03 ft/ft is 36.81 ft^3/sec.

Problem 1-3

The following data are obtained from $y(x) = x^4 + 3x^3 + 2x^2 + x + 5$. Show that a fourth-degree polynomial provides the best least squares approximation to the given data. Determine this polynomial.

Table P1-3

X	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Y	5.123	5.306	5.569	5.938	6.437	7.098	7.949

X	0.8	0.9
Y	9.025	10.363

Solution

Program PROG13 calculates (i) the coefficients for each degree of the polynomial, (ii) the variance, (iii) error sum of squares, (iv) total sum of squares, (v) coefficient of determination, and (vi) the correlation coefficient. The program shows that the fourth degree gives the lowest value of the variance, and therefore shows the best fit. The results are:

(i) the variance = 0.5557E-06
(ii) the error sum of squares = 0.2223E-05
(iii) the total sum of squares = 0.2620E+02
(iv) the correlation coefficient = 1.00

The calculated polynomial is:

$$y(x) = 4.999 + 1.011x + 1.965x^2 + 3.047x^3 + 0.977x^4$$

Problem 1-4

The final product from a chemical factory is made by blending four liquids (α, β, γ, δ) together. Each of these liquids contains four components A, B, C, and D. The product leaving the factory has to have a closely specified composition. Determine the relative quantities of α, β, γ, and δ required to meet the blend specifications in the following data:

Table P1-4

Component	w/w composition of α	β	γ	δ	w/w composition of specification
A	51.30	43.20	56.40	47.40	48.80
B	11.30	11.50	15.50	8.50	11.56
C	29.40	31.50	22.50	30.40	29.43
D	8.00	10.30	5.60	13.70	10.21

Source: B.Sc. Final year 1978, Aston University, Birmingham, U.K.

Solution

Program PROG14 uses the Gaussian elimination method to determine the quantities of each of α, β, γ, and δ and required to meet the blending specification. Answers:

$\alpha = 0.1172$

$\beta = 0.3789$

$\gamma = 0.2117$

$\delta = 0.3054$

Problem 1-5

A chemical reaction takes place in a series of four continuous stirred tank reactors arranged as shown. The chemical reaction is a first-order irreversible reaction of the type.

$$A \xrightarrow{k_i} B$$

The conditions of temperature in each reactor are such that the values of k_i and V_i are given in Table P1-5. Figure 1-8 shows four continuous stirred tanks with recycle streams.

The following assumptions are:

1. The system is at steady state.
2. The reactions are in the liquid phase.

Table P1-5

Reactor	Volume, V_i L	Rate Constant, k_i h^{-1}
1	1000	0.1
2	1500	0.2
3	100	0.4
4	500	0.3

Source: A. Constantinides [1], Applied Numerical Methods With Personal Computers, *Courtesy of McGraw-Hill Book Co., 1987.*

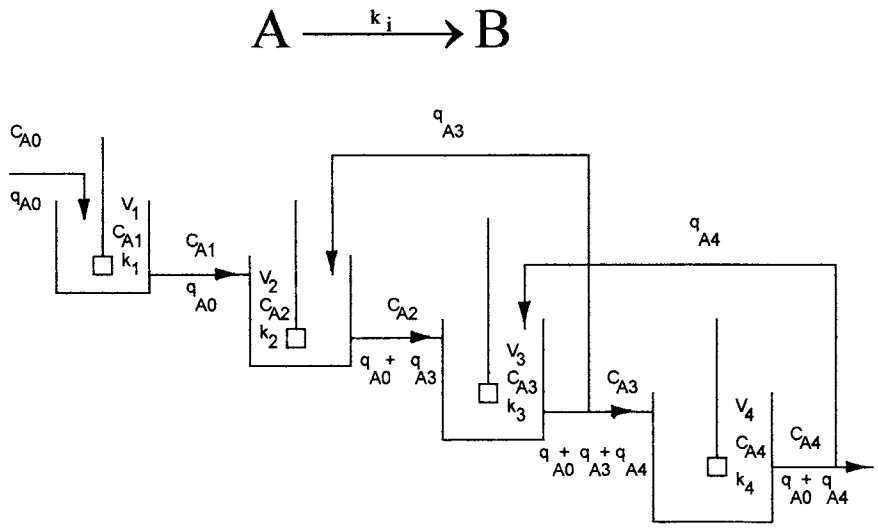


Figure 1-8. Chemical reaction with recycles in four continuous stirred tanks.

3. There is no change in volume or density of the liquid.
4. The rate of disappearance of component A in each in reactor is given by:

$$(-r_A) = kC_A$$

Set up the material balance equations for each of the four reactors, and use the Gauss-Seidel method to determine the exit concentration from each reactor.

Solution

The general unsteady state material balance for each reactor is:

Input = output + disappearance + accumulation by reaction

Mass balance for reactor 1.

$$q_{A0}C_{A0} = q_{A0}C_{A1} + (-r_A)V_1 + V_1\frac{dC_{A1}}{dt}$$

Because the system is at steady state, the accumulation is zero, the above equation becomes

$$q_{A0}C_{A0} = q_{A0}C_{A1} + k_1C_{A1}V_1$$

Mass balance for reactor 2.

$$q_{A0}C_{A1} + q_{A3}C_{A3} = (q_{A0} + q_{A3})C_{A2} + k_2C_{A2}V_2$$

Mass balance for reactor 3.

$$(q_{A0} + q_{A3})C_{A2} + q_{A4}C_{A4} = (q_{A0} + q_{A3} + q_{A4})C_{A3} + k_3V_3C_{A3}$$

Mass balance for reactor 4.

$$(q_{A0} + q_{A4})C_{A3} = (q_{A0} + q_{A4})C_{A4} + k_4V_4C_{A4}$$

where

$$q_{A0} = 1000\frac{\text{liters}}{\text{h}}, C_{A0} = 1\frac{\text{mol}}{\text{liter}}$$

$$q_{A3} = 100\frac{\text{liters}}{\text{h}}$$

$$q_{A4} = 100\frac{\text{liters}}{\text{h}}$$

$$(1000)(1) = 1000C_{A1} + (0.1)(C_{A1})(1000)$$

$$1000C_{A1} + 100C_{A3} = 1100C_{A2} + (0.2)(C_{A2})(1500)$$

$$1100C_{A2} + 100C_{A4} = 1200C_{A3} + (0.4)(C_{A3})(100)$$

$$1100C_{A3} = 1100C_{A4} + (0.3)(C_{A4})(500)$$

Rearranging the above equations:

$$\begin{aligned}
1100C_{A1} &= 1000 \\
1000C_{A1} - 1400C_{A2} + 100C_{A3} &= 0 \\
1100C_{A2} - 1240C_{A3} + 100C_{A4} &= 0 \\
1100C_{A3} - 1250C_{A4} &= 0
\end{aligned}$$

The above is a set of four simultaneous linear algebraic equations. This is a diagonal system of equations because the coefficients on the diagonal are larger in above value than the sum of the absolute values of the other coefficients. Therefore, the best method of solution for a diagonal system of linear algebraic equations is the Gauss-Seidel method. PROG15 is the Gauss-Seidel computer program for solving the above

equations. The four material balance equations are rearranged to solve for the unknown on the diagonal position of each equation.

$$C_{A1} = \frac{1000}{1100}$$

$$C_{A2} = \frac{1000C_{A1} + 100C_{A3}}{1400}$$

$$C_{A3} = \frac{1100C_{A2} + 100C_{A4}}{1240}$$

$$C_{A4} = \frac{1100C_{A3}}{1250}$$

An initial guess of C_{A1}, C_{A2}, C_{A3} and C_{A4} = 0.5 is used to start the Gauss-Seidel iterative method. The method converges after five iterations to the solutions:

$C_{A1} = 0.9091$

$C_{A2} = 0.6969$

$C_{A3} = 0.6654$

$C_{A4} = 0.5856$

Problem 1-6

The Colebrook implicit equation for the Darcy friction factor, f_D, for turbulent flow is:

$$F(f_D) = \frac{1}{f_D^{0.5}} + 0.86858 \ln\left[\frac{\varepsilon / D}{3.7} + \frac{2.51}{N_{Re} f_D^{0.5}}\right]$$

Determine the friction factor for N_{Re} = 184000, ε = 0.00015 ft and D = 0.17225ft (2.067 inch).

Solution

PROGRAM PROG16 solves the above implicit equation. The program uses Newton-Raphson method to give f_D = 0.02063 after three iterations.

Problem 1-7

In a fire tube boiler where gas flows inside the tubes and a steam-water mixture flows on the outside, the heat transfer coefficient inside h_i(Btu/ft^{2}hr.°F) can be expressed as:

$$h_i = 2.44 \frac{w^{0.8} F_1}{d^{0.8}}$$

where factors F_1, F_2, and F_3 are:

$$F_1 = \left(\frac{C_p}{\mu} \right)^{0.4} k^{0.6}$$

$$F_2 = \left(\frac{C_p}{\mu} \right)^{0.3} k^{0.7}$$

$$F_3 = \left(\frac{F_2}{C_p} \right) \mu^{=0.15}$$

Table P1-6 shows F_1, F_2, and F_3 for flue gas at a temperature range 200°F ≤ T ≤ 1200°F. Determine the values of F_1, F_2, and F_3 if the film temperature is 575°F.

Table P1-6

Temperature °F	F_1	F_2	F_3
200	0.1700	0.0954	0.5851
300	0.1770	0.1015	0.6059
400	0.1835	0.1071	0.6208
600	0.1943	0.1170	0.6457
800	0.2051	0.1264	0.6632
1000	0.2136	0.1340	0.6735
1200	0.2216	0.1413	0.6849

Source: V. Ganapathy, "Simplified Approach to Designing Heat Transfer Equipment," Courtesy of Chem. Eng., *April 13, 1987, pp. 88–87.*

Solution

PROG17 uses the interpolation methods of determine the values of F_1, F_2, and F_3 at t = 575°F. The values from the Stirling's central difference formula are:

$F_1 = 0.1932$
$F_2 = 0.1160$
$F_3 = 0.6431$

These values give the best interpolation at t = 575°F rather than the Newton-Gregory's forward or backward interpolation formulae.

Problem 1-8

A tracer experiment was carried out in a nozzle type reactor of volume V = 5.13L with liquid rate at 2.9 l/min. Table P1-7 shows data for the exit age distribution E(θ) against the dimensionless residence time θ. Determine the area under the distribution curve.

Table P1-7

θ	E(θ)	θ	E(θ)
0.000	0.000	1.243	0.403
0.113	0.308	1.356	0.355
0.226	0.995	1.469	0.313
0.339	0.876	1.582	0.275
0.452	0.786	1.695	0.237
0.565	0.720	1.808	0.213
0.678	0.663	1.921	0.171
0.791	0.606	2.034	0.142
0.904	0.545	2.147	0.123
1.017	0.497	2.260	0.109
1.130	0.450	2.373	0.095

Source: A. K. Coker, Ph.D., "A Study of Fast Reactions in Nozzle Type Reactors," 1985, Aston University, Birmingham, U.K.

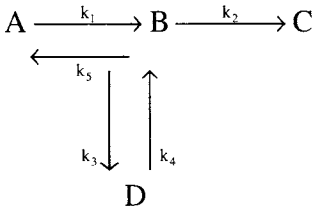
Solution

Program PROG18 uses Simpson's rule to compute the area of the exit age distribution of a tracer from a nozzle reactor.

The area under the distribution curve $\int_{0.0}^{2.373} E(\theta) = 1.0037$

Problem 1-9

Design of a batch reactor for complex reactions of the type



Determine the concentrations of A, B, C, and D over a period of ten minutes. The following rate constants and initial amounts for first-order reactions are:

Table P1-8

Rate Constant h^{-1}	Concentration at time $t = 0$ $\frac{mol}{m^3}$
$k_1 = 0.45$	$C_{A0} = 9.90$
$k_2 = 0.16$	$C_{B0} = 0.0$
$k_3 = 0.12$	$C_{C0} = 0.0$
$k_4 = 0.08$	$C_{D0} = 0.5$
$k_5 = 0.10$	

Solution

The mass balances for the batch reactor involving components A, B, C, and D are:

$$(-r_A)_{net} = k_1 C_A - k_5 C_B = -\frac{dC_A}{dt}$$

$$(-r_B)_{net} = (k_2 + k_3 + k_5)C_B - k_1C_A - k_4C_D = -\frac{dC_B}{dt}$$

$$(r_C)_{net} = k_2C_B = \frac{dC_C}{dt}$$

$$(-r_D)_{net} = k_4C_D - k_3C_B = -\frac{dC_D}{dt}$$

Rearranging the above equations

$$\frac{dC_A}{dt} = k_5C_B - k_1C_A$$

$$\frac{dC_B}{dt} = k_1C_A + k_4C_D - (k_2 + k_3 + k_5)C_B$$

Table P1-9
Runge-Kutta Fourth Order Method for a System of Ordinary Differential Equations

TIME	CONC. CA	CONC. CB	CONC. CC	CONC. CD
.00	9.90	.00	.00	.50
.50	8.15	1.65	.07	.54
1.00	6.77	2.72	.25	.66
1.50	5.68	3.39	.50	.83
2.00	4.82	3.77	.78	1.03
2.50	4.12	3.95	1.09	1.23
3.00	3.55	4.00	1.41	1.44
3.50	3.09	3.94	1.73	1.64
4.00	2.71	3.83	2.04	1.82
4.50	2.39	3.68	2.34	1.99
5.00	2.12	3.51	2.63	2.14
5.50	1.89	3.33	2.90	2.27
6.00	1.69	3.15	3.16	2.39
6.50	1.52	2.97	3.41	2.49
7.00	1.38	2.80	3.64	2.58
7.50	1.25	2.64	3.86	2.65
8.00	1.14	2.49	4.06	2.71
8.50	1.04	2.34	4.26	2.76
9.00	.96	2.21	4.44	2.79
9.50	.88	2.09	2.61	2.82
10.00	.81	1.97	4.77	2.84

$$\frac{dC_C}{dt} = k_2 C_B$$

$$\frac{dC_D}{dt} = k_3 C_B - k_4 C_D$$

where $F(1) = dC_A/dt$, $F(2) = dC_B/dt$, $F(3) = dC_C/dt$ and $F(4) = dC_D/dt$ and $X(1) = C_A$, $X(2) = C_B$, $X(3) = C_C$ and $X(4) = C_D$.

Thus, we can express the differential equations in the form of x-arrays as:

$$F(1) = k_5.X(2) - k_1.X(1)$$
$$F(2) = k_1.X(1) + k_4.X(4) - (k_2 + k_3 + k_5).X(2)$$
$$F(3) = k_2.X(2)$$
$$F(4) = k_3.X(2) - k_4.X(4)$$

Program PROG19 uses the Runge-Kutta fourth-order to solve the differential equations F(1), F(2), F(3), and F(4). Table P1-9 gives the computer printouts of C_A, C_B, C_C, and C_D from t= 0 to t=10 minutes. Figure 1-9 shows the profiles of concentrations from the start of the batch reaction to the final time of ten minutes.

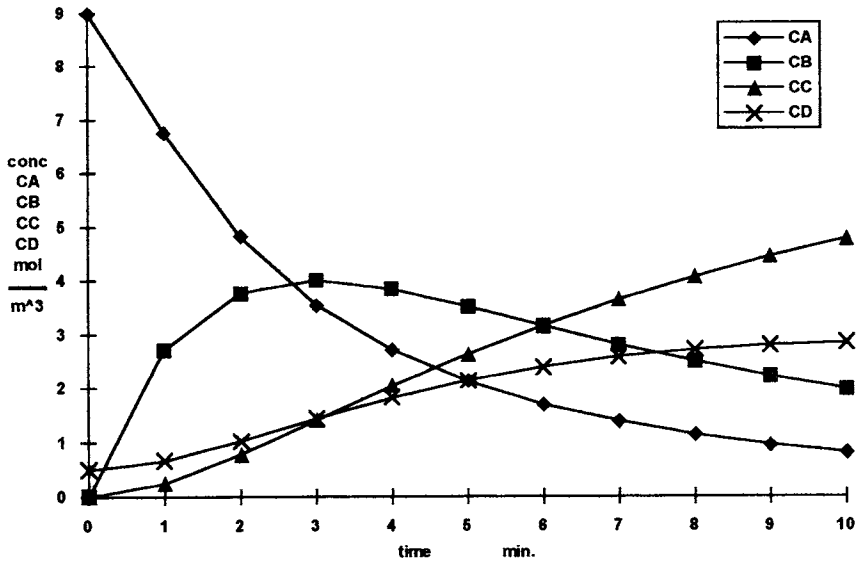


Figure 1-9. Concentration profiles of A, B, C, and D as a function of time in a batch reactor.

References

1. Constantinides, A., *Applied Numerical Methods With Personal Computers,* McGraw-Hill Book Company, New York, NY, 1987.
2. Nakamura, S., *Applied Numerical Methods With Software,* Prentice-Hall Inc., 1991.
3. Chapra, S. C. and R. P. Canale, *Numerical Methods For Engineers With Personal Computer Applications,* McGraw-Hill Book Company, New York, NY, 1984.
4. Pacher, J., *Handbook of Numerical Analysis Applications With Programs for Engineers and Scientists,* McGraw-Hill Book Company, New York, 1984.
5. Deutsch, D. J. and the Staff of Chemical Engineering, *Microcomputer Programs for Chemical Engineers,* McGraw-Hill Publication Company, New York, NY, 1984.
6. Draper, N. R. and H. Smith, *Applied Regression Analysis,* 2nd Ed., Wiley, New York, 1981.
7. Clare, B. W., "Nonlinear Regression on a Pocket Calculator," *Chem. Eng.,* August 23, 1982, pp. 83–89.
8. Colebrook, C. F., "Turbulent Flow in Pipe with Particular Reference to the Transition Region Between the Smooth and Rough Pipe Laws," *J. Inst. of Civil Eng.* (London), Vol. II, 1938-1939, pp. 133–156.
9. Redlich, O. and J. N. S. Kwong, "On the Thermodynamics of Solutions: An Equation of State Fugacities of Gaseous Solutions," *Chem. Rev.,* Vol. 44, 1949, p. 233.
10. Underwood, A. J. V., "Fractional Distillation of Multicomponent Mixtures Calculation of Minimum Reflux Ratio," *J. Inst. Petrol.,* Vol. 32, 1946, p. 614.
11. Gjumbir, M. and Z. Olujic, "Effective Ways to Solve Single Nonlinear Equations," *Chem. Eng.,* July 23, 1984, pp. 51–56.
12. Tao, B., "Finding Your Roots," *Chem. Eng.,* April 25, 1988, pp. 85–92.
13. Sargent, R. W. H., *A Review of Methods for Solving Nonlinear Algebraic Equation,* Foundations of Computed Aided Chemical Process Design, Engineering Foundation, New York, Vol. 1, 1980, pp. 27–76.
14. Danckwerts, P. V., "Continuous Flow Systems: Distribution of Residence Times," *Chem. Eng. Sci.,* 2, 1, 1953.
15. Coker, A. K., *A Study of Fast Reactions in Nozzle Type Reactors,* Ph.D. thesis, 1985, University of Aston, Birmingham, U.K.
16. Ganapathy, V., "Simplified Approach to Designing Heat Transfer Equipment," *Chem. Eng.,* April 13, 1987, pp. 81–87.

```
      PROGRAM PROG11

C     ********************************************************************
C     *  THE PROGRAM DETERMINES A RELATIONSHIP BETWEEN TWO VARIABLES     *
C     *  i.e. ONE DEPENDENT AND ONE INDEPENDENT VARIABLE. THE PROGRAM    *
C     *  SELECTS THE CURVE OF BEST FIT BY TESTING THE DATA AGAINST       *
C     *  EQUATIONS OF THE FORM: Y=a+b.F(x)                               *
C     *  THE PROGRAM GIVES THE CONSTANTS FOR THE EQUATION AND THE        *
C     *  CORRELATION COEFFICIENT.                                        *
C     ********************************************************************

C     THIS PROGRAM WILL CORRELATE X-Y DATA FOR THE FOLLOWING EQUATIONS:

C     1.  Y=a+b.X
C     2.  Y=a+b.X^2
C     3.  Y=a+b.X^0.5
C     4.  Y=a.exp(b.X)
C     5.  Y=a+b.ln(X)
C     6.  Y=a.X^b
C     7.  Y=a+b.(1/X)
C     8.  1/Y=a+b.(1/X)

C     X    =    VECTOR OF INDEPENDENT VARIABLE
C     Y    =    VECTOR OF DEPENDENT VARIABLE
C     YCAL =    VECTOR OF THE ESTIMATED DEPENDENT VARIABLE
C     N    =    NUMBER OF DATA
C     A, B =    CONSTANTS FOR THE EQUATION
C     COR  =    CORRELATION COEFFICIENT

C     ********************************************************************

        DIMENSION X(1:50),Y(1:50)

        DIMENSION YCAL1(1:50), YCAL2(1:50), YCAL3(1:50), YCAL4(1:50)
        DIMENSION YCAL5(1:50), YCAL6(1:50), YCAL7(1:50), YCAL8(1:50)

        COMMON/DATA/N,X,Y
        COMMON/DATA1/YCAL1,A1,B1
        COMMON/DATA2/YCAL2,A2,B2
        COMMON/DATA3/YCAL3, A3, B3
        COMMON/DATA4/YCAL4, A4, B4
        COMMON/DATA5/YCAL5, A5, B5
        COMMON/DATA6/YCAL6, A6, B6
        COMMON/DATA7/YCAL7, A7, B7
        COMMON/DATA8/YCAL8, A8, B8

        INTEGER N


        OPEN (UNIT=3, FILE='DATA11.DAT', STATUS='OLD', ERR=18)
        OPEN(UNIT=1,FILE='PRN')


        READ (3, *, ERR=19)N
        READ (3, *, ERR=19)(X(I), Y(I), I=1,N)
```

```
      GO TO 2

 18   WRITE (1, 111)
111   FORMAT (///,6X, 'FILE DOES NOT EXIST')
      GO TO 999
19    WRITE (1, 222)
222   FORMAT (///,6X, 'ERROR MESSAGE IN THE DATA VALUE')
      GO TO 999

2     WRITE(1, 100)
100   FORMAT(///,1H0,25X,'CURVE FITTING FOR TWO VARIABLES',
     *         /1H ,78(1H*))

      WRITE(1, 110)
110   FORMAT(///,1H0,25X,'X',15X,'Y',/1H ,78(1H-))

      DO 10 I=1,N

      WRITE(1, 120)I,X(I),Y(I)
120   FORMAT(1H0,4X,I4,6X,F14.3,6X,E14.6)
10    CONTINUE

      WRITE (1, 125)
125   FORMAT (1H ,78(1H-))


      WRITE(1, 130)
130   FORMAT(//, 1H ,25X,'THE RESULTS ARE:',/1H ,78(1H*))

      WRITE(1, 140)
140   FORMAT(/,10X,'EQUATION',10X,'REGRESSION COEFFICIENTS',
     *         5X,'CORRELATION')

      WRITE(1, 150)
150   FORMAT (28X,'INTECEPT: A',5X,'SLOPE: B',4X,'COEFFICIENT',
     *           /1H ,78(1H*))


       J=1

4     GO TO (5,15,25,35,45,55,65,75),J


C     CURVE FITTING FOR : Y=A+BX


 5    CALL LINFIT(X, Y, N, A1, B1, COR1)

      WRITE (1, 160)A1, B1, COR1
 160  FORMAT(2X,'1.',5X,'Y=A+BX',10X,3(E10.4,5X))

      DO 30 I=1,N
      YCAL1(I)=A1+B1*X(I)
30    CONTINUE

      J=J+1

      IF (J .EQ. 2) THEN
         GO TO 4
      ELSE
```

```
        GO TO 800
        ENDIF


C       CURVE FITTING FOR Y=A+BX^2


  15    DO 40 I=1,N
        X(I)=X(I)*X(I)
 40     CONTINUE
        CALL LINFIT(X, Y, N, A2, B2, COR2)

        WRITE(1, 170) A2, B2, COR2
 170    FORMAT (2X,'2.',5X,'Y=A+B*X^2',7X,3(E10.4,5X))

        DO 44 I=1,N
        YCAL2(I)=A2+B2*X(I)**2
   44   CONTINUE


        J=J+1
        IF (J .EQ. 3) THEN
        GO TO 4
        ELSE
        GO TO 800
        ENDIF


C       CURVE FITTING FOR Y=A+B.X^0.5


  25    DO 50 I=1,N
        X(I)=X(I)**0.5
 50     CONTINUE
        CALL LINFIT(X, Y, N, A3, B3, COR3)

        WRITE(1, 180) A3, B3, COR3
 180    FORMAT(2X,'3.',5X,'Y=A+B*X^0.5',5X,3(E10.4,5X))

        DO 54 I=1,N
        YCAL3(I)=A3+B3*X(I)**0.5
 54     CONTINUE


        J=J+1
        IF (J .EQ. 4) THEN
            GO TO 4
            ELSE
            GO TO 800
        ENDIF


C       CURVE FITTING FOR Y=A.EXP(B.X)
C       LINEARIZE TO Ln(Y)=Ln(A)+BX


  35    DO 60 I=1,N
```

```
      Y(I)=ALOG(Y(I))
 60   CONTINUE

      CALL LINFIT(X, Y, N, A4, B4, COR4)
      A4=EXP(A4)

      WRITE(1, 190) A4, B4, COR4
 190  FORMAT(2X,'4.',5X,'Y=A*EXP(B*X)',4X,3(E10.4,5X))

      DO 61 I=1,N
      Y(I)=EXP(Y(I))
      YCAL4(I)=A4*EXP(B4*X(I))
 61   CONTINUE


      J=J+1
      IF (J .EQ. 5) THEN
          GO TO 4
          ELSE
          GO TO 800
      ENDIF


C     CURVE FITTING FOR Y=A+B.Ln(X)



 45   DO 70 I=1,N
      X(I)=ALOG(X(I))
 70   CONTINUE
      CALL LINFIT(X, Y, N, A5, B5, COR5)

      WRITE(1, 200)A5, B5, COR5
 200  FORMAT(2X,'5.',5X,'Y=A+B*Ln(X)',5X,3(E10.4,5X))

      DO 71 I=1,N
      X(I)=EXP(X(I))
      YCAL5(I)=A5+B5*ALOG(X(I))
 71   CONTINUE

      J=J+1
      IF (J .EQ. 6) THEN
          GO TO 4
          ELSE
          GO TO 800
      ENDIF


C     CURVE FITTING FOR Y=A.X^B
C     LINEARIZE TO Ln(Y)=Ln(A)+B*Ln(X)


 55   DO 80 I=1,N
      X(I)=ALOG(X(I))
      Y(I)=ALOG(Y(I))
 80   CONTINUE

      CALL LINFIT(X, Y, N, A6, B6, COR6)
      A6=EXP(A6)
```

```
      WRITE(1,210)A6, B6, COR6
210   FORMAT(2X,'6.',5X,'Y=A*X^B',9X,3(E10.4,5X))

      DO 85 I=1,N
      X(I)=EXP(X(I))
      Y(I)=EXP(Y(I))
      YCAL6(I)=A6*(X(I)**B6)
 85   CONTINUE


      J=J+1
      IF (J .EQ. 7) THEN
          GO TO 4
          ELSE
          GO TO 800
      ENDIF


C     CURVE FITTING FOR Y=A+B.1/X


 65   DO 90 I=1,N
      X(I)=1/X(I)
 90   CONTINUE
      CALL LINFIT(X, Y, N, A7, B7, COR7)
      WRITE(1,220)A7, B7, COR7
  220 FORMAT(2X,'7.',5X,'Y=A+B*1/X',7X,3(E10.4,5X))

      DO 95 I=1,N
      X(I)=1/X(I)
      YCAL7(I)=A7+B7*1/X(I)
 95   CONTINUE


      J=J+1

      IF (J .EQ. 8) THEN
          GO TO 4
          ELSE
          GO TO 800
      ENDIF

C     CURVE FITTING FOR 1/Y=A+B.1/X


 75   DO 96 I=1,N
      X(I)=1/X(I)
      Y(I)=1/Y(I)
 96   CONTINUE

      CALL LINFIT(X, Y, N, A8, B8, COR8)

      WRITE(1, 230)A8, B8, COR8
 230  FORMAT(2X,'8.',5X,'1/Y=A+B*1/X',5X,3(E10.4,5X))


      DO 97 I=1,N
      X(I)=1/X(I)
      Y(I)=1/Y(I)
      YCAL8(I)=X(I)/(A8*X(I)+B8)
```

```
 97     CONTINUE


 800    CALL COMP(X, Y, N, COR1,COR2,COR3,COR4,COR5,COR6,COR7,COR8)

        CLOSE (3, STATUS='KEEP')
        CLOSE(1)

 999    STOP
        END


C       ********************************************************************
C       THIS PROGRAM PERFORMS A LINEAR REGRESSION ANALYSIS ON A SET
C       OF X-Y VALUES.
C       ********************************************************************

        SUBROUTINE LINFIT(X, Y, N, A,B,COR)
        DIMENSION X(1:50),Y(1:50)

        INTEGER N

        SUMX=0.0
        SUMY=0.0
        SUMXY=0.0
        SUMX2=0.0
        SUMY2=0.0
        DO 10 I=1,N
        XI=X(I)
        YI=Y(I)
        SUMX=SUMX+XI
        SUMY=SUMY+YI
        SUMXY=SUMXY+XI*YI
        SUMX2=SUMX2+XI*XI
        SUMY2=SUMY2+YI*YI
 10     CONTINUE
        SXX=N*SUMX2-SUMX*SUMX
        SXY=N*SUMXY-SUMX*SUMY
        SYY=N*SUMY2-SUMY*SUMY


C       CALCULATE THE CONSTANTS FOR THE EQUATION

        B=SXY/SXX
        XMEAN=SUMX/N
        YMEAN=SUMY/N

C       THIS PRINTS THE RESULTS ON THE SCREEN.

        WRITE(*, 50)XMEAN,YMEAN
 50     FORMAT(6X,'XMEAN:',F12.3,3X,'YMEAN:',F12.4)

        A=YMEAN-B*XMEAN


C       CALCULATE THE CORRELATION COEFFICIENT

        COR=ABS(SXY/SQRT(SXX*SYY))

        WRITE(*, 60)A, B, COR
```

```
 60     FORMAT(6X,'A=',F12.3,3X,'B=',F12.3,3X,'COR=',F12.4)

        RETURN
        END


C       ***************************************************************
C
C       THIS PROGRAM COMPARES THE VALUES OF THE CORRELATION COEFFICIENT
C       OF THE REGRESSION ANALYSES
C       ***************************************************************

       SUBROUTINE COMP(X, Y, N, COR1,COR2,COR3,COR4,COR5,COR6,COR7,COR8)
       DIMENSION X(1:50), Y(1:50)
       DIMENSION YCAL1(1:50), YCAL2(1:50), YCAL3(1:50), YCAL4(1:50)
       DIMENSION YCAL5(1:50), YCAL6(1:50), YCAL7(1:50), YCAL8(1:50)

       INTEGER N
       COMMON/DATA1/YCAL1,A1,B1
       COMMON/DATA2/YCAL2,A2,B2
       COMMON/DATA3/YCAL3,A3,B3
       COMMON/DATA4/YCAL4,A4,B4
       COMMON/DATA5/YCAL5,A5,B5
       COMMON/DATA6/YCAL6,A6,B6
       COMMON/DATA7/YCAL7,A7,B7
       COMMON/DATA8/YCAL8,A8,B8

C      FIND THE MAXIMUM VALUE OF THE CORRELATION COEFFICIENTS


       CORR=AMAX1(COR1,COR2,COR3,COR4,COR5,COR6,COR7, COR8)

       WRITE(1,100)CORR
 100   FORMAT(//,2X,'MAXIMUM CORRELATION COEFFICIENT:',F12.4)

C      COMPARE THIS VALUE WITH THE CALCULATED VALUES OF THE
C      REGRESSION ANALYSES.

       IF (CORR .EQ. COR1) THEN
           CALL OUTPUT(X, Y, YCAL1, N, A1,B1,COR1)
           GO TO 10
       ELSEIF (CORR .EQ. COR2) THEN

           CALL OUTPUT(X, Y, YCAL2, N, A2, B2, COR2)
           GO TO 10

       ELSEIF (CORR .EQ. COR3) THEN
         CALL OUTPUT(X, Y, YCAL3, N, A3, B3, COR3)
         GO TO 10

         ELSEIF (CORR .EQ. COR4) THEN
         CALL OUTPUT (X, Y, YCAL4, N, A4, B4, COR4)
         GO TO 10

         ELSEIF (CORR .EQ. COR5) THEN
         CALL OUTPUT(X, Y, YCAL5, N, A5, B5, COR5)
         GO TO 10

         ELSEIF (CORR .EQ. COR6) THEN
         CALL OUTPUT(X, Y, YCAL6, N, A6, B6, COR6)
```

```
      GO TO 10

      ELSEIF (CORR .EQ. COR7) THEN
      CALL OUTPUT (X, Y, YCAL7, N, A7, B7, COR7)
      GO TO 10

      ELSE IF (CORR .EQ. COR8) THEN
      CALL OUTPUT(X, Y, YCAL8, N, A8, B8, COR8)
      ENDIF

 10   RETURN
      END

C     ************************************************************
C     THIS PROGRAM OUTPUTS THE RESULTS OF THE REGRESSION ANALYSIS
C     WITH THE ESTIMATED VALUE OF THE DEPENDENT VARIABLE,
C     THE CONSTANTS AND THE CORRELATION COEFFICIENT.
C     ************************************************************

      SUBROUTINE OUTPUT (X, Y, YCAL, N, A, B, COR)
      DIMENSION X(1:50), Y(1:50), YCAL(1:50)
      INTEGER N

      WRITE(1, 100)
 100  FORMAT(/,1H0, 21X,'X-ACTUAL', 13X,'Y-ACTUAL', 11X,'Y-ESTIMATE',
     *       /,1H ,78(1H-))
      DO 10 I = 1,N

      WRITE(1, 110) I, X(I), Y(I), YCAL(I)
 110  FORMAT(1H0,4X,I4,6X,F14.3,2(6X,F14.6))
 10   CONTINUE

      WRITE (1, 120)
 120  FORMAT (1H ,78(1H-))

      WRITE (1, 130)A, B, COR
 130  FORMAT(//,1H0,6X,'CONSTANTS FOR THE EQUATION:',/1H0,6X,
     *       'A:', 6X, F12.4, /1H0, 6X, 'B:', 6X, F12.4,/1H0, 6X,
     *       'CORRELATION COEFFICIENT:',F12.4/,1H ,78(1H-))
      RETURN
      END
```

DATA11.DAT

```
6
1.0   50.0
3.0   18.0
5.0   11.0
7.0   7.90
9.0   6.20
13.0   5.0
```

CURVE FITTING FOR TWO VARIABLES

	X	Y
1	1.000	.500000E+02
2	3.000	.180000E+02
3	5.000	.110000E+02
4	7.000	.790000E+01
5	9.000	.620000E+01
6	13.000	.500000E+01

THE RESULTS ARE:

	EQUATION	REGRESSION COEFFICIENTS INTECEPT: A	SLOPE: B	CORRELATION COEFFICIENT
1.	Y=A+BX	.3577E+02	-.3066E+01	.7735E+00
2.	Y=A+B*X^2	.2553E+02	-.1650E+00	.6036E+00
3.	Y=A+B*X^0.5	.3577E+02	-.3066E+01	.7735E+00
4.	Y=A*EXP(B*X)	.3605E+02	-.1792E+00	.9150E+00
5.	Y=A+B*Ln(X)	.4416E+02	-.1772E+02	.9496E+00
6.	Y=A*X^B	.4913E+02	-.9203E+00	.9986E+00
7.	Y=A+B*1/X	.1123E+01	.4901E+02	.9998E+00
8.	1/Y=A+B*1/X	.1575E+00	-.1559E+00	.8137E+00

MAXIMUM CORRELATION COEFFICIENT: .9998

	X-ACTUAL	Y-ACTUAL	Y-ESTIMATE
1	1.000	50.000010	50.130660
2	3.000	18.000000	17.459000

3	5.000	11.000000	10.924670
4	7.000	7.900001	8.124240
5	9.000	6.200000	6.568446
6	13.000	5.000000	4.892976

CONSTANTS FOR THE EQUATION:

A: 1.1232

B: 49.0075

CORRELATION COEFFICIENT: .9998

```
      PROGRAM PROG12

C     ****************************************************************
C     LINEAR MULTIPLE REGRESSION ANALYSIS
C     THE PROGRAM PERFORMS A LINEAR MULTIPLE REGRESSION ANALYSIS
C     INVOLVING TWO OR MORE INDEPENDENT VARIABLES AND ONE DEPENDENT
C     VARIABLE USING THE GAUSS-ELIMINATION METHOD.
C
C     THE LINEAR MULTIPLE REGRESSION EQUATION IS OF THE FORM
C     Y=Co+C1*x1+C2*x2+......+Ck*Xk
C
C     DEPENDING ON THE NUMBER OF THE INDEPENDENT VARIABLES,
C     SIMULTANEOUS SOLUTION BY GAUSS ELIMINATION IS EMPLOYED TO
C     CALCULATE THE UNKNOWN REGRESSION CO-EFFICIENTS.
C
C
C     ****************************************************************
C
C     X1,.....Xk = THE INDEPENDENT VARIABLES
C     Y          = THE DEPENDENT VARIBALE
C     CO,C1,..Ck = THE UNKNOWN REGRESSION COEFFICIENTS
C     k          = THE NUMBER OF INDEPENDENT VARIABLES.
C     N          = THE NUMBER OF DATA VALUES
C     N1         = THE NUMBER OF EQUATIONS
C
C     ****************************************************************
C
C
      DIMENSION A(1:50,1:50),B(1:50),X1(1:50),X2(1:50)
      DIMENSION Y(1:50),YCAL(1:50)
      REAL MEANY


      DATA N/6/, N1/3/

      DATA X1(1)/1.00/, X2(1)/0.001/, Y(1)/1.5/
      DATA X1(2)/2.50/, X2(2)/0.005/, Y(2)/9.0/
      DATA X1(3)/3.00/, X2(3)/0.010/, Y(3)/25.0/
      DATA X1(4)/4.00/, X2(4)/0.010/, Y(4)/5.0/
      DATA X1(5)/1.00/, X2(5)/0.050/, Y(5)/30.0/
      DATA X1(6)/3.50/, X2(6)/0.050/, Y(6)/100.0/

C     THIS OPEN STATEMENT PRINTS THE RESULTS ONTO A PRINTER

      OPEN (UNIT=1, FILE='PRN')

C     LINEAR MULTIPLE REGRESSION ANALYSIS FOR AN EQUATION
C     Y=C0+C1*X1+C2*X2

      WRITE(1, 100)
  100 FORMAT(//,20X,'LINEAR MULTIPLE REGRESSION ANALYSIS',/1H ,20X,
     *'FOR AN EQUATION : Y=CO+C1*X1+C2*X2',/1H ,74(1H*))

      WRITE(1, 110)
  110 FORMAT(/,1H0,25X,'X1',15X,'X2',15X,'Y',/,1H ,74(1H-))

      DO 20 I=1,N
      WRITE(1, 115)I,X1(I),X2(I),Y(I)
  115 FORMAT(6X,I4,3(6X,F12.4))
   20 CONTINUE
```

```
      WRITE (1, 120)
  120 FORMAT (1H ,74(1H-))

C     LINEARIZE BOTH THE INDEPENDENT AND DEPENDENT VARIABLES.

      DO 12 J = 1, N
      X1(J)    = ALOG(X1(J))
      X2(J)    = ALOG(X2(J))
      Y(J)     = ALOG(Y(J))
   12 CONTINUE

      WRITE(1,121)
  121 FORMAT(/,1H ,20X,'LINEARIZED INDEPENDENT AND DEPENDENT VARIABLES')
      WRITE(1,122)
  122 FORMAT(/,3X,70(1H*)//,1H0,25X,'X1',15X,'X2',15X,'Y',/,1H ,74(1H-))


      DO 15 I = 1,N
      WRITE(1, 125)I, X1(I), X2(I), Y(I)
  125 FORMAT(6X,I4,3(6X,F12.4))
   15 CONTINUE

      WRITE (1, 126)
  126 FORMAT (1H ,74(1H-))

      WRITE(1, 130)N1
  130 FORMAT(//,1H0,6X,'THE NUMBER OF EQUATIONS:',3X,I4)


C     CALCULATE THE COEFFICIENTS OF MATRIX : A AND R.H.S. VECTOR : Y


   25 SX1=0.0
      SX2=0.0
      SX1X2=0.0
      SX1SQ=0.0
      SX2SQ=0.0
      SY=0.0
      SX1Y=0.0
      SX2Y=0.0
      DO 30 J=1,N
      SX1=SX1+X1(J)
      SX2=SX2+X2(J)
      SX1X2=SX1X2+X1(J)*X2(J)
      SX1SQ=SX1SQ+X1(J)**2
      SX2SQ=SX2SQ+X2(J)**2
      SY=SY+Y(J)
      SX1Y=SX1Y+X1(J)*Y(J)
      SX2Y=SX2Y+X2(J)*Y(J)
   30 CONTINUE
      MEANY=SY/N

      WRITE(*, 140)SX1,SX2,SY
  140 FORMAT(3X,'SUMX1=',F10.4,6X,'SUMX2=',F10.4,6X,'SUMY=',F10.4)


C     AUGMENT THE COEFFICIENTS OF MATRIX A WITH THE R.H.S. VECTOR Y

      A(1,1)=FLOAT(N)
```

```
      A(2,1)=SX1
      A(3,1)=SX2
      A(1,2)=A(2,1)
      A(2,2)=SX1SQ
      A(3,2)=SX1X2
      A(1,3)=A(3,1)
      A(2,3)=A(3,2)
      A(3,3)=SX2SQ
      A(1,4)=SY
      A(2,4)=SX1Y
      A(3,4)=SX2Y

C     NUMBER OF EQUATION: N1

      M=N1+1

      CALL GAUSS(A,B,N1,M,TEST)
      IF (TEST .EQ. 0.0) THEN
      GO TO 40
      ELSE
      WRITE(1, 160)
  160 FORMAT(//,1H0,6X,'ERROR FLAG')
      ENDIF
      GO TO 999

   40 WRITE(1, 165)
  165 FORMAT(//,1H0,74(1H*))
      CO=B(1)
      C1=B(2)
      C2=B(3)

C     CALCULATE THE ESTIMATED VALUES OF THE DEPENDENT VARIABLES

      DO 45 I=1,N
      YCAL(I)=CO+C1*X1(I)+C2*X2(I)
   45 CONTINUE

C     CALCULATE THE CORRELATION COEFFICIENT: COR

C     REGRESSION SUM OF SQUARES: SSR
C     TOTAL SUM OF SQUARES: SST
C     ERROR (OR RESIDUAL) SUM OF SQUARES: SSE

      SSR=0.0
      SST=0.0
      SSE=0.0
      DO 50 I=1,N
      SSR=SSR+(YCAL(I)-MEANY)**2
      SST=SST+(Y(I)-MEANY)**2
      SSE=SSE+(Y(I)-YCAL(I))**2
   50 CONTINUE
C
C     THE CORRELATION COEFFICIENT: COR
      COR=SQRT(1.0-(SSE/SST))
      CALL OUTPUT(X1,X2,Y,YCAL,B,N,SSR,SSE,SST,COR)
      CLOSE (UNIT= 1)

  999 STOP
      END
```

```
C
C     ****************************************************************
C     THIS PROGRAM SOLVES A SET OF LINEAR EQUATIONS USING THE
C     GAUSSIAN ELIMINATION METHOD.  A SEARCH IS MADE IN EACH COLUMN
C     FOR THE LARGEST ELEMENT.
C     ****************************************************************

      SUBROUTINE GAUSS(A,X,N,NP1,TEST)
      DIMENSION A(1:50,50),X(1:50)
      NM1=N-1
      DO 10 I=1,NM1
      KP1=I+1
      L=I
      DO 20 J=KP1,N
      IF(ABS(A(J,I)) .LE. ABS(A(L,I))) THEN
      GO TO 20
      ELSE
      L=J
      ENDIF
   20 CONTINUE
      IF(L .EQ. I) THEN
      GO TO 30
      ELSE
      IF(ABS(A(L,I)) .LE. 1E-7) THEN
      GO TO 40
      ELSE
      ENDIF
C
C     INTERCHANGE THE VALUES
C
      DO 50 J=1,NP1
      TEMP=A(I,J)
      A(I,J)=A(L,J)
      A(L,J)=TEMP
   50 CONTINUE
      ENDIF
C
C
C     ELIMINATE ALL ELEMENTS IN THE COLUMN BELOW MAIN DIAGONAL
C
   30  D=A(I,I)
      DO 60 J=1,NP1
      A(I,J)=A(I,J)/D
   60 CONTINUE
      DO 70 L=KP1,N
      E=A(L,I)
      DO 80 J=1,NP1
      A(L,J)=A(L,J)-E*A(I,J)
   80 CONTINUE
   70 CONTINUE
   10 CONTINUE

      IF(ABS(A(N,N)) .LE. 1E-7) THEN
      GO TO 40
      ELSE

C     BACK SUBSTITUTION

      X(N)=A(N,NP1)/A(N,N)
      ENDIF
```

```
      DO 85 J=1,NM1
      J1=N-J
      X(J1)=A(J1,NP1)
      DO 90 K=J1+1,N
      X(J1)=X(J1)-A(J1,K)*X(K)
   90 CONTINUE
   85 CONTINUE
      TEST=0.0
      RETURN
   40 TEST=1.0
      RETURN
      END

C     ****************************************************************
C     THIS PROGRAM GIVES THE RESULTS OF THE REGRESSION ANALYSIS WITH
C     THE ESTIMATED VALUE OF THE DEPENDENT VARIABLE, THE CONSTANTS
C     FOR THE EQUATION AND THE CORRELATION COEFFICIENT.
C     ****************************************************************

      SUBROUTINE OUTPUT(X1,X2,Y,YCAL,B,N,SSR,SSE,SST,COR)
      DIMENSION X1(1:50),X2(1:50),Y(1:50),YCAL(1:50),B(1:50)
      WRITE(1, 100)
  100 FORMAT(//,1H0,18X,'X1-ACTUAL',7X,'X2-ACTUAL',7X,'Y-ACTUAL',7X,
     *'Y-ESTIMATE',/,1H ,74(1H-))
      DO 10 I=1,N
      WRITE(1,110)I,X1(I),X2(I),Y(I),YCAL(I)
  110 FORMAT(1H0,4X,I4,6X,F10.3,6X,F10.3,6X,F10.4,6X,F10.4)
   10 CONTINUE

      WRITE (1, 115)
  115  FORMAT (1H ,74(1H-))

      CO=EXP(B(1))
      C1=B(2)
      C2=B(3)


C     CALCULATE THE PREDICTED FLOW FOR A PIPE WITH A DIAMETER OF 3.25ft.
C     AND A SLOPE OF 0.03 ft./ft.
C     Q=C0.D^C1.S^C2


      Q=CO*(3.25**C1)*(0.03**C2)
      WRITE(1, 120)CO,C1,C2
  120 FORMAT(//,1H0,6X,'COEFFICIENTS FOR THE EQUATION:',/1H0,6X,
     *'CO=',3X,F12.4,/1H0,6X,'C1=',3X,F12.4,/1H0,6X,'C2=',3X,F12.4)
      WRITE(1,130)SSR,SSE,SST,COR
  130 FORMAT(/,1H0,6X,'REGRESSION SUM OF SQUARES:',F12.4,/1H0,6X,
     *'ERROR SUM OF SQUARES:',F12.4,/1H0,6X,
     *'TOTAL SUM OF SQUARES:',F12.4,/1H0,6X,
     *'CORRELATION COEFFICIENT:',F12.4)


      WRITE(1, 140)Q
  140  FORMAT(//,1H ,6X,'THE PREDICTED FLOW FOR A PIPE WITH A DIAMETER',
     */1H ,6X,'OF 3.25ft AND SLOPE OF 0.03 ft/ft IS:',F8.2,1X,'ft.^3/s')

      WRITE (1, 150)
  150  FORMAT (1H ,74(1H-))

      RETURN
      END
```

LINEAR MULTIPLE REGRESSION ANALYSIS
FOR AN EQUATION : Y=CO+C1*X1+C2*X2

	X1	X2	Y
1	1.0000	.0010	1.5000
2	2.5000	.0050	9.0000
3	3.0000	.0100	25.0000
4	4.0000	.0100	5.0000
5	1.0000	.0500	30.0000
6	3.5000	.0500	100.0000

LINEARIZED INDEPENDENT AND DEPENDENT VARIABLES

	X1	X2	Y
1	.0000	-6.9078	.4055
2	.9163	-5.2983	2.1972
3	1.0986	-4.6052	3.2189
4	1.3863	-4.6052	1.6094
5	.0000	-2.9957	3.4012
6	1.2528	-2.9957	4.6052

THE NUMBER OF EQUATIONS: 3

	X1-ACTUAL	X2-ACTUAL	Y-ACTUAL	Y-ESTIMATE
1	.000	-6.908	.4055	.3269
2	.916	-5.298	2.1972	1.9761
3	1.099	-4.605	3.2189	2.6286
4	1.386	-4.605	1.6094	2.7069
5	.000	-2.996	3.4012	3.7287
6	1.253	-2.996	4.6052	4.0701

```
      COEFFICIENTS FOR THE EQUATION:

      CO=        563.2403

      C1=            .2725

      C2=            .8696

      REGRESSION SUM OF SQUARES:       8.9989

      ERROR SUM OF SQUARES:       2.0016

      TOTAL SUM OF SQUARES:      11.0006

      CORRELATION COEFFICIENT:        .9045

      THE PREDICTED FLOW FOR A PIPE WITH A DIAMETER
      OF 3.25ft AND SLOPE OF 0.03 ft/ft IS:   36.81 ft.^3/s
-------------------------------------------------------------------------------
```

```
      PROGRAM PROG13

C     ****************************************************************
C     THIS PROGRAM IS USED IN FITTING A POLYNOMIAL TO A SET OF DATA
C     N PAIRS OF THE INDEPENDENT AND DEPENDENT VARIABLES, X AND Y
C     ARE READ AND THE COEFFICIENTS OF THE NORMAL EQUATIONS FOR THE
C     LEAST SQUARES METHOD:
C
C     PARAMETERS ARE:-
C         X,Y     =   ARRAY OF X AND Y VALUES
C         N       =   NUMBER OF DATA PAIRS
C         MS, MF  =   THE RANGE OF DEGREE OF POLYNOMIALS TO BE COMPUTED
C         A       =   AUGMENTED ARRAY OF THE COEFFICIENTS OF THE NORMAL
C                     EQUATIONS
C         C       =   ARRAY OF COEFFICIENTS OF THE LEAST SQUARES
C                     POLYNOMIAL
C     ****************************************************************


      DIMENSION X(1:100), Y(1:100), C(1:10), A(1:10,1:11)
      DIMENSION XN(1:100)


      OPEN (UNIT = 1, FILE='PRN')


      DATA N/9/
      DATA X(1)/0.1/, Y(1)/5.123/, X(2)/0.2/, Y(2)/5.306/
      DATA X(3)/0.3/, Y(3)/5.569/, X(4)/0.4/, Y(4)/5.938/
      DATA X(5)/0.5/, Y(5)/6.437/, X(6)/0.6/, Y(6)/7.0988/
      DATA X(7)/0.7/, Y(7)/7.949/, X(8)/0.8/, Y(8)/9.025/
      DATA X(9)/0.9/, Y(9)/10.363/



      DATA MS/1/, MF/4/

      WRITE(1,100)
100   FORMAT(//,1H ,20X,'POLYNOMIAL REGRESSION ANALYSIS',/1H ,20X,
     *          'FOR AN EQUATION TO AN Nth DEGREE',/1H ,20X,
     *          ': Y=C0+C1.X+C2.X^2+C3.X^3+........+Cn.X^n',/1H ,
     * 74(1H*))



      IF (MF .LE.  (N-1)) THEN
      GO TO 5
      ELSE
      MF = N-1
      WRITE(*,*)

      WRITE(*,*) 'DEGREE OF POLYNOMIAL CANNOT EXCEED N-1.'
      WRITE(*,*) 'REQUESTED MAXIMUM DEGREE TOO LARGE- REDUCED TO',MF
      ENDIF

5     MFP1 = MF+1
      MFP2 = MF+2

      DO 10 I =1,N
      XN(I) =1.
```

```
10      CONTINUE

        YBAR = 0.0
        DO 11 J = 1,N
        YBAR = YBAR + Y(J)
 11     CONTINUE

        YBAR = YBAR/N


        DO 30 I = 1, MFP1
        A(I,1) = 0.0
        A(I, MFP2) = 0.

        DO 20 J = 1,N
        A(I,1) = A(I,1)+XN(J)
        A(I,MFP2) = A(I,MFP2)+Y(J)*XN(J)
        XN(J)=XN(J)*X(J)

20      CONTINUE
30      CONTINUE

        DO 50 I = 2,MFP1
        A(MFP1,I) = 0.

        DO 40 J = 1,N
        A(MFP1,I) = A(MFP1,I)+XN(J)
        XN(J) = XN(J)*X(J)

40      CONTINUE
50      CONTINUE

        DO 70 J = 2,MFP1
        DO 60 I = 1,MF
        A(I,J) = A(I+1,J-1)

60      CONTINUE
70      CONTINUE


        CALL LUDCMQ (A, MFP1, 10)


        MSP1 = MS +1
        DO 95 I = MSP1,MFP1
        DO 90 J = 1,I
        C(J) = A(J,MFP2)

90      CONTINUE

        CALL SOLNQ ( A, C, I, 10)
        IM1 = I - 1
        WRITE (1,110)IM1,(C(J), J=1,I)

110     FORMAT(//1H ,3X,'POLYNOMIAL OF DEGREE',I2,
     *         /1H ,3X,'COEFFICIENTS ARE:',3X,7F10.3)

        SSE = 0.0
        SST = 0.0
```

```
      DO 94 IPT = 1,N
      SUM = 0.0

          DO 93 ICOEF = 2,I
              JCOEF = I - ICOEF + 2

              SUM = (SUM + C(JCOEF))*X(IPT)

 93       CONTINUE


C     CALCULATE THE ERROR SUM OF SQUARES AND TOTAL SUM OF SQUARES

          SUM = SUM + C(1)
          SSE = SSE + (Y(IPT) - SUM)**2
          SST = SST + (Y(IPT) -YBAR)**2

 94   CONTINUE


C     CALCULATE THE VARIANCE


      VAR = SSE/(N-I)


      WRITE(1,120)VAR,SSE,SST

 120  FORMAT(/,1H ,3X,'VARIANCE IS:',11X,E12.4
     *       /,1H ,3X,'ERROR SUM OF SQUARES:', 2X,E12.4,
     *       /,1H ,3X,'TOTAL SUM OF SQUARES:',2X,E12.4)


C     CALCULATE CORRELATION OF DETERMINATION AND CORRELATION
C     COEFFICIENT.

      CORD = (1.0 - SSE/SST)
      COR = CORD**0.5


      WRITE(1,125)CORD, COR
 125  FORMAT(/,1H ,3X,'COEFFICIENT OF DETERMINATION IS:',2X, F10.4,
     *       /,1H ,3X,'CORRELATION COEFFICIENT:',2X,F10.4,
     *       /1H ,74(1H-))

 95   CONTINUE

      CLOSE (UNIT=1)

      STOP
      END

C     ***************************************************************
C     THIS PROGRAM FORMS  THE LU EQUIVALENT OF THE SQUARE
C     COEFFICIENT MATRIX A.  THE LU, IS THEN RETURNED IN THE
C     A MATRIX SPACE.  THE UPPER TRIANGULAR MATRIX U HAS ONES
C     ON ITS DIAGONAL.  THESE VALUES ARE NOT INCLUDED IN THE RESULT.
C     ***************************************************************

      SUBROUTINE LUDCMQ (A, N, NDIM)
```

```
      DIMENSION A(1:NDIM, 1:NDIM)

      DO 30 I = 1,N
      DO 30 J = 2,N
      SUM = 0.0
      IF (J .GT. I) THEN
      GO TO 15
      ELSE
      JM1 = J-1
      ENDIF

      DO 10 K = 1,JM1
      SUM = SUM + A(I,K)* A(K,J)
10    CONTINUE
      A(I,J) = A(I,J) - SUM
      GO TO 30

15    IM1 = I - 1

      IF (IM1 .EQ. 0.) THEN
      GO TO 25
      ELSE
      DO 20 K = 1, IM1
      SUM = SUM + A(I,K)*A(K,J)
20    CONTINUE
      ENDIF

25    IF (ABS(A(I,I)) .LT. 1.E-10) THEN
      GO TO 99
      ELSE
      A(I,J) = (A(I,J) - SUM)/A(I,I)
      ENDIF
30    CONTINUE
      RETURN

99    WRITE(*,*)'REDUCTION NOT COMPLETED AS SMALL VALUE FOUND '
      WRITE(*,*)'FOR DIVISOR IN ROW',I
      RETURN
      END

C     ****************************************************************
C     THIS PROGRAM FINDS THE SOLUTION TO A SET OF N LINEAR EQUATIONS
C     THAT CORRESPONDS TO THE RIGHT HAND SIDE VECTOR B.
C     THE A MATRIX IS THE LU DECOMPOSITION EQUIVALENT TO THE
C     COEFFICIENT MATRIX OF THE ORIGINAL EQUATIONS PRODUCED BY
C     LUDMQ.  THE SOLUTION VECTOR IS RETURNED IN THE B VECTOR.
C     DO THE REDUCTION STEP.
C     ****************************************************************

      SUBROUTINE SOLNQ ( A, B, N, NDIM)
      DIMENSION A(1:NDIM, 1:NDIM), B(1:NDIM)


      B(1) = B(1)/A(1,1)
      DO 20 I = 2,N
      IM1 = I - 1
      SUM = 0.0
      DO 10 K = 1, IM1
      SUM = SUM + A(I,K)*B(K)
```

```
 10     CONTINUE
        B(I) = (B(I) - SUM ) / A(I,I)

 20     CONTINUE

C       BACK SUBSTITUTION

        DO 40 J = 2,N
        NMJP2 = N - J + 2
        NMJP1 = N - J + 1
        SUM = 0.0

        DO 30 K = NMJP2, N
        SUM = SUM + A(NMJP1,K)*B(K)
 30     CONTINUE
        B(NMJP1) = B(NMJP1) - SUM
 40     CONTINUE
        RETURN
        END
```

```
                    POLYNOMIAL REGRESSION ANALYSIS
                    FOR AN EQUATION TO AN Nth DEGREE
                    : Y=C0+C1.X+C2.X^2+C3.X^3+........+Cn.X^n
*****************************************************************************


   POLYNOMIAL OF DEGREE 1
   COEFFICIENTS ARE:          3.809      6.340

   VARIANCE IS:                .2982E+00
   ERROR SUM OF SQUARES:       .2087E+01
   TOTAL SUM OF SQUARES:       .2620E+02

   COEFFICIENT OF DETERMINATION IS:          .9203
   CORRELATION COEFFICIENT:          .9593
-----------------------------------------------------------------------------


   POLYNOMIAL OF DEGREE 2
   COEFFICIENTS ARE:          5.305     -1.822      8.162

   VARIANCE IS:                .5950E-02
   ERROR SUM OF SQUARES:       .3570E-01
   TOTAL SUM OF SQUARES:       .2620E+02

   COEFFICIENT OF DETERMINATION IS:          .9986
   CORRELATION COEFFICIENT:          .9993
-----------------------------------------------------------------------------


   POLYNOMIAL OF DEGREE 3
   COEFFICIENTS ARE:          4.975      1.339       .659      5.002

   VARIANCE IS:                .1281E-04
   ERROR SUM OF SQUARES:       .6405E-04
   TOTAL SUM OF SQUARES:       .2620E+02

   COEFFICIENT OF DETERMINATION IS:         1.0000
   CORRELATION COEFFICIENT:         1.0000
-----------------------------------------------------------------------------


   POLYNOMIAL OF DEGREE 4
   COEFFICIENTS ARE:          4.999      1.011      1.965      3.047       .977

   VARIANCE IS:                .5557E-06
   ERROR SUM OF SQUARES:       .2223E-05
   TOTAL SUM OF SQUARES:       .2620E+02

   COEFFICIENT OF DETERMINATION IS:         1.0000
   CORRELATION COEFFICIENT:         1.0000
-----------------------------------------------------------------------------
```

```
      PROGRAM PROG14

C     ******************************************************************
C     THIS PROGRAM SOLVES A SET OF LINEAR EQUATIONS USING
C     THE GAUSS ELIMINATION METHOD WITH PARTIAL PIVOTING.
C
C     AB       = COEFFICIENT MATRIX AUGMENTED WITH R.H.S VECTORS.
C     N        = NUMBER OF EQUATIONS.
C     NP       = TOTAL NUMBER OF COLUMNS IN THE AUGMENTED MATRIX
C     NDIM     = FIRST DIMENSION OF MATRIX AB IN THE CALLING PROGRAM.
C                THE SOLUTION VECTORS ARE RETURNED IN THE AUGMENTATION
C                COLUMNS OF A
C     ******************************************************************

      DIMENSION A(1:10, 1:11)

      OPEN (UNIT=1, FILE='PRN')

      DATA N/4/
      DATA (A(1,J), J=1,5) /51.30, 43.20, 56.40, 47.40, 48.80/
      DATA (A(2,J), J=1,5) /11.30, 11.50, 15.50, 8.50, 11.56/
      DATA (A(3,J), J=1,5) /29.40, 31.50, 22.50, 30.40, 29.43/
      DATA (A(4,J), J=1,5) /8.00, 10.30, 5.60, 13.70, 10.21/


      WRITE(1,100)
100   FORMAT(1H ,10X,'GAUSSIAN ELIMINATION METHOD FOR A SET OF',\)
      WRITE(1,105)
105   FORMAT(' LINEAR EQUATIONS',/1H ,70(1H*),//)


      WRITE(1,110)
110   FORMAT(30X,'AUGMENTED MATRIX',//)

      DO 10 I=1,N
      WRITE(1,120)(A(I,J),J=1,N+1)
120   FORMAT(1X,1P6E12.4)

      NP =N +1
10    CONTINUE

      DET=1.0
C
C     CALL THE GAUSSIAN ELIMINATION

      CALL GAUSS (A, N, NP, 10, DET)

      WRITE(1,125)DET
125   FORMAT(//, 10X,'DETERMINANT=', 1X,E12.4)


      WRITE(1,130)
130   FORMAT(//,30X,'SOLUTION', /1H ,70(1H-))
      WRITE(1,140)
140   FORMAT(//,10X,'    I                  X(I)', /1H ,70(1H-))


      DO 20 I =1, N

      WRITE(1,150) I, A(I,N+1)
```

```
 150    FORMAT(10X,I5,10X,1PE16.6)
  20    CONTINUE
        WRITE(1,160)
 160    FORMAT(1H ,70(1H-))
        CLOSE (UNIT=1)
        STOP
        END


C       ************************************************************

        SUBROUTINE GAUSS (AB, N, NP, NDIM, DET)

        DIMENSION AB(1:NDIM, 1:NP)

C       BEGIN THE REDUCTION

        NM1=N-1

        DO 35 I=1,NM1

C       FIND THE ROW NUMBER OF THE PIVOT ROW AND THEN INTERCHANGE
C       ROWS INORDER TO PLACE THE PIVOT ELEMENT ON THE DIAGONAL.

        IPVT = I
        IP1 = I+1
        DO 10 J = IP1, N
        IF (ABS(AB(IPVT,I)) .LT. ABS(AB(J,I))) THEN
        IPVT =J
        ELSE
        ENDIF
 10     CONTINUE


C       CHECK THAT THE PIVOT ELEMENT IS NOT TOO SMALL. OTHERWISE
C       PRINT A MESSAGE AND RETURN

        IF ((ABS(AB(IPVT, I)) .LT. 1.E-5)) THEN
        DET=0.0
        GO TO 99
        ELSEIF( IPVT .EQ. I) THEN
        GO TO 25
        ENDIF

        DO 20 JCOL = I,NP
            SAVE=AB(I,JCOL)
            AB(I,JCOL)= AB(IPVT, JCOL)
            AB(IPVT,JCOL)=SAVE
 20     CONTINUE
        DET=-DET

C       REDUCE ALL ELEMENTS BELOW THE DIAGONAL IN THE I-TH ROW.
C       CHECK FIRST TO SEE IF A ZERO ALREADY PRESENT. IF SO,
C       SKIP REDUCTION FOR THAT ROW.

 25     DO 32 JROW = IP1,N
            IF (AB(JROW, I) .EQ. 0.0) THEN
            GO TO 32
            ELSE
            RATIO=AB(JROW,I)/AB(I,I)
```

```
            ENDIF

            DO 30 KCOL = IP1,NP
                AB(JROW, KCOL) = AB(JROW,KCOL)-RATIO*AB(I,KCOL)

 30         CONTINUE
 32     CONTINUE
 35     CONTINUE


C       WE STILL NEED TO CHECK A(N,N) FOR SIZE.

        IF (ABS(AB(N,N)). LT. 1.E-5) THEN
        GO TO 99
        ELSE
        ENDIF

C       BACK SUBSTITUTION

        NP1=N+1
        DO 50 KCOL =NP1,NP
            AB(N,KCOL}=AB(N,KCOL)/AB(N,N)
            DO 45 J= 2,N
            NVBL = NP1-J
            L=NVBL+1
            VALUE = AB(NVBL,KCOL)
            DO 40 K= L,N
                VALUE=VALUE-AB(NVBL,K)*AB(K,KCOL)
 40         CONTINUE
            AB(NVBL,KCOL)=VALUE/AB(NVBL,NVBL)

 45         CONTINUE
 50     CONTINUE


C       CALCULATE THE DETERMINANT OF THE MATRIX

        DO 60 I=1,N
        DET=DET*AB(I,I)
 60     CONTINUE
        RETURN


C       MESSAGE FOR A NEAR SINGULAR MATRIX.
 99     WRITE(1,100)
 100    FORMAT(//,10X,'SOLUTION NOT FEASIBLE. A NEAR ZERO PIVOT WAS',\)
        WRITE(1,110)
 110    FORMAT(' ENCOUNTERED')

        RETURN
        END
```

GAUSSIAN ELIMINATION METHOD FOR A SET OF LINEAR EQUATIONS

****

AUGMENTED MATRIX

5.1300E+01	4.3200E+01	5.6400E+01	4.7400E+01	4.8800E+01
1.1300E+01	1.1500E+01	1.5500E+01	8.5000E+00	1.1560E+01
2.9400E+01	3.1500E+01	2.2500E+01	3.0400E+01	2.9430E+01
8.0000E+00	1.0300E+01	5.6000E+00	1.3700E+01	1.0210E+01

DETERMINANT= -.1149E+05

SOLUTION

I	X(I)
1	1.172198E-01
2	3.789395E-01
3	2.117474E-01
4	3.053561E-01

```
      PROGRAM PROG15

C     ***************************************************************
C     GAUSS-SEIDEL ITERATION FOR N SIMULTANEOUS LINEAR EQUATIONS
C     THE ARRAY A CONTAINS THE N x N+1 AUGMENTED COEFFICIENT MATRIX
C     THE VECTOR X CONTAINS THE LATEST APPROXIMATION TO THE SOLUTION
C     THE COEFFICIENT MATRIX SHOULD BE DIAGONALLY DOMINANT.
C     AN INITIAL APPROXIMATION IS SENT TO THE SUBROUTINE IN THE VECTOR
C     X.  THE SOLUTION, AS APPROXIMATED BY THE SUBROUTINE IS RETURNED
C     IN X.  THE ITERATIONS ARE CONTINUED UNTIL THE MAXIMUM CHANGE IN
C     ANY X COMPONENT IS LESS THAN THE TOLERANCE, TOL.  IF THIS CANNOT
C     BE ACCOMPLISHED IN NITER ITERATIONS, A MESSAGE IS PRINTED.
C     IN TERMINATING THE ITERATIONS, NO ELEMENT OF X MAY UNDERGO A
C     MAGNITUDE CHANGE GREATER THAN TOLERANCE TOL FROM ONE ITERATION
C     TO THE NEXT.
C     ***************************************************************
C
C     PARAMETERS:
C
C     A    - AUGMENTED COEFFICIENT MATRIX
C     X    - INITIAL APPROXIMATION TO SOLUTION, ALSO RETURNS RESULTS
C     N    - NUMBER OF SIMULTANEOUS LINEAR EQUATIONS
C     NITER - LIMIT TO THE NUMBER OF ITERATIONS
C     TOL - TEST VALUE TO STOP THE ITERATING SOLUTION
C
C     ***************************************************************


      DIMENSION A(1:20,1:20), X(20)


      INTEGER N, NITER, I, J

      OPEN (UNIT=1, FILE='PRN')

      DATA N, NITER, TOL/4, 15, 0.0001/


      DATA A(1,1)/1100.0/, A(1,2)/0.0/, A(1,3)/0.0/, A(1,4)/0.0/
      DATA A(1,5)/1000.0/, A(2,1)/1000.0/, A(2,2)/-1400.0/
      DATA A(2,3)/100.0/, A(2,4)/0.0/, A(2,5)/0.0/, A(3,1)/0.0/
      DATA A(3,2)/1100.0/, A(3,3)/-1240.0/, A(3,4)/100.0/
      DATA A(3,5)/0.0/, A(4,1)/0.0/, A(4,2)/0.0/, A(4,3)/1100.0/
      DATA A(4,4)/-1250.0/, A(4,5)/0.0/


C     INITIAL GUESS VALUES

      DATA X(1), X(2), X(3), X(4)/0.5, 0.5, 0.5, 0.5/


      NP1 = N+1

      WRITE(1, 120)
 120  FORMAT(5X, 'SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS BY',\)
      WRITE(1,125)
 125  FORMAT(' GAUSS-SEIDEL METHOD', /5X, 70(1H*))

      WRITE(1,130) N, NITER, TOL
 130  FORMAT(//, 10X, 'NUMBER OF LINEAR EQUATIONS:',I4,/,
```

```
     + 10X,'MAXIMUM ITERATIONS',I4,/,10X,'CONVERGING TOLERANCE:',E14.6)

       WRITE(1,140)
 140   FORMAT(//,10X,'THE COEFFICIENT MATRIX A(1,1)...A(N+1,N+1) IS:',/
     +  5X, 70(1H-))

       DO 10  I = 1,N
       WRITE(1,150)(A(I,J),J=1,NP1)

 150   FORMAT(/,5(3X,F12.4),5X)

 10    CONTINUE
       WRITE(1,155)
 155   FORMAT(/,5X,70(1H-))

       WRITE(1,160)
 160   FORMAT(//,10X,'THE STARTING VECTOR X(1)....X(N) IS:')

       WRITE(1,170)(X(I),I=1,N)
 170   FORMAT(/,15X,1PE14.6)

       CALL GSITER (A, X, N, NITER, NP1, TOL)

       CLOSE (UNIT=1)
       STOP
       END


       SUBROUTINE GSITER (A, X, N, NITER, NP1, TOL)
       DIMENSION A(1:20, 1:20), X(20)
       INTEGER N, NITER, FLAG


C      NORMALIZE DIAGONAL ELEMENTS IN EACH ROW

       DO 10 I =1, N
       ASTAR = A(I,I)
       DO 10 J = 1,NP1
       A(I,J) = A(I,J)/ASTAR

 10    CONTINUE


C      NOW WE PERFORM THE ITERATIONS

       DO 30 ITER =1, NITER
       FLAG = 1
       DO 40 I = 1,N
       XSTAR = X(I)
       X(I) = A(I,NP1)

C      CALCULATE NEW SOLUTION VALUE, X(I)

       DO 50 J = 1,N
       IF ( I .EQ. J) THEN
       GO TO 50
       ELSE
       X(I) = X(I)-A(I,J)*X(J)
       ENDIF
```

```
 50     CONTINUE

C       TEST X(I) FOR CONVERGENCE

        IF (ABS(XSTAR - X(I)) .LE. TOL) THEN
        GO TO 40
        else
        FLAG = 0
        ENDIF

 40     CONTINUE


        IF (FLAG .NE. 1) THEN
        GO TO 30
        ELSE

      WRITE(1,100) ITER
 100  FORMAT(/,10X,'PROCEDURE CONVERGED AFTER',2X,I2,2X,'ITERATIONS',/)

      WRITE(1,110)(X(I),I =1,N)
 110  FORMAT(/10X,'SOLUTION VECTOR X(1).....X(N) IS:',//,(15X,1PE14.6))

        WRITE (1, 120)
 120    FORMAT (5X, 70(1H-))



        GO TO 60
        ENDIF

 30     CONTINUE


 60     RETURN

C       ****************************************************************
C       NORMAL EXIT FROM THE LOOP MEANS NON CONVERGENT IN NITER
C       ITERATIONS PRINT MESSAGE AND RETURN TO THE MAIN PROGRAM.
C       ****************************************************************

        WRITE(1,130)TOL,NITER
 130    FORMAT(//,10X,'DID NOT MEET TOLERANCE OF',E14.6,'IN',I4,
     *  'ITERATIONS')

        WRITE(1,140)(X(I),I=1,N)
 140    FORMAT(/,10X,'CURRENT VECTOR X(1).....X(N) IS :',/(15X,1PE14.6))
        RETURN
        END
```

```
SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS BY GAUSS-SEIDEL METHOD
************************************************************************


     NUMBER OF LINEAR EQUATIONS:   4
     MAXIMUM ITERATIONS  15
     CONVERGING TOLERANCE:    .100000E-03


     THE COEFFICIENT MATRIX A(1,1)...A(N+1,N+1) IS:
------------------------------------------------------------------------

 1100.0000          .0000          .0000          .0000       1000.0000

 1000.0000     -1400.0000       100.0000          .0000          .0000

     .0000      1100.0000     -1240.0000       100.0000          .0000

     .0000          .0000      1100.0000     -1250.0000          .0000
------------------------------------------------------------------------


     THE STARTING VECTOR X(1)....X(N) IS:

            5.000000E-01

            5.000000E-01

            5.000000E-01

            5.000000E-01

     PROCEDURE CONVERGED AFTER   5  ITERATIONS


     SOLUTION VECTOR X(1).....X(N) IS:

            9.090909E-01
            6.968780E-01
            6.654189E-01
            5.855687E-01
------------------------------------------------------------------------
```

```
       PROGRAM PROG16
C
C     ************************************************************
C     THIS PROGRAM EMPLOYS THE NEWTON-RAPHSON'S METHOD TO DETERMINE
C     THE ROOTS OF NONLINEAR EQUATIONS
C     ************************************************************

      EXTERNAL FUN,FDERV

      OPEN (UNIT=1, FILE='PRN')

C     INPUT DATA FOR INITIAL GUESS OF THE ROOT

      DATA X/0.015/


      WRITE(1,100)
 100  FORMAT(15X,'NEWTON-RAPHSON METHOD FOR A NON-LINEAR EQUATION',
     *          /1H ,76(1H*))


      WRITE(1,110)X
 110  FORMAT(//,6X,'INITIAL GUESS OF THE ROOT:',F12.4)


      XTOL=1.E-4
      FTOL=1.E-5
      NLIM=50
      I=0
      CALL NEWT(FUN,FDERV,X,XTOL,FTOL,NLIM,I)

      CLOSE (UNIT=1)

      STOP
      END

C     ************************************************************
C     THIS PROGRAM USES THE NEWTON-RAPHSON'S METHOD TO CALCULATE THE
C     ROOT OF NONLINEAR FUNCTIONS
C     ************************************************************

      SUBROUTINE NEWT(FUN,FDERV,X,XTOL,FTOL,NLIM,I)
C
C     FUN   = FUNCTION THAT CALCULATES F(X)
C     FDERV = FUNCTION THAT CALCULATES THE DERIVATIVE OF FUN
C     X     = VALUE OF APPROXIMATE ROOT
C     XTOL,FTOL = TOLERANCE VALUES FOR X, F(X)
C     NLIM  = LIMIT TO NUMBER OF ITERATIONS
C     I     =0 SHOWS HOW ROUTINE IS TERMINATED
C     I     =1 MEETS TOLERANCE FOR X VALUES
C     I     =2   "       "       "  F(X)
C     I     =-1 NLIM IS EXCEEDED
C     ************************************************************


      LOGICAL PRINT
      PRINT=.TRUE.
      IF (I .NE. 0) THEN
      PRINT=.FALSE.
```

```
      ELSE
      FX=FUN(X)
      ENDIF

      DO 10 J=1,NLIM
      DELTAX=FX/FDERV(X)
      X=X-DELTAX
      FX=FUN(X)
      FDX=FDERV(X)
      IF (.NOT. PRINT) THEN
      GO TO 20
      ELSE
      WRITE(1,140) J,X,FX
 140  FORMAT(//,1H0,3X,'AT ITERATION',I4,6X,'X=',E10.5,6X,
     *'F(X)=',E10.5)
      ENDIF
   20 IF (ABS(DELTAX) .LE. XTOL) THEN
      GO TO 30
      ELSEIF (ABS(FX) .LE. FTOL) THEN
      GO TO 40
      ELSE
      ENDIF
 10   CONTINUE


C     WHEN ITERATION IS COMPLETED, NLIM IS EXCEEDED

      I=-1
      WRITE(1,150)NLIM,X,FX
  150 FORMAT(//,1H0,3X,'TOLERANCE NOT MET AFTER',1X,I2,1X,
     *'ITERATIONS',1X,' THE FINAL ROOT:  X=',E10.5,2X,'F(X)=',E10.5)
      RETURN

C     THIS SECTION RETURNS AFTER MEETING XTOL

   30 I=1
      WRITE(1,160)J,X,FX
  160 FORMAT(//,1H0,3X,'TOLERANCE MET IN',1X,I2,1X,
     *'ITERATIONS',1X,'THE FINAL ROOT: X=',E10.5,2X,'F(X)=',E10.5)
      WRITE (1, 165)
 165  FORMAT (1H ,76(1H-))


      RETURN

C     THIS SECTION RETURNS AFTER MEETING F(X) TOLERANCE
   40 I=2
      WRITE(1,170)J,X,FX
 170  FORMAT(//,1H0,3X,'TOLERANCE MET IN',1X,I2,1X,
     *'ITERATIONS',1X,'THE FINAL ROOT: X=',E10.5,2X,'F(X)=',E10.5)
      WRITE (1, 175)
 175  FORMAT (1H ,76(1H-))

      RETURN
      END
C
C     ****************************************************************
C     THIS PROGRAM CALCULATES THE VALUE OF THE FUNCTION, FUN
C     ****************************************************************
```

```
      FUNCTION FUN(X)
C     The solution to F(X)=1/SQRT(X)+0.8686*ln(PR/3.7D+2.51/(RE*SQRT(X))
C
      COMMON/COKER1/PR,D,RE

      DATA PR/.0018/,D/2.067/,RE/184000/

      FUN=1/(X**.5)+0.8686*ALOG((PR/(3.7*D))+(2.51/(RE*(X**.5))))
      RETURN
      END

C     ***************************************************************
C     THIS PROGRAM CALCULATES THE DERIVATIVE OF THE FUNCTION

      FUNCTION FDERV(X)
      COMMON/COKER1/PR,D,RE

      FDERV=-0.5/X**1.5-(4.03329/((RE*(X**1.5)*PR/D)+9.287*X))
      RETURN
      END
```

```
              NEWTON-RAPHSON METHOD FOR A NON-LINEAR EQUATION
*******************************************************************************

     INITIAL GUESS OF THE ROOT:        .0150


   AT ITERATION   1       X=.19423E-01       F(X)=.22078E+00


   AT ITERATION   2       X=.20577E-01       F(X)=.93464E-02


   AT ITERATION   3       X=.20630E-01       F(X)=.18002E-04


   TOLERANCE MET IN  3 ITERATIONS THE FINAL ROOT: X=.20630E-01  F(X)=.18002E-04
-------------------------------------------------------------------------------
```

```
      PROGRAM PROG17

C     ***************************************************************
C     THIS PROGRAM PERFORMS INTERPOLATION WITHIN A SET OF (X, Y)
C     PAIRS  TO GIVE THE Y VALUE CORRESPONDING TO A GIVEN X VALUE.
C     THE PROGRAM USES THE NEWTON-GREGORY FORWARD AND BACKWARD INTER-
C     POLATIONS AND STIRLING'S CENTRAL DIFFERENCE METHOD.
C
C     THE NUMBER OF DATA AND (X, Y) PAIRS ARE IN A DATA FILE:DATA17.DAT
C     THE RESULTS ARE PRINTED IN A UNIT NUMBER (UNIT=1) AND FILE NAME
C     IS ASSIGNED TO PRINT IN 'PRN'.
C
C     X   =   ARRAY OF VALUES OF THE INDEPENDENT VARIABLE
C     Y   =   ARRAY OF VALUES OF THE DEPENDENT VARIABLE
C     XR  =   THE X - VALUE FOR WHICH THE ESTIMATE OF Y IS REQUIRED
C     M   =   THE NUMBER OF INDEPENDENT VARIABLE TO BE INTERPOLATED
C     N   =   THE NUMBER OF POINTS
C     ***************************************************************



      DIMENSION X(1:90),Y(1:90,1:10),XR(1:40)
      OPEN (UNIT=3,FILE='DATA17.DAT',STATUS='OLD',ERR=18)
      OPEN (UNIT=1,FILE='PRN')
      E=0.000005

      READ(3,*,ERR=19)N
      K=(2*N)-1
      READ(3,*,ERR=19)(X(I),Y(I,1),I=1,K,2)
      GO TO 10
 18   WRITE(*,11)
 11   FORMAT(3X,'FILE DOES NOT EXIST')
      GO TO 30
 19   WRITE(*,22)
 22   FORMAT(3X,'ERROR MESSAGE IN THE DATA VALUE')
      GO TO 30


 10   DO J=2,6
      N1=K-J
      DO I=J,K,2
      Y(I,J)=Y(I+1,J-1)-Y(I-1,J-1)
      IF (Y(I,J) .GT. E*2**(J-1)) THEN
      K1=J
      ELSE
      ENDIF
      END DO
      END DO
      WRITE(1,99)
 99   FORMAT(30X,'THE DIFFERENCE TABLE',/1H ,70(1H*))

      DO I1=1,3,2
      WRITE(1,100)X(I1),(Y(I1,J1),J1=1,I1,2)
      WRITE(1,110)(Y(I1+1,J1),J1=2,I1+1,2)
100   FORMAT(1H ,8X,F8.1,2(8X,F8.4))
110   FORMAT(1H ,32X,2(F8.4,8X))
      END DO

      DO I2=5,K-5,2
      WRITE(1,200)X(I2),(Y(I2,J2),J2=1,6,2)
```

```
      WRITE(1,250)(Y(I2+1,J2),J2=2,6,2)
200   FORMAT(1H ,8X,F8.1,3(8X,F8.4))
250   FORMAT(1H ,32X,3(F8.4,8X))
      END DO

      DO I3=K-4,K-1,2
      WRITE(1,300) X(I3),(Y(I3,J3),J3=1,K-I3+1,2)
      WRITE(1,350)(Y(I3+1,J3),J3=2,K-I3,2)
300   FORMAT(1H ,8X,F8.1,3(8X,F8.4))
350   FORMAT(1H ,32X,2(F8.4,8X))
      END DO

      WRITE(1,400) X(K),Y(K,1)
400   FORMAT(1H ,8X,F8.1,8X,F8.4)

      WRITE(1,450) E
450   FORMAT(//,10X,1H ,'ROUND-OFF ERROR:',E12.5,//)


C     READ THE NUMBER OF INDEPENDENT VARIABLES TO BE INTERPOLATED

      READ(3,*,ERR=19)M

      DO L=1,M

C     READ THE VALUE OF THE INDEPENDENT VARIABLE

      READ(3,*,ERR=19) XR(L)

      DO I=1,K-1,2
      IF (XR(L) .GT. X(I) .AND. XR(L) .LT. X(I+1)) THEN
      J=I
      ELSE
      END IF
      END DO

      RC=(XR(L)-X(J))/ABS(X(3)-X(1))
      R=(XR(L)-X(1))/ABS(X(3)-X(1))
      T=(XR(L)-X(K))/ABS(X(3)-X(1))

      CALL NFD(YF,Y,R)
      CALL NBD(YB,Y,T,K)
      CALL SCD(YC,Y,RC,J)

      WRITE(1,540)
      WRITE(1,550) XR(L),YF
      WRITE(1,600) XR(L),YB
      WRITE(1,650) XR(L),YC

540   FORMAT(20X,1H ,'INTERPOLATED VALUE OF THE INDEPENDENT VARIABLE',
     *       /1H ,70(1H*))

550   FORMAT(/,10X,'NEWTON FORWARD DIFF.',10X,'XR=',F8.2,10X,
     *'YF=',F8.4)
600   FORMAT(/,10X,'NEWTON BACKWAD DIFF.',10X,'XR=',F8.2,10X,
     *'YB=',F8.4)
650   FORMAT(/,10X,'STIRLING CENTRAL DIFF.',8X,'XR=',F8.2,10X,'YC=',
     *      F8.4)
      END DO
```

```
      WRITE (1, 700)
 700  FORMAT (1H ,70(1H-))

      CLOSE (UNIT=3,STATUS='KEEP')
      CLOSE (UNIT=1)

 30   STOP
      END
C
C     ****************************************************************
C     THIS PROGRAM PERFORMS THE NEWTON-GREGORY FORWARD INTERPOLATION
C     OF POLYNOMIALS
C     ****************************************************************

      SUBROUTINE NFD(B,C,D)
      DIMENSION C(1:90,1:10)
      B=C(1,1)+D*C(2,2)+(D*(D-1.)*C(3,3))/2.0
     1 +(D*(D-1.)*(D-2.)*C(4,4))/6.
      RETURN
      END

C     ****************************************************************
C     THIS PROGRAM PERFORMS THE NEWTON-GREGORY BACKWARD INTERPOLATION
C     OF POLYNOMIALS.
C     ****************************************************************

      SUBROUTINE NBD(F,G,H,J)
      DIMENSION G(1:90,1:10)
      F=G(J,1)+H*G(J-1,2)+(H*(H+1.)*G(J-2,3))/2.0
     1 +(H*(H+1.)*(H+2.)*G(J-3,4))/6.0
      RETURN
      END

C     ****************************************************************
C     THIS PROGRAM PERFORMS THE STIRLING'S CENTRAL DIFFERENCE
C     INTERPOLATION OF POLYNOMIALS.
C     ****************************************************************


      SUBROUTINE SCD(P,Q,S,J)
      DIMENSION Q(1:90,1:10)
      P=Q(J,1)+(S*((Q(J-1,2)+Q(J+1,2))/2.)/2.)+S*S*Q(J,3)
     1 +((S-1.)*S*(S-1.)*((Q(J-1,4)+Q(J+1,4))/2.)/6.)
      RETURN
      END
C     ****************************************************************
```

DATA17.DAT

```
7
200.0   0.5851
300.0   0.6059
400.0   0.6208
600.0   0.6457
800.0   0.6632
1000.0  0.6735
1200.0  0.6849
1
575.0
```

THE DIFFERENCE TABLE

200.0	.5851					
		.0208				
300.0	.6059		-.0059			
		.0149		.0159		
400.0	.6208		.0100		-.0333	
		.0249		-.0174		.0509
600.0	.6457		-.0074		.0176	
		.0175		.0002		-.0095
800.0	.6632		-.0072		.0081	
		.0103		.0083		
1000.0	.6735		.0011			
		.0114				
1200.0	.6849					

ROUND-OFF ERROR: .50000E-05

INTERPOLATED VALUE OF THE INDEPENDENT VARIABLE

NEWTON FORWARD DIFF.	XR= 575.00	YF= .6805
NEWTON BACKWAD DIFF.	XR= 575.00	YB= .4388
STIRLING CENTRAL DIFF.	XR= 575.00	YC= .6431

```
      PROGRAM PROG18

C     ************************************************************
C     THIS PROGRAM CALCULATES THE AREA OF A FUNCTION DEFINED BY
C     THE UNIFORM SPACING OF THE INDEPENDENT X VALUES
C
C     FUN      =   ARRAY OF VALUES OF THE FUNCTION
C     N        =   NUMBER OF VALUES
C     H        =   THE UNIFORM SPACING BETWEEN X VALUES
C     AREA     =   THE ESTIMATED INTERVAL OF THE FUNCTION
C
C     ************************************************************


      DIMENSION F(1:100)

      OPEN (UNIT=3, FILE='DATA18.DAT', STATUS='OLD', ERR=18)
      OPEN (UNIT=1, FILE='PRN')


C     READ THE NUMBER OF VALUES AND THE UNIFORM SPACING
C     BETWEEN THE X- VALUES

      READ(3,*,ERR=19)N,H

      DO 10 I =1, N
      READ (3,*,ERR=19)F(I)

 10   CONTINUE

      WRITE(1,100)
 100  FORMAT(1H ,15X,'SIMPSON RULE TO CALCULATE THE INTEGRATION',\)
      WRITE(1,105)
 105  FORMAT(1X,'OF A FUNCTION',/5X,70(1H*))

      GO TO 20
 19   WRITE(*, 110)
 110  FORMAT(3X,'FILE DOES NOT EXIST')

      GO TO 999
 18   WRITE(*,120)
 120  FORMAT(3X,'ERROR MESSAGE IN THE DATA VALUE')

      GO TO 999


 20   CALL SIMPS (F, N, H, AREA)

      WRITE(1,130)AREA
 130  FORMAT(//, 20X,'AREA OF THE FUNCTION:',F10.4)

      WRITE (1, 140)
 140  FORMAT (5X,70(1H-))

      CLOSE (UNIT=3, STATUS='KEEP')
      CLOSE (UNIT=1)

 999  STOP
      END
```

```
C       ********************************************************
C       THIS PROGRAM PERFORMS SIMPSON'S RULE INTEGRATION OF A
C       FUNCTION DEFINED BY A TABLE OF EQUISPACED VALUES.
C       ********************************************************

        SUBROUTINE SIMPS(F, N, H, AREA)

        DIMENSION F(1:100)

        INTEGER N, NP, NH, NB, NE

C       DETERMINE WHETHER NUMBER OF PANELS IS EVEN
C       NUMBER OF PANELS IS N-1

        NP=N-1
        NH=NP/2
        NB=1
        AREA=1.0

        IF ((NP-2*NH) .NE. 0) THEN

C       NUMBER OF PANELS IS ODD.  THEREFORE USE SIMPSON'S 3/8 RULE
C       ON FIRST THREE AND 1/3 RULE ON THE REMAINING FUNCTION VALUES.
C

        AREA=3.0*H/8*(F(1)+3.0*F(2)+3.0*F(3)+F(4))
        NB=4
        ELSEIF (N .EQ. 4) THEN
        RETURN
        ENDIF


C       USE 1/3 RULE ADD TO FIRST, SECOND AND LAST VALUES.

        AREA=AREA+H/3.0*(F(NB)+4.0*F(NB+1)+F(N))
        NB=NB+2
        IF (NB .EQ. N) THEN
        RETURN
        ENDIF

        NE=N-2
        DO 30 I=NB, NE, 2
        AREA=AREA+H/3.0*(2.0*F(I)+4.0*F(I+1))

 30     CONTINUE
        RETURN
        END
```

DATA18.DAT

```
22  0.113
0.000
0.308
0.995
0.876
0.786
0.720
0.663
0.606
0.545
0.497
0.450
0.403
0.355
0.313
0.275
0.237
0.213
0.171
0.142
0.123
0.109
0.095
```

```
            SIMPSON RULE TO CALCULATE THE INTEGRATION OF A FUNCTION
    ************************************************************************


                  AREA OF THE FUNCTION:    1.0037
    ------------------------------------------------------------------------
```

```
      PROGRAM PROG19

C     ********************************************************
C     THIS PROGRAM SOLVES A SYSTEM OF N FIRST ORDER DIFFENTIAL
C     EQUATIONS BY THE RUNGE-KUTTA FOURTH ORDER METHOD.
C     THE EQUATIONS ARE OF THE FORM DX1/DT = F1(X,T)
C     DX2/DT=F2(X,T), DX3/DT=F3(X,T), ETC.
C
C     DERIVS  =   A SUBROUTINE THAT CALCULATES VALUES OF THE
C                 N DERIVATIVES.  IT IS INVOKED BY THE STATEMENT
C                 CALL DERIVS (X, T, F, N)
C     T0      =   INITIAL VALUE OF THE INDEPENDENT VARIABLE.
C     H       =   THE STEP SIZE, THE INCREMENT TO T0.
C     X0      =   THE ARRAY THAT HOLDS THE INITIAL VALUES OF THE
C                 FUNCTIONS.
C     XEND    =   AN ARRAY THAT RETURNS THE FINAL VALUES OF THE
C                 FUNCTIONS.
C     XWRK    =   AN ARRAY USED TO HOLD INTERMEDIATE VALUES DURING
C                 CALCULATION.  IT MUST BE DIMENSIONED OF SIZE
C                 4 X N IN THE MAIN PROGRAM.
C     N       =   THE NUMBER OF EQUATIONS IN THE PROGRAM.
C     F       =   AN ARRAY THAT HOLDS VALUES OF THE DERIVATIVES.
C     TF      =   FINAL VALUE OF THE INDEPENDENT VARIABLE.
C     ********************************************************

      EXTERNAL DERIVS
      DIMENSION X0(1:4),F(1:4),XEND(1:4),XWRK(1:4,1:4)

      REAL K1, K2, K3, K4, K5
      COMMON/DATA1/X0
      COMMON/DATA2/K1,K2,K3,K4,K5
      COMMON/DATA3/TF

      DATA N,TF/4, 10.0/


      OPEN (UNIT=1, FILE='PRN')

      WRITE (1, 100)
 100  FORMAT (15X, 'RUNGE-KUTTA FOURTH ORDER METHOD FOR A SYSTEM OF')
      WRITE (1, 110)
 110  FORMAT (25X, 'ORDINARY DIFFERENTIAL EQUATIONS',/,5X, 74(1H*),//)

      WRITE (1, 120)
 120  FORMAT (18X,'TIME',3X,'CONC. CA',3X,'CONC. CB',3X,'CONC. CC',
     *          3X,'CONC. CD',/5X, 74(1H-),/)

      T0 = 0.0
      H = 0.5


      WRITE (1, 140) T0, (X0(I), I = 1, N)
 140  FORMAT (14X, F8.2, 5X, 4(F6.2, 5X))

5     CALL RKSYST (DERIVS, T0, H, X0, XEND, XWRK, F, N)
      T0 = T0 + H
      DO I = 1, N
      X0(I) = XEND(I)
      END DO
```

```
      WRITE (1, 150) T0, (X0(I), I = 1, N)
150   FORMAT (14X, F8.2, 5X, 4(F6.2, 5X))

      IF (T0 .LT. TF) THEN
      GO TO 5
      ELSE
      ENDIF

      WRITE (1, 160)
160   FORMAT (5X, 74(1H-))


C     FORM FEED THE PRINTING PAPER TO TOP OF THE NEXT PAGE.

      WRITE (1, *) CHAR(12)


      CLOSE (UNIT = 1)

      STOP
      END
C
C     ***************************************************************
C     THIS PROGRAM SOLVES A SYSTEM OF N FIRST ORDER DIFFERENTIAL
C     EQUATIONS BY THE RUNGE-KUTTA FOURTH ORDER METHOD.
C
C     ***************************************************************

      SUBROUTINE RKSYST (DERIVS, T0, H, X0, XEND, XWRK, F, N)
      DIMENSION X0(1:N), XEND(1:N), XWRK(1:4,1:N), F(1:N)
      COMMON/DATA3/TF

C     CALCULATE THE FIRST ESTIMATE OF THE DATA X'S

      CALL DERIVS (X0, T0, F, N)
      DO I = 1, N
          XWRK(1,I) = H * F(I)
          XEND(I) = X0(I) + XWRK(1,I)/2.
      END DO

C     CALCULATE THE SECOND ESTIMATE , THE XEND VECTOR
C     HOLDS THE X-VALUES

      CALL DERIVS (XEND, T0+H/2.0, F, N)
      DO I = 1, N
          XWRK(2,I) = H * F(I)
          XEND(I)= X0(I) + XWRK(2,I)/2.
      END DO

C     CALCULATE THE THIRD ESTIMATE

      CALL DERIVS (XEND, T0+H/2.0, F, N)
      DO I = 1, N
      XWRK(3,I) = H * F(I)
      XEND(I) = X0(I) + XWRK(3,I)
      END DO

C     CALCULATE THE FOURTH ESTIMATE
```

```
      CALL DERIVS (XEND, T0+H, F, N)
      DO I = 1, N
      XWRK(4,I) = H * F(I)
      END DO

C     COMPUTE THE X VALUES AT THE END OF THE INTERVAL FROM A
C     WEIGHTED AVERAGE OF THE FOURTH ESTIMATES.

      DO I=1,N
      XEND(I)=X0(I)+(XWRK(1,I)+2.0*XWRK(2,I)+2.*XWRK(3,I)+
     +  XWRK(4,I))/6.
      END DO


      RETURN
      END


C     ***************************************************************
C     THIS PROGRAM DEFINES THE FUNCTIONS OF THE ORDINARY DIFFERENTIAL
C     EQUATIONS IN TERMS OF THE F'S AND THE X'S.
C     ***************************************************************


      SUBROUTINE DERIVS (X, T, F, N)
      DIMENSION X(1:N),F(1:N)
      COMMON/DATA2/K1,K2,K3,K4,K5
      REAL K1,K2,K3,K4,K5
      F(1)=K5*X(2)-K1*X(1)
      F(2)=K1*X(1)+K4*X(4)-(K2+K3+K5)*X(2)
      F(3)=K2*X(2)
      F(4)=K3*X(2)-K4*X(4)
      RETURN
      END

C     ***************************************************************
C     THIS STORES THE DATA FOR THE RATE CONSTANTS AND THE INITIAL
C     STARTING VALUES OF THE CONCENTRATIONS.
C     ***************************************************************

      BLOCK DATA
      DIMENSION X0(4)
      COMMON/DATA1/X0
      COMMON/DATA2/K1,K2,K3,K4,K5
      REAL K1,K2,K3,K4,K5
      DATA X0/9.9,0.0,0.0,0.5/
      DATA K1/0.4/,K2/0.16/,K3/0.13/,K4/0.08/,K5/0.10/
      END
```