Appendix H

NUMERICAL TECHNIQUES

This appendix shows the various numerical techniques that can be employed in solving design problems, which could pose difficulty if an analytical method is used. The numerical methods can readily be incorporated into computer programs to obtain results of design problems.

H.1 SIMPSON’S RULE FOR AREA UNDER THE CURVE

Simpson’s rule is a numerical integration technique that is widely used in calculating the area under the curve. It is simple and has a greater degree of accuracy than the trapezoidal rule. The Simpson’s 1/3 rule is based on quadratic polynomial interpolation.

Figure H-1 shows a section of a curve and three coordinates erected to it at equally spaced intervals along the x-axis. Simpson’s rule states that the area \( P \) is given approximately by the formula

\[
\text{Area} = \frac{h}{3} (y_1 + 4y_2 + y_3)
\]

(H-1)

If we reduce the step size \( h \), the result becomes more accurate. The interval over which the integral is to be taken is divided into larger number of equal sub intervals as shown in Figure H-2.

We will divide the total area into four sections, namely \( \text{PRR}' \), \( \text{RSS}' \), \( \text{SSTT}' \), and \( \text{TQQ}' \).

We shall write down the expression for each area and sum them up to obtain the total area \( \text{PQQ}' \).

Area of \( \text{PRR}' \) = \( \frac{h}{3} (y_1 + 4y_2 + y_3) \)

Area of \( \text{RSS}' \) = \( \frac{h}{3} (y_3 + 4y_4 + y_5) \)

Area of \( \text{SSTT}' \) = \( \frac{h}{3} (y_5 + 4y_6 + y_7) \)

Area of \( \text{TQQ}' \) = \( \frac{h}{3} (y_7 + 4y_8 + y_9) \)

The total area is the sum of the areas \( \text{PRR}' \), \( \text{RSS}' \), \( \text{SSTT}' \), and \( \text{TQQ}' \).

\[
\text{Total area} = \frac{1}{3} h (y_1 + 4y_2 + y_3 + 4y_4 + y_5 + 4y_6 + y_7 + 4y_8 + y_9)
\]

(H-6)

The Simpson’s 1/3 rule for a quadratic integrated over two \( \Delta x \) intervals that are of uniform width or panel is

\[
\text{Area} = \int_a^b f(x) \, dx
\]

\[
= \frac{h}{3} (y_1 + 4y_2 + 2y_3 + 4y_4 + 2y_5 + \cdots + 2y_{n-1} + 4y_n + y_{n+1}) + E
\]

(H-7)

width \times \text{average height}

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

\[
\text{Area} = \int_a^b f(x) \, dx = \frac{3h}{8} (y_1 + 3y_2 + 3y_3 + 2y_4 + 3y_5 + \cdots + 3y_6 + \cdots + 2y_{n-2} + 3y_{n-1} + 3y_n + y_{n+1}) + E
\]

(H-8)

= width \times height
**EXAMPLE H-1**

A tracer experiment was carried out in a nozzle type reactor of volume $V = 5.13$ L, with liquid rate at 2.9 L/min. Table H-1 shows data for the exit age distribution $E(\theta)$ against the dimensionless residence time $\theta$. Determine the area under the distribution curve.

<table>
<thead>
<tr>
<th>$\theta$</th>
<th>$E(\theta)$</th>
<th>$\theta$</th>
<th>$E(\theta)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>1.243</td>
<td>0.403</td>
</tr>
<tr>
<td>0.113</td>
<td>0.308</td>
<td>1.356</td>
<td>0.355</td>
</tr>
<tr>
<td>0.226</td>
<td>0.956</td>
<td>1.469</td>
<td>0.313</td>
</tr>
<tr>
<td>0.339</td>
<td>0.876</td>
<td>1.582</td>
<td>0.275</td>
</tr>
<tr>
<td>0.452</td>
<td>0.786</td>
<td>1.695</td>
<td>0.237</td>
</tr>
<tr>
<td>0.565</td>
<td>0.720</td>
<td>1.808</td>
<td>0.213</td>
</tr>
<tr>
<td>0.678</td>
<td>0.663</td>
<td>1.921</td>
<td>0.171</td>
</tr>
<tr>
<td>0.791</td>
<td>0.606</td>
<td>2.034</td>
<td>0.142</td>
</tr>
<tr>
<td>0.904</td>
<td>0.545</td>
<td>2.147</td>
<td>0.123</td>
</tr>
<tr>
<td>1.017</td>
<td>0.497</td>
<td>2.260</td>
<td>0.109</td>
</tr>
<tr>
<td>1.130</td>
<td>0.450</td>
<td>2.373</td>
<td>0.095</td>
</tr>
</tbody>
</table>


A computer program PROG7 has been developed to determine the area under the curve from the residence time distribution function $E(t)$ and expressed as follows:

$$\int_0^\infty E(t) \, dt = 1 \quad (H-9)$$

The average time spent by material flowing at a rate $q$ through a volume $V$ equals $\bar{t} = \frac{V}{q} \quad (H-10)$

We can also express Eq. (H-9) in the form of dimensionless time where

$$\theta = \frac{tq}{V} \quad (H-11)$$

A computer program PROG7 has been developed to determine the area under the curve. Simpson's rule is also easy to use in Microsoft Excel spreadsheet. We can carry this out by entering the $x$-values in one column, the $y$-values in the next column. We then follow this procedure by an additional column containing the $y$-values multiplied by their appropriate constants (i.e., multiplied by 4 or 2, except the initial and final values). We sum the last column to obtain the value of the integral and multiply this by $\Delta x/3$.

**H.2 NON-LINEAR EQUATIONS**

The Newton–Raphson’s iterative method is a process for the determination of a real root of an equation $f(x) = 0$, given just one point close to the desired root (Figure H-4). If we let $x_0$ represent the known approximate value of the root of $f(x) = 0$, and $h$ be the difference between the true value $\alpha$ and the approximate value, we have

$$\alpha = x_0 + h$$

From Taylor series,

$$f(x) = f(x_0) + h f'(x_0) + \frac{h^2}{2!} f''(x_0) + \cdots + \frac{h^n}{n!} f^n(x_0)$$

(H-13)

about $x_0$, gives

$$f(\alpha) = f(x_0 + h) = f(x_0) + h f'(x_0) + \frac{h^2}{2!} f''(\xi)$$

(H-14)

where $\xi = x_0 + \theta h, 0 < \theta < 1$, lies between $\alpha$ and $x_0$. Ignoring the remainder term and writing $f(\alpha) = 0$, we have

$$f(x_0) + h f'(x_0) \approx 0$$

(H-15)

so that

$$h \approx \frac{f(x_0)}{f'(x_0)}$$

(H-16)
SOLUTION OF NON-LINEAR EQUATIONS

A developed computer program PROG6.FOR is used to solve non-linear equations as illustrated by the following example.

Figure H-4: Newton–Raphson’s method.

Therefore, the next root that gives a better estimate than $x_0$ is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \quad (H-17)$$

Better approximations may be obtained by repetition (iteration) of the process. We may write this as

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (H-18)$$

Each iteration provides the point at which the tangent at the original point cuts the $x$-axis as shown in Figure H-4. The equation of the tangent at the point $[x_n, f(x_n)]$ is

$$y - f(x_n) = f'(x_n)(x - x_n) \quad (H-19)$$

Therefore the point $(x_{n+1}, 0)$ corresponds to

$$-f(x_n) = f'(x_n)(x_{n+1} - x_n) \quad (H-20)$$

which gives

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (H-21)$$

A developed computer program PROG6.FOR is used to solve non-linear (e.g., Colebroke–White) equation using Newton–Raphson’s method of iteration in Chapter 9.

SOLUTION OF NON-LINEAR EQUATIONS

Consider a set of $N$ non-linear equations of the form

$$F(x, y, \ldots) = 0 \quad (H-22)$$
$$G(x, y, \ldots) = 0$$

where $x, y, \ldots$ are the roots of the $N$ equations. These equations can be solved explicitly for the roots. If we consider some points $(x_1, y_1)$ near the root of definition for the functions $F, G$, we can expand both functions by an $N$-dimensional Taylor series about the point $(x_1, y_1)$ as

$$F(x, y, \ldots) = F(x_1, y_1, \ldots) + \frac{\partial F}{\partial x} \bigg|_{(x_1, y_1)} (x-x_1) + \cdots$$
$$G(x, y, \ldots) = G(x_1, y_1, \ldots) + \frac{\partial G}{\partial x} \bigg|_{(x_1, y_1)} (x-x_1) + \cdots \quad (H-23)$$

Truncating the series after the first-order derivative and rewriting in matrix form will yield

$$\begin{bmatrix} \frac{\partial F}{\partial x} & \frac{\partial G}{\partial x} & \cdots \\ \frac{\partial F}{\partial y} & \frac{\partial G}{\partial y} & \cdots \end{bmatrix} \begin{bmatrix} x-x_1 \\ y-y_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} F-F_0 \\ G-G_0 \end{bmatrix} \quad (H-24)$$

We can solve for the roots $x, y, \ldots$, which give $F = G = \ldots = 0$ as

$$\begin{bmatrix} x \\ y \\ \vdots \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \end{bmatrix} + \begin{bmatrix} \frac{\partial F}{\partial x} & \frac{\partial G}{\partial x} & \cdots \\ \frac{\partial F}{\partial y} & \frac{\partial G}{\partial y} & \cdots \end{bmatrix}^{-1} \begin{bmatrix} F_0 \\ G_0 \end{bmatrix} \quad (H-25)$$

for $N = 1$

Equation (H-25) becomes

$$x = x_1 - \frac{F_0}{\frac{\partial F}{\partial x}} \quad (H-26)$$

This is Newton–Raphson’s method for finding the roots of an equation.

When $N = 2$

$$x = x_1 - \frac{F_0}{\frac{\partial F}{\partial x}} \bigg|_{(x_1, y_1)} - \frac{G_0}{\frac{\partial G}{\partial x}} \bigg|_{(x_1, y_1)}$$
$$y = y_1 - \frac{G_0}{\frac{\partial G}{\partial y}} \bigg|_{(x_1, y_1)} + \frac{F_0}{\frac{\partial F}{\partial y}} \bigg|_{(x_1, y_1)} \quad (H-27)$$

Equation (H-27) is a two-dimensional generalization of Newton’s method. This technique is often employed to solve large sets of non-linear algebraic equations. Care must be taken in choosing initial guess $(x_1, y_1)$ quite close to the final roots, as the algorithm may diverge. A developed computer program PROG8 is used to solve non-linear equations as illustrated by the following example.

The computer program PROG8 can be used to solve any number of non-linear equations. The partial derivatives of the functions are estimated by the difference quotients when a variable is perturbed by an amount equal to a small value $(\Delta)$ used in the program to perturb the X-values. Table H-2 shows the computer results of Example H-2.

H.3 SOLUTION OF SIMULTANEOUS, FIRST-ORDER, ORDINARY DIFFERENTIAL EQUATIONS

Analytical solutions to complex kinetic reactions in reactor systems are time-consuming and intractable. The designer must resort to
Example H-2

A pair of reactions A + B $\xrightarrow{k_1}$ 2C and A + C $\xrightarrow{k_2}$ D are conducted in a four-stage continuous flow stirred tank reactors (CFSTR) with $C_{A0} = 0.9$ mol/m$^3$, $C_{B0} = 0.3$ mol/m$^3$, and $C_{C0} = C_{D0} = 0$ mol/m$^3$. The residence time in each stage is 10 min. Determine the exit concentrations of A and B in the four CFSTRs. The rate expressions are

$$-r_A = k_1 C_A C_B + k_2 C_A C_C$$

and

$$-r_B = k_1 C_A C_B$$

where

$$k_1 = 0.3 \text{ mol/min}$$

$$k_2 = 0.15 \text{ mol/min}$$

Solution

Figure H-5 shows a battery of CFSTR with $C_{A0} = 0.9$ mol/m$^3$ and $C_{B0} = 0.3$ mol/m$^3$ in the first tank and where, $V_1 = V_2 = V_3 = V_4 = V$, $u = \text{volumetric flow rate}$, and the residence time, $t = V/u = 10$ min.

General mass balance

Input by flow = output by flow + disappearance by reaction + accumulation

Assuming that the operation is at steady state and, therefore, accumulation = 0

Mass balance on first CFSTR

The material balance on species A is:

$$u C_{A0} = u C_{A1} + (k_1 C_{A1} C_B + k_2 C_{A1} C_C) V_R$$

(H-28)

The stoichiometry between species A, B, and C is

$$C_C = 3(C_{B0} - C_B) - (C_{A0} - C_A)$$

$$= 0.9 - 3C_B - 0.9 + C_A$$

$$C_C = C_A - 3C_B$$

(H-29)

Substituting Eq. (H-29) into Eq. (H-28) gives

$$C_{A0} = C_{A1} + C_{A1} \tilde{t} (k_1 C_{B1} + k_2 (C_{A1} - 3C_{B1}))$$

or

$$0.9 = C_{A1} + 3C_{A1} C_{B1} + 1.5 C_{A1} (C_{A1} - 3C_{B1})$$

(H-30)

Material balance on species B is

$$u C_{B0} = u C_{B1} + (C_{A1} - 3C_{B2}) V_R$$

or

$$0.3 = C_{B1} + 3C_{A1} C_{B1}$$

(H-31)

Second CFSTR Material balance on species A is

$$u C_{A1} = u C_{A2} + [k_1 C_{A2} C_{B2} + k_2 C_{A2} (C_{A2} - 3C_{B2})] V_R$$

or

$$C_{A1} = C_{A2} + 10 [0.3 C_{A2} C_{B2} + 0.15 C_{A2} (C_{A2} - 3C_{B2})]$$

(H-32)

Material balance on species B is

$$u C_{B1} = u C_{B2} + C_{A2} C_{B2} V_R$$

or

$$C_{B1} = C_{B2} + 3C_{A2} C_{B2}$$

(H-33)

Rearranging Eq (H-35) and (H-36) yields

$$C_{A2} + 3C_{A2} C_{B2} + 1.5 C_{A2} (C_{A2} - 3C_{B2}) = C_{A1}$$

$$C_{B2} + 3C_{A2} C_{B2} = C_{B1}$$

(H-34)

Third CFSTR

The material balances on species A and B are

$$C_{A3} + 3C_{A3} C_{B3} + 1.5 C_{A3} (C_{A3} - 3C_{B3}) = C_{A2}$$

$$C_{B3} + 3C_{A3} C_{B3} = C_{B2}$$

(H-35)

Figure H-5  A battery of CFSTR.
**EXAMPLE H-2—(continued)**

Fourth CFSTR
The material balances on species A and B are

\[ C_{A_t} + 3C_{A_t}C_{B_t} + 1.5C_{A_t}(C_{A_t} - 3C_{B_t}) = C_{A_3} \]
\[ C_{B_t} + 3C_{A_t}C_{B_t} = C_{B_3} \]

These are eight non-linear equations involving the material balances of species A and B in the four stirred tank reactors. Rearranging these equations yields the following:

\[ 1.5C_{A_1}^2 - 1.5C_{A_1}C_{B_1} + C_{A_1} - 0.9 = 0 \]
\[ 3C_{A_1}C_{B_1} + C_{B_1} - 0.3 = 0 \]
\[ 1.5C_{A_2}^2 - 1.5C_{A_2}C_{B_2} + C_{A_2} - 0.1 = 0 \]
\[ 3C_{A_2}C_{B_2} + C_{B_2} - 0.1 = 0 \]
\[ 1.5C_{A_3}^2 - 1.5C_{A_3}C_{B_3} + C_{A_3} - 0.3 = 0 \]
\[ 3C_{A_3}C_{B_3} + C_{B_3} - 0.3 = 0 \]
\[ 1.5C_{A_4}^2 - 1.5C_{A_4}C_{B_4} + C_{A_4} - 0.1 = 0 \]
\[ 3C_{A_4}C_{B_4} + C_{B_4} - 0.1 = 0 \]

A computer program PROG8 was developed using the Newton–Raphson’s method to determine the outlet concentration of species A and B from the four stirred tank reactors. The eight equations (Eqs H-40) are supplied as functions in the subroutine of PROG8 as

\[ F(1) = 1.5 * X(1) * X(1) - 1.5 * X(1) * X(2) + X(1) - 0.9 \]
\[ F(2) = 3.0 * X(1) * X(2) + X(2) - 0.3 \]

Each member of the set of equations. This causes practical problems if the set is *stiff* where some members of these equations have characteristic times much smaller than other members of the equations. An example is the free-radical kinetics reaction, which has rates that may differ by three orders of magnitude.

In general, a system of \( n \)th first-order equations will be of the form

\[
\frac{dy}{dx} = f(x, y_1, y_2, \ldots, y_{n-1}), \quad i = 0, 1, 2, 3, \ldots, n
\]

(H-42)

with \( n \) initial conditions \( y_i(x_0) = A_i, \quad i = 0, 1, 2, \ldots, n \)

Consider the system of two equations:

\[
\frac{dy}{dx} = f(x, y, z)(y_0) = y_0 \quad \frac{dz}{dx} = g(x, y, z)(z_0) = z_0
\]

(H-43)

We may advance the solution of \( y \) and \( z \) to new values at \( x_1 = x_0 + h \) using any of the one-step or Runge–Kutta methods.

In general, our solutions will be advanced using expressions of the form

\[ y(x_1) = y(x_0) + K \quad z(x_1) = z(x_0) + L \]

where the nature of \( K \) or \( L \) depends on the method being applied. For the Runge–Kutta fourth order

\[ K = \frac{K_1 + 2K_2 + 2K_3 + K_4}{6} \]

(H-47)

**TABLE H-2** Newton Method For Solving Non-linear Equations

<table>
<thead>
<tr>
<th>Initial roots</th>
<th>Final roots</th>
</tr>
</thead>
<tbody>
<tr>
<td>( X(1) = 0.1000 )</td>
<td>( X(1) = 0.54599 )</td>
</tr>
<tr>
<td>( X(2) = 0.1000 )</td>
<td>( X(2) = 0.11372 )</td>
</tr>
<tr>
<td>( X(3) = 0.1000 )</td>
<td>( X(3) = 0.37030 )</td>
</tr>
<tr>
<td>( X(4) = 0.1000 )</td>
<td>( X(4) = 0.05386 )</td>
</tr>
<tr>
<td>( X(5) = 0.1000 )</td>
<td>( X(5) = 0.27180 )</td>
</tr>
<tr>
<td>( X(6) = 0.1000 )</td>
<td>( X(6) = 0.02958 )</td>
</tr>
<tr>
<td>( X(7) = 0.1000 )</td>
<td>( X(7) = 0.21107 )</td>
</tr>
<tr>
<td>( X(8) = 0.1000 )</td>
<td>( X(8) = 0.01792 )</td>
</tr>
</tbody>
</table>
968 NUMERICAL TECHNIQUES

and

\[ L = \frac{L_1 + 2L_2 + 2L_3 + L_4}{6} \]

where

\[ K_1 = hf(x_0, y_0, z_0) \]

\[ L_1 = hg(x_0, y_0, z_0) \]

\[ K_2 = hf \left( x_0 + \frac{1}{2} h, y_0 + \frac{1}{2} K_1, z_0 + \frac{1}{2} L_1 \right) \]

\[ L_2 = hg \left( x_0 + \frac{1}{2} h, y_0 + \frac{1}{2} K_1, z_0 + \frac{1}{2} L_1 \right) \]

\[ K_3 = hf \left( x_0 + \frac{1}{2} h, y_0 + \frac{1}{2} K_2, z_0 + \frac{1}{2} L_2 \right) \]

\[ L_3 = hg \left( x_0 + \frac{1}{2} h, y_0 + \frac{1}{2} K_2, z_0 + \frac{1}{2} L_2 \right) \]

\[ K_4 = hf \left( x_0 + h, y_0 + K_3, z_0 + L_3 \right) \]

\[ L_4 = hg(x_0 + h, y_0 + K_3, z_0 + L_3) \]

**H.4 EXTENSION OF RUNGE-KUTTA METHODS**

**RUNGE-KUTTA-GILL METHOD**

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

\[ \frac{dy}{dx} = f(x, y), y(x_0) = y_n \]

\[ y_{n+1} = y_n + \frac{1}{6} \left[ k_1 + 2 \left( 1 - \frac{1}{\sqrt{2}} \right) k_2 + 2 \left( 1 + \frac{1}{\sqrt{2}} \right) k_3 + k_4 \right] \] + \( O(h^5) \)

where

\[ k_1 = hf(x_n, y_n) \]

\[ k_2 = hf \left( x_n + \frac{h}{2}, y_n + \frac{1}{2} k_1 \right) \]

\[ k_3 = hf \left( x_n + \frac{h}{2}, y_n + \left[ \frac{-1}{2} + \frac{1}{\sqrt{2}} \right] k_1 + \left[ 1 - \frac{1}{\sqrt{2}} \right] k_2 \right) \]

\[ k_4 = hf \left( x_n + h, y_n - \frac{1}{\sqrt{2}} k_3 + \left[ 1 + \frac{1}{\sqrt{2}} \right] k_4 \right) \]

**THE RUNGE-KUTTA-MERSON METHOD**

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

\[ k_1 = hf(x_n, y_n) \]

\[ k_2 = hf \left( x_n + \frac{h}{3}, y_n + \frac{k_1}{3} \right) \]

\[ k_3 = hf \left( x_n + \frac{h}{3}, y_n + \frac{k_1}{6} + \frac{k_2}{3} \right) \]

The wave equation

\[ \frac{\partial^2 u}{\partial t^2} = \frac{T_0}{w} \frac{\partial^2 u}{\partial x^2} \]

is a hyperbolic type where the coefficients \( A = 1, B = 0, C = -1 \) and \( B^2 - 4AC = 4 \)

**PARABOLIC EQUATION**

The heat conduction equation

\[ \frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2} \]

is a parabolic type where the coefficients \( A = 0, B = 0, C = 1 \) and \( B^2 - 4AC = 0 \).
A method for solving the above partial differential equations is to replace the derivatives by difference quotients; that is, converting the equation to a difference equation. We can then write the difference equation that corresponds to each point at the intersections (nodes) of a grid work that subdivides the region of interest at which the function values are known. Solving these equations

**Example H-3**

Consider the dynamic response of component A in a CFSTR undergoing a first-order reaction represented by

\[-r_A = k C_A \left( \frac{\text{mol}}{\text{dm}^3 \cdot \text{min}} \right)\]

The rate constant decay is

\[k = k_o - a t^2 \left( \text{min}^{-1} \right)\]

The first-order differential equation from the mass balance is:

\[\frac{dC_A}{dt} = \frac{u}{V_R} C_{A0} - \left( \frac{u}{V_R} + k_o - a t^2 \right) C_A\] (H-62)

Spreadsheet programming: The Excel program (Prog8a.xls) is used to create numerical solutions using the fourth-order Runge–Kutta by rearranging the above equation to yield

\[\Delta C_A = C_A - C_{A0} = \left[ \frac{\alpha C_{A0}}{V_R} - \left( \frac{u}{V_R} + k_o - a t^2 \right) C_A \right] \Delta t\] (H-63)

The transient response concentration \(C_A\) is

\[C_A = C_{A0} + \left[ \frac{\alpha C_{A0}}{V_R} - \left( \frac{u}{V_R} + k_o - a t^2 \right) C_A \right] \Delta t\] (H-64)

Figure H-6 shows the spreadsheet snap shot of the numerical Runge–Kutta method. Further details are given by Coker, A.K. (Hydroc. Proc., Dec 2004, pp. 77–85).

![Figure H-6](image_url)

**Figure H-6** Simulation of CFSTR reaction with the Excel program.
NUMERICAL TECHNIQUES

simultaneously gives values for the function at each node that approximates the true values.

Let $h = \Delta x$ equal spacing of grid work in the $x$ direction as shown in the Figure H-7.

From Taylor series,

$$f(x_n + h) = f(x_n) + f'(x_n)h + \frac{f''(x_n)h^2}{2} + \frac{f'''(x_n)h^3}{6} + \ldots \text{ for } x_n < \xi_1 < x_n + h$$

and

$$f(x_n - h) = f(x_n) - f'(x_n)h + \frac{f''(x_n)h^2}{2} - \frac{f'''(x_n)h^3}{6} + \ldots \text{ for } x_n - h < \xi_2 < x_n$$

It follows that

$$f(x_n + h) + f(x_n - h) = 2f(x_n) + f''(x_n)h^2 + \frac{f'''(\xi)h^3}{12}$$

or

$$f''(x_n) = \frac{f''(\xi)h^2}{12} = \frac{f(x_n + h) - 2f(x_n) + f(x_n - h)}{h^2} \quad \text{for} \quad x_n - h < \xi < x_n + h$$

Using the subscript notation, we have

$$f'(x_n) + O(h^2) = \frac{f_{n+1} - 2f_n + f_{n-1}}{h^2}$$

where the subscripts of $f$ indicate the $x$-values at which it is evaluated. The order relation $O(h^2)$ shows that error approaches proportionality to $h^2$ as $h \to 0$.

Similarly, the first derivative is approximated to

$$f(x_n + h) - f(x_n - h) = 2f'(x_n)h + O(h^3)$$  \hspace{1cm} \text{(H-75)}

or

$$f'(x_n) = \frac{f(x_n + h) - f(x_n - h)}{2h}$$  \hspace{1cm} \text{(H-76)}

$$= \frac{f_{n+1} - f_{n-1}}{2h}$$  \hspace{1cm} \text{(H-77)}

The first derivative could also be approximated by the forward or backward difference, but would have an error of $O(h)$. The central difference approximation gives the more accurate approximation.

When $f$ is a function of both $x$ and $y$, we can obtain the second partial derivative with respect to $x, \frac{\partial^2 u}{\partial x^2}$, by holding $y$ constant and evaluating the function at three points where $x$ equals $x_n, x_n + h,$ and $x_n - h$. Correspondingly, the partial derivative $\frac{\partial^2 u}{\partial y^2}$ is determined by holding $x$ constant.

Consider the Laplace equation on a region in the $xy$ plane. We subdivide the region with equipspaced lines parallel to the $x$- and $y$-axes. Consider the region near $(x_i, y_j)$.

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$  \hspace{1cm} \text{(H-78)}

We can replace the derivatives by difference quotients which approximate the derivatives at point $(x_i, y_j)$. Then we have

$$\nabla^2 u(x_i, y_j) = \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{\Delta x^2} + \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{\Delta y^2} = 0$$  \hspace{1cm} \text{(H-79)}

or

$$\nabla^2 u(x_i, y_j) = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} = 0$$  \hspace{1cm} \text{(H-80)}

If we let $\Delta x = \Delta y = h$, the PDE becomes

$$\nabla^2 u_{i,j} = \frac{1}{h^2} \left( u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} \right) = 0$$  \hspace{1cm} \text{(H-81)}

This is known as the standard five-point formula, as five points are involved in the relationship of Eq. (H-81) which points to the right, left, above, and below the central point $(x_i, y_j)$. We can write Eq. (H-81) as

$$u_{i,j} = \frac{1}{4} \left( u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} \right)$$  \hspace{1cm} \text{(H-82)}
Instead of Eq. (H-82), we may also use the formula
\[ u_{i,j} = \frac{1}{4} \left( u_{i-1,j-1} + u_{i+1,j-1} + u_{i+1,j+1} + u_{i-1,j+1} \right) \]  
(H-83)

Therefore, the general procedure is to approximate the PDE by a finite difference transformation and then to obtain the solution at the mesh points, using the finite difference approximations. Other numerical methods of solution are the implicit Crank–Nicolson method or the alternating direction implicit scheme (ADI) by Peaceman and Rachford. Details of these methods are illustrated in Numerical analysis texts.

**H.6 THE EXPLICIT METHOD**

The explicit method divides space and time into discrete uniform subintervals and replaces both time and space derivatives by finite difference approximations, permitting one to easily compute values of the function at a time \( \Delta t \) after the initial time. These values are then used to compute a second set of values and the process is repeated. An approach to solving parabolic partial differential equations by a numerical method is to replace the partial derivatives by finite-difference approximations. An example is the one-dimensional heat flow equation.

\[ \frac{\partial^2 u}{\partial x^2} = \frac{cp}{\Delta t} \frac{\partial u}{\partial t} \]  
(H-84)

We can use the relations

\[ \frac{\partial^2 u}{\partial x^2} \bigg|_{x=x_i} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x^2) \]  
(H-85)

and

\[ \frac{\partial u}{\partial t} \bigg|_{t=t_j} = \frac{u^{j+1}_i - u^j_i}{\Delta t} + O(\Delta t) \]  
(H-86)

The subscripts are used to denote the position and superscripts for time.

Substituting Eqs (H-85) and (H-86) into Eq. (H-84) and solving for \( u^{j+1}_i \) gives the equation for the forward-difference method:

\[ u^{j+1}_i = \frac{k\Delta t}{\rho c^2} \left( u_{i+1} - 2u_i + u_{i-1} \right) + \left( 1 - \frac{2k\Delta t}{\rho c^2} \right) u^j_i \]  
(H-87)

Solving for \( u^{j+1}_i \) in terms of the temperatures at time \( t_j \) in Eq. (H-87) for a parabolic partial-differential equation involves subdividing the length into uniform subintervals and applying the finite-difference approximation to Eq. (H-84) at each point where \( u \) is not known. Eq. (H-87) then gives the values of \( u \) at each interior point at \( t = t_1 \) since the values at \( t = t_0 \) are given by the initial conditions. It can be used to get values at \( t_0 \) using the values at \( t_1 \) as initial conditions, so the solution can be stepped forward in time. At the end points, the boundary conditions will determine \( u \).

The relative size of the time and distance steps, \( \Delta t \) and \( \Delta x \), affects Eq. (H-87). If the ratio of \( \Delta t / (\Delta x)^2 \) is chosen so that \( k\Delta t / \rho c^2(\Delta x)^2 = 1/4 \), the equation is simplified in that the last term vanishes and we have

\[ u^{j+1}_i = \frac{1}{2} \left( u_{i+1}^j + u_{i-1}^j \right) \]  
(H-88)

If the value \( k\Delta t / \rho c^2(\Delta x)^2 \) is chosen as less than one-half, there will be improved accuracy (limited by the errors dependent on the size of \( \Delta x \)). If the value is chosen greater than one-half, which would reduce the number of calculations required to advance the solution through a given interval of time, the phenomenon of instability sets in. An example is used to illustrate the explicit method.

**H.7 INITIAL VALUE METHODS**

Another method for solving boundary value problems is to treat them like initial value problems. Since a second-order equation can be reduced to two first-order equations, two initial conditions are required. One condition will be known at a boundary: simply assume a value for the other dependent variable at that same boundary, integrate to the other side, and check if the required boundary condition is satisfied. If not, change the initial value and repeat the integration. The result of this method depends upon the skill with which the iterations are employed in the program.

**FINITE DIFFERENCE METHOD IN EXCEL**

The Microsoft Excel spreadsheet can be used to program the finite difference method and use the “Calculation” feature to handle the circular reference. First, turn off the iteration, prepare the spreadsheet, and then turn the calculation back on. Whether this converges depends upon the initial guess.

Consider a reaction and diffusion in a flat layer with a first-order reaction represented by

\[ \frac{dc}{dx} = kc, \text{ with boundary conditions } \frac{dc}{dx}(0) = 0, \quad c(R) = c_o \]  
(H-89)

where

- \( D \) = diffusivity
- \( c \) = concentration
- \( x \) = position (distance).

The idea in the finite difference method is that the differential equation, valid for all \( x \) positions, is replaced by a set of equations representing the equation only at the grid points. Using the finite difference method as derived above, the equation at the grid point is

\[ D \frac{c_i - c_{i+1}}{\Delta x^2} = kc_i, \text{ where } c_i = c(x) \]  
(H-90)

Re-arranging Eq. (H-90) to the following form gives

\[ c_i = \frac{c_{i+1} + c_{i-1}}{2(1 + k\Delta x^2/D)} \]  
(H-91)
Identify the cells (B1, C1, and D1) in the spreadsheet with the value of concentration at a node \( i \) as shown below.

<table>
<thead>
<tr>
<th>( k\Delta x^2/D )</th>
<th>B1</th>
<th>C1</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c_{i-1} )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( c_i )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( c_{i+1} )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Put the value of \( k\Delta x^2/D \) in cell A1. The equation for cell C1 is represented by Eq. (H-92) as

\[
= \frac{(D1 + B1)}{(2 + \Delta AS1)} \quad (H-92)
\]

Next, copy this equation over a series of cells that correspond to the number of grid points. For the first and last cell (grid point) use a different equation appropriate to the boundary condition. Then turn on the iteration feature to get the solution.

The next step is to resolve the problem using more grid points and a smaller \( \Delta x \). This entails that the spreadsheet must be programmed, which gives an indication of whether the answer changes much as the mesh is refined. This ensures that the problem is accurately solved.

### H.8 FINITE DIFFERENCE METHOD FOR ELLIPTIC EQUATIONS

Consider the heat condition equation with a heat generation term as:

\[
k \left( \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) = Q \quad (H-93)
\]

The finite difference form of Eq. (H-93) is solved on a worksheet by replacing the derivatives with central differences centered on grid point \((i, j)\):

\[
\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + \frac{T_{i,j+1} + T_{i,j-1}}{\Delta y^2} \frac{Q}{k} = \frac{T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}}{4} - \frac{\Delta^2 Q}{k} \quad (H-94)
\]

where \( T_{i,j} \) is the temperature at the \( i \)th location in the \( x \)-direction and the \( j \)th location in the \( y \)-direction. Assuming that the grid spacing \( \Delta x = \Delta y \), Eq. (H-94) is rearranged as

\[
T_{i,j} = \frac{T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}}{4} - \frac{\Delta^2 Q}{k} \quad (H-95)
\]

The spreadsheet is then arranged as shown below:

<table>
<thead>
<tr>
<th>A4 – ( \Delta^2 Q/k )</th>
<th>C4 – ( T_{i,j-1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>B5 – ( T_{i-1,j} )</td>
<td>C5 – ( T_{i,j} )</td>
</tr>
<tr>
<td>D5 – ( T_{i+1,j} )</td>
<td>C6 – ( T_{i,j+1} )</td>
</tr>
</tbody>
</table>

When the value of \( \Delta^2 Q/k \) is placed in cell A4, the equation for cell C5 is:

\[
= \frac{(D5 + B5 + C6 + C4)}{(4 + \Delta AS4)} \quad (H-96)
\]

In the worksheet, the boundaries of the problem are set with the fixed or derivative values of the boundary conditions, and the interior points are set with Eq. (H-96). Then copy this for every internal grid point, set the boundary equations, and turn on the iteration feature to obtain the solution. This is further carried out again with a finer mesh to assess the accuracy. If the heat generation term depends upon temperature, it is easy to include that complication just by inserting the formula in place of \( \Delta AS4 \).

### EXAMPLE H-4

A large flat steel plate is 2 cm thick. If the initial temperatures \( ^\circ C \) within the plate are given, as a function of the distance from one face, by the equations

\[
u = 100x \quad \text{for } 0 \leq x \leq 1
\]

\[
u = 100(2 - x) \quad \text{for } 1 \leq x \leq 2
\]

find the temperatures as a function of \( x \) and \( t \) if both faces are maintained at \( 0^\circ C \) [3].

Data:

\[
k = 0.13 \text{ cal/s cm } ^\circ C
\]

\[
c = 0.11 \text{ cal/g } ^\circ C
\]

\[
\rho = 7.8 \text{ g/cm}^3
\]

Solution

since the plate is large, the lateral flow of heat relative to the flow perpendicular to the faces can be neglected, and therefore Eq. (H-84) is used for heat flow in one direction. In order to use Eq. (H-88) as an approximation to the physical problem, we subdivide the total thickness into an integral number of spaces.

If we use \( \Delta x = 0.25 \), this gives eight subdivisions. From Eq. (H-88), \( \Delta t \) is fixed by the relation

\[
\frac{k\Delta t}{c\rho(\Delta x)^2} = \frac{1}{2}
\]

\[\Delta t = \frac{(0.11)(7.8)(0.25)^2}{(2)(0.13)} = 0.206 s\]

The boundary conditions are

\[
u(0, t) = 0 \quad \nu(2, t) = 0
\]

The initial conditions are

\[
u(x, 0) = 100x \quad \text{for } 0 \leq x \leq 1
\]

\[
u(x, 0) = 100(2 - x) \quad \text{for } 1 \leq x \leq 2
\]

The computer program PROG9 uses the simple algorithm of Eq. (H-88), which indicates that at each interior point the temperature at any point at the end of a time step is just the arithmetic average of the temperatures at the adjacent points at the beginning of that time step. The end temperatures are given by the boundary conditions. Because the temperatures are symmetrical on either side of the center line, we calculate only for \( x \leq 1.0 \). The temperature at \( x = 1.25 \) is the same as at \( x = 0.75 \). Table H-3 gives the results of the program.

(continued)
### TABLE H-3 | Potential Values in One Dimension by Explicit Method

For X = 0.0 to X = 10.0 with Delta X of 1.000

<table>
<thead>
<tr>
<th>X</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
<th>Value 4</th>
<th>Value 5</th>
<th>Value 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

(continued)
**Example H-4**—(continued)

**TABLE H-3**—(continued)

<table>
<thead>
<tr>
<th>VALUES AT T = 8.923</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000 5.368 10.965 17.083 24.097 32.435</td>
</tr>
<tr>
<td>42.486 54.476 68.358 83.756 100.000</td>
</tr>
</tbody>
</table>

VALUES AT T = 9.211
| 0.000 5.368 10.965 17.083 24.097 32.435  |
| 42.486 54.476 68.358 83.756 100.000        |

VALUES AT T = 9.498
| 0.000 5.486 11.231 17.534 24.757 33.281  |
| 43.438 55.401 69.097 84.168 100.000        |

VALUES AT T = 9.786
| 0.000 5.486 11.231 17.534 24.757 33.281  |
| 43.438 55.401 69.097 84.168 100.000        |

VALUES AT T = 10.074
| 0.000 5.618 11.514 17.995 25.405 34.088  |
| 44.326 56.250 69.769 84.539 100.000        |

VALUES AT T = 10.362
| 0.000 5.618 11.514 17.995 25.405 34.088  |
| 44.326 56.250 69.769 84.539 100.000        |

VALUES AT T = 10.650
| 0.000 5.759 11.809 18.459 26.037 34.856  |
| 45.155 57.031 70.381 84.876 100.000        |

VALUES AT T = 10.938
| 0.000 5.759 11.809 18.459 26.037 34.856  |
| 45.155 57.031 70.381 84.876 100.000        |

VALUES AT T = 11.225
| 0.000 5.906 12.110 18.922 26.653 35.587  |
| 45.931 57.754 70.942 85.184 100.000        |

VALUES AT T = 11.513
| 0.000 5.906 12.110 18.922 26.653 35.587  |
| 45.931 57.754 70.942 85.184 100.000        |

VALUES AT T = 11.801
| 0.000 6.056 12.414 19.380 27.250 36.283  |
| 46.659 58.424 71.458 85.465 100.000        |

VALUES AT T = 12.089
| 0.000 6.056 12.414 19.380 27.250 36.283  |
| 46.659 58.424 71.458 85.465 100.000        |

VALUES AT T = 12.377
| 0.000 6.207 12.717 19.830 27.827 36.946  |
| 47.343 59.048 71.935 85.724 100.000        |

VALUES AT T = 12.665
| 0.000 6.207 12.717 19.830 27.827 36.946  |
| 47.343 59.048 71.935 85.724 100.000        |

VALUES AT T = 12.952
| 0.000 6.358 13.017 20.269 28.383 37.577  |
| 47.987 59.629 72.378 85.963 100.000        |

VALUES AT T = 13.240
| 0.000 6.358 13.017 20.269 28.383 37.577  |
| 47.987 59.629 72.378 85.963 100.000        |

VALUES AT T = 13.528
| 0.000 6.508 13.312 20.697 28.918 38.178  |
| 48.594 60.174 72.789 86.185 100.000        |

VALUES AT T = 13.816
| 0.000 6.508 13.312 20.697 28.918 38.178  |
| 48.594 60.174 72.789 86.185 100.000        |

VALUES AT T = 14.104
| 0.000 6.655 13.601 21.112 29.432 38.749  |
| 49.167 60.684 73.173 86.391 100.000        |

(continued)
FURTHER READING
