derive the Newton-Raphson iterative method for two simultaneous equations.

3.10 The method of successive approximations given in Section 3.7 may be generalised to simultaneous equations as shown below:

$$x = f(x, y)$$
$$y = g(x, y).$$

The iterative method is

$$x_{i+1} = f(x_i, y_i)$$

and
$$y_{i+1} = g(x_i, y_i).$$

Show that the method will converge if

$$\left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial g}{\partial x}\right| < 1$$

and
$$\left|\frac{\partial f}{\partial y}\right| + \left|\frac{\partial g}{\partial y}\right| < 1$$

for all values of (x_i, y_i).

4 Solution of Simultaneous Algebraic Equations

4.1 Introduction

Simultaneous linear algebraic equations occur often in diverse fields of science and engineering and is an important area of study. In this chapter we will be mainly concerned with the solution of a set of n linear algebraic equations in n unknowns.

We are normally taught (in schools) a method of solving simultaneous equations, based on the evaluation of determinants, called Cramer's rule. For a small number of simultaneous equations (namely, 3 or 4) this rule is satisfactory. The number of multiplication operations needed when this method is used increases very rapidly as the number of equations increase. For example for 10 equations it is nearly 70 million arithmetic operations and with 50 equations it goes up to 10^{64} operations which is very large even for a fast computer. A different approach is thus needed for solving such equations on a computer.

In this chapter we will consider two techniques of solving simultaneous equations. These methods are particularly suited for computers. Typically these techniques would need about 1000 arithmetic operations for solving 10 simultaneous equations as compared to seventy million operations of Cramer's rule. The two methods are commonly known as the *direct method* and the *iterative method*. The direct method is based on the elimination of variables to transform the set of equations to a triangular form. The iterative method is a successive approximation procedure. Each one of these methods has its advantages and an understanding of the methods is needed to make a judicious choice when a set of equations is given.

4.2 The Gauss elimination method

To illustrate the method we shall consider three simultaneous equations in three unknowns:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = a_{14} \qquad \ldots(4.1)$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = a_{24} \qquad \ldots(4.2)$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = a_{34} \qquad \ldots(4.3)$$

The first step is to eliminate the first term from Equations (4.2) and (4.3). In order to do it Equation (4.1) is divided by a_{11} and multiplied by a_{21} and subtracted from (4.2). This eliminates x_1 from (4.2) as shown below:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = a_{14} \qquad \ldots(4.1)$$

$$\left(a_{21} - \frac{a_{11}}{a_{11}}\,a_{21}\right)x_1 + \left(a_{22} - \frac{a_{12}}{a_{11}}\,a_{21}\right)x_2 + \left(a_{23} - \frac{a_{13}}{a_{11}}\,a_{21}\right)x_3 = \left(a_{24} - \frac{a_{14}}{a_{11}}\,a_{21}\right) \qquad \ldots(4.4)$$

If we call $\dfrac{a_{21}}{a_{11}} = k_2$ Equation (4.4) may be rewritten as

$$(a_{21} - k_2a_{11})\,x_1 + (a_{22} - k_2a_{12})\,x_2 + (a_{23} - k_2a_{13})\,x_3 = (a_{24} - k_2a_{14}) \qquad \ldots(4.5)$$

Multiplying Equation (4.1) by $\left(\dfrac{a_{31}}{a_{11}}\right) = k_3$ and subtracting from (4.3) we get

$$(a_{31} - k_3a_{11})\,x_1 + (a_{32} - k_3a_{12})\,x_2 + (a_{33} - k_3a_{13})\,x_3 = (a_{34} - k_3a_{14}) \qquad \ldots(4.6)$$

Observe that $(a_{21} - k_2a_{11})$ and $(a_{31} - k_3a_{11})$ are both zero.

In the steps above we have assumed that a_{11} is not zero. We will consider the case when a_{11} is zero later so that too many issues are not confused at this time. The above elimination procedure is algorithmically expressed below.

Algorithm 4.1: Elimination of x_1 from 3 equations

```
1   For $i = 1$ to 3 and $j = 1$ to 4 in steps of 1 Read $a_{ij}$
```

```
2   For $i = 2$ to 3 in steps of 1 do
    begin
3       $u \leftarrow a_{i1}/a_{11}$
4       For $j = 1$ to 4 in steps of 1 do 5
5           $a_{ij} \leftarrow a_{ij} - u\,a_{1j}$
    end.
```

In the above algorithm instructions 2 and 4 both command that instructions up to 5 be repeated a certain number of times. The way it is executed is as follows:

Instruction 2 commands that all instructions up to and including 5 be repeated first with $i = 2$ and next with $i = 3$. Thus i is set equal to 2 and the next instruction taken up for execution. This instruction sets $u = a_{21}/a_{11}$. Instruction 4 commands that instruction 5 be done first with $j = 1$, then with $j = 2$, next with $j = 3$ and lastly with $j = 4$. Thus when instruction 5 is reached $i = 2$ and $j = 1$. After executing 5 with these values of i and j the inner loop (consisting of instruction 5) is executed again with $j = 2, 3, 4$. After the inner loop is executed for all values of j, the outer loop command (namely, instruction 2) is taken up for execution. Thus i is set equal to 3 and with this value the inner loop is again executed for all values of j, namely, 1, 2, 3 and 4.

This type of a structure of loops in an algorithm is called *nested* loops. In nested loops if the outer loop is to be repeated for $i = 1, n$ and the inner loop for $j = 1$ to m then the instructions inside the inner loop will be executed mn times and those in the outer loop n times. The sequence of values for (i, j) is shown below:

$$\begin{array}{ll} i & 1\;1\;1\;\ldots\;1\quad 2\;2\;2\;\ldots\;2\;\ldots\;n\;n\;n\;\ldots\;n \\ j & 1\;2\;3\;\ldots\;m\quad 1\;2\;3\;\ldots\;m\;\ldots\;1\;2\;3\;\ldots\;m \end{array}$$

The execution of Algorithm 4.1 is traced in the table below:

i	j	
2		$u \leftarrow a_{21}/a_{11}$
2	1	$a_{21} \leftarrow a_{21} - u\,a_{11}$ $\left(a_{21} \leftarrow a_{21} - \dfrac{a_{21}}{a_{11}}\,a_{11} = 0\right)$
2	2	$a_{22} \leftarrow a_{22} - u\,a_{12}$

```
2  3   $a_{23} \leftarrow a_{23} - u\,a_{13}$
2  4   $a_{24} \leftarrow a_{24} - u\,a_{14}$
3          $u \leftarrow a_{31}/a_{11}$
3  1   $a_{31} \leftarrow a_{31} - u\,a_{11}$
3  2   $a_{32} \leftarrow a_{32} - u\,a_{12}$
3  3   $a_{33} \leftarrow a_{33} - u\,a_{13}$
3  4   $a_{34} \leftarrow a_{34} - u\,a_{14}$
```

The reduced equations are:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = a_{14} \qquad \ldots(4.1)$$

$$a_{22}x_2 + a_{23}x_3 = a_{24} \qquad \ldots(4.7)$$

$$a_{32}x_2 + a_{33}x_3 = a_{34} \qquad \ldots(4.8)$$

The next step is to eliminate $a_{32}x_2$ from Equation (4.8). This is done by multiplying Equation (4.7) by $u = a_{32}/a_{22}$ and subtracting the resulting equation from (4.8). Algorithmically this is represented below.

Algorithm 4.2: For eliminating $\mathbf{x_2}$ from third equation

```
1  $i = 3$
2  $u \leftarrow a_{i2}/a_{22}$
3  For $j = 2$ to 4 in steps of 1 do 4
4  $a_{ij} \leftarrow a_{ij} - u\,a_{2j}$
```

Tracing the above algorithm we get the table;

```
i  j
3          $u \leftarrow a_{32}/a_{22}$
3  2   $a_{32} \leftarrow a_{32} - u\,a_{22} = a_{32} - \frac{a_{32}}{a_{22}}\,a_{22} = 0$
3  3   $a_{33} \leftarrow a_{33} - u\,a_{23}$
3  4   $a_{34} \leftarrow a_{34} - u\,a_{24}$
```

Thus at the end of this step the equations are

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = a_{14} \qquad \ldots(4.1)$$

$$a_{22}x_2 + a_{23}x_3 = a_{24} \qquad \ldots(4.7)$$

$$a_{33}x_3 = a_{34} \qquad \ldots(4.9)$$

The above set of equations are said to be in a *triangular* form.

Algorithm 4.3: Elimination of $\mathbf{x_1}$ extended to n unknowns

```
1  For $i = 1$ to $n$ in steps of 1 and $j = 1$ to $(n+1)$ in steps of
   1 do Read $a_{ij}$
2  For $i = 2$ to $n$ in steps of 1 do
   begin
3      $u \leftarrow a_{i1}/a_{11}$
4      For $j = 1$ to $(n+1)$ in steps of 1 do 5
5      $a_{ij} \leftarrow a_{ij} - u\,a_{1j}$
   end
```

Algorithm 4.4: Elimination of $\mathbf{x_n}$ extended to n unknowns

```
1  $i = n$
2  $u \leftarrow a_{i(n-1)}/a_{(n-1)(n-1)}$
3  For $j = (n-1)$ to $(n+1)$ in steps of 1 do 4
4  $a_{ij} \leftarrow a_{ij} - u\,a_{(n-1)j}$
```

In general while eliminating x_k the procedure would be:

Algorithm 4.5: Eliminating $\mathbf{x_k}$ from n equations

```
1  For $i = (k+1)$ to $n$ in steps of 1 do
   begin
2      $u \leftarrow a_{ik}/a_{kk}$
3      For $j = k$ to $(n+1)$ in steps of 1 do 4
4          $a_{ij} \leftarrow a_{ij} - u\,a_{kj}$
   end
```

Combining these ideas we may evolve the general algorithm given below to traingularise a set of n equations in n unknowns.

Algorithm 4.6: Triangularizing n equations in n unknowns

```
1  For $i = 1$ to $n$ in steps of 1 and $j = 1$ to $(n+1)$ in steps of 1
   Read $a_{ij}$
2  For $k = 1$ to $n-1$ in steps of 1 do
   begin
3      For $i = (k+1)$ to $n$ in steps of 1 do
       begin
4          $u \leftarrow a_{ik}/a_{kk}$
```

```
5          For $j = k$ to $(n+1)$ in steps of 1 do 6
6          $a_{ij} \leftarrow a_{ij} - u\, a_{kj}$
        end
    end
```

The student is urged to check the above algorithm for $n = 3$ by making a table.

From the triangular form obtained in Equations (4.1), (4.7) and (4.9) the values of x_3, x_2 and x_1 are obtained by *back substitution* as follows:

$$x_3 = a_{34}/a_{33} \quad \ldots(4.10)$$

$$x_2 = (a_{24} - a_{23}x_3)/a_{22} \quad \ldots(4.11)$$

$$x_1 = (a_{14} - a_{12}x_2 - a_{13}x_3)/a_{11} \quad \ldots(4.12)$$

Extending the above to n equations the back substitution may be expressed as the following equations:

$$x_n = a_{n\,(n+1)}/a_{nn}. \quad \ldots(4.13)$$

For $i = (n-1), (n-2), (n-3), \ldots 1$ we may write

$$x_i = (a_{in+1} - \sum_{j=(i+1)}^{n} a_{ij}\, x_j)/a_{ii} \quad \ldots(4.14)$$

The above equations may be expressed as algorithm 4.7 as given below.

Algorithm 4.7: Back substitution

```
1   $x_n \leftarrow a_{n(n+1)}/a_{nn}$
2   For $i = (n-1)$ to 1 in steps of $-1$ do
    begin
3   sum $\leftarrow 0$
4          For $j = (i+1)$ to $n$ in steps of 1 do 5
5          sum $\leftarrow$ sum $+ a_{ij}\, x_j$
6   $x_i \leftarrow (a_{i\,(n+1)} -$ sum$)/a_{ii}$
    end
```

4.3 Pivoting

In the triangularisation Algorithm 4.6 instruction 4 is

$$u \leftarrow a_{ik}/a_{kk}.$$

In this instruction it has been assumed that a_{kk} is not zero. If it happens to be zero or nearly zero the algorithm will lead to no results or meaningless results. An example of this type was considered in Chapter 2, Sec. 2.5 for two simultaneous equations. It was seen there that the two equations when interchanged gave a better solution. Similarly if any of the a_{kk} is small it would be necessary to reorder the equations. Observe that the value of a_{kk}s would be modified during the elimination process and there is no way of predicting their values at the start of the procedure.

Referring to Equations (4.1), (4.2) and (4.3) and Algorithm 4.1 it is seen that all elements except a_{11} in column 1 are made zero by that algorithm. The element a_{11} is called the *pivot* element. After a_{21} and a_{31} are eliminated the next step in triangularization is to eliminate all the terms below a_{22} in the second row (in this case a_{32} only). The pivot in this case is a_{22}.

In the elimination procedure the pivot should not be zero or a small number. In fact for maximum precision the pivot element should be the largest in absolute value of all the elements below it in its column. In other words a_{11} should be picked as the maximum of a_{11}, a_{21} and a_{31}. The element a_{22} should be greater than a_{32}. In general the pivot element a_{kk} should be the largest of

$$\{|a_{mk}|\} \text{ for } m = k+1, n$$

Thus during the Gauss elimination procedure $|a_{mk}|$s should be searched and the equation with the maximum value of $|a_{mk}|$ should be interchanged with the current equation. For example if during elimination we have the following situation:

$$x_1 + 2x_2 + 3x_3 = 4 \quad \ldots(4.15)$$

$$0.3x_2 + 4x_3 = 5 \quad \ldots(4.16)$$

$$-8x_2 + 3x_3 = 6 \quad \ldots(4.17)$$

then as $|-8| > 0.3$ equations (4.16) and (4.17) should be interchanged to yield

$$x_1 + 2x_2 + 3x_3 = 4 \quad \ldots(4.18)$$

$$-8x_2 + 3x_3 = 6 \quad \ldots(4.19)$$

$$0.3x_2 + 4x_3 = 5 \quad \ldots(4.20)$$

It should be remembered that interchange of equations does not affect the solution.

The algorithm for picking the largest element as the pivot and interchanging the equations is called *pivotal condensation*. This algorithm is to be inserted next to instruction 2 in Algorithm 4.6 for triangularization.

Algorithm 4.8: Pivotal condensation

Remarks: To be inserted between instructions 2 and 3 of Procedure 4.6

```
  2  For $k = 1$ to $(n - 1)$ in steps of 1 do
     begin
.....................................
201  max ← $|a_{kk}|$
202  $p \leftarrow k$
203  For $m = (k + 1)$ to $n$ in steps of 1 do
     begin
204       If ($|a_{mk}| >$ max) then
            begin
205           max ← $|a_{mk}|$
206           $p \leftarrow m$
          end
     end
207  If (max $< e$) then
       begin
208       Write 'Ill-conditioned equations'
209       Stop
       end
210    else If ($p = k$) then go to 3
211  For $q = k$ to $(n + 1)$ in steps of 1 do
     begin
212       temp ← $a_{kq}$
213         $a_{kq} \leftarrow a_{pq}$
214         $a_{pq} \leftarrow$ temp
     end
.....................................
  3  For $i = (k + 1)$ to $n$ in steps of 1 do
```

In the above algorithm an interesting new feature to be noted is the way of interchanging two numbers. In this procedure a_{kq} is to be interchanged with a_{pq}.

If one writes $a_{kq} \leftarrow a_{pq}$ then a_{pq} will *replace* a_{kq} and the original value in a_{kq} will be lost. For interchanging them a_{kq} is to be temporarily stored and then moved into a_{pq} as shown in Figure 4.1.

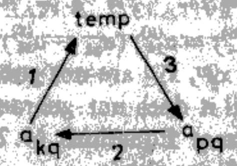

Fig. 4.1. Illustrating interchange of values.

The student is urged to trace through the above algorithm with a set of 4 simultaneous equations in 4 unknowns.

4.4 Illconditioned equations

If at any time during pivotal condensation it is found that all values of

$$\{|a_{mk}|\} \text{ for } m = k + 1 \text{ to } n$$

are less than a preassigned small quantity 'e' then the equations are illconditioned and no useful solution is obtained. Even when this does not happen the equations would be illconditioned if the determinant of the coefficients of the equation is small. (Remember Cramer's rule.) The value of the determinant is available as a byproduct after triangularisation as it is equal to the product $a_{11}\, a_{22}\, a_{33} \ldots a_{nn}$. Under these conditions some special techniques have to be used to get reasonably useful answers. One of these will be discussed in the next section.

4.5 Refinement of the solution obtained by Gaussian elimination

Even with row interchanges the solution will have some rounding error. We will discuss a method called *iterative refinement* which leads to reduced rounding errors and often a reasonable solution for some illconditioned problems is obtained.

Let $x_1^{(0)}, x_2^{(0)}, x_3^{(0)}$ be the approximate solution obtained from the Gaussian elimination procedure for Equations (4.1), (4.2) and

(4.3). If we substitute these solutions on the left hand side of these equations we obtain

$$a_{11}\,x_1^{(0)} + a_{12}\,x_2^{(0)} + a_{13}\,x_3^{(0)} = a_{14}^{(0)} \qquad \ldots(4.21)$$

$$a_{21}\,x_1^{(0)} + a_{22}\,x_2^{(0)} + a_{23}\,x_3^{(0)} = a_{24}^{(0)} \qquad \ldots(4.22)$$

$$a_{31}\,x_1^{(0)} + a_{32}\,x_2^{(0)} + a_{33}\,x_3^{(0)} = a_{34}^{(0)} \qquad \ldots(4.23)$$

If the differences $(a_{14} - a_{14}^{(0)}) = \alpha_{14}^{(0)}$, $(a_{24} - a_{24}^{(0)}) = \alpha_{24}^{(0)}$ and $(a_{34} - a_{34}^{(0)}) = \alpha_{34}^{(0)}$ are large then $x_1^{(0)}$, $x_2^{(0)}$, $x_3^{(0)}$ are not good solutions. (Even if these differences are small it does not guarantee that the solutions are correct.) If we subtract Equations (4.21), (4.22) and (4.23) from (4.1), (4.2) and (4.3) we obtain

$$a_{11}\,e_1^{(0)} + a_{12}\,e_2^{(0)} + a_{13}\,e_3^{(0)} = \alpha_{14}^{(0)} \qquad \ldots(4.24)$$

$$a_{21}\,e_1^{(0)} + a_{22}\,e_2^{(0)} + a_{23}\,e_3^{(0)} = \alpha_{24}^{(0)} \qquad \ldots(4.25)$$

$$a_{31}\,e_1^{(0)} + a_{32}\,e_2^{(0)} + a_{33}\,e_3^{(0)} = \alpha_{34}^{(0)} \qquad \ldots(4.26)$$

where $e_1^{(0)} = x_1 - x_1^{(0)}$, $e_2^{(0)} = x_2 - x_2^{(0)}$ and $e_3^{(0)} = x_3 - x_3^{(0)}$.

Equations (4.24), (4.25), (4.26) may be solved by Gaussian elimination and values of e_i's obtained. A new approximation to the solutions is then

$$x_i^{(1)} = x_i^{(0)} + e_i^{(0)}$$

This procedure may be repeated by substituting $x_i^{(1)}$'s in (4.1), (4.2), (4.3) and further refinements to solution obtained. It has been found that a few cycles of this refinement procedure lead to more accurate solutions even in illconditioned systems.

4.6 The Gauss-Seidel iterative method

The Gauss-Seidel method will be first illustrated with two simultaneous equations. We will later generalise it to n equations in n unknowns.

Consider the simultaneous equations given below:

$$x_1 + x_2 = 2 \qquad \ldots(4.27)$$

$$3x_1 - 10x_2 = 3 \qquad \ldots(4.28)$$

Start with an initial value of $x_2^0 = 0$. Substitute this in Equation

(4.27) and obtain the value of x_1^0 as 2. Use this guessed value in Equation (4.28) to obtain a refined guess for x_2. This is given by $x_2^1 = (3x_1^0 - 3)/10 = (6 - 3)/10 = .3$. Use this in Equation (4.27) to get another value of $x_1^{(2)}$. The progress of the iteration is shown in Table 4.1.

TABLE 4.1

ILLUSTRATING ITERATIVE METHOD

Iteration Number	x_1	x_2
1	2	.3
2	1.7	.21
3	1.79	.237
4	1.763	.229
5	1.771	.231
6	1.769	.231
7	1.769	.231

Initial guess $x_2^0 = 0$.

Observe that the values of x_1 and x_2 converge to 1.769 and .231 respectively which is the solution. As this is an iterative method iterations are stopped when successive values of x_1 as well as of x_2 are 'close enough'.

This is a successive approximation method and the technique is illustrated graphically in Figure 4.2. The student is urged to compare this with the iterative technique presented in Chapter 3, Section 3.7. The convergence of the iterative method for two simultaneous equations will be investigated now.

Consider the equations

$$a_{11}x_1 + a_{12}x_2 = a_{13} \qquad \ldots(4.29)$$

$$a_{21}x_1 + a_{22}x_2 = a_{23} \qquad \ldots(4.30)$$

The iterative method to proceed from the kth to the $(k+1)$th iteration is shown below:

$$a_{11}x_1^{k+1} = a_{13} - a_{12}x_2^k \qquad \ldots(4.31)$$

$$a_{22}x_2^{k+1} = a_{23} - a_{21}x_1^{k+1} \qquad \ldots(4.32)$$

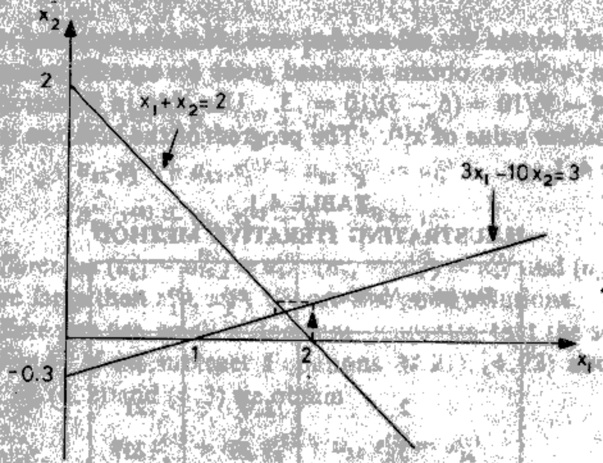

Fig. 4.2. Illustrating the Gauss-Seidel method.

Eliminating $x_1^{(k+1)}$ between Equations (4.31) and (4.32) we get

$$\frac{a_{11}}{a_{21}}\left[-a_{22}x_2^{k+1}+a_{23}\right]=a_{13}-a_{12}x_2^k \qquad \ldots(4.33)$$

Similarly for the $(k+1)$th to $(k+2)$th iteration we get

$$\frac{a_{11}}{a_{21}}\left[-a_{22}x_2^{k+2}+a_{23}\right]=a_{13}-a_{12}x_2^{k+1} \qquad \ldots(4.34)$$

Subtracting (4.33) from (4.34) and calling $(x_2^{k+1}-x_2^k)=e_2^k$ and $(x_2^{k+2}-x_2^{k+1})=e_2^{k+1}$ we get

$$e_2^{k+1}=\frac{a_{12}a_{21}}{a_{11}a_{22}}e_2^k \qquad \ldots(4.35)$$

Thus if the difference between successive iterations is to decrease we must satisfy the condition

$$\left|\frac{a_{12}a_{21}}{a_{11}a_{22}}\right|<1 \qquad \ldots(4.36)$$

If we eliminate x_2 and work with x_1 and call the difference between successive iterations in x_1, namely

$$(x_1^{k+1}-x_1^k)=e_1^k$$

then we obtain the equation

$$e_1^{k+1}=\frac{a_{12}a_{21}}{a_{11}a_{22}}\,e_1^k \qquad \ldots(4.37)$$

Again for the convergence of the iterative procedure we should ensure that

$$\left|\frac{a_{12}a_{21}}{a_{11}a_{22}}\right|<1 \qquad \ldots(4.36)$$

Convergence will thus be ensured if the diagonal terms are larger than the off diagonal terms. If $|a_{11}|>|a_{12}|$ and $|a_{22}|\geqslant a_{21}$ or if $|a_{11}|\geqslant a_{12}$ and $|a_{22}|>a_{21}$ the above condition (Equation 4.36) would be satisfied.

For Equations (4.27) and (4.28) we have

$$\left|\frac{a_{12}a_{21}}{a_{11}a_{22}}\right|=\left|\frac{3}{-10}\right|=0.3$$

and the procedure should converge. If the same equations are interchanged we get

$$3x_1-10x_2=3 \qquad \ldots(4.28)$$
$$x_1+x_2=2 \qquad \ldots(4.27)$$

For these equations

$$\left|\frac{a_{12}a_{21}}{a_{11}a_{22}}\right|=\left|\frac{-10\times 1}{3}\right|=3.33>1$$

Thus the same iterative procedure would diverge!

The student is urged to draw the straight lines corresponding to these equations and satisfy himself that the procedure diverges and answer why it diverges.

4.7 An algorithm to implement the Gauss-Seidel method

Consider the set of n equations in n unknowns

$$a_{11}x_1+a_{12}x_2+\ldots a_{1n}x_n=a_{1n+1} \qquad \ldots(4.38)$$
$$a_{21}x_1+a_{22}x_2+\ldots a_{2n}x_n=a_{2n+1} \qquad \ldots(4.39)$$
$$a_{31}x_1+a_{32}x_2+\ldots a_{3n}x_n=a_{3n+1} \qquad \ldots(4.40)$$
$$\vdots \qquad \vdots \qquad \vdots$$
$$a_{n1}x_1+a_{n2}x_2+\ldots a_{nn}x_n=a_{nn+1} \qquad \ldots(4.41)$$

In the Gauss-Seidel method $x_2^0\ldots x_n^0$ are set equal to zero and x_1^1 is calculated from (4.38). This value of x_1 and zeros are substituted for $x_3, x_4, \ldots, x_n$ in (4.39) and x_2 computed from Equation (4.39). The values of x_1^1 and x_2^1 along with zeros for $x_4, x_5, \ldots x_n$ are used in (4.40) to compute x_3^1. Finally $x_1^1, x_2^1, \ldots, x_{n-1}^1$ are used in (4.41) to compute x_n^1. With this the first iteration ends. In the second iteration $x_2^1, x_3^1 \ldots x_n^1$ are used to compute x_1^2. $x_1^2, x_3^1 \ldots x_n^1$ are used to get x_2^2 and so on. The main point to observe is that always the latest approximations for the values of variables are used in an iteration.

The complete algorithm to implement the Gauss-Seidel elimination is given as Algorithm 4.9.

Algorithm 4.9: Gauss-Seidel iterative method

```
1   For $i = 1$ to $n$ in steps of 1 and $j = 1$ to $n + 1$
    in steps of 1 Read $a_{ij}$
2   Read $e$, maxit
    Remarks: $e$ is the allowed relative error in the result vector.
             maxit is the maximum number of iterations
             allowed for the solution to converge.
3   For $i = 1$ to $n$ in steps of 1 do $x_i \leftarrow 0$
    Remarks: All $x_i$s are initialised to zero. The following
             For loop sets the limit maxit for the maximum
             number of iterations to be allowed.
4   For iter = 1 to maxit do
    begin
5   big $\leftarrow 0$
6         For $i = 1$ to $n$ in steps of 1 do
          begin
7             sum $\leftarrow 0$
8             For $j = 1$ to $n$ in steps of 1 do 9
9               if $(j \neq i)$ then sum $\leftarrow$ sum $+ a_{ij}x_j$
10            temp $\leftarrow (a_{in+1} -$ sum$)/a_{ii}$
11            relerror $\leftarrow |(x_i -$ temp$)/$temp$|$
12            if relerror > big then big $\leftarrow$ relerror
13            $x_i \leftarrow$ temp
          end
14  if (big $\leqslant e$) then
          begin Write 'Converges to a solution'
                For $i = 1$ to $n$ in steps of 1 Write $x_i$
                Stop end
    end
15  Write 'Does not converge in maxit iterations'
16  For $i = 1$ to $n$ in steps of 1  Write $x_i$
17  Stop
```

The main part of Algorithm 4.9 starts with statement 5. The variable big stores the largest magnitude of the relative error in the variables. It is initialised to 0 in statement 5. The loop consisting of statements 8 and 9 calculates for a given $i = k$

$$\sum_{\substack{j=1\\ j\neq k}}^{n} a_{kj}x_j$$

In statement 10 the latest value of x_k is calculated and temporarily stored giving

$$\text{temp} = \text{latest value of } x_k = (a_{k(n+1)} - \sum_{\substack{j=1\\ j\neq k}}^{n} a_{kj}x_j)/a_{kk}$$

Statement 11 calculates the relative error in x_i (for $i = k$) between two latest iterates. Statement 12 picks the biggest relative error in all x_is. In statement 13 x_i is set to the latest calculated value of the variable. Loop between 6 and 13 calculates the latest trial value of x_i for all i's. On coming out of the loop 'big' has the highest relative error

$$\underset{i}{\text{Max}} \left| x_i^{iter} - x_i^{iter-1} \right|$$

When big is smaller than the assigned error then the solution converges. Observe that if the relative error does not fall below the assigned error limit for a total number of iterations equal to maxit then the procedure stops after giving an appropriate message. Such a 'safety exit' is essential in all iterative methods. Typically the maximum number of iterations set would be 'reasonable', say, 25 iterations for a (20×20) set of simultaneous equations.

Observe that in the inner loop when a value of x_i for (say) $i = k$

is calculated the latest calculated values of all x_is for $i = 1$ to $(k-1)$ is used.

The analysis given to test the convergence of two simultaneous equations may be extended to n simultaneous equations. The condition is

$$|a_{ii}| \geqslant \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \quad \text{for all } i$$

and
$$|a_{ii}| > \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \quad \text{for at least one } i$$

These conditions are normally met when the diagonal elements dominate. Such cases occur particularly when simultaneous equations arise as a result of discretizing partial differential equations. In such applications a number of off diagonal elements are zero and sometimes the zeros form a regular pattern. When such a pattern is perceived the Gauss-Seidel algorithm may be modified to take advantage of the pattern and reduce the number of arithmetic operations.

4.8 Comparison of direct and iterative methods

Both direct and iterative methods have their strengths and weaknesses and a choice is based on the particular set of equations to be solved.

Gauss elimination will lead to a solution in a finite number of steps for *any* set of equations provided the determinant of coefficients is not very small. The disadvantages are:

(i) The computational effort is approximately $(2n^3/3)$ arithmetic operations in each iteration.

(ii) The amount of book keeping necessary (such as row interchanges) is considerable.

(iii) The rounding errors may become quite large particularly for illconditioned equations.

(iv) Any special structure in the matrix of coefficients is difficult to preserve during elimination. Thus savings cannot be made in calculations if there are many zero coefficients.

In contrast to elimination method iterative methods may not

always converge. In fact convergence may be guaranteed only under special conditions. When it converges, however, the iterative method is superior to the elimination method due to the following advantages:

(i) Computational effort is approximately $2n^2$ arithmetic operations per iteration and if convergence is achieved in less than n iterations it is significantly superior compared to the elimination method. Further, special pattern of zeros in the coefficient matrix could be used to tailor a procedure with reduced calculation effort.

(ii) Another important advantage of the iterative method is the small rounding error, the rounding error being the one committed in the last iteration. Thus for illconditioned system an iterative method is a good choice.

(iii) In Gauss-Seidel method we set x_i equal to the newly calculated value of the iterate stored in temp. (Algorithm 4.9.) Instead if we set

$$x_i \leftarrow x_i + w\,(\text{temp} - x_i)$$

then observe that $x_i \leftarrow$ temp if $w = 1$ and $x_i \leftarrow x_i$ when $w = 0$. It has been found in many problems that if w is given some intermediate value between 0 and 1, convergence of Gauss-Seidel method is faster. Unfortunately there is no way to find the best value of w. It is normally found by trial and error. With such an acceleration iteration method would be quite attractive.

References

1. Householder, A.S., *Principles of Numerical Analysis*, McGraw-Hill, 1953.
2. *Modern Computing Methods*, Philosophical Library, New York, 1961.
3. Southworth, R.W., and Delleuw, S.L., *Digital Computation and Numerical Methods*, McGraw-Hill, 1965.
4. Varga, R., *Matrix Iterative Analysis*, Prentice-Hall, Inc., 1962.
5. Dorn, W.S., and McCracken, D.D., *Numerical Methods with Fortran IV Case Studies*, John Wiley and Sons, Inc., New York, 1972.

Exercises

4.1 Solve the following set of equations by Gauss elimination:

(i)
$$\begin{aligned} x_1 + x_2 + x_3 &= 3 \\ 2x_1 + 3x_2 + x_3 &= 6 \\ x_1 - x_2 - x_3 &= -3 \end{aligned}$$

(ii)
$$\begin{aligned} 2x_1 + 4x_2 + 2x_3 &= 15 \\ 2x_1 + x_2 + 2x_3 &= -5 \\ 4x_1 + x_2 - 2x_3 &= 0. \end{aligned}$$

Is row interchange necessary for the above equations?

4.2 Write a computational procedure to evaluate the following determinants. Check your procedure by hand calculation.

(i) $\begin{vmatrix} 2 & 3 & 2 \\ 4 & -1 & 2 \\ 4 & 1 & -2 \end{vmatrix}$ (ii) $\begin{vmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 1 & -1 & -1 \end{vmatrix}$

(iii) $\begin{vmatrix} 1 & 7 & 2 & 0 \\ 4 & -6 & 0 & -1 \\ 3 & -2 & -2 & 1 \\ 0 & 2 & 9 & 4 \end{vmatrix}$

4.3 Solve the following simultaneous equations with complex coefficients. Write a computer procedure to solve n equations in n unknowns with complex coefficients by extending Gauss elimination.

$$\begin{aligned} (1 + 2i)\,x_1 + (1 - 2i)\,x_2 &= 3 + i \\ (4 + 5i)\,x_1 + (2 - 4i)\,x_2 &= -2 - 4i. \end{aligned}$$

4.4 In the Gauss elimination procedure the equations are triangularised by

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= a_{14} \\ a_{22}x_2 + a_{23}x_3 &= a_{24} \\ + a_{33}x_3 &= a_{34} \end{aligned}$$

eliminating the coefficients of variables below the pivot as shown above. It would be possible to eliminate the co-

efficients both above and below the pivot in a given column as shown below.

$$\begin{aligned} a_{11}x_1 + \qquad &= a_{14} \\ a_{22}x_2 \qquad &= a_{24} \\ + a_{33}x_3 &= a_{34} \end{aligned}$$

By this method the equations are *diagonalized* and the back-substitution step is eliminated. This procedure is called Gauss-Jordan elimination. Write a computational algorithm for the general case of n equations in n-unknowns.

4.5 Solve the following simultaneous equation by Gauss-Jordan elimination procedure explained in Problem 4.4.

$$\begin{aligned} 2x_1 + 6x_2 - x_3 &= -14 \\ 5x_1 - x_2 + 2x_3 &= 29 \\ -3x_1 - 4x_2 + x_3 &= 4. \end{aligned}$$

4.6 Modify Algorithms 4.6 and 4.7 for Gauss elimination and incorporate iterative refinement. There must be provision to refine the solution 3 times.

4.7 Solve the two simultaneous equations of Example 2.22 of Section 2.5 without row interchange by iterative refinement.

4.8 Repeat Problem 4.1 using Gauss-Seidel iterative method. Get your answer to 4 significant digits. Compare your answer with that obtained using the elimination method.

4.9 Solve the following equations by Gauss-Seidel procedure. The answer should be correct to 3 significant digits.

$$\begin{aligned} 9x_1 + 2x_2 + 4x_3 &= 20 \\ x_1 + 10x_2 + 4x_3 &= 6 \\ 2x_1 - 4x_2 + 10x_3 &= -15. \end{aligned}$$

4.10 Apply the accelerated convergence technique on the example of Section 4.6 with various *w*s. Comment.

4.11 Solve the following simultaneous equations

$$\begin{aligned} 2.5x_1 + 5.2\,x_2 &= 6.2 \\ 1.251x_1 + 2.605x_2 &= 3.152 \end{aligned}$$

by Gauss elimination using floating point arithmetic and get your answer to 4 significant digits. Improve the solution by iterative refinement.

4.12 Repeat Problem 4.11 using Gauss-Seidel method. Compare the answers obtained.

4.13 Modify Algorithm 4.9 to include the idea of accelerated convergence of the Gauss-Seidel method discussed in the last section.

4.14 Consider the following sparse set of equations

$$\begin{aligned} 2x_1 - 2x_2 \qquad\qquad &= 1 \\ -x_1 + 2x_2 - 3x_3 \qquad &= -2 \\ -2x_2 + 2x_3 - x_4 &= -1. \end{aligned}$$

(i) Are the zero coefficients preserved as zeros during Gauss elimination?

(ii) If yes, how would you modify the Gauss elimination algorithm to reduce the number of arithmetic operations?

4.15 Modify Gauss-Seidel algorithm to solve tridiagonal linear systems of equations so that unnecessary arithmetic operations are not performed.

$$\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ 0 \ + \ 0 \ + \ \ldots \ = b_1 \\ a_{12}x_1 + a_{22}x_2 + a_{23}x_3 + \ 0 \ + \ 0 \ = b_2 \\ 0 \ + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 \ + \ \ldots \ = b_3 \\ 0 \quad 0 \quad a_{43}x_3 + a_{44}x_4 + a_{45}x_5 + \ldots = b_4 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ 0 \quad 0 \quad 0 \quad a_{n-1n}x_{n-1} + a_{nn}x_n \ = b_n. \end{array}$$

5 Interpolation

5.1 Introduction

Quite often a function may be given as a table of values instead of as a closed form expression. Examples are tabulated functions such as Bessel functions. In these cases the value of the function $f(x)$ is tabulated at a discrete set of values of the argument x as shown in Table 5.1.

TABLE 5.1

x	x_1	x_2	x_3	$\ldots$	x_n
$f(x)$	$f(x_1)$	$f(x_2)$	$f(x_3)$	$\ldots$	$f(x_n)$

If the value of $f(x)$ is to be found at some point y in the interval $[x_1, x_n]$ and y is not one of the tabulated points then the value is estimated by using the known values of $f(x)$ at the surrounding points. This is called *interpolation*. In this chapter we will be mainly concerned with interpolation. If the point y is *outside* the interval $[x_1, x_n]$ then the estimation of $f(y)$ is called *extrapolation*. Extrapolation assumes that the behaviour of $f(x)$ outside the range $[x_1, x_n]$ is identical to that inside the range and this may not always be valid.

Historically interpolation has been the foundation of classical numerical analysis. The primary reason for this is the ease of using tables of values in hand computations. For instance it is very convenient to look up the values of a function like Sine in a