2

The Numerical Methods for Linear Equations and Matrices

•   •   •

We saw in the previous chapter that linear equations play an important role in transformation theory and that these equations could be simply expressed in terms of matrices. However, this is only a small segment of the importance of linear equations and matrix theory to the mathematical description of the physical world. Thus we should begin our study of numerical methods with a description of methods for manipulating matrices and solving systems of linear equations. However, before we begin any discussion of numerical methods, we must say something about the accuracy to which those calculations can be made.

2.1 Errors and Their Propagation

One of the most reliable aspects of numerical analysis programs for the electronic digital computer is that they almost always produce numbers. As a result of the considerable reliability of the machines, it is common to regard the results of their calculations with a certain air of infallibility. However, the results can be no better than the method of analysis and implementation program utilized by the computer and these are the works of highly fallible man. This is the origin of the aphorism "*garbage in – garbage out*". Because of the large number of calculations carried out by these machines, small errors at any given stage can rapidly propagate into large ones that destroy the validity of the result.

We can divide computational errors into two general categories: the first of these we will call round off error, and the second truncation error. Round off error is perhaps the more insidious of the two and is always present at some level. Indeed, its omnipresence indicates the first problem facing us. How accurate an answer do we require? Digital computers utilize a certain number of digits in their calculations and this base number of digits is known as the precision of the machine. Often it is possible to double or triple the number of digits and hence the phrase "double" or "triple" precision is commonly used to describe a calculation carried out using this expanded number of digits. It is a common practice to use more digits than are justified by the problem simply to be sure that one has "got it right". For the scientist, there is a subtle danger in this in that the temptation to publish all the digits presented by the computer is usually overwhelming. Thus published articles often contain numerical results consisting of many more decimal places than are justified by the calculation or the physics that went into the problem. This can lead to some reader unknowingly using the results at an unjustified level of precession thereby obtaining meaningless conclusions. Certainly the full machine precession is never justified, as after the first arithmetical calculation, there will usually be some uncertainty in the value of the last digit. This is the result of the first kind of error we called *round off error*. As an extreme example, consider a machine that keeps only one significant figure and the exponent of the calculation so that 6+3 will yield 9×10^0. However, 6+4, 6+5, and 6+8 will all yield the same answer namely 1×10^1. Since the machine only carries one digit, all the other information will be lost. It is not immediately obvious what the result of 6+9, or 7+9 will be. If the result is 2×10^1, then the machine is said to round off the calculation to the nearest significant digit. However, if the result remains 1×10^1, then the machine is said to truncate the addition to the nearest significant digit. Which is actually done by the computer will depend on both the physical architecture (hardware) of the machine and the programs (software) which instruct it to carry out the operation. Should a human operator be carrying out the calculation, it would usually be possible to see when this is happening and allow for it by keeping additional significant figures, but this is generally not the case with machines. Therefore, we must be careful about the propagation of round off error into the final computational result. It is tempting to say that the above example is only for a 1-digit machine and therefore unrealistic. However, consider the common 6-digit machine. It will be unable to distinguish between 1 million dollars and 1 million and nine dollars. Subtraction of those two numbers would yield zero. This would be significant to any accountant at a bank. Repeated operations of this sort can lead to a completely meaningless result in the first digit.

This emphasizes the question of 'how accurate an answer do we need?'. For the accountant, we clearly need enough digits to account for all the money at a level decided by the bank. For example, the Internal Revenue Service allows taxpayers to round all calculations to the nearest dollar. This sets a lower bound for the number of significant digits. One's income usually sets the upper bound. In the physical world very few constants of nature are known to more than four digits (the speed of light is a notable exception). Thus the results of physical modeling are rarely important beyond four figures. Again there are exceptions such as in null experiments, but in general, scientists should not deceive themselves into believing their answers are better answers than they are.

How do we detect the effects of round off error? Entire studies have been devoted to this subject by considering that round off errors occurs in basically a random fashion. Although computers are basically deterministic (i.e. given the same initial state, the computer will always arrive at the same answer), a large collection of arithmetic operations can be considered as producing a random collection of round-ups and round-downs. However, the number of digits that are affected will also be variable, and this makes the problem far more difficult to study in general. Thus in practice, when the effects of round off error are of great concern, the problem can be run in double precession. Should both calculations yield the same result at the acceptable level of precession, then round off error is probably not a problem. An additional "rule of thumb" for detecting the presence of round off error is the appearance of a large number of zeros at the right-hand side of the answers. Should the number of zeros depend on parameters of the problem that determine the size or numerical extent of the problem, then one should be concerned about round off error. Certainly one can think of exceptions to this rule, but in general, they are just that - exceptions.

The second form of error we called *truncation error* and it should not be confused with errors introduced by the "truncation" process that happens half the time in the case of round off errors. This type of error results from the inability of the approximation method to properly represent the solution to the problem. The magnitude of this kind of error depends on both the nature of the problem and the type of approximation technique. For example, consider a numerical approximation technique that will give exact answers should the solution to the problem of interest be a polynomial (we shall show in chapter 3 that the majority of methods of numerical analysis are indeed of this form). Since the solution is exact for polynomials, the extent that the correct solution differs from a polynomial will yield an error. However, there are many different kinds of polynomials and it may be that a polynomial of higher degree approximates the solution more accurately than one of lower degree.

This provides a hint for the practical evaluation of truncation errors. If the calculation is repeated at different levels of approximation (i.e. for approximation methods that are correct for different degree polynomials) and the answers change by an unacceptable amount, then it is likely that the truncation error is larger than the acceptable amount. There are formal ways of estimating the truncation error and some 'black-box' programs do this. Indeed, there are general programs for finding the solutions to differential equations that use estimates of the truncation error to adjust parameters of the solution process to optimize efficiency. However, one should remember that these estimates are just that - estimates subject to all the errors of calculation we have been discussing. It many cases the correct calculation of the truncation error is a more formidable problem than the one of interest. In general, it is useful for the analyst to have some prior knowledge of the behavior of the solution to the problem of interest before attempting a detailed numerical solution. Such knowledge will generally provide a 'feeling' for the form of the truncation error and the extent to which a particular numerical technique will manage it.

We must keep in mind that both round-off and truncation errors will be present at some level in any calculation and be wary lest they destroy the accuracy of the solution. The acceptable level of accuracy is determined by the analyst and he must be careful not to aim too high and carry out grossly inefficient calculations, or too low and obtain meaningless results.

We now turn to the solution of linear algebraic equations and problems involving matrices associated with those solutions. In general we can divide the approaches to the solution of linear algebraic equations into two broad areas. The first of these involve algorithms that lead directly to a solution of the problem after a finite number of steps while the second class involves an initial "guess" which then is improved by a succession of finite steps, each set of which we will call an iteration. If the process is applicable and properly formulated, a finite number of iterations will lead to a solution.

2.2 Direct Methods for the Solution of Linear Algebraic Equations

In general, we may write a system of linear algebraic equations in the form

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = c_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = c_2 \\ a_{31}x_1 + a_{312}x_2 + a_{33}x_3 + \cdots + a_{3n}x_n = c_3 \\ \quad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \\ \quad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \\ \quad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \qquad \cdot \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n = c_n \end{array}\right\} , \qquad (2.2.1)$$

which in vector notation is

$$\mathbf{A}\vec{x} = \vec{c} \ . \qquad (2.2.2)$$

Here x is an n-dimensional vector the elements of which represent the solution of the equations. $\vec{c}$ is the constant vector of the system of equations and **A** is the matrix of the system's coefficients.

We can write the solution to these equations as

$$\vec{x} = \mathbf{A}^{-1}\vec{c} \ , \qquad (2.2.3)$$

thereby reducing the solution of any algebraic system of linear equations to finding the inverse of the coefficient matrix. We shall spend some time describing a number of methods for doing just that. However, there are a number of methods that enable one to find the solution without finding the inverse of the matrix. Probably the best known of these is *Cramer's Rule*

a. Solution by Cramer's Rule

It is unfortunate that usually the only method for the solution of linear equations that students remember from secondary education is Cramer's rule or expansion by minors. As we shall see, this method is rather inefficient and relatively difficult to program for a computer. However, as it forms sort of a standard by which other methods can by judged, we will review it here. In Chapter 1 [equation (1.2.10)] we gave the form for the determinant of a 3×3 matrix. The more general definition is inductive so that the determinant of the matrix **A** would be given by

$$\text{Det}\,\mathbf{A} = \sum_{i=1}^{n}(-1)^{i+j}a_{ij}M_{ij}\ ,\forall j\ . \tag{2.2.4}$$

Here the summation may be taken over either i or j, or indeed, any monotonically increasing sequence of both. The quantity M_{ij} is the determinant of the matrix **A** with the ith row and jth column removed and, with the sign carried by $(-1)^{(i+j)}$ is called the *cofactor* of the *minor* element a_{ij}. With all this terminology, we can simply write the determinant as

$$\text{Det}\,\mathbf{A} = \sum_{i-1}^{n} C_{ij}a_{ij}\ \ ,\ \ \forall j\ ,\ = \sum_{j=1}^{n} a_{ij}C_{ij}\ \ ,\forall i\ \ . \tag{2.25}$$

By making use of theorems 2 and 7 in section 1.2, we can write the solution in terms of the determinant of **A** as

$$x_1\begin{vmatrix} a_{11} & a_{12} \cdots & a_{1n} \\ a_{21} & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & a_{nn} \end{vmatrix} = \begin{vmatrix} a_{11}x_1 & a_{12} \cdots & a_{1n} \\ a_{21}x_1 & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{n1}x_1 & a_{n2} & a_{nn} \end{vmatrix} = \begin{vmatrix} (a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n) & a_{12} \cdots & a_{1n} \\ (a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n) & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ (a_{1n}x_1 + a_{1n}x_2 + \cdots + a_{1n}x_n) & a_{n2} & a_{nn} \end{vmatrix}$$

$$= \begin{vmatrix} c_1 & a_{12} \cdots & a_{1n} \\ c_2 & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ c_n & a_{n2} & a_{nn} \end{vmatrix}\ , \tag{2.2.6}$$

which means that the general solution of equation (2.2.1) is given by

$$x_j = \begin{vmatrix} a_{11} \cdots a_{1\,j-1}c_1 & a_{1\,j+1} \cdots a_{1n} \\ a_{21} \cdots a_{2\,j-1}c_2 & a_{2\,j+1} \cdots a_{2n} \\ \cdot \quad\quad \cdot & \cdot \quad\quad \cdot \\ \cdot \quad\quad \cdot & \cdot \quad\quad \cdot \\ a_{n1} \cdots a_{n\,j-1}c_n & a_{n\,j+1} \cdots a_{nn} \end{vmatrix} \times\ [\text{Det}\,\mathbf{A}]^{-1}\ \ . \tag{2.2.7}$$

This requires evaluating the determinant of the matrix **A** as well as an augmented matrix where the jth column has been replaced by the elements of the constant vector c_i. Evaluation of the determinant of an n×n matrix requires about $3n^2$ operations and this must be repeated for each unknown, thus solution by Cramer's rule will require at least $3n^3$ operations. In addition, to maintain accuracy, an optimum path through the matrix (finding the least numerically sensitive cofactors) will require a significant amount of logic. Thus, solution by Cramer's rule is not a particularly desirable approach to the numerical solution of linear equations either for a computer or a hand calculation. Let us consider a simpler algorithm, which forms the basis for one of the most reliable and stable direct methods for the solution of linear equations. It also provides a method for the inversion of matrices. Let begin by describing the method and then trying to understand why it works.

b. Solution by Gaussian Elimination

Consider representing the set of linear equations given in equation (2.2.1) by

$$\begin{pmatrix} a_{11} & a_{12} \cdots & a_{1n} \\ a_{21} & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & a_{nn} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_n \end{pmatrix} . \tag{2.2.8}$$

Here we have suppressed the presence of the elements of the solution vector x_j. Now we will perform a series of operations on the rows and columns of the coefficient matrix **A** and we shall carry through the row operations to include the elements of the constant vector c_i. In other words, we shall treat the rows as if they were indeed the equations so that anything done to one element is done to all. One begins by dividing each row including the constant element by the lead element in the row. The first row is then subtracted from all the lower rows. Thus all rows but the first will have zero in the first column. Now repeat these operations for all but the first equation starting with the second element of the second equation producing ones in the second column of the remaining equations. Subtracting the resulting second line from all below will yield zeros in the first two columns of equation three and below. This process can be repeated until one has arrived at the last line representing the last equation. When the diagonal coefficient there is unity, the last term of the constant vector contains the value of x_n. This can be used in the (n-1)th equation represented by the second to the last line to obtain x_{n-1} and so on right up to the first line which will yield the value of x_1. The name of this method simply derives from the elimination of each unknown from the equations below it producing a triangular system of equations represented by

$$\begin{pmatrix} 1 & a'_{12} & \cdots & a'_{1n} \\ 0 & 1 & \cdots & a'_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \cdot\ \cdot\ \cdot & 1 \end{pmatrix} \begin{pmatrix} c'_1 \\ c'_2 \\ \cdot \\ \cdot \\ c'_n \end{pmatrix} , \tag{2.2.9}$$

which can then be easily solved by back substitution where

$$\left. \begin{aligned} x_n &= c'_n \\ x_i &= c'_i - \sum_{j=i+1}^{n} a'_{ij}\, x_j \end{aligned} \right\} . \tag{2.2.10}$$

One of the disadvantages of this approach is that errors (principally round off errors) from the successive subtractions build up through the process and accumulate in the last equation for x_n. The errors thus incurred are further magnified by the process of back substitution forcing the maximum effects of the round-off error into x_1. A simple modification to this process allows us to more evenly distribute the effects of round off error yielding a solution of more uniform accuracy. In addition, it will provide us with an efficient mechanism for calculation of the inverse of the matrix **A**.

c. Solution by Gauss Jordan Elimination

Let us begin by writing the system of linear equations as we did in equation (2.2.8), but now include a unit matrix with elements δ_{ij} on the right hand side of the expression. Thus,

$$\begin{pmatrix} a_{11} & a_{12} \cdots & a_{1n} \\ a_{21} & a_{22} \cdots & a_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & a_{nn} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_n \end{pmatrix} \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \cdots & 1 \end{pmatrix} . \tag{2.2.11}$$

We will treat the elements of this matrix as we do the elements of the constant vector c_i. Now proceed as we did with the Gauss elimination method producing zeros in the columns below and to the left of the diagonal element. However, in addition to subtracting the line whose diagonal element has been made unity from all those below it, also subtract from the equations above it as well. This will require that these equations be normalized so that the corresponding elements are made equal to one and the diagonal element will no longer be unity. In addition to operating on the rows of the matrix **A** and the elements of $\vec{C}$, we will operate on the elements of the additional matrix which is initially a unit matrix. Carrying out these operations row by row until the last row is completed will leave us with a system of equations that resemble

$$\begin{pmatrix} a'_{11} & 0 & \cdots & 0 \\ 0 & a'_{22} & \cdots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \cdots & a'_{nn} \end{pmatrix} \begin{pmatrix} c'_1 \\ c'_2 \\ \cdot \\ \cdot \\ c'_n \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \cdots & b_{1n} \\ b_{21} & b_{22} \cdots & b_{2n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ b_{n1} & b_{n2} \cdots & b_{nn} \end{pmatrix} . \tag{2.2.12}$$

If one examines the operations we have performed in light of theorems 2 and 7 from section 1.2, it is clear that so far we have done nothing to change the determinant of the original matrix **A** so that expansion by minors of the modified matrix represent by the elements a'_{ij} is simply accomplished by multiplying the diagonal elements a_{ii} together. A final step of dividing each row by a'_{ii} will yield the unit matrix on the left hand side and elements of the solution vector x_i will be found where the C'_i s were. The final elements of **B** will be the elements of the inverse matrix of **A**. Thus we have both solved the system of equations and found the inverse of the original matrix by performing the same steps on the constant vector as well as an additional unit matrix. Perhaps the simplest way to see why this works is to consider the system of linear equations and what the operations mean to them. Since all the operations are performed on entire rows including the constant vector, it is clear that they constitute legal algebraic operations that won't change the nature of the solution in any way. Indeed these are nothing more than the operations that one would preform by hand if he/she were solving the system by eliminating the appropriate variables. We have simply formalized that procedure so that it may be carried out in a systematic fashion. Such a procedure lends itself to computation by machine and may be relatively easily programmed. The reason for the algorithm yielding the matrix inverse is somewhat less easy to see. However, the product of **A** and **B** will be the unit matrix **1**, and the operations that go into that matrix-multiply are the inverse of those used to generate **B**.

To see specifically how the Gauss-Jordan routine works, consider the following system of equations:

$$\left.\begin{aligned} x_1 + 2x_2 + 3x_3 &= 12 \\ 3x_1 + 2x_2 + x_3 &= 24 \\ 2x_1 + x_2 + 3x_3 &= 36 \end{aligned}\right\} . \qquad (2.2.13)$$

If we put this in the form required by expression (2.2.11) we have

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix}\begin{pmatrix} 12 \\ 24 \\ 36 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} . \qquad (2.2.14)$$

Now normalize the all rows by factoring out the lead elements of the first column so that

$$(1)(3)(2)\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2/3 & 1/3 \\ 1 & 1/2 & 3/2 \end{pmatrix}\begin{pmatrix} 12 \\ 8 \\ 18 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/3 & 0 \\ 0 & 0 & 1/2 \end{pmatrix} . \qquad (2.2.15)$$

The first row can then be subtracted from the remaining rows (i.e. rows 2 and 3) to yield

$$(6)\begin{pmatrix} 1 & 2 & 3 \\ 0 & -4/3 & -8/3 \\ 0 & -3/2 & -3/2 \end{pmatrix}\begin{pmatrix} 12 \\ -4 \\ +6 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1/3 & 0 \\ -1 & 0 & 1/2 \end{pmatrix} . \qquad (2.2.16)$$

Now repeat the cycle normalizing by factoring out the elements of the second column getting

$$(6)\left(\frac{-4}{3}\right)\left(\frac{-3}{2}\right)(2)\begin{pmatrix} 1/2 & 1 & 3/2 \\ 0 & 1 & 2 \\ 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} +6 \\ +3 \\ -4 \end{pmatrix}\begin{pmatrix} 1/2 & 0 & 0 \\ 3/4 & 1/4 & 0 \\ 2/3 & 0 & -1/3 \end{pmatrix} . \qquad (2.2.17)$$

Subtracting the second row from the remaining rows (i.e. rows 1 and 3) gives

$$(24)\begin{pmatrix} 1/2 & 0 & -1/2 \\ 0 & 1 & 2 \\ 0 & 1 & -1 \end{pmatrix}\begin{pmatrix} +3 \\ +3 \\ -7 \end{pmatrix}\begin{pmatrix} -1/4 & 1/4 & 0 \\ 3/4 & -1/4 & 0 \\ -1/12 & 1/4 & -1/3 \end{pmatrix} . \qquad (2.2.18)$$

Again repeat the cycle normalizing by the elements of the third column so

$$(24)(-1/2)(2)(-1)\begin{pmatrix} -1 & 0 & 1 \\ 0 & 1/2 & 1 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} -6 \\ 3/2 \\ +7 \end{pmatrix}\begin{pmatrix} 1/2 & -1/2 & 0 \\ 3/8 & -1/8 & 0 \\ 1/12 & -1/4 & 1/3 \end{pmatrix} , \qquad (2.2.19)$$

and subtract from the remaining rows to yield

$$(24)\begin{pmatrix} -1 & 0 & 0 \\ 0 & \tfrac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} -13 \\ -11/2 \\ +7 \end{pmatrix}\begin{pmatrix} 5/12 & -1/4 & -1/3 \\ 7/24 & 1/8 & -1/3 \\ 1/12 & -1/4 & 1/3 \end{pmatrix} . \tag{2.2.20}$$

Finally normalize by the remaining elements so as to produce the unit matrix on the left hand side so that

$$(24)(-1)(1/2)(+1)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} +13 \\ -11 \\ +7 \end{pmatrix}\begin{pmatrix} -5/12 & 1/4 & 1/3 \\ 7/24 & 1/8 & -1/3 \\ 1/12 & -1/4 & 1/3 \end{pmatrix} . \tag{2.2.21}$$

The solution to the equations is now contained in the center vector while the right hand matrix contains the inverse of the original matrix that was on the left hand side of expression (2.2.14). The scalar quantity accumulating at the front of the matrix is the determinant as it represents factors of individual rows of the original matrix. Here we have repeatedly use theorem 2 and 7 given in section (1.2) in chapter 1. Theorem 2 allows us to build up the determinant by factoring out elements of the rows, while theorem 7 guarantees that the row subtraction shown in expressions (2.2.16), (2.2.18), and (2.2.20) will not change the value of the determinant. Since the determinant of the unit matrix on left side of expression (2.2.21) is one, the determinant of the original matrix is just the product of the factored elements. Thus our complete solution is

$$\left.\begin{aligned} &\vec{x} = [13, -11, +7] \\ \\ &\text{Det}\,\mathbf{A} = -12 \\ &\mathbf{A}^{-1} = \begin{pmatrix} -5/12 & 1/4 & 1/3 \\ 7/12 & 1/4 & -2/3 \\ 1/12 & -1/4 & 1/3 \end{pmatrix} \end{aligned}\right\} . \tag{2.2.22}$$

In carrying out this procedure, we have been careful to maintain full accuracy by keeping the fractions that explicitly appear as a result of the division. In general, this will not be practical and the perceptive student will have notice that there is the potential for great difficulty as a result of the division. Should any of the elements of the matrix **A** be zero when they are to play the role of divisor, then a numerical singularity will result. Indeed, should the diagonal elements be small, division would produce such large row elements that subtraction of them from the remaining rows would generate significant roundoff error. However, interchanging two rows or two columns of a system of equations doesn't alter the solution of these equations and, by theorem 5 of chapter 1 (sec 1.2), only the sign of the determinant is changed. Since the equations at each step represent a system of equations, which have the same solution as the original set, we may interchange rows and columns at any step in the procedure without altering the solution. Thus, most Gauss-Jordan programs include a search of the matrix to place the largest element on the diagonal prior to division by that element so as to minimize the effects of round off error. Should it be impossible to remove a zero from the division part of this algorithm, the one column of the matrix can be made to be completely zero. Such a matrix has a determinant, which is zero and the matrix is said to be *singular*. Systems of equations that are characterized by singular matrices have no unique solution.

It is clear that one could approach the singular state without actually reaching it. The result of this would be to produce a solution of only marginal accuracy. In such circumstances the initial matrix might have coefficients with six significant figures and the solution have one or less. While there is no *a priori* way of knowing how nearly singular the matrix may be, there are several "rules of thumb" which while not guaranteed to resolve the situation, generally work. First consider some characteristic of the matrix that measures the typical size of its elements. Most any reasonable criterion will do such as the absolute value of the largest element, the sum of the absolute values of the elements, or possibly the trace. Divide this characteristic by the absolute value of the determinant and if the result exceeds the machine precision, the result of the solution should be regarded with suspicion. Thus if we denote this characteristic of the matrix by M, then

$$N \gtrsim \log_{10}|M/d| \quad , \tag{2.2.23}$$

where d is the determinant of the original matrix. This should be regarded as a necessary, but not sufficient, condition for the solution to be accurate. Indeed a rough guess as to the number of significant figures in the resultant solution is

$$N_s \sim N - \log_{10}|M/d| \quad . \tag{2.2.24}$$

Since most Gauss-Jordan routines return the determinant as a byproduct of the solution, it is irresponsible to fail to check to see if the solution passes this test.

An additional test would be the substitution of the solution back into the original equations to see how accurately the elements of the constant vector are reproduced. For the inverse matrix, one can always multiply the original matrix by the inverse and see to what extent the unit matrix results. This raises an interesting question. What do we mean when we say that a solution to a system of equations is accurate. One could mean that each element of the solution vector contains a certain number of significant figures, or one might mean that the solution vector satisfies the equations at some acceptable level of accuracy (i.e. all elements of the constant vector are reproduced to some predetermined number of significant figures). It is worth noting that these two conditions are not necessarily the same. Consider the situation of a poorly conditioned system of equations where the constant vector is only weakly specified by one of the unknowns. Large changes in its value will make little change in the elements of the constant vector so that tight tolerances on the constant vector will not yield values of the that particular unknown with commensurate accuracy. This system would not pass the test given by equation (2.2.23). In general, there should always be an *a priori* specification of the required accuracy of the solution and an effort must be made to ascertain if that level of accuracy has been reached.

d. Solution by Matrix Factorization: The Crout Method

Consider two triangular matrices **U** and **V** with the following properties

$$\left.\begin{aligned} \mathbf{U} &= \begin{pmatrix} u_{ij} & i \le j \\ 0 & i > j \end{pmatrix} \\ \mathbf{V} &= \begin{pmatrix} 0 & i < j \\ v_{ij} & i \ge j \end{pmatrix} \end{aligned}\right\} \quad . \tag{2.2.25}$$

Further assume that **A** can be written in terms of these triangular matrices so that

$$\mathbf{A} = \mathbf{VU} \; . \tag{2.2.26}$$

Then our linear system of equations [equation (2.2.2)] could be written as

$$\mathbf{A}\vec{x} = \vec{c} = \mathbf{V}(\mathbf{U}\vec{x}) \ . \tag{2.2.27}$$

Multiplying by $\mathbf{V}^{-1}$ we have that the solution will be given by a different set of equations

$$\mathbf{U}\vec{x} = \mathbf{V}^{-1}\vec{c} = \vec{c}' \ , \tag{2.2.28}$$

where

$$\vec{c} = \mathbf{V}\vec{c}' \ . \tag{2.2.29}$$

If the vector $\vec{c}'$ can be determined, then equation (2.2.28) has the form of the result of the Gauss elimination and would resemble expression (2.2.9) and have a solution similar to equation (2.2.10). In addition, equation (2.2.29) is triangular and has a similarly simple solution for the vector $\vec{c}'$. Thus, we have replaced the general system of linear equations by two triangular systems. Now the constraints on **U** and **V** only depend on the matrix **A** and the triangular constraints. In no way do they depend on the constant vector $\vec{c}$. Thus, if one has a large number of equations differing only in the constant vector, the matrices **U** and **V** need only be found once.

The matrices **U** and **V** can be found from the matrix **A** in a fairly simple way by

$$\left.\begin{aligned} u_{ij} &= a_{ij} - \sum_{k=1}^{i-1} v_{ik} u_{kj} \\ v_{ij} &= \left(a_{ij} - \sum_{k=1}^{j-1} v_{ik} u_{kj} \right) \Big/ u_{ii} \end{aligned}\right\} \ , \tag{2.2.30}$$

which is justified by Hildebrandt[1]. The solution of the resulting triangular equations is then just

$$\left.\begin{aligned} c'_i &= \left(c_i - \sum_{k=1}^{i-1} v_{ik} c'_k \right) \Big/ v_{ii} \\ x_i &= \left(c'_i - \sum_{k=i+1}^{n} u_{ik} x_k \right) \Big/ u_{ii} \end{aligned}\right\} \ . \tag{2.2.31}$$

Both equations (2.2.30) and (2.2.31) are recursive in nature in that the unknown relies on previously determined values of the same set of unknowns. Thus round-off error will propagate systematically throughout the solution. So it is useful if one attempts to arrange the initial equations in a manner which minimizes the error propagation. However, the method involves a minimum of readily identifiable divisions and so tends to be exceptionally stable. The stability will clearly be improved as long as the system of equations contains large diagonal elements. Therefore the Crout method provides a method of similar or greater stability to Gauss-Jordan method and considerable efficiency in dealing with systems differing only in the constant vector. In instances where the matrix **A** is symmetric the equations for u_{ij} simplify to

$$u_{ij} = v_{ji}/u_{ii} \ . \tag{2.2.32}$$

As we shall see the normal equations for the least squares formalism always have this form so that the Crout method provides a good basis for their solution.

While equations (2.2.30) and (2.2.31) specifically delineate the elements of the factored matrices **U** and **V**, it is useful to see the manner in which they are obtained. Therefore let us consider the same equations that served as an example for the Gauss-Jordan method [i.e. equations (2.2.13)]. In order to implement the Crout method we wish to be able to express the coefficient matrix as

$$\mathbf{A} = \mathbf{VU} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix} = \begin{pmatrix} v_{11} & 0 & 0 \\ v_{12} & v_{22} & 0 \\ v_{13} & v_{23} & v_{33} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} . \qquad (2.2.33)$$

The constant vector $\vec{c}$ that appears in equation (2.2.31) is

$$\vec{c} = (\ 12, 24, 36\ ) \ . \qquad (2.2.34)$$

To factor the matrix **A** into the matrices **U** and **V** in accordance with equation (2.2.30), we proceed column by column through the matrix so that the necessary elements of **U** and **V** required by equation (2.2.30) are available when they are needed. Carrying out the factoring process specified by equations (2.2.30) sequentially column by column yields

$$\left.\begin{array}{l}
\left.\begin{array}{l}
u_{11} = a_{11} - 0 = 1 \\
v_{11} = (a_{11} - 0) \big/ u_{11} = 1 \\
v_{12} = (a_{12} - 0) \big/ u_{11} = 3 \\
v_{13} = (a_{13} - 0) \big/ u_{11} = 2
\end{array}\right\} \quad j = 1 \\[2em]
\left.\begin{array}{l}
u_{12} = a_{12} - 0 = 2 \\
u_{22} = [a_{22} - (v_{21}u_{21})] = 2 - (3 \times 2) = 4 \\
v_{22} = [a_{22} - (v_{21}u_{12})] \big/ u_{22} = [2 - (3 \times 2)] \big/ -4 = 1 \\[1em]
v_{32} = [a_{32} - (v_{31}u_{13})] \big/ u_{22} = [1 - (2 \times 2)] \big/ -4 = \frac{3}{4}
\end{array}\right\} \quad j = 2 \\[2em]
\left.\begin{array}{l}
u_{13} = a_{13} - 0 = 3 \\
u_{23} = a_{23} - (v_{21}u_{13}) = 1 - (3 \times 3) = -8 \\
u_{33} = a_{33} - (v_{31}u_{13} + v_{32}u_{23}) = 3 - [(2 \times 3) + (\tfrac{3}{4} \times -8)] = 3 \\[1em]
v_{33} = [a_{33} - (v_{31}u_{13} + v_{32}u_{23})] \big/ u_{33} = [3 - (2 \times 3) - (-8 \times \tfrac{3}{4})] / 3 = 1
\end{array}\right\} \quad j = 3
\end{array}\right\} . \qquad (2.2.35)$$

Therefore we can write the original matrix **A** in accordance with equation (2.2.33) as

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \tfrac{3}{4} & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 0 & -4 & -8 \\ 0 & 0 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & (6-4) & (9-8) \\ 2 & (4-3) & (6-6+3) \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix} . \qquad (2.2.36)$$

Here the explicit multiplication of the two factored matrices **U** and **V** demonstrates that the factoring has been done correctly.

Now we need to obtain the augmented constant vector $\vec{c}'$ specified by equations (2.2.31). These equations must be solved recursively so that the results appear in the order in which they are needed. Thus

$$\left.\begin{aligned} c'_1 &= (c_1 - 0)/v_{11} &&= 12/1 &&= 12 \\ c'_2 &= [c_2 - (v_{21}c'_1)]/v_{22} &&= [24-(3\times 12)]/1 && -12 \\ c'_3 &= [c_3 - (v_{31}c'_1 + v_{32}c'_2)]/v_{33} &&= [36-(2\times 2)+(12\times \tfrac{3}{4})]/1 &&= 1 \end{aligned}\right\} . \qquad (2.2.37)$$

Finally the complete solution can be obtained by back-solving the second set of equations (2.2.31) so that

$$\left.\begin{aligned} x_3 &= c'_3/u_{33} = 21/3 = 7 \\ x_2 &= (c'_2 - u_{23}x_3)/u_{22} &&= [-12+(8\times 7)]/(-4) &&= -11 \\ x_1 &= (c'_1 - u_{12}x_2 - u_{13}x_3)/u_{11} &&= [12-(2\times -11)-(3\times 7)]/1 &&= 13 \end{aligned}\right\} . \qquad (2.2.38)$$

As anticipated, we have obtained the same solution as in equation (2.2.22). The strength of the Crout method resides in the minimal number of operations required to solve a second set of equations differing only in the constant vector. The factoring of the matrix remains the same and only the steps specified by equations (2.2.37) and (2.2.38) need be repeated. In addition, the method is particularly stable.

e. The Solution of Tri-diagonal Systems of Linear Equations

All the methods described so far generally require about n^3 operations to obtain the solution. However, there is one frequently occurring system of equations for which extremely efficient solution algorithms exist. This system of equations is called tri-diagonal because there are never more than three unknowns in any equation and they can be arranged so that the coefficient matrix is composed of non-zero elements on the main diagonal and the diagonal immediately adjacent to either side. Thus such a system would have the form

$$\left.\begin{array}{ccccccccccc}
a_{11}x_1 + a_{12}x_2 + & 0 & + & 0 & + & \cdot\ \cdot\ \cdot\ \cdot & & \cdot\ \cdot\ \cdot\ \cdot\ \cdot & & + \ 0 & = c_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + & 0 & + & & & \cdot\ \cdot\ \cdot\ \cdot & & \cdot\ \cdot\ \cdot\ \cdot\ \cdot & & + \ 0 & = c_2 \\
0 \ + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 + & & & & & \cdot\ \cdot\ \cdot\ \cdot & & \cdot\ \cdot\ \cdot\ \cdot\ \cdot & & + \ 0 & = c_3 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 \ + \ 0 \ + \ 0 \ + \ 0 \ + & \cdot\ \cdot\ \cdot & + a_{n-1\,n-2}x_{n-2} & + a_{n-1\,n-1}x_{n-1} & + a_{n-1\,n}x_n & & & & & & = c_{n-1} \\
0 \ + \ 0 \ + \ 0 \ + \ 0 \ + & \cdot\ \cdot\ \cdot & + \quad 0 & + a_{n-1\,n}x_{n-1} & + a_{n\,n}x_n & & & & & & = c_n
\end{array}\right\} . \quad (2.2.39)$$

Equations of this type often occur as a result of using a finite difference operator to replace a differential operator for the solution of differential equations (see chapter 5). A routine that performed straight Gauss elimination would only be involved in one subtraction below the diagonal normalization element and so would reach its 'triangular' form after n steps. Since the resulting equations would only contain two terms, the back substitution would also only require two steps meaning that the entire process would require something of the order of 3n steps for the entire solution. This is so very much more efficient than the general solution and equations of this form occur sufficiently frequently that the student should be aware of this specialized solution.

2.3 Solution of Linear Equations by Iterative Methods

So far we have dealt with methods that will provide a solution to a set of linear equations after a finite number of steps (generally of the order of n^3). The accuracy of the solution at the end of this sequence of steps is fixed by the nature of the equations and to a lesser extent by the specific algorithm that is used. We will now consider a series of algorithms that provide answers to a linear system of equations in considerably fewer steps, but at a level of accuracy that will depend on the number of times the algorithm is applied. Such methods are generally referred to as iterative methods and they usually require of the order of n^2 steps for each iteration. Clearly for very large systems of equations, these methods may prove very much faster than direct methods providing they converge quickly to an accurate solution.

a. Solution by the Gauss and Gauss-Seidel Iteration Methods

All iterative schemes begin by assuming that an approximate answer is known and then the scheme proceeds to improve that answer. Thus we will have a solution vector that is constantly changing from iteration to iteration. In general, we will denote this by a superscript in parentheses so that $x^{(i)}$ will denote the value of x at the ith iteration. Therefore in order to begin, we will need an initial value of the solution vector $\vec{x}^{(0)}$. The concept of the Gauss iteration scheme is extremely simple. Take the system of linear equations as expressed in equations (2.2.1) and solve each one for the diagonal value of x so that

$$x_i = \frac{\left[c_i - \sum_{i\neq j}^{n} a_{ij}x_j\right]}{a_{ii}} \quad . \tag{2.3.1}$$

Now use the components of the initial value of on the right hand side of equation (2.3.1) to obtain an improved value for the elements. This procedure can be repeated until a solution of the desired accuracy is obtained. Thus the general iteration formula would have the form

$$x_i^{(k)} = \frac{\left[c_i - \sum_{i\neq j}^{n} a_{ij}x_j^{(k-1)}\right]}{a_{ii}} \quad . \tag{2.3.2}$$

It is clear, that should any of the diagonal elements be zero, there will be a problem with the stability of the method. Thus the order in which the equations are arranged will make a difference to in the manner in which this scheme proceeds. One might suppose that the value of the initial guess might influence whether or not the method would find the correct answer, but as we shall see in section 2.4 that is not the case. However, the choice of the initial guess will determine the number of iterations required to arrive at an acceptable answer.

The Gauss-Seidel scheme is an improvement on the basic method of Gauss. Let us rewrite equations (2.3.1) as follows:

$$x_i^{(k)} = \frac{\left[c_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right]}{a_{ii}} \quad . \tag{2.3.3}$$

When using this as a basis for an iteration scheme, we can note that all the values of x_j in the first summation for the kth iteration will have been determined before the value of $x_i^{(k)}$ so that we could write the iteration scheme as

$$x_i^{(k)} = \frac{\left[c_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right]}{a_{ii}} \quad . \tag{2.3.4}$$

Here the improved values of x_i are utilized as soon as they are obtained. As one might expect, this can lead to a faster rate of convergence, but there can be a price for the improved speed. The Gauss-Seidel scheme may not be as stable as the simple Gauss method. In general, there seems to be a trade off between speed of convergence and the stability of iteration schemes.

Indeed, if we were to apply either if the Gauss iterative methods to equations (2.2.13) that served as an example for the direct method, we would find that the iterative solutions would not converge. We shall see later (sections 2.3d and 2.4) that those equations fail to satisfy the simple sufficient convergence criteria given in section 2.3d and the necessary and sufficient condition of section 2.4. With that in mind, let us consider another 3×3 system of equations which does satisfy those conditions. These equations are much more strongly diagonal than those of equation (2.2.13) so

$$\left.\begin{aligned} 3x_1 + x_2 + x_3 &= 8 \\ x_1 + 4x_2 + 2x_3 &= 15 \\ 2x_1 + x_2 + 5x_3 &= 19 \end{aligned}\right\} . \tag{2.3.5}$$

For these equations, the solution under the Gauss-iteration scheme represented by equations (2.3.2) takes the form

$$\left.\begin{aligned} x_1^{(k+1)} &= \left[8 - x_2^{(k)} - x_3^{(k)}\right] \big/ 3 \\ x_2^{(k+1)} &= \left[15 - x_1^{(k)} - 2x_3^{(k)}\right] \big/ 4 \\ x_3^{(k+1)} &= \left[19 - 2x_1^{(k)} - x_2^{(k)}\right] \big/ 5 \end{aligned}\right\} . \tag{2.3.6}$$

However, if we were to solve equations (2.3.5) by means of the Gauss-Seidel method the iterative equations for the solution would be

$$\left.\begin{aligned} x_1^{(k+1)} &= \left[8 - x_2^{(k)} - x_3^{(k)}\right] \big/ 3 \\ x_2^{(k+1)} &= \left[15 - x_1^{(k+1)} - 2x_3^{(k)}\right] \big/ 4 \\ x_3^{(k+1)} &= \left[19 - 2x_1^{(k+1)} - x_2^{(k+1)}\right] \big/ 5 \end{aligned}\right\} . \tag{2.3.7}$$

If we take the initial guess to be

$$x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = 1 \ , \tag{2.3.8}$$

then repetitive use of equations (2.3.6) and (2.3.7) yield the results given in Table 2.1.

Table 2.1

Convergence of Gauss and Gauss-Seidel Iteration Schemes

K	0		1		2		3		4		5		10	
	G	GS	G	GS	G	GS	G	GS	G	GS	G	GS	G	GS
x_1	1.00	1.00	2.00	2.00	0.60	0.93	1.92	0.91	0.71	0.98	1.28	1.00	0.93	1.00
x_2	1.00	1.00	3.00	2.75	1.65	2.29	2.64	2.03	1.66	1.99	2.32	2.00	1.92	2.00
x_3	1.00	1.00	3.20	2.45	1.92	2.97	3.23	3.03	2.51	3.01	3.18	3.00	2.95	3.00

As is clear from the results labeled "G" in table 2.1, the Gauss-iteration scheme converges very slowly. The correct solution which would eventually be obtained is

$$\vec{x}^{(\infty)} = [\,1,2,3\,] \ . \tag{2.3.9}$$

There is a tendency for the solution to oscillate about the correct solution with the amplitude slowly damping out toward convergence. However, the Gauss-Seidel iteration method damps this oscillation very rapidly by employing the improved values of the solution as soon as they are obtained. As a result, the Gauss-Seidel scheme has converged on this problem in about 5 iterations while the straight Gauss scheme still shows significant error after 10 iterations.

b. The Method of Hotelling and Bodewig

Assume that the correct solution to equation (2.2.3) can be written as

$$\vec{x}_c = \mathbf{A}^{-1}\vec{c}\ , \tag{2.3.10}$$

but that the actual solution that is obtained by matrix inversion is really

$$\vec{x}^{(k)} = (\mathbf{A}^{-1})^{(k)}\vec{c} \ . \tag{2.3.11}$$

Substitution of this solution into the original equations would yield a slightly different constant vector, namely

$$\vec{c}^{(k)} = \mathbf{A}\vec{x}^{(k)} . \tag{2.3.12}$$

Let us define a residual vector in terms of the constant vector we started with and the one that results from the substitution of the correct solution into the original equations so that

$$\vec{R}^{(k)} = \vec{c}^{(k)} - \vec{c} = \mathbf{A}\vec{x}^{(k)} - \mathbf{A}\vec{x}_c = \mathbf{A}(\vec{x}^{(k)} - \vec{x}_c) . \tag{2.3.13}$$

Solving this for the true solution $\vec{x}_c$ we get

$$\vec{x}_c = \vec{x}^{(k)} - [\mathbf{A}^{-1}]^{(k)}\vec{R}^{(k)} = \vec{x}^{(k)} - [\mathbf{A}^{-1}]^{(k)}\vec{c}^{(k)} + [\mathbf{A}^{-1}]^{(k)}\vec{c} = [\mathbf{A}^{-1}]^{(k)}[2\vec{c} - \vec{c}^{(k)}] \ . \tag{2.3.14}$$

The solution of equation (2.3.13) will involve basically the same steps as required to solve equation (2.3.11). Thus the quantity $(\vec{x}^{(k)} - \vec{x}_c)$ will be found with the same accuracy as $\vec{x}^{(k)}$ providing $\vec{R}^{(k)}$ is not too large.

Now we can write $\vec{c}^{(k)}$ in terms $\vec{c}$ of by using equations (2.3.11, 12) and get

$$\vec{c}^{(k)} = \mathbf{A}\vec{x}^{(k)} = \mathbf{A}[\mathbf{A}^{-1}]^{(k)}\vec{c} \ . \tag{2.3.15}$$

Using this result to eliminate $\vec{c}^{(k)}$ from equation (2.3.14) we can write the "correct" solution $\vec{x}_c$ in terms of the approximate matrix inverse $[\mathbf{A}^{-1}]^{(k)}$ as

$$\vec{x}_c = [\mathbf{A}^{-1}]^{(k)}\{2\times\mathbf{1} - \mathbf{A}[\mathbf{A}^{-1}]^{(k)}\}\vec{c} . \tag{2.3.16}$$

Here **1** denotes the unit matrix with elements equal to the Kronecker delta δ_{ij}. Round-off error and other problems that gave rise to the initially inaccurate answer will in reality keep $\vec{x}_c$ from being the correct answer, but it may be regarded as an improvement over the original solution. It is tempting to use equation (2.3.16) as the basis for a continuous iteration scheme, but in practice very little improvement can be made over a single application as the errors that prevent equation (2.3.16) from producing the correct answer will prevent any further improvement over a single iteration.

If we compare equations (2.3.10) and (2.3.16), we see that this method provides us with a mechanism for improving the inverse of a matrix since

$$\mathbf{A}^{-1} = [\mathbf{A}^{-1}]^{(k)}\{2\times\mathbf{1} - \mathbf{A}[\mathbf{A}^{-1}]^{(k)}\} \ . \tag{2.3.17}$$

All of the problems of using equation (2.3.16) as an iteration formula are present in equation (2.3.17). However, the matrix inverse as obtained from equation (2.3.17) should be an improvement over $[\mathbf{A}^{-1}]^{(k)}$.

To see how this method works, consider the equations used to demonstrate the Gauss-Jordan and Crout methods. The exact matrix inverse is given in equations (2.2.22) so we will be able to compare the iterated matrix with the correct value to judge the improvement. For demonstration purposes, assume that the inverse in equation (2.2.22) is known only to two significant figures so that

$$(\mathbf{A}^{-1})^{(k)} = \begin{pmatrix} -0.42 & 0.25 & 0.33 \\ 0.58 & 0.25 & -0.67 \\ 0.08 & -0.25 & 0.33 \end{pmatrix} . \tag{2.3.18}$$

Taking the constant vector to be the same as equation (2.2.13), the solution obtained from the imperfect matrix inverse would be

$$\vec{x}^{(k)} = (\mathbf{A}^{-1})^{(k)}\vec{c} = \begin{pmatrix} -0.42 & 0.25 & 0.33 \\ 0.58 & 0.25 & -0.67 \\ 0.08 & -0.25 & 0.33 \end{pmatrix}\begin{pmatrix} 12 \\ 24 \\ 36 \end{pmatrix} = \begin{pmatrix} 12.84 \\ -11.16 \\ 6.84 \end{pmatrix} . \tag{2.3.19}$$

and substitution of this solution into the original equations [i.e. equation (2.2.13)] will yield the constant vector $\vec{c}_k$ with the elements

$$\mathbf{A}\vec{x}^{(k)} = \vec{c}^{(k)} = \begin{pmatrix} 1.0 & 2.0 & 3.0 \\ 3.0 & 2.0 & 1.0 \\ 2.0 & 1.0 & 3.0 \end{pmatrix}\begin{pmatrix} 12.84 \\ -11.16 \\ 6.84 \end{pmatrix} = \begin{pmatrix} 11.04 \\ 23.04 \\ 35.04 \end{pmatrix} , \tag{2.3.20}$$

that are used to obtain the residual vector in equation (2.3.13).

The method of Hotelling and Bodewig operates by basically finding an improved value for the matrix inverse and then using that with the original constant vector to obtain an improved solution. Therefore, using equation (2.3.17) to improve the matrix inverse we get

$$\mathbf{A}^{-1} = [\mathbf{A}^{-1}]^{(k)}\{2\times\mathbf{1}\text{-}\mathbf{A}[\mathbf{A}^{-1}]^{(k)}\},$$

or for example

$$\begin{aligned}
A^{-1} &= [A^{-1}]^{(k)}\left[\begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} - \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix}\begin{pmatrix} -0.42 & 0.25 & 0.33 \\ 0.58 & 0.25 & -0.67 \\ 0.08 & 0.25 & 0.33 \end{pmatrix}\right] \\
&= [A^{-1}]^{(k)}\left[\begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} - \begin{pmatrix} 0.98 & 0.00 & -0.02 \\ -0.02 & 1.00 & -0.02 \\ 0.02 & 0.00 & 0.98 \end{pmatrix}\right] \quad . \qquad (2.3.21) \\
&= [A^{-1}]^{(k)}\begin{pmatrix} 1.02 & 0.00 & 0.02 \\ 0.02 & 1.00 & 0.02 \\ 0.02 & 0.00 & 1.02 \end{pmatrix}
\end{aligned}$$

and performing the final matrix multiplication we have

$$\mathbf{A}^{-1} = \begin{pmatrix} -0.42 & 0.25 & 0.33 \\ 0.58 & 0.25 & -0.67 \\ 0.08 & -0.25 & 0.33 \end{pmatrix}\begin{pmatrix} 1.02 & 0.00 & 0.02 \\ 0.02 & 1.00 & 0.02 \\ 0.02 & 0.00 & 1.02 \end{pmatrix} = \begin{pmatrix} -0.4168 & 0.2500 & 0.3332 \\ 0.5832 & 0.2500 & -0.6668 \\ 0.0832 & -0.2500 & 0.3332 \end{pmatrix} . \qquad (2.3.22)$$

This can be compared with the six figure version of the exact inverse from equation (2.2.22) which is

$$\mathbf{A}^{-1} = \begin{pmatrix} -0.416667 & 0.250000 & 0.333333 \\ 0.583333 & 0.250000 & -0.666667 \\ 0.083333 & -0.250000 & 0.333333 \end{pmatrix} . \qquad (2.3.23)$$

Every element experienced a significant improvement over the two figure value [equation(2.3.18)]. It is interesting that the elements of the original inverse for which two figures yield an exact result (i.e $a_{12}^{-1}, a_{22}^{-1}, a_{32}^{-1}$) remain unchanged. This result can be traced back to the augmentation matrix [i.e. the right hand matrix in equation (2.3.21) third line]. The second column is identical to the unit matrix so that the second column of the initial inverse will be left unchanged.

We may now use this improved inverse to re-calculate the solution from the initial constant vector and get

$$\vec{x}_c = \mathbf{A}^{-1}\vec{c} = \begin{pmatrix} -0.4168 & 0.2500 & 0.3332 \\ 0.5832 & 0.2500 & -0.6668 \\ 0.0832 & -0.2500 & 0.3332 \end{pmatrix}\begin{pmatrix} 12 \\ 24 \\ 36 \end{pmatrix} = \begin{pmatrix} 12.99 \\ -11.00 \\ 6.994 \end{pmatrix} . \qquad (2.3.24)$$

As one would expect from the improved matrix inverse, the solution represents a significant improvement over the initial values given by equation (2.2.19). Indeed the difference between this solution and the exact solution given by equation (2.2.22) is in the fifth significant which is smaller than the calculation accuracy used to obtain the improved inverse. Thus we see that the method of Hotelling and Bodewig is a powerful algorithm for improving a matrix inverse and hence a solution to a system of linear algebraic equations.

c. *Relaxation Methods for the Solution of Linear Equations*

The Method of Hotelling and Bodewig is basically a specialized relaxation technique and such techniques can be used with virtually any iteration scheme. In general, relaxation methods tend to play off speed of convergence for stability. Rather than deal with the general theory of relaxation techniques, we will illustrate them by their application to linear equations.

As in equation (2.3.8) we can define a residual vector $\vec{R}^{(k)}$ as

$$\vec{R}^{(k)} = \mathbf{A}\vec{x}^{(k)} - \vec{c} . \tag{2.3.25}$$

Let us assume that each element of the solution vector $\vec{x}^{(k)}$ is subject to an improvement $\delta\vec{x}^{(k)}$ so that

$$x_j^{(k+1)} = x_j^{(k)} + \delta x_j . \tag{2.3.26}$$

Since each element of the solution vector may appear in each equation, a single correction to an element can change the entire residual vector. The elements of the new residual vector will differ from the initial residual vector by an amount

$$\delta R_{im} = - a_{im}\delta x_m . \tag{2.3.27}$$

Now search the elements of the matrix δR_{im} over the index m for the largest value of δR_{im} and reduce the corresponding residual by that maximum value so that

$$\rho_i^{(k)} \equiv -R_i^{(k)} / \underset{m}{\mathrm{Max}}(-\delta R_{im}) . \tag{2.3.28}$$

The parameter ρ_i is known as the *relaxation parameter* for the ith equation and may change from iteration to iteration. The iteration formula then takes the form

$$x_j^{(k+1)} = x_j^{(k)} + \rho_j^{(k)}\delta x_j . \tag{2.3.29}$$

Clearly the smaller ρ_i is, the smaller the correction to x_i will be and the longer the iteration will take to converge. The advantage of this technique is that it treats each unknown in an individual manner and thus tends to be extremely stable.

Providing a specific example of a relaxation process runs the risk of appearing to limit the concept. Unlike the other iterative procedures we have described, relaxation schemes leave the choice of the correction to the elements of the solution vector completely arbitrary. However, having picked the corrections, the scheme describes how much of them to apply by calculating a relaxation parameter for each element of the solution vector. While convergence of these methods is generally slow, their stability is often quite good. We shall demonstrate that by applying the method to the same system of equations used to demonstrate the other iterative processes [i.e. equations (2.3.5)].

We begin by choosing the same initial solution that we used for the initial guess of the iterative schemes [i.e. $\vec{x}$=(1, 1, 1)]. Inserting that initial guess into equation (2.3.5), we obtain the approximate constant vector $\vec{c}^{(k)}$, which yields a residual vector

$$\vec{R}_0 = \begin{pmatrix} 3 & 1 & 1 \\ 1 & 4 & 2 \\ 2 & 1 & 5 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 8 \\ 15 \\ 19 \end{pmatrix} = \begin{pmatrix} 5 \\ 7 \\ 8 \end{pmatrix} - \begin{pmatrix} 8 \\ 15 \\ 19 \end{pmatrix} = \begin{pmatrix} -3 \\ -8 \\ -11 \end{pmatrix} . \tag{2.3.30}$$

It should be emphasized that the initial guesses are somewhat arbitrary as they only define a place from which to start the iteration scheme. However, we will be able to compare the results given in Table 2.2 with the other iterative methods .

We will further arbitrarily choose to vary all the unknowns by the same amount so that

$$\delta x_m = 0.3 . \tag{2.3.31}$$

Now calculate the variational residual matrix specified by equation (2.3.27) and get

$$\delta R_{im} = -\begin{pmatrix} 3 & 1 & 1 \\ 1 & 4 & 2 \\ 2 & 1 & 5 \end{pmatrix} \times 0.3 = -\begin{pmatrix} 0.9 & 0.3 & 0.3 \\ 0.3 & 1.2 & 0.6 \\ 0.6 & 0.3 & 1.5 \end{pmatrix} . \tag{2.3.32}$$

The element of the matrix with the largest magnitude is δR_{33} = 1.5. We may now calculate the elements of the relaxation vector in accordance with equation (2.3.28) and modify the solution vector as in equation (2.3.29). Repeating this process we get the results in Table 2.2

Table 2.2

Sample Iterative Solution for the Relaxation Method

k	0	1	4	7	10	∞
	$i\backslash x_i$ ρ_i	x_i ρ_i	x_i ρ_i	x_i ρ_i	x_i ρ_i	x_i ρ_i
1	1.00 2.00	1.60 -1.07	1.103 –0.02	1.036-.107	0.998 .006	1.00 0.00
2	1.00 4.44	2.33 -1.55	2.072 +0.41	2.07 -.224	2.00 .002	2.00 0.00
3	1.00 7.33	3.20 -7.02	3.011 -.119	3.01 -.119	2.99 .012	3.00 0.00

We see that the solution does indeed converge at a rate that is intermediate between that obtain for the Gauss method and that of the Gauss-Seidel method. This application of relaxation techniques allows the relaxation vector to change approaching zero as the solution converges. Another approach is to use the relaxation parameter to change the correction given by another type of iteration scheme such as Gauss-Seidel. Under these conditions, it is the relaxation parameter that is chosen and usually held constant while the corrections approach zero.

There are many ways to arrive at suitable values for the relaxation parameter but the result will usually be in the range ½ρ½. For values of $\rho<½$, the rate of convergence is so slow that one is not sure when the solution has been obtained. On rare occasions one may choose a relaxation parameter greater than unity. Such a procedure is said to be *over relaxed* and is likely to become unstable. If $\rho \ \geq \ 2$, then instability is almost guaranteed. We have said a great deal about convergence, but little that is quantitative so let us turn to a brief discussion of convergence within the confines of fixed-point iteration theory.

d. Convergence and Fixed-point Iteration Theory

The problems of deciding when a correct numerical solution to a system of equations has been reached are somewhat more complicated for the iterative methods than with the direct methods. Not only does the practical problem of what constitutes a sufficiently accurate solution have to be dealt with, but the problem of whether or not the iteration method is approaching that solution has to be solved. The iteration method will most certainly produce a new solution set, but whether that set is any closer to the correct set is not immediately obvious. However, we may look to *fixed-point iteration theory* for some help with this problem.

Just as there is a large body of knowledge connected with relaxation theory, there is an equally large body of knowledge relating to fix-point iteration theory[2]. Before looking at iteration methods of many variables such as the Gauss iteration scheme, let us consider a much simpler iteration scheme of only one variable. We could write such a scheme as

$$x^{(k)} = \Phi[x^{(k-1)}] \ . \tag{2.3.33}$$

Here $\Phi[x^{(k-1)}]$ is any function or algorithm that produces an new value of x based on a previous value. Such a function is said to posses a *fixed-point* x_0 if

$$x_0 = \Phi(x_0) \ . \tag{2.3.34}$$

If $\Phi(x)$ provides a steady succession of values of x that approach the fixed-point x_0, then it can be said to be a convergent iterative function. There is a little-known theorem which states that a necessary and sufficient condition for $\Phi(x)$ to be a convergent iterative function is

$$\left|\frac{d\Phi(x)}{dx}\right| < 1 \quad \forall x \ \varepsilon \left|x^{(k)}\right| \leq |x| \leq |x_0| \ . \tag{2.3.35}$$

For multidimensional iterative functions of the form

$$x_i^{(k+1)} = \Phi_i(x_j^{(k)}) \, , \tag{2.3.36}$$

the theorem becomes

$$\sum_{j=1}^{n}\left|\frac{d\Phi_i(x_i)}{dx_j}\right|<1\ , \quad \forall x_i\ \varepsilon\left|x_i^{(k)}\right|\le\left|x_i\right|\le\left|x_{i0}\right| . \tag{2.3.37}$$

However, it no longer provides necessary conditions, only sufficient ones. If we apply this to the Gauss iteration scheme as described by equation (2.3.1) we have

$$\sum_{j\neq i}^{n}\left|\frac{a_{ij}}{a_{ii}}\right|<1\ , \quad \forall\, i\ . \tag{2.3.38}$$

It is clear that the convergence process is strongly influenced by the size of the diagonal elements present in the system of equations. Thus the equations should be initially arranged so that the largest possible elements are present on the main diagonal of the coefficient matrix. Since the equations are linear, the sufficient condition given in equation (2.2.23) means that the convergence of a system of equations under the Gauss iteration scheme is independent of the solution and hence the initial guess. If equation (2.2.23) is satisfied then the Gauss iteration method is guaranteed to converge. However, the number of iterations required to achieve that convergence will still depend on the accuracy of the initial guess.

If we apply these conditions to equations (2.2.13) which we used to demonstrate the direct methods of solution, we find that

$$\sum_{j\neq i}^{n}\left|\frac{a_{ij}}{a_{ii}}\right| = \begin{pmatrix} 5 \\ 2 \\ 1 \end{pmatrix} \quad . \tag{2.3.39}$$

Each equation fails to satisfy the sufficient convergence criteria given in equation (2.3.38). Thus it is unlikely that these equations can be solved by most iterative techniques. The fact that the method of Hotelling and Bodewig gave a significantly improved solution is a testament to the stability of that method. However, it must be remembered that the method of Hotelling and Bodewig is not meant to be used in an iterative fashion so comparison of iterative techniques with it is not completely justified.

The sufficient convergence criteria give by equation (2.3.38) essentially says that if the sum of the absolute values of the off-diagonal elements of every row is less than the absolute value of the diagonal element, then the iteration sequence will converge. The necessary and sufficient condition for convergence of this and the Gauss Seidel Scheme is that the eigenvalues of the matrix all be positive and less than one. Thus it is appropriate that we spend a little time to define what eigenvalues are, their importance to science, and how they may be obtained.

2.4 The Similarity Transformations and the Eigenvalues and Vectors of a Matrix

In Chapter 1 (section 1.3) we saw that it is often possible to represent one vector in terms of another by means of a system of linear algebraic equations which we called a coordinate transformation. If this transformation preserved the length of the vector, it was called an orthonormal transformation and the matrix of the transformation coefficients had some special properties. Many problems in science can be represented in terms of linear equations of the form

$$\vec{y} = \mathbf{A}\vec{x} . \tag{2.4.1}$$

In general, these problems could be made much simpler by finding a coordinate frame so that each element of the transformed vector is proportional to the corresponding element of the original vector. In other words, does there exist a space wherein the basis vectors are arranged so that the transformation is a diagonal matrix of the form

$$\vec{y}' = \mathbf{S}\vec{x}' , \tag{2.4.2}$$

where $\vec{x}'$ and $\vec{y}'$ represent the vectors $\vec{x}$ and $\vec{y}$ in this new space where the transformation matrix becomes diagonal. Such a transformation is called a *similarity transformation* as each element of $\vec{y}'$ would be similar (proportional) to the corresponding element of $\vec{x}'$. Now the space in which we express and is defined by a set of basis vectors $\hat{e}_i$ and the space in which $\vec{x}'$ and $\vec{y}'$ are expressed is spanned by $\hat{e}'_i$. If we let the transformation that relates the unprimed and primed coordinate frames be **D**, then the basis vectors are related by

$$\left.\begin{aligned} \hat{e}_i' &= \sum_j d_{ij}\hat{e}_j \\ \vec{e}' &= \mathbf{D}\vec{e} \end{aligned}\right\} . \tag{2.4.3}$$

Any linear transformation that relates the basis vectors of two coordinate frames will transform any vector from one frame to the other. Therefore

$$\left.\begin{aligned} \vec{x} &= \mathbf{D}^{-1}\vec{x}' \\ \vec{e}' &= \mathbf{D}\vec{e} \end{aligned}\right\} . \tag{2.4.4}$$

If we use the results of equation (2.4.4) to eliminate $\vec{x}$ and $\vec{y}$ from equation (2.4.1) in favor of $\vec{x}'$ and $\vec{y}'$ we get

$$\vec{y}' = [\mathbf{D}\mathbf{A}\mathbf{D}^{-1}]\vec{x}' = \mathbf{S}\vec{x}' \quad . \tag{2.4.5}$$

Comparing this result with equation (2.4.2) we see that the conditions for **S** to be diagonal are

$$\mathbf{DAD}^{-1} = \mathbf{S} \ , \tag{2.4.6}$$

which we can rewrite as

$$\mathbf{AD}^{T} = \mathbf{D}^{T}\mathbf{S} . \tag{2.4.7}$$

Here we have made use of an implicit assumption that the transformations are orthonormal and so preserve the length of vectors. Thus the conditions that lead to equation (1.3.8) are met and $\mathbf{D}^{-1} = \mathbf{D}^{T}$. We can write these equations in component form as

$$\sum_{k=1}^{n} a_{ik}d_{jk} = d_{ji}s_{jj} = \sum_{k=1}^{n} d_{jk}\delta_{ki}s_{jj} \quad , \quad i = 1\cdots n,\ j = 1\cdots n \qquad .(2.4.8)$$

These are n systems of linear homogeneous equations of the form

$$\sum_{k=1}^{n}(a_{ik} - \delta_{ki}s_{jj})d_{jk} = 0, \quad i = 1\cdots n,\ j = 1\cdots n \qquad , \tag{2.4.9}$$

which have a solution if and only if

$$\mathrm{Det}\left|a_{ik} - \delta_{ki} s_{jj}\right| = 0\ , \quad \forall j\ . \tag{2.4.10}$$

Now the nature of **D** and **S** depend only on the matrix **A** and in no way on the values of $\vec{x}$ or $\vec{y}$. Thus they may be regarded as properties of the matrix **A**. The elements s_{jj} are known as the *eigenvalues* (also as the proper values or characteristic values) of **A**, while the columns that make up **D** are called the *eigenvectors* (or proper vectors or characteristic vectors) of **A**. In addition, equation (2.4.10) is known as the *eigen* (or characteristic) *equation* of the matrix **A**. It is not obvious that a similarity transformation exists for all matrices and indeed, in general they do not. However, should the matrix be symmetric, then such a transformation is guaranteed to exist. Equation (2.4.10) suggests the manner by which we can find the eigenvalues of a matrix. The expansion of equation (2.4.10) by minors as in equation (1.2.10), or more generally in equation (2.2.5), makes it clear that the resulting expression will be a polynomial of degree n in s_{jj} which will have n roots which are the eigenvalues. Thus one approach to finding the eigenvalues of a matrix is equivalent to finding the roots of the eigen-equation (2.4.9). We shall say more about finding the roots of a polynomial in the next chapter so for the moment we will restrict ourselves to some special techniques for finding the eigenvalues and eigenvectors of a matrix.

We saw in section (2.2c) that diagonalization of a matrix will not change the value of its determinant. Since the application of the transformation matrix **D** and its inverse effectively accomplishes a diagonalization of **A** to the matrix **S** we should expect the determinant to remain unchanged. Since the determinant of **S** will just be the product of the diagonal elements we can write

$$\mathrm{Det}\left|\mathbf{A}\right| = \prod_i s_{ii}\ . \tag{2.4.11}$$

The trace of a matrix is also invariant to a similarity transformation so

$$\mathrm{Tr}\left|\mathbf{A}\right| = \sum_i s_{ii}\ . \tag{2.4.12}$$

These two constraints are always enough to enable one to find the eigenvalues of a 2□2 matrix and may be used to reduce the eigen-equation by two in its degree. However, for the more interesting case where n is large, we shall have to find a more general method. Since any such method will be equivalent to finding the roots of a polynomial, we may expect such methods to be fairly complicated as finding the roots of polynomials is one of the trickiest problems in numerical analysis. So it is with finding the eigenvalues of a matrix.

While we noted that the transformation that gives rise to **S** is a similarity transformation [equation (2.4.6)], not all similarity transformations need diagonalize a matrix, but simply have the form

$$\mathbf{B}^{-1}\mathbf{A}\mathbf{B} = \mathbf{Q}\ . \tag{2.4.13}$$

The invariance of the eigenvalues to similarity transformations provide the basis for the general strategy employed by most "canned" eigenvalue programs. The basic idea is to force the matrix **A** toward diagonal form by employing a series of similarity transformations. The details of such procedures are well beyond the scope of this book but can be found in the references suggested at the end of this chapter[3, 4]. However, whatever approach is selected, the prudent investigator will see how well the constraints given by equations (2.4.11, 12) are met before being satisfied that the "canned" package has actually found the correct eigenvalues of the matrix.

Having found the eigenvalues, the corresponding eigenvectors can be found by appealing to equation (2.4.9). However, these equations are still homogeneous, implying that the elements of the eigenvectors are not uniquely determined. Indeed, it is the magnitude of the eigenvector that is usually considered to be unspecified so that all that is missing is a scale factor to be applied to each eigenvector. A common approach is to simply define one of the elements of the eigenvector to be unity thereby making the system of equations (2.4.9) nonhomogeneous and of the form

$$\sum_{k=2}^{n}(a_{ik} - \delta_{ik}s_{jj})d_{jk} / d_{j1} = -a_{i1} \quad . \tag{2.4.14}$$

In this form the elements of the eigenvector will be found relative to the element d_{1j}.

Let us conclude our discussion of eigenvalues and eigen-vectors by again considering the matrix of the equations (2.2.13) used to illustrate the direct solution schemes. We have already seen from equation (2.3.39) that these equations failed the sufficient conditions for the existence of Gauss-Seidel iterative solution. By evaluating the eigenvalues for the matrix we can evaluate the *necessary and sufficient* conditions for convergence, namely that the eigenvalues all be positive and less than unity.

The matrix for equations (2.2.13) is

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{pmatrix} \quad , \tag{2.4.14}$$

so that the eigen-equation delineated by equation (2.4.10) becomes

$$\text{Det }\mathbf{A} = \mathbf{Det}\begin{vmatrix} (1\text{-s}) & 2 & 3 \\ 3 & (2-s) & 1 \\ 2 & 1 & (3-s) \end{vmatrix} = -s^3 + 6s^2 + 2s - 12 = 0 \quad . \tag{2.4.15}$$

The cubic polynomial that results has three roots which are the eigenvalues of the matrix. However before solving for the eigenvalues we can evaluate the constraints given by equations (2.4.11) and (2.4.12) and get

$$\left.\begin{array}{l} \text{Det}\,|\mathbf{A}| = \prod_i s_{ii} = -12 \\ \text{Tr}|\mathbf{A}| = \sum_i s_{ii} = +6 \end{array}\right\} \quad . \tag{2.4.16}$$

The determinant tells us that the eigenvalues cannot all be positive so that the *necessary and sufficient* conditions for the convergence of Gauss-Seidel are not fulfilled confirming the result of sufficient condition given by equation (2.3.39). The constraints given by equation (2.4.26) can also aid us in finding roots for the eigen-equation (2.4.15). The fact that the product of the roots is the negative of twice their sum suggests that two of the roots occur as a pair with opposite sign. This conjecture is supported by Descarte's "rule of signs" discussed in the next chapter (section 3.1a). With that knowledge coupled with the values for the trace and determinant we find that the roots are

$$s_i = \begin{pmatrix} 6 \\ +\sqrt{2} \\ -\sqrt{2} \end{pmatrix} \quad . \tag{2.4.17}$$

Thus, not only does one of the eigenvalues violate the necessary and sufficient convergence criteria by being negative, they all do as they all have a magnitude greater than unity.

We may complete the study of this matrix by finding the eigen-vectors with the aid of equation (2.4.9) so that

$$\begin{pmatrix} (1\text{-s}) & 2 & 3 \\ 3 & (2-s) & 1 \\ 2 & 1 & (3-s) \end{pmatrix} \begin{pmatrix} d_{i1} \\ d_{i2} \\ d_{i3} \end{pmatrix} = 0 \quad . \tag{2.4.18}$$

As we noted earlier, these equations are homogeneous so that they have no unique solution. This means that the length of the eigen-vectors is indeterminant. Many authors normalize them so that they are of unit length thereby constituting a set of unit basis vectors for further analysis. However, we shall simply take one component d_{11} to be unity thereby reducing the 3□3 system of homogeneous equations (2.4.18) to a 2□2 system of inhomogeneous equations,

$$\left.\begin{array}{l} (2-s_i)d_{i2} + d_{i3} = -3 \\ d_{i2} + (3-s_i)d_{13} = -2 \end{array}\right\} , \tag{2.4.19}$$

which have a unique solution for the remaining elements of the eigen-vectors. For our example the solution is

$$\left.\begin{array}{ll} s_1 = +6 : & \vec{D}_1 = [\,1.0, 1.0, 1.0\,] \\ s_2 = +\sqrt{2} : & \vec{D}_2 = [\,1.0, -(7+3\sqrt{2})/(7-5\sqrt{2}), +(2\sqrt{2}-1)/(7-5\sqrt{2}) \\ s_3 = -\sqrt{2} : & \vec{D}_3 = [\,1.0, -(7+3\sqrt{2})/(7+5\sqrt{2}), -(2\sqrt{2}+1)/(7+5\sqrt{2}) \end{array}\right\} \quad . \tag{2.4.20}$$

Should one wish to re-normalize these vectors to be unit vectors, one need only divide each element by the magnitude of the vectors. Each eigenvalue has its own associated eigen-vector so that equation (2.4.20) completes the analysis of the matrix **A**.

We introduced the notion of an eigenvalue initially to provide a necessary and sufficient condition for the convergence of the Gauss-Seidel iteration method for a system of linear equations. Clearly, this is an excellent example of the case where the error or convergence criteria pose a more difficult problem than the original problem. There is far more to the detailed determination of the eigenvalues of a matrix than merely the inversion of a matrix. All the different classes of matrices described in section 1.2 pose special problems even in the case where distinct eigenvalues exist. The solution of the eigen-equation (2.4.10) involves finding the roots of polynomials. We shall see in the next chapter that this is a tricky problem indeed.

Chapter 2 Exercises

1. Find the inverse, eigenvalues, and eigenvectors for

   $a_{ij} = (i+j-1)^{-1}$ for i5, j5 .

   Describe the accuracy of your answer and how you know.

2. Solve the following set of equations both by means of a direct method and iterative method. Describe the methods used and why you chose them.

$$\begin{aligned}
X_2 + 5X_3 - 7X_4 + 23X_5 - X_6 + 7X_7 + 8X_8 + X_9 - 5X_{10} &= 10 \\
17X_1 - 24X_3 - 75X_4 + 100X_5 - 18X_6 + 10X_7 - 8X_8 + 9X_9 - 50X_{10} &= -40 \\
3X_1 - 2X_2 + 15X_3 - 78X_5 - 90X_6 - 70X_7 + 18X_8 - 75X_9 + X_{10} &= -17 \\
5X_1 + 5X_2 - 10X_3 - 72X_5 - X_6 + 80X_7 - 3X_8 + 10X_9 - 18X_{10} &= 43 \\
100X_1 - 4X_2 - 75X_3 - 8X_4 + 83X_6 - 10X_7 - 75X_8 + 3X_9 - 8X_{10} &= -53 \\
70X_1 + 85X_2 - 4X_3 - 9X_4 + 2X_5 + 3X_7 - 17X_8 - X_9 - 21X_{10} &= 12 \\
X_1 + 15X_2 + 100X_3 - 4X_4 - 23X_5 + 13X_6 + 7X_8 - 3X_9 + 17X_{10} &= -60 \\
16X_1 + 2X_2 - 7X_3 + 89X_4 - 17X_5 + 11X_6 - 73X_7 - 8X_9 - 23X_{10} &= 100 \\
51X_1 + 47X_2 - 3X_3 + 5X_4 - 10X_5 + 18X_6 - 99X_7 - 18X_8 + 12X_{10} &= 0 \\
X_1 + X_2 + X_3 + X_4 + X_5 + X_6 + X_7 + X_8 + X_9 &= 100
\end{aligned}$$

3. Solve the equations $\mathbf{A}\vec{x} = \vec{c}$ where $a_{ij} = (i+j-1)^{-1}$, and $c_i = i$ for i5, and j5. Use both Gauss-Jordan and Gauss-Seidel methods and comment on which gives the better answer.

4. Solve the following system of equations by Gauss-Jordan and Gauss-Seidel iteration starting with an initial guess of X=Y=Z=1.

$$\begin{aligned}
8X + 3Y + 2Z &= 20.00 \\
16X + 6Y + 4.001Z &= 40.02 \\
4X + 1.501Y + Z &= 10.01 \ .
\end{aligned}$$

   Comment on the accuracy of your solution and the relative efficiency of the two methods.

5. Show that if **A** is an orthonormal matrix, the $\mathbf{A}^{-1} = \mathbf{A}^{T}$.

6. If $\vec{x} = \mathbf{A}\vec{x}'$ where

$$\mathbf{A} = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 1 & 1 \end{pmatrix},$$

   find the components of $\vec{x}$ ' in terms of the components of $\vec{x}$ for $\varphi = \pi/6$.

Chapter 2 References and Supplemental Reading

A reasonable complete description of the Crout factorization method is given by

1. Hildebrand, F.B., "Introduction to Numerical Analysis" (1956) McGraw-Hill Book Co., Inc., New York, Toronto, London.

A very nice introduction to fixed-point iteration theory is given by

2. Moursund, D.G., and Duris, C.S., "Elementary Theory and Applications of Numerical Analysis" (1988) Dover Publications, Inc. New York.

The next two references provide an excellent introduction to the determination of eigenvalues and eigenvectors. Householder's discussion is highly theoretical, but provides the underpinnings for contemporary methods. The work titled "Numerical Recipes" is just that with some description on how the recipes work. It represents probably the most complete and useful compilation of contemporary numerical algorithms currently available.

3. Householder, A.S., "Principles of Numerical Analysis" (1953) McGraw-Hill Book Co., Inc., New York, Toronto, London, pp.143-184.

4. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., "Numerical Recipes The Art of Scientific Computing" (1986), Cambridge University Press, Cambridge, New York, Melbourne, pp. 335-380.

Richard Hamming's most recent numerical analysis provides a good introduction to the methods for handling error analysis, while reference 6 is an excellent example of the type of effort one may find in the Russian literature on numerical methods. Their approach tends to be fundamentally different than the typical western approach and is often superior as they rely on analysis to a far greater degree than is common in the west.

5. Hamming, R.W., "Introduction to Applied Numerical Analysis" (1971) McGraw-Hill Book Co., Inc., New York, San Francisco, Toronto, London.

6. Faddeeva, V.N., "Computational Methods of Linear Algebra",(1959), Trans. C.D. Benster, Dover Publications, Inc. New York.