# 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 $9\times10^{0}$. However, 6+4, 6+5, and 6+8 will all yield the same answer namely $1\times10^{1}$. 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 $2\times10^{1}$, then the machine is said to round off the calculation to the nearest significant digit. However, if the result remains $1\times10^{1}$, 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

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

which in vector notation is

$$\mathbf{A}\vec{x} = \vec{c} \ . \tag{2.2.2}$$

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

$$\vec{x} = \mathbf{A}^{-1}\vec{c} \ , \tag{2.2.3}$$

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.

### *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 $\delta_{ij}$ 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:

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

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

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

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

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

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

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

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

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

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

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

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

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

and subtract from the remaining rows to yield

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

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

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

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

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

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

$$N \geq \log_{10} \left| M/d \right| \quad , \tag{2.2.23}$$

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

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

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

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

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

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

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

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

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

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

where

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

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

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

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

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

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

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

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

$$\left.\begin{aligned} u_{11} &= a_{11} - 0 = 1 \\ v_{11} &= (a_{11} - 0)/u_{11} = 1 \\ v_{12} &= (a_{12} - 0)/u_{11} = 3 \\ v_{13} &= (a_{13} - 0)/u_{11} = 2 \end{aligned}\right\} \quad j = 1$$

$$\left.\begin{aligned} u_{12} &= a_{12} - 0 = 2 \\ u_{22} &= [a_{22} - (v_{21}u_{21})] = 2 - (3 \times 2) = 4 \\ v_{22} &= [a_{22} - (v_{21}u_{12})]/u_{22} = [2 - (3 \times 2)]/-4 = 1 \\ v_{32} &= [a_{32} - (v_{31}u_{13})]/u_{22} = [1 - (2 \times 2)]/-4 = \frac{3}{4} \end{aligned}\right\} \quad j = 2$$

$$\left.\begin{aligned} u_{13} &= a_{13} - 0 = 3 \\ u_{23} &= a_{23} - (v_{21}u_{13}) = 1 - (3 \times 3) = -8 \\ u_{33} &= a_{33} - (v_{31}u_{13} + v_{32}u_{23}) = 3 - [(2 \times 3) + (\tfrac{3}{4} \times -8)] = 3 \\ v_{33} &= [a_{33} - (v_{31}u_{13} + v_{32}u_{23})]/u_{33} = [3 - (2 \times 3) - (-8 \times \tfrac{3}{4})]/3 = 1 \end{aligned}\right\} \quad j = 3 \qquad . \qquad (2.2.35)$$

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

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

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

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

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

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

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.