which (peeling of the $\mathbf{C}^{-1}$'s one at a time) implies a solution

$$\mathbf{x} = \mathbf{C}_1 \cdot \mathbf{C}_2 \cdot \mathbf{C}_3 \cdots \mathbf{b} \tag{2.1.8}$$

Notice the essential difference between equation (2.1.8) and equation (2.1.6). In the latter case, the **C**'s must be applied to **b** in the *reverse order* from that in which they become known. That is, they must all be stored along the way. This requirement greatly reduces the usefulness of column operations, generally restricting them to simple permutations, for example in support of full pivoting.

CITED REFERENCES AND FURTHER READING:

Wilkinson, J.H. 1965, *The Algebraic Eigenvalue Problem* (New York: Oxford University Press). [1]

Carnahan, B., Luther, H.A., and Wilkes, J.O. 1969, *Applied Numerical Methods* (New York: Wiley), Example 5.2, p. 282.

Bevington, P.R. 1969, *Data Reduction and Error Analysis for the Physical Sciences* (New York: McGraw-Hill), Program B-2, p. 298.

Westlake, J.R. 1968, *A Handbook of Numerical Matrix Inversion and Solution of Linear Equations* (New York: Wiley).

Ralston, A., and Rabinowitz, P. 1978, *A First Course in Numerical Analysis*, 2nd ed. (New York: McGraw-Hill), §9.3–1.

2.2 Gaussian Elimination with Backsubstitution

The usefulness of Gaussian elimination with backsubstitution is primarily pedagogical. It stands between full elimination schemes such as Gauss-Jordan, and triangular decomposition schemes such as will be discussed in the next section. Gaussian elimination reduces a matrix not all the way to the identity matrix, but only halfway, to a matrix whose components on the diagonal and above (say) remain nontrivial. Let us now see what advantages accrue.

Suppose that in doing Gauss-Jordan elimination, as described in §2.1, we at each stage subtract away rows only *below* the then-current pivot element. When a_{22} is the pivot element, for example, we divide the second row by its value (as before), but now use the pivot row to zero only a_{32} and a_{42}, not a_{12} (see equation 2.1.1). Suppose, also, that we do only partial pivoting, never interchanging columns, so that the order of the unknowns never needs to be modified.

Then, when we have done this for all the pivots, we will be left with a reduced equation that looks like this (in the case of a single right-hand side vector):

$$\begin{bmatrix} a'_{11} & a'_{12} & a'_{13} & a'_{14} \\ 0 & a'_{22} & a'_{23} & a'_{24} \\ 0 & 0 & a'_{33} & a'_{34} \\ 0 & 0 & 0 & a'_{44} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{bmatrix} \tag{2.2.1}$$

Here the primes signify that the a's and b's do not have their original numerical values, but have been modified by all the row operations in the elimination to this point. The procedure up to this point is termed *Gaussian elimination*.

Backsubstitution

But how do we solve for the x's? The last x (x_4 in this example) is already isolated, namely

$$x_4 = b'_4/a'_{44} \tag{2.2.2}$$

With the last x known we can move to the penultimate x,

$$x_3 = \frac{1}{a'_{33}}[b'_3 - x_4 a'_{34}] \tag{2.2.3}$$

and then proceed with the x before that one. The typical step is

$$x_i = \frac{1}{a'_{ii}}\left[b'_i - \sum_{j=i+1}^{N} a'_{ij}x_j\right] \tag{2.2.4}$$

The procedure defined by equation (2.2.4) is called *backsubstitution*. The combination of Gaussian elimination and backsubstitution yields a solution to the set of equations.

The advantage of Gaussian elimination and backsubstitution over Gauss-Jordan elimination is simply that the former is faster in raw operations count: The innermost loops of Gauss-Jordan elimination, each containing one subtraction and one multiplication, are executed N^3 and N^2M times (where there are N equations and M unknowns). The corresponding loops in Gaussian elimination are executed only $\frac{1}{3}N^3$ times (only half the matrix is reduced, and the increasing numbers of predictable zeros reduce the count to one-third), and $\frac{1}{2}N^2M$ times, respectively. Each backsubstitution of a right-hand side is $\frac{1}{2}N^2$ executions of a similar loop (one multiplication plus one subtraction). For $M \ll N$ (only a few right-hand sides) Gaussian elimination thus has about a factor three advantage over Gauss-Jordan. (We could reduce this advantage to a factor 1.5 by *not* computing the inverse matrix as part of the Gauss-Jordan scheme.)

For computing the inverse matrix (which we can view as the case of $M = N$ right-hand sides, namely the N unit vectors which are the columns of the identity matrix), Gaussian elimination and backsubstitution at first glance require $\frac{1}{3}N^3$ (matrix reduction) $+\frac{1}{2}N^3$ (right-hand side manipulations) $+\frac{1}{2}N^3$ (N backsubstitutions) $= \frac{4}{3}N^3$ loop executions, which is more than the N^3 for Gauss-Jordan. However, the unit vectors are quite special in containing all zeros except for one element. If this is taken into account, the right-side manipulations can be reduced to only $\frac{1}{6}N^3$ loop executions, and, for matrix inversion, the two methods have identical efficiencies.

Both Gaussian elimination and Gauss-Jordan elimination share the disadvantage that all right-hand sides must be known in advance. The LU decomposition method in the next section does not share that deficiency, and also has an equally small operations count, both for solution with any number of right-hand sides, and for matrix inversion. For this reason we will not implement the method of Gaussian elimination as a routine.

CITED REFERENCES AND FURTHER READING:

Ralston, A., and Rabinowitz, P. 1978, *A First Course in Numerical Analysis*, 2nd ed. (New York: McGraw-Hill), §9.3–1.

Isaacson, E., and Keller, H.B. 1966, *Analysis of Numerical Methods* (New York: Wiley), §2.1.

Johnson, L.W., and Riess, R.D. 1982, *Numerical Analysis*, 2nd ed. (Reading, MA: Addison-Wesley), §2.2.1.

Westlake, J.R. 1968, *A Handbook of Numerical Matrix Inversion and Solution of Linear Equations* (New York: Wiley).

2.3 LU Decomposition and Its Applications

Suppose we are able to write the matrix **A** as a product of two matrices,

$$\mathbf{L}\cdot\mathbf{U}=\mathbf{A} \tag{2.3.1}$$

where **L** is *lower triangular* (has elements only on the diagonal and below) and **U** is *upper triangular* (has elements only on the diagonal and above). For the case of a 4×4 matrix **A**, for example, equation (2.3.1) would look like this:

$$\begin{bmatrix}\alpha_{11} & 0 & 0 & 0\\ \alpha_{21} & \alpha_{22} & 0 & 0\\ \alpha_{31} & \alpha_{32} & \alpha_{33} & 0\\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44}\end{bmatrix}\cdot\begin{bmatrix}\beta_{11} & \beta_{12} & \beta_{13} & \beta_{14}\\ 0 & \beta_{22} & \beta_{23} & \beta_{24}\\ 0 & 0 & \beta_{33} & \beta_{34}\\ 0 & 0 & 0 & \beta_{44}\end{bmatrix}=\begin{bmatrix}a_{11} & a_{12} & a_{13} & a_{14}\\ a_{21} & a_{22} & a_{23} & a_{24}\\ a_{31} & a_{32} & a_{33} & a_{34}\\ a_{41} & a_{42} & a_{43} & a_{44}\end{bmatrix} \tag{2.3.2}$$

We can use a decomposition such as (2.3.1) to solve the linear set

$$\mathbf{A}\cdot\mathbf{x}=(\mathbf{L}\cdot\mathbf{U})\cdot\mathbf{x}=\mathbf{L}\cdot(\mathbf{U}\cdot\mathbf{x})=\mathbf{b} \tag{2.3.3}$$

by first solving for the vector **y** such that

$$\mathbf{L}\cdot\mathbf{y}=\mathbf{b} \tag{2.3.4}$$

and then solving

$$\mathbf{U}\cdot\mathbf{x}=\mathbf{y} \tag{2.3.5}$$

What is the advantage of breaking up one linear set into two successive ones? The advantage is that the solution of a triangular set of equations is quite trivial, as we have already seen in §2.2 (equation 2.2.4). Thus, equation (2.3.4) can be solved by *forward substitution* as follows,

$$\begin{aligned} y_1 &= \frac{b_1}{\alpha_{11}}\\ y_i &= \frac{1}{\alpha_{ii}}\left[b_i-\sum_{j=1}^{i-1}\alpha_{ij}y_j\right] \qquad i=2,3,\ldots,N \end{aligned} \tag{2.3.6}$$

while (2.3.5) can then be solved by *backsubstitution* exactly as in equations (2.2.2)–(2.2.4),

$$\begin{aligned} x_N &= \frac{y_N}{\beta_{NN}}\\ x_i &= \frac{1}{\beta_{ii}}\left[y_i-\sum_{j=i+1}^{N}\beta_{ij}x_j\right] \qquad i=N-1,N-2,\ldots,1 \end{aligned} \tag{2.3.7}$$