Doolittle Decomposition of a Matrix

It is always possible to factor a square matrix into a lower triangular matrix and an upper triangular matrix. That is,

$$[A] = [L][U]$$

EXAMPLE:

$$\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right] = \left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & -1 & 1 \end{array}\right] \left[\begin{array}{c|c|c} 2 & -1 & -1 \\ \hline & 4 & -1 \\ \hline & & 3 \end{array}\right]$$

Having the LU factors of a matrix is equivalent of having its inverse as far as ease in determining the solution of a set of simultaneous equations. Consider the following matrix equation:

$$[A]\{x\} = \{f\}$$

If the factors of $[A]$ are known, then

$$[L][U]\{x\} = \{F\}$$

Let (define)

$$\{y\} = [U]\{x\}$$

then

$$[L]\{y\} = \{F\}$$

Because $[L]$ is a triangular matrix, this equation can be used to solve for $\{y\}$ by a simple procedure of forward-substitution (the reverse of back-substitution after a Gauss elimination). Once $\{y\}$ is found then,

$$[U]\{x\} = \{y\}$$

is used to solve for $\{x\}$ by back-substitution.

EXAMPLE:

Let

$$\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right] \left\{\begin{array}{c} x1 \\ \hline x2 \\ \hline x3 \end{array}\right\} = \left\{\begin{array}{c} -1 \\ \hline 13 \\ \hline -6 \end{array}\right\}$$

then

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & -1 & 1 \end{array}\right] \left\{\begin{array}{c} y1 \\ \hline y2 \\ \hline y3 \end{array}\right\} = \left\{\begin{array}{c} -1 \\ \hline 13 \\ \hline -6 \end{array}\right\}$$

From this, it is clear that

$$\begin{aligned} y1 &= -1 \\ y2 = 13 + 2(y1) &= 11 \\ y3 = -6 + 2(y1) + 1(y2) &= 3 \end{aligned}$$

Therefore

$$\left[\begin{array}{c|c|c} 2 & -1 & -1 \\ \hline & 4 & -1 \\ \hline & & 3 \end{array}\right] \left\{\begin{array}{c} x1 \\ \hline x2 \\ \hline x3 \end{array}\right\} = \left\{\begin{array}{c} -1 \\ \hline 11 \\ \hline 3 \end{array}\right\}$$

from which, by back-substitution, we obtain

$$\begin{aligned} x3 = (3)/3 &= 1 \\ x2 = (11 + 1(y3))/4 &= 3 \\ x1 = (-1 + 1(y2) + 2(y3))/2 &= 2 \end{aligned}$$

Thus we have solved our set of equations by two simple matrix multiplications. For such a small problem the savings over a conventional Gauss elimination may not appear great; however, for large problems it is. As an example, for a set of N equations with a bandwidth of B, the number of numerical operations is approximately $NB^2 + NB$ whereas for a solution found using the LU factors, the number of numerical operations is approximately $2NB$.

The question is, of course, how does one obtain the factors. There are several ways - one being Gauss elimination itself. In fact, we find it is as much work to obtain the factors as doing a Gauss elimination. What, therefore, is the advantage? The advantage comes from needing to solve a set of equations for many different right hand sides. Particularly when each right hand side is not known until after the solution corresponding to the previous right hand side has been obtained. Such is the case, for example, during a transient or a nonlinear analyses by the finite element method.

Among the various methods to arrive at the factors, the Doolittle algorithm is particularly nice for two reasons. The first being that once understood, it is easily programmed for even fairly complex storage arrangements for sparse matrices. The other reason is that the algorithm itself is a proof that the factorization is unique.

The algorithm begins with the assumption that such a factorization does exist. Once that is assumed, then it becomes clear that for each of the terms in the two factors, there is one, and only one, linear equation for its value. The easiest way to explain the method is by example, hence:

EXAMPLE:

Begin with

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline \cdot & 1 & \\ \hline \cdot & \cdot & 1 \end{array}\right]\left[\begin{array}{c|c|c} \cdot & \cdot & \cdot \\ \hline & \cdot & \cdot \\ \hline & & \cdot \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

where the dots represent yet-to-be-determined entries. Note all diagonal terms for $[L]$ have been assigned a value of unity whereas the diagonal terms of $[U]$ are to be determined. Each value is now determined as follows:

Dot Row 1 of L with Columns 1-3 of U to obtain

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline \cdot & 1 & \\ \hline \cdot & \cdot & 1 \end{array}\right]\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline & \cdot & \cdot \\ \hline & & \cdot \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

Dot Rows 2-3 of L with Column 1 of U to obtain:

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & \cdot & 1 \end{array}\right]\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline & \cdot & \cdot \\ \hline & & \cdot \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

Dot Row 2 of L with Columns 2-3 of U to obtain:

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & \cdot & 1 \end{array}\right]\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline & 4 & -1 \\ \hline & & \cdot \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

Dot Row 3 of L with Column 2 of U to obtain:

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & -1 & 1 \end{array}\right]\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline & 4 & -1 \\ \hline & & \cdot \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

Dot Row 3 of L with Column 3 of U to obtain:

$$\left[\begin{array}{c|c|c} 1 & & \\ \hline -2 & 1 & \\ \hline -2 & -1 & 1 \end{array}\right]\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline & 4 & -1 \\ \hline & & 3 \end{array}\right]=\left[\begin{array}{c|c|c} 2 & -1 & -2 \\ \hline -4 & 6 & 3 \\ \hline -4 & -2 & 8 \end{array}\right]$$

The general algorithm is:

```
for i = 1:n
        for j = i:n
                L_ik U_kj = A_ij gives row i of U
        end
        for j = i+1:n
                L_jk U_ki = A_ji gives column i of L
        end
end
```

An important characteristic of the algorithm is that as each entry into either $[L]$ or $[U]$ is made, the corresponding entry in $[A]$ is no longer needed for the remainder of the algorithm. Hence, the space used for $[A]$ can be used to store both the $[L]$ and $[U]$ matrices. For our example, if this were done, the $[A]$ matrix would end up as:

$$[A] = \left[\begin{array}{r|r|r} 2 & -1 & -2 \\ \hline -2 & 4 & -1 \\ \hline -2 & -1 & 3 \end{array}\right]$$

On the next pages, two FORTRAN versions of the code are given. The first uses separate storage areas for the factors, and the second version uses the original matrix for the storage of the factors.

EGT: 2/5/98

```
      SUBROUTINE DL(AL,AU,A,Neq)
C     ==========================
      dimension AL(Neq,Neq),AU(Neq,Neq),A(Neq,Neq)
      do i=1,neq
         do j=i,neq
            au(i,j)=a(i,j)
            do k=1,i-1
               au(i,j)=au(i,j)-al(i,k)*au(k,j)
            enddo
         enddo
         do j=i+1,neq
            al(j,i)=a(j,i)
            do k=1,i-1
               al(j,i)=al(j,i)-al(j,k)*au(k,i)
            enddo
            al(j,i)=al(j,i)/au(i,i)
         enddo
      enddo
      return
      end


      SUBROUTINE DL(A,Neq)
C     ====================
      dimension A(Neq,Neq)
      do i=1,neq
         do j=i,neq
            do k=1,i-1
               a(i,j)=a(i,j)-a(i,k)*a(k,j)
            enddo
         enddo
         do j=i+1,neq
            do k=1,i-1
               a(j,i)=a(j,i)-a(j,k)*a(k,i)
            enddo
            a(j,i)=a(j,i)/a(i,i)
         enddo
      enddo
      return
```

```
end
```