2.9 Cholesky Decomposition

If a square matrix $A$ happens to be symmetric and positive definite, then it has a special, more efficient, triangular decomposition. Symmetric means that $a_{ij} = a_{ji}$ for $i, j = 1, \ldots, N$, while positive definite means that

$$v \cdot A \cdot v > 0$$

for all vectors $v$. (In Chapter 11 we will see that positive definite has the equivalent interpretation that $A$ has all positive eigenvalues.) While symmetric, positive definite matrices are rather special, they occur quite frequently in some applications, so their special factorization, called Cholesky decomposition, is good to know about. When you can use it, Cholesky decomposition is about a factor of two better than alternative methods for solving linear equations.

Instead of seeking arbitrary lower and upper triangular factors $L$ and $U$, Cholesky decomposition constructs a lower triangular matrix $L$ whose transpose $L^T$ can itself serve as the upper triangular part. In other words we replace equation (2.3.1) by

$$L \cdot L^T = A \quad (2.9.2)$$

This factorization is sometimes referred to as “taking the square root” of the matrix $A$. The components of $L^T$ are of course related to those of $L$ by

$$L_{ij}^T = L_{ji} \quad (2.9.3)$$

Writing out equation (2.9.2) in components, one readily obtains the analogs of equations (2.3.12)–(2.3.13),

$$L_{ii} = \left( a_{ii} - \sum_{k=1}^{i-1} L_{ik}^2 \right)^{1/2} \quad (2.9.4)$$

and

$$L_{ji} = \frac{1}{L_{ii}} \left( a_{ij} - \sum_{k=1}^{i-1} L_{ik} L_{jk} \right) \quad j = i + 1, i + 2, \ldots, N \quad (2.9.5)$$
If you apply equations (2.9.4) and (2.9.5) in the order \( i = 1, 2, \ldots, N \), you will see that the \( L \)'s that occur on the right-hand side are already determined by the time they are needed. Also, only components \( a_{ij} \) with \( j \geq i \) are referenced. (Since \( A \) is symmetric, these have complete information.) It is convenient, then, to have the factor \( L \) overwrite the subdiagonal (lower triangular but not including the diagonal) part of \( A \), preserving the input upper triangular values of \( A \). Only one extra vector of length \( N \) is needed to store the diagonal part of \( L \). The operations count is \( N^3/6 \) executions of the inner loop (consisting of one multiply and one subtract), with also \( N \) square roots. As already mentioned, this is about a factor 2 better than \( LU \) decomposition of \( A \) (where its symmetry would be ignored).

A straightforward implementation is

```fortran
SUBROUTINE choldc(a,n,np,p)
INTEGER n,np
REAL a(np,np),p(n)
Given a positive-definite symmetric matrix \( a(1:n,1:n) \), with physical dimension \( np \), this routine constructs its Cholesky decomposition, \( A = L \cdot L^T \). On input, only the upper triangle of \( A \) need be given; it is not modified. The Cholesky factor \( L \) is returned in the lower triangle of \( A \), except for its diagonal elements which are returned in \( p(1:n) \).

INTEGER i,j,k
REAL sum
DO 11 i=1,n
   DO 12 j=i,n
      sum=a(i,j)
      DO 11 k=i-1,1,-1
         sum=sum-a(i,k)*a(j,k)
      END DO 11
      IF(i.eq.j) THEN
         IF(sum.LE.0.) THEN
            PAUSE 'choldc failed'
            a, with rounding errors, is not positive definite.
            p(i)=SQRT(sum)
         ELSE
            a(j,i)=sum/p(i)
         ENDIF
      END DO 12
   END DO i
RETURN
END
```

You might at this point wonder about pivoting. The pleasant answer is that Cholesky decomposition is extremely stable numerically, without any pivoting at all. Failure of `choldc` simply indicates that the matrix \( A \) (or, with roundoff error, another very nearby matrix) is not positive definite. In fact, `choldc` is an efficient way to test whether a symmetric matrix is positive definite. (In this application, you will want to replace the `pause` with some less drastic signaling method.)

Once your matrix is decomposed, the triangular factor can be used to solve a linear equation by backsubstitution. The straightforward implementation of this is

```fortran
SUBROUTINE cholsl(a,n,np,p,b,x)
INTEGER n,np
REAL a(np,np),b(n),p(n),x(n)
Solves the set of \( n \) linear equations \( A \cdot x = b \), where \( A \) is a positive-definite symmetric matrix with physical dimension \( np \). \( a \) and \( p \) are input as the output of the routine `choldc`. Only the lower triangle of \( a \) is accessed. \( b(1:n) \) is input as the right-hand side vector. The solution vector is returned in \( x(1:n) \). \( a, \) \( np, \) and \( p \) are not modified and can be left in place for successive calls with different right-hand sides \( b \). \( b \) is not modified unless you identify \( b \) and \( x \) in the calling sequence, which is allowed.

INTEGER i,k
REAL sum
DO 12 i=1,n
   DO 12 k=i-1,1,-1
      sum=sum-a(i,k)*x(k)
   END DO 12
   x(i)=b(i)-sum
END DO i
RETURN
END
```
2.10 QR Decomposition

There is another matrix factorization that is sometimes very useful, the so-called QR decomposition,

\[ A = Q \cdot R \]  \hspace{1cm} (2.10.1)

Here \( R \) is upper triangular, while \( Q \) is orthogonal, that is,

\[ Q^T \cdot Q = I \]  \hspace{1cm} (2.10.2)

where \( Q^T \) is the transpose matrix of \( Q \). Although the decomposition exists for a general rectangular matrix, we shall restrict our treatment to the case when all the matrices are square, with dimensions \( N \times N \).

A typical use of \texttt{choldc} and \texttt{cholsl} is in the inversion of covariance matrices describing the fit of data to a model; see, e.g., §15.6. In this, and many other applications, one often needs \( L^{-1} \). The lower triangle of this matrix can be efficiently found from the output of \texttt{choldc}:

\begin{verbatim}
  do i=1,n
    a(i,i)=1./p(i)
  enddo
  do j=i+1,n
    sum=0.
    do k=i,j-1
      sum=sum-a(j,k)*a(k,i)
    enddo
    a(j,i)=sum/p(j)
  enddo
  enddo
\end{verbatim}

CITED REFERENCES AND FURTHER READING: