```
      SUBROUTINE FM_GENEQ(F,A,B,K,X,Y,N)
      USE FMZM
      IMPLICIT NONE

!  Generate the KxK matrix A and Kx1 vector B of normal equations for the least square
!  fit of the K-parameter model

!     Y = C(1)*F(1,X) + ... + C(K)*F(K,X)

!  to the data points (X(J),Y(J)), J = 1, 2, ..., N.

!  A and B are returned, and then the coefficients C can be found by solving the
!  linear system  A * C = B.

!  Function L in the model evaluated at X is referenced by F(L,X) in this routine,
!  and F should be supplied as an external function subprogram by the user.

      INTEGER :: K, N
      TYPE (FM), EXTERNAL :: F
      TYPE (FM) :: A(K,K), B(K), X(N), Y(N)
      TYPE (FM), ALLOCATABLE :: FXI(:)
      INTEGER :: I, J, L
      TYPE (FM) :: XI, YI, FXIL

      CALL FM_ENTER_USER_ROUTINE
      IF (N <= 0 .OR. K <= 0) THEN
          WRITE (*,"(/' Error in FM_GENEQ.  K,N=',2I8/)") K,N
          STOP
      ENDIF

      ALLOCATE(FXI(K),STAT=J)
      IF (J /= 0) THEN
          WRITE (*,"(/' Error in FM_GENEQ.  Unable to allocate FXI with size ',I8/)") K
          STOP
      ENDIF

!             Initialize the upper triangle of A.

      DO I = 1, K
         DO J = I, K
            A(I,J) = 0
         ENDDO
         B(I) = 0
      ENDDO

!             Loop over the data points.

      DO I = 1, N
         XI = X(I)
         YI = Y(I)

!             Compute the K function values at X(I).

         DO J = 1, K
            FXI(J) = F(J,XI)
```

```
         ENDDO

!             Multiply the function values and add the products to the matrix.

         DO L = 1, K
            FXIL = FXI(L)
            DO J = L, K
               A(L,J) = A(L,J) + FXIL*FXI(J)
            ENDDO

!             Sum the right-hand-side term.

            B(L) = B(L) + YI*FXIL
         ENDDO
      ENDDO

!             Fill the lower triangle of the A matrix using symmetry.

      IF (K >= 2) THEN
          DO L = 2, K
             DO J = 1, L-1
                A(L,J) = A(J,L)
             ENDDO
          ENDDO
      ENDIF

!             The FM_DEALLOCATE call marks the FXI type(fm) index numbers as free in the fm memory
!             database, so they can be re-used later.  The DEALLOCATE statement doesn't do that,
!             it just frees the compiler-generated type(fm) objects.
!             To avoid leaking memory, it is a good idea to call FM_DEALLOCATE before doing a
!             DEALLOCATE of any type fm, zm, or im array.

      CALL FM_DEALLOCATE(FXI)
      DEALLOCATE(FXI)
      CALL FM_EXIT_USER_ROUTINE
      END SUBROUTINE FM_GENEQ
```