NAG Library

Advice on Replacement Calls for Withdrawn/Superseded Routines

The following list gives the names of routines that are suitable replacements for routines that have either been withdrawn or superseded since Mark 19.

The list indicates the minimum change necessary, but many of the replacement routines have additional flexibility and you may wish to take advantage of new features. It is strongly recommended that you consult the routine documents.

C05 – Roots of One or More Transcendental Equations

C05ADF

Withdrawn at Mark 25.
Replaced by C05AYF.

```
Old: FUNCTION F(XX)
        ...
     END FUNCTION F
     ...
     CALL C05ADF(A,B,EPS,ETA,F,X,IFAIL)
New: FUNCTION F(XX,IUSER,RUSER)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION F
     ...
     INTEGER            :: IUSER(1)
     REAL (KIND=nag_wp) :: RUSER(1)
     ...
     CALL C05AYF(A,B,EPS,ETA,F,X,IUSER,RUSER,IFAIL)
```

C05AGF

Withdrawn at Mark 25.
Replaced by C05AUF.

```
Old: FUNCTION F(XX)
        ...
     END FUNCTION F
     ...
     CALL C05AGF(X,H,EPS,ETA,F,A,B,IFAIL)
New: FUNCTION F(XX,IUSER,RUSER)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION F
     ...
     INTEGER            :: IUSER(1)
     REAL (KIND=nag_wp) :: RUSER(1)
     ...
     CALL C05AUF(X,H,EPS,ETA,F,A,B,IUSER,RUSER,IFAIL)
```

C05AJF

Withdrawn at Mark 25.
Replaced by C05AWF.

```
Old: FUNCTION F(XX)
        ...
     END FUNCTION F
     ...
     CALL C05AJF(X,EPS,ETA,F,NFMAX,IFAIL)
New: FUNCTION F(XX,IUSER,RUSER)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION F
     ...
     INTEGER            :: IUSER(1)
     REAL (KIND=nag_wp) :: RUSER(1)
     ...
     CALL C05AWF(X,EPS,ETA,F,NFMAX,IUSER,RUSER,IFAIL)
```

C05NBF

Withdrawn at Mark 25.
Replaced by C05QBF.

```
Old: SUBROUTINE FCN(N,X,FVEC,IFLAG)
        ...
     END SUBROUTINE FCN
     ...
     CALL C05NBF(FCN,N,X,FVEC,XTOL,WA,LWA,IFAIL)
New: SUBROUTINE FCN(N,X,FVEC,IUSER,RUSER,IFLAG)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION FCN
     ...
     INTEGER            :: IUSER(1)
     REAL (KIND=nag_wp) :: RUSER(1)
     ...
     CALL C05QBF(FCN,N,X,FVEC,XTOL,IUSER,RUSER,IFAIL)
```

C05NCF

Withdrawn at Mark 25.
Replaced by C05QCF.

```
Old: SUBROUTINE FCN(N,X,FVEC,IFLAG)
        ...
     END SUBROUTINE FCN
     ...
     REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05NCF(FCN,N,X,FVEC,XTOL,MAXFEV,ML,MU,EPSFCN,DIAG,MODE,FACTOR, &
                 NPRINT,NFEV,FJAC,LDFJAC,R,LR,QTF,W,IFAIL)
New: SUBROUTINE FCN(N,X,FVEC,IUSER,RUSER,IFLAG)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION FCN
     ...
     INTEGER            :: IUSER(1)
     REAL (KIND=nag_wp) :: FJAC(N,N), RUSER(1)
     ...
     CALL C05QCF(FCN,N,X,FVEC,XTOL,MAXFEV,ML,MU,EPSFCN,MODE,DIAG,FACTOR, &
                 NPRINT,NFEV,FJAC,R,QTF,IUSER,RUSER,IFAIL)
```

C05NDF

Withdrawn at Mark 25.
Replaced by C05QDF.

```
Old: REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05NDF(IREVCM,N,X,FVEC,XTOL,ML,MU,EPSFCN,DIAG,MODE,FACTOR, &
                 FJAC,LDFJAC,R,LR,QTF,W,IFAIL)
New: REAL (KIND=nag_wp) :: FJAC(N,N), RWSAV(4*N+20)
     INTEGER            :: IWSAV(17)
     ...
     CALL C05QDF(IREVCM,N,X,FVEC,XTOL,ML,MU,EPSFCN,MODE,DIAG,FACTOR, &
                 FJAC,R,QTF,IWSAV,RWSAV,IFAIL)
```

C05PBF/C05PBA

Withdrawn at Mark 25.
Replaced by C05RBF.

```
Old: SUBROUTINE FCN_C05PBF(N,X,FVEC,FJAC,LDFJAC,IFLAG)
        ...
     END SUBROUTINE FCN_C05PBF
     ...
     REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05PBF(FCN_C05PBF,N,X,FVEC,FJAC,LDFJAC,XTOL,WA,LWA,IFAIL)
        or
     SUBROUTINE FCN_C05PBA(N,X,FVEC,FJAC,LDFJAC,IFLAG,IUSER,RUSER)
        ...
     END SUBROUTINE FCN_C05PBA
     ...
     REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05PBA(FCN_C05PBA,N,X,FVEC,FJAC,LDFJAC,XTOL,WA,LWA,IUSER,RUSER,IFAIL)
New: SUBROUTINE FCN(N,X,FVEC,FJAC,IUSER,RUSER,IFLAG)
        ...
     END SUBROUTINE FCN
     ...
     REAL (KIND=nag_wp) :: FJAC(N,N)
     ...
     CALL C05RBF(FCN,N,X,FVEC,FJAC,XTOL,IUSER,RUSER,IFAIL)
```

C05PCF/C05PCA

Withdrawn at Mark 25.
Replaced by C05RCF.

```
Old: SUBROUTINE FCN_C05PCF(N,X,FVEC,FJAC,LDFJAC,IFLAG)
        ...
     END SUBROUTINE FCN_C05PCF
     ...
     REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05PCF(FCN_C05PCF,N,X,FVEC,FJAC,LDFJAC,XTOL,MAXFEV,DIAG,MODE,FACTOR, &
                 NPRINT,NFEV,NJEV,R,LR,QTF,W,IFAIL)
        or
     SUBROUTINE FCN_C05PCA(N,X,FVEC,FJAC,LDFJAC,IFLAG,IUSER,RUSER)
        ...
     END SUBROUTINE FCN_C05PCA
     ...
     REAL (KIND=nag_wp) :: FJAC(LDFJAC,N)
     ...
     CALL C05PCA(FCN_C05PCA,N,X,FVEC,FJAC,LDFJAC,XTOL,MAXFEV,DIAG,MODE,FACTOR, &
                 NPRINT,NFEV,NJEV,R,LR,QTF,W,IUSER,RUSER,IFAIL)
New: SUBROUTINE FCN(N,X,FVEC,FJAC,IUSER,RUSER,IFLAG)
        ...
        INTEGER,            INTENT(INOUT) :: IUSER(*)
        REAL (KIND=nag_wp), INTENT(INOUT) :: RUSER(*)
        ...
     END FUNCTION FCN
     ...
     REAL (KIND=nag_wp) :: FJAC(N,N)
     ...
     CALL C05RCF(FCN,N,X,FVEC,FJAC,XTOL,MAXFEV,MODE,DIAG,FACTOR, &
                 NPRINT,NFEV,NJEV,R,QTF,IUSER,RUSER,IFAIL)
```

C05PDF/C05PDA

Withdrawn at Mark 25.
Replaced by C05RDF.

```
Old: REAL (KIND=nag_wp) :: FJAC(LDFJAC,N), RWSAV(10)
     INTEGER            :: IWSAV(15)
     ...
     CALL C05PDF(IREVCM,N,X,FVEC,FJAC,LDFJAC,XTOL,DIAG,MODE,FACTOR, &
                 R,LR,QTF,W,IFAIL)
        or
     CALL C05PDA(IREVCM,N,X,FVEC,FJAC,LDFJAC,XTOL,DIAG,MODE,FACTOR, &
                 R,LR,QTF,W,LWSAV,IWSAV,RWSAV,IFAIL)
New: REAL (KIND=nag_wp) :: FJAC(N,N), RWSAV(4*N+10)
     INTEGER            :: IWSAV(17)
     ...
     CALL C05RDF(IREVCM,N,X,FVEC,FJAC,XTOL,MODE,DIAG,FACTOR,        &
                 R,QTF,IWSAV,RWSAV,IFAIL)
```

C05ZAF

Withdrawn at Mark 25.
Replaced by C05ZDF.

```
Old: CALL C05ZAF(M,N,X,FVEC,FJAC,LDFJAC,XP,FVECP,MODE,ERR)
New: IFAIL = 0
     CALL C05ZDF(MODE,M,N,X,FVEC,FJAC,LDFJAC,XP,FVECP,ERR,IFAIL)
```

The array XP must now have dimension N regardless of the value of MODE, and likewise ERR must now have dimension M regardless. The argument IFAIL is the standard NAG argument for error trapping. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

C06 – Summation of Series

C06DBF

Withdrawn at Mark 25.
Replaced by C06DCF.

```
Old: DO I = 1, LX
        RES(I) = C06DBF(X(I),C,N,S)
     END DO
New: XMIN = -1.0D0
     XMAX = 1.0D0

     SELECT CASE (S)
     CASE (1,2,3)
        S_USE = S
     CASE DEFAULT
        S_USE = 2
     END SELECT

     IFAIL = 0
     CALL C06DCF(X,LX,XMIN,XMAX,C,N,S_USE,RES,IFAIL)
```

The old routine C06DBF returns a single sum at a time, whereas the new routine C06DCF returns a vector of LX values at once. The values supplied in X to C06DCF are un-normalized original variable values in the range $[\mathrm{XMIN}, \mathrm{XMAX}]$. The argument IFAIL is the standard NAG argument for error trapping. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

C06EAF

Withdrawn at Mark 26.
Replaced by C06PAF.

C06PAF removes restrictions on sequence length and combines transform directions.

```
Old: CALL C06EAF(X,N,IFAIL)
New: CALL C06PAF('F',X,N,WORK,IFAIL)
```

where WORK is a real array of length $3 \times \mathrm{N} + 100$ and the dimension of the array X has been extended from the original N to $\mathrm{N} + 2$. The output values X are stored in a different order with real and imaginary parts stored contiguously. The mapping of output elements is as follows:

$\mathrm{X}(2 \times i) \leftarrow \mathrm{X}(i)$, for $i = 0, 1, \ldots, \mathrm{N}/2$ and
$\mathrm{X}(2 \times i + 1) \leftarrow \mathrm{X}(\mathrm{N} - i)$, for $i = 1, 2, \ldots, (\mathrm{N} + 1)/2$.

C06EBF

Withdrawn at Mark 26.
Replaced by C06PAF.

C06PAF removes restrictions on sequence length and combines transform directions.

```
Old: CALL C06EBF(X,N,IFAIL)
New: CALL C06PAF('B',X,N,WORK,IFAIL)
```

where WORK is a real array of length $3 \times \mathrm{N} + 100$ and the dimension of the array X has been extended from the original N to $\mathrm{N} + 2$. The input values of X are stored in a different order with real and imaginary parts stored contiguously. Also C06PAF performs the inverse transform without the need to first conjugate. If prior conjugation of original array X is assumed then the mapping of input elements is:

$\mathrm{X}(2 \times i) \leftarrow \mathrm{X}(i)$, for $i = 0, 1, \ldots, \mathrm{N}/2$ and
$\mathrm{X}(2 \times i + 1) \leftarrow \mathrm{X}(\mathrm{N} - i)$, for $i = 1, 2, \ldots, (\mathrm{N} - 1)/2$.

C06ECF

Withdrawn at Mark 26.
Replaced by C06PCF.

C06PCF removes restrictions on sequence length, combines transform directions and uses complex types.

```
Old: CALL C06ECF(X,Y,N,IFAIL)
New: CALL C06PCF('F',Z,N,WORK,IFAIL)
```

where WORK is a complex array of length $2 \times \mathrm{N} + 15$ and Z is a complex array of length N such that $\mathrm{Z}(i) = \mathrm{CMPLX}(\mathrm{X}(i), \mathrm{Y}(i))$, for $i = 0, 1, \ldots \mathrm{N} - 1$ on input and output.

C06EKF

Withdrawn at Mark 26.
Replaced by C06FKF.

C06FKF removes restrictions on sequence length.

```
Old: CALL C06EKF(IJOB,X,Y,N,IFAIL)
New: CALL C06FKF(IJOB,X,Y,N,WORK,IFAIL)
```

where WORK is a real array of length N.

C06FPF

Scheduled for withdrawal at Mark 28.
Replaced by C06PQF.

C06PQF provides a simpler interface for both forward and backward transforms.

```
Old: CALL C06FPF(M,N,X,INIT,TRIG,WORK,IFAIL)
New: CALL C06PQF('F',N,M,X,WORK,IFAIL)
```

where the dimension of WORK has been extended from $\mathrm{M} \times \mathrm{N}$ to $\mathrm{M} \times \mathrm{N} + 2 \times \mathrm{N}$ (to include TRIG) and the dimension of the array X has been extended from the original $\mathrm{N} \times \mathrm{M}$ to $(\mathrm{N} + 2) \times \mathrm{M}$. The input values are stored stored slightly differently to allow for two extra storage spaces at the end of each sequence.

The mapping of input elements is as follows:

for $j = 1, 2, \ldots, \mathrm{M}$

$J_1 = (j - 1) \times \mathrm{N}$; $J_2 = (j - 1) \times (\mathrm{N} + 2)$;

$\mathrm{X}(J_2 + 2 \times i) \leftarrow \mathrm{X}(J_1 + i)$, for $i = 0, 1, \ldots, \mathrm{N}$;

$\mathrm{X}(J_2 + \mathrm{N})$ and $\mathrm{X}(J_2 + \mathrm{N} + 1)$ need not be set.

The output values X are stored in a different order with real and imaginary parts of each Hermitian sequence stored contiguously.

The mapping of output elements is as follows:

(Here X begins at element zero $\mathrm{X}(0)$.)

For $j = 1, 2, \ldots, \mathrm{M}$

$J_1 = (j - 1) \times \mathrm{N}$; $J_2 = (j - 1) \times (\mathrm{N} + 2)$;

$\mathrm{X}(J_2 + 2 \times i) \leftarrow \mathrm{X}(J_1 + i)$, for $i = 0, 1, \ldots, \mathrm{N}/2$ [real parts];

$\mathrm{X}(J_2 + 2 \times i + 1) \leftarrow \mathrm{X}(J_1 + \mathrm{N} - i)$, for $i = 1, 2, \ldots, \mathrm{N} + 1/2$ [imaginary parts];

$\mathrm{X}(J_2 + 1)$ is set to zero;

$\mathrm{X}(J_2 + \mathrm{N} + 1)$ is set to zero when N is even.

C06FQF

Scheduled for withdrawal at Mark 28.
Replaced by C06PQF.

C06PQF provides a simpler interface for both forward and backward transforms.

```
Old: CALL C06FQF(M,N,X,INIT,TRIG,WORK,IFAIL)
New: CALL C06PQF('B',N,M,X,WORK,IFAIL)
```

where the dimension of WORK has been extended from $\mathrm{M} \times \mathrm{N}$ to $\mathrm{M} \times \mathrm{N} + 2 \times \mathrm{N}$ (to include TRIG) and the dimension of the array X has been extended from the original $\mathrm{N} \times \mathrm{M}$ to $(\mathrm{N} + 2) \times \mathrm{M}$.

The input values X are stored in a different order with real and imaginary parts of each Hermitian sequence stored contiguously.

The mapping of input elements is as follows:

(Here X begins at element zero X(0).)

For $j = 1, 2, \ldots, \mathrm{M}$

$J_1 = (j-1) \times \mathrm{N}$; $J_2 = (j-1) \times (\mathrm{N}+2)$;

$\mathrm{X}(J_2 + 2 \times i) \leftarrow \mathrm{X}(J_1 + i)$, for $i = 0, 1, \ldots, \mathrm{N}/2$ [real parts];

$\mathrm{X}(J_2 + 2 \times i + 1) \leftarrow \mathrm{X}(J_1 + \mathrm{N} - i)$, for $i = 1, 2, \ldots, \mathrm{N} + 1/2$ [imaginary parts];

$\mathrm{X}(J_2 + 1)$ must be zero;

$\mathrm{X}(J_2 + \mathrm{N} + 1)$ must zero when N is even.

The output values are stored stored slightly differently to allow for two extra storage spaces at the end of each sequence.

The mapping of output elements is as follows:

For $j = 1, 2, \ldots, \mathrm{M}$

$J_1 = (j-1) \times \mathrm{N}$; $J_2 = (j-1) \times (\mathrm{N}+2)$;

$\mathrm{X}(J_2 + 2 \times i) \leftarrow \mathrm{X}(J_1 + i)$, for $i = 0, 1, \ldots, \mathrm{N}$;

$\mathrm{X}(J_2 + \mathrm{N})$ and $\mathrm{X}(J_2 + \mathrm{N} + 1)$ will be set to zero.

C06FRF

Withdrawn at Mark 26.
Replaced by C06PSF.

C06PSF provides a simpler interface for both forward and backward transforms.

```
Old: call C06FRF(M,N,X,Y,INIT,TRIG,WORK,IFAIL)
New: Do j = 1, m*n
         cx(j) = cmplx(x(j),y(j),kind=nag_wp)
     End Do
     Call C06PSF('F',M,N,CX,CWORK,IFAIL)
     x(1:m*n) = real(cx(1:m*n))
     y(1:m*n) = aimag(cx(1:m*n))
```

where cx and cwork are complex array of length $\mathrm{m} \times \mathrm{n}$ and $\mathrm{n} \times \mathrm{m} + 2 \times \mathrm{n} + 15$ respectively.

C06FUF

Withdrawn at Mark 26.
Replaced by C06PUF.

C06PUF provides a simpler interface for both forward and backward transforms.

```
Old: Call C06FUF(M,N,X,Y,INIT,TRIGM,TRIGN,WORK,IFAIL)
New: Do j = 1, m*n
         cx(j) = cmplx(x(j),y(j),kind=nag_wp)
     End Do
     Call C06PUF('F',M,N,CX,CWORK,IFAIL)
     x(1:m*n) = real(cx(1:m*n))
     y(1:m*n) = aimag(cx(1:m*n))
```

where cx and cwork are complex arrays of lengths m $\times$ n and n $\times$ m + 2 $\times$ n + 2 $\times$ m + 30 respectively.

C06GBF

Withdrawn at Mark 26.
There is no replacement for this routine.

C06GCF

Withdrawn at Mark 26.
There is no replacement for this routine.

C06GQF

Withdrawn at Mark 26.
There is no replacement for this routine.

C06GSF

Withdrawn at Mark 26.
There is no replacement for this routine.

C06HAF

Withdrawn at Mark 26.
Replaced by C06REF.

C06REF has a simpler interface, storing sequences by column.

```
Old: Call C06HAF(M,N,X,INIT,TRIG,WORK,IFAIL)
New: Call C06REF(M,N,Y,IFAIL)
```

where y$(1 : n-1 : m)$ is a two-dimensional real array such that $y(1 : n-1, j) = x(j : m \times (n-1) : m)$.

C06HBF

Withdrawn at Mark 26.
Replaced by C06RFF.

C06RFF has a simpler interface, storing sequences by column.

```
Old: Call C06HBF(M,N,X,INIT,TRIG,WORK,IFAIL)
New: Call C06RFF(M,N,Y,IFAIL)
```

where y$(0 : n : m)$ is a two-dimensional real array such that $y(0 : n, j) = x(j : m \times (n+1) : m)$.

C06HCF

Withdrawn at Mark 26.
Replaced by C06RGF.

C06RGF has a simpler interface, storing sequences by column.

```
Old: Call C06HCF(DIRECT,M,N,X,INIT,TRIG,WORK,IFAIL)
New: Call C06RGF(IDIR,M,N,Y,IFAIL)
```

where y$(1 : n : m)$ is a two-dimensional real array such that $y(1 : n, j) = x(j : m \times n : m)$; IDIR $= 1$ or -1 for forward and inverse transforms respectively.

C06HDF

Withdrawn at Mark 26.
Replaced by C06RHF.

C06RHF has a simpler interface, storing sequences by column.

```
Old: Call C06HDF(DIRECT,M,N,X,INIT,TRIG,WORK,IFAIL)
New: Call C06RHF(IDIR,M,N,Y,IFAIL)
```

where y$(0 : n - 1 : m)$ is a two-dimensional real array such that $y(0 : n - 1, j) = x(j : m \times n : m)$; IDIR $= 1$ or -1 for forward and inverse transforms respectively.

D01 – Quadrature

D01BAF

Withdrawn at Mark 26.
Replaced by D01UAF.

Withdrawn to provide thread safety in passing of data to user supplied function and a simpler interface to select the quadrature rule.

```
Old :  FUNCTION FUN(x)

         ...
       real(kind=nag_wp) :: FUN
       real(kind=nag_wp), intent(in) :: X
       FUN = ...
       END FUNCTION

       DINEST = D01BAF(D01XXX,A,B,N,FUN,IFAIL)

New :  SUBROUTINE F(X,NX,FV,IFLAG,IUSER,RUSER)
         ...
      ! see example below
         ...
       END SUBROUTINE F
        ...
        integer :: key
        integer, allocatable :: iuser(:)
        real(kind=nag_wp), allocatable :: ruser(:)

       ! set KEY according to quadrature formula
       ! KEY =  0 : (D01XXX=D01BAZ)
       ! KEY = -3 : (D01XXX=D01BAY)
       ! KEY = -4 : (D01XXX=D01BAW)
       ! KEY = -5 : (D01XXX=D01BAX)
       ! KEY = ABS(KEY) for normal weights
       KEY = 0

       allocate(iuser(liuser), ruser(lruser))

       CALL D01UAF(KEY,A,B,N,F,DINEST,IUSER,RUSER,IFAIL)
```

IUSER and RUSER are arrays available to allow you to pass information to the user-supplied subroutine F.

IFLAG is an integer which you may use to force an immediate exit from D01UAF in case of an error in the user-supplied subroutine F.

F may be used to call the original FUN as follows, although it may be more efficient to recode the integrand.

```
SUBROUTINE F(X,NX,FV,IFLAG,IUSER,RUSER)
 ...
 integer, intent(in) :: NX
 integer, intent(inout) :: iflag
 real(kind=nag_wp), intent(in) :: X(NX)
 real(kind=nag_wp), intent(out) :: fv(nx)
 real(kind=wp), intent(inout) :: ruser(*)
 integer, intent(inout) :: iuser(*)
 integer :: j
 external FUN

 do j=1,nx
    FV(j)  = FUN(x(j))
 enddo

END SUBROUTINE F
```

D01BBF

Withdrawn at Mark 26.
Replaced by D01TBF.

Withdrawn to provide thread safety in passing of data to the user-supplied routine and a simpler interface to select the quadrature rule.

```
Old :  CALL D01BBF(D01XXX,A,B,ITYPE,N,WEIGHT,ABSCIS,IFAIL)
New :  Integer :: key
       CALL D01TBF(KEY,A,B,N,WEIGHT,ABSICS,IFAIL)
```

The supplied subroutines D01XXX and the argument ITYPE have been combined into a single argument KEY. KEY < 0 is equivalent to ITYPE $= 1$ (adjusted weights). KEY > 0 is equivalent to ITYPE $= 0$ (normal weights). |KEY| indicates the quadrature rule.

|KEY| $= 0$: Gauss–Legendre (**D01XXX** = **D01BAZ**)

|KEY| $= 3$: Gauss–Laguerre (**D01XXX** = **D01BAX**)

|KEY| $= 4$: Gauss–Hermite (**D01XXX** = **D01BAW**)

|KEY| $= 5$: Rational Gauss (**D01XXX** = **D01BAY**)

D01RBF

Scheduled for withdrawal at Mark 27.
There is no replacement for this routine.

Withdrawn as a separate diagnostic routine is not required. The details of the computation, as stored in the parameters ICOM and COM, are specified in Section 10.1 in D01RAF.

See Section 10 in D01RAF for further details.

D02 – Ordinary Differential Equations

D02PCF

Withdrawn at Mark 26.
Replaced by D02PEF and associated D02P routines.

These replacements were made primarily for reasons of threadsafety.

```
Old: CALL D02PVF(N,TSTART,YINIT,TEND,TOL,THRESH,METHOD,'U',ERRASS, &
                 HSTART,W,LW,IFAIL)
     ...
     CALL D02PCF(F,TWANT,T,Y,YP,YMAX,W,IFAIL)

New: IF (.Not. ERRASS) METHOD = -METHOD
     CALL D02PQF(N,TSTART,TEND,YINIT,TOL,THRESH,METHOD,HSTART,IWSAV, &
                 RWSAV,IFAIL)
     ...
     CALL D02PEF(F2,N,TWANT,T,Y,YP,YMAX,IUSER,RUSER,IWSAV,RWSAV,IFAIL)
```

IWSAV is an integer array of length 130 and RWSAV is a real array of length $350 + 32 \times \mathrm{N}$.

IUSER and RUSER are arrays available to allow you to pass information to the user defined routine F2 (see F in D02PEF).

The definition of F2 (see F in D02PEF) can use the original routine F as follows:

```
   SUBROUTINE F2(T,N,Y,YP,IUSER,RUSER)
!     .. Scalar Arguments ..
      Real (Kind=wp), Intent (In)      :: t
      Integer, Intent (In)             :: n
!     .. Array Arguments ..
      Real (Kind=wp), Intent (Inout)   :: ruser(1)
      Real (Kind=wp), Intent (In)      :: y(n)
      Real (Kind=wp), Intent (Out)     :: yp(n)
      Integer, Intent (Inout)          :: iuser(1)
!     .. Procedure Arguments ..
      External                         :: f
!     .. Executable Statements ..
      Continue

      Call f(t,y,yp)

      Return
    End Subroutine F2
```

D02PDF

Withdrawn at Mark 26.
Replaced by D02PFF or D02PGF and associated D02P routines.

These replacements were made primarily for reasons of threadsafety. D02PGF also offers a reverse communication approach.

```
Old: CALL D02PVF(N,TSTART,YINIT,TEND,TOL,THRESH,METHOD,'U',ERRASS, &
                 HSTART,W,LW,IFAIL)
     ...
     CALL D02PDF(F,T,Y,YP,WORK,IFAIL)

New: IF (.Not. ERRASS) METHOD = -METHOD
     CALL D02PQF(N,TSTART,TEND,YINIT,TOL,THRESH,METHOD,HSTART,IWSAV, &
                 RWSAV,IFAIL)
     ...
     CALL D02PFF(F2,N,T,Y,YP,IUSER,RUSER,IWSAV,RWSAV,IFAIL)
```

IWSAV is an integer array of length 130 and RWSAV is a real array of length $350 + 32 \times \mathrm{N}$.

IUSER and RUSER are arrays available to allow you to pass information to the user defined routine F2 (see F in D02PEF).

The definition of F2 (see F in D02PEF) can use the original routine F as follows:

```
   SUBROUTINE F2(T,N,Y,YP,IUSER,RUSER)
!     .. Scalar Arguments ..
      Real (Kind=wp), Intent (In)      :: t
      Integer, Intent (In)             :: n
!     .. Array Arguments ..
      Real (Kind=wp), Intent (Inout)   :: ruser(1)
      Real (Kind=wp), Intent (In)      :: y(n)
      Real (Kind=wp), Intent (Out)     :: yp(n)
      Integer, Intent (Inout)          :: iuser(1)
!     .. Procedure Arguments ..
      External                         :: f
!     .. Executable Statements ..
      Continue

      Call f(t,y,yp)

      Return
    End Subroutine F2
```

D02PVF

Withdrawn at Mark 26.
Replaced by D02PQF.

This replacement was made primarily for reasons of threadsafety.

See D02PCF and D02PDF for further information.

D02PWF

Withdrawn at Mark 26.
Replaced by D02PRF.

This replacement was made primarily for reasons of threadsafety.

```
Old: CALL D02PWF(TENDNU,IFAIL)
New: CALL D02PRF(TENDNU,IWSAV,RWSAV,IFAIL)
```

IWSAV is an integer array of length 130 and RWSAV is a real array of length 350.

D02PXF

Withdrawn at Mark 26.
Replaced by D02PSF.

This replacement was made primarily for reasons of threadsafety.

```
Old: CALL D02PXF(TWANT,REQEST,NWANT,YWANT,YPWANT,F,WORK,WRKINT, &
                 LENINT,IFAIL)
New:
     If (REQEST=='S' .or. REQEST=='s') Then
       IDERIV = 0
     Else if (REQEST=='D' .or. REQEST=='d') Then
       IDERIV = 1
     Else
       IDERIV = 2
     End If
     CALL D02PSF(TWANT,IDERIV,NWANT,YWANT,YPWANT,F2,WORKINT, &
                 LENINT,IUSER,RUSER,IWSAV,RWSAV,IFAIL)
```

IWSAV is an integer array of length 130 and RWSAV is a real array of length $350 + 32 \times \mathrm{N}$.

IUSER and RUSER are arrays available to allow you to pass information to the user defined routine F2 (see F in D02PSF).

WCOMM is a real array of length LWCOMM. See the routine document for D02PSF for further information.

The definition of F2 (see F in D02PSF) can use the original routine F as follows:

```
   SUBROUTINE F2(T,N,Y,YP,IUSER,RUSER)
!     .. Scalar Arguments ..
      Real (Kind=wp), Intent (In)      :: t
      Integer, Intent (In)             :: n
!     .. Array Arguments ..
      Real (Kind=wp), Intent (Inout)   :: ruser(1)
      Real (Kind=wp), Intent (In)      :: y(n)
      Real (Kind=wp), Intent (Out)     :: yp(n)
      Integer, Intent (Inout)          :: iuser(1)
!     .. Procedure Arguments ..
      External                         :: f
!     .. Executable Statements ..
      Continue

      Call f(t,y,yp)

      Return
    End Subroutine F2
```

D02PYF

Withdrawn at Mark 26.
Replaced by D02PTF.

This replacement was made primarily for reasons of threadsafety.

```
Old: Call D02PYF(TOTFCN,STPCST,WASTE,STPSOK,HNEXT,IFAIL)
New: Call D02PTF(TOTFCN,STPCST,WASTE,STPSOK,HNEXT,IWSAV, &
                 RWSAV,IFAIL)
```

D02PZF

Withdrawn at Mark 26.
Replaced by D02PUF.

This replacement was made primarily for reasons of threadsafety.

```
Old: Call D02PZF(RMSERR,ERRMAX,TERRMX,WORK,IFAIL)
New: Call D02PUF(N,RMSERR,ERRMAX,TERRMX,IWSAV,RWSAV,IFAIL)
```

N must be unchanged from that passed to D02PQF.

IWSAV is an integer array of length 130 and RWSAV is a real array of length $350 + 32 \times \mathrm{N}$.

D02TKF

Scheduled for withdrawal at Mark 27.
Replaced by D02TLF.

This replacement was made primarily for reasons of threadsafety.

```
Old: Call D02TKF(FFUN,FJAC,GAFUN,GBFUN,GAJAC,GBJAC,GUESS,RCOMM,ICOMM,IFAIL)
New: Call D02TLF(FFUN,FJAC,GAFUN,GBFUN,GAJAC,GBJAC,GUESS,RCOMM,ICOMM,IUSER, &
                 RUSER,IFAIL)
```

The arrays IUSER and RUSER are also supplied as an additional two arguments to the seven user-supplied routines. These arrays are free to use to supply information to the seven routine arguments.

E01 – Interpolation

E01SEF

Withdrawn at Mark 20.
Replaced by E01SGF.

```
Old: CALL E01SEF(M,X,Y,F,RNW,RNQ,NW,NQ,FNODES,MINNQ,WRK,IFAIL)
New: CALL E01SGF(M,X,Y,F,NW,NQ,IQ,LIQ,RQ,LRQ,IFAIL)
```

E01SEF has been superseded by E01SGF which gives improved accuracy, facilities for obtaining gradient values and a consistent interface with E01TGF for interpolation of scattered data in three dimensions.

The interpolant generated by the two routines will not be identical, but similar results may be obtained by using the same values of NW and NQ. Details of the interpolant are passed to the evaluator through the arrays IQ and RQ rather than FNODES and RNW.

E01SFF

Withdrawn at Mark 20.
Replaced by E01SHF.

```
Old: CALL E01SFF(M,X,Y,F,RNW,FNODES,PX,PY,PF,IFAIL)
New: CALL E01SHF(M,X,Y,F,IQ,LIQ,RQ,LRQ,1,PX,PY,PF,QX,QY,IFAIL)
```

The two calls will not produce identical results due to differences in the generation routines E01SEF and E01SGF. Details of the interpolant are passed from E01SGF through the arrays IQ and RQ rather than FNODES and RNW.

E01SHF also returns gradient values in QX and QY and allows evaluation at arrays of points rather than just single points.

E02 – Curve and Surface Fitting

E02ACF

Scheduled for withdrawal at Mark 27.
Replaced by E02ALF.

```
Old: CALL E02ACF(X, Y, N, A, M1, REF)
New: CALL E02ALF(N, X, Y, M1, A, REF, IFAIL)
```

E04 – Minimizing or Maximizing a Function

E04CCF/E04CCA

Withdrawn at Mark 24.
Replaced by E04CBF.

```
Old: CALL E04CCF(N,X,F,TOL,IW,W1,W2,W3,W4,W5,W6,FUNCT,MONIT,MAXCAL,   &
                 IFAIL)
        or
     CALL E04CCA(N,X,F,TOL,IW,W1,W2,W3,W4,W5,W6,FUNCT2,MONIT2,MAXCAL, &
                 IUSER,RUSER,IFAIL)
New: CALL E04CBF(N,X,F,TOLF,TOLX,FUNCT2,MONIT3,MAXCAL,IUSER,RUSER,    &
                 IFAIL)

       SUBROUTINE          MONIT3(FMIN,FMAX,SIM,N,NCALL,SERROR,VRATIO,IUSER,
                                  RUSER)
       INTEGER             N, NCALL, IUSER(*)
       REAL (KIND=nag_wp) FMIN, FMAX, SIM(N+1,N), SERROR, VRATIO, RUSER(*)

       CALL MONIT2(FMIN,FMAX,SIM,N,N+1,NCALL,IUSER,RUSER)
!      Add code here to monitor the values of SERROR and VRATIO, if necessary
       RETURN
       END
```

E04FDF

Withdrawn at Mark 19.
Replaced by E04FYF.

```
Old: CALL E04FDF(M,N,X,FSUMSQ,IW,LIW,W,LW,IFAIL)
New: CALL E04FYF(M,N,LSFUN,X,FSUMSQ,W,LW,IUSER,USER,IFAIL)
```

LSFUN appears in the argument list instead of the fixed-name subroutine LSFUN1 of E04FDF. LSFUN must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In addition it has an extra two arguments, IUSER and USER, over and above those of LSFUN1. It may be derived from LSFUN1 as follows:

```
SUBROUTINE          LSFUN(M,N,XC,FVECC,IUSER,USER)
INTEGER             M, N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FVECC(M), USER(*)

CALL LSFUN1(M,N,XC,FVECC)

RETURN
END
```

In general the extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing. If however, a COMMON block was used to pass information into LSFUN1, or get information from LSFUN1, then the arrays IUSER and USER should be declared appropriately and used for this purpose.

E04GCF

Withdrawn at Mark 19.
Replaced by E04GYF.

```
Old: CALL E04GCF(M,N,X,FSUMSQ,IW,LIW,W,LW,IFAIL)
New: CALL E04GYF(M,N,LSFUN,X,FSUMSQ,W,LW,IUSER,USER,IFAIL)
```

LSFUN appears in the argument list instead of the fixed-name subroutine LSFUN2 of E04GCF. LSFUN must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In addition it has an extra two arguments, IUSER and USER, over and above those of LSFUN2. It may be derived from LSFUN2 as follows:

```
SUBROUTINE          LSFUN(M,N,XC,FVECC,FJACC,LJC,IUSER,USER)
INTEGER             M, N, LJC, IUSER(*)
REAL (KIND=nag_wp) XC(N), FVECC(M), FJACC(LJC,N), USER(*)

CALL LSFUN2(M,N,XC,FVECC,FJACC,LJC)

RETURN
END
```

In general the extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing. If however, a COMMON block was used to pass information through E04GCF into LSFUN2, or get information from LSFUN2, then the arrays IUSER and USER should be declared appropriately and used for this purpose.

E04GEF

Withdrawn at Mark 19.
Replaced by E04GZF.

```
Old: CALL E04GEF(M,N,X,FSUMSQ,IW,LIW,W,LW,IFAIL)
New: CALL E04GZF(M,N,LSFUN,X,FSUMSQ,W,LW,IUSER,USER,IFAIL)
```

LSFUN appears in the argument list instead of the fixed-name subroutine LSFUN2 of E04GEF. LSFUN must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In

addition it has an extra two arguments, IUSER and USER, over and above those of LSFUN2. It may be derived from LSFUN2 as follows:

```
SUBROUTINE          LSFUN(M,N,X,FVECC,FJACC,LJC,IUSER,USER)
INTEGER             M, N, LJC, IUSER(*)
REAL (KIND=nag_wp) XC(N), FVECC(M), FJACC(LJC,N), USER(*)

CALL LSFUN2(M,N,XC,FVECC,FJACC,LJC)

RETURN
END
```

In general the extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing. If however, a COMMON block was used to pass information through E04GEF into LSFUN2, or get information from LSFUN2, then the arrays IUSER and USER should be declared appropriately and used for this purpose.

E04HFF

Withdrawn at Mark 19.
Replaced by E04HYF.

```
Old: CALL E04HFF(M,N,X,FSUMSQ,IW,LIW,W,LW,IFAIL)
New: CALL E04HYF(M,N,LSFUN,LSHES,X,FSUMSQ,W,LW,IUSER,USER,IFAIL)
```

LSFUN and LSHES appear in the argument list instead of the fixed-name subroutines LSFUN2 and LSHES2 of E04HFF. LSFUN and LSHES must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In addition they have an extra two arguments, IUSER and USER, over and above those of LSFUN2 and LSHES2. They may be derived from LSFUN2 and LSHES2 as follows:

```
SUBROUTINE          LSFUN(M,N,XC,FVECC,FJACC,LJC,IUSER,USER)
INTEGER             M, N, LJC, IUSER(*)
REAL (KIND=nag_wp) XC(N), FVECC(M), FJACC(LJC,N), USER(*)

CALL LSFUN2(M,N,XC,FVECC,FJACC,LJC)

RETURN
END

SUBROUTINE          LSHES(M,N,FVECC,XC,B,LB,IUSER,USER)
INTEGER             M, N, LB, IUSER(*)
REAL (KIND=nag_wp) FVECC(M), XC(N), B(LB), USER(*)

CALL LSHES2(M,N,FVECC,XC,B,LB)

RETURN
END
```

In general, the extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing. If, however, a COMMON block was used to pass information through E04HFF into LSFUN2 or LSHES2, or to get information from LSFUN2 or LSHES2, then the arrays IUSER and RUSER should be declared appropriately and used for this purpose.

E04JAF

Withdrawn at Mark 19.
Replaced by E04JYF.

```
Old: CALL E04JAF(N,IBOUND,BL,BU,X,F,IW,LIW,LW,IFAIL)
New: CALL E04JYF(N,IBOUND,FUNCT,BL,BU,X,F,IW,LIW,W,LW,IUSER,USER,IFAIL)
```

FUNCT appears in the argument list instead of the fixed-name subroutine FUNCT1 of E04JAF. FUNCT must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In

addition it has an extra two arguments, IUSER and USER, over and above those of FUNCT1. It may be derived from FUNCT1 as follows:

```
SUBROUTINE          FUNCT(N,XC,FC,IUSER,USER)
INTEGER             N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FC, USER(*)

CALL FUNCT1(N,XC,FC)

RETURN
END
```

The extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing.

E04KAF

Withdrawn at Mark 19.
Replaced by E04KYF.

```
Old: CALL E04KAF(N,IBOUND,BL,BU,X,F,G,IW,LIW,W,LW,IFAIL)
New: CALL E04KYF(N,IBOUND,FUNCT,BL,BU,X,F,G,IW,LIW,W,LW,IUSER,USER,IFAIL)
```

FUNCT appears in the argument list instead of the fixed-name subroutine FUNCT2 of E04KAF. FUNCT must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub) program. In addition it has an extra two arguments, IUSER and USER, over and above those of FUNCT2. It may be derived from FUNCT2 as follows:

```
SUBROUTINE          FUNCT(N,XC,FC,GC,IUSER,USER)
INTEGER             N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FC, GC(N), USER(*)

CALL FUNCT2(N,XC,FC,GC)

RETURN
END
```

The extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing.

E04KCF

Withdrawn at Mark 19.
Replaced by E04KZF.

```
Old: CALL E04KCF(N,IBOUND,BL,BU,X,F,G,IW,LIW,W,LW,IFAIL)
New: CALL E04KZF(N,IBOUND,FUNCT,BL,BU,X,F,G,IW,LIW,W,LW,IUSER,USER,IFAIL)
```

FUNCT appears in the argument list instead of the fixed-name subroutine FUNCT2 of E04KCF. FUNCT must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub) program. In addition it has an extra two arguments, IUSER and USER, over and above those of FUNCT2. It may be derived from FUNCT2 as follows:

```
SUBROUTINE          FUNCT(N,XC,FC,GC,IUSER,USER)
INTEGER             N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FC, GC(N), USER(*)

CALL FUNCT2(N,XC,FC,GC)

RETURN
END
```

The extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing.

E04LAF

Withdrawn at Mark 19.
Replaced by E04LYF.

```
Old: CALL E04LAF(N,IBOUND,BL,BU,X,F,G,IW,LIW,W,LW,IFAIL)
New: CALL E04LYF(N,IBOUND,FUNCT,HESS,BL,BU,X,F,G,IW,LIW,W,LW,IUSER,USER, &
                 IFAIL)
```

FUNCT and HESS appear in the argument list instead of the fixed-name subroutines FUNCT2 and HESS2 of E04LAF. FUNCT and HESS must be declared as EXTERNAL or be a module subprogram USEd in the calling (sub)program. In addition they have an extra two arguments, IUSER and USER, over and above those of FUNCT2 and HESS2. They may be derived from FUNCT2 and HESS2 as follows:

```
SUBROUTINE          FUNCT(N,XC,FC,GC,IUSER,USER)
INTEGER             N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FC, GC(N), USER(*)

CALL FUNCT2(N,XC,FC,GC)

RETURN
END
```

```
SUBROUTINE          HESS(N,XC,HESLC,LH,HESDC,IUSER,USER)
INTEGER             N, LH, IUSER(*)
REAL (KIND=nag_wp) XC(N), HESLC(LH), HESDC(N), USER(*)

CALL HESS2(N,XC,HESLC,LH,HESDC)

RETURN
END
```

In general, the extra arguments, IUSER and USER, should be declared in the calling program as IUSER(1) and USER(1), but will not need initializing.

E04UNF

Withdrawn at Mark 22.
Replaced by E04USF/E04USA.

```
Old: CALL E04UNF(M,N,NCLIN,NCNLN,LDA,LDCJ,LDFJ,     &
                 LDR,A,BL,BU,Y,CONFUN,OBJFUN,ITER,  &
                 ISTATE,C,CJAC,F,FJAC,CLAMDA,OBJF,  &
                 R,X,IWORK,LIWORK,WORK,LWORK,IUSER, &
                 RUSER,IFAIL)
New: CALL E04USF(M,N,NCLIN,NCNLN,LDA,LDCJ,LDFJ,     &
                 LDR,A,BL,BU,Y,CONFUN,OBJFUN,ITER,  &
                 ISTATE,C,CJAC,F,FJAC,CLAMDA,OBJF,  &
                 R,X,IWORK,LIWORK,WORK,LWORK,IUSER, &
                 RUSER,IFAIL)
```

The specification of the subroutine OBJFUN must also be changed as follows:

```
Old: SUBROUTINE          OBJFUN(MODE,M,N,LDFJ,X,F,FJAC,NSTATE,IUSER,RUSER)
     INTEGER             MODE,M,N,LDFJ,NSTATE,IUSER(*)
     REAL (KIND=nag_wp) X(N),F(*),FJAC(LDFJ,*),RUSER(*)
New: SUBROUTINE          OBJFUN(MODE,M,N,LDFJ,NEEDFI,X,F,FJAC,NSTATE,      &
                         IUSER,RUSER)
     INTEGER             MODE,M,N,LDFJ,NEEDFI,NSTATE,IUSER(*)
     REAL (KIND=nag_wp) X(N),F(*),FJAC(LDFJ,*),RUSER(*)
```

See the routine documents for further information.

E04UPF

Withdrawn at Mark 19.
Replaced by E04USF/E04USA.

```
Old: CALL E04UPF(M,N,NCLIN,NCNLN,LDA,LDCJ,LDFJ,LDR,A,BL,BU, &
                 CONFUN,OBJFUN,ITER,ISTATE,C,CJAC,F,FJAC,   &
                 CLAMDA,OBJF,R,X,IWORK,LIWORK,WORK,LWORK,   &
                 IUSER,USER,IFAIL)
New: CALL E04USF(M,N,NCLIN,NCNLN,LDA,LDCJ,LDFJ,LDR,A,BL,BU, &
                 Y,CONFUN,OBJFUN,ITER,ISTATE,C,CJAC,F,FJAC, &
                 CLAMDA,OBJF, R,X,IWORK,LIWORK,WORK,LWORK,  &
                 IUSER,USER,IFAIL)
```

E04USF/E04USA contains one additional argument as follows:

Y(M) – real array.

Note that a call to E04UPF is the same as a call to E04USF/E04USA with $\mathrm{Y}(i) = 0.0$, for $i = 1, 2, \ldots, \mathrm{M}$.

The specification of the subroutine OBJFUN must also be changed as follows:

```
Old: SUBROUTINE          OBJFUN(MODE,M,N,LDFJ,X,F,FJAC,NSTATE,IUSER,USER)
     INTEGER             MODE,M,N,LDFJ,NSTATE,IUSER(*)
     REAL (KIND=nag_wp)  X(N),F(*),FJAC(LDFJ,*),USER(*)
New: SUBROUTINE          OBJFUN(MODE,M,N,LDFJ,NEEDFI,X,F,FJAC,NSTATE,     &
                         IUSER,USER)
     INTEGER             MODE,M,N,NEEFI,NSTATE,IUSER(*)
     REAL (KIND=nag_wp)  X(N),F(*),FJAC(LDFJ,*),USER(*)
```

See the routine documents for further information.

E04ZCF/E04ZCA

Withdrawn at Mark 24.
There is no replacement for this routine.

F01 – Matrix Operations, Including Inversion

F01MAF

Withdrawn at Mark 19.
Replaced by F11JAF.

Existing programs should be modified to call F11JAF. The interfaces are significantly different and therefore precise details of a replacement call cannot be given. Please consult the appropriate routine document.

F02 – Eigenvalues and Eigenvectors

F02BBF

Withdrawn at Mark 19.
Replaced by F08FBF (DSYEVX).

```
Old:  CALL F02BBF(A,LDA,N,RLB,RUB,M,MM,R,V,LDV,D,E,E2,X,G,C,  &
                  ICOUNT,IFAIL)
New:  CALL DSYEVX('V','V','L',N,A,LDA,RLB,RUB,   &
                  0,0,2*X02AMF(),MM,R,V,LDV,WORK,LWORK,IWORK, &
                  JFAIL,INFO)
```

where R must have dimension at least $\max(1, \mathrm{N})$, WORK is a real array of length at least $(4 \times \mathrm{N})$, LWORK is its actual length, JFAIL is an integer array of length at least $\max(1, \mathrm{N})$, and IWORK is an integer array of length at least $(5 \times \mathrm{N})$. Note that in the call to F02BBF R needs only to be of dimension (M). Larger values of LWORK, up to some optimal value, may improve performance. Arguments C, ICOUNT, X, G, E2, E and D are not used.

F02BCF

Withdrawn at Mark 19.
Replaced by F02ECF.

```
Old: CALL F02BCF(A,IA,N,ALB,UB,M,MM,RR,RI,VR,IVR,VI,IVI,   &
                 INTGER,ICNT,C,B,IB,U,V,IFAIL)
New: CALL F02ECF('Moduli',N,A,IA,ALB,UB,M,MM,RR,RI,VR,IVR, &
                 VI,IVI,WORK,LWORK,ICNT,C,IFAIL)
```

where WORK is a real array of length at least $(\mathrm{N} \times (\mathrm{N}+4))$ and LWORK is its actual length.

F02BDF

Withdrawn at Mark 19.
Replaced by F02GCF.

```
Old: CALL F02BDF(AR,IAR,AI,IAI,N,ALB,UB,M,MM,RR,RI,VR,IVR,  &
                 VI,IVI,INTGER,C,BR,IBR,BI,IBI,U,V,IFAIL)
New: DO 20 J = 1, N
        DO 10 I = 1, N
           A(I,J) = CMPLX(AR(I,J),AI(I,J),KIND=nag_wp)
  10    CONTINUE
  20 CONTINUE
     CALL F02GCF('Moduli',N,A,IA,ALB,UB,M,MM,R,V,IV,WORK,   &
                 LWORK,RWORK,INTGER,C,IFAIL)
     DO 30 I = 1, N
        RR(I) = REAL(R(I))
        RI(I) = AIMAG(R(I))
  30 CONTINUE
     DO 50 J = 1, MM
        DO 40 I = 1, N
           VR(I,J) = REAL(V(I,J))
           VI(I,J) = AIMAG(V(I,J))
  40    CONTINUE
  50 CONTINUE
```

where A is a complex array of dimension $(\mathrm{IA}, \mathrm{N})$, R is a complex array of dimension (N), V is a complex array of dimension $(\mathrm{IV}, \mathrm{M})$, WORK is a complex array of length at least $(\mathrm{N} \times (\mathrm{N}+2))$, LWORK is its actual length, and RWORK is a real array of length at least $(2 \times \mathrm{N})$.

F02BJF

Withdrawn at Mark 23.
Replaced by F08WAF (DGGEV).

```
Old: CALL F02BJF(N,A,LDA,B,LDB,EPS1,ALFR,ALFI,BETA,MATV,V,LDV,ITER,IFAIL)
New: IF (MATV) THEN
       JOBVR = 'V'
     ELSE
       JOBVR = 'N'
     ENDIF
     CALL DGGEV('N',JOBVR,N,A,LDA,B,LDB,ALFR,ALFI,BETA,VL,LDVL,          &
                VR,LDVL,WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
       ...
```

F02EAF

Withdrawn at Mark 23.
Replaced by F08PAF (DGEES).

```
Old: CALL F02EAF(JOB,N,A,LDA,WR,WI,Z,LDZ,WORK,LWORK,IFAIL)
New: LOGICAL SELECT
     EXTERNAL SELECT
     ...
     IF (JOB.EQ.'N') THEN
       JOBVS = 'N'
     ELSE
       JOBVS = 'V'
     END IF
     CALL DGEES(JOBVS,'N',SELECT,N,A,LDA,0,WR,WI,Z,LDZ,WORK, &
                LWORK,BWORK,INFO)
     IF (INFO.EQ.0) THEN
        ....
     LOGICAL FUNCTION SELECT(AR,AI)
     REAL (KIND=nag_wp) :: AR, AI
     SELECT = .TRUE.
     RETURN
     ENDK
```

F02EBF

Withdrawn at Mark 23.
Replaced by F08NAF (DGEEV).

```
Old: CALL F02EBF(JOB,N,A,LDA,WR,WI,VR,LDVR,VI,LDVI,WORK,LWORK, &
                 IFAIL)
New: IF (JOB.EQ.'N') THEN
        JOBVR = 'N'
     ELSE
        JOBVR = 'V'
     END IF
     CALL DGEEV('N',JOBVR,N,A,LDA,WR,WI,VL,LDVL,VR1,LDVR1,     &
                WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
!       Eigenvector information is stored differently.
!       For complex conjugate pairs (that is, corresponding
!       to the j-th eigenvector such that WI(j) is nonzero,
!       and WI(j) = -WI(j+1)), the real and imaginary parts
!       of the first of the pair of eigenvectors are stored
!       as consecutive columns of VR1: VR1(:,j), VR1(:,j+1).
!       The second in the pair is just the conjugate of the
!       first, so can be constructed by negating the
!       elements in VR1(:,j+1).
!       If the j-th eigenvector is real (WI(j)=0), the
!       corresponding real eigenvector is stored in the
!       j-th column of VR1, VR1(1:N,j).
```

F02FAF

Withdrawn at Mark 23.
Replaced by F08FAF (DSYEV).

```
Old: CALL F02FAF(JOB,UPLO,N,A,LDA,W,WORK,LWORK,IFAIL)
New: CALL DSYEV(JOB,UPLO,N,A,LDA,W,WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
       ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F02FCF

Withdrawn at Mark 23.
Replaced by F08FBF (DSYEVX).

```
Old: CALL F02FCF(JOB,RANGE,UPLO,N,A,LDA,WL,WU,IL,IU,MEST,M,    &
                 W,Z,LDZ,WORK,LWORK,IWORK,IFAIL)
New: CALL DSYEVX(JOB,RANGE,UPLO,N,A,LDA,WL,WU,IL,IU,ABSTOL,M, &
                 W,Z,LDZ,WORK,LWORK,IWORK,JFAIL,INFO)
     IF (INFO.EQ.0) THEN
       ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F02FDF

Withdrawn at Mark 23.
Replaced by F08SAF (DSYGV).

```
Old: CALL F02FDF(ITYPE,JOB,UPLO,N,A,LDA,B,LDB,W,WORK,LWORK,IFAIL)
New: CALL DSYGV(ITYPE,JOB,UPLO,N,A,LDA,B,LDB,W,WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
       ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F02FHF

Withdrawn at Mark 23.
Replaced by F08UAF (DSBGV).

```
Old:  CALL F02FHF(N,MA,A,LDA,MB,B,LDB,D,WORK,LWORK,IFAIL)
New:  CALL DSBGV('N','U',N,MA,MB,A,LDA,B,LDB,D,Z,LDZ,WORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The order of eigenvalues in D changes from descending to ascending.

The minimum workspace requirement has changed to become LWORK $= 3 \times$ N

F02GAF

Withdrawn at Mark 23.
Replaced by F08PNF (ZGEES).

```
Old: CALL F02GAF(JOB,N,A,LDA,W,Z,LDZ,RWORK,WORK,LWORK,IFAIL)
New: LOGICAL BWORK(1)
     LOGICAL SELECT
     EXTERNAL SELECT
        ...
     IF (JOB.EQ.'N') THEN
        JOBVS = 'N'
     ELSE
        JOBVS = 'V'
     END IF
     CALL ZGEES(JOBVS,'N',SELECT,N,A,LDA,0,W,Z,LDZ,          &
                WORK,LWORK,RWORK,BWORK,INFO)
     IF (INFO.NE.0) THEN
       ...
     LOGICAL FUNCTION SELECT(C)
     COMPLEX*16 C
     SELECT = .TRUE.
     RETURN
     END
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK = −1) should be used to determine the requirement for optimal performance.

F02GBF

Withdrawn at Mark 23.
Replaced by F08NNF (ZGEEV).

```
Old: CALL F02GBF(JOB,N,A,LDA,W,V,LDV,RWORK,WORK,LWORK,IFAIL)
New: CALL ZGEEV('N',JOB,N,A,LDA,W,VL,LDVL,V,LDV,             &
                WORK,LWORK,RWORK,INFO)
     IF (INFO.EQ.0) THEN
       ...
```

F02GJF

Withdrawn at Mark 23.
Replaced by F08WNF (ZGGEV).

```
Old:  CALL F02GJF(N,AR,LDAR,AI,LDAI,BR,LDBR,BI,LDBI,EPS1,ALFR,  &
                  ALFI,BETA,MATV,VR,LDVR,VI,LDVI,ITER,IFAIL)
New:  IF (MATV) THEN
         JOBVR = 'V'
      ELSE
         JOBVR = 'N'
      END IF

!     Set A=AR + iAI and B = BR+iBI

      CALL ZGGEV('N',JOBVR,N,A,LDA,B,LDB,ALPHA,BETA1,VL,LDVL,   &
                 V,LDV,WORK,LWORK,RWORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

Note that the separated real and imaginary parts of input and output data in F02GJF has been replaced by combined complex types in F08WNF (ZGGEV).

F02HAF

Withdrawn at Mark 23.
Replaced by F08FNF (ZHEEV).

```
Old:  CALL F02HAF(JOB,UPLO,N,A,LDA,W,RWORK,WORK,LWORK,IFAIL)
New:  CALL ZHEEV(JOB,UPLO,N,A,LDA,W,WORK,LWORK,RWORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK = −1) should be used to determine the requirement for optimal performance.

F02HCF

Withdrawn at Mark 23.
Replaced by F08FPF (ZHEEVX).

```
Old:  CALL F02HCF(JOB,RANGE,UPLO,N,A,LDA,WL,WU,IL,IU,MEST,M,   &
                  W,Z,LDZ,WORK,LWORK,RWORK,IWORK,IFAIL)
New:  CALL ZHEEVX(JOB,RANGE,UPLO,N,A,LDA,WL,WU,IL,IU,ABSTOL,M, &
                  W,Z,LDZ,WORK,LWORK,RWORK,IWORK,JFAIL,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK = −1) should be used to determine the requirement for optimal performance.

F02HDF

Withdrawn at Mark 23.
Replaced by F08SNF (ZHEGV).

```
Old:  CALL F02HDF(ITYPE,JOB,UPLO,N,A,LDA,B,LDB,W,RWORK,WORK, &
                  LWORK,IFAIL)
New:  CALL ZHEGV(ITYPE,JOB,UPLO,N,A,LDA,B,LDB,W,WORK,LWORK,  &
                  RWORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F02SDF

Scheduled for withdrawal at Mark 27.
Replaced by F12AGF and F12FGF.

The replacement routines F12FGF (symmetric case) and F12AGF (nonsymmetric case) are threaded for parallel execution in multithreaded implementations. These routines are based on the ARPACK package and make calls to BLAS/LAPACK routines. These may be threaded within the vendor library used by the implementation, which provides an additional opportunity for multithreaded performance.

```
Old: CALL F02SDF(N,MA+1,MB+1,A,LDA,B,LDB,SYM,RELEP,RMU,VEC,D,IWORK,WORK, &
                 LWORK,IFAIL)
New: LICOMM = 140
     LCOMM = 3*N + 3*NCV*NCV + 6*NCV + 60
     ALLOCATE (COMM(LCOMM),DR(NCV),DI(NCV),RESID(N),V(N,NCV), &
               ICOMM(LICOMM))
     !   B is symmetric definite:
     IF (B_symm_def) THEN
       CALL F12AFF(N,1,NCV,ICOMM,LICOMM,COMM,LCOMM,IFAIL)
       CALL F12ADF('Generalized',ICOMM,COMM,IFAIL)
       CALL F12ADF('Shifted Inverse',ICOMM,COMM,IFAIL)
       CALL F12AGF(KL,KU,A,LDA,B,LDB,RMU,0.0,NCONV,DR,DI,V,N,RESID, &
                   V,LDV,COMM,ICOMM,IFAIL)
       VEC(1:N) = V(1:N,1)
     ELSE
       CALL F12AAF(N,NEV,NCV,ICOMM,LICOMM,COMM,LCOMM,IFAIL)
       ALLOCATE(C(LDA,N),IPIV(N),X(N),MX(N))
       C = A - RMU*B
       CALL DGBTRF(N,N,KL,KU,C,LDA,IPIV,INFO)
       IREVCM = 0
       DO WHILE (IREVCM/=5)
         CALL F12ABF(IREVCM,RESID,V,LDV,X,MX,NSHIFT,COMM,ICOMM,IFAIL)
       IF (IREVCM==-1 .OR. IREVCM==1) THEN
     !     Perform  x <--- OP*x = inv[A-SIGMA*B]*Bx.
           CALL DGBMV('N',N,N,KL,KU,ONE,B,LDB,X,1,ZERO,MX,1)
           X(1:N) = MX(1:N)
           CALL DGBTRS('N',N,KL,KU,1,C,LDA,IPIV,X,N,INFO)
         END IF
       END DO
     ! Post-process using F12ACF to compute eigenvalue.
       CALL F12ACF(NCONV,DR,DI,V,LDV,RMU,0.0,RESID,V,N,COMM,ICOMM,IFAIL)
       LR = DR(1)/(DR(1)**2+DI(1)**2) + RMU
     END IF
```

F02WDF

Scheduled for withdrawal at Mark 27.
Replaced by F02WUF and F08AEF (DGEQRF).

This routine is replaced for multithreaded performance and ability to benefit from vendor library performance (BLAS/LAPACK).

Note: Only the multithreaded implementations of F02WDF were able to benefit from parallelism or vendor BLAS/LAPACK performance.

The Householder QU factorization part of the functionality can be achieved with F08AEF (DGEQRF). The action $Q^{\mathrm{T}}b$ can be computed by a call to F08AGF (DORMQR). The orthogonal matrix Q can be explicitly constructed, in-place, by a subsequent call to F08AFF (DORGQR).

If the singular value decomposition (SVD) of U is required, the result of F08AEF (DGEQRF) must be fed to F02WUF, remembering that the first orthogonal matrix of the SVD is called Q in F02WUF and R in F02WDF

```
Old: IFAIL = 0
     CALL F02WDF(M,N,A,LDA,WANTB,B,TOL,SVD,IRANK,Z,SV,WANTR,R, &
          LDR,WANTPT,PT,LDPT,WORK,LWORK,IFAIL)
New: LWORK = -1
     CALL DGEQRF(M,N,A,LDA,Z,WORK,LWORK,INFO)
     LWORK = ANINT(WORK(1))
     DEALLOCATE (WORK)
     ALLOCATE (WORK(LWORK))
     CALL DGEQRF(M,N,A,LDA,Z,WORK,LWORK,INFO)
     NCOLB = 1
     IF (WANTB) THEN
       CALL DORMQR('L','T',M,NCOLB,N,A,LDA,Z,B,M,WORK,LWORK,INFO)
     END IF
     IF (.NOT. SVD) THEN
       ! construct Q explicitly, overwrites A
       CALL DORGQR(M,M,A,LDA,Z,WORK,LWORK,INFO)
     ELSE
       ! SVD factorization, PT overwrites A
       DEALLOCATE (WORK)
       ALLOCATE (WORK(5*N))
       CALL F02WUF(N,A,LDA,NCOLB,B,M,WANTR,R,LDR,SV,WANTPT,WORK,IFAIL)
       ! compute rank
       IRANK = F06KLF(N,SV,1,TOL)
     END IF
```

F02WEF

Withdrawn at Mark 23.
Replaced by F08KBF (DGESVD).

```
Old:  CALL F02WEF(M,N,A,LDA,NCOLB,B,LDB,WANTQ,Q,LDQ,SV,WANTP,       &
                  PT,LDPT,WORK,IFAIL)
New:  IF (WANTQ) THEN
         JOBU = 'A'
      ELSE
         JOBU = 'N'
      END IF
      IF (WANTP) THEN
         JOBVT = 'A'
      ELSE
         JOBVT = 'N'
      END IF
      LWORK = -1
      CALL DGESVD(JOBU,JOBVT,M,N,A,LDA,SV,Q,LDQ,PT,LDPT,WORK,LWORK,INFO)
      LWORK = ANINT(WORK(1))
      ALLOCATE (W(LWORK))

      CALL DGESVD(JOBU,JOBVT,M,N,A,LDA,SV,Q,LDQ,PT,LDPT,W,LWORK,INFO)

      DEALLOCATE (W)
```

WORK must be a one-dimensional real array of length at least *lwork* given by: $\max(1, 3 \times \min(\mathrm{M}, \mathrm{N}) + \max(\mathrm{M}, \mathrm{N}), 5 \times \min(\mathrm{M}, \mathrm{N}))$

Larger values of LWORK, up to some optimal value, may improve performance.

Please note that the facility to return $Q^{\mathrm{T}}B$ is not provided so arguments WANTB and B are not required. Instead, F08KBF (DGESVD) has an option to return the entire M $*$ M orthogonal matrix Q, referred to as U in its documentation, through its 8th argument.

F02XEF

Withdrawn at Mark 23.
Replaced by F08KPF (ZGESVD).

```
Old:  CALL F02XEF(M,N,A,LDA,NCOLB,B,LDB,WANTQ,Q,LDQ,SV,WANTP,        &
                  PH,LDPH,RWORK,CWORK,IFAIL)
New:  IF (WANTQ) THEN
         JOBU = 'A'
      ELSE
         JOBU = 'N'
      END IF
      IF (WANTP) THEN
         JOBVT = 'A'
      ELSE
         JOBVT = 'N'
      END IF
      LWORK = -1
      CALL ZGESVD(JOBU,JOBVT,M,N,A,LDA,SV,Q,LDQ,PT,LDPT,WORK,        &
                  LWORK,RWORK,INFO)
      LWORK = ANINT(WORK(1))
      ALLOCATE (W(LWORK))

      CALL ZGESVD(JOBU,JOBVT,M,N,A,LDA,SV,Q,LDQ,PT,LDPT,W,           &
                  LWORK,RWORK,INFO)

      DEALLOCATE (W)
```

WORK must be a one-dimensional complex array of length at least *lwork* given by $\max(1, 2 \times \min(\mathrm{M}, \mathrm{N}) + \max(\mathrm{M}, \mathrm{N}))$

WORK must be a one-dimensional real array of length $\max(1, 5 \times \min(\mathrm{M}, \mathrm{N}))$.

Larger values of LWORK, up to some optimal value, may improve performance.

Please note that the facility to return $Q^{\mathrm{H}}B$ is not provided so arguments WANTB and B are not required. Instead, F08KPF (ZGESVD) has an option to return the entire M $*$ M unitary matrix Q, referred to as U in its documentation, through its 8th argument.

F03 – Determinants

F03AAF

Withdrawn at Mark 25.
Replaced by F07ADF (DGETRF) and F03BAF.

```
Old: IFAIL = 0
     CALL F03AAF(A,LDA,N,DET,WKSPCE,IFAIL)
New: INTEGER IPIV(N)
     ...

     CALL DGETRF(N,N,A,LDA,IPIV,INFO)
     IFAIL = 0
     CALL F03BAF(N,A,LDA,IPIV,D,ID,IFAIL)
     DET = D*2**ID
```

Note: the real array WKSPCE has been replaced by the integer array IPIV for holding the pivots of the factorization.

F03ABF

Withdrawn at Mark 25.
Replaced by F07FDF (DPOTRF) and F03BFF.

```
Old: IFAIL = 0
     CALL F03ABF(A,LDA,N,DET,WKSPCE,IFAIL)
New: CALL DPOTRF('U',N,A,LDA,INFO)
     IFAIL = 0
     CALL F03BFF(N,A,LDA,D,ID,IFAIL)
     DET = D*2**ID
```

Note: the real array WKSPCE is no longer required. Also the upper triangular part of A, stored in A, has been replaced here by its Cholesky factorization; the lower triangular part of A can be used and overwritten by replacing `U' by `L' in the call to DPOTRF above.

F03ACF

Withdrawn at Mark 25.
Replaced by F07HDF (DPBTRF) and F03BHF.

```
Old: IFAIL = 0
     CALL F03ACF(A,LDA,N,M,DET,RL,LDRL,M1,IFAIL)
New: CALL DPBTRF('L',N,M,AB,LDAB,INFO)
     IFAIL = 0
     CALL F03BHF('L',N,KD,AB,LDAB,D,ID,IFAIL)
     DET = D*2**ID
```

Note: the storage of A in arrays A and AB is different. In fact $\mathrm{AB}(i,j) = \mathrm{A}(j,i)$, for $i = 1,2,\ldots,m$ and $j = \max(1, i-m),\ldots,i$ which conforms to the LAPACK banded storage scheme. The factorization is returned in AB rather than in a separate array (RL). The upper part of matrix A can also be stored in AB on input to DPBTRF.

F03ADF

Withdrawn at Mark 25.
Replaced by F07ARF (ZGETRF) and F03BNF.

```
Old: IFAIL = 0
     CALL F03ADF(A,LDA,N,DETR,DETI,WKSPCE,IFAIL)
New: INTEGER IPIV(N)
     ...
     CALL ZGETRF(N,N,A,LDA,IPIV,INFO)
     IFAIL = 0
     CALL F03BNF(N,A,LDA,IPIV,D,ID,IFAIL)
     DETR = REAL(D)*2**ID(1)
     DETI = AIMAG(D)*2**ID(2)
```

Note: the real array WKSPCE has been replaced by the integer array IPIV for holding the pivots of the factorization. The real and imaginary parts of the determinant are independently scaled.

F03AEF

Withdrawn at Mark 25.
Replaced by F07FDF (DPOTRF) and F03BFF.

```
Old: IFAIL = 0
     CALL F03AEF(N,A,LDA,P,D1,ID,IFAIL)
New: CALL DPOTRF('U',N,A,LDA,INFO)
     IFAIL = 0
     CALL F03BFF(N,A,LDA,D1,ID,IFAIL)
```

Note: the upper triangular part of A, stored in A, has been replaced here by its Cholesky factorization; the lower triangular part of A can be used and overwritten by replacing UPLO = 'U' by UPLO = 'L' in the call to F07FDF (DPOTRF) above.

F03AFF

Withdrawn at Mark 25.
Replaced by F07ADF (DGETRF) and F03BAF.

```
Old: IFAIL = 0
     CALL F03AFF(N,EPS,A,LDA,D1,ID,P,IFAIL)
New: INTEGER IPIV(N)
     ...
     CALL DGETRF(N,N,A,LDA,IPIV,INFO)
     IFAIL = 0
     CALL F03BAF(N,A,LDA,IPIV,D1,ID,IFAIL)
```

Note: real array P has been replaced by the integer array IPIV for holding the pivots of the factorization.

F04 – Simultaneous Linear Equations

F04AAF

Withdrawn at Mark 23.
Replaced by F07AAF (DGESV).

```
Old: CALL F04AAF(A,LDA,B,LDB,N,M,C,LDC,WKSPCE,IFAIL)
New: CALL DGESV(N,M,A,LDA,IPIV,B,LDB,INFO)
     IF (INFO.EQ.0) THEN
!       Answer now in B
        ...
```

F04ABF

Scheduled for withdrawal at Mark 28.
Replaced by F07FBF (DPOSVX).

F04ABF and F04ASF have been replaced by F07FBF (DPOSVX) for performance. The replacement routine is threaded by NAG and may also be threaded in the vendor library (BLAS/LAPACK).

```
Old: CALL F04ABF(A,LDA,B,LDB,N,M,C,LDC,WKSPCE,BB,LDBB,IFAIL)
New: CALL f04abf_wrap(a,lda,b,ldb,n,m,c,ldc,wkspce,bb,ldbb,ifail)

    Subroutine f04abf_wrap(a,lda,b,ldb,n,m,c,ldc,wkspce,bb,ldbb,ifail)
!     .. Use Statements ..
      Use nag_library, Only: dposvx, dsymm, nag_wp
!     .. Scalar Arguments ..
      Integer, Intent (In)             :: lda, ldb, ldbb, ldc, m, n
      Integer, Intent (Inout)          :: ifail
!     .. Array Arguments ..
      Real (Kind=nag_wp), Intent (Inout) :: a(lda,*), b(ldb,*)
      Real (Kind=nag_wp), Intent (Out)   :: bb(ldbb,m), c(ldc,m), wkspce(1)
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: rcond, alpha, beta
      Integer                          :: info, ldaf
      Character (1)                    :: equed
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: s(1)
      Real (Kind=nag_wp), Allocatable  :: af(:,:), work(:), ferr(:), berr(:)
      Integer, Allocatable             :: iwork(:)

      ldaf = n
      Allocate (af(ldaf,n),ferr(m),berr(m),work(3*n),iwork(n))
!     The NAG name equivalent of dposvx is f07fbf
      Call dposvx('N','Upper',n,m,a,lda,af,ldaf,equed,s,b,ldb,  &
          c,ldc,rcond,ferr,berr,work,iwork,info)

      ifail = info
      bb(1:n,1:m) = b(1:n,1:m)
      alpha = -1.0_nag_wp
      beta = 1.0_nag_wp
!     The NAG name equivalent of dgemm is f06yaf
      Call dsymm('L','U',n,m,alpha,a,lda,c,ldc,beta,bb,ldbb)

    End Subroutine f04abf_wrap
```

F04ACF

Withdrawn at Mark 23.
Replaced by F07HAF (DPBSV).

```
Old:  CALL F04ACF(A,LDA,B,LDB,N,M,IR,C,LDC,RL,LDRL,M1,IFAIL)
New:  CALL DPBSV('U',N,M,IR,AB,LDAB,B,LDB,INFO)
      IF (INFO.EQ.0) THEN
!        A and AB are stored differently.
!        AB may be regarded as the transpose of A, with the 'U' option.
!        Thus LDAB might be M+1
!        Answer now in B
         ...
```

F04ADF

Withdrawn at Mark 23.
Replaced by F07ANF (ZGESV).

```
Old:  CALL F04ADF(A,LDA,B,LDB,N,M,C,LDC,WKSPCE,IFAIL)
New:  CALL ZGESV(N,M,A,LDA,IPIV,B,LDB,INFO)
      IF (INFO.EQ.0) THEN
!        Answer now in B
         ...
```

F04AEF

Scheduled for withdrawal at Mark 28.
Replaced by F07ABF (DGESVX).

F04AEF and F04ATF have been replaced by F07ABF (DGESVX) for performance. The replacement routine is threaded by NAG and may also be threaded in the vendor library (BLAS/LAPACK).

```
Old: CALL F04AEF(A,LDA,B,LDB,N,M,C,LDC,WKSPCE,AA,LDAA,BB,LDBB,IFAIL)
New: CALL f04aef_wrap(a,lda,b,ldb,n,m,c,ldc,wkspce,aa,ldaa,bb,ldbb,ifail)
    Subroutine f04aef_wrap(a,lda,b,ldb,n,m,c,ldc,wkspce,aa,ldaa,bb,ldbb,ifail)
!     .. Use Statements ..
      Use nag_library, Only: dgesvx, dgemm, nag_wp
!     .. Scalar Arguments ..
      Integer, Intent (In)             :: lda, ldaa, ldb, ldbb, ldc, m, n
      Integer, Intent (Inout)          :: ifail
!     .. Array Arguments ..
      Real (Kind=nag_wp), Intent (Inout) :: a(lda,*), b(ldb,*)
      Real (Kind=nag_wp), Intent (Out)   :: bb(ldbb,m), c(ldc,m), aa(ldaa,n), &
                                            wkspce(1)
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: rcond, alpha, beta
      Integer                          :: info
      Character (1)                    :: equed
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: cscl(1), rscl(1)
      Real (Kind=nag_wp), Allocatable  :: work(:), ferr(:), berr(:)
      Integer, Allocatable             :: ipiv(:), iwork(:)

      Allocate (berr(m),ferr(m),work(4*n),ipiv(n),iwork(n))
!     The NAG name equivalent of dgesvx is f07abf
      Call dgesvx('N','N',n,m,a,lda,aa,ldaa,ipiv,equed,rscl,cscl,b,ldb, &
          c,ldc,rcond,ferr,berr,work,iwork,info)

      ifail = info
      bb(1:n,1:m) = b(1:n,1:m)
      alpha = -1.0_nag_wp
      beta = 1.0_nag_wp
!     The NAG name equivalent of dgemm is f06yaf
      Call dgemm('N','N',n,m,n,alpha,a,lda,c,ldc,beta,bb,ldbb)

    End Subroutine f04aef_wrap
```

F04AFF

Withdrawn at Mark 25.
There is no replacement for this routine.

The factorization and solution of a positive definite linear system can be handled by calls to routines from Chapter F07, e.g., F07FBF (DPOSVX).

For example:

```
Old: IFAIL = 0
     CALL F03AEF(N,A,LDA,P,D1,ID,IFAIL)
     CALL F04AFF(N,NRHS,A,LDA,P,B,LDB,EPS,X,LDX,BB,LDBB,K,IFAIL)
New: CALL DPOSVX('equil','upper',N,NRHS,A,LDA,AF,LDAF,'Yes',P,B, &
                 LDB,X,LDX,RCOND,FERR,BERR,WORK,IWORK,INFO)
     IFAIL = 0
     CALL F03BFF(N,A,LDA,D1,ID,IFAIL)
```

F04AGF

Withdrawn at Mark 25.
There is no replacement for this routine.

The factorization and solution of a positive definite linear system can be handled by calls to routines from Chapter F07, e.g., F07FAF (DPOSV).

For example:

```
Old: IFAIL = 0
     CALL F03AEF(N,A,LDA,P,D1,ID,IFAIL)
     CALL F04AGF(N,NRHS,A,LDA,P,B,LDB,X,LDX)
New: CALL DPOSV('upper',N,NRHS,A,LDA,B,LDB,INFO)
     IFAIL = 0
     CALL F03BFF(N,A,LDA,D1,ID,IFAIL)
```

F04AHF

Withdrawn at Mark 25.
There is no replacement for this routine.

The factorization and solution of a real general linear system can be handled by calls to routines from the Chapter F07, e.g., F07ABF (DGESVX).

For example:

```
Old: IFAIL = 0
     CALL F03AFF(N,EPS,A,LDA,D1,ID,P,IFAIL)
     CALL F04AHF(N,NRHS,A,LDA,AA,LDAA,P,B,LDB,EPS,X,LDX,BB,    &
                 LDBB,K,IFAIL)
New: CALL DGESVX('Equil','No trans',N,NRHS,A,LDA,AA,LDAA,IPIV, &
                 'Yes',R,C,B,LDB,X,LDX,RCOND,FERR,BERR,WORK,   &
                 IWORK,INFO)
     IFAIL = 0
     CALL F03BAF(N,A,LDA,IPIV,D1,ID,IFAIL)
```

F04AJF

Withdrawn at Mark 25.
There is no replacement for this routine.

The factorization and solution of a real general linear system can be handled by calls to routines from Chapter F07, e.g., F07AAF (DGESV).

For example:

```
Old: IFAIL = 0
     CALL F03AFF(N,EPS,A,LDA,D1,ID,P,IFAIL)
     CALL F04AJF(N,NRHS,A,LDA,P,B,LDB)
New: CALL DGESV(N,NRHS,A,LDA,IPIV,B,LDB,INFO)
     IFAIL = 0
     CALL F03BAF(N,A,LDA,IPIV,D1,ID,IFAIL)
```

F04ARF

Withdrawn at Mark 23.
Replaced by F07AAF (DGESV).

```
Old:  CALL F04ARF(A,LDA,B,N,C,WKSPCE,IFAIL)
New:  CALL DGESV(N,1,A,LDA,IPIV,B,N,INFO)
      IF (INFO.EQ.0) THEN
!        Answer now in B
         ...
```

F04ASF

Scheduled for withdrawal at Mark 28.
Replaced by F07FBF (DPOSVX).

F04ABF and F04ASF have been replaced by F07FBF (DPOSVX) for performance. The replacement routine is threaded by NAG and may also be threaded in the vendor library (BLAS/LAPACK).

```
Old: CALL F04ASF(A,LDA,B,N,C,WK1,WK2,IFAIL)
New: CALL f04asf_wrap(a,lda,b,n,c,wk1,wk2,ifail)
      Subroutine f04asf_wrap(a,lda,b,n,c,wk1,wk2,ifail)
  !     .. Use Statements ..
        Use nag_library, Only: dposvx, nag_wp
  !     .. Scalar Arguments ..
        Integer, Intent (In)             :: lda, n
        Integer, Intent (Inout)          :: ifail
  !     .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout) :: a(lda,n), b(n)
        Real (Kind=nag_wp), Intent (Out) :: c(n), wk1(1), wk2(1)
  !     .. Local Scalars ..
        Real (Kind=nag_wp)               :: rcond
        Integer                          :: info, ldaf, m
        Character (1)                    :: equed
  !     .. Local Arrays ..
        Real (Kind=nag_wp)               :: s(1), ferr(1), berr(1)
        Real (Kind=nag_wp), Allocatable  :: af(:,:), work(:)
        Integer, Allocatable             :: iwork(:)

        ldaf = n
        m = 1
        Allocate (af(ldaf,n),work(3*n),iwork(n))
  !     The NAG name equivalent of dposvx is f07fbf
        Call dposvx('N','Upper',n,m,a,lda,af,ldaf,equed,s,b,n,  &
            c,n,rcond,ferr,berr,work,iwork,info)
        wk1(1) = rcond
        wk2(1) = berr(1)

        ifail = info
      End Subroutine f04asf_wrap
```

F04ATF

Scheduled for withdrawal at Mark 28.
Replaced by F07ABF (DGESVX).

F04AEF and F04ATF have been replaced by F07ABF (DGESVX) for performance. The replacement routine is threaded by NAG and may also be threaded in the vendor library (BLAS/LAPACK).

```
Old: Call F04ATF(A,LDA,B,N,C,AA,LDAA,WKS1,WKS2,IFAIL)
New: CALL f04atf_wrap(a,lda,b,n,c,aa,ldaa,wks1,wks2,ifail)
      Subroutine f04atf_wrap(a,lda,b,n,c,aa,ldaa,wks1,wks2,ifail)
  !     .. Use Statements ..
        Use nag_library, Only: dgesvx, nag_wp
  !     .. Scalar Arguments ..
        Integer, Intent (In)             :: lda, ldaa, n
        Integer, Intent (Inout)          :: ifail
  !     .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout) :: a(lda,*), b(n)
        Real (Kind=nag_wp), Intent (Out)   :: c(n), aa(ldaa,n), wks1(1),
  wks2(1)
  !     .. Local Scalars ..
        Real (Kind=nag_wp)               :: rcond
        Integer                          :: info, m
        Character (1)                    :: equed
  !     .. Local Arrays ..
        Real (Kind=nag_wp)               :: cscl(1), rscl(1), ferr(1),
  berr(1)
        Real (Kind=nag_wp), Allocatable  :: work(:)
        Integer, Allocatable             :: ipiv(:), iwork(:)

        m = 1
        Allocate (work(4*n),ipiv(n),iwork(n))
  !     The NAG name equivalent of dgesvx is f07abf
        Call dgesvx('N','N',n,m,a,lda,aa,ldaa,ipiv,equed,rscl,cscl,b,n, &
            c,n,rcond,ferr,berr,work,iwork,info)

        ifail = info
        wks1(1) = rcond
        wks2(1) = berr(1)

      End Subroutine f04atf_wrap
```

F04EAF

Withdrawn at Mark 23.
Replaced by F07CAF (DGTSV).

```
Old:  CALL F04EAF(N,D,DU,DL,B,IFAIL)
New:  CALL DGTSV(N,1,DL(2),D,DU(2),B,N,INFO)
      IF (INFO.EQ.0) THEN
!        Answer now in B
         ...
```

F04FAF

Withdrawn at Mark 23.
Replaced by F07JAF (DPTSV), or F07JDF (DPTTRF) and F07JEF (DPTTRS).

```
Old:  CALL F04FAF(JOB,N,D,E,B,IFAIL)
New:  CALL DPTSV(N,1,D,E(2),B,1,INFO)
         ...
```

F04JAF

Withdrawn at Mark 23.
Replaced by F08KAF (DGELSS).

```
Old:  CALL F04JAF(M,N,A,LDA,B,TOL,SIGMA,IRANK,WORK,LWORK,IFAIL)
New:  CALL DGELSS(M,N,1,A,LDA,B,1,S,RCOND,IRANK,WORK,LWORK,INFO)
      IF (INFO.EQ.0) THEN
!        Answer now in B
!        Singular values now in S, not WORK.
!        The standard error is not computed
         ...
```

The minimum workspace requirement has changed from $4 \times \mathrm{N}$ to $3 \times \min(\mathrm{N}, \mathrm{M}) + \max(2 \times \min(\mathrm{N}, \mathrm{M}), \max(\mathrm{M}, \mathrm{N}), 1)$.

F04JDF

Withdrawn at Mark 23.
Replaced by F08KAF (DGELSS).

```
Old:  CALL F04JDF(M,N,A,LDA,B,TOL,SIGMA,IRANK,WORK,LWORK,IFAIL)
New:  CALL DGELSS(M,N,1,A,LDA,B,1,S,RCOND,IRANK,WORK,LWORK,INFO)
!     Note workspace requirements are different.
      IF (INFO.EQ.0) THEN
!        Answer now in B
!        Singular values now in S, not WORK.
!        The standard error is not computed
         ...
```

The minimum workspace requirement has changed from $\mathrm{N} \times (\mathrm{M}+4)$ to $3 \times \min(\mathrm{N}, \mathrm{M}) + \max(2 \times \min(\mathrm{N}, \mathrm{M}), \max(\mathrm{M}, \mathrm{N}), 1)$.

F04JLF

Withdrawn at Mark 23.
Replaced by F08ZBF (DGGGLM).

```
Old:  CALL F04JLF(M,N,P,A,LDA,B,LDB,D,X,Y,WORK,LWORK,IFAIL)
New:  CALL DGGGLM(M,N,P,A,LDA,B,LDB,D,X,Y,WORK,LWORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F04JMF

Withdrawn at Mark 23.
Replaced by F08ZAF (DGGLSE).

```
Old:  CALL F04JMF(M,N,P,A,LDA,B,LDB,C,D,X,WORK,LWORK,IFAIL)
New:  CALL DGGLSE(M,N,P,A,LDA,B,LDB,C,D,X,WORK,LWORK,INFO)
      IF (INFO.EQ.0) THEN
         ...
```

The minimum workspace requirement has not increased but the requirement for optimal performance might be different. The workspace query mechanism (LWORK $= -1$) should be used to determine the requirement for optimal performance.

F04KLF

Withdrawn at Mark 23.
Replaced by F08ZPF (ZGGGLM).

```
Old: CALL F04KLF(M,N,P,A,LDA,B,LDB,D,X,Y,WORK,LWORK,IFAIL)
New: CALL ZGGGLM(M,N,P,A,LDA,B,LDB,D,X,Y,WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
        ...
```

F04KMF

Withdrawn at Mark 23.
Replaced by F08ZNF (ZGGLSE).

```
Old: CALL F04KMF(M,N,P,A,LDA,B,LDB,C,D,X,WORK,LWORK,IFAIL)
New: CALL ZGGLSE(M,N,P,A,LDA,B,LDB,C,D,X,WORK,LWORK,INFO)
     IF (INFO.EQ.0) THEN
        ...
```

F04MAF

Withdrawn at Mark 19.
Replaced by F11JCF.

Existing programs should be modified to call F11JCF. The interfaces are significantly different and therefore precise details of a replacement call cannot be given. Please consult the appropriate routine document.

F04MBF

Withdrawn at Mark 19.
Replaced by F11GDF, F11GEF and F11GFF (or F11JCF or F11JEF).

If a user-defined preconditioner is required existing programs should be modified to call F11GDF, F11GEF and F11GFF. Otherwise F11JCF or F11JEF may be used. The interfaces for these routines are significantly different from that for F04MBF and therefore precise details of a replacement call cannot be given. Please consult the appropriate routine document.

F04YCF

Withdrawn at Mark 26.
Replaced by F04YDF.

F04YDF employs a better algorithm (see Higham N J and Tisseur F (2000) A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra *SIAM J. Matrix. Anal. Appl.* **21** 1185–1201).

```
Old: CALL F04YCF(ICASE,N,X,ESTNRM,WORK,IWORK,IFAIL)
New: CALL F04YDF(IREVCM,M,N,X,LDX,Y,LDY,ESTNRM,T,SEED,WORK,IWORK,IFAIL)
```

F04YDF returns an estimate of the 1-norm of a rectangular $M \times N$ matrix, whereas F04YCF only works with square matrices. The real array X, which was previously used to return matrix–vector products to F04YCF, has been replaced with two real arrays $\mathrm{X}(\mathrm{LDX}, *)$ and $\mathrm{Y}(\mathrm{LDY}, *)$ which are used to return matrix-matrix products to F04YDF. Here, $\mathrm{LDX} \geq \mathrm{N}$, $\mathrm{LDY} \geq \mathrm{M}$ and the second dimensions of X and Y are at least of size T, where you can choose argument T. The sizes of the workspace arrays WORK and IWORK have been increased to $\mathrm{M} \times \mathrm{T}$ and $2 \times \mathrm{N} + 5 \times \mathrm{T} + 20$ respectively. The integer SEED provides a seed for the random number generator used by F04YDF. The integer ICASE has been replaced by IREVCM, which can take the values 0, 1 or 2. See the routine documentation for F04YDF further details about the reverse communication interface.

F04ZCF

Withdrawn at Mark 26.
Replaced by F04ZDF.

F04ZDF employs a better algorithm (see Higham N J and Tisseur F (2000) A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra *SIAM J. Matrix. Anal. Appl.* **21** 1185–1201).

```
Old: CALL F04ZCF(ICASE,N,X,ESTNRM,WORK,IFAIL)
New: CALL F04ZDF(IREVCM,M,N,X,LDX,Y,LDY,ESTNRM,T,SEED,WORK,RWORK,IWORK,IFAIL)
```

F04ZDF returns an estimate of the 1-norm of a rectangular $M \times N$ matrix, whereas F04ZCF only works with square matrices. The complex array X, which was previously used to return matrix–vector products to F04ZCF, has been replaced with two complex arrays $\mathrm{X}(\mathrm{LDX}, *)$ and $\mathrm{Y}(\mathrm{LDY}, *)$ which are used to return matrix-matrix products to F04ZDF. Here, $\mathrm{LDX} \geq \mathrm{N}$, $\mathrm{LDY} \geq \mathrm{M}$ and the second dimensions of X and Y are at least of size T, where you can choose the argument T. The sizes of the workspace arrays WORK and IWORK have been increased to $\mathrm{M} \times \mathrm{T}$ and $2 \times \mathrm{N} + 5 \times \mathrm{T} + 20$ respectively and there is an additional real workspace array RWORK of size $2 \times \mathrm{N}$. The integer SEED provides a seed for the random number generator used by F04ZDF. The integer ICASE has been replaced by IREVCM, which can take the values 0, 1 or 2. See the routine documentation for F04ZDF for further details about the reverse communication interface.

F11 – Large Scale Linear Systems

F11BAF

Withdrawn at Mark 21.
Replaced by F11BDF.

```
Old: CALL F11BAF(METHOD,PRECON,NORM,WEIGHT,ITERM,N,M,TOL,MAXITN, &
                 ANORM,SIGMAX,MONIT,LWREQ,IFAIL)
New: CALL F11BDF(METHOD,PRECON,NORM,WEIGHT,ITERM,N,M,TOL,MAXITN, &
                 ANORM,SIGMAX,MONIT,WORK,LWORK,LWREQ,IFAIL)
```

F11BDF contains two additional arguments as follows:

WORK(LWORK) – real array.

LWORK – integer.

See the routine document for further information.

F11BBF

Withdrawn at Mark 21.
Replaced by F11BEF.

```
Old: CALL F11BBF(IREVCM,U,V,WORK,LWORK,IFAIL)
New: CALL F11BEF(IREVCM,U,V,WGT,WORK,LWORK,IFAIL)
```

WGT must be a one-dimensional real array of length at least n (the order of the matrix) if weights are to be used in the termination criterion, and 1 otherwise. Note that the call to F11BEF requires the weights to be supplied in WGT$(1:n)$ rather than WORK$(1:n)$. The minimum value of the argument LWORK may also need to be changed.

F11BCF

Withdrawn at Mark 21.
Replaced by F11BFF.

```
Old: CALL F11BCF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,IFAIL)
New: CALL F11BFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,WORK,LWORK,IFAIL)
```

F11BFF contains two additional arguments as follows:

WORK(LWORK) – real array.

LWORK – integer.

See the routine document for further information.

F11GAF

Withdrawn at Mark 22.
Replaced by F11GDF.

```
Old: CALL F11GAF(METHOD,PRECON,SIGCMP,NORM,WEIGHT,ITERM,N,TOL,MAXITN,    &
                 ANORM,SIGMAX,SIGTOL,MAXITS,MONIT,LWREQ,IFAIL)
New: CALL F11GDF(METHOD,PRECON,SIGCMP,NORM,WEIGHT,ITERM,N,TOL,MAXITN,    &
                 ANORM,SIGMAX,SIGTOL,MAXITS,MONIT,LWREQ,WORK,LWORK,IFAIL)
```

F11GDF contains two additional arguments as follows:

WORK(LWORK) – real array.

LWORK – integer.

See the routine document for further information.

F11GBF

Withdrawn at Mark 22.
Replaced by F11GEF.

```
Old: CALL F11GBF(IREVCM,U,V,WORK,LWORK,IFAIL)
New: CALL F11GEF(IREVCM,U,V,WGT,WORK,LWORK,IFAIL)
```

WGT must be a one-dimensional real array of length at least n (the order of the matrix) if weights are to be used in the termination criterion, and 1 otherwise. Note that the call to F11GEF requires the weights to be supplied in WGT$(1:n)$ rather than WORK$(1:n)$. The minimum value of the argument LWORK may also need to be changed.

F11GCF

Withdrawn at Mark 22.
Replaced by F11GFF.

```
Old: CALL F11GCF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,ITS,SIGERR,IFAIL)
New: CALL F11GFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,ITS,SIGERR,       &
                 WORK,LWORK,IFAIL)
```

F11GFF contains two additional arguments as follows:

WORK(LWORK) – real array.

LWORK – integer.

See the routine document for further information.

G01 – Simple Calculations on Statistical Data

G01AAF

Withdrawn at Mark 26.
Replaced by G01ATF.

Withdrawn because on output, additional information was needed to allow large datasets to be processed in blocks and the results combined through a call to G01AUF. This information is returned in RCOMM.

```
Old:
     CALL G01AAF(N,X,IWT,WT,XMEAN,S2,S3,S4,XMIN,XMAX,WTSUM,IFAIL)
New:
     PN = 0
     CALL G01ATF(N,X,IWT,WT,PN,XMEAN,S2,S3,S4,XMIN,XMAX,RCOMM,IFAIL)
     IWT = PN
     WTSUM = RCOMM(1)
```

G05 – Random Number Generators

G05CAF

Withdrawn at Mark 22.
Replaced by G05SAF.

```
Old: DO 20 I = 1, N
        X(I) = G05CAF(X(I))
  20 CONTINUE
New: CALL G05SAF(N,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SAF contains information on the base generator being used. This array must have been initialized prior to calling G05SAF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SAF is likely to be different from those produced by G05CAF.

G05CBF

Withdrawn at Mark 22.
Replaced by G05KFF.

```
Old: CALL G05CBF(I)
New: LSEED = 1
     SEED(1) = I
     GENID = 1
     SUBID = 1
     CALL G05KFF(GENID,SUBID,SEED,LSEED,STATE,LSTATE,IFAIL)
```

The integer array STATE in the call to G05KFF contains information on the base generator being used. The base generator is chosen via the integer arguments GENID and SUBID. The required length of the array STATE depends on the base generator chosen. Due to changes in the underlying code a sequence of values produced by using a random number generator initialized via a call to G05KFF is likely to be different from a sequence produced by a generator initialized by G05CBF, even if the same value for I is used.

G05CCF

Withdrawn at Mark 22.
Replaced by G05KGF.

```
Old: CALL G05CCF
New: GENID = 1
     SUBID = 1
     CALL G05KGF(GENID,SUBID,STATE,LSTATE,IFAIL)
```

The integer array STATE in the call to G05KGF contains information on the base generator being used. The base generator is chosen via the integer arguments GENID and SUBID. The required length of the array STATE depends on the base generator chosen.

G05CFF

Withdrawn at Mark 22.
Replaced by F06DFF.

```
Old: CALL G05CFF(IA,NI,XA,NX,IFAIL)
New: LSTATE = STATE(1)
     CALL F06DFF(LSTATE,STATE,1,CSTATE,1)
```

The state of the base generator for the group of routines G05KFF, G05KGF, G05KHF, G05KJF, G05NCF, G05NDF, G05PDF–G05PZF, G05RCF–G05RZF, G05S and G05T can be saved by simply creating a local copy of the array STATE. The first element of the STATE array contains the number of elements that are used by the random number generating routines, therefore either this number of elements can be copied, or the whole array (as defined in the calling program).

G05CGF

Withdrawn at Mark 22.
Replaced by F06DFF.

```
Old: CALL G05CGF(IA,NI,XA,NX,IFAIL)
New: LSTATE = CSTATE(1)
     CALL F06DFF(LSTATE,CSTATE,1,STATE,1)
```

The state of the base generator for the group of routines G05KFF, G05KGF, G05KHF, G05KJF, G05NCF, G05NDF, G05PDF–G05PZF, G05RCF–G05RZF, G05S and G05T can be restored by simply copying back the previously saved copy of the STATE array. The first element of the STATE array contains the number of elements that are used by the random number generating routines, therefore either this number of elements can be copied, or the whole array (as defined in the calling program).

G05DAF

Withdrawn at Mark 22.
Replaced by G05SQF.

```
Old: DO 10 I = 1, N
        X(I) = G05DAF(AA,BB)
  10 CONTINUE
New: A = MIN(AA,BB)
     B = MAX(AA,BB)
     IFAIL = 0
     CALL G05SQF(N,A,B,STATE,X,IFAIL)
```

The old routine G05DAF returns a single variate at a time, whereas the new routine G05SQF returns a vector of N values in one go. In G05SQF the minimum value must be held in the argument A and the maximum in argument B, therefore $A < B$. This was not the case for the equivalent arguments in G05DAF.

The integer array STATE in the call to G05SQF contains information on the base generator being used. This array must have been initialized prior to calling G05SQF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SQF is likely to be different from those produced by G05DAF.

G05DBF

Withdrawn at Mark 22.
Replaced by G05SFF.

```
Old: DO 10 I = 1, N
        X(I) = G05DBF(AA)
  10 CONTINUE
New: A = ABS(AA)
     IFAIL = 0
     CALL G05SFF(N,A,STATE,X,IFAIL)
```

The old routine G05DBF returns a single variate at a time, whereas the new routine G05SFF returns a vector of N values in one go. In G05SFF argument A must be non-negative, this was not the case for the equivalent argument in G05DBF.

The integer array STATE in the call to G05SFF contains information on the base generator being used. This array must have been initialized prior to calling G05SFF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SFF is likely to be different from those produced by G05DBF.

G05DCF

Withdrawn at Mark 22.
Replaced by G05SLF.

```
Old: DO 10 I = 1, N
        X(I) = G05DCF(A,BB)
  10 CONTINUE
New: B = ABS(BB)
     IFAIL = 0
     CALL G05SLF(N,A,B,STATE,X,IFAIL)
```

The old routine G05DCF returns a single variate at a time, whereas the new routine G05SLF returns a vector of N values in one go. In G05SLF the spread (argument A) must be positive, this was not the case for the equivalent arguments in G05DCF.

The integer array STATE in the call to G05SLF contains information on the base generator being used. This array must have been initialized prior to calling G05SLF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SLF is likely to be different from those produced by G05DCF.

G05DDF

Withdrawn at Mark 22.
Replaced by G05SKF.

```
Old: DO 10 I = 1, N
        X(I) = G05DDF(XMU,SD)
  10 CONTINUE
New: VAR = SD**2
     IFAIL = 0
     CALL G05SKF(N,XMU,VAR,STATE,X,IFAIL)
```

The old routine G05DDF returns a single variate at a time, whereas the new routine G05SKF returns a vector of N values in one go. G05SKF expects the variance of the Normal distribution (argument VAR), compared to G05DDF which expected the standard deviation.

The integer array STATE in the call to G05SKF contains information on the base generator being used. This array must have been initialized prior to calling G05SKF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SKF is likely to be different from those produced by G05DDF.

G05DEF

Withdrawn at Mark 22.
Replaced by G05SMF.

```
Old: DO 10 I = 1, N
        X(I) = G05DEF(XMU,SD)
  10 CONTINUE
New: VAR = SD**2
     IFAIL = 0
     CALL G05SMF(N,XMU,VAR,STATE,X,IFAIL)
```

The old routine G05DEF returns a single variate at a time, whereas the new routine G05SMF returns a vector of N values in one go. G05SMF expects the variance of the corresponding Normal distribution (argument VAR), compared to G05DEF which expected the standard deviation.

The integer array STATE in the call to G05SMF contains information on the base generator being used. This array must have been initialized prior to calling G05SMF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SMF is likely to be different from those produced by G05DEF.

G05DFF

Withdrawn at Mark 22.
Replaced by G05SCF.

```
Old: DO 10 I = 1, N
        X(I) = G05DFF(XMED,B)
  10 CONTINUE
New: SEMIQR = ABS(B)
     IFAIL = 0
     CALL G05SCF(N,XMED,SEMIQR,STATE,X,IFAIL)
```

The old routine G05DFF returns a single variate at a time, whereas the new routine G05SCF returns a vector of N values in one go. G05SCF expects the semi-interquartile range (argument SEMIQR) to be non-negative, this was not the case for the equivalent argument in G05DFF.

The integer array STATE in the call to G05SCF contains information on the base generator being used. This array must have been initialized prior to calling G05SCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SCF is likely to be different from those produced by G05DFF.

G05DHF

Withdrawn at Mark 22.
Replaced by G05SDF.

```
Old: DO 10 I = 1, N
        X(I) = G05DHF(DF,IFAIL)
  10 CONTINUE
New: CALL G05SDF(N,DF,STATE,X,IFAIL)
```

The old routine G05DHF returns a single variate at a time, whereas the new routine G05SDF returns a vector of N values in one go.

The integer array STATE in the call to G05SDF contains information on the base generator being used. This array must have been initialized prior to calling G05SDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SDF is likely to be different from those produced by G05DHF.

G05DJF

Withdrawn at Mark 22.
Replaced by G05SNF.

```
Old: DO 10 I = 1, N
        X(I) = G05DJF(DF,IFAIL)
  10 CONTINUE
New: CALL G05SNF(N,DF,STATE,X,IFAIL)
```

The old routine G05DJF returns a single variate at a time, whereas the new routine G05SNF returns a vector of N values in one go.

The integer array STATE in the call to G05SNF contains information on the base generator being used. This array must have been initialized prior to calling G05SNF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SNF is likely to be different from those produced by G05DJF.

G05DKF

Withdrawn at Mark 22.
Replaced by G05SHF.

```
Old: DO 10 I = 1, N
        X(I) = G05DKF(DF1,DF2,IFAIL)
  10 CONTINUE
New: CALL G05SHF(N,DF1,DF2,STATE,X,IFAIL)
```

The old routine G05DKF returns a single variate at a time, whereas the new routine G05SHF returns a vector of N values in one go.

The integer array STATE in the call to G05SHF contains information on the base generator being used. This array must have been initialized prior to calling G05SHF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SHF is likely to be different from those produced by G05DKF.

G05DPF

Withdrawn at Mark 22.
Replaced by G05SSF.

```
Old: DO 10 I = 1, N
        X(I) = G05DPF(A,B,IFAIL)
  10 CONTINUE
New: CALL G05SSF(N,A,B,STATE,X,IFAIL)
```

The old routine G05DPF returns a single variate at a time, whereas the new routine G05SSF returns a vector of N values in one go.

The integer array STATE in the call to G05SSF contains information on the base generator being used. This array must have been initialized prior to calling G05SSF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SSF is likely to be different from those produced by G05DPF.

G05DRF

Withdrawn at Mark 22.
Replaced by G05TKF.

```
Old: DO 10 I = 1, N
        X(I) = G05DRF(LAMDA,IFAIL)
  10 CONTINUE
New: MODE = 3
     CALL G05TJF(MODE,N,LAMBDA,R,LR,STATE,X,IFAIL)
```

The old routine G05DRF returns a single variate at a time, whereas the new routine G05TJF returns a vector of N values in one go. For efficiency, the new routine can make use of a reference vector, R. If, as in this case, the integer argument MODE is set to 3, the real reference vector R is not referenced, and its length, LR, need only be at least one.

The integer array STATE in the call to G05TJF contains information on the base generator being used. This array must have been initialized prior to calling G05TJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TJF is likely to be different from those produced by G05DRF.

G05DYF

Withdrawn at Mark 22.
Replaced by G05TLF.

```
Old: DO 10 I = 1, N
        X(I) = G05DYF(AA,BB)
  10 CONTINUE
New: IFAIL = 0
     A = MIN(AA,BB)
     B = MAX(AA,BB)
     CALL G05TLF(N,A,B,STATE,X,IFAIL)
```

The old routine G05DYF returns a single variate at a time, whereas the new routine G05TLF returns a vector of N values in one go. In G05TLF the minimum value must be held in the argument A and the maximum in argument B, therefore $A \leq B$. This was not the case for the equivalent arguments in G05DYF.

The integer array STATE in the call to G05TLF contains information on the base generator being used. This array must have been initialized prior to calling G05TLF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TLF is likely to be different from those produced by G05DYF.

G05DZF

Withdrawn at Mark 22.
Replaced by G05TBF.

```
Old: DO 20 I = 1, N
        X(I) = G05DZF(PP)
  20 CONTINUE
New: P = MAX(0.0D0,MIN(PP,1.0D0))
     IFAIL = 0
     CALL G05TBF(N,P,STATE,X,IFAIL)
```

The old routine G05DZF returns a single variate at a time, whereas the new routine G05TBF returns a vector of N values in one go. The real argument P in G05TBF must not be less than zero or greater than one, this was not the case for the equivalent argument in G05DZF.

The integer array STATE in the call to G05TBF contains information on the base generator being used. This array must have been initialized prior to calling G05TBF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TBF is likely to be different from those produced by G05DZF.

G05EAF

Withdrawn at Mark 22.
Replaced by G05RZF.

```
Old: CALL G05EAF(XMU,M,C,LDC,EPS,R1,LR1,IFAIL)
New: MODE = 0
     CALL G05RZF(MODE,N,M,XMU,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The old routine G05EAF sets up a reference vector for use by G05EZF. The functionality of both these routines has been combined into the single new routine G05RZF. Setting MODE $= 0$ in the call to G05RZF only sets up the real reference vector R and hence mimics the functionality of G05EAF.

The length of the real reference vector, R, in G05RZF must be at least $M \times (M + 1) + 1$. In contrast to the equivalent argument in G05EAF, this array must be allocated in the calling program.

G05EBF

Withdrawn at Mark 22.
Replaced by G05TLF.

There is no direct replacement for routine G05EBF. G05EBF sets up a reference vector for use by G05EYF, this reference vector is no longer required. The replacement routine for G05EYF is G05TLF.

G05ECF

Withdrawn at Mark 22.
Replaced by G05TJF.

```
Old: CALL G05ECF(LAMBDA,R1,LR1,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EYF(R1,LR1)
  10 CONTINUE
New: MODE = 2
     CALL G05TJF(MODE,N,LAMBDA,R,LR,STATE,X,IFAIL)
```

The old routine G05ECF sets up a reference vector for use by G05EYF. The replacement routine G05TJF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05TJF sets up the real reference vector R and hence mimics the functionality of G05ECF. Setting MODE $= 1$ generates a series of variates from a reference vector mimicking the functionality of G05EYF for this particular distribution. Setting MODE $= 2$ initializes the reference vector and generates the variates in one go.

The routine G05EYF returns a single variate at a time, whereas the new routine G05TJF returns a vector of N values in one go.

The length of the real reference vector, R, in G05TJF, must be allocated in the calling program in contrast to the equivalent argument in G05ECF, see the documentation for more details.

The integer array STATE in the call to G05TJF contains information on the base generator being used. This array must have been initialized prior to calling G05TJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TJF is likely to be different from those produced by a combination of G05ECF and G05EYF.

G05EDF

Withdrawn at Mark 22.
Replaced by G05TAF.

```
Old: CALL G05EDF(M,P,R1,LR1,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EYF(R1,LR1)
  10 CONTINUE
New: MODE = 2
     CALL G05TAF(MODE,N,M,P,R,LR,STATE,X,IFAIL)
```

The old routine G05EDF sets up a reference vector for use by G05EYF. The replacement routine G05TAF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05TAF sets up the real reference vector R and hence mimics the functionality of G05EDF. Setting MODE $= 1$ generates a series of variates from a reference vector mimicking the functionality of G05EYF for this particular distribution. Setting MODE $= 2$ initializes the reference vector and generates the variates in one go.

The routine G05EYF returns a single variate at a time, whereas the new routine G05TAF returns a vector of N values in one go.

The length of the real reference vector, R, in G05TAF, needs to be a different length from the equivalent argument in G05EDF, see the documentation for more details.

The integer array STATE in the call to G05TAF contains information on the base generator being used. This array must have been initialized prior to calling G05TAF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TAF is likely to be different from those produced by a combination of G05EDF and G05EYF.

G05EEF

Withdrawn at Mark 22.
Replaced by G05THF.

```
Old: CALL G05EEF(M,P,R1,LR1,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EYF(R1,LR1)
  10 CONTINUE
New: MODE = 2
     CALL G05THF(MODE,N,M,P,R,LR,STATE,X,IFAIL)
```

The old routine G05EEF sets up a reference vector for use by G05EYF. The replacement routine G05THF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05THF sets up the real reference vector R and hence mimics the functionality of G05EEF. Setting MODE $= 1$ generates a series of variates from a reference vector mimicking the functionality of G05EYF for this particular distribution. Setting MODE $= 2$ initializes the reference vector and generates the variates in one go.

The routine G05EYF returns a single variate at a time, whereas the new routine G05THF returns a vector of N values in one go.

The length of the real reference vector, R, in G05THF, needs to be a different length from the equivalent argument in G05EEF, see the documentation for G05THF for more details.

The integer array STATE in the call to G05THF contains information on the base generator being used. This array must have been initialized prior to calling G05THF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05THF is likely to be different from those produced by a combination of G05EEF and G05EYF.

G05EFF

Withdrawn at Mark 22.
Replaced by G05TEF.

```
Old: CALL G05EFF(NS,M,NP,R1,LR1,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EYF(R1,LR1)
  10 CONTINUE
New: MODE = 2
     CALL G05TEF(MODE,N,NS,NP,M,R,LR,STATE,X,IFAIL)
```

The old routine G05EFF sets up a reference vector for use by G05EYF. The replacement routine G05TEF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05TEF sets up the real reference vector R and hence mimics the functionality of G05EFF. Setting MODE $= 1$ generates a series of variates from a reference vector mimicking the functionality of G05EYF for this particular distribution. Setting MODE $= 2$ initializes the reference vector and generates the variates in one go.

The routine G05EYF returns a single variate at a time, whereas the new routine G05TEF returns a vector of N values in one go.

The length of the real reference vector, R, in G05TEF, needs to be a different length from the equivalent argument in G05EFF, see the documentation for more details.

The integer array STATE in the call to G05TEF contains information on the base generator being used. This array must have been initialized prior to calling G05TEF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TEF is likely to be different from those produced by a combination of G05EFF and G05EYF.

G05EGF

Withdrawn at Mark 22.
Replaced by G05PHF.

```
Old: CALL G05EGF(E,A,NA,B,NB,R,NR,VAR,IFAIL)
New: AVAR = B(1)**2
     IQ = NB - 1
     IF (AVAR.GT.0.0D0) THEN
        DO 10 I = 1, IQ
           THETA(I) = -B(I+1)/B(1)
  10    CONTINUE
     ELSE
        DO 20 I = 1, IQ
           THETA(I) = 0.0D0
  20    CONTINUE
     END IF
     MODE = 0
     CALL G05PHF(MODE,N,E,NA,A,IQ,THETA,AVAR,R,LR,STATE,VAR,X,IFAIL)
```

The real vector THETA must be of length at least IQ $=$ NB $-$ 1.

The old routine G05EGF sets up a reference vector for use by G05EWF. The replacement routine G05PHF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05PHF sets up the real reference vector R and hence mimics the functionality of G05EGF. When MODE $= 0$, the integer array STATE in the call to G05PHF need not be set.

G05EHF

Withdrawn at Mark 22.
Replaced by G05NCF.

```
Old: CALL G05EHF(INDEX,N,IFAIL)
New: CALL G05NCF(INDEX,N,STATE,IFAIL)
```

The integer array STATE in the call to G05NCF contains information on the base generator being used. This array must have been initialized prior to calling G05NCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05NCF is likely to be different from those produced by G05EHF.

G05EJF

Withdrawn at Mark 22.
Replaced by G05NDF.

```
Old: CALL G05EJF(IA,N,IZ,M,IFAIL)
New: CALL G05NDF(IA,N,IZ,M,STATE,IFAIL)
```

The integer array STATE in the call to G05NDF contains information on the base generator being used. This array must have been initialized prior to calling G05NDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05NDF is likely to be different from those produced by G05EJF.

G05EWF

Withdrawn at Mark 22.
Replaced by G05PHF.

```
Old: CALL G05EGF(E,A,NA,B,NB,R,NR,VAR,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EWF(R,NR,IFAIL)
  10 CONTINUE
New: AVAR = B(1)**2
     IQ = NB - 1
     IF (AVAR.GT.0.0D0) THEN
        DO 10 I = 1, IQ
           THETA(I) = -B(I+1)/B(1)
  10    CONTINUE
     ELSE
        DO 20 I = 1, IQ
           THETA(I) = 0.0D0
  20    CONTINUE
     END IF
     MODE = 2
     CALL G05PHF(MODE,N,E,NA,A,NB-1,THETA,AVAR,VAR,R,LR,STATE,X,IFAIL)
```

The real vector THETA must be of length at least $\mathrm{IQ} = \mathrm{NB} - 1$.

The old routine G05EGF sets up a reference vector for use by G05EWF. The replacement routine G05PHF is now used to both set up a reference vector and generate the required variates. Setting the integer argument MODE to 0 in the call to G05PHF sets up the real reference vector R and hence mimics the functionality of G05EGF. Setting MODE to 1 generates a series of variates from a reference vector mimicking the functionality of G05EWF. Setting MODE to 2 initializes the reference vector and generates the variates in one go.

The integer array STATE in the call to G05PHF contains information on the base generator being used. This array must have been initialized prior to calling G05PHF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PHF is likely to be different from those produced by G05EGF.

G05EXF

Withdrawn at Mark 22.
Replaced by G05TDF.

```
Old: CALL G05EXF(P,NP,IP1,ITYPE,R1,LR1,IFAIL)
     DO 10 I = 1, N
        X(I) = G05EYF(R1,LR1)
  10 CONTINUE
New: MODE = 2
     CALL G05TDF(MODE,N,P,NP,IP1,ITYPE,R,LR,STATE,X,IFAIL)
```

The old routine G05EXF sets up a reference vector for use by G05EYF. The replacement routine G05TDF is now used to both set up a reference vector and generate the required variates. Setting MODE $= 0$ in the call to G05TDF sets up the real reference vector R and hence mimics the functionality of G05EXF. Setting MODE $= 1$ generates a series of variates from a reference vector mimicking the functionality of G05EYF for this particular distribution. Setting MODE $= 2$ initializes the reference vector and generates the variates in one go.

The routine G05EYF returns a single variate at a time, whereas the new routine G05TDF returns a vector of N values in one go.

The length of the real reference vector, R, in G05TDF must be allocated in the calling program in contrast to the equivalent argument in G05EXF, see the documentation for more details.

The integer array STATE in the call to G05TDF contains information on the base generator being used. This array must have been initialized prior to calling G05TDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05TDF is likely to be different from those produced by a combination of G05EXF and G05EYF.

G05EYF

Withdrawn at Mark 22.
Replaced by G05TDF.

There is no direct replacement routine for G05EYF.

G05EYF is designed to generate random draws from a distribution defined by a reference vector. These reference vectors are created by other routines in Chapter G05, for example G05EBF, which have themselves been superseded. In order to replace a call to G05EYF you must identify which NAG routine generated the reference vector being used and look up its replacement. For example, to replace a call to G05EYF preceded by a call to G05EBF, as in:

```
CALL G05EBF(M,IB,R,NR,IFAIL)
X = G05EYF(R,NR)
```

you would need to look at the replacement routine for G05EBF.

G05EZF

Withdrawn at Mark 22.
Replaced by G05RZF.

```
Old: CALL G05EAF(XMU,N,C,LDC,EPS,R1,LR1,IFAIL)
     DO 20 I = 1, N
        CALL G05EZF(CX,M,R,NR,IFAIL)
        DO 30 J = 1, M
           X(I,J) = CX(J)
  30    CONTINUE
  20 CONTINUE
New: MODE = 2
     CALL G05RZF(MODE,N,M,XMU,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The old routine G05EAF sets up a reference vector for use by G05EZF. The functionality of both these routines has been combined into the single new routine G05RZF. Setting MODE $= 2$ in the call to G05RZF sets up the real reference vector R and generates the draws from the multivariate Normal distribution in one go.

The old routine G05EZF returns a single (M-dimensional vector) draw from the multivariate Normal distribution at a time, whereas the new routine G05RZF returns an N by M matrix of N draws in one go.

The integer array STATE in the call to G05RZF contains information on the base generator being used. This array must have been initialized prior to calling G05RZF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05RZF is likely to be different from those produced by G05EZF.

G05FAF

Withdrawn at Mark 22.
Replaced by G05SQF.

```
Old: CALL G05FAF(AA,BB,N,X)
New: A = MIN(AA,BB)
     B = MAX(AA,BB)
     IFAIL = 0
     CALL G05SQF(N,A,B,STATE,X,IFAIL)
```

In G05SQF the minimum value must be held in the argument A and the maximum in argument B, therefore $A \leq B$. This was not the case for the equivalent arguments in G05FAF.

The integer array STATE in the call to G05SQF contains information on the base generator being used. This array must have been initialized prior to calling G05SQF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SQF is likely to be different from those produced by G05FAF.

G05FBF

Withdrawn at Mark 22.
Replaced by G05SFF.

```
Old: CALL G05FBF(AA,N,X)
New: A = ABS(AA)
     IFAIL = 0
     CALL G05SFF(N,A,STATE,X,IFAIL)
```

In G05SFF argument A must be non-negative, this was not the case for the equivalent argument in G05FBF.

The integer array STATE in the call to G05SFF contains information on the base generator being used. This array must have been initialized prior to calling G05SFF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SFF is likely to be different from those produced by G05FBF.

G05FDF

Withdrawn at Mark 22.
Replaced by G05SKF.

```
Old: CALL G05FDF(XMU,SD,N,X)
New: VAR = SD**2
     IFAIL = 0
     CALL G05SKF(N,XMU,VAR,STATE,X,IFAIL)
```

G05SKF expects the variance of the Normal distribution (argument VAR), compared to G05FDF which expected the standard deviation.

The integer array STATE in the call to G05SKF contains information on the base generator being used. This array must have been initialized prior to calling G05SKF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during

initialization. Due to changes in the underlying code the sequence of values produced by G05SKF is likely to be different from those produced by G05FDF.

G05FEF

Withdrawn at Mark 22.
Replaced by G05SBF.

```
Old: CALL G05FEF(A,B,N,X,IFAIL)
New: CALL G05SBF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SBF contains information on the base generator being used. This array must have been initialized prior to calling G05SBF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SBF is likely to be different from those produced by G05FEF.

G05FFF

Withdrawn at Mark 22.
Replaced by G05SJF.

```
Old: CALL G05FFF(A,B,N,X,IFAIL)
New: CALL G05SJF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SJF contains information on the base generator being used. This array must have been initialized prior to calling G05SJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SJF is likely to be different from those produced by G05FFF.

G05FSF

Withdrawn at Mark 22.
Replaced by G05SRF.

```
Old: CALL G05FSF(VK,N,X,IFAIL)
New: CALL G05SRF(N,VK,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SRF contains information on the base generator being used. This array must have been initialized prior to calling G05SRF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05SRF is likely to be different from those produced by G05FSF.

G05GAF

Withdrawn at Mark 22.
Replaced by G05PXF.

```
Old: CALL G05GAF(SIDE,INIT,M,N,A,LDA,WK,IFAIL)
New: CALL G05PXF(SIDE,INIT,M,N,STATE,A,LDA,IFAIL)
```

The integer array STATE in the call to G05PXF contains information on the base generator being used. This array must have been initialized prior to calling G05PXF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PXF is likely to be different from those produced by G05GAF.

G05GBF

Withdrawn at Mark 22.
Replaced by G05PYF.

```
Old: CALL G05GBF(N,D,C,LDC,EPS,WK,IFAIL)
New: CALL G05PYF(N,D,EPS,STATE,C,LDC,IFAIL)
```

The integer array STATE in the call to G05PYF contains information on the base generator being used. This array must have been initialized prior to calling G05PYF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PYF is likely to be different from those produced by G05GBF.

G05HDF

Withdrawn at Mark 22.
Replaced by G05PJF.

```
Old: CALL G05HDF(MODE,K,IP,IQ,MEAN,PAR,LPAR,QQ,LDQQ,N,W,REF,LREF,  &
                 IWORK,LIWORK,IFAIL)
New: IF (MODE.EQ.'S') THEN
        IMODE = 0
     ELSE IF (MODE.EQ.'C') THEN
        IMODE = 1
     ELSE IF (MODE.EQ.'R') THEN
        IMODE = 3
     END IF
     LL = 0
     DO 30 L = 1, IP
        DO 20 I = 1, K
           DO 10 J = 1, K
              LL = LL + 1
              PHI(I,J,L) = PAR(LL)
  10       CONTINUE
  20    CONTINUE
  30 CONTINUE
     DO 60 L = 1, IQ-1
        DO 50 I = 1, K
           DO 40 J = 1, K
              LL = LL + 1
              THETA(I,J,L) = PAR(LL)
  40       CONTINUE
  50    CONTINUE
  60 CONTINUE
     IF (MEAN.EQ.'M') THEN
        DO 70 I = 1, K
           LL = LL + 1
           XMEAN(I) = PAR(LL)
  70    CONTINUE
     ELSE
        DO 80 I = 1, K
           XMEAN(I) = 0.0D0
  80    CONTINUE
     END IF
     LDW = N
     CALL G05PJF(IMODE,N,K,XMEAN,IP,PHI,IQ,THETA,QQ,LDQQ,REF,LREF, &
                 STATE,W,LDW,IWORK,LIWORK,IFAIL)
```

The integer argument IMODE should be set to 0, 1 or 3 in place of the argument MODE having settings of `S', `C' or `R' respectively. The real array PHI should have length at least $\max(1, \mathrm{IP} \times (\mathrm{K} \times \mathrm{K}))$; if dimensioned as $\mathrm{PHI}(\mathrm{K}, \mathrm{K}, \mathrm{IP})$ (as in the above example) then $\mathrm{PHI}(i, j, l)$ will contain the element $\mathrm{PAR}((l-1) \times k \times k + (i-1) \times k + j)$. The real array THETA should have length at least $\max(1, \mathrm{IQ} \times (\mathrm{K} \times \mathrm{K}))$; if dimensioned as $\mathrm{THETA}(\mathrm{K}, \mathrm{K}, \mathrm{IQ})$ (as in the above example) then $\mathrm{THETA}(i, j, l)$ will contain the element $\mathrm{PAR}(\mathrm{IP} \times k \times k + (l-1) \times k \times k + (i-1) \times k + j)$. The real array XMEAN should have length at least K; if MEAN = 'M' then $\mathrm{XMEAN}(i)$ will contain the element $\mathrm{PAR}(\mathrm{IP} + \mathrm{IQ} \times k \times k + i)$, otherwise XMEAN should contain an array of zero values.

The integer array STATE in the call to G05PJF contains information on the base generator being used. This array must have been initialized prior to calling G05PJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PJF is likely to be different from those produced by G05HDF.

G05HKF

Withdrawn at Mark 24.
Replaced by G05PDF.

```
Old: CALL G05HKF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,RVEC,IGEN,  &
                 ISEED,RWSAV,IFAIL)
New: CALL G05PDF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,R,LR,STATE, &
                 IFAIL)
```

The integer array STATE in the call to G05PDF contains information on the base generator being used. This array must have been initialized prior to calling G05PDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PDF is likely to be different from those produced by G05HKF.

G05HLF

Withdrawn at Mark 24.
Replaced by G05PEF.

```
Old: CALL G05HLF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,RVEC,IGEN,  &
                 ISEED,RWSAV,IFAIL)
New: CALL G05PEF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,R,LR,STATE, &
                 IFAIL)
```

The integer array STATE in the call to G05PEF contains information on the base generator being used. This array must have been initialized prior to calling G05PEF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PEF is likely to be different from those produced by G05HLF.

G05HMF

Withdrawn at Mark 24.
Replaced by G05PFF.

```
Old: CALL G05HMF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,RVEC,IGEN,  &
                 ISEED,RWSAV,IFAIL)
New: CALL G05PFF(DIST,NUM,IP,IQ,THETA,GAMMA,DF,HT,ET,FCALL,R,LR,STATE, &
                 IFAIL)
```

The integer array STATE in the call to G05PFF contains information on the base generator being used. This array must have been initialized prior to calling G05PFF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization. Due to changes in the underlying code the sequence of values produced by G05PFF is likely to be different from those produced by G05HMF.

G05HNF

Withdrawn at Mark 24.
Replaced by G05PGF.

```
Old: CALL G05HNF(DIST,NUM,IP,IQ,THETA,DF,HT,ET,FCALL,RVEC,IGEN,ISEED, &
                 RWSAV,IFAIL)
New: CALL G05PGF(DIST,NUM,IP,IQ,THETA,DF,HT,ET,FCALL,RVEC,STATE,      &
                 IFAIL)
```

The integer array STATE in the call to G05PGF contains information on the base generator being used. This array must have been initialized prior to calling G05PGF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05KAF

Withdrawn at Mark 24.
Replaced by G05SAF.

```
Old: DO 20 I = 1, N
        X(I) = G05KAF(IGEN,ISEED)
  20 CONTINUE
New: CALL G05SAF(N,STATE,X,IFAIL)
```

The old routine G05KAF returns a single variate at a time, whereas the new routine G05SAF returns a vector of N values in one go.

The integer array STATE in the call to G05SAF contains information on the base generator being used. This array must have been initialized prior to calling G05SAF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05KBF

Withdrawn at Mark 24.
Replaced by G05KFF.

```
Old: G05KBF(IGEN,ISEED)
New: IF (IGEN.EQ.0) THEN
        CALL G05KFF(1,1,ISEED,LSEED,STATE,LSTATE,IFAIL)
     ELSE
        CALL G05KFF(2,IGEN,ISEED,LSEED,STATE,LSTATE,IFAIL)
     END IF
```

G05KCF

Withdrawn at Mark 24.
Replaced by G05KGF.

```
Old: CALL G05KCF(IGEN,ISEED)
New: IF (IGEN.EQ.0) THEN
        CALL G05KGF(1,1,STATE,LSTATE,IFAIL)
     ELSE
        CALL G05KGF(2,IGEN,STATE,LSTATE,IFAIL)
     END IF
```

G05KEF

Withdrawn at Mark 24.
Replaced by G05TBF.

```
Old: DO 20 I = 1, N
        X(I) = G05KEF(P,IGEN,ISEED,IFAIL)
  20 CONTINUE
New: CALL G05TBF(N,P,STATE,X,IFAIL)
```

The old routine G05KEF returns a single variate at a time, whereas the new routine G05TBF returns a vector of N values in one go.

The integer array STATE in the call to G05TBF contains information on the base generator being used. This array must have been initialized prior to calling G05TBF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LAF

Withdrawn at Mark 24.
Replaced by G05SKF.

```
Old: CALL G05LAF(XMU,VAR,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SKF(N,XMU,VAR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SKF contains information on the base generator being used. This array must have been initialized prior to calling G05SKF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LBF

Withdrawn at Mark 24.
Replaced by G05SNF.

```
Old: CALL G05LBF(DF,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SNF(N,DF,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SNF contains information on the base generator being used. This array must have been initialized prior to calling G05SNF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LCF

Withdrawn at Mark 24.
Replaced by G05SDF.

```
Old: CALL G05LCF(DF,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SDF(N,DF,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SDF contains information on the base generator being used. This array must have been initialized prior to calling G05SDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LDF

Withdrawn at Mark 24.
Replaced by G05SHF.

```
Old: CALL G05LDF(DF1,DF2,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SHF(N,DF1,DF2,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SHF contains information on the base generator being used. This array must have been initialized prior to calling G05SHF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LEF

Withdrawn at Mark 24.
Replaced by G05SBF.

```
Old: CALL G05LEF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SBF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SBF contains information on the base generator being used. This array must have been initialized prior to calling G05SBF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LFF

Withdrawn at Mark 24.
Replaced by G05SJF.

```
Old: CALL G05LFF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SJF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SJF contains information on the base generator being used. This array must have been initialized prior to calling G05SJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LGF

Withdrawn at Mark 24.
Replaced by G05SQF.

```
Old: CALL G05LGF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SQF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SQF contains information on the base generator being used. This array must have been initialized prior to calling G05SQF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LHF

Withdrawn at Mark 24.
Replaced by G05SPF.

```
Old: CALL G05LHF(XMIN,XMAX,XMED,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SPF(N,XMIN,XMED,XMAX,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SPF contains information on the base generator being used. This array must have been initialized prior to calling G05SPF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LJF

Withdrawn at Mark 24.
Replaced by G05SFF.

```
Old: CALL G05LJF(A,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SFF(N,A,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SFF contains information on the base generator being used. This array must have been initialized prior to calling G05SFF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LKF

Withdrawn at Mark 24.
Replaced by G05SMF.

```
Old: CALL G05LKF(XMU,VAR,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SMF(N,XMU,VAR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SMF contains information on the base generator being used. This array must have been initialized prior to calling G05SMF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LLF

Withdrawn at Mark 24.
Replaced by G05SJF.

```
Old: CALL G05LLF(XMED,SEMIQR,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SCF(N,XMED,SEMIQR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SCF contains information on the base generator being used. This array must have been initialized prior to calling G05SCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LMF

Withdrawn at Mark 24.
Replaced by G05SSF.

```
Old: CALL G05LMF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SSF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SSF contains information on the base generator being used. This array must have been initialized prior to calling G05SSF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LNF

Withdrawn at Mark 24.
Replaced by G05SLF.

```
Old: CALL G05LNF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SLF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SLF contains information on the base generator being used. This array must have been initialized prior to calling G05SLF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LPF

Withdrawn at Mark 24.
Replaced by G05SRF.

```
Old: CALL G05LPF(VK,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SRF(N,VK,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SRF contains information on the base generator being used. This array must have been initialized prior to calling G05SRF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LQF

Withdrawn at Mark 24.
Replaced by G05SGF.

```
Old: CALL G05LQF(NMIX,A,WGT,N,X,IGEN,ISEED,IFAIL)
New: CALL G05SGF(N,NMIX,A,WGT,STATE,X,IFAIL)
```

The integer array STATE in the call to G05SGF contains information on the base generator being used. This array must have been initialized prior to calling G05SGF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LXF

Withdrawn at Mark 24.
Replaced by G05RYF.

```
Old: CALL G05LXF(MODE,DF,M,XMU,C,LDC,N,X,LDX,IGEN,ISEED,R,LR,IFAIL)
New: IF (MODE == 0) THEN
       NMODE = 1
     ELSE IF (MODE == 1) THEN
       NMODE = 0
     ELSE
       NMODE = MODE
     END IF
     CALL G05RYF(NMODE,N,DF,M,XMU,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05RYF contains information on the base generator being used. This array must have been initialized prior to calling G05RYF with a call to either G05KFF or

G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LYF

Withdrawn at Mark 24.
Replaced by G05RZF.

```
Old: G05LYF(MODE,M,XMU,C,LDC,N,X,LDX,IGEN,ISEED,R,LR,IFAIL)
New: IF (MODE == 0) THEN
       NMODE = 1
     ELSE IF (MODE == 1) THEN
       NMODE = 0
     ELSE
       NMODE = MODE
     END IF
     CALL G05RZF(NMODE,N,M,XMU,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05RZF contains information on the base generator being used. This array must have been initialized prior to calling G05RZF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05LZF

Withdrawn at Mark 24.
Replaced by G05RZF.

```
Old: CALL G05LZF(MODE,M,XMU,C,LDC,X,IGEN,ISEED,R,LR,IFAIL)
New: IF (MODE == 0) THEN
       NMODE = 1
     ELSE IF (MODE == 1) THEN
       NMODE = 0
     ELSE
       NMODE = MODE
     END IF
     CALL G05RZF(NMODE,N,M,XMU,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05RZF contains information on the base generator being used. This array must have been initialized prior to calling G05RZF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MAF

Withdrawn at Mark 24.
Replaced by G05TLF.

```
Old: CALL G05MAF(A,B,N,X,IGEN,ISEED,IFAIL)
New: CALL G05TLF(N,A,B,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TLF contains information on the base generator being used. This array must have been initialized prior to calling G05TLF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MBF

Withdrawn at Mark 24.
Replaced by G05TCF.

```
Old: CALL G05MBF(MODE,P,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TCF(MODE,N,P,R,LR,STATE,X,IFAIL)
     DO 20 I = 1, N
        X(I) = X(I) + 1
  20 CONTINUE
```

G05MBF returned the number of trials required to get the first success, whereas G05TCF returns the number of failures before the first success, therefore the value returned by G05TCF is one less than the equivalent value returned from G05MBF.

The integer array STATE in the call to G05TCF contains information on the base generator being used. This array must have been initialized prior to calling G05TCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MCF

Withdrawn at Mark 24.
Replaced by G05THF.

```
Old: CALL G05MCF(MODE,M,P,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05THF(MODE,N,M,P,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05THF contains information on the base generator being used. This array must have been initialized prior to calling G05THF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MDF

Withdrawn at Mark 24.
Replaced by G05TFF.

```
Old: CALL G05MDF(MODE,A,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TFF(MODE,N,A,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TFF contains information on the base generator being used. This array must have been initialized prior to calling G05TFF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MEF

Withdrawn at Mark 24.
Replaced by G05TKF.

```
Old: CALL G05MEF(M,VLAMDA,X,IGEN,ISEED,IFAIL)
New: CALL G05TKF(M,VLAMDA,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TKF contains information on the base generator being used. This array must have been initialized prior to calling G05TKF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MJF

Withdrawn at Mark 24.
Replaced by G05TAF.

```
Old: CALL G05MJF(MODE,M,P,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TAF(MODE,N,M,P,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TAF contains information on the base generator being used. This array must have been initialized prior to calling G05TAF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MKF

Withdrawn at Mark 24.
Replaced by G05TJF.

```
Old: CALL G05MKF(MODE,LAMBDA,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TJF(MODE,N,LAMBDA,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TJF contains information on the base generator being used. This array must have been initialized prior to calling G05TJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MLF

Withdrawn at Mark 24.
Replaced by G05TEF.

```
Old: CALL G05MLF(MODE,NS,NP,M,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TEF(MODE,N,NS,NP,M,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TEF contains information on the base generator being used. This array must have been initialized prior to calling G05TEF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MRF

Withdrawn at Mark 24.
Replaced by G05TGF.

```
Old: CALL G05MRF(MODE,M,K,P,N,X,LDX,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TGF(MODE,N,M,K,P,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05TGF contains information on the base generator being used. This array must have been initialized prior to calling G05TGF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05MZF

Withdrawn at Mark 24.
Replaced by G05TDF.

```
Old: CALL G05MZF(MODE,P,NP,IP1,ITYPE,N,X,IGEN,ISEED,R,NR,IFAIL)
New: CALL G05TDF(MODE,N,P,NP,IP1,ITYPE,R,LR,STATE,X,IFAIL)
```

The integer array STATE in the call to G05TDF contains information on the base generator being used. This array must have been initialized prior to calling G05TDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05NAF

Withdrawn at Mark 24.
Replaced by G05NCF.

```
Old: CALL G05NAF(INDEX,N,IGEN,ISEED,IFAIL)
New: CALL G05NCF(INDEX,N,STATE,IFAIL)
```

The integer array STATE in the call to G05NCF contains information on the base generator being used. This array must have been initialized prior to calling G05NCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05NBF

Withdrawn at Mark 24.
Replaced by G05NDF.

```
Old: CALL G05NBF(IPOP,N,ISAMPL,M,IGEN,ISEED,IFAIL)
New: CALL G05NDF(IPOP,N,ISAMPL,M,STATE,IFAIL)
```

The integer array STATE in the call to G05NDF contains information on the base generator being used. This array must have been initialized prior to calling G05NDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05PAF

Withdrawn at Mark 24.
Replaced by G05PHF.

```
Old: CALL G05PAF(MODE,XMEAN,IP,PHI,IQ,THETA,AVAR,VAR,N,X,IGEN,ISEED,R, &
                 NR,IFAIL)
New: CALL G05PHF(MODE,N,XMEAN,IP,PHI,IQ,THETA,AVAR,R,LR,STATE,VAR,X,   &
                 IFAIL)
```

The integer array STATE in the call to G05PHF contains information on the base generator being used. This array must have been initialized prior to calling G05PHF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05PCF

Withdrawn at Mark 24.
Replaced by G05PJF.

```
Old: CALL G05PCF(MODE,K,XMEAN,IP,PHI,IQ,THETA,VAR,LDV,N,X,IGEN,ISEED,R,   &
                 NR,IWORK,LIWORK,IFAIL)
New: CALL G05PJF(MODE,N,K,XMEAN,IP,PHI,IQ,THETA,VAR,LDV,R,LR,STATE,X,LDX, &
                 IFAIL)
```

The integer array STATE in the call to G05PJF contains information on the base generator being used. This array must have been initialized prior to calling G05PJF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05QAF

Withdrawn at Mark 24.
Replaced by G05PXF.

```
Old: CALL G05QAF(SIDE,INIT,M,N,A,LDA,IGEN,ISEED,WK,IFAIL)
New: CALL G05PXF(SIDE,INIT,M,N,STATE,A,LDA,IFAIL)
```

The integer array STATE in the call to G05PXF contains information on the base generator being used. This array must have been initialized prior to calling G05PXF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05QBF

Withdrawn at Mark 24.
Replaced by G05PYF.

```
Old: CALL G05QBF(N,D,C,LDC,EPS,IGEN,ISEED,WK,IFAIL)
New: CALL G05PYF(N,D,EPS,STATE,C,LDC,IFAIL)
```

The integer array STATE in the call to G05PYF contains information on the base generator being used. This array must have been initialized prior to calling G05PYF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05QDF

Withdrawn at Mark 24.
Replaced by G05PZF.

```
Old: CALL G05QDF(MODE,NROW,NCOL,TOTR,TOTC,X,LDX,IGEN,ISEED,R,NR,IW,LIW, &
                 IFAIL)
New: CALL G05PZF(MODE,NROW,NCOL,TOTR,TOTC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05PZF contains information on the base generator being used. This array must have been initialized prior to calling G05PZF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05RAF

Withdrawn at Mark 24.
Replaced by G05RDF.

```
Old: CALL G05RAF(MODE,M,C,LDC,N,X,LDX,IGEN,ISEED,R,LR,IFAIL)
New: IF (MODE == 0) THEN
       NMODE = 1
     ELSE IF (MODE == 1) THEN
       NMODE = 0
     ELSE
       NMODE = MODE
     END IF
     CALL CALL G05RDF(NMODE,N,M,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05RDF contains information on the base generator being used. This array must have been initialized prior to calling G05RDF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05RBF

Withdrawn at Mark 24.
Replaced by G05RCF.

```
Old: CALL G05RBF(MODE,DF,M,C,LDC,N,X,LDX,IGEN,ISEED,R,LR,IFAIL)
New: IF (MODE == 0) THEN
       NMODE = 1
     ELSE IF (MODE == 1) THEN
       NMODE = 0
     ELSE
       NMODE = MODE
     END IF
     CALL CALL G05RCF(NMODE,N,DF,M,C,LDC,R,LR,STATE,X,LDX,IFAIL)
```

The integer array STATE in the call to G05RCF contains information on the base generator being used. This array must have been initialized prior to calling G05RCF with a call to either G05KFF or G05KGF. The required length of the array STATE will depend on the base generator chosen during initialization.

G05YAF

Withdrawn at Mark 23.
Replaced by G05YLF and G05YMF.

Faure quasi-random numbers

```
Old: CALL G05YAF(.TRUE.,'F',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YLF(4,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YAF(.FALSE.,'F',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YMF(1,2,QUAS,LDQUAS,IREF,IFAIL)
```

Sobol quasi-random numbers

```
Old: CALL G05YAF(.TRUE.,'S',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YLF(2,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YAF(.FALSE.,'S',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YMF(1,2,QUAS,LDQUAS,IREF,IFAIL)
```

Neiderreiter quasi-random numbers

```
Old: CALL G05YAF(.TRUE.,'N',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YLF(3,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YAF(.FALSE.,'N',ISKIP,IDIM,QUAS,IREF,IFAIL)
New: CALL G05YMF(1,2,QUAS,LDQUAS,IREF,IFAIL)
```

G05YBF

Withdrawn at Mark 23.
Replaced by G05YLF and either G05YJF or G05YKF.

This routine has been replaced by a suite of routines consisting of the relevant initialization routine followed by one of two possible generator routines.

Faure quasi-random numbers with Gaussian probability:

```
Old: CALL G05YBF(.TRUE.,'F',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(4,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'F',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YJF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

Sobol quasi-random numbers with Gaussian probability:

```
Old: CALL G05YBF(.TRUE.,'S',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(2,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'S',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YJF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

Neiderreiter quasi-random numbers with Gaussian probability:

```
Old: CALL G05YBF(.TRUE.,'N',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(3,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'N',.FALSE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YJF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

Faure quasi-random numbers with log Normal probability:

```
Old: CALL G05YBF(.TRUE.,'F',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(4,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'F',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YKF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

Sobol quasi-random numbers with log Normal probability:

```
Old: CALL G05YBF(.TRUE.,'S',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(2,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'S',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YKF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

Neiderreiter quasi-random numbers with log Normal probability:

```
Old: CALL G05YBF(.TRUE.,'N',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YLF(3,IDIM,IREF,LIREF,ISKIP,IFAIL)

Old: CALL G05YBF(.FALSE.,'N',.TRUE.,MEAN,STD,ISKIP,IDIM,QUASI,IREF,IFAIL)
New: CALL G05YKF(MEAN,STD,N,QUASI,IREF,IFAIL)
```

G05YCF

Withdrawn at Mark 24.
Replaced by G05YLF.

```
Old: CALL G05YCF(IDIM,IREF,IFAIL)
New: GENID = 4
     CALL G05YLF(GENID,IDIM,IREF,LIREF,ISKIP,IFAIL)
```

G05YDF

Withdrawn at Mark 24.
Replaced by G05YMF.

```
Old: CALL G05YDF(N,QUASI,IREF,IFAIL)
New: CALL G05YMF(N,QUAS,LDQUAS,IREF,IFAIL)
```

G05YEF

Withdrawn at Mark 24.
Replaced by G05YLF.

```
Old: CALL G05YEF(IDIM, IREF, ISKIP, IFAIL)
New: GENID = 2
     CALL G05YLF(GENID,IDIM,IREF,LIREF,ISKIP,IFAIL)
```

G05YFF

Withdrawn at Mark 24.
Replaced by G05YMF.

```
Old: CALL G05YFF(N,QUASI,IREF,IFAIL)
New: CALL G05YMF(N,QUAS,LDQUAS,IREF,IFAIL)
```

G05YGF

Withdrawn at Mark 24.
Replaced by G05YLF.

```
Old: CALL G05YGF(IDIM,IREF,ISKIP,IFAIL)
New: GENID = 3
     CALL G05YLF(GENID,IDIM,IREF,LIREF,ISKIP,IFAIL)
```

G05YHF

Withdrawn at Mark 24.
Replaced by G05YMF.

```
Old: CALL G05YHF(N,QUASI,IREF,IFAIL)
New: CALL G05YMF(N,RCORD,QUAS,LDQUAS,IREF,IFAIL)
```

G05ZAF

Withdrawn at Mark 22.
There is no replacement for this routine.

G05ZAF was used to select the underlying generator for the old style random number generation routines. These routines are no longer available and hence no direct replacement routine for G05ZAF is

required. See G05KFF and G05KGF for details on how to select the underlying generator for the newer style routines.

G10 – Smoothing in Statistics

G10BAF

Scheduled for withdrawal at Mark 27.
Replaced by G10BBF.

Withdrawn primarily due to threadsafety. The replacement routine alsointroduces new functionality with respect to the automatic selection of a suitable window width.

```
Old: CALL G10BAF(N,X,WINDOW,SLO,SHI,NS,SMOOTH,T,USEFFT,FFT,IFAIL)

New: ALLOCATE(RCOMM,NS+20)
     CALL G10BBF(N,X,1,WINDOW,SLO,SHI,NS,SMOOTH,T,USEFFT,RCOMM,IFAIL)
     ! the next step is only required if the information in FFT
     ! was being used outside another call to G10BAF
     FFT(1:NS) = RCOMM(21:NS+20)
```

G13 – Time Series Analysis

G13DCF

Withdrawn at Mark 24.
Replaced by G13DDF.

```
Old: CALL G13DCF(K,N,IP,IQ,MEAN,PAR,NPAR,QQ,KMAX,W,PARHLD,EXACT,IPRINT,  &
                 CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,CM,LDCM,WORK,LWORK, &
                 IW,LIW,IFAIL)
New: CALL G13DDF(K,N,IP,IQ,MEAN,PAR,NPAR,QQ,KMAX,W,PARHLD,EXACT,IPRINT,  &
                 CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,CM,LDCM,IFAIL)
```

The workspace arguments WORK, LWORK, IW and LIW are no longer required in the call to G13DDF.

P01 – Error Trapping

P01ABF

Withdrawn at Mark 24.
There is no replacement for this routine.

X02 – Machine Constants

X02DAF

Withdrawn at Mark 24.
There is no replacement for this routine.

X02DJF

Withdrawn at Mark 24.
There is no replacement for this routine.