NAG Library Routine Document

E04DJF/E04DJA

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

To supply optional parameters to E04DGF/E04DGA from an external file. More precisely, E04DJF must be used to supply optional parameters to E04DGF and E04DJA must be used to supply optional parameters to E04DGA.

E04DJA is a version of E04DJF that has additional arguments in order to make it safe for use in multithreaded applications (see Section 5). The initialization routine E04WBF **must** have been called before calling E04DJA.

2 Specification

2.1 Specification for E04DJF

```
SUBROUTINE E04DJF (IOPTNS, INFORM)
INTEGER IOPTNS, INFORM
```

2.2 Specification for E04DJA

```
SUBROUTINE E04DJA (IOPTNS, LWSAV, IWSAV, RWSAV, INFORM)
INTEGER            IOPTNS, IWSAV(610), INFORM
REAL (KIND=nag_wp) RWSAV(475)
LOGICAL            LWSAV(120)
```

3 Description

E04DJF/E04DJA may be used to supply values for optional parameters to E04DGF/E04DGA. E04DJF/E04DJA reads an external file and each line of the file defines a single optional parameter. It is only necessary to supply values for those arguments whose values are to be different from their default values.

Each optional parameter is defined by a single character string, of up to 72 characters, consisting of one or more items. The items associated with a given option must be separated by spaces, or equals signs [=]. Alphabetic characters may be upper or lower case. The string

```
Print Level = 1
```

is an example of a string used to set an optional parameter. For each option the string contains one or more of the following items:

– a mandatory keyword;

– a phrase that qualifies the keyword;

– a number that specifies an integer or real value. Such numbers may be up to 40 contiguous characters in Fortran's I, F, E or D formats, terminated by a space if this is not the last item on the line.

Blank strings and comments are ignored. A comment begins with an asterisk (*) and all subsequent characters in the string are regarded as part of the comment.

The file containing the options must start with `Begin` and must finish with `End`. An example of a valid options file is:

```
Begin * Example options file
    Print level = 5
 End
```

For E04DJF each line of the file is normally printed as it is read, on the current advisory message unit (see X04ABF), but printing may be suppressed using the keyword **Nolist**. To suppress printing of `Begin`, **Nolist** must be the first option supplied as in the file:

```
Begin
    Nolist
    Print level = 5
 End
```

Printing will automatically be turned on again after a call to E04DGF or E04DJF and may be turned on again at any time using the keyword **List**.

For E04DJA printing is turned off by default, but may be turned on at any time using the keyword **List**.

Optional parameter settings are preserved following a call to E04DGF/E04DGA and so the keyword **Defaults** is provided to allow you to reset all the optional parameters to their default values before a subsequent call to E04DGF/E04DGA.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04DGF/E04DGA.

4 References

None.

5 Arguments

1: IOPTNS – INTEGER *Input*

On entry: the unit number of the options file to be read.

Constraint: $0 \leq \text{IOPTNS} \leq 99$.

2: INFORM – INTEGER *Output*

Note: *for E04DJA, INFORM does not occur in this position in the argument list. See the additional arguments described below.*

On exit: contains zero if the options file has been successfully read and a value > 0 otherwise (see Section 6).

Note: *the following are additional arguments for specific use with E04DJA. Users of E04DJF therefore need not read the remainder of this description*.

3: LWSAV(120) – LOGICAL array *Communication Array*

4: IWSAV(610) – INTEGER array *Communication Array*

5: RWSAV(475) – REAL (KIND=nag_wp) array *Communication Array*

The arrays LWSAV, IWSAV and RWSAV **must not** be altered between calls to any of the routines E04DJA, E04DGA, E04DKA or E04WBF.

6: INFORM – INTEGER *Output*

Note: see the argument description for INFORM above.

6 Error Indicators and Warnings

INFORM $= 1$

IOPTNS is not in the range $[0, 99]$.

INFORM $= 2$

`Begin` was found, but end-of-file was found before `End` was found.

INFORM = 3

end-of-file was found before `Begin` was found.

INFORM = 4

Not used.

INFORM = 5

One or more lines of the options file is invalid. Check that all keywords are neither ambiguous nor misspelt.

7 Accuracy

Not applicable.

8 Parallelism and Performance

E04DJF/E04DJA is not threaded in any implementation.

9 Further Comments

E04DKF/E04DKA may also be used to supply optional parameters to E04DGF/E04DGA.

10 Example

This example solves the same problem as the example for E04DGF/E04DGA, but in addition illustrates the use of E04DJF/E04DJA and E04DKF/E04DKA to set optional parameters for E04DGF/E04DGA.

In this example the options file read by E04DJF/E04DJA is appended to the data file for the program (see Section 10.2). It would usually be more convenient in practice to keep the data file and the options file separate.

10.1 Program Text

the following program illustrates the use of E04DJF. An equivalent program illustrating the use of E04DJA is available with the supplied Library and is also available from the NAG web site.

```
!   E04DJF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.
    Module e04djfe_mod

!     E04DJF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                           :: objfn1
!     .. Parameters ..
      Integer, Parameter, Public       :: nin = 5, ninopt = 7, nout = 6
    Contains
      Subroutine objfn2(mode,n,x,objf,objgrd,nstate,iuser,ruser)
!       Routine to evaluate F(x)

!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: objf
        Integer, Intent (Inout)          :: mode
        Integer, Intent (In)             :: n, nstate
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: objgrd(n)
```

```
        Real (Kind=nag_wp), Intent (Inout) :: ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(n)
        Integer, Intent (Inout)        :: iuser(*)
!       .. Local Scalars ..
        Real (Kind=nag_wp)             :: x1, x2
!       .. Intrinsic Procedures ..
        Intrinsic                      :: exp
!       .. Executable Statements ..
        x1 = x(1)
        x2 = x(2)

        objf = exp(x1)*(4.0_nag_wp*x1**2+2.0_nag_wp*x2**2+4.0_nag_wp*x1*x2+    &
          2.0_nag_wp*x2+1.0_nag_wp)

        Return

      End Subroutine objfn2
      Subroutine objfn1(mode,n,x,objf,objgrd,nstate,iuser,ruser)
!       Routine to evaluate F(x) and approximate its 1st derivatives

!       .. Use Statements ..
        Use nag_library, Only: e04xaf
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: objf
        Integer, Intent (Inout)        :: mode
        Integer, Intent (In)           :: n, nstate
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: objgrd(n)
        Real (Kind=nag_wp), Intent (Inout) :: ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(n)
        Integer, Intent (Inout)        :: iuser(*)
!       .. Local Scalars ..
        Real (Kind=nag_wp)             :: epsrf
        Integer                        :: ifail, imode, iwarn, ldh, msglvl
!       .. Local Arrays ..
        Real (Kind=nag_wp), Allocatable :: h(:,:), hcntrl(:), hforw(:),        &
                                           work(:), xcopy(:)
        Integer, Allocatable           :: info(:)
!       .. Executable Statements ..
        Select Case (mode)
        Case (0)

!         Evaluate F(x) only

          Call objfn2(mode,n,x,objf,objgrd,nstate,iuser,ruser)

        Case (2)

!         Evaluate F(x) and approximate its 1st derivatives

          imode = 0
          ldh = n
          Allocate (info(n),hforw(n),hcntrl(n),h(ldh,1),work(n),xcopy(n))
          xcopy(1:n) = x(1:n)
          hforw(1:n) = 0.0_nag_wp
          msglvl = 0
          epsrf = 0.0_nag_wp

          ifail = 1
          Call e04xaf(msglvl,n,epsrf,xcopy,imode,objfn2,ldh,hforw,objf,objgrd, &
            hcntrl,h,iwarn,work,iuser,ruser,info,ifail)

        End Select

        Return

      End Subroutine objfn1
    End Module e04djfe_mod
    Program e04djfe

!     E04DJF Example Main Program
```

```
!     .. Use Statements ..
      Use nag_library, Only: e04dgf, e04djf, e04dkf, nag_wp, x04abf, x04acf,   &
                             x04baf
      Use e04djfe_mod, Only: nin, ninopt, nout, objfn1
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Character (*), Parameter         :: fname = 'e04djfe.opt'
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: objf
      Integer                          :: ifail, inform, iter, mode, n, outchn
      Character (80)                   :: rec
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: objgrd(:), work(:), x(:)
      Real (Kind=nag_wp)               :: ruser(1)
      Integer                          :: iuser(1)
      Integer, Allocatable             :: iwork(:)
!     .. Executable Statements ..
      Write (rec,99998) 'E04DJF Example Program Results'
      Call x04baf(nout,rec)

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) n
      Allocate (iwork(n+1),objgrd(n),x(n),work(13*n))

!     Set the unit number for advisory messages to OUTCHN

      outchn = nout
      Call x04abf(1,outchn)

      Read (nin,*) x(1:n)

!     Set two options using E04DKF

      Call e04dkf(' Verify Level = -1 ')

      Call e04dkf(' Maximum Step Length = 100.0 ')

!     Open the options file for reading

      mode = 0

      ifail = 0
      Call x04acf(ninopt,fname,mode,ifail)

!     Read the options file for the remaining options

      Call e04djf(ninopt,inform)

      If (inform/=0) Then
        Write (rec,99999) 'E04DJF terminated with INFORM = ', inform

        Call x04baf(nout,rec)

        Go To 100
      End If

!     Solve the problem

      ifail = -1
      Call e04dgf(n,objfn1,iter,objf,objgrd,x,iwork,work,iuser,ruser,ifail)

100   Continue

99999 Format (1X,A,I5)
99998 Format (1X,A)
    End Program e04djfe
```

10.2 Program Data

```
Begin   Example options file for  E04DJF
Iteration Limit  =  25                 *  (Default =  50)
Print Level      =   1                 *  (Default =  10)
End
```

```
E04DJF Example Program Data
  2                                    :Value of N
 -1.0  1.0                             :End of X
```

10.3 Program Results

```
 E04DJF Example Program Results


 Calls to E04DKF
 ---------------

       Verify Level = -1
       Maximum Step Length = 100.0


 OPTIONS file
 ------------

      Begin   Example options file for  E04DJF
      Iteration Limit  =  25                 *  (Default =  50)
      Print Level      =   1                 *  (Default =  10)
      End

 *** E04DGF

 Parameters
 ----------

 Variables..............         2

 Maximum step length....  1.00E+02       EPS (machine precision)  1.11E-16
 Optimality tolerance...  3.26E-12       Linesearch tolerance...  9.00E-01

 Est. opt. function val.      None       Function precision.....  4.37E-15
 Verify level...........        -1

 Iteration limit........        25       Print level............         1

 Exit from E04DGF after    10 iterations.


 Variable          Value      Gradient value
 Varbl    1     0.500000          9.1E-07
 Varbl    2    -1.00000           8.3E-07

 Exit E04DGF - Optimal solution found.

 Final objective value =   0.5235082E-13
```