

# NAG Library Routine Document

## E04VKF

**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

E04VKF may be used to supply optional parameters to E04VHF from an external file. The initialization routine E04VGF **must** have been called before calling E04VKF.

### 2 Specification

```
SUBROUTINE E04VKF (ISPECS, CW, IW, RW, IFAIL)
INTEGER           ISPECS, IW(*), IFAIL
REAL (KIND=nag_wp) RW(*)
CHARACTER(8)     CW(*)
```

### 3 Description

E04VKF may be used to supply values for optional parameters to E04VHF. E04VKF reads an external file and each line of the file defines a single optional parameter. It is only necessary to supply values for those parameters whose values are to be different from their default values.

Each optional parameter is defined by a single character string 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 16 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
```

Optional parameter settings are preserved following a call to E04VHF 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 E04VHF.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04VHF.

### 4 References

None.

## 5 Parameters

- 1: ISPECS – INTEGER *Input*  
*On entry:* the unit number of the option file to be read.  
*Constraint:* ISPECS is a valid unit open for reading.
- 2: CW(\*) – CHARACTER(8) array *Communication Array*  
**Note:** the dimension of the array CW must be at least LENCW (see E04VGF).
- 3: IW(\*) – INTEGER array *Communication Array*  
**Note:** the dimension of the array IW must be at least LENIW (see E04VGF).
- 4: RW(\*) – REAL (KIND=nag\_wp) array *Communication Array*  
**Note:** the dimension of the array RW must be at least LENRW (see E04VGF).
- 5: IFAIL – INTEGER *Input/Output*  
*On entry:* IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.  
 For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**  
*On exit:* IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

## 6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

The initialization routine E04VGF has not been called.

IFAIL = 2

At least one line of the options file is invalid.

Could not read options file on unit ISPECS = *<value>*.

*Could not read options file on unit ISPECS. This may be due to:*

- (a) ISPECS is not a valid unit number;
- (b) a file is not associated with unit ISPECS, or if it is, is unavailable for read access;
- (c) one or more lines of the options file is invalid. Check that all keywords are neither ambiguous nor misspelt;
- (d) Begin was found, but end-of-file was found before End was found;
- (e) end-of-file was found before Begin was found.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.  
See Section 3.8 in the Essential Introduction for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.  
See Section 3.7 in the Essential Introduction for further information.

IFAIL = -999

Dynamic memory allocation failed.  
See Section 3.6 in the Essential Introduction for further information.

## 7 Accuracy

Not applicable.

## 8 Parallelism and Performance

Not applicable.

## 9 Further Comments

E04VLF, E04VMF or E04VNF may also be used to supply optional parameters to E04VHF.

## 10 Example

This example solves the same problem as the example in the document for E04VHF, but sets and reads some optional parameters first. See Section 10 in E04VHF for further details.

The example in the document for E04VJF also solves the same problem (see Section 10 in E04VJF), but it first calls E04VJF to determine the sparsity pattern before calling E04VKF.

### 10.1 Program Text

```
! E04VKF Example Program Text
! Mark 25 Release. NAG Copyright 2014.

Module e04vkfe_mod

! E04VKF 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                                :: usrfun
! .. Parameters ..
Integer, Parameter, Public           :: lencw = 600, leniw = 600,      &
                                         lenrw = 600, nin = 5,          &
                                         ninopt = 7, nout = 6

Contains
Subroutine usrfun(status,n,x,needf,nf,f,needg,leng,g,cuser,iuser,ruser)

! .. Scalar Arguments ..
Integer, Intent (In)                  :: leng, n, needf, needg, nf
Integer, Intent (Inout)                :: status
! .. Array Arguments ..
```

```

Real (Kind=nag_wp), Intent (Inout)    :: f(nf), g(leng), ruser(*)
Real (Kind=nag_wp), Intent (In)      :: x(n)
Integer, Intent (Inout)               :: iuser(*)
Character (8), Intent (Inout)         :: cuser(*)
!
! .. Intrinsic Procedures ..
Intrinsic                             :: cos, sin
!
! .. Executable Statements ..
If (needf>0) Then

!     The nonlinear components of f_i(x) need to be assigned,
!     for i = 1 to NF

f(1) = 1000.0E+0_nag_wp*sin(-x(1)-0.25E+0_nag_wp) + &
      1000.0E+0_nag_wp*sin(-x(2)-0.25E+0_nag_wp)
f(2) = 1000.0E+0_nag_wp*sin(x(1)-0.25E+0_nag_wp) + &
      1000.0E+0_nag_wp*sin(x(1)-x(2)-0.25E+0_nag_wp)
f(3) = 1000.0E+0_nag_wp*sin(x(2)-x(1)-0.25E+0_nag_wp) + &
      1000.0E+0_nag_wp*sin(x(2)-0.25E+0_nag_wp)

!     N.B. in this example there is no need to assign for the wholly
!     linear components f_4(x) and f_5(x).

f(6) = 1.0E-6_nag_wp*x(3)**3 + 2.0E-6_nag_wp*x(4)**3/3.0E+0_nag_wp
End If

If (needg>0) Then

!     The derivatives of the function f_i(x) need to be assigned.
!     G(k) should be set to partial derivative df_i(x)/dx_j where
!     i = IGFUN(k) and j = IGVAR(k), for k = 1 to LENG.

g(1) = -1000.0E+0_nag_wp*cos(-x(1)-0.25E+0_nag_wp)
g(2) = -1000.0E+0_nag_wp*cos(-x(2)-0.25E+0_nag_wp)
g(3) = 1000.0E+0_nag_wp*cos(x(1)-0.25E+0_nag_wp) + &
      1000.0E+0_nag_wp*cos(x(1)-x(2)-0.25E+0_nag_wp)
g(4) = -1000.0E+0_nag_wp*cos(x(1)-x(2)-0.25E+0_nag_wp)
g(5) = -1000.0E+0_nag_wp*cos(x(2)-x(1)-0.25E+0_nag_wp)
g(6) = 1000.0E+0_nag_wp*cos(x(2)-x(1)-0.25E+0_nag_wp) + &
      1000.0E+0_nag_wp*cos(x(2)-0.25E+0_nag_wp)
g(7) = 3.0E-6_nag_wp*x(3)**2
g(8) = 2.0E-6_nag_wp*x(4)**2
End If

Return

End Subroutine usrfun
End Module e04vkfe_mod
Program e04vkfe

!     E04VKF Example Main Program

!
! .. Use Statements ..
Use nag_library, Only: e04vgf, e04vhf, e04vkf, e04vlf, e04vmf, e04vnf, &
      e04vrf, e04vsf, nag_wp, x04acf, x04baf
Use e04vkfe_mod, Only: lencw, leniw, lenrw, nin, ninopt, nout, usrfun
!
! .. Implicit None Statement ..
Implicit None
!
! .. Parameters ..
Character (*), Parameter               :: fname = 'e04vkfe.opt'
!
! .. Local Scalars ..
Real (Kind=nag_wp)                   :: bndinf, featol, objadd, sinf
Integer                               :: elmode, i, ifail, lena, leng, &
      mode, n, nea, neg, nf, nfname, &
      ninf, ns, nxname, objrow, start
Character (8)                         :: prob
Character (80)                        :: rec
!
! .. Local Arrays ..
Real (Kind=nag_wp), Allocatable       :: a(:), f(:), flow(:), fmul(:), &
      fupp(:), x(:), xlow(:), xmul(:), &
      xupp(:)
Real (Kind=nag_wp)                   :: ruser(1), rw(lenrw)

```

```

Integer, Allocatable                :: fstate(:), iafun(:), igfun(:), &
                                     javar(:), jgvar(:), xstate(:)
Integer                              :: iuser(1), iw(leniw)
Character (8)                        :: cuser(1), cw(lencw)
Character (8), Allocatable           :: fnames(:), xnames(:)
! .. Intrinsic Procedures ..
Intrinsic                            :: max
! .. Executable Statements ..
Write (rec,99995) 'E04VKF Example Program Results'
Call x04baf(nout,rec)

! This program demonstrates the use of routines to set and
! get values of optional parameters associated with E04VHF.

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

Read (nin,*) n, nf
Read (nin,*) nea, neg, objrow, start
lena = max(1,nea)
leng = max(1,neg)
nxname = n
nfname = nf
Allocate (iafun(lena),javar(lena),igfun(leng),jgvar(leng),xstate(n), &
         fstate(nf),a(lena),xlow(n),xupp(n),flow(nf),fupp(nf),x(n),xmul(n), &
         f(nf),fmul(nf),xnames(nxname),fnames(nfname))

! Read the variable names
Read (nin,*) xnames(1:nxname)

! Read the function names
Read (nin,*) fnames(1:nfname)

! Read the sparse matrix A, the linear part of F
Do i = 1, nea
!   For each element read row, column, A(row,column)
   Read (nin,*) iafun(i), javar(i), a(i)
End Do

! Read the structure of sparse matrix G, the nonlinear part of F
Do i = 1, neg
!   For each element read row, column
   Read (nin,*) igfun(i), jgvar(i)
End Do

! Read the lower and upper bounds on the variables
Do i = 1, n
   Read (nin,*) xlow(i), xupp(i)
End Do

! Read the lower and upper bounds on the functions
Do i = 1, nf
   Read (nin,*) flow(i), fupp(i)
End Do

! Initialise X, XSTATE, XMUL, F, FSTATE, FMUL
Read (nin,*) x(1:n)
Read (nin,*) xstate(1:n)
Read (nin,*) xmul(1:n)
Read (nin,*) f(1:nf)

```

```

Read (nin,*) fstate(1:nf)
Read (nin,*) fmul(1:nf)

objadd = 0.0E0_nag_wp
prob = ' '

! Call E04VGF to initialise E04VHF.

ifail = 0
Call e04vgf(cw,lencw,iw,leniw,rw,lenrw,ifail)

! By default E04VHF does not print monitoring
! information. Set the print file unit or the summary
! file unit to get information.

ifail = 0
Call e04vmf('Print file',nout,cw,iw,rw,ifail)

! Open the options file for reading

mode = 0

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

! Use E04VKF to read some options from the options file

ifail = 0
Call e04vkf(ninopt,cw,iw,rw,ifail)

Write (rec,'()')
Call x04baf(nout,rec)

! Use E04VRF to find the value of integer-valued option
! 'Elastic mode'.

ifail = 0
Call e04vrf('Elastic mode',elmode,cw,iw,rw,ifail)

Write (rec,99999) elmode
Call x04baf(nout,rec)

! Use E04VNF to set the value of real-valued option
! 'Infinite bound size'.

bndinf = 1.0E10_nag_wp

ifail = 0
Call e04vnf('Infinite bound size',bndinf,cw,iw,rw,ifail)

! Use E04VSF to find the value of real-valued option
! 'Feasibility tolerance'.

ifail = 0
Call e04vsf('Feasibility tolerance',featol,cw,iw,rw,ifail)

Write (rec,99998) featol
Call x04baf(nout,rec)

! Use E04VLF to set the option 'Major iterations limit'.

ifail = 0
Call e04vlf('Major iterations limit 50',cw,iw,rw,ifail)

! Solve the problem.

ifail = 0
Call e04vhf(start,nf,n,nxname,nfname,objadd,objrow,prob,usrfun,iafun, &
  javar,a,lena,nea,igfun,jgvar,leng,neg,xlow,xupp,xnames,flow,fupp, &
  fnames,x,xstate,xmul,f,fstate,fmul,ns,ninf,sinf,cw,lencw,iw,leniw,rw, &
  lenrw,cuser,iuser,ruser,ifail)

```

```

Write (rec, '()')
Call x04baf(nout, rec)
Write (rec, 99997) f(objrow)
Call x04baf(nout, rec)
Write (rec, 99996) x(1:n)
Call x04baf(nout, rec)

99999 Format (1X, 'Option ''Elastic mode'' has the value ', I3, '.')
99998 Format (1X, 'Option ''Feasibility tolerance'' has the value ', 1P, E13.5, &
          '.')
99997 Format (1X, 'Final objective value = ', F11.1)
99996 Format (1X, 'Optimal X = ', 7F9.2)
99995 Format (1X, A)
      End Program e04vkfe

```

## 10.2 Program Data

```

Begin example options file
* Comment lines like this begin with an asterisk.
* Switch off output of timing information:
Timing level 0
* Allow elastic variables:
Elastic mode 1
* Set the feasibility tolerance:
Feasibility tolerance 1.0D-4
End

E04VKF Example Program Data
  4   6           : Values of N and NF
  8   8   6   0   : Values of NEA, NEG, OBJROW and START

'X1' 'X2' 'X3' 'X4' : XNAMES
'NlnCon 1' 'NlnCon 2' 'NlnCon 3' 'LinCon 1' 'LinCon 2' 'Objectiv' : FNAMES

1  3 -1.0D0 : Nonzero elements of sparse matrix A, the linear part of F.
2  4 -1.0D0 : Each row IAFUN(i), JAVAR(i), A(IAFUN(i), JAVAR(i)), i = 1 to NEA
4  1 -1.0D0
4  2  1.0D0
5  1  1.0D0
5  2 -1.0D0
6  3  3.0D0
6  4  2.0D0

1  1           : Nonzero row/column structure of G, IGFUN(i), JGVAR(i), i = 1 to NEG
1  2
2  1
2  2
3  1
3  2
6  3
6  4

-0.55D0  0.55D0 : Bounds on the variables, XLOW(i), XUPP(i), for i = 1 to N
-0.55D0  0.55D0
  0.0D0 1200.0D0
  0.0D0 1200.0D0

-894.8D0 -894.8D0 : Bounds on the functions, FLOW(i), FUPP(i), for i = 1 to NF
-894.8D0 -894.8D0
-1294.8D0 -1294.8D0
-0.55D0  1.0D25
-0.55D0  1.0D25
-1.0D25  1.0D25

  0.0  0.0  0.0  0.0           : Initial values of X(i), for i = 1 to N
  0    0    0    0           : Initial values of XSTATE(i), for i = 1 to N

```

```

0.0  0.0  0.0  0.0           : Initial values of XMUL(i), for i = 1 to N
0.0  0.0  0.0  0.0  0.0  0.0 : Initial values of F(i), for i = 1 to NF
0   0   0   0   0   0       : Initial values of FSTATE(i), for i = 1 to NF
0.0  0.0  0.0  0.0  0.0  0.0 : Initial values of FMUL(i), for i = 1 to NF

```

### 10.3 Program Results

E04VKF Example Program Results

OPTIONS file

-----

```

Begin example options file
* Comment lines like this begin with an asterisk.
* Switch off output of timing information:
Timing level 0
* Allow elastic variables:
Elastic mode 1
* Set the feasibility tolerance:
Feasibility tolerance 1.0D-4
End

```

```

E04VKZ EXIT 100 -- finished successfully
E04VKZ INFO 101 -- OPTIONS file read

```

```

Option 'Elastic mode' has the value 1.
Option 'Feasibility tolerance' has the value 1.00000E-04.

```

Parameters

=====

Files

-----

Solution file.....	0	Old basis file .....	0	(Print file).....	6
Insert file.....	0	New basis file .....	0	(Summary file).....	0
Punch file.....	0	Backup basis file.....	0		
Load file.....	0	Dump file.....	0		

Frequencies

-----

Print frequency.....	100	Check frequency.....	60	Save new basis map....	100
Summary frequency.....	100	Factorization frequency	50	Expand frequency.....	10000

QP subproblems

-----

QP solver Cholesky.....					
Scale tolerance.....	0.900	Minor feasibility tol..	1.00E-04	Iteration limit.....	10000
Scale option.....	0	Minor optimality tol..	1.00E-06	Minor print level.....	1
Crash tolerance.....	0.100	Pivot tolerance.....	2.04E-11	Partial price.....	1
Crash option.....	3	Elastic weight.....	1.00E+04	Prtl price section ( A)	4
		New superbasics.....	99	Prtl price section (-I)	6

The SQP Method

-----

Minimize.....		Cold start.....		Proximal Point method..	1
Nonlinear objectiv vars	4	Objective Row.....	6	Function precision....	1.72E-13
Unbounded step size....	1.00E+10	Superbasics limit.....	4	Difference interval....	4.15E-07
Unbounded objective....	1.00E+15	Reduced Hessian dim....	4	Central difference int.	5.57E-05
Major step limit.....	2.00E+00	Derivative linesearch..		Derivative option.....	1
Major iterations limit.	50	Linesearch tolerance...	0.90000	Verify level.....	0
Minor iterations limit.	500	Penalty parameter.....	0.00E+00	Major Print Level.....	1
		Major optimality tol...	2.00E-06		

Hessian Approximation

-----

Full-Memory Hessian....		Hessian updates.....	99999999	Hessian frequency.....	99999999
				Hessian flush.....	99999999



```

Nonlinear constraints
-----
Nonlinear constraints..      3      Major feasibility tol.. 1.00E-06      Violation limit..... 1.00E+06
Nonlinear Jacobian vars    2
    
```

```

Miscellaneous
-----
LU factor tolerance....    3.99      LU singularity tol..... 2.04E-11      Timing level.....      0
LU update tolerance....    3.99      LU swap tolerance..... 1.03E-04      Debug level.....      0
LU partial pivoting...           eps (machine precision) 1.11E-16      System information.... No
    
```

```

Matrix statistics
-----
                Total      Normal      Free      Fixed      Bounded
Rows             6          2          1          3          0
Columns          4          0          0          0          4

No. of matrix elements          14      Density      58.333
Biggest          1.0000E+00 (excluding fixed columns,
Smallest          0.0000E+00 free rows, and RHS)

No. of objective coefficients          2
Biggest          3.0000E+00 (excluding fixed columns)
Smallest          2.0000E+00

Nonlinear constraints      3      Linear constraints      3
Nonlinear variables        4      Linear variables        0
Jacobian variables         2      Objective variables     4
Total constraints          6      Total variables          4
    
```

The user has defined 8 out of 8 first derivatives

Cheap test of user-supplied problem derivatives...

The constraint gradients seem to be OK.

--> The largest discrepancy was 2.20E-08 in constraint 6

The objective gradients seem to be OK.

```

Gradient projected in one direction 0.0000000000E+00
Difference approximation            4.48709939860E-21
    
```

Itns	Major	Minors	Step	nCon	Feasible	Optimal	MeritFunction	L+U	BSwap	nS	condHz	Penalty
3	0	3		1	8.0E+02	1.0E+00	0.0000000E+00	17		1	1.7E+07	_ r
5	1	2	1.2E-03	2	4.0E+02	9.9E-01	9.6317131E+05	16		1	4.8E+06	2.8E+00 _n r1
6	2	1	1.3E-03	3	2.7E+02	5.5E-01	9.6122945E+05	16				2.8E+00 _s 1
6	3	0	7.5E-03	4	8.8E+01	5.4E-01	9.4691061E+05	16				2.8E+00 _ 1
6	4	0	2.3E-02	5	2.9E+01	5.3E-01	9.0468403E+05	16				2.8E+00 _ 1
6	5	0	6.9E-02	6	8.9E+00	5.0E-01	7.8452897E+05	16				2.8E+00 _ 1
7	6	1	2.2E-01	7	2.3E+00	5.5E+01	4.8112339E+05	16		1	8.7E+03	2.8E+00 _ 1
8	7	1	8.3E-01	8	1.7E-01	4.2E+00	2.6898257E+04	16		1	7.6E+03	2.8E+00 _ 1
9	8	1	1.0E+00	9	1.8E-02	8.7E+01	6.2192920E+03	15	1	1	1.2E+02	2.8E+00 _
10	9	1	1.0E+00	10	1.7E-02	7.9E+00	5.4526185E+03	15		1	9.4E+01	2.8E+00 _
11	10	1	1.0E+00	11	1.7E-04	9.6E-01	5.1266089E+03	15		1	1.0E+02	2.8E+00 _
12	11	1	1.0E+00	12	1.7E-06	5.8E-02	5.1264988E+03	15		1	9.5E+01	2.8E+00 _
13	12	1	1.0E+00	13	( 1.2E-08)	6.9E-05	5.1264981E+03	15		1	9.5E+01	2.8E+00 _
14	13	1	1.0E+00	14	( 6.7E-15)( 3.0E-09)		5.1264981E+03	15		1	9.5E+01	6.0E+00 _

```

E04VHU EXIT 0 -- finished successfully
E04VHU INFO 1 -- optimality conditions satisfied
    
```

Problem name

```

No. of iterations          14 Objective value      5.1264981096E+03
No. of major iterations   13 Linear objective   4.0919702248E+03
Penalty parameter        6.038E+00 Nonlinear objective 1.0345278848E+03
No. of calls to funobj    15 No. of calls to funcon 15
No. of superbasics        1 No. of basic nonlinears 3
No. of degenerate steps   0 Percentage          0.00
Max x                     4 1.0E+03 Max pi              3 5.5E+00
Max Primal infeas        0 0.0E+00 Max Dual infeas     1 4.6E-08
Nonlinear constraint violn 5.7E-12
    
```

```

Name                               Objective Value      5.1264981096E+03
Status      Optimal Soln           Iteration    14 Superbasics    1
    
```

```

Objective      (Min)
RHS
Ranges
Bounds
    
```

Section 1 - Rows

Number	..Row..	State	...Activity...	Slack Activity	..Lower Limit.	..Upper Limit.	.Dual Activity	..i
5	NlnCon 1	EQ	-894.80000	0.00000	-894.80000	-894.80000	-4.38698	1
6	NlnCon 2	EQ	-894.80000	0.00000	-894.80000	-894.80000	-4.10563	2
7	NlnCon 3	EQ	-1294.80000	0.00000	-1294.80000	-1294.80000	-5.46328	3
8	LinCon 1	BS	-0.51511	0.03489	-0.55000	None	.	4
9	LinCon 2	BS	0.51511	1.06511	-0.55000	None	.	5
10	Objectiv	BS	4091.97022	4091.97022	None	None	-1.0	6

Section 2 - Columns

Number	.Column.	State	...Activity...	.Obj Gradient.	..Lower Limit.	..Upper Limit.	Reduced Gradnt	m+j
1	X1	BS	0.11888	.	-0.55000	0.55000	0.00000	7
2	X2	BS	-0.39623	.	-0.55000	0.55000	0.00000	8
3	X3	SBS	679.94532	4.38698	.	1200.00000	0.00000	9
4	X4	BS	1026.06713	4.10563	.	1200.00000	0.00000	10

```

Final objective value =      5126.5
Optimal X =           0.12 -0.40 679.95 1026.07
    
```