

NAG Library Routine Document

E04WEF

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

E04WEF may be used to supply optional parameters to E04WDF from an external file. The initialization routine E04WCF **must** have been called before calling E04WEF.

2 Specification

```
SUBROUTINE E04WEF (ISPECS, IW, RW, IFAIL)
  INTEGER          ISPECS, IW(*), IFAIL
  REAL (KIND=nag_wp) RW(*)
```

3 Description

E04WEF may be used to supply values for optional parameters to E04WDF. E04WEF 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
```

Optional parameter settings are preserved following a call to E04WDF 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 E04WDF.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04WDF.

4 References

Hock W and Schittkowski K (1981) *Test Examples for Nonlinear Programming Codes. Lecture Notes in Economics and Mathematical Systems* **187** Springer–Verlag

5 Arguments

- 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: IW(*) – INTEGER array *Communication Array*
Note: the dimension of the array IW must be at least LENIW (see E04WCF).
- 3: RW(*) – REAL (KIND=nag_wp) array *Communication Array*
Note: the dimension of the array RW must be at least LENRW (see E04WCF).
- 4: IFAIL – INTEGER *Input/Output*
On entry: IFAIL must be set to 0, -1 or 1. 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.
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 argument, 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 E04WCF 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 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

E04WEF is not threaded in any implementation.

9 Further Comments

E04WFF, E04WGF or E04WHF may also be used to supply optional parameters to E04WDF.

10 Example

This example is based on Problem 71 in Hock and Schittkowski (1981) and involves the minimization of the nonlinear function

$$F(x) = x_1x_4(x_1 + x_2 + x_3) + x_3$$

subject to the bounds

$$\begin{aligned} 1 &\leq x_1 \leq 5 \\ 1 &\leq x_2 \leq 5 \\ 1 &\leq x_3 \leq 5 \\ 1 &\leq x_4 \leq 5 \end{aligned}$$

to the general linear constraint

$$x_1 + x_2 + x_3 + x_4 \leq 20,$$

and to the nonlinear constraints

$$\begin{aligned} x_1^2 + x_2^2 + x_3^2 + x_4^2 &\leq 40, \\ x_1x_2x_3x_4 &\geq 25. \end{aligned}$$

The initial point, which is infeasible, is

$$x_0 = (1, 5, 5, 1)^T,$$

and $F(x_0) = 16$.

The optimal solution (to five figures) is

$$x^* = (1.0, 4.7430, 3.8211, 1.3794)^T,$$

and $F(x^*) = 17.014$. One bound constraint and both nonlinear constraints are active at the solution.

The document for E04WEF includes an example program to solve the same problem using some of the optional parameters described in Section 12 in E04WDF.

10.1 Program Text

```

!   E04WEF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

Module e04wefe_mod

!   E04WEF 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                                :: confun, objfun
!   .. Parameters ..
Integer, Parameter, Public            :: leniw = 600, lenrw = 600, nin = 5,    &
                                         ninopt = 7, nout = 6
Contains
Subroutine objfun(mode,n,x,objf,grad,nstate,iuser,ruser)
!   Routine to evaluate objective function and its 1st derivatives.

!   .. 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 (Inout) :: grad(n), ruser(*)
Real (Kind=nag_wp), Intent (In) :: x(n)
Integer, Intent (Inout)          :: iuser(*)
!   .. Executable Statements ..
If (mode==0 .Or. mode==2) Then
    objf = x(1)*x(4)*(x(1)+x(2)+x(3)) + x(3)
End If

If (mode==1 .Or. mode==2) Then
    grad(1) = x(4)*(2.0E0_nag_wp*x(1)+x(2)+x(3))
    grad(2) = x(1)*x(4)
    grad(3) = x(1)*x(4) + 1.0E0_nag_wp
    grad(4) = x(1)*(x(1)+x(2)+x(3))
End If

Return

End Subroutine objfun
Subroutine confun(mode,ncnln,n,ldcj,needc,x,ccon,cjac,nstate,iuser,    &
    ruser)
!   Routine to evaluate the nonlinear constraints and their 1st
!   derivatives.

!   .. Scalar Arguments ..
Integer, Intent (In)          :: ldcj, n, ncnln, nstate
Integer, Intent (Inout)       :: mode
!   .. Array Arguments ..
Real (Kind=nag_wp), Intent (Out) :: ccon(max(1,ncnln))
Real (Kind=nag_wp), Intent (Inout) :: cjac(ldcj,n), ruser(*)
Real (Kind=nag_wp), Intent (In) :: x(n)
Integer, Intent (Inout)       :: iuser(*)
Integer, Intent (In)          :: needc(ncnln)
!   .. Intrinsic Procedures ..
Intrinsic                    :: max
!   .. Executable Statements ..
If (nstate==1) Then

!       First call to CONFUN. Set all Jacobian elements to zero.
!       Note that this will only work when 'Derivative Level = 3'
!       (the default; see Section 11.2).

    cjac(1:ncnln,1:n) = 0.0E0_nag_wp

```

```

End If

If (needc(1)>0) Then

  If (mode==0 .Or. mode==2) Then
    ccon(1) = x(1)**2 + x(2)**2 + x(3)**2 + x(4)**2
  End If

  If (mode==1 .Or. mode==2) Then
    cjac(1,1) = 2.0E0_nag_wp*x(1)
    cjac(1,2) = 2.0E0_nag_wp*x(2)
    cjac(1,3) = 2.0E0_nag_wp*x(3)
    cjac(1,4) = 2.0E0_nag_wp*x(4)
  End If

End If

If (needc(2)>0) Then

  If (mode==0 .Or. mode==2) Then
    ccon(2) = x(1)*x(2)*x(3)*x(4)
  End If

  If (mode==1 .Or. mode==2) Then
    cjac(2,1) = x(2)*x(3)*x(4)
    cjac(2,2) = x(1)*x(3)*x(4)
    cjac(2,3) = x(1)*x(2)*x(4)
    cjac(2,4) = x(1)*x(2)*x(3)
  End If

End If

Return

End Subroutine confun
End Module e04wefe_mod
Program e04wefe

!      E04WEF Example Main Program

!      .. Use Statements ..
Use nag_library, Only: e04wcf, e04wdf, e04wef, e04wff, e04wgf, e04whf, &
                        e04wkf, e04wlf, nag_wp, x04acf, x04baf
Use e04wefe_mod, Only: confun, leniw, lenrw, nin, ninopt, nout, objfun
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Character (*), Parameter      :: fname = 'e04wefe.opt'
!      .. Local Scalars ..
Real (Kind=nag_wp)           :: bndinf, featol, objf
Integer                       :: elmode, i, ifail, lda, ldcj, ldh, &
                                majits, mode, n, nclin, ncnln, sda, &
                                sdcjac
Character (80)                :: rec
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable :: a(:,,:), bl(:), bu(:), ccon(:), &
                                cjac(:,,:), clamda(:), grad(:), &
                                h(:,,:), x(:)
Real (Kind=nag_wp)           :: ruser(1), rw(lenrw)
Integer, Allocatable         :: istate(:)
Integer                       :: iuser(1), iw(leniw)
!      .. Intrinsic Procedures ..
Intrinsic                    :: max
!      .. Executable Statements ..
Write (rec,99995) 'E04WEF Example Program Results'
Call x04baf(nout,rec)

!      This program demonstrates the use of routines to set and
!      get values of optional parameters associated with E04WDF.

!      Skip heading in data file

```

```

Read (nin,*)

Read (nin,*) n, nclin, ncnln
lda = max(1,nclin)

If (nclin>0) Then
  sda = n
Else
  sda = 1
End If

ldcj = max(1,ncnln)

If (ncnln>0) Then
  sdcjac = n
Else
  sdcjac = 1
End If

ldh = n
Allocate (istate(n+nclin+ncnln),a(lda,sda),bl(n+nclin+ncnln),      &
  bu(n+nclin+ncnln),ccon(max(1,ncnln)),cjac(ldcj,sdcjac),clamda(n+nclin+ &
  ncnln),grad(n),h(ldh,n),x(n))

If (nclin>0) Then
  Read (nin,*)(a(i,1:sd),i=1,nclin)
End If

Read (nin,*) bl(1:(n+nclin+ncnln))
Read (nin,*) bu(1:(n+nclin+ncnln))
Read (nin,*) x(1:n)

!   Call E04WCF to initialize E04WDF.

  ifail = 0
  Call e04wcf(iw,leniw,rw,lenrw,ifail)

!   By default E04WDF does not print monitoring
!   information. Set the print file unit or the summary
!   file unit to get information.

  ifail = 0
  Call e04wgf('Print file',nout,iw,rw,ifail)

!   Open the options file for reading

  mode = 0

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

!   Use E04WEF to read some options from the options file

  ifail = 0
  Call e04wef(ninopt,iw,rw,ifail)

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

!   Use E04WKF to find the value of integer-valued option
!   'Elastic mode'.

  ifail = 0
  Call e04wkf('Elastic mode',elmode,iw,rw,ifail)

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

!   Use E04WHF to set the value of real-valued option
!   'Infinite bound size'.

```

```

bndinf = 1.0E10_nag_wp

ifail = 0
Call e04whf('Infinite bound size',bndinf,iw,rw,ifail)

! Use E04WLF to find the value of real-valued option
! 'Feasibility tolerance'.

ifail = 0
Call e04wlf('Feasibility tolerance',featol,iw,rw,ifail)

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

! Use E04WFF to set the option 'Major iterations limit'.

ifail = 0
Call e04wff('Major iterations limit 50',iw,rw,ifail)

! Solve the problem.

ifail = 0
Call e04wdf(n,nclin,ncnln,lda,ldcj,ldh,a,bl,bu,confun,objfun,majits,      &
           ystate,ccon,cjac,clamda,objf,grad,h,x,iw,leniw,rw,lenrw,iuser,ruser, &
           ifail)

Write (rec,'()')
Call x04baf(nout,rec)
Write (rec,99997) objf
Call x04baf(nout,rec)
Write (rec,99996)(x(i),i=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 = ',F11.3)
99996 Format (1X,'Optimal X = ',7F9.2)
99995 Format (1X,A)
End Program e04wefe

```

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

```

E04WEF Example Program Data

4	1	2						: N, NCLIN and NCNLN
1.0	1.0	1.0	1.0					: Matrix A
1.0	1.0	1.0	1.0	-1.0E+25	-1.0E+25	25.0		: Lower bounds BL
5.0	5.0	5.0	5.0	20.0	40.0	1.0E+25		: Upper bounds BU
1.0	5.0	5.0	1.0					: Initial vector X

10.3 Program Results

E04WEF 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

```

```

E04WEZ EXIT 100 -- finished successfully
E04WEZ 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

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

The SQP Method

Minimize.....		Cold start.....		Proximal Point method..	1
Nonlinear objectiv vars	4	Major optimality tol...	2.00E-06	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 level.....	3
Major iterations limit.	50	Linesearch tolerance...	0.90000	Verify level.....	0
Minor iterations limit.	500	Penalty parameter.....	0.00E+00	Major Print Level.....	1

Hessian Approximation

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

Nonlinear constraints

Nonlinear constraints..	2	Major feasibility tol..	1.00E-06	Violation limit.....	1.00E+06
Nonlinear Jacobian vars	4				

Miscellaneous

LU factor tolerance....	1.10	LU singularity tol.....	2.04E-11	Timing level.....	0
LU update tolerance....	1.10	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	3	3	0	0	0
Columns	4	0	0	0	4
No. of matrix elements		12	Density	100.000	
Biggest		1.0000E+00	(excluding fixed columns,		
Smallest		0.0000E+00	free rows, and RHS)		

No. of objective coefficients 0

Nonlinear constraints	2	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	4	Objective variables	4
Total constraints	3	Total variables	4

The user has defined 8 out of 8 constraint gradients.
 The user has defined 4 out of 4 objective gradients.

Cheap test of user-supplied problem derivatives...

The constraint gradients seem to be OK.

--> The largest discrepancy was 1.84E-07 in constraint 6

The objective gradients seem to be OK.

Gradient projected in one direction 4.99993000077E+00
Difference approximation 4.99993303560E+00

Itns	Major	Minors	Step	nCon	Feasible	Optimal	MeritFunction	L+U	BSwap	nS	condHz	Penalty	
2	0	2		1	1.7E+00	2.8E+00	1.6000000E+01	7		2	1.0E+00		_ r
4	1	2	1.0E+00	2	1.3E-01	3.2E-01	1.7726188E+01	8		1	6.2E+00	8.3E-02	_n r
5	2	1	1.0E+00	3	3.7E-02	1.7E-01	1.7099571E+01	7		1	2.0E+00	8.3E-02	_s
6	3	1	1.0E+00	4	2.2E-02	1.1E-02	1.7014005E+01	7		1	1.8E+00	8.3E-02	_
7	4	1	1.0E+00	5	1.5E-04	6.0E-04	1.7014018E+01	7		1	1.8E+00	9.2E-02	_
8	5	1	1.0E+00	6	(3.3E-07)	2.3E-05	1.7014017E+01	7		1	1.9E+00	3.6E-01	_
9	6	1	1.0E+00	7	(4.2E-10)	(2.4E-08)	1.7014017E+01	7		1	1.9E+00	3.6E-01	_

E04WDM EXIT 0 -- finished successfully
E04WDM INFO 1 -- optimality conditions satisfied

Problem name		NLP
No. of iterations	9	Objective value 1.7014017287E+01
No. of major iterations	6	Linear objective 0.0000000000E+00
Penalty parameter	3.599E-01	Nonlinear objective 1.7014017287E+01
No. of calls to funobj	8	No. of calls to funcon 8
No. of superbasics	1	No. of basic nonlinears 2
No. of degenerate steps	0	Percentage 0.00
Max x	2 4.7E+00	Max pi 2 5.5E-01
Max Primal infeas	0 0.0E+00	Max Dual infeas 3 4.8E-08
Nonlinear constraint violn	2.7E-09	

Variable	State	Value	Lower bound	Upper bound	Lagr multiplier	Slack
variable	1 LL	1.000000	1.000000	5.000000	1.087871	.
variable	2 FR	4.743000	1.000000	5.000000	.	0.2570
variable	3 FR	3.821150	1.000000	5.000000	.	1.179
variable	4 FR	1.379408	1.000000	5.000000	.	0.3794

Linear constrnt	State	Value	Lower bound	Upper bound	Lagr multiplier	Slack
lincon	1 FR	10.94356	None	20.00000	.	9.056

Nonlin constrnt	State	Value	Lower bound	Upper bound	Lagr multiplier	Slack
nlncon	1 UL	40.00000	None	40.00000	-0.1614686	-0.2700E-08
nlncon	2 LL	25.00000	25.00000	None	0.5522937	-0.2215E-08

Final objective value = 17.014
Optimal X = 1.00 4.74 3.82 1.38