NAG Library Routine Document

D03FAF

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

D03FAF solves the Helmholtz equation in Cartesian coordinates in three dimensions using the standard seven-point finite difference approximation. This routine is designed to be particularly efficient on vector processors.

2 Specification

```
SUBROUTINE D03FAF (XS, XF, L, LBDCND, BDXS, BDXF, YS, YF, M, MBDCND, BDYS,      &
                   BDYF, ZS, ZF, N, NBDCND, BDZS, BDZF, LAMBDA, LDF, LDF2,      &
                   F, PERTRB, W, LWRK, IFAIL)
INTEGER            L, LBDCND, M, MBDCND, N, NBDCND, LDF, LDF2, LWRK, IFAIL
REAL (KIND=nag_wp) XS, XF, BDXS(LDF2,N+1), BDXF(LDF2,N+1), YS, YF,              &
                   BDYS(LDF,N+1), BDYF(LDF,N+1), ZS, ZF, BDZS(LDF,M+1),         &
                   BDZF(LDF,M+1), LAMBDA, F(LDF,LDF2,N+1), PERTRB, W(LWRK)
```

3 Description

D03FAF solves the three-dimensional Helmholtz equation in Cartesian coordinates:

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}+\lambda u=f(x,y,z).$$

This subroutine forms the system of linear equations resulting from the standard seven-point finite difference equations, and then solves the system using a method based on the fast Fourier transform (FFT) described by Swarztrauber (1984). This subroutine is based on the routine HW3CRT from FISHPACK (see Swarztrauber and Sweet (1979)).

More precisely, the routine replaces all the second derivatives by second-order central difference approximations, resulting in a block tridiagonal system of linear equations. The equations are modified to allow for the prescribed boundary conditions. Either the solution or the derivative of the solution may be specified on any of the boundaries, or the solution may be specified to be periodic in any of the three dimensions. By taking the discrete Fourier transform in the x- and y-directions, the equations are reduced to sets of tridiagonal systems of equations. The Fourier transforms required are computed using the multiple FFT routines found in Chapter C06.

4 References

Swarztrauber P N (1984) Fast Poisson solvers *Studies in Numerical Analysis* (ed G H Golub) Mathematical Association of America

Swarztrauber P N and Sweet R A (1979) Efficient Fortran subprograms for the solution of separable elliptic partial differential equations *ACM Trans. Math. Software* **5** 352–364

5 Parameters

1: XS – REAL (KIND=nag_wp) *Input*

On entry: the lower bound of the range of x, i.e., XS $\leq x \leq$ XF.

Constraint: XS $<$ XF.

2: XF – REAL (KIND=nag_wp) *Input*

On entry: the upper bound of the range of x, i.e., $\mathrm{XS} \leq x \leq \mathrm{XF}$.

Constraint: $\mathrm{XS} < \mathrm{XF}$.

3: L – INTEGER *Input*

On entry: the number of panels into which the interval (XS,XF) is subdivided. Hence, there will be $\mathrm{L}+1$ grid points in the x-direction given by $x_i = \mathrm{XS} + (i-1) \times \delta x$, for $i = 1, 2, \ldots, \mathrm{L}+1$, where $\delta x = (\mathrm{XF} - \mathrm{XS})/\mathrm{L}$ is the panel width.

Constraint: $\mathrm{L} \geq 5$.

4: LBDCND – INTEGER *Input*

On entry: indicates the type of boundary conditions at $x = \mathrm{XS}$ and $x = \mathrm{XF}$.

$\mathrm{LBDCND} = 0$
: If the solution is periodic in x, i.e., $u(\mathrm{XS}, y, z) = u(\mathrm{XF}, y, z)$.

$\mathrm{LBDCND} = 1$
: If the solution is specified at $x = \mathrm{XS}$ and $x = \mathrm{XF}$.

$\mathrm{LBDCND} = 2$
: If the solution is specified at $x = \mathrm{XS}$ and the derivative of the solution with respect to x is specified at $x = \mathrm{XF}$.

$\mathrm{LBDCND} = 3$
: If the derivative of the solution with respect to x is specified at $x = \mathrm{XS}$ and $x = \mathrm{XF}$.

$\mathrm{LBDCND} = 4$
: If the derivative of the solution with respect to x is specified at $x = \mathrm{XS}$ and the solution is specified at $x = \mathrm{XF}$.

Constraint: $0 \leq \mathrm{LBDCND} \leq 4$.

5: BDXS(LDF2,N + 1) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to x at $x = \mathrm{XS}$. When $\mathrm{LBDCND} = 3$ or 4, $\mathrm{BDXS}(j,k) = u_x(\mathrm{XS}, y_j, z_k)$, for $j = 1, 2, \ldots, \mathrm{M}+1$ and $k = 1, 2, \ldots, \mathrm{N}+1$.

When LBDCND has any other value, BDXS is not referenced.

6: BDXF(LDF2,N + 1) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to x at $x = \mathrm{XF}$. When $\mathrm{LBDCND} = 2$ or 3, $\mathrm{BDXF}(j,k) = u_x(\mathrm{XF}, y_j, z_k)$, for $j = 1, 2, \ldots, \mathrm{M}+1$ and $k = 1, 2, \ldots, \mathrm{N}+1$.

When LBDCND has any other value, BDXF is not referenced.

7: YS – REAL (KIND=nag_wp) *Input*

On entry: the lower bound of the range of y, i.e., $\mathrm{YS} \leq y \leq \mathrm{YF}$.

Constraint: $\mathrm{YS} < \mathrm{YF}$.

8: YF – REAL (KIND=nag_wp) *Input*

On entry: the upper bound of the range of y, i.e., $\mathrm{YS} \leq y \leq \mathrm{YF}$.

Constraint: $\mathrm{YS} < \mathrm{YF}$.

9: M – INTEGER *Input*

On entry: the number of panels into which the interval (YS,YF) is subdivided. Hence, there will be $\mathrm{M}+1$ grid points in the y-direction given by $y_j = \mathrm{YS} + (j-1) \times \delta y$, for $j = 1, 2, \ldots, \mathrm{M}+1$, where $\delta y = (\mathrm{YF} - \mathrm{YS})/\mathrm{M}$ is the panel width.

Constraint: $\mathrm{M} \geq 5$.

10: MBDCND – INTEGER *Input*

On entry: indicates the type of boundary conditions at $y = \mathrm{YS}$ and $y = \mathrm{YF}$.

$\mathrm{MBDCND} = 0$
If the solution is periodic in y, i.e., $u(x, \mathrm{YF}, z) = u(x, \mathrm{YS}, z)$.

$\mathrm{MBDCND} = 1$
If the solution is specified at $y = \mathrm{YS}$ and $y = \mathrm{YF}$.

$\mathrm{MBDCND} = 2$
If the solution is specified at $y = \mathrm{YS}$ and the derivative of the solution with respect to y is specified at $y = \mathrm{YF}$.

$\mathrm{MBDCND} = 3$
If the derivative of the solution with respect to y is specified at $y = \mathrm{YS}$ and $y = \mathrm{YF}$.

$\mathrm{MBDCND} = 4$
If the derivative of the solution with respect to y is specified at $y = \mathrm{YS}$ and the solution is specified at $y = \mathrm{YF}$.

Constraint: $0 \leq \mathrm{MBDCND} \leq 4$.

11: BDYS(LDF,N + 1) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to y at $y = \mathrm{YS}$. When $\mathrm{MBDCND} = 3$ or 4, $\mathrm{BDYS}(i, k) = u_y(x_i, \mathrm{YS}, z_k)$, for $i = 1, 2, \ldots, \mathrm{L}+1$ and $k = 1, 2, \ldots, \mathrm{N}+1$.

When MBDCND has any other value, BDYS is not referenced.

12: BDYF(LDF,N + 1) – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to y at $y = \mathrm{YF}$. When $\mathrm{MBDCND} = 2$ or 3, $\mathrm{BDYF}(i, k) = u_y(x_i, \mathrm{YF}, z_k)$, for $i = 1, 2, \ldots, \mathrm{L}+1$ and $k = 1, 2, \ldots, \mathrm{N}+1$.

When MBDCND has any other value, BDYF is not referenced.

13: ZS – REAL (KIND=nag_wp) *Input*

On entry: the lower bound of the range of z, i.e., $\mathrm{ZS} \leq z \leq \mathrm{ZF}$.

Constraint: $\mathrm{ZS} < \mathrm{ZF}$.

14: ZF – REAL (KIND=nag_wp) *Input*

On entry: the upper bound of the range of z, i.e., $\mathrm{ZS} \leq z \leq \mathrm{ZF}$.

Constraint: $\mathrm{ZS} < \mathrm{ZF}$.

15: N – INTEGER *Input*

On entry: the number of panels into which the interval (ZS,ZF) is subdivided. Hence, there will be $\mathrm{N}+1$ grid points in the z-direction given by $z_k = \mathrm{ZS} + (k-1) \times \delta z$, for $k = 1, 2, \ldots, \mathrm{N}+1$, where $\delta z = (\mathrm{ZF} - \mathrm{ZS})/\mathrm{N}$ is the panel width.

Constraint: $\mathrm{N} \geq 5$.

16: NBDCND – INTEGER *Input*

On entry: specifies the type of boundary conditions at $z = \text{ZS}$ and $z = \text{ZF}$.

NBDCND $= 0$
: if the solution is periodic in z, i.e., $u(x, y, \text{ZF}) = u(x, y, \text{ZS})$.

NBDCND $= 1$
: if the solution is specified at $z = \text{ZS}$ and $z = \text{ZF}$.

NBDCND $= 2$
: if the solution is specified at $z = \text{ZS}$ and the derivative of the solution with respect to z is specified at $z = \text{ZF}$.

NBDCND $= 3$
: if the derivative of the solution with respect to z is specified at $z = \text{ZS}$ and $z = \text{ZF}$.

NBDCND $= 4$
: if the derivative of the solution with respect to z is specified at $z = \text{ZS}$ and the solution is specified at $z = \text{ZF}$.

Constraint: $0 \leq \text{NBDCND} \leq 4$.

17: BDZS$(\text{LDF},\text{M}+1)$ – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to z at $z = \text{ZS}$. When NBDCND $= 3$ or 4, $\text{BDZS}(i, j) = u_z(x_i, y_j, \text{ZS})$, for $i = 1, 2, \ldots, \text{L} + 1$ and $j = 1, 2, \ldots, \text{M} + 1$.

When NBDCND has any other value, BDZS is not referenced.

18: BDZF$(\text{LDF},\text{M}+1)$ – REAL (KIND=nag_wp) array *Input*

On entry: the values of the derivative of the solution with respect to z at $z = \text{ZF}$. When NBDCND $= 2$ or 3, $\text{BDZF}(i, j) = u_z(x_i, y_j, \text{ZF})$, for $i = 1, 2, \ldots, \text{L} + 1$ and $j = 1, 2, \ldots, \text{M} + 1$.

When NBDCND has any other value, BDZF is not referenced.

19: LAMBDA – REAL (KIND=nag_wp) *Input*

On entry: the constant λ in the Helmholtz equation. For certain positive values of λ a solution to the differential equation may not exist, and close to these values the solution of the discretized problem will be extremely ill-conditioned. If $\lambda > 0$, then D03FAF will set IFAIL $= 3$, but will still attempt to find a solution. However, since in general the values of λ for which no solution exists cannot be predicted *a priori*, you are advised to treat any results computed with $\lambda > 0$ with great caution.

20: LDF – INTEGER *Input*

On entry: the first dimension of the arrays F, BDYS, BDYF, BDZS and BDZF as declared in the (sub)program from which D03FAF is called.

Constraint: $\text{LDF} \geq \text{L} + 1$.

21: LDF2 – INTEGER *Input*

On entry: the second dimension of the array F and the first dimension of the arrays BDXS and BDXF as declared in the (sub)program from which D03FAF is called.

Constraint: $\text{LDF2} \geq \text{M} + 1$.

22: F$(\text{LDF},\text{LDF2},\text{N}+1)$ – REAL (KIND=nag_wp) array *Input/Output*

On entry: the values of the right-side of the Helmholtz equation and boundary values (if any).

$$\text{F}(i, j, k) = f(x_i, y_j, z_k) \qquad i = 2, 3, \ldots, \text{L}, j = 2, 3, \ldots, \text{M} \text{ and } k = 2, 3, \ldots, \text{N}.$$

On the boundaries F is defined by

LBDCND	$\mathrm{F}(1, j, k)$	$\mathrm{F}(\mathrm{L}+1, j, k)$	
0	$f(\mathrm{XS}, y_j, z_k)$	$f(\mathrm{XS}, y_j, z_k)$	
1	$u(\mathrm{XS}, y_j, z_k)$	$u(\mathrm{XF}, y_j, z_k)$	
2	$u(\mathrm{XS}, y_j, z_k)$	$f(\mathrm{XF}, y_j, z_k)$	$j = 1, 2, \ldots, \mathrm{M}+1$
3	$f(\mathrm{XS}, y_j, z_k)$	$f(\mathrm{XF}, y_j, z_k)$	$k = 1, 2, \ldots, \mathrm{N}+1$
4	$f(\mathrm{XS}, y_j, z_k)$	$u(\mathrm{XF}, y_j, z_k)$	

MBDCND	$\mathrm{F}(i, 1, k)$	$\mathrm{F}(i, \mathrm{M}+1, k)$	
0	$f(x_i, \mathrm{YS}, z_k)$	$f(x_i, \mathrm{YS}, z_k)$	
1	$u(\mathrm{YS}, x_i, z_k)$	$u(\mathrm{YF}, x_i, z_k)$	
2	$u(x_i, \mathrm{YS}, z_k)$	$f(x_i, \mathrm{YF}, z_k)$	$i = 1, 2, \ldots, \mathrm{L}+1$
3	$f(x_i, \mathrm{YS}, z_k)$	$f(x_i, \mathrm{YF}, z_k)$	$k = 1, 2, \ldots, \mathrm{N}+1$
4	$f(x_i, \mathrm{YS}, z_k)$	$u(x_i, \mathrm{YF}, z_k)$	

NBDCND	$\mathrm{F}(i, j, 1)$	$\mathrm{F}(i, j, \mathrm{N}+1)$	
0	$f(x_i, y_j, \mathrm{ZS})$	$f(x_i, y_j, \mathrm{ZS})$	
1	$u(x_i, y_j, \mathrm{ZS})$	$u(x_i, y_j, \mathrm{ZF})$	
2	$u(x_i, y_j, \mathrm{ZS})$	$f(x_i, y_j, \mathrm{ZF})$	$i = 1, 2, \ldots, \mathrm{L}+1$
3	$f(x_i, y_j, \mathrm{ZS})$	$f(x_i, y_j, \mathrm{ZF})$	$j = 1, 2, \ldots, \mathrm{M}+1$
4	$f(x_i, y_j, \mathrm{ZS})$	$u(x_i, y_j, \mathrm{ZF})$	

Note: if the table calls for both the solution u and the right-hand side f on a boundary, then the solution must be specified.

On exit: contains the solution $u(i, j, k)$ of the finite difference approximation for the grid point (x_i, y_j, z_k), for $i = 1, 2, \ldots, \mathrm{L}+1$, $j = 1, 2, \ldots, \mathrm{M}+1$ and $k = 1, 2, \ldots, \mathrm{N}+1$.

23: PERTRB – REAL (KIND=nag_wp) *Output*

On exit: $\mathrm{PERTRB} = 0$, unless a solution to Poisson's equation $(\lambda = 0)$ is required with a combination of periodic or derivative boundary conditions (LBDCND, MBDCND and $\mathrm{NBDCND} = 0$ or 3). In this case a solution may not exist. PERTRB is a constant, calculated and subtracted from the array F, which ensures that a solution exists. D03FAF then computes this solution, which is a least squares solution to the original approximation. This solution is not unique and is unnormalized. The value of PERTRB should be small compared to the right-hand side F, otherwise a solution has been obtained to an essentially different problem. This comparison should always be made to ensure that a meaningful solution has been obtained.

24: W(LWRK) – REAL (KIND=nag_wp) array *Workspace*

25: LWRK – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D03FAF is called. $2 \times (\mathrm{N}+1) \times \max(\mathrm{L}, \mathrm{M}) + 3 \times \mathrm{L} + 3 \times \mathrm{M} + 4 \times \mathrm{N} + 6$ is an upper bound on the required size of W. If LWRK is too small, the routine exits with $\mathrm{IFAIL} = 2$, and if on entry $\mathrm{IFAIL} = 0$ or -1, a message is output giving the exact value of LWRK required to solve the current problem.

26: 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: $\mathrm{IFAIL} = 0$ 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 $= 1$

On entry, XS $\geq$ XF,
or L < 5,
or LBDCND < 0,
or LBDCND > 4,
or YS $\geq$ YF,
or M < 5,
or MBDCND < 0,
or MBDCND > 4,
or ZS $\geq$ ZF,
or N < 5,
or NBDCND < 0,
or NBDCND > 4,
or LDF $<$ L $+ 1$,
or LDF2 $<$ M $+ 1$.

IFAIL $= 2$

On entry, LWRK is too small.

IFAIL $= 3$

On entry, $\lambda > 0$.

7 Accuracy

Not applicable.

8 Further Comments

The execution time is roughly proportional to $\mathrm{L} \times \mathrm{M} \times \mathrm{N} \times (\log_2 \mathrm{L} + \log_2 \mathrm{M} + 5)$, but also depends on input parameters LBDCND and MBDCND.

9 Example

This example solves the Helmholz equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} + \lambda u = f(x, y, z)$$

for $(x, y, z) \in [0, 1] \times [0, 2\pi] \times \left[0, \frac{\pi}{2}\right]$, where $\lambda = -2$, and $f(x, y, z)$ is derived from the exact solution

$$u(x, y, z) = x^4 \sin y \cos z.$$

The equation is subject to the following boundary conditions, again derived from the exact solution given above.

$u(0, y, z)$ and $u(1, y, z)$ are prescribed (i.e., LBDCND $= 1$).

$u(x, 0, z) = u(x, 2\pi, z)$ (i.e., MBDCND $= 0$).

$u(x, y, 0)$ and $u_x\left(x, y, \frac{\pi}{2}\right)$ are prescribed (i.e., NBDCND $= 2$).

9.1 Program Text

```
    Program d03fafe

!     D03FAF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: d03faf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: dx, dy, dz, error, lambda, pertrb,   &
                                         t, xf, xs, yf, ys, yy, zf, zs, zz
      Integer                         :: i, ifail, j, k, l, lbdcnd, ldf,      &
                                         ldf2, lwrk, m, maxlm, mbdcnd, n,     &
                                         nbdcnd
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: bdxf(:,:), bdxs(:,:), bdyf(:,:),     &
                                         bdys(:,:), bdzf(:,:), bdzs(:,:),     &
                                         f(:,:,:), w(:), x(:), y(:), z(:)
!     .. Intrinsic Procedures ..
      Intrinsic                       :: abs, cos, real, sin
!     .. Executable Statements ..
      Write (nout,*) 'D03FAF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) l, m, n, maxlm
      ldf = l + 1
      ldf2 = m + 1
      lwrk = 2*(n+1)*maxlm + 3*l + 3*m + 4*n + 6000
      Allocate (bdxf(ldf2,n+1),bdxs(ldf2,n+1),bdyf(ldf,n+1),bdys(ldf,n+1), &
        bdzf(ldf,m+1),bdzs(ldf,m+1),f(ldf,ldf2,n+1),w(lwrk),x(l+1),y(m+1), &
        z(n+1))
      Read (nin,*) lambda
      Read (nin,*) xs, xf
      Read (nin,*) ys, yf
      Read (nin,*) zs, zf
      Read (nin,*) lbdcnd, mbdcnd, nbdcnd

!     Define the grid points for later use.

      dx = (xf-xs)/real(l,kind=nag_wp)
      Do i = 1, l + 1
        x(i) = xs + real(i-1,kind=nag_wp)*dx
      End Do
      dy = (yf-ys)/real(m,kind=nag_wp)
      Do j = 1, m + 1
        y(j) = ys + real(j-1,kind=nag_wp)*dy
      End Do
      dz = (zf-zs)/real(n,kind=nag_wp)
      Do k = 1, n + 1
        z(k) = zs + real(k-1,kind=nag_wp)*dz
      End Do

!     Define the array of derivative boundary values.

      Do j = 1, m + 1
        yy = sin(y(j))
        bdzf(1:l+1,j) = -yy*x(1:l+1)**4
      End Do

!     Note that for this example all other boundary arrays are
!     dummy variables.

!     We define the function boundary values in the F array.

      f(1,1:m+1,1:n+1) = 0.0_nag_wp
```

```
      Do k = 1, n + 1
        zz = cos(z(k))
        Do j = 1, m + 1
          f(l+1,j,k) = zz*sin(y(j))
        End Do
      End Do
      f(1:l+1,1:m+1,1) = -bdzf(1:l+1,1:m+1)

!     Define the values of the right hand side of the Helmholtz
!     equation.

      Do k = 2, n + 1
        zz = 4.0_nag_wp*cos(z(k))
        Do j = 1, m + 1
          yy = sin(y(j))*zz
          Do i = 2, l
            f(i,j,k) = yy*x(i)**2*(3.0_nag_wp-x(i)**2)
          End Do
        End Do
      End Do

!     Call D03FAF to generate and solve the finite difference equation.
!     ifail: behaviour on error exit
!            =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call d03faf(xs,xf,l,lbdcnd,bdxs,bdxf,ys,yf,m,mbdcnd,bdys,bdyf,zs,zf,n, &
        nbdcnd,bdzs,bdzf,lambda,ldf,ldf2,f,pertrb,w,lwrk,ifail)

!     Compute discretization error.  The exact solution to the
!     problem is

!     U(X,Y,Z) = X**4*SIN(Y)*COS(Z)

      error = 0.0_nag_wp
      Do k = 1, n + 1
        zz = cos(z(k))
        Do j = 1, m + 1
          yy = sin(y(j))*zz
          Do i = 1, l + 1
            t = abs(f(i,j,k)-yy*x(i)**4)
            If (t>error) error = t
          End Do
        End Do
      End Do
      Write (nout,*)
      Write (nout,99999) error

99999 Format (1X,'Maximum component of discretization error =',1P,E13.6)
    End Program d03fafe
```

9.2 Program Data

```
D03FAF Example Program Data
  16 32 20 32                 : l, m, n, maxlm
  -2.0                        : lambda
  0.0 1.0                     : xs, xf
  0.0 6.28318530717958647692  : ys, yf
  0.0 1.57079632679489661923  : zs, zf
  1 0 2                       : lbdcnd, mbdcnd, nbdcnd
```

9.3 Program Results

```
 D03FAF Example Program Results

 Maximum component of discretization error = 5.176553E-04
```