NAG Library Routine Document

D06DAF

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

D06DAF is a utility which performs an affine transformation of a given mesh.

2 Specification

```
SUBROUTINE D06DAF (NV, NEDGE, NELT, NTRANS, ITYPE, TRANS, COORI, EDGEI,     &
                   CONNI, COORO, EDGEO, CONNO, ITRACE, RWORK, LRWORK,       &
                   IFAIL)
INTEGER            NV, NEDGE, NELT, NTRANS, ITYPE(NTRANS),                  &
                   EDGEI(3,NEDGE), CONNI(3,NELT), EDGEO(3,NEDGE),           &
                   CONNO(3,NELT), ITRACE, LRWORK, IFAIL
REAL (KIND=nag_wp) TRANS(6,NTRANS), COORI(2,NV), COORO(2,NV),               &
                   RWORK(LRWORK)
```

3 Description

D06DAF generates a mesh (coordinates, triangle/vertex connectivities and edge/vertex connectivities) resulting from an affine transformation of a given mesh. This transformation is of the form $Y = A \times X + B$, where

Y, X and B are in $\mathbb{R}^2$, and

A is a real 2 by 2 matrix.

Such a transformation includes a translation, a rotation, a scale reduction or increase, a symmetric transformation with respect to a user-supplied line, a user-supplied analytic transformation, or a composition of several transformations.

This routine is partly derived from material in the MODULEF package from INRIA (Institut National de Recherche en Informatique et Automatique).

4 References

None.

5 Arguments

1: NV – INTEGER *Input*

On entry: the total number of vertices in the input mesh.

Constraint: NV ≥ 3.

2: NEDGE – INTEGER *Input*

On entry: the number of the boundary or interface edges in the input mesh.

Constraint: NEDGE ≥ 1.

3: NELT – INTEGER *Input*

On entry: the number of triangles in the input mesh.

Constraint: NELT $\leq 2 \times$ NV $- 1$.

4: NTRANS – INTEGER *Input*

On entry: the number of transformations of the input mesh.

Constraint: NTRANS ≥ 1.

5: ITYPE(NTRANS) – INTEGER array *Input*

On entry: ITYPE(i), for $i = 1, 2, \ldots,$ NTRANS, indicates the type of each transformation as follows:

ITYPE(i) $= 0$
Identity transformation.

ITYPE(i) $= 1$
Translation.

ITYPE(i) $= 2$
Symmetric transformation with respect to a user-supplied line.

ITYPE(i) $= 3$
Rotation.

ITYPE(i) $= 4$
Scaling.

ITYPE(i) $= 10$
User-supplied analytic transformation.

Note that the transformations are applied in the order described in ITYPE.

Constraint: ITYPE(i) $= 0$, 1, 2, 3, 4 or 10, for $i = 1, 2, \ldots,$ NTRANS.

6: TRANS(6, NTRANS) – REAL (KIND=nag_wp) array *Input*

On entry: the arguments for each transformation. For $i = 1, 2, \ldots,$ NTRANS, TRANS$(1, i)$ to TRANS$(6, i)$ contain the arguments of the ith transformation.

If ITYPE(i) $= 0$, elements TRANS$(1, i)$ to TRANS$(6, i)$ are not referenced.

If ITYPE(i) $= 1$, the translation vector is $\vec{u} = \begin{pmatrix} a \\ b \end{pmatrix}$, where $a =$ TRANS$(1, i)$ and $b =$ TRANS$(2, i)$, while elements TRANS$(3, i)$ to TRANS$(6, i)$ are not referenced.

If ITYPE(i) $= 2$, the user-supplied line is the curve $\{(x, y) \in \mathbb{R}^2;$ such that $ax + by + c = 0\}$, where $a =$ TRANS$(1, i)$, $b =$ TRANS$(2, i)$ and $c =$ TRANS$(3, i)$, while elements TRANS$(4, i)$ to TRANS$(6, i)$ are not referenced.

If ITYPE(i) $= 3$, the centre of the rotation is (x_0, y_0) where $x_0 =$ TRANS$(1, i)$ and $y_0 =$ TRANS$(2, i)$, $\theta =$ TRANS$(3, i)$ is its angle in degrees, while elements TRANS$(4, i)$ to TRANS$(6, i)$ are not referenced.

If ITYPE(i) $= 4$, $a =$ TRANS$(1, i)$ is the scaling coefficient in the x-direction, $b =$ TRANS$(2, i)$ is the scaling coefficient in the y-direction, and (x_0, y_0) are the scaling centre coordinates, with $x_0 =$ TRANS$(3, i)$ and $y_0 =$ TRANS$(4, i)$; while elements TRANS$(5, i)$ to TRANS$(6, i)$ are not referenced.

If ITYPE(i) $= 10$, the user-supplied analytic affine transformation $Y = A \times X + B$ is such that $A = (a_{kl})_{1 \leq k,l \leq 2}$ and $B = (b_k)_{1 \leq k \leq 2}$ where $a_{kl} =$ TRANS$(2 \times (k - 1) + l, i)$, and $b_k =$ TRANS$(4 + k, i)$ with $k, l = 1, 2$.

7: COORI(2, NV) – REAL (KIND=nag_wp) array *Input/Output*

On entry: COORI$(1, i)$ contains the x coordinate of the ith vertex of the input mesh, for $i = 1, 2, \ldots,$ NV; while COORI$(2, i)$ contains the corresponding y coordinate.

On exit: see Section 9.

8: EDGEI(3, NEDGE) – INTEGER array *Input/Output*

On entry: the specification of the boundary or interface edges. EDGEI$(1, j)$ and EDGEI$(2, j)$ contain the vertex numbers of the two end points of the jth boundary edge. EDGEI$(3, j)$ is a user-supplied tag for the jth boundary edge.

Constraint: $1 \leq$ EDGEI$(i, j) \leq$ NV and EDGEI$(1, j) \neq$ EDGEI$(2, j)$, for $i = 1, 2$ and $j = 1, 2, \ldots,$ NEDGE.

On exit: see Section 9.

9: CONNI(3, NELT) – INTEGER array *Input/Output*

On entry: the connectivity of the input mesh between triangles and vertices. For each triangle j, CONNI(i, j) gives the indices of its three vertices (in anticlockwise order), for $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NELT.

Constraints:

$1 \leq$ CONNI$(i, j) \leq$ NV;
CONNI$(1, j) \neq$ CONNI$(2, j)$;
CONNI$(1, j) \neq$ CONNI$(3, j)$ and CONNI$(2, j) \neq$ CONNI$(3, j)$, for $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NELT.

On exit: see Section 9.

10: COORO(2, NV) – REAL (KIND=nag_wp) array *Output*

On exit: COORO$(1, i)$ will contain the x coordinate of the ith vertex of the transformed mesh, for $i = 1, 2, \ldots,$ NV; while COORO$(2, i)$ will contain the corresponding y coordinate.

11: EDGEO(3, NEDGE) – INTEGER array *Output*

On exit: the specification of the boundary or interface edges of the transformed mesh. If the number of symmetric transformations is even or zero thenEDGEO$(i, j) =$ EDGEI(i, j), for $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NEDGE; otherwise EDGEO$(1, j) =$ EDGEI$(2, j)$,EDGEO$(2, j) =$ EDGEI$(1, j)$ and EDGEO$(3, j) =$ EDGEI$(3, j)$, for $j = 1, 2, \ldots,$ NEDGE.

12: CONNO(3, NELT) – INTEGER array *Output*

On exit: the connectivity of the transformed mesh between triangles and vertices. If the number of symmetric transformations is even or zero thenCONNO$(i, j) =$ CONNI(i, j), for $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NELT; otherwise CONNO$(1, j) =$ CONNI$(1, j)$, CONNO$(2, j) =$ CONNI$(3, j)$ and CONNO$(3, j) =$ CONNI$(2, j)$, for $j = 1, 2, \ldots,$ NELT.

13: ITRACE – INTEGER *Input*

On entry: the level of trace information required from D06DAF.

ITRACE ≤ 0
No output is generated.

ITRACE ≥ 1
Details of each transformation, the matrix A and the vector B of the final transformation, which is the composition of all the NTRANS transformations, are printed on the current advisory message unit (see X04ABF).

14: RWORK(LRWORK) – REAL (KIND=nag_wp) array *Workspace*

15: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D06DAF is called.

Constraint: LRWORK $\geq 12 \times$ NTRANS.

16: 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 $= 1$

On entry, NV < 3;
or NELT $> 2 \times$ NV $- 1$;
or NEDGE < 1;
or EDGEI$(i,j) < 1$ or EDGEI$(i,j) >$ NV for some $i = 1, 2$ and $j = 1, 2, \ldots,$ NEDGE;
or EDGEI$(1,j) =$ EDGEI$(2,j)$ for some $j = 1, 2, \ldots,$ NEDGE;
or CONNI$(i,j) < 1$ or CONNI$(i,j) >$ NV for some $i = 1, 2, 3$ and $j = 1, 2, \ldots,$ NELT;
or CONNI$(1,j) =$ CONNI$(2,j)$ or CONNI$(1,j) =$ CONNI$(3,j)$ or
CONNI$(2,j) =$ CONNI$(3,j)$ for some $j = 1, 2, \ldots,$ NELT;
or NTRANS < 1;
or ITYPE$(i) \neq$ 0, 1, 2, 3, 4 or 10 for some $i = 1, 2, \ldots,$ NTRANS;
or LRWORK $< 12 \times$ NTRANS.

IFAIL $= 2$

A serious error has occurred in an internal call to an auxiliary routine. Check the input mesh especially the triangles/vertices and the edges/vertices connectivities as well as the details of each transformations.

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

D06DAF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

You may not wish to save the input mesh (COORI, EDGEI and CONNI) and could call D06DAF using the same arguments for the input and the output (transformed) mesh.

10 Example

For an example of the use of this utility routine, see Section 10 in D06DBF.