NAG Toolbox

nag_pde_2d_gen_order2_checkgrid (d03ry)

1 Purpose

nag_pde_2d_gen_order2_checkgrid (d03ry) is designed to be used in conjunction with nag_pde_2d_gen_order2_rectilinear (d03rb). It can be called from the **inidom** to check the user-specified initial grid data and to obtain a simple graphical representation of the initial grid.

2 Syntax

```
[pgrid, ifail] = nag_pde_2d_gen_order2_checkgrid(nx, ny, lrow, irow, icol, llbnd, ilbnd, lbnd, 'npts', npts, 'nrows', nrows, 'nbnds', nbnds, 'nbpts', nbpts)
[pgrid, ifail] = d03ry(nx, ny, lrow, irow, icol, llbnd, ilbnd, lbnd, 'npts', npts, 'nrows', nrows, 'nbnds', nbnds, 'nbpts', nbpts)
```

3 Description

nag_pde_2d_gen_order2_checkgrid (d03ry) outputs a character array which can be printed to provide a simple graphical representation of the virtual and base grids supplied to nag_pde_2d_gen_order2_rectilinear (d03rb). It must be called only from within the **inidom** after all output arguments of **inidom** (other than **ierr**) have been set. nag_pde_2d_gen_order2_checkgrid (d03ry) also checks the validity of the grid data specified in **inidom**.

You are strongly advised to call nag_pde_2d_gen_order2_checkgrid (d03ry) during the initial call of nag_pde_2d_gen_order2_rectilinear (d03rb) (at least) and to print the resulting character array in order to check that the base grid is exactly as required.

nag_pde_2d_gen_order2_checkgrid (d03ry) writes a representation of each point in the virtual and base grids to the character array **pgrid** as follows:

– internal base grid points are written as two dots (`..`);

– boundary base grid points are written as the **ilbnd** value (i.e., the type) of the boundary;

– points external to the base grid are written as `XX`.

As an example, consider a rectangular domain with a rectangular hole in which the virtual domain extends by one base grid point beyond the actual domain in all directions. The output when each row of **pgrid** is printed consecutively is as follows:

```
XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 XX 23  3  3  3  3  3  3  3  3  3  3 34 XX
 XX  2 .. .. .. .. .. .. .. .. .. ..  4 XX
 XX  2 .. .. .. .. .. .. .. .. .. ..  4 XX
 XX  2 .. .. 14  1  1  1  1 21 .. ..  4 XX
 XX  2 .. ..  4 XX XX XX XX  2 .. ..  4 XX
 XX  2 .. ..  4 XX XX XX XX  2 .. ..  4 XX
 XX  2 .. ..  4 XX XX XX XX  2 .. ..  4 XX
 XX  2 .. ..  4 XX XX XX XX  2 .. ..  4 XX
 XX  2 .. ..  4 XX XX XX XX  2 .. ..  4 XX
 XX  2 .. .. 43  3  3  3  3 32 .. ..  4 XX
 XX  2 .. .. .. .. .. .. .. .. .. ..  4 XX
 XX  2 .. .. .. .. .. .. .. .. .. ..  4 XX
 XX 12  1  1  1  1  1  1  1  1  1  1 41 XX
 XX XX XX XX XX XX XX XX XX XX XX XX XX XX
```

4 References

None.

5 Parameters

5.1 Compulsory Input Parameters

1: **nx** – INTEGER

2: **ny** – INTEGER

The number of virtual grid points in the x- and y-direction respectively (including the boundary points).

Constraint: **nx** and **ny** ≥ 4.

3: **lrow**(**nrows**) – INTEGER array

lrow(i), for $i = 1, 2, \ldots,$ **nrows**, contains the base grid index of the first grid point in base grid row i.

Constraints:

$1 \leq$ **lrow**$(i) \leq$ **npts**, for $i = 1, 2, \ldots,$ **nrows**;
lrow$(i-1) <$ **lrow**(i), for $i = 2, 3, \ldots,$ **nrows**.

4: **irow**(**nrows**) – INTEGER array

irow(i), for $i = 1, 2, \ldots,$ **nrows**, contains the virtual grid row number that corresponds to base grid row i.

Constraints:

$0 \leq$ **irow**$(i) \leq$ **ny**, for $i = 1, 2, \ldots,$ **nrows**;
irow$(i-1) <$ **irow**(i), for $i = 2, 3, \ldots,$ **nrows**.

5: **icol**(**npts**) – INTEGER array

icol(i), for $i = 1, 2, \ldots,$ **npts**, contains the virtual grid column number that contains base grid point i.

Constraint: $0 \leq$ **icol**$(i) \leq$ **nx**, for $i = 1, 2, \ldots,$ **npts**.

6: **llbnd**(**nbnds**) – INTEGER array

llbnd(i), for $i = 1, 2, \ldots,$ **nbnds**, contains the element of **lbnd** corresponding to the start of the ith boundary (or corner).

Constraints:

$1 \leq$ **llbnd**$(i) \leq$ **nbpts**, for $i = 1, 2, \ldots,$ **nbnds**;
llbnd$(i-1) <$ **llbnd**(i), for $i = 2, 3, \ldots,$ **nbnds**.

7: **ilbnd**(**nbnds**) – INTEGER array

ilbnd(i), for $i = 1, 2, \ldots,$ **nbnds**, contains the type of the ith boundary (or corner), as defined in nag_pde_2d_gen_order2_rectilinear (d03rb).

Constraint: **ilbnd**(i) must be equal to one of the following: 1, 2, 3, 4, 12, 23, 34, 41, 21, 32, 43 or 14, for $i = 1, 2, \ldots,$ **nbnds**.

8: **lbnd**(**nbpts**) – INTEGER array

lbnd(i), for $i = 1, 2, \ldots,$ **nbpts**, contains the grid index of the ith boundary point.

Constraint: $1 \leq$ **lbnd**$(i) \leq$ **npts**, for $i = 1, 2, \ldots,$ **nbpts**.

5.2 Optional Input Parameters

1: **npts** – INTEGER

Default: the dimension of the array **icol**.

The total number of points in the base grid.

Constraint: **npts** $\leq$ **nx** $\times$ **ny**.

2: **nrows** – INTEGER

Default: the dimension of the arrays **lrow**, **irow**. (An error is raised if these dimensions are not equal.)

The total number of rows of the virtual grid that contain base grid points.

Constraint: $4 \leq$ **nrows** $\leq$ **ny**.

3: **nbnds** – INTEGER

Default: the dimension of the arrays **llbnd**, **ilbnd**. (An error is raised if these dimensions are not equal.)

The total number of physical boundaries and corners in the base grid.

Constraint: **nbnds** ≥ 8.

4: **nbpts** – INTEGER

Default: the dimension of the array **lbnd**.

The total number of boundary points in the base grid.

Constraint: $12 \leq$ **nbpts** $<$ **npts**.

5.3 Output Parameters

1: **pgrid**(**ny**) – CHARACTER(*) array

pgrid(i), for $i = 1, 2, \ldots,$ **ny**, contains a graphical representation of row **ny** $- i + 1$ of the virtual grid (see Section 3).

2: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **nx** or **ny** < 4,
or **npts** $>$ **nx** $\times$ **ny**,
or **nrows** < 4,
or **nrows** $>$ **ny**,
or **nbnds** < 8,
or **nbpts** < 12,
or **nbpts** $\geq$ **npts**,
or **lrow**(i) < 1, for some $i = 1, 2, \ldots,$ **nrows**,
or **lrow**(i) $>$ **npts**, for some $i = 1, 2, \ldots,$ **nrows**,
or **lrow**(i) $\leq$ **lrow**($i - 1$), for some $i = 2, 3, \ldots,$ **nrows**,
or **irow**(i) < 0, for some $i = 1, 2, \ldots,$ **nrows**,
or **irow**(i) $>$ **ny**, for some $i = 1, 2, \ldots,$ **nrows**,
or **irow**(i) $\leq$ **irow**($i - 1$), for some $i = 2, 3, \ldots,$ **nrows**,

or **icol**$(i) < 0$, for some $i = 1, 2, \ldots,$ **npts**,
or **icol**$(i) >$ **nx**, for some $i = 1, 2, \ldots,$ **npts**,
or **llbnd**$(i) < 1$, for some $i = 1, 2, \ldots,$ **nbnds**,
or **llbnd**$(i) >$ **nbpts**, for some $i = 1, 2, \ldots,$ **nbnds**,
or **llbnd**$(i) \leq$ **llbnd**$(i-1)$, for some $i = 2, 3, \ldots,$ **nbpts**,
or **ilbnd**$(i) \neq 1$, 2, 3, 4, 12, 23, 34, 41, 21, 32, 43 or 14, for some $i = 1, 2, \ldots,$ **nbnds**,
or **lbnd**$(i) < 1$, for some $i = 1, 2, \ldots,$ **nbpts**,
or **lbnd**$(i) >$ **npts**, for some $i = 1, 2, \ldots,$ **nbpts**,
or $leniwk <$ **nx** $\times$ **ny** $+ 1$,
or LEN (**pgrid**(1)) $< 3 \times$ **nx**.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

See Section 10 in nag_pde_2d_gen_order2_rectilinear (d03rb).

9.1 Program Text

```
function d03ry_example

fprintf('d03ry example results\n\n');

nx = nag_int(11);
ny = nag_int(11);
lrow = nag_int([1, 4, 15:11:37, 46:11:79, 88, 97]);
irow = nag_int([0:10]);
icol = nag_int([0:2, 0:10, 0:10, 0:10, 0:5, 8:10, ...
                0:10, 0:10, 0:10, 0:8, 0:9, 0:8]);
llbnd = nag_int([1, 2,11,18,19,24,31,37,42,48,53,55,56,58,59,60,61,62, ...
                 63,64,65,66,67,68,69,70,71,72]);
ilbnd = nag_int([1, 2, 3, 4, 1, 4, 1, 2, 3, 4, 3, 4, 1, 2,12,23,34,41, ...
                 14,41,12,23,34,41,43,14,21,32]);
lbnd  = nag_int([2, 4, 15:11:37, 46:11:79, 88, 98:104, ...
                96, 86:-1:82, 70:-11:48, 39:-11:6, 8:13, ...
                18:11:40, 49, 60, 72:77, 67:-11:45, 36, 25, ...
                33, 32, 42, 52, 53, 43, 1, 97, 105, 87, 81, ...
                3, 7, 71, 78, 14, 31, 51, 54, 34]);

[pgrid, ifail] = d03ry(nx, ny, lrow, irow, icol, llbnd, ilbnd, lbnd);

disp('Textual representation of grid:');
disp(pgrid);
```

9.2 Program Results

```
     d03ry example results

Textual representation of grid:
    '  3  3  3  3  3  3 34 .. 23 XX'
    '  2 .. .. .. .. .. .. ..  4 XX XX'
    '  2 .. 14  1  1  1  1  1 41 XX XX'
    '  2 ..  4 23  3  3  3  3  3  3 34'
    '  2 ..  4  2 .. .. .. .. .. ..  4'
    '  2 ..  4  2 .. 14  1  1 21 ..  4'
    '  2 ..  4  2 ..  4 XX XX  2 ..  4'
    '  2 ..  4  2 .. 43  3  3 32 ..  4'
    '  2 ..  4  2 .. .. .. .. .. ..  4'
    '  2 ..  4 12  1  1  1  1  1  1 41'
    ' 12  1 41 XX XX XX XX XX XX XX XX'
```