NAG Toolbox for MATLAB

d02nu

1 Purpose

d02nu is a setup function which must be called prior to an integrator in sub-chapter D02M/N, if sparse matrix linear algebra is required.

2 Syntax

```
[jacpvt, rwork, ifail] = d02nu(neq, ldysav, jceval, nwkjac, ia, ja,
njcpvt, sens, u, eta, lblock, rwork, 'nia', nia, 'nja', nja, 'isplit',
isplit)
```

3 Description

d02nu defines the linear algebra to be used as sparse matrix linear algebra, permits you to specify the method for calculating the Jacobian and its structure, and checks the validity of certain input values.

4 References

See the D02M/N Sub-chapter Introduction.

5 Parameters

5.1 Compulsory Input Parameters

1: **neq – int32 scalar**

The number of differential equations.

Constraint: $1 \leq$ **neq** $\leq$ **ldysav**.

2: **ldysav – int32 scalar**

A bound on the maximum number of differential equations to be solved during the integration.

Constraint: **neq** $\leq$ **ldysav**.

3: **jceval – string**

Specifies the technique to be used to compute the Jacobian.

jceval = 'N'

The sparsity structure and the value of the Jacobian are to be determined numerically by the integrator.

jceval = 'S'

The sparsity structure of the Jacobian is supplied in the arrays **ia** and **ja** but its value is to be determined numerically. This is the recommended mode of operation unless it is a simple matter to supply the Jacobian.

jceval = 'A'

The Jacobian will be evaluated by calls to (sub)program **jac**. The sparsity structure will be estimated by calls to **jac**; that is, no explicit sparsity structure need be supplied in the arrays **ia** and **ja**.

jceval = 'F'

The sparsity structure of the Jacobian is supplied in **ia** and **ja**, and its value will be determined by calls to (sub)program **jac**. This is the recommended mode of operation if the **jac** is simple to form.

jceval = 'D'

The default choice is to be made. In this case 'D' is interpreted as 'S'.

If the sparsity structure is supplied in arrays **ia** and **ja**, then any evidence from the numerical or analytical formation of the Jacobian that this structure is not correct, is ignored.

Only the first character of the actual parameter **jceval** is passed to d02nu; hence it is permissible for the actual argument to be more descriptive, e.g., 'Numerical', 'Structural', 'Analytical', 'Full information' or 'Default' in a call to d02nu.

If the option **jceval** = 'N', 'S' or 'D' is used then the actual argument corresponding to (sub)program **jac** in the call to d02nd or d02nj must be either "temp_tag_xref_d02ndz d02njz"done**d02ndz d02njz**missing_entity d02ndz d02njz respectively.

Constraint: **jceval** = 'N', 'S', 'A', 'F' or 'D'.

4: **nwkjac – int32 scalar**

The size of the array **wkjac**, which you are supplying to the integrator, as declared in the (sub)program from which d02nu is called.

Suggested value: **nwkjac** = 4 × **ldysav** if **jceval** = 'N' or 'A'. If **nwkjac** is less than this estimate, then a message is printed on the current advisory message unit (see x04ab), and execution continues.

Constraint: if **jceval** = 'S', 'F' or 'D', **nwkjac** $\geq nelement + 2 \times$ **neq**, where $nelement$ is the total number of nonzeros.

5: **ia**(**nia**) **– int32 array**

If **jceval** = 'S', 'F' or 'D', **ia** must contain details of the sparsity pattern to be used for the Jacobian. See **ja**.

ia is not used if **jceval** = 'N' or 'A'.

6: **ja**(**nja**) **– int32 array**

If **jceval** = 'S', 'F' or 'D', **ja** must contain details of the sparsity pattern to be used for the Jacobian. **ja** contains the row indices where nonzero elements occur, reading in column-wise order, and **ia** contains the starting locations in **ja** of the descriptions of columns $1, 2, \ldots,$ **neq** in that order, with **ia**$(1) = 1$. Thus for each column index $j = 1, 2, \ldots,$ **neq**, the values of the row index i in column j where a nonzero element may occur are given by

$$i = \mathbf{ja}(k)$$

where $\mathbf{ia}(j) \leq k < \mathbf{ia}(j+1)$.

Thus the total number of nonzeros, $nelement$, must be $\mathbf{ia}(\mathbf{neq}+1)-1$. For example, for the following matrix

$$\begin{pmatrix} x & 0 & x & 0 & 0 \\ 0 & x & x & x & 0 \\ x & x & x & 0 & 0 \\ x & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \end{pmatrix}$$

where x represents nonzero elements (13 in all) the arrays **ia** and **ja** should be

$\mathbf{ia}(k)$	1	4	6	9	12	14							
$\mathbf{ja}(k)$	1	3	4	2	3	1	2	3	2	4	5	4	5

ja is not used if **jceval** = 'N' or 'A'.

7: **njcpvt – int32 scalar**

the length of the array **jacpvt**, which you are supplying to the integrator, as dimensioned in the sub(program) from which d02nu is called.

Suggested value: **njcpvt** $= 20 \times$ **ldysav** if **jceval** = 'N' or 'A'. If **njcpvt** is less than this estimate, then a message is printed on the current advisory message unit (see x04ab), and execution continues.

Constraint: if **jceval** = 'S', 'F' or 'D', **njcpvt** $\geq 3 \times nelement + 14 \times$ **neq**, where $nelement$ is the total number of nonzeros.

8: **sens – double scalar**

A threshold parameter used to determine whether or not a matrix element is zero; when **sens** is set to 0.0 on entry, the function will use **sens** $= 100.0 \times$ ***machine precision***. Otherwise the absolute value of **sens** is used.

9: **u – double scalar**

Should have a value between 0.0 and 0.9999. Otherwise a default value of 0.1 is used. When the sparsity pattern has been evaluated, the first Jacobian computed is decomposed with **u** governing the choice of pivots; subsequent Jacobian decompositions use the same pattern of decomposition until the sparsity pattern is re-evaluated. When searching a row for a pivot, any element is excluded from the search which is less than **u** times the largest of those elements in the row available as pivots. Thus decreasing **u** biases the algorithm towards maintaining sparsity at the expense of numerical stability.

10: **eta – double scalar**

A relative pivot threshold, below which on subsequent decompositions (as described under **u**), an internal error is provoked.

eta > 1.0

No check on pivot size is made.

eta ≤ 0.0

The default value **eta** $=$ 1.0D−4 is used.

11: **lblock – logical scalar**

Indicates if preordering is used before decomposition.

If **lblock** = **true**, on entry, the Jacobian matrix is preordered to block lower triangular form before a decomposition is performed (this is the recommended mode). If you know the structure of the Jacobian to be irreducible, that is not permutable to block lower triangular form, then you should set **lblock** = **false**. For example, a Jacobian arising from using the method of lines for parabolic partial differential equations would normally be irreducible. (See the specification of d02nx for optional output concerning **lblock**.)

12: **rwork**(**50** + **4** × **ldysav**) – **double array**

This must be the same workspace array as the array **rwork** supplied to the integrator. It is used to pass information from the setup function to the integrator and therefore the contents of this array must not be changed before calling the integrator.

5.2 Optional Input Parameters

1: **nia** – **int32 scalar**

Default: The dimension of the array **ia**.

Constraints:

if **jceval** = 'S', 'F' or 'D', **nia** ≥ **neq** + 1;
nia ≥ 1 otherwise.

2: **nja** – **int32 scalar**

Default: The dimension of the array **ja**.

Constraints:

if **jceval** = 'S', 'F' or 'D', **nja** ≥ **ia**(**neq** + 1) − 1;
nja ≥ 1 otherwise.

3: **isplit** – **int32 scalar**

This parameter is used for splitting the integer workspace **jacpvt** to effect an efficient decomposition. It must satisfy 1 ≤ **isplit** ≤ 99. If **isplit** lies outside this range on entry, a default value of 73 is used. An appropriate value for **isplit** for subsequent runs on similar problems is available via the optional output d02nx.

Suggested value: **isplit** = 73, unless you have information from a previous run of a similar problem.

Default: 73

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **jacpvt**(**njcpvt**) – **int32 array**

Data relating to the Jacobian sparsity structure.

2: **rwork**(**50** + **4** × **ldysav**) – **double array**

This must be the same workspace array as the array **rwork** supplied to the integrator. It is used to pass information from the setup function to the integrator and therefore the contents of this array must not be changed before calling the integrator.

3: **ifail** – **int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, an illegal input was detected.

7 Accuracy

Not applicable.

8 Further Comments

d02nu must be called as a setup function before a call to either d02nd or d02nj and may be called as the linear algebra setup function before a call to d02nm or d02nn.

9 Example

d02nd_fcn.m

```
function [f, ires] = fcn(neq, t, y, ires)
      f = zeros(3,1);
      f(1) = -0.04d0*y(1) + 1.0d4*y(2)*y(3);
      f(2) = 0.04d0*y(1) - 1.0d4*y(2)*y(3) - 3.0d7*y(2)*y(2);
      f(3) = 3.0d7*y(2)*y(2);
```

d02nd_jac.m

```
function p = jac(neq, t, y, h, d)
  p = zeros(neq, neq);
  hxd = h*d;
  p(1,1) = 1.0d0 - hxd*(-0.04d0);
  p(1,2) = -hxd*(1.0d4*y(3));
  p(1,3) = -hxd*(1.0d4*y(2));
  p(2,1) = -hxd*(0.04d0);
  p(2,2) = 1.0d0 - hxd*(-1.0d4*y(3)-6.0d7*y(2));
  p(2,3) = -hxd*(-1.0d4*y(2));
  p(3,2) = -hxd*(6.0d7*y(2));
  p(3,3) = 1.0d0 - hxd*(0.0d0);
```

d02nd_monitr.m

```
function [hnext, y, imon, inln, hmin, hmax] = ...
    monitr(neq, neqmax, t, hlast, hnext, y, ydot, ysave, r, acor, imon,
hmin, hmax, nqu)
  inln=int32(0);
```

```
neq = int32(3);
t = 0;
tout = 10;
y = [1;
     0;
     0];
rwork = zeros(62, 1);
rtol = [0.0001];
atol = [1e-07];
itol = int32(1);
inform = zeros(23, 1, 'int32');
```

```
ysave = zeros(3, 6);
wkjac = zeros(100, 1);
jacpvt = zeros(150, 1, 'int32');
itask = int32(1);
itrace = int32(0);
[const, rwork, ifail] = ...
    d02nv(int32(3), int32(6), int32(5), 'Newton', false, zeros(6), ...
    0, 1e-10, 10, 0, int32(200), int32(5), 'Average-L2', rwork);
[jacpvt, rwork, ifail] = ...
    d02nu(int32(3), int32(3), 'Numerical', int32(100), int32(0), ...
    int32(0), int32(150), 0, 0.1, 1e-4, true, rwork);
[tOut, yOut, ydot, rworkOut, informOut, ysaveOut, wkjacOut, ...
 jacpvtOut, ifail] = ...
    d02nd(neq, t, tout, y, rwork, rtol, atol, itol, inform, ...
    'd02nd_fcn', ysave, 'd02nd_jac', wkjac, jacpvt, 'd02nd_monitr', ...
    itask, itrace)
```

```
tOut =
    10
yOut =
    0.8414
    0.0000
    0.1586
ydot =
   -0.0075
   -0.0000
    0.0075
rworkOut =
     array elided
informOut =
          55
         136
          16
           4
           4
          78
           0
           3
           0
         100
         150
          29
          71
          16
           7
           3
          73
        1108
           1
           0
           0
           0
           0
ysaveOut =
    0.8355   -0.0067    0.0002   -0.0000    0.0000   -0.0000
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000
    0.1645    0.0067   -0.0002    0.0000   -0.0000    0.0000
wkjacOut =
     array elided
jacpvtOut =
     array elided
ifail =
           0
```