NAG Toolbox for MATLAB

d02lz

1 Purpose

d02lz interpolates components of the solution of a non-stiff system of second-order differential equations from information provided by the integrator d02la, when the low-order method has been used.

2 Syntax

```
[ywant, ypwant, ifail] = d02lz(t, y, yp, nwant, twant, rwork, 'neq',
neq, 'lrwork', lrwork)
```

3 Description

d02lz evaluates the first **nwant** ($\leq$ **neq**) components of the solution of a non-stiff system of second-order ordinary differential equations at any point using a special Runge–Kutta–Nystrom formula (see Dormand and Prince 1986c) and information generated by d02la when the low-order method has been used. This information must be presented unchanged to d02lz. d02lz should not normally be used to extrapolate outside the range of the values from d02la.

4 References

Dormand J R and Prince P J 1986c Runge–Kutta–Nystrom triples *Mathematical Report TP-CS-86-05* Teesside Polytechnic

5 Parameters

5.1 Compulsory Input Parameters

1: **t – double scalar**

t, the current value at which the solution and its derivative have been computed (as returned in parameter **t** on output from d02la).

2: **y**(**neq**) **– double array**

The ith component of the solution at t, for $i = 1, 2, \ldots,$ **neq**, as returned from d02la.

3: **yp**(**neq**) **– double array**

The ith component of the derivative at t, for $i = 1, 2, \ldots,$ **neq**, as returned from d02la.

4: **nwant – int32 scalar**

the number of components of the solution and derivative whose values at **twant** are required. The first **nwant** components are evaluated.

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

5: **twant – double scalar**

The point at which components of the solution and derivative are to be evaluated. **twant** should not normally be an extrapolation point, that is **twant** should satisfy

$$told \leq \mathbf{twant} \leq \mathbf{t},$$

or if integration is proceeding in the negative direction

$$told \geq \mathbf{twant} \geq \mathbf{t},$$

where *told* is the previous integration point which is held in an element of the array **rwork** and is, to within rounding, **t** − **hused**. (**hused** is given by d02ly.) Extrapolation is permitted but not recommended, and **ifail** = 2 is returned whenever extrapolation is attempted.

6: **rwork**(**lrwork**) – **double array**

This **must** be the same parameter **rwork** as supplied to d02la. It is used to pass information from d02la to d02lz and therefore the contents of this array **must not** be changed before calling d02lz.

5.2 Optional Input Parameters

1: **neq – int32 scalar**

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

the number of second-order ordinary differential equations being solved by d02la. It must contain the same value as the parameter **neq** in a prior call of d02la.

2: **lrwork – int32 scalar**

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

This must be the same parameter **lrwork** as supplied to the setup function d02lx.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **ywant**(**nwant**) – **double array**

The calculated value of the ith component of the solution at $t =$ **twant**, for $i = 1, 2, \ldots,$ **nwant**.

2: **ypwant**(**nwant**) – **double array**

The calculated value of the ith component of the derivative at $t =$ **twant**, for $i = 1, 2, \ldots,$ **nwant**.

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

Illegal input detected, i.e., one of the following conditions:

- d02la has not been called;
- one or both of the parameters **neq** and **lrwork** does not match the corresponding parameter supplied to the setup function d02lx;
- no integration steps have been taken since the last call to d02lx with **start** = **true**;
- **nwant** < 1 or **nwant** > **neq**.

This error exit can be caused if elements of **rwork** have been overwritten.

ifail = 2

d02lz has been called for extrapolation. The values of the solution and its derivative at **twant** have been calculated and placed in **ywant** and **ypwant** before returning with this error number (see Section 7).

ifail = 3

d02la last used the high order method to integrate the system of differential equations. Interpolation is not permitted with this method.

7 Accuracy

The error in interpolation is of a similar order to the error arising from the integration using d02la with the lower order method.

The same order of accuracy can be expected when extrapolating using d02lz. However, the actual error in extrapolation will, in general, be much larger than for interpolation.

8 Further Comments

When interpolation for only a few components is required then it is more efficient to order the components of interest so that they are numbered first.

9 Example

```
d02la_fcn.m
```

```
function f = fcn(neq, t, y)
  r = sqrt(y(1)^2+y(2)^2)^3;
  f = zeros(2,1);
  f(1) = -y(1)/r;
  f(2) = -y(2)/r;
```

```
t = 0;
tend = 20;
y = [0.5;
     0];
yp = [0;
     1.732050807568877];
ydp = zeros(2, 1);
rwork = zeros(56,1);
twant = 2;
tnext = 2;
nwant = int32(2);
[startOut, rwork, ifail] = ...
     d02lx(0, 1e-4, zeros(2,1), zeros(2,1), int32(0), true, true, false,
rwork);
fprintf(’\n  T       Y(1)       Y(2)\n’);
fprintf(’%4.1f %10.5f %10.5f\n’, t, y(1), y(2));
while (t < tend && ifail == 0)
  [t, y, yp, ydp, rwork, ifail] = d02la(’d02la_fcn’, t, tend, y, yp, ydp,
rwork);
  while (tnext <= t && ifail == 0)
    [ywant, ypwant, ifail] = d02lz(t, y, yp, nwant, tnext, rwork);
    fprintf(’%4.1f %10.5f %10.5f\n’, tnext, ywant(1), ywant(2));
    tnext = tnext + 2;
  end
end
if (ifail == 0)
  [hnext, hused, hstart, nsucc, nfail, natt, thres, thresp, ifail] =  ...
    d02ly(nwant, rwork);

  fprintf(’\n Number of successful steps = %d\n’, nsucc);
  fprintf(’ Number of failed steps     = %d\n’, nfail);
end
```

```
T       Y(1)       Y(2)
0.0    0.50000    0.00000
```

```
2.0   -1.20573    0.61357
 4.0   -1.33476   -0.47685
 6.0    0.35748   -0.44558
 8.0   -1.03762    0.73022
10.0   -1.42617   -0.32658
12.0    0.05515   -0.72032
14.0   -0.82880    0.81788
16.0   -1.48103   -0.16788
18.0   -0.26719   -0.84223
20.0   -0.57803    0.86339
 Number of successful steps = 108
 Number of failed steps     = 16
```