NAG Toolbox for MATLAB

d03py

1 Purpose

d03py may be used in conjunction with either d03pd or d03pj. It computes the solution and its first derivative at user-specified points in the spatial co-ordinate.

2 Syntax

```
[up, rsave, ifail] = d03py(u, xbkpts, npoly, xp, itype, rsave, 'npde', npde, 'nbkpts', nbkpts, 'npts', npts, 'intpts', intpts, 'lrsave', lrsave)
```

3 Description

d03py is an interpolation function for evaluating the solution of a system of partial differential equations (PDEs), or the PDE components of a system of PDEs with coupled ordinary differential equations (ODEs), at a set of user-specified points. The solution of a system of equations can be computed using d03pd or d03pj on a set of mesh points; d03py can then be employed to compute the solution at a set of points other than those originally used in d03pd or d03pj. It can also evaluate the first derivative of the solution. Polynomial interpolation is used between each of the break points $\mathbf{xbkpts}(i)$, for $i = 1, 2, \ldots, \mathbf{nbkpts}$. When the derivative is needed ($\mathbf{itype} = 2$), the array $\mathbf{xp}(\mathbf{intpts})$ must not contain any of the break points, as the method, and consequently the interpolation scheme, assumes that only the solution is continuous at these points.

4 References

None.

5 Parameters

Note: the parameters **u**, **npts**, **npde**, **xbkpts**, **nbkpts**, **rsave** and **lrsave** must be supplied unchanged from either d03pd or d03pj.

5.1 Compulsory Input Parameters

1: **u**(**npde**,**npts**) – **double array**

The PDE part of the original solution returned in the parameter **u** by the function d03pd or d03pj.

2: **xbkpts**(**nbkpts**) – **double array**

$\mathbf{xbkpts}(i)$, for $i = 1, 2, \ldots, \mathbf{nbkpts}$, must contain the break points as used by d03pd or d03pj.

Constraint: $\mathbf{xbkpts}(1) < \mathbf{xbkpts}(2) < \cdots < \mathbf{xbkpts}(\mathbf{nbkpts})$.

3: **npoly** – **int32 scalar**

The degree of the Chebyshev polynomial used for approximation as used by d03pd or d03pj.

Constraint: $1 \leq \mathbf{npoly} \leq 49$.

4: **xp**(**intpts**) – **double array**

$\mathbf{xp}(i)$, for $i = 1, 2, \ldots, \mathbf{intpts}$, must contain the spatial interpolation points.

Constraint: $\mathbf{xbkpts}(1) \leq \mathbf{xp}(1) < \mathbf{xp}(2) < \cdots < \mathbf{xp}(\mathbf{intpts}) \leq \mathbf{xbkpts}(\mathbf{nbkpts})$.

When **itype** $= 2$, $\mathbf{xp}(i) \neq \mathbf{xbkpts}(j)$, for $i = 1, 2, \ldots, \mathbf{intpts}$ and $j = 2, 3, \ldots, \mathbf{nbkpts} - 1$

5: **itype – int32 scalar**

specifies the interpolation to be performed.

itype $= 1$

The solution at the interpolation points are computed.

itype $= 2$

Both the solution and the first derivative at the interpolation points are computed.

Constraint: **itype** $= 1$ or 2.

6: **rsave**(**lrsave**) **– double array**

The array **rsave** contains information required by d03py as returned by d03pd or d03pj. The contents of **rsave** must not be changed from the call to d03pd or d03pj. Some elements of this array are overwritten on exit.

5.2 Optional Input Parameters

1: **npde – int32 scalar**

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

the number of PDEs.

Constraint: **npde** ≥ 1.

2: **nbkpts – int32 scalar**

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

the number of break points.

Constraint: **nbkpts** ≥ 2.

3: **npts – int32 scalar**

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

the number of mesh points as used by d03pd or d03pj.

Constraint: **npts** $= (\mathbf{nbkpts} - 1) \times \mathbf{npoly} + 1$.

4: **intpts – int32 scalar**

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

the number of interpolation points.

Constraint: **intpts** ≥ 1.

5: **lrsave – int32 scalar**

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

the size of the workspace **rsave**, as in d03pd or d03pj.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **up**(**npde**,**intpts**,**itype**) – **double array**

If **itype** $= 1$, **up**$(i,j,1)$, contains the value of the solution $U_i(x_j, t_{\text{out}})$, at the interpolation points $x_j =$ **xp**(j), for $j = 1, 2, \ldots,$ **intpts** and $i = 1, 2, \ldots,$ **npde**.

If **itype** $= 2$, **up**$(i,j,1)$ contains $U_i(x_j, t_{\text{out}})$ and **up**$(i,j,2)$ contains $\dfrac{\partial U_i}{\partial x}$ at these points.

2: **rsave**(**lrsave**) – **double array**

The array **rsave** contains information required by d03py as returned by d03pd or d03pj. The contents of **rsave** must not be changed from the call to d03pd or d03pj. Some elements of this array are overwritten on exit.

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, **itype** $\neq 1$ or 2,
or **npoly** < 1,
or **npde** < 1,
or **nbkpts** < 2,
or **intpts** < 1,
or **npts** $\neq$ (**nbkpts** $- 1$) $\times$ **npoly** $+ 1$,
or **xbkpts**(i), for $i = 1, \ldots,$ **nbkpts**, are not ordered.

ifail $= 2$

On entry, the interpolation points **xp**(i), for $i = 1, \ldots,$ **intpts**, are not in strictly increasing order, or when **itype** $= 2$, at least one of the interpolation points stored in **xp** is equal to one of the break points stored in **xbkpts**.

ifail $= 3$

You are attempting extrapolation, that is, one of the interpolation points **xp**(i), for some i, lies outside the interval [**xbkpts**(1), **xbkpts**(**nbkpts**)]. Extrapolation is not permitted.

7 Accuracy

See the documents for d03pd or d03pj.

8 Further Comments

None.

9 Example

d03pd_bndary.m

```
function [beta, gamma, ires, user] = bndary(npde, t, u, ux, ibnd, ires,
user)
  beta = zeros(npde, 1);
```

```
  gamma = zeros(npde, 1);

  if (ibnd == 0)
    beta(1) = 1;
    gamma(1) = 0;
    beta(2) = 0;
    gamma(2) = u(1) - 1;
  else
    beta(1) = 1;
    gamma(1) = 0;
    beta(2) = 0;
    gamma(2) = u(1) + 1;
  end
```

d03pd_pdedef.m

```
function [p, q, r, ires, user] = pdedef(npde, t, x, nptl, u, ux, ires,
user)
  p = zeros(npde, npde, nptl);
  q = zeros(npde, nptl);
  r = zeros(npde, nptl);

  for i = 1:nptl
    q(1,i) = u(2,i);
    q(2,i) = u(1,i)*ux(2,i) - ux(1,i)*u(2,i);
    r(1,i) = ux(1,i);
    r(2,i) = ux(2,i);
    p(1,1,i) = 0;
    p(1,2,i) = 0;
    p(2,1,i) = 0;
    p(2,2,i) = 1;
  end;
```

d03pd_uinit.m

```
function [u, user] = uinit(npde, npts, x, user)
  u = zeros(npde, npts);

  piby2 = pi/2;
  for i = 1:npts
    u(1,i) = -sin(piby2*x(i));
    u(2,i) = -piby2*piby2*u(1,i);
  end
```

```
m = int32(0);
ts = 0;
tout = 0.0001;
u = zeros(2, 28);
xbkpts = [-1;
     -0.7777777777777778;
     -0.5555555555555556;
     -0.3333333333333333;
     -0.1111111111111111;
     0.1111111111111111;
     0.3333333333333333;
     0.5555555555555556;
     0.7777777777777778;
     1];
npoly = int32(3);
acc = 0.0001;
rsave = zeros(2085, 1);
isave = zeros(80, 1, 'int32');
itask = int32(1);
itrace = int32(0);
ind = int32(0);
```

```
cwsav = {'''; '''; '''; '''; '''; '''; '''; '''; '''; '''};
lwsav = false(100, 1);
iwsav = zeros(505, 1, 'int32');
rwsav = zeros(1100, 1);
xp = [-1;
     -0.6;
     -0.2;
     0.2;
     0.6;
     1];
itype = int32(1);
[ts, u, x, rsave, isave, ind, user, cwsav, lwsav, iwsav, rwsav, ifail] =
...
     d03pd(m, ts, tout, 'd03pd_pdedef', 'd03pd_bndary', u, xbkpts, npoly,
...
    'd03pd_uinit', acc, rsave, isave, itask, itrace, ind, ...
    cwsav, lwsav, iwsav, rwsav);
[up, rsaveOut, ifail] = d03py(u, xbkpts, npoly, xp, itype, rsave)
```

```
up =
    1.0000    0.8090    0.3090   -0.3090   -0.8090   -1.0000
   -2.4850   -1.9957   -0.7623    0.7623    1.9957    2.4850
rsaveOut =
     array elided
ifail =
           0
```