NAG C Library Function Document

nag_pde_interp_1d_fd (d03pzc)

1 Purpose

nag_pde_interp_1d_fd (d03pzc) interpolates in the spatial co-ordinate the solution and derivative of a system of partial differential equations (PDEs). The solution must first be computed using one of the finite difference schemes nag_pde_parab_1d_fd (d03pcc), nag_pde_parab_1d_fd_ode (d03phc) or nag_pde_parab_1d_fd_ode_remesh (d03ppc), or one of the Keller box schemes nag_pde_parab_1d_keller (d03pec), nag_pde_parab_1d_keller_ode (d03pkc) or nag_pde_parab_1d_keller_ode_remesh (d03prc).

2 Specification

```
void nag_pde_interp_1d_fd (Integer npde, Integer m, const double u[],
     Integer npts, const double x[], const double xp[], Integer intpts, Integer itype,
     double up[], NagError *fail)
```

3 Description

nag_pde_interp_1d_fd (d03pzc) is an interpolation function for evaluating the solution of a system of partial differential equations (PDEs), at a set of user-specified points. The solution of the system of equations (possibly with coupled ordinary differential equations) must be computed using a finite difference scheme or a Keller box scheme on a set of mesh points. nag_pde_interp_1d_fd (d03pzc) can then be employed to compute the solution at a set of points anywhere in the range of the mesh. It can also evaluate the first spatial derivative of the solution. It uses linear interpolation for approximating the solution.

4 References

None.

5 Parameters

Note: the parameters **x**, **m**, **u**, **npts** and **npde** must be supplied unchanged from the PDE function.

1: **npde** – Integer *Input*

On entry: the number of PDEs.

Constraint: **npde** ≥ 1.

2: **m** – Integer *Input*

On entry: the co-ordinate system used. If the call to nag_pde_interp_1d_fd (d03pzc) follows one of the finite difference functions then **m** must be the same parameter **m** as used in that call. For the Keller box scheme only Cartesian co-ordinate systems are valid and so **m must** be set to zero. No check will be made by nag_pde_interp_1d_fd (d03pzc) in this case.

m $= 0$

Indicates Cartesian co-ordinates.

m $= 1$

Indicates cylindrical polar co-ordinates.

m $= 2$

Indicates spherical polar co-ordinates.

Constraints:

$0 \leq \mathbf{m} \leq 2$ following a finite difference function;
$\mathbf{m} = 0$ following a Keller box scheme function.

3: **u**[**npde** × **npts**] – const double *Input*

Note: where $\mathbf{U}(i,j)$ appears in this document it refers to the array element **u**[**npde** $\times (j-1)+i-1$]. We recommend using a #define to make the same definition in your calling program.

On entry: the PDE part of the original solution returned in the parameter **u** by the PDE function.

Constraint: **npde** ≥ 1.

4: **npts** – Integer *Input*

On entry: the number of mesh points.

Constraint: **npts** ≥ 3.

5: **x**[**npts**] – const double *Input*

On entry: **x**[$i-1$], for $i = 1,2,\ldots,$ **npts**, must contain the mesh points as used by the PDE function.

6: **xp**[**intpts**] – const double *Input*

On entry: **xp**[$i-1$], for $i = 1,2,\ldots,$ **intpts**, must contain the spatial interpolation points.

Constraint: **x**[0] $\leq$ **xp**[0] $<$ **xp**[1] $< \cdots <$ **xp**[**intpts** -1] $\leq$ **x**[**npts** -1].

7: **intpts** – Integer *Input*

On entry: the number of interpolation points.

Constraint: **intpts** ≥ 1.

8: **itype** – Integer *Input*

On entry: specifies the interpolation to be performed.

If **itype** $= 1$, the solutions at the interpolation points are computed. If **itype** $= 2$, both the solutions and their first derivatives at the interpolation points are computed.

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

9: **up**[**npde** × **intpts** × **itype**] – double *Output*

Note: where $\mathbf{UP}(i,j,k)$ appears in this document it refers to the array element **up**[**npde** $\times$ (**intpts** $\times (k-1)+j-1)+i-1$]. We recommend using a #define to make the same definition in your calling program.

On exit: if **itype** $= 1$, $\mathbf{UP}(i,j,1)$, contains the value of the solution $U_i(x_j, t_{\text{out}})$, at the interpolation points $x_j =$ **xp**[$j-1$], for $j = 1,2,\ldots,$ **intpts**; $i = 1,2,\ldots,$ **npde**.

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

10: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **itype** is not equal to 1 or 2: **itype** = ⟨*value*⟩.

On entry, **m** is not equal to 0, 1, or 2: **m** = ⟨*value*⟩.

O n entry, **intpts** ≤ 0: **intpts** = ⟨*value*⟩.

On entry, **npts** = ⟨*value*⟩.
Constraint: **npts** > 2.

On entry, **npde** = ⟨*value*⟩.
Constraint: **npde** > 0.

NE_EXTRAPOLATION

On entry, interpolating point ⟨*value*⟩ with the value ⟨*value*⟩ is outside the **x** range.

NE_NOT_STRICTLY_INCREASING

On entry, interpolation points **xp** badly ordered: i = ⟨*value*⟩, **xp**$[i-1]$ = ⟨*value*⟩ j = ⟨*value*⟩, **xp**$[j-1]$ = ⟨*value*⟩.

On entry, mesh points **x** badly ordered: i = ⟨*value*⟩, **x**$[i-1]$ = ⟨*value*⟩ j = ⟨*value*⟩, **x**$[j-1]$ = ⟨*value*⟩.

NE_BAD_PARAM

On entry, parameter ⟨*value*⟩ had an illegal value.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy

See the PDE function documents.

8 Further Comments

None.

9 Example

See Section 9 of the documents for nag_pde_parab_1d_fd (d03pcc), nag_pde_parab_1d_fd_ode_remesh (d03ppc) and nag_pde_parab_1d_keller_ode_remesh (d03prc).