NAG Library Routine Document

D03PUF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

D03PUF calculates a numerical flux function using Roe's Approximate Riemann Solver for the Euler equations in conservative form. It is designed primarily for use with the upwind discretization schemes D03PFF, D03PLF or D03PSF, but may also be applicable to other conservative upwind schemes requiring numerical flux functions.

2 Specification

```
SUBROUTINE D03PUF (ULEFT, URIGHT, GAMMA, FLUX, IFAIL)

INTEGER            IFAIL
REAL (KIND=nag_wp) ULEFT(3), URIGHT(3), GAMMA, FLUX(3)
```

3 Description

D03PUF calculates a numerical flux function at a single spatial point using Roe's Approximate Riemann Solver (see Roe (1981)) for the Euler equations (for a perfect gas) in conservative form. You must supply the *left* and *right* solution values at the point where the numerical flux is required, i.e., the initial left and right states of the Riemann problem defined below.

In the routines D03PFF, D03PLF and D03PSF, the left and right solution values are derived automatically from the solution values at adjacent spatial points and supplied to the subroutine argument NUMFLX from which you may call D03PUF.

The Euler equations for a perfect gas in conservative form are:

$$\frac{\partial U}{\partial t}+\frac{\partial F}{\partial x}=0, \tag{1}$$

with

$$U=\begin{bmatrix}\rho\\ m\\ e\end{bmatrix} \quad \text{and} \quad F=\begin{bmatrix}m\\ \frac{m^2}{\rho}+(\gamma-1)\left(e-\frac{m^2}{2\rho}\right)\\ \frac{me}{\rho}+\frac{m}{\rho}(\gamma-1)\left(e-\frac{m^2}{2\rho}\right)\end{bmatrix}, \tag{2}$$

where ρ is the density, m is the momentum, e is the specific total energy, and γ is the (constant) ratio of specific heats. The pressure p is given by

$$p=(\gamma-1)\left(e-\frac{\rho u^2}{2}\right), \tag{3}$$

where $u=m/\rho$ is the velocity.

The routine calculates the Roe approximation to the numerical flux function $F(U_L,U_R)=F(U^*(U_L,U_R))$, where $U=U_L$ and $U=U_R$ are the left and right solution values, and $U^*(U_L,U_R)$ is the intermediate state $\omega(0)$ arising from the similarity solution $U(y,t)=\omega(y/t)$ of the Riemann problem defined by

$$\frac{\partial U}{\partial t}+\frac{\partial F}{\partial y}=0, \tag{4}$$

with U and F as in (2), and initial piecewise constant values $U=U_L$ for $y<0$ and $U=U_R$ for $y>0$. The spatial domain is $-\infty<y<\infty$, where $y=0$ is the point at which the numerical flux is required.

This implementation of Roe's scheme for the Euler equations uses the so-called parameter-vector method described in Roe (1981).

4 References

LeVeque R J (1990) *Numerical Methods for Conservation Laws* Birkhäuser Verlag

Quirk J J (1994) A contribution to the great Riemann solver debate *Internat. J. Numer. Methods Fluids* **18** 555–574

Roe P L (1981) Approximate Riemann solvers, parameter vectors, and difference schemes *J. Comput. Phys.* **43** 357–372

5 Parameters

1: ULEFT(3) – REAL (KIND=nag_wp) array *Input*

On entry: ULEFT(i) must contain the left value of the component U_i, for $i = 1, 2, 3$. That is, ULEFT(1) must contain the left value of ρ, ULEFT(2) must contain the left value of m and ULEFT(3) must contain the left value of e.

Constraints:

ULEFT(1) ≥ 0.0;
Left pressure, $pl \geq 0.0$, where pl is calculated using (3).

2: URIGHT(3) – REAL (KIND=nag_wp) array *Input*

On entry: URIGHT(i) must contain the right value of the component U_i, for $i = 1, 2, 3$. That is, URIGHT(1) must contain the right value of ρ, URIGHT(2) must contain the right value of m and URIGHT(3) must contain the right value of e.

Constraints:

URIGHT(1) ≥ 0.0;
Right pressure, $pr \geq 0.0$, where pr is calculated using (3).

3: GAMMA – REAL (KIND=nag_wp) *Input*

On entry: the ratio of specific heats, γ.

Constraint: GAMMA > 0.0.

4: FLUX(3) – REAL (KIND=nag_wp) array *Output*

On exit: FLUX(i) contains the numerical flux component $\hat{F}_i$, for $i = 1, 2, 3$.

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

Note: if the left and/or right values of ρ or p (from (3)) are found to be negative, then the routine will terminate with an error exit (IFAIL $= 2$). If the routine is being called from the NUMFLX etc., then a **soft fail** option (IFAIL $= 1$ or -1) is recommended so that a recalculation of the current time step can be forced using the NUMFLX parameter IRES (see D03PFF or D03PLF).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, GAMMA ≤ 0.0.

IFAIL $= 2$

On entry, the left and/or right density or pressure value is less than 0.0.

7 Accuracy

D03PUF performs an exact calculation of the Roe numerical flux function, and so the result will be accurate to ***machine precision***.

8 Further Comments

D03PUF must only be used to calculate the numerical flux for the Euler equations in exactly the form given by (2), with ULEFT(i) and URIGHT(i) containing the left and right values of ρ, m and e, for $i = 1, 2, 3$, respectively. It should be noted that Roe's scheme, in common with all Riemann solvers, may be unsuitable for some problems (see Quirk (1994) for examples). In particular Roe's scheme does not satisfy an 'entropy condition' which guarantees that the approximate solution of the PDE converges to the correct physical solution, and hence it may admit non-physical solutions such as expansion shocks. The algorithm used in this routine does not detect or correct any entropy violation. The time taken is independent of the input parameters.

9 Example

See Section 9 in D03PLF.