NAG Library Routine Document

D02LZF

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

D02LZF interpolates components of the solution of a non-stiff system of second-order differential equations from information provided by the integrator D02LAF, when the low-order method has been used.

2 Specification

```
SUBROUTINE D02LZF (NEQ, T, Y, YP, NWANT, TWANT, YWANT, YPWANT, RWORK,      &
                   LRWORK, IFAIL)
INTEGER            NEQ, NWANT, LRWORK, IFAIL
REAL (KIND=nag_wp) T, Y(NEQ), YP(NEQ), TWANT, YWANT(NWANT),               &
                   YPWANT(NWANT), RWORK(LRWORK)
```

3 Description

D02LZF 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 (1986)) and information generated by D02LAF when the low-order method has been used. This information must be presented unchanged to D02LZF. D02LZF should not normally be used to extrapolate outside the range of the values from D02LAF.

4 References

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

5 Arguments

1: NEQ – INTEGER *Input*

On entry: the number of second-order ordinary differential equations being solved by D02LAF. It must contain the same value as the argument NEQ in a prior call to D02LAF.

2: T – REAL (KIND=nag_wp) *Input*

On entry: t, the current value at which the solution and its derivative have been computed (as returned in argument T on output from D02LAF).

3: Y(NEQ) – REAL (KIND=nag_wp) array *Input*

On entry: the ith component of the solution at t, for $i = 1, 2, \ldots, \text{NEQ}$, as returned from D02LAF.

4: YP(NEQ) – REAL (KIND=nag_wp) array *Input*

On entry: the ith component of the derivative at t, for $i = 1, 2, \ldots, \text{NEQ}$, as returned from D02LAF.

5: NWANT – INTEGER *Input*

On entry: the number of components of the solution and derivative whose values at TWANT are required. The first NWANT components are evaluated.

Constraint: $1 \leq \text{NWANT} \leq \text{NEQ}$.

6: TWANT – REAL (KIND=nag_wp) *Input*

On entry: 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

$$told \leq \text{TWANT} \leq \text{T},$$

or if integration is proceeding in the negative direction

$$told \geq \text{TWANT} \geq \text{T},$$

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 D02LYF.) Extrapolation is permitted but not recommended, and IFAIL $= 2$ is returned whenever extrapolation is attempted.

7: YWANT(NWANT) – REAL (KIND=nag_wp) array *Output*

On exit: the calculated value of the ith component of the solution at $t = \text{TWANT}$, for $i = 1, 2, \ldots, \text{NWANT}$.

8: YPWANT(NWANT) – REAL (KIND=nag_wp) array *Output*

On exit: the calculated value of the ith component of the derivative at $t = \text{TWANT}$, for $i = 1, 2, \ldots, \text{NWANT}$.

9: RWORK(LRWORK) – REAL (KIND=nag_wp) array *Communication Array*

On entry: this **must** be the same argument RWORK as supplied to D02LAF. It is used to pass information from D02LAF to D02LZF and therefore the contents of this array **must not** be changed before calling D02LZF.

10: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D02LZF is called.

This must be the same argument LRWORK as supplied to the setup routine D02LXF.

11: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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).

If D02LZF is to be used for extrapolation at TWANT, IFAIL should be set to 1 before entry. It is then essential to test the value of IFAIL on exit.

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 $= 1$

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

– D02LAF has not been called;

– one or both of the arguments NEQ and LRWORK does not match the corresponding argument supplied to the setup routine D02LXF;

– no integration steps have been taken since the last call to D02LXF with START $=$ .TRUE.;

– NWANT < 1 or NWANT $>$ NEQ.

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

IFAIL $= 2$

D02LZF 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 $= 3$

D02LAF last used the high order method to integrate the system of differential equations. Interpolation is not permitted with this method.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The error in interpolation is of a similar order to the error arising from the integration using D02LAF with the lower order method.

The same order of accuracy can be expected when extrapolating using D02LZF. However, the actual error in extrapolation will, in general, be much larger than for interpolation.

8 Parallelism and Performance

D02LZF is not thread safe and should not be called from a multithreaded user program. Please see Section 3.12.1 in How to Use the NAG Library and its Documentation for more information on thread safety.

D02LZF is not threaded in any implementation.

9 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.

10 Example

See Section 10 in D02LAF.