NAG Library Routine Document

D02NVF

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

D02NVF is a setup routine which must be called prior to linear algebra setup routines and integrators from the SPRINT suite of routines, if Backward Differentiation Formulae (BDF) are to be used.

2 Specification

```
SUBROUTINE D02NVF (NEQMAX, SDYSAV, MAXORD, METHOD, PETZLD, CON, TCRIT,     &
                   HMIN, HMAX, H0, MAXSTP, MXHNIL, NORM, RWORK, IFAIL)

INTEGER             NEQMAX, SDYSAV, MAXORD, MAXSTP, MXHNIL, IFAIL
REAL (KIND=nag_wp) CON(6), TCRIT, HMIN, HMAX, H0, RWORK(50+4*NEQMAX)
LOGICAL             PETZLD
CHARACTER(1)        METHOD, NORM
```

3 Description

An integrator setup routine must be called before the call to any linear algebra setup routine or integrator from the SPRINT suite of routines in this sub-chapter. This setup routine, D02NVF, makes the choice of the BDF integrator and permits you to define options appropriate to this choice. Alternative choices of integrator from this suite are the BLEND method and the DASSL implementation of the BDF method which can be chosen by initial calls to D02NWF or D02MVF respectively.

4 References

See the D02M–N Sub-chapter Introduction.

5 Arguments

1: NEQMAX – INTEGER *Input*

On entry: a bound on the maximum number of differential equations to be solved.

Constraint: NEQMAX ≥ 1.

2: SDYSAV – INTEGER *Input*

On entry: the second dimension of the array YSAV that will be supplied to the integrator, as declared in the (sub)program from which the integrator is called.

Constraint: SDYSAV $\geq$ MAXORD $+ 1$.

3: MAXORD – INTEGER *Input*

On entry: the maximum order to be used for the BDF method.

Constraint: $0 <$ MAXORD ≤ 5.

4: METHOD – CHARACTER(1) *Input*

On entry: specifies the method to be used to solve the system of nonlinear equations arising on each step of the BDF code.

METHOD = 'N'
A modified Newton iteration is used.

METHOD = 'F'
Functional iteration is used.

METHOD = 'D'
A modified Newton iteration is used.

Note: a linear algebra setup routine must be called even when using functional iteration, since if difficulty is encountered a switch is made to a modified Newton method.

Only the first character of the actual argument METHOD is passed to D02NVF; hence it is permissible for the actual argument to be more descriptive e.g., 'Newton', 'Functional iteration' or 'Default' in a call to D02NVF.

Constraint: METHOD = 'N', 'F' or 'D'.

5: PETZLD – LOGICAL *Input*

On entry: specifies whether the Petzold local error test is to be used. If PETZLD is set to .TRUE. on entry, then the Petzold local error test is used, otherwise a conventional test is used. The Petzold test results in extra overhead cost but is more stable and reliable for differential/algebraic equations.

6: CON(6) – REAL (KIND=nag_wp) array *Input/Output*

On entry: values to be used to control step size choice during integration. If any $\mathrm{CON}(i) = 0.0$ on entry, it is replaced by its default value described below. In most cases this is the recommended setting.

CON(1), CON(2), and CON(3) are factors used to bound step size changes. If the current step size h fails, then the modulus of the next step size is bounded by $\mathrm{CON}(1) \times |h|$. The default value of CON(1) is 2.0. Note that the new step size may be used with a method of different order to the failed step. If the initial step size is h, then the modulus of the step size on the second step is bounded by $\mathrm{CON}(3) \times |h|$. At any other stage in the integration, if the current step size is h, then the modulus of the next step size is bounded by $\mathrm{CON}(2) \times |h|$. The default values are 10.0 for CON(2) and 1000.0 for CON(3).

CON(4), CON(5) and CON(6) are 'tuning' constants used in determining the next order and step size. They are used to scale the error estimates used in determining whether to keep the same order of the BDF method, decrease the order or increase the order respectively. The larger the value of $\mathrm{CON}(i)$, for $i = 4, 5, 6$, the less likely the choice of the corresponding order. The default values are: $\mathrm{CON}(4) = 1.2$, $\mathrm{CON}(5) = 1.3$, $\mathrm{CON}(6) = 1.4$.

Constraints:

These constraints must be satisfied after any zero values have been replaced by their default values.

$0.0 < \mathrm{CON}(1) \leq \mathrm{CON}(2) \leq \mathrm{CON}(3)$;
$\mathrm{CON}(i) \geq 1.0$, for $i = 2, 3, \ldots, 6$.

On exit: the values actually used by D02NVF.

7: TCRIT – REAL (KIND=nag_wp) *Input*

On entry: a point beyond which integration must not be attempted. The use of TCRIT is described under the argument ITASK in the specification for the integrator (e.g., see D02NBF). A value, 0.0 say, must be specified even if ITASK subsequently specifies that TCRIT will not be used.

8: HMIN – REAL (KIND=nag_wp) *Input*

On entry: the minimum absolute step size to be allowed. Set HMIN $= 0.0$ if this option is not required.

9: HMAX – REAL (KIND=nag_wp) *Input*

On entry: the maximum absolute step size to be allowed. Set HMAX $= 0.0$ if this option is not required.

10: H0 – REAL (KIND=nag_wp) *Input*

On entry: the step size to be attempted on the first step. Set H0 $= 0.0$ if the initial step size is calculated internally.

11: MAXSTP – INTEGER *Input*

On entry: the maximum number of steps to be attempted during one call to the integrator after which it will return with IFAIL $= 2$. Set MAXSTP $= 0$ if no limit is to be imposed.

12: MXHNIL – INTEGER *Input*

On entry: the maximum number of warnings printed (if ITRACE ≥ 0) per problem when $t + h = t$ on a step ($h =$ current step size). If MXHNIL ≤ 0, a default value of 10 is assumed.

13: NORM – CHARACTER(1) *Input*

On entry: indicates the type of norm to be used.

NORM = 'M'
Maximum norm.

NORM = 'A'
Averaged L2 norm.

NORM = 'D'
Is the same as NORM = 'A'.

If $vnorm$ denotes the norm of the vector v of length n, then for the averaged L2 norm

$$vnorm = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(v_i/w_i)^2},$$

while for the maximum norm

$$vnorm = \max_i |v_i/w_i|.$$

If you wish to weight the maximum norm or the L2 norm, then RTOL and ATOL should be scaled appropriately on input to the integrator (see under ITOL in the specification of the integrator for the formulation of the weight vector w_i from RTOL and ATOL, e.g., see D02NBF).

Only the first character to the actual argument NORM is passed to D02NVF; hence it is permissible for the actual argument to be more descriptive e.g., 'Maximum', 'Average L2' or 'Default' in a call to D02NVF.

Constraint: NORM = 'M', 'A' or 'D'.

14: RWORK$(50 + 4 \times$ NEQMAX$)$ – REAL (KIND=nag_wp) array *Communication Array*

This must be the same workspace array as the array RWORK supplied to the integrator. It is used to pass information from the setup routine to the integrator and therefore the contents of this array must not be changed before calling the integrator.

15: 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).

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, an illegal input was detected.

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

Not applicable.

8 Parallelism and Performance

D02NVF is not threaded in any implementation.

9 Further Comments

None.

10 Example

See Section 10 in D02NBF.