NAG Library Function Document

nag_ode_ivp_rkts_diag (d02ptc)

1 Purpose

nag_ode_ivp_rkts_diag (d02ptc) provides details about an integration performed by either nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc) or nag_ode_ivp_rk_step_rev comm (d02pgc).

2 Specification

```
#include <nag.h>
#include <nagd02.h>
void nag_ode_ivp_rkts_diag (Integer *fevals, Integer *stepcost,
     double *waste, Integer *stepsok, double *hnext, Integer iwsav[],
     const double rwsav[], NagError *fail)
```

3 Description

nag_ode_ivp_rkts_diag (d02ptc) and its associated functions (nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc), nag_ode_ivp_rk_step_revcomm (d02pgc), nag_ode_ivp_rk_interp_ setup (d02phc), nag_ode_ivp_rk_interp_eval (d02pjc), nag_ode_ivp_rkts_setup (d02pqc), nag_o de_ivp_rkts_reset_tend (d02prc), nag_ode_ivp_rkts_interp (d02psc) and nag_ode_ivp_rkts_errass (d02puc)) solve the initial value problem for a first-order system of ordinary differential equations. The functions, based on Runge–Kutta methods and derived from RKSUITE (see Brankin *et al.* (1991)), integrate

$$y' = f(t, y) \quad \text{given} \quad y(t_0) = y_0$$

where y is the vector of n solution components and t is the independent variable.

After a call to nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc) or nag_o de_ivp_rk_step_revcomm (d02pgc), nag_ode_ivp_rkts_diag (d02ptc) can be called to obtain information about the cost of the integration and the size of the next step.

4 References

Brankin R W, Gladwell I and Shampine L F (1991) RKSUITE: A suite of Runge–Kutta codes for the initial value problems for ODEs *SoftReport 91-S1* Southern Methodist University

5 Arguments

1: **fevals** – Integer * *Output*

On exit: the total number of evaluations of f used in the integration so far; this includes evaluations of f required for the secondary integration necessary if nag_ode_ivp_rkts_setup (d02pqc) had previously been called with **errass** = Nag_ErrorAssess_on.

2: **stepcost** – Integer * *Output*

On exit: the cost in terms of number of evaluations of f of a typical step with the method being used for the integration. The method is specified by the argument **method** in a prior call to nag_ode_ivp_rkts_setup (d02pqc).

3: **waste** – double * *Output*

On exit: the number of attempted steps that failed to meet the local error requirement divided by the total number of steps attempted so far in the integration. A 'large' fraction indicates that the integrator is having trouble with the problem being solved. This can happen when the problem is 'stiff' and also when the solution has discontinuities in a low-order derivative.

4: **stepsok** – Integer * *Output*

On exit: the number of accepted steps.

5: **hnext** – double * *Output*

On exit: the step size the integrator will attempt to use for the next step.

6: **iwsav**[**130**] – Integer *Communication Array*

7: **rwsav**[**350**] – const double *Communication Array*

Note: the communication **rwsav** used by the other functions in the suite must be used here however, only the first 350 elements will be referenced.

On entry: these must be the same arrays supplied in a previous call to nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc) or nag_ode_ivp_rk_step_revcomm (d02pgc). They must remain unchanged between calls.

On exit: information about the integration for use on subsequent calls to nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc) or nag_ode_ivp_rk_step_revcomm (d02pgc) or other associated functions.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in How to Use the NAG Library and its Documentation for further information.

NE_MISSING_CALL

You cannot call this function before you have called the integrator.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in How to Use the NAG Library and its Documentation for further information.

NE_PREV_CALL

On entry, a previous call to the setup function has not been made or the communication arrays have become corrupted, or a catastrophic error has already been detected elsewhere.
You cannot continue integrating the problem.

NE_RK_INVALID_CALL

You have already made one call to this function after the integrator could not achieve specified accuracy.
You cannot call this function again.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_ode_ivp_rkts_diag (d02ptc) is not threaded in any implementation.

9 Further Comments

When a secondary integration has taken place, that is when global error assessment has been specified using **errass** = Nag_ErrorAssess_on in a prior call to nag_ode_ivp_rkts_setup (d02pqc), then the approximate number of evaluations of f used in this secondary integration is given by $2 \times$ **stepsok** $\times$ **stepcost** for **method** = Nag_RK_4_5 or Nag_RK_7_8 and $3 \times$ **stepsok** $\times$ **stepcost** for **method** = Nag_RK_2_3.

10 Example

See Section 10 in nag_ode_ivp_rkts_range (d02pec), nag_ode_ivp_rkts_onestep (d02pfc), nag_ode_ivp_rkts_reset_tend (d02prc), nag_ode_ivp_rkts_interp (d02psc) and nag_ode_ivp_rkts_errass (d02puc).