NAG Library Function Document

nag_ode_bvp_ps_lin_cgl_vals (d02ubc)

1 Purpose

nag_ode_bvp_ps_lin_cgl_vals (d02ubc) evaluates a function, or one of its lower order derivatives, from its Chebyshev series representation at Chebyshev Gauss–Lobatto points on $[a, b]$. The coefficients of the Chebyshev series representation required are usually derived from those returned by nag_ode_bvp_ps_lin_coeffs (d02uac) or nag_ode_bvp_ps_lin_solve (d02uec).

2 Specification

```
#include <nag.h>
#include <nagd02.h>
void nag_ode_bvp_ps_lin_cgl_vals (Integer n, double a, double b, Integer q,
     const double c[], double f[], NagError *fail)
```

3 Description

nag_ode_bvp_ps_lin_cgl_vals (d02ubc) evaluates the Chebyshev series

$$S(\bar{x}) = \tfrac{1}{2}c_1T_0(\bar{x}) + c_2T_1(\bar{x}) + c_3T_2(\bar{x}) + \cdots + c_{n+1}T_n(\bar{x}),$$

or its derivative (up to fourth order) at the Chebyshev Gauss–Lobatto points on $[a, b]$. Here $T_j(\bar{x})$ denotes the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$ defined on $[-1, 1]$. In terms of your original variable, x say, the input values at which the function values are to be provided are

$$x_r = -\tfrac{1}{2}(b-a)\cos(\pi(r-1)/n) + \tfrac{1}{2}(b+a), \quad r = 1, 2, \ldots, n+1,$$

where b and a are respectively the upper and lower ends of the range of x over which the function is required.

The calculation is implemented by a forward one-dimensional discrete Fast Fourier Transform (DFT).

4 References

Canuto C (1988) *Spectral Methods in Fluid Dynamics* 502 Springer

Canuto C, Hussaini M Y, Quarteroni A and Zang T A (2006) *Spectral Methods: Fundamentals in Single Domains* Springer

Trefethen L N (2000) *Spectral Methods in MATLAB* SIAM

5 Arguments

1: **n** – Integer *Input*

On entry: n, where the number of grid points is $n + 1$. This is also the largest order of Chebyshev polynomial in the Chebyshev series to be computed.

Constraint: **n** > 0 and **n** is even.

2: **a** – double *Input*

On entry: a, the lower bound of domain $[a, b]$.

Constraint: **a** $<$ **b**.

3: **b** – double *Input*

On entry: b, the upper bound of domain $[a, b]$.

Constraint: $\mathbf{b} > \mathbf{a}$.

4: **q** – Integer *Input*

On entry: the order, q, of the derivative to evaluate.

Constraint: $0 \leq \mathbf{q} \leq 4$.

5: **c**[**n** + **1**] – const double *Input*

On entry: the Chebyshev coefficients, c_i, for $i = 1, 2, \ldots, n + 1$.

6: **f**[**n** + **1**] – double *Output*

On exit: the derivatives $S^{(q)}x_i$, for $i = 1, 2, \ldots, n + 1$, of the Chebyshev series, S.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_BAD_PARAM

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

NE_INT

On entry, $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{n} > 0$.

On entry, $\mathbf{n} = \langle value \rangle$.
Constraint: **n** is even.

On entry, $\mathbf{q} = \langle value \rangle$.
Constraint: $0 \leq \mathbf{q} \leq 4$.

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.

NE_REAL_2

On entry, $\mathbf{a} = \langle value \rangle$ and $\mathbf{b} = \langle value \rangle$.
Constraint: $\mathbf{a} < \mathbf{b}$.

7 Accuracy

Evaluations of DFT to obtain function or derivative values should be an order n multiple of ***machine precision*** assuming full accuracy to ***machine precision*** in the given Chebyshev series representation.

8 Parallelism and Performance

nag_ode_bvp_ps_lin_cgl_vals (d02ubc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_ode_bvp_ps_lin_cgl_vals (d02ubc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The number of operations is of the order $n\log(n)$ and the memory requirements are $O(n)$; thus the computation remains efficient and practical for very fine discretizations (very large values of n).

10 Example

See Section 10 in nag_ode_bvp_ps_lin_solve (d02uec).