NAG Library Function Document

nag_inteq_fredholm2_split (d05aac)

1 Purpose

nag_inteq_fredholm2_split (d05aac) solves a linear, nonsingular Fredholm equation of the second kind with a split kernel.

2 Specification

```
#include <nag.h>
#include <nagd05.h>
```

```
void nag_inteq_fredholm2_split (double lambda, double a, double b, Integer n,
     double (*k1)(double x, double s, Nag_Comm *comm),
     double (*k2)(double x, double s, Nag_Comm *comm),
     double (*g)(double x, Nag_Comm *comm),
     Nag_KernelForm kform, double f[], double c[], Nag_Comm *comm,
     NagError *fail)
```

3 Description

nag_inteq_fredholm2_split (d05aac) solves an integral equation of the form

$$f(x) - \lambda \int_a^b k(x,s) f(s)\, ds = g(x)$$

for $a \leq x \leq b$, when the kernel k is defined in two parts: $k = k_1$ for $a \leq s \leq x$ and $k = k_2$ for $x < s \leq b$. The method used is that of El–Gendi (1969) for which, it is important to note, each of the functions k_1 and k_2 must be defined, smooth and nonsingular, for all x and s in the interval $[a, b]$.

An approximation to the solution $f(x)$ is found in the form of an n term Chebyshev series $\sum_{i=1}^{n}{}' c_i T_i(x)$, where $'$ indicates that the first term is halved in the sum. The coefficients c_i, for $i = 1, 2, \ldots, n$, of this series are determined directly from approximate values f_i, for $i = 1, 2, \ldots, n$, of the function $f(x)$ at the first n of a set of $m + 1$ Chebyshev points:

$$x_i = \tfrac{1}{2}(a + b + (b - a)\cos[(i - 1)\pi/m]), \quad i = 1, 2, \ldots, m + 1.$$

The values f_i are obtained by solving simultaneous linear algebraic equations formed by applying a quadrature formula (equivalent to the scheme of Clenshaw and Curtis (1960)) to the integral equation at the above points.

In general $m = n - 1$. However, if the kernel k is centro-symmetric in the interval $[a, b]$, i.e., if $k(x, s) = k(a + b - x, a + b - s)$, then the function is designed to take advantage of this fact in the formation and solution of the algebraic equations. In this case, symmetry in the function $g(x)$ implies symmetry in the function $f(x)$. In particular, if $g(x)$ is even about the mid-point of the range of integration, then so also is $f(x)$, which may be approximated by an even Chebyshev series with $m = 2n - 1$. Similarly, if $g(x)$ is odd about the mid-point then $f(x)$ may be approximated by an odd series with $m = 2n$.

4 References

Clenshaw C W and Curtis A R (1960) A method for numerical integration on an automatic computer *Numer. Math.* **2** 197–205

El–Gendi S E (1969) Chebyshev solution of differential, integral and integro-differential equations *Comput. J.* **12** 282–287

5 Arguments

1: **lambda** – double *Input*

On entry: the value of the parameter λ of the integral equation.

2: **a** – double *Input*

On entry: a, the lower limit of integration.

3: **b** – double *Input*

On entry: b, the upper limit of integration.

Constraint: **b** $>$ **a**.

4: **n** – Integer *Input*

On entry: the number of terms in the Chebyshev series required to approximate $f(x)$.

Constraint: **n** ≥ 1.

5: **k1** – function, supplied by the user *External Function*

k1 must evaluate the kernel $k(x,s) = k_1(x,s)$ of the integral equation for $a \leq s \leq x$.

The specification of **k1** is:

```
double k1 (double x, double s, Nag_Comm *comm)
```

1: **x** – double *Input*

2: **s** – double *Input*

On entry: the values of x and s at which $k_1(x,s)$ is to be evaluated.

3: **comm** – Nag_Comm *

Pointer to structure of type Nag_Comm; the following members are relevant to **k1**.

user – double *
iuser – Integer *
p – Pointer

The type Pointer will be `void *`. Before calling nag_inteq_fredholm2_split (d05aac) you may allocate memory and initialize these pointers with various quantities for use by **k1** when called from nag_inteq_fredholm2_split (d05aac) (see Section 3.2.1.1 in the Essential Introduction).

6: **k2** – function, supplied by the user *External Function*

k2 must evaluate the kernel $k(x,s) = k_2(x,s)$ of the integral equation for $x < s \leq b$.

The specification of **k2** is:

```
double k2 (double x, double s, Nag_Comm *comm)
```

1: **x** – double *Input*

2: **s** – double *Input*

On entry: the values of x and s at which $k_2(x,s)$ is to be evaluated.

3: **comm** – Nag_Comm *

Pointer to structure of type Nag_Comm; the following members are relevant to **k2**.

user – double *
iuser – Integer *
p – Pointer

The type Pointer will be `void *`. Before calling nag_inteq_fredholm2_split (d05aac) you may allocate memory and initialize these pointers with various quantities for use by **k2** when called from nag_inteq_fredholm2_split (d05aac) (see Section 3.2.1.1 in the Essential Introduction).

Note that the functions k_1 and k_2 must be defined, smooth and nonsingular for all x and s in the interval $[a,b]$.

7: **g** – function, supplied by the user *External Function*

g must evaluate the function $g(x)$ for $a \leq x \leq b$.

The specification of **g** is:

```
double g (double x, Nag_Comm *comm)
```

1: **x** – double *Input*

On entry: the values of x at which $g(x)$ is to be evaluated.

2: **comm** – Nag_Comm *

Pointer to structure of type Nag_Comm; the following members are relevant to **g**.

user – double *
iuser – Integer *
p – Pointer

The type Pointer will be `void *`. Before calling nag_inteq_fredholm2_split (d05aac) you may allocate memory and initialize these pointers with various quantities for use by **g** when called from nag_inteq_fredholm2_split (d05aac) (see Section 3.2.1.1 in the Essential Introduction).

8: **kform** – Nag_KernelForm *Input*

On entry: determines the forms of the kernel, $k(x,s)$, and the function $g(x)$.

kform = Nag_NoCentroSymm
$k(x,s)$ is not centro-symmetric (or no account is to be taken of centro-symmetry).

kform = Nag_CentroSymmOdd
$k(x,s)$ is centro-symmetric and $g(x)$ is odd.

kform = Nag_CentroSymmEven
$k(x,s)$ is centro-symmetric and $g(x)$ is even.

kform = Nag_CentroSymmNeither
$k(x,s)$ is centro-symmetric but $g(x)$ is neither odd nor even.

Constraint: **kform** = Nag_NoCentroSymm, Nag_CentroSymmOdd, Nag_CentroSymmEven or Nag_CentroSymmNeither.

9: **f**[**n**] – double *Output*

On exit: the approximate values f_i, for $i = 1, 2, \ldots, \mathbf{n}$, of $f(x)$ evaluated at the first **n** of $m + 1$ Chebyshev points x_i, (see Section 3).

If **kform** = Nag_NoCentroSymm or Nag_CentroSymmNeither, $m = \mathbf{n} - 1$.

If **kform** = Nag_CentroSymmOdd, $m = 2 \times \mathbf{n}$.

If **kform** = Nag_CentroSymmEven, $m = 2 \times \mathbf{n} - 1$.

10: **c**[**n**] – double *Output*

On exit: the coefficients c_i, for $i = 1, 2, \ldots, \mathbf{n}$, of the Chebyshev series approximation to $f(x)$.

If **kform** = Nag_CentroSymmOdd this series contains polynomials of odd order only and if **kform** = Nag_CentroSymmEven the series contains even order polynomials only.

11: **comm** – Nag_Comm *

The NAG communication argument (see Section 3.2.1.1 in the Essential Introduction).

12: **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.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

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

NE_EIGENVALUES

A failure has occurred due to proximity of an eigenvalue.

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: $\mathbf{n} \geq 1$.

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 the Essential Introduction for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

NE_REAL_2

On entry, **a** = ⟨*value*⟩ and **b** = ⟨*value*⟩.
Constraint: $\mathbf{b} > \mathbf{a}$.

7 Accuracy

No explicit error estimate is provided by the function but it is usually possible to obtain a good indication of the accuracy of the solution either

(i) by examining the size of the later Chebyshev coefficients c_i, or

(ii) by comparing the coefficients c_i or the function values f_i for two or more values of **n**.

8 Parallelism and Performance

nag_inteq_fredholm2_split (d05aac) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_inteq_fredholm2_split (d05aac) 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 X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken by nag_inteq_fredholm2_split (d05aac) increases with **n**.

This function may be used to solve an equation with a continuous kernel by defining **k1** and **k2** to be identical.

This function may also be used to solve a Volterra equation by defining **k2** (or **k1**) to be identically zero.

10 Example

This example solves the equation

$$f(x) - \int_0^1 k(x,s) f(s)\, ds = \left(1 - \frac{1}{\pi^2}\right) \sin(\pi x)$$

where

$$k(x,s) = \begin{cases} s(1-x) & \text{for } 0 \leq s \leq x, \\ x(1-s) & \text{for } x < s \leq 1. \end{cases}$$

Five terms of the Chebyshev series are sought, taking advantage of the centro-symmetry of the $k(x,s)$ and even nature of $g(x)$ about the mid-point of the range $[0,1]$.

The approximate solution at the point $x = 0.1$ is calculated by calling nag_sum_cheby_series (c06dcc).

10.1 Program Text

```
/* nag_inteq_fredholm2_split (d05aac) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 23, 2011.
 */
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagc06.h>
#include <nagd05.h>
#include <nagx01.h>

#ifdef __cplusplus
extern "C" {
#endif
```

```
static double NAG_CALL k1(double x, double s, Nag_Comm *comm);
static double NAG_CALL k2(double x, double s, Nag_Comm *comm);
static double NAG_CALL g(double x, Nag_Comm *comm);

#ifdef __cplusplus
}
#endif

int main(void)
{
  /* Scalars */
  double  a = 0.0, b = 1.0, lambda = 1.0, x = 0.1;
  double  res;
  Integer exit_status = 0;
  Integer n = 5;
  Integer i;
  /* Arrays */
  static double ruser[3] = {-1.0, -1.0, -1.0};
  double  *c = 0, *f = 0;
  /* NAG types */
  Nag_Comm comm;
  NagError fail;
  Nag_KernelForm kform = Nag_CentroSymmEven;
  Nag_Series s = Nag_SeriesEven;

  INIT_FAIL(fail);

  printf("nag_inteq_fredholm2_split (d05aac) Example Program Results\n");

  /* For communication with user-supplied functions: */
  comm.user = ruser;

  if (
      !(f = NAG_ALLOC(n, double)) ||
      !(c = NAG_ALLOC(n, double))
      )
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  /*
    nag_inteq_fredholm2_split (d05aac).
    Linear non-singular Fredholm integral equation, second kind, split kernel.
  */
  nag_inteq_fredholm2_split(lambda, a, b, n, k1, k2, g, kform, f, c, &comm,
                            &fail);

  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_inteq_fredholm2_split (d05aac).\n%s\n",
             fail.message);
      exit_status = 1;
      goto END;
    }

  printf("\nKernel is centro-symmetric and g is even, "
         "so the solution is even\n\n");

  printf("Chebyshev coefficients of the approximation to f(x)\n\n");
  for (i = 0; i < n; i++) printf("%14.4e", c[i]);
  printf("\n\n");

  /*
    nag_sum_cheby_series (c06dcc).
    Sum of a Chebyshev series at a set of points.
  */
  nag_sum_cheby_series(&x, 1, a, b, c, n, s, &res, &fail);
```

```
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_sum_cheby_series (c06dcc).\n%s\n",
             fail.message);
      exit_status = 1;
      goto END;
    }

  printf(" Solution: x = %5.2f and  f(x) = %10.4f\n", x, res);

 END:

  NAG_FREE(c);
  NAG_FREE(f);

  return exit_status;
}
static double NAG_CALL k1(double x, double s, Nag_Comm *comm)
{
  if (comm->user[0] == -1.0)
    {
      printf("(User-supplied callback k1, first invocation.)\n");
      comm->user[0] = 0.0;
    }
  return s * (1.0 - x);
}
static double NAG_CALL k2(double x, double s, Nag_Comm *comm)
{
  if (comm->user[1] == -1.0)
    {
      printf("(User-supplied callback k2, first invocation.)\n");
      comm->user[1] = 0.0;
    }
  return x * (1.0 - s);
}
static double NAG_CALL g(double x, Nag_Comm *comm)
{
  if (comm->user[2] == -1.0)
    {
      printf("(User-supplied callback g, first invocation.)\n");
      comm->user[2] = 0.0;
    }
  return (1.0 - 1.0 / pow(nag_pi, 2)) * sin(nag_pi * x);
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_inteq_fredholm2_split (d05aac) Example Program Results
(User-supplied callback g, first invocation.)
(User-supplied callback k1, first invocation.)
(User-supplied callback k2, first invocation.)

Kernel is centro-symmetric and g is even, so the solution is even

Chebyshev coefficients of the approximation to f(x)

    9.4400e-01   -4.9940e-01    2.7992e-02   -5.9669e-04    6.6578e-06

 Solution: x =  0.10 and  f(x) =     0.3090
```