NAG Toolbox

nag_ode_bvp_coll_sys (d02jb)

1 Purpose

nag_ode_bvp_coll_sys (d02jb) solves a regular linear two-point boundary value problem for a system of ordinary differential equations by Chebyshev series using collocation and least squares.

2 Syntax

```
[c, ifail] = nag_ode_bvp_coll_sys(n, cf, bc, x0, x1, k1, kp)
[c, ifail] = d02jb(n, cf, bc, x0, x1, k1, kp)
```

3 Description

nag_ode_bvp_coll_sys (d02jb) calculates the solution of a regular two-point boundary value problem for a regular linear nth-order system of first-order ordinary differential equations as a Chebyshev series in the interval (x_0, x_1). The differential equation

$$y' = A(x)y + r(x)$$

is defined by **cf**, and the boundary conditions at the points x_0 and x_1 are defined by **bc**.

You specify the degree of Chebyshev series required, **k1** $- 1$, and the number of collocation points, **kp**. The function sets up a system of linear equations for the Chebyshev coefficients, n equations for each collocation point and one for each boundary condition. The boundary conditions are solved exactly, and the remaining equations are then solved by a least squares method. The result produced is a set of coefficients for a Chebyshev series solution for each component of the solution of the system of differential equations on an interval normalized to $(-1, 1)$.

nag_fit_1dcheb_eval2 (e02ak) can be used to evaluate the components of the solution at any point on the interval (x_0, x_1) – see Section 10 for an example. nag_fit_1dcheb_deriv (e02ah) followed by nag_fit_1dcheb_eval2 (e02ak) can be used to evaluate their derivatives.

4 References

Picken S M (1970) Algorithms for the solution of differential equations in Chebyshev-series by the selected points method *Report Math. 94* National Physical Laboratory

5 Parameters

5.1 Compulsory Input Parameters

1: **n** – INTEGER

n, the order of the system of differential equations.

Constraint: **n** ≥ 1.

2: **cf** – REAL (KIND=nag_wp) FUNCTION, supplied by the user.

cf defines the system of differential equations (see Section 3). It must return the value of a coefficient function $a_{i,j}(x)$, of A, at a given point x, or of a right-hand side function $r_i(x)$ if **j** $= 0$.

```
[result] = cf(ii, j, x)
```

Input Parameters

1: **ii** – INTEGER

2: **j** – INTEGER

Indicate the function to be evaluated, namely $a_{i,j}(x)$ if $1 \leq \mathbf{j} \leq n$, or $r_i(x)$ if $\mathbf{j} = 0$. $1 \leq \mathbf{ii} \leq n$, $0 \leq \mathbf{j} \leq n$.

3: **x** – REAL (KIND=nag_wp)

The point at which the function is to be evaluated.

Output Parameters

1: **result**

The value of a coefficient function $a_{i,j}(x)$, of A, at a given point x, or of a right-hand side function $r_i(x)$ if $\mathbf{j} = 0$.

3: **bc** – SUBROUTINE, supplied by the user.

bc defines the n boundary conditions, which have the form $y_k(x_0) = s$ or $y_k(x_1) = s$. The boundary conditions may be specified in any order.

```
[j, rhs] = bc(ii)
```

Input Parameters

1: **ii** – INTEGER

The index of the boundary condition to be defined.

Output Parameters

1: **j** – INTEGER

Must be set to $-k$ if the ith boundary condition is $y_k(x_0) = s$, or to $+k$ if it is $y_k(x_1) = s$.

j must not be set to the same value k for two different values of **ii**.

2: **rhs** – REAL (KIND=nag_wp)

The value s.

4: **x0** – REAL (KIND=nag_wp)

5: **x1** – REAL (KIND=nag_wp)

The left- and right-hand boundaries, x_0 and x_1, respectively.

Constraint: $\mathbf{x1} > \mathbf{x0}$.

6: **k1** – INTEGER

The number of coefficients to be returned in the Chebyshev series representation of the components of the solution (hence the degree of the polynomial approximation is $\mathbf{k1} - 1$).

Constraint: $\mathbf{k1} \geq 2$.

7: **kp** – INTEGER

The number of collocation points to be used.

Constraint: **kp** $\geq$ **k1** $-$ 1.

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **c**$(ldc, \mathbf{n})$ – REAL (KIND=nag_wp) array

The computed Chebyshev coefficients of the kth component of the solution, y_k; that is, the computed solution is:

$$y_k = \sum_{i=1}^{\mathbf{k1}} \mathbf{c}(i,k) T_{i-1}(x), \quad 1 \leq k \leq n$$

where $T_i(x)$ is the ith Chebyshev polynomial of the first kind, and $\sum$ denotes that the first coefficient, $\mathbf{c}(1,k)$, is halved.

2: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **n** $<$ 1,
or **x0** $\geq$ **x1**,
or **k1** $<$ 2,
or **kp** $<$ **k1** $-$ 1,
or $ldc <$ **k1**.

ifail $= 2$

On entry, $lw < 2 \times \mathbf{n} \times (\mathbf{kp}+1) \times (\mathbf{n} \times \mathbf{k1}+1) + 7 \times \mathbf{n} \times \mathbf{k1}$,
or $liw < \mathbf{n} \times (\mathbf{k1}+2)$ (i.e., insufficient workspace).

ifail $= 3$

Either the boundary conditions are not linearly independent (that is, in **bc** the variable **j** is set to the same value k for two different values of **ii**), or the rank of the matrix of equations for the coefficients is less than the number of unknowns. Increasing **kp** may overcome this latter problem.

ifail $= 4$

The least squares function nag_linsys_real_gen_lsqsol (f04am) has failed to correct the first approximate solution (see nag_linsys_real_gen_lsqsol (f04am)).

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The Chebyshev coefficients are determined by a stable numerical method. The accuracy of the approximate solution may be checked by varying the degree of the polynomials and the number of collocation points (see Section 9).

8 Further Comments

The time taken by nag_ode_bvp_coll_sys (d02jb) depends on the size and complexity of the differential system, the degree of the polynomial solution, and the number of matching points.

The collocation points in the interval (x_0, x_1) are chosen to be the extrema of the appropriate shifted Chebyshev polynomial. If $\mathbf{kp} = \mathbf{k1} - 1$, then the least squares solution reduces to the solution of a system of linear equations, and true collocation results.

The accuracy of the solution may be checked by repeating the calculation with different values of **k1** and with **kp** fixed but $\mathbf{kp} \gg \mathbf{k1} - 1$. If the Chebyshev coefficients decrease rapidly for each component (and consistently for various **k1** and **kp**), the size of the last two or three gives an indication of the error. If the Chebyshev coefficients do not decay rapidly, it is likely that the solution cannot be well-represented by Chebyshev series. Note that the Chebyshev coefficients are calculated for the interval $(-1, 1)$.

Linear systems of high-order equations in their original form, singular problems, and, indirectly, nonlinear problems can be solved using nag_ode_bvp_coll_nth_comp (d02tg).

9 Example

This example solves the equation

$$y'' + y = 1$$

with boundary conditions

$$y(-1) = y(1) = 0.$$

The equation is written as the first-order system

$$\begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

for solution by nag_ode_bvp_coll_sys (d02jb) and the boundary conditions are written

$$y_1(-1) = y_1(1) = 0.$$

We use $\mathbf{k1} = 4$, 6 and 8, and $\mathbf{kp} = 10$ and 15, so that the different Chebyshev series may be compared. The solution for $\mathbf{k1} = 8$ and $\mathbf{kp} = 15$ is evaluated by nag_fit_1dcheb_eval2 (e02ak) at nine equally spaced points over the interval $(-1, 1)$.

9.1 Program Text

```
function d02jb_example

fprintf('d02jb example results\n\n');

% Set up initial values.
n     = nag_int(2);
k1max = 8;
kpmax = 15;
x0    = -1.0;
x1    = 1.0;
c     = zeros(k1max, n);
```

```
fprintf('  KP   K1   Chebyshev coefficients\n');
for kp = nag_int(10:5:kpmax)
  for k1 = nag_int(4:2:k1max)
    [c, ifail] = d02jb(...
                       n, @cf, @bc, x0, x1, k1, kp);
    % Output results.
    fprintf('%4d ',kp, k1);
    for jord = 1:n
      for kind = 1:k1
        fprintf('%7.4f ',c(kind,jord));
        if mod(kind, 8) == 0 && kind ~= k1
          fprintf('\n          ');
        end
      end
      fprintf('\n          ');
    end
    fprintf('\n');
  end
end
% Now prepare to evaluate and plot the last solution.
fprintf('\n');
k1   = 8;
k1m1 = nag_int(k1-1);
m    = 9;
ia1  = nag_int(1);
x    = zeros(m, 1);
y    = zeros(m, n+1);

fprintf(['Last computed solution evaluated at %1d equally spaced ', ...
    'points\n\n'], m);
fprintf('    X       Y(1)      Y(2)\n');
for i = 1:m
  x(i) = (x0*(m-i) + x1*(i-1))/(m-1);
  fprintf('%8.4f  ', x(i));

  for jord = 1:n
    % Calling e02ak to evaluate the polynomial from its Chebyshev
    % representation.
    [y(i, jord), ifail] = e02ak(...
                                k1m1, x0, x1, c(:,jord), ia1, x(i));
    fprintf('%8.4f  ',y(i, jord));
  end
  fprintf('\n');

  % Calculate the absolute error at this point.
  y(i,n+1) = abs(y(i,1) - 1 + cos(x(i))/cos(1));
end
% Plot results.
fig1 = figure;
display_plot(x, y, fig1);

function [j, rhs] = bc(i)
  % Define the boundary conditions.
  rhs = 0;
  if (i == 1)
    j = nag_int(1);
  else
    j = nag_int(-1);
  end

function result = cf(ii, jj, x)
  % Define the differential equation to be solved.
  if (jj == ii)
    result = 0;
  elseif (ii == 1 && jj == 2)
    result = 1;
  elseif (ii == 2 && jj == 1)
    result = -1;
  elseif (ii == 1)
    result = 0;
```

```
  else
    result = 1;
  end

function display_plot(x, y, fig1)
  axes1 = axes('Parent',fig1,'YMinorTick','on','YTick',[-2 -1 0 1 2],...
              'YColor',[0 0.447 0.741],...
              'XMinorTick','on');
  hold(axes1,'on');
  ylabel('y and y''');
  xlabel('x');
  title({'Two-point BVP for ODE System using Chebyshev series',...
         ' with Collocation & Least-Squares'});

  % Create multiple lines using matrix input to plot
  plot1 = plot(x,y(:,1:2),'Parent',axes1);
  set(plot1(1),'DisplayName','y''','Marker','+','LineStyle','--');
  set(plot1(2),'DisplayName','y','Marker','x');

  % Create axes
  axes2 = axes('Parent',fig1,'HitTest','off','Color','none',...
              'YMinorTick','on',...
              'YTick',[0 5e-07 1e-06 1.5e-06 2e-06],...
              'YColor',[0.85 0.325 0.098],...
              'YAxisLocation','right',...
              'Position',[0.13 0.11 0.775 0.815]);
  hold(axes2,'on');
  ylabel('Absolute Error');

  plot(x,y(:,3),'Parent',axes2,'DisplayName','y error','Marker','*',...
       'LineStyle',':','Color',[0.85 0.325 0.098]);

  legend1 = legend(axes1,'show');
  set(legend1,'Position',[0.17 0.77 0.09 0.11]);
```

9.2 Program Results

```
     d02jb example results

  KP   K1   Chebyshev coefficients
  10    4 -0.7798  0.0000  0.3899 -0.0000
             0.0000  1.5751  0.0000 -0.0629

  10    6 -0.8326 -0.0000  0.4253  0.0000 -0.0090 -0.0000
            -0.0000  1.6290  0.0000 -0.0724 -0.0000  0.0009

  10    8 -0.8325 -0.0000  0.4253  0.0000 -0.0092 -0.0000  0.0001  0.0000
            -0.0000  1.6289  0.0000 -0.0724 -0.0000  0.0009  0.0000 -0.0000

  15    4 -0.7829  0.0000  0.3914 -0.0000
             0.0000  1.5778  0.0000 -0.0631

  15    6 -0.8326 -0.0000  0.4253  0.0000 -0.0090  0.0000
             0.0000  1.6290  0.0000 -0.0724 -0.0000  0.0009

  15    8 -0.8325 -0.0000  0.4253  0.0000 -0.0092  0.0000  0.0001 -0.0000
             0.0000  1.6289  0.0000 -0.0724 -0.0000  0.0009  0.0000 -0.0000


Last computed solution evaluated at 9 equally spaced points

     X       Y(1)      Y(2)
 -1.0000    0.0000   -1.5574
 -0.7500   -0.3542   -1.2616
 -0.5000   -0.6242   -0.8873
 -0.2500   -0.7933   -0.4579
```

```
 0.0000   -0.8508    0.0000
 0.2500   -0.7933    0.4579
 0.5000   -0.6242    0.8873
 0.7500   -0.3542    1.2616
 1.0000    0.0000    1.5574
```