MA50174 Advanced Numerical Methods

Assignment 4 Two point boundary value problems

- **Set:** Monday 5th December
- **To be handed in:** Friday 16th December, by 12.00 to Maths Department Reception Desk. You need to sign your work personally together with a signed coursework cover sheet.
- **Handing in your work.** Please write your MATLAB codes to a CD and hand them in together with the written part of your work. There will be no marks for typesetting your work (using, for example LaTeX) but it is your responsibility to make sure your work is well-presented and readable.

All work which you hand in should be your own. Cheating is a serious offence and will be dealt with under the University disciplinary procedures.

This assignment should typically take about 15 hours. If you spend significantly more than this on an assignment then you may disadvantage yourself with regard to other courses.

You have a limit of 10 pages (excluding Matlab codes).

Two point boundary value problems differ from initial value problems in that the values of a differential equation are specified at *boundary points.* Thus unlike initial value problems where you have complete information at one point and you can simply march forward in time, you have to solve a global problem "linking" data at boundary points. This is a rather harder task, especially if the problem has boundary layers and/or is nonlinear, and the solution does not always exist and/or is unique. Two point boundary value problems can be solved by shooting, or by solving a larger system of global equations based on a variety of discretizations: finite difference, finite element, and spectral collocation methods. In this sheet we look at shooting and finite difference based global solution methods.

Shooting Methods

1. The *shooting method* solves a two point boundary value problem by solving an associated initial value problem. Consider the two point boundary value problem for $u(t)$:

$$\frac{d^4u}{dt^4} - (1+t^2)\left(\frac{d^2u}{dt^2}\right)^2 + 5u^2 = 0, \tag{1}$$

with boundary conditions

$$u(0) = 1, \quad u'(0) = 0, \quad u''(1) = -2 \quad \text{and} \quad u'''(1) = -3. \tag{2}$$

To solve this equation by *shooting* we need to solve an *initial value problem.*

(a) Rewrite problem (1) as a system of four first order ODE's

$$\frac{dY}{dt} = F(t, Y), \quad Y \in \mathbb{R}^4, \tag{3}$$

where $F(t, Y)$ is a function you need to find. Write a MATLAB function file `ivp.m` which returns the value of $F(t, Y)$. [1]

(b) Rewrite problem (1)-(2) as the combination of an initial value problem with initial conditions depending on z_1 and z_2 (the unknown pieces of initial data at $t = 0$) and conditions that need to be satisfied at the other boundary. Write a MATLAB function file `shootingfunction.m` which, taking z_1 and z_2 as input, gives you the solution to the initial value problem (3) up to $t = 1$ as output. [1]

(c) Use the Matlab function `fsolve` with starting guess $z_1 = 0$ and $z_2 = 0$ to solve the system of nonlinear equations given in part (b) such that the solution to the initial value problem (3) coincides with the solution to the boundary value problem (1)-(2). Plot the solution $u(t)$ satisfying (1) and (2) and its derivatives $u'(t)$, $u''(t)$, $u'''(t)$ for $t \in [0, 1]$ in the same figure. [1]

2. The *Allen-Cahn* equation is a two point boundary value problem which models the transition between phases for a binary alloy or the equilibrium population density of a logistically governed population of organisms with spatial mobility. In the simplest case, it takes the form

$$\frac{d^2u}{dx^2} + \lambda^2 u(1 - u) = 0, \quad u(0) = u(1) = 0. \tag{4}$$

For physical reasons we are interested in *positive* solutions with $u > 0$ when $0 < x < 1$. For large values of λ the non-zero solutions have a *boundary layer* close to $x = 0$ and $x = 1$. This makes them hard to solve by using a shooting method.

(a) Show that in the phase plane $(u, v) = (u, du/dx)$, the solutions of the differential equation satisfy the identity

$$\frac{1}{2}v^2 + \lambda^2 \left(\frac{u^2}{2} - \frac{u^3}{3} \right) = C, \tag{5}$$

where C is a constant, determined by the boundary conditions. Use Matlab to draw the solution curves satisfying (5) for $\lambda = 2$ and varying values of $C = 0 : 1/6 : 2$ in the (u, v) phase-plane. (One possibility is to use the `contour` command) [1]

(b) A *homoclinic* solution of the initial value problem is one that tends to the *same* fixed point in the phase plane (in this case $u = 1, v = 0$) as $x \to \pm\infty$. Show that for such a solution the constant C above is given by

$$C = \frac{\lambda^2}{6}.$$

Hence show that if there is a point x^* such that $u(x^*) = 0$ on such a solution then

$$|u'(x^*)| = v_H \equiv \lambda/\sqrt{3}. \tag{6}$$

As the equation (4) is invariant under translations in space, we may (without loss of generality) take $x^* = 0$. Hence, explain why we can conclude that the solution to (4) must satisfy $0 < u'(0) < \lambda/\sqrt{3}$. Highlight the homoclinic orbit on the same graph as in (a). (Note that the full homoclinic orbit will not have $u > 0$.) [1]

(c) Set up the differential equation (4) as an initial value problem with a shooting parameter $z = v(0) = u'(0)$. Solve this using `ode15s` with all tolerances set to 10^{-10}. Taking $\lambda = 3.5$ and an initial guess of $z = 1$, calculate $z^* = u'(0)$ which solves (4) using `fzero`. Plot the resulting solution $u(x)$. [1]

(d) Now steadily increase λ in steps of 0.25 from 3.5 to 8. At each stage use the value of z^* calculated for the *previous* value of λ as the starting guess for the value at the *next* value of λ. Create a vector of the values of λ and z^* and plot z^* as a function of λ. This plot is called a *bifurcation diagram*. Conjecture where the graph intersects the $\lambda-$axis (i.e. what is the minimum value of λ for which there is a solution.) [1]

(e) For larger values of λ the solution calculated in (b) converges to a portion of the homoclinic solution discussed above. Solutions of (4) close to the homoclinic solution are notoriously hard to calculate by using shooting methods as very small changes to the value of $u'(0)$ have a dramatic effect on the resulting solution $u(x)$ (the problem is *ill conditioned*). To see this, repeat the calculation of (d) up to $\lambda = 9$ and plot all the solution curves (on one plot) and the bifurcation diagram. What do you observe? Interpret your answer using the phase plane portrait of part (b). Now reduce the step size in λ to 0.1. How far along the bifurcation curve can you go? Discuss what you think is happening. [2]

Finite Difference Methods

Whilst shooting methods are often efficient and simple to use to solve two-point boundary value problems, they can (see Q.2) be highly ill-conditioned. We now look at finite difference methods which can perform better.

3. Consider the following differential equation with *Dirichlet* boundary conditions

$$\frac{d^2u}{dx^2} + u = x^2, \quad u(0) = u(1) = 0. \tag{7}$$

(a) Find the *exact* solution to this equation. [1]

(b) Replace the boundary conditions in (7) by $u(0) = u(\pi) = 0$ and find an exact solution. [1]

(c) Consider a division of the interval $[0, 1]$ into intervals of width h where $h = 1/(N-1)$. In this division we set $x_i = (i-1)h$ for $i = 1 \dots N$. Show that

$$u''(x_i) = (u(x_{i+1}) - 2u(x_i) + u(x_{i-1}))/h^2 - h^2 u''''(x_i)/12 + \mathcal{O}(h^4), \quad i = 2, \dots, N-1.$$

By using this result and incorporating the boundary conditions, show that, up to an error of $\mathcal{O}(h^2)$, we can approximate (7) with a linear equation of the form

$$A\mathbf{U} \equiv B_h\mathbf{U} + \mathbf{U} = \mathbf{b} \tag{8}$$

where you should determine A, B_h, $\mathbf{U}$ and $\mathbf{b}$. [2]

(d) By using the `diag` and `ones` commands, write a MATLAB script file `bvp.m` which constructs the matrix `A` and the vector `b` for general N. Using this routine solve the system (8) for $N = 2^i + 1, \quad i = 6, \dots, 12$ using the built-in backslash command. Using `tic` and `toc` (which are more reliable than `cputime`) record the time taken for your program to find $\mathbf{U}$ for each N and estimate how this time varies as a function of N. [2]

(e) By comparing the numerical solution with the exact solution, demonstrate numerically that there is a constant K so that for these odd values of N

$$U_{(N+1)/2} = u(1/2) + Kh^2 + \mathcal{O}(h^4).$$

[1]

4. The Allen-Cahn equation is described in Question 2. Shooting is a poor way of solving this equation and finite difference methods do much better.

(a) Show that we can discretise the differential equation (4) together with the boundary conditions in the form

$$\mathbf{G}(\mathbf{U}) = B_h\mathbf{U} + \lambda^2\mathbf{F}(\mathbf{U}) = \mathbf{0}, \tag{9}$$

where $\mathbf{F}$ is an appropriate nonlinear function of $\mathbf{U}$ and B_h is the matrix given in (8). [1]

(b) The nonlinear system (9) can be solved by using the Newton method. This method generates a series of approximations $\mathbf{U}^n$ to the solution via the iteration

$$J\left(\mathbf{U}^{n+1} - \mathbf{U}^n\right) = -\mathbf{G}(\mathbf{U}^n)$$

where J is the Jacobian matrix defined by $J_{i,j} = \partial G_i / \partial U_j$. Show that if $\mathbf{W}^{n+1} = \mathbf{U}^n - \mathbf{U}^{n+1}$, then $\mathbf{W}^{n+1}$ solves the linear system

$$B_h \mathbf{W}^{n+1} + \lambda^2 D \mathbf{W}^{n+1} = \mathbf{R}^n, \tag{10}$$

$D =$ `diag(1-2u,0)` and $\mathbf{R}^n$ is the *residual vector* given by $\mathbf{R}^n = \mathbf{G}(\mathbf{U}^n)$. The value of $\|\mathbf{R}^n\|_2$ gives a measure of the error of the solution and you should monitor convergence with this norm. [1]

(c) Write a MATLAB script file to implement the Newton-Raphson iteration to solve (9). This file should start with an appropriate initial guess $\mathbf{U}^0$ for $\mathbf{U}$ and should terminate when the residual norm $\|\mathbf{R}\|_2 < 10^{-6}$.
Set $N = 40$. Initially take $\lambda = 4$ and and $\mathbf{U}^0{}_i = 2x_i(1 - x_i)$. Now increase λ to 20 in steps of 1, using the previously calculated value of $\mathbf{U}$ as the initial guess for the next calculation. Plot (on the same graph) the solution in each case as a function of x. Monitor how the residual decreases with each iteration and comment whether it is the expected rate for Newton's method. [2]

ASSESSMENT. Please, do questions 1–4. The optional questions 5 is useful but does not count for assessment.

Paul Milewski

5. *This question is optional but is a useful introduction to some of the problems that you will meet in semester 2.*

The equation

$$\epsilon u'' + u' = -1, \quad u(0) = u(1) = 0. \tag{11}$$

models a convecting and diffusing fluid. For small ϵ the fluid flow is *convection dominated.*

(a) Find the exact solution to this problem and show that it ϵ is small that it has a boundary layer of approximate width ϵ at the origin.

(b) We can discretise the diffusive second derivative term $\epsilon u''$ exactly as before. However, there are several ways in which we can discretise the convective term u'. The three possibilities are

$$\begin{aligned} \text{central difference:} \quad & u' \approx \left(U_{n+1} - U_{n-1}\right)/2h, \\ \text{upwind difference:} \quad & u' \approx \left(U_{n+1} - U_n\right)/h, \\ \text{downwind difference:} \quad & u' \approx \left(U_n - U_{n-1}\right)/h. \end{aligned}$$

Show that the central difference method leads to an error proportional to h^2 whereas upwind and downwind differencing both give errors proportional to h.

(c) By incorporating the boundary conditions show that a central difference discretisation of (11) leads to a problem of the form

$$A\mathbf{U} \equiv \epsilon B_h \mathbf{U} + D_h \mathbf{U} = \mathbf{f} \tag{12}$$

where B_h is as before and

$$D_h = \frac{1}{2h}\begin{pmatrix} 1 & 0 & 0 & 0 & .. & 0 & 0 \\ -1 & 0 & 1 & 0 & .. & 0 & 0 \\ 0 & -1 & 0 & 1 & .. & 0 & 0 \\ .. & .. & .. & .. & .. & .. & .. \\ 0 & 0 & 0 & 0 & .. & 0 & 1 \\ 0 & 0 & 0 & 0 & .. & 0 & 1 \end{pmatrix}, \quad \mathbf{f} = \begin{pmatrix} 0 \\ -1 \\ -1 \\ .. \\ -1 \\ 0 \end{pmatrix} \tag{13}$$

Derive similar matrices D_h corresponding to the case of upwind and downwind differencing.

(d) Write a MATLAB script-file which takes as input the number N of mesh points and ϵ and requests whether you want central, upwind or downwind differences. Then construct the system (12) and solves it for $\mathbf{U}$ by using the Thomas algorithm. Plot the resulting solution against x.

(e) Using central differences and taking $\epsilon = 0.01$ and $N = 10, 20, 40, 50, 60, 100$ compute and plot the resulting solutions, comparing them with the exact solution. What do you notice as N passes through the value of 50? The plots for $N < 50$ are physically incorrect and are due to an ill conditioning in the matrix $C = \epsilon B_h + D_h$ which leads to spurious behaviour when the mesh size is too coarse. Indeed, when N is odd, the matrix D_h has an eigenvector with eigenvalue 0. Determine this eigenvector and use it to help explain your results.

(f) This spurious behaviour can be avoided by taking h so that the matrix C is *diagonally dominant* i.e. if

$$|C_{ii}| \geq \sum_{i \neq j} |C_{ij}|.$$

Show that C is diagonally dominant provided that

$$2\epsilon > |\epsilon - h/2| + |\epsilon + h/2|,$$

and hence it is diagonally dominant only if $h < 2\epsilon$. Show that this is consistent with what you have observed. Unfortunately, this places a severe restriction on the value of h which may make this method unusable in problems with very low diffusion.

(g) An alternative way of avoiding the spurious behaviour is to use either upwind differencing or downwind differencing. By analysing the matrix C show that *one* of these choices always leads to a diagonally dominant matrix. Taking this choice, repeat the calculation in *(e)* above and show that this leads to much more regular behaviour, and by comparing with the exact solution, show that this is at the expense of lower accuracy for small values of h.