MATH 4093/5093 Homework 8 Due Mon, 11/22/10

Problem 1. The polynomials

$$\begin{aligned} S_0(x) &= 4 + 4(x-1) + 13(x-1)^2 - 9(x-1)^3 \ , \quad x \in [1,2] \ , \\ S_1(x) &= a + b(x-2) + c(x-2)^2 + d(x-2)^3 \ , \quad x \in [2,3] \end{aligned}$$

form the *clamped* cubic spline interpolant for some function $f(x)$ that is known to satisfy $f'(1) = -f'(3)$.

(a) Use this information to find the values of the coefficients a, b, c, and d.

(b) Compute $S'(2.5)$.

Problem 2. Download Brady's codes `newton_sys.m`, `LUfactor.m` and `LUsolve.m` from `www.pcs.cnu.edu/~bbradie/msystems.html`. The code `newton_sys.m` finds roots of systems of algebraic equations by using Newton's method. This code uses the codes `LUfactor.m` and `LUsolve.m`, which perform an LU decomposition, respectively forward and backward substitution, and are called in each step of the Newton's iteration.

Solve the nonlinear system

$$\begin{aligned} x_1 - x_2 - x_1^3 &= -9 \\ x_1 + x_2 - x_2^3 &= -22 \end{aligned}$$

by using `newton_sys.m`, starting from the initial approximation $\mathbf{x}^{(0)} = (3\ 4)^T$, with tolerance 10^{-6}. If the files defining the nonlinear system and the Jacobian are called `eqns_newt.m` and `jac_newt.m`, then you have to run `newton_sys.m` as follows:

```
newton_sys ( @eqns_newt, @jac_newt, [3 4], 1e-6, 100 )
```

The exact solution is $\mathbf{x}^* = (2\ 3)^T$. Compute the ℓ_2- norms of the errors at each step: $\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_2$, $\|\mathbf{x}^{(1)} - \mathbf{x}^*\|_2$, $\|\mathbf{x}^{(2)} - \mathbf{x}^*\|_2$, ... by using the Matlab command `norm` (type `help norm` in Matlab for help).

Please write in detail all your calculations (with paper and pencil) that you needed to write your codes, and attach printouts of your Matlab codes and output.

Problem 3. From the web-site `www.pcs.cnu.edu/~bbradie/mbvps.html` download the Matlab code `linfd.m`, and from `www.pcs.cnu.edu/~bbradie/msystems.html` download `tridiagonal.m` (this code is needed for `linfd.m`).

(a) The code `linfd.m` uses finite difference method to solve a linear boundary-value problem with Dirichlet, Neumann, or Robin boundary conditions (as explained in Section 8.2 of the book and, more succinctly, in the file `notes-pde1-5093-f10.pdf` of the lecture notes). Run this code to solve the linear Dirichlet boundary value problem

$$\begin{aligned} &y'' = 4xy' + (1-4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ , \\ &y(0) = 1 \ , \quad y\left(\tfrac{\pi}{2}\right) = 2\mathrm{e}^{\pi^2/4} \ , \end{aligned}$$

whose exact solution is

$$y_{\text{exact}}(x) = (1+\sin x)\mathrm{e}^{x^2} \ .$$

Plot both the exact and the approximate solution on the same graph, using lines for the exact solution and stars for the approximate solution. Note that the code `linfd.m`

Suppose that you want to test the accuracy of the code by dividing the interval $[a,b] = [0, \frac{\pi}{2}]$ into $N = 100$ subintervals of equal length, in which case you have to create a uniform grid of $(N+1)$ points x_j, $j = 0, 1, \ldots, N$. Then you want to compute the values of the exact solution at each of of the exact solution at each of the grid points x_j, $j = 0, 1, \ldots, N$. You can do this with the commands

```
N=100;
ti=linspace(0,pi/2,N+1);
theor=(1+sin(ti)).*exp(ti.^2);
```

Then you run the code `linfd.m` with $N = 100$, and create a vector `wi` of length $N+1$ with the approximate values of the solution of the boundary value problem.

(b) Solve the problem from part (a) with $N = 10$, 100, 1000 and 10000, and compute the norms $\|\texttt{exact} - \texttt{wi}\|_\infty$ of the absolute errors for each N (you can do this with the command `norm(theor-wi,Inf)`). What seems to be the order of convergence?

(c) Solve the boundary value problem

$$\begin{aligned} &y'' = 4xy' + (1-4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ , \\ &y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) + y'\left(\tfrac{\pi}{2}\right) = 4\pi\mathrm{e}^{\pi^2/4} \ , \end{aligned}$$

and then the boundary value problem

$$\begin{aligned} &y'' = 4xy' + (1-4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ , \\ &y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = 0 \ , \end{aligned}$$

in both cases use $N = 10000$. These two boundary value problems have the same solution as the one in part (a). For each of the two boundary value problems in this part plot both the theoretic and the numerical solution. What do you observe?

(d) Do the same as in part (c) with the boundary value problems

$$y'' = 4xy' + (1-4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = 0.0001 \ ,$$

and

$$y'' = 4xy' + (1-4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = -0.0001 \ .$$

What is the moral of your observations?

Problem 4. From the web-site `www.pcs.cnu.edu/~bbradie/mbvps.html` download the Matlab code `linshoot.m`. In this problem you will study this code and its performance only for the case of Dirichlet boundary value problems. In the notations of this code, the boundary conditions are defined by the 3-dimensional vectors `alpha` and `beta` in the following way:

$$\begin{aligned} \texttt{alpha}(1)\, u(a) + \texttt{alpha}(2)\, u'(a) &= \texttt{alpha}(3) \\ \texttt{beta}(1)\, u(b) + \texttt{beta}(2)\, u'(b) &= \texttt{beta}(3) \ , \end{aligned}$$

where $[a, b]$ is the interval in which we are solving the boundary value problem. Therefore, the case of Dirichlet boundary conditions corresponds to $\texttt{alpha}(2) = \texttt{beta}(2) = 0$.

Note that the output of this code is not a vector, but a $2 \times (N+1)$ matrix, the first column of which contains the approximate values of the solution evaluated at the nodes, and the second column contains the approximate values of the derivatives of the solution evaluated at the nodes. If you call the code like this:

```
wi = linshoot(@coeffs, 0.0, pi/2.0, 100, [1 0 1], [1 0 2*exp(pi^2/4)] );
```

then `wi` will be a 2×101 matrix. If you then want to use only the first row of `wi`, you can type `wi(1,:)`, which will be a vector of dimension 101.

(a) Explain in detail what the code `linshoot.m`, make connection with the theory studied in class, write all necessary formulae.

(b) Do with this code the same as in Problem 3(a).

(c) Do with this code the same as in Problem 3(b).

Problem 5 (FOOD FOR THOUGHT ONLY, NOT TO BE TURNED IN!)
Download the code `nonlinshoot.m` from `www.pcs.cnu.edu/~bbradie/mbvps.html`.

(a) Explain what this code does in the case of Dirichlet boundary conditions; the Dirichlet boundary conditions are encoded in the same way as in the code `linshoot.m`, studied in problem 4 (using the 3-dimensional vectors `alpha` and `beta`).

(b) In this part of the problem you will solve the nonlinear Dirichlet boundary value problem

$$y'' = -2y^2 + 8x^2y^3 \ , \qquad x \in [0,1] \ ,$$
$$y(0) = 1 \ , \quad y(1) = \tfrac{1}{2} \ ,$$

whose exact solution is $y_{\text{exact}}(x) = \frac{1}{1+x^2}$.

Create the file `coeffs_nonlin.m` containing the following two lines:

```
function rhs = coeffs_nonlin( x, u, up )
  rhs = - 2 * u^2 + 8 * x^2 * u^3 ;
```

and run the following Matlab commands for $N = 10$, $N = 100$, and $N = 1000$:

```
ti=linspace(0,1,N+1);
theor=1./(1+ti.^2);
wi=nonlinshoot(@coeffs_nonlin,0.0,1.0,N,[1 0 1],[1 0 0.5],1e-6,100,[0 1]);
norm_diffs=norm(theor-wi(1,:),Inf)
plot(ti,theor,'-',ti,wi(1,:),'*')
```

(c) What is the observed order of convergence from the data collected in part (b)?