CSC 446/2310 ASSIGNMENT #3 Due: 5 April 2019.

This assignment is due at the start of our lecture on Friday, 5 April 2019. You can hand the assignment in late, with a penalty of 25%, on Monday, 8 April 2019. Since classes end on Friday, 5 April 2019, if you do submit your assignment late, either give it to me in my office, BA 4228, or email it to me before 11:59 PM on Monday, 8 April 2019. For more details about late assignments, see the course outline `http://www.cs.toronto.edu/~krj/courses/446-2310/course.outline.2019.pdf`.

For the questions that require you to write a MatLab program, hand in the program and its output as well as any written answers requested in the question. Your program should conform to the usual CS standards for comments, good programming style, etc. Try to format the output from your program so that it is easy for your TA to read and to understand your results.

Before writing your MatLab programs, you might find it useful to read the MatLab documentation on our course webpage `http://www.cs.toronto.edu/~krj/courses/446-2310/`. You should use sparse matrices as much as possible. Read help in MatLab on sparfun, sparse and spdiags. After you have initialized a matrix using the MatLab sparse matrix routines, you can solve a system $Ax = b$ in MatLab by `x = A \ b`. MatLab will use an efficient sparse matrix factorization to solve the system.

Everyone who has registered for this course should have an account on the CS Teaching Labs Computer System (i.e., formerly called the CDF System). You should be able to access the system remotely over the internet. There is more information about accessing your account at the start of Assignment 1. If you have any trouble with this, let me know and I will try to help.

Throughout this assignment, I refer to MatLab, but you can use Octave or one of the other MatLab clones instead. (See "MatLab Clones" on the course webpage `http://www.cs.toronto.edu/~krj/courses/446-2310/`.) However, MatLab clones are not 100% compatible with MatLab. So, run each of your final programs through MatLab to make sure that your program really runs under MatLab, since, when your TA marks your program, he will want to see a working MatLab program.

1. [10 marks]

Consider the two-point boundary value problem (BVP)

$$\begin{aligned} -y'' + y &= (\pi^2 + 1)\sin(\pi x), \quad x \in (0,1) \\ y(0) &= 1, \quad y'(1) = \frac{1}{2}\left(\mathrm{e} - \frac{1}{\mathrm{e}}\right) - \pi \end{aligned} \tag{1}$$

It is easy to verify that the solution to this problem is

$$y(x) = \frac{1}{2}\left(\mathrm{e}^x + \mathrm{e}^{-x}\right) + \sin(\pi x)$$

(Note that MatLab has a built-in value for π. Read "help pi" in MatLab. Moreover, you can compute e in MatLab from `exp(1)`.)

Write a MatLab program that uses the Ritz-Galerkin method with piecewise linear hat (i.e., chapeau) basis functions (defined on page 176 of your textbook) on an equally spaced grid to solve the BVP (1). That is, let the gridpoints be $x_i = ih$ for $i = 0, 1, \ldots, m$ and $h = 1/m$, where m is an integer. Note that $x_0 = 0$ and $x_m = 1$. (See below for the choices of m.)

The approximate solution generated by the Ritz-Galerkin method has the form

$$y_m(x) = \varphi_0(x) + \sum_{k=1}^{m} \gamma_k \varphi_k(x)$$

where $\varphi_0(x)$ satisfies the boundary conditions in (1), $\varphi_k(x)$, $k \in \{1, 2, \ldots, m\}$, is the hat basis function defined on page 176 of your textbook and the γ_k are determined by solving the Galerkin equations (see equation (9.7) on page 174 of your textbook).

Also, read the section on *natural boundary conditions* on page 182 of your textbook.

For each of $m = 10, 20, 40, 80, 160, 320, 640$, use your program to compute the Ritz-Galerkin solution $y_m(x)$ to the BVP (1).

The maximum error in the numerical solution at the gridpoints $\{x_i : i = 1, \ldots, m\}$,

$$\max\{|y(x_i) - y_m(x_i)| : i = 1, \ldots, m\} \tag{2}$$

is a good approximation to the infinity norm of the error in the numerical solution

$$\|y - y_m\|_\infty = \max\{|y(x) - y_m(x)| : x \in [0,1]\}$$

Compute and print the maximum error in the numerical solution at the gridpoints (2) for $m = 10, 20, 40, 80, 160, 320, 640$.

How does this error decrease with $h = 1/m$?

2. [10 marks]

Repeat Question 1, but use the cubic B-spline basis functions defined below in place of the piecewise linear hat (i.e., chapeau) basis functions that you used in Question 1.

For each m, let the associated grid be $\{x_i = i/m : i = 0, 1, \ldots, m\}$ and the associated stepsize $h = 1/m$, as in Question 1. For each $i = 0, 1, \ldots, m-1$, define the "regular" cubic B-spline basis function on this grid by

$$B_i(x) = \begin{cases} \frac{1}{6}\left(\frac{x - x_i}{h}\right)^3 & \text{if } x \in [x_i, x_{i+1}] \\ \frac{2}{3} - 2\left(\frac{x - x_i}{h}\right) + 2\left(\frac{x - x_i}{h}\right)^2 - \frac{1}{2}\left(\frac{x - x_i}{h}\right)^3 & \text{if } x \in [x_{i+1}, x_{i+2}] \\ -\frac{22}{3} + 10\left(\frac{x - x_i}{h}\right) - 4\left(\frac{x - x_i}{h}\right)^2 + \frac{1}{2}\left(\frac{x - x_i}{h}\right)^3 & \text{if } x \in [x_{i+2}, x_{i+3}] \\ \frac{1}{6}\left(\frac{x_{i+4} - x}{h}\right)^3 & \text{if } x \in [x_{i+3}, x_{i+4}] \\ 0 & \text{otherwise} \end{cases}$$

We also need two "special" cubic B-spline basis function at the left end of the grid:

$$B_{-1}(x) = \begin{cases} \frac{3}{2}\left(\frac{x}{h}\right)^2 - \frac{11}{12}\left(\frac{x}{h}\right)^3 & \text{if } x \in [0, h] \\ -\frac{3}{2} + \frac{9}{2}\left(\frac{x}{h}\right) - 3\left(\frac{x}{h}\right)^2 + \frac{7}{12}\left(\frac{x}{h}\right)^3 & \text{if } x \in [h, 2h] \\ \frac{1}{6}\left(\frac{3h - x}{h}\right)^3 & \text{if } x \in [2h, 3h] \\ 0 & \text{otherwise} \end{cases}$$

$$B_{-2}(x) = \begin{cases} 3\left(\frac{x}{h}\right) - \frac{9}{2}\left(\frac{x}{h}\right)^2 + \frac{7}{4}\left(\frac{x}{h}\right)^3 & \text{if } x \in [0, h] \\ \frac{1}{4}\left(\frac{2h - x}{h}\right)^3 & \text{if } x \in [h, 2h] \\ 0 & \text{otherwise} \end{cases}$$

Write a MatLab program that uses the Ritz-Galerkin method with the cubic B-spline basis functions defined above on an equally spaced grid to solve the BVP (1).

The approximate solution generated by the Ritz-Galerkin method has the form

$$y_m(x) = \varphi_0(x) + \sum_{k=-2}^{m-1} \gamma_k B_k(x)$$

where $\varphi_0(x)$ satisfies the boundary conditions in (1).

For each of $m = 10, 20, 40, 80, 160, 320, 640,$ use your program to compute the Ritz-Galerkin solution $y_m(x)$ to the BVP (1).

The maximum error in the numerical solution at the gridpoints $\{x_i : i = 1, \ldots, m\}$,

$$\max \{|y(x_i) - y_m(x_i)| : i = 1, \ldots, m\} \tag{3}$$

is a good approximation to the infinity norm of the error in the numerical solution

$$\|y - y_m\|_\infty = \max \{|y(x) - y_m(x)| : x \in [0, 1]\}$$

Compute and print the maximum error in the numerical solution at the gridpoints (3) for $m = 10, 20, 40, 80, 160, 320, 640$.

How does this error decrease with $h = 1/m$?

How does the error for this question compare with the error for Question 1?

3. [10 marks]

Consider the two-point boundary value problem (BVP)

$$\begin{aligned} -y'' + 10^4\, y &= 0, \quad x \in (0,1) \\ y(0) &= y(1) = 1. \end{aligned} \tag{4}$$

It is easy to verify that the solution to this problem is

$$y(x) = c_1 \mathrm{e}^{100x} + c_2 \mathrm{e}^{-100x}$$

where

$$c_1 = \frac{1 - \mathrm{e}^{-100}}{\mathrm{e}^{100} - \mathrm{e}^{-100}} \qquad c_2 = \frac{\mathrm{e}^{100} - 1}{\mathrm{e}^{100} - \mathrm{e}^{-100}}$$

Write a MatLab program that uses the Ritz-Galerkin method with piecewise linear hat (i.e., chapeau) basis functions (defined on page 176 of your textbook) on an equally spaced grid to solve the BVP (4). That is, let the gridpoints be $x_i = ih$ for $i = 0, 1, \ldots, m+1$ and $h = 1/(m+1)$, where m is an integer. (See below for the choices of m.)

Write another MatLab program that uses the Ritz-Galerkin method with piecewise linear hat (i.e., chapeau) basis functions on an unequally spaced grid to solve the BVP (4). In particular, make the stepsize $h_i = x_{i+1} - x_i$ smaller where the solution to (4) varies more rapidly and make the stepsize $h_i = x_{i+1} - x_i$ larger where the the solution to (4) is smoother. However, keep the same number of gridpoints, m, in the unequally spaced grid as in the equally spaced one described above.

The piecewise linear hat (i.e., chapeau) basis functions defined on page 176 of your textbook assume that the gridpoints are equally spaced. However, it is easy to extend the definition of the piecewise linear hat basis functions to unequally spaced gridpoints. Let

$$\varphi_k(x) = \begin{cases} \dfrac{x - x_{k-1}}{x_k - x_{k-1}} & \text{if } x \in [x_{k-1}, x_k] \\ \dfrac{x_{k+1} - x}{x_{k+1} - x_k} & \text{if } x \in [x_k, x_{k+1}] \\ 0 & \text{otherwise} \end{cases}$$

The approximate solution generated by the Ritz-Galerkin method has the form

$$y_m(x) = \varphi_0(x) + \sum_{k=1}^{m} \gamma_k \varphi_k(x)$$

where $\varphi_0(x)$ satisfies the boundary conditions in (4), $\varphi_k(x)$, $k \in \{1, 2, \ldots, m\}$, is the hat basis function defined above and the γ_k are determined by solving the Galerkin equations (see (9.7) on page 174 of your textbook).

For each of $m = 9, 19, 39, 79, 159, 319, 639,$ use each of your two programs (one for the equally spaced grid and the other for the unequally spaced grip) to compute the Ritz-Galerkin solution $y_m(x)$ to the BVP (4).

The maximum error in the numerical solution at the gridpoints $\{x_i : i = 1, \ldots, m\}$,

$$\max \{|y(x_i) - y_m(x_i)| : i = 1, \ldots, m\} \tag{5}$$

is a good approximation to the infinity norm of the error in the numerical solution

$$\|y - y_m\|_\infty = \max \{|y(x) - y_m(x)| : x \in [0, 1]\}$$

Compute and print the maximum error in the numerical solution at the gridpoints (5) for $m = 9, 19, 39, 79, 159, 319, 639$.

Try to choose an unequally spaced grid so that the error for the Ritz-Galerkin method on the unequally spaced grid is much smaller than the error for the Ritz-Galerkin method on the equally spaced grid.

4. [10 marks]

As I explained briefly in class, you can extend the Ritz-Galerkin method from 1–D problems to 2–D or 3–D problems by using a tensor product approach, provided that the domain of the problem is a rectangle (or some other suitable domain).

To see how this works, consider the 2–D boundary-value problem (BVP)

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = 32x(1-x) + 32y(1-y) \qquad \text{for } x \in (0,1) \text{ and } y \in (0,1)$$

with Dirichlet boundary conditions $u(x,y) = 0$ if $x = 0$, $x = 1$, $y = 0$ or $y = 1$. It is easy to check that the solution to this problem is

$$u(x,y) = 16x(1-x)y(1-y)$$

In this example, we'll use the piecewise linear hat (i.e., chapeau) basis functions, but this approach works just as well for any other basis functions. In particular, we could use the cubic B-spline basis functions defined in Question 2.

Let $\varphi_k(x)$ be the 1–D piecewise linear hat basis function defined on page 176 of your textbook. Similarly let $\varphi_l(y)$ be a 1–D piecewise linear hat basis function, but in the y variable, rather than the x variable. For the 2–D piecewise linear hat basis function, let $\varphi_{k,l}(x,y) = \varphi_k(x)\varphi_l(y)$.

Then the Ritz-Galerkin method can be formulated using the 2–D basis functions $\varphi_{k,l}(x,y)$ in almost exactly the same way as for the 1–D case, except that all integrals in the 2–D case are 2–D integrals.

Solve the 2–D BVP above on the unit square with a uniform grid (x_i, y_j), where $x_i = i/(m+1)$ for $i = 1, \ldots, m$ and $y_j = j/(m+1)$ for $j = 1, \ldots, m$.

The approximate solution generated by the Ritz-Galerkin method has the form

$$u_m(x,y) = \sum_{k=1}^{m} \sum_{l=1}^{m} \gamma_{k,l} \varphi_{k,l}(x,y)$$

where $\varphi_{k,l}(x,y)$ is the 2–D hat basis function described above and the $\gamma_{k,l}$ are determined by solving the Galerkin equations (similar to (9.7) on page 174 of your textbook, but extended from 1–D to 2–D).

Note that we don't need at $\varphi_0(x,y)$ in this problem because we already have zero boundary conditions.

For each of $m = 9, 19, 39, 79, 159, 319, 639$, use your program to compute the Ritz-Galerkin solution $u_m(x,y)$ to the 2–D BVP above.

The maximum error in the numerical solution at the gridpoints $\{(x_i, y_j) : i = 1, \ldots, m,\ j = 1, \ldots, m\}$,

$$\max \{|u(x_i, y_j) - u_m(x_i, y_j)| : i = 1, \ldots, m,\ j = 1, \ldots, m\} \tag{6}$$

is a good approximation to the infinity norm of the error in the numerical solution

$$\|u - u_m\|_\infty = \max\{|u(x,y) - u_m(x,y)| : x \in [0,1],\ y \in [0,1]\}$$

Compute and print the maximum error in the numerical solution at the gridpoints (6) for $m = 9, 19, 39, 79, 159, 319, 639$.