HW1, Math 228A

Date due and handed in 10/12/2010

UC Davis, California Fall 2010

Nasser M. Abbasi

Fall 2010 Compiled on June 18, 2019 at 7:26pm [public]

Contents

1 Problem description

Math 228A
Homework 1
Due Tuesday, 10/12/10

1. Let L be the linear operator $Lu = u_{xx}$, $u_x(0) = u_x(1) = 0$.
   (a) Find the eigenfunctions and corresponding eigenvalues of L.
   (b) Show that the eigenfunctions are orthogonal in the $L^2[0,1]$ inner product:
   $$\langle u, v\rangle = \int_0^1 uv\,dx.$$
   (c) It can be shown that the eigenfunctions, $\phi_j(x)$, form a complete set in $L^2[0,1]$. This means that for any $f \in L^2[0,1]$, $f(x) = \sum_j \alpha_j \phi_j(x)$. Express the solution to
   $$u_{xx} = f, \quad u_x(0) = u_x(1) = 0, \tag{1}$$
   as a series solution of the eigenfunctions.
   (d) Note that equation (1) does not have a solution for all f. Express the condition for existence of a solution in terms of the eigenfunctions of L.
2. Define the functional $F : X \to \Re$ by
   $$F(u) = \int_0^1 \frac{1}{2}(u_x)^2 + fu\,dx,$$
   where X is the space of real valued functions on $[0,1]$ that have at least one continuous derivative and are zero at $x = 0$ and $x = 1$. The Frechet derivative of F at a point u is defined to be the linear operator $F'(u)$ for which
   $$F(u+v) = F(u) + F'(u)v + R(v),$$
   where
   $$\lim_{||v||\to 0} \frac{||R(v)||}{||v||} = 0.$$
   One way to compute the derivative is
   $$F'(u)v = \lim_{\epsilon\to 0} \frac{F(u+\epsilon v) - F(u)}{\epsilon}.$$
   Note that this looks just like a directional derivative.
   (a) Compute the Frechet derivative of F.
   (b) $u \in X$ is a critical point of F if $F'(u)v = 0$ for all $v \in X$. Show that if u is a solution to the Poisson equation
   $$u_{xx} = f, \quad u(0) = u(1) = 0,$$
   then it is a critical point of F.

   Finite element methods are based on these "weak formulations" of the problem. The Ritz method is based on minimizing F and the Galerkin method is based on finding the critical points of $F'(u)$.

1

Figure 1: problem description

2 Problem 1

L is a second order differential operator defined by $Lu \equiv u_{xx}$ with boundary conditions on u given as $u_x(0) = u_x(1) = 0$

2.1 part a

Let $\phi(x)$ be an eigenfunction of the operator L associated with an eigenvalue λ. To obtain the eigenfunctions and eigenvalues, we solve an eigenvalue problem $L\phi = \lambda\phi$ where λ is scalar. Hence the problem is to solve the differential equation

$$\phi_{xx} - \lambda\phi = 0 \tag{1}$$

with B.C. given as $\phi'(0) = \phi'(1) = 0$. The characteristic equation is

$$r^2 - \lambda = 0$$

The roots are $r = \pm\sqrt{\lambda}$, therefore the solution to the eigenvalue problem (1) is

$$\phi(x) = c_1 e^{\sqrt{\lambda}x} + c_2 e^{-\sqrt{\lambda}x} \tag{2}$$

Where c_1,c_2 are constants.

$$\phi'(x) = c_1\sqrt{\lambda}e^{\sqrt{\lambda}x} - \sqrt{\lambda}c_2 e^{-\sqrt{\lambda}x} \tag{3}$$

First we determine the allowed values of the eigenvalues λ which satisfies the boundary conditions.

1. Assume $\lambda = 0$ The solution (2) becomes $\phi(x) = c_1 + c_2$. Hence the solution is a constant. In other words, when the eigenvalue is zero, the eigenfunction is a constant. Let us now see if this eigenfunction satisfies the B.C. Since $\phi(x)$ is constant, then $\phi'(x) = 0$, and this does satisfy the B.C. at both $x = 0$ and $x = 1$. Hence $\lambda = 0$ is an eigenvalue with a corresponding eigenfunction being a constant. We can take the constant as 1.

2. Assume $\lambda > 0$ From the first BC we have, from (3), that $\phi'(0) = 0 = c_1\sqrt{\lambda} - \sqrt{\lambda}c_2$ or
$$c_1 = c_2$$
and from the second BC we have that $\phi'(1) = 0 = c_1\sqrt{\lambda}e^{\sqrt{\lambda}} - \sqrt{\lambda}c_2e^{-\sqrt{\lambda}}$ or
$$c_1e^{\sqrt{\lambda}} - c_2e^{-\sqrt{\lambda}} = 0$$
From the above 2 equations, we find that $e^{\sqrt{\lambda}} = e^{-\sqrt{\lambda}}$ which is not possible for positive λ. Hence λ can not be positive.

3. Assume $\lambda < 0$. Let $\lambda = -\beta^2$ form some positive β. Then the solution (2) becomes
$$\phi(x) = c_1e^{i\beta x} + c_2e^{-i\beta x}$$
which can be transformed using the Euler relation to obtain
$$\begin{aligned}\phi(x) &= c_1\cos\beta x + c_2\sin\beta x\\ \phi'(x) &= -c_1\beta\sin\beta x + c_2\beta\cos\beta x \qquad (4)\end{aligned}$$
Now consider the BC's. Since $\phi'(0) = 0$ we obtain $c_2 = 0$ and from $\phi'(1) = 0$ we obtain $0 = c_1\beta\sin\beta$ and hence for non trivial solution, i.e. for $c_1 \neq 0$, we must have that
$$\sin\beta = 0$$
or
$$\beta = \pm n\pi$$
but since β is positive, we consider only $\beta_n = n\pi$, where n is positive integer $n = 1, 2, 3, \cdots$

Conclusion: The eigenvalues are
$$\lambda_n = -(\beta_n)^2 = -(n\pi)^2 = \left\{0, -\pi^2, -(2\pi)^2, -(3\pi)^2, \cdots\right\}$$
And the corresponding eigenfunctions are$\phi_n(x) = \cos\beta_n x = \cos n\pi x = \{1, \cos\pi x, \cos 2\pi x, \cos 3\pi x, \cdots\}$

where $n = 0, 1, 2, \cdots$

2.2 part (b)

Given inner product defined as $\langle u, v\rangle = \int_0^1 uv dx$, then
$$\begin{aligned}\langle\phi_n, \phi_m\rangle &= \int_0^1 (\cos\beta_n x)(\cos\beta_m x)\,dx\\ &= \int_0^1 (\cos n\pi x)(\cos m\pi x)\,dx\\ &= \begin{cases} 0 & n \neq m\\ \frac{1}{2} & n = m\end{cases}\end{aligned}$$
Also, the first eigenfucntion, $\phi_0(x) = 1$ is orthogonal to all other eigenfunctions, since $\int_0^1 (\cos n\pi x)\,dx = \frac{1}{n\pi}[\sin n\pi x]_0^1 = 0$ for any integer $n > 0$.

Hence all the eigenfunctions are orthogonal to each others in $L^2[0, 1]$ space.

2.3 Part (c)

Given

$$u_{xx} = f$$

$u_x(0) = u_x(1) = 0$. This is $Lu = f$. We have found the eigenfunctions $\phi(x)$ of L above. These are basis of the function space of L where f resides in. We can express f as a linear combination of the eigenfunctions of the operator L, hence we write

$$f(x) = \sum_{n=0}^{\infty} a_n \phi_n(x)$$

where $\phi_n(x)$ is the n^{th} eigenfunction of L and a_n is the corresponding coordinate (scalar).

Therefore the differential equation above can be written as

$$Lu = f(x) = \sum_{n=0} a_n \phi_n(x) \tag{1}$$

But since

$$L\phi_n = \lambda_n \phi_n$$

Then

$$L^{-1} = \frac{1}{\lambda_n}$$

Therefore, using (1), the solution is

$$u(x) = \sum_n \left(\frac{a_n}{\lambda_n}\right) \phi_n(x) \tag{2}$$

Now to find a_n, using $f(x) = \sum_n a_n \phi_n(x)$, we multiply each side by an eigenfunction, say $\phi_m(x)$ and integrate

$$\begin{aligned}
\int_0^1 \phi_m(x) f(x)\, dx &= \int_0^1 \phi_m(x) \sum_n a_n \phi_n(x)(x)\, dx \\
&= \int_0^1 \sum_n a_n \phi_m(x) \phi_n(x)\, dx \\
&= \sum_n a_n \int_0^1 \phi_m(x) \phi_n(x)\, dx
\end{aligned}$$

The RHS is $1/2$ when $n = m$ and zero otherwise, hence the above becomes

$$\int_0^1 \phi_n(x) f(x)\, dx = \frac{a_n}{2}$$

Or

$$a_n = 2 \int_0^1 \cos(n\pi x) f(x)\, dx \tag{3}$$

Where a_n as given by (3).

If we know $f(x)$ we can determine a_n and hence the solution is now found.

2.4 part (d)

The solution found above

$$u(x) = \sum_n \left(\frac{a_n}{\lambda_n}\right) \phi_n(x)$$

Is not possible for all f. Only an f which has $a_0 = 0$ is possible. This is because $\lambda_0 = 0$, then a_0 has to be zero to obtain a solution (since L^{-1} does not exist if an eigenvalue is zero).

$a_0 = 0$ implies, by looking at (3) above, that when $n = 0$ we have

$$0 = \int_0^1 f(x)\, dx$$

So only the functions $f(x)$ which satisfy the above can be a solution to $Lu = f$ with the B.C. given.

To review: We found that $\lambda = 0$ to be a valid eigenvalue due to the B.C. being Von Neumann boundary conditions. This in resulted in a_0 having to be zero. This implied that $\int_0^1 f(x)\, dx = 0.$

Having a zero eigenvalue effectively removes one of the space dimensions that $f(x)$ can resides in.

In addition to this restriction, the function $f(x)$ is *assumed* to meet the Dirichlet conditions for Fourier series expansion, and these are

1. $f(x)$ must have a finite number of extrema in any given interval
2. $f(x)$ must have a finite number of discontinuities in any given interval
3. $f(x)$ must be absolutely integrable over a period.
4. $f(x)$ must be bounded

3 Problem 2

3.1 Part (a)

Applying the definition given

$$F'(u)\, v = \lim_{\varepsilon \to 0} \frac{F(u + \varepsilon v) - F(u)}{\varepsilon} \tag{1}$$

And using $F(u) = \int_0^1 \frac{1}{2}(u_x)^2 + f\ u\ dx$, then (1) becomes

$$F'(u)\, v = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \left(\int_0^1 \frac{1}{2}\left[(u + \varepsilon v)_x\right]^2 + f\ \ (u + \varepsilon v)\ \ dx - \int_0^1 \frac{1}{2}(u_x)^2 + f\ u\ dx \right)$$

Simplify the above, we obtain

$$F'(u)\, v = \lim_{\varepsilon \to 0} \left(\int_0^1 \frac{\varepsilon}{2} v_x^2 dx + \int_0^1 u_x v_x dx + \int_0^1 f v dx \right)$$

Hence, as $\varepsilon \to 0$ only the first integral above vanishes (since v_x is bounded), and we have

$$F'(u)\, v = \int_0^1 u_x v_x + f v dx \tag{1A}$$

3.2 Part (b)

The solution to $u_{xx} = f(x)$ with $u(0) = u(1) = 0$ was found in class to be

$$u(x) = \sum_n \left(\frac{a_n}{\lambda_n} \right) \phi_n(x) \tag{2}$$

where

$$\phi_n(x) = \sin(n\pi x)$$

are the eigenfunctions associated with the eigenvalues $\lambda_n = -n^2\pi^2$.

Now we can use this solution in the definition of $F'(u)v$ found in (1A) from part (a). Substitute $u(x)$ from (2) into (1A), and also substitute $f = \sum_n a_n \phi_n(x)$ into (1A), we obtain

$$F'(u)v = \int_0^1 \left(\sum_n \left(\frac{a_n}{\lambda_n} \right) \phi_n(x) \right)' v' + \left(\sum_n a_n \phi_n(x) \right) v \; dx \tag{4}$$

We need to show that the above becomes zero for any $v(x) \in X$.

$$\begin{aligned} F'(u)v &= \int_0^1 \sum_n v' \left(\frac{a_n}{\lambda_n} \right) \phi_n'(x) + \sum_n v a_n \phi_n(x)\, dx \\ &= \int_0^1 \sum_n \left(v' \left(\frac{a_n}{\lambda_n} \right) \phi_n'(x) + v a_n \phi_n(x) \right) dx \\ &= \sum_n a_n \left(\int_0^1 \frac{1}{\lambda_n} v' \phi_n'(x) + v \phi_n(x)\, dx \right) \end{aligned} \tag{5}$$

Now we pay attention to the integral term above. If we can show this is zero, then we are done.

$$\begin{aligned} I &= \frac{1}{\lambda_n} \int_0^1 v' \phi_n'(x) + \int_0^1 v \phi_n(x)\, dx \\ &= I_1 + I_2 \end{aligned} \tag{6}$$

Integrate by parts I_1

$$\begin{aligned} I_1 &= \frac{1}{\lambda_n} \int_0^1 \overbrace{\phi_n'(x)\, v' dx}^{udv} \\ &= \frac{1}{\lambda_n} \left([\phi_n'(x)\, v]_0^1 - \int_0^1 v(x)\, \phi_n''(x)\, dx \right) \\ &= \frac{1}{\lambda_n} \left(\overbrace{[v(1)\, \phi_n'(1) - v(0)\, \phi_n'(0)]}^{\text{zero due to boundaries on } v(x) \in X} - \int_0^1 v(x)\, \phi_n''(x)\, dx \right) \\ &= -\frac{1}{\lambda_n} \int_0^1 v(x)\, \phi_n''(x)\, dx \end{aligned}$$

But since $\phi_n(x) = \sin n\pi x$, then $\phi_n'(x) = n\pi \cos n\pi x$ and $\phi_n''(x) = -n^2\pi^2 \sin n\pi x = -n^2\pi^2$ $\phi_n(x)$ then

$$I_1 = \frac{n^2\pi^2}{\lambda_n} \int_0^1 v(x)\, \phi_n(x)\, dx$$

But also $\lambda_n = -n^2\pi^2$ hence the above becomes

$$I_1 = -\int_0^1 v(x)\, \phi_n(x)\, dx$$

Therefore (6) can be written as

$$\begin{aligned} I &= I_1 + I_2 \\ &= -\int_0^1 v(x)\, \phi_n(x)\, dx + \int_0^1 v(x)\, \phi_n(x)\, dx \\ &= 0 \end{aligned}$$

Therefore, from (5), we see that

$$F'(u)v = 0$$

Hence we showed that if u is solution to $u_{xx} = f$ with $u(0) = u(1) = 0$, then $F'(u)v = 0$.

4 Problem (3)

4.1 Part (a)

Notations used: let $\tilde{f}$ to mean the approximate discrete solution at a grid point. Let f to mean the exact solution.

Using the method of undetermined coefficients, let the second derivative approximation be

$$\tilde{f}''(x) = af\left(x - \frac{h}{2}\right) + bf(x) + cf(x+h) \tag{1}$$

Where a, b, c are constants to be found. Now using Taylor expansion, since

$$f(x+\Delta) = f(x) + \Delta f'(x) + \frac{\Delta^2}{2!}f''(x) + \frac{\Delta^3}{3!}f'''(x) + O\left(h^4\right)$$

Hence apply the above to each of the terms in the RHS of (1) and simplify

$$\begin{aligned}
f\left(x - \frac{h}{2}\right) &= f(x) - \frac{h}{2}f'(x) + \frac{\left(-\frac{h}{2}\right)^2}{2!}f''(x) + \frac{\left(-\frac{h}{2}\right)^3}{3!}f''(x) + \frac{\left(-\frac{h}{2}\right)^4}{4!}f^{(4)}(x) + O\left(h^5\right)\\
f(x) &= f(x)\\
f(x+h) &= f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{3!}f''(x) + \frac{h^4}{4!}f^{(4)}(x) + O\left(h^5\right)
\end{aligned}$$

Substitute the above 3 terms in (1)

$$\begin{aligned}
\tilde{f}''(x) = a&\left(f(x) - \frac{h}{2}f'(x) + \frac{h^2}{8}f''(x) - \frac{h^3}{8\times 6}f'''(x) + \frac{h^4}{16\times 24}f^{(4)}(x) + O\left(h^5\right)\right)\\
&+ bf(x)\\
&+ c\left(f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{6}f'''(x) + \frac{h^4}{24}f^{(4)}(x) + O\left(h^5\right)\right)
\end{aligned}$$

Collect terms

$$\begin{aligned}
\tilde{f}''(x) = (a+b+c)f(x) + f'(x)h\left(-\frac{a}{2}+c\right) + f''(x)h^2\left(\frac{a}{8}+\frac{c}{2}\right) + f'''(x)h^3\left(\frac{-a}{8\times 6}+\frac{c}{6}\right) \\
+ f^{(4)}h^4\left(\frac{a}{16\times 24}+\frac{c}{24}\right) + O\left(h^5\right)
\end{aligned} \tag{2}$$

Hence for $\tilde{f}''(x)$ to best approximate $f''(x)$, we need

$$\begin{aligned}
(a+b+c) &= 0\\
-\frac{a}{2}+c &= 0\\
h^2\left(\frac{a}{8}+\frac{c}{2}\right) &= 1
\end{aligned}$$

Solving the above 3 equations we find

$$\begin{aligned}
a &= \frac{8}{3h^2}\\
b &= -\frac{4}{h^2}\\
c &= \frac{4}{3h^2}
\end{aligned}$$

Hence (1) becomes

$$\begin{aligned}
\tilde{f}''(x) &= af\left(x - \frac{h}{2}\right) + bf(x) + cf(x+h)\\
&= \frac{8}{3h^2}f\left(x - \frac{h}{2}\right) - \frac{4}{h^2}f(x) + \frac{4}{3h^2}f(x+h)
\end{aligned}$$

To examine the local truncation error, from (2), and using the solution we just found for a, b, c we find

$$\begin{aligned}
\tilde{f}''(x) &= f''(x) + f'''(x)\,h^3\left(\frac{-\left(\frac{8}{3h^2}\right)}{8\times 6} + \frac{\left(\frac{4}{3h^2}\right)}{6}\right) + f^{(4)}h^4\left(\frac{\left(\frac{8}{3h^2}\right)}{16\times 24} + \frac{\left(\frac{4}{3h^2}\right)}{24}\right) + O\left(h^5\right) \\
&= f''(x) + f'''(x)\,h^3\left(\frac{1}{6h^2}\right) + f^{(4)}h^4\left(\frac{1}{16h^2}\right) + O\left(h^5\right) \\
&= f''(x) + f'''(x)\left(\frac{h}{6}\right) + f^{(4)}h^2\left(\frac{1}{16}\right) + O\left(h^5\right)
\end{aligned}$$

We can truncate at either $f'''(x)$ or $f^{(4)}$. In the first case, we obtain

$$\tilde{f}''(x) = f''(x) + O(h)$$

Where $O(h) = \frac{f'''(x)}{6}h$, hence $p = 1$ in this case, and with the truncation error $\tau = \frac{f'''(x_j)}{6}h$ at each grid point.

In the second case, we obtain

$$\tilde{f}''(x) = f''(x) + \frac{f'''(x)}{6}h + O\left(h^2\right)$$

Where $O(h^2) = \frac{f^{(4)}}{16}h^2$ and $p = 2$ in this case, and with the truncation error $\tau = \frac{f^{(4)}(x_j)}{16}h^2$ at each grid point. We see that τ is smaller if we use $p = 2$ than $p = 1$.

The accuracy then depends on where we decide to truncate. For example, at $p = 1$, the error is dominated by $O(h)$, and at $p = 2$, it is $O(h^2)$.

4.2 part (b) Refinement study

Given $f(x) = \cos(2\pi x)$, first, let us find the accuracy of this scheme. The finite difference approximation formula found is

$$\tilde{f}''(x) = \frac{8}{3h^2}f\left(x - \frac{h}{2}\right) - \frac{4}{h^2}f(x) + \frac{4}{3h^2}f(x+h) \tag{1}$$

And the exact value is

$$\frac{d^2}{dx^2}\cos(2\pi x) = -4\pi^2\cos 2\pi x \tag{2}$$

To find the local error τ

$$\tau = \tilde{f}''(x) - f''(x)$$

Substitute $f(x) = \cos(2\pi x)$ in the RHS of (1) to find the approximation of the second derivative and subtract the exact result value of the second derivative from it.

Plug $f(x) = \cos(2\pi x)$ in RHS of (1) we obtain

$$\begin{aligned}
\tilde{f}''(x) &= \frac{8}{3h^2}\cos\left(2\pi\left(x - \frac{h}{2}\right)\right) - \frac{4}{h^2}\cos(2\pi x) + \frac{4}{3h^2}\cos(2\pi(x+h)) \\
&= \frac{8}{3h^2}\cos(2\pi x - \pi h) - \frac{4}{h^2}\cos(2\pi x) + \frac{4}{3h^2}\cos(2\pi x + 2\pi h)
\end{aligned}$$

Hence the local error τ is

$$\begin{aligned}
\tau &= \tilde{f}''(x) - f''(x) \\
&= \left[\frac{8}{3h^2}\cos(2\pi x - \pi h) - \frac{4}{h^2}\cos(2\pi x) + \frac{4}{3h^2}\cos(2\pi x + 2\pi h)\right] + 4\pi^2\cos 2\pi x
\end{aligned}$$

We notice that τ depends on h and x. At $x = 1$,

$$\begin{aligned}
\tau &= \left[\frac{8}{3h^2}\cos(2\pi - \pi h) - \frac{4}{h^2} + \frac{4}{3h^2}\cos(2\pi + 2\pi h)\right] + 4\pi^2 \\
&= \frac{4}{3h^2}\left(\cos(2\pi h) + 2\cos(\pi h) + 3h^2\pi^2 - 3\right)
\end{aligned}$$

In the following we plot local error τ as a function of h in linear scale and log scale. Here is the result.

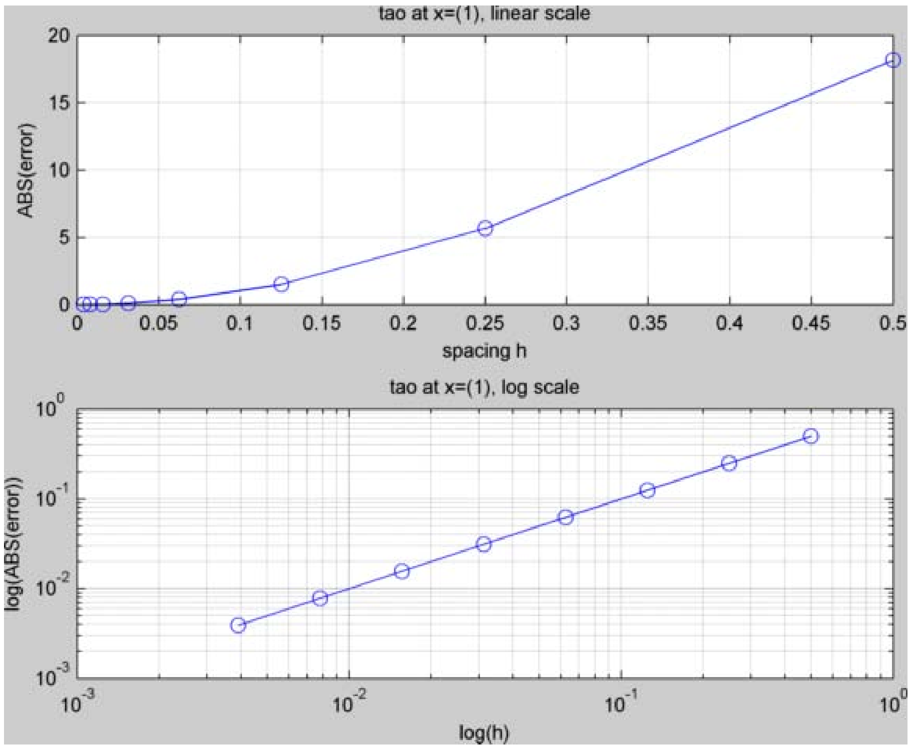


Figure 2: matlab HW1 partb

We notice that the log plot shows the slope $p = 2$ and not $p = 1$. This is because the $O(h)$ part turned out to be zero at $x = 1$ this is because $O(h) = \frac{f'''(x)}{6}h = \frac{(8\pi^3 \sin 2\pi x)}{6}h$ and this term is zero at $x = 1$, so the dominant error term became $O(h^2)$ which is $\frac{f^{(4)}(x_j)}{16}h^2 = \frac{16\pi^4 \cos 2\pi x}{16}h^2$ or $\pi^4 h^2$ or $O(h^2)$.

This is why we obtained $p = 2$ and not $p = 1$ at $x = 1$.

The following table show the ratio of the local error between each 2 successive halving of the spacing h. Each time h is halved, and the ratio of the error (absolute local error) is shown. We see for $x = 1$ that the ratio approaches 4. This indicates that $p = 2$.

```
EDU>> nma_HW1_partb()
h                   error               ratio
5.0000E-001       1.8145E+001       0.0000E+000
2.5000E-001       5.6483E+000       3.2125E+000
1.2500E-001       1.4936E+000       3.7816E+000
6.2500E-002       3.7872E-001       3.9439E+000
3.1250E-002       9.5014E-002       3.9859E+000
1.5625E-002       2.3775E-002       3.9965E+000
7.8125E-003       5.9449E-003       3.9991E+000
3.9063E-003       1.4863E-003       3.9998E+000
```

4.3 Part(c)

The refinement study in part (b) showed that the local error became smaller as h become smaller, and the error was $O(h^2)$ since $p = 2$ in the log plot.

But this is not a good test as it was done only for one point $x = 1$. We need to examine the approximation scheme at other points as well. The reason is the local error at an x location is

$$\tau = \left[\frac{8}{3h^2}\cos(2\pi x - \pi h) - \frac{4}{h^2}\cos(2\pi x) + \frac{4}{3h^2}\cos(2\pi x + 2\pi h)\right] + 4\pi^2 \cos 2\pi x$$

which can be seen to be a function of x and h. In (b) we found that at $x = 1$, $\tau = O\left(h^2\right)$ and this was because the dominant error term $O(h)$ happened to vanish at $x = 1$.

But if we examine τ at different point, say $x = 0.2$, then we will see that τ is $O\left(h\right)$ and $p = 1$.

Here is a plot of τ at $x = 0.2$ and at $x = 1$. Both showing what happens as h becomes smaller. We see that the at $x = 1$ the approximation was more accurate ($p = 2$) but at $x = 0.2$ the approximation was less accurate ($p = 1$). What we conclude from this, is that a single test is not sufficient for determine the accuracy for all points. More tests are needed at other points to have more confidence. To verify that at $x = 0.2$ we indeed have $p = 1$, we generate the error table as shown above, but for $x = 0.2$ this time.

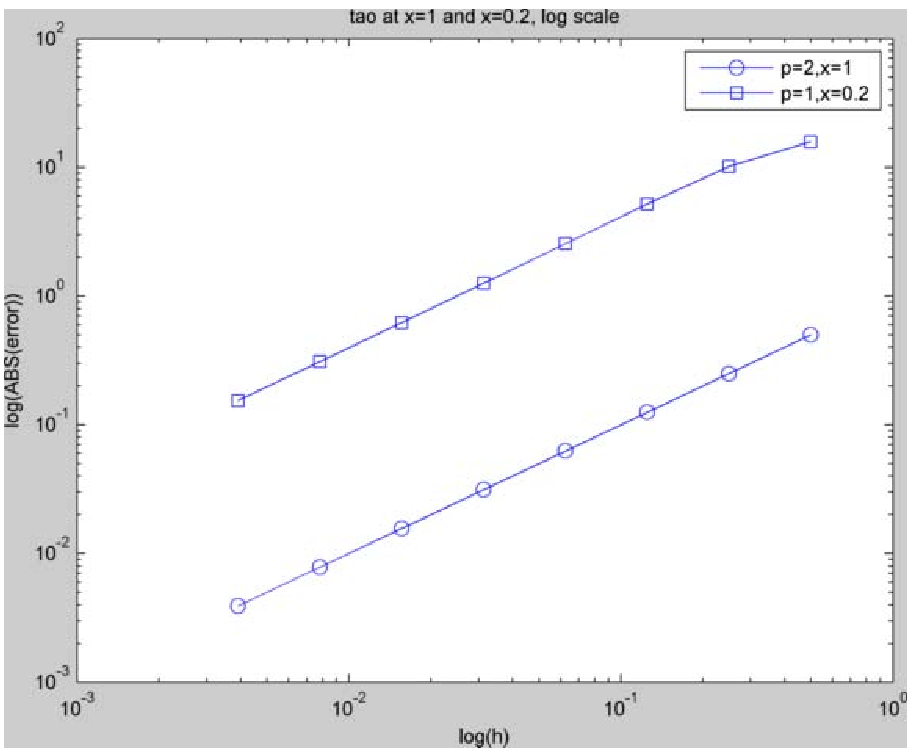


Figure 3: matlab HW1 partc

```
EDU>> nma_HW1_partc()
  h                   error               ratio
5.0000E-001         1.5752E+001         0.0000E+000
2.5000E-001         1.0149E+001         1.5520E+000
1.2500E-001         5.1898E+000         1.9557E+000
6.2500E-002         2.5508E+000         2.0345E+000
3.1250E-002         1.2551E+000         2.0324E+000
1.5625E-002         6.2133E-001         2.0200E+000
7.8125E-003         3.0897E-001         2.0110E+000
3.9063E-003         1.5404E-001         2.0057E+000
```

We see that the ratio becomes 2 this time, not 4 as we half the spacing each time. This mean $p = 1$. This means the accuracy of the formula used can depend on the location.

4.4 Part (d)

The points that we need to interpolate are $\left[\left[x - \frac{h}{2}, u\left(x - \frac{h}{2}\right)\right], [x, u(x)], [x + h, u(x + h)]\right]$ where $u = \cos(2\pi x)$

Since we require a quadratic polynomial, then we write

$$p(x) = a + bx + cx^2$$

Where $p(x)$ is the interpolant. Evaluate the above at each of the 3 points. Choose $x=1$, hence the points are

$$\begin{pmatrix} 1-\frac{h}{2}, u\left(1-\frac{h}{2}\right) \\ 1, u(x) \\ 1+h, u(1+h) \end{pmatrix}$$

Evaluate $p(x)$ at each of these points

$$\begin{aligned} p\left(1-\frac{h}{2}\right) &= \cos\left(2\pi\left(1-\frac{h}{2}\right)\right) = a+b\left(1-\frac{h}{2}\right)+c\left(1-\frac{h}{2}\right)^2 \\ p(1) &= \cos(2\pi) = a+b+c \\ p(1+h) &= \cos(2\pi(1+h)) = a+b(1+h)+c(1+h)^2 \end{aligned}$$

or

$$\begin{pmatrix} \left(1-\frac{h}{2}\right)^2 & \left(1-\frac{h}{2}\right) & 1 \\ 1 & 1 & 1 \\ (1+h)^2 & (1+h) & 1 \end{pmatrix} \begin{pmatrix} c \\ b \\ a \end{pmatrix} = \begin{pmatrix} \cos\left(2\pi\left(1-\frac{h}{2}\right)\right) \\ \cos(2\pi 1) \\ \cos(2\pi(1+h)) \end{pmatrix}$$
$$Av = b$$

Solving the above Vandermonde system, we obtain

$$\begin{aligned} a &= \frac{1}{3h^2}\left(4(1+h)\cos(h\pi)+(h-2)(3+3h-\cos(2\pi h))\right) \\ b &= \frac{-1}{3h^2}4\left((h-4)\cos(\pi h)-h-8\right)\sin^2\left(\frac{\pi h}{2}\right) \\ c &= \frac{2}{3h^2}\left(2\cos(\pi h)-3+\cos(2\pi h)\right) \end{aligned}$$

Hence

$$\begin{aligned} p(x) = &\left[\frac{1}{3h^2}\left(4(1+h)\cos(h\pi)+(h-2)(3+3h-\cos(2\pi h))\right)\right] \\ &-\left[\frac{1}{3h^2}4\left((h-4)\cos(\pi h)-h-8\right)\sin^2\left(\frac{\pi h}{2}\right)\right]x \\ &+\left[\frac{2}{3h^2}\left(2\cos(\pi h)-3+\cos(2\pi h)\right)\right]x^2 \end{aligned} \tag{1}$$

Recall, that we found, for $u=\cos(2\pi x)$, the finite difference formula was

$$\tilde{u}''(x) = \left[\frac{8}{3h^2}\cos(2\pi x-\pi h)-\frac{4}{h^2}\cos(2\pi x)+\frac{4}{3h^2}\cos(2\pi x+2\pi h)\right] \tag{2}$$

Take the second derivative of $p(x)$ shown in (1) above

$$p''(x) = \frac{4}{3h^2}\left(2\cos(\pi h)+\cos(2\pi h)-3\right) \tag{3}$$

But we notice that $\tilde{u}''(x)$ evaluated at $x=1$ is

$$\tilde{u}''(1) = \frac{4}{3h^2}\left(2\cos(\pi h)+\cos(2\pi h)-3\right)$$

which is the same as $p''(x)$.

Therefore, $p''(x)$ is the same as as the finite difference approximation evaluated at the central point of the 3 points, used to generate p.

In other words, given 3 points

$$\begin{pmatrix} x_0-\frac{h}{2}, u(x_0) \\ x_0, u(x_0) \\ x_0+h, u(x_0+h) \end{pmatrix}$$

Where $u(x)$ is some function (here it was $\cos(2\pi x)$), and we generate a quadratic interpolant polynomial $p(x)$ using the above 3 points, then $p''(x)$ will given the same value as the finite difference formula evaluated at x_0.

$$p''(x)|_{x=x_0} = \tilde{u}(x)|_{x=x_0}$$

For this to be valid, $p(x)$ must have been generated with the center point being x_0. If we pick another center point x_1, and therefore have the 3 points $x_1 - h/2, x_1, x_1 + h$, and then generate a polynomial $q(x)$ as above, then we will find

$$q''(x)|_{x=x_1} = \tilde{u}(x)|_{x=x_1}$$

This is illustrated by the following diagram

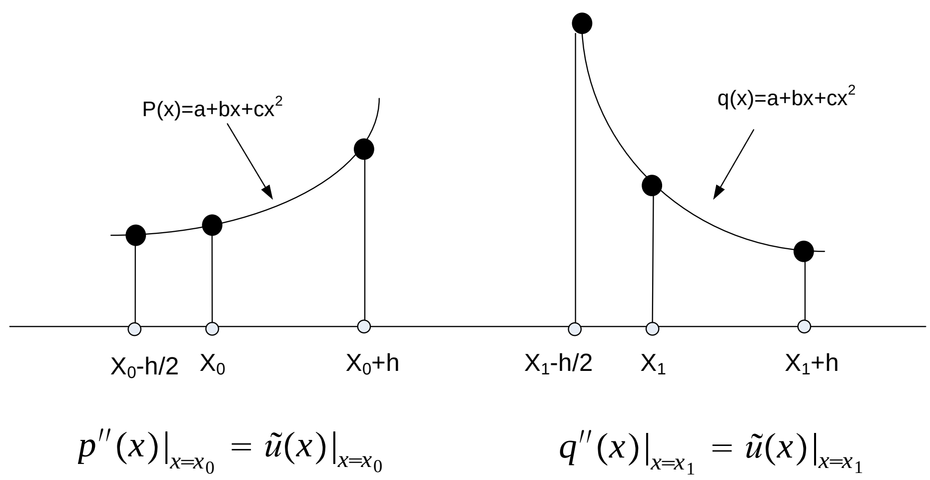


Figure 4: prob3 c

5 Appendix (Source code)

5.1 Matlab

```
%-- by Nasser M. Abbasi, Math 228A, UC Davis, Fall 2010
%-- implement part b, problem 3
function nma_HW1_partb()
%-- Generate h values to use, and define tao(h) function
N = 8;
pointAt=1;
data  = arrayfun( @(i) [1/(2^i) , local_error(1/(2^i),pointAt)] ,1:N, ...
                  'UniformOutput',false);
data = reshape(cell2mat(data),2,N)';

%-- plot the tao(h) in linear and log scale
set(0,'defaultaxesfontsize',8) ;
set(0,'defaulttextfontsize',8);

subplot(2,1,1);
plot(data(:,1),data(:,2),'-o'); grid on;
title('tao at x=(1), linear scale');
xlabel('spacing h'); ylabel('ABS(error)');

subplot(2,1,2);
loglog(data(:,1),data(:,1),'-o'); grid on;
title('tao at x=(1), log scale');
xlabel('log(h)'); ylabel('log(ABS(error))');
export_fig matlab_HW1_partb.png

%-- now generate the error table, find ratio first
error_ratio = zeros(N,1);
for i=2:N
```

```
    error_ratio(i) = data((i-1),2)/data(i,2);
end

%-- print table
fprintf('h\t\t\t\t error\t\t\t\t ratio\n');
for i=1:N
   fprintf('%6.4E\t\t%6.4E\t\t%6.4E\n',data(i,1),data(i,2),error_ratio(i));
end

end

function tao=local_error(h,x)
tao=8/(3*h^2)*cos(2*pi*(x-h/2))-4/h^2*cos(2*pi*x)+4/(3*h^2)*...
    cos(2*pi*(x+h))+(2*pi)^2*cos(2*pi*x);
end
```

```
%-- by Nasser M. Abbasi, Math 228A, UC Davis, Fall 2010
%-- implement part c, problem 3
function nma_HW1_partc()

%-- Generate h values to use, and define tao(h) function
%-- plot the tao(h) in linear and log scale
set(0,'defaultaxesfontsize',8) ;
set(0,'defaulttextfontsize',8);

%build data, x-axis is spacing h, y-axis is error
N = 8;
pointAt=1.0;
data  = arrayfun( @(i) [1/(2^i) , local_error(1/(2^i),pointAt)] ,1:N, ...
                 'UniformOutput',false);
data = reshape(cell2mat(data),2,N)';

loglog(data(:,1),data(:,1),'-o'); grid off;
title('tao at x=1 and x=0.2, log scale');
xlabel('log(h)'); ylabel('log(ABS(error))');
hold on;

pointAt=0.2;
data  = arrayfun( @(i) [1/(2^i) , local_error(1/(2^i),pointAt)] ,1:N,...
                 'UniformOutput',false);
data = reshape(cell2mat(data),2,N)';
loglog(data(:,1),data(:,2),'-s');
legend('p=2,x=1','p=1,x=0.2');

export_fig matlab_HW1_partc.png

%-- now generate the error table, find ratio first
error_ratio = zeros(N,1);
for i=2:N
    error_ratio(i) = data((i-1),2)/data(i,2);
end

%-- print table
fprintf('h\t\t\t\t error\t\t\t\t ratio\n');
for i=1:N
   fprintf('%6.4E\t\t%6.4E\t\t%6.4E\n',data(i,1),data(i,2),error_ratio(i));
end

end

function tao=local_error(h,x)
tao=8/(3*h^2)*cos(2*pi*(x-h/2))-4/h^2*cos(2*pi*x)+4/(3*h^2)*...
    cos(2*pi*(x+h))+(2*pi)^2*cos(2*pi*x);
end
```

5.2 Mathematica

HW 1, problem 3, computational part. math 228A UC davis fall 2010

Nasser M. Abbasi

This is the code used to generate the plots and tables used in HW1

define local error function

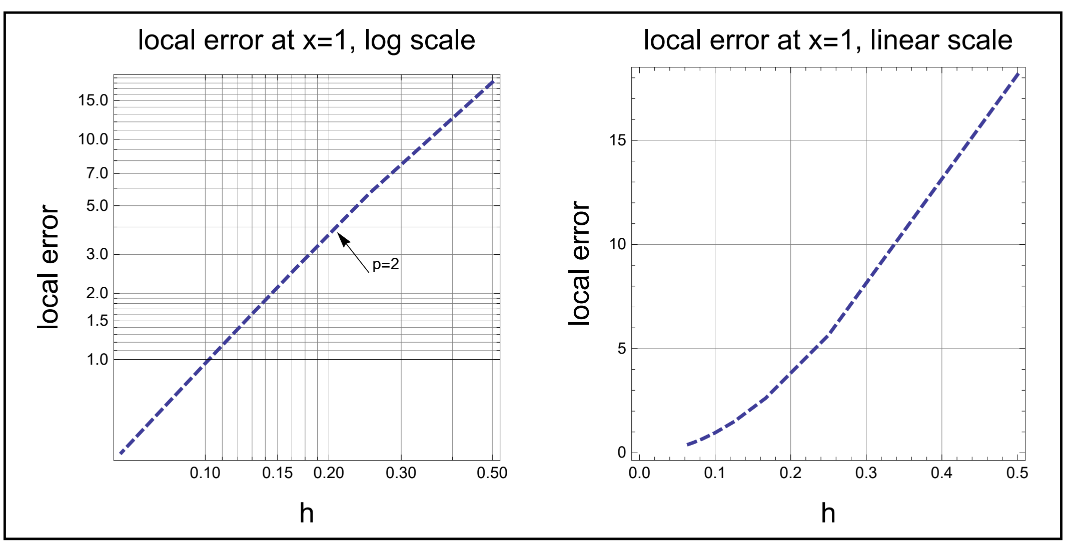
```
localError[h_, x_] :=
  Module[{}, 8/(3 h^2) Cos[2 π x - π h] - 4/h^2 Cos[2 π x] + 4/(3 h^2) Cos[2 π x + 2 π h]] + (2 π)^2 Cos[2 π x];
```

define a function to make the plots

```
makePlot[x_, s_, title_, xlabel_, ylabel_, f_] := Module[{data, n = 8},
  data = Table[{1/(2^i), Abs@localError[1/(2^i), x] }, {i, 1, n}];
  f[data, Joined → True, AxesOrigin → {0, 0},
   GridLines → Automatic, AspectRatio → 1, Frame → True, PlotRange → All,
   FrameLabel → {{ylabel, None}, {xlabel, title}}, PlotStyle → s, ImageSize → Full]
 ]
```

make plot for problem 3, part b

```
title = Style["local error at x=1, log scale", 16];
xlabel = Style["h", 16]; ylabel = Style["local error", 16];
p1 = makePlot[1, {Thick, Dashed}, title, xlabel, ylabel, ListLogLogPlot];
title = Style["local error at x=1, linear scale", 16];
p2 = makePlot[1, {Thick, Dashed}, title, xlabel, ylabel, ListPlot];
Framed[Grid[{{p1, p2}}], ImageSize → {600, 300}]
```



Generate error table, problem 3, part b

```
n = 14;
x = 1;
data = Table[{1/(2^i), Abs@localError[1/(2^i), x]}, {i, 1, n}];
data = Table[{data[[i, 1]], data[[i, 2]], If[i == 1, 0, data[[i - 1, 2]]/data[[i, 2]]]}, {i, 1, n}];
t = TableForm[N[data, $MachinePrecision], TableHeadings →
    {None, {"h", "local error τ", "ratio"}}, TableSpacing → {1, 6}, TableAlignments → Left];
Labeled[Framed@ScientificForm[t, {8, 6}, NumberFormat → (Row[{#1, "e", #3}] &),
   NumberPadding → {"", "0"}], Style["local error as function of h at x=1", 14], Top]
```

local error as function of h at x=1

h	local error τ	ratio
5.000000e-1	1.814508e1	0.000000e
2.500000e-1	5.648307e	3.212482e
1.250000e-1	1.493636e	3.781581e
6.250000e-2	3.787161e-1	3.943947e
3.125000e-2	9.501408e-2	3.985895e
1.562500e-2	2.377451e-2	3.996468e
7.812500e-3	5.944941e-3	3.999117e
3.906250e-3	1.486317e-3	3.999779e
1.953125e-3	3.715845e-4	3.999945e
9.765625e-4	9.289644e-5	3.999986e
4.882812e-4	2.322413e-5	3.999997e
2.441406e-4	5.806034e-6	3.999999e
1.220703e-4	1.451509e-6	4.000000e
6.103516e-5	3.628771e-7	4.000000e

Generate table for problem 3, part (c)

```
n = 14;
x = 0.2;
data = Table[{1/(2^i), Abs@localError[1/(2^i), x]}, {i, 1, n}];
data = Table[{data[[i, 1]], data[[i, 2]], If[i == 1, 0, data[[i - 1, 2]]/data[[i, 2]]]}, {i, 1, n}];
t = TableForm[N[data, $MachinePrecision], TableHeadings →
    {None, {"h", "local error τ", "ratio"}}, TableSpacing → {1, 6}, TableAlignments → Left];
Labeled[Framed@ScientificForm[t, {8, 6}, NumberFormat → (Row[{#1, "e", #3}] &),
   NumberPadding → {"", "0"}], Style["local error as function of h at x=0.2", 14], Top]
```

local error as function of h at x=0.2

h	local error τ	ratio
5.000000e-1	1.575174e1	0.000000e
2.500000e-1	1.014949e1	1.551974e
1.250000e-1	5.189762e	1.955675e
6.250000e-2	2.550829e	2.034539e
3.125000e-2	1.255100e	2.032371e
1.562500e-2	6.213251e-1	2.020037e
7.812500e-3	3.089650e-1	2.010989e
3.906250e-3	1.540406e-1	2.005737e
1.953125e-3	7.690765e-2	2.002930e
9.765625e-4	3.842539e-2	2.001480e
4.882812e-4	1.920555e-2	2.000744e
2.441406e-4	9.600990e-3	2.000372e
1.220703e-4	4.800048e-3	2.000186e
6.103516e-5	2.399851e-3	2.000144e

Generate plot for part (C)

```
title = Style["local error at different x locations, log scale", 16];
xlabel = Style["h", 16]; ylabel = Style["local error", 16];
p1 = makePlot[1, {Thick, Dashed}, title, xlabel, ylabel, ListLogLogPlot];
p2 = makePlot[0.2, {Thick, Black}, title, xlabel, ylabel, ListLogLogPlot];
Show[{p1, p2}, ImageSize → 500]
```