Kyle Kondrat

MATH 572

HW#1

02/02/2016

Problem #1

a)

Starting from the Taylor expansions of $u(x)$, $(x+h)$, $u(x-h)$, $u(x+2h)$, $u(x-2h)$; for $u(x)$ the expansion is $u(x)$, for $u(x+h)$ about x

$$u(x+h) = u(x) + hu^{'}(x) + \frac{h^2u^{''}(x)}{2} + \frac{h^3u^{'''}(x)}{6} + \frac{h^4u^{''''}(x)}{24}, \tag{1}$$

for $u(x-h)$ about x

$$u(x+h) = u(x) - hu^{'}(x) + \frac{h^2u^{''}(x)}{2} - \frac{h^3u^{'''}(x)}{6} + \frac{h^4u^{''''}(x)}{24}, \tag{2}$$

for $u(x+2h)$ about x

$$u(x+2h) = u(x) + 2hu^{'}(x) + \frac{4h^2u^{''}(x)}{2} + \frac{8h^3u^{'''}(x)}{6} + \frac{16h^4u^{''''}(x)}{24}, \tag{3}$$

and for $u(x-2h)$ about x

$$u(x+2h) = u(x) - 2hu^{'}(x) + \frac{4h^2u^{''}(x)}{2} - \frac{8h^3u^{'''}(x)}{6} + \frac{16h^4u^{''''}(x)}{24}. \tag{4}$$

$$\begin{bmatrix} u(x-2h) & u(x-h) & u(x) & u(x+h) & u(x+2h) \end{bmatrix}$$

$$\begin{bmatrix} u(x) & u(x) & u(x) & u(x) & u(x) \\ -2hu^{'}(x) & -hu^{'}(x) & 0 & hu^{'}(x) & 2hu^{'}(x) \\ \frac{4h^2u^{''}(x)}{2} & \frac{h^2u^{''}(x)}{2} & 0 & \frac{h^2u^{''}(x)}{2} & \frac{4h^2u^{''}(x)}{2} \\ -\frac{8h^3u^{'''}(x)}{6} & -\frac{h^3u^{'''}(x)}{6} & 0 & \frac{h^3u^{'''}(x)}{6} & \frac{8h^3u^{'''}(x)}{6} \\ \frac{16h^4u^{''''}(x)}{24} & \frac{h^4u^{''''}(x)}{24} & 0 & \frac{h^4u^{''''}(x)}{24} & \frac{16h^4u^{''''}(x)}{24} \end{bmatrix}, \tag{5}$$

Now we can set up the 5×5 Vandermonde Matrix from the coefficients in the matrix above

$$V = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ -2h & -1h & 0 & 1h & 2h \\ \frac{4}{2}h^2 & \frac{1}{2}h^2 & 0 & \frac{1}{2}h^2 & \frac{4}{2}h^2 \\ -\frac{8}{6}h^3 & -\frac{1}{6}h^3 & 0 & \frac{1}{6}h^3 & \frac{8}{6}h^3 \\ \frac{16}{24}h^4 & \frac{1}{24}h^4 & 0 & \frac{1}{24}h^4 & \frac{16}{24}h^4 \end{bmatrix} \tag{6}$$

and from Eqn. 6 we can set up the Vandermonde system to solve for the coefficients

$$V \cdot C = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ -2h & -1h & 0 & 1h & 2h \\ \frac{4}{2}h^2 & \frac{1}{2}h^2 & 0 & \frac{1}{2}h^2 & \frac{4}{2}h^2 \\ -\frac{8}{6}h^3 & -\frac{1}{6}h^3 & 0 & \frac{1}{6}h^3 & \frac{8}{6}h^3 \\ \frac{16}{24}h^4 & \frac{1}{24}h^4 & 0 & \frac{1}{24}h^4 & \frac{16}{24}h^4 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}. \tag{7}$$

From the Vandermonde system we get a system of equations to solve

$$\begin{cases} 0 &= c_1 + c_2 + c_3 + c_4 + c_5 \\ 0 &= -2c_1 - c_2 + c_4 + 2c_5 \\ 2h^2 &= 4c_1 + c_2 + c_4 + 4c_5 \\ 0 &= -8c_1 - c_2 + c_4 + 8c_5 \\ 0 &= 16c_1 + c_2 + c_5 + 16c_5 \end{cases} \tag{8}$$

Using a linear systems of equation solver, we see that the coefficients are

$$\begin{cases} c_1 &= -\frac{1}{12h^2} \\ c_2 &= \frac{4}{3h^2} \\ c_3 &= -\frac{5}{2h^2} \\ c_4 &= \frac{4}{3h^2} \\ c_5 &= -\frac{1}{12h^2} \end{cases} . \tag{9}$$

So our equation becomes

$$u^{''}(x) = -\frac{1}{12}u\left(x-2h\right) + \frac{4}{3}u\left(x-h\right) - \frac{5}{2}u\left(x\right) + \frac{4}{3}u\left(x+h\right) - \frac{1}{12}u\left(x+2h\right) + \mathcal{O}(h^4). \tag{10}$$

b)

The results from the code are:

```
The derivative u^(2) of u at x0 is approximated by


          1/h^2 * [

                     -8.333333333333333e-02 * u(x0-2*h) +

                      1.333333333333333e+00 * u(x0-1*h) +

                     -2.500000000000000e+00 * u(x0) +

                      1.333333333333334e+00 * u(x0+1*h) +

                     -8.333333333333333e-02 * u(x0+2*h) ]


For smooth u,

       Error = 0 * h^3*u^(5) + -0.0111111 * h^4*u^(6) + ...
```

, which verify the result found in part a.

c)

Using the finite difference method from part a), $u^{''}(1)$ was approximated for $u(x) = \sin(x)$ with the values in the array hvals=logspace$(-1, -2, 13)$. The following output of the table for the h values, numerical error, and predicted error is:

h	D2error	Predicted Error
1.0000e-01	6.4431e-05	6.4661e-05
5.6234e-02	6.4588e-06	6.4661e-06
3.1623e-02	6.4638e-07	6.4661e-07
1.7783e-02	6.4653e-08	6.4661e-08
1.0000e-02	6.4608e-09	6.4661e-09
5.6234e-03	6.3753e-10	6.4661e-10
3.1623e-03	5.2040e-11	6.4661e-11
1.7783e-03	3.9182e-11	6.4661e-12
1.0000e-03	5.0307e-10	6.4661e-13
5.6234e-04	4.8242e-10	6.4661e-14
3.1623e-04	4.3333e-09	6.4661e-15
1.7783e-04	9.9178e-09	6.4661e-16
1.0000e-04	3.3754e-08	6.4661e-17

The following figure is the log-log plot of the errors vs. h

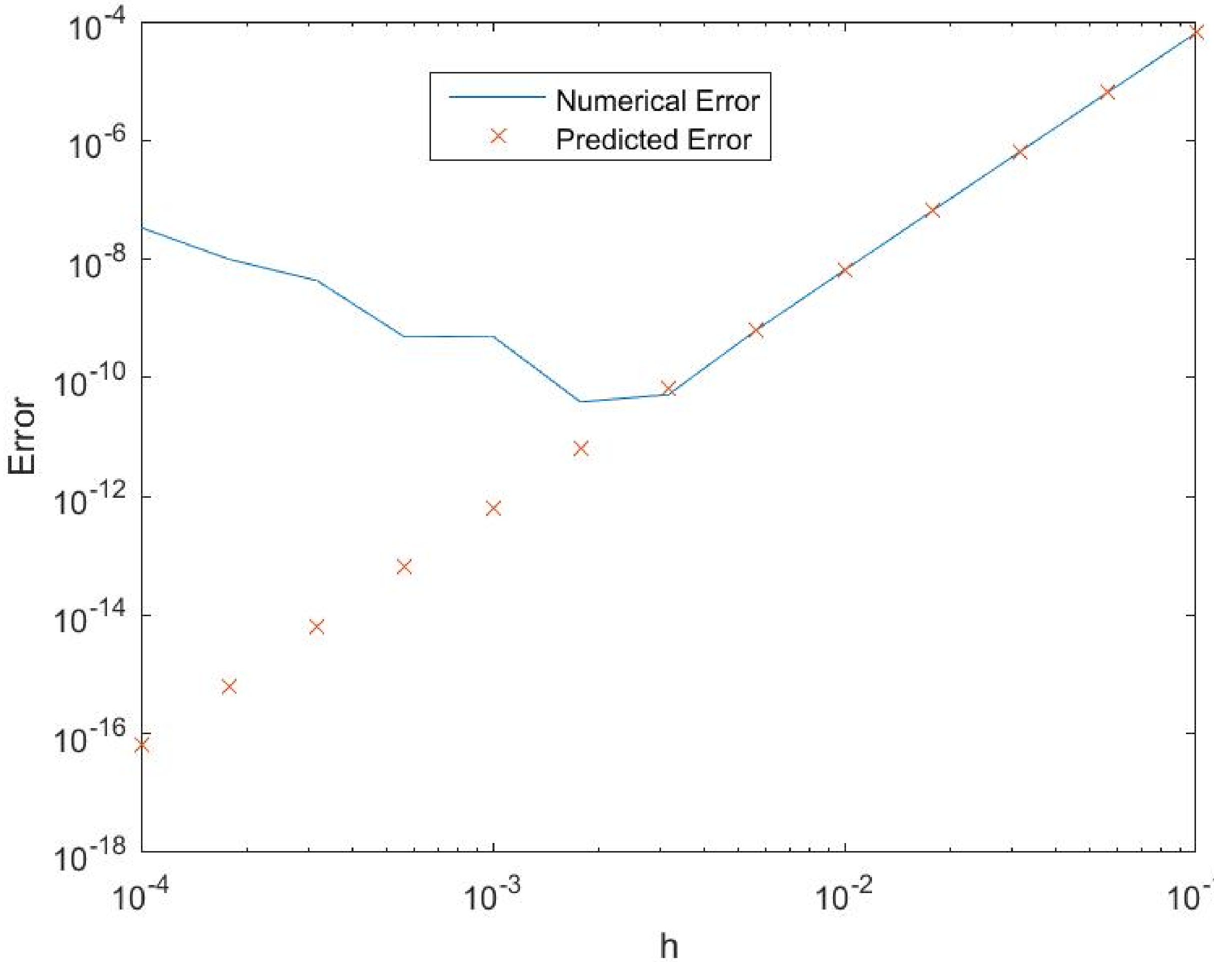


Figure 1: Problem1: log-log plot of errors vs. h

Problem #2

Using Lagrange method for an interpolating polynomial, the interpolating polynomial an be found to be

$$p(x) = u\left(x_0\right) \frac{\left(x-\left(x_0-h\right)\right)\left(x-\left(x_0-2h\right)\right)}{\left(x_0-\left(x_0-h\right)\right)\left(x_0-\left(x_0-2h\right)\right)} + u\left(x_0-h\right) \frac{\left(x-\left(x_0\right)\right)\left(x-\left(x_0-2h\right)\right)}{\left(x_0-\left(x_0\right)\right)\left(x_0-\left(x_0-2h\right)\right)} + u\left(x_0-2h\right) \frac{\left(x-\left(x_0-h\right)\right)\left(x-\left(x_0\right)\right)}{\left(x_0-\left(x_0-h\right)\right)\left(x_0-\left(x_0\right)\right)}. \tag{11}$$

Expanding this we get

$$p(x) = \frac{u\left(x_0\right)}{2h^2}\left[x^2-2xx_0+3xh-3x_0h+2h^2+x_0^2\right] - \frac{u\left(x_0-h\right)}{h^2}\left[x^2-2xx_0+2xh-2x_0h+x_0^2\right]$$

$$+\frac{u\left(x_0-2h\right)}{2h^2}\left[x^2-2xx_0+xh-x_0h+x_0^2\right], \tag{12}$$

and then taking the first derivative

$$p^{'}(x)=\frac{u\left(x_0\right)}{2h^2}\left[2x-2x_0+3h\right]-\frac{u\left(x_0-h\right)}{h^2}\left[2x-2x_0+2h\right]+\frac{u\left(x_0-2h\right)}{2h^2}\left[2x-2x_0+h\right], \tag{13}$$

then evaluate x at x_0

$$p^{'}(x_0)=\frac{u\left(x_0\right)}{2h^2}\left[2x_0-2x_0+3h\right]-\frac{u\left(x_0-h\right)}{h^2}\left[2x_0-2x_0+2h\right]+\frac{u\left(x_0-2h\right)}{2h^2}\left[2x_0-2x_0+h\right] \tag{14}$$

$$p^{'}(x_0)\frac{u\left(x_0\right)}{2h^2}\left[3h\right]-\frac{u\left(x_0-h\right)}{h^2}\left[2h\right]+\frac{u\left(x_0-2h\right)}{2h^2}\left[h\right]=\frac{1}{2h}\left[3u\left(x_0\right)-4u\left(x_0-h\right)+u\left(x_0-2h\right)\right]. \tag{15}$$

This gives us the same answer as shown in class. We get this from assuming the form of

$$D_2u(x_0)=au(x_0)+bu(x_0-h)+cu(x_0-2h), \tag{16}$$

where we then take the Taylor series expansions for the $u(x)$ to get

$$D_2u(x)=\left(a+b+c\right)u(x)-\left(b+2c\right)hu^{'}(x)+\frac{1}{2}\left(b+4c\right)h^2u^{''}(x)-\frac{1}{6}\left(b+8c\right)h^3u^{'''}(x)+..., \tag{17}$$

then we enfore that

$$\begin{cases} a+b+c & =0 \\ -\left(b+2c\right)h & =1 \\ \frac{1}{2}\left(b+4c\right)h^2 & =0 \end{cases}, \tag{18}$$

then solve for the coefficients to get

$$\begin{cases} a & =\frac{3}{2}h \\ b & =-\frac{2}{h}, \\ b & =\frac{1}{2}h \end{cases} \tag{19}$$

therefore

$$D_2u(x)=\frac{1}{2h}\left[3u(x)-4u(x-h)+u(x-2h)\right]. \tag{20}$$

Problem #3

a)

Modify the m-file bvp2.m so that it implements a Dirichlet boundary coundition at $x=a$ and a Neumann condition at $x=b$ and test the modified program.

For this the following changes were made to the bvp2.m program

```
ax = 0;
bx = 3;
sigma = -5; % Neumann boundary condition at ax
beta = 3; % Dirichlet boundary condtion at bx
```

was changed to

```
ax = 0;
bx = 3;
sigma = -5;
alpha = 22 - exp(3); % Dirichlet boundary condtion at ax
beta = 3;
beta_new = exp(3)-6; % Neumann boundary condition at bx
```

To change the boundary conditions. Then these other changes were made to the code for the boundary condition changes, with the original bvp2.m program having

```
% first row for Neumann BC at ax:
A(1,1:3) = fdcoeffF(1, x(1), x(1:3));
and
% last row for Dirichlet BC at bx:
A(m2,m:m2) = fdcoeffF(0,x(m2),x(m:m2));
and
% Right hand side:
F = f(x);
F(1) = sigma; k
F(m2) = beta;
```

was changed to

```
% first row for Dirichlet BC at ax:
A(1,1:3) = fdcoeffF(0, x(1), x(1:3));
and
% last row for Neumann BC at bx:
A(m2,m:m2) = fdcoeffF(1,x(m2),x(m:m2));
and
% Right hand side:
F = f(x);
F(1) = alpha;
F(m2) = beta_new;
```

The output for the original program bvp2.m is

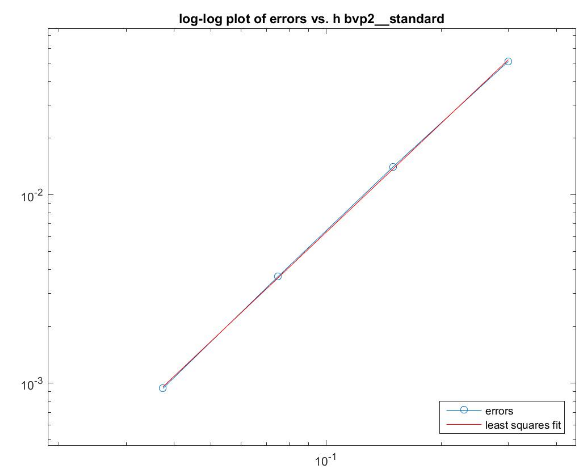
```
   h        error       ratio       observed order

 0.30000   5.08091e-02       NaN             NaN

 0.15000   1.40105e-02   3.62650         1.85858

 0.07500   3.66223e-03   3.82568         1.93572

 0.03750   9.34417e-04   3.91927         1.97058


Least squares fit gives E(h) = 0.525438 * h^1.92303
```

and the output for the modification is

h	error	ratio	observed order
0.30000	1.74886e+00	NaN	NaN
0.15000	4.80811e-01	3.63732	1.86288
0.07500	1.26086e-01	3.81336	1.93106
0.03750	3.22858e-02	3.90531	1.96544

```
Least squares fit gives E(h) = 18.0049 * h^1.92092
```

From this we can see that the accurcy has gone remained about the same $\mathcal{O}(h^{1.92303})$ for the original program to $\mathcal{O}(h^{1.92092})$ for the modified program. The plots for these two cases are



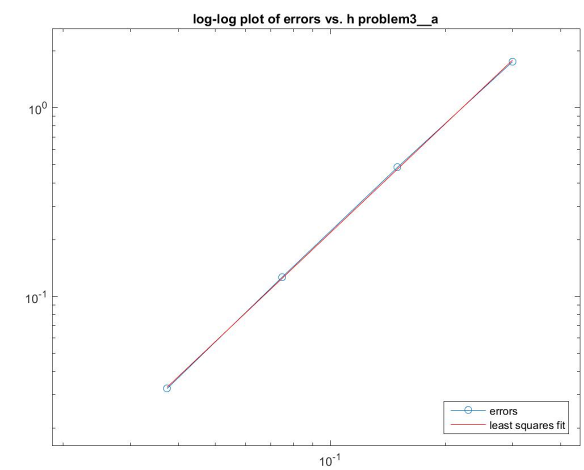


Figure 2: Problem 2 Part a. log-log plots (bvp2.m on right side and modification on left side)

b)

Make the same modification to the m-file bvp4.m, which implements a fourth order accurate method. Again test the modified program.

For this the following changes were made to the bvp2.m program

```
ax = 0;
bx = 3;
sigma = -5; % Neumann boundary condition at ax
beta = 3; % Dirichlet boundary condtion at bx
```

was changed to

```
ax = 0;
bx = 3;
sigma = -5;
alpha = 22 - exp(3); % Dirichlet boundary condtion at ax
beta = 3;
beta_new = exp(3)-6; % Neumann boundary condition at bx
```

To change the boundary conditions. Then these other changes were made to the code for the boundary condition changes, with the original bvp2.m program having

```
% first row for Neumann BC on u'(x(1))
A(1,1:5) = fdcoeffF(1, x(1), x(1:5));
and
% last row for Dirichlet BC on u(x(m+2))
A(m2,m-2:m2) = fdcoeffF(0,x(m2),x(m-2:m2));
and
% Right hand side:
F = f(x);
F(1) = sigma;
F(m2) = beta;
```

was changed to

```
% first row for Dirichlet BC on u(x(1))
A(1,1:5) = fdcoeffF(0, x(1), x(1:5));
and
% last row for Neumann BC on u'(x(m+2))
A(m2,m-2:m2) = fdcoeffF(1,x(m2),x(m-2:m2));
and
% Right hand side:
F = f(x);
F(1) = alpha;
F(m2) = beta_new;
```

The output for the original program bvp4.m is

```
   h         error        ratio        observed order

 0.30000   6.05673e-03        NaN             NaN

 0.15000   4.01813e-04   15.07352         3.91394

 0.07500   2.51825e-05   15.95603         3.99603

 0.03750   1.56256e-06   16.11621         4.01044


Least squares fit gives E(h) = 0.740188 * h^3.97573
```

and the output for the modification is

h	error	ratio	observed order
0.30000	7.22610e-02	NaN	NaN
0.15000	5.21206e-03	13.86421	3.79329
0.07500	3.47061e-04	15.01768	3.90859
0.03750	2.23260e-05	15.54515	3.95839

```
Least squares fit gives E(h) = 8.04933 * h^3.88894
```

From this we can see that the accurcy has gone down a litte from $\mathcal{O}(h^{3.97573})$ to $\mathcal{O}(h^{3.88894})$ for the modification. The plots for these two cases are

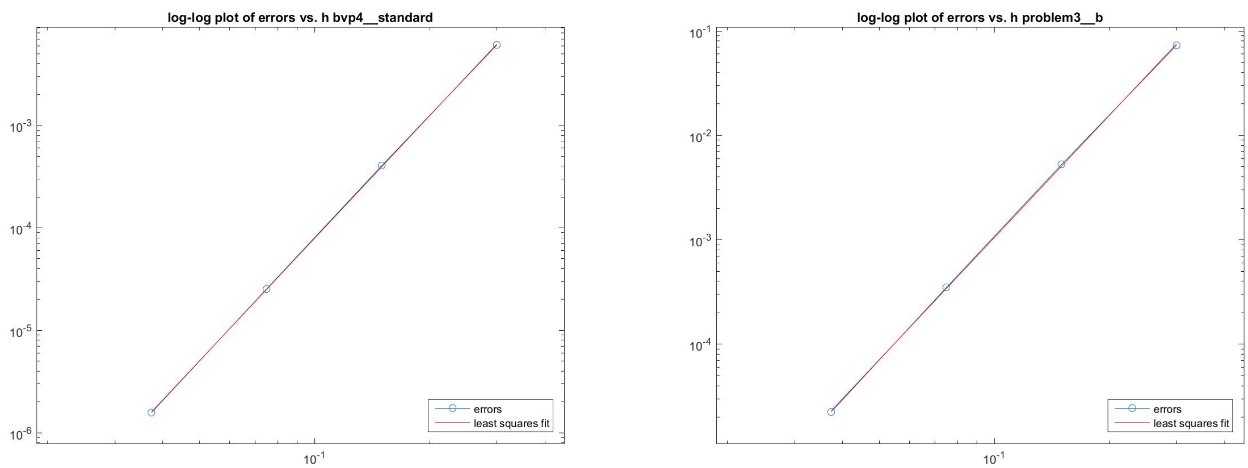


Figure 3: Problem 2 Part b. log-log plots (bvp4.m on right side and modification on left side)

Problem #4

a)

Setting gridchoice to be 'rtlayer'. The output we get from this is:

h	error	ratio	observed order
0.30000	5.67023e-01	NaN	NaN

```
0.15000    1.26585e-01    4.47940         2.16330

0.07500    2.98496e-02    4.24076         2.08432

0.03750    7.24452e-03    4.12030         2.04275
```

```
Least squares fit gives E(h) = 6.91323 * h^2.09554
```

From this we can see that we have second accurate accuracy with $\mathcal{O}(h^2)$. We can also see this from the output figure as well

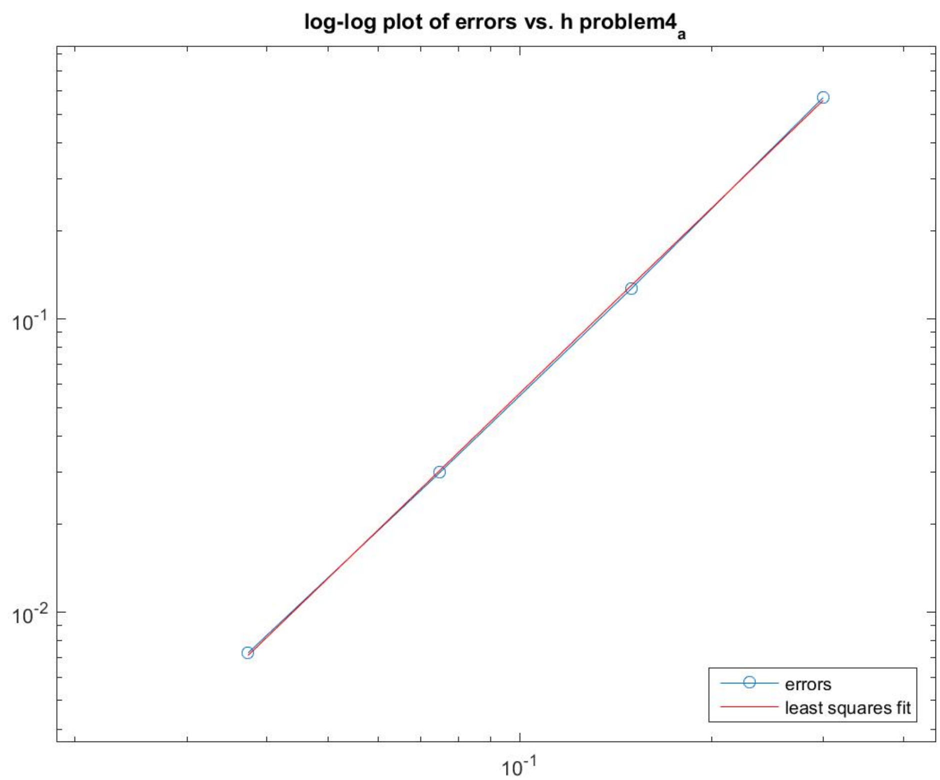


Figure 4: Problem 4 part b, log-log plot

b)

The local truncation error is defined by

$$\tau = c_{i-1}u\left(x_{i-1}\right) + c_iu\left(x_i\right) + c_{i+1}u\left(x_{i+1}\right) - u^{''}\left(x_i\right), \tag{21}$$

where are coefficients are

$$\begin{cases} c_{i-1} & = \frac{2}{h_{i-1}(h_{i-1}+h_i)} \\ c_i & = \frac{-2}{h_{i-1}h_i} \\ c_{i+1} & = \frac{2}{h_i(h_{i-1}+h_i)} \end{cases}, \tag{22}$$

where $h_{i-1} = x_i - x_{i-1}$ and $h_i = x_{i+1} - x_i$. We also have the following approximation of $x_i - x_{i-1} \approx hX'(x_i)$, therefore our coefficeints become

$$\begin{cases} c_{i-1} & = \frac{2}{h^2X'(x_i)(X'(x_i)+X'(x_{i+1}))} \\ c_i & = -\frac{2}{h^2X'(x_i)X'(x_{i+1})} \\ c_{i+1} & = \frac{2}{h^2X'(x_{i+1})(X'(x_i)+X'(x_{i+1}))} \end{cases}. \tag{23}$$

Now take Taylor expansions of $u(x_{i-1})$ and $u(x_{i+1})$,

$$u(x_{i-1}) = u(x_i) + u^{'}(x_i)(x_{i-1} - x_i) + \frac{u^{''}(x_i)(x_{i-1} - x_i)^2}{2} + \frac{u^{'''}(x_i)(x_{i-1} - x_i)^3}{6} + \frac{u^{''''}(x_i)(x_{i-1} - x_i)^4}{24}, \tag{24}$$

and

$$u(x_{i+1}) = u(x_i) + u^{'}(x_i)(x_{i+1} - x_i) + \frac{u^{''}(x_i)(x_{i+1} - x_i)^2}{2} + \frac{u^{'''}(x_i)(x_{i+1} - x_i)^3}{6} + \frac{u^{''''}(x_i)(x_{i+1} - x_i)^4}{24}. \tag{25}$$

Now subsitute in the approximation into the expansions to get

$$u(x_{i-1}) = u(x_i) - u^{'}(x_i)hX'(x_i) + \frac{u^{''}(x_i)h^2X'(x_i)^2}{2} - \frac{u^{'''}(x_i)h^3X'(x_i)^3}{6} + \frac{u^{''''}(x_i)h^4X'(x_i)^4}{24}, \tag{26}$$

and

$$u(x_{i-1}) = u(x_i) + u^{'}(x_i)hX'(x_{i+1}) + \frac{u^{''}(x_i)h^2X'(x_{i+1})^2}{2} + \frac{u^{'''}(x_i)h^3X'(x_{i+1})^3}{6} + \frac{u^{''''}(x_i)h^4X'(x_{i+1})^4}{24}. \tag{27}$$

Frist look at the first term $u(x_i)$ with its coefficients and we have

$$\begin{aligned} & u(x_i) \cdot \left[\frac{2}{h^2X'(x_i)\left(X'(x_i) + X'(x_{i+1})\right)} - \frac{2}{h^2X'(x_i)X'(x_{i+1})} + \frac{2}{h^2X'(x_{i+1})\left(X'(x_i) + X'(x_{i+1})\right)} \right] \\ & = \frac{u(x_i)}{h^2} \left[\frac{2X'(x_{i+1}) - 2\left(X'(x_i) + X'(x_{i+1})\right) + 2X'(x_i)}{X'(x_{i+1})\left(X'(x_i) + X'(x_{i+1})\right)X'(x_i)} \right] = 0, \end{aligned} \tag{28}$$

which eliminates the first term. Now look at the $u^{'}(x_i)$ term with the coefficients and we get

$$\frac{-2u^{'}(x_i)hX'(x_i)}{h^2X'(x_i)\left(X'(x_i) + X'(x_{i+1})\right)} + \frac{2u^{'}(x_i)hX'(x_{i+1})}{h^2X'(x_{i+1})\left(X'(x_i) + X'(x_{i+1})\right)} = \frac{-2u^{'}(x_i) + 2u^{'}(x_i)}{h\left(X'(x_i) + X'(x_{i+1})\right)} = 0, \tag{29}$$

which eliminates the 1st derivative term. Now look at the $u^{''}(x_i)$ terms with the coefficients and we get

$$\begin{aligned} & \frac{2u^{''}(x_i)h^2X'(x_i)^2}{h^2X'(x_i)\left(X'(x_i) + X'(x_{i+1})\right)2} + \frac{2u^{''}(x_i)h^2X'(x_{i+1})^2}{h^2X'(x_{i+1})\left(X'(x_i) + X'(x_{i+1})\right)2} - u^{''}(x_i) \\ & = \frac{u^{''}(x_i)X'(x_i) + u^{''}(x_i)X'(x_{i+1})}{\left(X'(x_i) + X'(x_{i+1})\right)} - u^{''}(x_i) = 0. \end{aligned} \tag{30}$$

Now our local truncation error is

$$\tau = \frac{2}{h^2X'(x_i)\left(X'(x_i)+X'(x_{i+1})\right)}\cdot\left[-\frac{u^{'''}(x_i)h^3X'(x_i)^3}{6}+\frac{u^{''''}(x_i)h^4X'(x_i)^4}{24}\right]$$
$$+\frac{2}{h^2X'(x_{i+1})\left(X'(x_i)+X'(x_{i+1})\right)}\cdot\left[\frac{u^{'''}(x_i)h^3X'(x_{i+1})^3}{6}+\frac{u^{''''}(x_i)h^4X'(x_{i+1})^4}{24}\right]. \tag{31}$$

Collect the $u^{'''}(x_i)$ terms to show the cancelation

$$-\frac{2u^{'''}(x_i)h^3X'(x_i)^3}{h^2X'(x_i)\left(X'(x_i)+X'(x_{i+1})\right)6}+\frac{2u^{'''}(x_i)h^3X'(x_{i+1})^3}{h^2X'(x_{i+1})\left(X'(x_i)+X'(x_{i+1})\right)6}=\frac{hu^{'''}(x_i)\left[X'(x_{i+1})^2-X'(x_i)^2\right]}{3\left(X'(x_i)+X'(x_{i+1})\right)}$$
$$\frac{1}{3}hu^{'''}(x_i)\frac{\left[X'(x_{i+1})+X'(x_i)\right]}{X'(x_{i+1})+X'(x_i)}\cdot\left[X'(x_{i+1})-X'(x_i)\right]=\frac{1}{3}hu^{'''}(x_i)\left[X'(x_{i+1})-X'(x_i)\right], \tag{32}$$

and if this is a smooth enough then $X'(x_i)\approx X'(x_{i+1})$, thefore the $u^{'''}(x_i)$, can be eliminated, and we are left with

$$\tau = \frac{2}{h^2X'(x_i)\left(X'(x_i)+X'(x_{i+1})\right)}\cdot\left[\frac{u^{''''}(x_i)h^4X'(x_i)^4}{24}\right]+\frac{2}{h^2X'(x_{i+1})\left(X'(x_i)+X'(x_{i+1})\right)}\cdot\left[\frac{u^{''''}(x_i)h^4X'(x_{i+1})^4}{24}\right]$$
$$=h^2\frac{u^{''''}(x_i)\left[X'(x_i)^3+X'(x_{i+1})^3\right]}{12\left(X'(x_i)+X'(x_{i+1})\right)}, \tag{33}$$

which is $\mathcal{O}(h^2)$, for the local truncation error.

c)

For this problem the gridchioce was set to 'random' and the mvals=round(logspace$(1,3,50)$); to do 50 tests for values between 10 and 1000. The output from this is:

```
   h         error        ratio       observed order

0.30000   1.03455e-01       NaN              NaN

0.27273   3.09755e+00   0.03340        -35.66495

0.25000   1.31142e+00   2.36198          9.87804

0.23077   1.88213e+00   0.69677         -4.51376

0.20000   4.07625e-01   4.61731         10.69044

0.18750   2.86107e-01   1.42473          5.48483

0.16667   1.33476e-01   2.14351          6.47330
```

0.15789	2.16270e-01	0.61717	-8.92604
0.14286	6.25518e-02	3.45745	12.39497
0.13043	1.31840e-01	0.47445	-8.19587
0.11538	8.30575e-02	1.58733	3.76872
0.10714	7.96189e-02	1.04319	0.57055
0.09677	2.85435e-02	2.78939	10.07855
0.08824	1.38157e-01	0.20660	-17.07161
0.08108	3.80493e-01	0.36310	-11.98096
0.07317	8.65950e-03	43.93943	36.85006
0.06667	1.27935e-01	0.06769	-28.92740
0.06122	1.53809e-02	8.31778	24.87612
0.05556	2.27839e-02	0.67508	-4.04398
0.05000	5.50789e-02	0.41366	-8.37803
0.04545	9.33110e-03	5.90273	18.62775
0.04167	1.73652e-02	0.53734	-7.13835
0.03797	1.87520e-02	0.92605	-0.82805
0.03448	7.78873e-03	2.40758	9.10863
0.03158	1.21303e-02	0.64209	-5.03623
0.02857	5.25001e-03	2.31053	8.36781
0.02609	1.07162e-02	0.48991	-7.84342
0.02381	2.30405e-02	0.46510	-8.37979
0.02158	3.04564e-03	7.56507	20.60800
0.01961	1.55053e-03	1.96426	7.03510
0.01786	6.38858e-03	0.24270	-15.13929
0.01630	1.64307e-03	3.88821	14.92713

0.01485	2.52427e-03	0.65091	-4.60066
0.01351	4.24860e-03	0.59414	-5.51465
0.01230	1.22618e-03	3.46492	13.15143
0.01119	1.85625e-03	0.66057	-4.41979
0.01017	2.85995e-04	6.49051	19.48508
0.00926	3.09968e-04	0.92266	-0.85844
0.00843	3.92073e-04	0.79059	-2.49482
0.00767	1.66909e-03	0.23490	-15.44719
0.00699	3.84353e-04	4.34261	15.83272
0.00637	5.07775e-04	0.75694	-2.98152
0.00579	3.19535e-04	1.58910	4.86948
0.00527	2.64155e-04	1.20965	2.02683
0.00480	1.97289e-04	1.33892	3.10921
0.00437	1.73438e-04	1.13752	1.36233
0.00398	1.83517e-04	0.94508	-0.60704
0.00362	7.64580e-05	2.40024	9.23324
0.00330	5.24937e-05	1.45652	4.03380
0.00300	8.01179e-05	0.65521	-4.48313

```
Least squares fit gives E(h) = 9.43204 * h^2.01101
```

, which we can see that the average accuracy is observed to be around $\mathcal{O}(h^2)$. The plot from these conditions is

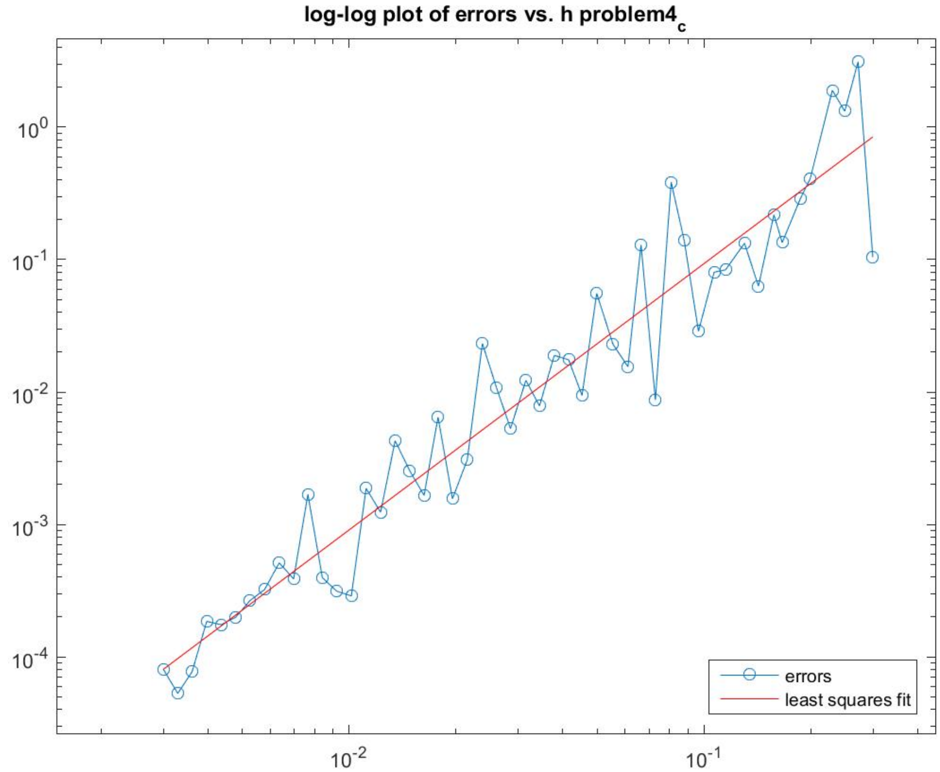


Figure 5: Problem 4 part c, log-log plot