NENG 685 Fall 2017
HW 1 Due Oct. 11, 2017

Name: Kevin Choe

On homework:

- If you work with anyone else, document what you worked on together.
- Show your work.
- Always clearly label plots (axis labels, a title, and a legend if applicable).
- Homework should be done "by hand" (i.e. not with a numerical program such as MATLAB, Python, or Wolfram Alpha) unless otherwise specified. You may use a numerical program to check your work.
- If you use a numerical program to solve a problem, submit the associated code, input, and output (email submission is fine).

Do not write in the table to the right.

Problem	Points	Score
1	20	
2	30	
3	15	
4	30	
Total:	95	

1. (20 points) Using four points on the interval $[x_0, x_3]$, do the following:

   (a) (4 points) Construct all of the Lagrange polynomials $L_j(x)$ that correspond to the points x_0, x_1, x_2, and x_3 by hand.

   (b) (4 points) Use these Lagrange polynomials to construct the interpolating polynomial, $P_3(x)$, that interpolates the function $f(x)$ at the points x_0, x_1, x_2, and x_3 by hand.

   (c) (8 points) Using the $P_3(x)$ you derived, create an interpolant for

   $$f(x) = \sin(\frac{\pi}{2}x) + \frac{x^2}{4}$$

   over $[x_0, x_3]$ with $x_0 = 0, x_1 = 2, x_2 = 3$, and $x_3 = 4$. You may do this using something like Python or MATLAB, but *write your own functions rather than using the built in ones.*

   Plot the actual function and your interpolant using 100 equally spaced points for x between -0.5 and 4.5.

   (d) (4 points) Repeat what you did in part c but instead use $x_0 = 0, x_1 = 1, x_2 = 2.5$, and $x_3 = 4$.

   Discuss the differences in how well the function is interpolated using the different point sets.

2. (15 points) Using the interpolant $P_3(x)$ derived in question 1:
   (a) (3 points) Write the general expression for the error term, $err(x) = |f(x) - P_3(x)|$.
   (b) (4 points) Given
   $$f(x) = \sin(\frac{\pi}{2}x) + \frac{x^2}{4} \,,$$
   use information about the function to bound the error expression.
   (c) (8 points) Use the values $x_0 = 0, x_1 = 2, x_2 = 3$, and $x_3 = 4$ to get the upper bound of $err(x)$ over this interval. That is, insert the points into the expression from part b, find an expression for x that maximizes error, and then find the x that gives the maximum. Present one final number. You may use a mathematical package to assist you in solving for x.

3. (15 points) We have the following data:
   $$x = [1, 2, 3, 4, 5, 6, 7] \,,$$
   $$f(x) = [1, 4, 10, 12, 5, 4, 0] \,.$$
   (a) (10 points) Using *built in* Python or MATLAB functions, interpolate this data using
   - Piecewise linear interpolation
   - Lagrange polynomial interpolation
   - Spline interpolation

   Create a subplot for each of your interpolants over $[0.75, 6.25]$ using a fine mesh spacing, e.g. 0.05 (note that to use scipy's piecewise linear polynomial interpolation you will need to restrict the range to the exact endpoints, 1.0 and 6.0). Include the data points on the interpolation plots.
   (b) (5 points) Briefly discuss the differences between the resulting interpolations.

4. (30 points) The errors generated by a numerical method on a test problem with various grid resolutions have been recorded in the following table:

Grid Spacing (h)	Error (E)
5.00000e-02	1.036126e-01
2.50000e-02	3.333834e-02
1.25000e-02	1.375409e-02
6.25000e-03	4.177237e-03
3.12500e-03	1.103962e-03
1.56250e-03	2.824698e-04
7.81250e-04	7.185644e-05
3.90625e-04	1.813937e-05

For this numerical method, the error should be of the form

$$E = kh^p$$

(a) (3 points) Write this problem as a linear system $\mathbf{A}\vec{x} = \vec{b}$, where $x = \begin{pmatrix} \ln(k) \\ p \end{pmatrix}$ is the vector of unknowns.

(b) (5 points) Derive the normal equations for this over-determined system: write the matrices in $\mathbf{A}\vec{x} = \vec{b}$ form, where you include formulas / values for each entry.

(c) (9 points) Solve, using the program/language of your choice, the normal equations to obtain a least squares estimate to the parameters k and p.

(d) (6 points) Solve for the parameters k and p using SciPy's CurveFit function

(e) (7 points) Make a log-log and a lin-lin plot that displays both the input data and the function $E = kh^p$. Comment on the differences between the two approximations. (Checkout Python's matplotlib.pyplt.loglog command.)

BONUS (5 points): submit your code by providing read/clone access to an online version control repository where your code is stored (e.g. github or bitbucket). If you don't know what that means and want to learn about it, come talk to me or check out resources here: `http://software-carpentry.org/lessons.html` For Windows, you want to setup the Git GUI. NOTE: You will not be able to do this from AFIT's systems.

1(a) construct Lagrange polynomials $L_i(x)$ that correspond to the points $x_0 x_1, x_2, x_3$ by hand.

$$L_3(x) = \prod_{\substack{i=0 \\ i\neq k}}^{n} \frac{(x-x_2)}{(x_k-x_2)} \qquad n=3 \text{ in this case}$$

$$i=0,\ L_0(x) = \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)}$$

$$i=1,\ L_1(x) = \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)}$$

$$i=2,\ L_2(x) = \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}$$

$$i=3,\ L_3(x) = \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$$

(b) use L.P. to construct interpolating polynomials, $P_3(x)$ by hand.

$$P_3(x) = \sum_{k=0}^{3} f(x_k) L_k(x)$$

$$= f(x_0)\cdot L_0(x) + f(x_1)\cdot L_1(x) + f(x_2)\cdot L_2(x) + f(x_3)\cdot L_3(x)$$

(c) use $P_3(x)$ derived to create interpolant for

$$f(x) = \sin\left(\frac{\pi}{2}x\right) + \frac{x^2}{4}$$

I used pyton for this problem; solution can be found in Github under "GradFall17 NENG 685" by "dabotop2"

(d) repeat (c) with different x values and discuss the differences in how well the function is interpolating

- you can find plots for (c) and (d) for comparison and discussion statement in "HW1_1_d.py" in Github.

2. (a) write the general expression for the error term.

$$\text{error}(x) = |f(x) - P_3(x)| = \frac{f^{4}\xi}{4!}(x-x_0)(x-x_1)(x-x_2)(x-x_3)$$

(b) Given $f(x) = \sin\left(\frac{\pi}{2}x\right) + \frac{x^2}{4}$, use information about the function to bound the error expression.

$$\frac{d^4}{dx^4}(f(x)) = \frac{d^4}{dx^4}\left(\sin\left(\frac{\pi x}{2}\right) + \frac{x^2}{4}\right)$$

$$= \frac{d^3}{dx^3}\left(\frac{d}{dx}\left(\sin\left(\frac{\pi x}{2}\right) + \frac{x^2}{4}\right)\right) = \frac{d^2}{dx^2}\left(\frac{d}{dx}\left(\frac{\pi\cos\left(\frac{\pi x}{2}\right) + x}{2}\right)\right)$$

$$= \frac{d}{dx}\left(\frac{d}{dx}\left(\frac{1}{2}\left(-\frac{\pi^2}{2}\sin\left(\frac{\pi x}{2}\right) + 1\right)\right)\right) = \frac{d}{dx}\left(-\frac{\pi^3}{8}\cos\left(\frac{\pi x}{2}\right)\right)$$

$$= \frac{\pi^4}{16}\sin\left(\frac{\pi x}{2}\right)$$

Since its a sin function we can bound it between -1 and 1

$$-1 \leq \underbrace{\frac{\pi^4}{16}\sin\left(\frac{\pi x}{2}\right) \leq 1}$$

we are interested in finding max error, this term

so my function(e) becomes

$$\text{error}(x) = \frac{\frac{\pi^4}{16}\underbrace{\sin\left(\frac{\pi x}{2}\right)\xi}_{\leq 1}}{4!}(x-x_0)(x-x_1)(x-x_2)(x-x_3)$$

$$\underbrace{\cdots \leq}\; e \leq \frac{\pi^4}{16}\cdot\frac{1}{4!}(x-x_0)(x-x_1)(x-x_2)(x-x_3)$$

we can ignore this term for now since we are looking for e_{max}

Plug in the given values $x_0 = 0, x_1 = 2, x_2 = 3, x_3 = 4$

$$e \leq \frac{\pi^4}{384}(x-0)(x-2)(x-3)(x-4)$$

now find x that maximizes e!

I used wolfram maximum find calculator (Attached in the back) to find it.

$e_{max} \approx 0.2388$ @ $x \approx 2.469$

Max/Min Finder

Find the maximum

of \frac{\pi ^4}{:

Submit

(no global maxima found)

Need a step by step solution for this problem? >>

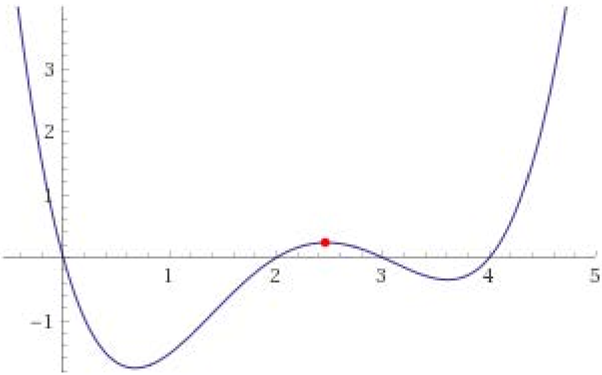
Local maximum:

More digits | Exact form

$$\max\left\{\frac{1}{384}\pi^4 (x-0)(x-2)(x-3)(x-4)\right\} \approx 0.238799 \text{ at } x \approx 2.46909$$

Need a step by step solution for this problem? >>

Plot:



(x from -0.4 to 4.9)

WolframAlpha Get this widget

3. (a) with the give following data for x and $f(x)$, interpolate these data using python "built in" function.
- Piecewise linear interpolation
- L.P. interpol
- Spline interpol

· You can find my code, 3 plots for each methods (w/ data points plot) in "HW1_3.Py" in Github.

(b) Briefly discuss the differences btw the resulting interpol's
· Discussion statement is included in "HW1_3.Py" in Github.

4. (a) Given error form, $E=kh^p$; write this problem as a linear system $A\vec{x}=\vec{b}$, where $x=\binom{\ln(k)}{p}$ is the vector of unknowns.

$E=kh^p \Rightarrow \ln(E)=\ln(kh^p)=\ln(k)+\ln(h^p)=\ln(k)+p\ln(h)$

$$\underbrace{\ln(E)}_{\vec{b}} = \underbrace{\ln(k)}_{x_0} + \underbrace{p}_{x_1}\ \underbrace{\ln(h)}_{\vec{A}}$$

$E_0 \cdots E_8$ $\quad h_0 \cdots h_8$ ← I'm not writing 8 repetitive eq's with differen E ($E_0 \sim E_8$) and h ($h_0 \sim h_8$). This form represent the idea.

(b) Derive the normal equations for this sys: write the matrices in $\vec{A}\vec{x}=\vec{b}$ form.

$$\underbrace{\begin{pmatrix} 1 & \ln(h_0) \\ 1 & \ln(h_1) \\ 1 & \vdots \\ 1 & \\ 1 & \ln(h_m) \end{pmatrix}}_{\vec{A}} \underbrace{\begin{pmatrix} \ln(k) \\ p \end{pmatrix}}_{\vec{x}} = \begin{pmatrix} \ln(E_0) \\ \ln(E_1) \\ \vdots \\ \vdots \\ \ln(E_8) \end{pmatrix}$$

where m=8

Use transpose, A^T : $A^TA\vec{x}=A^T\vec{b}$, we can get $\vec{x}$ matrix.

(c) use python to solve (and w/ normal equation) for k and p via least squares methods

· Code, Plot, and k, p values can be found in "HW1_4" in Github.

$p = 1.777$ $\qquad$ $k = 27.053$

(d) Solve for k, p using Scipy's Curve fit.

· Code, Plot, and k, p values can be found in "HW1_4"

$p = 1.571$ $\qquad$ $k = 11.432$

(e) log-log plot, lin-lin plot, comments can be found in "HW1_4"