Ay190 – Worksheet 02
Chatarin (Mee) Wong-u-railertkun
Date: January 16, 2014

1 An Unstable Calculation

Create a python function to calculate sequence and recurrence relation described in the problem, called **recur13**, in the file `Ay190_Lib_2.py`

n	Recur Value	Closed form Value	Absolute Error	Relative Error
0	1.0	1.0	0.0	0.0
1	0.333333	0.333333	0.0	0.0
2	0.111111	0.111111	-4.47035e-08	-4.02331e-07
3	0.0370372	0.037037	-1.82539e-07	-4.92856e-06
4	0.0123464	0.0123457	-7.31088e-07	-5.92181e-05
5	0.00411815	0.00411523	-2.92482e-06	-0.000710731
6	0.00138344	0.00137174	-1.16988e-05	-0.00852843
7	0.000504043	0.000457247	-4.67952e-05	-0.102341
8	0.000339597	0.000152416	-0.000187181	-1.22809
9	0.000799529	5.08053e-05	-0.000748724	-14.7371
10	0.00301183	1.69351e-05	-0.00299489	-176.846
11	0.0119852	5.64503e-06	-0.0119796	-2122.15
12	0.0479202	1.88168e-06	-0.0479183	-25465.8
13	0.191674	6.27225e-07	-0.191673	-305589.0
14	0.766693	2.09075e-07	-0.766693	-3.66707e+06
15	3.06677	6.96917e-08	-3.06677	-4.40048e+07

Table 1: Table shows values from recurring method, and closed form, including absolute and relative errors for n = 0, 1, 2, ..., 15. Numbers are shown with single digit precision FP numbers.

Thus, even at such a low n as 15, the relative error grows up to 10^7 already.

2 Finite Difference Approximation and Convergence

Create a python function to calculate first-order derivative using forward differencing and central differencing, called **forward** and **central**, in the file `Ay190_Lib_2.py`, respectively.

It turns out that even at step size $h = 10^{-3}$, the differences between these two methods are hard to notice when plotted against each other. Thus, in figure 1, I plot with $h = 1$ to intentionally exaggerate the error. Moreover, I plot the analytical answer for the first order derivative of the function.

Figure 2 and figure 3 illustrate the convergence rate of forward and central differencing, respectively. We can see that the absolute error for forward differencing method is reduced linearly with step size h as expected. However, the absolute error for the central differencing method doesn't go down quadratically as expected.

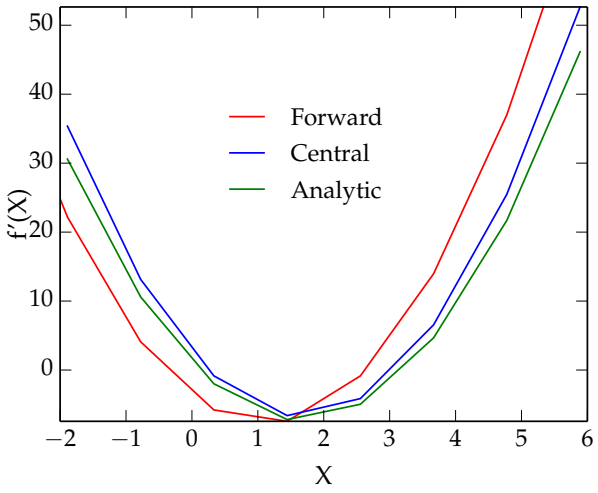

Figure 1: Comparing answer from two different methods in finding first-order derivative, against the analytical answer. The step size is 1, to exaggerate the error.

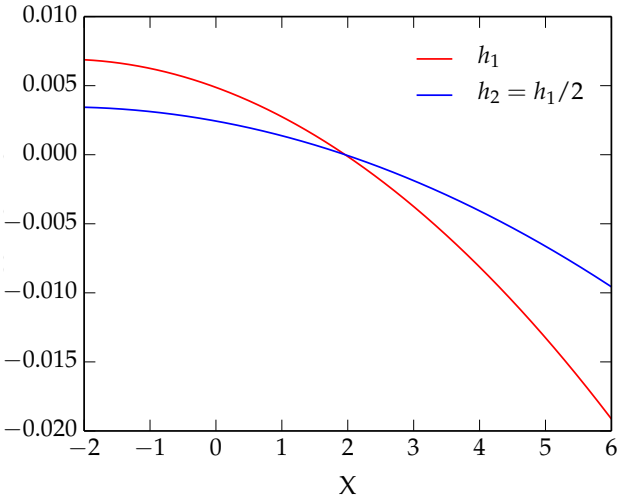

Figure 2: Showing absolute errors of forward differencing method with two different h, with $h_1 = 10^{-3}$. We can see that the error reduce linearly with h as expected.

3 Second Derivative

Similar to in the class notes, we use Taylor's theorem to find an approximation for f' at x_o:

$$f(x_o + h) = f(x_o) + hf'(x_o) + \frac{h^2}{2}f''(x_o) + \frac{h^3}{6}f'''(x_o) + O(h^4)$$
$$f(x_o - h) = f(x_o) - hf'(x_o) + \frac{h^2}{2}f''(x_o) - \frac{h^3}{6}f'''(x_o) + O(h^4)$$

Combine these two equations together

$$f(x_o + h) + f(x_o - h) = 2f(x_o) + h^2 f''(x_o) + O(h^4)$$

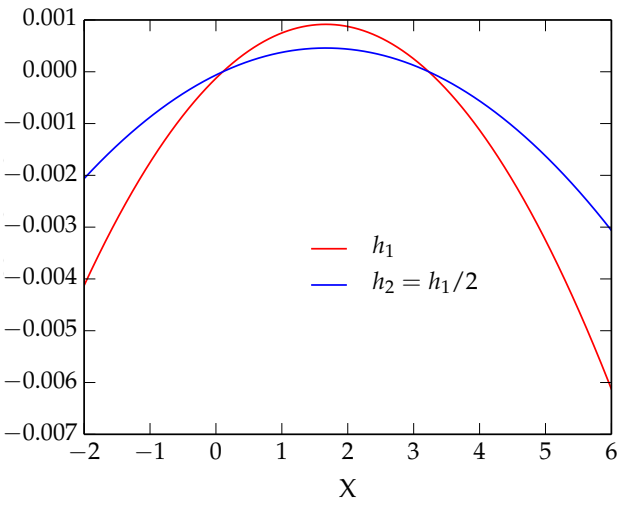

Figure 3: Showing absolute errors of central differencing method with two different h, with $h_1 = 10^{-3}$. However, the error reduce linearly with h, instead of quadratically as expected.

Then, by solving for the second-order derivative

$$f''(x_o) = \frac{f(x_o + h) - 2f(x_o) + f(x_o - h)}{h^2} + O(h^2)$$

4 Interpolation: Cepheid Lightcurve

4.1 Global Lagrange Interpolation

Figure 4 illustrates the global Lagrange interpolation against its 9 known data points. We could see the Runge's phenomenon of non-convergence toward the boundary points.

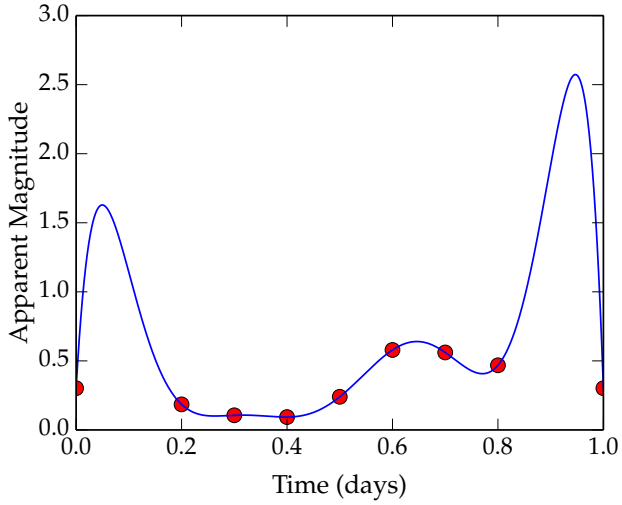

Figure 4: Plotting a single global Lagrange interpolation polynomial against its 9 known data points.

4.2 Linear Vs. Quadratic Interpolation

Figure 5 compares results from linear and quadratic interpolation of the same data.

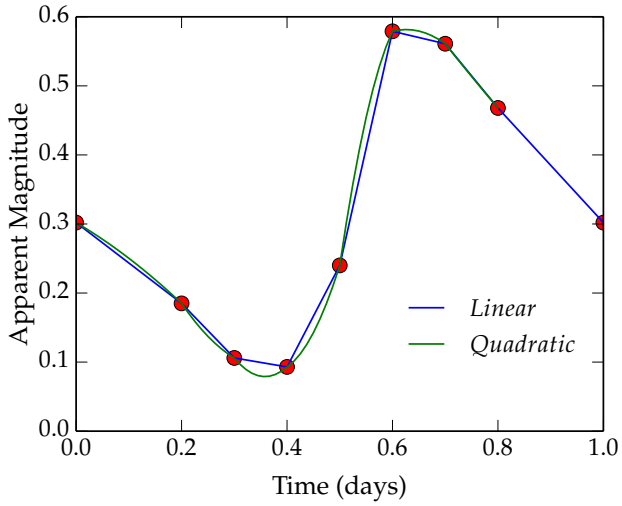

Figure 5: Comparing results from two different methods, linear and quadratic interpolation, on the same set of data

5 More Cepheid Lightcurve Interpolation

5.1 Piecewise Cubic Hermite Interpolation

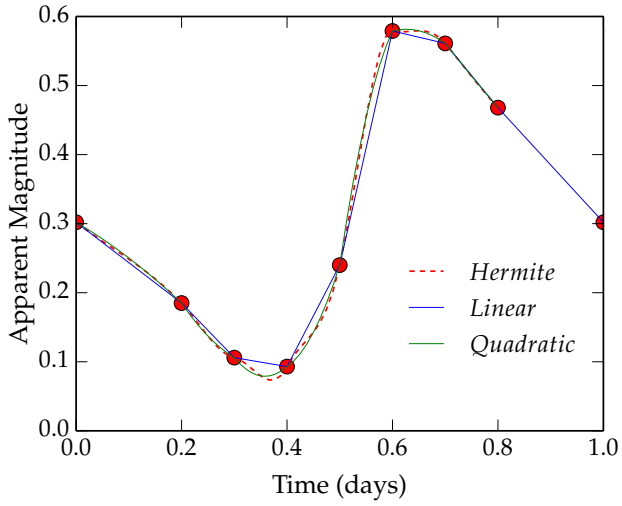

Figure 6

5.2 Cubic Spline Interpolation

Using Scipy.interpolate to perform cubic spline interpolation, instead of writing my own function.

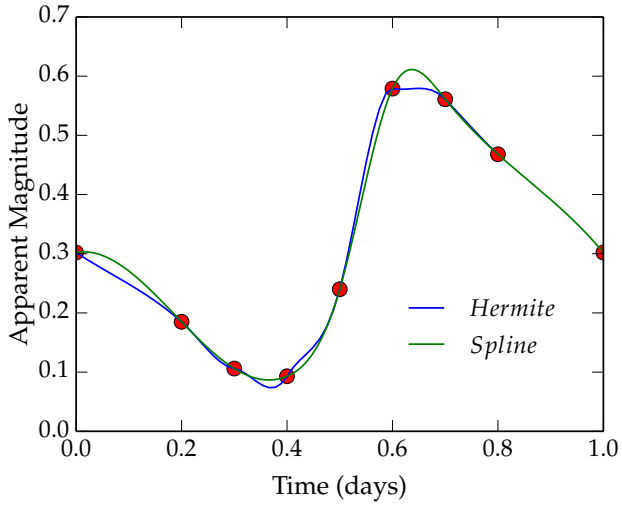

Figure 7

6 Note

All python functions created are grouped together in the file `Ay190_Lib_2.py`. Other python scripts, used to create figures, are named in the format of Q[question number]-[order of figure in that question].py