Problem set 5

[Revised Jan 31, 3:49 PM]

Interpolation in 1D

1. For this question you will use the linear and quadratic interpolation polynomials $P_1(x)$ and $P_2(x)$. Recall these polynomials are constructed using discretely sampled values for x_i and $f(x_i), i = 0, \ldots, n$. The two interpolation polynomials are given by

$$P_1(x) = f_i + (x - x_i)\left[\frac{f_{i+1} - f_i}{x_{i+1} - x_i}\right] \qquad x \in [x_i, x_{i+1}]$$

and

$$P_2(x) = f_i + (x - x_i)\left[\frac{f_{i+1} - f_{i-1}}{2h}\right] + \frac{1}{2}(x - x_i)^2\left[\frac{f_{i+1} - 2f_i + f_{i-1}}{h^2}\right] \qquad x \in [x_{i-1}, x_{i+1}]$$

where we assumed that $h = x_i - x_{i-1} = x_{i+1} - x_i$.

Calculate approximate values of $f(x)$ at $x = 0.06, 0.16, 0.26, 0.36, 0.46$ using

- linear interpolation
- quadratric interpolation (over the three nearest points)

given the data

```
x = numpy.array( [0.0, 0.1, 0.2, 0.3, 0.4, 0.5] )

f = numpy.array( [1.0000, 0.9950, 0.9801, \
                  0.9553, 0.9211, 0.8776] )
```

This question should be solved using Python code.

2. Calculate approximate values of f at $x = 0.15$ using quadratic interpolation with

- $x_0 = 0.1,\ x_1 = 0.2,\ x_2 = 0.3$
- $x_0 = 0.0,\ x_1 = 0.3,\ x_2 = 0.6$

where f is given by the table of values below

```
x = numpy.array( [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6] )

f = numpy.array( [7.389056, 7.028688, 6.685894, \
                  6.359820, 6.049647, 5.754603, 5.473947] )
```

Given that the exact value of $f(0.15) = 6.855149$, show that the error decreases in the manner predicted by the theoretical estimate ($O(h^3)$) as the interval of the width $2h$ is decreased.

This question should be solved using Python code.

3. Interpolate the data in Question 1 using

   - `numpy.interp` (linear interpolation)
   - `CubicSpline` from `scipy.interpolate` (cubic spline interpolation).

   Evaluate both polynomials on a high resolution set of evaluation points. Create a plot showing both of interpolated values on the high-resolution evaluation points. Save this plot to a PDF called `ps_5_qu_3_interp.py`.

   Please refer to `interpolate_with_scipy.ipynb` for more details and examples of how to do this. This question should be solved using Python code.

4. Using the data from Question 1 we wish to try out `RBFInterpolator` from `scipy.interpolate` (radial basis function interpolation). Refer here for example usages of this interpolation

   Create two RBF's using

   - `kernel = "linear"`
   - `kernel = "quintic"`

   Evaluate both polynomials on a high resolution set of evaluation points. Create a plot showing both of interpolated values on the high-resolution evaluation points. Save this plot to a PDF called `ps_5_qu_4_interp_rbf.py`. How does the RBF interpolations compare with the linear and cubic spline approache used in Question 3? What do you think the benefits of using RBFs are?

   This question should be solved using Python code.