Numerical Analysis - Midterm

Submission instructions:

Please submit a PDF file that includes all answers and all the code you used (even in questions that don't require to hand in a *.py file). Do not use print screen to add the code to the PDF - it should appear in the PDF file as text.

If at any point you call functions from imported python libraries, all import statements must be included with the code.

In addition, you should submit two *.py files, **divdiff.py** and **interpnewt.py**

Compress the PDF file and the python files into a single *.zip or *.rar file, named yourID.zip.

Each exercise is worth 25 points.

Question 1: Interpolation code

1. Implement a function that constructs the coefficients (divided differences) required for evaluating the Newton form of the interpolation polynomial, that is $f[x_0, \dots, x_k]$ for $0 \leq k \leq n$. This function should have the signature divdiff(x,y) and return the output c where
   - x is the given sampling points
   - y is the given function values
   - c is the coefficients for the newton inerpolation polynomial.
2. Implement a function which evaluates Newton's interpolation polynomial at a new set of points **xnew.** Your function should have the signature **interpnewt(c,x,xnew**) and return the output **[ynew]** where
   - c is the interpolation coefficients computed by **divdiff**
   - x is the same x from **divdiff**
   - xnew are the new points to evaluate the interpolating polynomial at.
   - ynew are the values of the interpolating polynomial at **xnew**.

Important instructions:

- The variables **x, c, y, xnew, ynew** should all be numpy arrays with dimension 1×(number of points \ coefficients)
- You can use Horner's rule to make running time of the second function linear in the degree of the interpolation polynomial.

Question 2: Error approximation

Given the functions

$$\cos(5x),\ x \in [0, 2\pi] \qquad e^x,\ x \in [0, 10], \qquad \frac{1}{x^2+1},\ x \in [-5, 5]$$

For each function, estimate the maximal error derived by approximating $f(x)$ using an inerpolation polinomial numerically, and plot the error (on a logarithmic scale) as a function of the umber of points n for equally spaced points and for Chebyshev points. Explain how you computed the estimates. Choose different values of n so that you get an informaitve graph. Add the plots to the PDF.

Question 3: Edge detection in images - the Sobel operator

Download the image added in the submission box from Moodle. Load the image into an **numpy ndarray M** [Read the image into an ndarray variable in Python named 'M' using the imread function of matplotlib library].

Now, theoretically, for each pixel in the image we wish to calculate the central difference in the x direction and save it to a numpy ndarray Dx. In practice, instead of the central difference in the pixel (i, j) we will calculate

$$\begin{aligned} Dx(i,j) = &M(i-1,j-1) - M(i-1,j+1) \\ &+ 2 \cdot M(i,j-1) - 2 \cdot M(i,j+1) \\ &+ M(i+1,j-1) - M(i+1,j+1) \end{aligned}$$

Note that this is not possible for pixels on the margins of the image, i.e of the form $[0, j], [511, j], [i, 0], [i, 511]$, so do not compute the central difference there, that is, $Dx(i, j)$ should be of size 510×510. Do the same for the y direction and save it into a numpy ndarray Dy. Now create a numpy ndarray $G(i, j) = \sqrt{Dx^2 + Dy^2}$, normalize it between 0 and 1, and use the **imshow** and **imsave** functions of **matplotlib** library to display and save it.

- Add the resulting image to your PDF
- What do we see in the image? How is the operation $G(i, j)$ interpreted in the image?

Question 4: Three Point Endpoint formula approximation

In recitation 3, we derived together an approximation to the derivative of a function f using three sampling points

$$f'(x_0) \approx \frac{1}{2h}\left[-3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h)\right]$$

Implement a function that generates approximate values of a given mathematical function **f** at the points of the form $x, x + h, x + 2h$ with random errors bounded by $0.5 \cdot 10^{-5}$.

For each of the values $h = 10^{-1}, \ldots, 10^{-8}$ and each of the functions $\cos(x)$, e^x, $\frac{1}{x^2+1}$, use the three point formula and the approximate values generated by your previous code to approximate the derivative at $x = 1.2$. Plot an appropriate graph for each of the functions.

Explain the results. Add any code you wrote for this exercise to the PDF.