AA 2021/2022 Computational Methods
Lesson 2 - Numerical Differentiation

Almaz Khabibrakhmanov*, Mario Galante†, Alexandre Tkatchenko

Theoretical Chemical Physics group, FSTM,
Campus Limpertsberg, University of Luxembourg

Exercise

First derivative and the error of differentiation

1. Create a new file `diff.py` in your directory using `Spyder`. Based on the script `forwDer.py` from Moodle, define a function `centDer` which implements central differences scheme:
$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} + O(h^2) \tag{1}$$

2. Create a `numpy` 1D array, `hList`, containing 50 equally spaced points between 0.05 and 0.5 using the `linspace` function.

3. Take $f(x) = sin(x)$ and calculate its first derivative at point $x_0 = 0.5$ for the values of h from `hList` using functions `forwDer` and `centDer`. As a result, you should have two 1D arrays `forw` and `cent` containing the corresponding derivatives for all h. Do not forget to take the advantage of the `numpy` functionality.

4. To obtain the arrays with differentiation errors `errorForw` and `errorCent`, subtract the analytically derived derivative from your results:
$$p(h) = |f'_{num}(x_0, h) - f'_{an}(x_0)|, \tag{2}$$
where $f'_{an}(x_0)$ is the analytical derivative, and $f'_{num}(x_0, h)$ is the numerical derivative. Calculate $p(h)$ using both `forwDer` and `centDer`.

5. Plot the obtained results using `matplotlib.pyplot` library. What do you see?

6. Use the script `plotter.py` from Moodle as a guide to modify the appearance of your plot: add title; set axis title and ticks; specify line colors, widths, and labels; add the legend.

*almaz.khabibrakhmanov@uni.lu
†mario.galante@uni.lu

Second derivative and optimal choice of step h

7. Create a new file `diffErr.py` in your directory using `Spyder`. Paste function `centDer` here. Do not forget to import `numpy` and `matplotlib.pyplot` libraries before using them.

8. The second derivative can be calculated via the formula:

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} + O(h^2) \tag{3}$$

   Analogously to 1), define a function `secDer` implementing this numerical scheme.

9. Define a 1D array of numbers evenly spaced in logarithmic scale using the `numpy` function `logspace`. Google the documentation of `numpy.logspace` and define `hList` with 200 points between 10^{-20} and 10^0.

10. As in 3), take $f(x) = sin(x)$ and $x_0 = 0.5$. For every h calculate the error of differentiation according to the Eq. (2). Calculate $p(h)$ using both `centDer` and `secDer`.

11. Take a look at the figure `firstDerErrors.png` from Moodle. This plot shows the dependence of total error of differentiation for the forward differences scheme. Note the logarithmic scale of both x- and y-axis.

12. Now the task is create a similar looking plot for `centDer` and `secDer`. You may use `loglog` function of `matplotlib.pyplot`. Look in the `plotter.py` or Google for more details. There is no need to plot black line, just focus on the red points.

13. Save your plots as `centDerErrors.png` and `secDerErrors.png` in your directory using function `savefig` from `matplotlib.pyplot`. See `plotter.py` or Google for the proper syntax.

14. Compare all three plots that you have. What can you see? Which conclusion can you make about the influence of round-off errors on the first and second numerical derivatives?

Advanced/facultative tasks

15*. Repeat 1) - 6) for the fourth order numerical differentiation scheme (`fourDer`):

$$f'(x_0) = \frac{f(x_0-2h) - 8f(x_0-h) + 8f(x_0+h) - f(x_0+2h)}{12h} + O(h^4) \tag{4}$$

Simply complement the script `diff.py` that you already have and then compare the results for all three schemes (`forwDer`, `centDer` and `fourDer`).

16*. Repeat 10) - 13) for `fourDer`. Analyze the results.

17*. Prove mathematically (using paper, pen, and your knowledge of Calculus) that Eqs. (3) and (4) define the numerical differentiation methods of the order $O(h^2)$ and $O(h^4)$, respectively.