MENUM Lab

Finite Differences

The objective of this practical work is to be able to program finite differences using `python` . The first part deals with the 1D setting, and the second one with the 2D setting.

Reports : They have to be done by groups of two, and the next rules have to be followed :
— The work has to be submitted on the course website (one submission for each group).
— The report has to be a pdf file.
— The name of the file has to be : `MENUM_FD_Name1_Name2.pdf`
— You must also submit your scripts under the following name : `MENUM_FD_FILES_Name1_Name2_GRPY.zip` (with `Y` is your lab group letter (`A`, `B` or `C`))
— Do not copy and paste your whole code into the report. However you can paste some excerpts of your code, but the structure (tabulations) must be preserved.
— **Note that the questions proposed here are "crossing points" to help you for the work. Just answering to these questions will not be considered as a good report : you have to synthetize and explain your results.**

1. 1D problem

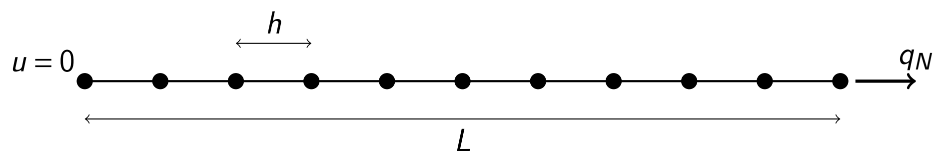


FIGURE 1 – Unidimensional example

Consider the following strong form, defined for $x \in]0, L[$, with $L = 1$:

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + r(x) = 0 \quad \forall x \in]0, L[\\ u(0) = 0 \\ \dfrac{\partial u}{\partial x}(L) = q_N \end{cases} \tag{1}$$

In your program, you will consider the following cases : $r(x) = 1. =$ constant, $r(x) = \sin\left(\frac{\pi x}{L}\right)$ and $r(x) = \sin\left(\frac{\pi x}{2L}\right)$. **Select** $q_N = 1$ **for all these cases.**

We want to find the solution of the problem by means of finite differences :

1. Write the discretization of the PDE using a 3 points centred approximation of the second order derivative.
   What is the order of truncature of this scheme ?
2. How would you take into account the boundary condition at $x = 0$?
   Choose the additional equation approach.
3. Treat the Neumann BC with a backward approximation.
   What is the order of truncature of this scheme ?
4. Allow your code to be able to use a centred approximation for the Neumann BC ("fictitious point method").
5. Write the algorithm of your procedure.
6. Program this scheme using `python` and `numpy` (see below for some informations).
7. Compare the results with the analytical solution (that you have to express for the three proposed cases).
8. For the three given source terms : Plot the evolution of the error ε with respect to the inverse of the grid distance ($1/h$) **using a log-log plot** (see appendix A for more details).

   $$\varepsilon = \max(|numericalSolution(x) - exactSolution(x)|)$$

   Do it for the two Neumann approaches proposed above.
   This leads to three graphs (three possible source terms) each one involving two curves (two possibilities for the Neumann BCs).

Setting up your work environment :

— You can code using `spyder` which is a `numpy` development environment. The interface is quite close to matlab.
— To launch spyder with **english** language : "Tools" – "Preferences" – "Advanced Settings" tab : "English"
— **Advice :** Check the following options under "Tools" – "Preferences" – "IPython console" – "Graphics" tab :
☑Automatically load pylab and numpy modules
☑Graphics backend : Qt5. (**relaunch spider!**)

Useful numpy commands :

A showcase of the commands you will use during the lab is available in the file `numpySyntax.py` which is available on hippocampus (along with this pdf). You can also refer to the following resources :

2. 2D problem

Consider the following PDE, defined for $(x, y) \in]-L/2, L/2[^2$; $L = 2.$:

$$\begin{cases} \Delta u + 1 = 0 \\ u = 0 \quad \text{on} \quad \partial\Omega \end{cases}$$

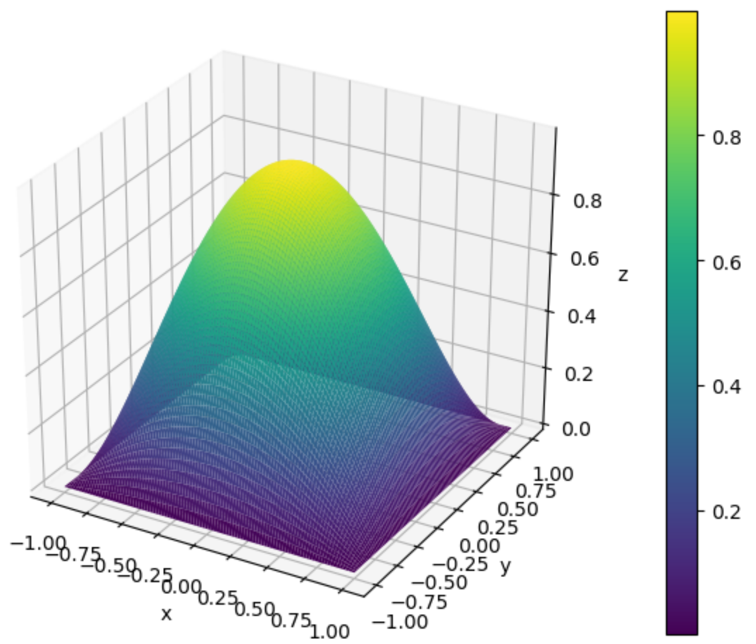


FIGURE 2 – Bi-dimensional example

Questions :

1. Discretize the space using equi-spaced nodes on the domain Ω;
2. Write the finite difference approximation of the PDE by means of a 5 points approximation of the Laplacian operator;
3. What is the size of the linear system that will arise?
4. Propose a mapping from nodal coordinates (i,j) to degrees of freedom's numbering in the linear system;
5. Explain how to take into account the boundary conditions;
6. Write the algorithm of your procedure;
7. Program the F.D. scheme;
8. Plot the numerical solution;
9. Compare your results with the analytical solution of the problem :

$$u(x,y) = \sum_{odd\,i}^{\infty} \sum_{odd\,j}^{\infty} \frac{64}{\pi^4(i^2+j^2)ij} \sin\left(\frac{i\pi(x+1)}{2}\right) \sin\left(\frac{j\pi(y+1)}{2}\right) \tag{2}$$

10. Plot the evolution of the error ($\max(|numericalSolution - exactSolution|)$) with respect to the grid spacing $(1/h)$ (**using a log-log plot**);
11. Compare your results with the following theorem :

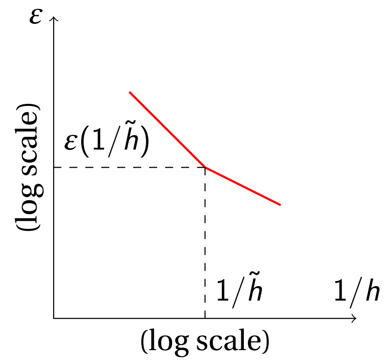
    Let $u(x,y)$ be the exact solution of a Poisson's boundary value problem on a domain $\Omega \in \mathbb{R}^2$. If u and all its partial derivatives are continuous up to fourth order on Ω, then a positive constant C exists such that :

$$\max |u(X_i, Y_j) - u^h(X_i, Y_j)| \leq C\,M\,h^2 \tag{3}$$

where M is the maximal value of the fourth order derivatives on Ω, and u^h is the discrete solution obtained by mean of a five points finite difference scheme of second order.
Hint : what should be the the decrease rate of the error in a log-log plot?

A. Plotting the evolution of the error

In the 1D problem, you are asked to *"Plot the evolution of the error ε with respect to the inverse of the grid distance using a log-log plot"*.



Each point defining the curve in this graph corresponds to the error associated to a given number of nodes $\tilde{N}$. Knowing $\tilde{N}$ allows to compute $1/\tilde{h}$ (absisca in the plot). The associated error $\varepsilon(1/\tilde{h})$ is used to obtain the ordinate in the plot. Solving the FD problem for a set of $\tilde{N}$ ($[4, 8, 16, 32, 64]$ for example) allows to compute two sets : a set of $1/\tilde{h}$ and a set of $\varepsilon(1/\tilde{h})$ that can be plotted in loglog scale using `matplotlib`'s `loglog` command, and obtain the corresponding curve.