1 Numerical integration of the Barotropic Vorticity Equation

1. Integrate numerically the system

$$\begin{cases} \frac{\partial L}{\partial t} = J\left(hL + f, Z\right) \\ h = \frac{gm^2}{f} \\ \nabla^2 \frac{\partial Z}{\partial t} = \frac{\partial L}{\partial t} \end{cases} \tag{1}$$

where $m = \frac{2}{1+\sin\left(\phi(x,y)\right)}$, f is the Coriolis parameter, $L(x,y) = \nabla^2 Z(x,y)$ and $J(a,b) = \frac{\partial b}{\partial y}\frac{\partial a}{\partial x} - \frac{\partial b}{\partial x}\frac{\partial a}{\partial y}$ from the initial condition on Z at $t = 0$ up to t=24h with a time step of 0.5 h. To do this use the functions

- make_Laplacian
- make_Jacobian
- Poisson_solver

and the rectangular domain provided and described in the attached file *BVE_ENIAC_functions.py* and in the lecture slides. To integrate the system, use the Leapfrog method to update z and compute L as the laplacian of z. Extrapolate linearly the values of L to the boundary at the first time step, then keep z and L constant at the boundary. The initial condition is provided in the file *Case1-1949010**5**03.Z00* and is a field of Z at 500 hPa derived from observations.

2. Plot the forecast of Z(t) at $t = t_0 + 24h$ with the analysis Z24 at $t = t_0 + 24$ h provided in the file *Case1-1949010**6**03.Z00*.
You may compare also the tendencies $Z(t) - Z0$ **and** $Z24 - Z0$ where Z0 is the observed field at $t = t_0$. You may also compute the RMSE for the computed forecast and for a persistence forecast defined as a stationary state $Z(t) = Z(t_0)$.

3. Suggest and briefly discuss at least 3 changes of the approach to reduce the error of the forecast **(max 500 words)** and write a short report **(max 3 pages)**.

SUMMARY OF SUGGESTIONS FOR POINT 1:

-Remember that the first time step should be performed with a two-point method (Euler forward).
-Note that at the first time step you need to compute L as the Laplacian of Z (with make_laplacian) and extrapolate it to the boundary, after the first time step you should use the function make_laplacian to compute it in the inner domain and copy the values from the previous time step at the boundary.
-A simple way to proceed is to copy the value of Z at the previous time step into the new one and then apply Euler/Leapfrog in the inner domain only. With Poisson_solver you get the time derivative of Z in the inner domain.
-The whole task can be done with a single *for* cycle over the time steps ('for n in range(0,nt):' where nt is the number of time steps) and one or two (easier) *if* cycles
if n==0:
...
else:
...
and in less than twenty lines in total. This means that there is no need to loop over each point of the domain.
-You may choose to implement the first time step, check it and then implement the other time steps.
-Remember that for an array v of size M the first element of the array is $v[0]$, the last element is v[M-1], while v[0:M] correspond to the whole array (v[0:M-1] excludes the last element)
-Think before writing code.