

Weekly milestones. First semester.

■ Milestone 1 : First steps with the NumericalHUB

Install the Microsoft Visual Studio and the Intel Fortran compiler by following the detailed installation manual:

[Book: Programming with Visual Studio](#)

A pdf file of this book can be found in:

[PDF file: Programming with Visual Studio](#)

Download and open the NumericalHUB by following the instructions given in:

[Software: NumericalHUB](#)

Open the `main_NumericalHUB.f90` file and follow the folder structure and menu. Focus on the Cauchy Problem which is explained in the book:

[Book: How to learn applied mathematics through modern Fortran](#)

[PDF file: How to learn applied mathematics through modern Fortran](#)

Once the NumericalHUB is open, revise the first order and second order examples to understand how to reduce any ordinary differential equation to a system of first order differential equations.

Write a Python modules for the following milestones and compare the results with those obtained with the Python library `scipy`

■ Milestone 2 : Numerical integration of orbits.

1. Write a prototype to integrate Kepler orbits with an Euler method.
2. Write a function to integrate a Cauchy problem.
3. Write a function to express the force of the Kepler movement. Put emphasis on the way the function of the Cauchy problem is written:
$$F = [\dot{r}, -r/|r|^3]$$
 where $r, \dot{r} \in \mathbb{R}^2$
4. Write an Euler, an Inverse Euler, a Crank Nicolson and a fourth order Runge Kutta method.
5. Integrate a Kepler with these latter schemes and explain the results.
6. Increase and decrease the time step and explained the results.
7. Compare these results with those obtained with the library `scipy`.

■ **Milestone 3 : Error estimation of numerical solutions.**

1. Write a function to evaluate errors of numerical integration by means of Richardson extrapolation. This function should be based on the Cauchy problem solution implemented in milestone 2.
2. Numerical error of different temporal schemes: Euler, Inverse Euler, Crank Nicolson and fourth order Runge Kutta method.
3. Write a function to evaluate the convergence rate of different temporal schemes.
4. Convergence rate of the different methods with the time step.

■ **Milestone 4 : Linear problems. Regions of absolute stability.**

1. Integrate the linear oscillator $\ddot{x} + x = 0$ with some initial conditions. Use Euler, Inverse Euler, Leap-Frog, Crank-Nicolson and fourth order Runge Kutta method.
2. Regions of absolute stability of the above methods.
3. Explain the numerical results based on regions of absolute stability.

■ **Milestone 5 : N body problem.**

1. Write a function to integrate the N body problem.
2. Simulate an example and discuss the results.

■ **Milestone 6 : Lagrange points and their stability.**

1. Write function to simulate the circular restricted three body problem.
2. Determination of the Lagrange points $F(U) = 0$.
3. Stability of the Lagrange points: L_1, L_2, L_3, L_4, L_5 .
4. Orbits around the Lagrange points.

■ **Milestone 7 : Orbits of the circular restricted three body problem.**

1. Investigate existing temporal schemes to integrate Cauchy problems. Fortran and Python programming codes. Use ODE integrators of the library scipy (odeint, solve_ivp). LSODA, ODEPACK, ODEX.
2. Integrate Arenstorf's periodic orbit. Compare results among GBS, RK and AM methods.
3. Stability of Lyapunov orbits. Shampine and Gordon orbits.
4. Error tolerance and computational effort.
5. Mixing Fortran and Python: calling Fortran from Python and viceversa.
6. Parallel programming with GPUs. N-body problem.
7. Standalone Python codes.
8. Automatic testing with GitHub.
9. Mixing Fortran, C++ and Python in one executable file.
10. Python script for GMAT.