AMPLIACION DE MATEMATICAS I

MILESTONES REPORT

MASTER UNIVERSITARIO EN SISTEMAS ESPACIALES

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA AERONÁUTICA Y DEL ESPACIO

UNIVERSIDAD POLITÉCNICA DE MADRID

Change Control Log

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Author/s | Comments |
| 01 | 05 Oct 2023 | David Calleja | Initial Release: including milestone 1 and 2 |
| 02 | 06 Oct 2023 | David Calleja | Revision of milestone 2 |

Table of Contents

[Introduction 7](#_Toc147516230)

[1 Milestone 1 7](#_Toc147516231)

[1.1 Objective 7](#_Toc147516232)

[1.2 Results 8](#_Toc147516233)

[2 Milestone 2 9](#_Toc147516234)

[2.1 Objective 9](#_Toc147516235)

[2.2 Results 9](#_Toc147516236)

Index of Figures

Figure 1. Kepler integrated with Euler Method before creating functions (code removed from the final script) 7

Figure 2. Kepler as a function in a temporal scheme 7

Figure 3. Kepler as a function of Inverse Euler before it crashes 10

Figure 4. Kepler as a function of Inverse Euler for N>20 10

Figure 4. Harmonic Oscillator using Cauchy function and Euler temporal scheme 11

Index of Tables

**No table of figures entries found.**

PAGE INTENTIONALLY LEFT BLANK

# Introduction

This document summarizes the different results obtained from the weekly milestones of the course AMPLIACION DE MATEMATICAS I. It also explains the programming process for some key items.

# Milestone 1

## Objective

A close up of text

Description automatically generated

Although initially created without functions, just as formulas as they are show in the following script for Kepler (Figure 1), they naturally evolved while programming to functions in the time domain. Which provides a cleaner and easier code to develop, read, and correct.

Therefore, this script was developed with functions.

|  |
| --- |
| # Old version before moving to time domain  U = array( [1,0,0,1] )  N = 10000  x = array( zeros(N) )  y = array( zeros(N) )  x[0] = U[0]  y[0] = U[1]  for i in range(1,N):  F = array( [ U[2], U[3], -U[0]/(U[0]\*\*2+U[1]\*\*2)\*\*1.5, -U[1]/(U[0]\*\*2+U[1]\*\*2)\*\*1.5 ] )  U = U + dt\*F  x[i] = U[0]  y[i] = U[1] |

Figure 1. Kepler integrated with Euler Method before creating functions (code removed from the final script)

|  |
| --- |
| def Kepler(U, t):  x = U[0]; y = U[1]; dxdt = U[2]; dydt = U[3]  d = ( x\*\*2 +y\*\*2 )\*\*1.5  return array( [ dxdt, dydt, -x/d, -y/d ] ) |

Figure 2. Kepler as a function in a temporal scheme

## Results

|  |  |
| --- | --- |
|  |  |
|  |  |
| A blue circle with a line in the middle  Description automatically generated | A graph of a graph with a blue circle  Description automatically generated with medium confidence |

Time step and iterations is changed to evaluate the results. While Euler requires a relatively small-time step (dt) and significant number of iterations to obtain a non-closed orbit. RK4 and Crank Nicolson provide good results with a bigger dt and a significantly reduced number of iterations.

# Milestone 2

## Objective

A white text with black text

Description automatically generated

## Results

Since 3 out of the 4 required functions were developed under the milestone 1 (Euler, Crank-Nicolson and RK4), only Inverse\_Euler was created. The function to represent the Kepler movement was also part of the Milestone 1, which was also created as a function. A module with the 4 methods and the Kepler function was called Temporal\_Schemes. In the same module Kepler was integrated into the 4 methods and plots where created.

However, in order to organize the different modules separating between:

1. Temporal\_schemes: Where the numerical methods are located (RK4, CN…)
2. Physics: Where physics are located (Kepler, harmonic oscillator…)
3. Operations: Where temporal\_schemes and physics modules are called to perform operations with them and make plots and representations

On one hand, items 1, 2, 3 and 4 of this milestone (Euler, CN, RK4, Inverse\_Euler) are part of: “Temporal\_Schemes”. While the item 6 (Kepler) of this milestone is part of the module “physics” (it was also added to Physics.py the harmonic oscillator, which by the way, its state vector is a matrix with 2 rows and 1 column, not like Kepler where U is a matrix of 4 rows and 1 columns).

The module “**MILESTONE\_2\_Operations**” makes the respective calls to “Temporal\_Schemes” and “Physics” to integrate Kepler into each different method, responding to the item 7 and 8 of this milestone.

The results from Euler, RK4 and CN are explained in the previous chapter. Inverse Euler however is tested for first time in this milestone, detecting that after 20 iterations (N+20) it fails to converge.

A graph of a function

Description automatically generated

Figure 3. Kepler as a function of Inverse Euler before it crashes

A screenshot of a computer program

Description automatically generated

Figure 4. Kepler as a function of Inverse Euler for N>20

Finally, a different module called “Cauchy\_problem” is created containing a function to integrate a Cauchy problem.

* Input arguments of the function:
  + Temporal\_scheme: any of the numerical methods used to resolve the problem (RK4, Euler…)
  + Time\_domain:
    - t: time partition t (vector of length N+1)
    - dt: time step
  + Uo: initial condition at t=0
  + F(U,t) : Function dU/dt = F(U,t) : it is the physics of the problem
* Return:
  + The state vector U at the different time steps of the time domain: U: matrix[Nv, N+1], Nv state values at N+1 time steps
  + To make plots I collect the different values of U matrix (state vector):
    - 1st row: x
    - 2nd row: y
    - 3rd row: dx/dt
    - 4th row: dy/dt

*Note:*

*from Uo I know the number of rows of the matrix U*

*from t I know the number of columns of the matrix N*

While initially I created the different temporal schemas based on t1 and t2, being dt=t2-t1, I added dt to all the schemas and dt as an input parameter of the Cauchy problem. Then I tested using the Harmonic Oscillator from Physics.py, and using Euler. It worked out since it tends to the “explosion” following the theory:

A graph with a blue line

Description automatically generated

Figure 5. Harmonic Oscillator using Cauchy function and Euler temporal scheme

**END OF DOCUMENT**