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 |

Table of Contents

[Introduction 7](#_Toc147327519)

[1 Milestone 1 7](#_Toc147327520)

[1.1 Objective 7](#_Toc147327521)

[1.2 Results 8](#_Toc147327522)

[2 Milestone 2 9](#_Toc147327523)

[2.1 Objective 9](#_Toc147327524)

[2.2 Results 9](#_Toc147327525)

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. Milestone 2 for one step (U1) for dt=0.1 9

Figure 4. Milestone 2 for one step (U1) for dt=0.001 10

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

Since the functions were created in the previous milestone, they were simplified to only one step, providing the results of N=1.

Inverse\_Euler for one step was added, and finally, Kepler was integrated into Euler, CN, RK4 and Inverse Euler.

To facilitate the item 8, increase and decrease analysis of the time step, a simple console script is developed.

## Results

A black and white screen with numbers and letters

Description automatically generated

Figure 3. Milestone 2 for one step (U1) for dt=0.1

A screen shot of a computer

Description automatically generated

Figure 4. Milestone 2 for one step (U1) for dt=0.001

Smaller dt shows a closer value of U1 to U0 and more similarity between methods, while bigger dt shows less accuracy on implicit methods.

**END OF DOCUMENT**