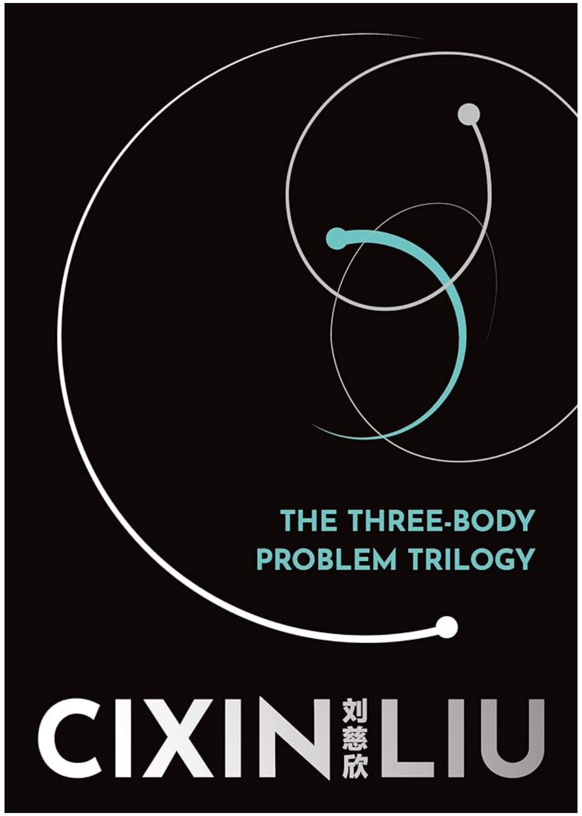
Coursework 1 intro: week 4

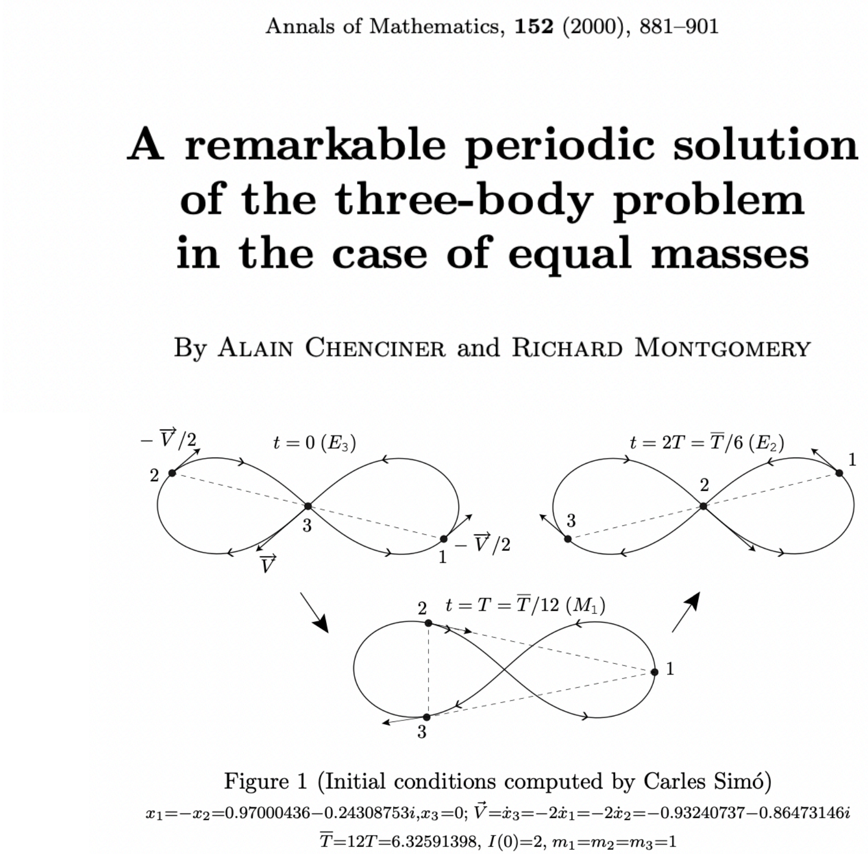
Trisolaris: the three body problem

Dr K Clough, Topics in Scientific computing, Autumn term 2023

Coursework - Trisolaris, the three body problem

Coursework - Trisolaris, the three body problem





Annals of Mathematics, **152** (2000), 881–901

A remarkable periodic solution of the three-body problem in the case of equal masses

By Alain Chenciner and Richard Montgomery

Figure 1 (Initial conditions computed by Carles Simó)

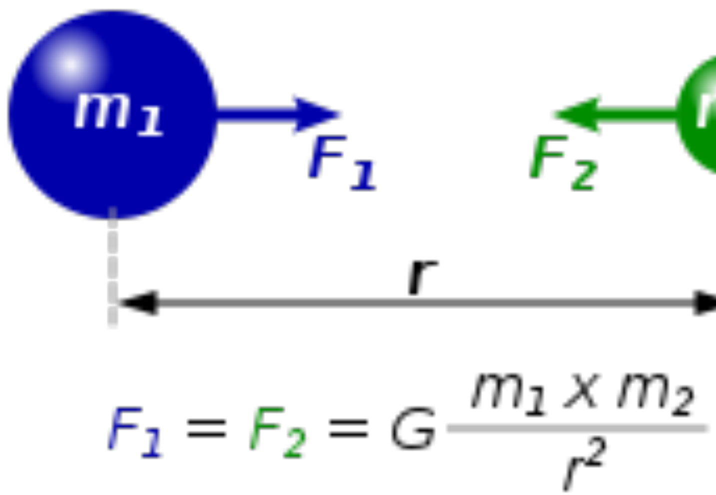
$x_1 = -x_2 = 0.97000436 - 0.24308753i, x_3 = 0;\ \vec{V} = \dot{x}_3 = -2\dot{x}_1 = -2\dot{x}_2 = -0.93240737 - 0.86473146i$

$\overline{T} = 12T = 6.32591398,\ I(0) = 2,\ m_1 = m_2 = m_3 = 1$

Coursework - Trisolaris, the three body problem

1. Revise Newtonian mechanics
2. Discuss problems
3. Discuss strategy
4. Q&A
5. Complete feedback questionnaire

Let’s revise Newtonian mechanics



$$\vec{F}_{12} = -\frac{G M_1 M_2}{r_{12}^2} \frac{\vec{r}_{12}}{|\vec{r}_{12}|}$$

Let’s revise Newtonian mechanics

$$\vec{F} = M\,\frac{d^2\vec{x}}{dt^2}$$

Let's revise Newtonian mechanics

The ODE for the x position of each star is:

$$\frac{d^2 x_i}{dt^2} = \sum_{j=1, j \neq i}^{j=N} \frac{GM_j}{r_{ij}^2} \cos(\theta)$$

and for the y position:

$$\frac{d^2 y_i}{dt^2} = \sum_{j=1, j \neq i}^{j=N} \frac{GM_j}{r_{ij}^2} \sin(\theta)$$

where

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

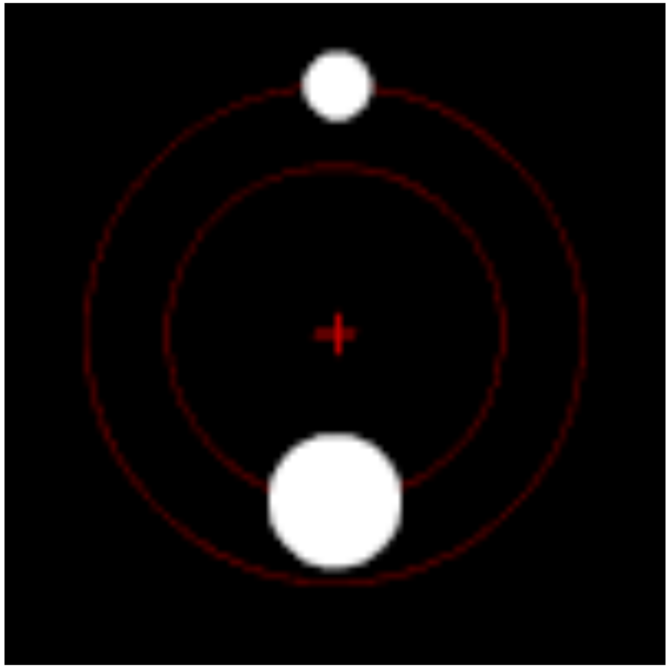
$$\cos\theta = (x_j - x_i)/r_{ij}$$

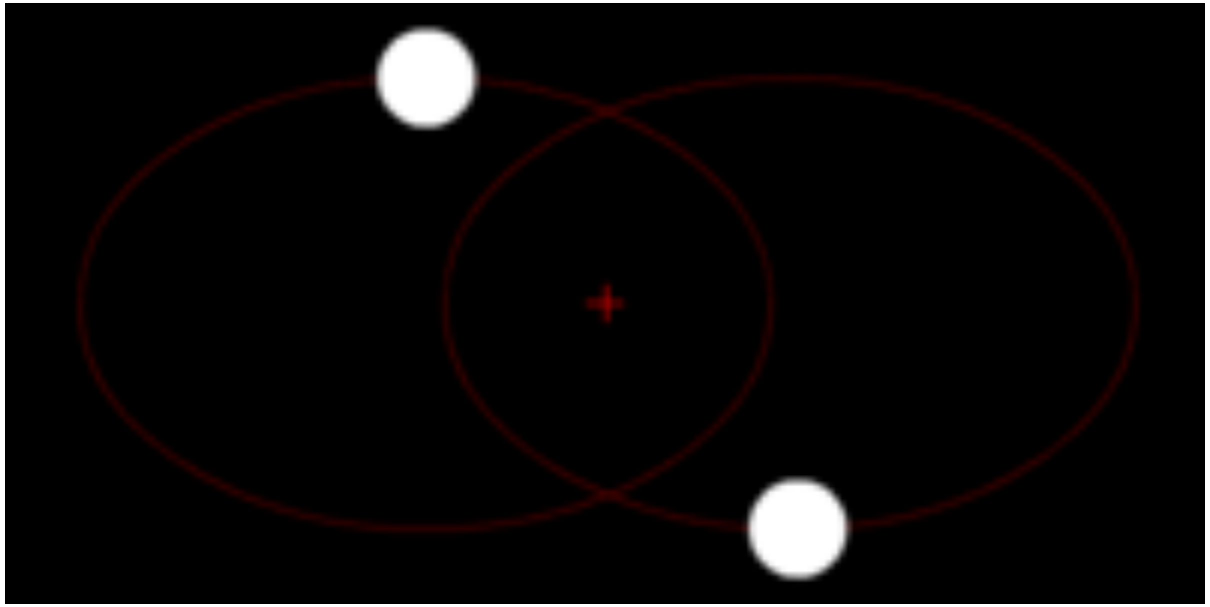
$$\sin\theta = (y_j - y_i)/r_{ij}$$

Coursework - Trisolaris, the three body problem

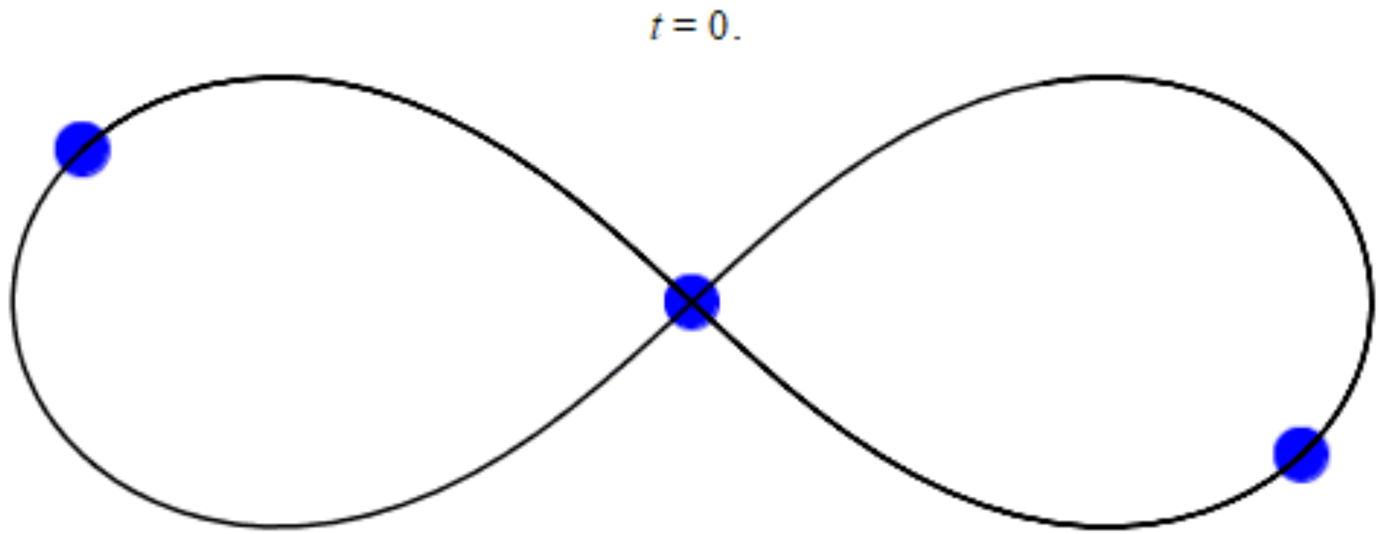
1. ~~Revise Newtonian mechanics~~
2. Discuss problems
3. Discuss strategy
4. Q&A
5. Complete feedback questionnaire

Problem 1: Binary system





Coursework - Problem 2a stable triple

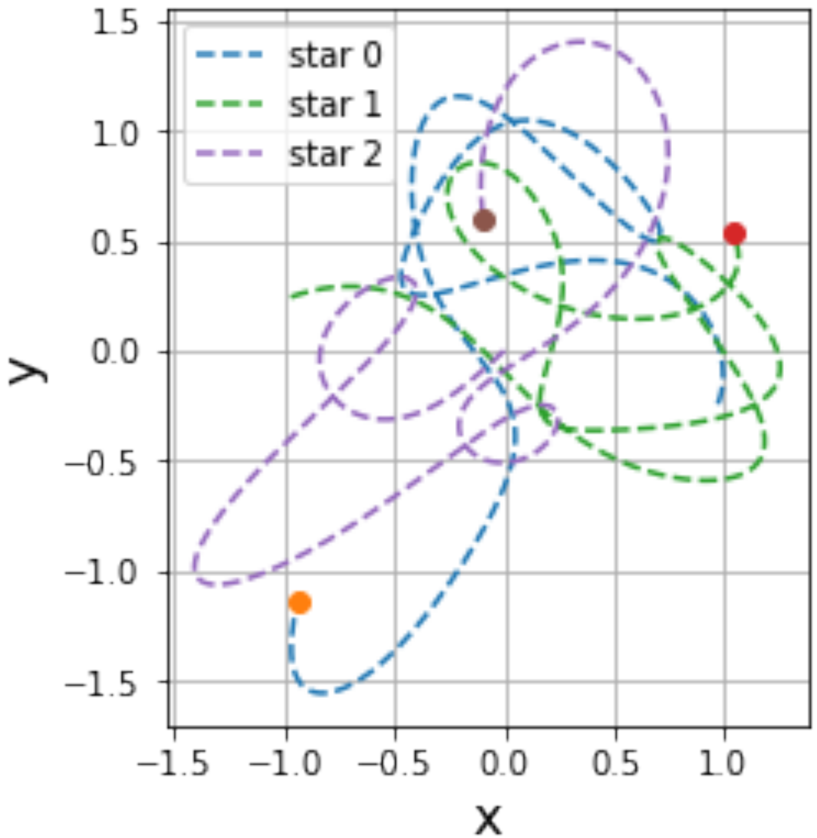


2. Three body problem

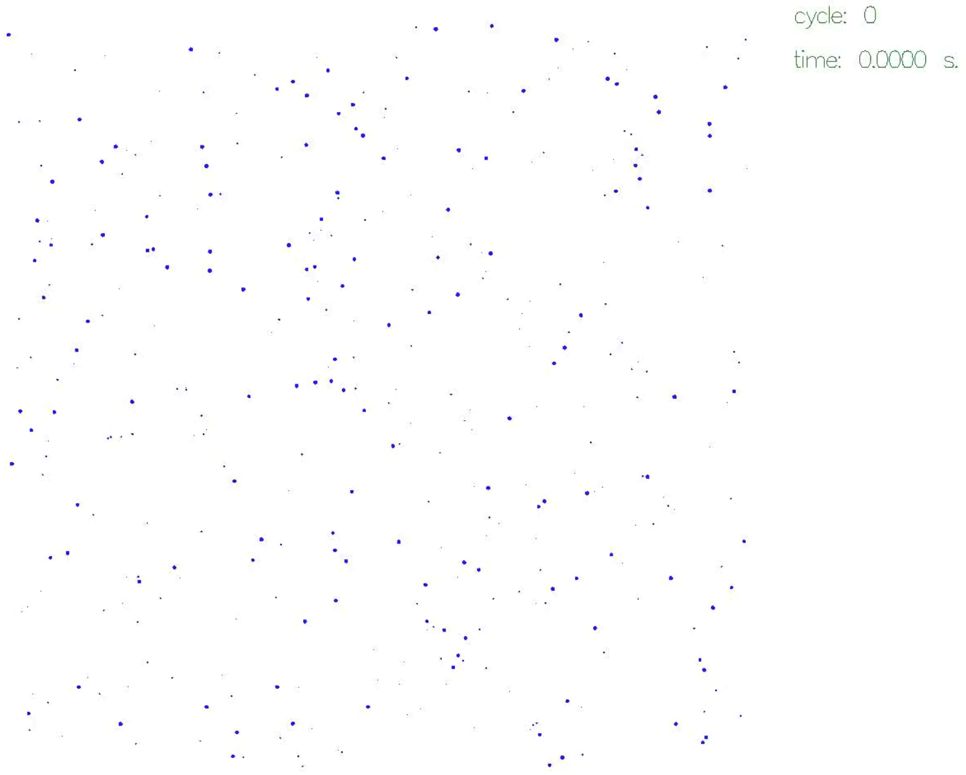
Coursework - Problem 2a stable triple

Raffi discovers a stable octuple system that contains a hidden message (Star Trek: Picard)

Coursework - Problem 2b chaotic triple

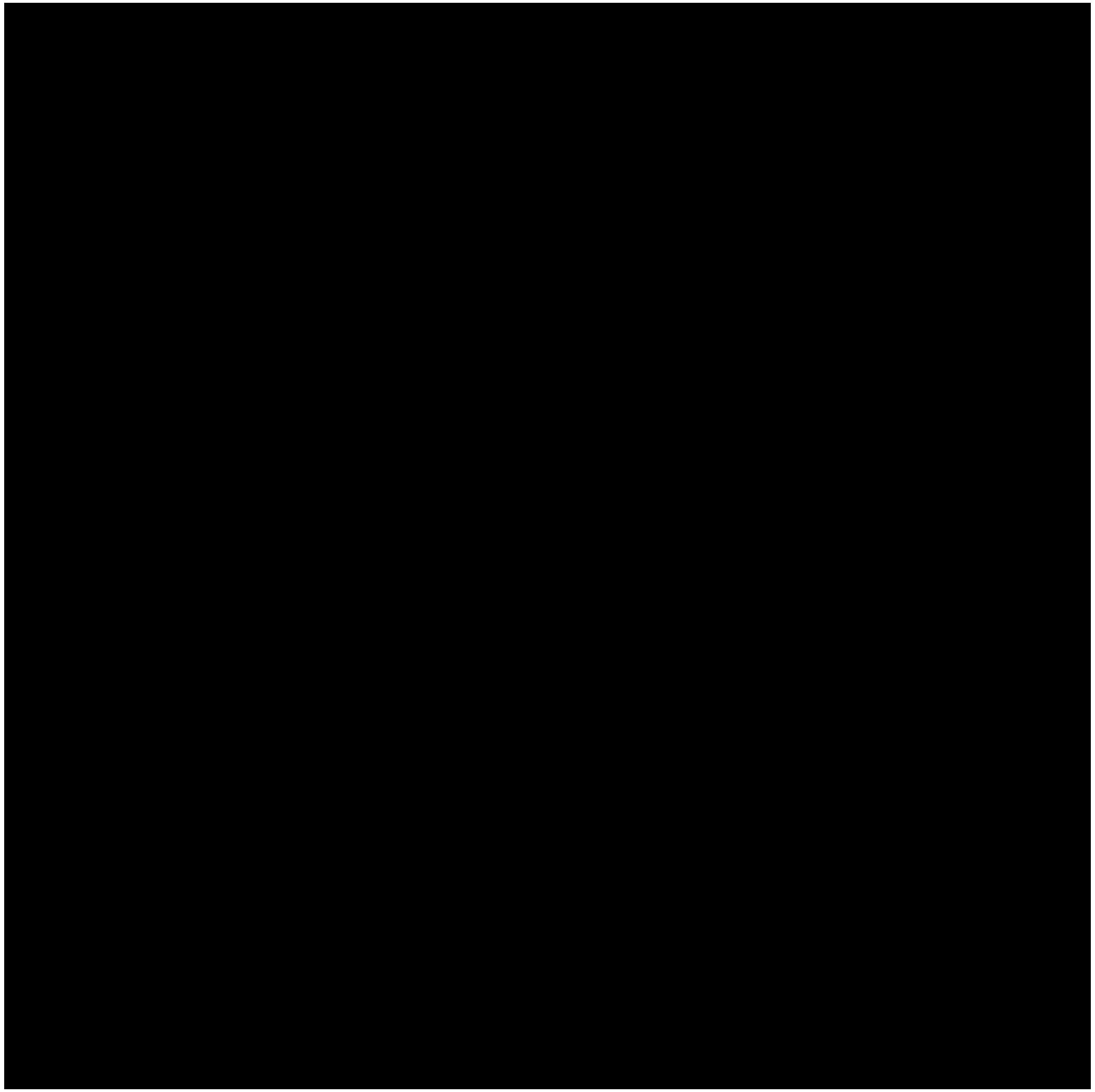


Coursework Problem 3 - merging hypergiants



(You only have to do 3 of these!)

Not in the coursework - build a universe!



Coursework - Trisolaris, the three body problem

1. ~~Revise Newtonian mechanics~~
2. ~~Discuss problems~~
3. Discuss strategy
4. Q&A
5. Complete feedback questionnaire

Coursework - strategy

You have to read the full instructions.
Here I will only highlight the key points.

Coursework - strategy

Marking scheme

The marks available for each section and each question are indicated. Broadly speaking, for each section, the marks available will be split as follows:

- 60% for working code that correctly implements all of the requested components
- 20% for use of defensive programming techniques - asserts and tests implemented to prevent user error and check the code is functioning correctly, including convergence testing.
- 10% for readability of code, following the agreed naming conventions of the course, appropriate commenting
- 10% for appropriate documentation of the code implemented in markdown format around the code.

Key takeaways - make your marker happy

1. Make your marker happy - first impressions count - if I open your file and it contains mess I will be annoyed despite my deep belief in the stoic principle of externals.

2. Intersperse your code with documentation to lead a user through it - target audience is not me as marker, but new student level - would you have been happy if you received your notebook in one of the tutorials? Would you have known what to do with it?

3. Spelling check for markdown cells https://pypi.org/project/jupyterlab-spellchecker/ can be conda installed

4. Make sure you are plotting frequently enough to show smooth curves - remember the role of time vector in solve_ivp() is not related to solution but to outputs

5. Avoid animations - you cannot assume I have installed the same packages as you - if you really must then provide documentation for the user and test it on another system (the lab computers). Always back up with static plots - sometimes animations are not the clearest way to communicate data. There are no extra marks for animation, so better to use your time getting easier marks…

Key takeaways - be strategic

1. **It is unforgivable to lose marks because you have no documentation and no asserts/tests.** This is not about ticking boxes - in any real world job or research position these things will be mandatory, and for good reason. Good researchers and coders do not work in isolation, and need to consider and communicate to others.

2. Check lecture notes for buzz phrases for any “comment on” questions. Say the obvious things! “So we can see that X is bigger than Y, which is what we expected.”

3. Take the hint: If there is a HINT there is probably a mark attached to it

4. If you can’t do something, don’t leave it blank, give me something to work with. Describing a convergence test might give you more marks than struggling with hypergiants, for example.

5. Better to be honest if something isn’t working and explain what you would do to debug it, rather than try to hide it in a confusing plot.

Coursework - strategy

Check the required components for each part

3.1 Problem 1: Binary system (40% of marks)

Model a two body system made up of two stars undergoing multiple stable orbits (which may be elliptic).

Required Components

1. The first star should start at position $[x_1, y_1] = [2.0, 0.0]$ with velocity $\left[\frac{dx1}{dt}, \frac{dy_1}{dt}\right] = [0.0, -0.1]$. Put the second star at $[x_1, y_1] = [-2.0, 0.0]$ and use the expression above to find its initial velocity.
2. The mass ratio of the stars should be $M_1/M_2 = 2$
3. You should write one class to represent stars, and another for star systems
4. Provide plots of the trajectories and a phase diagram for the position and velocity of one of the component stars

NB I prefer that you do not provide animations, as these usually don't work and require additional python packages, but if you do you must also provide static plots. It is an important and useful skill to be able to represent time varying data using static plots.

5. Compare two explicit integration techniques - the first should be scipy's solve_ivp() and the second should be the midpoint method (or another Runge Kutta method of specified order)
6. For the midpoint/Runge Kutta method, provide a convergence test confirming that the solution converges at the required order
7. Include defensive programming techniques such as asserts and tests of key functionality
8. Provide some documentation of the code implemented in markdown

Coursework - strategy

Take the hint(s)

(HINT 1: you may want to investigate the rtol parameter for solve_ivp())

(HINT 2: try to think ahead - what features of problem 1 might you want to reuse in problems 2 and 3? You can save time coding by making it sufficiently general from the start)

Coursework - strategy

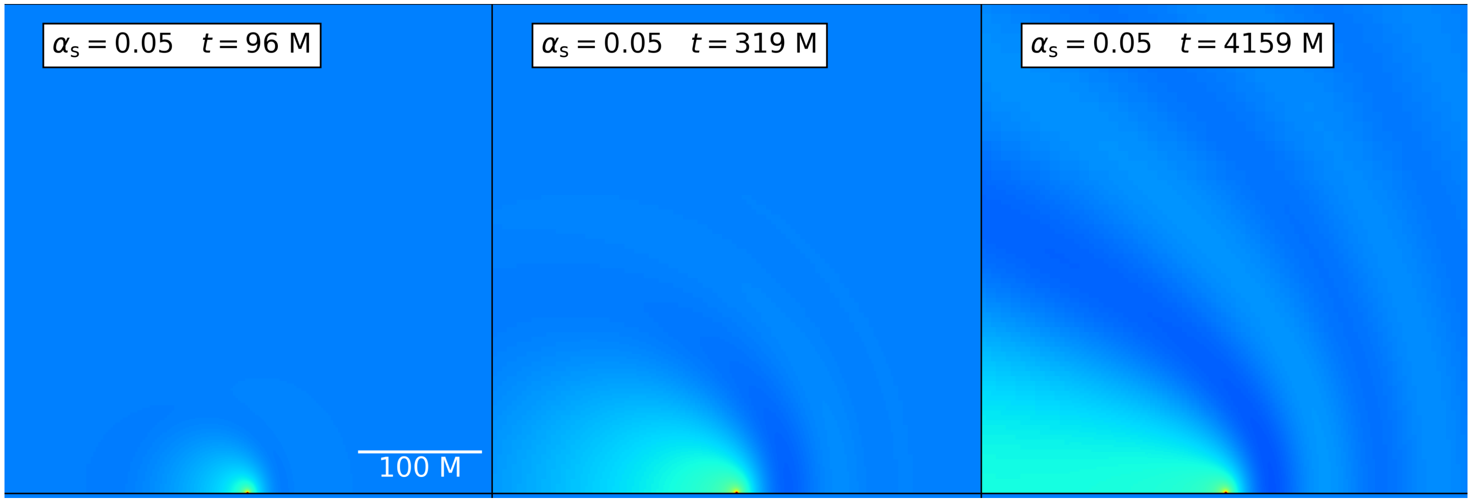
NEVER NEVER NEVER leave a part blank

The project consists of 3 problems, each containing a single goal with multiple component parts. You are expected and encouraged to attempt every problem in this assignment. Fractional marks may be awarded for meaningful partial solutions, for example, a description in words/pseudocode of what you would do to solve the problem, or implement a test. It will always be better to write something rather than nothing, since you can only gain (you won't lose marks for wrong answers).

Coursework - strategy

I will upload an example animation with the Week 4 Tutorial Solutions, but you do not have to use animation in the coursework, in fact I prefer if you don't.

One nice method of seeing evolution is to use a series of snapshots.



Coursework - strategy

Other methods to visualise motion:

How could you give a star a “tail”? Investigate the parameter “alpha” that changes the opacity of the line, noting that it can be a vector…

Coursework - Trisolaris, the three body problem

1. ~~Revise Newtonian mechanics~~
2. ~~Discuss problems~~
3. ~~Discuss strategy~~
4. Q&A
5. Complete feedback questionnaire