# Project 3 FYS4150

Kjetil Karlsen and Vilde Mari Reinertsen

October 27, 2017

**Abstract**

The program used in this project can be found at Github.
Test: Energy conservation, modulus "position" (lengde vektor) bevart Alle vectorer samme str.
Printe + plotte energi stabilitet mellom euler og verlet.
To do: OBS Unit tests
3b: Forklare objektorientering, hvorfor deler kan generaliseres.
3c: Plotte ulike dt-er Plott energi som funk av ulike dt Referere til convergens - funksjonen. Vise at angulærmoment bevart: Oppdatere til L = mr x v i Planet + egen verdi tilhørene hver planet Vise tid ulike algoritmer.
3d: Exact løsning escape vel Plots ulike init.hastigheter Bytte gravitasjonskrefter...
3e- 3body 3 ulike masser Plotte alle banene Stabilitet: Energi-plot
3f Hvordan velge origio???????? IKKE GJORT NOENITING FORELØPIG
3g: IKKE GJORT NOENITING FORELØPIG FLOPS euler/Verlet
Konklusjon: Kunne strukturert annerledes: Solver class - kun euler, verlet Egen klasse for å kjøre algoritmer i (tidsteg, skrive fil)
SPR: is the force, work positive/negative?????? PLAN: Torsdag: Kjetil: Teori, metode, (intro?), Fikse angulærmoment Vilde: Plots, andre resultater (CPU-tid...), (skrive resultater)

## Contents

# 1 Introduction

There are many physical problems that deals with differential equations. A simple example of this is the ordinary differential equations that governs how objects move in relations to each other, for example the solar system. The aim of this project is to model the our solar system using a relatively simple model, relativistic model for the movement of the planets and the sun. In order to do this efficiently to different integration methods will be explored, the Euler method and the Velocity Verlet method.

This report starts by the explaining basic theoretical physics that governs the movement of the objects in the solar system. These equations can be scaled and implemented in a general algorithm for solving 2. order differential equations. The two different integration models will then be discussed alongside how we chose to implement these in our experiment.

The results of benchmark and stability tests of the different methods will be discussed together with more general results of how the planets are moving. This is all wrapped up in a conclusion at the end.

# 2 Theory

## 2.1 Classical Solar system

In the classical description of the solar system, there is only a single force working:

$$F_G = -G\frac{M_1 M_2}{r^2} \tag{1}$$

By applying Newtons 2.law on component form we achieve two more equations, in the case of a two dimensional system.

$$\frac{\mathrm{d}^2\vec{r}}{\mathrm{d}t^2} = \frac{\vec{F}_G}{M_2} \tag{2}$$

Equation 2 is in reality two seperate, independent equations, one for the x-direction and the other for the y-direction.

## 2.2 Units and scaling

A computer has a limited bit-resolution and the distances and timescale are large when computing the solar system. This means that it is important to use appropriate units. The distance between the sun and the earth is defined as 1 Astronomical unit (1 Au) and the timescale used in this project will be in units of 1 year.

For simplicity we will define a new unite of energy: $J_{ast} = \frac{m_{planet}}{|M_{sun}|}\left(\frac{Au}{year}\right)^2$, where $|M_{sun}|$. In reallity this is simply Joule with a prefactor. This prefactor can be obtained by inserting the values into $J_{ast}$:

$$J_{ast} = \frac{m_{planet}}{|M_{sun}|}\left(\frac{Au}{year}\right)^2 \tag{3}$$

$$= \frac{m_{planet}}{2\cdot 10^{30}}\left(|v|\frac{1.5\cdot 10^{11}m}{360*24*60*60s}\right)^2 \tag{4}$$

$$\simeq 1.163\cdot 10^{-23}J \tag{5}$$

The dimensionality of the variables are as follows: $[v] = m/s$ and $[m_{planet}] = kg$.

We know that the earth needs one year to orbit the sun, meaning that $v = \frac{2\pi r}{1\text{ year}}$. This can be rewritten with $v = \tilde{v}v_0$ and $r = \tilde{r}r_0$. The units of $r$ and $v$ are contained in $v_0 = \frac{1\text{Au}}{1\text{ year}}$ and $r_0 = 1$ Au, giving that $\tilde{v}^2\tilde{r} = 4\pi^2$. In the same way $t = \tilde{t}t_0$, with $t_0 = 1$ year.

For the case of the earth - sun system one can assume that the sun is stationary, as $M_{sun} >> M_{earth}$. The force experienced by the earth is thus centrifugal, which means that $a = \frac{v^2}{r}$, with $v = 2\pi r/1year$. Combining this with equations 1 and 2 it is possible to scale the equations in the following manner:

$$a_E = \frac{F_E}{M_E} = G\frac{M_{sun}}{r^2} = \frac{v^2}{r} \tag{6}$$

$$GM_{sun} = v^2 r = 4\pi^2\frac{(1\text{ Au})^3}{(1\text{ year})^2} \tag{7}$$

$$\tag{8}$$

This gives that the dimensionless expression can be stated as:

$$\frac{\mathrm{d}\tilde{v}}{\mathrm{d}\tilde{t}} = \frac{4\pi^2}{\tilde{r}^2} \tag{9}$$

In the two dimensional system $r = (x, y) = (r\cos\theta, r\sin\theta)$. Using the notation $\dot{p} = \frac{\mathrm{d}p}{\mathrm{d}t}$, this gives the following coupled differential equations:

$$\dot{v}_x = \frac{4\pi^2 r\cos\theta}{r^3} = \frac{4\pi^2 x}{r^3} \tag{10}$$

$$\dot{v}_y = -\frac{4\pi^2 r\sin\theta}{r^3} = \frac{4\pi^2 y}{r^3} \tag{11}$$

$$\dot{x} = v_x \tag{12}$$

$$\dot{y} = v_y \tag{13}$$

$$\tag{14}$$

## 2.3 Energy considerations

As with any physical system, the total energy has to be conserved. The potential energy $E_p = \int_{r'}^{\infty}\vec{F}(r')\cdot d\vec{r}' = -\frac{GM_1M_2}{r}$, while the kinetic energy is $E_K = \frac{1}{2}mv^2$. This gives a total energy of:

$$E_{tot} = \frac{1}{2}mv^2 - \frac{GM_1M_2}{r} \tag{15}$$

In addition, the angular momentum ($\vec{L}$) of the system has to be preserved. This is because there are no additional sources of torque ($\vec{\tau}$) once the system has been initialized and $\vec{\tau} = \frac{d\vec{L}}{dt} = 0$. As a result all the absolute value of $\vec{L}$ hast to be constant.

In order to maintain in the gravitational field of another object, the distance between them needs to be smaller than $\infty$. However, with a large enough velocity it is possible to escape. By setting equation 15 equal to 0 one obtains the lowest escape velocity $v_{esc}$:

$$\frac{1}{2}mv_{esc}^2 - \frac{GM_1M_2}{r} = 0 \tag{16}$$

$$v_{esc} = \sqrt{\frac{2GM_1M_2}{mr}} \tag{17}$$

Nevne approksimasjon med barrycentre v sol i sentrum v pos sol @ t=0: Vi har ingen beregninger med sol i sentrum, men bruker sol ved t=0 som posisjon.

Hvorfor bør egentlig være barrycentre til univers?

Can the observed perihelion precession of Mercury be explained by the general theory of relativity?

# 3 Method

In this project we tested two numerical integration techniques, the Euler method and the velocity Verlet method. This section is largely based on Hjort-Jensen [1]

## 3.1 The Euler method

When evaluating the function $x(t)$ in the interval between $t$ and $t+h$ it is natural to do a Taylor expansion:

$$x(t+h) = x(t) + \sum_{n=1}^{\infty} \frac{h^n}{n!}\frac{d^n x(t)}{dt^n} \tag{18}$$

By choosing a small $h$, it is sufficient to truncate the sum after n=1, giving $x(t+h) = x(t) + h\frac{dx}{dt} + O(h^2)$. The term$O(h^2)$ contains the rest of the infinite sum, often called the truncation error. In order for a computer to use this method it is necessary to discretise the expression, substituting $x(t) \to x(t_i) \to x_i$. The Euler method can thus be expressed as

$$x_{i+1} = x_i + h\dot{x}_i + O(h^2) \tag{19}$$

Combining this with equations 10 and 12 we get an algorithm for doing a 1 dimentional integration for the solar system:

$$a_i = F(t_i) \tag{20}$$

$$v_{i+1} = v_i + ha_i \tag{21}$$

$$x_{i+1} = x_i + hx_i \tag{22}$$

**This gives in total 4 floating points operations per iteration, per dimension. For our two dimentional system this results in a total of 8N FLOPS per iteration, with $N = \frac{1}{h}$.**

## 3.2 The Velocity Verlet method

The velocity Verlet can be derived from the same Taylor expansion as the Euler method, equation 18. However, here we will truncate the expansion after $n = 2$ and compare the two steps $x(t \pm h)$. A simple substitution $h \to -h'$ gives the $x_{i-1}$ expression, when compared to the originale Tayler expansion:

$$x_{i+1} = x_i + h\dot{x}_i + \frac{h^2}{2}\ddot{x}_i + O(h^3) \tag{23}$$

$$x_{i-1} = x_i - h\dot{x}_i + \frac{h^2}{2}\ddot{x}_i - O(h^3) \tag{24}$$

When we add $x_{i+1} + x_{i-1}$ we see that the first order expression disappears, together with the truncation error of order $h^3$, resulting in:

$$x_{i+1} = 2x_i - x_{i-1} + h^2\ddot{x} + O(h^4) \tag{25}$$

Verlet velocity method uses forward euler method as a fundament, but introduces a new taylor expansion of the velocity $v'(x)$ .

$$v'(x+h) = v'(x) + hv''(x) + \mathcal{O}(h^2)$$

While discretizing the variables and using the relation $a(x) = v'(x)$, $v_i''$ may be expressed using $a_i$ and $a_{i+1}$.

$$v_i'' = \frac{a_{i+1} - a_i}{h} + \mathcal{O}(h)$$

Inserted into a discretized second order taylor expansion of $v(x)$, an expression for $v_{i+1}$ is achieved.

$$v_{i+1} = v_i + h\frac{a_{i+1} + a_i}{2} + \mathcal{O}(h^3) \tag{26}$$

A second order taylor expansion is also used to approximate $x_{i+1}$ where $x_i' = v_i$ and $x_i'' = a_i$

$$x_{i+1} = x_i + hv_i + \frac{h^2}{2}a_i + \mathcal{O}(h^3) \tag{27}$$

Euler Velocity verlet FLOPS ulike metoder

## 3.3 Choice of origin

As every planet in the solar system is moving, choosing a point of origin is not straight forwards. For the smaller systems, ie. the "sun-earth"-system, we chose to select the suns position at the start as the origin. This allows the sun to move. Another choice of origin is the solar system barycentre, which we utilized when calculating the entire solar system. Using the barrycentre can give a prettier picture of the physics which is (nummerically) unfolding, as every object will rotate around this point.

### 3.4 Unit test

UNIT TESTS ??????

### 3.5 Object orientation

In order to simplify the calculation of an ensemble of planets, each with velocities, positions, energies and angular momentum, it is useful to generalize the code in an object oriented way. We chose to create one class for the planets, where all the internal dynamics (energies, position, velocity, ...) were stored. The forces experienced by the planets are specific for this project and we kept this in the Planet class, so that the second class, the Solver class, could be more general and easier reused.

This Solver class is were all the technicalities are located, including the different integration methods. For each time step one need the location of every planet in the system and Solver includes therefore a function that for each time step loop over all the planets. In order to update their position it is necessary to again loop over all the different planets in order to find the total gravitational force exerted on the current planet.

When all the calculations are taken care of by the solver class and all the properties of the different planets are stored in each planet object. This means that the main part of the program only needs to initialize the different planetary instances , adding these to the a instance of the Solver class which is initialized according to what output we want to achieve, see the snippet below for how this looks in 'main'.

```
        Planet earth("Initializing inputs");
                Planet sun("Initializing inputs");
        Solver verlet("Initializing inputs");
        verlet.add(earth);
        verlet.add(sun);
        verlet.add(mercury);
        verlet.algorithm("input variables");
```

Classes Instanser deklerasjon friend class

## 4 Result

with steps per year: 7*3600*360 Running velocity verlet Perihelion position after 100 years: 0.307498, -0.000933806 Perihelion angle after 100 years: -626.38 arc seconds CPU time: 3248.07

with steps per year: 2*7*3600*360

## 5 Discussion

## 6 Conclusion

## References

[1] Morten Hjorth-Jensen. Computational physics: Lecture notes fall 2015. Department of Physics, University of Oslo, 8 2015. Chapter 2 and 6.