# Investigating the motion of a rocket in orbit

Francis Taylor
*Level 5 Laboratory, School of Physics, University of Bristol.*
(Dated: April 22, 2020)

beans

## INTRODUCTION AND THEORY

A satellite in orbit around a body follows an elliptical path, with the barycenter at one of the foci of the ellipse, as described by Kepler's First Law [1]. This path is defined by the changing velocity of the satellite as it orbits, being lowest when it is at the apoapsis, which is derived from Newton's laws of motion and gravitation. The force on two bodies with mass is given by the latter, being [2]

$$\boldsymbol{F_{12}} = -\frac{Gm_1m_2}{\mid \boldsymbol{r_{12}} \mid^2}\hat{\boldsymbol{r}}, \tag{1}$$

where $\boldsymbol{F_{12}}$ is the force between objects 1 and 2, $G$ is the gravitational constant, $\boldsymbol{r_{12}}$ is the distance between the center of the objects (see figure 1), and $m_1$ and $m_2$ are the masses of the objects.

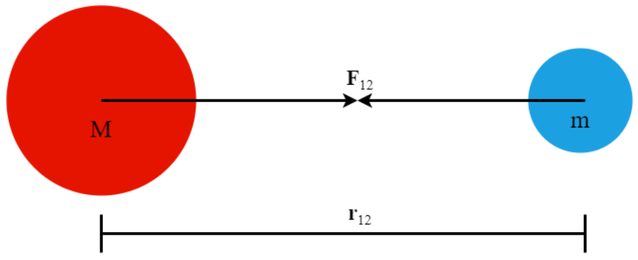

FIG. 1. Newton's Law of Universal Gravitation

From this, the equation of motion for an orbiting satellite can be derived using Newton's second law, giving [2]

$$m\ddot{\boldsymbol{r}} = -\frac{mMG}{\mid \boldsymbol{r} \mid^3}\boldsymbol{r}, \tag{2}$$

where $m$ is the mass of the satellite, $M$ is the mass of the large body, and $r$ is the position of the satellite relative to the centre of the large body. This is an ordinary differential equation, which can be solved using numerical analysis. One solution of this is using the Runge-Kutta family of iterative methods.

### Runge-Kutta Methods

The Runge-Kutta methods are used to provide approximate solutions to ordinary differential equations. Such methods discretise a continuous function in both space and time, allowing for integration over discrete time intervals, a technique easily done using computer simulations [3]. For this simulation, the 4th-order Runge-Kutta (RK4) method was used. This method evaluates the slope of a function with known initial values at four different points in a given interval, shown in figure 2, given by the following [4, 5]

$$k_1 = f(x_n, y_n) \tag{3}$$

$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) \tag{4}$$

$$k_3 = f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \tag{5}$$

$$k_4 = f(x_n + h, y_n + hk_3) \tag{6}$$

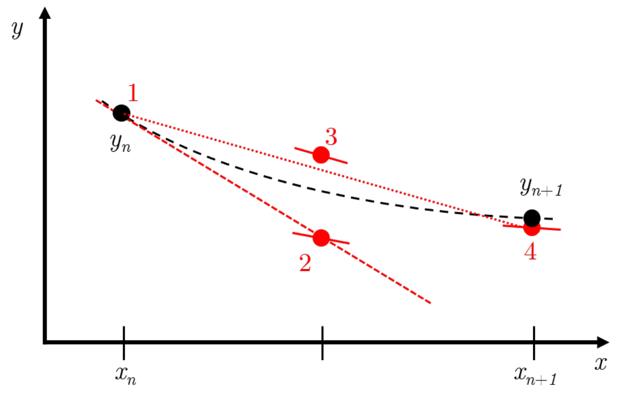

FIG. 2. Slopes used by the 4th-order Runge-Kutta method.

In the context of solving equation 2, $h$ is the time step, and $x$ and $y$ are the positional coordinates of the rocket. $k_1$ is the slope at the beginning of the interval, calculated using $y$. $k_2$ is the slope at the midpoint of the interval, calculated using $y$ and $k_1$. $k_2$ is also the slope at the midpoint of the interval, but is instead calculated using $k_2$ instead of $k_1$. Finally, $k_4$ is the slope at the end of the interval, calculated using $y$ and $k_4$.

The solution to this is then given by the weighted average of the increments [4]

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{7}$$

$$t_{n+1} = t_n + h. \tag{8}$$

As this method is of 4th-order, the global truncation error is of 4th-order, so this method is much more accurate than comparable methods, such as Euler's method, which is 2nd-order [5, 6]. Also of note is that if $f$ is independent of $y$, then RK4 simplifies to Simpson's rule [6].

A two-dimensional orbit with the larger body at the origin requires several variables that need to be evaluated in four functions, as RK4 couples the variables together:

$$f_1(t, xx, y, v_x, v_y) = \frac{dx}{dt} = v_x \tag{9}$$

$$f_2(t, xx, y, v_x, v_y) = \frac{dy}{dt} = v_y \tag{10}$$

$$f_3(t, xx, y, v_x, v_y) = \frac{dv_x}{dt} = \frac{-GMx}{(x^2+y^2)^{3/2}} \tag{11}$$

$$f_4(t, xx, y, v_x, v_y) = \frac{dv_y}{dt} = \frac{-GMy}{(x^2+y^2)^{3/2}} \tag{12}$$

Ignoring the arguments that are not needed, we can apply these functions to each of equations 3-6 to obtain values for $x$, $y$, $v_x$ and $v_y$ at each increment in the interval. Using this, and applying it to equation 7, a series of time-stepping equations is obtained, where $y$ is replaced by each of the variables listed. These can easily be calculated using a computer, by iterating through a loop.

## RESULTS AND DISCUSSION

A circular medium Earth orbit was modelled for 60000s with a time step of 2s, an initial height of 3621km above the Earth's surface and initial velocity of 6313ms$^{-1}$, shown in figure 3. The orbit was stable, completing several revolutions in this time period, and this is also confirmed by the energy graph shown in 4, as the energy remains constant over time. It can be inferred from this that the rocket remained on an gravitational equipotential line, hence it's height above the surface is constant.

An elliptical orbit was also modelled for the same time period and initial height, but with an initial velocity of 7500ms$^{-1}$, shown in figure 5. Again, this orbit was stable, completing several orbits in the time period, and this is also confirmed by figure 6, which shows the energy of the system over time. Several clear periods can be seen, and while the gravitational and kinetic energies oscillate, the total energy remains constant. The periodic nature of the energy is caused by the rocket coming closer and moving away from the Earth, and is a direct consequence of Kepler's Second Law [1]. The magnitude of the kinetic and gravitational energy is highest when the rocket is closest to the Earth, and the least when at apoapsis.

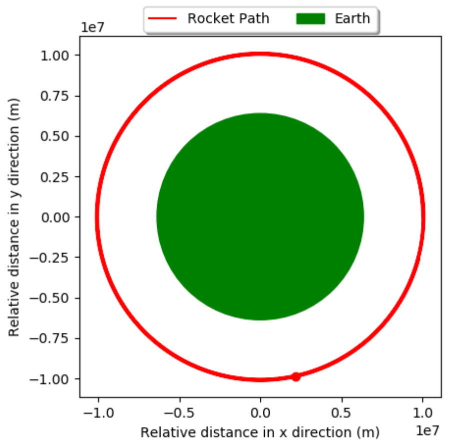

FIG. 3. Position of the orbit of a rocket with initial height of 3621km above the Earth's surface and velocity 6313ms$^{-1}$. The path of the rocket is shown in red, the red dot represents the current position of the rocket, and the Earth is represented by the green circle.

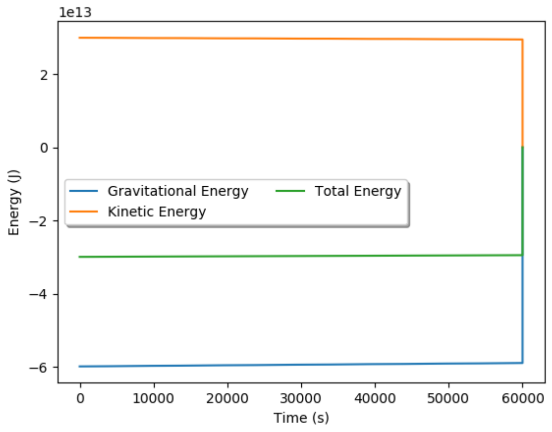

FIG. 4. A plot of energy against time for the circular orbit shown in figure 3. Gravitational energy is shown by the blue line, kinetic energy by the orange line, and total energy by the green line in between.

## IMPROVEMENTS TO CODE

As can be seen in the energy graphs (for example figure 4), at the end of the simulation, all the energy values jump to 0. This may be because there is an extra time value included in the array, where none exists in the energy arrays, but this could be fixed to remove this, by using validation checks to remove extra zeros at the end of the array. As well as this, the animation used for the graphs crashes when the end of the array is reached. I spent several hours trying to debug this, but could not understand why an extra index was being used for $N$, the total number of iterations, when this was the same as the size of the $x$ and $y$ arrays being passed to the function controlling the animation. An improvement would be to ensure that the animation function was being passed the correct number of iterations. Another improvement to the code would be to make the size of the Earth and Moon on the graphs to scale, as often when the rocket crashes, it is displayed as be-

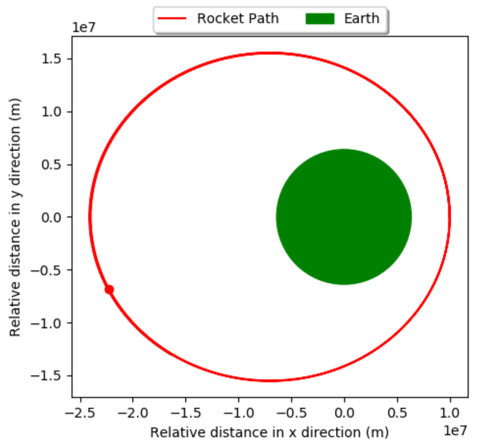

FIG. 5. Position of the orbit of a rocket with initial height of 3621km above the Earth's surface and initial velocity 7500ms$^{-1}$. The path of the rocket is shown in red, the red dot represents the current position of the rocket, and the Earth is represented by the green circle.

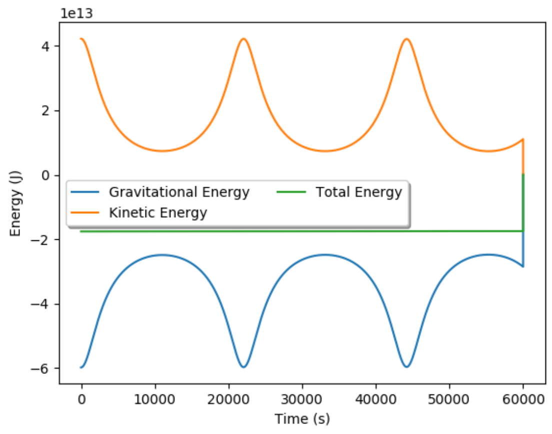

FIG. 6. A plot of energy against time for the circular orbit shown in figure 5. Gravitational energy is shown by the blue line, kinetic energy by the orange line, and total energy by the green line in between.

ing in space, when in actuality it has hit the Moon/Earth, and this isn't displayed very well. To improve the simulation, the equation of motion could be adapted to account for relativistic gravitational effects, however as this is outside my area of knowledge I did not feel comfortable implementing this. One final improvement that could be made would be to move a large amount of the code into more functions, as several sections are reused very often. This is not difficult, and was only restricted by time, but an example of a useful function to help remove repetition would be implementing a function to handle all the Pythagoras, which would make the code easier to read.

## CONCLUSIONS

## APPENDIX

## REFERENCES

[1] J. Kepler and W. H. Donahue, *New Astronomy*, Cambridge University Press, Cambridge; New York, 1992.

[2] I. Newton, A. Motte, and N. W. Chittenden, *Newton's Principia. The Mathematical Principles of Natural Philosophy*, D. Adee, New-York, 1st edition, 1848.

[3] P. L. DeVries and J. E. Hasbun, *A First Course in Computational Physics*, Jones and Bartlett Publishers, Sudbury, Mass., 2nd edition, 2011.

[4] W. H. Press and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge Univ. Press, Cambridge, 3rd edition, 2007.

[5] K. E. Atkinson, *An Introduction to Numerical Analysis*, John Wiley & Sons, New York, 2nd edition, 1989.

[6] E. Süli and D. F. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, Cambridge, 2003.