On the Mechanics of Planetary Orbits

Vincent Razo

California State University, Fullerton

Abstract

In this paper, we will model, study, and analyze the orbit of a planet around a sun. We will consider measurements for the Earth and the sun, then apply Runge-Kutta methods to plot the orbits. We would then like to study the stability of our model in terms of our parameters.

1 Introduction

In this project, we will consider the work of Isaac Newton along with Johannes Kepler, Leonhard Euler, and Joseph-Luis Lagrange. We would like to derive a model for the motion of a planet around a sun using classic physics and Euler-Lagrange calculus.

For this system, we consider a two-body problem where M is the big mass and m is our little mass with a position vector $\vec{r}$ relative to M. Consider the Cartesian xy-plane for our basis and place M at the origin. For planetary orbits, we consider only the xy-plane and not 3-dimensions because planets exist in the ecliptic plane. There are many ways to show this, and if the reader is interested a proof will be included in the Appendix.

Using Newton's Laws of Motion[4], Kepler's Laws[4], and Lagrange Mechanics[1], we can derive two equations for the change in the angle and the change in the radius of m. If the reader is interested in the derivation of these equations, the proof will be included in the Appendix. After many steps applying our classical physics laws, we derive the two equations:

$$\dot{\theta} = \frac{\ell}{mr^2}. \tag{1}$$

$$\ddot{r} = \frac{\ell^2}{m^2r^3} - \frac{GM}{r^2}, \tag{2}$$

where G is the gravitational constant.

Finally, we have obtained two equations modeling the motion of our planet m. We can now analyze these equations using methods from numerical analysis. For the start of this project, I hypothesize that we will obtain at least one stable critical point and that the orbits will look elliptical. I arrived at this hypothesis because that is what scientists have observed about planetary orbits.

2 Methods

There are two specific methods of numerical analysis that we will utilize for analyzing our system of equations. We will use the Runge-Kutta method[3] to obtain solution vectors for our system. We will then apply stability analysis[2] to determine if our system is stable and what the stability conditions will be.

In order to apply Runge-Kutta to our system of equations, we must first convert our second-order differential equation to a system of first-order differential equations[3]. Here, let $r = u_1$, $\dot{u_1} = u_2$, and $\dot{u_2} = \frac{\ell^2}{m^2u_1^3} - \frac{GM}{u_1^2}$. Also, let $\dot{\theta} = \frac{\ell}{mu_1^2}$. Then, we can apply Runge-Kutta to $\dot{\theta}, \dot{u_1}$, and $\dot{u_2}$ in order to obtain solution vectors to r and θ. Using those solution vectors we can convert to Cartesian coordinates using the formulas $x = rcos(\theta)$ and $y = rsin(\theta)$. Examples using Runge-kutta will be discussed later.

Next, we would like to analyze the stability of our equations. Consider our new system of equations:

$$\begin{aligned}
\dot{\theta} &= \frac{\ell}{mu_1^2} \\
\dot{u_1} &= u_2 \\
\dot{u_2} &= \frac{\ell^2}{m^2u_1^3} - \frac{GM}{u_1^2}
\end{aligned}$$

We will carry out our analysis in 5 steps. First, we find the critical points of our system. Second, we compute the Jacobian. Third, compute the eigenvalues of the Jacobian. Fourth, we substitute for our critical points. Lastly, we classify the stability at each critical point.

To find the critical points of our system, we must set each of our equations equal to zero and solve for our variables. What variables do we want to solve for? Well, since we have 3 differential equations for 3 variables, we would like to solve for those variables. That is, we solve for (θ, u_1, u_2). Now we analyze

$$0 = \frac{\ell}{mu_1^2} \tag{3}$$

$$0 = u_2 \tag{4}$$

$$0 = \frac{\ell^2}{m^2u_1^3} - \frac{GM}{u_1^2}. \tag{5}$$

From 3, we cannot obtain any information for our critical point. From 4, we can see that $u_2 = 0$ gives one coordinate for our critical point. From 5, we can apply algebra to the system and solve for u_1 to obtain the coordinate $u_1 = \frac{\ell^2}{m^2GM}$. Since our system is independent of the angle θ, we can take θ to be arbitrary. For simplicity, we will choose θ to be zero.

Next, we compute the Jacobian. The Jacobian matrix for a system of 3 equations is defined as

$$J = \begin{bmatrix} \frac{d\dot{\theta}}{d\theta} & \frac{d\dot{\theta}}{du_1} & \frac{d\dot{\theta}}{du_2} \\ \frac{d\dot{u_1}}{d\theta} & \frac{d\dot{u_1}}{du_1} & \frac{d\dot{u_1}}{du_2} \\ \frac{d\dot{u_2}}{d\theta} & \frac{d\dot{u_2}}{du_1} & \frac{d\dot{u_2}}{du_2} \end{bmatrix} = \begin{bmatrix} 0 & \frac{-2\ell}{mu_1^3} & 0 \\ 0 & 0 & 1 \\ 0 & \frac{-3\ell^2}{m^2u_1^4} + \frac{2GM}{u_1^3} & 0 \end{bmatrix} \tag{6}$$

Now, we need to find the eigenvalues of the Jacobian. To do this, we find the characteristic polynomial of the Jacobian matrix and solve for λ. We solve for the equation $det(J - \lambda I) = 0$ to obtain the equation

$$0 = \lambda(\lambda^2 + \frac{3\ell^2}{m^2 u_1^4} - \frac{2GM}{u_1^3}) \tag{7}$$

For the fourth step in our stability analysis, we substitute $u_1 = \frac{\ell^2}{m^2 GM}$ into 7. Then we solve for λ,

$$\lambda_1 = 0, \quad \lambda_{2,3} = \pm \frac{G^2 M^2 m^2}{\ell^3}\sqrt{3 - 2GM} \tag{8}$$

From $\lambda_1 = 0$, we cannot obtain any information for the stability of our system. In $\lambda_{2,3}$, we note that for the sun, $GM \approx 1e20$. Then from the square root, we obtain two imaginary eigenvalues with no real parts. That is $\lambda_2 = i\mu$, $\lambda_3 = -i\mu$ where $\mu = \frac{G^2 M^2 m^2 i}{\ell^3}\sqrt{3 - 2GM}$. In our book for stability analysis[2], these two eigenvalues give us a stable center node. In the next section, we would like to illustrate our analysis with examples.

3 Example with Code

Here, we include our first example, Figure 1. We want to plot the Earth's orbit with real parameters for one Earth Year(AU). We can see that the orbit of Earth looks almost circular. However, the maximum and minimum position vectors of the orbits are approximately 1.55×10^{11} and 1.5×10^{11}, respectively. When analyzing the plot, it is hard to tell the difference between the minimum and maximum points of our ellipse. (The code for Runge-Kutta will be included in the Appendix)

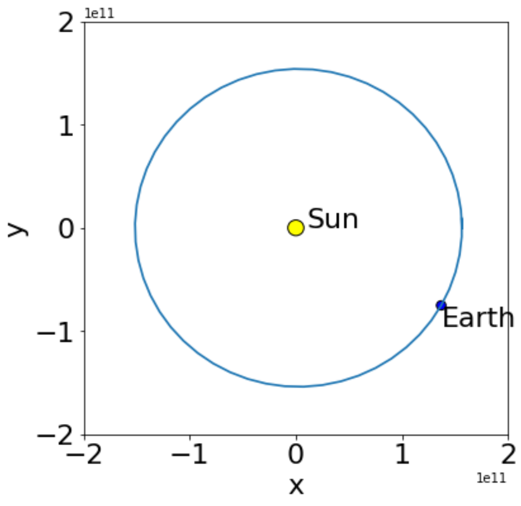

Figure 1: Solution orbit of the earth and the sun using RKM04 and real measurements for our variables(ℓ, G, M, m)

For our next example, we would like to demonstrate how our system acts at and around our critical point. Remember, for our critical point we have $u_2 = 0$ and $u_1 = \frac{\ell^2}{m^2 GM}$. Substituting real-life values for Earth and the sun into u_1, we obtain the critical point $u_1 \approx 1.52 \times 10^{11}$. We will illustrate the dynamics of our system around the critical point with two plots. For both plots, we create an array of initial values for u_1 around our critical point. We also have our initial value $u_2 = 0$. Then, we run a loop for each initial value in our array and plot the resulting solution vectors. This can be seen in Figure 2 a.) b.)

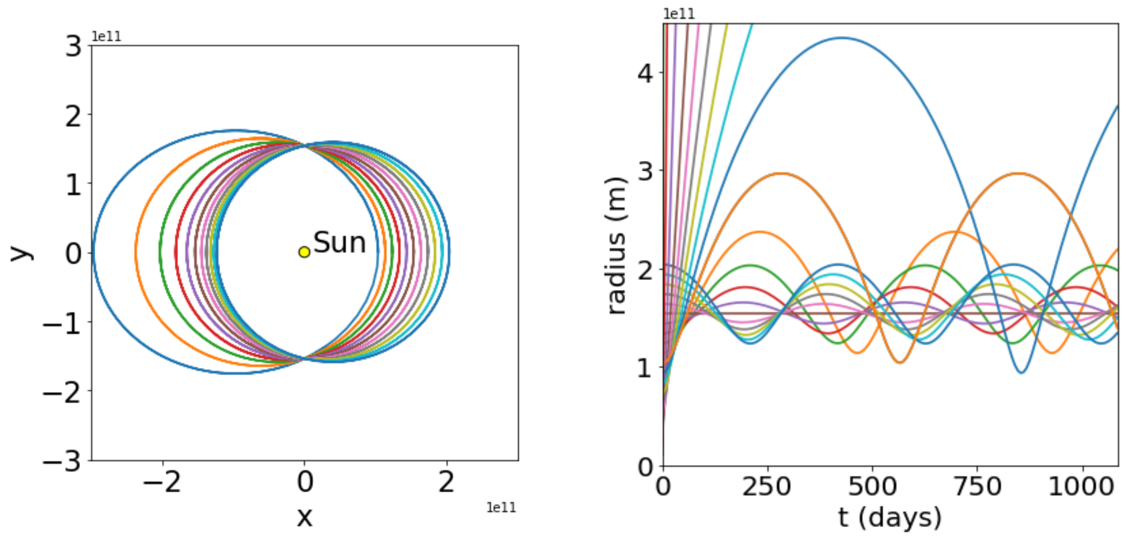

Figure 2: a.) "Phase plane". Solution trajectories are converted to Cartesian Coordinates b.) Time Course Plot. Solution trajectories are plotted over $3AU$.

In Figure 2 we note the critical point for u_1 as the brown trajectory. We can see that in the time course, the radius does not change. For the "Phase Plane," the brown trajectory represents a perfect circle. That is, for the "Phase Plane" there is no change in the radius but we still have a change in the angle. We also note that as we go away from the critical point in either direction, the radius can be seen as waves in the time course plot. For the Phase Plane, we can see that the trajectories become elliptical.

4 Discussion

After deriving a model for the planetary orbit of one body about a larger body, we analyzed our system and arrived at some fascinating results. As hypothesized, we can conclude that our system of equations gives us exactly one stable critical point and elliptical motion for our body. We came to this conclusion with the use of numerical solvers and stability analysis. Finally, we illustrated these results using examples.

References

[1] Bentley, V. *Lagrangian Planetary Orbits* [Video]. Youtube. Published July 4, 2019. https://www.youtube.com/watch?v=EYD79CkdYIQ&t=361s.

[2] Boyce W.E., DiPrima R.C. *Elementary Differential Equations and Boundary Value Problems*. 11th edition. Wiley; 2012.

[3] Faires J.D., Burden R.L. *Numerical Methods*. 4th ed. Cengage Learning; 2012.

[4] Young H.D., Freedman R.A. *University Physics with Modern Physics*. 15th ed. Pearson; 2019.

5 Appendix

Here, we include a proof deriving the formulas for our system.

Proof. In [4] we know that m has a force vector, due to gravity, pointing in the opposing direction of the position vector $\vec{r}$. We also know that the little mass m has a velocity vector $\vec{v}$ with an arbitrary direction in the xy-plane. The velocity vector will have a radial portion and an angular portion. Now, we derive our equations.

In [4] we know that the potential energy for m is given by $PE = -\frac{GMm}{r}$, where G is the gravitational constant, and the kinetic energy for m is given as $KE = \frac{1}{2}mv^2$. Since $\vec{v}$ has a radial and angular portion, we will have to apply the chain rule to our position vector to obtain our velocity in terms of r and θ. That is, $KE = \frac{1}{2}m(\dot{r} + r\dot{\theta})^2$,where $\dot{}$ represents $\frac{d}{dt}$.

Now, using Lagrange mechanics[1] we know that the the Lagrange for m is given as $L = KE + PE$. Substituting for kinetic and potential energy,

$$L = \frac{1}{2}m(\dot{r} + r\dot{\theta})^2 + \frac{GMm}{r} \tag{9}$$

Consider the Euler-Lagrange calculus of variations equation[1] for $(t, r, \dot{r})$:

$$\frac{dL}{dr} = \frac{d}{dt}(\frac{dL}{d\dot{r}}). \tag{10}$$

Taking partials of 9 and substituting into 10, we obtain the equation $mr\dot{\theta}^2 - \frac{GMm}{r^2} = \frac{d}{dt}(\frac{1}{2}m\dot{r}^2)$. Next, we apply the derivative and solve for $\ddot{r}$ to obtain

$$\ddot{r} = r\dot{\theta} - \frac{GM}{r^2}. \tag{11}$$

We now have our equation modeling the change of our radial vector. Now, we need to obtain our equation for the change in the angle. Consider again the Euler-Lagrange calculus of variations equation for $(t, \theta, \dot{\theta})$: $\frac{dL}{d\theta} = \frac{d}{dt}(\frac{dL}{d\dot{\theta}})$. Taking the partials of 9, we obtain $0 = \frac{d}{dt}(mr^2\dot{\theta})$. Instead of applying the derivative, we integrate on both sides to obtain $\ell = mr^2\dot{\theta}$. By the properties of integral calculus, we know ℓ must be a constant.

Note that in Kepler's laws, we know the angular momentum is equal to the change in angle by the moment of inertia, where the inertia of m is given by mr^2. Then, we can see from our equation that ℓ is the angular momentum of m. Solving for $\dot{\theta}$, we obtain our equation

$$\dot{\theta} = \frac{\ell}{mr^2}. \tag{12}$$

Q.E.D.

All code will be included here.

```
import numpy as np

#intialize variables for Sun and Earth
a = 0
b = 3.335e7
```

```
years = 3
n = years*52
h = 604800
G = 6.6743*10**(-11)
M = 1.9891*10**(30)
m = 5.9722*10**(24)
el = 2.7*10**(40)
rmin = 147.1*10**(9)
rmax = 152.1*10**(9)
theta = 0
t = np.arange(0,years*365-7,7)

#initialize symbols
u1 = sym.symbols('u1')
u2 = sym.symbols('u2')

#initialize functions
f = el/(m*u1**2)                                        #dtheta/dt
du1 = u2                                                #dr/dt
du2 = (el**2/(m**2*u1**3)) - G*M/(u1**2)

#Runge-Kutta function
def RungeKutta(n,h,r,rdot,theta,f,du1,du2):
    u = np.array([[r],[rdot]])
    the = np.array([theta])
    x = np.array([r*np.cos(theta)])
    y = np.array([r*np.sin(theta)])

    for i in np.arange(0,n-1):
        k11 = h*du1.subs(u2,u[1,i])
        k12 = h*du2.subs(u1,u[0,i])
        k13 = h*f.subs(u1,u[0,i])

        k21 = h*du1.subs(u2,u[1,i]+k12/2)
        k22 = h*du2.subs(u1,u[0,i]+k11/2)
        k23 = h*f.subs(u1,u[0,i]+k11/2)

        k31 = h*du1.subs(u2,u[1,i]+k22/2)
        k32 = h*du2.subs(u1,u[0,i]+k21/2)
        k33 = h*f.subs(u1,u[0,i]+k21/2)

        k41 = h*du1.subs(u2,u[1,i]+k32)
        k42 = h*du2.subs(u1,u[0,i]+k31)
        k43 = h*f.subs(u1,u[0,i]+k31)

        w1 = u[0,i]+(1/6)*(k11+2*k21+2*k31+k41)
        w2 = u[1,i]+(1/6)*(k12+2*k22+2*k32+k42)
        u = np.hstack((u,np.array([[w1],[w2]])))
        the = np.append(the,the[i]+(1/6)*(k13+2*k23+2*k33+k43))
        x = np.append(x,[sym.cos(the[i+1])*u[0,i+1]])
        y = np.append(y,[sym.sin(the[i+1])*u[0,i+1]])

    return(u,the,x,y)

#find the C.P for u1:
u1CP = sym.solve(du2,u1)
u1CP = float(u1CP[0])

#Initialize interval around C.P.
kstep = 1e10
u1interval1 = np.arange(u1CP-5*kstep,u1CP+6*kstep,kstep)
zerointerval = np.linspace(0,u1CP-5*kstep,np.size(u1interval1))
uinterval = np.vstack((u1interval1,zerointerval))

#Earth Orbit Plot
rad,angle,x,y = RungeKutta(n+4,h,r,rdot,thet,f,du1,du2)
plt.figure(2,figsize=(6,6))
plt.plot(x,y, lw = 1.7)

#Phase Plane Plot
plt.figure(2,figsize=(6,6))

for k in np.arange(0,np.size(u1interval1)):

    rdot = 0
    r = u1interval1[k]
    thet = 0
    rad,angle,x,y = RungeKutta(n,h,r,rdot,thet,f,du1,du2)
    plt.plot(x,y, lw = 1.7)

#Time Course Plot
plt.figure(2,figsize=(6,6))

for i in np.arange(0,2):
    for k in np.arange(0,np.size(u1interval1)):

        rdot = 0
        r = uinterval[i,k]
        thet = 0
        rad,angle,x,y = RungeKutta(n,h,r,rdot,thet,f,du1,du2)
        plt.plot(t,rad[0,:], lw = 1.7)
```