Tristan Zook

Computational Mechanics

Mighty Big Problem Write Up

11/16/2023

Mighty Big Problem Write Up

Part A:

In part A, we'd like to simulate Mercury's orbit around the Sun as a two-body central force system. In order to do this, we need to start with Newton's Second Law applied to Mercury, which yields $\Sigma\overrightarrow{F_m} = M_m * \overrightarrow{a_m}$ where $\overrightarrow{F_m}$ is the force on Mercury, M_m is Mercury's mass, and $\vec{a}$ will be the second time derivative of the position vector $\vec{r}$. So, we have $\Sigma\overrightarrow{F_m} = M_m * \frac{d^2\vec{r}}{dt^2}$. Now we must look to Newton's universal law of gravitation. Newton's law of universal gravitation in general states

$$\overrightarrow{F_{m1m2}} = -\frac{G*m_1*m_2}{|r|^2} * \hat{r} \qquad (i)$$

Where the m1 and m2 are two bodies, and the each exerts a force on the other that is equal to the right-hand side, where $|r|$ is the magnitude of the vector between the two, ie the distance, and $\hat{r}$ points the from the body acting to the body being acted on, ie the opposite direction of the force. This law applied to Mercury and the Sun results in the equation

$$\overrightarrow{F_{sm}} = -\frac{G*M_m*M_S}{|r|^2} * \hat{r}$$

In this case, $\overrightarrow{F_{sm}}$ is the force of the Sun on Mercury, G is the gravitational constant, M_s is the mass of the Sun, and $|r|^2$ is the magnitude of the position vector of Mercury r. Before moving on, we first must convert G into units that align with the data we are provided. We are provided data in AU's and yrs. G's units are $\frac{m^3}{kg\ s^2}$. So, say we have w equal to the amount of meters in and

AU, and q equal to the amount of seconds in a year, then we must take $G * \frac{q^2}{w^3}$ to convert G to units of $\frac{AU^3}{kg\ yrs^2}$. Now, back to Newton's law of universal gravitation. We are looking at

$$\overrightarrow{F_{sm}} = -\frac{G*M_m*M_s}{|r|^2} * \hat{r} \qquad (1)$$

Typically, our denominator here would be the distance between the bodies we are considering in our gravitational force. But here we set the Sun to be the origin of our coordinate system, so we have this distance being equal to the magnitude of the position vector of Mercury. And finally, $\hat{r}$ is the unit vector that points from the Sun to Mercury. Our force is therefore negative because it acts in the opposite direction of $\hat{r}$, as the the Sun is pulling Mercury towards it. We now want to get this equation in to something we can work with. To do this, we must determine more concretely what $\hat{r}$ is. Recall that a unit vector in the direction of any given vector A is defined as $\frac{\vec{A}}{|A|}$. So, using this definition, we can conclude that $\hat{r} = \frac{\vec{r}}{|r|}$. Substituting this in to Newton's law of universal gravitation for the Sun and Mercury we get

$$\overrightarrow{F_{sm}} = -\frac{G * M_m * M_s}{|r|^2} * \hat{r} \ \rightarrow \overrightarrow{F_{sm}} = -\frac{G * M_m * M_s}{|r|^2} * \frac{\vec{r}}{|r|} \rightarrow \overrightarrow{F_{sm}} = -G * M_m * M_s * \frac{\vec{r}}{|r|^3}$$

$$\overrightarrow{F_{sm}} = -G * M_m * M_s * \frac{\vec{r}}{|r|^3} \qquad (2)$$

Now, since the Sun is the only body acting on Mars, we can say $\overrightarrow{F_{sm}} = \Sigma\overrightarrow{F_m}$. So we now have two equations for $\overrightarrow{F_m}$. The next step is setting these two equations equal to each other. Doing this yields

$$M_m * \frac{d^2\vec{r}}{dt^2} = -G * M_m * M_s * \frac{\vec{r}}{|r|^3}$$

Right away we can see that the mass of Mercury can be divided across on both sides of the equation. This yields

$$\frac{d^2\vec{r}}{dt^2} = -G * M_s * \frac{\vec{r}}{|r|^3} \qquad (3)$$

This is our now simplified differential equation that describes the motion of Mercury about the Sun with no other planets involved. However, this equation is currently in vector form. We'd like to break it in to its components. What we can do here, and throughout the rest of our investigation is approximate the Sun and all the planets to be in the same plane. This allows us to ignore the z direction, and only worry about the x and y coordinates of our position vectors. So, we'd like to break our current differential equation in to two separate equations, one for each the x and y components of r. We can ignore the z coordinate here, as we assume that Mercury and the Sun exist within the same plane, thus having the same z coordinate. We make this same assumption throughout the entirety of this problem. Recall here we are in cartesian, so our vector $\vec{r} = x_m\hat{\imath} + y_m\hat{\jmath}$, where x_m and y_m are the x and y coordinates for Mercury respectively. Both $\hat{\imath}$ and $\hat{\jmath}$ are unchanging in time, so the when we take $\frac{d^2\vec{r}}{dt^2}$ we get

$$\frac{d^2\vec{r}}{dt^2} = \frac{d^2x_m}{dt^2}\hat{\imath} + \frac{d^2y_m}{dt^2}\hat{\jmath} \qquad (4)$$

We'll use this information to break our equation up in to component form. Moving forward, I will not explicitly write $\hat{\imath}$ or $\hat{\jmath}$, understanding that equations for the x and y components will be in the $\hat{\imath}$ and $\hat{\jmath}$ direction respectively. Also note that $|r| = \sqrt{x_m^2 + y_m^2}$, So using what we've derived above, our two equations that follow from breaking our equation of motion in to component form are

$$\frac{d^2x_m}{dt^2} = -G * M_s * \frac{x_m}{(\sqrt{x_m^2+y_m^2})^3} \qquad (5)$$

for the x component, and

$$\frac{d^2y_m}{dt^2} = -G * Ms * \frac{y_m}{(\sqrt{x_m{}^2+y_m^2})^3} \qquad (6)$$

for the y component. We now have a system of second order ODEs. We'd like to solve them using odeint in python. However, odeint requires first order ODEs. So we must break our two equations for the x and y components down further in to two sets of coupled equations for both components. For the x component, our coupled equations become

$$\frac{dvx_m}{dt} = -G * M_s * \frac{x_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} \;\&\&\; \frac{dx_m}{dt} = vx_m$$

where vx_m is the velocity in the x direction for Mercury. For the y component, we get

$$\frac{dvy_m}{dt} = -G * M_s * \frac{y_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} \;\&\&\; \frac{dy_m}{dt} = vy_m$$

where vy_m is the velocity in the y direction for Mercury.

We now have a couple equation describing Mercury's orbit about the Sun without other planets, which we can solve using odeint. However, before plugging this in to odeint we must also find our initial conditions that describe this system. To start doing this, we must first make some clarifications about our coordinate system. As mentioned before, we define the Sun to exist at the origin of our coordinate system. To make finding our initial conditions simple, we'd like to pick our initial conditions such that the x axis passes through the perihelion. This will result in our initial conditions for both y_{m0} and vx_{m0}, initial y position and x-direction velocity respectively, to both be zero. This is because y is zero along the x axis, and at any point of the orbit that falls on the x axis will have all its velocity going in the y direction. So with this we just need to find x_{m0} and vy_{m0}. Since we chose our perihelion to be on our x-axis, we have our initial x_{m0} being equal to the perihelion. We know that for an elliptical orbit, the perihelion is described by the equation

$$p = a(1 - \epsilon) \qquad (7)$$

where a is the semimajor radius of orbit, which we have in AU's, and ϵ is the eccentricity of the orbit. So we set $x_{m0} = p = a(1 - \epsilon)$. Now we must find the initial y velocity, vy_{m0}. To do this, we can use an equation describing angular momentum that follows from conservation of angular momentum in orbit about a central force, namely the equation

$$L = m(x * vy - y * vx) \qquad (8)$$

Applying this equation to our case, we have m equal to the mass of mercury, and the term y*vx=0 due to how we have set up our initial conditions. So at the initial point chosen we have

$$L = M_m(x_{m0} \ * vy_{m0}\ )$$

We can use this, along with another equation that describes the perihelion of an elliptical orbit, namely

$$p = \frac{L^2}{G*Ms*(Mm)^2} * \frac{1}{1+\epsilon}$$

We can now substitute our equation for L in to this equation which yields

$$p = \frac{\left(M_m(x_{m0} \ * vy_{m0}\ )\right)^2}{G * M_s * (M_m)^2} * \frac{1}{1 + \epsilon}$$

We also know how we've defined our coordinate system that $x_{m0}\ p$. So, plugging this in our equation becomes

$$p = \frac{\left(M_m(p*vy_{m0}\ )\right)^2}{G*Ms*(M_m)^2} * \frac{1}{1+\epsilon} \rightarrow p = \frac{(M_m^2*p^2*vy_{m0}^2)}{G*Ms*(M_m)^2} * \frac{1}{1+\epsilon} \rightarrow * (1 + \epsilon)(G * M_s) * p = p^2 vy_{m0}^2 \rightarrow$$

$$\frac{(1+\epsilon)(G*M_s)}{p} = vy_{m0}^2 \rightarrow \sqrt{\frac{(1+\epsilon)(G*M_s)}{p}} = vy_{m0}$$

We can then calculate the value of the perihelion using our equation $p = a(1 - \epsilon)$, plug it in to our equation we've derived for vy_{m0}, and allow python to do these calculations for us. So we've found our initial conditions. We have,

$$x_{m0} = p,\ y_{m0} = 0, vx_{m0} = 0, vy_{m0} = \sqrt{\frac{(1+\epsilon)(G*M_S)}{p}}.$$

We can use python to calculate these values explicitly. When doing this we find that, $p \approx .3074$ and $vym0 \approx 12.4327$ Now that we have our system of equations and its initial conditions, we can move on to solving it.

Solving this system of coupled equations using odeint and plotting our result yields us the plot seen in Fig 1.

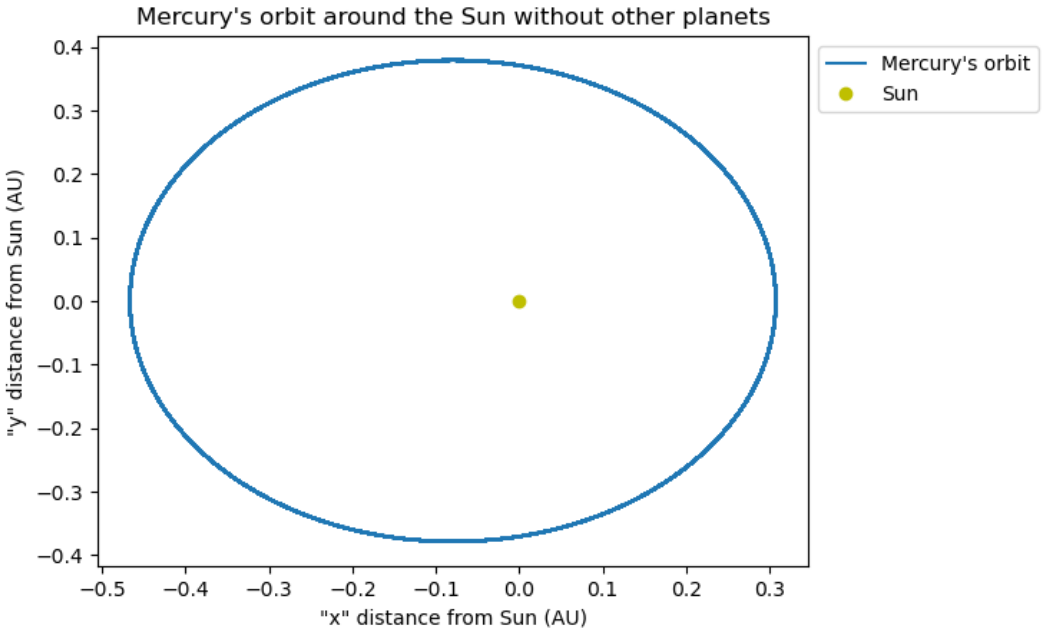

Fig 1. Mercury's orbit about the Sun

We can see this looks as expected. A roughly elliptical orbit with the perigee occurring along the x axis, at around our expect value of slightly over .3 AU's. We are now interested in looking at how much the perihelion precesses in this system containing just the Sun and Mercury. Theoretically, the perihelion should not precess at all. But in order to achieve this result computationally we would need infinite points in order to land exactly at the perihelion each time. Thus, since this is not possible we expect to see some minimal error. So, in order to approach evaluating this precession we must first create a way to track the perihelion. This involves some clever coding. Currently, when we solve our system using odeint, it spits out an array, containing values for the x and y coordinates of however many points we chose to plot, as well as the x direction and y direction velocities at these points. From this array, we can extract the x and y values in to separate arrays. We can then break these arrays of x and y values into equal sized sub arrays, where the number of sub arrays is equal to however many orbits we've sent mercury through in our simulation. So each sub array represents one orbit. We can then loop through each of these sub arrays for each orbit, and find the value with the least distance to the sun. This can be done using the distance formula with the x and y coordinates in the arrays. Looping through and finding the point with least distance should give the perihelion for each orbit. We can then append the corresponding x and y values in to another 2D array to store the values of all of our perihelions for that specific simulation. However, when we loop through like this we actually need to start at the first aphelion in order to avoid error that may occur if we loop through starting at the perihelion. Do this, and our tracker works great. So let's see how our tracker shows the precession of the perihelion of this orbit. First, let's zoom way in around the perihelion from Fig 1, plot our perihelions, and see what it looks like

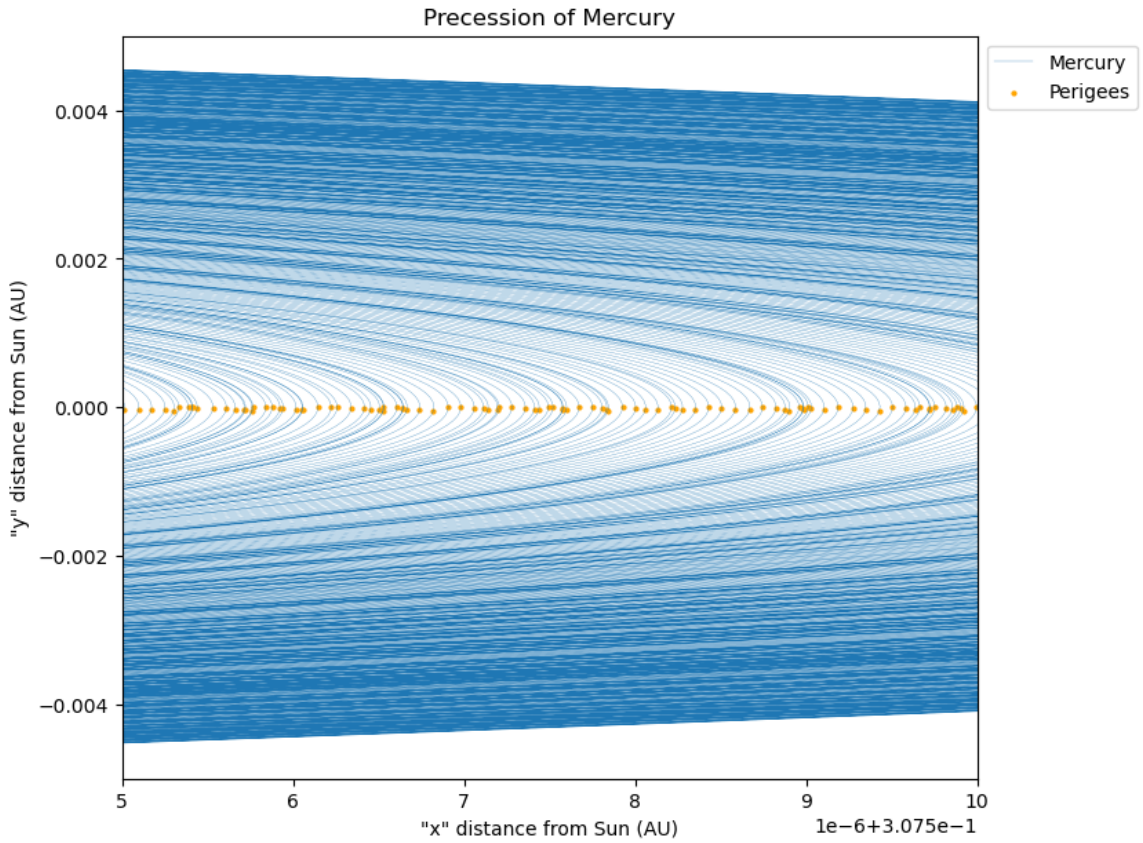

Fig 2. Precession of the perihelion with just Mercury over the sun

This looks like a large amount of precession. However, we can see our x axis is an extremely small range of AU's, and our y axis is small as well, however not as much as the x axis. This is also over 691 orbits of Mercury. This explains why we see so many orbits, and so much small precession in Fig 2. Let's now look at the plot of arcseconds versus time. In order to examine this graph in python, we must first convert all of our x y coordinates for the perihelions in to arcseconds. Do this with an arctan function (dividing each y value by each x value for each point), wrapped in a degree function to convert to degrees from radians, and then multiply by 3600 to convert to arcseconds. When doing this, we can see our plot is the following,

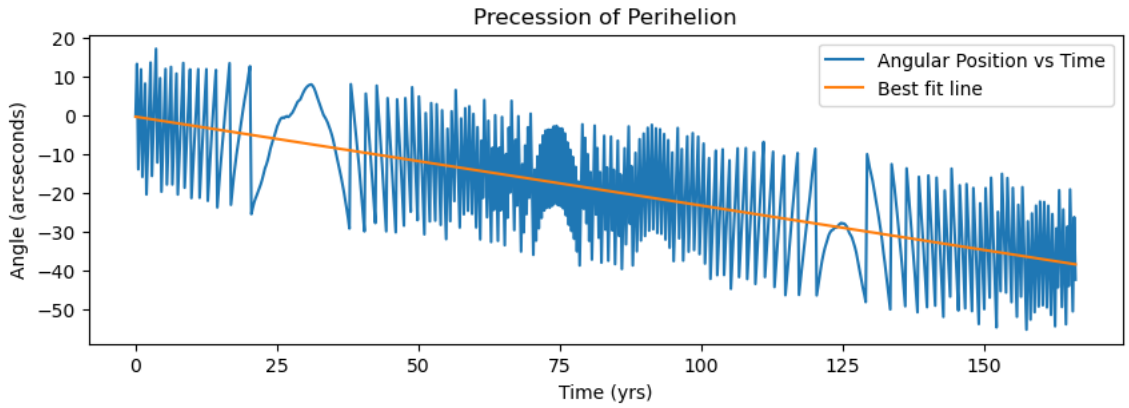

Fig 3. Precession of perihelion in arcseconds versus time with just Mercury and the Sun

Note here the time steps we plot over are one period of Mercury per step, under the assumption that the perihelion will occur around the time point where Mercury finishes its period, since we start at the perihelion as our initial condition. We see in this plot that we get a lot of small arcsecond precession over our 691 orbits. Also on this plot we've added a best fit line. We add this best fit line so we can get a good idea of the average precession of the perihelion, and a best fit line is the most statistically sound way of performing this task. The slope of our best fit line will give us our average precession rate of the perihelion in arcseconds per year. So, to find the average precession rate over a century, we can take this and multiply by 100. When doing this in python, we see our precession rate is around -22.9 arcseconds per century. This is a relatively small precession rate, and is to be expected due to the inaccuracies of the computation system. From this, we can expect any of our precession rates we find later in our investigation to possibly be off by an order of ± -22.9 arcseconds.

Part B:

We now would like to begin adding other planets. We estimate the Jupiter will surely have the larges effect of Mercury's precession versus all the other planets, due to its mass being 3

to 4 orders of magnitude higher than the rest of the planets, besides Saturn. However, Saturn is much further away, so it's effect on Mercury will drop off as a function of this distance cubed. So, we'd like to now begin modeling our system that contains the Sun, Mercury, and Jupiter. To start, we must look back at equation (i), Newton's universal law of gravitation. We know have both the Sun and Jupiter acting Mercury. So we have to instances of Newton's law of gravitation, one for each body acting on Mercury. Doing this we get

$$\overrightarrow{F_{sm}} = -\frac{G * M_m * M_s}{|r_m|^2} * \hat{r}1$$

$$\overrightarrow{F_{jm}} = -\frac{G * M_m * M_j}{|r_m - r_j|^2} * \hat{r}2$$

Here, we do $r_m - r_j$, because this will give us the distance between Mercury and Jupiter, as r_m is the position vector for Mercury, and r_j is the position vector of Jupiter. Also $\hat{r}1$ points from the Sun to Mercury, and $\hat{r}2$ points from Jupiter to Mercury. Just as we did above, we'd like to change our $\hat{r}1\ and\ \hat{r}2$ using the definition of the unit vector. We've already done this for the Sun above in equation (2). Doing this with in our Jupiter case yields

$$\overrightarrow{F_{jm}} = -\frac{G*M_m*M_j}{|r_m-r_j|^2} * \hat{r}2 \ \rightarrow \overrightarrow{F_{jm}} = -\frac{G*M_m*M_j}{|r_m-r_j|^2} * \frac{\overrightarrow{(r_m-r_j)}}{|r_m-r_j|} \rightarrow \overrightarrow{F_{jm}} = -G * M_m * M_j * \frac{\overrightarrow{(r_m-r_j)}}{|r_m-r_j|^3}$$

Now when we look at Newton's second law applied to Mercury, since we have new forces, we have

$$\Sigma\overrightarrow{F_m} = M_m * \frac{d^2\overrightarrow{r_m}}{dt^2} = \overrightarrow{F_{sm}} + \overrightarrow{F_{jm}} = -G * M_m * M_s * \frac{\overrightarrow{r_m}}{|r_m|^3} - G * M_m * M_j * \frac{\overrightarrow{(r - r_j)}}{|r - r_j|^3}$$

Which gives us our equation of motion,

$$M_m * \frac{d^2\overrightarrow{r_m}}{dt^2} = -G * M_m * M_s * \frac{\overrightarrow{r_m}}{|r_m|^3} - G * M_m * M_j * \frac{\overrightarrow{(r_m - r_j)}}{|r_m - r_j|^3}$$

And dividing out Mercury's mass we get

$$\frac{d^2\overrightarrow{r_m}}{dt^2} = -G * M_s * \frac{\overrightarrow{r_m}}{|r_m|^3} - G * M_j * \frac{\overrightarrow{(r_m - r_j)}}{|r_m - r_j|^3}$$

However, we also need to have an equation of motion for Jupiter, as its position is changing as well. In order to do this, we must apply Newton's second law to Jupiter to find the force acting on Jupiter, which yields

$$\Sigma\vec{F_J} = M_j * \frac{d^2\vec{r_J}}{dt^2}$$

We can then apply Newton's law of universal gravitation to Jupiter and the Sun to find the force the sun exerts on Jupiter is equal to

$$\overrightarrow{F_{sJ}} = -\frac{G * M_j * M_s}{|r_j|^2} * \hat{r}j$$

Where $\hat{r}$j points from the Sun to Jupiter. Now, technically we should also consider Newton's universal law of gravitation between Mercury and Jupiter to find the force that Mercury exerts on Jupiter. However, this effect is negligible for the purposes of our computation, resulting in only a couple arcseconds difference. In fact, as we move through and add more planets, we will not consider any of the forces of the planets acting on each other. Rather, we will consider for Mercury the force from the Sun and all the other planets, and for all the other planets we will only consider the force from the Sun. This is for the same reason as before, as these forces are negligible for the purposes of our calculations, resulting in only a few arcseconds difference while streamlining computation.

Now, since we are considering only the force of the Sun acting on Jupiter, we can now say that $\Sigma\vec{F_J} = \overrightarrow{F_{sJ}}$ which we can then put together to see that

$$M_j * \frac{d^2\vec{r_j}}{dt^2} = -\frac{G * M_j * M_s}{|r_j|^2} * \hat{r}j$$

And finally dividing through by Jupiter's mass we have

$$\frac{d^2\vec{r_j}}{dt^2} = -\frac{G * M_s}{|r_j|^2} * \hat{r}j$$

So we now are looking at a system of two vector differential equations, namely

$$\frac{d^2\overrightarrow{r_m}}{dt^2} = -G * M_s * \frac{\overrightarrow{r_m}}{|r_m|^3} - G * M_j * \frac{\overrightarrow{(r_m - r_j)}}{|r_m - r_j|^3}$$

$$\frac{d^2\vec{r_j}}{dt^2} = -\frac{G * M_s}{|r_j|^2} * \hat{r}j$$

Which describe Mercury's and Jupiter's orbit respectively. We now need to break these in to components, and then in to coupled first order differential equations just as we did for Mercury alone. Breaking these in to components yields,

$$\frac{d^2x_m}{dt^2} = -G * M_s * \frac{x_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} - G * M_j * \frac{(x_m - x_j)}{\left(\sqrt{(x_m - x_j)^2 + (y_m - y_j)^2}\right)^3}$$

$$\frac{d^2y_m}{dt^2} = -G * M_s * \frac{y_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} - G * M_j * \frac{(y_m - y_j)}{\left(\sqrt{(x_m - x_j)^2 + (y_m - y_j)^2}\right)^3}$$

for the equation describing Mercury, and

$$\frac{d^2x_j}{dt^2} = -G * M_s * \frac{x_j}{\left(\sqrt{x_j^2 + y_j^2}\right)^3}$$

$$\frac{d^2y_j}{dt^2} = -G * M_s * \frac{y_j}{\left(\sqrt{x_j^2 + y_j^2}\right)^3}$$

for Jupiter. Breaking further in to coupled equations we get

$$\frac{dvx_m}{dt} = -G * M_s * \frac{x_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} - G * M_j * \frac{(x_m - x_j)}{\left(\sqrt{(x_m - x_j)^2 + (y_m - y_j)^2}\right)^3} \;\&\&\; \frac{dx_m}{dt} = vx_m$$

$$\frac{dvy_m}{dt} = -G * M_s * \frac{y_m}{\left(\sqrt{x_m^2 + y_m^2}\right)^3} - G * M_j * \frac{(y_m - y_j)}{\left(\sqrt{(x_m - x_j)^2 + (y_m - y_j)^2}\right)^3} \;\&\&\; \frac{dy_m}{dt} = vy_m$$

for Mercury, and

$$\frac{dvx_j}{dt} = -G * M_s * \frac{x_j}{\left(\sqrt{x_j^2 + y_j^2}\right)^3} \;\&\&\; \frac{dx_j}{dt} = vx_j$$

$$\frac{dvy_j}{dt} = -G * M_s * \frac{y_j}{\left(\sqrt{x_j^2 + y_j^2}\right)^3} \;\&\&\; \frac{dy_j}{dt} = vy_j$$

for Jupiter.

We now have a system of four first order ODEs that odeint can solve. We only now need to find our initial conditions. To do this, we do the same as we did above for just Mercury. We start both planets at their perihelion's, and have the x axis pass through those points so we can say that the initial y position and x velocity is zero for both planets. We then use the same equations we used for Mercury to calculate the perihelion Jupiter, as well as it's initial y velocity, and say these are our initial conditions. So our initial conditions here become,

$$x_{m0} = p_m, \qquad y_{m0} = 0, vx_{m0} = 0, vy_{m0} = \sqrt{\frac{(1 + \epsilon_m)(G * M_s)}{p_m}}$$

$$x_{j0} = p_j, \qquad y_{j0} = 0, vx_{j0} = 0, vy_{j0} = \sqrt{\frac{(1 + \epsilon_j)(G * M_s)}{p_j}}$$

We then use python to do our calculations, and use odeint to solve this system. We can now plot our orbits, and we get the plot

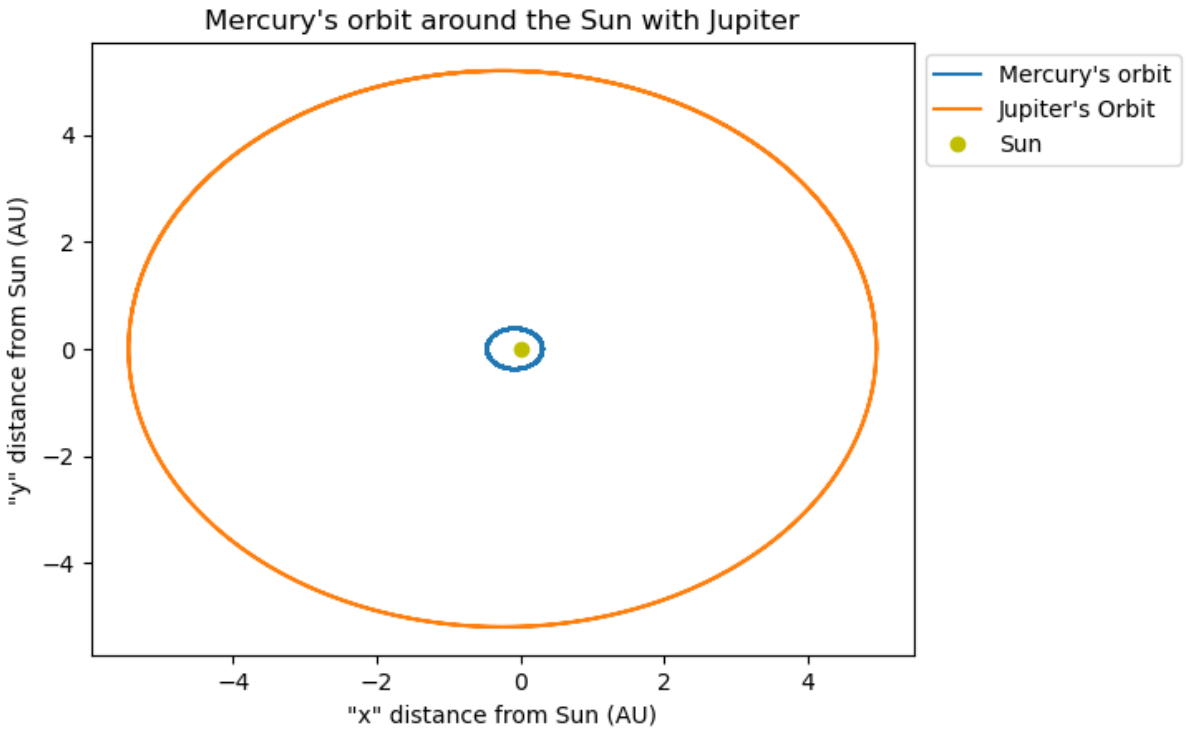

Fig 4. Orbit with Jupiter added

This looks like what we'd expect. Cool. Now, we presumed that Jupiter would have the greatest effect on Mercury's precession, so let us plot the precession of the perihelion to see what it looks like. This time, let's just plot the angles versus time. Again in our code, we will use our perihelion tracker, then convert the points to arcsecond angular positions. Plotting this over time (with the same time steps as described above) we get,

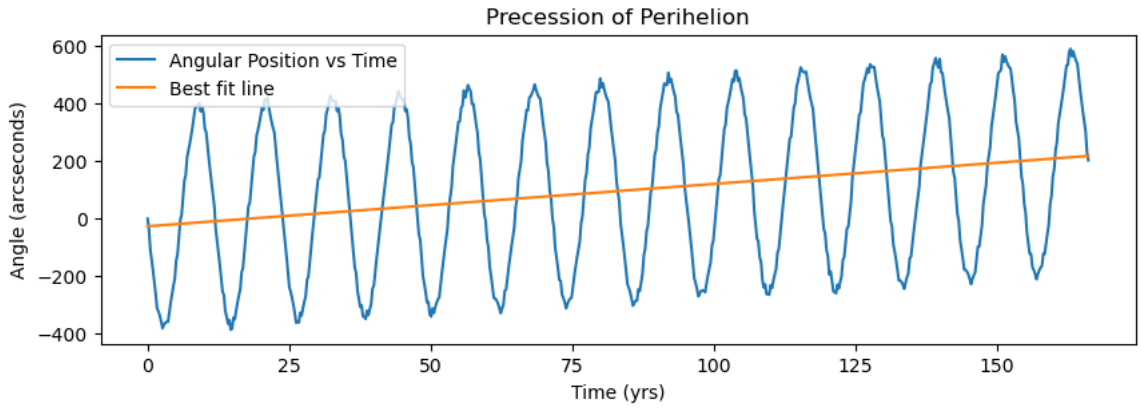

Fig 5. Precession of Mercury's orbit in arcseconds vs time with just Jupiter

We see that in Fig 5, we are getting much more precession in our angle than with just Mercury, and the precession almost creates a sinusoidal wave. This makes sense, as each peak and trough here correspond with a Jupiter orbit it, as Jupiter pulls Mercury one way or the other depending on where it is in its orbit. When calculating the slope of our best fit line, and multiplying by 100 just as above, we get a precession rate of around 147 arcseconds per century. This is much more than we got before with just Mercury, as expected. But still far off from around 530 arcseconds per century that we expect before adding relativity. So, let's go ahead and add the other planets.

For adding the other planets, we will omit explicit details in deriving their equations, along with the initial conditions for the system. This is just repeating the process described above in both the Mercury system, as well as the Mercury and Jupiter system. However, keep in mind that as mentioned, we are ignoring the effects the other planets have on each other, and only considering how the Sun acts on these planets. We only consider the effects these planets have on Mercury. Again, we do this because adding these forces results in negligible changes to our final calculated precession, and not having these forces streamlines computation. So, we start by adding each planet one by one. Let's look at each of these orbits successively

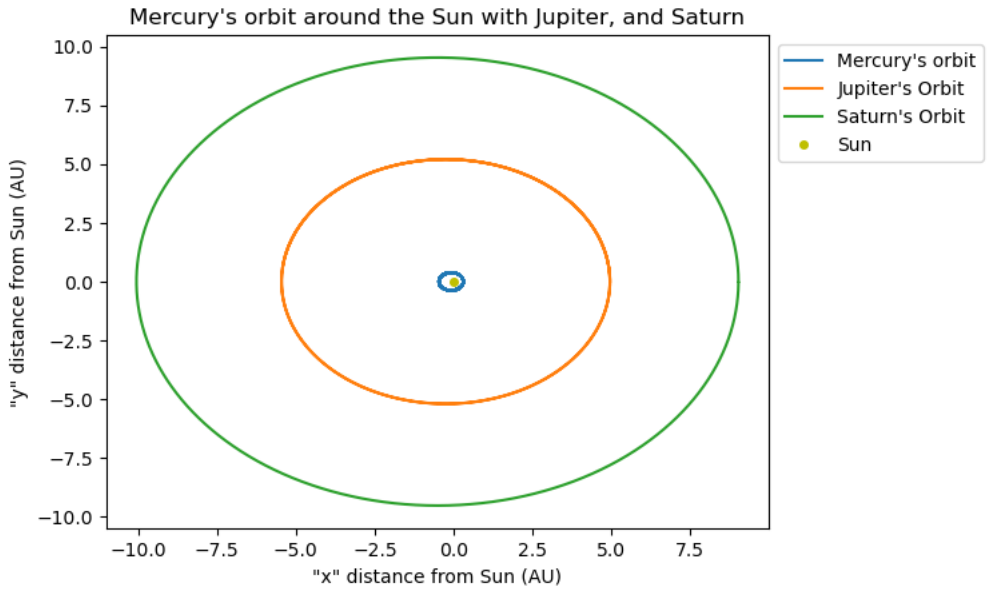

Fig 6. Orbit with Jupiter and Saturn added

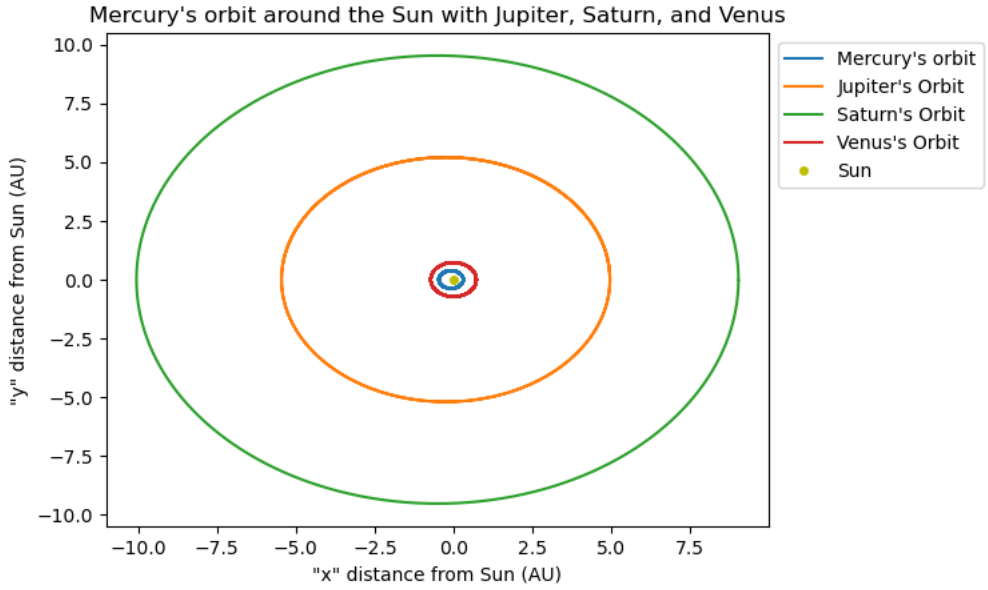

Fig 7. Orbit with Jupiter, Saturn, and Venus added

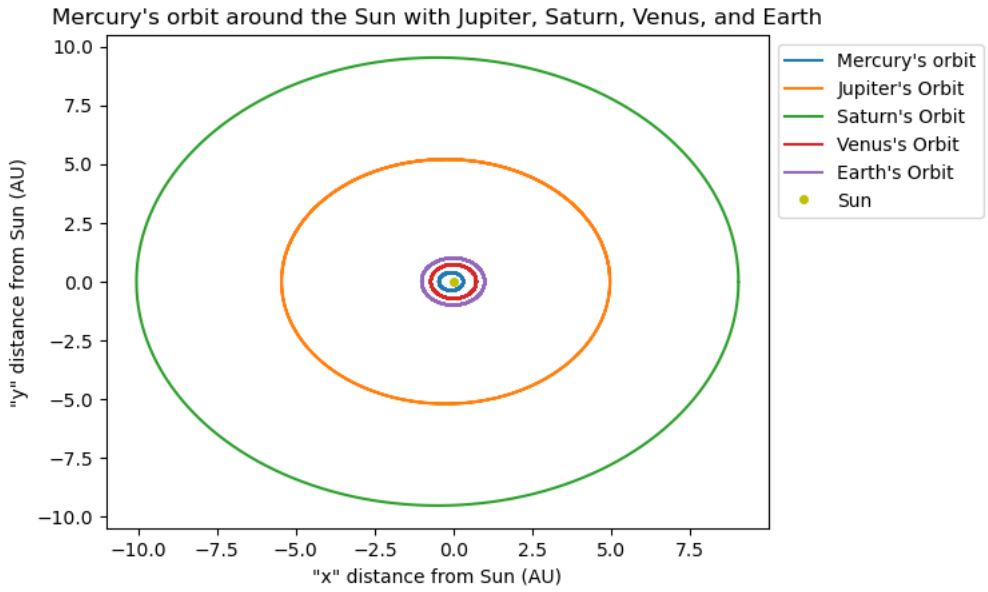

Fig 8. Orbit with Jupiter, Saturn, Venus, and Earth added

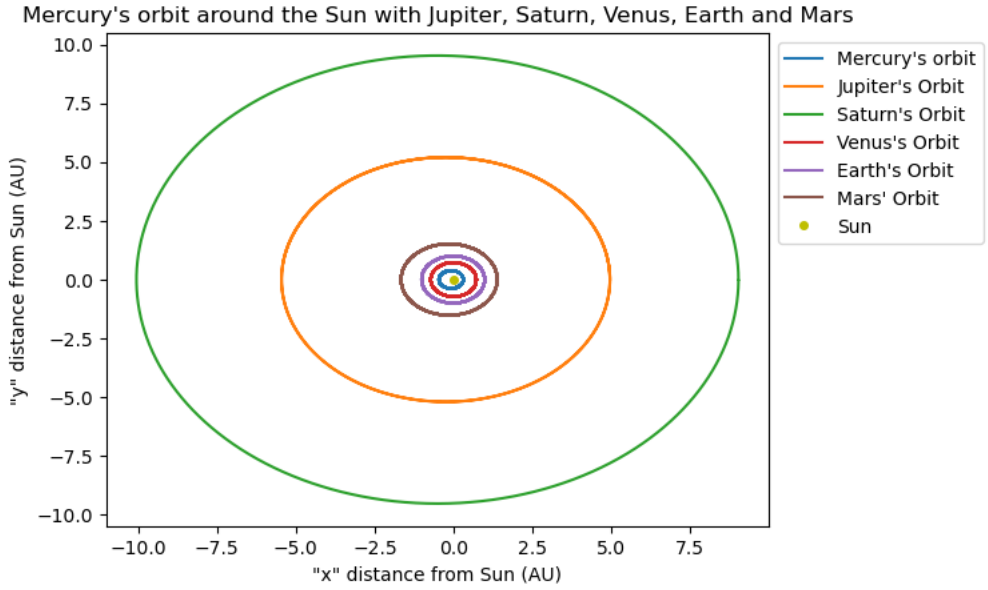

Fig 9. Orbit with Jupiter, Saturn, Venus, Earth, and Mars added

We now have all of our planets plotted, and they all look just as they should. Cool. Let's now investigate the precession for our system that includes all planets and see if it matches up with what we'd expect it to. First let's plot a zoomed in view of our precession to see how it

looks compared to our original zoomed in plot above. Then, lets plot the arcsecond position over time, and take the best fit line. Again, track our perihelion over all of our orbits, and convert that array of points in cartesian to an array of their angular positions in arcseconds.

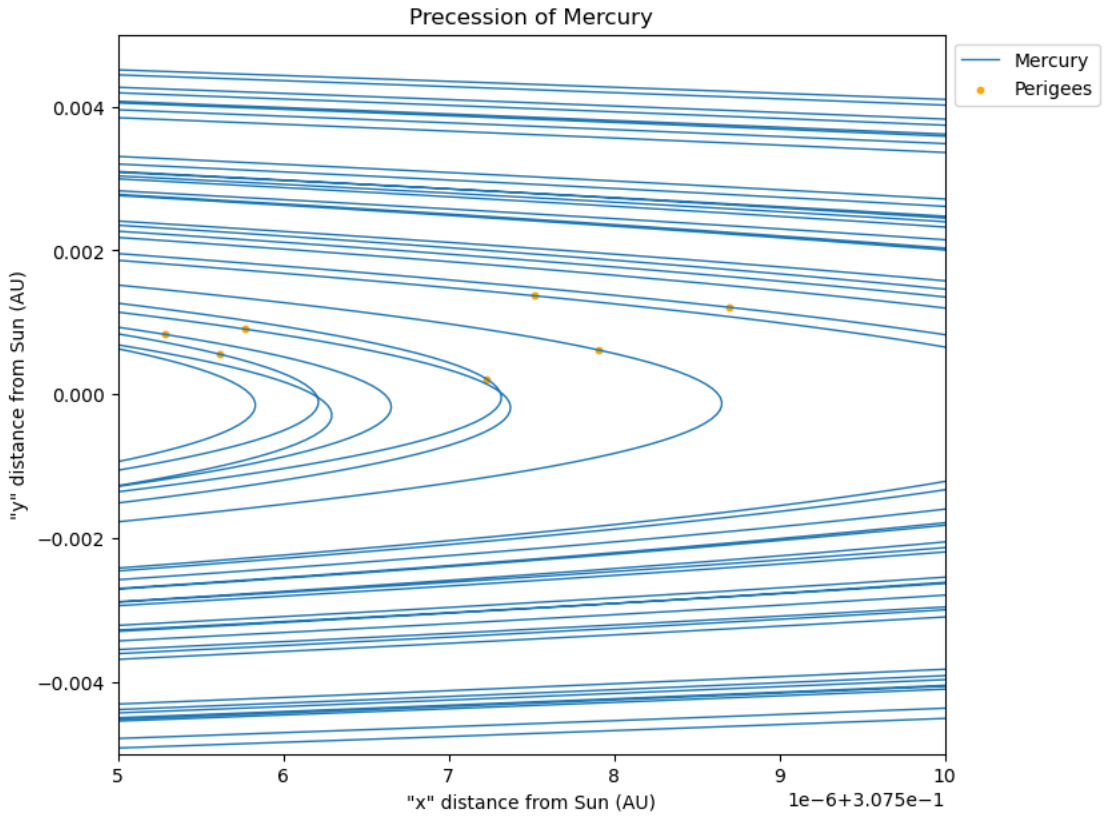

Fig 10. Zoomed in precession of Mercury with all planets in System

Here we have plotted this precession with the same number of orbits, and the same limit on the x axis. We can see clearly that our precession is much more spread out, with more being out of the frame. This shows us that the orbit precesses by a larger amount, which is exactly what we'd expect. Now let's look at the angles in arcseconds vs time, again using our time steps described above for this job.

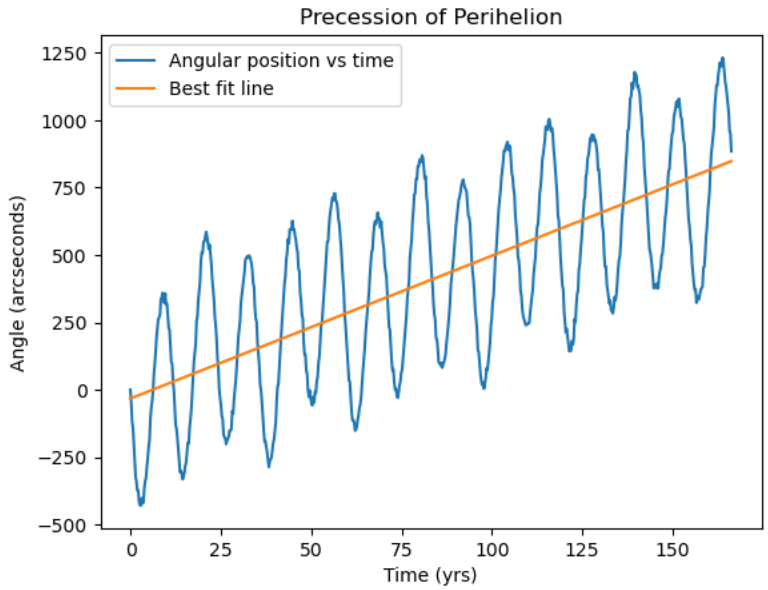

Fig 11. Precession of the perihelion in arcseconds versus time with all planets in the system

This is the same amount of orbits and same time steps as our plot for Jupiter. We can see clearly that it slopes much sharper now, showing much greater average precession. We also see that the peaks and troughs don't seem to follow the slope of the best fit line as well as they do in the Jupiter case. This makes sense, because while each peak and trough still corresponds to a Jupiter orbit, the other planets are playing a significant role as well and causing this different oscillation. When calculating this slope of the best fit line, and multiplying by 100 to find average precession over a century just as we did above, we get an average precession per century of 530.64 arcseconds per century. We expected to get around 532 arcseconds per century, so we have gotten a very accurate model. Any error can be attributed to error in computation, just as described in why we are seeing precession with just Mercury. So we've done very well in our accuracy. Awesome.

Part C:

Now we'd like to add relativity. We know we must add the term $(1 + \frac{\alpha}{|r|^2})$ on to Newton's universal law of gravitation to approximate relativity, turn equation (i) in to the equation

$$\overrightarrow{F_{m1m2}} = -\frac{G * m_1 * m_2}{|r|^2} * \left(1 + \frac{\alpha}{|r|^2}\right) * \hat{r}$$

Where $\alpha \approx 1.1 * 10^{-8} AU^2$. So, we just use our system we defined above for the planets. However, we only apply this α on to the term involving the Sun from Newton's universal law of gravitation on Mercury. This is because the effects that this relativity has on the other planets being much smaller and close to negligible, as the further distances make the relativity term become much closer to one than it is for just Mercury and the Sun. So, multiplying by this term is close to multiplying by one, making its effect more negligible. So, we can omit it to streamline our computation. Lets quick plot all our orbits to make sure they look right.

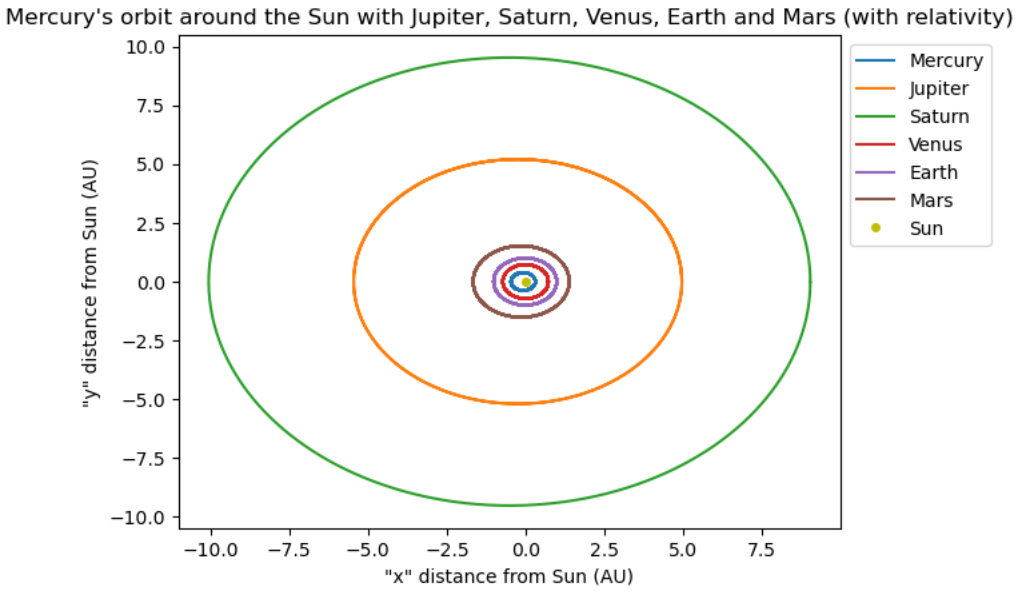

Fig 12. Orbits of all planets with relativity

These look correct, and as we'd expect. This looks no different from our plot of all the orbits above. Lets now plot our zoomed in precession, as well as our precession of arcsecond position over time. Again, do the same conversion to arcseconds as above.

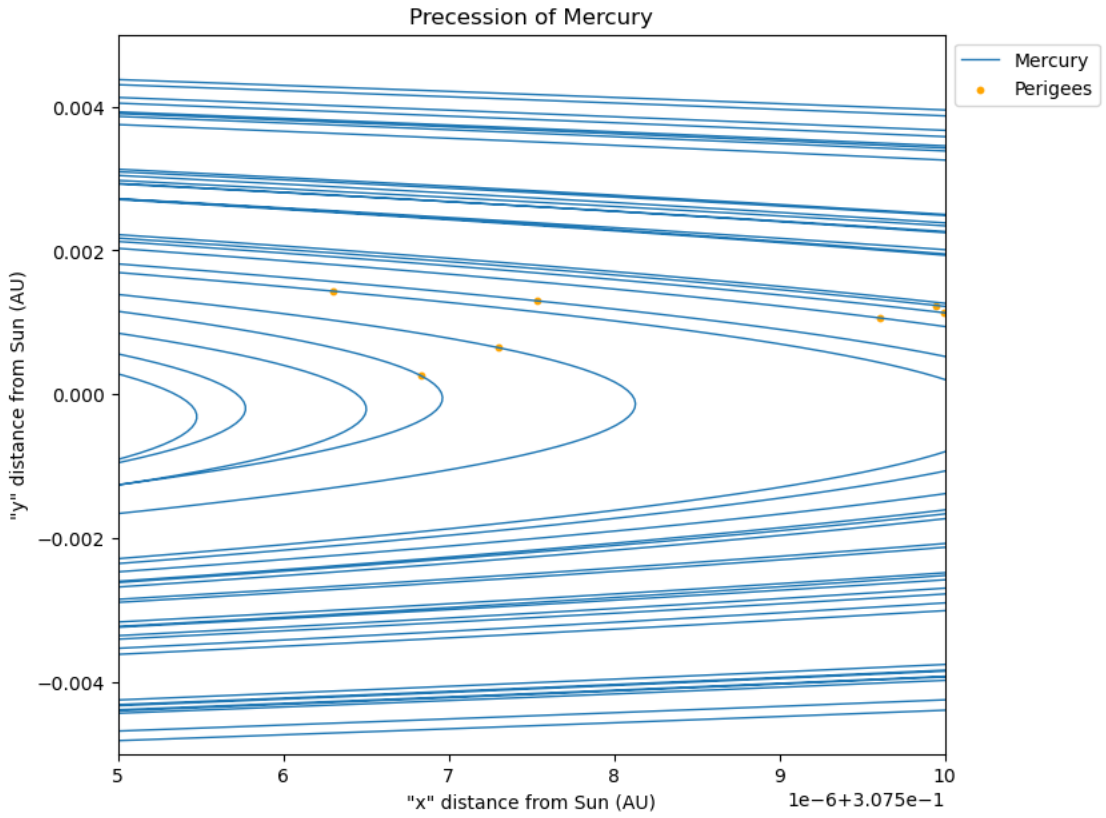

Fig. 13 Zoomed in precession of Mercury with all planets in system plus relativity

We see on our zoomed in plot, over the same time steps and domain as used in the ones above, we get a similar result to that of the precession for all the planets. This is to be expected, as relativity should not affect it that greatly. However, it does appear as there may be a few less lines in this graph, suggested slightly more precession which is what we expect from adding relativity. Now let's look at our angular position vs time.

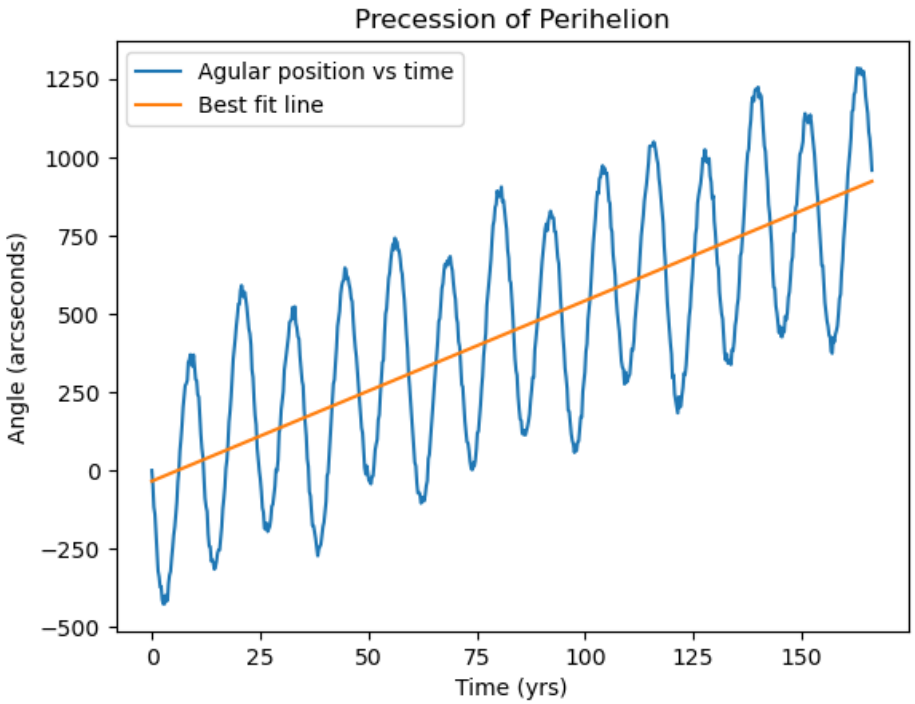

Fig 14. Precession of the perihelion in arcseconds versus time with all planets in the system plus relativity

Here, just as our zoomed in precession, this looks nearly indistinguishable from our arcsecond precession vs time for our system with all planets without relativity. There may be some signs of greater precession, however it looks very similar just looking at. However, when we calculate our average precession per century using our slope of our best fit line, we get around 575.8 arcseconds per century. This is very close to our expected value of 575.19. This shows again that we've created a very accurate simulation. Any error out to this degree is expected from the nature of this computation. So we've done very well. Awesome! The fact that we have solved a problem that a Nobel laureate sweated over, and have done it more accurately than he did is so awesome. It really shows how far this field has come, (and not to mention how awesome computers are in helping us with computation). So cool. MBP Complete!