Celestial Mechanics Note Set 3: General Three Body Problem and the Orbital Configurations of Euler and Lagrange

J.D. Mireles James

January 1, 2007

Contents

1 Introduction

This set of notes contains material about the general three body problem. This problem has well known equilibria solutions, when cast in rotating coordinates. We present several numerical experiments devoted to the properties of these special solutions. (The derivations of the Euler and Lagrange solutions are taken for granted in this present version of the notes, although I would like to fix that at some later date).

Aside from the material on these special solutions, this set of notes also contains some analytic results that are useful in the study of the three body problem. Specifically, the equations of motion are developed in Jacobi Coordinates, and the expressions for velocity and acceleration in a rotating reference frame are derived.

This last item can be used to derive the equations of motion for the circular restricted three body problem, which is the subject of the next set of notes. We will choose not to user it there, in favor of a completely bottom up approach. Nevertheless it is a useful result to have easy reference to, and the derivation given here differs from the ones usually given in physics texts in that no appeal is made to intuition nor is any use made of "infinitesimal rotations". In fact it it not even necessary to make any limiting arguments; all that is needed is a messy computation.

2 Numerical Study of Lagrange's Equilateral Triangular Solutions

This section makes a study of the equilateral triangle solutions to the general three body problem, following the method put forward in the axillary notes [Ocampo 2006]. Given the initial data $G = 1$, that the initial position of the center of mass for the system is $\mathbf{r}_{cm}(0) = (0, 0, 0)$, that the initial velocity of the center of mass is $\mathbf{v}_{cm}(0) = (-1, 1, 0)$, and that the masses are $m_1 = 1.0$, $m_2 = 2$, $m_3 = 3$ the program *hw3prob2* computes a set of initial conditions, given in inertial coordinates, which will put the three bodies into co-planar periodic orbits such that at any given instant each body is at a vertex of an equilateral triangle with side lengths $r = 1$. If is desired that the periodic orbits lie in the xy plane of the inertial frame. This is achieved by specifying the plane be perpendicular to the unit normal vector

Note here that the program is designed so that all of these quantities are global parameters and may be changed to suit the problem at hand. Given any such set of initial date the output will be a set of initial conditions which put three bodies into such a coplanar Lagrange equilateral triangle configuration. The program also simulates these initial conditions and transforms the resulting numerical trajectories into non-inertial coordinates rotating with the angular velocity of the Lagrange solution. The period of the solutions is computed and the number of periods over which the simulation is to run is specified by the user. By adjusting these initial data one can experiment with both the dynamical and numerical stability of these solutions.

For the data specified in the assignment we obtain the following results. Following the notes we compute

$$\omega = \sqrt{\frac{G}{r^3}(m_1 + m_2 + m_3)} \approx 2.4495$$

Then the distance from the center of the triangle to any vertex will be

$$l = \frac{r}{\sqrt{3}} \approx 0.5774$$

Next the locations of the masses in a rotating frame are specified. These must be in a triangular configuration with the desired l and r. Then we can choose

$$\mathbf{r}_{1_{tc}} = \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0.5774 \\ 0 \end{bmatrix}$$

$$\mathbf{r}_{2_{tc}} = \begin{bmatrix} l\cos 210^\circ \\ l\sin 210^\circ \\ 0 \end{bmatrix} \approx \begin{bmatrix} -0.5000 \\ -0.2887 \\ 0 \end{bmatrix}$$

$$\mathbf{r}_{3_{tc}} = \begin{bmatrix} l\cos 330^\circ \\ l\sin 330^\circ \\ 0 \end{bmatrix} \approx \begin{bmatrix} 0.5000 \\ -0.2887 \\ 0 \end{bmatrix}$$

Next the center of mass is located and the frame frame is translated to have this as it's center. The center of mass is found to be

$$\mathbf{r}_{cm_{tc}} = \frac{1}{m_1 + m_2 + m_3}(m_1\mathbf{r}_{1_{tc}} + m_2\mathbf{r}_{2_{tc}} + m_3\mathbf{r}_{3_{tc}}) = \begin{bmatrix} 0.0833 \\ -0.1443 \\ 0 \end{bmatrix}$$

Then the translated coordinates become

$$\mathbf{r}_{i_{cm}} = \mathbf{r}_{i_{tc}} - \mathbf{r}_{cm_{tc}} \qquad \mathbf{v}_{icm} = (0, 0, 0)$$

Without stating the numbers so obtained we can outline the remainder of the procedure. A set of coordinates is derived which are inertial and and have the mass center as the origin. The transformation R^{mi} will be the matrix which takes vectors in the rotating frame centered at the CM to the corresponding vectors in the inertial frame centered at the center of mass. The inertial coordinates centered at the CM the initial conditions are

$$\mathbf{r}_{im}(t_0) = \mathbf{r}_{icm} \qquad \mathbf{v}_{im}(t_0) = \mathbf{v}_{icm} + \omega \times \mathbf{r}_{icm}$$

The matrix R^{mi} can now be used to transform these to the specified inertial coordinates via the formulas

$$\mathbf{r}_i(t_0) = \mathbf{r}_{cm}(t_0) + R^{mi}\mathbf{r}_{im}(t_0)$$

and

$$\mathbf{v}_i(t_0) = \mathbf{v}_{cm}(t_0) + R^{mi}\mathbf{v}_{im}(t_0)$$

Carrying out these computations one has that

$$\mathbf{r}_1(t_0) \approx \begin{bmatrix} -0.0833 \\ 0.7217 \\ 0 \end{bmatrix}$$

$$\mathbf{r}_2(t_0) \approx \begin{bmatrix} -0.5833 \\ -0.1443 \\ 0 \end{bmatrix}$$

$$\mathbf{r}_3(t_0) \approx \begin{bmatrix} 0.4167 \\ -0.1443 \\ 0 \end{bmatrix}$$

$$\mathbf{v}_1(t_0) \approx \begin{bmatrix} -2.7678 \\ 0.7959 \\ 0 \end{bmatrix}$$

$$\mathbf{v}_2(t_0) \approx \begin{bmatrix} -0.6464 \\ -0.4289 \\ 0 \end{bmatrix}$$

$$\mathbf{v}_3(t_0) \approx \begin{bmatrix} -0.6464 \\ 2.0206 \\ 0 \end{bmatrix}$$

The $\approx$ in these expressions is a reminder of the fact that these are numerical approximations of the theoretical initial conditions which would lead to an equilateral triangle solution. Since the values only approximate the true solution we are already working with a perturbation of the desired initial conditions.

Now we integrate the general three body problem with these numerical approximations of the Lagrange initial conditions for this initial data. One expects that the trajectories so obtained will stay on an equilateral triangle for some finite time, but that eventually numerical error and the initial perturbation of the theoretical values may cause the system to evolve away from this initial configuration.

First we will numerically confirm that the orbits we have obtained are in fact circular. This is confirmed by computing the norm of the displacement of each body from the center of mass at each time increment. If the results are constant then the orbits are circular.

The first figure shows the results of this experiment. The red line is the displacement of mass one. Blue is mass two and green is mass three. The results show that to a high degree of accuracy the orbits are in fact circular.

Further conformation that the numerical results are in good agreement with the theoretical predictions can be obtained by considering the direction of the forces. The derivation of the Lagrange solutions tells us that instantaneous force on each particle should point to the center of mass.

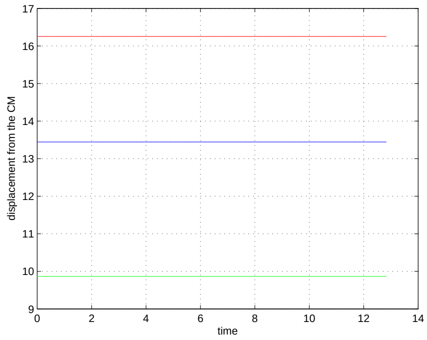

Figure 1: Distance from each body to the center of mass as a function of time

We can compute the force by evaluating the vector field for the n body problem a each time t (here we consider the system of second order equations, rather than the first order system). This gives the total force $\mathbf{F}_i(t)$ on body i at each time. The force $\mathbf{F}_i(t)$ is pointing in the direction of the center of mass if the cross product

$$\mathbf{F}_i(t) \times (\mathbf{r}_{cm}(t) - \mathbf{r}_i(t)) = \mathbf{0}$$

where these vectors are in the inertial reference frame. The results of this experiment are shown in Figure 2.

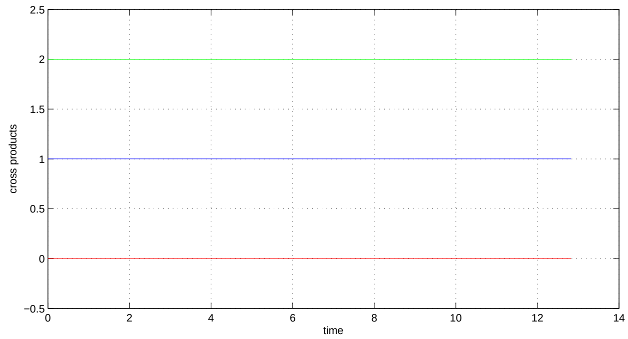

Figure 2: cross product of force and displacement from CM as a function of time

Here we have added one to the results for mass two and two to the results for mass three in order that the lines do not coincide. The plot then shows that the numerical forces have the correct direction at least up to time 12, and gives more evidence that the numerical trajectories are good approximations of the theoretical.

Now, since the orbits are approximately circular and the angular velocity $\omega \approx 2.4495$, it follows that the period of the orbits is $T = 2\pi/\omega \approx 2.5651$ time units. This allows us to consider questions about how long the orbits remain in the triangular configuration using the natural clock of the system; its own period.

We simulate the system over a time interval

$$[0, t_f] = [0, 5 * T] \approx [0, 12.8255]$$

and plot the first coordinate of the position vector of the first mass as a function of time (in the inertial frame centered at the center of mass). This should be a sinusoid and we expect to see five periods of it. This is shown in the third figure, and confirms our expectations.

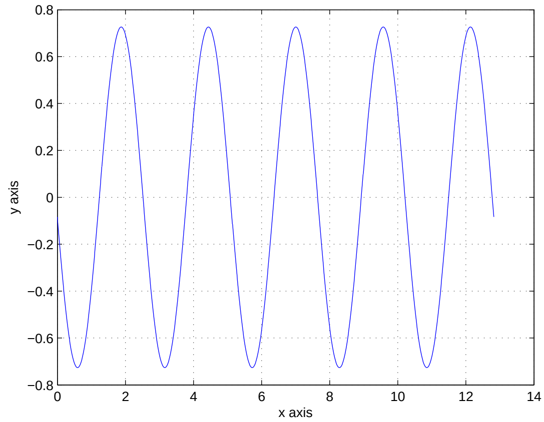

Figure 3: Five periods of the oscillation of one component of position

The next two figures show the trajectories of the masses, first in the inertial frame and then in a rotating frame whose angular velocity matches that of the system.

Figure Five gives the impression that, in the rotating frame, the masses sit in equilibrium at the vertices of an equilateral triangle. This qualitative impression can be measured quantitatively by considering the amount of drift, in the non-inertial reference frame,o f the points away from the equilibrium. This is the

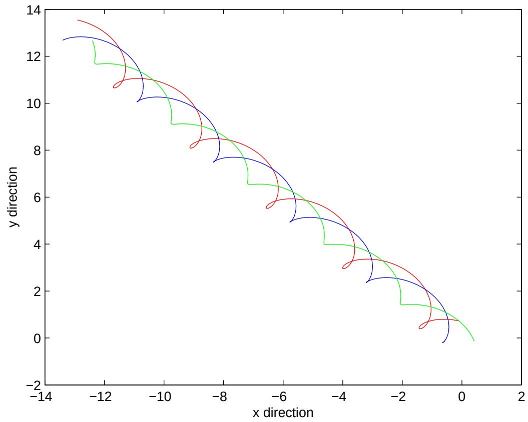

Figure 4: Trajectories of the bodies in the inertial frame

distance of the masses at a given time from their initial positions, all taken in the rotating frame.

Specifically let $\mathbf{x}_m(t) \in \mathbb{R}^{18}$ be the state of the system at time t in the rotating frame and define an error norm $d_{error} : \mathbb{R}^{18} \times \mathbb{R} \to \mathbb{R}^+$ by

$$d_{error}(\mathbf{x}_m(t)) = ||\mathbf{x}_m(t) - \mathbf{x}_m(0)||$$

where $||\ ||$ is the standard Euclidean norm on $\mathbf{R}^{18}$.

We compute the error norm as a function of time and give the results in Figure 6. If this norm is computed for five periods the results are zero to within $1.5 \cdot 10^{-6}$. Then during the first five periods the configuration is at equilibrium to six significant figures of accuracy, which sheds no light on how long this state of affairs will persist. Figure 6 shows the error norm over eight periods, and shows clearly both that the numerical system does move away from the equilibrium state, and when this begins to happen.

After eight periods the error is roughly of order one. Since the initial distances between the masses is also roughly order one, this is error is significant and we consider the system to have diverged from the equilibrium. However at this scale the error norm is essentially zero until roughly time $t = 17.5$. Then the time interval in which we can consider the numerical simulation to agree well with the theoretical predictions can be taken to be $[0, 17.5]$.

If we compute the length of the sides over this interval we obtain Figure 7.

Here we see the familiar straight line plot that comes up so often in this problem. For the first seven periods the lengths of the sides are constant and very close to one. It is interesting to note that by the end of seven periods

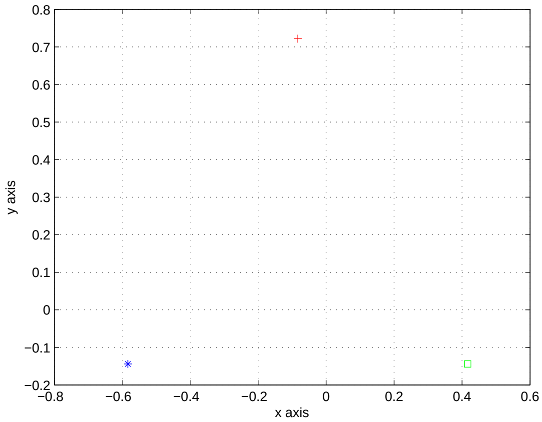

Figure 5: Trajectories in the rotating frame

the state is a distance almost one from the equilibrium, however the lengths if the sides are still one. The triangular configuration seems to have a little more stability than the equilibrium configuration itself.

On the other hand this state of affairs will not persist indefinitely. If we examine the same plot, integrated for a substantially longer interval. This is shown in Figure eight, where we have integrated over 20 periods.

The figure certainly indicates that the state leaves the equilibrium and may not return. However, at least at this time scale the state stays within a bounded distance for the initial configuration. Looking at the actual trajectories in the inertial frame gives some indication of why this may be true. Figure 9 illustrates the behavior of the system for the same twenty periods in this frame.

Again, on this limited time scale, the system is in a 'braided' orbit, and the bodies remain within a neighborhood of one another. Wether or not this is a permanent condition or not we cannot tell from only these simple experiments.

3 Numerical Study of Euler's Collinear Solution

In this section we examine Euler's collinear solutions for the general three body problem. It is required that three bodies m_1, m_2, and m_3 initially lie on the same line. Then we can choose coordinates so that the line is the x axis, the origin is the center of mass and the position of m_1 is to the left of the position of m_2, which is itself to the left of m_3. We specify (arbitrarily) an angular velocity of $\omega = 1$ (and we take $G = 1$).

We will consider several configurations, the first being the case when $m_1 = 3$, $m_2 = 2$, and $m_3 = 1$. For this data we compute that $x_1 \approx -0.9723$, $x_2 \approx$

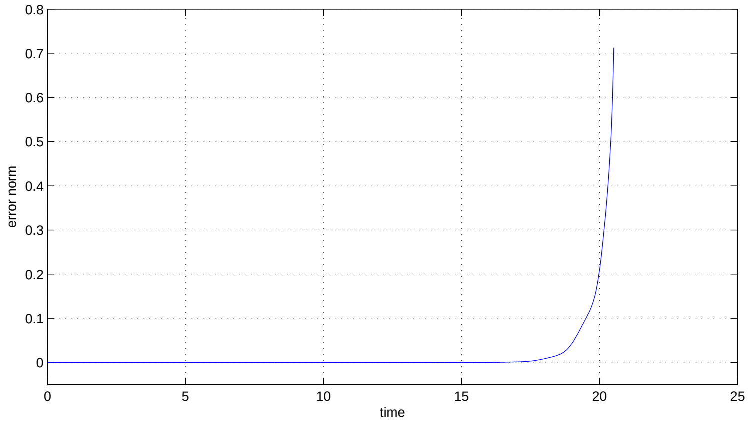

Figure 6: Distance of the current state from the equilibrium state as a function of time over eight periods

0.5708, and $x_3 = 1.7755$, with initial velocities the same magnitudes but in perpendicular directions. We obtain a period of $T \approx 6.2832$, and integrate until $t_f = 2.5133$ Figure ten shows the trajectories of the orbits over this time interval, a little less than half a period, in inertial coordinates with the origin at the center of mass.

The black line is to help visualize the dynamics of the particles. The line would be flat along the x axis at time zero and all three particles would lie on the line. Then we see that m_2 and m_3 (shown in green and blue respectively) have velocities in the positive y direction, while m_1 has initial velocity in the negative y direction. This is because for these values of the initial data m_2 initially lies to the right of the center of mass, and so it will have initial velocity in the same direction as m_3.

The two outer bodies will always have to have velocities in the opposite directions. The variable will be wether m_2 follows m_3 or wether it follows m_1. In the critical case where m_2 finds itself initially at the center of mass is sits in equilibrium.

Consider now the case when $m_1 = 3$, $m_2 = 1$, and $m_3 = 2$. We show a full period of the trajectories that result from this initial data in Figure 11.

For this initial data the initial positions are $x_1 = -0.9949$, $x_2 = 0.2599$, and $x_3 = 1.3625$ with perpendicular initial velocities of the same magnitude and period $T = 6.2832$. From the discussion above we have again that body m_1 initially have velocity in the negative y direction and bodies m_2, and m_3 have initial velocities in the opposite direction, as m_2 is once again initially located

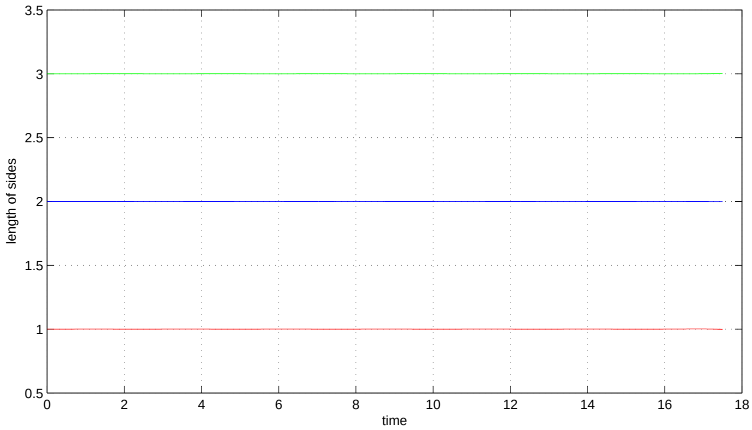

Figure 7: Lengths of the sides of the triangle for seven periods.

to the right of the center of mass. This time however body two is further from body three, even though body two still follows body three.

For this data we also consider the trajectories in rotating coordinates with the same angular velocity as the straight line solution. On the time scale considered the bodies remain near the equilibrium to almost ten significant figures.

The third data to be considered is $m_1 = 1$, $m_2 = 3$, and $m_3 = 2$. For this configuration we compute $x_1 = -1.7121$, $x_2 = -0.2937$, and $x_3 = 1.2966$. Here we finally see m_1 initially to the left of the center of mass so that it follows m_1 rather than m_2 and initially has velocity in the negative y direction.

Figure 13 returns to a partial period and includes the diagonal line to illustrate that the new behavior of m_2. When the trajectories corresponding to this data are plotted in the rotating frame (plot not shown here) the results are within twelve decimal places of the equilibrium through the entire period.

Changing gears a little and returning the system to the first set of initial data introduced, namely $m_1 = 3$, $m_2 = 2$, and $m_3 = 1$, we integrate the data using the program 'hw3prob3'. This program implements a tolerance algorithm that runs the integrator so long as both, a) the current energy differs from the initial energy at the twelfth decimal place or less, and b) the magnitude of the displacement of each particle from the center of mass differs from its initial value by less than 5%. When this difference exceeds five percent we consider the system to have left the Euler configuration.

The algorithm integrates for 1/10 of a period in a while loop. Each time it exits the loop the tolerances are checked and , if the energy is in bounds, and the

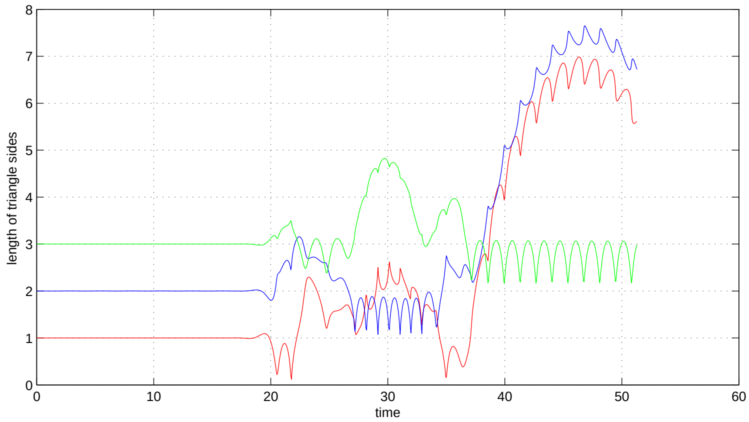

Figure 8: The lengths of the sides over twenty periods

configuration is in tolerance the program computes another tenth of a period. In this way we look for the time $t = t_{escape}$ when the trajectories diverge from the Euler configuration.

Running this program we find that if the loop runs 60 times (roughly six periods) then at the end of the 60^{th} loop the error (discrepancy between the initial and present distances from the center of mass)is roughly 8.24%. By rolling the clock back one iteration we obtain a terminal time for which both the energy and the error tolerances are met. In fact with just a little tweaking the program parameters we find that we can integrate the initial conditions on the time interval $[0, 18.8087]$ without exceeding any of the tolerances and that at $t = t_{escape} = 18.8087$ the error is 0.0499 which is 5% to three significant figures.

If we plot the trajectories from $t = 0$ to $t = 1.5t_{escape}$ we see that the Euler configuration degenerates quite badly once it diverges.

Figure 14 seems to show that the green trajectory is ejected, at least temporarily, from the system. The green path is m_3 which is the least massive of the three bodies. On the other hand if we look at the orbits of m_1, and m_2 we see that they seem remain bonded at least for the time being.

Certainly Figure 15 gives the impression that while the smaller body has been ejected, the two larger bodies m_1 and m_2 remain in a bounded region of one another. In fact is looks as though they are in approximately circular orbits about the center of mass.

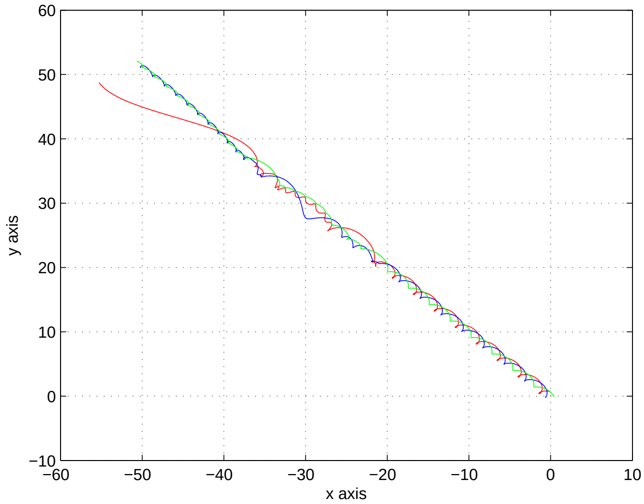

Figure 9: 20 periods in inertial frame

4 Jacobi Coordinates

Consider three masses m_1, m_2, and m_3 in three space with positions $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$. Define

$$\mathbf{r}_{cm} = \frac{1}{m_1 + m_2 + m_3} \sum_{i=1}^{3} m_i \mathbf{r}_i$$

and

$$\mathbf{r}_{cm_{12}} = \frac{1}{m_1 + m_2} \sum_{i=1}^{2} m_i \mathbf{r}_i$$

and let

$$\mathbf{r} = \mathbf{r}_2 - \mathbf{r}_1$$

and

$$\rho = \mathbf{r}_3 - \mathbf{r}_{cm_{12}}$$

Then we have the following

Theorem 1 *In these coordinates*

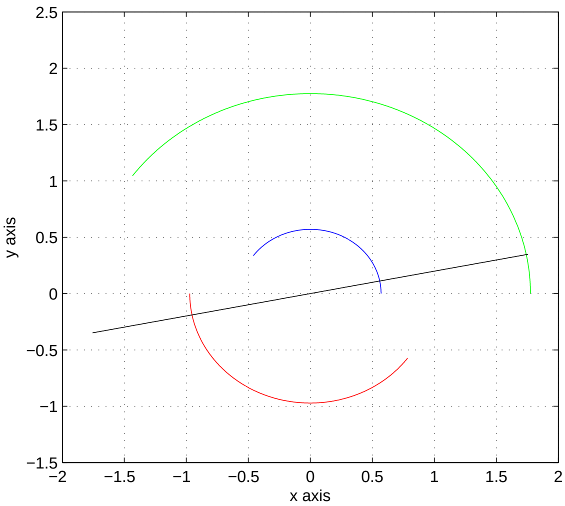

Figure 10: 0.4 periods in inertial frame

(i)

$$\mathbf{r}_3 - \mathbf{r}_1 = \rho + \frac{m_2}{m_1 + m_2}\mathbf{r} \tag{1}$$

(ii)

$$\mathbf{r}_3 - \mathbf{r}_2 = \rho + \frac{m_1}{m_1 + m_2}\mathbf{r} \tag{2}$$

(iii)

$$\ddot{\mathbf{r}} = \frac{G\mu}{|\mathbf{r}|^3}\mathbf{r} + Gm_3\left(\frac{\rho - (m_1/\mu)\mathbf{r}}{r_{23}^3} - \frac{\rho + (m_2/\mu)\mathbf{r}}{r_{13}^3}\right) \tag{3}$$

(iv)

$$\ddot{\rho} = \frac{MG(m_1/\mu)}{r_{13}^3}[\rho + (m_2/\mu)\mathbf{r}] - \frac{MG(m_2/\mu)}{r_{23}^3}[\rho - (m_1/\mu)\mathbf{r}] \tag{4}$$

where $M = (m_1 + m_2 + m_3)$, $\mu = (m_1 + m_2)$, $r_{12} = |\mathbf{r}_3 - \mathbf{r}_1|$, *and* $r_{23} = |\mathbf{r}_3 - \mathbf{r}_2|$

Proof Each part is a computation.

(i)

$$\begin{aligned}
\mathbf{r}_3 - \mathbf{r}_1 &= \mathbf{r}_3 - \mathbf{r}_{cm_{12}} + \mathbf{r}_{cm_{12}} - \mathbf{r}_1 \\
&= \mathbf{r}_3 - \mathbf{r}_{cm_{12}} + \mathbf{r}_{cm_{12}} - \mathbf{r}_2 + \mathbf{r}_2 - \mathbf{r}_1 \\
&= \rho + \mathbf{r}_{cm_{12}} - \mathbf{r}_2 + \mathbf{r} \\
&= \rho + \frac{1}{m_1 + m_2}\sum_{i=1}^{2} m_i\mathbf{r}_i - \mathbf{r}_2 + \mathbf{r}
\end{aligned}$$

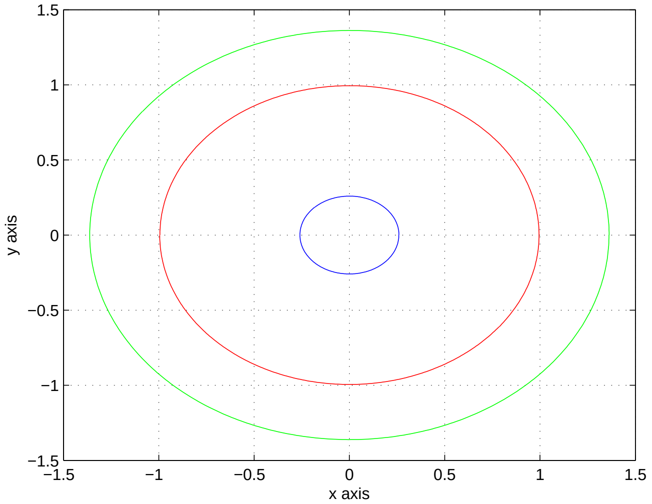

Figure 11: One period of configuration 2 in the inertial frame

$$
\begin{aligned}
&= \rho + -\frac{m_1}{m_1+m_2}(\mathbf{r}_2-\mathbf{r}_1)+\mathbf{r} \\
&= \rho + \frac{m_2}{m_1+m_2}\mathbf{r}
\end{aligned}
$$

(ii)

$$
\begin{aligned}
\mathbf{r}_3-\mathbf{r}_2 &= \mathbf{r}_3-\mathbf{r}_{cm_{12}}+\mathbf{r}_{cm_{12}}-\mathbf{r}_1 \\
&= \mathbf{r}_3-\mathbf{r}_{cm_{12}}+\mathbf{r}_{cm_{12}}-\mathbf{r}_1+\mathbf{r}_1-\mathbf{r}_2 \\
&= \rho+\mathbf{r}_{cm_{12}}-\mathbf{r}_1+\mathbf{r} \\
&= \rho+\frac{1}{m_2+m_1}\sum_{i=1}^{2} m_i\mathbf{r}_i-\mathbf{r}_1+\mathbf{r} \\
&= \rho+-\frac{m_2}{m_1+m_2}(\mathbf{r}_2-\mathbf{r}_1)-\mathbf{r} \\
&= \rho-\frac{m_2}{m_1+m_2}\mathbf{r}
\end{aligned}
$$

(iii)

$$
\begin{aligned}
\ddot{\mathbf{r}} &= \ddot{\mathbf{r}}_2-\ddot{\mathbf{r}}_1 \\
&= G\left(\frac{m_1}{|\mathbf{r}_{21}|^3}\mathbf{r}_{21}+\frac{m_3}{|\mathbf{r}_{23}|^3}\mathbf{r}_{23}-\frac{m_2}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12}-\frac{m_3}{|\mathbf{r}_{13}|^3}\mathbf{r}_{13}\right)
\end{aligned}
$$

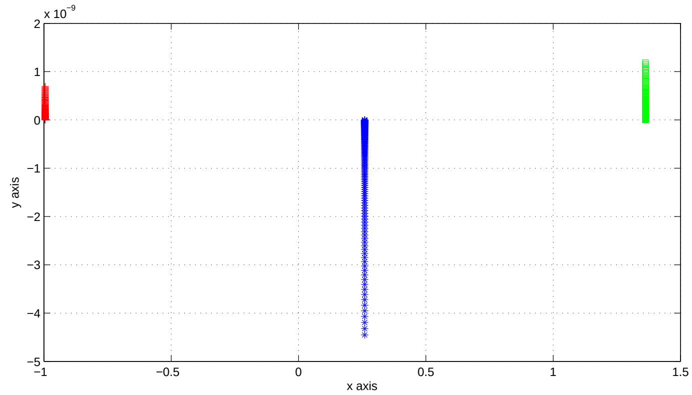

Figure 12: One period in the non-inertial frame with vertical scale of ten to the minus nine

$$
\begin{aligned}
&= G\left(-\frac{m_1}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12} + \frac{m_3}{|\mathbf{r}_{23}|^3}\mathbf{r}_{23} - \frac{m_2}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12} - \frac{m_3}{|\mathbf{r}_{13}|^3}\mathbf{r}_{13}\right)\\
&= G\left(-\frac{m_1+m_2}{|\mathbf{r}|^3}\mathbf{r} + \frac{m_3}{|\mathbf{r}_{23}|^3}[\mathbf{r}_3-\mathbf{r}_2] - \frac{m_3}{|\mathbf{r}_{13}|^3}[\mathbf{r}_3-\mathbf{r}_1]\right)\\
&= \frac{G\mu}{|\mathbf{r}|^3}\mathbf{r} + Gm_3\left(\frac{\rho-(m_1/\mu)\mathbf{r}}{r_{23}^3} - \frac{\rho+(m_2/\mu)\mathbf{r}}{r_{13}^3}\right)
\end{aligned}
$$

(iv) First we need an identity relating $\mathbf{r}_{cm_{12}}$ and $\mathbf{r}_3$. Shuffling terms gives

$$
\begin{aligned}
\ddot{\mathbf{r}}_{cm_{12}} &= \frac{1}{m_1+m_2}[m_1\ddot{\mathbf{r}}_1 - m_2\ddot{\mathbf{r}}_2]\\
&= \frac{1}{m_1+m_2}G\left(\frac{m_1m_2}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12} + \frac{m_1m_3}{|\mathbf{r}_{13}|^3}\mathbf{r}_{13} + \frac{m_2m_1}{|\mathbf{r}_{21}|^3}\mathbf{r}_{21} + \frac{m_2m_3}{|\mathbf{r}_{23}|^3}\mathbf{r}_{23}\right)\\
&= \frac{1}{m_1+m_2}G\left(\frac{m_1m_2}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12} + \frac{m_1m_3}{|\mathbf{r}_{13}|^3}\mathbf{r}_{13} - \frac{m_2m_1}{|\mathbf{r}_{12}|^3}\mathbf{r}_{12} + \frac{m_2m_3}{|\mathbf{r}_{23}|^3}\mathbf{r}_{23}\right)\\
&= \frac{m_3}{m_1+m_2}G\left(-\frac{m_1}{|\mathbf{r}_{31}|^3}\mathbf{r}_{31} - \frac{m_2}{|\mathbf{r}_{32}|^3}\mathbf{r}_{32}\right)\\
&= -\frac{m_3}{m_1+m_2}\ddot{\mathbf{r}}_3
\end{aligned}
$$

Then

$$
\begin{aligned}
\ddot{\rho} &= \ddot{\mathbf{r}}_3 - \ddot{\mathbf{r}}_{cm_{12}}\\
&= G\left(\frac{m_1}{|\mathbf{r}_{31}|^3}\mathbf{r}_{31} + \frac{m_2}{|\mathbf{r}_{32}|^3}\mathbf{r}_{32}\right) - \left(-\frac{m_3}{m_1+m_2}\ddot{\mathbf{r}}_3\right)
\end{aligned}
$$

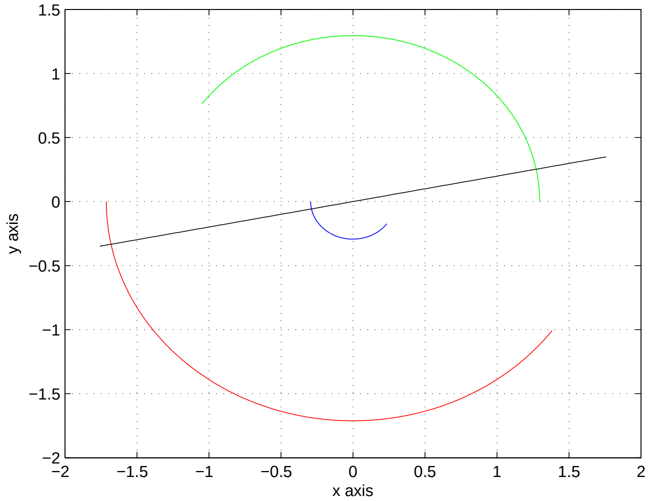

Figure 13: A partial period of the third data set.

$$
\begin{aligned}
&= G\left(\frac{m_1}{|\mathbf{r}_{31}|^3}\mathbf{r}_{31} + \frac{m_2}{|\mathbf{r}_{32}|^3}\mathbf{r}_{32}\right) + G\frac{m_3}{m_1+m_2}\left(\frac{m_1}{|\mathbf{r}_{31}|^3}\mathbf{r}_{31} + \frac{m_2}{|\mathbf{r}_{32}|^3}\mathbf{r}_{32}\right) \\
&= G\frac{m_1}{|\mathbf{r}_{31}|^3}\left(\frac{m_3}{m_1+m_2}\mathbf{r}_{31} + \mathbf{r}_{31}\right) + G\frac{m_2}{|\mathbf{r}_{23}|^3}\left(\frac{m_3}{m_1+m_2}\mathbf{r}_{32} + \mathbf{r}_{32}\right) \\
&= G\frac{m_1}{|\mathbf{r}_{31}|^3}\frac{-m_3-m_1-m_2}{m_1+m_2}(\mathbf{r}_3-\mathbf{r}_1) + G\frac{m_2}{|\mathbf{r}_{23}|^3}\frac{-m_3-m_1-m_2}{m_1+m_2}(\mathbf{r}_3-\mathbf{r}_2) \\
&= \frac{MG(m_1/\mu)}{r_{13}^3}[\rho + (m_2/\mu)\mathbf{r}] - \frac{MG(m_2/\mu)}{r_{23}^3}[\rho - (m_1/\mu)\mathbf{r}]
\end{aligned}
$$

□

Suppose we change coordinates to an inertial frame with origin at the center of mass. Our new coordinates will be

$$\mathbf{r}_{01} = \mathbf{r}_{cm} - \mathbf{r}_1 \qquad \mathbf{v}_{01} = \mathbf{v}_{cm} - \mathbf{v}_1$$

$$\mathbf{r}_{02} = \mathbf{r}_{cm} - \mathbf{r}_2 \qquad \mathbf{v}_{02} = \mathbf{v}_{cm} - \mathbf{v}_2$$

$$\mathbf{r}_{03} = \mathbf{r}_{cm} - \mathbf{r}_3 \qquad \mathbf{v}_{03} = \mathbf{v}_{cm} - \mathbf{v}_3$$

We want to compute the angular momentum in these coordinates. This gives

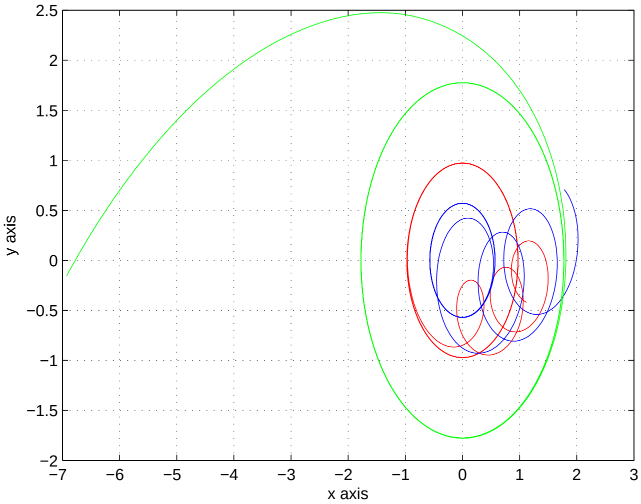

Figure 14: trajectories to 1.5 times the escape time

$$
\begin{aligned}
\mathbf{h} &= \sum_{i=1}^{3} m_i \ \mathbf{r}_{0i} \times \mathbf{v}_{0i} \\
&= \sum_{i=1}^{3} m_i (\mathbf{r}_{cm} - \mathbf{r}_i) \times (\mathbf{v}_{cm} - \mathbf{v}_i) \\
&= \sum_{i=1}^{3} \frac{m_i}{M} \left[\left(\sum_{j=1}^{3} m_j \mathbf{r}_j - \mathbf{r}_i \right) \times \left(\sum_{i=1}^{3} m_j \mathbf{v}_j - \mathbf{v}_i \right) \right]
\end{aligned}
$$

This computation gets quite painful, but many of the terms are similar to each other. For example, consider the two terms

$$
\begin{aligned}
\mathbf{r}_{cm} - \mathbf{r}_1 &= m_1 \left(\frac{1}{M} (m_1 \mathbf{r}_1 + m_2 \mathbf{r}_2 + m_3 \mathbf{r}_3) - \mathbf{r}_1 \right) \\
&= \frac{m_1}{M} (m_2 \mathbf{r}_2 - m_2 \mathbf{r}_1 + m_3 \mathbf{r}_3 - m_3 \mathbf{r}_1) \\
&= \frac{m_1}{M} \left[m_2 \mathbf{r} + m_3 \left(\rho + \frac{m_2}{\mu} \mathbf{r} \right) \right] \\
&= \frac{m_1}{\mu M} (m_1 m_2 + m_2 m_2 + m_3 m_2) \mathbf{r} + m_3 \mu \rho)
\end{aligned}
$$

and

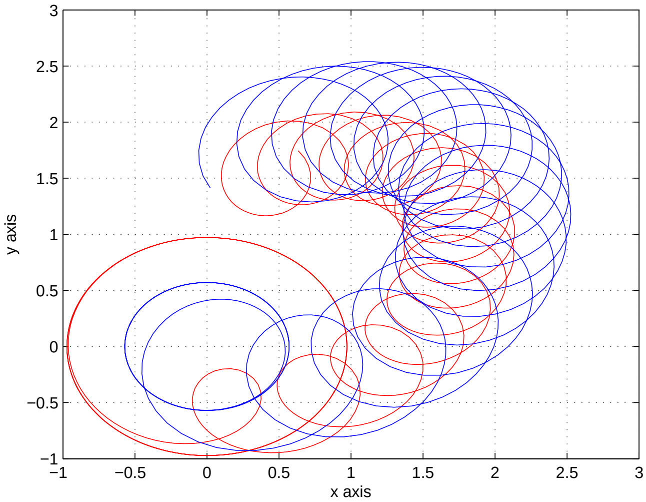

Figure 15: The two more massive bodies after the ejection of body three

$$\begin{aligned}
\mathbf{v}_{cm} - \mathbf{v}_1 &= \frac{1}{M}\left[m_2\dot{\mathbf{r}} + m_3\left(\dot{\rho} + \frac{m_2}{\mu}\dot{\mathbf{r}}\right)\right] \\
&= \frac{1}{M\mu}[(m_1m_2 + m_2m_2 + m_2m_3)\dot{\mathbf{r}} + m_3\mu\dot{\rho}]
\end{aligned}$$

Evaluating the cross product gives

$$(\mathbf{r}_{cm} - \mathbf{r}_1) \times (\mathbf{v}_{cm} - \mathbf{v}_1) \quad =$$

$$\frac{m_1}{\mu M}(m_1m_2 + m_2m_2 + m_3m_2)\mathbf{r} + m_3\mu\rho) \times \frac{1}{M\mu}[(m_1m_2 + m_2m_2 + m_2m_3)\dot{\mathbf{r}} + m_3\mu\dot{\rho}] =$$

$$\frac{m_1}{M^2}\left[\frac{m_2m_2M^2}{\mu^2}\mathbf{r} \times \dot{\mathbf{r}} + \frac{m_3m_2M}{\mu^2}\rho \times \dot{\mathbf{r}} + \frac{m_3m_2M}{\mu}\mathbf{r} \times \dot{\rho} + m_3m_3\rho \times \dot{\rho}\right]$$

The other terms are similar to these, though certainly not identical. In fact, carrying out the remaining algebra, one finds that

$$\mathbf{h} =$$

$$\frac{m_1}{M^2}\left[\frac{m_2 m_2 M^2}{\mu^2}\mathbf{r}\times\dot{\mathbf{r}} + \frac{m_3 m_2 M}{\mu^2}\rho\times\dot{\mathbf{r}} + \frac{m_3 m_2 M}{\mu}\mathbf{r}\times\dot{\rho} + m_3 m_3 \rho\times\dot{\rho}\right] +$$

$$\frac{m_2}{M^2}\left[\frac{m_1 m_1 M^2}{\mu^2}\mathbf{r}\times\dot{\mathbf{r}} - \frac{m_1 m_3 M}{\mu^2}\dot{\mathbf{r}}\times\rho - \frac{m_1 m_3 M}{\mu}\dot{\rho}\times\mathbf{r} + m_3 m_3 \rho\times\dot{\rho}\right] +$$

$$+\frac{m_3\mu}{M^2}\rho\times\dot{\rho}$$

$$= \frac{m_1 m_2 m_2 + m_2 m_1 m_1}{\mu^2}\mathbf{r}\times\dot{\mathbf{r}} + \frac{m_1 m_3 m_3 + m_2 m_3 m_3 + m_3\mu^2}{M^2}\rho\times\dot{\rho}$$

$$= \frac{m_1 m_2}{\mu}\mathbf{r}\times\dot{\mathbf{r}} + m_3\frac{\mu}{M}\rho\times\dot{\rho}$$

which is a convenient expression for the angular momentum in Jacobi Coordinates.

5 Acceleration in Rotating Reference Frames

We recall that by definition an inertial reference frame is one where Newton's first law of motion holds; i.e. one in which a body which is not acted upon by any external force will move in a straight line with constant velocity. It is a fundamental assumption of classical mechanics that such frames exist.

Consider two vectors $\mathbf{r}_0$, $\mathbf{r}_m \in \mathbb{R}^2$, and their sum $\mathbf{r}_i = \mathbf{r}_0 + \mathbf{r}_m$. Let $\mathbf{r}_m$ give the position at time $t = 0$ of a particle P with mass m and assume the standard basis $\{\mathbf{i}, \mathbf{j}\}$ of $\mathbb{R}^2$, where $\mathbf{i} = (1,0,0), \mathbf{j} = (0,1,0)$, and $\mathbf{k} = (0,0,1)$ is an inertial frame for this particle.

Now we want to consider a second, orthonormal non-inertial reference frame with origin at $\mathbf{r}_0$ and rotating with angular velocity ω. There are two common and natural descriptions of this coordinate system, and the relation between the inertial and non-inertial kinematics, and we hope to show that these are equivalent.

First, we can describe the relationships in terms of linear and affine transformations of $\mathbf{R}^2$. Suppose that the position of the origin of the rotating reference frame $\mathbf{r}_0$ is a known function of time expressed in the inertial coordinates. We denote this as $\mathbf{r}_0^I(t)$ where the I expresses the fact that the geometric object object $\mathbf{r}_0$ is given in the inertial coordinates.

There are two (instantaneously) orthonormal vectors $\{\mathbf{e}_1, \mathbf{e}_2\}$ based at $\mathbf{r}_0$ which are rotating with angular velocity $\boldsymbol{\omega}$. Explicitly let $R^{im}(t)$ be a 2×2 rotation matrix. Any such matrix is uniquely defined by an angle of rotation θ as

$$R^{im} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

taking ω to be the instantaneous angular velocity we have $\dot{\theta} = \omega$. Then the basis vectors in the rotating frame are explicit functions of the standard basis vectors. For example

$$\begin{aligned} \mathbf{e}_1^I(t) &= \mathbf{r}_0(t) + R^{im}(t)\mathbf{i} \\ &= \mathbf{r}_0(t) + R^{im}(t)\begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \mathbf{r}_0(t) + \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \end{aligned}$$

Now, the velocity of $\mathbf{e}_1(t)$ is

$$\frac{d}{dt}\mathbf{e}_1^I = \dot{\mathbf{r}_0} + \dot{R^{im}\mathbf{i}} = \mathbf{v}_0 + \frac{d}{dt}(R^{im}\mathbf{i})$$

The derivative of the matrix can be explicitly computed.

$$\begin{aligned} \frac{d}{dt}(R^{im}\mathbf{i}) &= \frac{d}{dt}\left(R^{im}\begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) \\ &= \frac{d}{dt}\left(\begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}\right) \\ &= \begin{bmatrix} -(\sin\theta)\ \dot{\theta} \\ (\cos\theta)\ \dot{\theta} \end{bmatrix} \\ &= \begin{bmatrix} -(\sin\theta)\ \omega \\ (\cos\theta)\ \omega \end{bmatrix} \end{aligned}$$

This vector has magnitude ω and is perpendicular to $R^{im}(t)\mathbf{i}$.
Similarly

$$\begin{aligned} \mathbf{e}_2^I(t) &= \mathbf{r}_0(t) + R^{im}(t)\mathbf{j} \\ &= \mathbf{r}_0(t) + \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} \end{aligned}$$

and

$$\frac{d}{dt}(R^{im}\mathbf{j}) = \begin{bmatrix} -(\cos\theta)\ \omega \\ -(\sin\theta)\ \omega \end{bmatrix}$$

Then let

$$\mathbf{r}_m^M = a\,\mathbf{e}_1 + b\,\mathbf{e}_2$$

be an arbitrary vector expressed in the rotating frame. Compute

$$\begin{aligned}
\mathbf{r}_m^I &= \mathbf{r}_0 + R^{im}\mathbf{r}_m^M \\
&= \mathbf{r}_0 + R^{im}\begin{bmatrix} a \\ b \end{bmatrix} \\
&= \mathbf{r}_0 + a\,R^{im}\mathbf{i} + b\,R^{im}\mathbf{j} \\
&= \mathbf{r}_0 + a\begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} + b\begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} \\
&= \mathbf{r}_0 + \begin{bmatrix} a\cos\theta - b\sin\theta \\ a\sin\theta + b\cos\theta \end{bmatrix}
\end{aligned}$$

We name this last vector

$$\mathbf{r}_m^0 = (a\cos\theta - b\sin\theta, a\sin\theta + b\cos\theta)^T = (c,d)T \tag{5}$$

as it is the geometric object $\mathbf{r}_m$ expressed in inertial coordinates that are centered at $\mathbf{r}_0$; the origin of the rotating frame. Then

$$\begin{aligned}
\frac{d}{dt}\mathbf{r}_m^I &= \mathbf{v}_0 + a\dot{R^{im}}\mathbf{i} + b\dot{R^{im}}\mathbf{j} && (6) \\
&= \mathbf{v}_0 + a\begin{bmatrix} -(\sin\theta)\ \omega \\ (\cos\theta)\ \omega \end{bmatrix} + b\begin{bmatrix} -(\cos\theta)\ \omega \\ -(\sin\theta)\ \omega \end{bmatrix} && (7) \\
&= \mathbf{v}_0 + -\omega\begin{bmatrix} a\sin\theta + b\cos\theta \\ -a\cos\theta + b\sin\theta \end{bmatrix} && (8) \\
&= \mathbf{v}_0 - \omega(a\sin\theta + b\cos\theta, -a\cos\theta + b\sin\theta)^T && (9) \\
&= \mathbf{v}_0 - \omega(d,-c)^T && (10) \\
& && (11)
\end{aligned}$$

Then, letting $\boldsymbol{\omega} = (0,0,\omega)$ and observing that

$$\boldsymbol{\omega} \times (c,d)^T = \omega(-d,c)^T = -\omega(d,-c)^T$$

we have

$$\begin{aligned}
\frac{d}{dt}\mathbf{r}_m^I &= \mathbf{v}_0 + \frac{d}{dt}\mathbf{r}_m^0 \\
&= \mathbf{v}_0 + \boldsymbol{\omega} \times (c,d)^T \\
&= \mathbf{v}_0 + \boldsymbol{\omega} \times \mathbf{r}_m^0
\end{aligned}$$

which is the transformation law as commonly found in physics texts and literature. This is assuming that $\mathbf{r}_m^M$ and hence a, and b are constant in the rotating frame. If in addition the vector is moving in this frame we will have additional terms in (3) of the form

$$\dot{a}\begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} + \dot{b}\begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} \dot{a} \\ \dot{b} \end{bmatrix} = R^{im}\dot{\mathbf{r}}_m^M = R^{im}\mathbf{v}_m^M$$

Then to transform the coordinates of a particle experiencing general motion in the rotating into coordinates in the inertial frame we must employ the transformation

$$\begin{aligned} \frac{d}{dt}\mathbf{r}_m^I(t) &= \mathbf{v}_0 + R^{im}\mathbf{v}_m^M + \frac{d}{dt}\mathbf{r}_m^0 \\ &= \mathbf{v}_0 + \mathbf{v}_m^0 + \boldsymbol{\omega} \times \mathbf{r}_m^0 \end{aligned}$$

But now, since this law applies to any vector in the rotating frame, it applies just as well to $\mathbf{v}_m^M$ in which case

$$\begin{aligned} \frac{d}{dt}\mathbf{v}_m^I(t) &= \frac{d}{dt}\mathbf{v}_0 + \frac{d}{dt}\mathbf{v}_m^0 + \frac{d}{dt}\omega \times \mathbf{r}_m^0 && (12) \\ &= \mathbf{a}_0 + (R^{im}\mathbf{v}_m^0)' + (\omega R^{im}\mathbf{r}_m^0)' && (13) \\ &= \mathbf{a}_0 + \omega \times \mathbf{v}_m^0 + \mathbf{a}_m^0 + \dot{\omega} \times \mathbf{r}_m^0 + \omega \times (\omega \times \mathbf{r}_m^0 + \mathbf{v}_m^0) && (14) \\ &= \mathbf{a}_0 + \mathbf{a}_m^0 + 2\omega \times \mathbf{v}_m^0 + \dot{\omega} \times \mathbf{r}_m^0 + \omega \times \omega \times \mathbf{r}_m^0 && (15) \end{aligned}$$

as the preceding computations show that $\frac{d}{dt}R^{im} = \omega\times$.

A second approach to deriving the expressions for the acceleration in the inertial coordinates is simply to differentiate the expression

$$\mathbf{r}_m^I = \mathbf{r}_0 + R^{im}\mathbf{r}_n^M$$

twice with respect to time. This is

$$\begin{aligned} \mathbf{a}_m^I &= \frac{d^2}{dt^2}\mathbf{r}_m^I(t) && (16) \\ &= \frac{d}{dt}\frac{d}{dt}\left(\mathbf{r}_0 + R^{im}\mathbf{r}_m^M\right) && (17) \\ &= \mathbf{a}_0 + \frac{d}{dt}\left((R^{im})'\mathbf{r}_m^M + R^{im}(\mathbf{r}_m^M)'\right) && (18) \\ &= \mathbf{a}_0 + (R^{im})''\mathbf{r}_m^M + (R^{im})'(\mathbf{r}_m^M)' + (R^{im})'(\mathbf{r}_m^M)' + R^{im}(\mathbf{r}_m^M)'' && (19) \\ &= \mathbf{a}_0 + \mathbf{a}_m^0 + 2(R^{im})'(\mathbf{v}_m^M) + (R^{im})''\mathbf{r}_m^M && (20) \end{aligned}$$

Equating (11) and 16 gives the desired result. (Actually to obtain the result in the homework apply R^{mi} to both sides and use that $(R^{im})^{-1} = (R^{mi})^T$, but the computations are almost identical).

The computation has been carried out in only two dimensions. However, all rotations in three space have an axis of symmetry and can be expressed as a planar rotation in the proper coordinates. The coordinate change that takes the three dimensional rotation to a planar one is an affine coordinate change (a single fixed rotation with determinant absolute value one) and as such does not alter the equations of motion. (Proof of this was given in note set one).

By combining these results the case of a three dimensional rotational change of coordinates is covered by the planar case. Then The virtue of the method used here to derive these kinematic equations is that it does not involve any infinitesimals or any appeal to the intuitive suggestion that for small enough angles rotations commute.

A MatLab Code

A.1 The program for the equilateral solutions

```
%Working with equilateral triangle configurations of N-Body problem
%The givens are specified below.  For any set of given date the program
%computes the initial data for a triangular orbit and simulates the
%system

%-------------------------DATA---------------------------------------------

%Clear the workspace
clear
%This will be a three body problem
N=3;

%Givens:
    %gravitational constant
    G=1.0;
    %initial position of center of mass in inertial frame
    Rcm=[0.0; 0.0; 0.0];
    %initial velociety of center of mass in inertial frame
    Vcm=[-1.0; 1.0; 0.0];
    %components of normal vector to the desired plane of motion of the
    %triangular orbit (specifies the desired plane)
    x=0.0;
    y=0.0;
    z=1.0;
    %Masses
    m1=1.0;
    m2=2.0;
    m3=3.0;
    Mass=[m1 m2 m3];
```

```
    M=m1+m2+m3;

%Compute a set of initial conditions which put these bodies in a
%equilateral configuration with side of length r;

%length of triangle
   r=1.0;

%---------------------------------------------------------------------------
%-----------------Computation of Initial Conditions-------------------------
%---------------------------------------------------------------------------


%These conditions determine such an orbital configuration.
%The configuration can be computed by;

%Compute the angular velociety;  (Output to screne)

omega=sqrt((G/(r)^3)*M)

%Distance from center of triangle/origin to any vertex is;

l=r/sqrt(3)

%Positions referenced from the center of the triangle (coordinate
%frame with origin at centerr of triangle)

r1_tc=[0;l;0]
r2_tc=[l*cos((210/180)*pi); l*sin((210/180)*pi); 0]
r3_tc=[l*cos((330/180)*pi); l*sin((330/180)*pi); 0]

%Compute the center of Mass in this frame

Rcm_tc=(1/M)*(m1*r1_tc + m2*r2_tc + m3*r3_tc)

%Change to a frame with the center of mass as the origin

r1_cm=r1_tc-Rcm_tc
r2_cm=r2_tc-Rcm_tc
r3_cm=r3_tc-Rcm_tc

%this is thought of as a rotating frame with the CM at the origin
%the frame rotates with angular velociety

omega_vector=[0;0; omega]
```

```
%In this frame the positions and velocities are

r1_rot=r1_cm;
r2_rot=r2_cm;
r3_rot=r3_cm;
v1_rot=[0.0; 0.0; 0.0];
v2_rot=[0.0; 0.0; 0.0];
v3_rot=[0.0; 0.0; 0.0];

%Now define yet another frame; the m-frame with unitvectors xhat, yhat and
%zhat based at the center of mass as follows

zhat=([x;y;z])/norm([x;y;z]);

%The zhat vector defined above will be normal to the plane of the
%triangular orbit.  We are free to choose it to point in any direction we
%like here, which amounts to being able to put the triangular orbit in any
%desired plane.  (if x=0, y=0, z=1 then we have specified the xy-plane)

%From zhat we work out the complete inertial frame

ihat=[1; 0; 0];
jhat=[0; 1; 0];
khat=[0; 0; 1];

ahat=(cross(ihat,zhat)/norm(cross(ihat,zhat)));
xhat=cross(zhat,ahat);
yhat=cross(zhat,xhat);

%now define the matrix which takes vectors in the CM frame to the inertial
%reference frame is

R_mi=[dot(ihat,xhat), dot(ihat,yhat), dot(ihat,zhat);
      dot(jhat,xhat), dot(jhat,yhat), dot(jhat,zhat);
      dot(khat,xhat), dot(khat,yhat), dot(khat,zhat)];


%The initial position and velociety of the center of mass in the inertial
%frame are free as well, however these are given in the problem data above
%So, compute the positions and velocities in the inertial frame with
%coordinates in the rotating frame

r1_im=r1_rot;
r2_im=r2_rot;
r3_im=r3_rot;
v1_im=v1_rot+cross(omega_vector, r1_rot);
```

```
v2_im=v2_rot+cross(omega_vector, r2_rot);
v3_im=v3_rot+cross(omega_vector, r3_rot);

%Now these can be transformed into inertial coordinates to give the desired
%initial positions and velocities in inertial coordinates

r1_i=Rcm+R_mi*r1_im
r2_i=Rcm+R_mi*r2_im
r3_i=Rcm+R_mi*r3_im
v1_i=Vcm+R_mi*v1_im
v2_i=Vcm+R_mi*v2_im
v3_i=Vcm+R_mi*v3_im



%---------------------------------------------------------------------------
%----SIMULATE the system with these parameters and this initial state------
%---------------------------------------------------------------------------

%initial state
y=[r1_i; r2_i; r3_i; v1_i; v2_i; v3_i];

%integrate over some number of periods
Period=(2*pi)/omega
%number of periods to integrate over
tf=(20)*Period
tspan=linspace(0,tf,1000);

%numerical integration
y0=y';                                                      %inital condition
options=odeset('RelTol',1e-14,'AbsTol',1e-22);              %set tolerences
%initialize x
x=0;
[t,x]=ode113('ThreeBodyWpar',tspan,y0,options,[],G,m1,m2,m3);


%--------------------------------------------------------------------------
%----------------------------------ANALYSIS------------------------------
%--------------------------------------------------------------------------



%want to plot the solution in a frame centered at the CM and rotating with
%the bodies

%get the size of the time vector
```

```
checkX=size(x)
temp=size(t)
numTimeSteps=temp(1,1)
%The time step 'h' is
h=tf/(numTimeSteps-1);

%initialize loop variables
s1=0;
s2=0;
s3=0;

for n=1:numTimeSteps
    %compute the instaneous center of mass
    Rcm_ti=(1/M)*(m1*x(n,1:3)' + m2*x(n,4:6)' + m3*x(n,7:9)');
    Vcm_ti=(1/M)*(m1*x(n,10:12)' + m2*x(n,13:15)' + m3*x(n,16:18)');
    %compute the instaneous rotation matrix
    theta_ti=omega*(n-1)*h;
    RotMat_ti=[cos(theta_ti),sin(theta_ti),0;
               -sin(theta_ti),cos(theta_ti),0;
               0,0,0];
    %compute the instaneous state in the rotating frame
    O=zeros(3);
    x_cmi=[Rcm_ti; Rcm_ti; Rcm_ti; Vcm_ti; Vcm_ti; Vcm_ti];
    x_om(n,1:18)=[x(n,:) - x_cmi'];
    x_m(n,1:18)=([RotMat_ti, O, O, O, O, O;
                 O, RotMat_ti, O, O, O, O;
                 O, O, RotMat_ti, O, O, O;
                 O, O, O, RotMat_ti, O, O;
                 O, O, O, O, RotMat_ti, O;
                 O, O, O, O, O, RotMat_ti]*x_om(n,1:18)')';
    %compute the length of the sides of the triangle
    side1(n)=norm(x(n,1:3)-x(n,4:6));
    side2(n)=norm(x(n,1:3)-x(n,7:9));
    side3(n)=norm(x(n,4:6)-x(n,7:9));
    %compute the difference between the initial x_m and the current x_m
    %(Actually only the position components)
    error(n,1)=norm(x_m(1,1:9)-x_m(n,1:9));
    %Compute the forces
    for i=1:N
    for j=1:N
            Rij=(x(n,3*i-2)-x(n,3*j-2))^2+(x(n,3*i-1)-x(n,3*j-1))^2+(x(n,3*i)-x(n,3*j))^
            %compute the three components of acceleration i
            if j~=i
               s1=s1+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j-2)-x(n,3*i-2));
               s2=s2+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j-1)-x(n,3*i-1));
```

```
                s3=s3+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j)-x(n,3*i));
             else
                s1=s1+0;
                s2=s2+0;
                s3=s3+0;
             end
     end
     force(n,3*i-2)=G*Mass(1,i)*s1;
     force(n,3*i-1)=G*Mass(1,i)*s2;
     force(n,3*i)=G*Mass(1,i)*s3;
     s1=0;
     s2=0;
     s3=0;
      %Check that the force points to the center of mass; It does if the
      %cross product is allways zero
      forceVector_i=[force(n,3*i-2);force(n,3*i-1);force(n,3*i)];
      forceCmCheck(n,i)=(i-1)+norm(cross(forceVector_i,x_om(n,(3*i-2):(3*i))'));
     end
end


%-------------------------------------------------------------------------
%-------------------------PLOTS-------------------------------------------
%-------------------------------------------------------------------------

%Force crossed with the direction vector between mass and CM
%plot(t(:),forceCmCheck(:,1),'r',t(:),forceCmCheck(:,2),'b',t(:),forceCmCheck(:,3),'g')

%'x' coordinate of mass one as a function of time; numerically confirms
%the period
%plot(t(:),x_om(:,1),'b')

%Length of sides as a function of time
plot(t(:),side1(:),'r',t(:),side2(:)+1,'b',t(:),side3(:)+2,'g')

%CM centered frame  (NOT ROTATING)
%plot(x_om(:,1), x_om(:,2),'r', x_om(:,4), x_om(:,5),'b', x_om(:,7), x_om(:,8),'g')

%CM centered Rotating Frame
%plot(x_m(:,1), x_m(:,2),'r+', x_m(:,4), x_m(:,5),'b*', x_m(:,7), x_m(:,8),'gs')

%This plot examines the drift in the deviation from stationary in the
%rotating frame
%plot(t(:), error(:,1), 'b')

%plot an error estimate as time progresses
```

```
%plot(t(:), error(:),'b')

%Plot the length of the radius vector.  If constant then the orbit is a
%circle.  If it oscillates then the orbit is an ellipse and the peiod of
%the sineusiod gives the period of the rotation
%plot(t(:),norm(x_om(:,1:3)),'r', t(:), norm(x_om(:,4:6)),'b', t(:),norm(x_om(:,7:9)),'g
%(Result:  they look constant which suggests each body is in a circular
%orbit).

%plot the trajectory in inertial frame
%plot(x(:,1), x(:,2),'r', x(:,4), x(:,5),'b', x(:,7), x(:,8),'g')


grid on
xlabel('x axis'), ylabel('y axis')
```

A.2 Euler Straight Line Programs

```
%This program studies the Euler straight line solutions of the general
%three body problem.  Primarly it is desinged to be used to estimates find
%escape times; times when the system is outside a given tolerence from
%the equilibrium state

%-------------------------DATA---------------------------------------------

%clear workspace
clear

%This will be a three body problem
N=3;

%Givens:
    %Gravatational constant
    G=1.0;
    %Angular velociety
    omega=1.0
    %masses
    m1=3.0;
    m2=2.0;
    m3=1.0;
```

```
    M=m1+m2+m3;
    Mass=[m1 m2 m3];
    %initial center of mass in inertial coordinates
    Rcm=[0; 0; 0];
    %Initial velociety of CM in inertial coordinates
    Vcm=[0; 0; 0];
    %The line the particls lie on must fall through the CM at time zero but
    %may have any desired orientation.  Specify a direction vector for the
    %line
    d=[1; 0; 0];
    %Convert to unit vector
    u_d=d/norm(d);
    %This line rotates in some plane.Specify the plane by giving the normal
    %vector to it
    u_n=[0; 0; 1];

%-------------------------------------------------------------------------
%-------------Computation of initial conditions---------------------------
%-------------------------------------------------------------------------

%Let r1=x1, r2=x2, r3=x3 and suppose x1<x2<x3.
%Then follow the standard procedure for porducing coloiear orbits

%First solve the polynomial
%  0=(m2 + m3) + (2*m2 + 3*m3)b + (3*m3 + m2)b^2
%              - (3*m1 + m2)b^3 - (3*m1 + 2*m2)b^4 - (m1 + m2)b^5
%for b

%polynominal in matlab form
p=[-(m1+m2) -(3*m1+2*m2) -(3*m1+m2) (3*m3+m2) (2*m2+3*m3) (m2+m3)];

%solve for b
r=roots(p)

%for physical reasons p has only one positive real root.  Find it and call
%it 'b'.
k=find(r>0);
b=r(k)

%then compute 'a'

a=((1/(m2+m3*(1+b)))*((G*M)/(omega^2))*(m2+m3/(1+b)^2))^(1/3)

%Name the relative distances;
%             R12=x2-x1
%             R13=x3-x1
```

```
%              R31=x1-x3
%              R32=x2-x3
%then
R12=a;
R13=a+a*b;
R31=-(a+a*b);
R32=-a*b;

%then explicitly compute the x coordinates of the masses (in the rotating
%reference frame)
x1=-(G/omega^2)*(m2/(R12^2)+m3/(R13^2))
x3=(G/omega^2)*(m1/(R31^2)+m2/(R32^2))
x2=-(m1*x1+m3*x3)/m2

%These are the positions in the rotating frame centered at the CM
%rotating with angular velociety omega.  We use this information to compute
%the initial velocities.  The xi coordinates are the distances from the
%center of mass, and the points move in a circle around the CM with radius
%equal to xi and angular velociety equal to omega then the velocitiel can
%be computed via the formula v=omega*xi

xdot1=omega*x1
xdot2=omega*x2
xdot3=omega*x3

%To change to inertial coordinates we "vectorize" all quantities (here we
%are essentially changing coordinates to an inertial frame centered at the
%center of mass
r1_cm=[x1; 0; 0];
r2_cm=[x2; 0; 0];
r3_cm=[x3; 0; 0];

%Now convert to inertial frame
r1_i=Rcm+x1*d;
r2_i=Rcm+x2*d;
r3_i=Rcm+x3*d;

%the velociety vectors lie in the plane determined by the line through the
%cm with direction u_d and with normal vector u_n.  A first direction in the
%plane is give by d.  A second direction, perpendicular to d and u_n is
%given by u_n x u_d.

u_v=cross(u_n, u_d)/norm(cross(u_n, u_d));

%Now the velociety in the inertial frame centered at the cm can be computed
```

```
v1_cm=xdot1*u_v;
v2_cm=xdot2*u_v;
v3_cm=xdot3*u_v;

%and the innital velocities in the inertial frame are
v1_i=Vcm+v1_cm
v2_i=Vcm+v2_cm
v3_i=Vcm+v3_cm

%---------------------------------------------------------------------------
%------------------------Simulate the System-------------------------------
%---------------------------------------------------------------------------



%initial state
y=[r1_i; r2_i; r3_i; v1_i; v2_i; v3_i];

%integrate over some number of periods
Period=(2*pi)/omega

%Now we want to integrate the system, and hold the energy error under
%10^-12.  We want to let the integration run untill the distance from one
%of the particles to the center of mass is more than five percent of its
%initial value.

%Start by integrating the system for a short time
    tf=(0.05)*Period
    tspan=linspace(0,tf,500);
    %numerical integration
    y0=y';                                                  %inital condition
    options=odeset('RelTol',1e-14,'AbsTol',1e-22);          %set tolerences
    [t,x]=ode113('ThreeBodyWpar',tspan,y0,options,[],G,m1,m2,m3);


%From this we can do and an initial energy computation.

A=size(x);                 %vector containing (rows,columns)
timeMax=A(1,1);             %number of rows of y is number of time steps

%Energy: (T-U)(t)

%Potential Energy:  U(t)
U=0;
for i=1:timeMax
    S=0;             %Initialize S
```

```
        for j=1:N
            s=0;
            for k=1:N
                Rjk=(x(i,3*j-2)-x(i,3*k-2))^2+(x(i,3*j-1)-x(i,3*k-1))^2+(x(i,3*j)-x(i,3*
                if k~=j
                    s=s+Mass(1,k)/[sqrt(Rjk)];
                else
                   s=s+0;
                end
            end
            S=S+Mass(1,j)*s;
        end
    U(i)=(G/2)*S;
end
%Kenetic Energy;  T(t)
T=0;
for i=1:timeMax
    S=0;              %Initialize S
    for j=1:N
        S=S+Mass(1,j)*[(x(i,3*N+(3*j-2)))^2+(x(i,3*N+(3*j-1)))^2+(x(i,3*N+(3*j)))^2];
    end
    T(i)=S/2;
end

%Tenth integral; Energy   E(t)
E=0;
for i=1:timeMax
    E(i)=T(i)-U(i);
end

%E(1) is the true initial energy, not a loop variable
%Then give it a special name
initEnergy=E(1);

%initialize the energy error 'deltEnergy'
deltaEnergy=abs(initEnergy-E(timeMax));

%initialize the measure of the percent difference of the distances from
%their initial value
%compute the initial center of mass
%This is not a loop variable either and should not be changed in the loop
Rcm_t0=(1/M)*(m1*x(1,1:3)' + m2*x(1,4:6)' + m3*x(1,7:9)');
%compute the initial distance from the bodies to the CM
%Again, these are not loop variables and will not be changed in the loop
InitDistM1CM=norm(x(1,1:3)'-Rcm_t0);
InitDistM2CM=norm(x(1,4:6)'-Rcm_t0);
```

```
InitDistM3CM=norm(x(1,7:9)'-Rcm_t0);
%compute the current center of mass
Rcm_tf=(1/M)*(m1*x(timeMax,1:3)' + m2*x(timeMax,4:6)' + m3*x(timeMax,7:9)');
%compute the current distance from body to CM
finalDistM1CM=norm(x(timeMax,1:3)'-Rcm_tf);
finalDistM2CM=norm(x(timeMax,4:6)'-Rcm_tf);
finalDistM3CM=norm(x(timeMax,7:9)'-Rcm_tf);
%compute the relative error between the current and initial distances to
%the mass center
error1=finalDistM1CM/InitDistM1CM;
error2=finalDistM2CM/InitDistM2CM;
error3=finalDistM3CM/InitDistM3CM;
%the actual percent err or will be the worst of these
percentError=max([abs(1-error1); abs(1-error2); abs(1-error3)]);

%-----------------------LOOP----------------------------------------------
%-----------Now start the loop.  Each time integrate a little further------
%-----------holding the energy estimate in bounds, and checking untill-----
%-----------the distance tolerence is exceeded----------------------------

%initialize the counter
n=0;
%h is the fraction of an orbit that will be traversed each loop
h=0.05;

%START THE LOOP
while percentError<0.05 & deltaEnergy<10^(-12)
%if deltaEnergy<10^(-12) & percentError<0.05
    %how many times have we done this already? (output answer)
    n=n+1
    %each time through the loop
    %we integrate over an additional
    %tenth of a period
    %reinitialize variables
    tf=0;
    tf=h*Period;
    tspan=0;
    tspan=linspace(0,tf,500);
    %during this loop we only integrate
    %the final conditions of the last
    %loop for one tenth of a period
    y0=x(timeMax,:);
    options=odeset('RelTol',5e-14,'AbsTol',1e-22);
    [t,x]=ode113('ThreeBodyWpar',tspan,y0,options,[],G,m1,m2,m3);
    %now compute the energy error for this
    %time through the loop
```

```
A=size(x);                 %vector containing (rows,columns)
timeMax=A(1,1);             %number of rows of y is number of time steps
%TEST CODE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!------------------------------
checkFinalCondLoop=x(timeMax,1:9)'
%--------------------------------------------------------------------
%Potential Energy:  U(t)
U=0;
for i=1:timeMax
    S=0;              %Initialize S
     for j=1:N
            s=0;
            for k=1:N
                Rjk=(x(i,3*j-2)-x(i,3*k-2))^2+(x(i,3*j-1)-x(i,3*k-1))^2+(x(i,3*j)-x(
                if k~=j
                  s=s+Mass(1,k)/[sqrt(Rjk)];
                else
                s=s+0;
                end
            end
        S=S+Mass(1,j)*s;
     end
    U(i)=(G/2)*S;
end
%Kenetic Energy;  T(t)
T=0;
for i=1:timeMax
    S=0;              %Initialize S
 for j=1:N
       S=S+Mass(1,j)*[(x(i,3*N+(3*j-2)))^2+(x(i,3*N+(3*j-1)))^2+(x(i,3*N+(3*j)))^2];
   end
   T(i)=S/2;
end
%Tenth integral; Energy   E(t)
E=0;
for i=1:timeMax
    E(i)=T(i)-U(i);
end
%energy error for this loop
deltaEnergy=abs(initEnergy-E(timeMax));
%Now compute the percent error for this time through the loop
%compute the current center of mass
Rcm_tf=(1/M)*(m1*x(timeMax,1:3)' + m2*x(timeMax,4:6)' + m3*x(timeMax,7:9)');
%compute the current distance from body to CM
finalDistM1CM=norm(x(timeMax,1:3)'-Rcm_tf);
finalDistM2CM=norm(x(timeMax,4:6)'-Rcm_tf);
finalDistM3CM=norm(x(timeMax,7:9)'-Rcm_tf);
```

```
    %compute the relative error between the current and initial distances to
    %the mass center
    error1=finalDistM1CM/InitDistM1CM;
    error2=finalDistM2CM/InitDistM2CM;
    error3=finalDistM3CM/InitDistM3CM;
    %the actual percent err or will be the worst of these
    percentError=max([abs(1-error1); abs(1-error2); abs(1-error3)])
end

%We recompute the orbit from zero to the break away time..
%return to the initial state as computed initially
y=[r1_i; r2_i; r3_i; v1_i; v2_i; v3_i];
%integrate untill the time just before the overflow
%reinitialize tf
tf=0;

%--------------------------------------------------------------------------
%NOTE: There is a unexpected side effect in the looping scheme used above.
%The loop integrates for h*Period, then checks the accuracy.
%
%If the accuracy is in bounds the program continues from the conditions it
%stopped at, and integrates another h*Period.  If the accuracy is out of
%bounds it halts and disreguards the last h*Period which forced the
%overflow.  T
%
%The unexpected effect is that the stoping and starting, each time
%integrating over h*Period, seems to interfeer with the step size selection
%of the integration routine.  So after the loop when we integrate from the
%same starting time to the same ending time we get slightly different
%answers (in roughly the 5th decmal place).  It is my guess that this is
%due to the step size issues just described, however there may be another
%but in the program.
%
%At any rate, the is the reason for the use of (0.0435)*Period below
%instead of (0.05)*Period as used above in the initialization run.
%-------------------------------------------------------------------------
tf=(0.0435)*Period + (n-1)*h*Period
%reinitialize tspan
tspan=0;
tspan=linspace(0, (3.5)*tf, 1000);
y0=y';
options=odeset('RelTol',1e-14,'AbsTol',1e-22);
[t,x]=ode113('ThreeBodyWpar',tspan,y0,options,[],G,m1,m2,m3);

A=size(x);                 %vector containing (rows,columns)
timeMax=A(1,1);             %number of rows of y is number of time steps
```

```
%Now compute the percent error for this time through the loop
    %compute the current center of mass
    Rcm_tf=(1/M)*(m1*x(timeMax,1:3)' + m2*x(timeMax,4:6)' + m3*x(timeMax,7:9)');
    %compute the current distance from body to CM
    finalDistM1CM=norm(x(timeMax,1:3)'-Rcm_tf);
    finalDistM2CM=norm(x(timeMax,4:6)'-Rcm_tf);
    finalDistM3CM=norm(x(timeMax,7:9)'-Rcm_tf);
    %compute the relative error between the current and initial distances to
    %the mass center
    error1=finalDistM1CM/InitDistM1CM;
    error2=finalDistM2CM/InitDistM2CM;
    error3=finalDistM3CM/InitDistM3CM;
    %the actual percent err or will be the worst of these
    percentError=max([abs(1-error1); abs(1-error2); abs(1-error3)])


%TEST CODE!!!!!!!!!!!!!!!!!!!!!!!-----------------------------------------
checkFinalState=x(timeMax,1:9)'
%------------------------------------------------------------------------



%-------------------------------------------------------------------------
%----------------------------------ANALYSIS-------------------------------
%-------------------------------------------------------------------------

%want to plot the solution in a frame centered at the CM and rotating with
%the bodies

%get the size of the time vector

temp=size(t);
numTimeSteps=temp(1,1);
%The time step 'h' is
h=tf/(numTimeSteps-1);

%Initialize loop variables
s1=0;
s2=0;
s3=0;

for n=1:numTimeSteps
    %compute the instaneous center of mass
    Rcm_ti=(1/M)*(m1*x(n,1:3)' + m2*x(n,4:6)' + m3*x(n,7:9)');
    Vcm_ti=(1/M)*(m1*x(n,10:12)' + m2*x(n,13:15)' + m3*x(n,16:18)');
    %compute the instaneous rotation matrix
```

```
theta_ti=omega*(n-1)*h;
RotMat_ti=[cos(theta_ti),sin(theta_ti),0;
           -sin(theta_ti),cos(theta_ti),0;
           0,0,0];
%compute the instaneous state in the rotating frame
O=zeros(3);
x_cmi=[Rcm_ti; Rcm_ti; Rcm_ti; Vcm_ti; Vcm_ti; Vcm_ti];
x_om(n,1:18)=[x(n,:) - x_cmi'];
x_m(n,1:18)=([RotMat_ti, O, O, O, O, O;
             O, RotMat_ti, O, O, O, O;
             O, O, RotMat_ti, O, O, O;
             O, O, O, RotMat_ti, O, O;
             O, O, O, O, RotMat_ti, O;
             O, O, O, O, O, RotMat_ti]*x_om(n,1:18)')';
%compute the length of the sides of the triangle
side1(n)=norm(x(n,1:3)-x(n,4:6));
side2(n)=norm(x(n,1:3)-x(n,7:9));
side3(n)=norm(x(n,4:6)-x(n,7:9));
%compute the difference between the initial x_m and the current x_m
%(Actually only the position components)
error(n)=norm(x_m(1,1:9)-x_m(n,1:9));
%Compute the forces
for i=1:N
for j=1:N
        Rij=(x(n,3*i-2)-x(n,3*j-2))^2+(x(n,3*i-1)-x(n,3*j-1))^2+(x(n,3*i)-x(n,3*j))^
        %compute the three components of acceleration i
        if j~=i
           s1=s1+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j-2)-x(n,3*i-2));
           s2=s2+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j-1)-x(n,3*i-1));
           s3=s3+(Mass(1,j)/(sqrt(Rij))^3)*(x(n,3*j)-x(n,3*i));
        else
           s1=s1+0;
           s2=s2+0;
           s3=s3+0;
        end
end
force(n,3*i-2)=G*Mass(1,i)*s1;
force(n,3*i-1)=G*Mass(1,i)*s2;
force(n,3*i)=G*Mass(1,i)*s3;
s1=0;
s2=0;
s3=0;
 %Check that the force points to the center of mass; It does if the
 %cross product is allways zero
 forceVector_i=[force(n,3*i-2);force(n,3*i-1);force(n,3*i)];
 forceCmCheck(n,i)=(i-1)+norm(cross(forceVector_i,x_om(n,(3*i-2):(3*i))'));
```

```
    end
end


%---------------------------------------------------------------------------
%-------------------Ploting-------------------------------------------------
%---------------------------------------------------------------------------




%'x' coordinate of mass one as a function of time; numerically confirms
%the period
%plot(t(:),x_om(:,1),'b')


%CM centered frame  (NOT ROTATING)
%plot(x_om(:,1), x_om(:,2),'r', x_om(:,4), x_om(:,5),'b', x_om(:,7), x_om(:,8),'g')

%CM centered Rotating Frame
%plot(x_m(:,1), x_m(:,2),'r+', x_m(:,4), x_m(:,5),'b*', x_m(:,7), x_m(:,8),'gs')

%This plot examines the drift in the deviation from stationary in the
%rotating frame
%plot(t(:),x_m(:,1),'b')

%plot an error estimate as time progresses
%plot(t(:),(1)*error(:),'b')

%Plot the length of the radius vector.  If constant then the orbit is a
%circle.  If it oscillates then the orbit is an ellipse and the peiod of
%the sineusiod gives the period of the rotation
%plot(t(:),norm(x_om(:,1:3)),'r')
%(Result:  they look constant which suggests each body is in a circular
%orbit).

%plot the trajectory in inertial frame
%plot(x(:,1), x(:,2),'r', x(:,4), x(:,5),'b', x(:,7), x(:,8),'g')

%the primaries in case (a)
plot(x(:,1), x(:,2),'r', x(:,4), x(:,5),'b')


% vvvvvvvvv Leave uncommented always vvvvvvvvvv
grid on
xlabel('x axis'), ylabel('y axis')
```

References

[1] C. Ocampo, The Lagrange Equilateral Solution for the G3BP, Unpublished Notes

[2] C. Ocampo, The Euler Straight Line Solution for the G3BP, Unpublished Notes