Orbital Dynamics with Maple (sll --- v1.0, February 2012)

Kepler's Laws of Orbital Motion

Orbital theory is one of the great triumphs mathematical astronomy. The first understanding of orbits was published by Johannes Kepler in 1609 and 1619. Kepler was describing the motion of planets, but as the later derivation of the Laws of Planetary Motion from Newtonian gravity demonstrate, they apply to any gravitating system.

Kepler's Three Laws are:

- ▷ Orbits are ellipses. When one mass dominates (like the Sun in relation to the planets) the dominant mass is at one focus of the ellipse. Generically, the focal point of a two-body system is located at the *barycenter* — the center of mass of the two bodies.
- ▷ As the orbit evolves, the area swept out by the radius vector (pointing from the focus of the orbit to the body) per unit time is a constant: $dA/dt = const.$
- ▷ The square of the orbital period P is proportional to the cube of the semi-major axis a of the orbit: $P^2 \propto a^3$. The exact relation may be written in many useful ways. If the masses in a mutual orbit are m_1 and m_2, and noting that $\omega = 2\pi/P$ then Kepler III may be written as

$$G(m_1 + m_2) = \omega^2 a^3 \qquad \rightarrow \qquad P^2 = \frac{4\pi^2}{G}\frac{a^3}{m_1 + m_2}$$

For many applications in astrophysics, the determination of orbits is an essential part of extracting the fundamental behavior of the system from data. Orbital determination is also often important for explaining or determining other physical phenomena (such as light curves and gravitational wave emission).

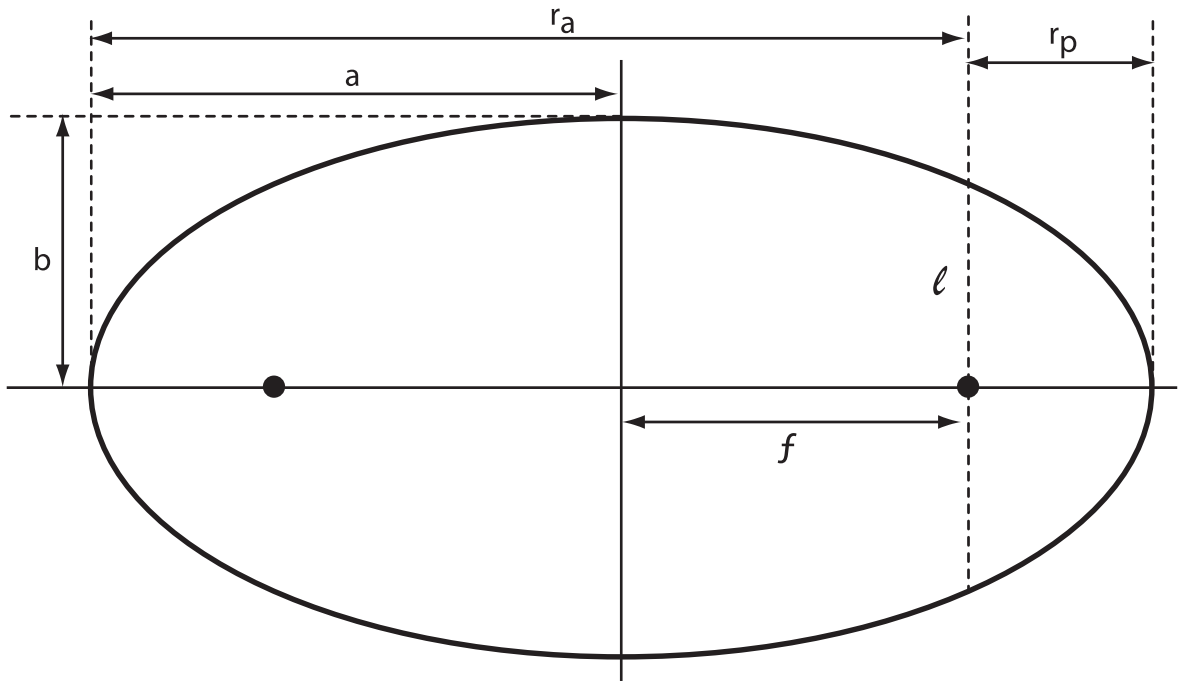
Fundamentally, and orbit is the solution to the equation of motion for bodies moving under the influence of a central $1/r^2$ force — gravity. For a given situation, these equations can be solved analytically or numerically. But under certain reasonable and useful restrictions, the equations of motion can be solved and transformed into a set of fundamental relationships for the geometry of the orbit, capturing the essential physical behavior in closed form solutions that be easily manipulated and used for your own devious purposes.

Rosetta Stone: Orbital Mumbo Jumbo

There are many ways to describe the properties of orbits in terms of measurable parameters, and many of them are interconnected and related.

Ellipses ▶
Since orbits are defined in terms of ellipses, it is useful to recall the basic geometric definitions that describe ellipse geometry.

- **a = semi-major axis.** The *major axis* is the long axis of the ellipse. The semi-major axis is 1/2 this length.
- **b = semi-minor axis.** The *minor axis* is the short axis of the ellipse. The semi-minor axis is 1/2 this length.



- **e = eccentricity.** The eccentricity characterizes the deviation of the ellipse from circular; when $e = 0$ the ellipse is a circle, and when $e = 1$ the ellipse is a parabola. The eccentricity is defined in terms of the semi-major and semi-minor axes as

$$e = \sqrt{1 - (b/a)^2}$$

- **f = focus.** The distance from the geometric center of the ellipse (where the semi-major and semi-minor axes cross) to either focus is

$$f = ae$$

- ℓ **= semi-latus rectum.** The distance from the focus to the ellipse, measured along a line parallel to the semi-minor axis, and has length

$$\ell = b^2/a$$

- r_p **= periapsis.** The periapsis is the distance from the focus to the nearest point of approach of the ellipse; this will be along the semi-major axis and is equal to

$$r_p = a(1 - e)$$

- r_a **= apoapsis.** The apoapsis is the distance from the focus to the farthest point of approach of the ellipse; this will be along the semi-major axis and is equal to

$$r_a = a(1 + e)$$

Basic Geometric Definitions ▶ .....

The game of orbits is always about locating the positions of the masses. For planar orbits (the usual situation we encounter in most astrophysical applications) one can think of the position of the mass m_i in terms of the Cartesian coordinates $\{x_i, y_i\}$, or in terms of some polar coordinates $\{r_i, \theta_i\}$.

The value of the components of these location vectors generically depends on the coordinates used to describe them. The most common coordinates used in orbital theory are called *barycentric coordinates*, with the origin located at the focus between the two bodies.

▷ **The Shape Equation**. The shape equation gives the distance of the orbiting body (*"particle"*) from the focus of the orbit as a function of polar angle θ. It can be expressed in various ways depending on the parameters you find most convenient to describe the orbit.

$$r = \frac{a(1-e^2)}{1+e\cos\theta} \qquad \rightarrow \qquad r = \frac{r_p(1+e)}{1+e\cos\theta} \qquad \rightarrow \qquad r = \frac{r_a(1-e)}{1+e\cos\theta}$$

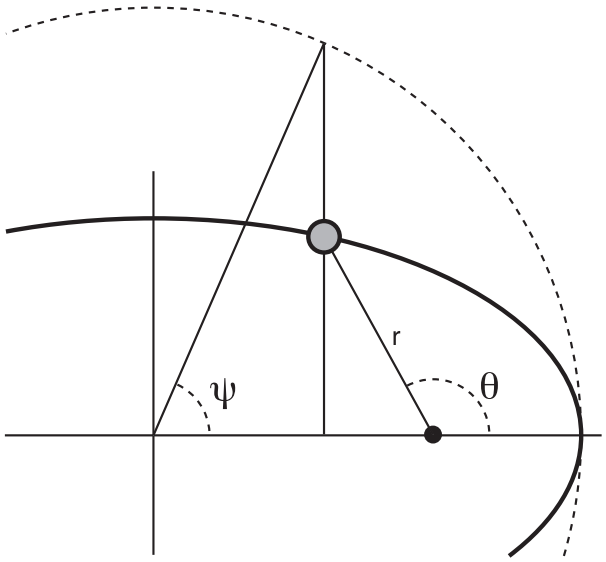
▷ **The Anomaly**. Astronomers refer to the angular position of the body as the *anomaly*. There are three different anomalies of interest.

• θ **= true anomaly.** This is the polar angle θ measured in barycentric coordinates.

• $\mathcal{M}$ **= mean anomaly.** This is the phase of the orbit expressed in terms of the time t since the particle last passed a reference point, generally taken to be $\theta = 0$

$$\mathcal{M} = \frac{2\pi}{P}t$$

Note that for *circular orbits*, $\theta = \mathcal{M}$.



• ψ **= eccentric anomaly.** This is a geometrically defined angle measured from the center of the ellipse to a point on a circumferential circle with radius equal to the semi-major axis of the ellipse. The point on the circle is geometrically located by drawing a perpendicular line from the semi-major axis of the ellipse through the location of the particle. The eccentric anomaly is important for locating the position of the particle as a function of time (as we shall see shortly).

Parametric Plots

An elliptical trajectory is not a function that can be plotted — that is to say the trajectory is not some dependent variable y that can be unambiguously expressed in terms of the value of some independent variable x. This is most obvious when you look at the ellipse plotted in the $x-y$ plane; there are multiple values of y for every value of x!

Mathematically we can still plot continuous trajectories (like the shape equation) by making a *parametric plot*. To define a parametric plot, we need two equations that define the x and y position of the orbital trajectory in terms of a third variable (the *parameter*). For each value of the parameter, there is an unambiguous value for both x and y.

For the purposes of plotting elliptical orbits, we will use the shape equation to build a set of parametric equations. The shape equation gives the distance r from the focus of the orbit to the

particle as a function of the anomaly θ. This can be used to define a set of parametric equations for x and y position of the particle as a function of the parameter θ

$$x = r \cdot \cos\theta \qquad \rightarrow \qquad x = \frac{r_p(1+e)\cdot\cos\theta}{1+e\cos\theta}$$

$$y = r \cdot \sin\theta \qquad \rightarrow \qquad y = \frac{r_p(1+e)\cdot\sin\theta}{1+e\cos\theta}$$

Parametric plots in Maple ▶ .....
Let's implement the plotting of the shape equation. First we define the shape equation as a function of θ so we can just call it

```
[> r := theta -> rp*(1+ecc)/(1 + ecc*cos(theta));
```

We need to define the parameters of the orbit for our problem. For the moment, let's choose generic values of $r_p = 1$ and $e = 0.5$:

```
[> rp := 1;

[> ecc := 0.5;
```

Now define the x and y parametric positions:

```
[> xPos := theta -> r(theta)*cos(theta);

[> yPos := theta -> r(theta)*sin(theta);
```

`Maple` can make parametric plots using the `plot()` command. Usually the arguments for a call to `plot()` are the function you would like plotted, followed by the range of the independent variable: `[> plot(sin(x),x=0..3*Pi);`

To make a parametric plot, the argument of `plot()` is changed to the form (including the square brackets): `[xEquation, yEquation, parameter = lowValue..hiValue]`

So the parametric plot of the shape equation may be implemented by

```
[> plot([xPos(theta),yPos(theta),theta=0..2*Pi],scaling=constrained);
```

The option `scaling=constrained` forces `Maple` to use the same scale on the x and y axes, allowing you to see the ellipse in the proper proportions. You can change the values of the parameters and plot different orbits:

```
[> rp := 0.7;

[> ecc := 0.9;

[> plot([xPos(theta),yPos(theta),theta=0..2*Pi],scaling=constrained);
```

Suppose I wanted to compare the orbit of Mars ($r_p = 1.3818$ AU, $e = 0.9833$) to the orbit of Earth ($r_p = 0.9833$ AU, $e = 0.167$). Since we are going to now have two orbits, each with different

parameters, it is convenient to define a shape equation with more arguments, so we can make specifications for each planet:

```
[> rPlanet := (rp,ecc,theta) -> rp*(1+ecc)/(1 + ecc*cos(theta));
```

First define two shape equations, one for each planet:

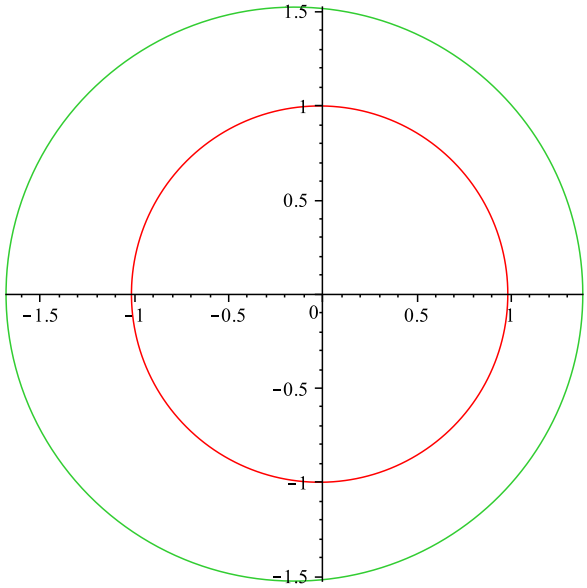
```
[> rEarth := theta -> rPlanet(0.9833,0.0167,theta);
```

```
[> rMars := theta -> rPlanet(1.3818,0.09833,theta);
```

`Maple` can stack two plots by including the two plot arguments in curly braces { }.

```
[> plot({[rEarth(theta)*cos(theta),rEarth(theta)*sin(theta),theta=0..2*Pi],
    [rMars(theta)*cos(theta),rMars(theta)*sin(theta),theta=0..2*Pi]},
    scaling=constrained);
```

Note that the multi-line entry is useful for making your worksheet readable. To create a multi-line entry without evaluating the expression you are entering, type *shift return*.



Planet	**rp (AU)**	**eccentricity**
Mercury	0.30749	0.20563593
Venus	0.71843	0.00677672
Earth	0.98329	0.01671123
Mars	1.38140	0.09339410
Jupiter	4.95113	0.04838624
Saturn	9.02301	0.05386179
Uranus	18.2823	0.04725744
Neptune	29.8116	0.00859048
Pluto	29.6578	0.24882730

Table 1: Orbital elements for the planets in the Solar System, suitable to creating overlapping plots like the one produced above for all the planets.

Orbital Motion

One of the powerful abilities of programs like *Maple* and *Mathematica* is letting you visualize physical systems in ways that you could not otherwise. There are two prominent ways that are most useful are plotting of two-dimensional quantities (either as *surfaces* or *contour plots*), and in animations. Let's look at animations.

General Binary Setup ▶ .....
We label the masses in binary systems such that $m_1 > m_2$; m_1 is called the *primary* and m_2 is called the *secondary*. It is often convenient to define the mass ratio $\beta = m_2/m_1$.

In the usual expression of Kepler III, $m_2 \ll m_1$. If you know the orbital period, you simply use Kepler III to figure out the size of the orbit (the semi-major axis, a). When you have a system comprised of a large mass and a small mass (like the Sun and a planet) the large mass stays (more or less) fixed at the focus of the ellipse that describes the orbit of the small body, so this procedure makes sense.

In the case where the secondary is not significantly less massive than the primary, *both masses are tracing out orbits in space!* These systems are usually called *visual binaries*.

The trajectory traced out by each mass is an ellipse described by the shape equation. Each orbit is traced out with the same period P. Each ellipse has the same eccentricity e, but a semi-major axis that depends on the mass of the body[1]:

$$r_i = \frac{a_i(1-e^2)}{1+e\cos\theta_i} \qquad \rightarrow \qquad r_i = \frac{r_{pi}(1+e)}{1+e\cos\theta_i} \qquad \rightarrow \qquad r_i = \frac{r_{ai}(1-e)}{1+e\cos\theta_i}$$

Since the stars are in a mutual orbit, the line connecting them (the *binary axis*) passes through the focus (the *barycenter*) of the orbit. This means that the angles θ_i are simply related by

$$\theta_2 = \theta_1 + \pi \qquad \rightarrow \qquad \cos\theta_2 = \cos(\theta_1+\pi) = -\cos\theta_1$$

At any given moment, the relative distance between each mass and the barycenter is defined by the mass ratio β. If r_1 is the distance from the barycenter to the primary, then the distance from the barycenter to the secondary is $r_2 = r_1/\beta$. To see this, think of the barycenter as being the location where you would balance a see-saw with the masses, so the torques must equal out:

$$r_1 \cdot m_1 = r_2 \cdot m_2 \qquad \rightarrow \qquad r_2 = r_1\frac{m_1}{m_2} = \frac{r_1}{\beta}$$

The generalized form of Kepler's Third Law is

$$G(m_1+m_2) = \omega^2(a_1+a_2)^3 \qquad \rightarrow \qquad P^2 = \frac{4\pi^2}{G}\frac{(a_1+a_2)^3}{m_1+m_2}$$

Equal Mass, Circular Binary Example ▶ .....
When doing numerical calculations, we often choose mass and length units that are *convenient*. As a general rule, it is often not convenient to work in SI units for orbital dynamics. For instance,

[1] This result is stated without proof; it results from deriving Kepler's Laws from Newtonian gravity.

working in the solar system working in SI units would cause *Maple* to plot numbers like 4×10^{11} m. If we instead work with astronomical units as the basic unit of length, then *Maple* plots numbers in with values like 3.5 — much easier to see/read/interpret! ***Just keep track of the units you work in, otherwise when the time comes to produce a number for a paper/talk/research report, you won't know what you did!***.

Let's consider an equal mass binary. I will choose to work in mass units of solar masses ($M_\odot$) and lengths of astronomical units (AU). The physical parameters I choose for this simulation are

$$m_1 = m_2 = 10M_\odot \qquad P_{orb} = 28.8724 \text{ day} \qquad e = 0$$

If these are main sequence stars, then consulting Appendix G in BOB, then I see that $m = 10M_\odot$ stars are ∼B2 type, which have radii of $R \sim 4R_\odot = 0.0186$ AU (this is a good thing to bear in mind; if the orbit were smaller than this, the stars would be in contact!).

Applying the general form of Kepler III and solving for the size of the orbit, we find

$$a_1 + a_2 = 7.479 \times 10^{10} \text{ m} = 0.5 \text{ AU}$$

Since this is an *equal mass binary*, $\beta = 1$ and at any time $r_1 = r_2$, thus $a_1 = a_2 = 0.25$ AU.

Let's set this up in `Maple`. First we enter the parameter values:

```
[> m1 := 10;

[> m2 := 10;

[> beta := m2/m1;

[> a1 := 0.25;

[> a2 := a1/beta;

[> ecc := 0.0;

[> rp1 := a1*(1 - ecc);

[> rp2 := a2*(1 - ecc);
```

Note that I have set this up in a more complicated way that is necessary for a circular, equal mass binary. I've done this to lend flexibility to my `Maple` code, so it can be used for future, more complicated problems.

Next I define a generic shape equation, then define the barycentric distance for each star:

```
[> rStar := (rp,ecc,theta) -> rp*(1+ecc)/(1 + ecc*cos(theta));

[> r1 := theta -> rStar(rp1,0.0,theta);

[> r2 := theta -> rStar(rp2,0.0,theta + Pi);
```

Animation in `Maple` ▶ .....

If we are going to generate an animation of the orbit, we must have something to *animate*. For this exercise, we'll draw small circles to represent the stars and move them around on the screen[2].

We can't draw the circles to scale, otherwise they would not be visible on your computer monitor, so we will draw exaggerated circles, whose centers follow the orbital trajectories. Because we've decided to measure distances in terms of AU, we have to specify the size that `Maple` should draw our circles in terms of their diameter in AU. The diameter of our orbit was found above to be $a_1 + a_2 = 0.5$ AU, so let's fix the size of the stars to be a small fraction of an AU:

```
[> dStar := a2/20;
```

For `Maple` to draw circles, we give it a set of parametric equations for the circle. Nominally this would be

$$x_{star} = dStar \cdot \cos\phi \qquad\qquad y_{star} = dStar \cdot \sin\phi$$

where $\phi = 0..2\pi$. If I just fed this set of parametric equations to `Maple`, it would draw a circle centered on the origin, but what we want is a circle *centered at the location of the star*! So the parametric equations for a given star are

$$x_{star} = x_i + dStar \cdot \cos\phi \qquad\qquad y_{star} = y_i + dStar \cdot \sin\phi$$

where $\{x_i, y_i\}$ are the Cartesian coordinates of the star, given by the parametric equations

$$x_i = r_i \cdot \cos\theta_i \qquad\qquad y_i = r_i \cdot \sin\theta_i$$

Implementing this in `Maple` can be done in several ways. Adhering to our philosophy of making our code flexible, do the following:

```
[> xStar1 := (phi,theta) -> dStar*cos(phi) + r1(theta)*cos(theta);

[> yStar1 := (phi,theta) -> dStar*sin(phi) + r1(theta)*sin(theta);

[> xStar2 := (phi,theta) -> dStar*cos(phi) + r2(theta)*cos(theta + Pi);

[> yStar2 := (phi,theta) -> dStar*sin(phi) + r2(theta)*sin(theta + Pi);
```

In order to run an animation in `Maple`, we must load the full `plot()` package:

```
[> with(plots);
```

All of our shenanigans above have been geared toward making the `Maple` animation command easier to understand and read.

```
[> animate({[xStar1(phi,theta),yStar1(phi,theta), phi=0..2*Pi],
    [xStar2(phi,theta),yStar2(phi,theta), phi=0..2*Pi]},
    theta=0..2*Pi, frames=75, scaling=constrained);
```

Again, we have used *shift return* to spread the `Maple` entry across several lines and make it more readable. To make the animation run, right-click on the graphic and select `Play` from the `Animation` submenu.

[2]For those of you who are programming aficionados, you will recognize the moving object is a *sprite*.

The Unequal Mass, Circular Binary ▶ .....

We set this up to be very general, so let's use it to demonstrate this by looking at the unequal mass case. Let's consider

$$m_1 = 100 M_\odot \qquad m_2 = 10 M_\odot \qquad P_{orb} = 28.8724 \text{ day} \qquad e = 0$$

In this case the mass ratio is $\beta = 1/10$. Using the general form of Kepler III we find that for this orbital period the corresponding orbital size is

$$a_1 + a_2 = 1.32 \times 10^{11} \text{ m} = 0.8826 \text{ AU}$$

Since $\beta = 0.1$, this implies that $a_2 = a_1/\beta$, which together with the result above (2 equations, 2 unknowns) gives

$$a_1 = 0.8024 \text{ AU} \qquad a_2 = 8.024 \text{ AU}$$

To implement this orbit, we go back through our worksheet, and simply make the appropriate updates, then re-evaluate all the cells:

```
[> m1 := 100;

[> m2 := 10;

[> beta := m2/m1;

[> a1 := 0.8024;

[> a2 := a1/beta;
```