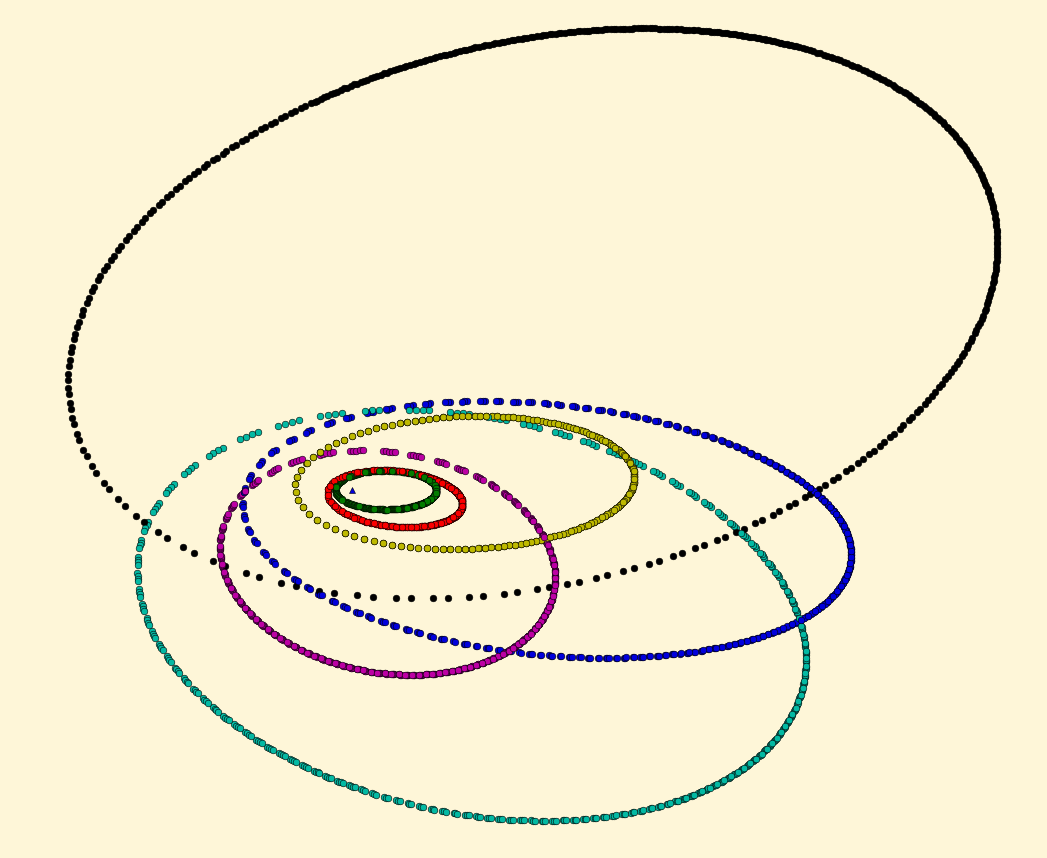
When you want to launch yourself to another planet, you would like to know where that planet is, exactly. This is what I was thinking when I started building my simulation of the system I was in. I already had the data for where all the planets were, their speed and the masses of them and the sun, so all I had to do was calculate the force between the planets and the sun, see how much it moved them, then calculate the force at the new position and voila! Repeat endlessly for orbits of the planets. Here Newton was a great friend in calculating the motion of rocks (unlike Johnson, who preferred the rocks moving towards me at high speeds).

, were the formulas so kindly provided by Sir Isaac, which I used. I used the distance from the sun to find the change to the velocity, which I then used to find the change to the position, where I would find the change to the velocity again. Since I was using a computer, I couldn’t do this perfectly accurately, so I had to move the planets for short steps of time (I moved them for about 30mins before calculating again, and while this may seem like a long time, on a planetary scale I can assure you it isn’t)

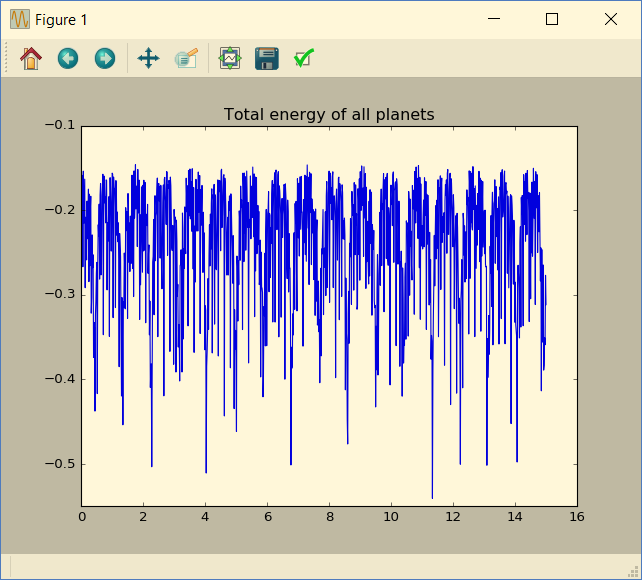
Now, some smart people might recommend that I also account for the gravitational pull on the planets from each other (Yes Lene, I know you have an unceasing need for everything to be exactly how you want it, but doing that would be more coding and I can’t be arsed), but the effect is small compared to that of the sun, so I decided to just not.

Eventually, I ended up with some wonky orbits:

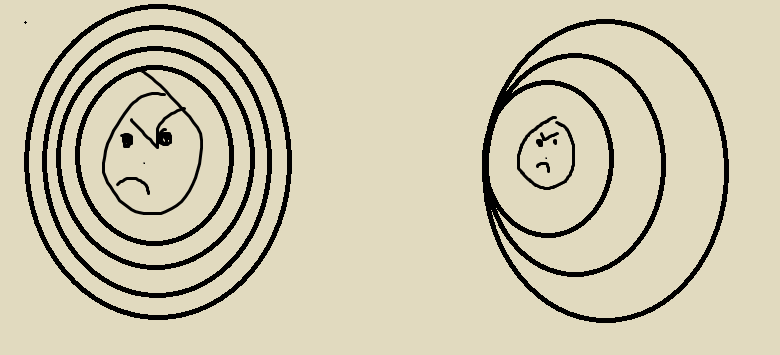


Even if they are a bit wonky, some planets orbit pretty wonkily, and these guys end up in the spot they started, so I trust my simulations. At least your simulations won’t leave you stranded on a bloody empty planet.

Now, I wanted to check if I broke any laws of thermodynamics by energy entering or leaving the system, by checking that the potential energy of the planets and the kinetic energy of the planets remained constant through the orbit, but I somehow couldn’t get the numbers to work out (just like Johnson, you fat fuck). I assume that is because of a fault in my programming and not a fault in the simulation, because the orbits orbit pretty damn nicely, even if it is a bit wonky.

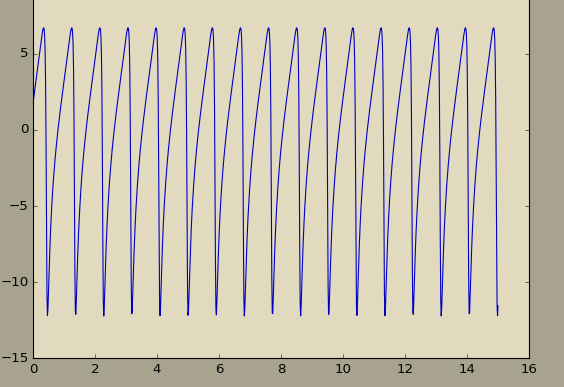
  
This line is supposed to be flat, but it isn’t, so fuck me, am I right?

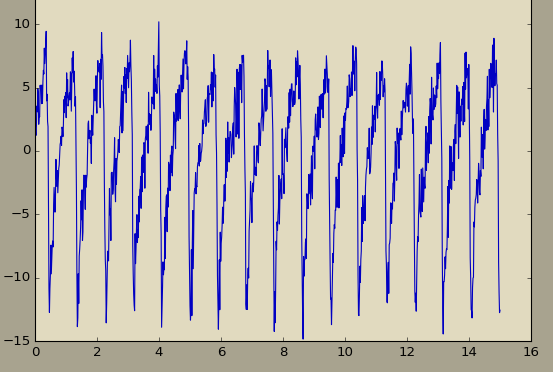
While working on the orbit of the planets, they got me thinking: Can anyone see these planets?  
Does anyone besides my dick crewmembers know that this solar system even has habitable planets? I realised after making a program to check the visibility of the planets in this solar system that of course they do, we were sent up here after all, but done is done, and now I can tell how visible these planets are, or at least the largest ones. Every action has an equal or opposite reaction, and when a star moves a planet around itself, so does the planet move the star. It doesn’t move it close to as much as the planet moves, but depending on the size of the planet, it just might be noticeable. You see (Yes, that is a pun, you will *see* why later), when objects producing waves, such as soundwaves or lightwaves (technically something that is not quite a particle nor a wave, but it behaves enough like a wave to do this), moves they change the frequency of the waves they are producing. We officially call this the doppler effect. This can be noticed when a hovercar moves past you, screaming obscenities at you for standing so close to the fucking road. When the asshole drives towards you, his incoherent garbling is in a higher pitch than normal, as he is moving towards you, making the wavelengths shorter (I made a pretty drawing for you to explain), and when he passes you, you hear a sudden schwoom as the pitch moves from higher than normal to lower than normal, and the asshole is lost in the crowd forever, hopefully to never scream at you again, though that probably doesn’t stop any other drivers from doing it.

  
in the left picture, the asshole is standing still, emitting noise at a constant pace, with the same distance between each wave constant. in the right picture, he is moving his douchebaggery leftwards, still emitting noise at a steady pace, but now each wave is made a tiny bit the left of the last one, making the distance between them shorter on the left and large on the right

The same happens with light waves, and can be noticed in stars, as you will *see* (hehe). When something is moving towards you, the light frequency is shifted a bit, moving the light to one side of the spectrum, like when a comet is hurtling towards us, it looks a bit bluer not because it’s sad that it’s about to kill a bunch of people, but because the light from it has been blueshifted (the opposite being redshift), which is the technical term for it *because apparently we need a technical term for both instead of just calling it the doppler effect.* Knowing what elements the surface of the star is made of (it’s almost always mostly hydrogen), can help us know what wavelengths the star *should* put out, and then compare it to what we *seek* it put out.

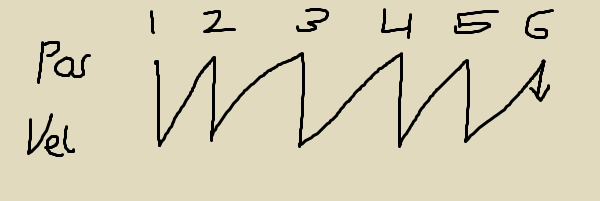
So I used this and found how people would notice the biggest planet orbiting the star

  
I then realised that people would probably look at it through an atmosphere, where the air would fuck with the light coming through, so I added a bit of noise to make it seem more accurate to what other people would see. The noise is just a bunch of random numbers I added to the graph, positive or negative, that have a higher probability of being small, but can still be large. Centred random effects like these are called gaussian distributions, with an average and a standard deviation to show how spread the random numbers are. The average show what the random numbers will be centred around, and the standard deviation shows how far away from the average the random numbers go. As an example, there is a 68% chance of the random numbers being between the average – one deviation and the average + one deviation. The standard deviation is always a number, and so is the average, here they were 0 (average) and one fifth of the highest velocity recorded for the standard deviation, because that seems reasonable.

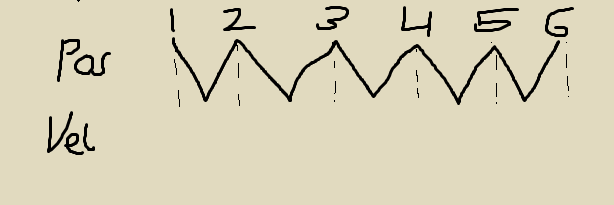
  
What the graph ended up looking like

For the more technical explanations, as I’m sure I will need eventually:

When calculating the orbit, you could just calculate the velocity from the position and then use that new position to calculate a new velocity and so on and so forth, and the direction of calculation would look something like this, where the arrow-thing shows what is being calculated after what:

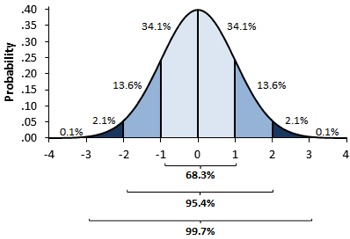


But there is a better method; The leapfrog method! What you can do, for a better accuracy with equally many calculations (ok, one more calculation, but that is a small amount when you use orders of magnitude to say how many calculations you want), is take the initial velocity and calculate it half a timestep forward and then calculate it whole steps afterwards:



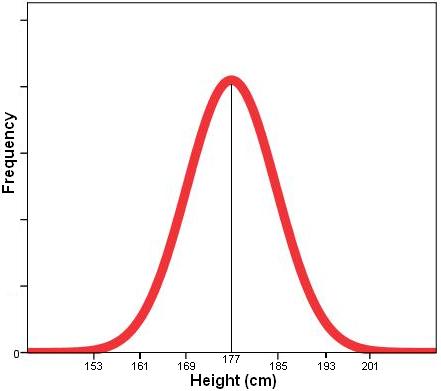
This means that you get twice as accurate for the same amount of computing time! This is what I used, and it works damned well!

The doppler effect can be written as , where is the wavelength we *should* see, is the wavelength we *do* see, is the speed of the object moving relative to the observer and is the speed of the wave (speed of light or speed of sound or whatever, depending on your circumstance). What I did with my data of observed wavelength (in one giant list of wavelengths with accompanying timestamps for when they were observed) was just put the entire list through that formula, isolated for v, and then found out how the observed star moved in terms of away or towards whomever observed it. This doesn’t really account for whether the direction the star-planet duo spins in is oriented towards the observer or not (like when you look at the hands on a watch, if you are looking directly at the clock face, the hands will always have the same distance towards you, while if it is lying face down on a table or something, they can move away from or towards you), but I can at least put a minimum for how fast the planet-star duo is spinning, because while it might move faster than observed, it *certainly* can’t move slower.



This is a normalised distribution curve, showing the type of event happening in terms of how many standard deviations it is away from the average on the horizontal axis and the probability of that happening on the vertical axis.

For example, this is the distribution of height in a population:



As you can see, the average is at 177cm, and the standard deviation is (185-177) = 8cm. This means that 68.2% of the population is between 169cm and 185cm tall.

The curve is described by a formula, which I won’t go into detail on how is made, but I’ll just paste it in here so you can have a look at it:

is the standard distribution and is the average