Maxine Hartnett
4/26/2017

Using QR Factorization to Analyze TRAPPIST-1 Transit Data

To begin with, I took the trappist-1 data and applied a matrix based least squares equation. This involves creating a matrix A, with the first column being the x values, here the epoch data, and the second being all ones. We set that equal to a 1xn matrix of the y data, here the barycentric Julian days. This basically means we can solve for y = ax+b, where y and x are from the data and a and b will be the slope and y-intercept of the best fit line.

However, the A matrix is very ill-conditioned because there is a wide range of values. This means that there will be a large error in my calculation. Using this method, I got 0.662, which is very wrong. This method also amplifies ill-conditioned matrices, so I ruled out the easy solution. A way to reduce errors due to ill-conditioned matrices is QR factorization. I decided to use householder reflectors to generate Q and R. This involves reflecting n columns from the A matrix through an n-1 dimensional plane. This method is more numerically stable than other methods of generating the QR factorization.

Once we have Q and R, we can solve the equation $\hat{R}\overline{x}$ = d, where d is the upper n values of $Q^T b$, and $\hat{R}$ is the upper nxn matrix of R. In this case, since I'm solving for a two-dimensional a and b, the value of n is two. Solving this equation will give us $\overline{x}$, the slope and intercept of the best-fit line.

I ran into quite a few problems with this method. I first wrote the code using a much simpler example, so I could verify by hand that I was getting the right answer. However, when I ran through the data, I had to make modifications to make the method accept any amount of data, and generalize it a lot more. I also kept getting negative numbers for the y-intercept, which would throw my whole line off of the data. However, I eventually solved this problem by realizing that while I was plotting my data points in Mathematica with the epoch as the x value, and bjd as the y value, while my code, the two were reversed. Once I switched them in my code, I got a good fit line.

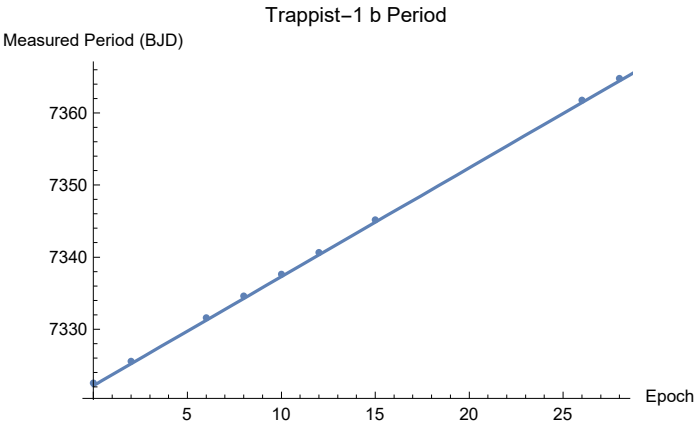
In this line, the real value I was looking at was the slope. The y-intercept was just to align it with the times taken in the data, essentially just an offset from the start date of observations. However, the slope was in units of days/epoch, meaning that this would give the value of the period as generated from this data.

The first value I got was 1.35, where the value determined by astronomers for the period of Trappist-1 b was 1.511. I was looking through my data, maybe seeing if I could eliminate some obvious outliers to get a better value, when I actually realized I was including data from Trappist-1 c as well! Once I deleted these values, I got the slope of 1.51086151. The official period, in days, of Trappist-1 b is 1.51087081±0.00000060. This means I have a relative error of $1.94 * 10^{-5}$. My estimation is off from the official period by 2.53 seconds.

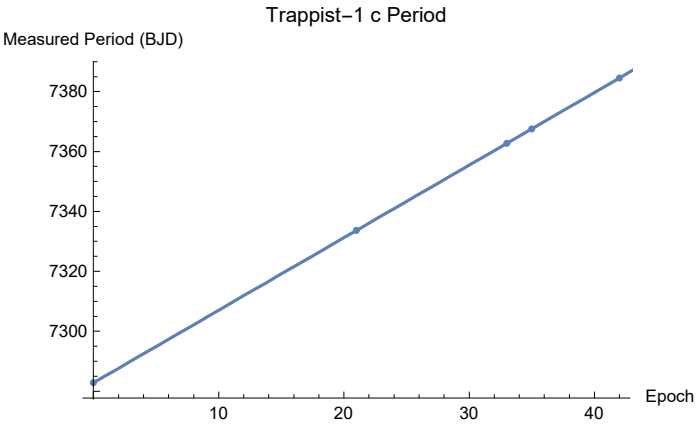
You can see the plot below for a comparison of my data line to the actual data points.

```
epochb = {0, 2, 6, 8, 8, 10, 12, 15, 26, 28};
bjdb = {7322.5161, 7325.5391, 7331.5803, 7334.6032,
   7334.60490, 7337.6249, 7340.6474, 7345.18011, 7361.79960, 7364.82137};

Show[ListPlot[Thread[{epochb, bjdb}]], Plot[1.51089151 x + 7322.1649, {x, 0, 45}],
 PlotLabel → "Trappist-1 b Period", AxesLabel → {"Epoch", "Measured Period (BJD)"}]
```




```
epochc = {0, 21, 33, 35, 42};
bjyc = {7282.8058, 7333.6633, 7362.72623, 7367.5699, 7384.5230};
Show[ListPlot[Thread[{epochc, bjyc}]], Plot[2.42184005 x + 7282.80544, {x, 0, 45}],
 PlotLabel → "Trappist-1 c Period", AxesLabel → {"Epoch", "Measured Period (BJD)"}]
```



Using the Trappist-1 c data, we found another line using the same method. This gives us a slope of 2.42184. The official estimated period for Trappist-1 c is 2.4218233±0.0000017. This gives us a relative error of $6.89563 * 10^{-6}$. This estimate is off by only 1.44 seconds from the official value. Therefore, this method has proven to be effective for finding the periods of planets from transit data.

We can use this data to calculate the semi-major axis, with the mass of Trappist-1 and Newton's version of Kepler's third law, $P^2 = \frac{GM}{4\pi^2} * a^3$. With this calculation, I got 0.011094 AU as the semi-major axis of Trappist-1 b. The actual predicted value is 0.01111 AU. This means the relative error is 0.00144. I got 0.01519609 AU as my prediction for the semi-major axis of Trappist-1 c. The official

value is 0.01521 AU, making the relative error $9 * 10^{-4}$.

References:
Confirmed planet data from the exoplanet archive, http://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=planets

Data about the star from:
trappist.one/#system

Planet	Instrument	Epoch	Mid-transit timing (BJD_{TDB}-2,450,000)
TRAPPIST-1b	TRAPPIST	0	$7322.5161^{+0.0013}_{-0.0010}$
	TRAPPIST	2	$7325.5391^{+0.0035}_{-0.0013}$
	TRAPPIST	6	7331.5803±0.0013
	TRAPPIST	8	7334.6038±0.0012
	VLT/HAWK-I	8	7334.60490±0.00020
	TRAPPIST	10	7337.6249±0.0010
	TRAPPIST	12	$7340.6474^{+0.0010}_{-0.0022}$
	HCT/HFOSC	15	7345.18011±0.00089
	UKIRT/WFCAM	26	7361.79960±0.00030
	UKIRT/WFCAM	28	7364.82137±0.00056
TRAPPIST-1c	TRAPPIST	0	7282.8058±0.0010
	TRAPPIST	21	7333.6633±0.0010
	UKIRT/WFCAM	33	7362.72623±0.00040
	TRAPPIST	35	7367.5699±0.0012
	TRAPPIST	42	7384.5230±0.0011
TRAPPIST-1d	TRAPPIST	0	7294.7736±0.0014
	TRAPPIST	?	7367. 5818±0.0015