# Mars Orbit Module

Abhishek Ranjan MTech (Res) Sr No. - 18390

## 1. Introduction

In the early centuries, astronomers and space enthusiasts were keen on knowing the motion of different planets and celestial bodies, but due to lack of the proper equipment and data, they were unable to devise the trajectory. The only thing they knew was that The Sun, the s, and planets move in some circular path of radius, depending on their latitude and the observer's latitude.

Scientists can observe the planet's path based on some distant stars' collections like zodiacs. The plane on which the Earth and the sun lie is the Ecliptic. **Tycho Brahe,** one of the prominent observers in the early centuries, created several instruments to measure the longitude and latitude of planets. He observed Mars from 1580-1601. After him, **Kepler** obtained the data obtained by Brahe and spent many years in analysis before stating his famous **three laws of planetary motion.** In this analysis, we are trying to predict the orbit of Mars based on Brahe's data up to a discrepancy of 4 minutes.

## 2. Code Architecture

The Dataset contains observations from 1580 to 1604 with an interval of 2 years consisting of 15 features.

- Calculate the "Mars Heliocentric longitude" values in degree form for each entry using the formula given below:

Longitude = ZodiacIndex * 30 + Degree + Minute/60 + second/3600

- calculate the difference in the number of days between two entries provided in the Dataset.
- Create a matrix named oppositions that store the "Mars Heliocentric longitude" array and "time difference array".

*This opposition array will help in finding the positions of Mars in its orbit*

- We can use oppositions data to get spokes with respect to the sun-Aries axis and with respect to equant 0.

- Define a global function "getIntersectionPoint(h, k, theta, r, c)" with *h* and *k* as the coordinate of intercepting points; *theta* makes an angle with the x-axis, *r* is the radius of the orbit, and *c* is the angle at which center of the orbit is placed with respect to the sun.
- For Q1, Define a function "MarsEquantModel()" with following parameters:

  c: - orbit of circle with Centre specified at angle c(degrees)

  r: - radius of orbit (in units of sun-center distance)

  e1, e2 are coordinates of equant where e1 is the distance from the sun and e2 is the angle in degree with respect to equant 0.

  The reference' equant 0' is given by angle with respect to Aries.

  S represents the angular velocity of Mars around the equant.

  It outputs an error list of all the 12 oppositions and the Maximum error.

- For Q2, we fix the value of **r** and **s** and do a grid search over parameters c, e1, e2, and z to minimize the angular error for a particular r and s value. So, a function is defined as "bestOrbitInnerParams(r, s, oppositions)". It outputs parameters like c, e1, e2, z, error of 12 oppositions, and least max error.
- For Q3, we fix the value of **r** by doing a discretized search in the neighborhood of 360 degrees over 687 days. I have used function bestOrbitInnerParams() for each value of s. So, a function is defined as "bestS(r, oppositions)". It outputs optimized value for s, error of 12 oppositions, and least max error.
- For Q4, we fix the value of **s** do a discretized search about the average distance of black dots *(The intersections of the resp dotted (lines from the equant) and solid lines (lines from the sun) yield the positions of Mars)*. So, a function is defined as "bestR(s, oppositions)". It internally calls function bestOrbitInnerParams() for each value of r. This function gives outputs as optimized value for r, error of 12 oppositions, and least max error.

- For Q5, a function is defined as "bestMarsOrbitParams()," in which I did the grid search in the neighborhood of best initial guess for r and s. It internally calls function bestOrbitInnerParams() for each value of r and s. It outputs the best parameter value for c, z, r, e1, e2, r, s, error list for 12 oppositions, and best Max Error.

## 3. Derivations and Definitions

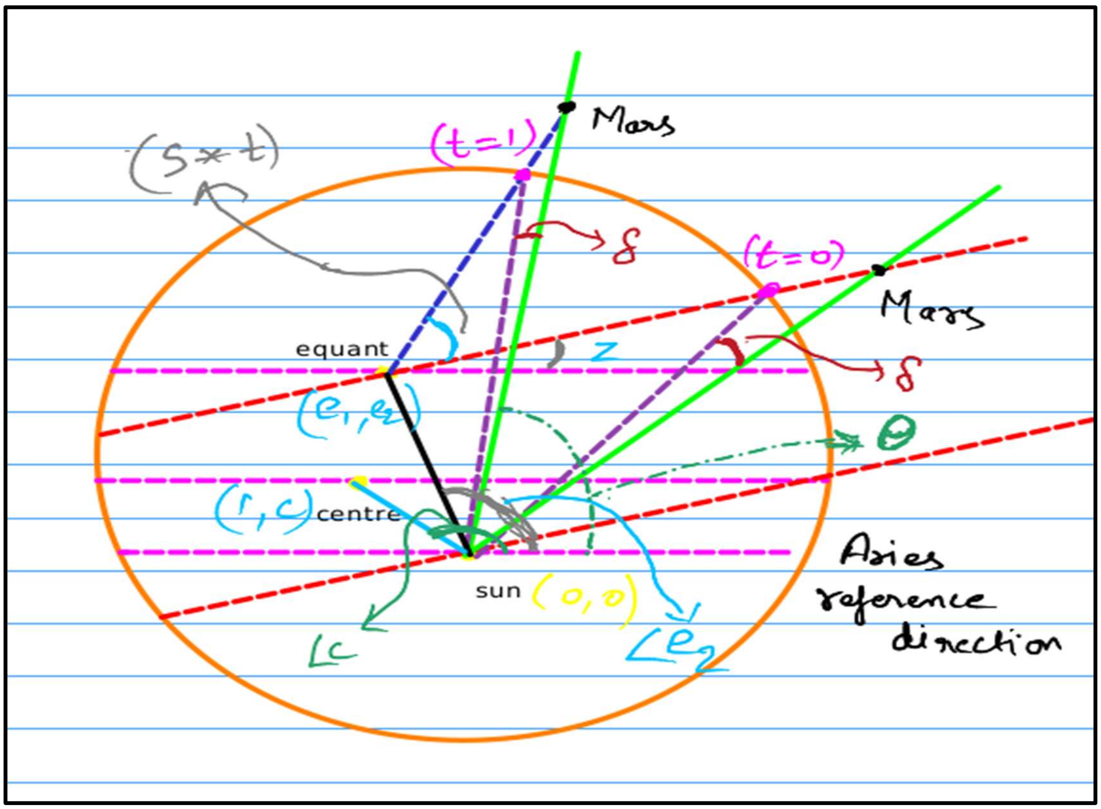


Fig 1: a sketch of mars orbit

Here in the diagram, M represents the position of Mars. Delta gives the angle discrepancy between the observed position of mars from Earth with respect to the sun and the model position with respect to equant.

Now, using the diagram showed above, we can say that :-

Points of line :→

- Intercepts Point $(h, k)$ of line with the circle as we are given radius '$r$' and coordinates of Centre $(1, c)$ in polar form. Here $c$ is angle of Centre from the sun & 1 is distance.

$\boxed{\text{• } h, k \text{ are Cartesian coordinate of } (e_1, e_2)}$

- $(x_1, y_1)$ are Centre of circle in cartesian form

$$\therefore \quad x_1 = \cos c, \quad y_1 = \sin c$$

- line from equant $(e_1, e_2)$ to $t=1$, Pos. of Mars makes angle $\phi$ with the x-axis.

$$\phi = z + (s * t) \longrightarrow \text{this is called as theta in code.}$$

Each Point on line is described as $(h + l\cos\phi,\ k + l\sin\phi)$

Here, $l$ is some running variable. as we vary '$l$', we get different points along the line.

we know, that centre is not at origin. Hence Equation of circle is :

$$(x - x_1)^2 + (y - y_1)^2 = r^2$$

where, $x = (h + l\cos\phi)$, $y = (k + l\sin\phi)$, $x_1 = \cos c$, $y_1 = \sin c$.

$$\Rightarrow ((h + l\cos\phi) - \cos c)^2 + (k + l\sin\phi - \sin c)^2 = r^2$$

on solving this equation, we get,

$$l^2 + 2\underbrace{(h\cos\phi + k\sin\phi - \cos c \cdot \cos\phi - \sin c \cdot \sin\phi)}_{\text{let it be } b} \cdot l + \underbrace{k^2 + h^2 + 1 - 2h\cos c - 2k\sin c - r^2}_{\text{let, it be } c} = 0$$

then $\quad l^2 + 2bl + c = 0$

$$\text{or} \quad l = \frac{-b \pm \sqrt{b^2 - 4c}}{2}$$

$\boxed{\text{Take Positive root of } l \text{ for finding intersection Point :- } x, y = ((h + l\cos\phi), (k + l\sin\phi))}$

we can do, $\tan^{-1}(y/x)$ from equant to find the angle of observation and if we substract it from the observed angle w.r.t Sun we will get the discrepancy of angle.

## 4. Summaries of outputs

Angle of projection for Mars at the best value of parameters: - c=49.0, r = 8.5999, e1 =1.60000000001, e2 = 93.199999999, z = 55.800000001, s = 0.524.

```
projection of mars -in degree for spoke  0 :- 66.41568548485075

projection of mars -in degree for spoke  1 :- 106.91987148757997

projection of mars -in degree for spoke  2 :- 141.61699958798587

projection of mars -in degree for spoke  3 :- 175.71365621014655

projection of mars -in degree for spoke  4 :- 214.40803212542343

projection of mars -in degree for spoke  5 :- 266.6763801958711

projection of mars -in degree for spoke  6 :- 342.28732724662217

projection of mars -in degree for spoke  7 :- 47.5911838652876

projection of mars -in degree for spoke  8 :- 92.51508723879607

projection of mars -in degree for spoke  9 :- 128.68197103916125

projection of mars -in degree for spoke  10 :- 162.47209374138987

projection of mars -in degree for spoke  11 :- 198.67794720835812

12 spokes errors-: [0.060703404038136455, 0.005128512420043307, 0.014221810208113084, 0.0030104565201156674, 0.024
69879209010628, 0.040286470795535934, 0.020660579955517733, 0.06340608750981858, 0.04842057212940176, 0.0486377058
2792881, 0.02209374138988096, 0.0585027639136797]
-----------------------------------------------------------
maximum error:- 0.06340608750981858
```

Fig 2:- Best Max Error result

- Experimentation

| Best MaxError | optimumCVal | optimume1Val | optimume2Val | optimumZVal | optimumRVal | optimumSVal |
|---|---|---|---|---|---|---|
| 0.2672 | 156 | 1.6 | 93 | 56 | 8.1 | 0.524017467 |
| 0.1327 | 156 | 1.48 | 93 | 55.7 | 8 | 0.524017467 |
| 0.0981 | 156 | 1.56 | 93.3 | 55.7 | 8.4 | 0.524017467 |
| 0.0757 | 149.1 | 1.56 | 93 | 55.8 | 8.4 | 0.524017467 |
| 0.0677 | 149 | 1.6 | 93.19999999 | 55.8 | 8.6 | 0.524017467 |
| **0.063406088** | **149.0000001** | **1.6** | **93.19999999** | **55.8** | **8.6** | **0.524017467** |

Table 1: Various experimentation results

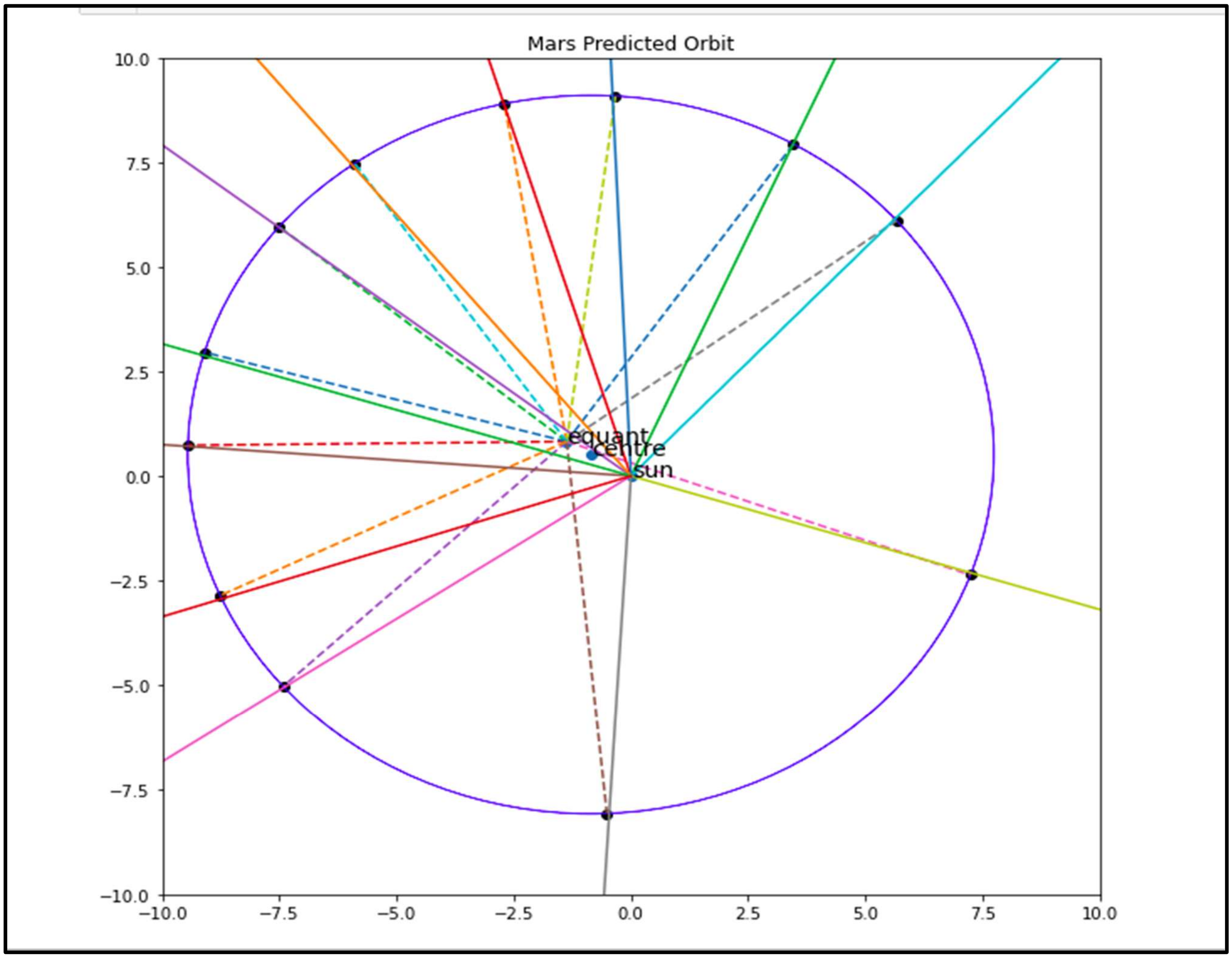


Fig 3:- Mars positions on-orbit when the error is least

## 5. Conclusion

This module finds the max error angle between spokes from the Sun-Aries reference axis and spokes from the equant; the error must be less than 4 minutes or 0.06 degrees. I found this error to be 0.06340608750981858 degrees which are 3.804 minutes.

I learned how to use the Dataset to create new fields and do visualization of data. This assignment also helps me to understand the essential libraries. It helped me to explore the Grid search method and approximation of values to optimize hyperparameters.

For next year's students, specify some particular range of values for parameters to search. It will help them to save some computing time. And Instructor might increase the complexity of the assignment by asking them to use some optimization method instead of using the brute force method to optimize the variables.