Name: Aditya Gupta
SR Number: 22205 UG BTech 3rd year

Q1	Q2	T
5	~~3~~ 5	~~8~~ 10

EO 259 Data Analytics
Quiz: 05/10/2024, 4:00 – 5:00 pm
Module: Mars Orbit

Instructions:

1. Please write your name and SR number on each page of the answer script now.
2. The quiz is in lieu of Assignment 2. There are two questions. Each carries five marks.
3. Each question is on a separate sheet of paper. Restrict your response to just that paper (front and back). You can use extra paper if you wish, don't attach them.
4. Please do not approach the TAs or the proctors for any clarifications. If there is something ambiguous, please state your assumptions and proceed to provide your response.
5. The duration of the test is 1 hr. No extra time will be given.

Question 1: Let $lo_1, lo_2, lo_3, \dots, lo_k$ be the geocentric longitudes of Mars observed at times of k distinct oppositions that occur at times $t_1, t_2, t_3, \dots, t_k$, where the longitudes are in radians and times are in minutes, from some base year far enough back. Assume an equant at position (x, y) and the sun at (0,0). Assume Mars' orbit is a circle centred at the equant.

a) What other unknown parameters do you need to fully define Mars' orbit?
b) Write a pseudocode to derive the oppositions discrepancy at each of the k oppositions assuming the above orbit.

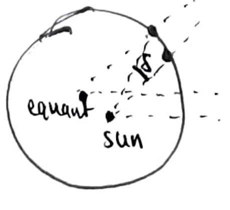
You can assume you have the following readymade functions; write any others you need:

(i) $Line(p_1, p_2)$: Returns a line passing through two given points p_1, p_2
(ii) $Point(l_1, l_2)$: Returns the intersection point of two given lines l_1, l_2
(iii) $Angle(l_1, l_2, p)$: Returns the angle between lines l_1, l_2, both passing through p
(iv) $Circle(l, p, r)$: Returns the points of intersection of line l with a circle with centre p and radius r.

(a) To define mars' orbit, we need the radius of the orbit (r) and the angular velocity of mars around the equant (s). Additionally, to find mars' position at a given time, we would need to know its longitude (z) at time t_1 wrt the equant (assuming longitude taken from Sun-Aries line)

-2

(b) The opposition discrepancy is defined as the angle between the lines connecting sun to mars' observed & predicted position on the circle centred at the centre.



sun-aries reference

Start with the longitudes observed from sun

long_sun = [lo_1 , lo_2 , ... lo_K]

initial long. angular speed

Assuming we have z & s fixed, get longitudes from equant.

long_equant = [z, z + $(t_2-t_1)s$, z + $(t_3-t_1)s$, ..., z + $(t_K-t_1)s$] % 2π

since its an angle ≤ 2π

We have sun at (0,0), equant & centre at (x,y)

Define a function line_fromangle

def line_fromangle (p_1, a_1):
    returns line passing through p_1 with angle a_1 from x-axis

Now we need to calculate angle of equant's predicted mars from sun.

for i in range(K):

l_1 = line_fromangle ((x,y), long_equant [i])

p_1 = circle (l_1, (x,y), r)   # assuming we fixed r.

l_2 = line (p_1, (0,0))   # line from sun to equant pred

l_3 = line_fromangle ((0,0), long_sun [i])   # observed line from sun

δ_i = angle (l_2, l_3, (0,0))   # discrepancy

Thus we get discrepancy δ_i for all K times.

2.5 + 0.5 = 3

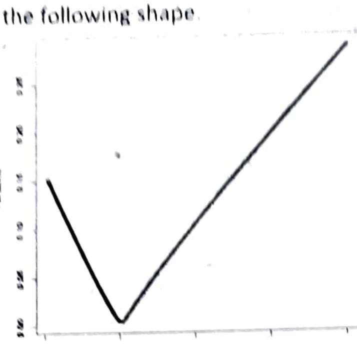
Name: Aditya Gupta

SR Number: 22205, UG BTech 3rd year

Question 2: You are trying to fit an ellipse with focus at (0,0) to a few given points on a plane. The equation of the ellipse is given by the following in polar coordinates

$$r = l/(1 + ecc\cos(\theta))$$

where there are two parameters, l and ecc. Suppose l can vary from 10 to 20 and ecc can vary from 0.0 to 0.4. For any given value of l in the above range, the dependence of the root-mean-squared-error of the fit on ecc is given by the following shape.

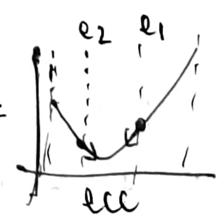


5

Describe in pseudocode an algorithm to find a close to optimum value for l and ecc, with an eye on efficiency. You can assume you have access to a function $RMSE(l, ecc)$ that, given l and ecc will give you the RMSE value of the fit.

From the graph, it appears that there is only one optimum value of ecc for a given l. Also, the graph has a minima at the optimum point, & monotonic growth on ~~the~~ both sides of minima.

Thus, to find the best ecc for a given l, we need to perform gradient descent. However, since we only have one variable here, we can simply use a binary search algorithm.




```
def best_ecc (l, tolerance):
    start = 0.0
    end = 0.4
    while |start - end| > tolerance:
        e = (start + end)/2
        if RMSE(l, e) > RMSE(l, e×1.00001):   # RMSE of e & point slightl
                                               ahead of e
            start = e
        else
            end = e
    return (start + end)/2      # return final best ecc for l
```

best_ecc will return ~~the~~ ~~best~~ an ecc very close to the actual minima (within small tolerance). Since it is a binary search, it will converge fast.

#Now that we have best_ecc for an l, we calculate it for all l's in the given range.

best = []
for l in range (10, 20, 0.01):   # 10 to 20 with step size 0.01
Total 1000 values
    ecc = best_ecc (l, tolerance)
    best.append ~~(ecc)~~ (RMSE(l, ecc)}, l, ecc)

best ecc will give ~~best~~ RMSE for any l ?

The above loop gives us best RMSE for all 1000 values of l. For best fit, we need RMSE to be the lowest. So we take top 1~~0~~% RMSE and do a finer search on them.

sort (best)   # sort in ascending order based on RMSE

new = best [0 : 10]   # Here new gets the corresponding l values for the top 1% RMSE

final = []
for l in new:
    for l_1 in range (l-0.005, l+0.005, 0.0001):   # 100 values
        ecc = best_ecc (l_1, tolerance)
        final.append (RMSE(l_1, ecc)}, l, ecc)

Now final has the best possible values of RMSE.
select best ~~R~~ RMSE from it.

sort (final)   # sort in ascending order based on RMSE

best_fit = final [0]   # lowest RMSE pair

Thus we get optimum value of l, ecc

Aditya Gupta, SR No: 22205, UG BTech 3rd year.

Note that we can make this more accurate by increasing the ~~f~~ ranges (ie 10000 values instead of 1000) for initial search.

For the final search, we can also choose the ranges to be a percentage interval (ie 99% l to 101% l) & shrink the range based on binary search upto some tolerance.