Assignment 2: part 1

The aim of the task was to produce coordinates in the RA (Right Ascension) system of 17 satellites in orbit at an instantaneous time. However, with respect to the data obtained only the orbital coordinates could be calculated which thereafter had to be converted to the RA system. The orbital system is the coordinate system used by the satellites.

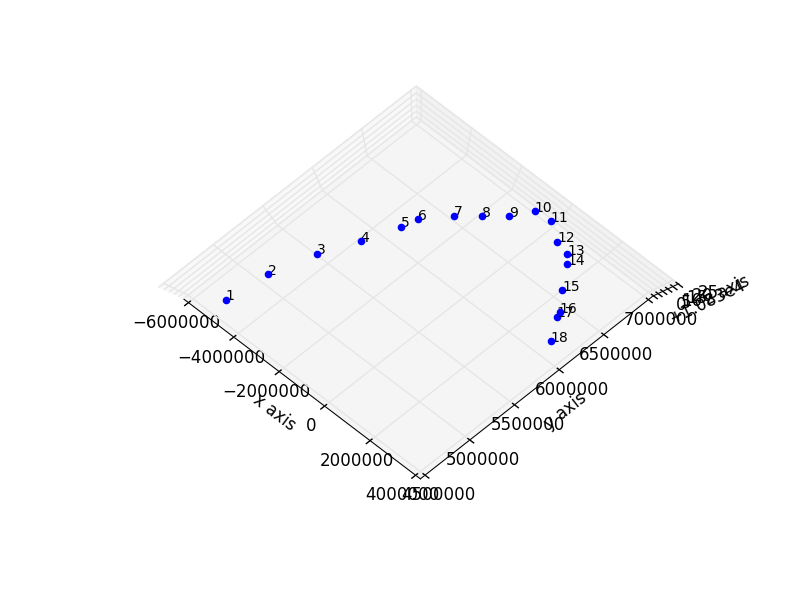
The implementation of acquiring data was a simple process. By accessing the NASA spaceflight data page, information regarding the position and time of the satellite was obtained to compute results. The data on the page was then copied and made into a text file for processing. To initiate the calculations the data first had to be sorted as not everything in the text file was relevant. Using python (a programming language), the data could all be read and split based on characters such as tab spaces and next line symbols. The reason these specific characters were chosen is to avoid splitting lines of data even though there had been space bar symbols between them. The individual line elements (strings) was then appended to a list which made it convenient to perform operations on each element. Once the full lists was made all that needed to be done was then to filter out those elements which were of relevance.

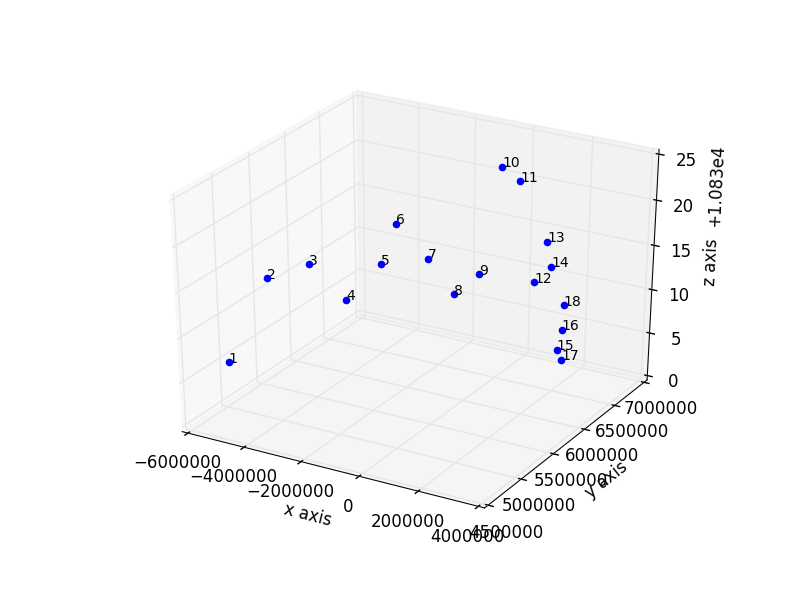
Based on the position of the element “TWO LINE MEAN SET” in the lists an “if” statement was set up to grab elements line 1 and line 2 of which each were appended into two lists. The lists generated were then inspected and sliced accordingly. Seeing as each line was one string, slicing with the associated positional values could be done, most of these slice values were obtained from the celestrak website which explained the meaning of the numbers. Furthermore the individual data was divided and then the calculations were performed to compute the Eccentric Anomaly and oriented true anomaly which were then used to compute the orbital coordinates of each satellite. Following the procedures to convert from orbital to the RA system a simple rotation was computed about the z axis using argument of perigee as the angle, then about the x axis using the Inclination as the angle and then again about the z axis finally using the Right Ascension as the angle. Please note all angles were converted to radians and revolutions per day to radians per second.

A text file “DATA2” was also created to contain all the information about the satellite and can be seen below. In addition a module matplotlib was used to produce 3D scatter plots of the calculated coordinate data in python (please use spyder and enable graphic under preferences to get a full 360 degree plot).

Graphs:

From the top we get a semi ellipse :



A more scaled view from the side:

Coordinates in RA system:

|  |  |  |  |
| --- | --- | --- | --- |
| Satellite number | X | Y | Z |
| 1 | -5031609.79498 | 4679832.94993 | 10837.08328 |
| 2 | -4621318.06008 | 5085380.43587 | 10844.69719 |
| 3 | -4147988.76344 | 5478409.13021 | 10844.659 |
| 4 | -3673871.33672 | 5807040.60977 | 10839.38627 |
| 5 | -3139791.15219 | 6112405.56622 | 10842.29141 |
| 6 | -2897702.88362 | 6231182.21477 | 10846.43103 |
| 7 | -2328691.71581 | 6465380.68441 | 10841.78629 |
| 8 | -1778173.94276 | 6637854.30043 | 10837.24009 |
| 9 | -1177234.31541 | 6770418.04369 | 10839.29269 |
| 10 | -605001.62166 | 6845294.80029 | 10851.44954 |
| 11 | -28772.56259 | 6872005.85182 | 10850.10269 |
| 12 | 586168.48979 | 6846935.2511 | 10839.27072 |
| 13 | 1158425.88138 | 6773480.3356 | 10844.6522 |
| 14 | 1422850.03367 | 6722733.27213 | 10842.21576 |
| 15 | 2017957.26714 | 6568563.11074 | 10833.90151 |
| 16 | 2561425.04276 | 6376089.1275 | 10837.62413 |
| 17 | 2632566.76901 | 6346895.43046 | 10834.38294 |
| 18 | 3155313.05351 | 6103724.2478 | 10842.26161 |

The orbital parameters can found in the document labelled orbital parameters (question 1).

Code script:

import math

import numpy as np

from mpl\_toolkits.mplot3d import Axes3D

import matplotlib.pyplot as plt

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

#print ax.azim

#ax.view\_init(azim=60)

f = open("DATA.txt","r")

data = f.read()

spl = data.splitlines()

list1 =[]

list2 =[]

X=[]

Y=[]

Z=[]

f.close()

f2 = open("DATA2.txt","w")

for i in range(len(spl)):

if spl[i] == 'TWO LINE MEAN ELEMENT SET':

list1.append(spl[i+3])

list2.append(spl[i+4])

f2.write('sat'+'\t'+'EpochYear'+'\t'+'EDay'+'\t'+'EHour'+'\t'+'EMin'+'\t'+'ESec'+'\t'+'Inclination'+'\t'+'RightAscension'+'\t'+'Eccentricity'+'\t'+'Perigee'+' \t'+'MeanAnomoly'+'\t'+'MeanMotion'+'\t'+'RevsNo'+' '+'\t'+'\t'+'RadPs'+'\t'+'\t'+'\t'+'TrueAnomoly'+'\n'+'\n')

for i in range(len(list1)):

position1 = list1.pop(0)

EpochYear = float(position1[19:20])

EpochDay = float(position1[21:32])

EDay = int(EpochDay)

EHour = int((EpochDay - EDay)\*24)

EMin = int((((EpochDay - EDay)\*24 -EHour))\*60)

ESec = int((((((EpochDay - EDay)\*24 -EHour))\*60) - EMin)\*60)

position2 = list2.pop(0)

Inclination = math.radians(float(position2[9:16]))

RightAscension = math.radians(float(position2[17:26]))

Eccentricity = float(position2[27:33])\*(10\*\*-7)

Perigee = math.radians(float(position2[34:42]))

MeanAnomoly = float(position2[43:51])

MeanMotion = float(position2[52:63])

RevsNo = float(position2[64:69])

pie = math.pi

RadPs = (MeanMotion\*(2\*pie))/86400

M = math.radians(MeanAnomoly)

E = M

def Efunction(X):

global E

E = M +e\*math.sin(X)

return E;

e = Eccentricity

iteration = 5

while iteration > 0:

iteration -=1

EccentricA = Efunction(E)

#print EccentricA

Ft = (((1-e\*\*2)\*\*0.5)\*math.sin(EccentricA))

Fb = (math.cos(EccentricA)-e)

TrueAnomoly = math.atan(Ft/Fb)

#print TrueAnomoly

if Ft >0 and Fb<0:

TrueAnomoly = TrueAnomoly + math.pi

elif Ft<0 and Fb<0:

TrueAnomoly = TrueAnomoly + math.pi

elif Ft <0 and Fb>0:

TrueAnomoly = TrueAnomoly+2\*(math.pi)

else:

TrueAnomoly= TrueAnomoly

#print TrueAnomoly

a = float(6871000)

r = (a\*(1-(e\*\*2))/(1+(e\*(math.cos(TrueAnomoly)))))

x = r\*(math.cos(TrueAnomoly))

y = r\*(math.sin(TrueAnomoly))

z = 0

#print(x,y,z)

xRp = (x\*(math.cos(-1\*Perigee))) + (y\*(math.sin(-1\*Perigee))) #rotation about z-axis , angle of perigee

yRp = -1\*x\*(math.sin(-1\*Perigee)) + (y\*math.cos(-1\*Perigee))

zRp = z

xRi = xRp

yRi = zRp\*(math.sin(-1\*Inclination)) + (yRp\*math.cos(-1\*Inclination)) #rotation about x-axis, angle of inclination

zRi = zRp\*(math.cos(-1\*Inclination)) - (yRp\*math.sin(-1\*Inclination))

xRa = (xRi\*(math.cos(-1\*RightAscension))) + (yRi\*(math.sin(-1\*RightAscension))) #rotation about z-axis, angle of rightascension

yRa = -1\*xRi\*(math.sin(-1\*RightAscension)) + (yRi\*math.cos(-1\*RightAscension))

zRa = zRi

f2.write('sat'+str([i+1])+'\t'+str(EpochYear)+'\t'+'\t'+str(EDay)+'\t'+str(EHour)+'\t'+str(EMin)+'\t'+str(ESec)+'\t'+str(Inclination)+'\t'+str(RightAscension)+'\t'+str(Eccentricity)+'\t'+str(Perigee)+' \t'+str(MeanAnomoly)+' '+'\t'+str(MeanMotion)+'\t'+str(RevsNo)+'\t'+'\t'+str(RadPs)+' '+'\t'+str(TrueAnomoly)+'\n')

X.append(xRa)

Y.append(yRa)

Z.append(zRa)

#print (round(xRa,5),'\t',round(yRa,5),'\t',round(zRa,5))

for i in range(len(X)): #plot each point + it's index as text above

ax.scatter(X[i],Y[i],Z[i], color='b', marker = 'o')

ax.text(X[i],Y[i],Z[i], '%s' % (str(i+1)), size=10, zorder=1, color='k')

#f2.write("Satellite coordinates in RA system"+'\n'+'\n'+'\t'+"Xra"+'\t'+"Yra"+'\t'+"Zra"+'\n')

ax.set\_xlabel('x axis')

ax.set\_ylabel('y axis')

ax.set\_zlabel('z axis')

plt.show()