Error Comparison for Propagated Starlink Satellites

David Roman Garcia *UROP, December 2022*

The SGP4 model is an analytical tool for orbit prediction from two-line element sets (TLEs). Normally, it is assumed that the error arising from such calculations is no more than satellites deviating 1-3 km per day from their predicted orbits [1]. Particularly, the Python implementation is expected to deviate no more than 0.1 mm from the standard SGP4 standard source code on C++ [2]. Nevertheless, there does not exist many sources to validate these claims [3]. Moreover, TLEs do not provide any kind of accuracy information. In this document, we propagate low-earth orbit satellites (LEOs) from the Starlink constellation through SGP4 and compare our results to those found in Privateer. By analyzing the relative errors in velocity and position, we find that errors do not increase drastically around the 30 second time period needed for our calculations. Nevertheless, they do appear to grow significantly over longer time intervals and we are left with important questions without having access to the proprietary tools from Privateer to collect and process space objects.

As a first case, we consider the object Starlink-4633. Its TLE information is freely available on Celestrak and it roughly covers the same time period than the information available at Privateer. Particularly, the TLE used for Starlink-4633 from Celestrak is dated on 2022-11-03T04:00:01.000. Similarly, the data for the same object from Privateer ranges from from 2022-11-01T12:00:00.000 to 2022-11-06T12:00:00.000. This allows us to compare the position and velocity outputs from SGP4 and the proprietary tools from Privateer, resulting in useful information that mostly agree with our results for short time intervals smaller than a few minutes. First, we calculate the magnitude of the relative error between both methods. We do this for the position vector magnitude and each of its components:

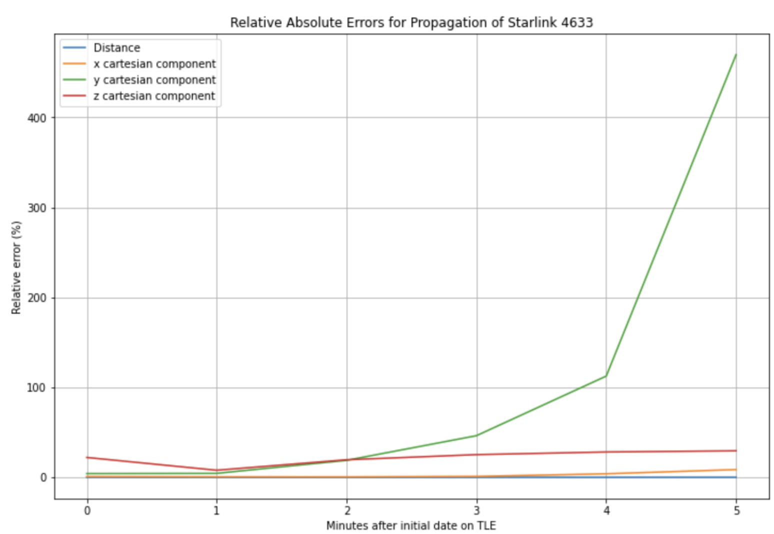

Figure 1: Magnitude of relative error for position magnitude and components of Starlink-4633

Similarly, we also compare the velocities from both sources obtaining similar results. The fact that the significant surge in velocity error for the x component does not produce an error increase for any of the position variables in the next iteration indicates that privateer may use significantly different methods for propagating or obtaining data from satellites.

As a second case, we consider the object Starlink-1007. Its TLE information is freely available

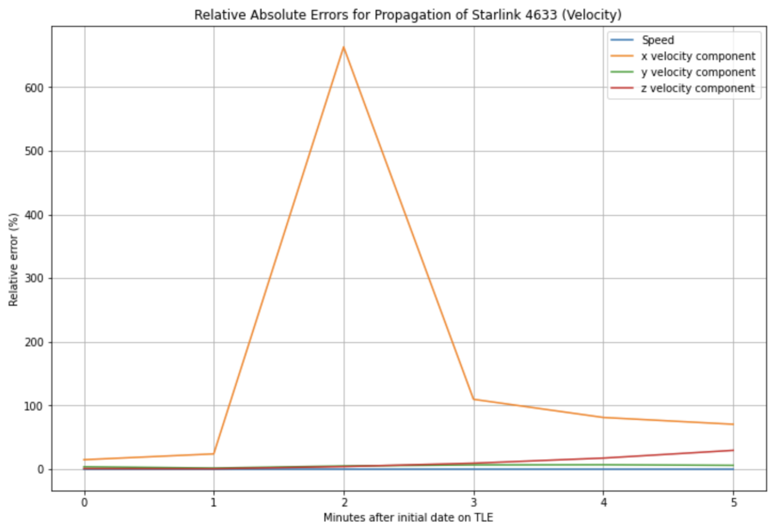

Figure 2: Magnitude of relative error for velocity magnitude and components of Starlink-4633

on Celestrak and it roughly covers the same time period than the information available at Privateer. Particularly, the TLE used for Starlink-1007 from Celestrak is dated on 2022-12-08T18:43:06.171. Similarly, the data for the same object from Privateer ranges from 2022-12-06T12:00:00.000 to 2022-12-11T12:00:00.000. This allows us to compare the position and velocity outputs from SGP4 and the proprietary tools from Privateer, resulting in useful information that mostly agree with our results for short time intervals smaller than a few minutes. First, we calculate the magnitude of the relative error between both methods. We do this for the position vector magnitude and each of its components:

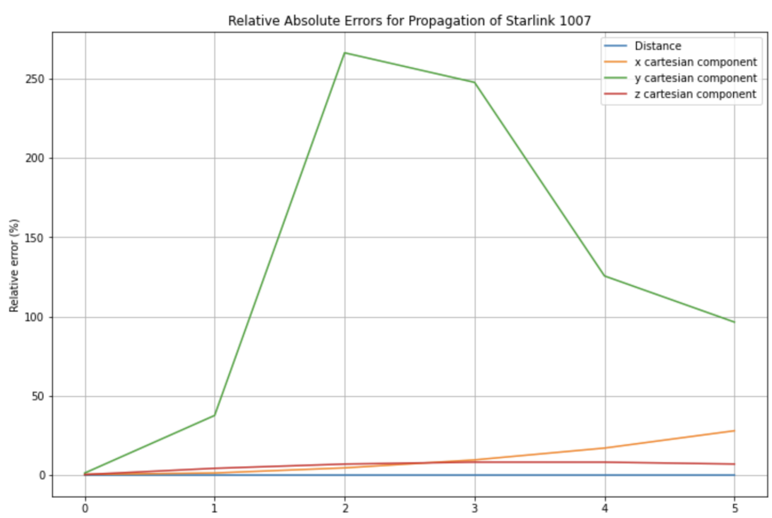

Figure 3: Magnitude of relative error for position magnitude and components of Starlink-1007

Similarly, we also compare the velocities from both sources obtaining similar results. The fact that the significant surge in velocity error for the x component does not produce an error increase for any of the position variables in the next iteration indicates that privateer may use significantly different methods for propagating or obtaining data from satellites.

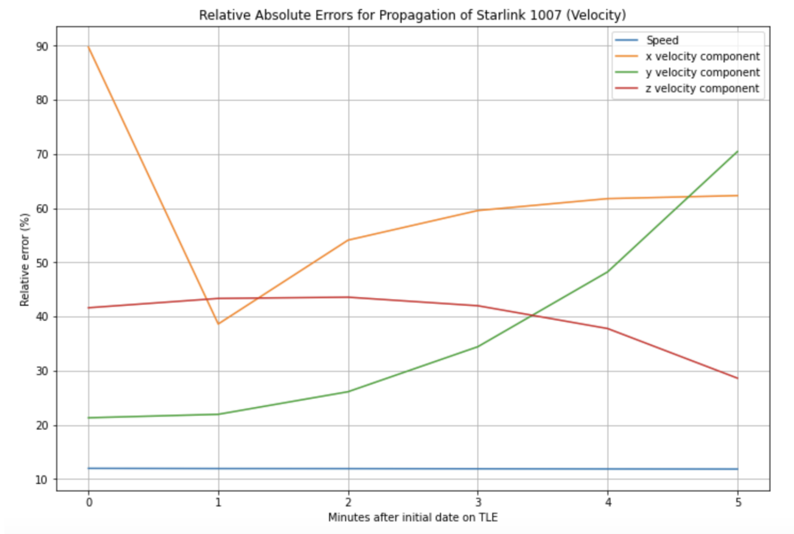

Figure 4: Magnitude of relative error for position magnitude and components of Starlink-1007

In conclusion, the data does not seem to spiral out of control for short amounts of time. Nevertheless, the error does not seem to stabilize or converge to a specific value, as expected. Further analysis with more satellites and perhaps longer time lapses is needed to determine the usefulness of our approach. The error in position is very encouraging for the first satellite, and mostly for the second one as well. The fact that sudden surges in velocity error do not result in obvious changes for the position seems to indicate that both methods may be significantly different. The method used for calculating the error is currently inadequate and relies at times on manual calculation for some data. It will be improved.

References

[1] S. Aida and M. Kirschner, “Accuracy assessment of sgp4 orbit information conversion into osculating elements”, (2013).

[2] D. Oltrogge, R. AGI, and A. Jens, “Parametric characterization of sgp4 theory and tle positional accuracy”, (2014).

[3] T. Kelso et al., “Validation of sgp4 and is-gps-200d against gps precision ephemerides”, (2007).


```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


import pandas as pd
import numpy as np
import math


```

```
# In[2]:


import spiceypy as sp
import astropy.coordinates
import re
import sgp4.api as sg
import astropy.units as u
from astropy.coordinates import SkyCoord


# In[3]:


import matplotlib.pyplot as plt
import os
import sys
from timeit import default_timer as timer
from astropy.time import Time
from astropy.time import TimeDelta
import datetime as dt
import timeit
import skyfield
# from skyfield.framelib import ecliptic_frame # For rotation matrices
from skyfield.api import EarthSatellite # For time calculations
from skyfield.api import load
from skyfield.api import N,S,E,W, wgs84
from skyfield.positionlib import Barycentric


# Starlink 4633

# In[71]:


# Starlink 4633 comparison

s = '1 53964U 22125A   22307.16667824 -.01246499  00000+0 -33583-1 0  9992'
t = '2 53964  53.2173 313.7382 0001472  50.2418 222.4739 15.40357252  4806'

satellite = sg.Satrec.twoline2rv(s, t)


# In[75]:


# Privateer data ranges from 2022-11-01T12:00:00.000 to 2022-11-06T12:00:00.000

t = Time(59884.50000000, format='mjd')  # convert to Time format (MJD)
t_change = TimeDelta(60, format='sec')

count = 1

while count <= 7200: # 5 days
    jd_t = t + 2400000.5
    fr, whole = math.modf(float(str(jd_t)))  # fr = digits after decimal of MJD

    e, r, v = satellite.sgp4(float(str(jd_t)), round(fr, 12))  # r is [x,y,z] for
    propagated satellite
```

```python
    x, y, z = r[0], r[1], r[2]
    vx, vy, vz = v[0], v[1], v[2]
    # print(x, y, z)
    # print(vx, vy, vz)

    t += t_change # 1 minute intervals
    count += 1


# In[4]:


# Position Error Plots: Relative error of magnitude and components

sgp4_data = [[5712.812466306943, -3561.55784734158, 1009.2470802383032],
             [5861.615790327355, -3001.073198487873, 1720.9926416647727],
             [5903.7789920140285, -2385.989091040445, 2401.340192203756],
             [5838.536396469713, -1727.4964597956075, 3037.8774007050497],
             [5667.0880765163565, -1037.5791548017778, 3618.9976176844516],
             [5392.574326885202, -328.7912409280184, 4134.11348873785]]

privateer_data = [[5758.101045387836, -3414.100482326623, 1231.5507692745161],
             [5830.709224574032, -3131.754532231567, 1586.4744375004084],
             [5876.716085143028, -2835.1267755439203, 1934.1842689184464],
             [5895.898257087167, -2525.564771718134, 2273.044557197647],
             [5888.260674021064, -2204.510632667524, 2601.5025094598245],
             [5853.815592134328, -1873.4198032164295, 2918.1053452860433]]


# r relative error
rerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(np.linalg.norm(privateer_data[i]) - np.linalg.norm(sgp4_data[
    i]))/np.linalg.norm(sgp4_data[i]))
    rerr.append(rel_err)

# x relative error
xerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][0] - sgp4_data[i][0])/sgp4_data[i][0])
    xerr.append(rel_err)

# y relative error
yerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][1] - sgp4_data[i][1])/sgp4_data[i][1])
    yerr.append(rel_err)

# z relative error
zerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][2] - sgp4_data[i][2])/sgp4_data[i][2])
    zerr.append(rel_err)

fig, ax = plt.subplots(figsize=(12,8))
plt.grid()
title = "Relative Absolute Errors for Propagation of Starlink 4633"
plt.title(title)
plt.ylabel('Relative error (%)')
```

```python
plt.xlabel('Minutes after initial date on TLE')
plt.plot(rerr)
plt.plot(xerr)
plt.plot(yerr)
plt.plot(zerr)
plt.legend(['Distance','x cartesian component','y cartesian component','z cartesian
    component'])


# In[6]:


# Velocity Error Plots: Relative error of magnitude and components

sgp4_data = [[1.6781468207971149, 4.414141579436392, 6.026288270952519],
             [0.7981121501322271, 4.913119459768193, 5.818111998978295],
             [-0.0964848532920048, 5.322735762912325, 5.503774433488314],
             [-0.9892617498478192, 5.63549150209669, 5.089050297534009],
             [-1.8638927044954192, 5.845690731805137, 4.581569248851075],
             [-2.7044187345291975, 5.949538483018555, 3.9906596968137675]]

privateer_data = [[1.4288563821009157, 4.579989910384911, 5.967043303150862],
             [0.9898348798815465, 4.828229669527856, 5.859597279401854],
             [0.5432572062270737, 5.055559935625433, 5.726015055699247],
             [0.0962195262428886, 5.259148202615219, 5.565042239729997],
             [-0.35082970512957984, 5.438630705079241, 5.3795699415615985],
             [-0.7969567812342215, 5.593558148568212, 5.1697218234047675]]


# v relative error
rerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(np.linalg.norm(privateer_data[i]) - np.linalg.norm(sgp4_data[
    i]))/np.linalg.norm(sgp4_data[i]))
    rerr.append(rel_err)

# vx relative error
xerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][0] - sgp4_data[i][0])/sgp4_data[i][0])
    xerr.append(rel_err)

# vy relative error
yerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][1] - sgp4_data[i][1])/sgp4_data[i][1])
    yerr.append(rel_err)

# vz relative error
zerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][2] - sgp4_data[i][2])/sgp4_data[i][2])
    zerr.append(rel_err)

fig, ax = plt.subplots(figsize=(12,8))
plt.grid()
title = "Relative Absolute Errors for Propagation of Starlink 4633 (Velocity)"
plt.title(title)
plt.ylabel('Relative error (%)')
```

```
plt.xlabel('Minutes after initial date on TLE')
plt.plot(rerr)
plt.plot(xerr)
plt.plot(yerr)
plt.plot(zerr)
plt.legend(['Speed','x velocity component','y velocity component','z velocity component'])


# Starlink 1007

# In[60]:


# Starlink 1007 comparison

s = '1 44713U 19074A   22342.77993253  .00004591  00000+0  32676-3 0  9993'
t = '2 44713  53.0559 114.8779 0001409  66.6824 293.4313 15.06412418169857'

satellite = sg.Satrec.twoline2rv(s, t)


# In[79]:


# Privateer ranges from 2022-12-06T12:00:00.000 to 2022-12-11T12:00:00.000

t = Time(59919.50000000, format='mjd')  # convert to Time format (MJD)
t_change = TimeDelta(60, format='sec')

count = 1

while count <= 7200: # 5 days
    jd_t = t + 2400000.5
    fr, whole = math.modf(float(str(jd_t)))  # fr = digits after decimal of MJD

    e, r, v = satellite.sgp4(float(str(jd_t)), round(fr, 12))  # r is [x,y,z] for propagated satellite
    x, y, z = r[0], r[1], r[2]
    vx, vy, vz = v[0], v[1], v[2]
    # print(x, y, z)
    # print(vx, vy, vz)

    t += t_change # 1 minute intervals
    count += 1


# In[7]:


# Error plots: Relative error of magnitude and components

sgp4_data = [[5186.0057779413455, -1893.762729553324, -4189.5021984017185],
             [5038.387605290658, -1111.2401400264566, -4627.737167688791],
             [4804.176228659034, -309.61768658116233, -4986.218366835685],
             [4487.443114903397, 497.32423493657575, -5258.815489381309],
             [4093.6625808621675, 1295.728201693788, -5440.872611072728],
             [3629.6162975570533, 2071.892168589236, -5529.2824735242775]]

```

```
privateer_data = [[5166.391626209741, -1917.5339389812832, -4202.57948966722],
                  [5105.0020788551765, -1529.0912008323176, -4430.990788230552],
                  [5021.593250212112, -1134.096519957054, -4640.260746862787],
                  [4916.584410325542, -734.2576921365774, -4829.480783468913],
                  [4790.465354410709, -331.2398352442685, -4997.894350747439],
                  [4643.719974763799, 73.20737424057101, -5144.775018462607]]


# r relative error
rerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(np.linalg.norm(privateer_data[i]) - np.linalg.norm(sgp4_data[i]))/np.linalg.norm(sgp4_data[i]))
    rerr.append(rel_err)

# x relative error
xerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][0] - sgp4_data[i][0])/sgp4_data[i][0])
    xerr.append(rel_err)

# y relative error
yerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][1] - sgp4_data[i][1])/sgp4_data[i][1])
    yerr.append(rel_err)

# z relative error
zerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][2] - sgp4_data[i][2])/sgp4_data[i][2])
    zerr.append(rel_err)

fig, ax = plt.subplots(figsize=(12,8))
plt.grid()
title = "Relative Absolute Errors for Propagation of Starlink 1007"
plt.title(title)
plt.ylabel('Relative error (%)')
plt.xlabel('Minutes after initial date on TLE')
plt.plot(rerr)
plt.plot(xerr)
plt.plot(yerr)
plt.plot(zerr)
plt.legend(['Distance','x cartesian component','y cartesian component','z cartesian component'])


# In[78]:


# Velocity Error Plots: Relative error of magnitude and components

sgp4_data = [[-0.44215625127094654, 5.283927607930954, -6.783419863760003],
             [-1.9666985116670517, 5.358343603456702, -6.441890733123883],
             [-3.424399338854064, 5.256008098660114, -5.888052417148206],
             [-4.767315233757892, 4.980849020064989, -5.140888797721124],
             [-5.951714890621503, 4.542472702999641, -4.225620342527024],
             [-6.939424772919748, 3.9557108233188387, -3.1727203902931262]]
```

```
privateer_data = [[-0.8389760433209092, 6.409333368267421, -3.9603373004505813],
                  [-1.2071902732539261, 6.5337180586906225, -3.6502909779148585],
                  [-1.571691939583276, 6.628120635201371, -3.322918788553278],
                  [-1.9272610192447481, 6.695247298390183, -2.982348618103537],
                  [-2.2754415122956626, 6.733781193543453, -2.6293922986241522],
                  [-2.6141743610286317, 6.742933589686734, -2.2648373639270662]]


# v relative error
rerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(np.linalg.norm(privateer_data[i]) - np.linalg.norm(sgp4_data[i]))/np.linalg.norm(sgp4_data[i]))
    rerr.append(rel_err)

# vx relative error
xerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][0] - sgp4_data[i][0])/sgp4_data[i][0])
    xerr.append(rel_err)

# vy relative error
yerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][1] - sgp4_data[i][1])/sgp4_data[i][1])
    yerr.append(rel_err)

# vz relative error
zerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][2] - sgp4_data[i][2])/sgp4_data[i][2])
    zerr.append(rel_err)

fig, ax = plt.subplots(figsize=(12,8))
plt.grid()
title = "Relative Absolute Errors for Propagation of Starlink 1007 (Velocity)"
plt.title(title)
plt.ylabel('Relative error (%)')
plt.xlabel('Minutes after initial date on TLE')
plt.plot(rerr)
plt.plot(xerr)
plt.plot(yerr)
plt.plot(zerr)
plt.legend(['Speed','x velocity component','y velocity component','z velocity component'])


# In[64]:


# SES 20 comparison

s = '1 53960U 22123A   22306.61541150  .00000104  00000+0  00000+0 0  9992'
t = '2 53960   0.0492 259.4379 0002986 115.4657 107.3032  1.00272189   290'

satellite = sg.Satrec.twoline2rv(s, t)


# In[65]:
```

```python

# Privateer ranges from 2022-11-01T12:00:00.000 to 2022-11-06T12:00:00.000

t = Time(59884.50000000, format='mjd')  # convert to Time format (MJD)
t_change = TimeDelta(60, format='sec')

count = 1

while count <= 7200: # 5 days
    jd_t = t + 2400000.5
    fr, whole = math.modf(float(str(jd_t)))  # fr = digits after decimal of MJD

    e, r, v = satellite.sgp4(float(str(jd_t)), round(fr, 12))  # r is [x,y,z] for propagated satellite
    x, y, z = r[0], r[1], r[2]
    print(x, y, z)

    t += t_change # 1 minute intervals
    count += 1


# In[8]:


# Error plots: Relative error of magnitude and components

sgp4_data = [[7636.830959860919, 41461.344836723765, -1.4322440379173866],
             [7273.656023976714, 41526.69803436998, -1.8799321920885201],
             [6909.923915966385, 41588.8702172758, -2.327433528709936],
             [6545.662502101271, 41647.85664572445, -2.7747136408960613],
             [6180.899688813813, 41703.65282418863, -3.2217381424753424],
             [5815.663420556347, 41756.25450166828, -3.6684726706332995]]

privateer_data = [[7849.573469709278, 41421.11211177337, -19.054606588777077],
             [668.22619853303, 41455.1143314384, -18.88149102901193],
             [7486.732073363855, 41488.3226068779, -18.70799211941946],
             [7305.094569714989, 41520.736292534675, -18.534106991124183],
             [7123.317165430435, 41552.35477788082, -18.359842548883744],
             [6941.403340281259, 41583.17747608506, -18.185209952749158]]


# r relative error
rerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(np.linalg.norm(privateer_data[i]) - np.linalg.norm(sgp4_data[i]))/np.linalg.norm(sgp4_data[i]))
    rerr.append(rel_err)

# x relative error
xerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][0] - sgp4_data[i][0])/sgp4_data[i][0])
    xerr.append(rel_err)

# y relative error
yerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][1] - sgp4_data[i][1])/sgp4_data[i][1])
```

```
    yerr.append(rel_err)

# z relative error
zerr = []
for i in range(0, len(sgp4_data)):
    rel_err = abs(100*(privateer_data[i][2] - sgp4_data[i][2])/sgp4_data[i][2])
    zerr.append(rel_err)

fig, ax = plt.subplots(figsize=(12,8))
plt.grid()
title = "Relative Absolute Errors for Propagation of SES 20"
plt.title(title)
plt.ylabel('Relative error (%)')
plt.xlabel('Minutes after initial date on TLE')
plt.plot(rerr)
plt.plot(xerr)
plt.plot(yerr)
plt.plot(zerr)
plt.legend(['Distance','x cartesian component','y cartesian component','z cartesian
    component'])


# In[ ]:
```

Listing 1: Code used for calculations.