Problem 1.

Basic orbital parameters: remind yourself:

(a) Determine the kinetic, potential, and total energy per unit mass, and the magnitude of the moment of momentum (or, angular momentum due to orbital motion) for the HST

$$p = \frac{h^2}{\mu} = \frac{52519.6585^2}{398600.4} = 6920 km$$
$$r = \frac{p}{1 + e\cos\theta} = \frac{6920}{1 + 0.0002935\cos 0.5652} = 6918.285 km$$

$$\begin{aligned}
\text{alt} &= \big(a(1 - e\cos E)\big) - 6378 \text{ km} \\
&= \big(6920(1 - 0.0002935\cos 5.71815)\big) - 6378 \\
&= 540.28 \text{ km}
\end{aligned}$$

$$\begin{aligned}
\text{vel} &= \sqrt{\mu\left(\frac{2}{\text{alt} + 6378} - \frac{1}{a}\right)} \\
&= \sqrt{398600.4\left(\frac{2}{540.28 + 6378} - \frac{1}{6920}\right)} \\
&= 13.1465 \text{ km/s}
\end{aligned}$$

$$\begin{aligned}
\epsilon_p &= -\frac{\mu}{r} = -\frac{398600.4}{6918.285} = -57.615\frac{J}{kg} \\
\epsilon_k &= \frac{v^2}{2} = \frac{13.1465^2}{2} = 86.41614\frac{J}{kg} \\
\epsilon &= \epsilon_k + \epsilon_p = \frac{v^2}{2} - \frac{\mu}{r} = 28.80114\frac{J}{kg}
\end{aligned}$$

$$\begin{aligned}
h &= \sqrt{a(1 - e^2)\mu} \\
&= \sqrt{6920(1 - (0.0002935)^2)398600.4} \\
&= 52519.6585 \text{ km}^2\text{/s}
\end{aligned}$$

Problem 2.

Assume your spacecraft is an elliptical transfer orbit, with perigee of 150km above Earth mean surface, and apogee at HST mean altitude.

HST mean altitude is 552.7km.

(a) Compute the value (deg) of the true anomaly one hour after perigee passage.

$$\begin{aligned}
R_a &= 552.7 \text{ km} + 6378 \text{ km} = 6930.7 \text{ km} \\
R_p &= 150 \text{ km} + 6378 \text{ km} = 6528 \text{ km} \\
a &= \frac{R_a + R_p}{2} = 6729.35 \text{ km} \\
e &= \frac{R_a - R_p}{R_a + R_p} = 0.0299 \\
M_m &= \sqrt{\frac{\mu}{a^3}} = \sqrt{\frac{398600.4}{(6729.35)^3}} = 0.0011 \\
M &= M_m \cdot t = 0.0011 \cdot 60 \cdot 60 = 3.96 \text{ rads} \\
E &= E_o - \frac{E_o - e \sin E_o - M}{1 - e \cos E_o} \\
&= M - \frac{M - e \sin M - M}{1 - e \cos M} \\
&= 3.96 - \frac{3.96 - 0.299 \sin 3.96 - 3.96}{1 - 0.299 \cos 3.96} \\
&= 3.7787 \text{ rads} \\
\nu &= \cos^{-1}\left(\frac{\cos E - e}{1 - e \cos E}\right) \\
&= \cos^{-1}\left(\frac{\cos 3.7787 - 0.0299}{1 - 0.0299 \cos 3.7787}\right) \\
&= 2.5220
\end{aligned}$$

(b) Compute the magnitude of the Earth-relative velocity of your spacecraft at this same point

$$\begin{aligned}
\text{alt} &= \big(a(1 - e \cos E)\big) - 6378 \text{ km} \\
&= \big(6729.35(1 - 0.0299 \cos 3.7787)\big) - 6378 \\
&= 513.0846 \text{ km} \\
\text{vel} &= \sqrt{\mu\left(\frac{2}{\text{alt} + 6378} - \frac{1}{a}\right)} \\
&= \sqrt{398600.4\left(\frac{2}{513.0846 + 6378} - \frac{1}{6729.35}\right)} \\
&= 7.5135 \text{ km/s}
\end{aligned}$$

Problem 3.

Obtain the Two-Line-Element for HST at an epoch of your choosing.

```
HST
1 20580U 90037B   16031.61163492  .00001273  00000-0  69179-4 0  9996
2 20580  28.4704  87.0856 0002935 165.2440 327.6400 15.08104961214326
```

```
Epoch Time: Sun Jan 31 2016 06:40:45 GMT-0800 (PST)
```

(a) Write down the 6 orbital elements h, e, I, Ω, ω, and θ
I = inclination = 28.4704 deg = 0.4969 rad
Ω = right ascension of ascending node = 87.0856 deg = 1.5199 rad
e = eccentricity = 0.0002935
ω = argument of perigee = 165.2440 deg = 2.8841 rad
M = Mean Anomaly = 327.640 deg = 5.7184 rad
n = Mean Motion = 15.08104961 rev/day = 0.00109672488 rads/s

$$\begin{aligned}
\theta = \text{true anomaly} &= \cos^{-1}\left(\frac{\cos E - e}{1 - e\cos E}\right) \\
&= \cos^{-1}\left(\frac{\cos 5.71815 - 0.0002935}{1 - 0.0002935\cos 5.71815}\right) \\
&= 0.5652 \text{ rad} \\
a &= \left(\frac{\mu}{n^2}\right)^{1/3} \\
&= \left(\frac{398600.4}{(0.00109672488)^2}\right)^{1/3} \\
&= 6920 \text{ km} \\
h = \text{angular momentum} &= \sqrt{a(1-e^2)\mu} \\
&= \sqrt{6920(1-(0.0002935)^2)398600.4} \\
&= 52519.6585 \text{ km}^2/\text{s}
\end{aligned}$$

(b) Also compute the eccentric anomaly E.

$$\begin{aligned}
M &= E - e\sin E \\
5.7184 &= E - 0.0002935\sin E \\
E &= 5.71815 \text{ rad, via Newton-Raphson method}
\end{aligned}$$

(c) Using class notes (Lecture 7, Thurs 1/26/16, "Compute State Vector from Orbital Elements"), find the HST state vector at this time, in the geocentric equatorial reference frame. (Also see Curtis book, Sec 4.6 and App D2 (SmartSite).

```
import numpy as np


def sv_from_coe(coe, mu):
    '''
    This function computes the state vector (r,v) from the
    classical orbital elements (coe).

    mu   - gravitational parameter (km^3/s^2)
    coe  - orbital elements [h e RA incl w TA]
           where
               h    = angular momentum (km^2/s)
               e    = eccentricity
               RA   = right ascension of the ascending node (rad)
               incl = inclination of the orbit (rad)
               w    = argument of perigee (rad)
               TA   = true anomaly (rad)
    R3_w - Rotation matrix about the z-axis through the angle w
    R1_i - Rotation matrix about the x-axis through the angle i
    R3_W - Rotation matrix about the z-axis through the angle RA
    Q_pX - Matrix of the transformation from perifocal to geocentric
           equatorial frame
    rp   - position vector in the perifocal frame (km)
    vp   - velocity vector in the perifocal frame (km/s)
    r    - position vector in the geocentric equatorial frame (km)
    v    - velocity vector in the geocentric equatorial frame (km/s)
    '''

    h    = coe[0]
    e    = coe[1]
    RA   = coe[2]
    incl = coe[3]
    w    = coe[4]
    TA   = coe[5]

    #...Equations 4.45 and 4.46 (rp and vp are column vectors):
    rp = (h**2/mu) * (1/(1 + e*np.cos(TA))) * np.array([np.cos(TA), np.sin(TA), 0])
    vp = (mu/h) * np.array([-np.sin(TA), (e + np.cos(TA)), 0])

    #...Equation 4.34:
    R3_W = np.array([[ np.cos(RA),  np.sin(RA),  0],
                     [-np.sin(RA),  np.cos(RA),  0],
                     [          0,           0,  1]])

    #...Equation 4.32:
    R1_i = np.array([[1,              0,             0],
                     [0,   np.cos(incl),  np.sin(incl)],
                     [0,  -np.sin(incl),  np.cos(incl)]])

    #...Equation 4.34:
    R3_w = np.array([[ np.cos(w),  np.sin(w),  0],
                     [-np.sin(w),  np.cos(w),  0],
                     [         0,          0,  1]]);
```

```

    #...Equation 4.49:
    Q_pX = (R3_w @ R1_i @ R3_W).T

    #...Equations 4.51:
    r = Q_pX @ rp;
    v = Q_pX @ vp;

    return r, v

coe = np.array([52519.6585, 0.0002935, 1.5199, 0.4969, 2.8841, 0.5652])
mu = 398600.4 #km^3/s^2
sv_from_coe(coe, mu)

#(array([ 1504.15011252, -6678.50694298,  -998.86945097]),
# array([ 6.46890726,  1.97156421, -3.44905071]))
```

Problem 4.

Plot the magnitudes vs θ of the three vector components of the perturbing gravitational potential $\boldsymbol{b}$ for one orbit of the HST (Lecture 8, p14, Thurs 1/28/16, Curtis eqn 12.30)

$$p_r = -\frac{\mu}{r^2}\frac{3}{2}J_2\left(\frac{R}{r}\right)^2\left[1 - 3\sin^2 i\sin^2(\omega+\theta)\right]$$
$$p_\perp = -\frac{\mu}{r^2}\frac{3}{2}J_2\left(\frac{R}{r}\right)^2\sin^2 i\sin[2(\omega+\theta)]$$
$$p_h = -\frac{\mu}{r^2}\frac{3}{2}J_2\left(\frac{R}{r}\right)^2\sin 2i\sin(\omega+\theta)$$

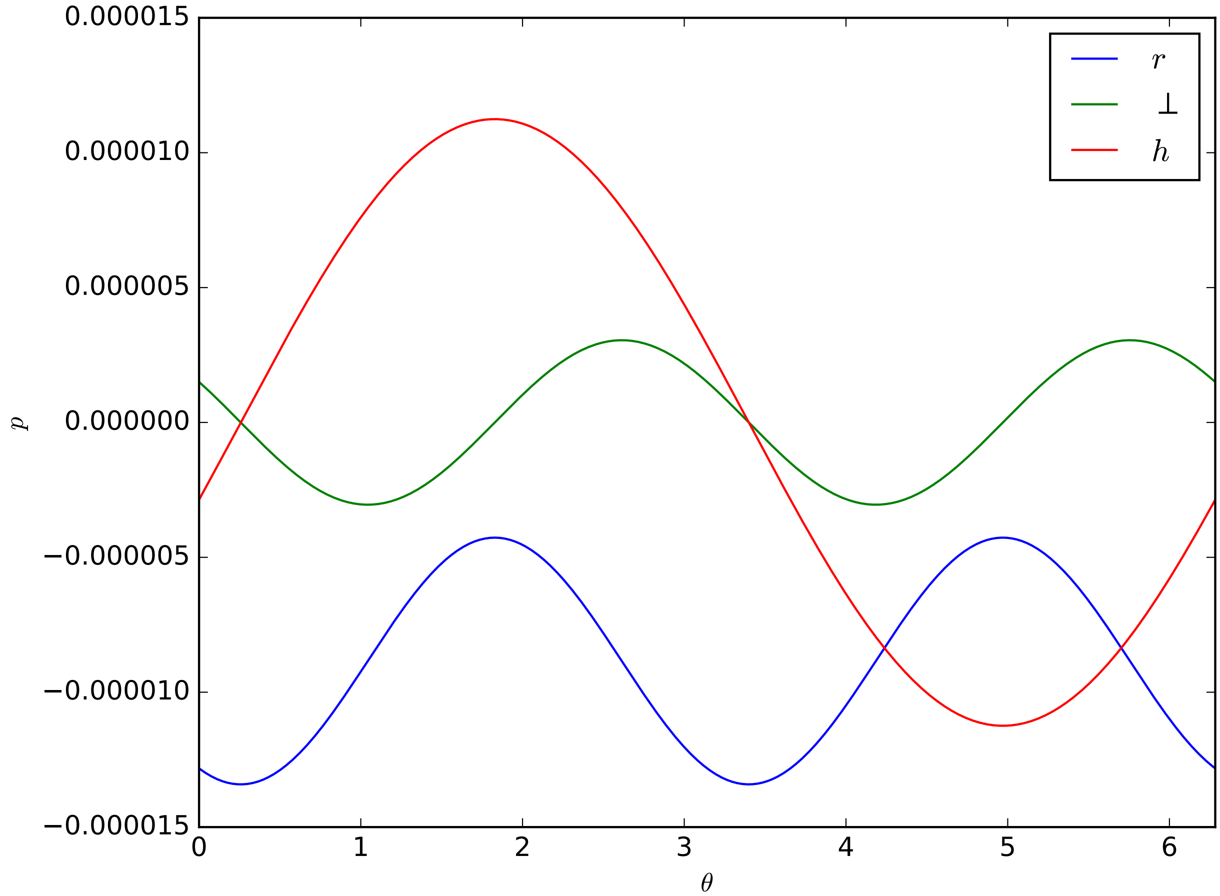

Figure 1: Magnitude of perturbing gravitational effects, assuming a constant $R = r =$ 6946km.

Problem 5.

Drag forces:

(a) Estimate the drag force imposed on the HST at its actual altitude, and also as if it were at ISS altitude. Do this for two cases, solar min and solar max, using the NASA atmospheric model you used in HW #2. Assume a non-rotating Earth. List all other assumptions.

	Low	High
ISS	6.370e-16	5.138e-15
HST	4.362e-17	4.536e-16

Table 1: ρ, atmospheric density, at ISS (400km) and HST (500km) altitudes during low and high solar activity (g/cm^{-3}).

$$D = -\frac{1}{2}\rho S C_D v_r^2 \left(\frac{v_r}{|v_r|}\right)$$

We must assume several things to fill out this equation:

(i) We guess the 'exposed' surface area, as a constant $S \approx 13.2 \text{ m} \cdot 4.2 \text{ m} = 55.44 \text{ m}$

(ii) We guess a value for the coefficient of drag, $C_D \approx 2.5$

(iii) We specify an approximate velocity of HST, $v_r = 6.47\hat{x} + 1.97\hat{y} - 3.45\hat{z}$ km/s

Plugging in these values, we arrive at:

	Low	High
ISS	2.54e-09	2.05e-08
HST	1.74e-10	1.81e-09

Table 2: D, atmospheric drag, at ISS (400km) and HST (500km) altitudes during low and high solar activity (Newtons).

(b) Explain why drag tends to circularize an elliptical orbit.

Drag is largest at perigee, as this is when both velocity and atmospheric density are the largest. Energy is lost at lower altitudes of elliptical orbits, leaving less velocity to push back out to apogee. Velocity continues to be reduced the most at the lowest part of the orbit until drag is constant throughout the orbit, at which point the orbit is circularized.

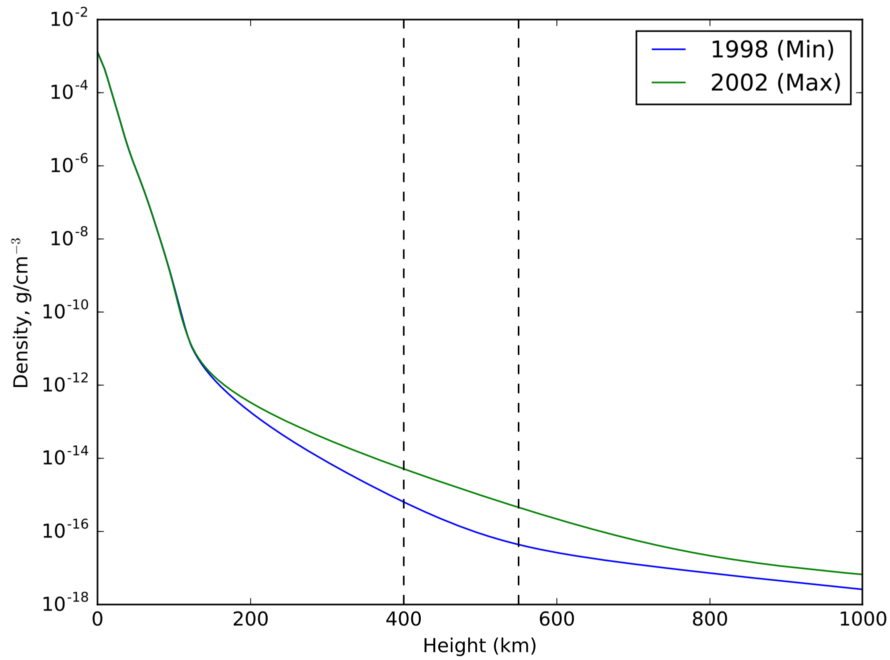

Figure 2: Atmospheric density by altitude during solar min and solar max.

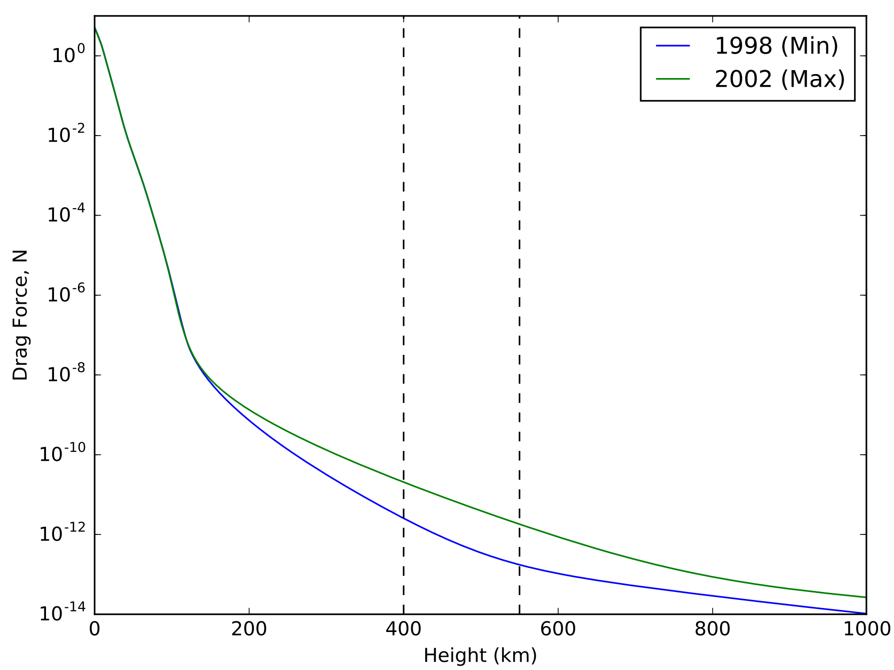

Figure 3: Atmospheric drag by altitude during solar min and solar max.

Problem 6.

Referring to the Gaussian form of the Lagrange Planetary Equations (eqn 4.34 in the text):

(a) For a retrograde burn, which orbital parameters will change, and by which sign?

T is defined along the positive direction with the spacecraft's velocity vector. Assuming a burn purely along the anti-velocity vector, and with all else being equal, this would lead to $\frac{da}{d\theta} \propto -T$, $\frac{de}{d\theta} \propto -T$, $\frac{d\omega}{d\theta} \propto -T$, and $\frac{dt}{d\theta} \propto T$. $\frac{di}{d\theta}$, and $\frac{d\Omega}{d\theta}$ would be be affected.

(b) To change the orbital inclination, you must burn "out of plane". What other orbital parameters will this change, if any?

W is defined normal to the orbit plane giving a right-handed system of accelerations. Assuming a burn purely along this direction, and with all else being equal, this would lead to changes in $\frac{di}{d\theta}$, and $\frac{d\Omega}{d\theta}$. Assuming $i \neq \pi n - \frac{\pi}{2}, n \in Z$, $\frac{d\omega}{d\theta}$ would also be changed. $\frac{da}{d\theta}$, $\frac{de}{d\theta}$, and $\frac{dt}{d\theta}$ would be unaffected.

(c) To increase the argument of perigee, in which direction(s) could you generate thrust?

To increase the argument of perigee, ω, you could generate thrust in the anti-radial direction, $-S$, or the $+T$ direction.

(d) For a purely in-plane burn, what is the relationship between tangential and radial thrust required to leave the argument of perigee unchanged?

$$\frac{d\omega}{d\theta} = \frac{r^2}{\mu e}\left[-\cos\theta S + \left(1 + \frac{r}{p}\right)\sin\theta T\right] - \cos i \frac{d\Omega}{d\theta} \qquad \text{Definition.}$$

$$\frac{d\omega}{d\theta} = \frac{r^2}{\mu e}\left[-\cos\theta S + \left(1 + \frac{r}{p}\right)\sin\theta T\right] \qquad \text{Purely in plane, } W = 0.$$

$$0 = \frac{r^2}{\mu e}\left[-\cos\theta S + \left(1 + \frac{r}{p}\right)\sin\theta T\right] \qquad \frac{d\omega}{d\theta} = 0$$

$$S = \left[\left(1 + \frac{r}{p}\right)\tan\theta\right] T \qquad \text{Rearranging.}$$

Problem 7.

Numerical propagation of perturbed orbits:

(a) Use the HST state vector found in Problem 3 as your initial conditions.

(b) Write the orbital equation of motion for perturbed orbits as two first order ODEs in **r** and **v**:

$$\frac{d}{dt}\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix} = \begin{bmatrix}\mathbf{v}\\ \mathbf{a}\end{bmatrix} = \begin{bmatrix}\mathbf{v}\\ -\mu\frac{\mathbf{r}}{r^3}+\mathbf{p}\end{bmatrix}$$

(c) The perturbation vector **p** is due to drag only, $\mathbf{p} = -1/2\rho v(C_D S)\mathbf{v}$.

(d) Using RK4, solve for **r** and **v** on the time interval of one orbit, once with drag and once without. Plot the difference over time for the magnitudes of **r** and **v**.

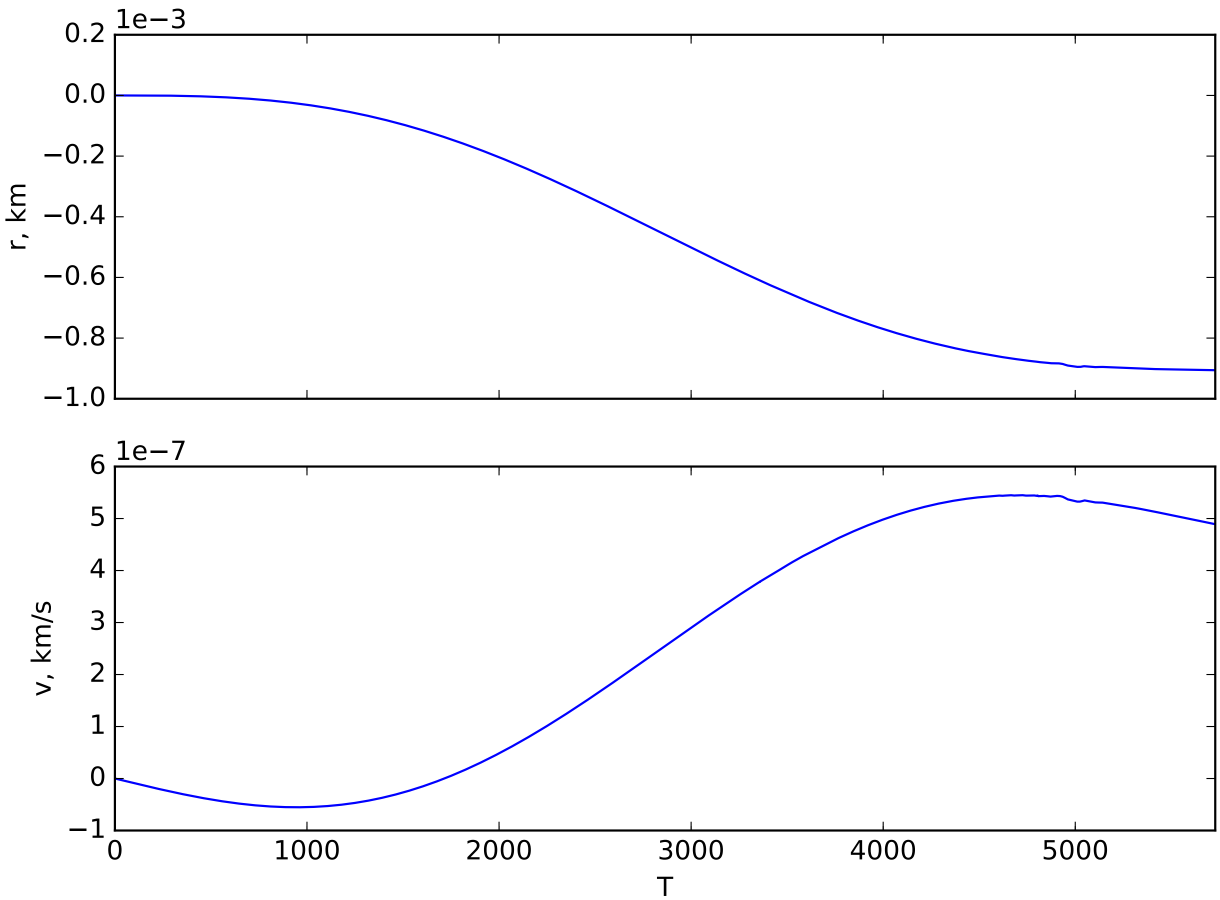

Figure 4: Difference in magnitude of **r** and **v** between drag and no drag cases for one orbit.

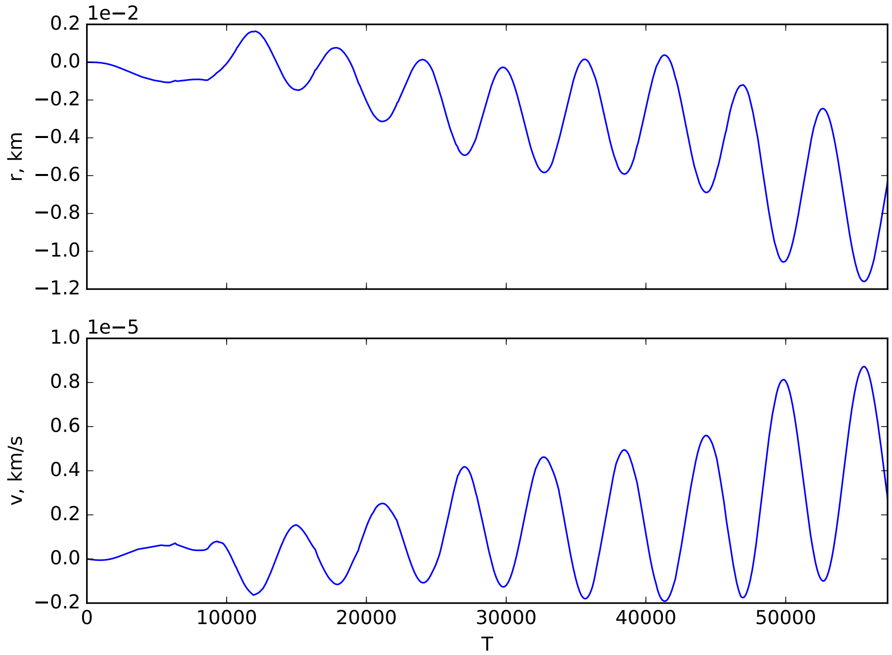

Figure 5: Difference in magnitude of **r** and **v** over ten orbits.

```python
import pandas as pd
import numpy as np
from numpy import pi
import matplotlib.pyplot as plt
from sv_from_coe import sv_from_coe
from scipy.integrate import odeint


# Constants
mu = 398600                          # Gravitational parameter (km**3/s**2)
RE = 6378                            # Earth's radius (km)
wE = np.array([0, 0, 7.2921159e-5])  # Earth's angular velocity (rad/s)

# Satellite data:
CD = 2.5                             # Drag codfficient
m = 11110                            # Mass (kg)
A = 55.44                            # Area (m**2)


def orbit(drag):
    coe0 = np.array([52519.6585, 0.0002935, 1.5199, 0.4969, 2.8841, 0.5652])

    # Obtain the initial state vector
    R0, V0 = sv_from_coe(coe0, mu)
    y0 = np.array([R0, V0]).flatten()

    t0, tf, nout = 0, 5728.8, 10000
```

```
    tspan = np.linspace(t0, tf, nout)
    y = odeint(rates, y0, tspan, args=(drag,))

    return y, tspan


def rates(f, t, drag=False):
    '''
    This function calculates the spacecraft acceleration from its position and
    velocity at time t.
    '''

    R = f[0:3]                          # Position vector (km/s)
    r = np.linalg.norm(R)               # Distance from earth's center (km)
    alt = r - RE                        # Altitude (km)
    rho = atmosphere(alt)               # Air density from US Standard Model (kf/m**3)
    V = f[3:6]                          # Velocity vector (km/s)
    Vrel = V - np.cross(wE, R)          # Velocity relative to the atmosphere (km/s)
    vrel = np.linalg.norm(Vrel)         # Speed relative to the atmosphere (km/s)
    uv = Vrel / vrel                    # Relative velocity unit vector
    ap = (-CD * A / m * rho *           # Acceleration due to drag (m/s**2)
          (1000 * vrel)**2 / 2 * uv)  # (converting units of vrel from km/s to m/s)
    a0 = -mu * R / r**3                 # Gravitational ecceleration (km/s**2)
    if drag:
        a = a0 + ap / 1000
    else:
        a = a0

    dfdt = np.array([V, a]).flatten()
    return dfdt


def atmosphere(z):
    '''
    Calculates density for altitudes from sea level through 1000 km using
    exponential interpolation.
    '''

    # Geometric altitudes (km):
    h = (
        [0, 25, 30, 40, 50, 60, 70,
         80, 90, 100, 110, 120, 130, 140,
         150, 180, 200, 250, 300, 350, 400,
         450, 500, 600, 700, 800, 900, 1000])

    # Corresponding densities (kg/m^3) from USSA76:
    r = (
        [1.225, 4.008e-2, 1.841e-2, 3.996e-3, 1.027e-3, 3.097e-4, 8.283e-5,
         1.846e-5, 3.416e-6, 5.606e-7, 9.708e-8, 2.222e-8, 8.152e-9, 3.831e-9,
         2.076e-9, 5.194e-10, 2.541e-10, 6.073e-11, 1.916e-11, 7.014e-12, 2.803e-12,
         1.184e-12, 5.215e-13, 1.137e-13, 3.070e-14, 1.136e-14, 5.759e-15, 3.561e-15])

    # Scale heights (km):
    H = (
```

```
        [7.310, 6.427, 6.546, 7.360, 8.342, 7.583, 6.661,
         5.927, 5.533, 5.703, 6.782, 9.973, 13.243, 16.322,
         21.652, 27.974, 34.934, 43.342, 49.755, 54.513, 58.019,
         60.980, 65.654, 76.377, 100.587, 147.203, 208.020, 270.010])

    # Handle altitudes outside of the range:
    if z > 1000:
        z = 1000
    elif z < 0:
        z = 0

    # Determine the interpolation interval:
    for j in range(len(h) - 1):
        if z >= h[j] and z < h[j + 1]:
            i = j
    if z == 1000:
        i = len(h) - 1

    # Exponential interpolation:
    density = r[i] * np.exp(-(z - h[i]) / H[i])
    return density


def process_orbit(drag):
    y, t = orbit(drag)
    res = pd.DataFrame(y)
    res.columns = ['R1', 'R2', 'R3', 'V1', 'V2', 'V3']
    res['T'] = t
    res['R'] = res.apply(lambda x: np.linalg.norm(np.array(x[['R1', 'R2', 'R3']])), axis=1)
    res['V'] = res.apply(lambda x: np.linalg.norm(np.array(x[['V1', 'V2', 'V3']])), axis=1)
    res['Drag'] = drag
    return res


drag = process_orbit(True)
no_drag = process_orbit(False)
res = pd.concat((drag, no_drag))
diff = res.query('Drag') - res.query('not Drag')
diff['T'] = res.query('Drag')['T']

f, ax = plt.subplots(2, sharex=True)
ax[0].set_ylabel('r, km')
ax[1].set_ylabel('v, km/s')
diff.plot(x='T', y='R', ax=ax[0], legend=False)
diff.plot(x='T', y='V', ax=ax[1], legend=False)
ax[0].ticklabel_format(style='sci', axis='y', scilimits=(0,0))
plt.tight_layout()
#plt.savefig('p7.pdf')
plt.show()
```