Table of Contents


```
%-------- HW 4 MATLAB code --------%
% Romeo Perlstein, section 0101 %
```

Q1

Matlab portion of Q1, for solving for numbers:

```
ei = 0.3;
i = 45;
omega = 270;
w = 30;
mew_mars = 4.28*10^4; % km^3/s^2
rp = 4000;
```

a

Find deltaV for transfer

```
ai = rp/(1-ei); % Find the semi-major axis
ra = ai*(1+ei); % Find the radius at apoapsis
rf = ra; % Get the final radius (of the circular orbit)
va = sqrt((2*mew_mars)/ra - mew_mars/ai); % Get the velocity at the apoapsis
 of the first orbit
vf = sqrt(mew_mars/rf); % Get the velocity of the circular orbit (constant)

% Get the deltaV
deltaV = vf-va
```

b

The overall magnitude of the velocity will increase since we need to move faster to change to a bigger orbit, and then become constant once we are moving in a circular orbit. However, since the velocity of an ellipse is not uniform throughout, our final velocity compared to the velocity of the original orbit at the periapsis results in a decrease of magnitude. This is because the maximum velocity that an orbit achieves is at the periapsis, but since we are conducting the maneuver at the apoapsis which is the slowest point of an orbit, changing our orbit to a larger circular, while increasing the velocity in magnitude, requires less deltaV

```
vp = sqrt((2*mew_mars)/rp - mew_mars/ai) % demonstration of periapsis velocity
```

```
deltaV =

    0.3921


vp =

    3.7296
```

Q2

Matlab portion of of Q2, for solving numbers:

```
mew_earth = 3.986*10^5;
rp2 = 7000;
ra2 = 10000;
i2 = 12;
omega2 = 90;
w2 = 0;
e2 = (rp2-ra2)/(-ra2-rp2);
a2 = rp2/(1-e2);
r2_des = 8000;

% [r_des, v_des, spef_energy] = orbitalElementsToCart(a2, e2, i2, omega2, w2,
 true_anom, mew_earth, "deg");
% rf2 = norm(r_des) % Easier eq to do this but I wanted to use my function :)

% Find true anomaly with respect to r
true_anom = acosd(((a2*(1-e2^2))-r2_des)/(r2_des*e2)); % value in radians!

% Now pick a location and find velocity at true anomaly:
ve2 = sqrt((2*mew_earth)/r2_des - (2*mew_earth)/(ra2+rp2));
vc2 = sqrt(mew_earth/r2_des);

% Find flight path angle
p2  = a2*(1-e2^2);
flight_path_angle = acosd(sqrt((mew_earth*p2))/(r2_des*ve2)); % angle in
 degrees

% Find delta V
deltaV2 = sqrt(ve2^2+vc2^2-(2*ve2*vc2*cosd(-flight_path_angle)))
```

C

If the burn was conducted at the other location, roughly -80 degrees, then the delta V required would be roughly the same! This is because all of the orbital parameters in use are not only constant, but also the flight path angle would be the same, so none of the parameters would be different from the first burn location

```
deltaV2 =

    1.2149
```

Q3

a

Get periapsis and apoapsis

```
a3 = 20000;
e3 = 0.3;
i3 = 5;
omega3 = 30;
w3 = 45;
w3f = 30;

rp3 = a3*(1-e3)
ra3 = a3*(1+e3)
```

b

Find delta V at intersection closest to periapsis

```
true_anom3_close_final = (45-30)/2 % degrees
true_anom3_far_final = 180+((45-30)/2) % degrees

r_inter3 = (a3*(1-e3^2))/(1+e3*cosd(true_anom3_close_final)); % Doing the
 closer one as per problem statement
v3 = sqrt(mew_earth/r_inter3);
deltaV3 = sqrt(2*v3^2*(1-cosd(45-30)))
```

```
rp3 =

       14000


ra3 =

       26000


true_anom3_close_final =

    7.5000


true_anom3_far_final =

  187.5000
```

```
deltaV3 =

    1.3916
```

Q4

a

```
mew_saturn = 3.7931187*10^7;
r4 = 60000;
i4 = 10;
v4 = sqrt(mew_saturn/r4);
deltaV4 = sqrt(2*v4^2*(1-cosd(15-10)))
```

```
deltaV4 =

    2.1935
```

Published with MATLAB® R2022b