Homework 7

Tycho Bogdanowitsch
AA 279A - Space Mechanics

March 12, 2025

Problem 1.

Part (a)

```
function [statedot] = derivative(t, state)
    % state vector is [rx ry rz vx vy vz]
    %
    % although required by form, input value t will go unused
       here
    statedot = zeros(6, 1);
    statedot(1:3) = state(4:6); % rdot = velocity

    r = state(1:3);
    accel_g = -(mu/(norm(r))^3)*r;

    statedot(4:6) = accel_g;
end % terminates MATLAB function
```

The acceleration vector for the gravitational force on the orbiting object (with direction) is given by:

$$\mathbf{a}_g = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r}$$

Part (b)

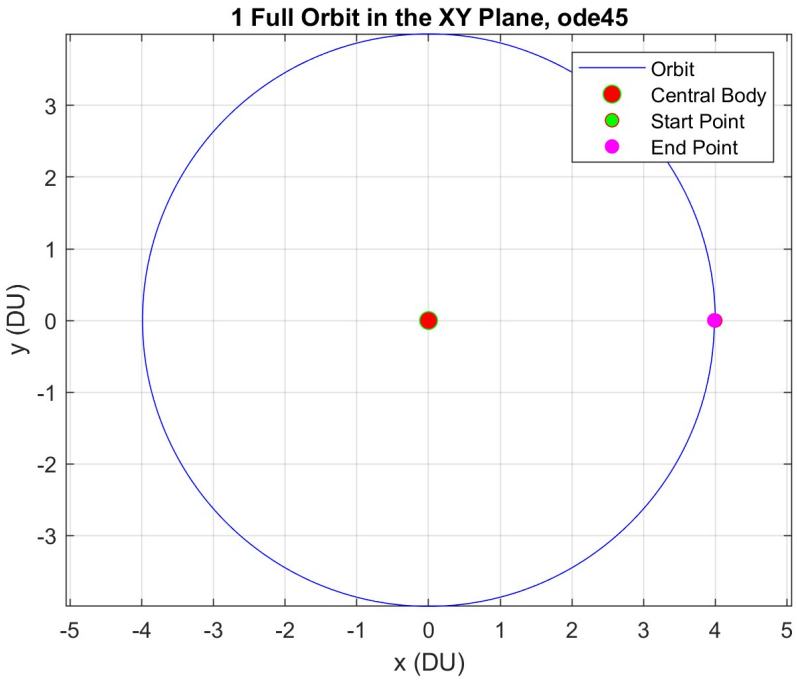

Figure 1: One Full Orbit with ode45

Part (c)

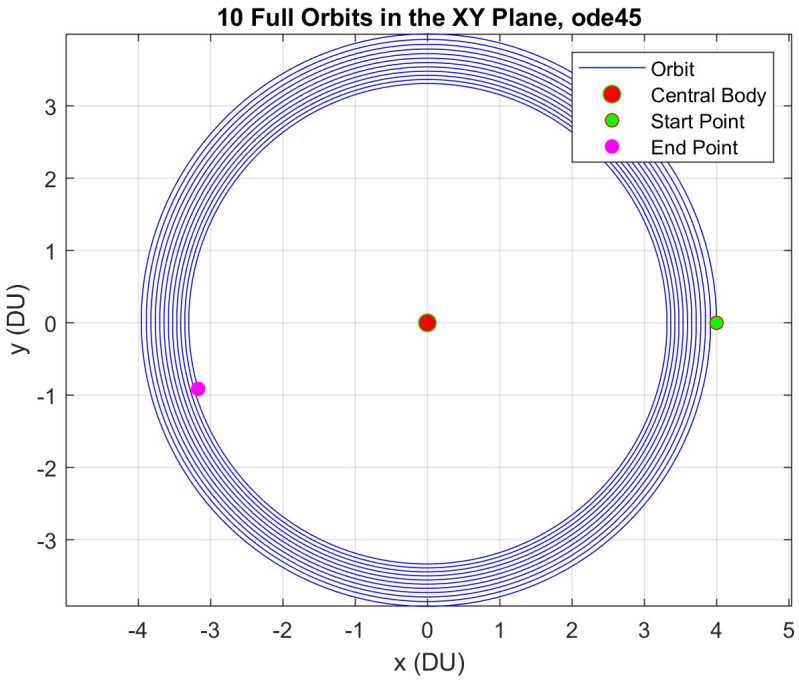

Figure 2: 10 Full Orbits with ode45

Using ode45, each successive orbit has a smaller semi-major axis. This slow decrease is a numerical effect, because only gravity is acting on this orbit and there are no perturbative forces so it should remain constant with the same semi-major axis.

Part (d)

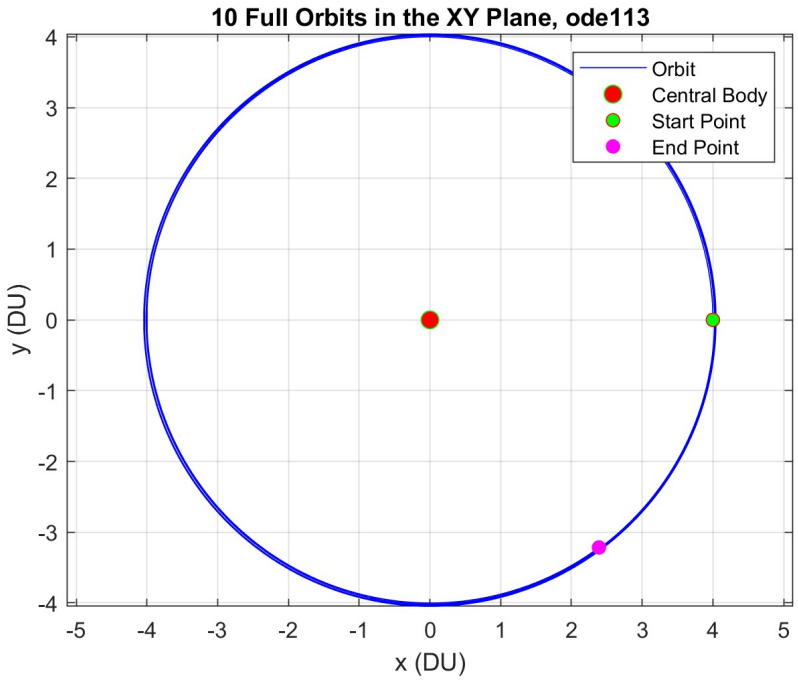

Figure 3: 10 Full Orbits with ode113

ode113 performs much better than ode45, with a much smaller variation in semi-major axis for each successive orbit. This is because ode113 is a multistep method that saves information from previous time steps to improve the current timestep. However, there are still unexpected variations over 10 orbits.

Part (e)

To reuse the Pluto model for this problem, we would need to enter new constants for initial position, initial velocity, and mu (changing from the Sun to Central Body).

Problem 2.

Part (a)

```
function [statedot] = derivative_2(t, state)
    N = 5; % number of bodies
    set_len = 7; % elements in each set
    statedot = zeros(set_len*N, 1);
    mu = zeros(1,N);
    r = zeros(3,N);
    v = zeros(3,N);
    a = zeros(3,N);

    for i = 1:N % getting the mu, r, v for each body
        index = set_len*(i-1) + 1; % 1,8,...
        mu(i) = state(index);
        r(:,i) = state(index+1:index+3);
        v(:,i) = state(index+4:index+6);
    end

    for i=1:N
        for j = 1:N
            if j~=i
                r_ij = r(:,j) - r(:,i); % get vector from body
                   i to body j
                a(:,i) = a(:,i) + mu(j)* r_ij / norm(r_ij)^3; %
                    sum up all forces
            end
        end
    end

    for i=1:N
        index = set_len*(i-1) + 1; % 1,8,...
        statedot(index) = 0; % derivative of mu is zero
        statedot(index+1:index+3) = v(:,i); % rdot = v
        statedot(index+4:index+6) = a(:,i);
    end
end
```

The gravitational acceleration for each body i due to all other bodies j is given by:

$$\mathbf{a}_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\mu_j}{\|\mathbf{r}_j - \mathbf{r}_i\|^3} (\mathbf{r}_j - \mathbf{r}_i)$$

where $\mathbf{a}_i$ is the acceleration of body i, μ_j is the gravitational parameter of body j, $\mathbf{r}_i$ and

$\mathbf{r}_j$ are the position vectors of bodies i and j, respectively, and $\|\mathbf{r}_j - \mathbf{r}_i\|$ is the distance between the two bodies. This expression is applied to every body to determine the resulting acceleration.

Part (b)

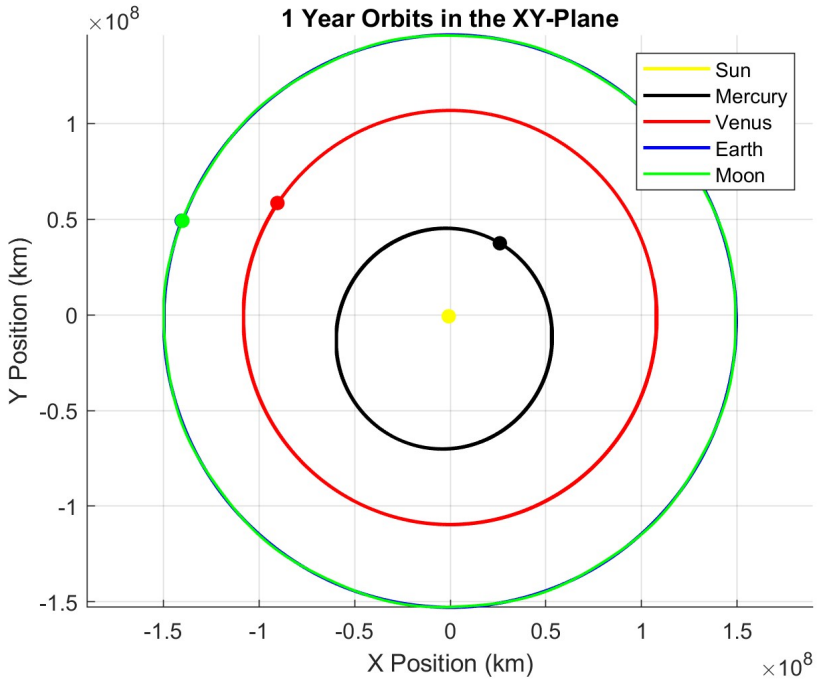

Figure 4: Inner Solar System Orbits in 1 Year

Part (c)

Counting the orbits, the Moon orbits the Earth 13 full times and about 40 percent of a full orbit in 1 year.

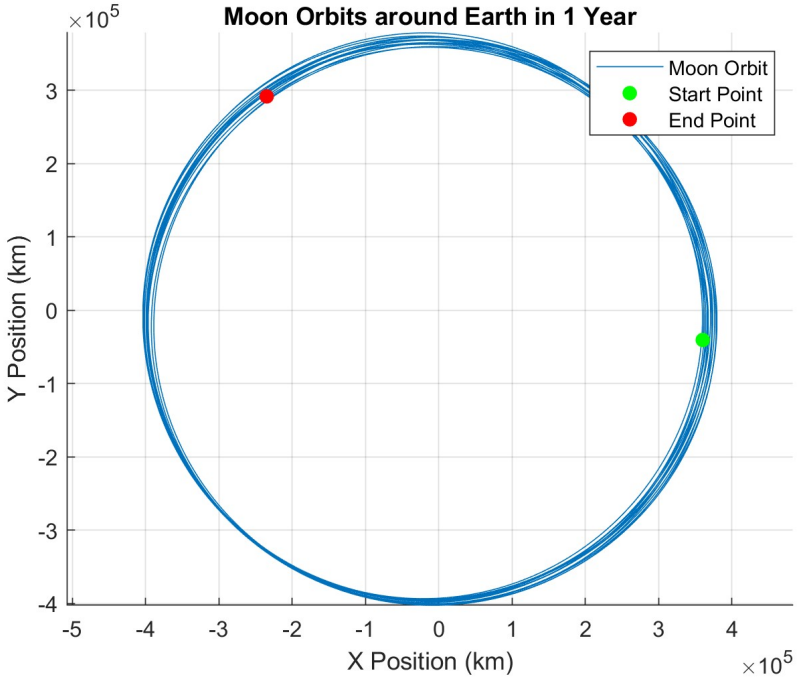

Figure 5: Moon Orbits around Earth in 1 Year

Problem 3.

Part (a)

Using the vis-viva equation with the elliptical transfer orbit and evaluating at points right before and after the two burns, the following expression can be derived for total delta V for a Hohhman transfer:

$$\Delta V_A = \left| \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_a} - \frac{1}{r_a + r_b} \right)} - \sqrt{\frac{\mu_{\text{Earth}}}{r_a}} \right|$$

$$\Delta V_B = \left| \sqrt{\frac{\mu_{\text{Earth}}}{r_b}} - \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_b} - \frac{1}{r_a + r_b} \right)} \right|$$

$$\Delta V_{\text{tot}} = \Delta V_A + \Delta V_B$$

Additionally, by finding half the period of the elliptical transfer orbit, the transfer time can be found:

$$t_{\text{trans}} = \pi \sqrt{\frac{(r_a + r_b)^3}{8\mu_{\text{Earth}}}}$$

Evaluating this for a Hohmann transfer to GEO from a circular equatorial 200 km orbit, the total Delta V is **3.932 km/s** and the transfer time is **5.259 hrs**.

Part (b)

Evaluating these same expressions for a Hohmann transfer to the Moon's orbital radius from a circular equatorial 200 km orbit, the total Delta V is **3.963 km/s** and the transfer time is **119.471 hrs**.

Problem 4.

Part (a)

The same process can be followed for finding the Delta V and transfer time for bi-elliptic transfers.

$$\Delta V_A = \left| \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_a} - \frac{1}{r_a + r_b} \right)} - \sqrt{\frac{\mu_{\text{Earth}}}{r_a}} \right|$$

$$\Delta V_B = \left| \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_b} - \frac{1}{r_b + r_c} \right)} - \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_b} - \frac{1}{r_a + r_b} \right)} \right|$$

$$\Delta V_C = \left| \sqrt{\frac{\mu_{\text{Earth}}}{r_c}} - \sqrt{2\mu_{\text{Earth}} \left(\frac{1}{r_c} - \frac{1}{r_b + r_c} \right)} \right|$$

$$\Delta V_{\text{tot}} = \Delta V_A + \Delta V_B + \Delta V_C$$

$$t_{\text{trans}_1} = \pi \sqrt{\frac{(r_a + r_b)^3}{8\mu_{\text{Earth}}}}$$

$$t_{\text{trans}_2} = \pi \sqrt{\frac{(r_b + r_c)^3}{8\mu_{\text{Earth}}}}$$

$$t_{\text{trans}_{\text{tot}}} = t_{\text{trans}_1} + t_{\text{trans}_2}$$

Evaluating this for a bi-elliptic transfer to GEO from a circular equatorial 200 km orbit, the total Delta V is **4.480 km/s** and the transfer time is **1013.442 hrs**.

Part (b)

Evaluating these same expressions for a bi-elliptic transfer to the Moon's orbital radius from a circular equatorial 200 km orbit, the total Delta V is **3.792 km/s** and the transfer time is **1289.544 hrs**.

Part (c)

Compared to the results for the Hohmann transfers, the bi-elliptic transfer minimized total Delta V for the transfer to the Moon's orbital radius but not for the transfer to GEO. The transfer time for both is significantly higher. For crewed missions, this large transfer time is undesireable because it requires more consumables to sustain the crew, leading to more mass and less performance. It also exposes the crew to the harsh conditions of space for longer periods, especially the Van Allen radiation belts in MEO. This is true even for uncrewed missions, such as GEO sats, which want to minimize the time spent in the radiation belts as much as possible.

Problem 5.

Part (a)

```
function delta_v_tot = combined(delta_i_A)
    mu_earth = 3.986e5; % km^3/s^2
    r_earth = 6378; % km

    i_initial = 28.5; % deg
    delta_i_B = i_initial - delta_i_A; % deg
    r_a = r_earth + 200; % km
    r_b = r_earth + 35786; % km
    v_pre_A = sqrt(mu_earth/r_a);
    v_post_A = sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))));
    v_pre_B = sqrt(2*mu_earth*((1/r_b)-(1/(r_a+r_b))));
    v_post_B = sqrt(mu_earth/r_b);
    delta_v_A = sqrt((v_pre_A)^2+(v_post_A)^2-2*v_pre_A*
       v_post_A*cosd(delta_i_A));
    delta_v_B = sqrt((v_pre_B)^2+(v_post_B)^2-2*v_pre_B*
       v_post_B*cosd(delta_i_B));
    delta_v_tot = delta_v_A + delta_v_B;
end
```

To determine the total Delta V for the combined manuever given a change in inclination at point A, the delta V for the Hohmann transfer is combined with the delta V for the inclination change, as given in these equations:

$$v_{\text{pre},A} = \sqrt{\frac{\mu_{\text{Earth}}}{r_a}}$$

$$v_{\text{post},A} = \sqrt{2\mu_{\text{Earth}}\left(\frac{1}{r_a} - \frac{1}{r_a + r_b}\right)}$$

$$v_{\text{pre},B} = \sqrt{2\mu_{\text{Earth}}\left(\frac{1}{r_b} - \frac{1}{r_a + r_b}\right)}$$

$$v_{\text{post},B} = \sqrt{\frac{\mu_{\text{Earth}}}{r_b}}$$

$$\Delta V_A = \sqrt{v_{\text{pre},A}^2 + v_{\text{post},A}^2 - 2v_{\text{pre},A}v_{\text{post},A}\cos\Delta i_A}$$

$$\Delta V_B = \sqrt{v_{\text{pre},B}^2 + v_{\text{post},B}^2 - 2v_{\text{pre},B}v_{\text{post},B}\cos\Delta i_B}$$

$$\Delta V_{\text{tot}} = \Delta V_A + \Delta V_B$$

Part (b)

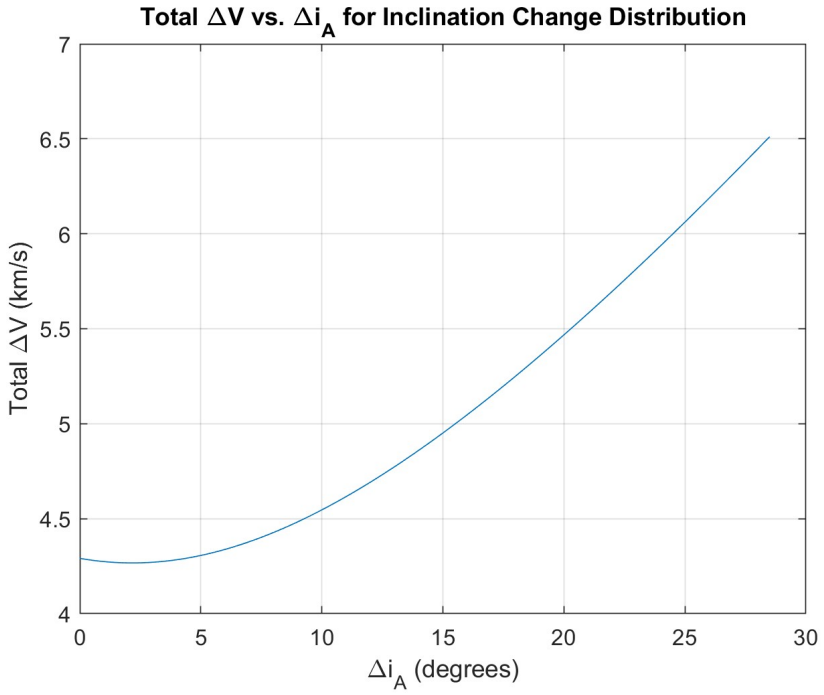

Figure 6: Delta V vs Delta i

Part (c)

Doing all the inclination change at the perigee kick, which corresponds to the last point on this curve, leads to the largest total Delta V needed. This is because the velocity vector is largest at the perigee, so it requires the most Delta V to level out and get rid of the entire inclination.

Part (d)

Using MATLAB's **fminbnd** function, the minimum Delta V is **4.267 km/s** and occurs at the optimum value of **2.169 deg**.

Part (e)

The direction of the apogee burn expressed in components in the equatorial plane and normal to the equatorial plane is given by:

$$\mathbf{d}_B = \begin{bmatrix} sind(\Delta i_B) \\ cosd(\Delta i_B) \end{bmatrix} = \begin{bmatrix} 0.4436 \\ 0.8962 \end{bmatrix}$$

Part (f)

The transfer time is the same as the transfer time for the Hohmann transfer to GEO, **5.259 hrs**.

Appendix 1

```
%% Space Mechanics Homework 7
% Tycho Bogdanowitsch
clc; clear;
% Constants
R_earth = 6378; % km
DU_earth = R_earth; % km
AU = 1.496e8; % km
mu_earth = 3.986e5; % km^3/s^2
mu_mars = 4.283e4; % km^3/s^2
mu_sun = 1.327e11; % km^3/s^2
R_sun = 6.963e5; % km

%% Problem 1
disp('Problem 1')
close all;
r_CB = [0;
    0;
    0;]; % DU
mu = 1; % DU/TU

% Part a
function [statedot] = derivative(t, state)
    % state vector is [rx ry rz vx vy vz]
    %
    % although required by form, input value t will go unused
       here
    statedot = zeros(6, 1);
    statedot(1:3) = state(4:6); % rdot = velocity

    mu = 1; % DU/TU
    r = state(1:3);
    accel_g = -(mu/(norm(r))^3)*r;

    statedot(4:6) = accel_g;
end % terminates MATLAB function

% Part b
r_0 = [4;0;0;]; % DU
v_0 = [0,0.5,0]; % DU/TU
y_initial = zeros(6,1);
y_initial(1:3) = r_0;
y_initial(4:6) = v_0;
```

```
tstart = 0; % Start time (TU)
t_orbit = 50.1;
tend = t_orbit; % End time (TU)
tint = 0.01; % Time step (TU)

options = odeset('RelTol', 1e-3, 'AbsTol', 1e-6);
[t_out, y_out] = ode45(@derivative, [tstart:tint:tend],
   y_initial, options);

x = y_out(:,1);
y = y_out(:,2);

figure(1);
plot(x, y, 'b');
hold on;
plot(0, 0, 'go', 'MarkerFaceColor', 'r', 'MarkerSize', 8); %
   Central body
plot(x(1), y(1), 'ro', 'MarkerFaceColor', 'g', 'MarkerSize', 6)
   ; % Start point
plot(x(end), y(end), 'mo', 'MarkerFaceColor', 'm', 'MarkerSize'
   , 6); % End point
xlabel('x (DU)');
ylabel('y (DU)');
title('1 Full Orbit in the XY Plane, ode45');
grid on;
axis equal;
legend('Orbit', 'Central Body', 'Start Point', 'End Point');
hold off;

% Part c
tstart = 0; % Start time (TU)
tend = 10*t_orbit; % End time (TU)
tint = 0.01; % Time step (TU)

options = odeset('RelTol', 1e-3, 'AbsTol', 1e-6);
[t_out, y_out] = ode45(@derivative, [tstart:tint:tend],
   y_initial, options);

x = y_out(:,1);
y = y_out(:,2);

figure(2);
plot(x, y, 'b');
hold on;
```

```
plot(0, 0, 'go', 'MarkerFaceColor', 'r', 'MarkerSize', 8); %
   Central body
plot(x(1), y(1), 'ro', 'MarkerFaceColor', 'g', 'MarkerSize', 6)
   ; % Start point
plot(x(end), y(end), 'mo', 'MarkerFaceColor', 'm', 'MarkerSize'
   , 6); % End point
xlabel('x (DU)');
ylabel('y (DU)');
title('10 Full Orbits in the XY Plane, ode45');
grid on;
axis equal;
legend('Orbit', 'Central Body', 'Start Point', 'End Point');
hold off;

% Part d
tstart = 0; % Start time (TU)
tend = 10*t_orbit; % End time (TU)
tint = 0.01; % Time step (TU)

options = odeset('RelTol', 1e-3, 'AbsTol', 1e-6);
[t_out, y_out] = ode113(@derivative, [tstart:tint:tend],
   y_initial, options);

x = y_out(:,1);
y = y_out(:,2);

figure(3);
plot(x, y, 'b');
hold on;
plot(0, 0, 'go', 'MarkerFaceColor', 'r', 'MarkerSize', 8); %
   Central body
plot(x(1), y(1), 'ro', 'MarkerFaceColor', 'g', 'MarkerSize', 6)
   ; % Start point
plot(x(end), y(end), 'mo', 'MarkerFaceColor', 'm', 'MarkerSize'
   , 6); % End point
xlabel('x (DU)');
ylabel('y (DU)');
title('10 Full Orbits in the XY Plane, ode113');
grid on;
axis equal;
legend('Orbit', 'Central Body', 'Start Point', 'End Point');
hold off;
```

```
%% Problem 2
disp('Problem 2');
close all; clear; clc;

% Physical Constants for Problem 7.2
% AA 279A Winter 2025
% Andrew K. Barrows

% Physical constants from https://ssd.jpl.nasa.gov/horizons/
% Valid for 0000h 1 March 2025 TDB
musun = 132712440041.93938; % [km^3/sec^2]
rsun0 = [-7.934185894828094E+05; ...
         -7.637131464755005E+05; ...
         +2.578283849613235E+04];
vsun0 = [+1.260654814883448E-02; ...
         -5.078486758688182E-03; ...
         -2.148008120450161E-04];

mumerc = 22031.86855; % [km^3/sec^2]
rmerc0 = [+2.594180273870160E+07; ...
          +3.740281863092749E+07; ...
          +6.926260996986236E+05];
vmerc0 = [-4.957858286093838E+01; ...
          +2.991861316756679E+01; ...
          +6.993763405403170E+00];

muvenus = 324858.592; % [km^3/sec^2]
rvenus0 = [-9.032880117596443E+07; ...
           +5.843626378287782E+07; ...
           +6.005029413579255E+06];
vvenus0 = [-1.944223863442384E+01; ...
           -2.939125952142425E+01; ...
           +7.187313483472693E-01];

muearth = 398600.435436; % [km^3/sec^2]
rearth0 = [-1.403457468394584E+08; ...
           +4.919962589267671E+07; ...
           +2.376299004800990E+04];
vearth0 = [-1.051430807577604E+01; ...
           -2.817482001172409E+01; ...
           +2.355436484258178E-04];

mumoon = 4902.800066; % [km^3/sec^2]
rmoon0 = [-1.399854526961165E+08; ...
          +4.915894474568726E+07; ...
```

```
          +2.172652641501278E+04];
vmoon0 = [-1.040859940096287E+01; ...
          -2.709796856217922E+01; ...
          +9.985242071252820E-02];

% Part a
function [statedot] = derivative_2(t, state)
    N = 5; % number of bodies
    set_len = 7; % elements in each set
    statedot = zeros(set_len*N, 1);
    mu = zeros(1,N);
    r = zeros(3,N);
    v = zeros(3,N);
    a = zeros(3,N);

    for i = 1:N % getting the mu, r, v for each body
        index = set_len*(i-1) + 1; % 1,8,...
        mu(i) = state(index);
        r(:,i) = state(index+1:index+3);
        v(:,i) = state(index+4:index+6);
    end

    for i=1:N
        for j = 1:N
            if j~=i
                r_ij = r(:,j) - r(:,i); % get vector from body
                   i to body j
                a(:,i) = a(:,i) + mu(j)* r_ij / norm(r_ij)^3; %
                    sum up all forces
            end
        end
    end

    for i=1:N
        index = set_len*(i-1) + 1; % 1,8,...
        statedot(index) = 0; % derivative of mu is zero
        statedot(index+1:index+3) = v(:,i); % rdot = v
        statedot(index+4:index+6) = a(:,i);
    end
end

r_0 = [4;0;0;]; % DU
v_0 = [0,0.5,0]; % DU/TU
y_initial = zeros(35,1);
y_initial(1) = musun;
```

```
y_initial(2:4) = rsun0;
y_initial(5:7) = vsun0;
y_initial(8) = mumerc;
y_initial(9:11) = rmerc0;
y_initial(12:14) = vmerc0;
y_initial(15) = muvenus;
y_initial(16:18) = rvenus0;
y_initial(19:21) = vvenus0;
y_initial(22) = muearth;
y_initial(23:25) = rearth0;
y_initial(26:28) = vearth0;
y_initial(29) = mumoon;
y_initial(30:32) = rmoon0;
y_initial(33:35) = vmoon0;


tstart = 0; % Start time (s)
tend = 365*86400; % end time (s)
tint = 100; % Time step (TU)

options = odeset('RelTol', 1e-6, 'AbsTol', 1e-9);
[t_out, y_out] = ode113(@derivative_2, [tstart:tint:tend],
   y_initial, options);

x_sun = y_out(:,2);
y_sun = y_out(:,3);
z_sun = y_out(:,4);
x_merc = y_out(:,9);
y_merc = y_out(:,10);
z_merc = y_out(:,11);
x_venus = y_out(:,16);
y_venus = y_out(:,17);
z_venus = y_out(:,18);
x_earth = y_out(:,23);
y_earth = y_out(:,24);
z_earth = y_out(:,25);
x_moon = y_out(:,30);
y_moon = y_out(:,31);
z_moon = y_out(:,32);

figure(1);
hold on;
plot(x_sun,   y_sun,   'y', 'LineWidth', 1.5);
plot(x_merc,  y_merc,  'k', 'LineWidth', 1.5);
plot(x_venus, y_venus, 'r', 'LineWidth', 1.5);
```

```
plot(x_earth, y_earth, 'b', 'LineWidth', 1.5);
plot(x_moon,  y_moon,  'g', 'LineWidth', 1.2);

plot(x_sun(1),   y_sun(1),   'yo', 'MarkerFaceColor', 'y', '
   MarkerSize', 6);
plot(x_merc(1),  y_merc(1),  'ko', 'MarkerFaceColor', 'k', '
   MarkerSize', 6);
plot(x_venus(1), y_venus(1), 'ro', 'MarkerFaceColor', 'r', '
   MarkerSize', 6);
plot(x_earth(1), y_earth(1), 'bo', 'MarkerFaceColor', 'b', '
   MarkerSize', 6);
plot(x_moon(1),  y_moon(1),  'go', 'MarkerFaceColor', 'g', '
   MarkerSize', 6);

xlabel('X Position (km)');
ylabel('Y Position (km)');
title('1 Year Orbits in the XY-Plane');
grid on;
axis equal;
legend('Sun', 'Mercury', 'Venus', 'Earth', 'Moon');
hold off;

figure(3);
hold on;
scatter3(x_sun,   y_sun, z_sun,  'y');
plot3(x_merc,  y_merc, z_merc, 'k');
plot3(x_venus, y_venus, z_venus, 'r');
plot3(x_earth, y_earth, z_earth, 'b');
plot3(x_moon,  y_moon, z_moon, 'g');


xlabel('X [AU]');
ylabel('Y [AU]');
zlabel('Z [AU]');
axis equal;
title('1 Year Orbits in 3D');
view(3); % Set the 3D perspective
grid on;

legend('Sun', 'Mercury', 'Venus', 'Earth', 'Moon');
hold off;

% Part c
earth_moon_x = x_moon - x_earth;
earth_moon_y = y_moon - y_earth;
```

```
figure(2);
hold on;
plot(earth_moon_x,earth_moon_y);
plot(earth_moon_x(1), earth_moon_y(1), 'go', 'MarkerFaceColor',
    'g', 'MarkerSize', 6); % Start point
plot(earth_moon_x(end), earth_moon_y(end), 'ro', '
   MarkerFaceColor', 'r', 'MarkerSize', 6); % End point

xlabel('X Position (km)');
ylabel('Y Position (km)');
title('Moon Orbits around Earth in 1 Year');
grid on;
axis equal;
legend('Moon Orbit','Start Point','End Point');
hold off;

%% Problem 3
close all; clear; clc;
disp('Problem 3');
format long;

mu_earth = 3.986e5; % km^3/s^2
r_earth = 6378; % km

% Part a
r_a = r_earth + 200; % km
r_b = r_earth + 35786; % km

delta_v_A = abs(sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))))-sqrt(
   mu_earth/r_a));
delta_v_B = abs(sqrt(mu_earth/r_b)-sqrt(2*mu_earth*((1/r_b)
   -(1/(r_a+r_b)))));
delta_v_tot = delta_v_A + delta_v_B;
t_trans = pi*sqrt((r_a+r_b)^3/(8*mu_earth));

fprintf('Delta V to GEO: %.3f km/s\n', delta_v_tot);
fprintf('Transfer Time to GEO: %.3f hrs\n', t_trans/3600);

% Part b
r_a = r_earth + 200; % km
r_b = 384400; % km

delta_v_A = abs(sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))))-sqrt(
   mu_earth/r_a));
```

```
delta_v_B = abs(sqrt(mu_earth/r_b)-sqrt(2*mu_earth*((1/r_b)
   -(1/(r_a+r_b)))));
delta_v_tot = delta_v_A + delta_v_B;
t_trans = pi*sqrt((r_a+r_b)^3/(8*mu_earth));

fprintf('Delta V to Moon Orbit: %.3f km/s\n', delta_v_tot);
fprintf('Transfer Time to Moon Orbit: %.3f hrs\n', t_trans
   /3600);

%% Problem 4
close all; clear; clc;
disp('Problem 4');
format long;

mu_earth = 3.986e5; % km^3/s^2
r_earth = 6378; % km

% Part a
r_a = r_earth + 200; % km
r_b = 1000000; % km
r_c = r_earth + 35786; % km

delta_v_A = abs(sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))))-sqrt(
   mu_earth/r_a));
delta_v_B = abs(sqrt(2*mu_earth*((1/r_b)-(1/(r_b+r_c))))-sqrt
   (2*mu_earth*((1/r_b)-(1/(r_a+r_b)))));
delta_v_C = abs(sqrt(mu_earth/r_c)-sqrt(2*mu_earth*((1/r_c)
   -(1/(r_b+r_c)))));
delta_v_tot = delta_v_A + delta_v_B + delta_v_C;
t_trans_1 = pi*sqrt((r_a+r_b)^3/(8*mu_earth));
t_trans_2 = pi*sqrt((r_b+r_c)^3/(8*mu_earth));
t_trans_tot = t_trans_1 + t_trans_2;

fprintf('Delta V to GEO: %.3f km/s\n', delta_v_tot);
fprintf('Transfer Time to GEO: %.3f hrs\n', t_trans_tot/3600);

% Part b
r_a = r_earth + 200; % km
r_b = 1000000; % km
r_c = 384400; % km

delta_v_A = abs(sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))))-sqrt(
   mu_earth/r_a));
delta_v_B = abs(sqrt(2*mu_earth*((1/r_b)-(1/(r_b+r_c))))-sqrt
   (2*mu_earth*((1/r_b)-(1/(r_a+r_b)))));
```

```
delta_v_C = abs(sqrt(mu_earth/r_c)-sqrt(2*mu_earth*((1/r_c)
   -(1/(r_b+r_c)))));
delta_v_tot = delta_v_A + delta_v_B + delta_v_C;
t_trans_1 = pi*sqrt((r_a+r_b)^3/(8*mu_earth));
t_trans_2 = pi*sqrt((r_b+r_c)^3/(8*mu_earth));
t_trans_tot = t_trans_1 + t_trans_2;

fprintf('Delta V to Moon Orbit: %.3f km/s\n', delta_v_tot);
fprintf('Transfer Time to Moon Orbit: %.3f hrs\n', t_trans_tot
   /3600);

%% Problem 5
disp('Problem 5');
close all; clear; clc;

% Part a
function delta_v_tot = combined(delta_i_A)
    mu_earth = 3.986e5; % km^3/s^2
    r_earth = 6378; % km

    i_initial = 28.5; % deg
    delta_i_B = i_initial - delta_i_A; % deg
    r_a = r_earth + 200; % km
    r_b = r_earth + 35786; % km
    v_pre_A = sqrt(mu_earth/r_a);
    v_post_A = sqrt(2*mu_earth*((1/r_a)-(1/(r_a+r_b))));
    v_pre_B = sqrt(2*mu_earth*((1/r_b)-(1/(r_a+r_b))));
    v_post_B = sqrt(mu_earth/r_b);
    delta_v_A = sqrt((v_pre_A)^2+(v_post_A)^2-2*v_pre_A*
       v_post_A*cosd(delta_i_A));
    delta_v_B = sqrt((v_pre_B)^2+(v_post_B)^2-2*v_pre_B*
       v_post_B*cosd(delta_i_B));
    delta_v_tot = delta_v_A + delta_v_B;
end

% Part b
delta_i_A_range = linspace(0,28.5,100);
delta_v_tot_range = zeros(size(delta_i_A_range));

for i = 1:length(delta_i_A_range)
    delta_v_tot_range(i)=combined(delta_i_A_range(i));
end

figure(1);
plot(delta_i_A_range, delta_v_tot_range);
```

```
xlabel('\Deltai_A (degrees)');
ylabel('Total \DeltaV (km/s)');
title('Total \DeltaV vs. \Deltai_A for Inclination Change
   Distribution');
grid on;

% Part d
min_delta_i_A = fminbnd(@combined,0,28.5);
min_delta_v = combined(min_delta_i_A);
fprintf('Optminum value of Delta i_A: %.3f deg\n',
   min_delta_i_A);
fprintf('Minimum total Delta V: %.3f km/s\n', min_delta_v);

% Part e
i_initial = 28.5;
delta_i_B = i_initial-min_delta_i_A;
dir_B = [cosd(delta_i_B),sind(delta_i_B)]
%disp(dir_B);

% Part f
mu_earth = 3.986e5; % km^3/s^2
r_earth = 6378; % km

r_a = r_earth + 200; % km
r_b = r_earth + 35786; % km

t_trans = pi*sqrt((r_a+r_b)^3/(8*mu_earth));
fprintf('Transfer Time to GEO: %.3f hrs\n', t_trans/3600);
```