A diagram of earth orbit

Description automatically generated

Mars Arrival

Earth Departure

Figure Transfer Between Earth and Mars

% This is the code for HW3 - Problem 3

% By Nicholas Luis (PSU ID 930841391)

clear; clc; close all;

% Constants

MU = 1.327\*(10^11); % km^3 / s^2

PI = 3.141592654;

AU = 149.6\*(10^6); % km

% Earth Orbit

rE = 1\*AU;

aE = rE; % True for circular orbits

vE = getVelo(rE, aE);

% Mars Orbit

rM = 1.524\*AU;

aM = rM; % True for circular orbits

vM = getVelo(rM, aM);

% Transfer Orbit

TOF = 1.58112\*(10^6); % Time Of Flight in seconds

delta\_f = 135.8624; % Degrees

a\_T = get\_aMin(rE, rM, delta\_f);

p\_T = getPT(a\_T, rE, rM, delta\_f);

e\_T = (p\_T / rE) - 1;

vET = getVelo(rE, a\_T); % Velocity on the TO at inersection w/ Earth orbit

vMT = getVelo(rM, a\_T); % Velocity on the TO at inersection w/ Mars orbit

% Delta V

delta\_VE = getDeltaV(vE, vET, 0);

delta\_VM = getDeltaV(vM, vMT, 0);

%% Plotting

fvec = 0:0.1:360; % Creates a list of true anomaly values to iterate through (in degrees)

% X Y values. Note that (0,0) will be located on the Sun

xvec1 = rE.\*cosd(fvec); % List of the x coordinates of the departure orbit

yvec1 = rE.\*sind(fvec); % List of the y coordinates of the departure orbit

rTvec = (a\_T\*(1-e\_T^2)) ./ (1+e\_T\*cosd(fvec)); % List of the radii of the transfer orbit

xvecT = rTvec.\*cosd(fvec); % List of the x coordinates of the transfer orbit

yvecT = rTvec.\*sind(fvec); % List of the y coordinates of the transfer orbit

xvec2 = rM.\*cosd(fvec-30); % List of the x coordinates of the arrival orbit

yvec2 = rM.\*sind(fvec-30); % List of the y coordinates of the arrival orbit

figure(1)

hold on

plot(xvec1, yvec1, LineWidth=2)

plot(xvec2, yvec2, LineWidth=2)

plot(xvecT(1:1801), yvecT(1:1801), ':', LineWidth=2) % Only plotting half of the transfer orbit

title('Earth-Mars Transfer')

xlabel("x (km)")

ylabel("y (km)")

legend('Earth Orbit', 'Mars Orbit', 'Transfer Orbit')

hold off

exportgraphics(gca,"HW2\_Problem3\_Figure1.jpg");

%% Functions

function P\_output = getPT(amin, r1\_in, r2\_in, df)

% This funcion gets the semilatus rectum of a transfer orbit given r1,

% r2, and the change in f

k = r1\_in\*r2\_in\*(1-cosd(df));

m = r1\_in\*r2\_in\*(1+cosd(df));

l = r1\_in + r2\_in;

P\_output = (k\*m - 2\*amin\*k\*l) / (4\*amin\*m - 2\*amin\*l\*l);

end

function velo = getVelo(r\_input, a\_input)

% This function gets the velocity using energy given MU, r, and a

MU = 1.327\*(10^11); % km^3 / s^2

velo = sqrt(2 \* MU \* ( (1/r\_input) - (1 / (2\*a\_input)) ) );

end

function dvelo = getDeltaV(vi, vf, gamma)

dvelo = sqrt( vf^2 + vi^2 - 2\*vi\*vf\*cosd(gamma) );

end

function aMinT = get\_aMin(r1\_in, r2\_in, df)

% This function gets the semimajor axis of minimum energy transfer

% orbit given r1, r2, and the change in f

sqrtTerm = sqrt( r1\_in^2 + r2\_in^2 - 2\*r1\_in\*r2\_in\*cosd(df) ) ;

aMinT = 0.25 \* (r1\_in + r2\_in + sqrtTerm);

end

function fOut = getF(e, p, r)

fOut = acosd((1/e)\*((p/r)-1));

end

function gammaOut = getGamma(e, f, df)

gammaOut = atand((e\*sind(f)) / (1+e\*cosd(f)));

if (df > 180)

gammaOut = abs(gammaOut);

else

gammaOut = - abs(gammaOut);

end

end