Homework 4

Tycho Bogdanowitsch
AA 279A - Space Mechanics

February 12, 2025

Problem 1.

Part (a)

True, orbital elements fully define an orbit in space.

Part (b)

False, in circular, equatorial orbits, the line of apsis and nodes are undefined, meaning that the argument of periapsis is not defined. Instead the true longitude is used, which is the distance from the vernal equinox.

Part (c)

True, in a solar day the Earth rotates an extra amount in order to point towards the sun since it is simultaneously rotating about the sun and its own axis.

Part (d)

True, a geostationary orbit by definition has a 0 degree inclination and by design stays above a fixed point on Earth.

Part (e)

True, the inclination of an orbit determines how the orbital plane is inclined relative to the equatorial plane, which when projected on to Earth corresponds to the maximum latitude of the satellite's ground track.

Part (f)

False, the period of an orbit can be determined from its ground track by looking at the change in longitude and then accounting for Earth's spin. Then, the semi-major axis can be determined from the period.

Problem 2.

Part (a)

```
function MJD = Cal2MJD(UT1)
    M = UT1(1);
    D = UT1(2);
    Y = UT1(3);
    if M<=2
        y = Y-1;
        m = M+12;
    else
        y = Y;
        m = M;
    end

    if (y<=1582 && m<=8 && D<= 4)
        B = -2 + (y+4716) / 4 - 1179;
    else
        B = y/400 - y/100 + y/4;
    end

    MJD = 365*y - 679004 + floor(B) + floor(30.6001 * (m+1))+D;
end
```

Using the above function, the MJD of February 1, 2024 12:00h UT1 is **60341.5 days**.

Part (b)

```
function GMST = MJD2GMST(MJD)
    d = MJD - 51544.5;
    GMST = 280.4606 + 360.9856476*d;
end
```

Using the above function, the GMST at this same epoch is **5.432 rad**.

Part (c)

```
function Rot = CRF2TRFRot(GMST)
    Rot = [cos(GMST) sin(GMST) 0;
        -sin(GMST) cos(GMST) 0;
        0 0 1];
end
```

Using the above function, the rotation matrix from CRF to TRF at this same epoch is:

$$R_{CRF,TRF} = \begin{bmatrix} 0.6587 & -0.7524 & 0 \\ 0.7524 & 0.6587 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

Problem 3.

Part (a)

The ISS orbit is prograde because inclination is less than 90 degrees, elliptical because eccentricity is not equal to 0, and inclined because inclination is not equal to 0.

Part (b)

```
function E = NewtonRaphson(M,e,epsilon)
    E = M;

    while true
        fE = E - e*sin(E) - M;
        fprimeE = 1 - e*cos(E);
        delta = -fE/fprimeE;
        E_next = E+delta;
        if abs(delta)<epsilon
            break;
        end

        E = E_next;
    end
end
```

```
function [x,y] = OE2Perifocal(a,E,e)
    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
    r = a*(1-e^2)/(1+e*cos(nu));
    x = r*cos(nu);
    y = r*sin(nu);
end
```

Using the same Simulink model as last homework, with adding an initial mean anomaly, the ISS orbit in perifocal coordinates over 1 day can be plotted. The two functions used for this part of the simulation are shown above. The initial mean anomaly was calculated by propagating the given mean anomaly at t = 30 days to t = 100 days, using the following relations:

$$T = 2\pi\sqrt{\frac{a^3}{\mu_{\text{earth}}}} \quad \text{(seconds)}$$

$$n = \frac{2\pi}{T} \quad \text{(rad/s)}$$

$$\Delta t_{\text{30-100}} = 70 \times 24 \times 3600 \quad \text{(days to seconds)}$$

$$M_0 = \text{mod}(M + n\Delta t_{\text{30-100}}, 2\pi) \quad \text{(rad)}$$

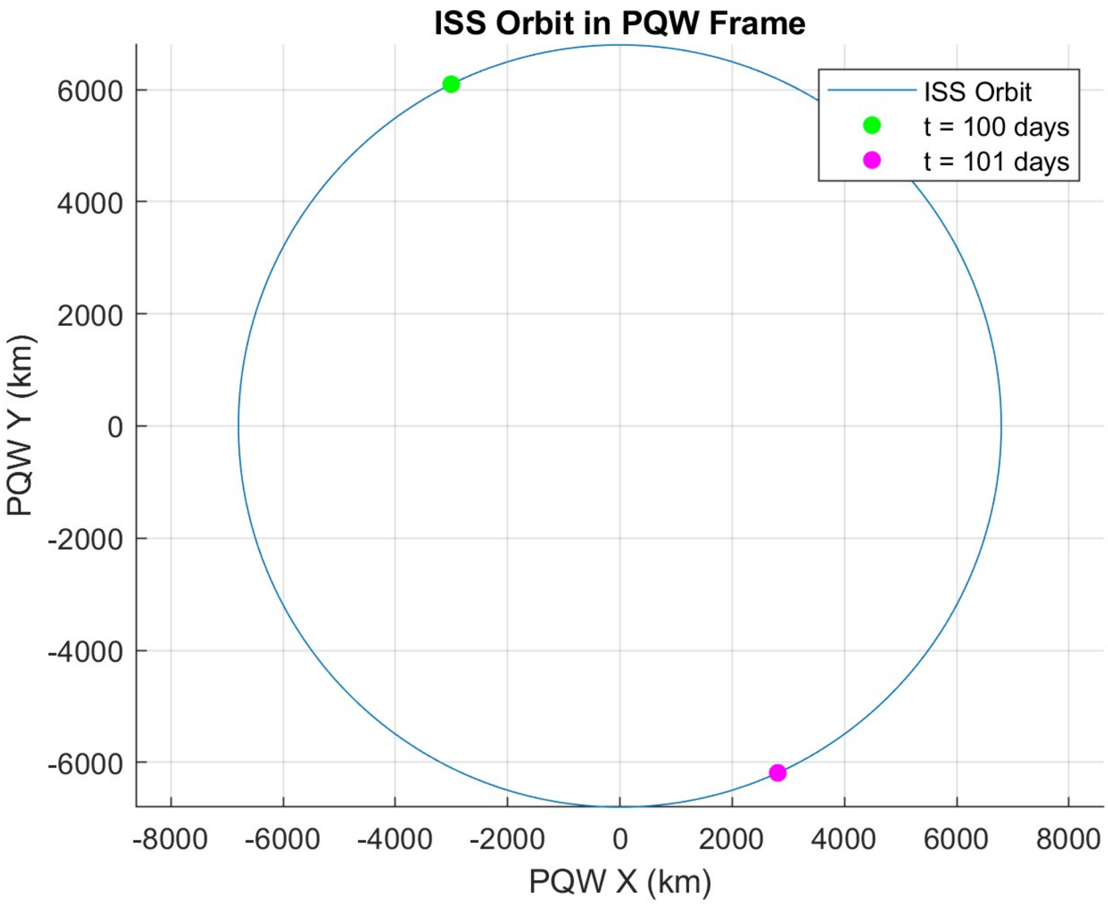

Figure 1: ISS Orbit in PQW Frame

Part (c)

```
function r_ECI = OE2ECI(a,E,e,i,RAAN,omega)
    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
    r = a*(1-e^2)/(1+e*cos(nu));
    r_PQW = [r * cos(nu); r * sin(nu); 0];

    R_PQR_IJK = [cos(RAAN)*cos(omega)-sin(RAAN)*cos(i)*sin(
       omega),...
        -cos(RAAN)*sin(omega)-sin(RAAN)*cos(i)*cos(omega),...
        sin(RAAN)*sin(i);
        sin(RAAN)*cos(omega)+cos(RAAN)*cos(i)*sin(omega),...
        -sin(RAAN)*sin(omega)+cos(RAAN)*cos(i)*cos(omega),...
        -cos(RAAN)*sin(i);
        sin(i)*sin(omega), sin(i)*cos(omega), cos(i)];

    r_ECI = R_PQR_IJK*r_PQW;
```

```
end
```

Now, using the above function in the extended Simulink model, the orbit in the ECI frame over the same timeframe can be plotted.

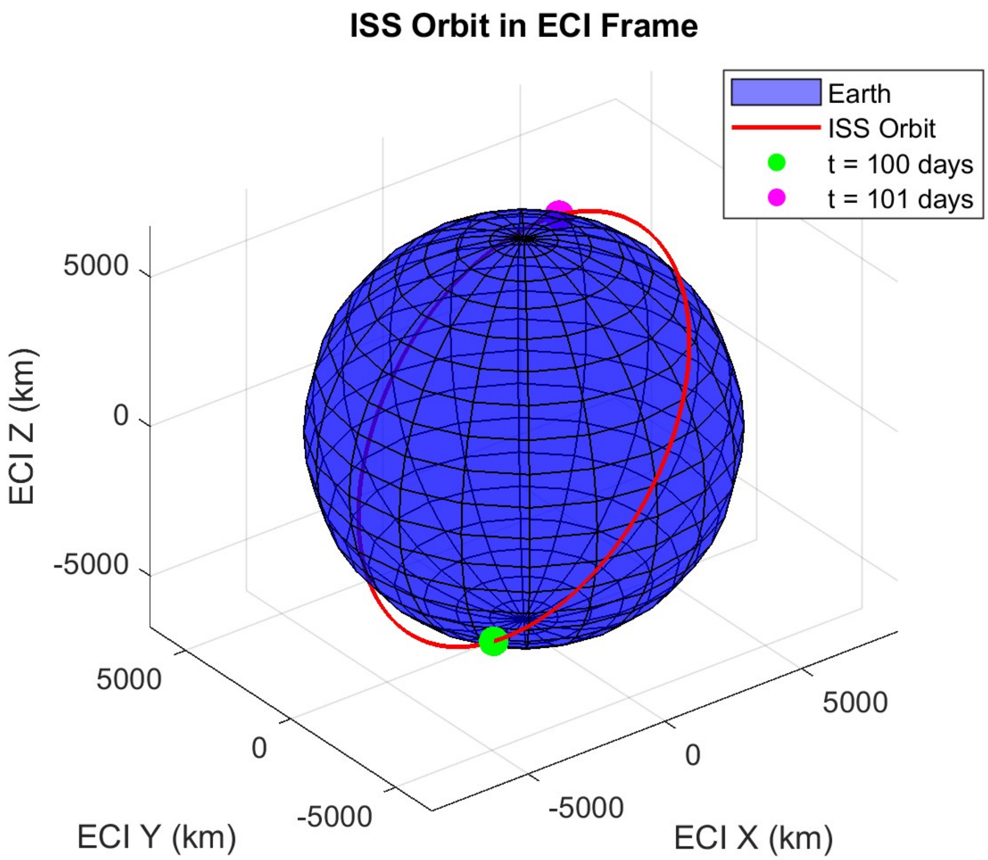

Figure 2: ISS Orbit in ECI Frame

Part (d)

```
function r_ECEF = ECEF(a,E,e,i,RAAN,omega,UT1,t)
    M = UT1(1);
    D = UT1(2);
    Y = UT1(3);
    t_days = t/86400;
    D = D+t_days;
    if M<=2
        y = Y-1;
        m = M+12;
    else
        y = Y;
        m = M;
    end

    if (y<=1582 && m<=8 && D<= 4)
        B = -2 + (y+4716) / 4 - 1179;
    else
        B = y/400 - y/100 + y/4;
    end

    MJD = 365*y - 679004 + floor(B) + floor(30.6001 * (m+1))+D;

    d = MJD - 51544.5;
    GMST = 280.4606 + 360.9856476*d;

    GMST_rad_wrapped = wrapTo2Pi(GMST*pi/180);

    R_ECI_ECEF = [cos(GMST_rad_wrapped) sin(GMST_rad_wrapped)
       0;
        -sin(GMST_rad_wrapped) cos(GMST_rad_wrapped) 0;
        0 0 1];

    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
    r = a*(1-e^2)/(1+e*cos(nu));
    r_PQW = [r * cos(nu); r * sin(nu); 0];

    R_PQR_IJK = [cos(RAAN)*cos(omega)-sin(RAAN)*cos(i)*sin(
       omega),...
        -cos(RAAN)*sin(omega)-sin(RAAN)*cos(i)*cos(omega),...
        sin(RAAN)*sin(i);
        sin(RAAN)*cos(omega)+cos(RAAN)*cos(i)*sin(omega),...
        -sin(RAAN)*sin(omega)+cos(RAAN)*cos(i)*cos(omega),...
        -cos(RAAN)*sin(i);
```

```
        sin(i)*sin(omega), sin(i)*cos(omega), cos(i)];

    r_ECI = R_PQR_IJK*r_PQW;

    r_ECEF = R_ECI_ECEF*r_ECI;
end
```

Finally, the above function is added to complete the Simulink model, shown below. Using this extended Simulink model, the orbit in ECEF coordinates can be plotted. Note that the above function takes in the UT1 start time and the current time in seconds to calculate the appropriate rotation matrix.

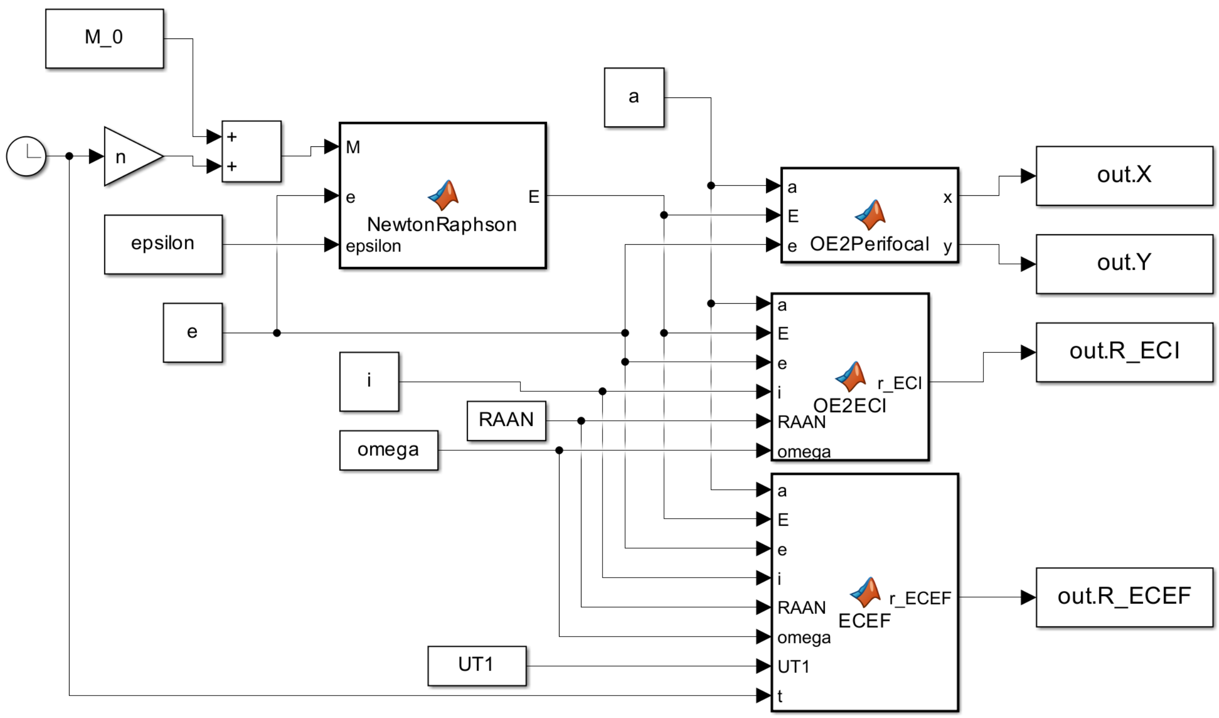

Figure 3: Full ISS Orbit Simulink Model

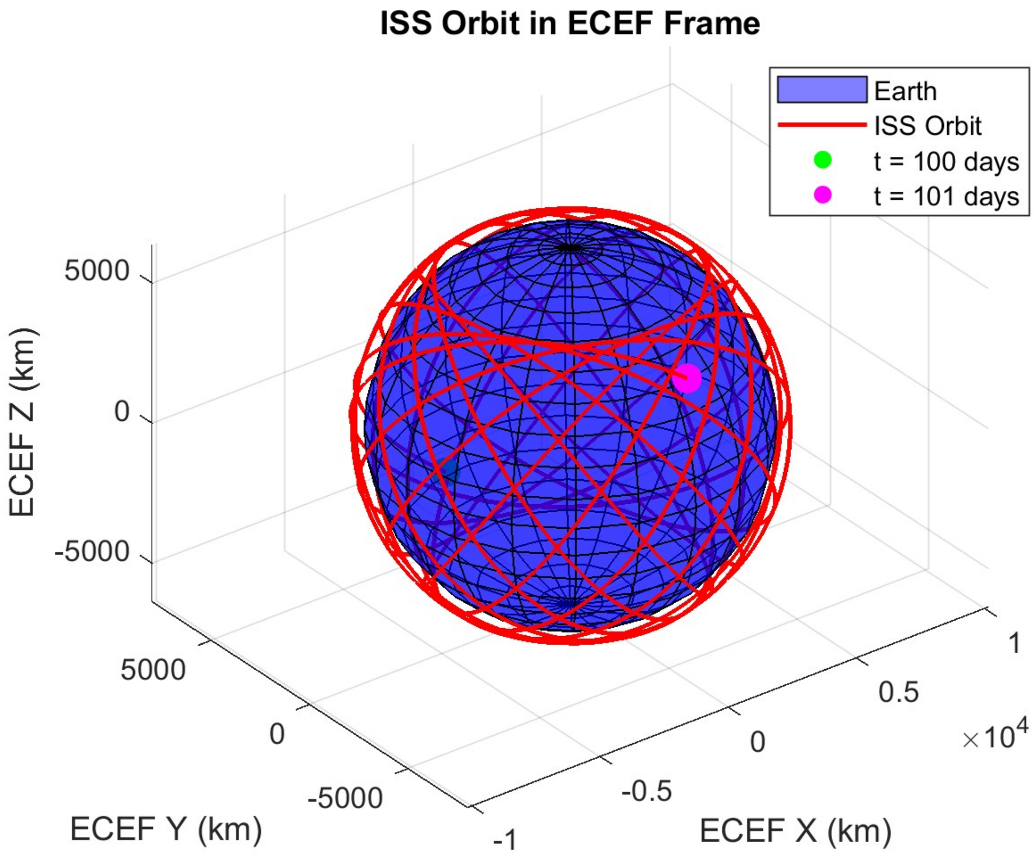

Figure 4: ISS Orbit in ECEF Frame

Problem 4.

Part (a)

```
function [lambda,psi] = ECEF2Geocentric(ECEF)
    r_x = ECEF(1);
    r_y = ECEF(2);
    r_z = ECEF(3);

    lambda = atan2d(r_y,r_x);
    psi = asind(r_z/sqrt(r_x^2+r_y^2+r_z^2));
end
```

Using the function shown above, the ECEF position of the spacecraft can be converted to Geocentric coordinates. The result is: $\phi = \mathbf{36.091}°, \lambda = -\mathbf{115.183}°$. This means that in Geocentric coordinates, the spacecraft is passing over the stadium.

Part (b)

```
function [lambda_prime,psi_prime] = ECEF2Geodetic(ECEF)
    r_x = ECEF(1);
    r_y = ECEF(2);
    r_z = ECEF(3);

    R_earth = 6378.1; % km
    e_earth = 0.0818;
    epsilon = 0.001; % deg

    r_xy = sqrt(r_x^2+r_y^2);

    lambda_prime = atan2d(r_y,r_x);

    psi_prime = asind(r_z/sqrt(r_x^2+r_y^2+r_z^2));

    while true
        N = R_earth/(sqrt(1-(e_earth^2)*(sind(psi_prime))^2));

        psi_prime_next = atan2d(r_z+N*(e_earth^2)*sind(
           psi_prime),r_xy);
        delta = psi_prime_next - psi_prime;

        if abs(delta)<epsilon
            psi_prime = psi_prime_next;
            break;
        end

        psi_prime = psi_prime_next;
    end
end
```

Similarly, using the function above, the ECEF position of the spacecraft can be converted to Geodetic coordinates. The result is: $\phi = \mathbf{36.274}°, \lambda = -\mathbf{115.183}°$. This means that when accounting for the oblateness of the Earth, the spacecraft is not passing over the stadium. Rather, it is **20.333 km away in the Northward direction**.

Appendix 1

```
%% Space Mechanics Homework 4
% Tycho Bogdanowitsch
clc; clear;
% Constants
R_earth = 6371; % 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 2
fprintf(['\nProblem 2\n']);
% Part a
UT1 = [2 1.5 2024];

MJD = Cal2MJD(UT1);

fprintf('The MJD is: %.3f days\n', MJD);

function MJD = Cal2MJD(UT1)
    M = UT1(1);
    D = UT1(2);
    Y = UT1(3);
    if M<=2
        y = Y-1;
        m = M+12;
    else
        y = Y;
        m = M;
    end

    if (y<=1582 && m<=8 && D<= 4)
        B = -2 + (y+4716) / 4 - 1179;
    else
        B = y/400 - y/100 + y/4;
    end

    MJD = 365*y - 679004 + floor(B) + floor(30.6001 * (m+1))+D;
end
```

```
% Part b
GMST = MJD2GMST(MJD);
GMST_rad_wrapped = wrapTo2Pi(GMST*pi/180);

fprintf('The GMST is: %.3f rad\n', GMST_rad_wrapped);

function GMST = MJD2GMST(MJD)
    d = MJD - 51544.5;
    GMST = 280.4606 + 360.9856476*d;
end

% Part c
Rot = CRF2TRFRot(GMST_rad_wrapped);
fprintf('The CRF to TRF rotation matrix is: \n');
disp(Rot)

function Rot = CRF2TRFRot(GMST)
    Rot = [cos(GMST) sin(GMST) 0;
        -sin(GMST) cos(GMST) 0;
        0 0 1];
end

%% Problem 3
close all;
fprintf(['\nProblem 3\n']);

% Part b
function E = NewtonRaphson(M,e,epsilon)
    E = M;

    while true
        fE = E - e*sin(E) - M;
        fprimeE = 1 - e*cos(E);
        delta = -fE/fprimeE;
        E_next = E+delta;
        if abs(delta)<epsilon
            break;
        end

        E = E_next;
    end
end

function [x,y] = OE2Perifocal(a,E,e)
    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
```

```
    r = a*(1-e^2)/(1+e*cos(nu));
    x = r*cos(nu);
    y = r*sin(nu);
end

a = 6796.3; % km
RAAN = deg2rad(257.7630); % deg --> rad
e = 0.00023;
omega = deg2rad(195.9983); % deg --> rad
i = deg2rad(51.6412); % deg --> rad
M = deg2rad(240.3224); % deg --> rad
UT1 = [2 1 2024];

epsilon = 1e-10;

T = 2*pi*sqrt(a^3/mu_earth); % seconds
n = 2*pi/T; % rad/s

delta_t_30_100 = 70*24*3600; % days --> seconds

M_0 = wrapTo2Pi(M + n*delta_t_30_100); % rad

stop_time = 1*24*3600; % days --> seconds

x = out.X;
y = out.Y;

x_start = x(1);
y_start = y(1);

x_end = x(end);
y_end = y(end);

figure(1);
hold on;
plot(x, y,'DisplayName','ISS Orbit');
axis equal;
xlabel('PQW X (km)');
ylabel('PQW Y (km)');
title('ISS Orbit in PQW Frame');
scatter(x_start,y_start,'g','filled','DisplayName', 't = 100
   days');
scatter(x_end,y_end,'m','filled','DisplayName', 't = 101 days')
   ;
grid on;
```

```
legend show;
hold off;

% Part c
function r_ECI = OE2ECI(a,E,e,i,RAAN,omega)
    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
    r = a*(1-e^2)/(1+e*cos(nu));
    r_PQW = [r * cos(nu); r * sin(nu); 0];

    R_PQR_IJK = [cos(RAAN)*cos(omega)-sin(RAAN)*cos(i)*sin(
       omega),...
        -cos(RAAN)*sin(omega)-sin(RAAN)*cos(i)*cos(omega),...
        sin(RAAN)*sin(i);
        sin(RAAN)*cos(omega)+cos(RAAN)*cos(i)*sin(omega),...
        -sin(RAAN)*sin(omega)+cos(RAAN)*cos(i)*cos(omega),...
        -cos(RAAN)*sin(i);
        sin(i)*sin(omega), sin(i)*cos(omega), cos(i)];

    r_ECI = R_PQR_IJK*r_PQW;
end

r_ECI = out.R_ECI;

[xE,yE,zE] = ellipsoid(0,0,0,R_earth,R_earth,R_earth,20);

figure(2)
hold on;

surface(xE, yE, zE, 'FaceColor', 'blue', 'EdgeColor', 'black',
   'FaceAlpha', 0.5,'DisplayName', 'Earth');

plot3(r_ECI(1,:), r_ECI(2,:), r_ECI(3,:),'r', 'LineWidth', 1.5,
   'DisplayName', 'ISS Orbit');

scatter3(r_ECI(1,1), r_ECI(2,1), r_ECI(3,1), 100, 'g', 'filled'
   ,'DisplayName', 't = 100 days');
scatter3(r_ECI(1,end), r_ECI(2,end), r_ECI(3,end), 100, 'm', '
   filled','DisplayName', 't = 101 days');

axis equal;
grid on;
legend show;
xlabel('ECI X (km)');
ylabel('ECI Y (km)');
zlabel('ECI Z (km)');
```

```
title('ISS Orbit in ECI Frame');
view(3);
hold off;

% Part d


function r_ECEF = ECEF(a,E,e,i,RAAN,omega,UT1,t)
    M = UT1(1);
    D = UT1(2);
    Y = UT1(3);
    t_days = t/86400;
    D = D+t_days;
    if M<=2
        y = Y-1;
        m = M+12;
    else
        y = Y;
        m = M;
    end

    if (y<=1582 && m<=8 && D<= 4)
        B = -2 + (y+4716) / 4 - 1179;
    else
        B = y/400 - y/100 + y/4;
    end

    MJD = 365*y - 679004 + floor(B) + floor(30.6001 * (m+1))+D;

    d = MJD - 51544.5;
    GMST = 280.4606 + 360.9856476*d;

    GMST_rad_wrapped = wrapTo2Pi(GMST*pi/180);

    R_ECI_ECEF = [cos(GMST_rad_wrapped) sin(GMST_rad_wrapped)
       0;
        -sin(GMST_rad_wrapped) cos(GMST_rad_wrapped) 0;
        0 0 1];

    nu = 2*atan2(sqrt(1+e)*tan(E/2),sqrt(1-e));
    r = a*(1-e^2)/(1+e*cos(nu));
    r_PQW = [r * cos(nu); r * sin(nu); 0];

    R_PQR_IJK = [cos(RAAN)*cos(omega)-sin(RAAN)*cos(i)*sin(
       omega),...
```

```
        -cos(RAAN)*sin(omega)-sin(RAAN)*cos(i)*cos(omega),...
        sin(RAAN)*sin(i);
        sin(RAAN)*cos(omega)+cos(RAAN)*cos(i)*sin(omega),...
        -sin(RAAN)*sin(omega)+cos(RAAN)*cos(i)*cos(omega),...
        -cos(RAAN)*sin(i);
        sin(i)*sin(omega), sin(i)*cos(omega), cos(i)];

    r_ECI = R_PQR_IJK*r_PQW;

    r_ECEF = R_ECI_ECEF*r_ECI;
end

r_ECEF = out.R_ECEF;

figure(3)
hold on;

surface(xE, yE, zE, 'FaceColor', 'blue', 'EdgeColor', 'black',
   'FaceAlpha', 0.5,'DisplayName', 'Earth');

plot3(r_ECEF(1,:), r_ECEF(2,:), r_ECEF(3,:),'r', 'LineWidth',
   1.5,'DisplayName', 'ISS Orbit');

scatter3(r_ECEF(1,1), r_ECEF(2,1), r_ECEF(3,1), 100, 'g', '
   filled','DisplayName', 't = 100 days');
scatter3(r_ECEF(1,end), r_ECEF(2,end), r_ECEF(3,end), 100, 'm',
    'filled','DisplayName', 't = 101 days');

axis equal;
grid on;
legend show;
xlabel('ECEF X (km)');
ylabel('ECEF Y (km)');
zlabel('ECEF Z (km)');
title('ISS Orbit in ECEF Frame');
view(3);
hold off;

%% Problem 4
fprintf(['\nProblem 4\n']);
R_earth = 6378.1; % km
e_earth = 0.0818;

function [lambda,psi] = ECEF2Geocentric(ECEF)
    r_x = ECEF(1);
```

```
    r_y = ECEF(2);
    r_z = ECEF(3);

    lambda = atan2d(r_y,r_x);
    psi = asind(r_z/sqrt(r_x^2+r_y^2+r_z^2));
end

r_ECEF = [-2195.7 -4669.6 3761.5];

[lambda,psi] = ECEF2Geocentric(r_ECEF);

fprintf('The Geocentric latitude is: %.3f deg\n', psi);
fprintf('The Geocentric longitude is: %.3f deg\n', lambda);

fprintf('In Geocentric coordinates, the spacecraft is passing
   over the stadium\n');

function [lambda_prime,psi_prime] = ECEF2Geodetic(ECEF)
    r_x = ECEF(1);
    r_y = ECEF(2);
    r_z = ECEF(3);

    R_earth = 6378.1; % km
    e_earth = 0.0818;
    epsilon = 0.001; % deg

    r_xy = sqrt(r_x^2+r_y^2);

    lambda_prime = atan2d(r_y,r_x);

    psi_prime = asind(r_z/sqrt(r_x^2+r_y^2+r_z^2));

    while true
        N = R_earth/(sqrt(1-(e_earth^2)*(sind(psi_prime))^2));

        psi_prime_next = atan2d(r_z+N*(e_earth^2)*sind(
           psi_prime),r_xy);
        delta = psi_prime_next - psi_prime;

        if abs(delta)<epsilon
            psi_prime = psi_prime_next;
            break;
        end

        psi_prime = psi_prime_next;
```

```
    end
end

[lambda_prime,psi_prime] = ECEF2Geodetic(r_ECEF);

fprintf('The Geodetic latitude is: %.3f deg\n', psi_prime);
fprintf('The Geodetic longitude is: %.3f deg\n', lambda_prime);


arc_dist = 2*pi*R_earth*((psi_prime-36.0909)/360);
fprintf('In Geodetic coordinates, the spacecraft is %.3f km
   away in the Northward direction\n', arc_dist);

%geoc2geod(psi,R_earth*10e3)
```