**Nicholas Luis (PSU ID 930841391)**

**AERSP 450 HW3**

Below is the data I collected about the spacecraft’s velocity for each case, which was calculated using Gibb’s method and was compared to a given reference speed.







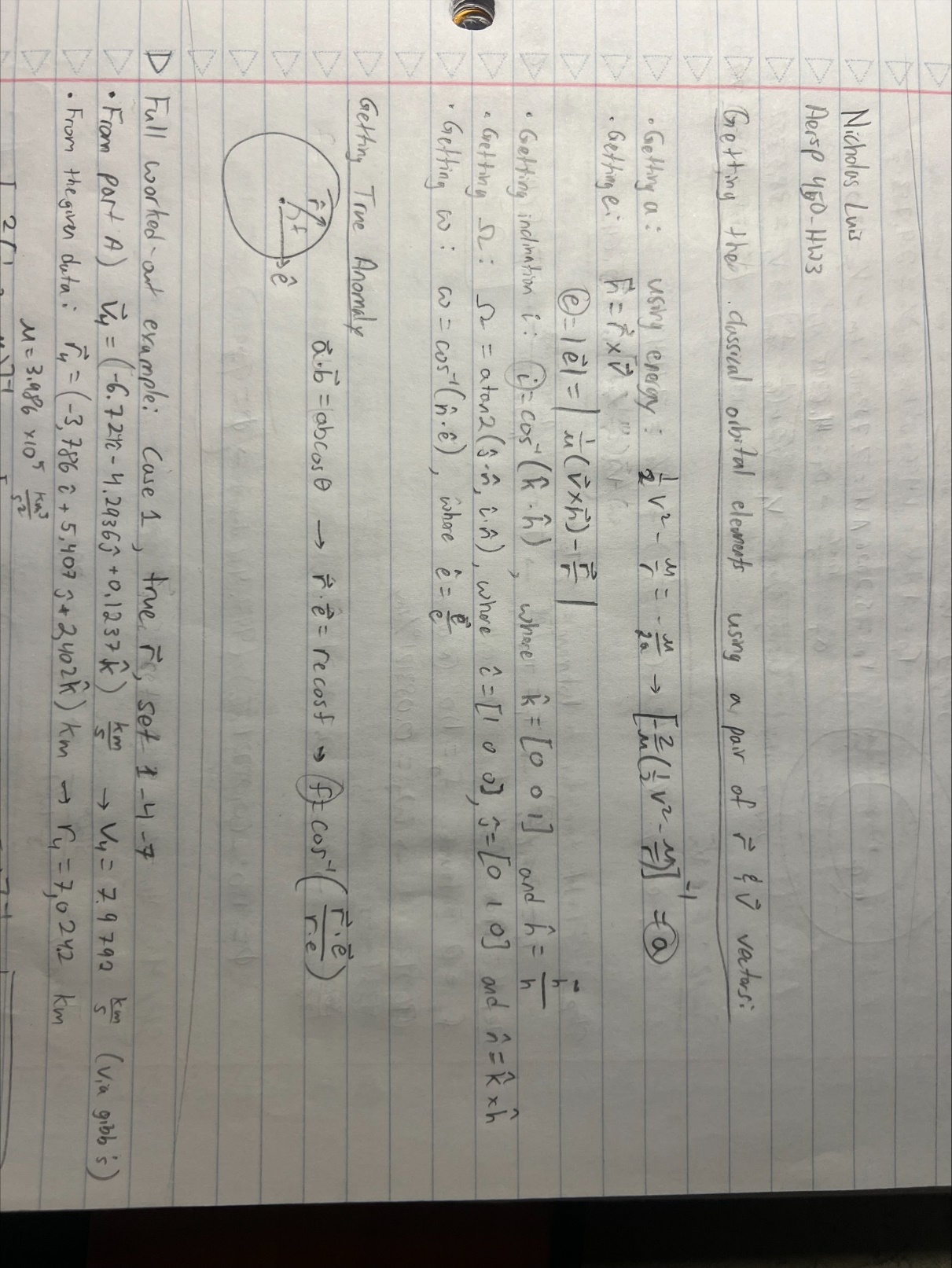
Below is the data I collected regarding the orbital elements and the change in true anomaly between measurements locations.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Obs | a (km) | e | I (deg) | Ω (deg) | ω (deg) | Δf (deg) |
| Case 1 | Perfect | 3,4,5 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 1.00 |
| 2,4,6 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 2.00 |
| 1,4,7 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 3.00 |
| Corrupted | 3,4,5 | 7946.80 | 0.125 | 20.00 | 45.00 | 66.05 | 1.0001 |
| 2,4,6 | 7986.78 | 0.129 | 20.00 | 45.00 | 66.76 | 2.0001 |
| 1,4,7 | 7980.31 | 0.128 | 20.00 | 45.00 | 66.67 | 3.0001 |

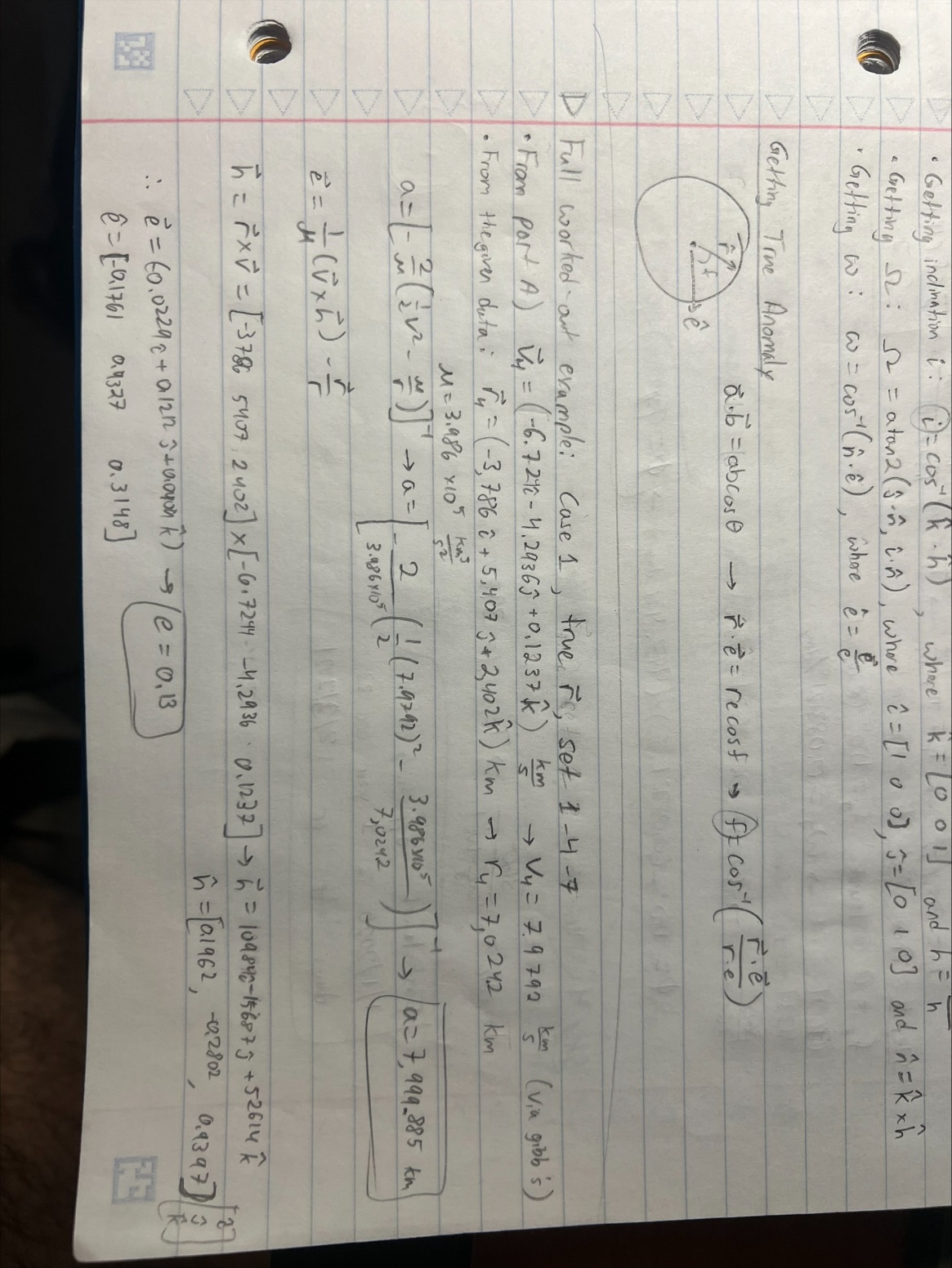
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Obs | a (km) | e | I (deg) | Ω (deg) | ω (deg) | Δf (deg) |
| Case 2 | Perfect | 3,4,5 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 10.00 |
| 2,4,6 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 20.00 |
| 1,4,7 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 30.00 |
| Corrupted | 3,4,5 | 7998.62 | 0.129 | 20.00 | 45.00 | 66.96 | 10.0001 |
| 2,4,6 | 8000.15 | 0.130 | 20.00 | 45.00 | 67.01 | 20.0000 |
| 1,4,7 | 8000.01 | 0.130 | 20.00 | 45.00 | 67.00 | 30.0001 |

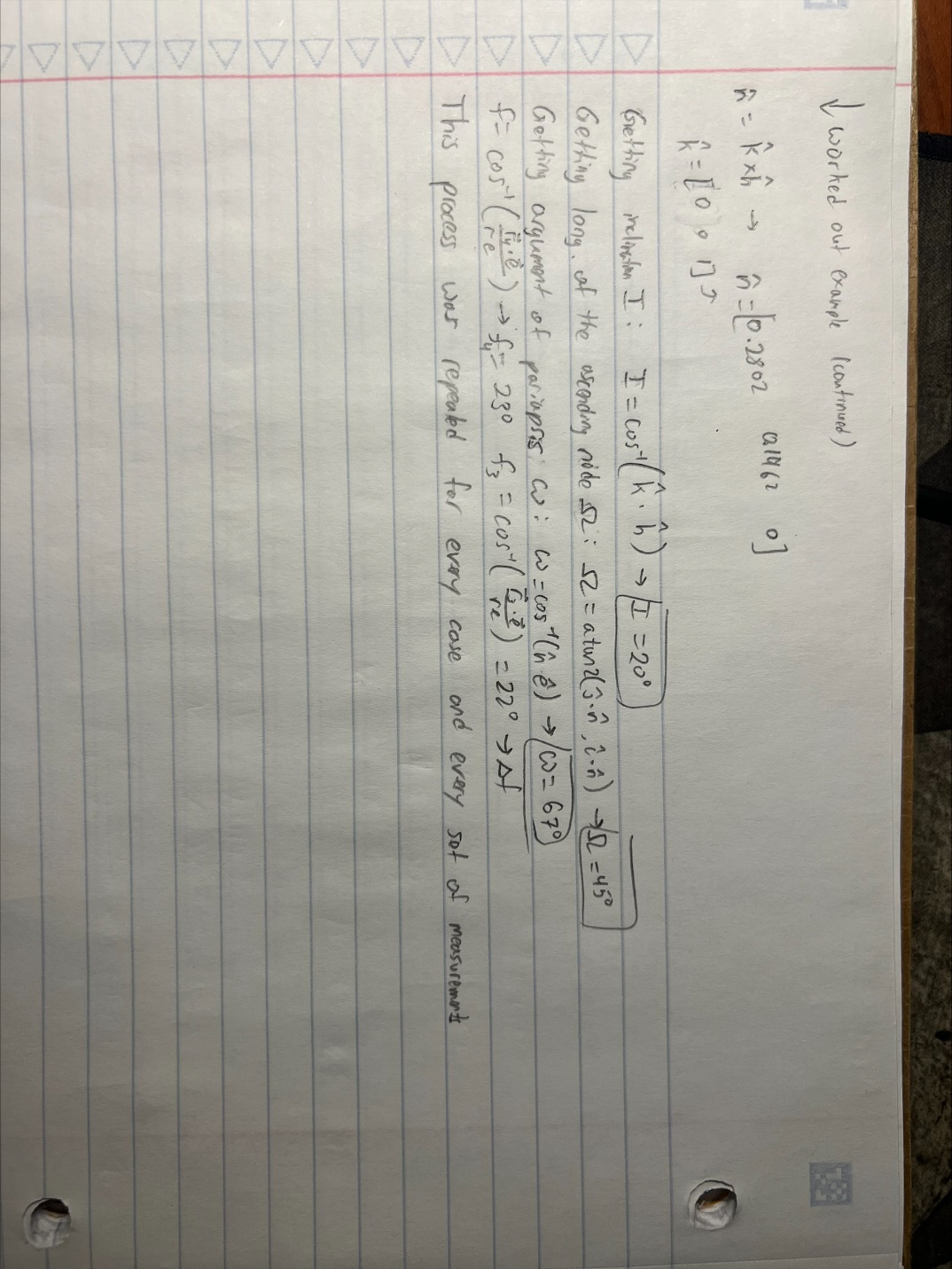
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Obs | a (km) | e | I (deg) | Ω (deg) | ω (deg) | Δf (deg) |
| Case 3 | Perfect | 3,4,5 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 30.00 |
| 2,4,6 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 60.00 |
| 1,4,7 | 8000.00 | 0.130 | 20.00 | 45.00 | 67.00 | 90.00 |
| Corrupted | 3,4,5 | 7999.91 | 0.130 | 19.99 | 45.00 | 66.99 | 30.0001 |
| 2,4,6 | 8000.00 | 0.130 | 19.99 | 45.00 | 66.99 | 59.9999 |
| 1,4,7 | 7999.99 | 0.130 | 20.00 | 45.00 | 66.99 | 90.0001 |

Below documents my solution process for obtaining this orbital data.



Below shows a worked out example of this process. Note that beyond this example, all data was obtained through a Matlab scipt.





**Answers to questions**

1. The three cases displayed the various true anomaly differences in between measurements. In Case 1, the measurements are taken in increments of 1 degrees. Case 2 sees increments of 10 degrees. Finally, Case 3 has the measurements taken every 30 degrees.
2. When comparing the perfect data to that of the corrupted (measured) data in each case, it seems that a moderate Δf of 20-30 degrees between measurements is ideal for minimizing the error. This is because taking data points too close together is not good for extrapolating the orbital elements, velocities, etc. because any small error in the measurements can be exacerbated.
3. According to the data collected, Gibb’s method is better for far apart observations, which is seen in Case 3. It has a consistently low % error of roughly 10-4. Case 3 used observations taken 30, 60, and 90 degrees apart, and it all had roughly the same low error.
4. The orbital elements observed were all the same, within the margin of error. This is expected because we are trying to study the effect of Δf while keeping everything else the same in order to better understand its effects on the accuracy of Gibb’s method.

% Made by Nicholas Luis (PSU ID 930841391)

% AERSP 450 HW 3 (Part A)

clear; clc;

% Constants

MU = 3.986\*(10^14); % m^3 / s^2

%% Case 1 - Perfect Observations

clear;

load('Case1.mat')

% Perfect Observations

r1 = Rtrue(1,:);

r2 = Rtrue(2,:);

r3 = Rtrue(3,:);

r4 = Rtrue(4,:);

r5 = Rtrue(5,:);

r6 = Rtrue(6,:);

r7 = Rtrue(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for perfect observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Case 1 - Corrupted Observations

clear;

load('Case1.mat')

% Corrupted Observations

r1 = RMeas(1,:);

r2 = RMeas(2,:);

r3 = RMeas(3,:);

r4 = RMeas(4,:);

r5 = RMeas(5,:);

r6 = RMeas(6,:);

r7 = RMeas(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for measured observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Case 2 - Perfect Observations

fprintf("\n\n\n-------BEGINNING OF CASE 2-------\n\n")

clear;

load('Case2.mat')

% Perfect Observations

r1 = Rtrue(1,:);

r2 = Rtrue(2,:);

r3 = Rtrue(3,:);

r4 = Rtrue(4,:);

r5 = Rtrue(5,:);

r6 = Rtrue(6,:);

r7 = Rtrue(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for perfect observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Case 2 - Corrupted Observations

clear;

load('Case2.mat')

% Corrupted Observations

r1 = RMeas(1,:);

r2 = RMeas(2,:);

r3 = RMeas(3,:);

r4 = RMeas(4,:);

r5 = RMeas(5,:);

r6 = RMeas(6,:);

r7 = RMeas(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for measured observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Case 3 - Perfect Observations

fprintf("\n\n\n-------BEGINNING OF CASE 3-------\n\n")

clear;

load('Case3.mat')

% Perfect Observations

r1 = Rtrue(1,:);

r2 = Rtrue(2,:);

r3 = Rtrue(3,:);

r4 = Rtrue(4,:);

r5 = Rtrue(5,:);

r6 = Rtrue(6,:);

r7 = Rtrue(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for perfect observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Case 3 - Corrupted Observations

load('Case3.mat')

% Corrupted Observations

r1 = RMeas(1,:);

r2 = RMeas(2,:);

r3 = RMeas(3,:);

r4 = RMeas(4,:);

r5 = RMeas(5,:);

r6 = RMeas(6,:);

r7 = RMeas(7,:);

if coplanarCheck(r3,r4,r5)~=0, fprintf("r 3,4,5 are not coplanar!"), end

if coplanarCheck(r2,r4,r6)~=0, fprintf("r 2,4,6 are not coplanar!"), end

if coplanarCheck(r1,r4,r7)~=0, fprintf("r 1,4,7 are not coplanar!"), end

% Remove semicolons to print the data to copy + paste into excel

%fprintf("Below are the values for measured observations: \n")

v345 = getVelo(r3,r4,r5); norm(v345);

v246 = getVelo(r2,r4,r6); norm(v246);

v147 = getVelo(r1,r4,r7); norm(v147);

%fprintf("Below is the δV (deviation from expected): \n")

dV345 = getDiff(v345 , V2true);

dV246 = getDiff(v246 , V2true);

dV147 = getDiff(v147 , V2true);

%fprintf("Below is the %% Error (deviation from expected): \n")

pV345 = 100\*dV345/norm(V2true);

pV246 = 100\*dV246/norm(V2true);

pV147 = 100\*dV147/norm(V2true);

% PART B

% Getting Orbital Elements (Uncomment to output the data)

fprintf("Orbit elements for the 3-4-5 data set: \n")

orbitElements = getOrbitalElements(r4, v345);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 2-4-6 data set: \n")

orbitElements = getOrbitalElements(r4, v246);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

fprintf("Orbit elements for the 1-4-7 data set: \n")

orbitElements = getOrbitalElements(r4, v147);

a = orbitElements(1,1);

e = orbitElements(:,2);

e\_mag = norm(e);

i = orbitElements(1, 3);

Omega = orbitElements(1, 4);

omega = orbitElements(1,5);

df = getF(e, r4) - getF(e, r3);

df = getF(e, r5) - getF(e, r4);

%% Functions

function trueAnomaly = getF(E, R)

% This function gets the true anomaly (in deg) given eccentricity and position vectors

e = norm(E);

r = norm(R);

trueAnomaly = acos(dot(R,E) / (r\*e)) \* (180/pi);

end

function orbitalElements = getOrbitalElements(R,V)

% This function gets the classical orbital elements given r and v vectors

R = R / 1000; % Converting to km

V = V / 1000; % Converting to km/s

r = norm(R);

v = norm(V);

MU = 3.986\*(10^5); % km^3 / s^2

i = [1 0 0]; j = [0 1 0]; k = [0 0 1];

% Getting semi major axis, a

a = ((-2/MU)\*(0.5\*v^2 - MU/r))^(-1);

% Intermediate Steps

h = cross(R,V);

n = cross(k,h);

% Calculating eccentricity

e = (cross(V,h)/MU) - (R/r);

% Calculating inclination

I = acos(dot(k,h/norm(h))) \* (180/pi);

% Calculating longitude of ascending node

Omega = atan2(dot(j,n/norm(n)), dot(j,n/norm(n))) \* (180/pi);

% Calculating argument of periapsis

omega = acos(dot(n/norm(n),e/norm(e))) \* (180/pi);

orbitalElements = NaN(3,5); % 5 Columns, 3 Rows in each

% Column 1 = a, Semi major axis (scalar stored in first element)

orbitalElements(1,1) = a;

% Column 2 = e, eccentricity (vector)

orbitalElements(:,2) = e';

% Column 3 = i, inclination (scalar stored in first element)

orbitalElements(1,3) = I;

% Column 4 = Ω, long. of the ascending node (scalar; first element)

orbitalElements(1,4) = Omega;

% Column 5 = ω, arg. of pariapsis (sclar; first element)

orbitalElements(1,5) = omega;

end

function delta = getDiff(va, vb)

% This function gets magnitude difference between two vectors of length 3

delta = sqrt( (va(1)-vb(1))^2 + (va(2)-vb(2))^2 + (va(3)-vb(3))^2 );

end

function vOut = getVelo(ra, rb, rc)

MU = 3.986\*(10^14); % m^3 / s^2

% This function gets velocity at the middle position

n = (norm(ra).\*cross(rb,rc)) + (norm(rb).\*cross(rc,ra)) + (norm(rc).\*cross(ra,rb));

d = cross(ra,rb) + cross(rb,rc) + cross(rc,ra);

s = (ra.\*(norm(rb)-norm(rc))) + (rb.\*(norm(rc)-norm(ra))) + (rc.\*(norm(ra)-norm(rb)));

vOut = ((cross(d,rb)/norm(rb))+s).\*sqrt(MU/(norm(n)\*norm(d)));

end

function boolean = coplanarCheck(ra, rb, rc)

% This function checks if all the vectors are coplanar (within reason)

% Converting to unit vectors

ra = ra ./ norm(ra);

rb = rb ./ norm(rb);

rc = rc ./ norm(rc);

boolean = round( dot(ra, cross(rb,rc)), 10 ); % Rounds if close enough to 0

end