# Detumbling using B-Dot Law

April 2, 2019

## 1 Introduction

In this paper we try recreating Example 7.5 from the book by Markley and Crassidis for which we write a MATLAB code. To recreate the Example we use Newton's Law of Gravitation for the motion of the satellite and B-Dot Law for Detumbling.

## 2 Method

We first initialize all the values from mentioned Example. Next we run a loop where we start from time $t = 0$ and increment $t$ by $dt$ assuming constant acceleration during time interval $dt$. The acceleraton on satellite is due to gravitational force exerted by the Earth. In the loop we use previous known values calculated from pervious iteration to calculate current values. The loop runs till we reach $T$. During the time interval $dt$, we convert ecef coordinates to lattitude, longitude and altitude and find magnetic field at a point using predefined MATLAB functions ecef2lla() and igrfmagm() respectively.

Then we apply B-Dot Law such that

$$m = -\frac{k}{||B||}\omega \times b$$

$$L = m \times B$$

where $k$ can be calculated by

$$k = \frac{4\pi}{T_{orb}}(1 + sin\xi_m)J_{min}$$

By Rotational Equations of Motion

$$L = I\alpha$$

$$\omega_f = \omega_i + \alpha \times dt$$

Now to find next coordinate we use Equations of Motion

$$r_f - r_i = v_i \times dt + \frac{1}{2} a \times dt^2$$

$$v_f = v_i + a \times dt$$

To find acceleration due gravity at $r_f$ towards the center of the Earth

$$a = \frac{GM}{r_f^2} \hat{r_f} = \frac{GM}{r_f^2} \frac{\vec{r_f}}{|r_f|}$$

Now these updated values are used in next iteration for calculating next values and so on.

# 3 Code

## 3.1 Initialization

First we simply initialize or set the intial values $I, r_i, v_i, \omega_i$ and $\alpha$

```
% The inertia vector of satellite
    inertia = [6400,-76.4,-25.6;-76.4,4730,-40;-25.6,-40,8160]' * 1e-7; % in kg*m^2
% The initial position and velocity of satellite in Earth-Centered Earth-Fixed
    ini_pos = [1029.7743e3;6699.3469e3;3.7896e3]'; % in m
    ini_vel = [6.2119e3;0.9524e3;4.3946e3]'; % in m/s
% Initial angular velocity and angular acceleration of satellite
    cang = [0.1;0.1;0.1]'; % rad/s
    angacc = [0;0;0]'; % rad/s^2
% Constants
    G = 6.67428e-11; % Earth gravitational constant in m^3/kg*s^2
    M = 5.972e24; % Earth mass in kg
    Re = 6371.2e3; % Radius of earth in m

% Minimum Principal Momentum
    J = 4726.01952; % in kg*m^2
```

## 3.2 Storing the Output

We create array initialized to zero to store the output or the answers required to pllot the graph

```
% Final Answers
    ang_vel = zeros(length(t),3); % To store the angular velocity of satellite after dt time
    torque = zeros(length(t),3); % To store the torque of satellite after dt time
```

### 3.3 Calculated Values

We calculate required values of different variables

```
%% Calculated Values
% Scalar linear velocity of satellite
    linvel = sqrt(dot(ini_vel, ini_vel)); % in m/s
% The altitude of satellite from earth's surface
    lla = ecef2lla(ini_pos); % ecef2lla() converts ecef coordinates to latitude, longitude and altitude
    alti = lla(3); % in meters
% Distance of satellite from center of earth
    Rc = Re + alti; % in m
% Time period of satellite
    timePeriod = 2*pi/sqrt(dot(cang, cang)); % in s^-1 Since Time Period = 2pi/(Angular Velocity)
% Accelaration of satellite due to earths gravity
    ini_acc = -ini_pos * (G * M / Rc^2) / sqrt(dot(-ini_pos, -ini_pos)); % in m/s^2 Since acceleration has value of GM/R^2 and in direction opposite to position vector
% Position of satellite after dt time
    ini_pos1 = ini_pos + ini_vel * dt + 0.5 * ini_acc * dt^2; % in m Assuming constant accelaration for dt time Since d = u*t + 0.5*a*t^2
% Angle Of Inclination Of Orbit
    normal = cross(ini_pos, ini_pos1); % Normal to orbit plane
    normalDotK = normal(3) / sqrt(dot(normal, normal)); % Dot product of unit normal vector with k^
    angleOfInclinationOfOrbit = acos(normalDotK); % in rad
% Positive Scalar Gain of Bdot Law
    k = 4*pi*(1 + sin(angleOfInclinationOfOrbit)*J)/timePeriod; % in kg*m^2*s Since k = 4*pi*(1+sin(angle of inclination))*Jmin/Torb

%% Initializations for loop
pos = ini_pos;
vel = ini_vel;
acc = ini_acc;
```

### 3.4 Main Loop

Now finally we run the loop for time interval $dt$ such that $t_i = 0$ and after each iteration $t_f = t_i + dt$. First we store all the variables in temporary variables for resusing later. Then we use ecef2lla() to find lattitude, longitude and altitude. Next we use igrfmagm() to find magnetic field $B$ at current position. And now we finally update the values and store them in a 2D array.

```
for i=1:length(t)

    veli = vel;
    posi = pos;
    acci = acc;
    cangi = cang;

    lla = ecef2lla(posi);
    lat = lla(1); % Latitude
    long = lla(2); % Longitude
    alti = lla(3); % Altitude
    alti = min(alti, 6e5); % Since igrfmagm has a limit on altitude of 6e5

    [mag_field_vector1,hor_intensity,declination,inclination,total_intensity] = igrfmagm(alti,lat,long,decyear(2015,7,4),12); % igrfmagm used to calculate the magnetic field of earth at particular position
    mag_field_vector = mag_field_vector1 * 1e-9; % in T Since the function returns the value in nT
    mag_field_vector = mag_field_vector.'; % Taking transpose of the magnetic feild

%% B-dot
%   Determinant of Magnetic Field
        detb = sqrt(dot((mag_field_vector),(mag_field_vector)));
%   Magnetic Dipole Moment
        m = ((k)/detb*norm(mag_field_vector))*cross(cang, mag_field_vector); % in A*m^2 Since m = -k (w x b)/||B||
%   Torque
        vtorque = cross(m, mag_field_vector); % in N*m Since T = m x B
%   Angular Accelaration of Satellite
        angacc = vtorque * inv(inertia); % in rad/s^2 Since I x angacc = T
%% Updating loop values

    cang = cangi + (angacc * dt); % calculates the new angular velocity
```

```
    pos = posi + (veli * dt) + (0.5 * acci * dt^2); % calculates
     the new position assuming constant acc for dt time
    vel = veli + (acci * dt); % calculates the new velocity
    acc = -pos * (G * M / Rc^2) / sqrt(dot(pos, pos)); %
    calculates the new acceleration
% Display the new angular velocity
    disp('new angular velocity');
    disp(cang);
    ang_vel(i,:) = cang;
    torque(i,:) = vtorque;

 end
```

### 3.5 Plots

We now plot the graphs from obtained values

```
%% Plots
% Plot for Angular velocity
    figure(1)
    subplot(3,1,1)
    plot(t/60,ang_vel(:,1));
    subplot(3,1,2)
    plot(t/60,ang_vel(:,2));
    subplot(3,1,3)
    plot(t/60,ang_vel(:,3));
% Plot for Torque
    figure(2)
    subplot(3,1,1)
    plot(t,torque(:,1));
    subplot(3,1,2)
    plot(t,torque(:,2));
    subplot(3,1,3)
    plot(t,torque(:,3));
```

### 3.6 Final Code

```
clear all;
clc;

%% Specify time, where dt is step size and T (in sec) is total
    time upto which the algorithm will run
dt = 0.30; T = 90*60*3;

t = 0:dt:T;
%% Initializations
% The inertia vector of satellite
```

```
    inertia = [6400,-76.4,-25.6;-76.4,4730,-40;-25.6,-40,8160]' * 1e-7; % in kg*m^2
% The initial position and velocity of satellite in Earth-Centered Earth-Fixed
    ini_pos = [1029.7743e3;6699.3469e3;3.7896e3]'; % in m
    ini_vel = [6.2119e3;0.9524e3;4.3946e3]'; % in m/s
% Initial angular velocity and angular acceleration of satellite
    cang = [0.1;0.1;0.1]'; % rad/s
    angacc = [0;0;0]'; % rad/s^2
% Constants
    G = 6.67428e-11; % Earth gravitational constant in m^3/kg*s^2
    M = 5.972e24; % Earth mass in kg
    Re = 6371.2e3; % Radius of earth in m
% Final Answers
    ang_vel = zeros(length(t),3); % To store the angular velocity of satellite after dt time
    torque = zeros(length(t),3); % To store the torque of satellite after dt time
% Minimum Principal Momentum
    J = 4726.01952; % in kg*m^2

%% Calculated Values
% Scalar linear velocity of satellite
    linvel = sqrt(dot(ini_vel, ini_vel)); % in m/s
% The altitude of satellite from earth's surface
    lla = ecef2lla(ini_pos); % ecef2lla() converts ecef coordinates to latitude, longitude and altitude
    alti = lla(3); % in meters
% Distance of satellite from center of earth
    Rc = Re + alti; % in m
% Time period of satellite
    timePeriod = 2*pi/sqrt(dot(cang, cang)); % in s^-1 Since Time Period = 2pi/(Angular Velocity)
% Accelaration of satellite due to earths gravity
    ini_acc = -ini_pos * (G * M / Rc^2) / sqrt(dot(-ini_pos, -ini_pos)); % in m/s^2 Since acceleration has value of GM/R^2 and in direction opposite to position vector
% Position of satellite after dt time
    ini_pos1 = ini_pos + ini_vel * dt + 0.5 * ini_acc * dt^2; % in m Assuming constant accelaration for dt time Since d = u*t + 0.5*a*t^2
% Angle Of Inclination Of Orbit
    normal = cross(ini_pos, ini_pos1); % Normal to orbit plane
    normalDotK = normal(3) / sqrt(dot(normal, normal)); % Dot product of unit normal vector with k^
    angleOfInclinationOfOrbit = acos(normalDotK); % in rad
% Positive Scalar Gain of Bdot Law
    k = 4*pi*(1 + sin(angleOfInclinationOfOrbit)*J)/timePeriod;
```

```
    % in kg*m^2*s Since k = 4*pi*(1+sin(angle of inclination))*Jmin/Torb

%% Initializations for loop
pos = ini_pos;
vel = ini_vel;
acc = ini_acc;

%% Main loop
 for i=1:length(t)

     veli = vel;
     posi = pos;
     acci = acc;
     cangi = cang;

     lla = ecef2lla(posi);
     lat = lla(1); % Latitude
     long = lla(2); % Longitude
     alti = lla(3); % Altitude
     alti = min(alti, 6e5); % Since igrfmagm has a limit on altitude of 6e5

     [mag_field_vector1,hor_intensity,declination,inclination,total_intensity] = igrfmagm(alti,lat,long,decyear(2015,7,4),12); % igrfmagm used to calculate the magnetic field of earth at particular position
     mag_field_vector = mag_field_vector1 * 1e-9; % in T Since the function returns the value in nT
     mag_field_vector = mag_field_vector.'; % Taking transpose of the magnetic feild

%% B-dot
%   Determinant of Magnetic Field
         detb = sqrt(dot((mag_field_vector),(mag_field_vector)));
%   Magnetic Dipole Moment
        m = ((k)/detb*norm(mag_field_vector))*cross(cang,mag_field_vector); % in A*m^2 Since m = -k (w x b)/||B||
%   Torque
         vtorque = cross(m, mag_field_vector); % in N*m Since T = m x B
%   Angular Accelaration of Satellite
         angacc = vtorque * inv(inertia); % in rad/s^2 Since I x angacc = T
%% Updating loop values

     cang = cangi + (angacc * dt); % calculates the new angular velocity
     pos = posi + (veli * dt) + (0.5 * acci * dt^2); % calculates
```

```
    the new position assuming constant acc for dt time
    vel = veli + (acci * dt); % calculates the new velocity
    acc = -pos * (G * M / Rc^2) / sqrt(dot(pos, pos)); %
    calculates the new acceleration
% Display the new angular velocity
    disp('new angular velocity');
    disp(cang);
    ang_vel(i,:) = cang;
    torque(i,:) = vtorque;

 end

%% Plots
% Plot for Angular velocity
    figure(1)
    subplot(3,1,1)
    plot(t/60,ang_vel(:,1));
    subplot(3,1,2)
    plot(t/60,ang_vel(:,2));
    subplot(3,1,3)
    plot(t/60,ang_vel(:,3));
% Plot for Torque
    figure(2)
    subplot(3,1,1)
    plot(t,torque(:,1));
    subplot(3,1,2)
    plot(t,torque(:,2));
    subplot(3,1,3)
    plot(t,torque(:,3));
```

# 4 Results

Below are the plots of $\omega_1, \omega_2$ and $\omega_3$ in Figure 1 and plots of $L_1, L_2$ and $L_3$ in Figure 2

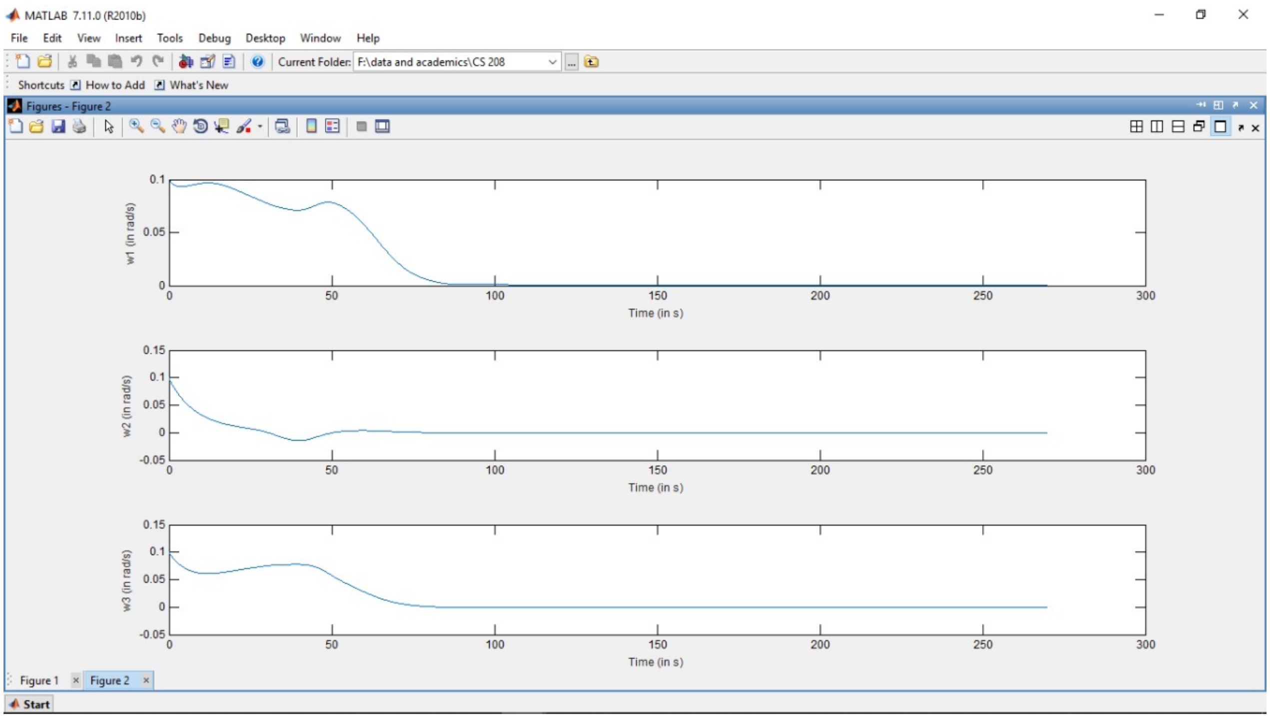
MATLAB 7.11.0 (R2010b)
File Edit View Insert Tools Debug Desktop Window Help
Current Folder: F:\data and academics\CS 208
Shortcuts How to Add What's New
Figures - Figure 2
w1 (in rad/s)
Time (in s)
w2 (in rad/s)
Time (in s)
w3 (in rad/s)
Time (in s)
Figure 1
Figure 2
Start

Figure 1

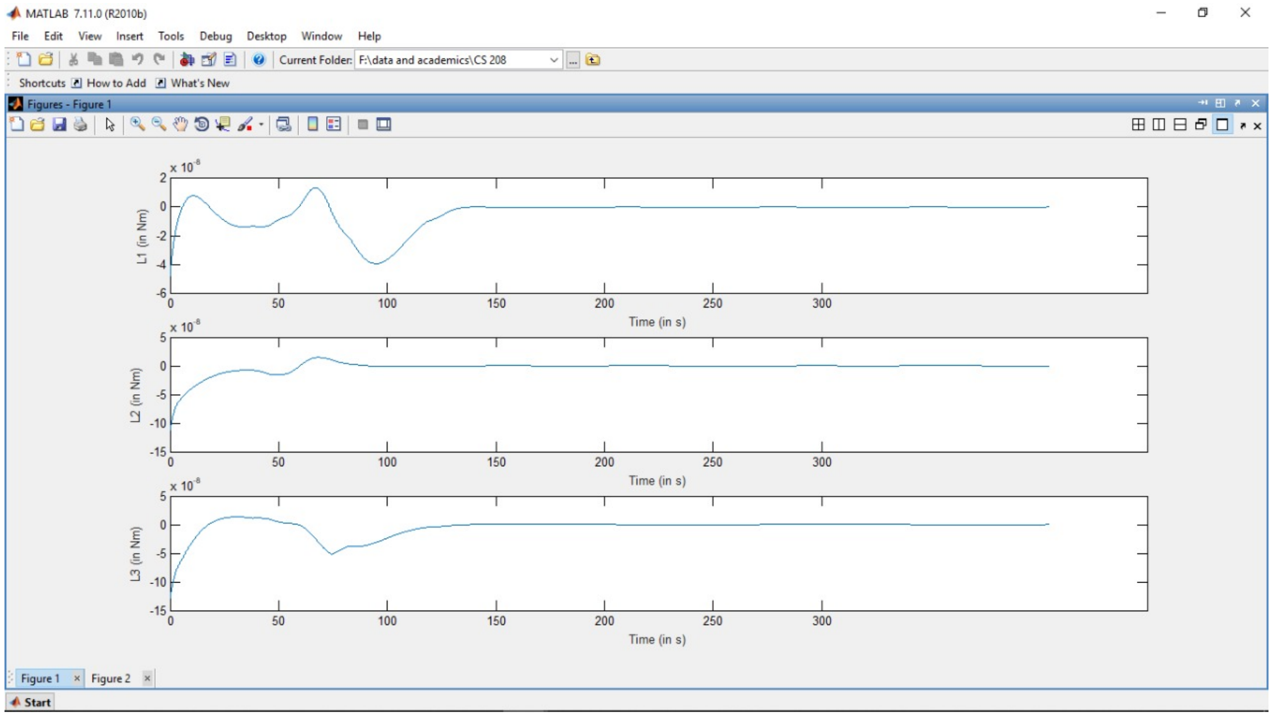
MATLAB 7.11.0 (R2010b)
File Edit View Insert Tools Debug Desktop Window Help
Current Folder: F:\data and academics\CS 208
Shortcuts How to Add What's New
Figures - Figure 1
L1 (in Nm)
L2 (in Nm)
L3 (in Nm)
Time (in s)
Figure 1
Figure 2
Start

Figure 2