# UUM571E - Spacecraft Dynamics Project - Part 2 - Fall 2022/2023

Name: **Resul Dagdanov**, Student ID: **511211135**

***Abstract*— This part of the project illustrates the usage of the Extended Kalman Filter (EKF) for estimating Sun direction during eclipse periods on Mars. By applying EKF for nonlinear sensor measurements from multiple sensors, the approximate direction of the Sun could be estimated with and without rate-gyro sensors. The accuracy of predictions will be compared with actual measurements in terms of Root Mean Square (RMS) error. From project part 1, the location and altitude of the space vehicle on Mars parking orbit of 300 km is calculated. By using previous knowledge about the position, attitude, and velocity of the spacecraft, the accurate estimation of the sun-sensor measurements will be conducted with the EKF algorithm.**

***Index Terms*— Spacecraft Dynamics, Sun-Sensor Estimation, Extended Kalman Filter, Mars Eclipse**

## I. Overview

In Section II, general information about Kalman Filter and Extended Kalman Filter is explained in detail. In Section III, the approach to detect Mars eclipse or shadow in parking orbit is handled. Section IV explains the construction of the EKF algorithm with measurement and state models specifically for this project. In this section, the required two experimental cases are also handled. In Section V, experimental results are discussed with visualization graphs. In the latest Section VI, general comments about the project part 2 and conclusion are given.

## II. Background

### A. Kalman Filter

The Kalman filter is a mathematical method used to estimate the state of a system based on noisy and incomplete observations. It is a recursive algorithm that uses a set of equations to predict the state of the system at the next time step, given the current state and observations. The Kalman filter is commonly used in engineering and control systems to estimate the state of a system and make predictions about its future behavior [1].

The basic form of the Kalman filter consists of two steps: prediction and update. In the prediction step, the filter uses the current state of the system and a model of the system's dynamics to predict the state at the next time step. This prediction is then refined in the update step, using new observations of the system.

The Kalman filter is formulated using a set of linear equations. These equations can be written in matrix form as follows:

$$x_k = A_k x_{k-1} + B_k u_k + w_k \tag{1}$$

Resul Dagdanov is studying in the Department of Aeronautical and Astronautical Engineering, Istanbul Technical University, Turkey dagdanov21 at itu.edu.tr

$$y_k = C_k x_k + v_k \tag{2}$$

where, $x_k$ is the state of the system at time $k$, $u_k$ is the input to the system at time $k$, and $y_k$ is the observation of the system at time k. $A_k$, $B_k$, and $C_k$ are matrices that describe the dynamics of the system, and $w_k$ and $v_k$ are noise terms.

The Kalman filter estimates the state of the system by iteratively applying the prediction and update steps. In the prediction step, the filter uses the current state estimate and the system dynamics to predict the state at the next time step:

$$x_k^* = A_k x_{k-1} + B_k u_k \tag{3}$$

In the update step, the filter uses the prediction and the new observation to refine the state estimate:

$$K_k = P_k^* C_k^T (C_k P_k^* C_k^T + R_k)^{-1} \tag{4}$$

$$x_k = x_k^* + K_k (y_k - C_k x_k^*) \tag{5}$$

$$P_k = (I - K_k C_k) P_k^* \tag{6}$$

where, $K_k$ is the Kalman gain, $P_k$ is the estimate of the state covariance, and $R_k$ is the covariance of the observation noise.

### B. Extended Kalman Filter

The Extended Kalman Filter (EKF) is a variation of the Kalman filter that is used to estimate the state of nonlinear systems. Like the Kalman filter, the EKF uses a recursive algorithm to estimate the state of the system based on noisy and incomplete observations. However, the EKF is able to handle nonlinear systems by linearizing the system dynamics around the current state estimate.

The basic form of the EKF consists of two steps: prediction and update. In the prediction step, the EKF uses the current state of the system and a model of the system's dynamics to predict the state at the next time step. This prediction is then refined in the update step, using new observations of the system.

The EKF is formulated using a set of nonlinear equations. These equations can be written in the following form:

$$x_k = f(x_{k-1}, u_k) + w_k \tag{7}$$

$$y_k = h(x_k) + v_k \tag{8}$$

where, $x_k$ is the state of the system at time $k$, $u_k$ is the input to the system at time $k$, and $y_k$ is the observation of the system at time $k$. $f(x, u)$ is a nonlinear function that describes the dynamics of the system, and $h(x)$ is a nonlinear function that relates the state to the observations. $w_k$ and $v_k$ are noise terms.

To estimate the state of the system using the EKF, the filter linearizes the nonlinear functions $f(x, u)$ and $h(x)$ around the current state estimate $x_k^*$. The linearized forms of these functions are denoted as $f_k(x)$ and $h_k(x)$, respectively. The linearized versions of the state and observation equations are then given by:

$$x_k^* = f_k(x_{k-1}) + B_k u_k \tag{9}$$

$$y_k^* = h_k(x_k^*) \tag{10}$$

where, $B_k$ is the Jacobian matrix of $f_k(x)$ with respect to x. In the update step, the EKF uses the prediction and the new observation to refine the state estimate equations are the same as Kalman Filter equations given in the previous subsection.

The prediction step of the EKF involves using the current estimate of the state of the system (also known as the "a priori" estimate) to predict the state of the system at the next time step. This prediction is made by using the system's dynamics model, which describes how the state of the system changes over time. The predicted state estimate is then used as the new "a priori" estimate for the next iteration of the EKF.

The update step of the EKF involves using new measurements of the state of the system to improve the accuracy of the state estimate. This is done by combining the "a priori" estimate from the prediction step with the new measurements using a weighted average, where the weights are determined by the variance (or uncertainty) of the "a priori" estimate and the measurement. The resulting estimate is called the "a posteriori" estimate, and it is used as the new "a priori" estimate for the next iteration of the EKF.

Overall, the EKF works by iteratively improving the estimate of the state of the system by using a combination of the system's dynamics model and new measurements. The EKF is particularly useful in situations where the system being modeled is nonlinear since it can handle such systems by linearizing the dynamics model around the current estimate of the state.

*1) Prediction Step:* The prediction step involves using the current estimate of the state of the system, $x(k-1)$, and the system's dynamics model, $f$, to predict the state of the system at the next time step, $x(k)$. This prediction is made using the following formula:

$$x(k) = f(x(k-1), u(k-1), k-1) \tag{11}$$

where $u(k-1)$ is the control input to the system at time $k-1$ [2].

*2) Update Step:* The update step involves using new measurements, $z(k)$, of the state of the system to improve the accuracy of the state estimate. This is done by combining the "a priori" estimate from the predict step, $x(k|k-1)$, with the new measurements using a weighted average. The resulting estimate is the "a posteriori" estimate, $x(k|k)$.

The update step is typically split into two sub-steps: the prediction of the measurement, $h(x(k))$, and the correction of the state estimate based on the prediction error, $y(k) = z(k) - h(x(k))$. The correction step is given by the following formula:

$$x(k|k) = x(k|k-1) + K(k) * y(k) \tag{12}$$

where $K(k)$ is the Kalman gain, which determines the weight of the measurement in the weighted average. The Kalman gain is given by [2]:

$$\begin{aligned} K(k) =& P(k|k-1) * H(k)^T * \\ & (H(k) * P(k|k-1) * H(k)^T + R(k))^{-1} \end{aligned} \tag{13}$$

## III. Mars Eclipse

To detect the position and intensity of sunlight while the spacecraft is in Mars parking orbit, digital sun-sensors [3] are used instead of analog sun-sensors, as these sensors are more accurate and more complex. The sun-sensors measure the sun vector in sensor coordinates with enough accuracy. As the sensor orientation on the spacecraft is known, it is possible to convert sensor coordinate frame measurements to a space vehicle body frame. During an eclipse, the sun's light is partially or entirely blocked by the planet Mars, causing a decrease in the intensity of sunlight. Digital sun-sensors can detect this decrease in sunlight intensity and send a signal to the spacecraft's control systems, which can then adjust the orientation of the spacecraft to compensate for the eclipse.

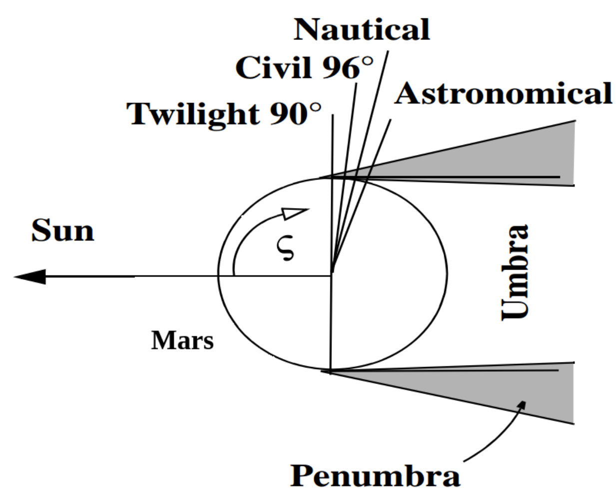


Fig. 1. Geometry for Mars Solar Illumination. Visualizing umbra and penumbra regions during a solar eclipse on the planet Mars.

As visualized in Figure 1, if the spacecraft is orbiting Mars at the time of a solar eclipse, it will experience a partial eclipse if it is within the penumbra, and a total eclipse if the spacecraft is within the umbra [2].

From Project Part 1, the spacecraft will be on a 300 km parking orbit of Mars on the date **2024 year**, **January month**, **24th day**, **hour 10 a.m.**, **59 minutes**, and **59 seconds**.

The position of the Sun and Mars are calculated by applying Algorithm 10.2 in reference [4]. The main steps are given below with corresponding equations and itemization below in order.

(1) Julian day conversion number is calculated with the following Eq. (14) and Eq. (15):

$$J_D = J_0 + \frac{UT}{24} \tag{14}$$

where $J_0$ is the symbol for the Julian day number at 0 h UT as could be calculated with Eq. (15).

$$\begin{aligned} J_0 = 367y - INT\left[\frac{7\left[y + INT(\frac{m+9}{12})\right]}{4}\right] \\ + INT\left(\frac{275m}{9}\right) + d + 1,721,013.5 \end{aligned} \tag{15}$$

where y is a year in a range of [1901, 2099], m is a month between a range of [1, 12], and d is a day number in the range [1, 31]. By applying the calculated date when the spacecraft reached the parking orbit of Mars, the Julian day at 0h UT is $\boxed{J_0 = 2.4603335 \text{ x } 10^6}$. By applying the conversion according to the hour-minute-second exact time, the Julian day is $\boxed{J_D = 2.4603339 \text{ x } 10^6}$.

(2) The number of days since J2000 is calculated with the following Eq. (16).

$$n = J_D + 2451545 \tag{16}$$

where $J_D$ is a Julian day. After applying the Eq. (16), $\boxed{n = 8.788958 \text{ x } 10^3}$.

(3) The mean anomaly is computed with the following equation in degrees valid between $(0^o \le M \le 360^o)$.

$$M = mod(357.528^o + (n \text{ x } 0.9856003^o),\ 360^o) \tag{17}$$

The mean anomaly of the planet Mars, while the spacecraft is just entered the parking orbit, is $\boxed{M = 19.9280^o}$.

(4) The mean solar longitude $L$ is computed with Eq. (18).

$$L = mod(280.459^o + (n \text{ x } 0.98564736^o),\ 360^o) \tag{18}$$

The mean solar longitude of the planet Mars, while the spacecraft is just entered the parking orbit, is $\boxed{L = 303.2726^o}$.

(5) By using the Astronomical Almanac [5] information, apparent solar ecliptic longitude $\lambda$ (in degrees) is computed as in Eq. (19).

$$\lambda = L + 1.915^o sin(M) + 0.020^o sin(2M) \tag{19}$$

where $L$ is the mean solar longitude and $M$ is the mean anomaly of the planet Mars when the spacecraft is entered the parking orbit of 300 km. The solar ecliptic longitude is calculated as $\boxed{\lambda = 303.9381^o}$.

(6) The obliquity $\epsilon$ is obtained using Eq. (20).

$$\epsilon = 23.439^o - (3.56 \text{ x } 10^{-7} \text{ x } n) \tag{20}$$

When the spacecraft is entered the parking orbit, an obliquity is $\boxed{\epsilon = 23.4355^o}$.

(7) The unit vector in an inertial frame from Mars to the Sun $\hat{u}$ is calculated using Eq. (21).

$$\hat{u} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\epsilon) & -sin(\epsilon) \\ 0 & sin(\epsilon) & cos(\epsilon) \end{bmatrix} \begin{Bmatrix} cos(\lambda) \\ sin(\lambda) \\ 0 \end{Bmatrix} \tag{21}$$

$$\hat{u} = \begin{Bmatrix} cos(\lambda) \\ cos(\epsilon)sin(\lambda) \\ sin(\epsilon)sin(\lambda) \end{Bmatrix} \tag{22}$$

At the initial conditions when the spacecraft just entered the parking orbit of Mars, the unit vector $\hat{u}$ of the sun direction from Mars in the inertial frame is calculated as follows:

$$\boxed{\hat{u} = [0.5583, \quad -0.7508, \quad -0.3531]'} \tag{23}$$

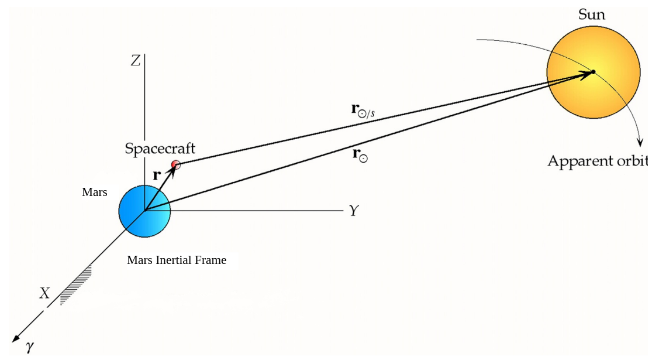


Fig. 2. Visualization of the sun position vector, spacecraft position vector on Mars inertial frame coordinates.

(8) The distance from Mars to the Sun is taken at approximately 1.52 x AU, where the astronomical unit (AU) is 149597870.691 km. The inertial frame sun position vector, as illustrated in Fig. 2, is calculated with the following formula:

$$\begin{aligned} r_{sun} = \hat{u} \text{ x } (1.52 \text{ x } AU) \text{ x} \\ (1.00014 - 0.0933941 cos(M) - 0.00014 cos(2M)) \end{aligned} \tag{24}$$

By applying the above equation with previously calculated initial mean anomaly $M$, the sun position vector $r_{sun}$ in an inertial frame of Mars could be computed as follows:

$$\boxed{r_{sun} = [0.7619, \quad -1.0245, \quad -0.4818]' \text{ x } 10^8 km} \tag{25}$$

An initial value of the right ascension of the ascending node $\sigma_0 = 49.57854^o$ and an initial value of the argument of perigee $\omega_0 = 336.04084^o$ are taken from reference [6].

When the position state vector of the spacecraft and the position vector of the sun in the Mars inertial frame is given, it is possible to determine whether the spacecraft is in the shadow of Mars or not.

$$\theta = cos^{-1}\left(\frac{r_{sun}.r_{sc}}{||r_{sun}||.||r_{sc}||}\right) \tag{26}$$

$$\theta_1 = cos^{-1}(R_{mars}/||r_{sc}||) \tag{27}$$

$$\theta_2 = cos^{-1}(R_{mars}/||r_{sun}||) \tag{28}$$

where, $R_{mars}$ is a radius of the planet Mars, $r_{sc}$ and $r_{sun}$ are inertial frame position vectors of the spacecraft and the Sun, respectively. The spacecraft is in the shadow of Mars when $\theta_1 + \theta_2 \le \theta$ and vise-versa otherwise.

## IV. EKF Approach

EKF algorithm is described in Section II in detail. The nonlinear functions $f[.]$ and $h[.]$ if equations Eq. (7) and Eq. (8), respectively, are dynamic and measurement functions. This system of function is modeled as a zero-mean Gaussian noise $w(k)$ with covariance $Q(k)$ and another zero-mean Gaussian noise $v(k)$ with covariance $R(k)$ where noise values are uncorrelated with zero mean as indicated below:

$$E[w(k)] = E[v(k)] = 0, \quad \forall k \tag{29}$$

The main goal of the project is to design an extended Kalman filter (EKF) for estimating the direction of the Sun while the spacecraft is in the parking orbit of Mars. The sun heading as well as its rate of change will be estimated in the body frame of the orbiting space vehicle [7].

The measurement model of the Coarse Sun-Sensor (CSS) is given below:

$$h_i(x) = n_i * d \tag{30}$$

where $*$ is a dot product of the sun-line heading and the normal to the $i^{th}$ sensor. This will yield easy partial derivatives for the rows of transposed normal vectors ($H$ matrix). It is expected that this $H$ matrix will change its size with respect to the amounts of measurements.

$$H = \left[\frac{\partial h(x, t_i)}{\partial x}\right] \tag{31}$$

where $x$ is a state vector at time step $t_i$. Alternatively, the measurement matrix $H$ could be defined as follows:

$$H(x) = [n_1^T, n_2^T, ..., n_i^T]^T \tag{32}$$

In this part of the project, it is expected to apply EKF for both with and without rate gyro sensor measurements.

*1) Case 1:* The first case is to find the Sun heading angle with the EKF algorithm while rate gyro sensor measurements are **available**.

$$x = [d] \tag{33}$$

The propagation error is calculated as follows:

$$x' = F(x) = d' = -\omega_{\beta/N} \text{ x } d = -\left[\widetilde{\omega}_{\beta/N}\right]^{\beta} d \tag{34}$$

$$d_{k+1} = d_k - \Delta t.\omega_{\beta/N} \text{ x } d_k \tag{35}$$

Then the partial derivative of the nonlinear function could be found as follows:

$$\left[\frac{\partial F(d, t_i)}{\partial d}\right] = -\left[\widetilde{\omega}_{\beta/N}\right] \tag{36}$$

*2) Case 2:* The second case is to find the Sun heading angle with the EKF algorithm while rate gyro sensor measurements are **not available**.

Because gyro sensor results are not measured by the sensor, the gyro calculations will not be added to the filter. As a result, the sun heading vector estimate will be as follows:

$$w_k = \frac{1}{\Delta t}\frac{d_k \text{ x } d_{k-1}}{||d_k \text{ x } d_{k-1}||} \ . \ arccos\left(\frac{d_k.d_{k-1}}{||d_k||.||d_{k-1}||}\right) \tag{37}$$

The EKF function takes as input the state transition matrix $F$, the observation matrix $H$, the process noise covariance $Q$, the measurement noise covariance $R$, the initial state estimate $x$, the initial covariance estimate $P$, and the measurement vector $z$. It then performs the prediction step, which involves predicting the state at the next time step using the state transition matrix and the current state estimate, and updating the covariance estimate to account for the process noise. Next, the EKF function performs the correction step, which involves using the measurement to update the state estimate. It computes the Kalman gain, which is a weighting factor that determines how much the measurement should be incorporated into the updated state estimate. The updated state and covariance estimates are then returned as output arguments.

In total, 8 different sun sensors are placed on the spacecraft to detect sunlight while orbiting in the parking orbit of Mars. The body frame position configuration of these sensors is given as follows:

$$H = \begin{Bmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \\ n_5 \\ n_6 \\ n_7 \\ n_8 \end{Bmatrix} = \begin{bmatrix} a & -b & b \\ a & -b & -b \\ a & b & -b \\ a & b & b \\ -a & -b & b \\ -a & -b & -b \\ -a & b & -b \\ -a & b & b \end{bmatrix} \tag{38}$$

where $\boxed{a = \frac{\sqrt{2}}{2}}$ and $\boxed{b = 0.5}$. Each vector $n_i$ represents $i^{th}$ sun sensor on the body referenced frame of the spacecraft.

## V. Experiments and Results

Results are obtained using Matlab and all custom scripts are included in the uploaded zip file. By running the Matlab scripts, the visualization and all calculations that are demonstrated above could be obtained along with RMS errors.

As seen in Fig. 6, the sensor measurement of the Sun direction vector and actual Sun direction vector angle difference is plotted. The eclipse period is computed as $\boxed{T_{eclipse} = 1145sec}$. As the spacecraft measurements are started at the true anomaly of $\theta_0 = 200^o$, the total duration until reaching the eclipse location is 4390 seconds. It is observed that the eclipse occurs between $\theta^0_{eclipse} = 71.6611^o$ and $\theta^1_{eclipse} = 132.0830^o$.

*1) RMS Errors:*

- Case 1: RMS Error Angle Between True and Measured: $3.7806e7km$
- Case 1: RMS Error Angle Between True and Estimated: $217.2029km$
- Case 2: RMS Error Angle Between True and Measured: $3.7407e7km$
- Case 2: RMS Error Angle Between True and Estimated: $16.1830km$

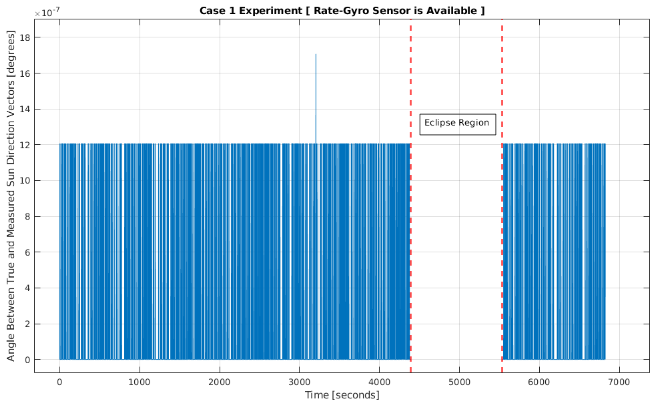


Fig. 3. Angle Between True and Measured Sun Direction Vectors when Rate-Gyro Sensor is Available.

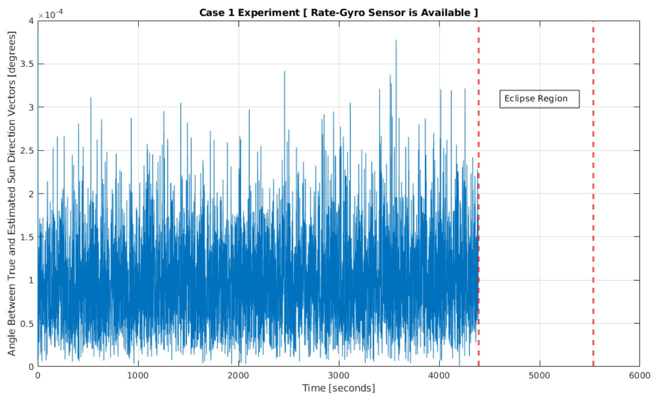


Fig. 4. Angle Between True and Estimated Sun Direction Vectors when Rate-Gyro Sensor is Available.

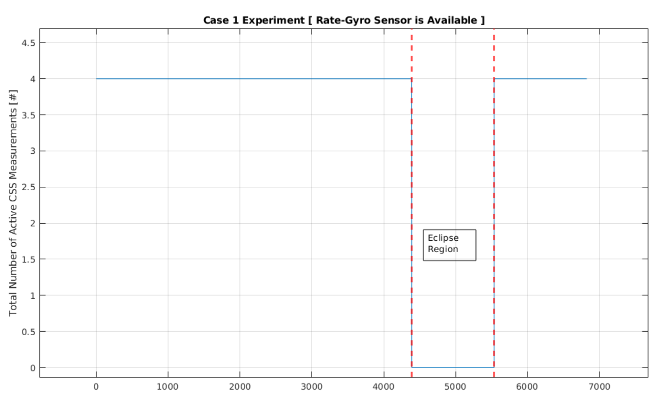


Fig. 5. Total Number of Active CSS Measurements when Rate-Gyro Sensor is Available.

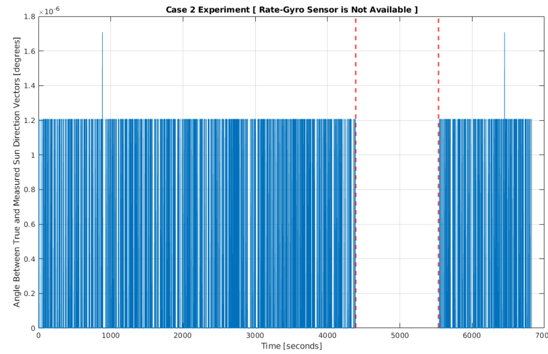


Fig. 6. Angle Between True and Measured Sun Direction Vectors when Rate-Gyro Sensor is not Available.

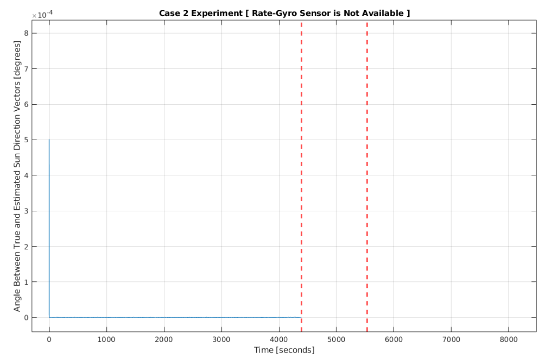


Fig. 7. Angle Between True and Estimated Sun Direction Vectors when Rate-Gyro Sensor is not Available.

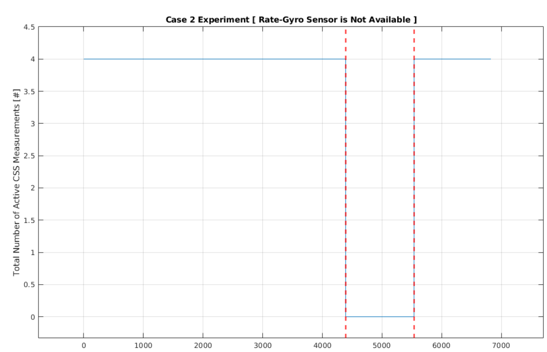


Fig. 8. Total Number of Active CSS Measurements when Rate-Gyro Sensor is not Available.

# VI. Conclusion

In conclusion, all of the calculations and simulations of part 2 of this project are completed. All simulations are calculated with custom scripts in MATLAB. Each script line has a comment for better readability and in these comments, useful equation numbers in reference [4] are given.

# References


[1] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[2] D. A. Vallado, *Fundamentals of astrodynamics and applications*. Springer Science & Business Media, 2001, vol. 12.

[3] M. D. Griffin, *Space vehicle design*. AIAA, 2004.

[4] H. Curtis, *Orbital mechanics for engineering students*. Butterworth-Heinemann, 2013.

[5] U. S. N. O. N. A. Office, G. B. N. A. Office, E. R. C. G. Britain), R. A. Laboratory, C. for the Central Laboratory of the Research Councils (Great Britain), U. S. D. of the Navy, U. S. Congress, and G. B. H. Office, *The Astronomical Almanac*. US Government Printing Office, 2008.

[6] E. M. Standish, J. G. Williams *et al.*, "Orbital ephemerides of the sun, moon, and planets," *Explanatory supplement to the astronomical almanac*, pp. 279–323, 1992.

[7] C. Hajiyev and D. Cilden-guler, "Nanosatellite attitude estimation in sun and eclipse periods without gyroscopes," *APPLIED AND COMPUTATIONAL MATHEMATICS*, vol. 21, no. 3, pp. 246–258, 2022.