# AE4878 - Mission Geometry and Orbit Design
# Part 6 - Error Analysis

Ali Nawaz
3ME, LR, Delft University of Technology.

September 2, 2018

## 1 Why analyse errors?

No system can be perfectly designed. Let it be mechanical tolerances or algorithmic limitations, all engineering practices come with limited precision. This precision is quantified with the help of errors. Errors define the accuracy of any given engineering system under consideration. To understand the bounds of perfection of a system, errors present in a system must be analysed. Error analysis of any system come with the following aspects: error sources (where in the system does the error take place?), error characterisation (what type of error is it?), error propagation (what effects does the error have?) and error budget (what's the total error?).

For this assignment, the sources and types of errors are given. The objective is to use different error propagation techniques to evaluate the effect of the errors and obtain the corresponding error budget. This report will look into two different techniques of error propagation, 1. an analytic approach 2. Monte Carlo approach.

For analytic approach, analytic relations between error and effect are assessed and propagated. While for Monte Carlo based approach, N trials are carried out with variations in error assumptions. The errors are defined within a confidence interval and randomly generated, later they are propagated.

First the problem is defined in Section 2. Section 3 generates and verifies analytic error propagation. Section 4 evaluates the impact of systematic errors on analytic error propagation. Section 5 conducts a Monte Carlo error propagation and Section 6 evaluates the impact of systematic error on Monte Carlo based error propagation.

## 2 Problem Definition

Figure 1[1][Fig.5-11,pg.265] represents a sketch of the problem under consideration. The orbit altitude h, is considered to be 1000km with an elevation angle $\varepsilon$ of $21.6052°$ as seen from the target Earth. Flat Earth model is considered. $\theta$ defined the Nadir angle. The distance from sub-satellite point (SSP) to Target is defined as $D_{nom}$ and the distance from the satellite to the target is defined as D.

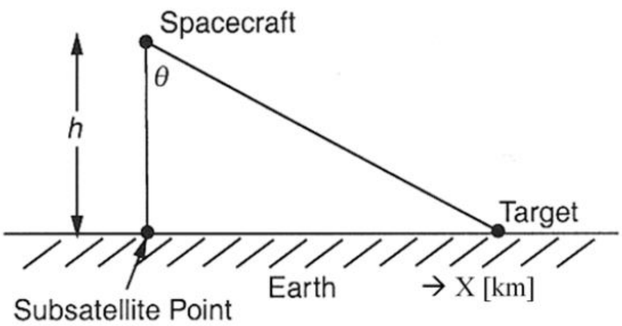


Figure 1: 2D overview of the problem under consideration.

Relations for error propagation are Tabulated with the aid of Figure2[1][Table5.7,pg.254]. Note the Nadir angle $\eta$ is considered to be equivalent to $\theta$ throughout this report.

**TABLE 5-7. Mapping and Pointing Error Formulas.** $\varepsilon$ is the observation angle of the spacecraft as seen from the target, *lat* is the latitude of the target, $\phi$ is the target azimuth relative to the ground track, $\lambda$ is the Earth central angle from the target to the satellite, $D$ is the distance from the satellite to the target, $R_T$ is the distance from the Earth's center to the target (typically ~ $R_E$, the Earth's radius), and $R_S$ is the distance from the Earth's center to the satellite. See Fig. 5-5.

| Error Source | Error Magnitude (units) | Magnitude of Mapping Error (km) | Magnitude of Pointing Error (rad) | Direction of Error |
|---|---|---|---|---|
| ***Attitude Errors:***(1) | | | | |
| *Azimuth* | $\Delta\phi$(rad) | $\Delta\phi D \sin\eta$ | $\Delta\phi \sin\eta$ | Azimuthal |
| *Nadir Angle* | $\Delta\eta$ (rad) | $\Delta\eta\ D / \sin\varepsilon$ | $\Delta\eta$ | Toward nadir |
| ***Position Errors:*** | | | | |
| *In-Track* | $\Delta I$ (km) | $\Delta I\ (R_T/R_S)\cos H^{(2)}$ | $(\Delta I/D)\sin Y_I^{(5)}$ | Parallel to ground track |
| *Cross-Track* | $\Delta C$ (km) | $\Delta C\ (R_T/R_S)\cos G^{(3)}$ | $(\Delta C/D)\sin Y_C^{(6)}$ | Perpendicular to ground track |
| *Radial* | $\Delta R_S$ (km) | $\Delta R_S\ \sin\eta / \sin\varepsilon$ | $(\Delta R_S/D)\sin\eta$ | Toward nadir |
| ***Other Errors:*** | | | | |
| *Target Altitude* | $\Delta R_T$ (km) | $\Delta R_T / \tan\varepsilon$ | — | Toward nadir |
| *S/C Clock* | $\Delta T$ (s) | $\Delta T\ V_e \cos(lat)^{(4)}$ | $\Delta T\ (V_e/D)\cos(lat) \cdot \sin J^{(7)}$ | Parallel to Earth's equator |

Notes:
(1) Includes attitude determination error, instrument mounting error, stability over exposure time (mapping only), and control error (pointing only). The formulas given assume that the attitude is measured with respect to the Earth.
(2) $\sin H = \sin\lambda \sin\phi$.
(3) $\sin G = \sin\lambda \cos\phi$.
(4) $V_e$ = 464 m/s (Earth rotation velocity at the equator).
(5) $\cos Y_I = \cos\phi \sin\eta$.
(6) $\cos Y_C = \sin\phi \sin\eta$.
(7) $\cos J = \cos\phi_E \cos\varepsilon$, where $\phi_E$ = azimuth relative to East.

Figure 2: Relations used for error propagation.

# 3 Analytic error propagation

The objective is to reconstruct the mapping budget contributions, tabulated with the aid of Figure3[1][Table 5.18, pg.277]. A script used to produce the results is presented in Section 7. First the distance from Spacecraft to the target is found with the aid of Equation 1.

$$D_{target} = \frac{h}{cos(\theta)} \tag{1}$$

Impact on mapping budget for errors associated with star sensor (measurement, mounting error and star catalogue accuracy), attitude computation error, payload sensor (measurement and mounting error), target centroiding error and transformation of target location to inertial co-ordinates are estimated with the aid of Equation 2. The results of this is tabulated with the aid of Table 1. Shaded boxes indicate the values which are not identical to the values presented in the literature[1]. However, the offset is quite small in order of 0.1m.

$$\text{impact on mapping budget} = \frac{\Delta \cdot D_{target}}{sin(\epsilon)} \quad [m] \tag{2}$$
$$\Delta = \text{Assumed error value in radians.}$$

**TABLE 5-18. Projection Error Budget.** See text for discussion. See Table 5-7 in Sec. 5.3 for formulas.

| Error Source | Assumed Error Value | Impact on Mapping Budget (m) | Comments |
|---|---|---|---|
| *Star Sensor Measurement Error* | 0.0015 deg | 193.1 | Typically larger error in rotation about sensor axis |
| *Star Sensor Mounting Error* | 0.0020 deg | 257.5 | May vary from sunlight to eclipse |
| *Star Catalog Accuracy* | 0.0001 deg | 12.9 | Have to eliminate double stars |
| *Attitude Computation Error* | 0.0001 deg | 12.9 | Computation errors typically small |
| *Payload Sensor Measurement Errors* | 0.0010 deg | 128.8 | Sum of many internal error sources |
| *Target Centroiding Error* | 0.0020 deg | 257.5 | Depends on size and shape of target |
| *Payload Sensor Mounting Error* | 0.0010 deg | 128.8 | Payload sensor opposite side of spacecraft from star sensor |
| *Transformation of Target Location to Inertial Coordinates* | 0.0001 deg | 12.9 | Only small arithmetic error (attitude errors accounted for above) |
| *Orbit Determination Error* | 100 m | 100.0 | Along-track component much larger than others |
| *Timing Error* | 50 ms | 367.5 | LEO spacecraft moving at 7.35 km/s (1,000 km altitude) |
| *Total Pointing Error With Respect to Nadir* | — | **590.5** | RSS of pointing components above; worst in radial direction |

Figure 3: Mapping budget to be reproduced and verified with analytic error propagation.

For orbit determination error the maximum among in-track, cross-track and along track is taken. Along track is larger than in-track/cross-track. In-track and cross-track error's impact on mapping budget is calculated with the aid of Equations ( in-track, cross-track) presented in Figure 2. The impact of timing error is calculated with the aid of Equation S/C clock presented in Figure 2. Ve is taken to be 7.5 km/s as indicated in Figure 3. While lattitude is taken to be 0, i.e. at equator.

Table 1: Reconstructed impact on mapping budget with the aid of analytic error propagation.

| Error Source | Impact on Mapping Budget [m] |
|---|---|
| Star Sensor Measurement | 193.09898046 |
| Star Sensor Mounting | 257.46530728 |
| Star Catalog Accuracy | 12.87326536 |
| Attitude Computation | 12.87326536 |
| Payload Sensor Measurement | 128.73265364 |
| Target Centroiding | 257.46530728 |
| Payload Sensor Mounting | 128.73265364 |
| Transformation of target location to inertial co-ordinates | 12.87326536 |
| Orbit determination | 100.00000000 |
| Timing | 367.50000000 |
| ***Total pointing error with respect to Nadir*** | 590.39104860 |

Assuming that all errors are random and completely uncorrelated, the total pointing error can be calculated by taking the ”Root Sum Square” RSS of all the individual error impacts. This can be mathematically represented with the aid of Equation3. Where N is the total number of errors analysed. Total pointing error wrt Nadir is outlined in Table 1.

$$\text{Total pointing error} = \sqrt{\sum_{i=1}^{N} \text{Error}_i^2} \tag{3}$$

From the above analysis it can be concluded that the analytic model for error propagation is verified to be in line with the results tabulated in literature (Figure 3). A copy of the script used to construct the above results can be found in Section 7.

# 4 Effect of systematic error on analytic solution

The objective of this Section is to analyse the impact of systematic error on the total pointing error. First the systematic error is generated, later they are propagated for the best case and the worst case scenarios.

An additional systematic error (i.e. constant) of 0.005 degree is introduced in the payload sensor mounting in addition to the random errors. This systematic error in the payload sensor mounting can be calculated with the aid of Equation 2. Next RSS of this systematic error is estimated with the aid of Equation 5.

Next two cases are evaluated. For Case 1, a pessimistic approach is taken. Sum of systematic and random errors are taken to be correlated and linear. In other words, RSS of total random pointing errors and systematic errors are added linearly. For Case 2, an optimistic approach is taken. Sum of systematic random errors are taken to be uncorrelated and RSS is conducted on the both the errors. The result of this is presented in Table 2.

Table 2: Optimistic and pessimistic impact of systematic error on the total mapping budget.

| **Error Type** | **Impact on Mapping Budget [m]** |
|---|---|
| RSS of random error | 590.3910 |
| RSS of systematic error | 643.6633 |
| ***Case 1*** | 1234.0543 |
| ***Case 2*** | 873.4209 |

From Table2 it can be concluded that in presence of systematic errors, the overall impact on the mapping budget increases, lesser for the optimistic case and more for the pessimistic case.

# 5 Monte Carlo based error propagation

Monte Carlo based error propagation can be conducted in different ways. One way is to randomise the assumed values of errors, within a desired confidence interval of standard deviation $\sigma$. ”N” numbers of data points are generated, from which the errors are randomly picked. The picked error values can then be used in the analytic model described in Section 3, to obtain the total pointing error and the total pointing error with systematic errors. However, one immediately realises that this method is not attractive (for the case of $1\sigma$ of random errors) for two reasons, 1. the computational effort is N times more than analytic solution, ignoring the computation power consumed in generating N random numbers. 2. The value of total pointing error will in general be smaller and at best equivalent to the analytic solution. There is no point in undertaking this method unless the $\sigma$ is increased to higher intervals i.e. $\geq 2\sigma$. A copy of this method is presented in the script for the curious readers to try out.

However, another approach which will be presented in the following sections, which is rather attractive for the $1\sigma$ case. The objective is to use the nominal model and calculate the nominal target location as indicated in Figure1. Next the errors are introduced in the model and the corresponding target location is updated. The process is repeated N times, where at every instance a random value of the error is picked in the interval $1\sigma$ of error. The difference between the nominal and the new location is stored for every run. And the end of simulation, the mean and rms of the differences is analysed to provide insight into the impact of total pointing error (wrt Nadir) on the mapping budget. Equation 5 outlines the nominal target solution. While, Equation **??** outlines the target location for a new random simulation. Where $\Delta$ indicated the random errors introduced, ss indicates star sensor, sc = star catalogue,att = attitude compensation, ps = payload sensor, tc = target centroiding, tran = transformation of target location to inertial co-ordinates, orb = orbit determination error and time = timing error.

$$x_{\text{nominal target}} = h * tan(\theta) \tag{4}$$

$$x_{\text{updated target}} = h \cdot tan(\theta + \Delta_{ss\_meas} + \Delta_{ss\_mount} + \Delta_{sc\_acc} + \Delta_{att} + \Delta_{ps\_meas} + \Delta_{ps\_mount} + \Delta_{tc} + \Delta_{tran}) + \Delta x_{orb} + \Delta x_{time} \tag{5}$$

The simulation is run for 3 times for 4 different data point sizes ( 10, 1000, 10000, 100000). Matlab command ”*randn*” is used to generate the data points. Standard deviation of $1\sigma$ is used. Expressions for $\Delta x_{orb}$ and $\Delta x_{time}$ are the same as the analytic case and can be found in Figure 2. The impact of total random pointing error on the mapping budget is outlined in Table 3. For increasing data points, it can be observed that the rms of total impact on mapping budget $\Delta x$ is getting closer to the analytic total impact on mapping budget as outlined in Table1.

Table 3: Monte Carlo simulation of total impact on mapping budget $\Delta x$

| $\Delta x$ | **Run 1** | | **Run 2** | | **Run 3** | |
|---|---|---|---|---|---|---|
| **Datapoint** | mean [m] | rms[m] | mean [m] | rms [m] | mean [m] | rms [m] |
| 100 | 0.1903 | 548.2911 | 40.1289 | 648.7165 | -69.4567 | 595.3122 |
| 1000 | 28.7337 | 585.8341 | -5.9516 | 587.3025 | -17.0918 | 565.3290 |
| 10000 | -0.2170 | 591.0601 | -2.5325 | 582.1630 | -4.0780 | 593.4750 |
| 100000 | 2.5344 | 590.5176 | -0.8578 | 589.0559 | 1.2926 | 590.8461 |

# 6 Effect of systematic error on Monte Carlo solution

In case of a constant systematic error on the payload mounting, a new ”constant” term $\Delta_{systematic}$ = 0.005 deg is added to the model. This is indicated with the aid of Equation 6.

$$x_{\text{updated target}} = h * tan(\theta + \Delta_{systematic} + \Delta_{ss\_meas} + \Delta_{ss\_mount} + \Delta_{sc\_acc} + \Delta_{att} + \Delta_{ps\_meas} + \Delta_{ps\_mount} + \Delta_{tc} + \Delta_{tran})\ldots$$
$$+ \Delta x_{orb} + \Delta x_{time} \tag{6}$$

Simulation is run 3 times, for 4 different data point sizes and the results are outlined in Table 4. An interesting observation is that with increasing data points the total mapping error budget reaches the optimistic case (case 2) of analytic solution as seen in Table 2.

Table 4: Impact of systematic error on the mapping error budget with Monte Carlo.

| $\Delta x$ | **Run 1** | | **Run 2** | | **Run 3** | |
|---|---|---|---|---|---|---|
| **Datapoint** | mean [m] | rms[m] | mean [m] | rms [m] | mean [m] | rms [m] |
| 100 | 602.9215 | 856.1260 | 780.3592 | 962.4202 | 592.8838 | 881.0682 |
| 1000 | 653.8683 | 873.6976 | 628.7222 | 861.1940 | 662.6669 | 891.2925 |
| 10000 | 647.0354 | 869.1803 | 643.0922 | 876.2583 | 653.3000 | 881.3668 |
| 100000 | 640.5384 | 870.1111 | 645.5183 | 875.2766 | 645.8447 | 875.5356 |
| 1000000 | 642.6453 | 872.3490 | 644.1882 | 873.6003 | 643.7526 | 873.8586 |

What can be concluded after all this analysis? It can be concluded that Monte Carlo simulation provides am alluring alternative to analytic approach. Especially for larger systems involving high number of parameters, Monte Carlo can be used to evaluate one simple equation for N data points (very attractive from vectorization perspective, can be solved in one step) rather one evaluating one equation for every single parameter involved.

# 7 Matlab Scipt

Following script is used to generate the above presented results.

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% AE4878 Mission Geometry and Orbit Design %%%%%%%%%%%
%%%%% Assignment 6 - Week 4.2 - DESIGN 7 - Version 2 %%%%%
%%%%% Author Info: Ali Nawaz; Student ID - 4276477 %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Parameters
h = 1000; % altitude [km]
elevd = 21.6052; %elevation angle [deg]
elevr = deg2rad(21.6052); % elevation angle [rad]
theta = pi - pi/2 - elevr; % Theta/Eta Nadir angle [rad]
lambda = pi/2 - elevr - theta; % Earth central angle [rad] Eqn-9.6 OCDM
phi = 0; % target azimuth relative to groundtrack [rad]
H = asin(sin(lambda)*sin(phi)); % H angle [rad]
G = asin(sin(lambda)*cos(phi)); % G angle [rad]
Dtnom = h*tan(theta); % Nominal target distance from SSP [km]
Dnom = h/cos(theta); % Nominal distance from S/C to Target [km]
RE = 6378.136; % Radius of Earth [km], taken from OCDM/intro lecture slides
RT = RE; % Distance from Earth's center to target [km]
RS = RE+h; % Distance from Earth's center to the satellite [km]

%% Part 1: Assumed error values and corresponding impact on mapping budget [m] - Analytical ...
    Approach
ss_meas_err = deg2rad(0.0015) ; %Star Sensor (SS) measurement error [rad]
ss_meas_map = (ss_meas_err*(Dnom/ (sin(elevr)) ))*10^3; % Impact on mapping budget[m]

ss_mount_err = deg2rad(0.0020); % SS mounting error [rad]
ss_mount_map = (ss_mount_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

ss_cata_acc = deg2rad(0.0001); % SS catalogue accuracy [rad]
ss_cata_map = (ss_cata_acc*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

att_comp_err = deg2rad(0.0001); % attitude computation error [rad]
att_comp_map = (att_comp_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

ps_meas_err = deg2rad(0.0010); % Payload sensor mounting error [rad]
ps_meas_map = (ps_meas_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

targ_cent_err = deg2rad(0.0020); % Target centroiding error [rad]
targ_cent_map = (targ_cent_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

ps_mount_err = deg2rad(0.0010); % Payload sensor mounting error [rad]
ps_mount_map = (ps_mount_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

trans_err = deg2rad(0.0001); % Transformation of target location to inertial co-ordinates [rad]
trans_map = (trans_err*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

orb_det_err = 100*10^(-3); % Orbit determination error [km]
orb_det_map_rad = orb_det_err*(sin(theta)/sin(elevr))*10^(3); % Radial impact on mapping ...
    budget [m]
orb_det_map_int = orb_det_err*(RT/RS)*cos(H)*10^(3); % Intrack impact on error budget [m] ...
    displacement in the direction of the velocity vector
orb_det_map_ct = orb_det_err*(RT/RS)*cos(G)*10^(3); % Cross track impact on error budget [m]
orb_det_map_along = orb_det_err*10^(3); % Along track impact on error budget [m] transverse ...
    displacement normal to the position vector.


timing_err = 50*10^(-3); % Timing error [s]
Ve = 7.35*10^(3); % [m/s] Velocity at 1000 km altitude
lat = deg2rad(0); % Lattitude at equator[rad]
timing_map = timing_err*Ve*cos(lat); % Impact of timing error on mapping budget [m]

% total pointing error impact on budget wrt nadir, random and uncorrelated errors i.e. RSS
total_pointing_error = sqrt( ss_meas_map^2 + ss_mount_map^2 + ss_cata_map^2 + att_comp_map^2 ...
    + ps_meas_map^2 + targ_cent_map^2 + ps_mount_map^2 + trans_map^2 + orb_det_map_along^2 + ...
    timing_map^2); %[m]

% Tabulated mapping error/ impact on budget [m]
tab_map_part1 = [ss_meas_map, ss_mount_map, ss_cata_map, att_comp_map, ps_meas_map, ...
    targ_cent_map, ps_mount_map, trans_map, orb_det_map_along, timing_map];

%% Part 2: Addition of systematic payload mounting error

% Additional systematic error for payload mounting
ps_mount_err_sys = deg2rad(0.005); % Payload sensor mounting error [rad]
ps_mount_map_sys = (ps_mount_err_sys*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

```

```
rss_sys = sqrt(ps_mount_map_sys^2); % RSS of systematic error(s)

% Case 1: Sum of random and systematic errors (correlated)(linear) -
% Pessimistic
case1_error = total_pointing_error + rss_sys; % [m]

% Case 2: Sum of random and systematic errors (uncorrelated)(RSS) -
% Optimistic
case2_error = sqrt(total_pointing_error^2 + rss_sys^2); %[m]

%% PART 3  [ Monte Carlo Extension to Analytic Approach ] - Assumed error values and ...
    corresponding impact on mapping budget [m]
datapoints = 1000000; % no. of random values generated
lb = 0; % Lower bound on error

% Generate random normally distributed
val = randn(datapoints,1);
mss_meas_err = deg2rad(0.0015) ; %Star Sensor (SS) measurement error [rad]
val1 = val*mss_meas_err; % Random errors generated for pure monte carlo approach in part 5
mss_meas_map = (mss_meas_err*val*(Dnom/ (sin(elevr)) ))*10^3; % Impact on mapping budget[m]

val = randn(datapoints,1);
mss_mount_err = deg2rad(0.0020); % SS mounting error [rad]
val2 = val*mss_mount_err;% Random errors generated for pure monte carlo approach in part 5
mss_mount_map = (mss_mount_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
mss_cata_acc = deg2rad(0.0001); % SS catalogue accuracy [rad]
val3 = val*mss_cata_acc; % Random errors generated for pure monte carlo approach in part 5
mss_cata_map = (mss_cata_acc*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
matt_comp_err = deg2rad(0.0001); % attitude computation error [rad]
val4 = val*matt_comp_err;% Random errors generated for pure monte carlo approach in part 5
matt_comp_map = (matt_comp_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
mps_meas_err = deg2rad(0.0010); % Payload sensor mounting error [rad]
val5 = val*mps_meas_err;% Random errors generated for pure monte carlo approach in part 5
mps_meas_map = (mps_meas_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
mtarg_cent_err = deg2rad(0.0020); % Target centroiding error [rad]
val6 = val*mtarg_cent_err;% Random errors generated for pure monte carlo approach in part 5
mtarg_cent_map = (mtarg_cent_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
mps_mount_err = deg2rad(0.0010); % Payload sensor mounting error [rad]
val7 = val*mps_mount_err;% Random errors generated for pure monte carlo approach in part 5
mps_mount_map = (mps_mount_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
mtrans_err = deg2rad(0.0001); % Transformation of target location to inertial co-ordinates [rad]
val8 = val*mtrans_err;% Random errors generated for pure monte carlo approach in part 5
mtrans_map = (mtrans_err*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping budget [m]

val = randn(datapoints,1);
morb_det_err = 100*10^(-3); % Orbit determination error [km]
morb_det_map_rad = morb_det_err*val*(sin(theta)/sin(elevr))*10^(3); % Radial impact on ...
    mapping budget [m]
morb_det_map_int = morb_det_err*val*(RT/RS)*cos(H)*10^(3); % Intrack impact on error budget ...
    [m] displacement in the direction of the velocity vector
morb_det_map_ct = morb_det_err*val*(RT/RS)*cos(G)*10^(3); % Cross track impact on error ...
    budget [m]
morb_det_map_along = (-morb_det_err + 2*morb_det_err*val); % Along track impact on error ...
    budget [m] transverse displacement normal to the position vector.
morb_det_map_along = morb_det_err*val*10^(3); % Along track impact on error budget [m] ...
    transverse displacement normal to the position vector.
val9 = morb_det_map_along; % Random errors generated for pure monte carlo approach in part 5

val = randn(datapoints,1);
mtiming_err = 50*10^(-3); % Timing error [s]
Ve = 7.35*10^(3); % [m/s] Velocity at 1000 km altitude
lat = deg2rad(0); % Lattitude at equator[rad]
val10 = val*mtiming_err;% Random errors generated for pure monte carlo approach in part 5
mtiming_map = mtiming_err*val*Ve*cos(lat); % Impact of timing error on mapping budget [m]
```

```
% total pointing error with monte carlo approach to analytic solution
mtotal_pointing_error = sqrt( mss_meas_map.^2 + mss_mount_map.^2 + mss_cata_map.^2 + ...
    matt_comp_map.^2 + mps_meas_map.^2 + mtarg_cent_map.^2 + mps_mount_map.^2 + mtrans_map.^2 ...
    + morb_det_map_along.^2 + mtiming_map.^2); %[m]
% mean and rms of corresponding total pointing error
mean_mtotal_pointing_error = mean(mtotal_pointing_error); % mean of total pointing error
rms_mtotal_pointing_error = rms(mtotal_pointing_error); % rms of total pointing error

Visualising the behaviour of Monte Carlo generated pointing error
figure(1)
mean_list = (mean_mtotal_pointing_error*ones(length(mtotal_pointing_error)));
rms_list = (rms_mtotal_pointing_error*ones(length(mtotal_pointing_error)));
plot(1:length(mtotal_pointing_error), ...
    mtotal_pointing_error,'ro',1:length(mtotal_pointing_error),mean_list,'b-', ...
    1:length(mtotal_pointing_error),rms_list,'k--');
title('Total pointing error behaviour for different Monte Carlo runs');
legend('Total pointing error','Mean','RMS');
grid on
xlabel('Simulation no.')
ylabel('Total pointing error [m]')
% Storing individual impact on error budget
tab_map_part2 = [mss_meas_map, mss_mount_map, mss_cata_map, matt_comp_map, mps_meas_map, ...
    mtarg_cent_map, mps_mount_map, mtrans_map, morb_det_map_along, mtiming_map];

%% PART 4 - [ Monte Carlo Extension to Analytic Approach ] Additional systematic error for ...
    payload mounting
val = randn(datapoints,1);
mps_mount_err_sys = deg2rad(0.005); % Payload sensor mounting error [rad]
mps_mount_map_sys = (mps_mount_err_sys*val*(Dnom/ (sin(elevr)) ))*10^3;% Impact on mapping ...
    budget [m]

mrss_sys = sqrt(mps_mount_map_sys.^2); % RSS of systematic error(s)

% Case 1: Sum of random and systematic errors (correlated)(linear) -
% Pessimistic
mcase1_error = mtotal_pointing_error + mrss_sys; % [m]

% Case 2: Sum of random and systematic errors (uncorrelated)(RSS) -
% Optimistic
mcase2_error = sqrt(mtotal_pointing_error.^2 + mrss_sys.^2); % [m]

%% Part 5: Pure Monte Carlo Approach
x_nom = h*tan(theta)*10^(3); % Nominal value of target location from SSP [m]

% Updated target value with random error selection
x_target_new = h*tan(theta + val1+ val2 + val3 + val4 + val5 + val6 + val7 + val8)*10^(3) + ...
    val9 + mtiming_map;
%Updated target value with random and systematic error selection
x_target_new_sys = h*tan(theta + val1+ val2 + val3 + val4 + val5 + val6 + val7 + val8 + ...
    mps_mount_err_sys)*10^(3) + val9 + mtiming_map;

% differences between target and nominal models
diff = x_target_new - x_nom; % random error case
diff_sys = x_target_new_sys - x_nom; % random error and systematic error case

% mean and rms of impact on budget for random error case
mean_diff = mean(diff);
rms_diff = rms(diff);

% mean and rms of impact on budget for combined random and systematic error
% case
mean_diff_sys = mean(diff_sys)
rms_diff_sys = rms(diff_sys)

%%%%% END OF SCRIPT %%%%%
```

# References


[1] James R. Wertz. *Orbit Constellation Design Management*. Springer, 2009.