Assignment 02 - Navigation

Spacecraft Guidance and Navigation

AY 2023-2024

POLITECNICO
MILANO 1863

Exam

➢ **Remarks**:

- The deadline for the submission of assignment 2 is on **22/12/2023**.
  - ➢ An extension of a few days is possible only for well motivated issues.
- Those who will not be able to submit within the deadline will perform the exam on the following exam date (currently on January 30).
  - ➢ The deadline for taking part to this second session is on **10/01/2024** for both Assignment 1 and 2.

Laboratory sessions

- Laboratory sessions **will not be recorded**
  - we are here to give you answers while you are working
- When writing in the forum:
  - Reply to the proper thread
  - Check if your question was already posted
  - Be patient (do not re-ask a question that still has to receive an answer)
  - Do **not** attach/send code asking for debugging

Timetable		Room
29 Nov	14.15-16.15	BL.27.05 (+ Bonaccorsi virtual room)
01 Dec	09.15-12.15	LM.3 (+ Bonaccorsi virtual room)
05 Dec	16.15-18.15	B8.07 (+ Bonaccorsi virtual room)
06 Dec	14.15-16.15	BL.27.05 (+ Bonaccorsi virtual room)
12 Dec	16.15-18.15	B8.07 (+ Bonaccorsi virtual room)
13 Dec	14.15-16.15	BL.27.05 (+ Bonaccorsi virtual room)
22 Dec	**23.30 CET**	**Assignment 2 deadline**

Delivery of assignments

Assigment will be delivered through **WeBeep**:

DEADLINE ⇨ Dec 22, 2023, 23:30:00 CET

1) Click on the link to **load Assignment 2** in your Overleaf

   https://bit.ly/SGN_23_Assignment2

2) **Fill the report** and be sure it is compiled properly
3) **Download the PDF** and merge it in a **zipped file** with MATLAB code.
   Rename it `lastname123456_Assign2.zip`
4) **Submit the compressed file** by uploading it on Webeep
   - **Max Size**: 10 Mb

Do **NOT** put spaces in file names

Assignment 02 – Topics

3 Exercises

- Uncertainty propagation
- Batch filters
- Sequential filters

Suggested MATLAB Version: R2021b (or newer)

Report

1 Impulsive guidance

Exercise 1

Let $\mathbf{x}(t) = \varphi(\mathbf{x}_0, t_0; t)$ be the flow of the geocentric two-body model. 1) Using one of Matlab's built-in integrators, implement and validate a propagator that returns $\mathbf{x}(t)$ for given $\mathbf{x}_0$, t_0, t, and μ. 2) Given the pairs $\{\mathbf{r}_1, \mathbf{r}_2\}$ and $\{t_1, t_2\}$, develop a solver that finds $\mathbf{v}_1$ such that $\mathbf{r}(t_2) = \mathbf{r}_2$, where $(\mathbf{r}(t), \mathbf{v}(t))^\top = \varphi((\mathbf{r}_1, \mathbf{v}_1)^\top, t_1; t_2)$ (Lambert's problem). To compute the derivatives of the shooting function, use either a) finite differences or b) the state transition matrix $\Phi = \mathrm{d}\varphi/\mathrm{d}\mathbf{x}_0$. Validate the algorithms against the classic Lambert solver. 3) Using the propagator of point 1) in the heliocentric case, and reading the motion of the Earth and Mars from SPICE, solve the shooting problem

$$\min_{\mathbf{x}_1, t_1, t_2} \Delta v \quad \text{s.t.} \quad \begin{cases} \mathbf{r}_1 = \mathbf{r}_E(t_1) \\ \mathbf{r}(t_2) = \mathbf{r}_M(t_2) \\ t_1^L \le t_1 \le t_1^U \\ t_2^L \le t_2 \le t_2^U \\ t_2 \ge t_1 \end{cases} \tag{1}$$

where $\Delta v = \Delta v_1 + \Delta v_2$, $\Delta \mathbf{v}_1 = \mathbf{v}_1 - \mathbf{v}_E(t_1)$, $\Delta \mathbf{v}_2 = \mathbf{v}(t_2) - \mathbf{v}_M(t_2)$. $\mathbf{x}_1 = (\mathbf{r}_1, \mathbf{v}_1)^\top$, and $(\mathbf{r}(t), \mathbf{v}(t))^\top = \varphi(\mathbf{x}_1, t_1; t_2)$. Define lower and upper bounds, and make sure to solve the problem stated in Eq. (1) for different initial guesses.

Write your answer here

- Develop the exercises in one Matlab script; name the file `lastname123456_Assign1.m`
- Organize the script in sections, one for each exercise; use local functions if needed.
- Download the PDF from the Main menu.
- Create a single .zip file containing both the report in PDF and the MATLAB file. The name shall be `lastname123456_Assign1.zip`.
- Red text indicates where answers are needed; be sure there is no red stuff in your report.
- In your answers, be concise: to the point.
- **Deadline for the submission: Nov 11 2021, 23:30.**
- **Load the compressed file to the Assignments folder on Webeep.**

Exercise

Answer here

NB: The code is not your report!

Answer the question **in the report** and add any plot you think is relevant for your answers there.

Any description or result missing in the report but present in the code **will not contribute** to the evaluation!

Script

- Name of the script (one per exercise): `lastname123456_Assign2_Ex1.m`
- Header:

```
% Spacecraft Guidance and Navigation (2023/2024)
% Assignment # 2, Exercise 1
% Author: Name Lastname
```

- Functions in a section at the end of the script: `%% Functions`

Exercise 1 – Uncertainty propagation

Exercise 1: Uncertainty propagation

The Prototype Research Instruments and Space Mission Technology Advancement (PRISMA) is a technology in-orbit test-bed mission for demonstrating Formation Flying (FF) and rendezvous technologies, as well as flight testing of new sensors and actuator equipment. It was launched on June 15, 2010, and it involves two satellites: Mango (Satellite 1, ID 36599), the chaser, and Tango (Satellite 2, ID 36827), the target.

You have been provided with an estimate of the states of Satellites 1 and 2 at the separation epoch t_{sep} = `2010-08-12T05:27:39.114` (UTC) in terms of mean and covariance, as reported in Table 1. Assume Keplerian motion can be used to model the spacecraft dynamics.

1. Propagate the initial mean and covariance for both satellites within a time grid going from t_{sep} to $t_{sep} + N\,T_1$, with a step equal to T_1, where T_1 is the orbital period of satellite 1 and $N = 10$, using both a Linearized Approach (LinCov) and the Unscented Transform (UT). We suggest to use $\alpha = 0.1$ and $\beta = 2$ for tuning the UT in this case.

Parameter	Value
Ref. epoch t_{sep} [UTC]	2010-08-12T05:27:39.114
Mean state $\hat{\boldsymbol{x}}_{0,\text{sat1}}$ [km, km/s]	$\hat{\boldsymbol{r}}_{0,\text{sat1}} = [4622.232026629,\ 5399.3369588058,\ -0.0212138165769957]$ $\hat{\boldsymbol{v}}_{0,\text{sat1}} = [0.812221125483763,\ -0.721512914578826,\ 7.42665302729053]$
Mean state $\hat{\boldsymbol{x}}_{0,\text{sat2}}$ [km, km/s]	$\hat{\boldsymbol{r}}_{0,\text{sat2}} = [4621.69343340281,\ 5399.26386352847,\ -3.09039248714313]$ $\hat{\boldsymbol{v}}_{0,\text{sat2}} = [0.813960847513811,\ -0.719449862738607,\ 7.42706066911294]$
Covariance P_0 [km^2, km^2/s, km^2/s^2]	$\begin{bmatrix} +5.6e-7 & +3.5e-7 & -7.1e-8 & 0 & 0 & 0 \\ +3.5e-7 & +9.7e-7 & +7.6e-8 & 0 & 0 & 0 \\ -7.1e-8 & +7.6e-8 & +8.1e-8 & 0 & 0 & 0 \\ 0 & 0 & 0 & +2.8e-11 & 0 & 0 \\ 0 & 0 & 0 & 0 & +2.7e-11 & 0 \\ 0 & 0 & 0 & 0 & 0 & +9.6e-12 \end{bmatrix}$

2. Considering that the two satellites are in close formation, you have to guarantee a sufficient accuracy about the knowledge of their state over time to monitor potential risky situations. For this reason, at each revolution, you shall compute:
   - the norm of the relative position (Δr), and
   - the sum of the two covariances associated to the position elements of the states of the two satellites (P_{sum})

   The critical conditions which triggers a collision warning is defined by the following relationship:
   $$\Delta r < 3\sqrt{\max(\lambda_i(P_{\text{sum}}))}$$
   where $\lambda_i(P_{\text{sum}})$ are the eigenvalues of P_{sum}. Identify the revolution N_c at which this condition occurs and elaborate on the results and the differences between the two approaches (UT and LinCov).

3. Perform the same uncertainty propagation process on the same time grid using a Monte Carlo (MC) simulation *. Compute the sample mean and sample covariance and compare them with the estimates obtained at Point 1). Provide the plots of:
   - the time evolution for all three approaches (MC, LinCov, and UT) of $3\sqrt{\max(\lambda_i(P_{r,\text{i}}))}$ and $3\sqrt{\max(\lambda_i(P_{v,\text{i}}))}$, where $i = 1, 2$ is the satellite number and P_r and P_v are the 3x3 position and velocity covariance submatrices.
   - the propagated samples of the MC simulation, together with the mean and covariance obtained with all methods, projected on the orbital plane.

   Compare the results and discuss on the validity of the linear and Guassian assumption for uncertainty propagation.

Exercise 2 – Batch filters

Exercise 2: Batch filters

You have been asked to track Mango to improve the accuracy of its state estimate. To this aim, you shall schedule the observations from the two ground stations reported in Table 2.

1. *Compute visibility windows.* By using the mean state reported in Table 1 and by assuming Keplerian motion, predict the trajectory of the satellite over a uniform time grid (with a time step of 60 seconds) and compute all the visibility time windows from the available stations in the time interval from t_0 = `2010-08-12T05:30:00.000` (UTC) to t_f = `2022-11-2010-08-12T11:00:00.000` (UTC). Plot the resulting predicted Azimuth and Elevation profiles in the visibility windows.

2. *Simulate measurements.* The Two-Line Elements (TLE) set of Mango are reported in Table 3 (and in WeBeep as 36599.3le). Use SGP4 and the provided TLEs to simulate the measurements acquired by the sensor network in Table 2 by:

   (a) Computing the spacecraft position over the visibility windows identified in Point 1 and deriving the associated expected measurements.

   (b) Simulating the measurements by adding a random error to the expected measurements (assume a Gaussian model to generate the random error, with noise provided in Table 2). Discard any measurements (i.e., after applying the noise) that does not fulfill the visibility condition for the considered station.

3. *Solve the navigation problem.* Using the measurements simulated at the previous point:

   (a) Find the least squares (minimum variance) solution to the navigation problem without a priori information using

   - the epoch t_0 as reference epoch;
   - the reference state as the state derived from the TLE set in Table 3 at the reference epoch;
   - the simulated measurements obtained for the KOROU ground station only;
   - pure Keplerian motion to model the spacecraft dynamics.

   (b) Repeat step 3a by using all simulated measurements from both ground stations.

   (c) Repeat step 3b by using J2-perturbed motion to model the spacecraft dynamics.

4. Provide the obtained navigation solutions and elaborate on the results, comparing the different solutions.

5. Select the best combination of dynamical model and ground stations and perform the orbit determination for the other satellite.

Table 3: TLE of Mango.

```
1 36599U 10028B   10224.22752732 -.00000576  00000-0 -16475-3 0  9998
2 36599 098.2803 049.5758 0043871 021.7908 338.5082 14.40871350  8293
```

Table 4: TLE of Tango.

```
1 36827U 10028F   10224.22753605  .00278492  00000-0  82287-1 0  9996
2 36827 098.2797 049.5751 0044602 022.4408 337.8871 14.40890217    55
```

! Do not copy-paste from PDF

Exercise 2 – Batch filters

Table 2: Sensor network to track Mango and Tango: list of stations, including their features.

Station name	KOUROU	SVALBARD
Coordinates	LAT = 5.25144° LON = -52.80466° ALT = -14.67 m	LAT = 78.229772° LON = 15.407786° ALT = 458 m
Type	Radar (monostatic)	Radar (monostatic)
Provided measurements	Az, El [deg] Range (one-way) [km]	Az, El [deg] Range (one-way) [km]
Measurements noise (diagonal noise matrix R)	$\sigma_{Az,El}$ = 100 mdeg σ_{range} = 0.01 km	$\sigma_{Az,El}$ = 125 mdeg σ_{range} = 0.01 km
Minimum elevation	10 deg	5 deg

Exercise 3 – Sequential filters

Exercise 3: Sequential filters

According to the Formation Flying In Orbit Ranging Demonstration experiment (FFIORD), PRISMA's primary objectives include testing and validation of GNC hardware, software, and algorithms for autonomous formation flying, proximity operations, and final approach and recede operations. The cornerstone of FFIORD is a Formation Flying Radio Frequency (FFRF) metrology subsystem designed for future outer space formation flying missions.

FFRF subsystem is in charge of the relative positioning of 2 to 4 satellites flying in formation. Each spacecraft produces relative position, velocity and line-of-sight (LOS) of all its companions.

You have been asked to track Mango to improve the accuracy of the estimate of its absolute state and then, according to the objectives of the PRISMA mission, validate the autonomous formation flying navigation operations by estimating the relative state between Mango and Tango by exploiting the relative measurements acquired by the FFRF subsystem. The Two-Line Elements (TLE) set of Mango and Tango satellites are reported in Tables 3 and 4 (and in WeBeep as 36599.3le, and 36827.3le).

The relative motion between the two satellites can be modelled through the linear, Clohessy-Wiltshire (CW) equations[†]

$$\begin{aligned} \ddot{x} &= 3n^2 x + 2n\dot{y} \\ \ddot{y} &= -2n\dot{x} \\ \ddot{z} &= -n^2 z \end{aligned} \tag{1}$$

where x, y,and z are the relative position components expressed in the LVLH frame, whereas n is the mean motion of Mango, which is assumed to be constant and equal to:

$$n = \sqrt{\frac{GM}{R^3}} \tag{2}$$

where R is the position of Mango at t_0.

The unit vectors of the LVLH reference frame are defined as follows:

$$\hat{\boldsymbol{i}} = \frac{\boldsymbol{r}}{r}, \quad \hat{\boldsymbol{j}} = \hat{\boldsymbol{k}} \times \hat{\boldsymbol{i}}, \quad \hat{\boldsymbol{k}} = \frac{\boldsymbol{h}}{h} = \frac{\boldsymbol{r} \times \boldsymbol{v}}{\|\boldsymbol{r} \times \boldsymbol{v}\|} \tag{3}$$

To perform the requested tasks you should:

1. *Estimate Mango absolute state.* You are asked to develop a sequential filter to narrow down the uncertainty on the knowledge of Mango absolute state vector. To this aim, you shall schedule the observations from the SVALBARD ground station[‡] reported in Table 2, and then proceed with the state estimation procedure by following these steps:

   (a) By using the mean state reported in Table 1 and by assuming Keplerian motion, predict the trajectory of the satellite over a uniform time grid (with a time step of 5 seconds) and compute the first visibility time window from the SVALBARD station in the time interval from t_0 = `2010-08-12T05:30:00.000` (UTC) to t_f = `2022-11-2010-08-12T06:30:00.000` (UTC).

   (b) Use SGP4 and the provided TLE to simulate the measurements acquired by the SVALBARD station for the Mango satellite only. For doing it, compute the spacecraft position over the visibility window using a time-step of 5 seconds, and derive the associated expected measurements. Finally, simulate the measurements by adding a random error (assume a Gaussian model to generate the random error, with noise provided in Table 2).

[†]Notice that the system is linear, therefore it has an analytic solution of the state transition matrix $\boldsymbol{\Phi}$

[‡]Note that these are the same ones computed in Exercise 2

Exercise 3 – Sequential filters

(c) Using an Unscented Kalman Filter (UKF), provide an estimate of the spacecraft state (in terms of mean and covariance) by sequentially processing the acquired measurements in chronological order. Plot the time evolution of the error estimate together with the 3σ of the estimated covariance for both position and velocity.

2. *Estimate the relative state.* To validate the formation flying operations, you are also asked to develop a sequential filter to narrow down the uncertainty on the knowledge of the relative state vector. To this aim, you can exploit the relative azimuth, elevation, and range measurements obtained by the FFRF subsystem, whose features are reported in Table 5, and then proceed with the state estimation procedure by following these steps:

   (a) Use SGP4 and the provided TLEs to propagate the states of both satellites at epoch t_0 in order to compute the relative state in LVLH frame at that specific epoch.

   (b) Use the relative state as initial condition to integrate the CW equations over the time grid defined in Point 1a. Finally, simulate the relative measurements acquired by the Mango satellite through its FFRF subsystem by adding a random error to the expected measurements. Assume a Gaussian model to generate the random error, with noise provided in Table 5.

   (c) Consider a time interval of 20 minutes starting from the first epoch after the visibility window (always with a time step of 5 seconds). Use an UKF to provide an estimate of the spacecraft relative state in the LVLH reference frame (in terms of mean and covariance) by sequentially processing the measurements acquired during those time instants in chronological order. Plot the time evolution of the error estimate together with the 3σ of the estimated covariance for both relative position and velocity.

3. *Reconstruct Tango absolute covariance.* Starting from the knowledge of the estimated covariance of the absolute state of Mango, computed in Point 1, and the estimated covariance of the relative state in the LVLH frame, you are asked to provide an estimate of the covariance of the absolute state of Tango. You can perform this operation as follows:

   (a) Pick the estimated covariance of the absolute state of Mango at the last epoch of the visibility window, and propagate it within the time grid defined in Point 2c.

   (b) Rotate the estimated covariance of the relative state from the LVLH reference frame to the ECI one within the same time grid.

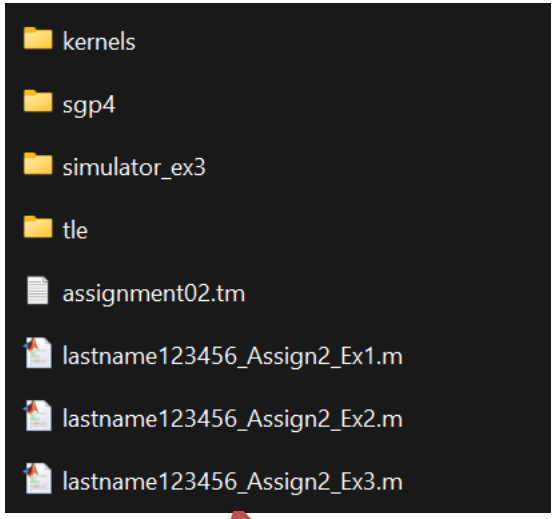
   (c) Sum the two to obtain an estimate of the covariance of the absolute state of Tango. Plot the time evolution of the 3σ for both position and velocity and elaborate on the results.

Table 5: Parameters of FFRF.

Parameter	Value
Measurements noise $\sigma_{Az,El}$	1 deg
(diagonal noise matrix R) σ_{range}	1 cm

Supporting material

- Available on **WeBeep**, inside the folder Laboratories
- Compressed folder **Assignment02.zip** contains:
  - Subfolder **`kernels`** with all required SPICE kernels
  - Subfolder **`sgp4`** with all sgp4 functions (Lab04)
  - Subfolder **`tle`** with two-line elements
  - Meta-kernel **`assignment02.tm`**



lastname123456_Assign2_Ex1.m

lastname123456_Assign2_Ex2.m

lastname123456_Assign2_Ex3.m

How to prepare your script

- Unzip the folder and work on your Matlab script inside it
- Remember to **load the meta-kernel** and to **add the path to sgp4 subfolder** in your script
  - Suggestion: use relative paths
- **NB:** do not to upload the supporting material when preparing your delivery on WeBeep, we have it.

A quick recap (1) – Orbit propagator with J2

NB: The J2 perturbation is **positional** and depends on the satellite position in **ECEF** frame.

1. Compute the matrix to convert position from ECI to ECEF

   rotm = cspice_pxform('J2000', 'ITRF93', et);

2. Convert the ECI position in ECEF

   pos_ECEF = rotm * pos_ECI;

3. Compute the J2 acceleration ($J_2 = 0.0010826269$) in ECEF:

$$\boldsymbol{a}_{J_2} = \frac{3}{2}\mu J_2 \frac{\boldsymbol{r}}{r^3}\left(\frac{R_E}{r}\right)^2\left(5\left(\frac{Z}{r}\right)^2 - \begin{Bmatrix}1\\1\\3\end{Bmatrix}\right)$$

4. Convert it to ECI
5. Add it to the Keplerian acceleration

A quick recap (2) – SGP4/SDP4

Usage of SGP4

a. Initialize the satellite record with
   - `twoline2rv` (requires tle strings)
   - `read_TLE` (requires TLE file name or sat id)

b. Call SGP4
   - Provide time since reference epoch (minutes)

c. Convert from/to TEME (if needed)
   - Compute `ttt`
     - Centuries from TDT 2000 January 1 00:00:00
   - TEME acceleration can be set to [0,0,0]
   - Get offsets dPsi and dEpsilon
     - Convert them in rad if provided in arcsec
   - Call functions `teme2eci` or `eci2teme`

```
typerun    = 'u';  % user-provided inputs to SGP4 Matlab function
opsmode    = 'a';  % afspc approach ('air force space command')
whichconst =  72;  % WGS72 constants (radius, gravitational parameter)
```

```
% initialize the satrec structure, using the function twoline2rv
satrec = twoline2rv(longstr1, longstr2, typerun,'e', opsmode, whichconst)
```

```
satrec = read_TLE(123456, whichconst);
```

```
% Evaluate the TLE
et0 = cspice_str2et('TLE REF EPOCH'); % Reference time of the TLE
et = 12345.6789;  % Ephemeris time for the SGP4 propagation
tsince = (et-et0)/60; % Time since TLE reference epoch [min]
[~,rteme,vteme] = sgp4(satrec, tsince);

% Centuries from TDT 2000 January 1 00:00:00.000
ttt = cspice_unitim(et, 'ET', 'TDT')/cspice_jyear()/100;
```

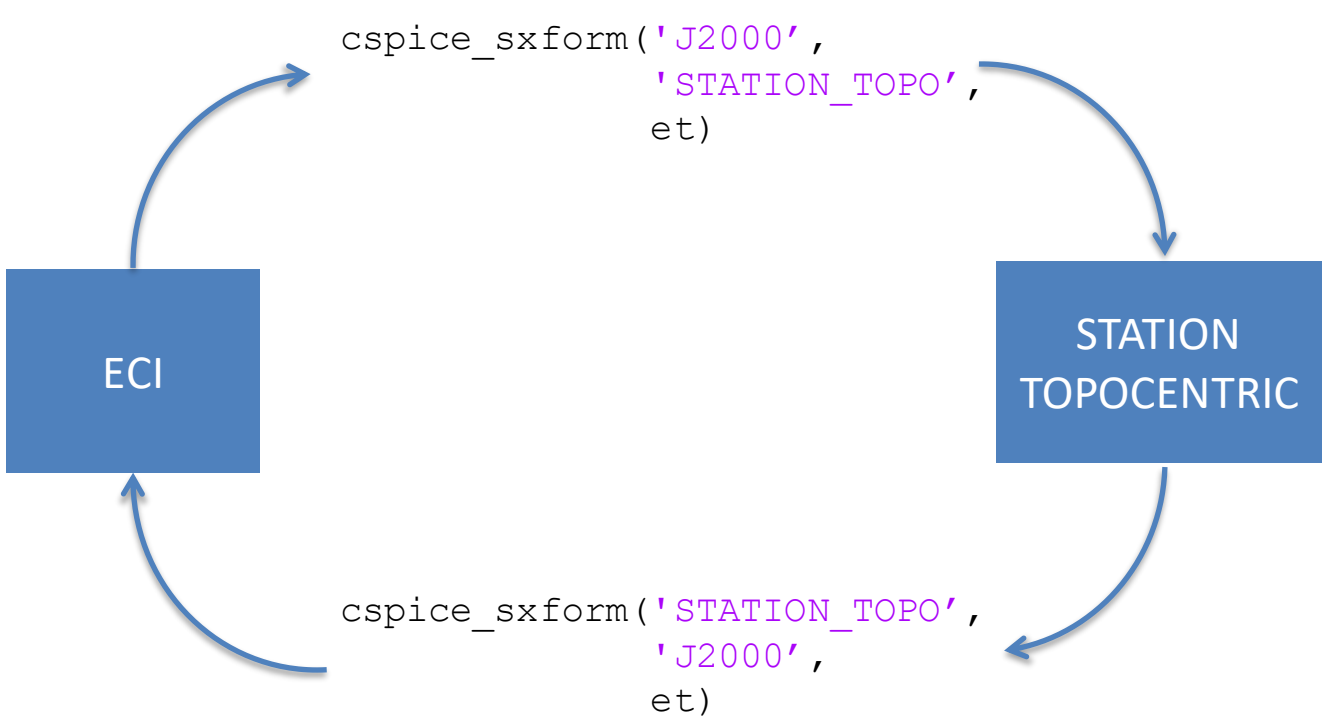
```
% -------- teme2eci    - transform teme to eci vectors
ateme = [0;0;0];
[reci, veci, aeci] = teme2eci(rteme, vteme, ateme, ttt, ddpsi, ddeps);
```

```
% -------- eci2teme    - transform eci to teme vectors
[rteme, vteme] = eci2teme(reci, veci, aeci, ttt, ddpsi, ddeps);
```

A quick recap (3) – Reference frames

Reference frame conversions

With station kernel generated with `pinpoint`



Without station kernel

```
% Define station coordinates
lat = 45.50122; % deg
lon = 9.15461; % deg
alt = 20; % m

% Get Earth radii (equatorial and polar)
% (requires pck00010.tpc kernel)
radii = cspice_bodvrd('EARTH', 'RADII', 3);
re = radii(1); rp = radii(3);

% Compute flattening
flat = (re - rp) / re;

% Convert to radians and km
lat_rad = deg2rad(lat); lon_rad = deg2rad(lon);
alt_km = alt / 1000;

% Compute station pos wrt Earth center (ECEF)
pos_ECEF = cspice_pgrrec(...
    'EARTH', lon_rad, lat_rad, alt_km, re, flat);
% Compute ECEF2TOPO rotation matrix
ECEF2TOPO = cspice_eul2m(...
    lat_rad - pi, pi - lon_rad, pi / 2, 2, 1, 2);
```

A quick recap (4a) – Generation of multivariate normal random numbers

Use `mvnrnd` to generate multivariate normal random numbers

- **Generate sample points for Monte Carlo simulations**
  Example: generate `100` vectors with given mean `[1 3]` and **covariance** `diag([0.01 0.02])`:

```
n = 100; mu = [1 3]; Sigma = diag([0.01 0.02]);
R = mvnrnd(mu, Sigma, n);  % R size: 100x2
```

- **Add random noise to measurements**
  Example: add noise with **covariance** `diag([0.01 0.02])` to a set of angular measurements :

```
mu = [1 3; 1.5 2; 2 1]; Sigma = diag([0.01 0.02]);
R = mvnrnd(mu, Sigma);  % R size: 3x2
```

Notes:

- When Sigma is diagonal, you can directly pass the diagonal (e.g. `Sigma = [0.01 0.02];`)
- It is possible to define a different Sigma for each measurement (using the 3rd dimension)

A quick recap (4b) – Correlation coefficients

A covariance matrix can be also expressed in terms of correlation coefficients ρ

$$P = \begin{bmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y & \rho_{xz}\sigma_x\sigma_z \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 & \rho_{yz}\sigma_y\sigma_z \\ \rho_{xz}\sigma_x\sigma_z & \rho_{yz}\sigma_y\sigma_z & \sigma_z^2 \end{bmatrix} \text{ with } \begin{cases} \rho_{xy} = \frac{\sigma_{xy}}{\sigma_x\sigma_y} \\ \rho_{xz} = \frac{\sigma_{xz}}{\sigma_x\sigma_z} \\ \rho_{yz} = \frac{\sigma_{yz}}{\sigma_y\sigma_z} \end{cases}$$

Measurements noise (noise matrix R)	$\sigma_{Az} = 1.5$ mdeg $\sigma_{El} = 1.3$ mdeg $\sigma_{range} = 75$ m $\rho_{Az-El} = 0.1$ $\rho_{Az-rng} = \rho_{El-rng} = 0$	$\sigma_{Az} = 1.5$ mdeg $\sigma_{El} = 1.3$ mdeg $\sigma_{range} = 75$ m $\rho_{Az-El} = 0.1$ $\rho_{Az-rng} = \rho_{El-rng} = 0$

A quick recap (5a) – Non-linear least squares in MATLAB

Solve nonlinear least-squares problems in MATLAB

```
[x, resnorm, residual, exitflag, ~, ~, jac] = lsqnonlin(fun, x0, lb, ub, opt);
```

Inputs:

- `fun`: handle of the cost function to be minimized
  - `fun` must take as input only the x and must return the residual
  - the residual can be a vector, `lsqnonlin` implicitly computes the norm
- `x0`: initial point
- `lb, ub`: lower and upper bounds
- `opt`: to be defined using the `optimoptions` function

Outputs:

- `x`: solution
- `resnorm`: squared norm of the residual
- `residual`: value of `fun(x)`
- `exitflag`: reason the solver stopped (convergence if `exitflag` > 0)
- `jac`: jacobian matrix of `fun` with respect to `x`

A quick recap (5b) – Non-linear least squares in MATLAB example

Solve nonlinear least-squares problems in MATLAB - example

```
% Variables of the problem
x_0 = ...; var_1 = ...; var_2 = ...;

% Encapsulate the variables in the function handle
fun = @(x) costfunction(x, var_1, var_2, ...);

% Optimization options
opt = optimoptions('lsqnonlin', 'Algorithm', 'levenberg-marquardt', 'Display', 'iter');

% Execute least squares
[x, resnorm, residual, exitflag, ~, ~, jac] = lsqnonlin(fun, x_0, [], [], opt);

% Compute resulting covariance
Jac = full(jac);
P_ls = resnorm / (length(residual)-length(x_0)) .* inv(Jac'*Jac);

function residual = costfunction(x, var_1, var_2, ... )
    residual = []; % Initialize output variable
    % Propagate x to the epochs of the measurements
    x_prop = ...;
    % Compute predicted measurements
    meas_pred = ...;
    % Compute the residual of the measurements and append it to the output
    diff_meas_weighted = W_m * (meas_pred - meas_real);
    residual = [residual; diff_meas_weighted(:)];
end
```

Weight for measurements:

```
W_m = diag(1 ./ sigma_meas);
```

A quick recap (6) – UT

$\mathbf{x}, \mathbf{P_x}$

Create sigma points

$\boldsymbol{\chi}_0 = \mathbf{x} \qquad c = \alpha^2(N + \kappa)$

$\boldsymbol{\chi}_i = \mathbf{x} + \boldsymbol{\Delta}\mathbf{x}_i \text{ for } i = 1, \dots, 2N$

$\boldsymbol{\Delta}\mathbf{x}_i = \left(\sqrt{c\mathbf{P_x}}\right)_i \text{ for } i = 1, \dots, N$

$\boldsymbol{\Delta}\mathbf{x}_{i+N} = -\left(\sqrt{c\mathbf{P_x}}\right)_i \text{ for } i = 1, \dots, N$

$\boldsymbol{\chi}$

Apply transformation

$\boldsymbol{\gamma} = \mathrm{f}(\boldsymbol{\chi})$

NB:

- N **is the size of the state**
- $\left(\sqrt{c\mathbf{P_x}}\right)_i$ **indicates the i-th column of the matrix square root** (*sqrtm* matlab)

Parameters to be selected (with common values)

$\alpha = 1e-3$

$\beta = 2$

$\kappa = 0$

Compute mean and covariance

$$W_0^{(m)} = 1 - \frac{N}{\alpha^2(N+\kappa)} \qquad W_0^{(c)} = (2 - \alpha^2 + \beta) - \frac{N}{\alpha^2(N+\kappa)}$$

$$W_i^{(m)} = \frac{1}{2\alpha^2(N+\kappa)} \text{ for } i = 1, \dots, 2N \qquad W_i^{(c)} = \frac{1}{2\alpha^2(N+\kappa)} \text{ for } i = 1, \dots, 2N$$

$$\mathbf{y} = \sum_{i=0}^{2N} W_i^{(m)} \boldsymbol{\gamma}_i \qquad \mathbf{P_y} = \sum_{i=0}^{2N} W_i^{(c)} [\boldsymbol{\gamma}_i - \mathbf{y}][\boldsymbol{\gamma}_i - \mathbf{y}]^{\mathrm{T}}$$

REF: Van der Merwe, Rudolph, and Eric A. Wan. "The Square-Root Unscented Kalman Filter for State and Parameter-Estimation." 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221), 6:3461–64. Salt Lake City, UT, USA: IEEE, 2001. https://doi.org/10.1109/ICASSP.2001.940586.

A quick recap (7) – EKF

Extended Kalman Filter (EKF)

Previous state with covariance

$$\hat{\mathbf{x}}_{k-1}^{+} \quad \mathbf{P}_{k-1}^{+}$$

Propagate dynamics

Next state and STM

$$\hat{\mathbf{x}}_{k}^{-} \quad \boldsymbol{\Phi}_{k-1,k}$$

Propagate covariance

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_{k-1,k}\mathbf{P}_{k-1}^{+}\boldsymbol{\Phi}_{k-1,k}^T$$

Compute measurements and their Jacobians

$$\hat{\mathbf{y}}_k^- = \mathbf{h}(\hat{\mathbf{x}}_k^-)$$

$$\boldsymbol{H}_k = \left.\frac{\partial \mathbf{h}(\hat{\mathbf{x}}_k^-)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k^-}$$

Update state

$$\mathbf{K}_k = \mathbf{P}_k^- \boldsymbol{H}_k^T \left(\boldsymbol{H}_k \mathbf{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}\right)^{-1}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)$$

$$\mathbf{P}_k^+ = (\boldsymbol{I} - \mathbf{K}_k \boldsymbol{H}_k)\mathbf{P}_k^-$$

Reference:
Crassidis & Junkins, Optimal Estimation of Dynamic Systems

A quick recap (8) – UKF

Unscented Kalman Filter (UKF)

Previous state with covariance

$\hat{\mathbf{x}}_{k-1}^{+} \quad \mathbf{P}_{k-1}^{+}$

Create sigma points → $\boldsymbol{\chi}_{i,k-1}$

Apply propagation function

$\boldsymbol{\chi}_{i,k} = \mathbf{f}(\boldsymbol{\chi}_{i,k-1})$ → $\boldsymbol{\chi}_{i,k}$

Apply measurement function

$\boldsymbol{\gamma}_{i,k} = \mathbf{h}(\boldsymbol{\chi}_{i,k})$ → $\boldsymbol{\gamma}_{i,k}$

Apply UT weights to compute means and covariances

$$\hat{\mathbf{x}}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \boldsymbol{\chi}_{i,k} \qquad \mathbf{P}_k^- = \sum_{i=0}^{2L} W_i^{(c)} [\boldsymbol{\chi}_{i,k} - \hat{\mathbf{x}}_k^-][\boldsymbol{\chi}_{i,k} - \hat{\mathbf{x}}_k^-]^{\mathrm{T}}$$

$$\hat{\mathbf{y}}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \boldsymbol{\gamma}_{i,k} \qquad \mathbf{P}_{ee,k} = \sum_{i=0}^{2L} W_i^{(c)} [\boldsymbol{\gamma}_{i,k} - \hat{\mathbf{y}}_k^-][\boldsymbol{\gamma}_{i,k} - \hat{\mathbf{y}}_k^-]^{\mathrm{T}} + \mathbf{R}_k$$

$$\mathbf{P}_{xy,k} = \sum_{i=0}^{2L} W_i^{(c)} [\boldsymbol{\chi}_{i,k} - \hat{\mathbf{x}}_k^-][\boldsymbol{\gamma}_{i,k} - \hat{\mathbf{y}}_k^-]^{\mathrm{T}}$$

Update state

$$\mathbf{K}_k = \mathbf{P}_{xy,k}\, \mathbf{P}_{ee,k}^{-1}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)$$

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{P}_{ee,k} \mathbf{K}_k^{\mathrm{T}}$$

Reference:
Crassidis & Junkins, Optimal Estimation of Dynamic Systems

General hints

Report content

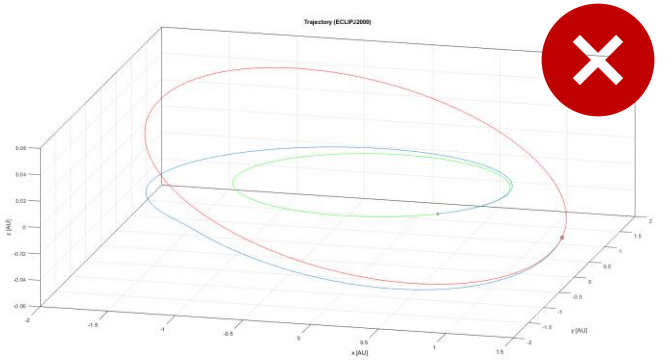
- **State vectors** must be provided with corresponding **reference frame and origin**
- Always put the (correct!) units
- Do not mix position and velocities when reporting errors

r_x [km]	r_y [km]	r_z [km]
1234.567	123.456	234.567

Satellite position (@Sun ECLIPJ2000)

Plots

- Make sure that axis labels and titles are clearly readable with page zoom at 100%
- Remember to put the (correct!) units on each axis
- When plotting trajectories specify the **reference frame and origin** in the plot title or caption



General hints

LaTeX

- Do not put multiplication symbols, especially as asterisks
- Clearly distinguish between vectors and scalar: write vector using underline, arrow or bold
  - Latin alphabet: `\mathbf{x}` or `\mathbf{r}`
  - Greek symbols: `\boldsymbol{\lambda}`
  - **Obs:** No need to use norm with this notation $\|\boldsymbol{\lambda}_x\| \leftrightarrow \lambda_x$
- Write scalar product using `\cdot`
- Prefer the use of `\dfrac{}{}` over `\frac{}{}` when writing equations
- When inserting text in an equation remember to use `\mathrm{}`