

Orbit Mechanics Tutorial

Exercise 2: Numerical Integration of Satellite Orbits

Tutor: Zhang Ke

ge85taz@mytum.de

Directed by: Prof. Urs Hugentobler

urs.hugentobler@tum.de

2 tutorials in the semester:

1. Keplerian Orbits in Space-Fixed, Earth-fixed and Topocentric Systems
2. Numerical Integration of Satellite Orbits

Satellite to analyse: Sentinel 3, consider 3 orbit revolutions and compute the undisturbed and disturbed case.

Undisturbed case:

- Ideal situation, where the orbit is a perfect ellipse and only the gravity is considered.

Disturbed case:

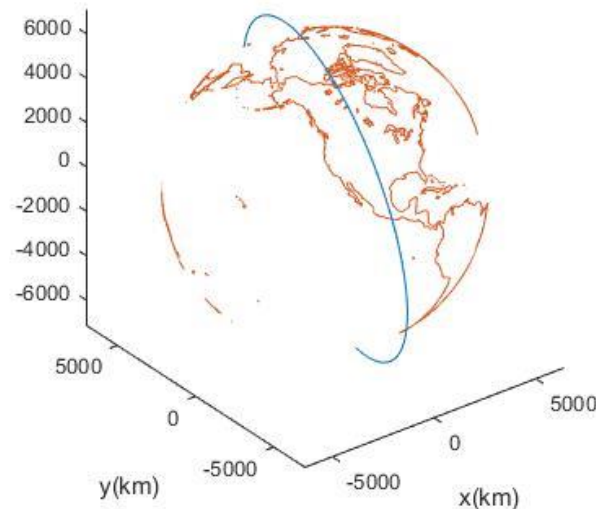
- Orbit perturbation due to Earth oblateness. Numerical integration and equations for motion are used.

Keplerian elements:

Satellite	$a[\text{km}]$	e	$i[^\circ]$	$\Omega[^\circ]$	$\omega[^\circ]$	$T_p[\text{sec}]$
Sentinel-3	7192	0.004	98.3	257.7	144.2	00:00

Task 1: Plot 3 revolutions for the undisturbed case

Trajectory in Space-fixed system



Functions were already created for the exercise 1, simply reuse them!

Load again the `Earth_coast()` function.

Task 2: Write program yprime

There are many methods to solve numerical integration such as Euler, Runge-Kutta, etc., single-step methods, multi-step methods.

Input is the differential equation (i.e., the change of the parameters as function of the parameters themselves and time)

$$\frac{dx}{dt} = f(x, t)$$

Given this analytical function and a step size they compute the integral $x(t)$

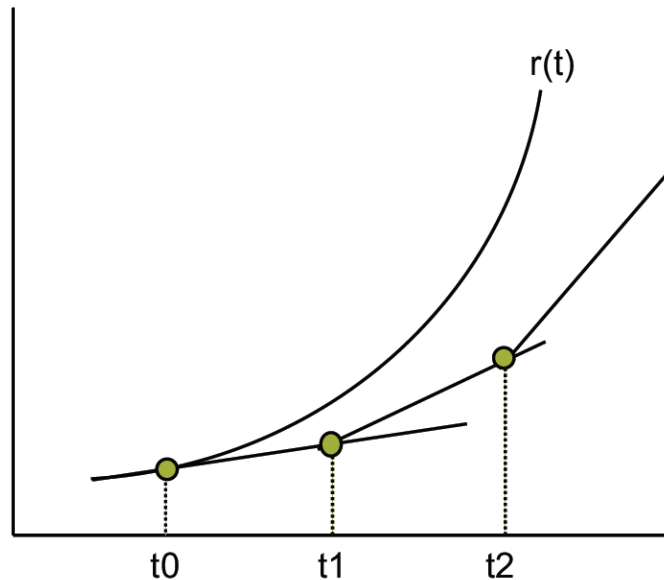
A too large step size may cause approximation errors.

A too small step size may cause rounding errors.

The different integration methods minimize the accumulation of these errors, e.g. using high order polynomial approximations and/or automatic selection of optimal step size.

Task 2: Write program yprime

Example for simple method (Euler method):



$r(t_0) = a$, known (initial value)
 $r(t_1), r(t_2), \dots, r(t_n)$ to be computed

$dr/dt = g(r,t)$ is given


Integration goes like this:

$r(t_0) = a$
 $r(t_1) \approx r(t_0) + g(r, t_0) * (t_1 - t_0)$
 $r(t_2) \approx r(t_1) + g(r, t_1) * (t_2 - t_1)$

... and so on

Task 2: Write program yprime

In orbit mechanics we deal with a system of three ordinary differential equations of second order

$$\ddot{\mathbf{r}} = -\frac{GM}{r^3}\mathbf{r} + \mathbf{f}_s$$


Central force of the gravity field Disturbing forces

Usually numerical integrators are solving differential equations of first order. With the method of Cowell the above system of equations can be converted into six equations of first order:

With $\mathbf{y} = \begin{pmatrix} \mathbf{r} \\ \mathbf{v} \end{pmatrix}$ we get $\dot{\mathbf{y}} = \mathbf{F}(\mathbf{y}) = \begin{pmatrix} \dot{\mathbf{r}} \\ \dot{\mathbf{v}} \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ \ddot{\mathbf{r}} \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ -\frac{GM}{r^3}\mathbf{r} + \mathbf{f}_s \end{pmatrix}$

\mathbf{y} is the 6-dimensional state vector.

Task 2: Write program yprime

Matlab provides various numerical integrator functions

$$\ddot{\mathbf{r}} = -\frac{GM}{r^3}\mathbf{r} + \mathbf{f}_s$$

Integrator
function

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ v_{x_0} \\ v_{y_0} \\ v_{z_0} \end{bmatrix}$$

Initial position and
velocity (e.g.
from kep2cart)

$$\begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \\ v_{x_0} \\ v_{y_0} \\ v_{z_0} \end{bmatrix}$$

Apply formula
from previous
slide

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ v_{x_1} \\ v_{y_1} \\ v_{z_1} \end{bmatrix}$$

Next position and
velocity after time
step

[...]

Repeat the process. Use
different time steps.
Results are here for 5
and 50 secs

Task 2: Write program yprime

Try with different integrators from MATLAB like ode23, ode45, ode113 (check MATLAB help for more details).

```
y0=[x0,y0,z0,vx0,vy0,vz0];
```

define initial values

```
options=odeset('InitialStep',5, 'MaxStep',5);
```

define time steps, etc.

```
[t y]=ode23('yprime', times, y0, options);
```

call integrator, with start/end time,
output is a time vector and vector
of state vectors

User-defined function containing the first-order differential equation:

```
function yp =yprime(t,y)
```

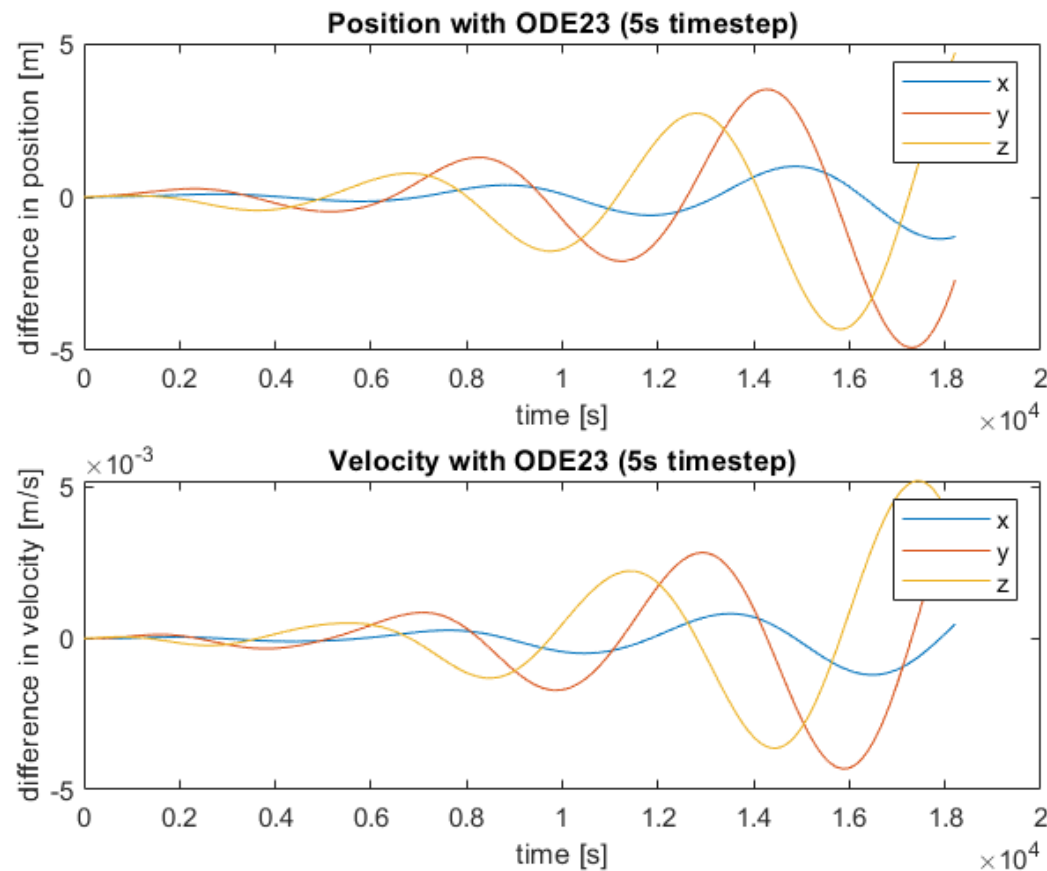
```
% Input: epoch and state vector
```

```
% Output: time derivative if state vector
```

```
...
```

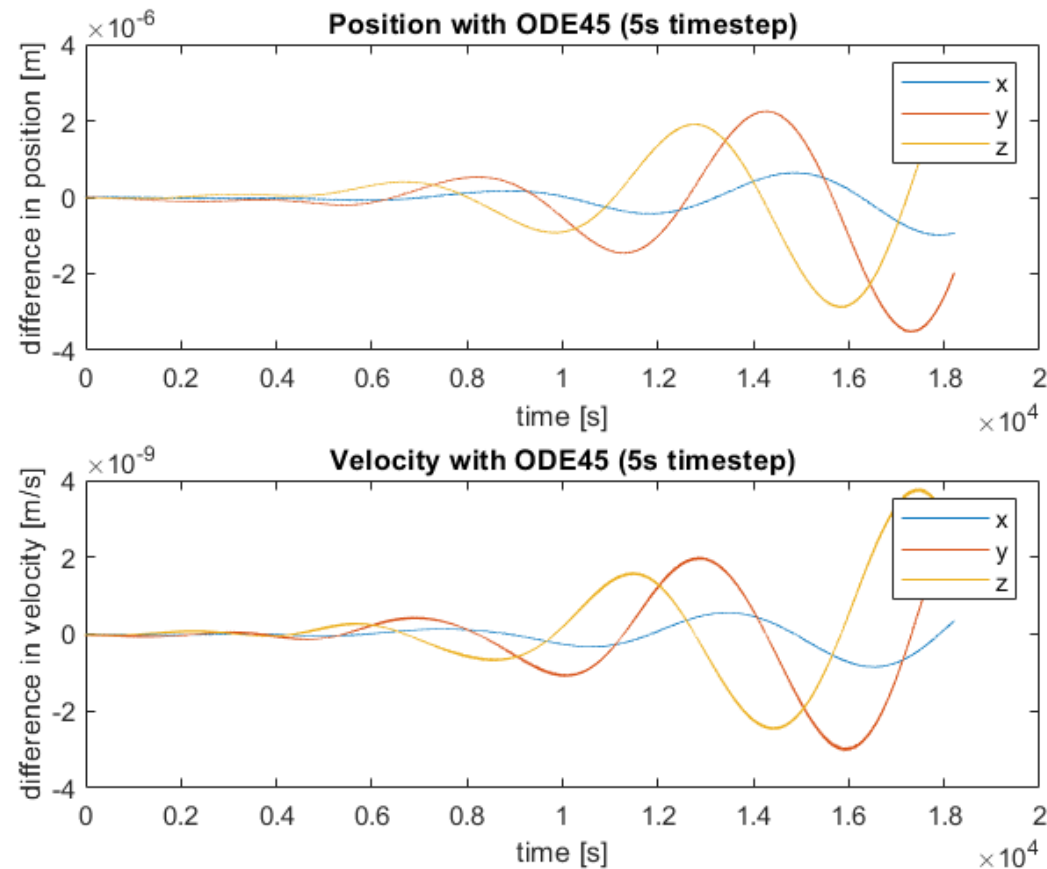
Task 3: Plot the results

Here for 5 seconds



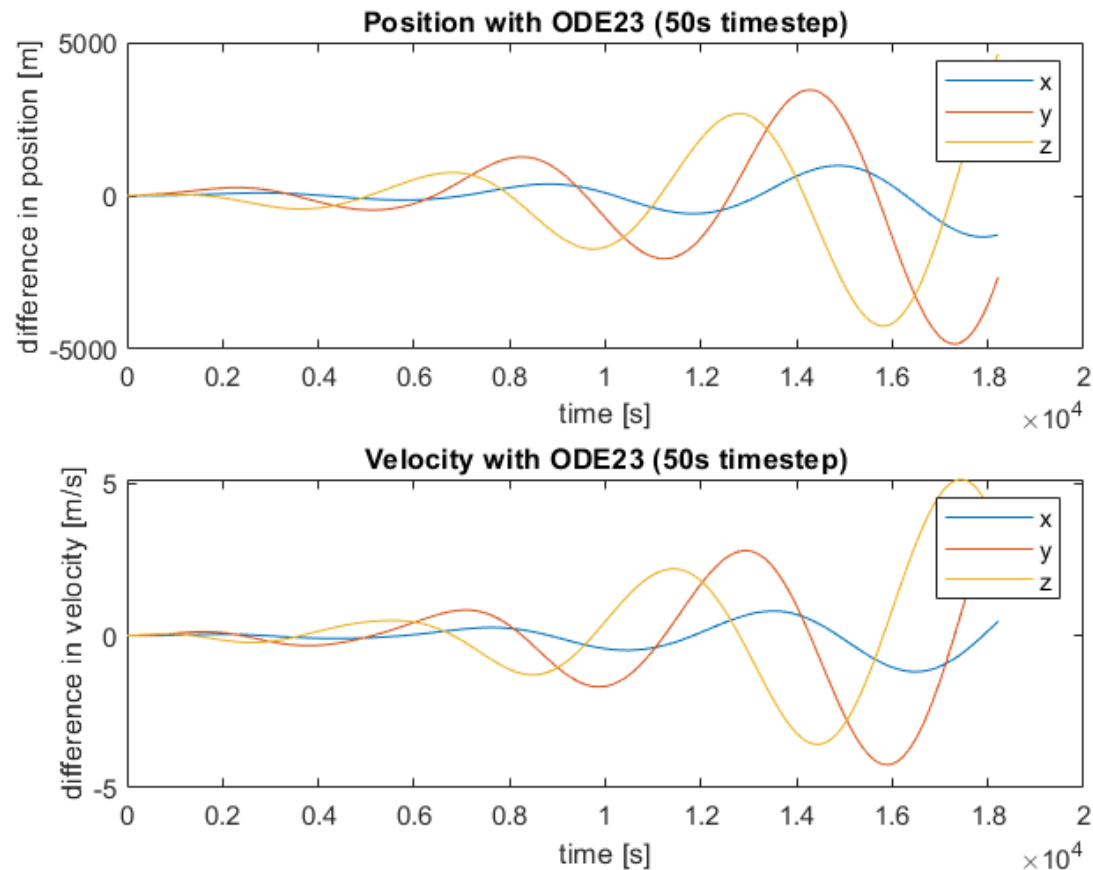
Task 3: Plot the results

Using ode45 for 5 sec.



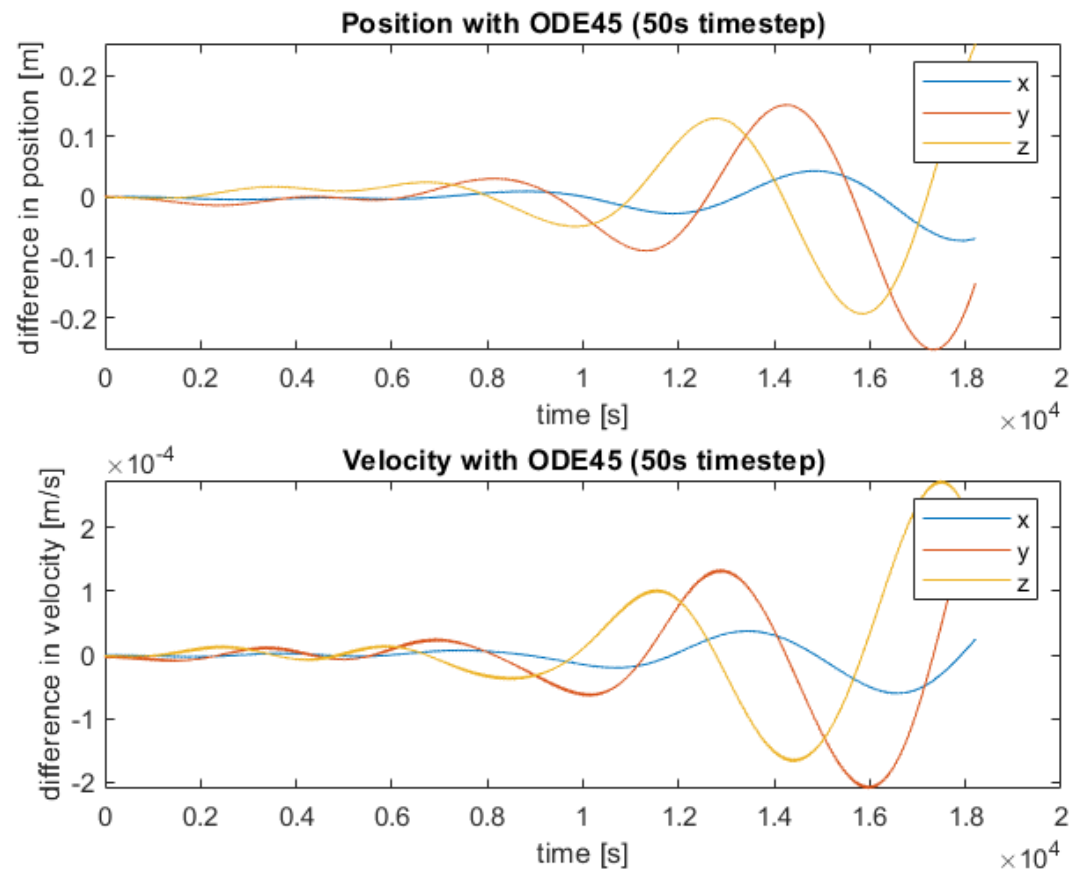
Task 3: Plot the results

Ode23 for 50 seconds



Task 3: Plot the results

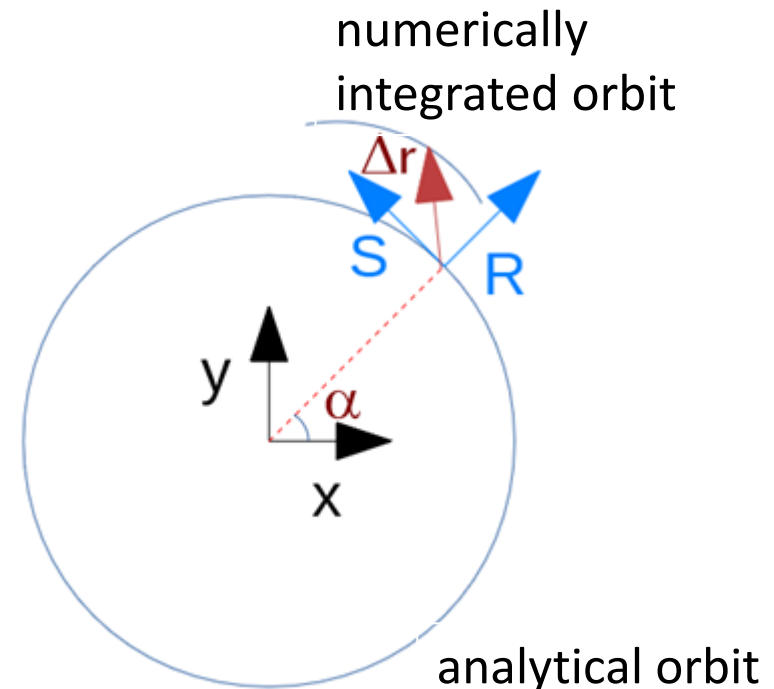
Ode45 for 50 seconds



Task 4: Plot results in radial, alongtrack, crosstrack

Observe the differences in the RSW system.

- 1) Compute the base unit vectors of the RSW system based on the analytical orbit (resp. unperturbed orbit), e.g. starting from position and velocity vectors
- 2) Decompose the difference vector between numerically integrated orbit (resp. the perturbed and unperturbed orbit) into this base.



$$\mathbf{e}_R = \frac{\mathbf{r}}{|\mathbf{r}|} \quad \mathbf{e}_W = \frac{\mathbf{r} \times \mathbf{v}}{|\mathbf{r} \times \mathbf{v}|} \quad \mathbf{e}_S = \mathbf{e}_W \times \mathbf{e}_R$$

$$\Delta r_R = \mathbf{e}_R \cdot \Delta \mathbf{r} \quad \Delta r_S = \mathbf{e}_S \cdot \Delta \mathbf{r} \quad \Delta r_W = \mathbf{e}_W \cdot \Delta \mathbf{r}$$

Task 5: Consider the disturbed case

Use the acceleration for the oblate Earth given in the lecture:

$$\ddot{\mathbf{r}} = -\frac{GM}{r^3} \mathbf{r} - \frac{3}{2} J_2 \frac{a_e^2}{r^2} \begin{pmatrix} x \left(5 \left(\frac{z}{r} \right)^2 - 1 \right) \\ y \left(5 \left(\frac{z}{r} \right)^2 - 1 \right) \\ z \left(5 \left(\frac{z}{r} \right)^2 - 3 \right) \end{pmatrix}$$

$a_e = 6378 \text{ km}$

: Earth equatorial radius

$J_2 = 0.00108263$

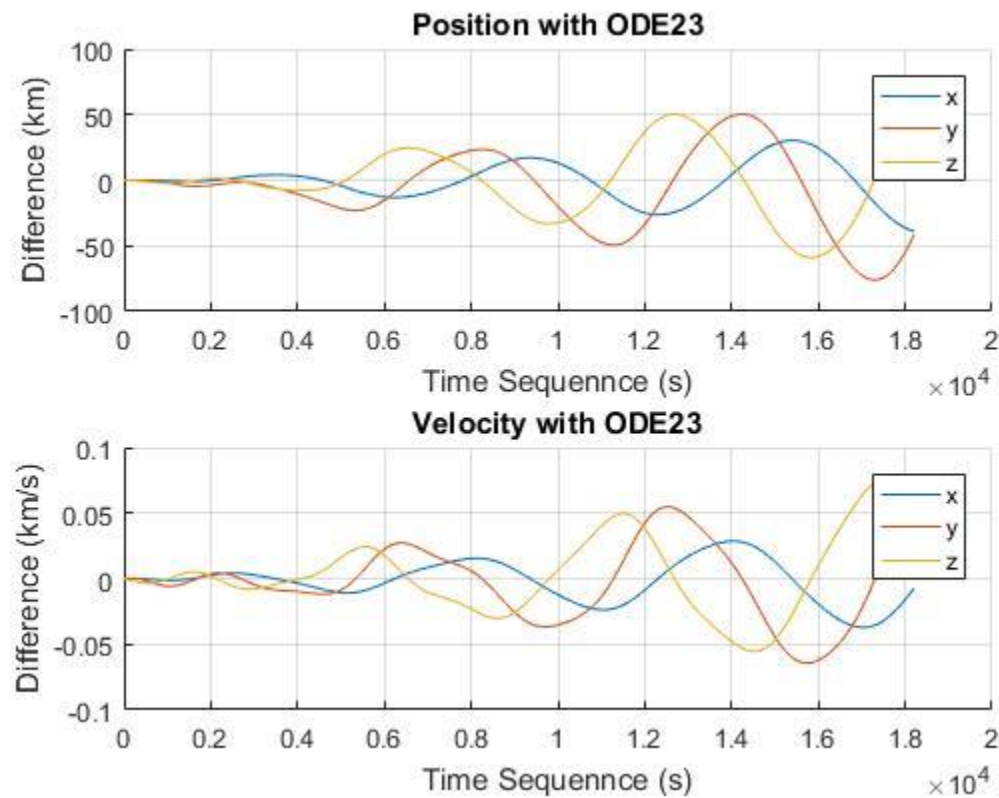
: gravity field parameter describing Earth oblateness

$GM = 398.6005 \cdot 10^{12} \text{ m}^3/\text{s}^2$

: gravitational parameter of Earth

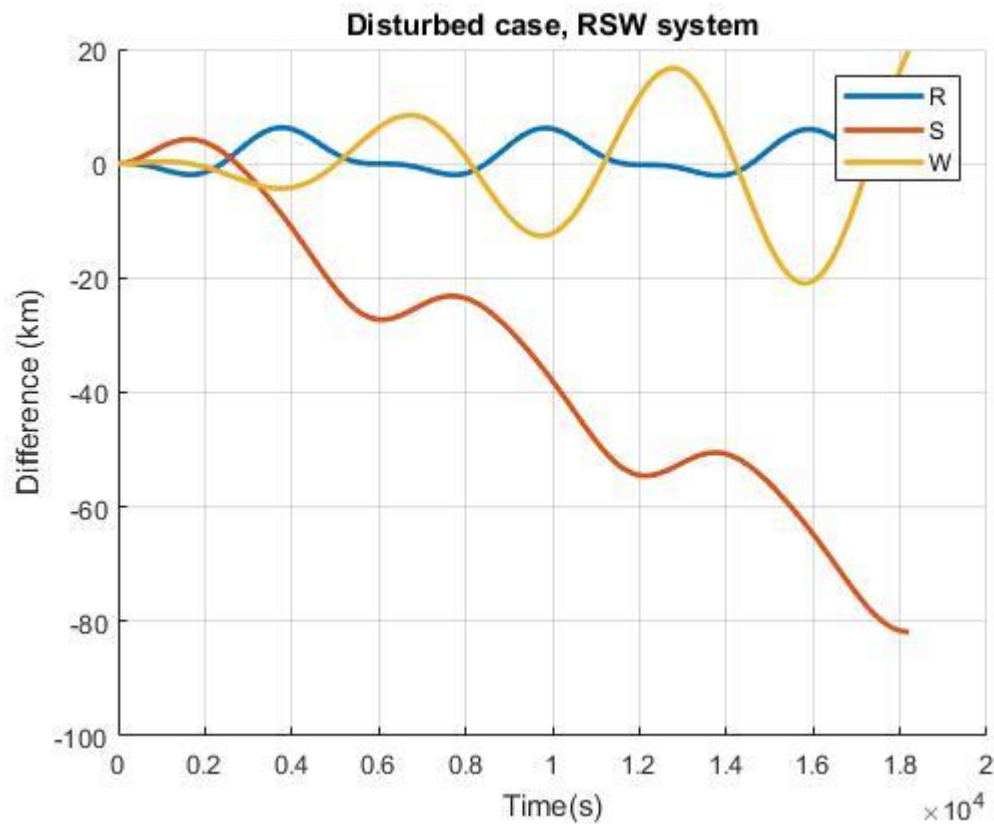
Task 5: Consider the disturbed case

Plot the results. Here ode23 for 5 sec.



Task 5: Consider the disturbed case

Plot the result. Ode23, step=5, here given in RSW-frame.



Task 6: Own integrator for the undisturbed case

Consider e.g. Euler and Runge-Kutta.

Euler

$$y(t_{n+1}) = y(t_n) + \dot{y}(t_n) * \Delta t$$

Runge-Kutta

$$\dot{y}(t_{n+1}) = f(t_n, y(t_n))$$

$$k_1 = f(t_n, y(t_n)) * \Delta t$$

$$k_2 = f(t_n + \frac{h}{2}, y(t_n) + \frac{k_1}{2}) * \Delta t$$

$$k_3 = f(t_n + \frac{h}{2}, y(t_n) + \frac{k_2}{2}) * \Delta t$$

$$k_4 = f(t_n + h, y(t_n) + k_3) * \Delta t$$

A way to do it in MATLAB:

Write the integrator function that accepts the name of the function yprime as a handle, e.g.:

`function y=OwnIntegrator(fh,tspan,tstep,y0,sel)`

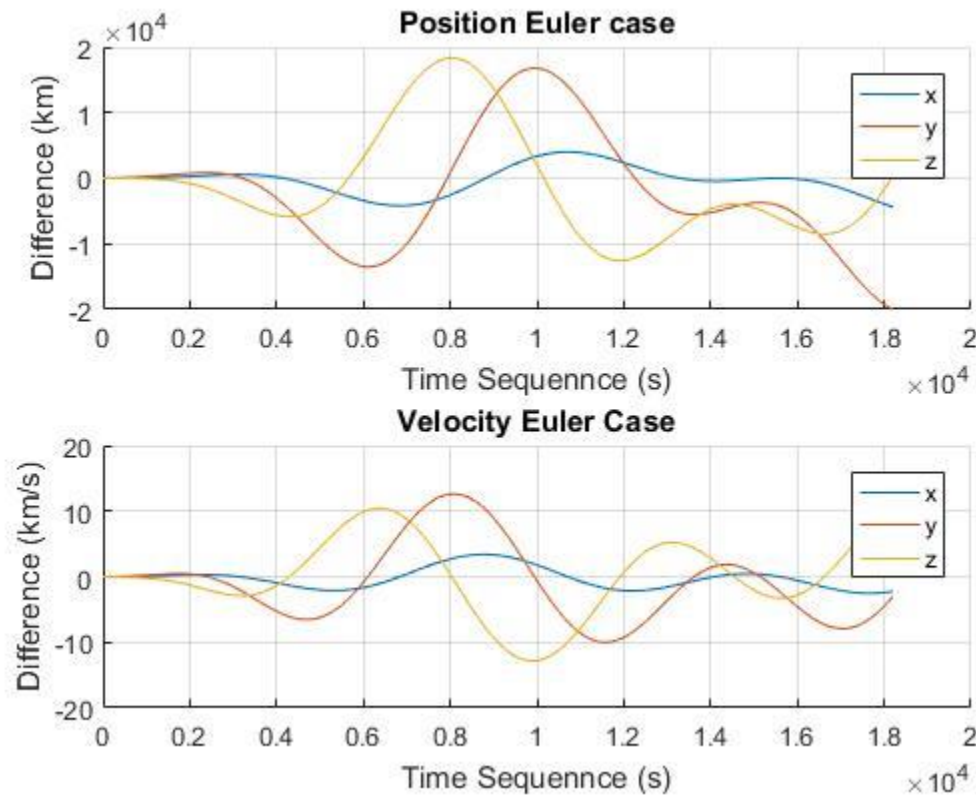
fh: function handle, call in SR with `fh(t,x)`
tspan: array with start and end time
tstep: step size
y0: initial state vector
sel: selector for Euler or Runge-Kutta

Call integrator with

`y=OwnIntegrator(@yprime,tspan,tstep,y0,sel)`

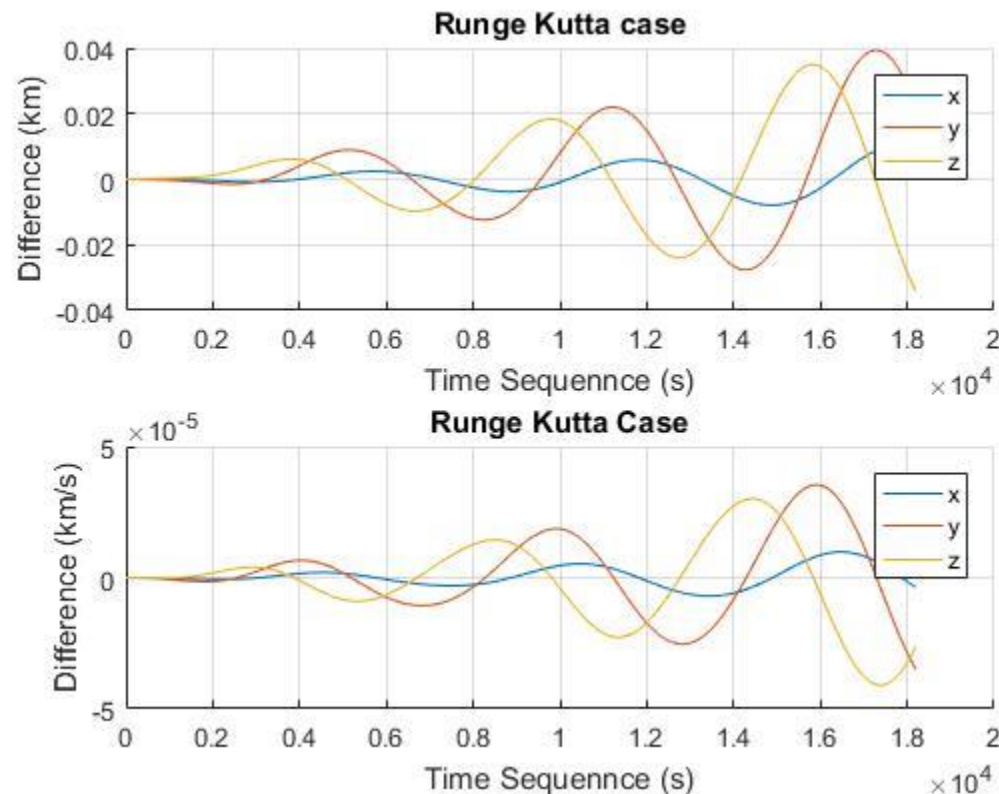
Task 5: Own integrator for the undisturbed case

Result for Euler 50s.



Task 5: Own integrator for the undisturbed case

Result for RK 50s.



To consider

Results will have some variations due to the MATLAB version, internal parameters and the features of your own computer. You will have similar results but don't be afraid if they vary slightly to the ones in the slides or to the ones of your classmates. Adapt the step size in the ode23, ode45 and ode113 if required, but keep the step for the time vector.

Provide more cases in your results as the ones given here, these are just to give a hint of the result, but cases for at least two size steps and two MATLAB methods should be done.

Submission until: Feb. 12, 2024