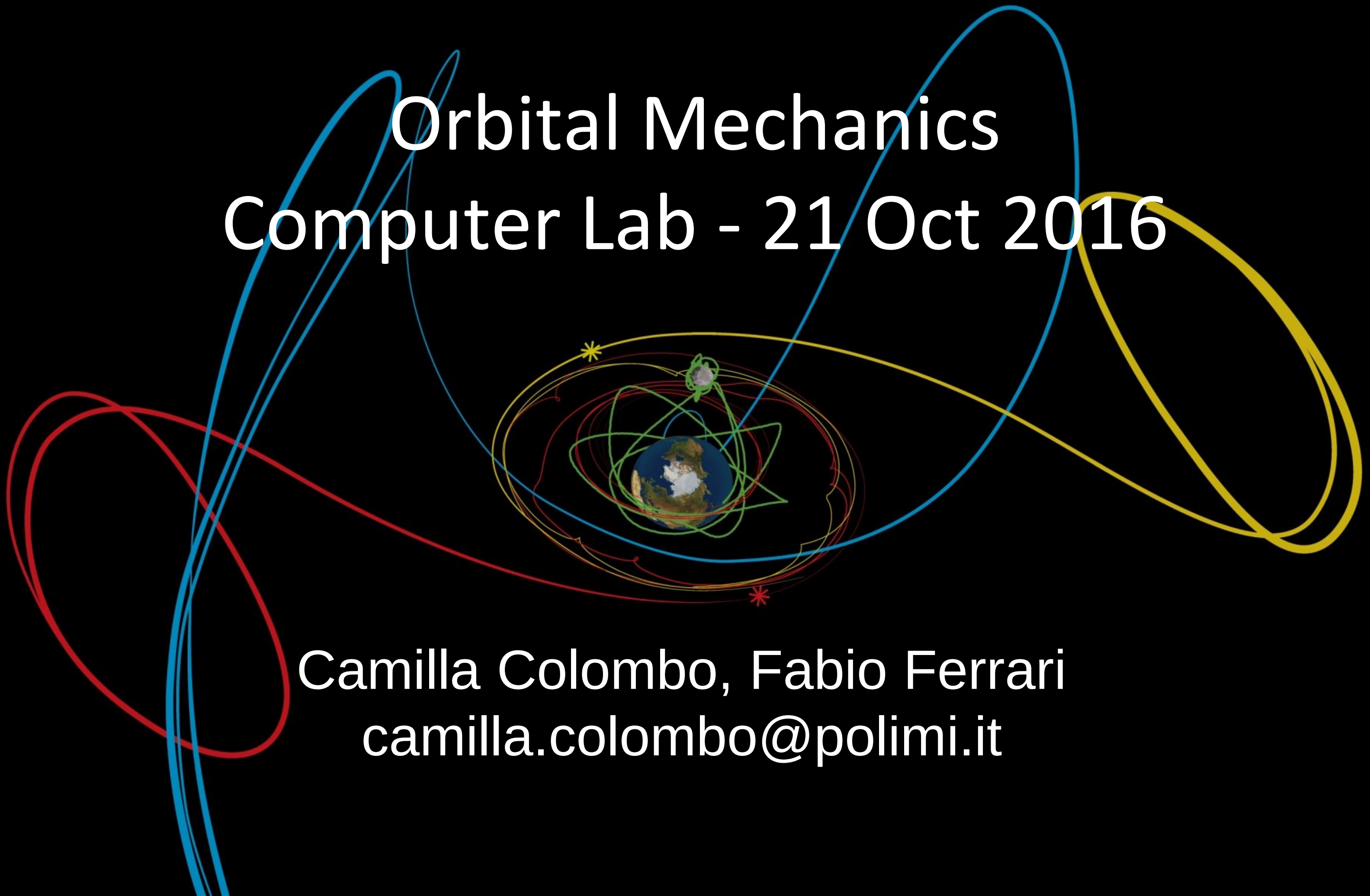
Orbital Mechanics
Computer Lab - 21 Oct 2016



Camilla Colombo, Fabio Ferrari
camilla.colombo@polimi.it

Preparing the basic ingredients for trajectory design

2. ORBIT REPRESENTATION

Lab 2.1: Cartesian to Keplerian

1. Write a function car2kep.m that takes as input [state = (x,y,z,vx,vy,vz)] and returns as output: [kep=(a,e,i,Om,om,theta)]
2. Write a function kep2car.m that takes as input [kep=(a,e,i,Om,om,theta)] and returns as output: [state = (x,y,z,vx,vy,vz)]
3. Write a function plotOrbit.m that plot an orbit given 5 orbital elements: kep=(a,e,i,Om,om,theta) and returns the handle to the figure

NB: Also mu is an input to the function. For testing the function write a script so that all the input you use are stored.

Lab 2.2: Keplerian orbit

1. Numerically integrate with ode45 the equation $\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r} = 0$
   function dyn_2BP.m

   Initial conditions: positions and velocity in LEO at altitude of 500 km, inclination = 30 degrees, RAAN = 40 degrees, omega = 0 degrees, e =0.
2. Plot r(t), v(t)
3. Demonstrate with a graph that h(t) and e(t) are constant in magnitude and direction and that e(t) is perpendicular to h(t)
4. Convert r(t) and v(t) to kep(t)
5. Plot a(t), e(t), i(t), Om(t), om(t), theta(t)
6. Plot vr(theta) and vtheta(theta)

NB: declare all initial parameters and numerical constants at the beginning of the script, use scripts!

Lab 2.3: Integrating the dynamics with MatLab

```
[T,X]=ode45(@dynamics,tspan,X0)
```

integration time
(n_T x 1 array)

state evolution in time
(n_T x n_X matrix)

time span for integration

- $[T_0\ T_f]$
- $[T_0\ T_1\ \dots\ T_i\ \dots\ T_f]$ (1 x n_T array)

initial state
(1 x n_X array)

function $[X_d]$=dynamics(t,X)

…

< dynamics written as system of first order ode >

…

Xd=[…]’; % column vector

n_X = length of state vector

n_T = length of integration time array

Lab 2.3: Integrating the dynamics with MatLab

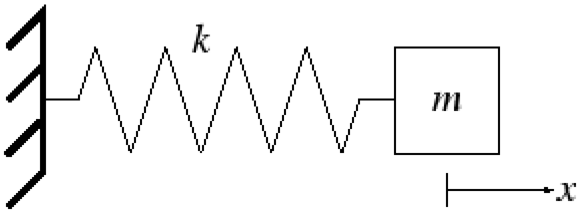
<u>Example: harmonic oscillator</u>

Equation of motion

$$m\ddot{x} + kx = 0$$

$$\updownarrow$$

$$\ddot{x} + \omega_0^2 x = 0 \qquad \omega_0 = \sqrt{\frac{k}{m}}$$



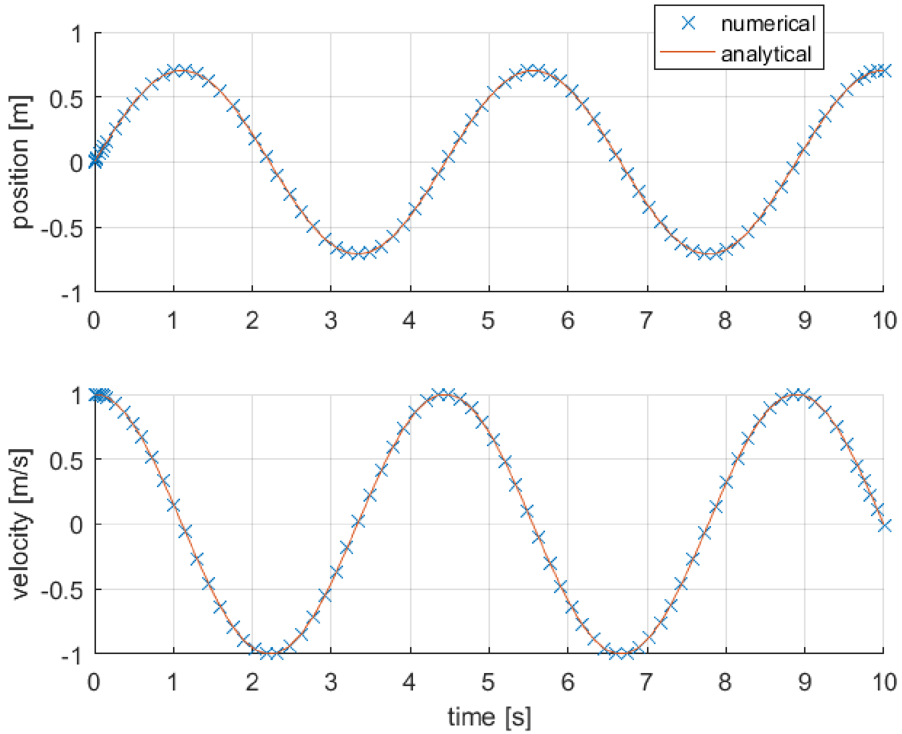
Analytical solution

$$x(t) = \frac{\dot{x}_0}{\omega_0}\sin(\omega_0 t) + x_0\cos(\omega_0 t)$$

- Compare analytical solution with numerical integrated one

Lab 2.3: Integrating the dynamics with MatLab

1. Write the equation of motion (second order differential equation) into a system of ode into a function
2. Call the function using ode45
3. Plot the results and compare with analytical solution



Lab 2.3: Integrating the dynamics with MatLab

<u>Astrodynamics problems</u>

- Require high accuracy and more stringent tolerances

```
options=odeset('Reltol',1e-13,'AbsTol',1e-14);
```

Default values:

- Reltol = 1e-3
- Abstol = 1e-6

```
[T,X]=ode45(@dynamics,tspan,X0,options)
```

```
[T,X]=ode113(@dynamics,tspan,X0,options)
```

NB: Sometimes ode113 works better than ode45 for astrodynamics problems

Important notes

- In dyn_2BP.m use mu as an external parameter
  [T,X]=ode45(@dynamics,tspan,X0,options,parameters)
  [X_d]=dynamics(t,X,parameters)
- Use vector notation in Matlab
  r=X(1:3); v=X(4:6)
- At the beginning of the main script use clc; clear; (clear allows using debug mode)
- Use debug mode to validate your code
- help function to read the help of a Matlab function