Exercise 09: Symbolic Computation with Sympy

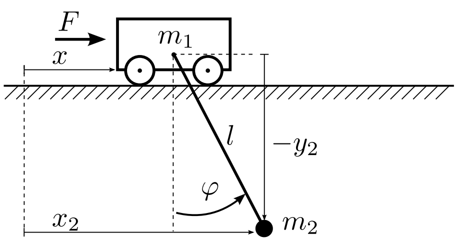
The aim of the exercise is to get acquainted with the package `sympy` using the example of derivation of the equations of motion of a mechanical system ('Euler-Lagrange equations').

Prerequisites

Sympy: symbols, functions, differentiating, substituting, solving equations.
Python: loops, lists and tuples, dictionaries.

System under consideration: 2D crane with fixed rope length

The equations of motion are a system of differential equations which describe (for the depicted mechanical system) the relation between the time-dependent quantities $x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$, $\varphi(t)$, $\dot{\varphi}(t)$, and $\ddot{\varphi}(t)$. They can be derived by evaluating the so-called Euler-Lagrange equations. The necessary calculation steps are to be carried out with `sympy`. The parameters m_1, m_2, l and g are assumed to be constant and known.



coordinates: $q_1(t) = x(t), \;\; q_2(t) = \varphi(t)$
auxiliary geometric quantities:
$x_2(t) := x(t) + l\sin\varphi(t)$, $y_2(t) := -l\cos\varphi(t)$
constant parameters: m_1, m_2, l, g
kin. energy: $T = \frac{1}{2}m_1\dot{x}(t)^2 + \frac{1}{2}m_2\left(\dot{x}_2(t)^2 + \dot{y}_2(t)^2\right)$
pot. energy: $U = m_2 g y_2(t)$
Lagrange function: $L(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = T - U$

Lagrange equations

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_1} - \frac{\partial L}{\partial q_1} = Q_1 \qquad (1a)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_2} - \frac{\partial L}{\partial q_2} = Q_2 \qquad (1b)$$

External forces and torques: $Q_1 = F, \;\; Q_2 = 0$

General Remarks:

- Physical understanding of the task and knowledge of the "Lagrange" method are helpful but **not** mandatory.
  The subtasks give the solution.
- Follow the given script (`lagrange.py`) and the comments contained in it.
- Pay attention to `sys.exit()` and move it down step by step. (The source code after that is still incomplete for now).
- Choose variable names that are as meaningful as possible.
- If you find this exercise too complicated, you can safely skip it.

Tasks

1. Create all required symbols for the constant parameters (m_1, ...).
2. Create time functions for $x(t)$ and $\varphi(t)$.
3. Form the time derivatives $\dot{x}(t), \dot{\varphi}(t), \ddot{x}(t)$, and $\ddot{\varphi}(t)$.

4. Calculate the auxiliary geometric quantities $x_2(t), y_2(t)$ (formulas: see above).
5. Calculate T, U and L (formulas: see above).
6. Generate the following four auxiliary terms: $\frac{\partial L}{\partial q_1(t)}$ and $\frac{\partial L}{\partial \dot{q}_1(t)}$ and $\frac{\partial L}{\partial q_2(t)}$ and $\frac{\partial L}{\partial \dot{q}_2(t)}$.
7. Calculate $\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i}$ (one term each for $i = 1$ and $i = 2$).
8. Now construct the two equations of motion

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = F \quad \text{and} \quad \frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} - \frac{\partial L}{\partial \varphi} = 0.$$

   Note: These two equations form a linear algebraic system of equations with respect to the accelerations $\ddot{x}$ and $\ddot{\varphi}$.

9. Solve the linear system of equations for the accelerations with `res = sp.solve(..)`, so that two equations $\ddot{x} = \dots$ and $\ddot{\varphi} = \dots$ result (respectively the right sides of these equations).
10. Show the data type of `res` and the obtained expressions for $\ddot{x}$ and $\ddot{\varphi}$ e.g. using `sp.pprint(...)`.
11. For both expressions, use `sp.lambdify(...)` to generate a function to calculate the respective acceleration.

    Notes: First substitute

    - the time functions and their derivatives by appropriately named symbols (starting with the highest derivative order, see course slides resp. example-notebook).
    - the system-parameters with the following numeric values:
      `[(m1, 0.8), (m2, 0.3), (l, 0.5), (g, 9.81)]`.

    → The expressions then depend only on the following five symbols: the force F, the coordinates (x, φ) and the velocities $(\dot{x}, \dot{\varphi})$. The Pythonfunctions created by `lambdify` will be needed to simulate the system in the next exercise.