# Exercise 8: Nonlinear Least Squares + Linear Kalman Filter
(Solution of Task 1)

Prof. Dr. Moritz Diehl, Katrin Baumgärtner, Naya Baslan, Jakob Harzer, Doga Can Öner

Parameter estimation for output error minimization   **(5 points + 2 bonus points)**

In this exercise we will consider the model of a differential drive robot with unicycle dynamics. The movement of the robot depends on the angular velocities of the left and the right wheel ω_{L} and ω_{R}, as well as on their radii R_{L} and R_{R}. Differing radii influence the behaviour of the robot.
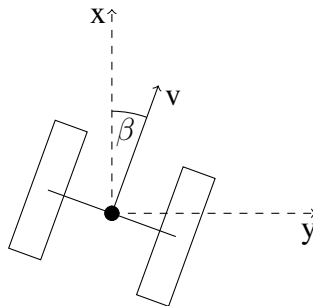
The system can be described by a state space model with three internal states. The state vector $\mathbf{x} = [x,\ y,\ \beta]^\top$ contains the position of the robot in the $X - Y$ plane and the deviation β from its initial orientation. The system can be controlled by the angular velocities of the wheels: $\mathbf{u} = [\omega_{\mathrm{L}},\ \omega_{\mathrm{R}}]^\top$. The output of the system is the position of the robot: $\mathbf{y} = [x,\ y]^\top$. The model follows as

$$\dot{\mathbf{x}} = \begin{bmatrix} v \cdot \cos \beta \\ v \cdot \sin \beta \\ \frac{\omega_{\mathrm{L}} R_{\mathrm{L}} - \omega_{\mathrm{R}} R_{\mathrm{R}}}{L} \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{1}$$

with L being the length of the axis between the two wheels and the velocity v being

$$v = \frac{\omega_{\mathrm{L}} \cdot R_{\mathrm{L}} + \omega_{\mathrm{R}} \cdot R_{\mathrm{R}}}{2}.$$

The system state is $\mathbf{x} = [x, y, \beta]^\top$ and is equal to the robot's pose. The system can be controlled by the angular velocities of the wheels: $\mathbf{u} = [\omega_L\ \omega_R]^\top$. The output $\mathbf{y}$ is the position of the robot and measured with a sampling time of $\Delta t = 0.01$ s.

    We already provided you some functions to simulate the position of the two-wheel-robot using the state space model. For reference use chapter 6.2 'Numerical Integration Methods' from the lecture notes. Please go through all of the provided functions and try to understand what they are doing.

- `[xdot] = robot_ode(x,u,p)` evaluates the right-hand side of the ODE $\dot{x} = f(x, u, p)$, with parameters $p = [R_L, R_R, L]$. Use the following values: $R_L = 0.2$ m, $R_R = 0.2$ m and $L = 0.6$ m.

- `[x_next] = RK4_step(h, x0, u, ode, p)` performs one RK4 integration step for a general ODE $\dot{x} = f(x, u, p)$ starting at x_0, with input u, parameters p and step size h.

- `[x_sim] = sim_RK4(t,x0, u, ode, p)` simulates the robot's behaviour at times $\mathbf{t}$ given a set of inputs $\mathbf{u}$, starting at $x_0 = [0\ 0\ 0]^\top$.

In this task, we would like to estimate the dimensions of the robot $\theta = [R_L, R_R, L]^\top$ using `lsqnonlin`[1]. Assuming that the robot system has only output errors, and that these errors are Gaussian with zero mean and variances $\sigma_x^2 = 1.6 \cdot 10^{-3}$ m^2 and $\sigma_y^2 = 4 \cdot 10^{-4}$ m^2, then the Maximum Likelihood Estimation problem to estimate θ is:

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^3} \sum_{k=0}^{N} \|\mathbf{y_k} - M_k(\mathbf{U}, \mathbf{x_0}, \theta)\|_{\Sigma_y^{-1}}^2,$$

where $\mathbf{y_k} = (x, y)^\top \in \mathbb{R}^2$ with x and y being the coordinates of the robot and N is the number of measurements; Σ_y is the weighing matrix containing the variances on the x and y measurements,

$$\Sigma_y = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix};$$

$M_k(\mathbf{U}, \mathbf{x_0}, \theta)$ denotes the modeled position at timestep k for given $\mathbf{U}, \mathbf{x_0}, \theta$ where $\mathbf{U} \in \mathbb{R}^{(N-1) \times 2}$ is a matrix that contains all applied control inputs $u_1, \ldots, u_N$, each consisting of the angular velocity of the left and right wheel respectively (ω_L and ω_R); $\mathbf{x_0}$ contains the robot's initial pose $\mathbf{x_0} = [x_0, y_0, \beta_0]^\top = [0, 0, 0]^\top$ which we assume to be perfectly known.

1. ON PAPER: Formulate the output model

$$M_k : \mathbb{R}^{(N-1) \times 2} \times \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^2, \ (\mathbf{U}, \mathbf{x_0}, \theta) \mapsto \hat{\mathbf{y}}_\mathbf{k}$$

where you may use a function $F : \mathbb{R}^3 \times \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^3$ to denote the discretized system dynamics describing the mapping $(x_k, u_k, \theta) \mapsto x_{k+1}$.   (1 point)

Solution: First, we use the discretized system dynamics F to define the k-step forward simulation F_k:

$$
\begin{aligned}
x_0 &= F_0(x_0, \theta) && := x_0 \\
x_1 &= F_1(x_0, u_0, \theta) && := F(x_0, u_0, \theta) \\
x_2 &= F_2(x_0, u_0, u_1, \theta) && := F(F(x_0, u_0, \theta), u_1, \theta) \\
x_3 &= F_3(x_0, u_0, u_1, u_2, \theta) && := F(F(F(x_0, u_0, \theta), u_1, \theta), u_2, \theta) \\
&\ \ \vdots \\
x_k &= F_k(x_0, u_0, u_1, u_2, \ldots, u_{k-1}, \theta) && := F(\ldots F(F(F(x_0, u_0, \theta), u_1, \theta), u_2, \theta) \ldots), u_{k-1}, \theta)
\end{aligned}
$$

Using F_k, we can formulate the output model $M_k : (\mathbf{U}, \mathbf{x_0}, \theta) \mapsto \hat{\mathbf{y}}_\mathbf{k}$ as

$$M_k(\mathbf{U}, \mathbf{x_0}, \theta) := C \cdot F_k(x_0, u_0, u_1, u_2, \ldots, u_{k-1}, \theta)$$

with

$$C := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

Note that M_k depends only on $u_0, \ldots, u_{k-1}$, but we still use $\mathbf{U}$ here to keep the notation simple.

---

[1] `lsqnonlin` takes as input a vector function $f(\theta) = [f_1(\theta), \ldots, f_N(\theta)]$, and minimizes $\|f(\theta)\|_2^2$ with respect to θ. Thus, you have to stack the residuals obtained for different timesteps to obtain a *single* residual vector.