Exercises for Course on Modeling and System Identification (MSI)
Albert-Ludwigs-Universität Freiburg – Winter Term 2020-2021

Exercise 7: Recursive Least Squares

(to be returned on Jan 20th, 2021, 9:00)

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

In this exercise you will implement a Recursive Least Squares (RLS) estimator and a forward simulation of a differential drive robot with unicycle dynamics. We will apply the RLS algorithm to position data of a 2-DOF movement in the X-Y plane, measured with a sampling time of $0.0159\,\mathrm{s}$.

1. **Recursive Least Squares applied to position data**

   In this task you will implement the Recursive Least Squares (RLS) algorithm in MATLAB and tune the *forgetting factors*. We approximate the position data by a fourth order polynomial in order to obtain a linear-in-the-parameters (LIP) model. You can assume that the noise on the X and Y measurements is independent. The experiment starts at $t = 0\,\mathrm{s}$.

   (a) MATLAB: Fit a 4-th order polynomial through the data using linear least-squares. Plot the data and the fit for the X- and Y-coordinate.
   Hint: You need one estimator for each coordinate.
   PAPER: Does the fit seem reasonable? Why do you think that is? (1 point)

   (b) MATLAB: Implement the RLS algorithm as described in the script *(Check section 5.3.1)* to estimate 4-th order polynomials to fit the data. Do not use forgetting factors yet. Plot the result against the data.
   PAPER: Compare the LS estimator from (a) with the RLS estimator you obtain after processing N measurements. Please give an explanation for your observation. (2 points)

   (c) MATLAB: Add a forgetting factor α to your algorithm and try different values for α. Plot the results on the same plot as the previous question.
   PAPER: How does α influence the fit? What is a reasonable value for α? (1 point)

   (d) PAPER: How can you compute the covariance Σ_p of the position, if you know the covariance of the estimator $\Sigma_{\hat{\theta}}$?
   Hint: For a random variable $\gamma = A\theta$, where A is a matrix, $\mathrm{cov}(\gamma) = A\mathrm{cov}(\theta)A^{\mathrm{T}}$. (1 point)

   (e) MATLAB: Compute the *one-step-ahead* prediction at each point (i.e. extrapolate your polynomial fit to the next time step). We also provided code to plot the 1-σ confidence ellipsoid around this point, and the data.
   PAPER: Do the confidence ellipsoids grow bigger or smaller as you take more measurements? Why do you think that is? (2 points)

2. **Covariance approximation**
   Consider a nonlinear function $f : \mathbb{R}^n \to \mathbb{R}$ that maps a random vector $X = (X_1, \ldots, X_n)^\top$ to a scalar random variable Y, i.e.
   $$Y = f(X) = f(X_1, \ldots, X_n).$$
   We have $\mathbb{E}\{X\} = \mu_x = (\mu_1, \ldots, \mu_n)^\top$ and $\mathrm{cov}(X) = \Sigma_x \in \mathbb{R}^{n\times n}$.

   (a) ON PAPER: Give an approximation of the expected value $\mathbb{E}\{Y\}$ and the covariance matrix $\mathrm{cov}(Y)$ of Y using a first order Taylor expansion of f around μ_x. (2 points)

(b) ON PAPER: Suppose $X_1, \ldots, X_n$ are independent. Simplify your covariance approximation from part (a). (1 point)

This sheet gives in total 10 points