System Identification – Practical Assignment 8
Identifying an output-error model of the DC motor with Gauss-Newton optimization

Logistics

Please reread the logistics part of lab 2, the same rules will apply to this lab. The only things that change are the Teams assignment, which for this lab is "Lab 8 (OE)", and of course the lab number in the file name.

Assignment description

In this assignment we will identify an OE model of the DC motor system using the prediction error method. See the course material, Part VII: *General Prediction Error Methods*.

We know from the physics of the DC motor that it is a first-order system, and due to serial communication issues the system may exhibit a time delay of a few steps. Due among other things to the usage of discrete differences to compute the rotation velocity output, this signal is affected by noise. This means that the following Output Error form is appropriate to model it:

$$y(k) = \frac{b}{1 + fq^{-1}} u(k - nk) + e(k)$$

with the parameters $\theta = [f, b]^\top$. Our objective will be to implement the prediction error method for this particular model structure, using Gauss-Newton optimization. The algorithm is summarized at the end of this description so as to help with implementation. Note that, differently from the lectures, we need to take particular care to handle the delay nk; in particular, since the error and its derivative will depend on $u(k - nk)$, we start the recursion at $k = nk + 1$.

Requirements:

- Find the recursion formulas required by the algorithm, on paper or at the whiteboard. **Hint:** While imposing $\varepsilon(k) = e(k)$, find $\varepsilon(k)$ using the model equation, and then compute the partial derivatives with respect to the two parameters.
- To keep things simple, we will create a single, longer sequence of data containing both the identification and validation data. We will use a sampling rate of 0.01 s (10 ms). After a short range of zero inputs, apply a PRBS signal with amplitudes in the interval $[-0.7, 0.7]$ and a length of about 200 samples, followed by another range of zero inputs, and then a step signal of magnitude around 0.4 and around 70 samples in length. You can either use `idinput` or your own PRBS code from the previous labs, but in the latter case use a sufficiently large number of bits so that the signal does not repeat.
- Apply the signal generated to the DC motor. The output is the rotational velocity. Isolate the data range corresponding to the PRBS input and copy it to new input and output vectors; this will be our identification data. **Important note**: To minimize system wear, separate the code that generates the data from the code that performs the rest of the steps below (easiest using different script sections, see *Code Sections* in the Matlab documentation), and regenerate the data only when necessary.

- Plot and examine the data obtained.
- Implement the algorithm, and run it on the identification data starting from *nonzero* initial parameters, while tuning the delay nk.
- For the near-optimal values of f and b obtained, create an OE model in the system identification toolbox format, using `idpoly`. The syntax of this function is `idpoly(A,B,C,D,F,0,Ts)` where you need to specify the nk leading zeros in B, the leading 1 in F, and the sampling time. The polynomials that you don't use can be set to 1. Use `compare` to see how the model performs on the validation data.
- If the model is not satisfactory, tune α, δ and $\ell_{\max}$ (as well as perhaps θ_0), so as to improve performance.

---

choose stepsize α, initial parameters θ_0, threshold δ, and max iterations $\ell_{\max}$
initialize iteration counter $\ell = 0$
compute recursion formulas for $\varepsilon(k)$, $\frac{d\varepsilon(k)}{d\theta} = [\frac{d\varepsilon(k)}{df}, \frac{d\varepsilon(k)}{db}]^\top$
repeat
  with the current parameters θ_ℓ:
  initialize $\varepsilon(k) = y(k)$, $\frac{d\varepsilon(k)}{d\theta} = [0,0]^\top$ for $k = 1, \ldots, nk$
  for $k = nk+1$ to N **do**
    apply recursion formulas to find $\varepsilon(k)$, $\frac{d\varepsilon(k)}{d\theta}$
  end for
  compute gradient of the objective function, $\frac{dV}{d\theta} = \frac{2}{N-nk}\sum_{k=nk+1}^{N} \varepsilon(k)\frac{d\varepsilon(k)}{d\theta}$
  compute approximate Hessian of the objective function, $\mathcal{H} = \frac{2}{N-nk}\sum_{k=nk+1}^{N} \frac{d\varepsilon(k)}{d\theta}\left[\frac{d\varepsilon(k)}{d\theta}\right]^\top$
  apply Gauss-Newton update formula: $\theta_{\ell+1} = \theta_\ell - \alpha\mathcal{H}^{-1}\frac{dV}{d\theta}$
  increment counter: $\ell = \ell + 1$
until $\|\theta_\ell - \theta_{\ell-1}\| \le \delta$, or $\ell_{\max}$ was reached

---