Some Practical Exercises for System Identification

Fall 2017

Introduction

The objective of the computer exercise sessions and the practical project is to implement different algorithms learned during the lectures and make the student familiar with MATLAB and its system identification toolbox. Three computer exercises and one project are scheduled. For each computer exercise the groups of two students should prepare a report and submit it in the *moodle* of the course in due time. The reports will be corrected by the assistants and if necessary comments are added. The report of the final project will be corrected by the teacher and has 40 points. The final grad will be given after individual meeting with each student and a deep evaluation of the reports in January 2018. A time schedule for the evaluation meeting is prepared by the end of December 2017.

1 CE-1 : Time-domain methods

In order to solve the following exercises, you will require Matlab2014 or later. Make sure your version is up to date. During the first session of the exercises you will create a Simulink model of a second order transfer function, which is a very common model that can be found in many applications.

1.1 Step response

1. Create a new Simulink model. Create a second order `Transfer Function` block with the following parameters :

$$G(s) = \frac{4}{s^2 + s + 4}$$

2. Define the sample time $T_e = 0.2$ s (Is this a reasonable choice ?).
3. Simulate measurement noise by creating a `Random Number` block with a variance of 0.1 and a sample time of T_e and adding it to the output of the transfer function block.

4. Create an input `Saturation` block with an upper limit of 0.5 and a lower limit of -0.5.
5. Finish the block diagram by creating a `From Workspace` source and a `To Workspace` sink to exchange data with the workspace. Set the sample time of the From Workspace and To Workspace blocks to T_e.

6. Apply a step with a magnitude equal to the upper saturation limit at the input and plot the response of the system. To do this, create an M-file. First we need to generate the input signal. Create a struct `simin` with the following fields :
   – `simin.signals.values` : A column vector representing the input signal (i.e. the step)
   – `simin.time` : The time vector corresponding to the input signal

   The time vector should have a length of 50 s and the step should occur after 1 s. Note that the simulation time should be at least 50 s (in the simulation parameters of Simulink).
7. Use the command `sim` to run the Simulink model with the above input signal and plot the output of the system.
8. In the same way compute the impulse response of the system.

1.2 Auto Correlation of a PRBS signal

1. Download the file `prbs.m` from the Moodle page of the course. The function `prbs(n,p)` generates a PRBS of `p` periods with an `n`-bit shift register.
2. Write a function for Matlab `[R,h] = intcor(u,y)` that computes $R_{uy}(h)$ for the periodic signals.
3. Check your function by computing the autocorrelation of a PRBS signal.

1.3 Impulse response by deconvolution method

The objective is to compute the impulse response of the system in simulink using the numerical deconvolution method. Note that the simulation time should be less than 100 s and the random signal should not exceed the saturation values. For this purpose,

1. Generate a random signal (see `help rand`) with a sampling time of $T_e = 0.2$s as the input to the model.
2. Use `toeplitz` command to construct the input matrix.
3. Generate a time vector for simulink : `t=0:Te:(N-1)*Te`, where N is the length of your input signal.
4. Compute the impulse response using the measured output and the matrix of the input signal.
5. Compare your results with the true impulse response of the discrete-time system obtained by `zoh` option for transformation (see `tf, c2d, impulse`).

1.4 Impulse response by correlation approach

1. Generate an input sequence `Uprbs` using the `prbs` command (with $p = 4$ and $n = 7$) and apply it to your system. Set the sample time to $T_e = 0.2$ s.
2. Using your `intcor` function that you have already developed, compute the impulse response of the discrete-time system $g(k)$ using the correlation approach.
3. Compute the impulse response using `xcorr` function of Matlab.
4. Compare your results (using `xcorr` and `intcor`) with the true impulse response of the discrete-time system.

2 CE-2 : Frequency domain methods

The objective of this exercise is to identify a frequency-domain model for our system in CE-1. The sampling period should be set to $T_e = 0.2$s and the data length of experiment should be about $N \approx 2000$.

2.1 Frequency domain Identification (Periodic signal)

Aim : Use the Fourier analysis method to identify the frequency response of a model excited by a PRBS signal.

1. Choose a PRBS signal and apply it to the Simulink model of CE-1.
2. Compute the Fourier transform of the input and output signal (use `fft`) for each period and use the average.
3. Compute a frequency vector associated to the computed values (Slide 30, Chap. 2).
4. Generate a frequency-domain model in Matlab using the `frd` command.
5. Plot the Bode diagram of the identified model and compare it with the true one.

2.2 Frequency domain Identification (Random signal)

Aim : Use the spectral analysis method to identify the frequency response of a model excited by a random signal.

1. Apply a random signal of length $N = 1000$ to the system. Discuss the characteristics of the random signal (Gaussian or uniform, binary or multi-level) to obtain the highest energy of the excitation signal.
2. Compute the frequency-response of the model using the spectral analysis method.
3. Use a Hann or Hamming window to improve the results.
4. Cut the data to m groups and try to improve the results by averaging.
5. Plot the Bode diagrams of different methods and compare them.

3 CE-3 : Parametric Identification Methods

The objective of this exercise is to practice the parametric identification methods.

3.1 Identification of a laser beam stabilizing system

The Laser Beam Stabilization device consists of a low-power stationary laser beam source pointing at a single-axis, fast steering mirror. The reflected beam is picked up by a high-resolution position sensing detector (PSD). The PSD measures the relative displacement of the beam from the nominal position and the mirror mechanism is actuated using a high-bandwidth voice coil.

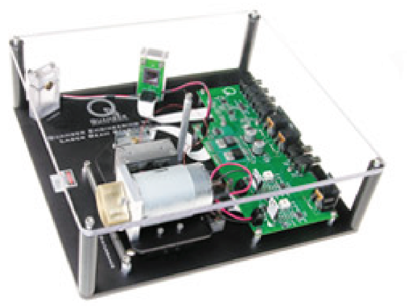

Figure 1. Laser beam stabilizing system

Since the open-loop system contains an integrator, a proportional controller is used to stabilize the system. The objective is to identify the whole closed-loop system. The excitation signal u is a PRBS and the beam position y is measured with a sampling frequency of 1 kHz. The input and output signal are saved in `laserbeamdataN.mat`. This file can be downloaded from the moodle of the course.

3.2 FIR model identification

Assume an FIR model for the data with $m = 50$ parameters :

$$\hat{y}(k,\theta) = b_1 u(k-1) + \cdots + b_m u(k-m)$$

1. Compute the vector of parameters $\hat{\theta}^T = [b_1 \ \dots \ b_m]$ using the least squares algorithm. Note that the matrix Φ is a Toeplitz matrix.
2. Compute the predicted output of the identified model, $\hat{y}(k,\hat{\theta})$. Plot the measured output $y(k)$ and the predicted output in the same figure and compute the loss function $J(\hat{\theta})$.
3. Compute the covariance of the parameter estimates assuming that the noise is white. Plot the finite impulse response of the system (the vector θ) together with $\pm 2\sigma$ confidence interval (use `errorbar`).

3.3 ARX model identification

Assume a second order ARX model for the data with the following predictor :

$$\hat{y}(k,\theta) = -a_1 y(k-1) - a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2)$$

1. Compute the vector of the parameters $\hat{\theta}^T = [a_1\ ,\ a_2\ ,\ b_1\ ,\ b_2]$ using the least squares algorithm.
2. Compute the predicted output of the identified model, $\hat{y}(k,\hat{\theta})$. Plot the measured output $y(k)$ and the predicted output in the same figure and compute the loss function $J(\hat{\theta})$.
3. Compute the output of the identified model, $y_m(k)$, for the input signal, $u(k)$, using `lsim`. Plot the measured output $y(k)$ and $y_m(k)$ in the same figure and compute the sum of squares of the error.
4. Try to improve the results using the Instrumental Variable method (note that $y_m(k)$ is a noiseless estimate of $y(k)$). Compare the result with that of ARX model.

3.4 State-space model identification

The objective of this part is to identify a sate-space model for the laser beam stabilizing system using the subspace method based on the extended observability matrix.

1. Construct the matrix $\boldsymbol{Y}$ and $\boldsymbol{U}$ based on the measured data (see Eq. 3.69 and 3.70 in the course-notes) and compute $Q = \boldsymbol{Y}\boldsymbol{U}^{\perp}$.
2. Compute the singular values of Q and conclude the number of states n (use `svd`). Compute the observability matrix O_r (the first n columns of Q).
3. Compute the matrix A and C from the observability matrix.
4. Assume that $D = 0$ and estimate B using the least squares algorithm.
5. Compute the output of the identified model, $y_m(k)$ for the input signal $u(k)$ using `lsim`. Plot the measured output $y(k)$ and $y_m(k)$ in the same figure and compute the the sum of squares of the error.

3.5 Parametric Identification of a Flexible Link

The objective of this part is to identify and validate a parametric black-box model for a rotary flexible link. In the first part the order of the system id identified using the data and in the second part different parametric models are identified and validated.

The system contains two units : the Rotary Servo Base Unit which is a geared servo-mechanism system and a thin stainless steel flexible link (see Fig. 1). The Rotary Servo Base Unit consists of a DC motor that drives a small pinion gear through an internal gearbox. The pinion gear is fixed to a larger middle gear that rotates on the load shaft. The chassis of the Rotary Flexible Joint is mounted on the load gear of the Servo Base Unit and can rotate freely. The DC motor on the Rotary Servo Base Unit is used to rotate the flexible link from one end in the horizontal plane. The motor end of the link is instrumented with a strain gage that can detect the deflection of the tip. The strain gage outputs an analog signal proportional to the deflection of the link.

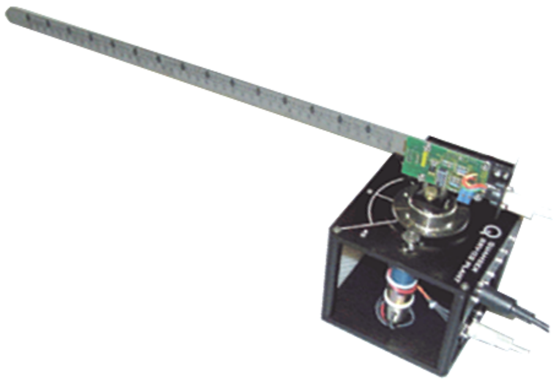

Fig. 1 Flexible link system

A model between the current reference for the DC motor (which is proportional to its mechanical torque) and the tip position of the flexible link should be identified. Download the file `CE3.mat` that contains the experimental data of the flexible link system containing the input signal `u` and the sampled output signal `y` with a sampling period ($T_e = 0.015$ s). Use the command `iddata` to generate a data object that can be used for the System Identification Toolbox. Remove the mean value of the data using the command `detrend`.

Remark : Matlab uses the following notation :

$$A(q^{-1})y(t) = \frac{B(q^{-1})}{F(q^{-1})}u(t-n_k) + \frac{C(q^{-1})}{D(q^{-1})}e(t)$$

where :

$$\begin{aligned} A(q^{-1}) &= 1 + a_1q^{-1} + \cdots + a_{n_a}q^{-n_a} & B(q^{-1}) &= b_1 + b_2q^{-1} + \cdots + b_{n_b}q^{-n_b+1} \\ C(q^{-1}) &= 1 + c_1q^{-1} + \cdots + c_{n_c}q^{-n_c} & D(q^{-1}) &= 1 + d_1q^{-1} + \cdots + d_{n_d}q^{-n_d} \\ F(q^{-1}) &= 1 + f_1q^{-1} + \cdots + f_{n_f}q^{-n_f} \end{aligned}$$

For ARX structure we have $n_c = n_d = n_f = 0$, for ARMAX $n_d = n_f = 0$, for OE : $n_a = n_c = n_d = 0$ and for BJ : $n_a = 0$.

3.5.1 Order estimation

1. Using the `arx` command, identify 10 models of orders from 1 to 10 ($n_k = 1, n_a = n_b = n, n = 1, \ldots, 10$) and plot the "loss function" of the models versus the model order (loss function is available for the model `M` in the following variable : `M.EstimationInfo.LossFcn`). Estimate the order of the system from this curve.
2. Validate the chosen order by checking the zero/pole cancellation approach using several models identified by the ARMAX structure ($n_k = 1, n_a = n_b = n_c = n, n = n_{\min}, \ldots, n_{\max}$). Use `h=iopzplot(M)` to plot the zeros and poles of M and `showConfidence(h,2)` to plot their confidence intervals ($\pm 2\sigma$).

3. Estimate the delay n_k by inspecting the coefficients of $B(q^{-1})$ and their standard deviations (use `M.b` and `M.db`. Compute the number of parameters in the numerator $(1 \leq n_b \leq n - n_k + 1)$.
4. Compare your results with those proposed by `struc, arxstruc, selstruc`.

3.5.2 Parametric identification and validation

1. Divide the data into two parts and use one part for identification and the other part for validation.
2. Using the estimated values of n_a, n_b and n_k, identify different parametric models by the following methods : `arx, iv4, armax, oe, bj, n4sid`. For the structures with noise model take $n_c = n_d = n_a$. For the state-space model choose number of states equal to the global order n.
3. Compare the output of the identified models with the measured output using the command `compare`. What is the best model ?
4. Validate the identified models by the whiteness test of the residuals as well as the cross-correlation of the residuals and past inputs using the command `resid`. Which models are validated ?