System Identification
227-0689-00L
Midterm examination: instructions

Due: 10:00 on Wednesday, November 6$^{\text{th}}$, 2019

Overview

The midterm exam is composed of two problems. To answer these problems you need to write two separate MATLAB functions (the details will be provided in the following pages).

Each function will analyze the data associated with the corresponding problem part.

This work must be done individually. You may not discuss these problems with anyone else.

Your functions will be submitted by email. In this email you must also include a scan of the signed declaration of originality form. In the form the title of work should be specified as `System Identification midterm exam HS2019` and the **scanned** document should be named as `HS2019_SysID_midterm_D0_leginumber.pdf`, where '`leginumber`' must be your leginumber, without any dash, slash, backslash, etc., e.g.,

`HS2019_SysID_midterm_D0_11235813.pdf`

If you do not have a leginumber, before the due date of exam, you should send an email to `sysid@ee.ethz.ch` and ask for a temporary number to be assigned.

Your submission must be emailed to `sysid@ee.ethz.ch` by the due date and time given above. The subject of the email must be `SystemID HS2019 Midterm Exam`. The email should contain the exact given number of separate attachments, including the requested MATLAB m-files and the PDF (the details will be provided in the following pages).

Your grade for the midterm will be evaluated based on your performance in the two problems weighted in the following way: 40% for Problem 1 and 60% for Problem 2.

Downloadable data

On the Piazza course website

`https://piazza.com/ethz.ch/fall2019/227068900l/resources`

the required files for the individual parts will be provided. Here is the list of uploaded files:

File	Problem	Content
HS2019_SysID_midterm_p1_12345678.m	Problem 1:	Template file
HS2019_SysID_midterm_p1_GenerateData.p	Problem 1:	p-file
HS2019_SysID_midterm_p2_12345678.m	Problem 2:	Template file
HS2019_SysID_midterm_p2_system_sim.p	Problem 2:	p-file
HS2019_SysID_midterm_p2_validation.p	Problem 2:	p-file
HS2019_SysID_midterm_DO_12345678.pdf		pdf-file

MATLAB function format

You should use the provided templates for your solution to Problem 1 and Problem 2. Replace the number '12345678' with your own leginumber in both the file name of the template files as well as in the function name inside the template file.

In case you have functions of your own that you want to use in your solutions, include them as *local function* by putting them at the end of the main function. Do not use or submit them in separate files. You can find more details in help of MATLAB and also, here:

`https://www.mathworks.com/help/matlab/matlab_prog/local-functions.html`

This means that MATLAB treats them as subfunctions and calls them instead of any other functions in the same path with the same name. This also ensures that your solution calls your own functions and not another student's functions. Generally, it is recommended that you read the MATLAB documentation on functions and programming for examples of this.

The following are generic considerations for the behavior of your code. They apply to each of the problems.

- As your function runs it must print out in the command window (use the `disp` or `printf` functions) a description of what it is doing and what choices you are making (i.e. number of points, frequency range, DFT calculations, windowing, model order, model parameters, etc.). You will be graded on the quality of your explanation.

- For any figures requested in the problem use commands `figure(1)`, `figure(2)`, etc. to ensure that all figures remain visible on the screen after your code terminates. Plots that have been overwritten during the running of your algorithm will not count towards your grade.
- Each MATLAB figure can contain at most two plots.
- Specify all axis limits via the MATLAB command `axis(...)`. Auto-scaling must be avoided as it works differently on different machines.
- If your plot uses a legend use the `'Location'` flag in the `legend` command to make sure that relevant data is not covered.
- The script should run unattended and at the end the command window will describe what you have done, with any requested figures visible. Do not use `pause` command inside your functions.
- Do not define any global variables in your functions or subfunctions.
- Use the template files for your solutions. Do not change the order or names of the output variables in the template files. Add your solution to the template and replace the empty brackets, i.e. '`[]`', assigned to the output variables with your solution (keep the empty bracket in case you have no solution).
- You can assume that the functions `WfHann.m` and `WtHann.m` (from the Piazza class website) will be available in the Matlab path.

Before submitting your file, restart MATLAB and test it by running it in a folder/directory with only the auxiliary .p functions. This will ensure that it can run in a stand-alone manner.

Problem 1 (Weight: 40%)

To solve this problem, you should write a function of the following form

```
[p1_R, p1_omega, p1_a, p1_var] = HS2019_SysID_midterm_p1_LegiNumber()
```

using the provided sample solution file `HS2019_SysID_midterm_p1_12345678.m`. Modify *both* the file and the function names by replacing 12345678 with your Legi number. As visible from the sample solution, the function `HS2019_SysID_midterm_p1_GenerateData()` is called and a variable named `p1_U` is saved in your workspace. Note that this requires the file `HS2019_SysID_midterm_p1_GenerateData.p` to be in your working directory. The variable `p1_U` must be used to solve the problem tasks given below. Do not change the order or the names of the output variables as written in the template.

The goal of this problem is to identify an unknown signal $\mathtt{u}(t)$, which is a combination of M sinusoids. The signal $\mathtt{u}(t)$ can be decomposed into its components as

$$\mathtt{u}(t) = \sum_{i=1}^{M} \mathtt{p1\_a}[i] \cdot \cos(\mathtt{p1\_omega}[i] \cdot t)$$

where `p1_a` and `p1_omega` are vector-valued parameters to be identified, and M is the number of sinusoids in the signal.

The signal u(t) was measured using a sensor with sampling time $T_s = 0.5$ s. The sensor only worked during 5 distinct time windows, each window consisting of K consecutive time steps. Denoting by T the period of the underlying signal, you can assume that $K \geq 2T$, that is each measurement window contains at least 2 full periods of `u`. Measurement data are provided to you in the $5 \times K$ matrix `p1_U`. Each row i of the matrix corresponds to the K measurements obtained in data record i. The sensor's measurement noise can be assumed to be zero-mean Gaussian with a flat spectrum (in expectation).

You are now given the following tasks.

1) Identify the period T and compute the periodic autocorrelation $R_u(\tau)$ of the signal measured in the first window (first row in the matrix `p1_U`). Use the first T values of the signal for this computation. Your function should return a $1 \times T$ vector `p1_R` containing the autocorrelation vector $R(\tau)$, corresponding to lags $\tau = -\frac{T}{2} + 1, \ldots, \frac{T}{2}$.

   In the MATLAB command window, *concisely* explain the procedure you used to identify the period T and compute the autocorrelation vector `p1_R`. In addition, plot one period of the autocorrelation function in a separate MATLAB Figure.

2) Identify the input signal parameters `p1_omega` and `p1_a`. Return `p1_omega` and `p1_a` as $1 \times M$ vectors. The values of frequencies must be arranged in the *increasing order* of their magnitude, and the values in the vector `p1_a` must be corresponding to the frequencies in `p1_omega`. Remember that the frequencies `p1_omega` must be in [rad/sec].

Explain the procedure which was followed to identify the parameters in the MATLAB command window in a *concise* manner. Your explanation must include the procedure used to estimate the frequencies `p1_omega`, and the formulae used to estimate the vector `p1_a`.

3) Compute an estimate of the variance of the sensor's measurement noise, and return it as a scalar in the variable `p1_var`. *Concisely* explain the procedure which was followed to produce such estimate in the MATLAB command window.

Your grade for this problem will depend on:

- The correctness of the returned autocorrelation values `p1_R`.
- The clarity and accuracy of your explanation to compute the autocorrelation `p1_R`.
- The correctness and clarity of your autocorrelation plot.
- The correctness of the parameters of the input signal `p1_omega` and `p1_a`.
- The clarity and accuracy of your explanation to identify the parameters `p1_omega` and `p1_a`.
- The accuracy of the estimated noise variance `p1_var`
- The clarity and accuracy of your explanation to identify `p1_var`.
- The overall clarity and correctness of your explanations.

Problem 2 (Weight: 60%)

Write a MATLAB function of the following form:

```
[p2_umin, p2_umax, p2_M, p2_Np, p2_u_etfe, p2_y_etfe, p2_omega,
p2_estimate_no_window, p2_estimate_windowed, p2_gamma_best] =
            HS2019_SysID_midterm_p2_LegiNumber()
```

You should use the provided template solution `HS2019_SysID_midterm_p2_12345678.m` for writing your code. You need to modify both the file and the function names by replacing '`12345678`' with your own Leginumber. Do not change the order or the names of the output variables as written in the template.

Consider an identification experiment on a system of the form $y(k) = G(e^{j\omega})u(k) + v(k)$, with input $u(k)$, output $y(k)$, and output measurement noise $v(k)$ (the sampling time is 1s). The noise $v(k)$ is i.i.d. and Gaussian with variance $\sigma^2 = 0.25$. The plant $G(e^{j\omega})$ has lower and upper input saturation limits $u_{\min}$ and $u_{\max}$. Your task is to design an experiment and identify the plant $G(e^{j\omega})$.

We have supplied the following plant-function that simulates the above system:

`p2_y = HS2019_SysID_midterm_p2_system_sim(legi_num,p2_u).`

This plant-function takes as input:

- `legi_num`: Your Legi number, entered as shown in the Solution template file. The Legi number is used to create a plant for you to estimate.
- `p2_u`: An input signal $u(k)$ of your choice. See further details below.

The plant-function returns `p2_y`, which is the measured output $y(k)$ of the system.

For cross-validation, you are provided with a validation-function:

`p2_u_cv = HS2019_SysID_midterm_p2_validation(p2_M).`

This function returns a signal u_{cv}, which you should use for cross-validation experiments. The signal, u_{cv}, contains 6 periods of length `p2_M`.

Your MATLAB function should return the following outputs in the order listed:

- `p2_umin` (scalar, rounded to first decimal): Estimate of the lower limit of the plant $G(e^{j\omega})$ input saturation, $u_{\min}$.
- `p2_umax` (scalar, rounded to first decimal): Estimate of the upper limit of the plant $G(e^{j\omega})$ input saturation, $u_{\max}$.

- `p2_M` (scalar integer): Length M of one period of the designed PRBS input signal $u(k)$.
- `p2_Np` (scalar integer): Number N_p of periods in the designed PRBS input signal $u(k)$.
- `p2_u_etfe` (vector): Designed minimum length PRBS input signal $u(k)$.
- `p2_y_etfe` (vector): Output $y(k)$ obtained from the plant-function when applying `p2_u_etfe`.
- `p2_omega` (vector): Vector ω of equally-spaced frequencies in the range $(0, \pi)$ (not inclusive) [rad/sec].
- `p2_estimate_no_window` (vector): Unsmoothed plant estimate $\hat{G}_u(e^{j\omega})$ at the frequencies contained in ω.
- `p2_estimate_windowed` (vector): Smoothed plant estimate $\tilde{G}_u^{H_\gamma^*}(e^{j\omega})$ using the optimal choice of Hann window width γ^*.
- `p2_gamma_best` (scalar): The optimal choice of the Hann window width, γ^* based on a cross-validation with input signal $u_{\text{cv}}(k)$.

Using the provided plant-function and validation-function, you should do the following:

1. Compute an estimate for the plant's input saturation limits $u_{\min}$ and $u_{\max}$. The true limits lie in the range of $[-5, 5]$ and are multiples of 0.1. Provide your estimates rounded to the first decimal.

   In the MATLAB command window, *concisely* explain how you estimated the saturation limits.

2. Design the input signal $u(k)$ as a minimum length periodic PRBS signal of period length M, and number of periods N_p, such that

   - the frequency resolution is larger than $\frac{\pi}{200}$,
   - the variance of a unsmoothed transfer function estimate due to measurement noise is less than $\sigma^2 = 0.04$,
   - its application to ETFE gives an unbiased estimate in practice.

   In the MATLAB command window, *concisely* explain how you made your choice of period length and number of periods for $u(k)$.

3. Calculate the unsmoothed ETFE $\hat{G}_u(e^{j\omega})$ of the plant $G(e^{j\omega})$. Define the vector of frequencies ω for which $|U(e^{j\omega})| \neq 0$ covering the range $(0, \pi)$ (not inclusive) and return the estimate $\hat{G}_u(e^{j\omega})$ at each of the frequencies in ω.

   In the MATLAB command window, *concisely* explain how you calculated the unsmoothed ETFE. Display a Bode plot (gain and phase) of the unsmoothed ETFE. Specify the axis limits and labels of your plot.

4. Use frequency-domain Hann windowing to smooth the transfer function estimate $\hat{G}_u(e^{j\omega})$ obtained in Task 3. The aim thereby is to find the optimal Hann window width γ^*. To this end, use the cross-validation input signal $u_{\text{cv}}(k)$ which you can obtain from the

validation-function provided. With this, consider the following validation problem over the set $\gamma = [50 : 50 : 500]$.

$$\min_{\gamma} \quad ||Y_{\text{cv}}(e^{j\omega}) - \tilde{G}_u^{H_\gamma}(e^{j\omega})U_{\text{cv}}(e^{j\omega})||_2, \tag{1}$$

where $\tilde{G}_u^{H_\gamma}(e^{j\omega})$ is the Hann window smoothed estimate using γ. Return the γ^* which minimizes the error equation (1), and the corresponding $\tilde{G}_u^{H_\gamma^*}(e^{j\omega})$ estimate.

Show a MATLAB figure in which you plot the value of the error equation (1) obtained for each of the γ values. Specify the axis limits and labels of your plot.

In the MATLAB command window, *concisely* explain how and why the value of error equation changes with γ. Also, answer the following question: If you were taking more periods for your input signal $u(k)$, how would you expect the optimal Hann window width γ^* to change? And why?

Important notes:

- *Only use the provided validation input $u_{cv}(k)$ to validate your best estimate $\tilde{G}_u^{H_\gamma^*}(e^{j\omega})$.*
- *We expect the length of a concise explanation to be around 2-4 sentences.*
- *Make sure to have exactly 10 outputs, and to keep the order as displayed above and written in the template. If you don't have results for one or more of the outputs, keep the empty array as in the template.*

Your grade for this problem will depend on:

- The correctness of the outputs `p2_umin, p2_umax, p2_M` and `p2_Np, p2_omega`.
- The accuracy of the outputs `p2_estimate_no_window, p2_estimate_windowed` and `p2_gamma_best`. We will compare your provided estimates with the correct estimates obtained for your individual plant and your individual validation signal using the methods specified in the Task description.
- The correctness of the figures.
- The quality, clarity and accuracy of your explanations and answers in all parts.