Project Assignment
System Identification 2022-2023

Logistics

This MATLAB-based project assignment is a compulsory part of the System Identification course in the Control Engineering B.Sc. program of the Technical University of Cluj-Napoca. It will be graded and the mark counts for 30% in the final grade of the course (15% for part 1, and 15% for part 2). The assignment is carried out in groups of **three** students, and should take around 20 hours per person to solve, depending on your experience with MATLAB. Each group will receive their own data sets. To receive them, form groups and send as soon as possible an e-mail to the project teacher (Zoltán Nagy at `zoltan.nagy@aut.utcluj.ro`). Please, mention the name and email address of each member of the group.

The assignment consists of two parts. The evaluation is performed differently for the two parts, see each part for details. **Crucial rule:** it is strictly forbidden to copy code, text, or results from other students or from online resources. Automated tools are in place to check this, and there will be absolutely zero tolerance for copying: failing to obey this rule automatically and immediately leads to ineligibility for the exam. So, be extremely careful!

Part 1. Time series modeling using Fourier basis functions

We will consider a time series with the monthly quantities of a certain product sold by a store. This store mostly sells building engineering products, like pipes, fittings, boilers etc. Therefore, there are products that are sold mainly in autumn, like boilers, since people are preparing for the cold season, and others that are sold mainly in spring-time, for example irrigation systems. Moreover, there are products that have an increasing trend, for example products used for insulation are becoming more popular. The data records the time (in months) and quantity of product units sold for a certain multi-year duration.

The data set is given as a MATLAB data file, containing two vectors of identical sizes: `time`, where each element contains the index k of the month, and `y`, which contains the product quantity $y(k)$ corresponding to each month.

Prior to any modeling, the dataset must be split into an identification part and a validation part: the first 80% of the data should be used for identification, and the last 20% of the data should be kept for validation.

We will create models of the following type for this time series:

$$\hat{y}(k) = t_0 + t_1 k + \sum_{i=1}^{m} \left[a_i \cos\left(\frac{2\pi i\, k}{P}\right) + b_i \sin\left(\frac{2\pi i\, k}{P}\right) \right]$$

The formula includes both a first-order, *linear trend* component $t_0 + t_1 k$, and a *Fourier basis* with a configurable number of terms m. Note that the Fourier basis is used because the data is expected to exhibit periodicity (yearly, trimester-wise, etc.). In particular, we have monthly data and assume an at most yearly periodicity, therefore by default the period $P = 12$. Further, note that each Fourier term gives two basis functions, one with $\cos$ and one with $\sin$. The regressors of this model contain the linear-trend components (1 and k) and the Fourier basis functions ($\cos$ and $\sin$ of various frequencies); whereas the parameter vector is $\theta = [t_0, t_1, a_1, b_1, \ldots, a_m, b_m]^\top$, containing $2 + 2m$ parameters in total.

For example, for $m = 1$ and the chosen $P = 12$, the approximator has the form:

$$\begin{aligned}\hat{y}(k) =& t_0 + t_1 k + a_1 \cos\left(\frac{2\pi\, k}{12}\right) + b_1 \sin\left(\frac{2\pi\, k}{12}\right) \\ =& \varphi^\top(k)\theta = \left[1,\, k,\, \cos\left(\frac{2\pi k}{12}\right),\, \sin\left(\frac{2\pi k}{12}\right)\right] \cdot \begin{bmatrix} t_0 \\ t_1 \\ a_1 \\ b_1 \end{bmatrix}\end{aligned}$$

and for $m = 2$:

$$\begin{aligned}\hat{y}(k) =& t_0 + t_1 k + a_1 \cos\left(\frac{2\pi\, k}{12}\right) + b_1 \sin\left(\frac{2\pi\, k}{12}\right) + a_2 \cos\left(\frac{4\pi\, k}{12}\right) + b_2 \sin\left(\frac{4\pi\, k}{P}\right) \\ =& \varphi^\top(k)\theta = \left[1,\, k,\, \cos\left(\frac{2\pi k}{12}\right),\, \sin\left(\frac{2\pi k}{12}\right),\, \cos\left(\frac{4\pi k}{12}\right),\, \sin\left(\frac{4\pi k}{12}\right)\right] \cdot \begin{bmatrix} t_0 \\ t_1 \\ a_1 \\ b_1 \\ a_2 \\ b_2 \end{bmatrix}\end{aligned}$$

For a given m, model fitting consists of finding the optimal parameter vector θ^* so that $\hat{y}$ best matches y on the identification dataset, in a least-squares sense. This can be done with linear regression. Details can be found in the lectures, Part 2: *Mathematical Background*, see the linear regression sections.

The **requirements** are given next. Program such an approximator with configurable number of Fourier terms m. Try to fit approximators with varying $m = 1, \ldots, 7$, so as to obtain the most accurate one. Validation (also for the purpose of comparing e.g. different values of m) should always be performed on the different, validation dataset. Report the mean squared errors as a function of m for both sets and show a *representative plot* for the fit on the training and the validation data sets (true values compared to approximator outputs, for the best value of m). *Discuss* the results, including the choice of m and the quality of the model fit on the two data sets, relating them to the discussion during lectures on model choice and overfitting in regression.

Your report must be written coherently, concisely, and in a self-contained manner. It should include at least the following elements:

- An introductory part, including a description of the problem.
- A brief description of the approximator structure, and the procedure to find the parameters.
- Any key features of your own individual solution (do not include trivial implementation details).
- Tuning results (at least the MSE as a function of m, either as a graph or a table).
- The representative plots pointed out above, for the optimal value of m.
- The discussion pointed out above, and an overall conclusion.

A more detailed guide on report writing and style in general is available on the course website.

Evaluation of part 1

Part 1 must be worked out in the form of a short written report (in English, one report per group), with the associated code. The deadline for the report and code is **November 13th 2022, 24:00**. In case of

delays, each newly entered day of delay results in a 2 point decrease in the maximum grade (for instance, delivering the report on November 15th at 00:10 AM leads to a maximum grade of 6 since the second day of delay has been entered).

Please **pay attention and follow to the letter** the following rules for delivery. A uniformized, semi-automated processing of solutions is essential for efficient grading, and any deviation from the rules makes your submission require additional, manual processing time, which will likely be unavailable and may therefore mean that your solution cannot be graded!

- Your submission must consist of exactly two files, named exactly like this: LN1LN2LN3.pdf and LN1LN2LN3.zip, where LNi is the last (family) name of student "i" in your project group, without diacritics. For example: IonescuFarkasBonta.pdf and IonescuFarkasBonta.zip.
- The first file is the report, which must be delivered **in PDF format** (not DOCX or any other source format; PDF only).
- For identification purposes, the first page of the report should prominently include names and the datafile index of your group.
- It is required to include in the report complete listings of the MATLAB code (functions and scripts) that you developed for solving the assignment problems.
- The second file contains your code itself, sent separately as a **ZIP archive** (not RAR, and not 7Zip or other formats; classical ZIP only). Important: There should be no subdirectories in this archive, all the m-files must be top-level.
- These files will be submitted via a DropBox file request, the link to which will be supplied before the deadline. Do not submit multiple versions as these cannot be taken into account; only your first submission will be considered, so make sure it is complete and correct.
- The creation date of the file on DropBox is taken for the purpose of delay computation per the rules above.

Part 2. Nonlinear ARX identification

To work on this part of the project, you need to understand first *linear* ARX models, which are handled in the lectures, part *ARX identification*. Nonlinear ARX models are also introduced in the appendix of that part.

A dataset is given, measured on an unknown **dynamic system** with one input and one output. The order of the dynamics is not larger than three, and the dynamics may be nonlinear while the output may be affected by noise. Your task is to develop a black-box model for this system, using a nonlinear ARX model that is a polynomial in the previous inputs and outputs. A second data set measured on the same system is provided for validating the developed model. The two data sets will be given in a MATLAB data file, with variables `id` and `val` containing the two sets as objects of type `iddata` from the System Identification toolbox. Recall that the input, output, and sampling time are available on fields `u, y, Ts` respectively. Only as a backup in case the system identification toolbox is not installed on the computer, `id_array` and `val_array` contain the same two datasets but now in an array format, with the structure: time values on the first column, input on the second column, and output on the last column.

Consider model orders na, nb, and delay nk, following the convention of the `arx` MATLAB function. Then, the nonlinear ARX model is:

$$\begin{aligned}\hat{y}(k) &= p(y(k-1),\ldots,y(k-na),u(k-nk),u(k-nk-1),\ldots,u(k-nk-nb+1)) \\ &= p(d(k))\end{aligned} \tag{1}$$

where the vector of delayed outputs and inputs is denoted by $d(k) = [y(k-1),\ldots,y(k-na),u(k-nk),u(k-nk-1),\ldots,u(k-nk-nb+1)]^T$, and p is a polynomial of degree m in these variables.

For instance, if $na = nb = nk = 1$, then $d = [y(k-1),u(k-1)]^T$, and if we take degree $m = 2$, we can write the polynomial explicitly as:

$$y(k) = ay(k-1) + bu(k-1) + cy(k-1)^2 + vu(k-1)^2 + wu(k-1)y(k-1) + z \tag{2}$$

where a, b, c, v, w, z are real coefficients, and the parameters of the model. Note that the model is nonlinear, since it contains squares and products of the inputs and outputs as opposed to the ARX model which would only contain the terms linear in $y(k-1)$ and $u(k-1)$). Crucially however, the model is still linear in the parameters so linear regression can still be used to identify these parameters.

Note that the linear ARX form is a special case of the general form (1), obtained by taking degree $m = 1$, which leads to:

$$\hat{y}(k) = ay(k-1) + bu(k-1) + c$$

and further imposing that the free term $c = 0$ (without this condition, the model would be called affine).

The **requirements** follow. Code a function that generates such an ARX model, for configurable model orders na, nb, and polynomial degree m; the delay nk can be taken equal to 1. Code also the linear regression procedure to identify the parameters, and the usage of the model on arbitrary input data. Note that the model should be usable in two modes:

- One-step-ahead prediction, which uses knowledge of the real delayed outputs of the system; in the example, we would apply (2) at step k with variables $y(k-1), u(k-1)$ on the right-hand side.
- Simulation, in which knowledge about the real outputs is not available, so we can only use previous outputs of the model itself; in the example we would replace $y(k-1)$ on the right-hand side of (2) by the previously simulated value $\hat{y}(k-1)$.

Identify such a nonlinear ARX model using the identification data, and validate it on the validation data. Tune carefully the model orders and the polynomial degree so as to have maximal performance on the validation data. To reduce the search space you may take $na = nb$. Your presentation should include at least the following elements:

- An introductory part, including a description of the problem.
- A brief description of the approximator structure, and the procedure to find the parameters.
- Any key features of your own individual solution (do not include trivial implementation details).
- Tuning results (at least the MSE as a function of $na = nb$ and of m, either as a table or a graph).
- For the best model obtained above, representative plots with the approximated model output versus the real output, for both simulation and prediction, separately for the training and the validation data sets.
- Corresponding to these plots, report the one-step-ahead prediction error, and the simulation error for both the identification and the validation sets (use the mean squared error).

- A discussion of the results, including the quality of the model fit on the two data sets; and an overall conclusion.

The task description above is self-contained, but you may e.g. look at the following papers for additional technical insight:

1. H. Peng et al., *RBF-ARX model-based nonlinear system modeling and predictive control with application to a NOx decomposition process*, Control Engineering Practice 12, pages 191–203, 2007. Here the model is explained in Sections 2.1-2.2, and uses tunable radial basis functions instead of polynomials.
2. L. Ljung, *System Identification*, Wiley Encyclopedia of Electrical and Electronics Engineering, 2007. Available as technical report LiTH-ISY-R-2809. See Section 4 for nonlinear models, again mainly using basis functions.

Evaluation of part 2

Part 2 will be presented orally, and the slides and code must be submitted in advance of the presentation. The presentations will take place at the end of the semester. The exact program (which group presents when) will be communicated a sufficient time in advance. Each group gets 15 minutes, out of which 8 or 9 are allocated for the presentation itself, and at least 6 minutes for questions. Be there 15 minutes in advance. All three members of the group must attend and answer questions; excepting force majeure, any student who is absent will fail the project and thus be ineligibile for the exam. The grade will consider three aspects: the solution itself (code and results); the presentation; and the question answers. Questions will be targeted to differentiate the contribution of each group member, so each student will likely get a different grade.

The deadline for the presentation itself as well as the associated code is **December 23rd 2022, 24:00**. In case of delays, each newly entered day of delay results in a 2 point decrease in the maximum grade, as for part 1 above.

Please follow the following rules for delivery, as accurately as for part 1.

- Your submission must consist of at least two files, named exactly like this: LN1LN2LN3.pdf, and LN1LN2LN3.zip, where LNi is the last (family) name of student "i" in your project group, without diacritics. For example: IonescuFarkasBonta.pdf and IonescuFarkasBonta.zip.
- The first file is the presentation; only the **PDF** format is acceptable, although optionally you might also include a PPT or PPTX presentation if you wish, and it will be used if possible during the session. Irrespective of the format, please make sure that you send a presentation (slides)! **Do not send a report instead**.
- For identification purposes, the first slide of the presentation should prominently include the names and indices of your group, the latter in the format N/M where N is the part 1 datafile index, and M the part 2 datafile index.
- The second file contains your code itself, sent as a classical **ZIP archive** (no other formats accepted). Important: there should be no subdirectories in this archive, all the m-files must be top-level.
- These files will be submitted via a DropBox file request, the link to which will be supplied before the deadline. Do not submit multiple versions as these cannot be taken into account; only your first submission will be considered, so make sure it is complete and correct.

- The creation date of the file on DropBox is taken for the purpose of delay computation per the rules above.

Matlab programming and other remarks

If you are less familiar with programming in MATLAB, the following pointers may help. Type `doc` at the command line to access the documentation. A good initial read is the *Getting Started with Matlab* node of the documentation. *Matrices and Arrays*, *Programming Basics*, and *Plotting Basics* are also useful.

Strive for a compact and elegant MATLAB code, avoid the use of loops (`for`, `while`, etc.) and `if-then-else` constructs where vector operations would be easier and more readable. Search for "vectorization" in the MATLAB help system for helpful tips on the proper MATLAB programming style. However, do not exaggerate with applying vectorization: if the code is clearer with loops or if statements, use them.