Project 2: Estimating Transfer Functions

SSY 230, System Identification
Jonas Sjöberg
Electrical Engineering, Chalmers

March 31, 2018

In this project you will develop a functions for estimating transfer function. You will use the functions from the first project as building blocks. You will also write functions for analysing the model and for using it for prediction and simulation.

1 Write an ARX estimator

(a) Write a command `arxfit(z,[na nb nk])` which takes a data data set `z`, and three integer parameters as inputs and which delivers an ARX model. `na` indicates the number of past outputs in the regressor, `nb` number of past inputs, and `nk` delays in the input signal. To validate the function, generate some noise-free data. and verify that you re-obtain the correct parameter values. Store the result in an object like the ones you produced in *project 1*. Add fields to the object, one indicating that it is an ARX model and one with the indexes. These fields will be useful to recognise what type of model it is.

(b) Write a function `id2tf(arx)` which convert an ARX model to Matlab's transfer function (see `tf`). Using that function, you can evaluate your estimated models using `ltiview`.

(c) Write functions `idpredict(model,z,horizon)` and `idsimulate(model,z)`. For `idsimulate` yo can have good help of `lsim`.

(d) (optional) Write a function `idcompare(z,model,horizon)` which plots the simulated/predicted output together with the true output. Plot also the uncertainty. This is an excellent function for validation where you can test the estimated model on validation data and evaluate the performance. If you don't implement this function, consider at least how you could have evaluated the uncertainty and write this in the report.

2 Write an OE estimator

You will now write a function `oefit(z,[nb nf nk])` which returns an OE-model.

(a) Write a function `oefit(z,[nb nf nk])` which performs the following steps

- Estimate a high order ARX model of order `na=4*nf`, `nb=4*nb` and `nk` unchanged. This gives you an ARX model and let us call the numerator B_h and the denominator A_h.
- Simulate the ARX model giving the output y_s.
- Here you have two options:
  (1) (approximate method) Estimate a new ARX model using the original input, the simulated output, y_s, and the original model order. The result is your OE model.
  (2) (optimal method) Filter both the original input and simulated output, y_s, through the FIR filer A_h. Estimate a new ARX model using the filtered signals and the original model order. The result is your OE model.

  Change the appropriate field so that it indicates that it is an OE model.
- Explain why yo obtain an OE model in this way.

Verify your estimator. Add white noise to the output signal and verify that you re-obtain the true parameter values when the number of data is high. Verify also that this is not the case if you identify with arxfit.

(b) Modify `id2tf` so it works for OE-models too.

(c) Modify `idpredict` so it works for OE-models too.

(d) Modify `idsimulate` so it works for OE-models too.

(e) (Optional) Modify `idcompare` so it works for OE-models too.

3 Identify two systems

Now it is time to use your functions on some data sets from two different systems.

For the first system you find the data in the file `exercise1` and the input is in the variable u and the output in y.

The second system data you find in `exercise2` and the data is placed in $z1$, estimation data and $z2$, validation data. The input is in the second column and the input is in the first column.

For both systems, estimate good models taking the following into account.

- Try different orders of ARX and OE models and evaluate the model quality.
- Investigate model properties by looking at, eg, poles and zeros, and impulse response (using `ltiview`).
- Try prediction with different horizon and simulation on validation data. Is the same model best in all aspects?
- Illustrate prediction performance using your command `idcompare`.