University of Alberta Computer Process Control Group

Subspace Closed-loop Identification

Limited Trial Version

Written by: CPC Control Group, University of Alberta

Version 2.0

Table of Contents

List of Figures

Introduction

The “Subspace Closed-loop Identification” toolbox was developed by the Computer Process Control Group at the University of Alberta to allow closed-loop identification to be performed in MATLAB using the subspace approach.

A “Quick Start” approach to using this algorithm is presented, along with a detailed section containing full explanations and examples for using this algorithm.

System Requirements

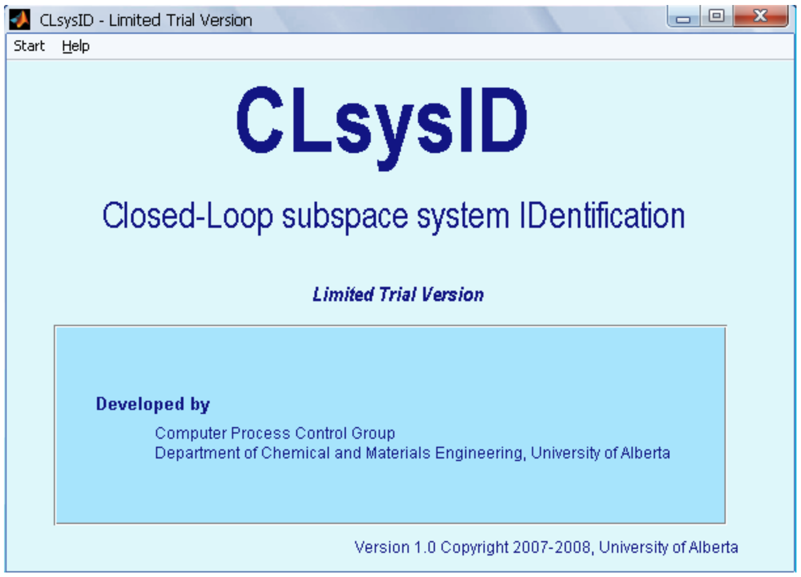
In order to run this program properly, the following programmes are required:

1) MATLAB 2006a (MATLAB 7.1) or better. It should be noted that the newest version of MATLAB (MATLAB 2008a) makes the toolbox run slower.
2) The SYSTEM IDENTIFICATION TOOLBOX from MATLAB is required.

Quick Start

For quickly using the toolbox, the following steps should be followed:

1) Unzip the files to the desired location.
2) Start MATLAB, and point the current directory to the location of the unzipped files.
3) At the command prompt, type >>**main_CLsysID** to start the toolbox. The GUI shown in Figure 1 should appear.



CLsysID

Closed-Loop subspace system IDentification

Limited Trial Version

Developed by
Computer Process Control Group
Department of Chemical and Materials Engineering, University of Alberta

Version 1.0 Copyright 2007-2008, University of Alberta

Figure 1: The first GUI that appears

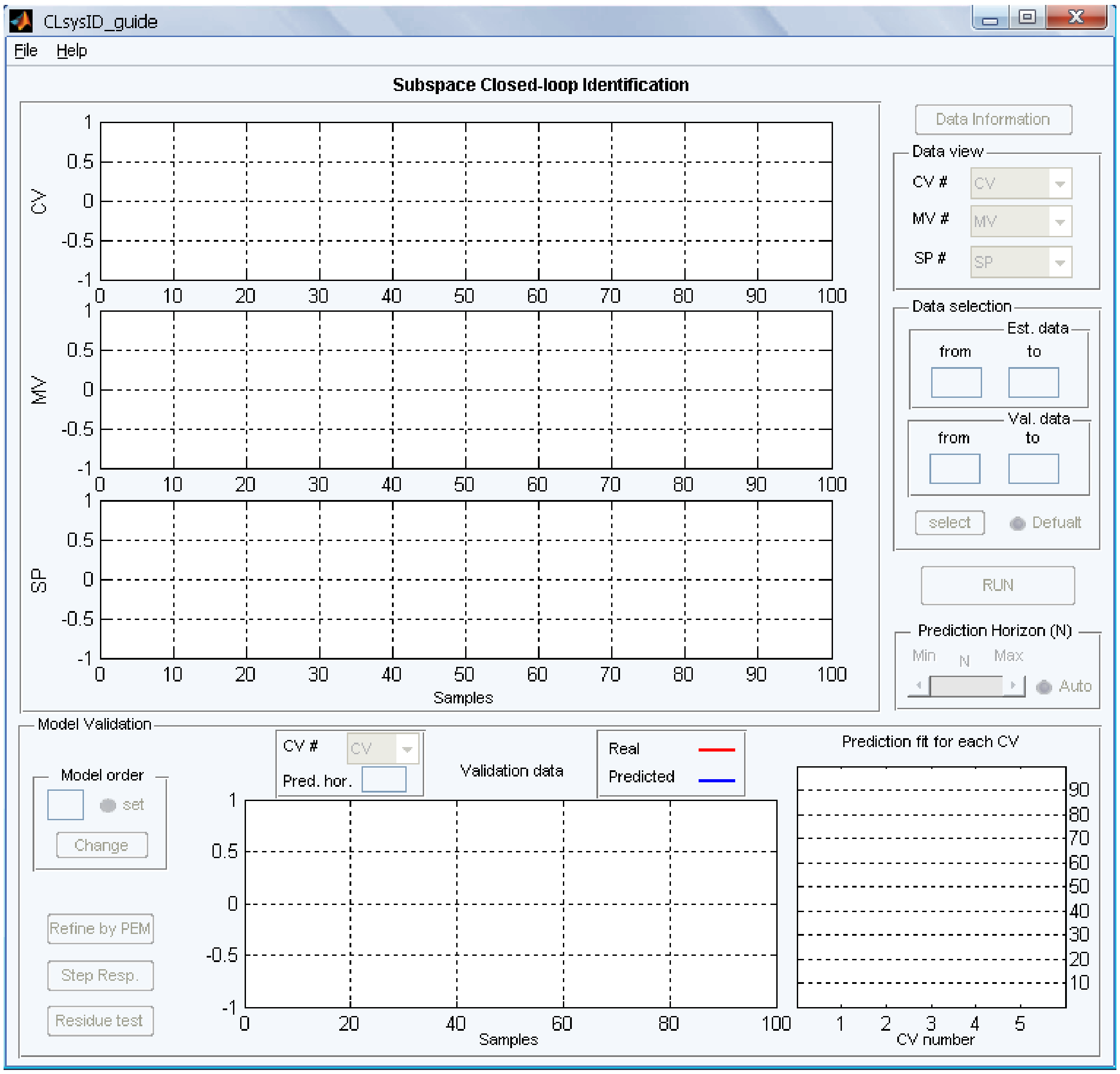


Figure 2: The main GUI

4) Press the **Run** button in the **Start menu**. A new GUI will appear that is shown in Figure 2.
5) This GUI tool provides closed-loop subspace identification using the joint-output identification method.
6) You have to load a set of closed-loop experiment data in *Compact* format (to be explained shortly) using the **Data** submenu in the **File** menu.

7) For the purpose of this quick start, we will use the sample data provided from a MIMO example.
8) When the data is loaded, the corresponding information about it can be obtained by pressing the **Data Information** button. After loading the data, the GUI should resemble Figure 3.

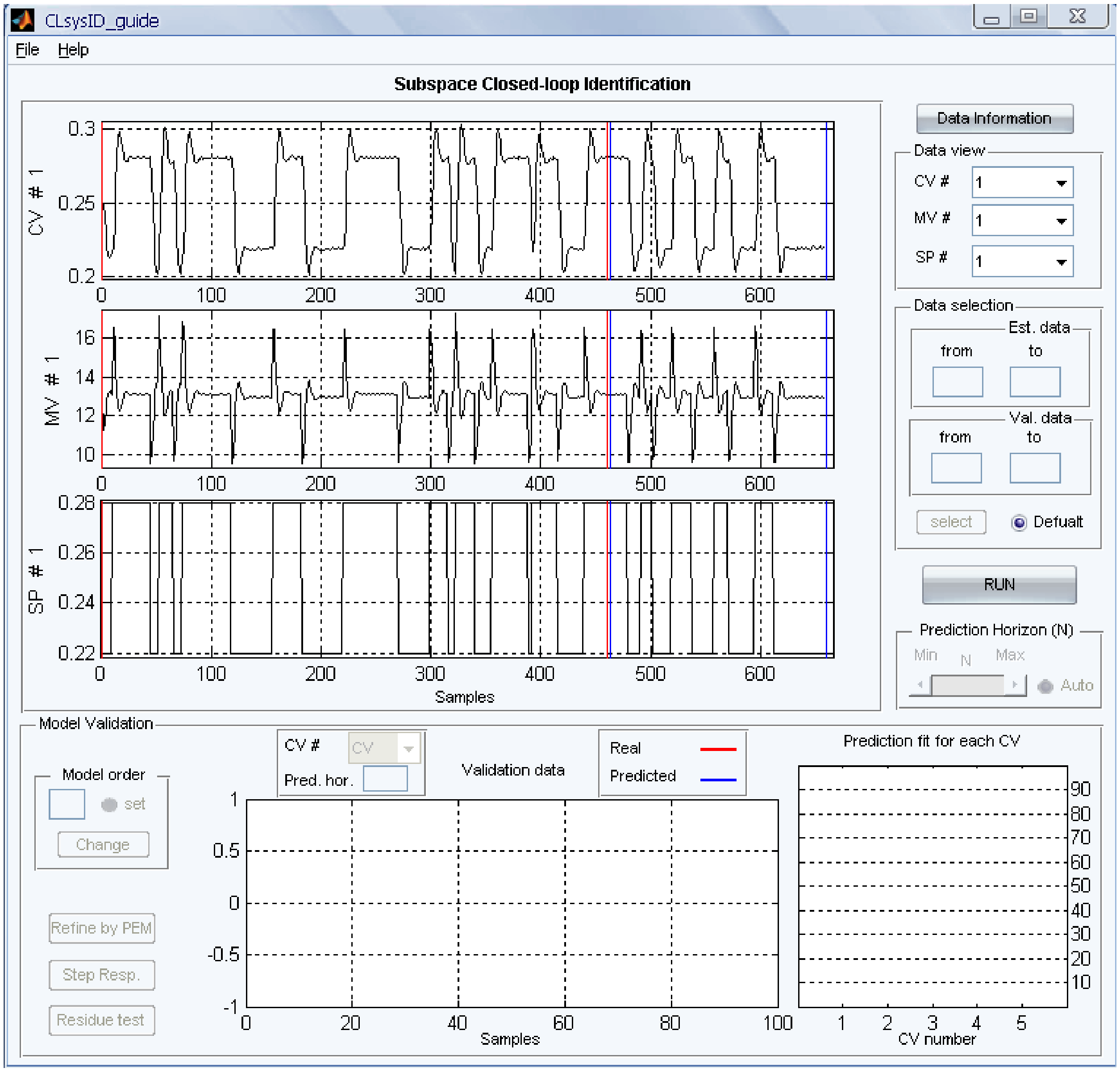
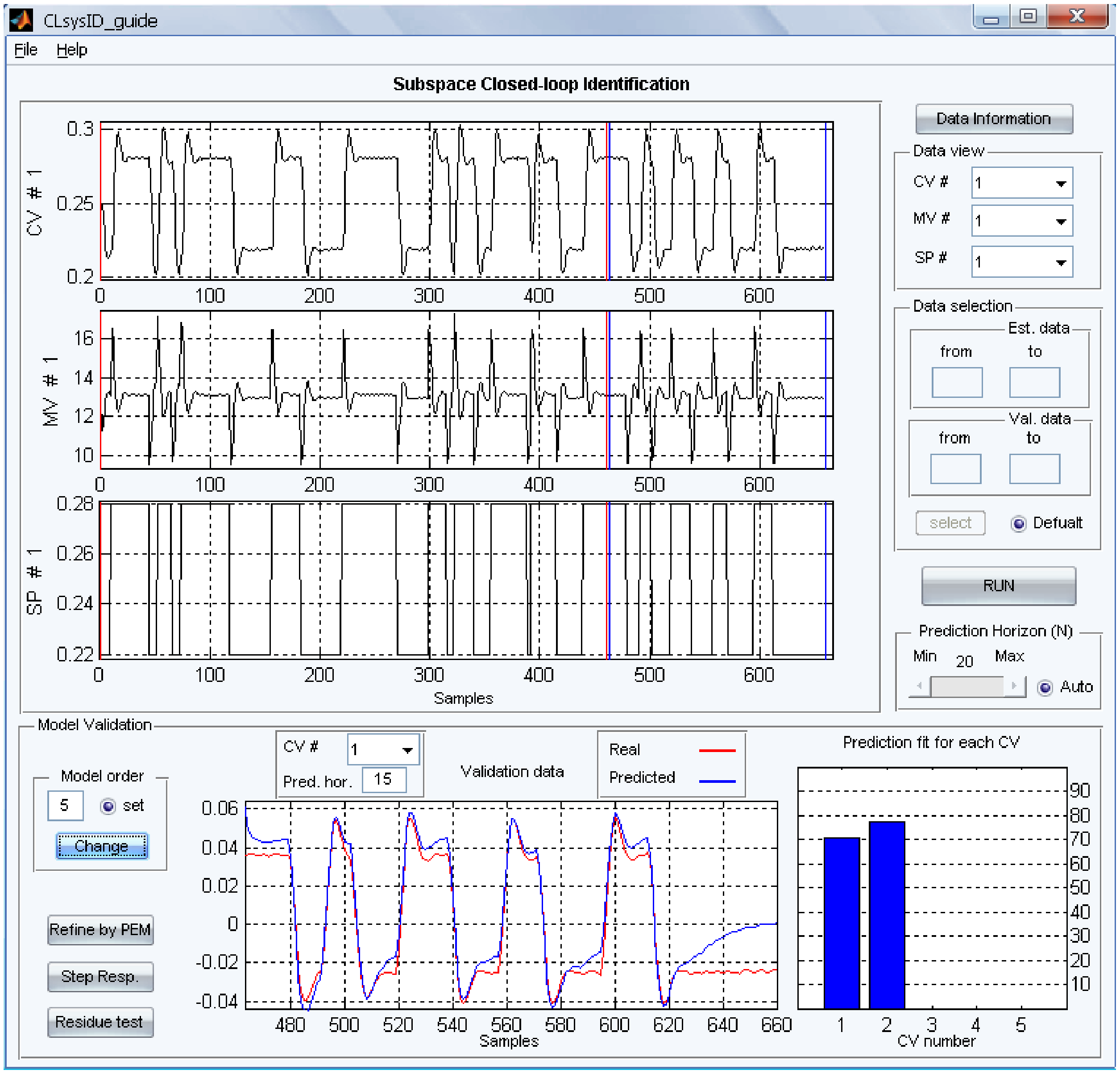


Figure 3: Main GUI with the closed-loop data loaded

9) Use the **Run** button to estimate the process model from closed-loop data. Depending on the type of process dynamics, the number of inputs and outputs, and the number of data

points, it may take between a few seconds to couple of minutes to obtain a model. The results of pressing this button are shown in Figure 4.



CV # 1

MV # 1

SP # 1

Min 20 Max

Model order 5

CV # 1

Pred. hor. 15

Figure 4: The estimated state-space model and fit analysis

10) The results of model validation are provided in the **Validation** panel including the prediction fit and residual test results. Further information about the residuals can be found by pressing the **Residue test** button. The results are shown in Figure 5.

11) The estimated model can be refined using the prediction error method. You can use the **Refine by PEM** button in the **Validation** panel.

12) Direct step response estimation and continuous-time transfer function models between each input and output can also be obtained using the **Step Resp.** button in the **Validation** panel. The results are shown in Figure 6.

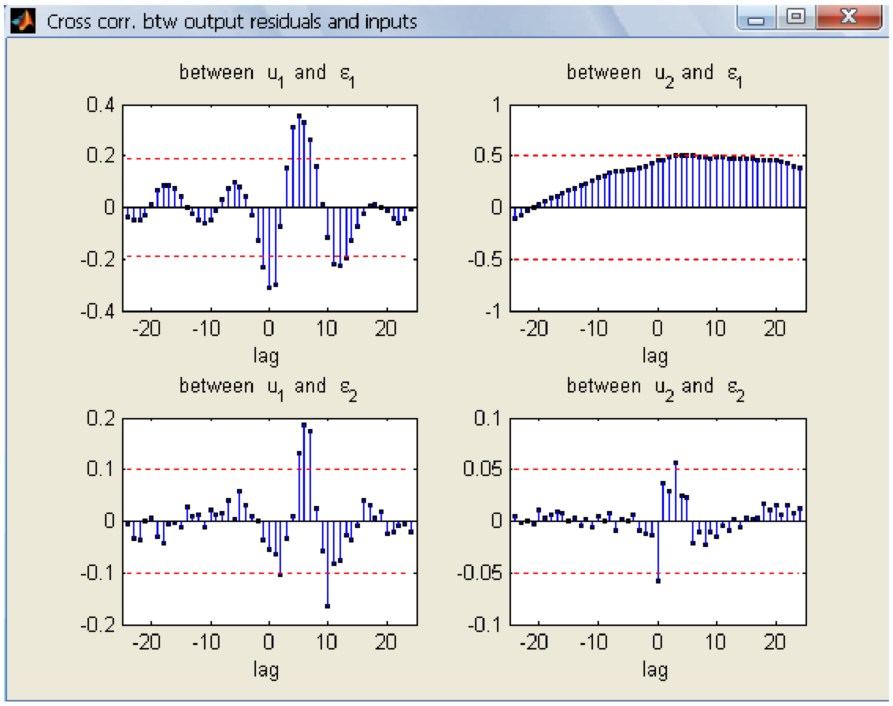


Figure 5: Residue test results

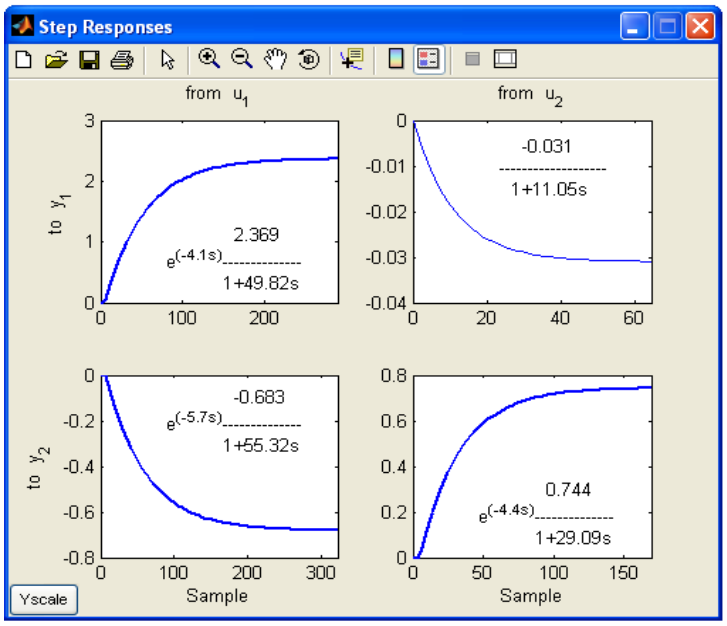


Figure 6: Step response model results

Detailed Instructions

Theory

Subspace identification methods provide an alternative approach to the classical system identification methods like the prediction error method (Ljung L. , 1999) and the instrument variable method (Söderström & Stoica, 1989). Subspace methods use efficient computational algorithms such as QR-factorization and singular value decomposition to compute the parameter values. This makes these methods numerically robust. Different subspace identification methods have been developed in the last two decades, including the regression analysis approach, N4SID (**n**umerical **s**ub**s**pace **s**tate **s**pace **id**entification), MOESP (**M**IMO **o**utput **e**rror **s**tate s**p**ace), CVA (**c**anonical **v**ariate **a**nalysis) (Knudsen, 2001; Larimore, 1996; Moonena, De Moora, Vandenberghea, & Vandewalle, 1989; Van Overschee & De Moor, 1994; Van Overschee & De Moor, 1995; Verhaegena & Dewilde, 1992; De Moor & Van Overschee, 1996). As well, work has been devoted to implementing closed-loop, subspace system identification for control-relevant identification and controller performance assessment. Many subspace methods have been extended to closed-loop identification (Knudsen, 2001; Ljung & McKelvey, 1996; Tangirala, Lakshminarayanan, & Shah, 1997; Van Overschee & De Moor, 1997; Kadali & Huang, 2002). An improved closed-loop subspace identification method was also presented in (Danesh Pour, Huang, & Shah, 2008) using the joint input-output identification approach. An alternative formulation of this method with guaranteed consistency is proposed in(Huang & Kadali, 2008). This programme is based on the last algorithm.

Subspace Identification

Consider the following state space representation for a linear system with *l* inputs and *m* outputs:

$$\begin{cases} x_{t+1} = Ax_t + Bu_t + Ke_t \\ y_t = Cx_t + Du_t + e_t \end{cases} \tag{1}$$

where $x_t \in R^n$ is the state, $u_t \in R^l$ is the input, $y_t \in R^m$ is the output, and $e_t \in R^m$ is the white noise disturbance. System (1) can be rewritten in terms of the basic subspace equations for output and states, that is,

$$\begin{aligned} Y_f &= \Gamma_N X_f + L_u U_f + L_e E_f \\ X_f &= A^N X_p + \Delta_N^d U_p + \Delta_N^s E_p \end{aligned} \tag{2}$$

where *p* represents past values of the system, *f* represents future values of the system, $X_f \in R^{n\times j}$ is the subspace matrix of future states given by

$$X_f = \left(x_N, x_{N+1}, \ldots x_{N+j-1}\right) \tag{3}$$

$\Gamma_N \in R^{mN\times n}$ is the extended observability matrix defined as

$$\Gamma_N = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{N-1} \end{pmatrix} \tag{4}$$

$L_u \in R^{mN\times lN}$ is the extended process dynamics matrix defined as

$$L_u = \begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ CA^{N-2}B & CA^{N-3}B & \cdots & D \end{bmatrix} \tag{5}$$

$L_e \in R^{mN\times mN}$ is the extended error dynamics matrix defined as

$$L_e = \begin{bmatrix} I & 0 & \cdots & 0 \\ CK & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ CA^{N-2}K & CA^{N-3}K & \cdots & I \end{bmatrix} \tag{6}$$

$\Delta^d_N \in R^{n\times lN}$ is the transposed extended controllability matrix of (*A*, *B*)

$$\Delta_N^d = \left(A^{N-1}B, A^{N-2}B, \ldots, AB, B\right) \tag{7}$$

$\Delta^s_N \in R^{n\times mN}$ is the transposed extended controllability matrix of (*A*, *K*)

$$\Delta_N^s = \left(A^{N-1}K, A^{N-2}K, \ldots, AK, K\right) \tag{8}$$

$U_p \in R^{nlN\times j}$ is the data Hankel matrix for past inputs defined as

$$U_p = U_{0|N-1} = \begin{bmatrix} u_0 & u_1 & \cdots & u_{j-1} \\ u_1 & u_2 & \cdots & u_j \\ \vdots & \vdots & & \vdots \\ u_{N-1} & u_N & \cdots & u_{N+j-2} \end{bmatrix} \tag{9}$$

$U_f \in R^{lN\times j}$ is the data Hankel matrix for future inputs defined as

$$U_f = U_{N|2N-1} = \begin{bmatrix} u_N & u_{N+1} & \cdots & u_{N+j-1} \\ u_{N+1} & u_{N+2} & \cdots & u_{N+j} \\ \vdots & \vdots & & \vdots \\ u_{2N-1} & u_{2N} & \cdots & u_{2N+j-2} \end{bmatrix} \tag{10}$$

$Y_p \in R^{mN\times j}$ is the data Hankel matrix for past outputs defined similarly to Equation (9), $Y_f \in R^{mN\times j}$ is the data Hankel matrix for future outputs defined similarly to Equation (10), $E_p \in R^{mN\times j}$ is the data Hankel matrix for past errors defined similarly to Equation (9), and $E_f \in R^{mN\times j}$ is the data Hankel matrix for future errors defined similarly to Equation (10).

In open-loop subspace identification in order to reduce sensitivity to noise, *j* should be much larger than

$$\max\left(mN, lN\right) \tag{11}$$

In many ways, *j* plays the same role as the number of observations in classical identification. The number of rows in the data Hankel matrices, *N*, is related to the order of the system.

Using a regression analysis approach (Knudsen, 2001), it can be shown that

$$Y_f = L_w W_p + L_u U_f + L_e E_f \tag{12}$$

where $L_w \in R^{mN\times(l+m)N}$ and W_p is defined as

$$W_p = \begin{bmatrix} Y_p \\ U_p \end{bmatrix} \tag{13}$$

This gives an input-output equation in the subspace framework.

Subspace identification of a closed-loop system

For the closed-loop system shown in Figure 7, two separate open-loop models can be defined: a model from the setpoint r_t to the output y_t and another model from r_t to the controller output u_t. Similarly to Equation (12), these two systems can be represented by the following input-output relationships (Huang & Kadali, 2008):

$$Y_f = L_Y \begin{bmatrix} Y_p \\ R_p \end{bmatrix} + L_{YR} R_f + L_{YE} E_f \tag{14}$$

$$U_f = L_U \begin{bmatrix} U_p \\ R_p \end{bmatrix} + L_{UR} R_f + L_{UE} E_f \tag{15}$$

where

$$W_p^Y = \begin{bmatrix} Y_p \\ R_p \end{bmatrix}$$
$$W_p^U = \begin{bmatrix} U_p \\ R_p \end{bmatrix} \qquad (16)$$

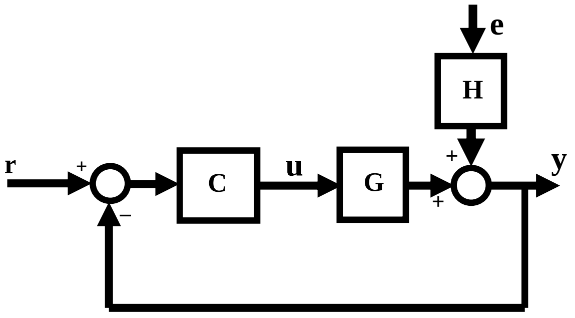


Figure 7: Closed-loop system

If it is assumed that R_f is uncorrelated with E_f, and $W^U{}_p$ with $W^Y{}_p$, each of the open-loop problems can be solved using the least square method. In (Huang & Kadali, 2008), it has been shown that the open-loop subspace matrices L_u, L_e, and Γ_N, and the input noise covariance matrix can be estimated from the closed-loop subspace matrices, L_{YR}, L_{UR}, and L_{UE}, using a joint input-output framework. The state-space matrices can be obtained as follows:

$$\begin{aligned}
\Gamma_N^u &= \Gamma_N\left(1:(N-1)m,:\right) \\
\Gamma_N^y &= \Gamma_N\left(m+1:Nm,:\right) \\
\hat{C} &= \Gamma_N\left(1:m,1:n\right) \\
\hat{A} &= \left(\Gamma_N^u\right)^+ \Gamma_N^y \\
\hat{B} &= \left(\Gamma_N^u\right)^+ L_u\left(m+1:Nm,1:l\right) \\
\hat{D} &= L_u\left(1:m,1:l\right) \\
\hat{K} &= \left(\Gamma_N^u\right)^+ \hat{P}_v\left(m+1:Nm,1:m\right)\hat{R}^{-1}
\end{aligned} \qquad (17)$$

where + represents the Penrose pseudo-inverse, $\hat{\circ}$ represents an estimated value, P_v is the noise covariance matrix, and R is the parameter covariance matrix.

The L_u and L_e matrices as defined by Equations (5) and (6) contain the impulse response coefficients (Markov parameters) of the process. Therefore, the step response coefficients between each input and output can be directly extracted from these subspace matrices without requiring an explicit process model. In fact, this approach provides more reliable estimation of the step response, especially when the final model is not accurate. In this toolbox, nonlinear regression is used to fit a continuous-time transfer function model to each set of step response coefficients. Both the **f**irst-**o**rder **p**lus **t**ime **d**elay (FOPTD) and **s**econd-**o**rder **p**lus **t**ime **d**elay (SOPTD) models are fitted to each set and the model with the better fit is selected.

Data Storage

Data Storage Format

The data storage format is called a **compact** format. Assume that there are *j* samples of output, input, and setpoint data with a total of *m* control variables and *l* manipulated variables with a total of *q* tags. In the compact data storage method, the data is stored as an object containing the following entries:

1) **controller_Status**: This is a *j*-by-1 double matrix that contains the status of each of the controllers, where 1 represents a controller that is "on" and 0 represents a controller that is "off."
2) **cell_char_TagList**: This is a *q*-by-1 cell matrix that contains the name of each of the tags that are presented in the process.
3) **cell_char_TimeStamp**: This is a *j*-by-1 cell matrix that contains the time stamp for each of the samples.
4) **dbl_Compact_Data**: This is a *j*-by-(2*m*+*l*) double matrix that contains the values for each of the outputs, inputs and setpoints at sample periods.
5) **dbl_SamplingTime**: This is a scalar double that contains the sampling time for the data.
6) **int_CVNumber**: This is a scalar integer that contains the number of controlled variables in the process, that is, *m*.
7) **int_MVNumber**: This is a scalar integer that contains the number of manipulated variables in the process, that is, *l*.
8) **status**: This is a *j*-by-(*l* + *m*) double matrix that stores the data in the following manner: The first *t* columns contain the status of the controller variables, while the remaining *s* columns contain the status of the manipulated variables. A value of 1 signifies that the data is good.

Data Generation

The data storage file can be generated using **gen_cmpct_data.p**. A set of closed-loop experiment data (identification data) should be present in the MATLAB workspace. If your data is ready in the workspace, run the code. The followings will appear in the command window:

```
*************************************************************
Consider a process with m inputs and p outputs:
The following information should be available in WORKSPACE:

 1. CL Excitation data; Output, Input and Setpoint data (y,u,r)
 2. Sampling time
*************************************************************
Press ENTER to continue or type X to exit :
```

Press enter to continue if the data has already been created; otherwise quit the programme, create the data, and try again. Once enter is pressed, the following will appear:

```
Provide Closed-loop excitation data matrix [y u r]:
```

The data should be arranged in the given order, that is, output, input, and setpoint. Next, the number of process outputs must be entered:

```
number of outputs:
```

Then, the sampling time will be entered:

```
Sampling time:
```

Finally, the name of the file to which the data should be saved is entered at the prompt:

```
The file name to save compact data (e.g. test_cmpct_data):
```

This programme will create a ".mat" file that is saved to the current path in MATLAB.

Using the Toolbox

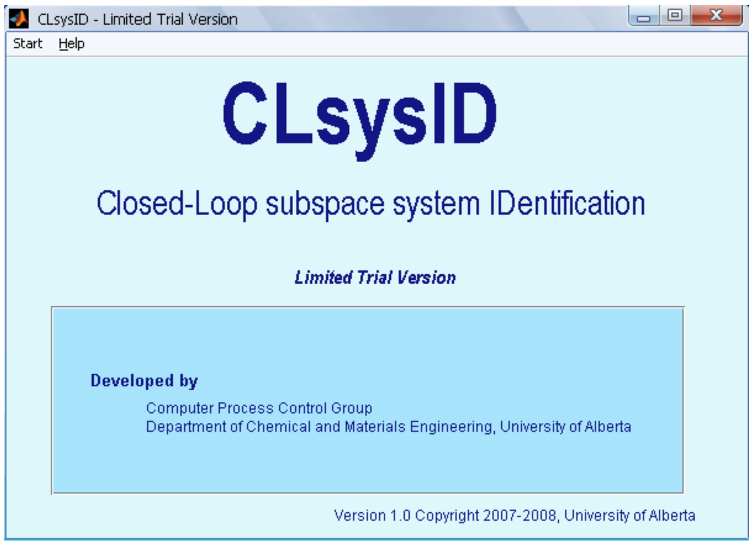
Installation

The toolbox can be installed by simply unzipping the files to any desired location. In order for the toolbox to function properly, the System Identification Toolbox should be installed.

Starting the Toolbox

The toolbox can be accessed from MATLAB using the following sequence of commands.

First MATLAB itself should be started from the directory pointing to the folder containing the files for this toolbox. Next, at the command prompt, type >> main_CLsysID. The GUI shown in should appear. This GUI is the main access to the toolbox. To start a session of the toolbox, click on the **Run** submenu in the **Start menu**. This will bring up a new GUI, which is shown in Figure 9. Each of the main parts of the GUI in Figure 9 will be discussed separately later.



CLsysID

Closed-Loop subspace system IDentification

Limited Trial Version

Developed by

Computer Process Control Group
Department of Chemical and Materials Engineering, University of Alberta

Version 1.0 Copyright 2007-2008, University of Alberta

Figure 8: The first GUI that appears

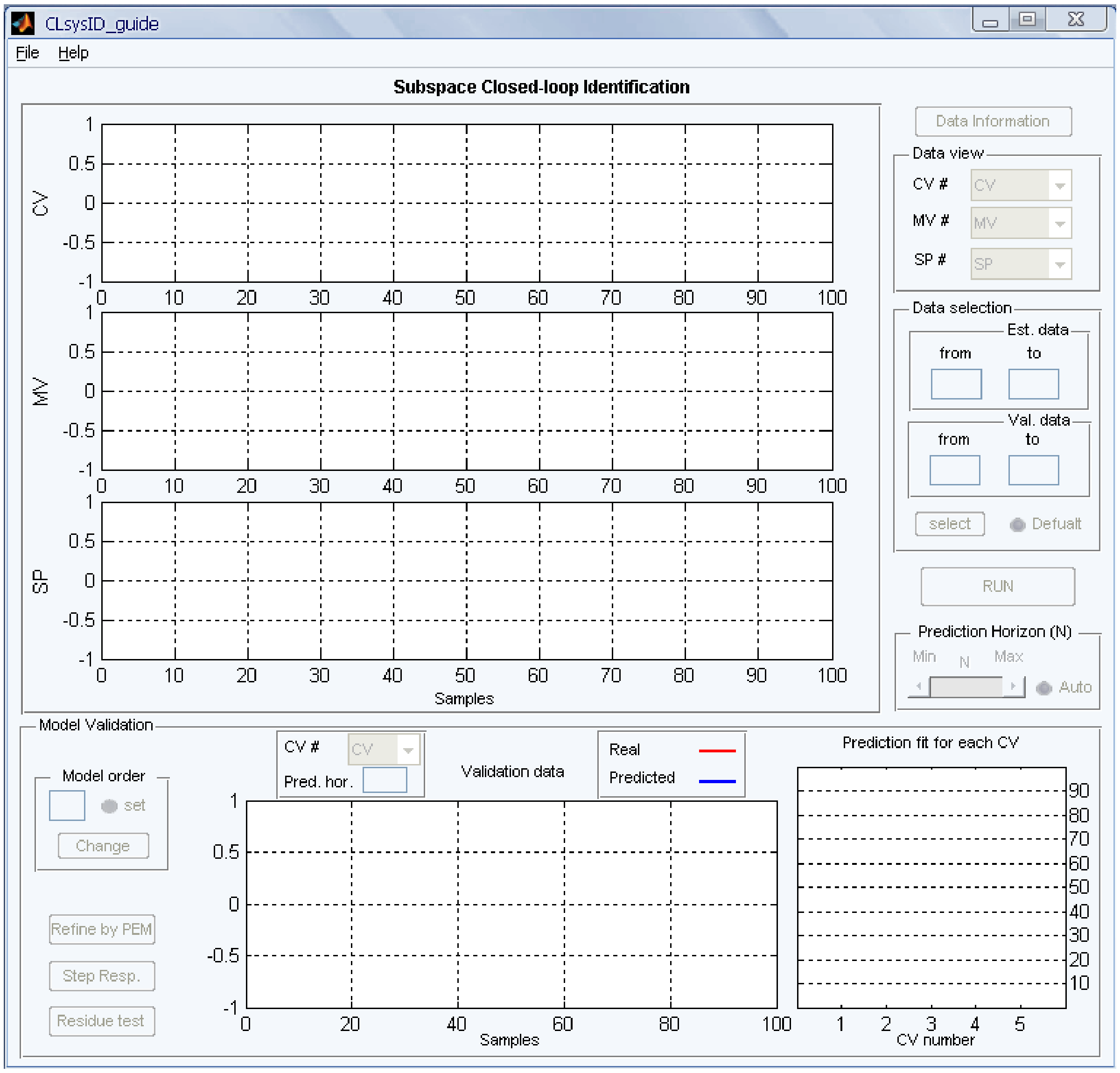


Figure 9: The main GUI for this toolbox

In-depth Discussion of the Toolbox

Section 1: Main Menu

The Main Menu consists of the following areas:

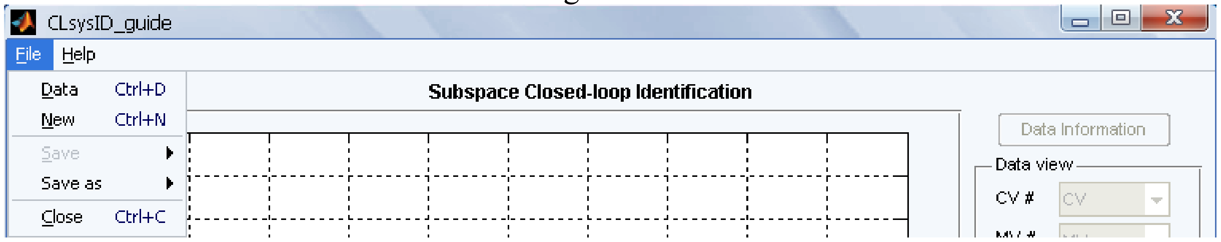


Figure 10: The “File” menu

1) **Data:** Clicking this menu will allow the user to load the closed-loop experiment data for analysis.
2) **New:** Clicking this menu will clear all the data from the current GUI and allow the user to restart the analysis from a clean layout.
3) **Save:** Clicking this menu will allow the user to save the resulted state space model or step response coefficients on the disk.
4) **Save as:** Clicking this menu will allow the user to re-save the resulted state space model or step response coefficients.

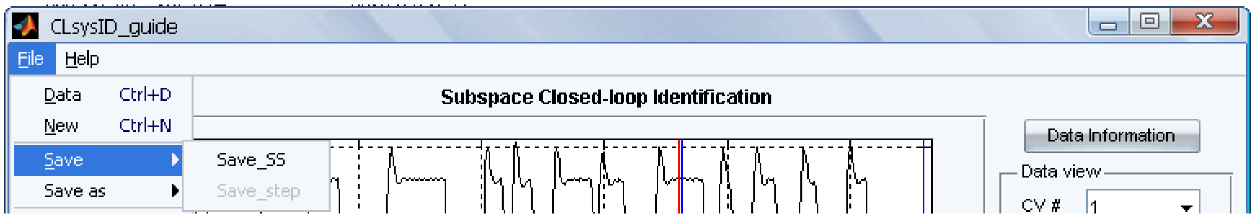
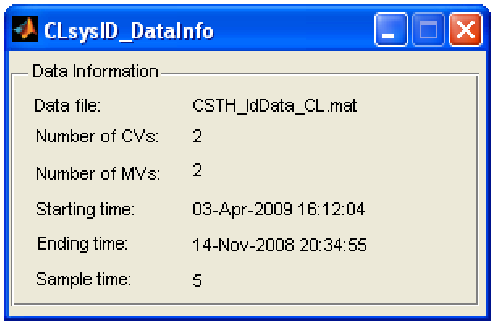


Figure 11: The “Save” menu

Section 2: Data Information Button

This button opens a new window which shows information about the loaded experiment data including the data file name, the number of controller variables (CV), the number of manipulated variables (MV), the sampling time, the starting time, and end time. A sample example of the data provided is shown in Figure 9.



Data file:	CSTH_IdData_CL.mat
Number of CVs:	2
Number of MVs:	2
Starting time:	03-Apr-2009 16:12:04
Ending time:	14-Nov-2008 20:34:55
Sample time:	5

Figure 9: Data Information Window

Section 3: Data View and Selection Panels

The data view and selection panels are shown in Figure 10.

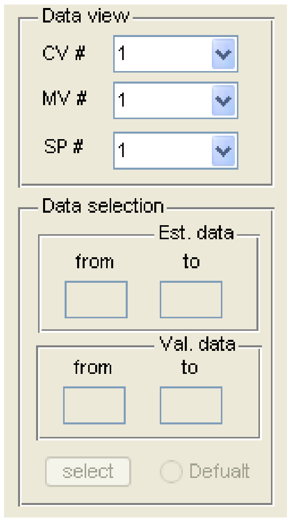


Figure 10: Data View and Selection Panel

The following are the most important sections:

1) **Data view:** This pane enables you to choose the controlled variable (CV), manipulated variable (MV), or setpoint variable (SP) which you want to be shown on the corresponding axes.
2) **Data selection:** In this filed, you can specify which data sections are to be used for identification (est. data) and validation (val. data). The end value for each selection must be greater than its starting value. In order to register the values, click the **select** button.

Section 4: Run Panel

A close-up view of the run panel is shown in Figure 11.

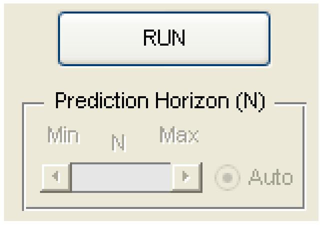


Figure 11: Run Panel

1) **Run:** This button is used to run the main closed-loop identification algorithm. The button is disabled until the data is loaded.
2) **Prediction Horizon (N) Panel:** The slider can be used to set the appropriate values for the prediction horizon, *N*. The range of values selected is determined by the algorithm on

first run. From then on, the user is able to select the appropriate value. In order to enable, manual selection, the radio button beside auto should be ticked **off**; otherwise an automatically selected value will be used.

Section 5: Plot Data

Three axes, as shown in Figure 12, are used to display the controlled variables, manipulated variables, and setpoint data. Since only a single variable of each type can be displayed at a time, the selections are made in the data selection panel (see Section 3: Data View and Selection Panels).

Section 6: Validation Panel

In this panel, model validation and residual test results are presented. A close-up of this section is shown in Figure 13. As well, in this panel the model order can be changed and the step response functions determined. The main parts of this panel are:

1) **Model Order Subpanel:** The estimated order for the state space model is shown in this subpanel. If the **Set** option is enabled, then the model order can be changed manually. When the new order is entered, press the **Change** button to rerun the model estimation algorithm to display the new results.

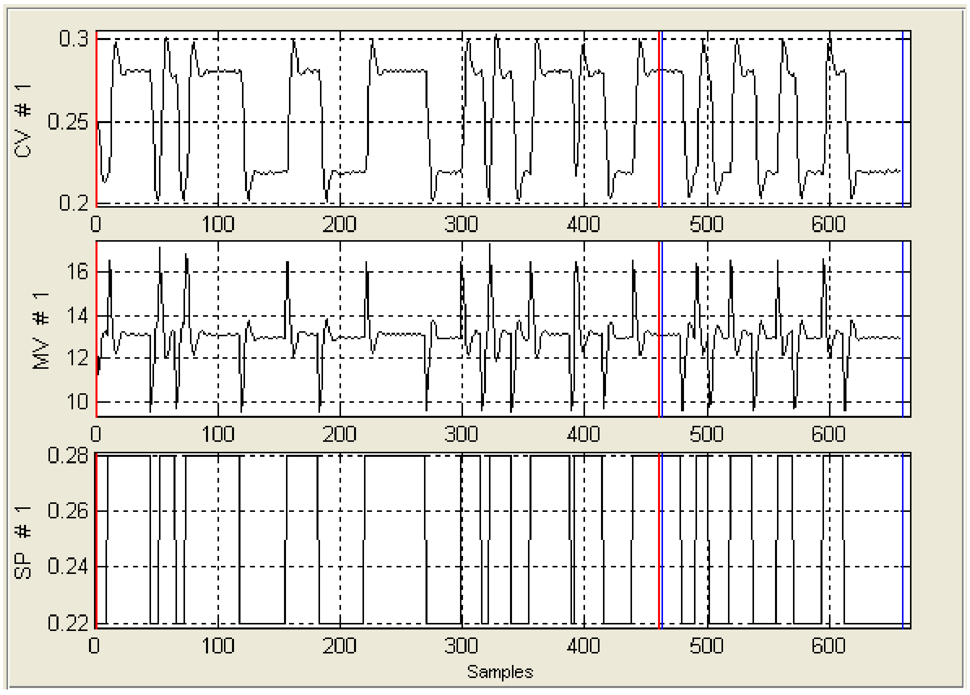


Figure 12: Data Plot Axes

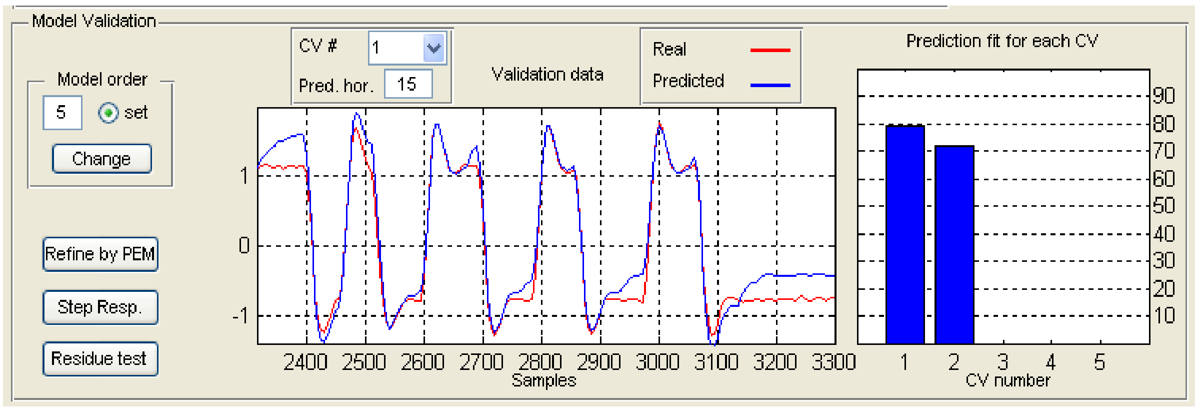


Figure 13: Validation Panel

2) **Validation Data Plot:** This plot shows the validation data as well as the model prediction. Only one pair of data points can be shown at a single time, so the user has to choose which output is to be displayed using the **CV** drop-down box located at the top of the panel. The prediction horizon can be changed using the **Pred. hor.** textbox. The default value for the prediction horizon is 15 samples.

3) **Prediction Fit for Each CV Plot:** This plot shows the results of comparing the validation output to the model prediction for each process output. Each bar represents the percent fit for one controller variable.

The three push buttons, located on the left of this panel, are called as follows:

1) **Refine by PEM:** This button will run the **pem** function from MATLAB's system identification toolbox to refine the identified model obtained subspace identification. It should be noted that this need not produce a better model. If the new model is not better, then the previously estimated subspace model can be restored by pressing the **Change** button.

2) **Residue Test:** This button can be used to view the results of residue tests on the last estimated model. Only the cross-correlation test is performed, so the disturbance model is not tested. The results are shown in Figure 14.

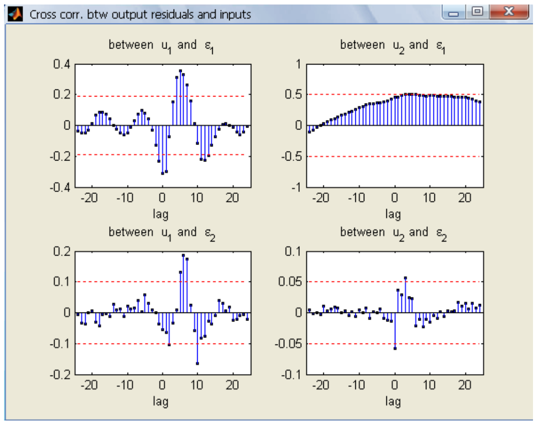

Cross corr. btw output residuals and inputs
between u_1 and ε_1
between u_2 and ε_1
between u_1 and ε_2
between u_2 and ε_2
lag

Figure 14: Residual Test Window

3) **Step Resp.:** This button will open a new window that allows the user to estimate the step response coefficients directly from the subspace matrices. This is shown in Figure 15. The **Export** menu is used to save the step response coefficients. The **"Cont. Model"** menu is used to obtain the continuous-time models (FOPTD or SOPTD) for the estimated step response coefficients. A window as shown in Figure 19 will be opened to enter the model orders for each step response. The nonlinear regression algorithm from MATLAB's statistics toolbox is used to fit a continuous-time model to each step response. The results will be shown in a new window, similar to that shown in Figure 17. **Note that the results of this calculation may not be reliable.**

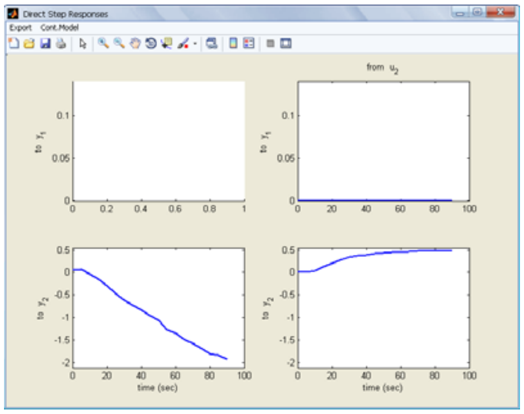
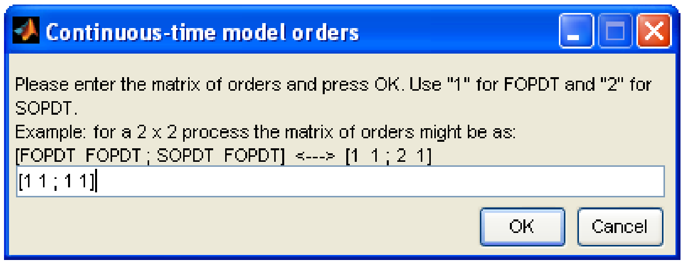


Figure 15: Estimated Step Response Coefficients



Please enter the matrix of orders and press OK. Use "1" for FOPDT and "2" for SOPDT.
Example: for a 2 x 2 process the matrix of orders might be as:
[FOPDT FOPDT ; SOPDT FOPDT] <---> [1 1 ; 2 1]

```
[1 1 ; 1 1]
```

Figure 16: Window for Entering the Continuous-Time Model Orders

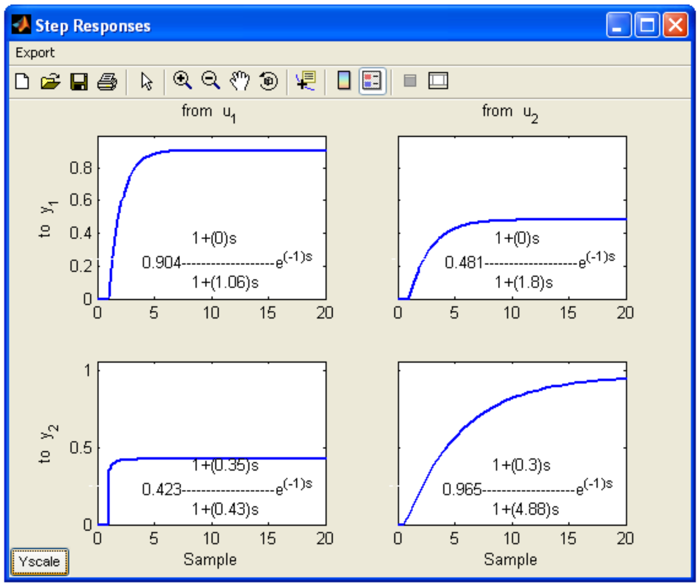


Figure 17: Estimated Continuous-Time Models

Example 1

A set of closed-loop identification data generated from the following system is provided with the toolbox in a file called **SISO_IdData_CL.mat**:

$$x_{t+1} = \begin{pmatrix} 0.6 & 0.6 & 0 \\ -0.6 & 0.6 & 0 \\ 0 & 0 & 0.7 \end{pmatrix} x_t + \begin{pmatrix} 1.6161 \\ -0.3481 \\ 2.6319 \end{pmatrix} u_t + \begin{pmatrix} -1.1472 \\ -1.5204 \\ -3.1993 \end{pmatrix} e_t \tag{18}$$
$$y_t = \begin{pmatrix} -0.4373 & -0.5046 & 0.0936 \end{pmatrix} x_t - 0.7759 u_t + e_t$$

A continuous-time **p**roportional-**i**ntegral (PI) controller,

$$u_t = \left(0.1 + \frac{0.05}{s}\right) y_t \tag{19}$$

is used to control the process. The Simulink diagram shown in Figure 18 can be used to collect the data. The identification test signal, *r*(*t*), is created using MATLAB's idinput command

```
r=idinput(1000,'rbs',[0,0.06], [-1,1]);
```

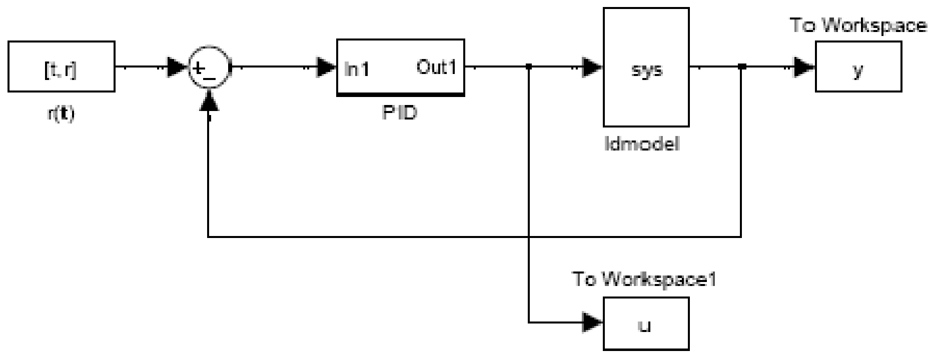


Figure 18: Simulink Model for Collecting the Data

The idmodel block needs a variable called sys to be defined in the workspace. This can be done as follows:

```
A=[0.6 0.6 0;-0.6 0.6 0;0 0 0.7];
B=[1.6161; -0.3481; 2.6319];
C=[-0.4373 -0.5046 0.0936];
Du=-0.7759;
K=[-1.1472; -1.5204; -3.1993];
sys=idss(A,B,C,D,K,[0;0;0],1);
sys.NoiseVariance=0.01;
```

In the block parameters of idmodel, there is an option called “Add noise” which must be selected. Once the required variables have been defined in the workspace and the Simulink file has been run to obtain the simulated data, run gen_cmpct_data.p to create and save the data file using the *Compact* format. The data file is called SISO_IdData_CL. Loading this data file into GUI, the state space model and continuous-time transfer function model of this process can be estimated. Click the **run** button to obtain the model estimates. The GUI should look similar to that shown in Figure 19.

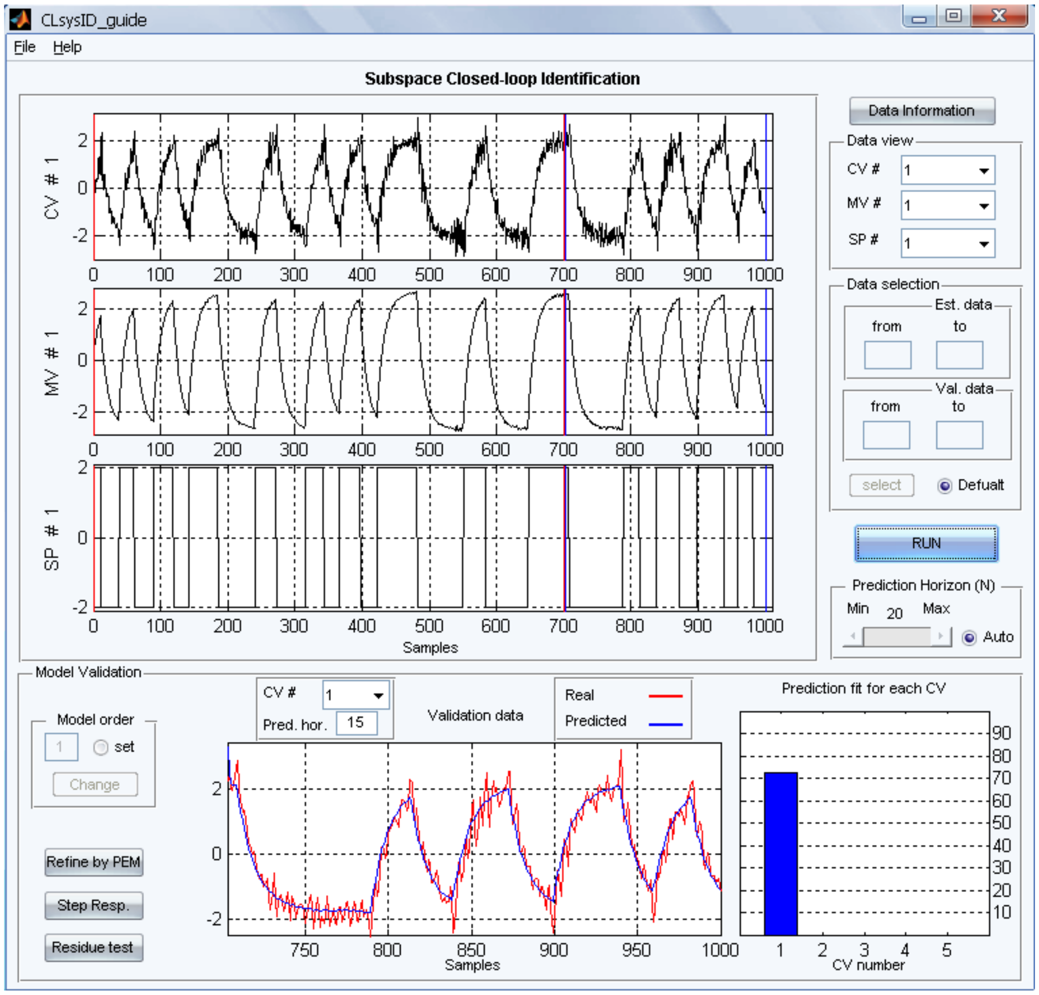


Figure 19: Results of Using the Given Data

The residual values are shown in Figure 20.

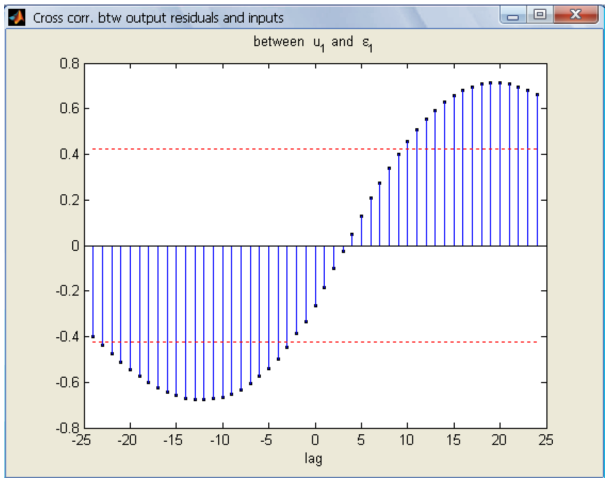


Figure 20: Residual Test Results for Example 1

Example 2

A MIMO example is also provided with the toolbox. The process to be controlled has two inputs and two outputs with transfer functions:

$$G_p = \begin{pmatrix} \dfrac{z^{-1}}{1-0.4z^{-1}} & \dfrac{0.5z^{-2}}{1-0.1z^{-1}} \\ \dfrac{0.3z^{-1}}{1-0.4z^{-1}} & \dfrac{z^{-2}}{1-0.8z^{-1}} \end{pmatrix}$$
$$G_l = \begin{pmatrix} \dfrac{1}{1-0.5z^{-1}} & \dfrac{-0.6z^{-1}}{1-0.5z^{-1}} \\ \dfrac{0.5z^{-1}}{1-0.5z^{-1}} & \dfrac{1}{1-0.5z^{-1}} \end{pmatrix} \tag{20}$$

The controller transfer function is given as

$$G_c = \begin{pmatrix} \dfrac{0.5-0.2z^{-1}}{1-0.5z^{-1}} & 0 \\ 0 & \dfrac{0.25-0.2z^{-1}}{(1-0.5z^{-1})(1+0.5z^{-1})} \end{pmatrix} \tag{21}$$

The required closed-loop data can be generated using the following commands:

```
Gp=tf({[1],[4];[0.3],[1]},{[0 1 -.4],[1 -0.1 0];[ 0 1 -0.1],[1 -0.8 0]},1);
Gl=tf({[1 0],[-.6];[0.5],[1 0]},{[1 -.5],[1 -.5];[1 -.5],[1 -.5]},1);
Gc=tf({[.5 -.2],[0];[0],[.25 -.2 0]},{[1 -.5],[1];[1],[1 0 -.25]},1);
r1=idinput(1000,'rbs',[0,0.03], [-5,5]);
r2=idinput(1000,'rbs',[0,0.03], [-5,5]);
r = [r1 r2];
t=[1:1000]';
[A,B,C,D,K] = tf2ssGpGl(Gp,Gl);
% controller
cont=idss(Gc);
Ac = cont.a; Bc = cont.b; Cc = cont.c; Dc = cont.d;
seeds = [1 2];
```

where tf2ssGpGl is a user-specified function given as

```
function [A,B,C,D,K] = tf2ssGpGl(Gp,Gl)
   ny = size(Gp.OutputDelay,1);
   nu = size(Gp.InputDelay,1);
   NUMGt={Gp.num Gl.num};
   DENGt={Gp.den Gl.den};
```

```
        Gt = tf([NUMGt{1,1} NUMGt{1,2}] ,[DENGt{1,1} DENGt{1,2}] ,1);
        set(Gt,'InputGroup',struct('Noise',[nu+1:nu+ny]))
        model = idss(Gt);
        A = model.a; B = model.b; C = model.c;
        D = model.d; K = model.k;
    end
```

A similar Simulink model to that used in Example 1 can be used to generate the data. After saving the simulated data using the compact format, loading it into the GUI, and pressing the run button, the results similar to that shown in Figure 21 should appear. The residual test results are shown in Figure 22.

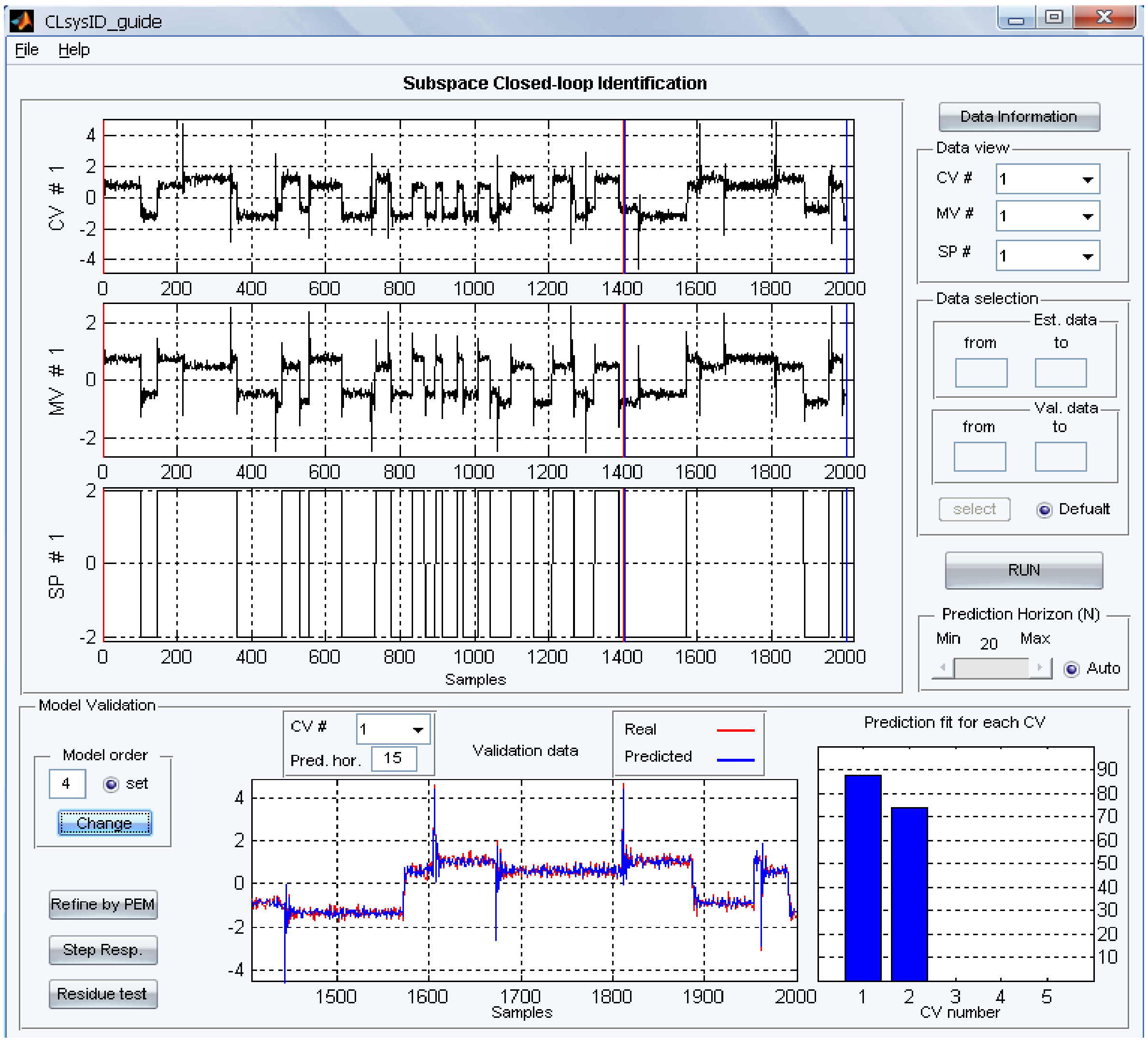


Figure 21: Results of Closed-Loop Identification for Example 2

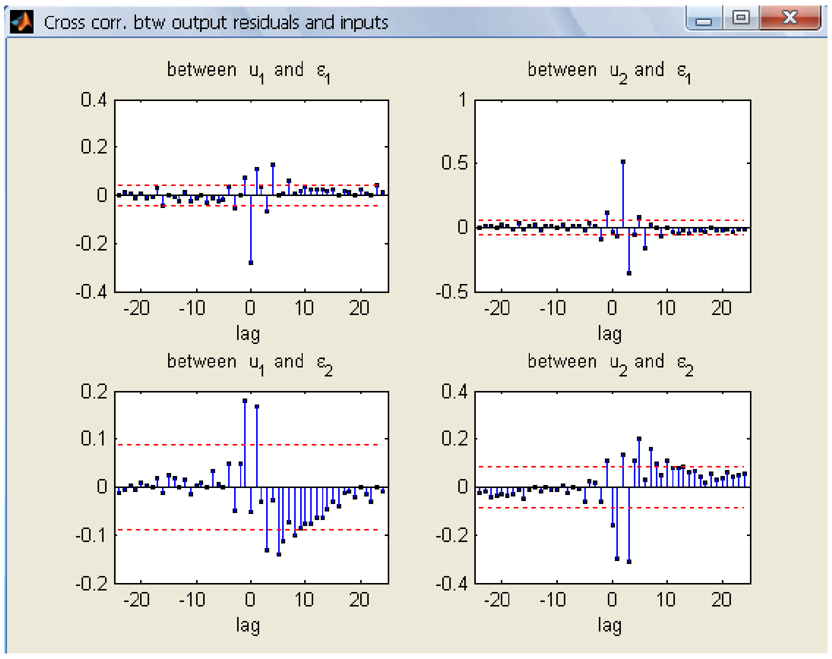


Figure 22: Residual Test Results for Example 2

References

Danesh Pour, N., Huang, B., & Shah, S. L. (2008). Closed-loop subspace identification for performance assessment. *Journal of Process Control* .

De Moor, B., & Van Overschee, P. (1996). *Subspace identification for Linear Systems: Theory, Implementation, Application.* Berlin, Germany: Springer.

Huang, B., & Kadali, R. (2008). *Dynamic Modeling, Predictive Control, and Performance Monitoring.* Springer-Verlag.

Kadali, R., & Huang, B. (2002). Estiamtion of dynamic matrix and noise model for predictive control using closed-loop data. *Industrial and Engineering Chemical Research , 41*, 842-852.

Knudsen, T. (2001). Consistency analysis of subspace identification methods based on linear regression approach. *Automatica , 37*, 81-89.

Larimore, W. E. (1996). Statistical optimality and canonical variate analysis system identification. *Signal Processing , 52*, 131-144.

Ljung, L. (1999). *System Identification Theory for the User.* Upper Saddle River, New Jersey, United States of America: Prentice Hall PTR.

Ljung, L., & McKelvey, T. (1996). Subspace identification from closed-loop data. *Signal Processing , 52*, 209-215.

Moonena, M., De Moora, B., Vandenberghea, L., & Vandewalle, J. (1989). On- and off-line identification of linear state-space models. *International Journal of Control , 49* (1), 219-232.

Söderström, T., & Stoica, P. (1989). *System Identification.* New York, New York, United States of America: Prentice Hall.

Tangirala, A. K., Lakshminarayanan, S., & Shah, S. L. (1997). Closed-loop identification using canonical variate analysis. *47th CSChE Conference.* Edmonton, Alberta, Canada.

Van Overschee, P., & De Moor, B. (1995). A unifying theorem for three subspace system identification algorithms. *Automatica , 31* (12), 1877-1883.

Van Overschee, P., & De Moor, B. (1997). Closed-loop subspace system identification. *Proceedings Of The 36th IEEE Conference On Decision And Control*, *2*, pp. 1834-1853.

Van Overschee, P., & De Moor, B. (1994). N4sid: Subspace algorithm for the identification of combined deterministic-stocahstic systems. *Automatica , 30* (1), 75-93.

Verhaegena, M., & Dewilde, P. (1992). Subspace model identification part 1: the output-error state-space model identification class of algorithms. *International Journal of Control , 56* (5), 1187-1210.