University of Alberta Computer Process Control Group

LQG benchmark for Performance Assessment: Subspace Approach

Limited Trial Version

Written by: CPC Control Group, University of Alberta

Version 1.0

Table of Contents

List of Figures

Introduction

The "LQG benchmark for Performance Assessment: Subspace Approach" toolbox was developed by the Computer Process Control Group at the University of Alberta to allow performance assessment to be performed in MATLAB using a 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_LQGPA`"** to start the toolbox. The GUI shown in Figure 1 should appear.

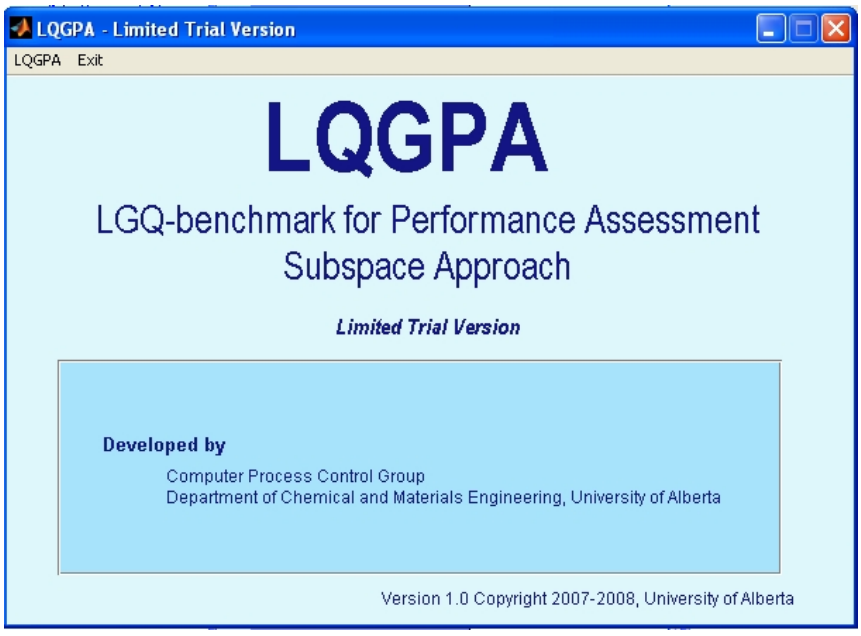

Figure 1: The First GUI that appears

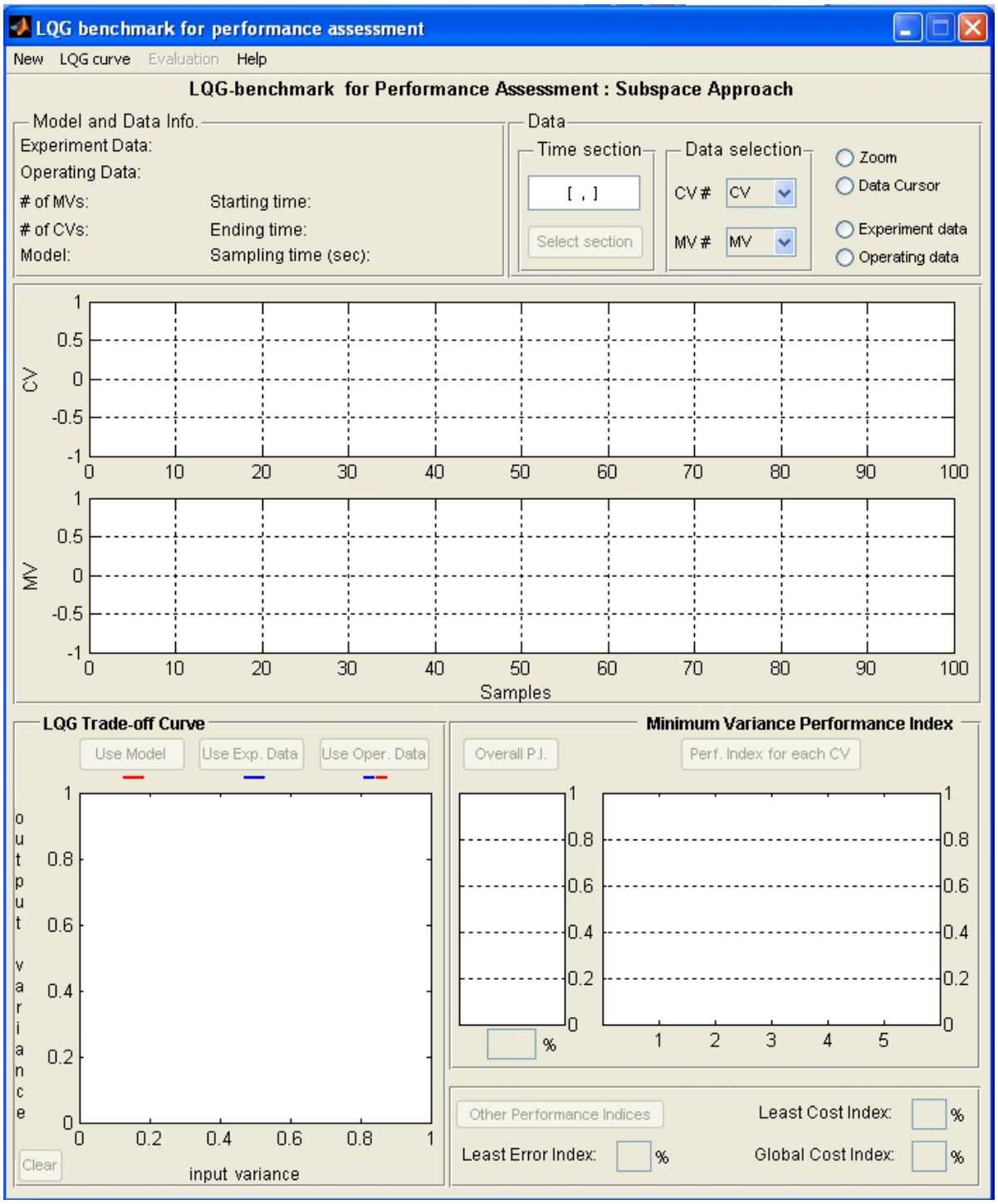

Figure 2: The main GUI for this toolbox

4) Press the **'LQGPA' menu**. A new GUI will appear that is shown in Figure 2.
5) This GUI tool provides different modes of operations to load a model. The GUI accepts any of the following options:

a. Loading an open-loop model of the process (both process and disturbance models), in transfer function or state space form, the LQG trade-off curve can be produced.
b. Loading a set of open-loop experiment (identification) data, the LQG trade-off curve can be produced.
c. Loading a set of closed-loop experiment (identification) data, the LQG trade-off curve can be produced.
d. Loading only process model (with no disturbance model) and a set of closed-loop routine operating data, the LQG trade-off curve can be produced.

6) After load the model or data, as a measure of current performance, a set of closed-loop routine operating data should be loaded.
7) For the purpose of this quick start, we will use the sample data provided for a SISO example in the zip file containing models of process in TF and SS format, a set of open-loop experiment data, a set of closed-loop experiment data and a set of closed-loop operating data.
8) To perform one of the tasks (a) – (d) of section 5, use the "LQGcurve" or "Evaluation" menu to upload model or data. When model or data is uploaded, the corresponding information about it will be shown in the **'Model and Data Info.' panel** (see Figure 3 and 4).

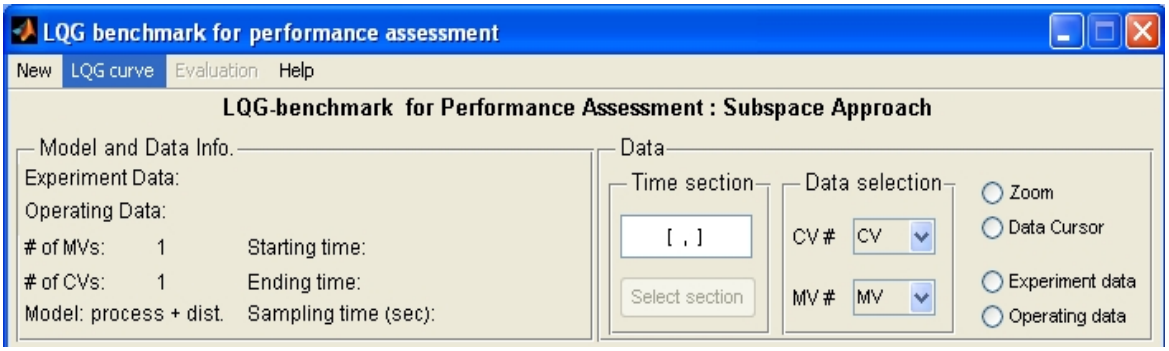

Figure 3: TF model loaded in

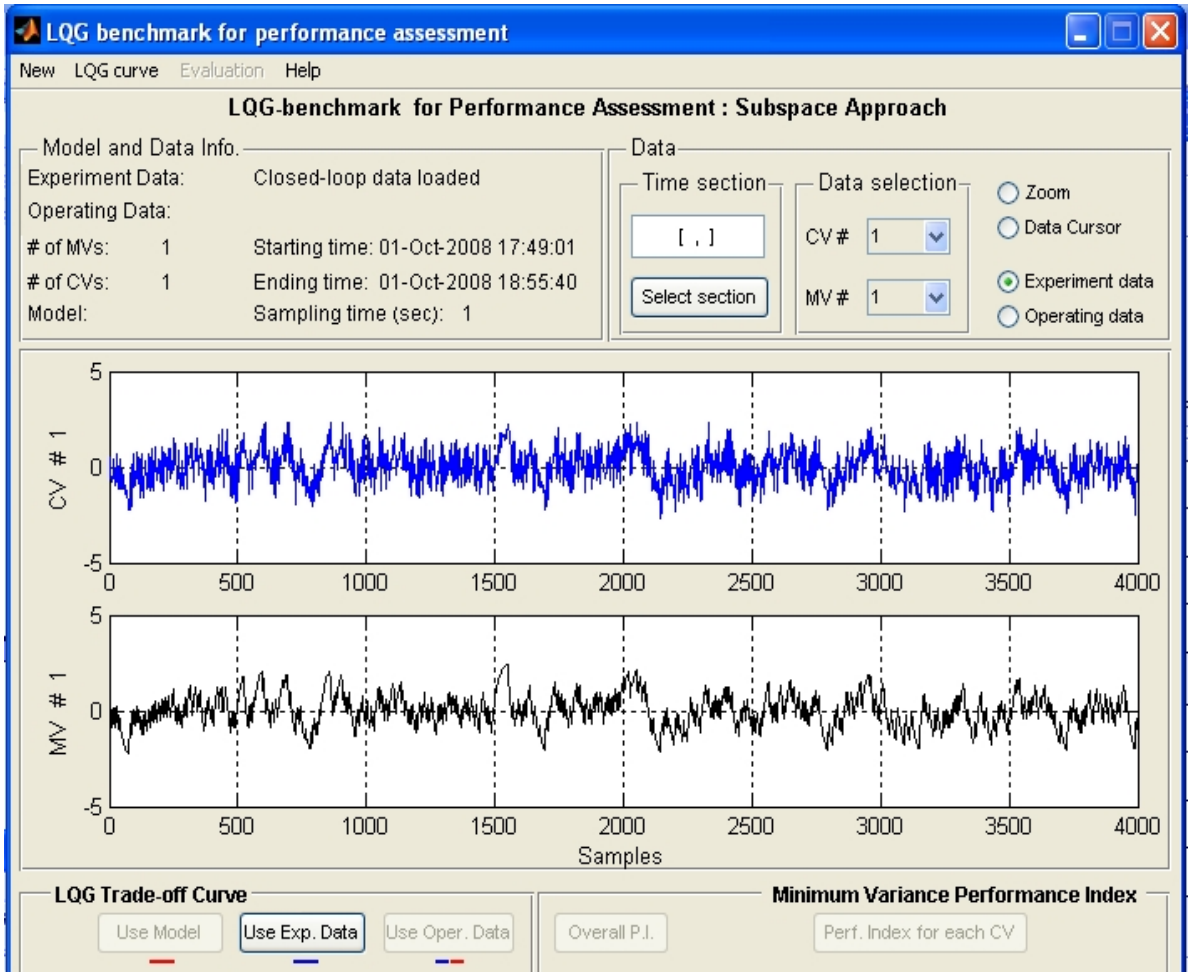

Figure 4: Data information and plots

9) If an open-loop model is loaded, you may press the **'Use Model' button** to generate trade-off curve (in red) as in Figure 5. Depending on the type of process dynamics and number of inputs and outputs, it takes a few seconds to a few minutes to perform the calculation.

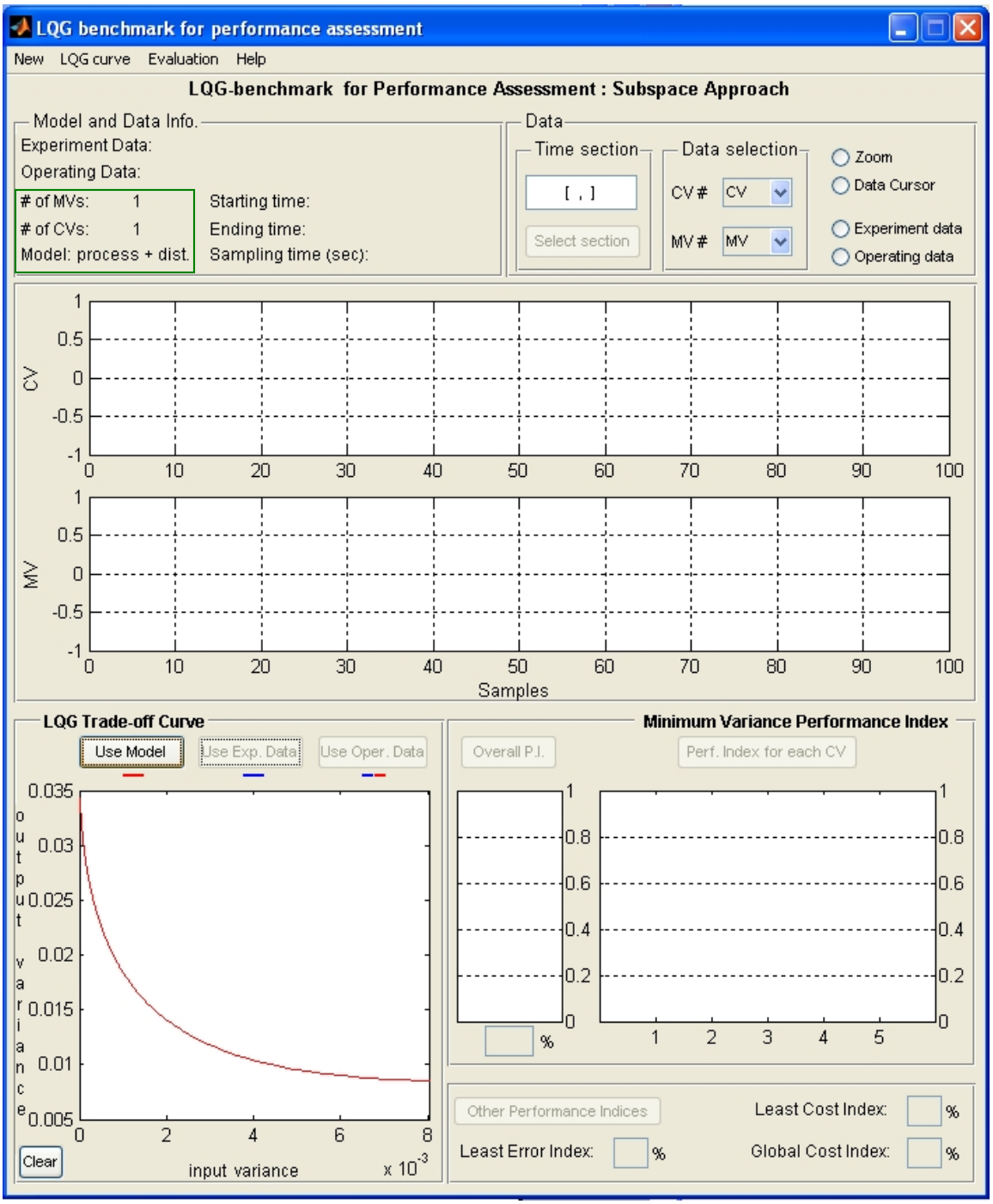

Figure 5: LQG curve calculated using the provided a complete model

10) If a set of closed-loop or open-loop experiment is loaded, you can press ‘Use Exp. Data’ to estimate and plot the curve (in blue) as in Figure 6. Depending on the type of process dynamics, number of inputs and outputs and number of data points, it will need a few seconds to a few minutes to perform the estimation.

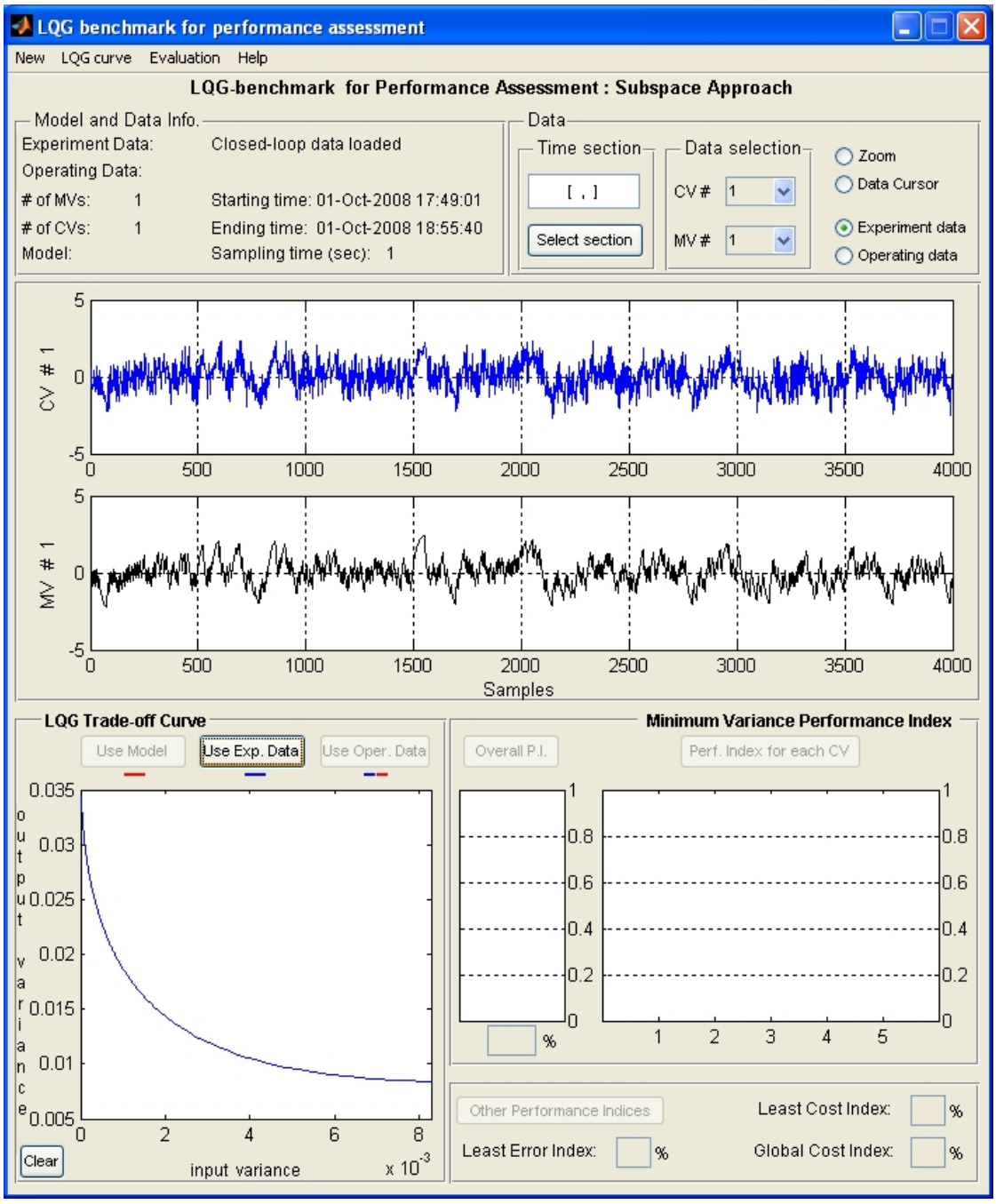

Figure 6: LQG curve estimated form closed-loop experiment data

11) If an open-loop model along with a set of closed-loop operating data is loaded, then you can get the curve by pressing **'Use Oper. Data' button**. The result will be plotted in red-blue dash line style.
12) To perform a comparison between the real curve and estimated curve (when both model and data are available), you may upload a model and a set of identification data and

generate both true curve (computed form the model) and estimated curve (estimated from data) by pressing both buttons one after the other.

13) Upload a set of closed-loop routine operating data. The corresponding input-output variance point (actual control operating point) will appear on the figure as in Figure 7.

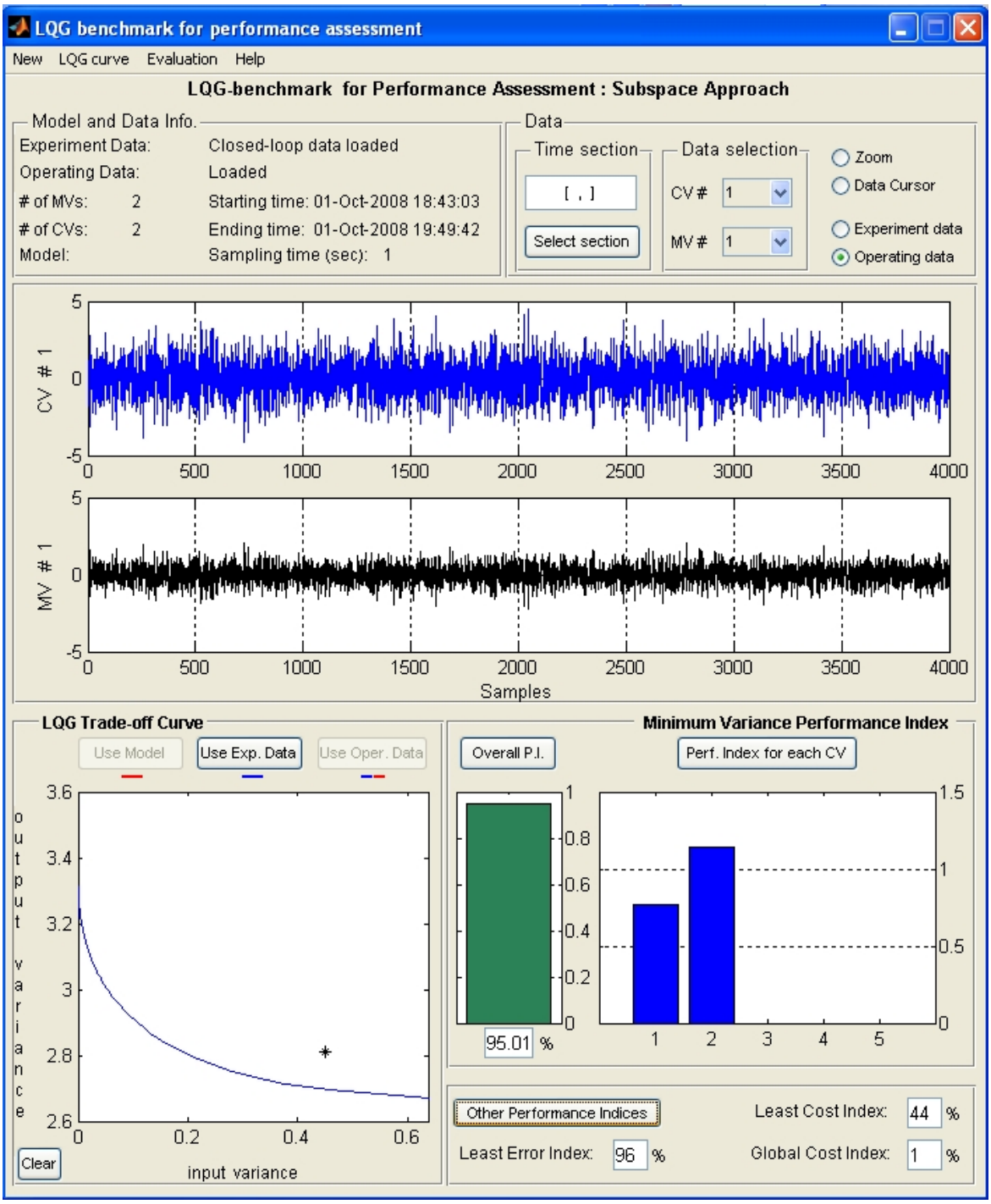

Figure 7: Performance indices obtained for a MIMO process

14) At this time you can have all the performance indices. To get the overall minimum variance performance index press the **'Overall P.I.' button** in the **'Minimum Variance Performance Index' panel**. To find MVC index for each controlled variable, press **'Perf. Index for each CV' button** in the **'Minimum Variance Performance Index' panel**. There are 3 more indices available in the **'Other Performance Indices' panel**. Please see Figure 7. For the definitions of different indices, please see "Detailed Instruction" section of the manual. Note that if either the curve has not been generated yet or operating data has not been uploaded (no actual point), you will receive an error message box when pressing any of the performance buttons.
15) To start a new problem, press **'New' Menu**.

Detailed Instructions

Theory

Optimal LQG controller can be used as a benchmark for performance assessment. The question in performance assessment using LQG-benchmark is how far away is the controller performance from the best achievable performance with the same control effort. In other words, one should solve the following problem:

$$\text{Given } \boldsymbol{E}[\boldsymbol{u}_t^2] < a \text{ , what is min } \boldsymbol{E}[\boldsymbol{u}_t^2] \text{ ?}$$

Solution of this problem is given by a curve named 'trade-off' curve which can be obtained by solving the LQG problem

$$\boldsymbol{J}(l) = \boldsymbol{E}[\boldsymbol{y}_t^2] + l\ \boldsymbol{E}[\boldsymbol{u}_t^2]$$

for different values of l . In this way, various solutions for $\boldsymbol{E}[\boldsymbol{u}_t^2]$ and $\boldsymbol{E}[\boldsymbol{y}_t^2]$ can be calculated and then a curve with the optimal $\boldsymbol{E}[\boldsymbol{y}_t^2]$ as the abscissa and $\boldsymbol{E}[\boldsymbol{u}_t^2]$ as the ordinate is formed from these solutions. Any linear controller can only perform in the region above this curve. In other words, this curve displays the minimal achievable variance of the controlled variable versus the variance of the manipulated variable. Performance of any controller can be evaluated comparing current input and output variances with the trade-off curve.

A sample LQG trade-off curve is shown in Figure 8. In the Figure, the current working point of the control system is also shown. Having the curve and current input and output variances, one

can define some performance indices to compare the performance of current controller to the optimal LQG controller [1,2]. MVC benchmark can also be found from this curve, as shown in the figure which is the case when $l \to 0$. As an example, assume that the actual input and output variances are presented by $\boldsymbol{V_u}$ and $\boldsymbol{V_y}$, respectively. The optimal output variance corresponding to $\boldsymbol{V_u}$ is shown by $\boldsymbol{V_u}^0$ and the optimal input variance corresponding to $\boldsymbol{V_y}$ is defined as $\boldsymbol{V_y}^0$ (see Figure 8). Then following performance indices for output and input variances can be defined:

$$h = \frac{\boldsymbol{V_y}^0}{\boldsymbol{V_y}}$$

$$\boldsymbol{E} = \frac{\boldsymbol{V_u}^0}{\boldsymbol{V_u}}$$

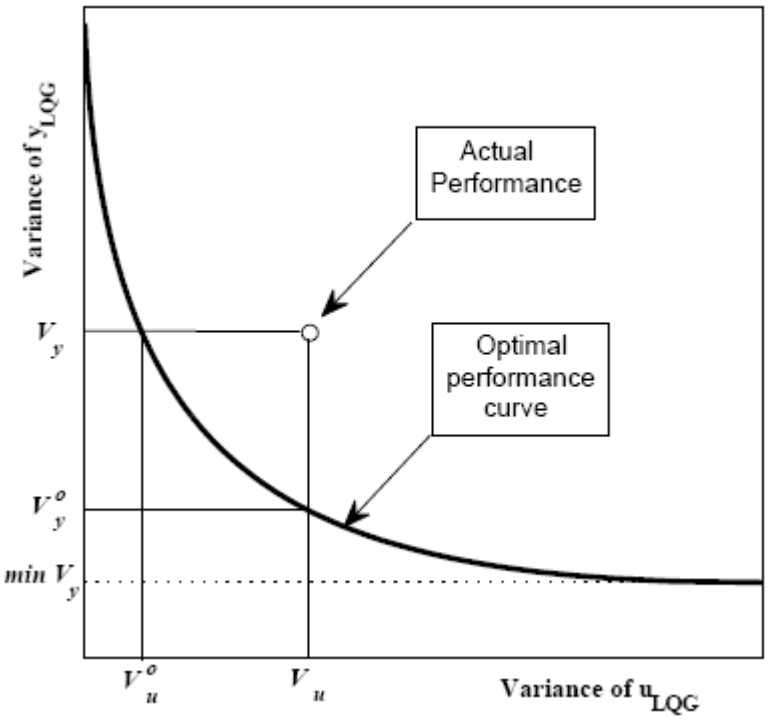

Figure 8: A sample LQG trade-off curve

In contrast to Minimum Variance Control (MVC) benchmark, this approach focuses on both variances of input and output and provides a 'trade-off' curve which represents a feasible range of performance for linear controllers [1,2]. Furthermore, some other indices can be obtained from this method besides the MVC index. A detailed discussion on this issue and the definition of

various indices are presented by Huang (2003) in [3]. Similar to MVC benchmark, this method is also based on a model of the process which has to be obtained through process identification. Kadali and Huang (2002) have presented a method for calculating the LQG trade-off curve in a subspace framework [2,4] based on the LQG design given by [5,6]. A closed-loop identification method for estimation of the required open-loop subspace matrices from closed-loop data is also provided by them in [7].

An improved version of this identification method is recently proposed in [8] which proves the consistency of noise variance estimation. This work also presents the method of disturbance model estimation using an available process model and a set of closed-loop routine operating data (no identification experiments required). This GUI is based on the work proposed in [8]. For details of the algorithms, **please see Ref. 2 and 8**.

Data Storage

There is a code provided in file "**gen_cmpct_data_LQG.p**" which can be used to generate data or model objects and save them on the disk for using in the GUI. Model and data formats generated by this code are as follows:

Model Storage Format

The model storage file contains information about the multivariate model that is being used to describe the system. It must be entered as a transfer function object (tf) or state space object (ss). The transfer function matrix, for each of the process or disturbance models, can be created using the following steps:

1) Assume that a process with *n* inputs and *m* outputs is being analyzed. The transfer function between the j^{th} input and the i^{th} output is a rational function of either *s* or *z*, given as *Aij* / *Bij*, where *A* is the numerator and *B* is the denominator , and both are vector containing in descending order of powers of *s* or *z* the corresponding co-efficients of A and B, respectively.
2) It is now necessary to create 2 cell arrays that contain all the information about the process. The first cell array, **A** , is created as follows:
   The first row contains the transfer function between the first output and each of the inputs, while the second row contains the transfer function between the second output and each

of the inputs. It should be noted that the curly brackets, {}, are used to create a cell array. In order to access, the (*i*, *j*) entry of the array, the command would be **A**{*i*, *j*}. A similar cell array containing the values of the denominator, denoted as **B** , is also created.

3) The time delay for the model can be created by forming the matrix **D** , such that the (*i*, *j*) entry contains the time delay associated with the transfer function for the *i*th output and *j*th input. The array simply contains the numeric values.
4) The continuous transfer object can then be created using the following MATLAB command: **"`>>Atf=tf(A,B,'ioDelay',D)`"**, where **A** is the cell array **A** , **B** is the cell array **B** , and **D** is the time-delay matrix, **D** . To create a discrete time transfer function, the MATLAB command would be **"`>>Atf=tf(A,B,Ts,'ioDelay',D)`"**, where **Ts** is the sampling time. If it is desired to create a discrete model that contains powers of "z^{-1}", then the command would be
   "`>>Atf=tf(A,B,Ts,'ioDelay',D,'Variable','z^-1')`"
5) Creating a MIMO state space model can be easily done by providing A,B,C,D,K matrices to the MATLAB function **"`>>sys=ss(A,B,C,D,Ts)`"**. For continuous-time model, **Ts** should be left blank.

Data Storage Format

The data format is called *Compact* format. Assume that there are ***p*** samples of output, input and setpoint data with a total of ***t*** controlled variables and ***s*** 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 *p*-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 *p*-by-1 cell matrix that contains the time stamp for each of the samples.

4) **dbl_Compact_Data**: This is a (2*p*+*t*)-by-*q* double matrix (for CL experiment data) or a (*p*+*t*)-by-*q* double matrix (for OL experiment data and CL operating data) that contains the values for each of the tags and sample periods.
5) **dbl_SamplingTime**: This is a scalar double that contains the sampling time for the process.
6) **int_CVNumber**: This is a scalar integer that contains the number of controlled variables in the process, that is, *t*.
7) **int_MVNumber**: This is a scalar integer that contains the number of manipulated variables in the process, that is, *s*.
8) **status**: This is a *p*-by-(*s* + *t*) 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 and Model Generation

The data and model storage files can be generated using "**gen_cmpct_data_LQG.p**". Based on the step-by-step comments after running the code, you would be able to create your model or data file in a *Compact* format. To generate a complete model (process + disturbance), you have two choices:

1) A **TF**-object as process model and a **TF**-object as the disturbance model must be ready in the Workspace.
2) State space matrices **A,B,C,D** and **K** must be ready in your Workspace.

Note that you can only use one of these two types of model to generate a *Compact* model.

For process model (only process model, no disturbance model),

1) It can be provided by state space matrices **A,B,C** and **D**.
2) It can be provided as one **TF**-object.

Note that a set of closed-loop routine operating data is required with this choice of model.

A set of **closed-loop** or **open-loop** experiment data (identification data) must be available in the Workspace, if you want to estimate the LQG curve from experiment data.

For creating closed-loop **operating** data in *Compact* format (for the **evaluation** purpose), you have to get your data ready in the Workspace and run the code.

By running the code one time, you would be able to generate and save one model, one set of experiment data and one set of operating data at the same time. Note that you may skip some steps based on your choices.

Run the code:

If your data and/or model information are ready in the Workspace, run the code. The followings will appear on the command window:

```
**************************************************************
Consider a process with m inputs , p outputs and d disturbances:
The following variables should be available in WORKSPACE:

 1. LQG CURVE REQUIREMENTS:
     - process and disturbance model in the form of
       p x m and p x d transfer function matrices (tf-object)
       And White noise covariance matrix
 OR
     - process and disturbance model in the form of
       a State-Space model: A,B,C,D,K matrices
       And White noise covariance matrix
 OR
     - only process model
       in SS format (A,B,C,D) of TF format (p x m tf-object)
       And
       CL Operating data; output and Input data (y,u):  N x(p+m) matrix

 AND / OR

     - CL Excitation data; Output, Input and Setpoint data (y,u,r): N x
(p+m+p) matrix

 AND / OR
     - OL Excitation data; Output and Input data (y,u): N x(p+m) matrix

 2. PERFOMANCE EVALUATION REQUIRMENT
     - CL Operating data; output and Input data (y,u):  N x(p+m) matrix
```

```
 3. Sampling time

************************************************************
Press ENTER to continue or type X to exit :
```

At this stage, you may want to stop going further (by typing X) to prepare your model matrices or load data into the Workspace.

You may press ENTER to go further. **Step 1** is to provide a complete model (process + disturbance), if applicable. You will see the first step in the command window:

```
1. Provide process and disturbance models....

press ENTER to provide TF model or type X to go to the next part :
```

If presses ENTER, the following will appear:

```
process model transfer function matrix (tf-object):
```

Give the name that you have defined for your **process** model tf-object, e.g. Gp, and press enter. The following will appear:

```
disturbance model transfer function matrix (tf-object):
```

Provide the name that you have defined for your **disturbance** model tf-object, e.g. Gd, and press enter. The following will appear:

```
noise covariance matrix:
```

Give the input noise covariance matrix.

If you skip transfer function part (by typing X), you will be asked for a model in state space format:

```
press ENTER to provide SS model or type X to go to the next part :
```

By pressing ENTER you will be asked for state space matrices:

```
state space A matrix:
state space B matrix:
state space C matrix:
state space D matrix:
state space K matrix:
noise covariance matrix:
```

You can skip this part as well. In this case, you will have the opportunity to provide a process model (no noise model) as follows (**Step 2**):

```
2. Provide process model and CL operating data....
press ENTER to provide TF model or type X to go to the next part :
```

By pressing ENTER, the following will appear:

```
process model transfer function matrix (tf-object):
```

By typing X, the following will appear:

```
press ENTER to provide SS model or type X to go to the next part :
```

If you choose to provide a state space model, the code will ask for the matrices:

```
state space A matrix:
state space B matrix:
state space C matrix:
state space D matrix:
```

Closed-loop operating data should then be provided:

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

Either skipping steps (1) and (2) or not , the next step (**Step 3**) asks for closed-loop excitation data (if you want to provide):

```
3. Provide Closed-loop excitation data matrix [y u r],
Or press ENTER, if you want to skip this part,
```

Give the data arranged in the given order: output, input and setpoint. Keep the '[' and ']' symbols. If presses enter without providing closed-loop experiment data, you may provide open-loop experiment data after the following (**Step 4**):

```
4. Provide Open-loop excitation data matrix [y u],
Or press ENTER, if you want to skip this part,
```

If you want to generate and save closed-loop operating data, the next command (**Step 5**) you see will allow you to do that:

```
5. Provide Closed-loop operating data matrix [y u]:
```

Note that you can **not** generate both experiment and operating data in *Compact* format by running the code once.
You will be asked about the number of process outputs for any of these set of data that you provided (if you have not provided any model).

Sampling time will be asked at this stage:

```
Sampling time:
```

Names of the files for saving the *Compact* model or *Compact* data are requested at the last step:

```
The file name to save compact data (e.g. test_cmpct_data):
The file name to save transfer function model (e.g. tf_model):
The file name to save state space model (e.g. ss_model):
```

`The file name to save mpc model (e.g. mpc_model):`

The “gen_cmpct_data_LQG” code, will save some .mat files on the current active path of MATLAB which you can upload and use in the GUI for analysis.

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_LQGPA`”.**

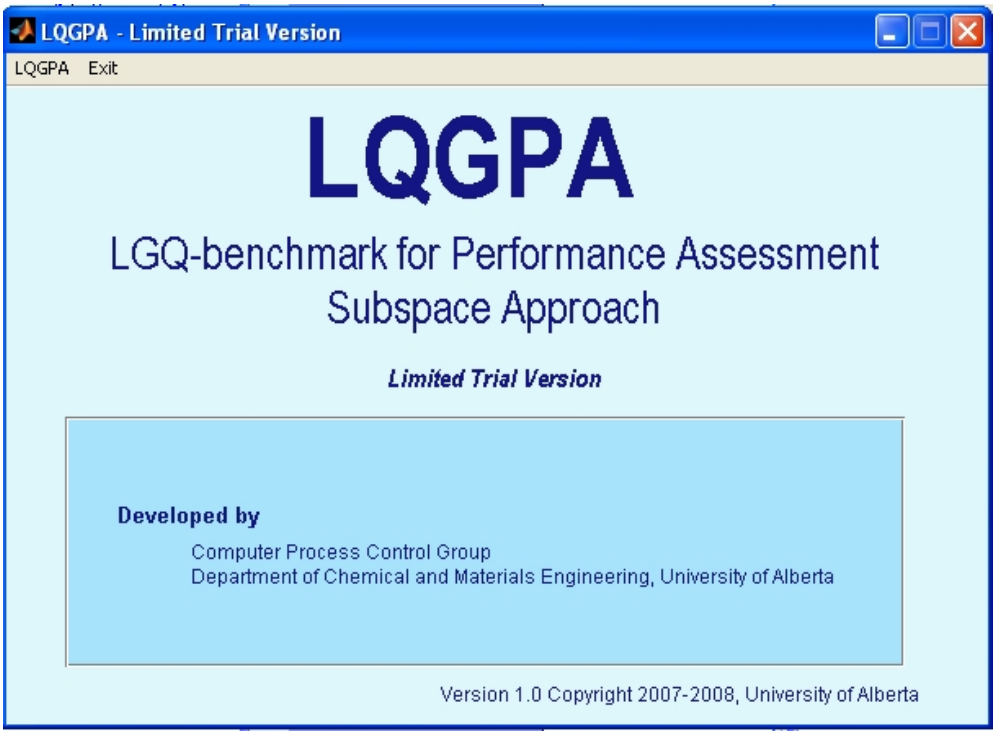

Figure 9: The first GUI that appears

The GUI shown in Figure 9 should appear. This GUI is the main access to the toolbox. To start a session of the toolbox, click on the **'LQGPA' menu**. This will bring up a new GUI, which is shown in Figure 10. In Figure 10, each of the main parts of the GUI is separated by panels and will be discussed separately.

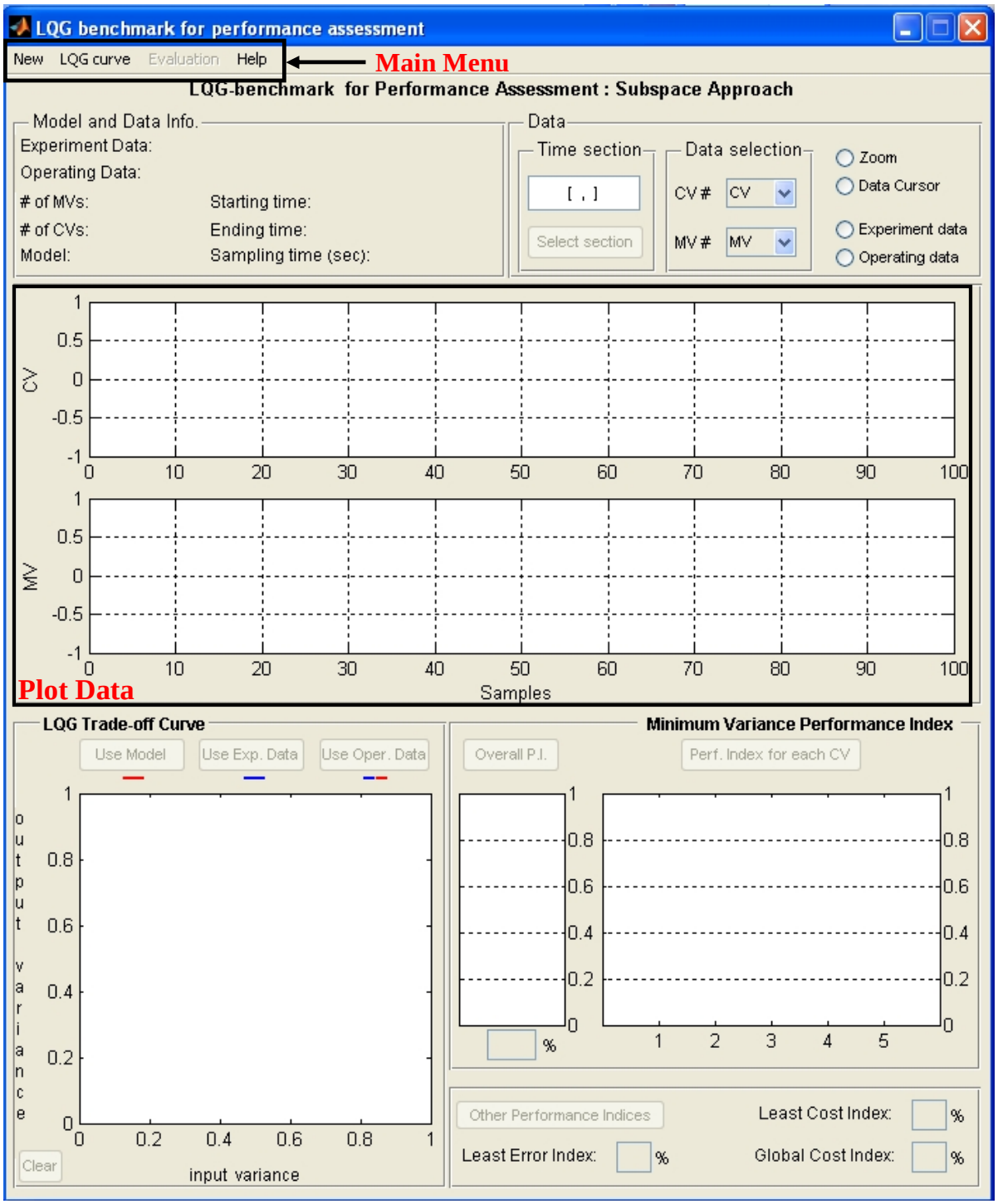

Figure 10: The main GUI for this toolbox

In-depth Discussion of the Toolbox

Section 1: Main Menu

The **'Main' Menu** consists of the following 3 areas:

1) **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.
2) **LQGcurve:** Clicking this menu will allow the user to select different options of closed-loop or open-loop data or models (generated by "gen_cmpct_data_LQG.p") that will be used in the analysis. Four choices are available:

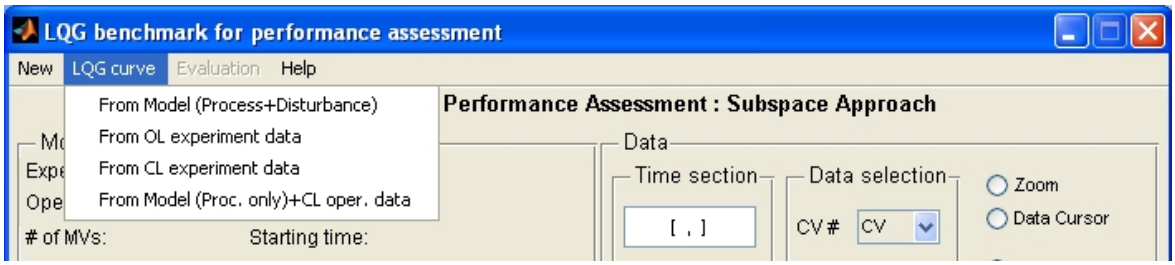

Figure 11: The 'LQGcurve' menu from the 'Main' menu

   a. **From Model (Process + Disturbance):** This option allows the user to load a complete model of the process which contains both process and disturbance models to be used for the calculation of LQG curve.
   b. **From OL experiment data:** This assumes that the data to be used for estimation of the curve is a set of open-loop experiment (identification) data stored in *Compact* format.
   c. **From CL experiment data:** This assumes that the data to be used for estimation of the curve is a set of closed-loop experiment (identification) data stored in *Compact* format.
   d. **From Model (Proc. only)+CL oper. data:** You can use this choice to load a process model and a set of closed-loop operating data (in one file). Note that when you upload this file, the process model and operating data will be used to calculate the curve, so for the evaluation purpose you may want to upload another set of operating data (which can be the same set) from the **'Evaluation' menu**.

3) **Evaluation:** Clicking this menu will allow the user to select and upload a set of closed-loop routine operating data (generated by "gen_cmpct_data_LQG.p") for evaluating the performance of current controller. Only one choice is available:

   a. **Load CL operating data:** This allows the user to upload a set of closed-loop routine operating data.

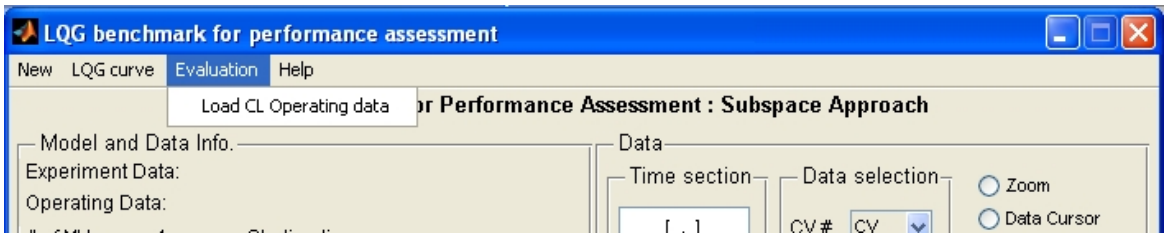

Figure 12: The 'Evaluation' menu from the 'Main' menu

Section 2:' Model and Data Info.' panel

This panel shows the information about the type of model you have loaded in (if any) as well as experiment data (if any) and operating data. Number of CVs and MVs, sampling time, Starting and Ending time of the data are other information provided in this panel.

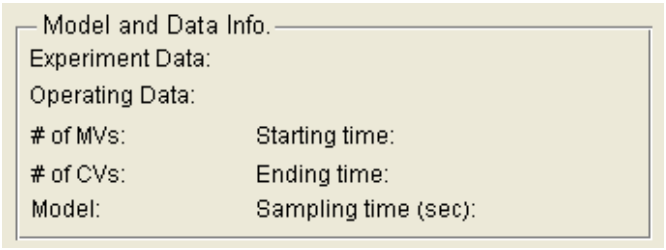

Figure 13: The 'Model and Data Info.' panel

Section 3: 'Data' panel

There are some tools in this panel which can be used for data selection and vision as follows:

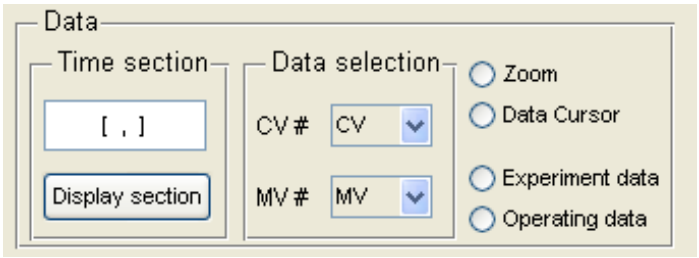

Figure 14: The 'Data ' panel

1) **Time section:** In this filed, the user can specify what part of the total data is to be used in the analysis. The format for this entry is "[start sample value, end sample value]". A comma must separate the 2 values and the end value must be greater than the start value. As well, there must a sufficient number of data samples in order for the computer to estimate a model. Clicking "Select section" will display the selected samples in the 'plot data' section.
2) **Data selection:** This sub-pane enables you to choose the controlled variable (CV) and manipulated variable (MV) that you want to show in the 'plot data' section.
3) **Zoom:** This allows the user to zoom in onto a certain section of the graph. However, there is no zoom-out button, so you have to double click the left mouse button to get back to the original zoom.
4) **Data Cursor:** This can be used to monitor the exact value of any data points.
5) **Experiment/Operating data:** By choosing each of these two radio buttons, you can choose to show either experiment data or operating data in the 'plot data' section.

Section 4: Plot data

Two axes have been provided in this section which allows the user to view MV and CV data. Each one can only show one set of data, so you should use **'Data selection' sub-panel** in the **'Data' panel** to choose the number of CV or MV you want to see.

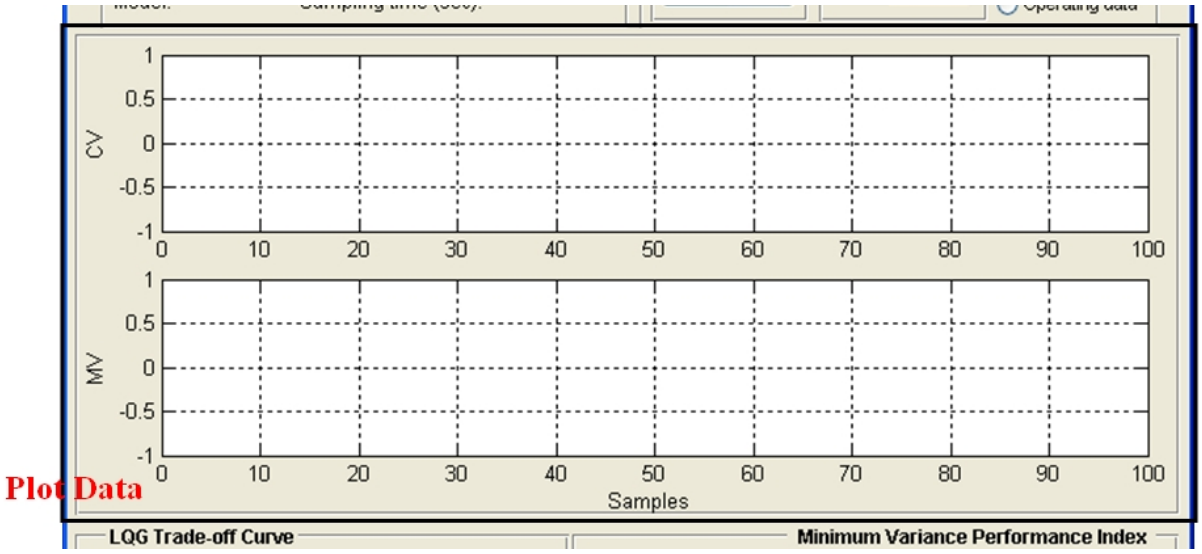

Figure 15: The panel being used for plotting data

Section 5: 'LQG Trade-off Curve' panel

In this panel, the LQG trade-off curve calculated using the model, estimated from data or both at the same time is plotted on the provided graph. Four push buttons are available in this panel:

1) **Use Model:** This button is enabled after you loaded a complete model (Process + Disturbance) in the GUI. Press it to calculate and plot the trade-off curve (in red).

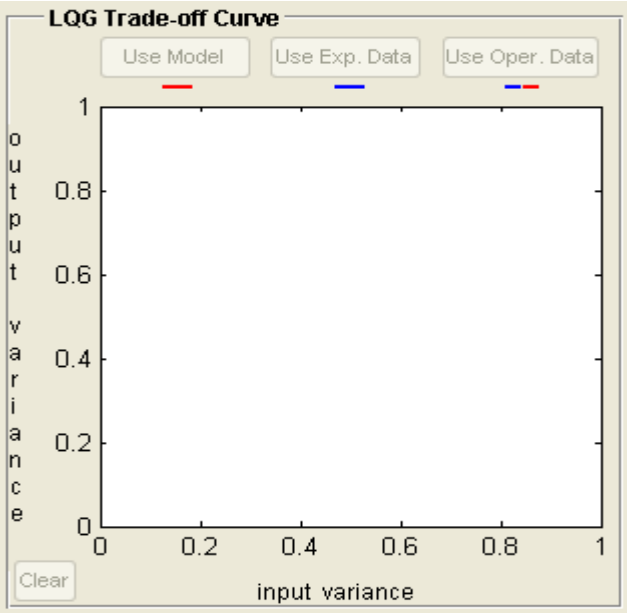

Figure 16: 'LGQ Trade-off Curve' panel

2) **Use Exp. Data:** This button is enabled, if you have provided a set of closed-loop or open-loop experiment data to estimate and plot the trade-off curve (in blue).
3) **Use Oper. Data:** This button is enabled, if you have provided process model and a set of closed-loop operating data to estimate and plot the trade-off curve (in dashed blue-red).
4) **Clear:** You may use this button to clear the graph.

Section 6: 'Minimum Variance Performance Index' panel

In this panel, Minimum Variance Control (MVC) benchmark (obtained from the trade-off curve) is used to calculate the MVC performance index. Note MVC benchmark calculated here automatically takes all non-minimum phase zeros into account. See Chapter 10 of book by Huang and Shah (1999) [1] for explanation of MVC benchmark for non-minimum phase systems.

There are two axes in this panel: one for overall MVC performance index and the other one for individual performance indices for each CV. Two corresponding push buttons are also provided:

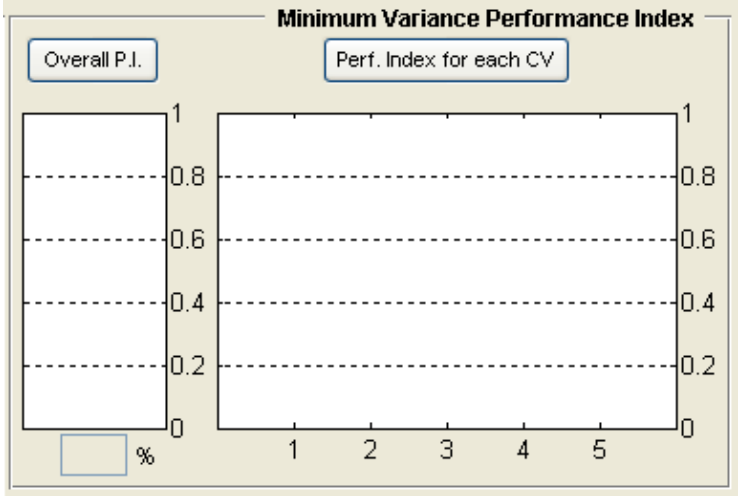

Figure 17: 'Minimum Variance Performance Index' panel

1) **Overall P.I.:** Press this button after you a trade-off curve has been calculated or estimated and a set of closed-loop operating data is also loaded in (the actual variance point is present on the graph of **'LQG Trade-off Curve' panel**) to see the overall MVC performance index for the process.
2) **Perf. Index for each CV:** Press this button after a trade-off curve has been calculated or estimated and a set of closed-loop operating data is also loaded in (the actual variance point is present on the graph of **'LQG Trade-off Curve' panel**) to see the MVC performance index for each of the controlled variables of the process.

Section 7: 'Other Performance Indices' panel

Three other performance indices can also be found from the LQG trade-off curve which are achievable in this panel by pressing the embedded push button. Note that based on the position of actual variance point in relation to the trade-off curve, some of these indices might not be calculable.

If more that one curve is present in the graph. i.e. both 'True' and 'Estimated' curves, the one that has been produced most recent will be used for indices calculations.

Figure 18: 'Other Performance Indices' panel

Example 1

A set of models and data generated by "gen_cmpct_data_LQG" are provided with this toolbox which come from the following system:

$$\boldsymbol{x}_{t+1} = \begin{pmatrix} 0.6 & 0.6 & 0 \\ -0.6 & 0.6 & 0 \\ 0 & 0 & 0.7 \end{pmatrix} \boldsymbol{x}_t + \begin{pmatrix} 1.6161 \\ -0.3481 \\ 2.6319 \end{pmatrix} \boldsymbol{u}_t + \begin{pmatrix} -1.1472 \\ -1.5204 \\ -3.1993 \end{pmatrix} \boldsymbol{e}_t$$

$$\boldsymbol{y}_t = \begin{pmatrix} -0.4373 & -0.5046 & 0.0936 \end{pmatrix} \boldsymbol{x}_t - 0.7759 \boldsymbol{u}_t + \boldsymbol{e}_t$$

A PI controller, $[\,0.1 + \frac{0.05}{\boldsymbol{s}}\,]$ is used for this process. A simple Simulink model has been prepared to collect closed-loop experimental and operating data (see Figure 11). $\boldsymbol{r}(\boldsymbol{t})$ is the identification test signal designed by '*idinput*' command:

```
r=idinput(6001,'rbs',[0,0.06], [-1,1]);
```

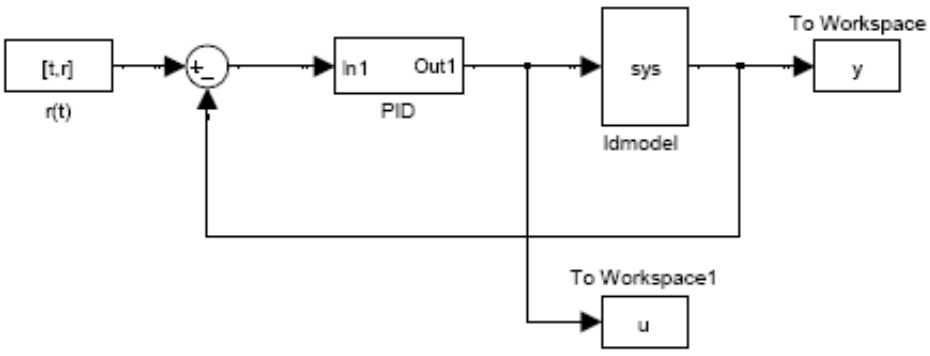

Figure 19: Simulink model for collecting closed-loop data

The ‘idmodel’ block needs a variable named ‘sys’ to be defined and be ready 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 “Add noise” which must be checked. To get a set of routine operating data $\boldsymbol{r}(\boldsymbol{t})$ must be replaced by a zero column vector of length 6001.

A similar model has been used to generate open-loop experiment data.

When model and data variables are present in Workspace, we run “gen_cmpct_data_LQG” to create and save model and data files in *Compact* format. All the files related to this example have names started with “**SISO_**”.

Loading model and data files in GUI, the LQG trade-off curve can be estimated and various performance indices can be calculated. Final results are shown in Figure 20 which indicates relatively poor performance of the controller in all aspects.

Example 2

A MIMO example is also provided with this toolbox. The process to be controlled is a 2 input, 2 output process described by the following transfer function matrices:

$$\boldsymbol{G_p} = \begin{pmatrix} \dfrac{z^{-1}}{1-0.4z^{-1}} & \dfrac{0.5z^{-2}}{1-0.1z^{-1}} \\ \dfrac{0.3z^{-1}}{1-0.4z^{-1}} & \dfrac{z^{-2}}{1-0.8z^{-1}} \end{pmatrix} \qquad \textit{and} \qquad \boldsymbol{G_l} = \begin{pmatrix} \dfrac{1}{1-0.4z^{-1}} & \dfrac{-z^{-1}}{1-0.1z^{-1}} \\ \dfrac{z^{-1}}{1-0.7z^{-1}} & \dfrac{1}{1-0.8z^{-1}} \end{pmatrix}$$

The following controller is implemented on the process:

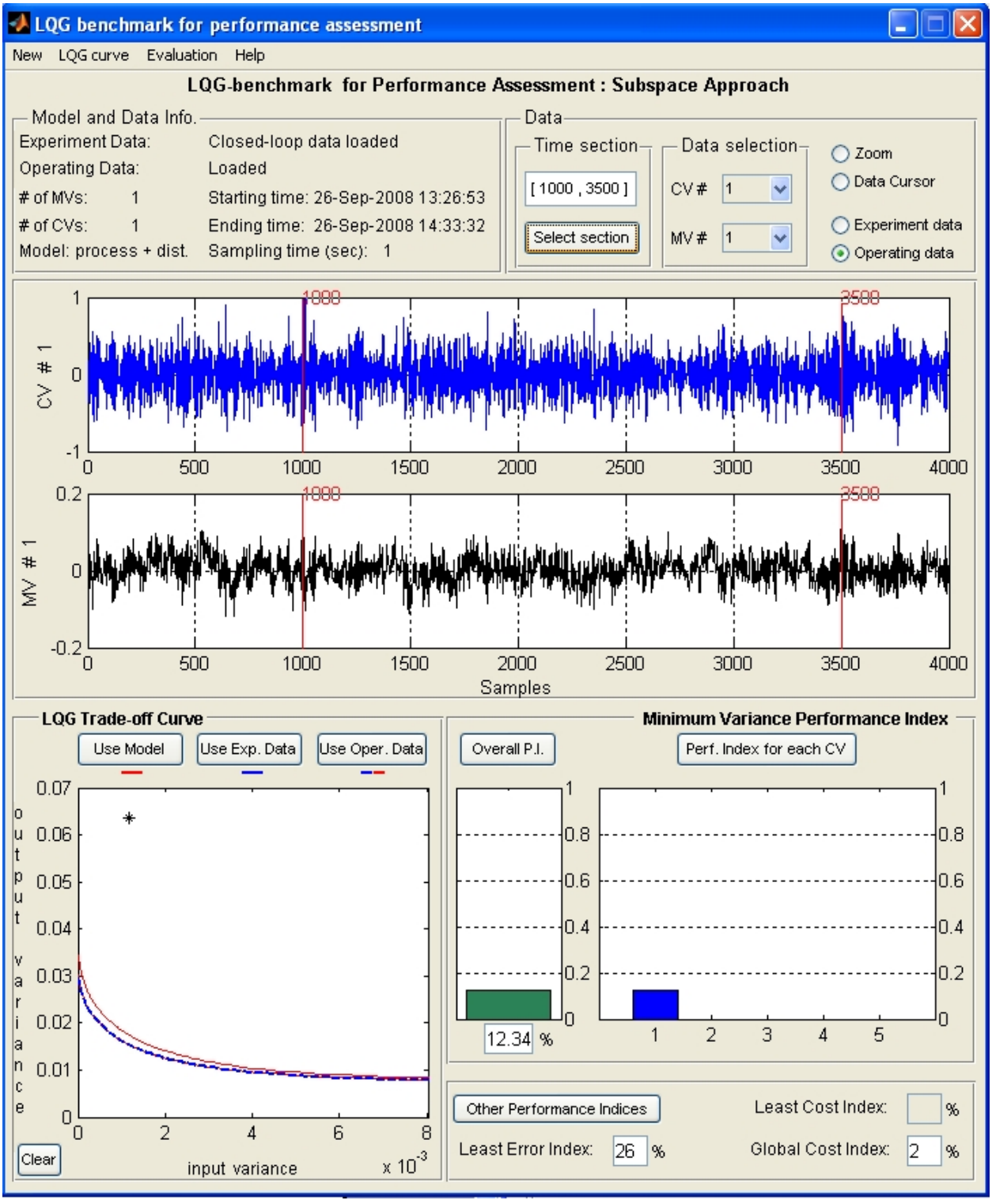

Figure 20: Results of performance assessment for example 1

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

The following procedure can be followed to generate some set of closed-loop experiment data or operating data from this process:

```
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(3000,'rbs',[0,0.1], [-5,5]);
r2=idinput(3000,'rbs',[0,0.1], [-5,5]);
r = [r1 r2];
% for operating data
% r = zeros(3000,2);
t=[1:3000]';
[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];


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
```

Then use a Simulink model similar to Example 1 to generate data.

The same procedure as the previous example is followed to create and save *Compact* model and data files, load them in the GUI and obtain the trade-off curve as well as performance indices. All the files related to this example have names started with "**MIMO_**". Results are shown in Figure 21 which indicates that the overall control system is working similar to the minimum variance controller with the cost of a very small Global Cost index which indicates that the control action could be reduced significantly for the same output variance.

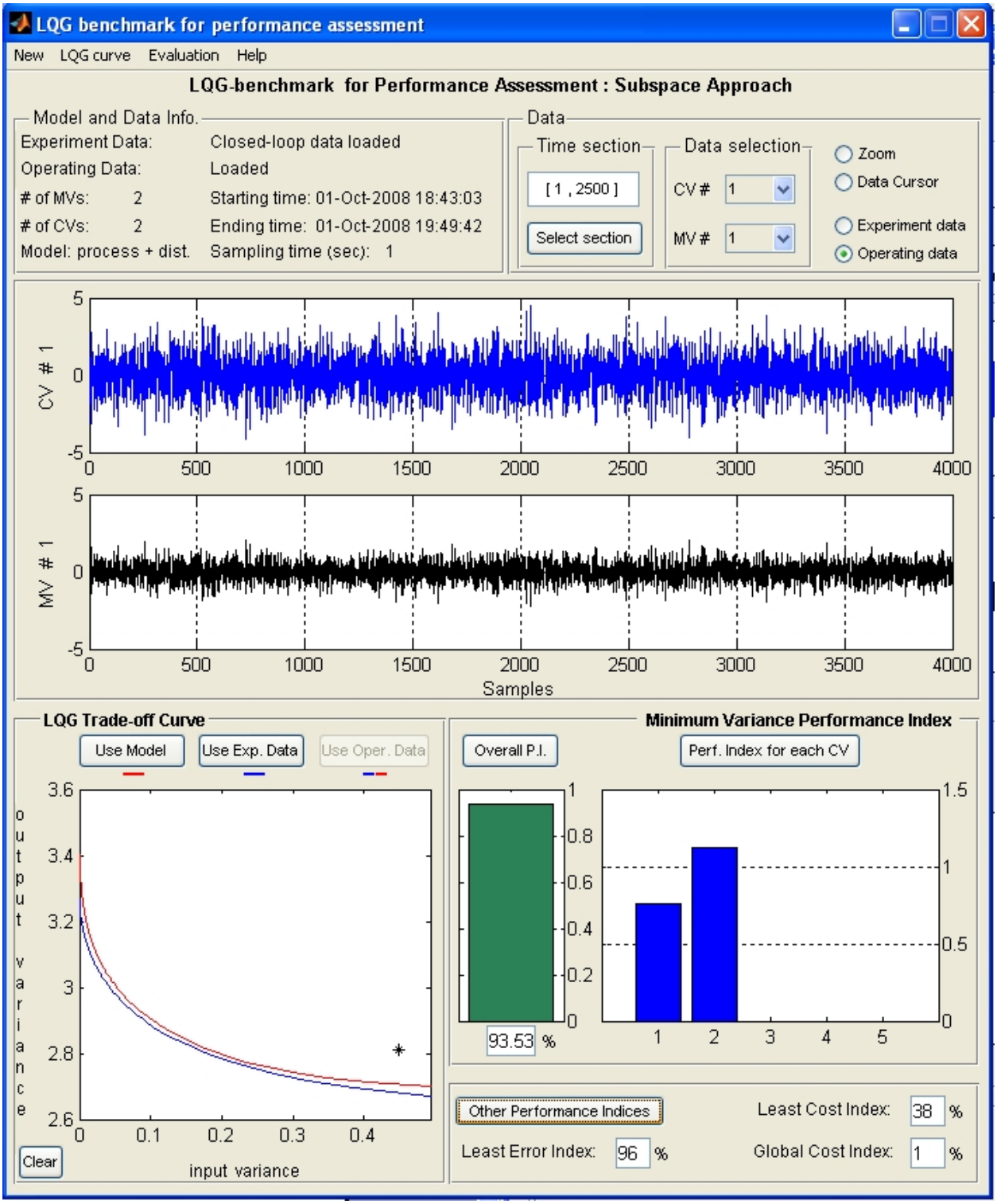

Figure 21: Results of performance assessment for example 2

References

1- Huang, B. and Shah S.L. *Performance Assessment of Control Loops: Theory and Applications*. Springer, 1999.

2- Huang, B. and Kadali, R. *Dynamic Modelling, Predictive Control and Performance Monitoring; A data-driven subspace approach*. Springer-Verlag, London, UK, 2008.

3- Huang, B. A pragmatic approach towards assessment of control loop performance. *Int. J. Adapt. Control and Signal Processing*, 17: 589–608, 2003.

4- Kadali R. and Huang B. Controller performance assessment using LQG-benchmark obtained under closed loop. *ISA Transactions*, 41: 521-537, 2002.

5- Favoreel W., DeMoor B., Gevers M. and van Overschee P. Closed loop model-free subspace-based LQG-design. Tch. rep. 98-108, Katholieke Universiteit Leuven, 1998.

6- Favoreel W., DeMoor B., van Overschee P.and Gevers M. Model-free subspace-based LQG-design. In Proc. of the American Control Conference, pp. 3372-3376, 1999.

7- Kadali R. and Huang B. Estimation of dynamic matrix and noise model for model predictive control using closed-loop data. *Ind. Eng. Chem. Res.* 41, 842-852, 2002.

8- Danesh Pour, N. Huang, B. and Shah S.L. Closed-loop subspace identification for performance assessment. Submitted to *Journal of Process Control*, October 2008.