#### Getting Started with the System Identification Assistant

June 2008, 371246B–01

This help file teaches you how to acquire and analyze signals and models with the LabVIEW System Identification Assistant. You can use the System Identification Assistant to identify the characteristics of a system through signal processing and mathematics.

This help file contains exercises to teach you how to load and analyze sample signals and estimate and compare sample system models using the System Identification Assistant. You can learn how to complete the following procedures:

- Launch the LabVIEW System Identification Assistant in the LabVIEW SignalExpress environment
- Load signals and rebuild missing data
- Split data to use for model estimation and validation
- Normalize data for ease of calculation
- Estimate parameters of a system model
- <u>View model information</u> in the SignalExpress environment
- <u>Compute the pole-zero plot</u> for a model
- <u>Validate</u> the accuracy of a model
- <u>Compare system models</u> through <u>model validation</u>
- Save models to a file

The exercises in this help file take a short amount of time to complete. However, each section of this help file builds upon previous sections so complete each exercise in order.

Next: System Identification Assistant Related Documentation

© 2004–2008 National Instruments Corporation. All rights reserved.

# System Identification Assistant Related Documentation

The following documents contain information that you might find helpful as you use the LabVIEW System Identification Assistant.

Note You must install LabVIEW and the LabVIEW System Identification Toolkit to access the following manuals and help files.

- LabVIEW System Identification Toolkit Help—Use this help file to learn more about the system identification process and the System Identification VIs. Access this help file by launching LabVIEW and selecting **Help**»**Search the LabVIEW Help**.
- Getting Started with LabVIEW—Use this manual as a tutorial to familiarize yourself with the LabVIEW graphical programming environment and the basic LabVIEW features you use to build data acquisition and instrument control applications. Access this manual by navigating to the labview\manuals directory and double-clicking LV\_Getting\_Started.pdf.
- LabVIEW Fundamentals—Use this manual to learn about LabVIEW programming concepts, techniques, features, VIs, and functions you can use to create test and measurement, data acquisition, instrument control, datalogging, measurement analysis, and report generation applications. Access this manual by navigating to the labview\manuals directory and double-clicking LV\_Fundamentals.pdf.

Previous: Getting Started with the System Identification Assistant Next: Before You Begin

## **Before You Begin**

#### **System Requirements**

The LabVIEW System Identification Assistant requires a minimum of 128 MB of RAM and a Pentium III or later or Celeron 600 MHz or equivalent processor. National Instruments recommends 256 MB of RAM and a Pentium 4 or equivalent processor.

You can install the System Identification Assistant without installing LabVIEW on the host computer. However, to <u>convert projects created</u> <u>with the System Identification Assistant into LabVIEW block diagrams</u>, you must install and be a properly licensed user of the LabVIEW 8.6 Full or Professional Development System.

Refer to <u>Activating Your Software with the NI License Manager</u> for information about activating this software.

#### **Printing the Help File**

You can print this help file if you want to refer to a printed copy while completing the exercises. Complete the following steps to print this help file.

- 1. Click the **Contents** tab of the *LabVIEW SignalExpress Help* to display the contents.
- 2. Expand the **Getting Started** book.
- 3. Right-click the **Getting Started with the System Identification Assistant** book and select **Print** from the shortcut menu to open the **Print Topics** dialog box.
- 4. Select Print the selected heading and all subtopics.
- 5. Click the **OK** button.

Previous: System Identification Assistant Related Documentation Next: 1. Launching the System Identification Assistant

# **1. Launching the System Identification Assistant**

You can use the LabVIEW System Identification Assistant to create dynamic models of system data. In the following exercises, you create a dynamic model that relates the voltage of a DC motor to its motor speed.

Complete the following steps to launch the System Identification Assistant in the default configuration if you have not already done so.

1. Launch the System Identification Assistant by selecting Start»All Programs»National Instruments»LabVIEW System Identification Toolkit»LabVIEW System Identification Assistant. The System Identification Assistant launches within the LabVIEW SignalExpress environment.

You also can launch the System Identification Assistant from Windows Explorer. Browse to National Instruments\MAX\Assistants\SysId Assistant and double-click **SystemIdAssistant.exe**.

To launch the System Identification Assistant from LabVIEW, select **Tools**»**Control Design and Simulation**»**Launch System Identification Assistant**.

- 2. The **Welcome** dialog box and the LabVIEW SignalExpress environment open in the default configuration.
  - Note You can place a checkmark in the Show At Startup checkbox in the Welcome dialog box to specify whether the Welcome dialog box appears when you launch the System Identification Assistant. You can display this dialog box at any time by selecting Help»Show Welcome Dialog.

The Project View is the blank white rectangle on the left side of the LabVIEW SignalExpress environment. The Data View is the green coordinate grid on the black background in the middle of the LabVIEW SignalExpress environment. The context help window is the window on the right side of the LabVIEW SignalExpress environment that displays the How do I get help? and Context Help topics.

3. Click the **Close** button to close the **Welcome** dialog box.

- 4. If the graph does not appear in the Data View, select **View»Data View** to display the Data View.
- 5. If the context help window does not appear, select **Help\*Context Help** to display the context help window.

Previous: Before You Begin

Next: 2. Loading Files

# 2. Loading Files

You can use the LabVIEW System Identification Assistant to load measurement data files (.lvm). A measurement data file is a tab-delimited text file you can open with a spreadsheet or text-editing application. A measurement data file includes information about the data, such as the date and time the data was generated. You also can use the System Identification Assistant to load ASCII, SPICE, PSpice, or Multisim files.

Complete the following steps to load a measurement data file containing sample input and output signals. This data file contains signals acquired from a DC motor.

1. Click the **Add Step** button, shown below, to open the <u>Add Step</u> menu.



A step is a configurable function that acquires, generates, analyzes, loads, or stores signals. Steps process input signals and produce output signals. You can configure the operation of steps by specifying values in the <u>Configuration View</u>, or **Step Setup** tab, for each step. You create projects by adding steps to the <u>Project View</u>.

- 2. Select Load/Save Signals»Load from LVM to add a Load from LVM step to the Project View. The Configuration View for the Load from LVM step appears.
  - Tip When you move the cursor over a step in the Add Step menu, a description of that step appears in the help window at the bottom of the Add Step menu.
- 3. In the Configuration View, click the **Browse** button, shown below, next to the **Import file path** text box. Navigate to the National Instruments/Express Workbench/System Identification Assistant/Examples/getting\_started directory.



4. Select dc\_motor\_data.lvm and click the **OK** button to load this file. Notice that this file contains two signals, **Motor Input** and **Motor**  Output, which appear in the Signals in file table.

5. Verify that the **Import** checkbox next to **Motor Input** has a checkmark. This checkbox configures the System Identification Assistant to load the **Motor Input** signal from this measurement data file. This signal is a square periodic function of the voltage. You will use this stimulus data to estimate and validate the dynamic model of the motor.

Notice that the signal you load appears on the **Imported Signal** graph in the Configuration View. Also notice that information about the data, including the date and time the data was generated, appears on the **ID** tab in the **File and Signal Selection** section. The other tabs in this section contain additional information about the data.

You can create a probe to obtain information about a signal. In the Project View, expand the **dc\_motor\_data.lvm** item, right-click the **Motor Input** signal, and select **Probe** from the shortcut menu to create the probe. Notice that the probe displays information about the signal, including which step you used to generate the signal.

- 6. Close the probe.
- 7. In the Project View, expand the **dc\_motor\_data.lvm** item, rightclick the **Motor Input** signal, and select **Rename** from the shortcut menu to rename the signal. Change the name of this signal to Raw Stimulus Data.
  - $\overline{\mathbb{N}}$

**Note** In the Project View, a blue arrow indicates an output of a step. A red arrow indicates an input of a step. The blue and red arrows are shown below.

▶ step output

🕨 step input

8. In the Configuration View, place a checkmark in the **Import** checkbox next to **Motor Output** to configure the System Identification Assistant to load the **Motor Output** signal from this measurement data file. This signal is the speed of the motor over time and is dependent on the voltage data points in **Raw Stimulus Data**. You will use this data to estimate and validate the model for the motor.

Notice that this signal appears to be periodic except for sharp spikes near sample times 1.0 and 2.5. These spikes indicate bad or missing data caused by sensor failures. You will learn how to rebuild this data in the <u>Rebuilding Missing Data</u> exercise.

- 9. In the Project View, rename the **Motor Output** signal to Raw Response Data.
- 10. Select **File**»**Save Project As** and save this project as Sample Project.seproj to an easily accessible location. A .seproj file is a workbench project, which refers to the LabVIEW SignalExpress framework in which the System Identification Assistant resides.

Previous: 1. Launching the System Identification Assistant Next: 3. Rebuilding Missing Data

# **3. Rebuilding Missing Data**

You can use the LabVIEW System Identification Assistant to rebuild bad or missing stimulus and response data sets by defining the expected range of the measurement.

In the Loading Files exercise, you loaded two signals from a measurement data file. Spikes in the signals indicate missing data. When sensor failures occur, the measuring instrument records the voltage or motor speed as the default maximum value of the instrument. These default values are out of the range of the measuring instrument and indicate that the measurement for that particular sample time is missing. Complete the following steps to rebuild the missing data points in the signals.

1. Right-click the empty white area of the <u>Project View</u> and select **System Identification**»Preprocessing»Rebuild Missing Data from the shortcut menu to add a <u>Rebuild Missing Data</u> step.

Notice that the **Step Setup** tab changes from displaying the <u>Configuration View</u> of the Load from LVM step to displaying the Configuration View of the Rebuild Missing Data step.

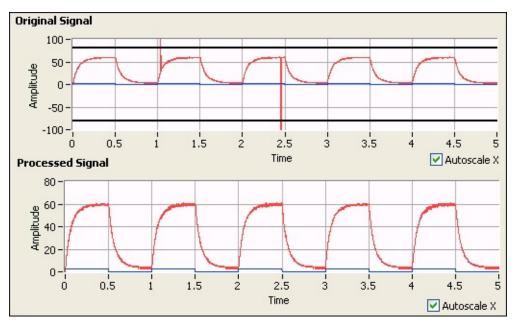
- 2. On the **Input Signals** tab in the Configuration View, verify that **Raw Stimulus Data** is selected in the **Stimulus Signal** pull-down menu. This pull-down menu specifies the stimulus signal you want to rebuild.
- 3. Verify that **Raw Response Data** is selected in the **Response Signal** pull-down menu. This pull-down menu specifies the response signal you want to rebuild.
- 4. On the **Configuration** tab in the Configuration View, place checkmarks in the **Specify upper limit** and **Specify lower limit** checkboxes.
- 5. Set **Specify upper limit** to 80. The Rebuild Missing Data step rebuilds all data with values above the upper limit.
- 6. Set **Specify lower limit** to -80. The Rebuild Missing Data step rebuilds all data with values below the lower limit.

Notice that horizontal lines representing the upper and lower limits appear on the **Original Signal** graph. You can drag these

lines up and down to set the upper and lower limits visually.

7. Compare the Original Signal graph with the Processed Signal graph. The Original Signal graph displays the input signals for this step. The Processed Signal graph displays the rebuilt signals. Preprocessed Input 1 in the Project View is the rebuilt signal corresponding to Raw Stimulus Data in the Configuration View. Preprocessed Input in the Project View is the rebuilt signal corresponding to Raw Response Data in the Configuration View.

In the **Processed Signal** graph, notice that the rebuilt signal corresponding to **Raw Response Data** does not contain the spikes near sample times 1.0 and 2.5, as shown in the following figure.



- 8. In the Project View, rename the **Preprocessed Input 1** signal to Rebuilt Stimulus.
  - **Tip** When you move the cursor over a renamed input or output signal in the Project View, a dialog bubble appears with both the new name and the original name of the signal. You also can use this bubble to view the entire name of the signal if the width of the Project View is too small to view the full name.
- 9. Rename the **Preprocessed Input** signal to Rebuilt Response.

10. Select File»Save Project to save the project.

Previous: 2. Loading Files

Next: 4. Splitting Data

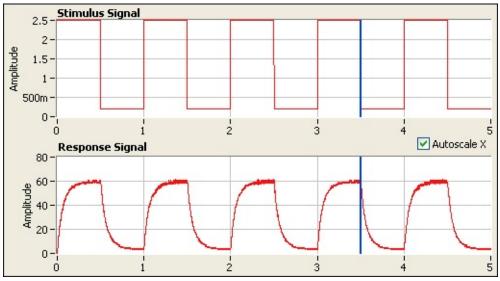
# 4. Splitting Data

You can use the LabVIEW System Identification Assistant to split signals into two data sets. You can use one data set for model estimation and the other data set for model validation.

Complete the following steps to split the rebuilt signals into two data sets.

- Right-click the empty white area of the <u>Project View</u> and select System Identification»Preprocessing»Split Data from the shortcut menu to add a <u>Split Data</u> step.
- 2. On the **Configuration** tab in the <u>Configuration View</u>, verify that **Rebuilt Stimulus** is selected in the **Stimulus Signal** pull-down menu. This pull-down menu specifies the stimulus signal you want to split.
- 3. Verify that **Rebuilt Response** is selected in the **Response Signal** pull-down menu. This pull-down menu specifies the response signal you want to split.
- 4. Set **Split Position** to 70.00% to split **Rebuilt Stimulus** and **Rebuilt Response** at 70% of their length. You will use the first 70% of each signal to estimate the model for the motor and the remaining 30% of each signal to validate the model.

Notice that the blue vertical line on the **Stimulus Signal** and **Response Signal** graphs moves when you set the **Split Position**. This line visually displays the position where the signals split, as shown in the following figure.



Note You also can set the location of the signal split by dragging the blue line left or right inside the **Stimulus Signal** or **Response Signal** graphs.

Notice that if you move the blue line in one graph, the line moves to the same position on the x-axis in the other graph. The number of samples in corresponding portions of the stimulus and response signals must be equal.

- 5. Restore **Split Position** to 70.00%.
- 6. In the Project View, rename the **Rebuilt Stimulus Estimation Data** signal to Estimation Stimulus.
- 7. Rename the **Rebuilt Stimulus Validation Data** signal to Validation Stimulus.
- 8. Rename the **Rebuilt Response Estimation Data** signal to Estimation Response.
- 9. Rename the **Rebuilt Response Validation Data** signal to Validation Response.
- 10. Select File»Save Project to save the project.

Previous: 3. Rebuilding Missing Next: 5. Normalizing Estimation
Data
Data

## **5. Normalizing Estimation Data**

You can use the LabVIEW System Identification Assistant to normalize signals to set their mean and standard deviation to 0 and 1 respectively. Normalizing signals makes calculations more manageable when estimating models, particularly when a small change in the stimulus can result in a large change in the response.

Complete the following steps to normalize the data you want to use for model estimation.

- 1. From the LabVIEW SignalExpress menu bar, select Add Step»System Identification»Preprocessing»Normalize Data to add a <u>Normalize Data</u> step to the <u>Project View</u>.
- 2. On the **Input Signals** tab in the <u>Configuration View</u>, verify that **Estimation Stimulus** is selected in the **Stimulus signal** pull-down menu. This pull-down menu specifies the stimulus signal you want to normalize.
- 3. Select **Estimation Response** from the **Response signal** pulldown menu to specify **Estimation Response** as the response signal you want to normalize.
- 4. On the Mean Source Configuration tab, verify that the Auto calculated option is selected in both the Stimulus mean source and Response mean source sections. When you select Auto calculated, the System Identification Assistant automatically calculates the mean of the stimulus or response signal and normalizes the signal using that mean value.
- 5. Similarly, on the **Standard Deviation Source Configuration** tab, verify that the **Auto calculated** option is selected in both the **Stimulus standard deviation source** and **Response standard deviation source** sections. When you select **Auto calculated**, the System Identification Assistant automatically calculates the standard deviation of the stimulus or response signal and normalizes the signal using that standard deviation value.
- 6. Observe the two graphs in the Configuration View. **Stimulus Signals** displays the original **Estimation Stimulus** signal and the same signal after normalization. **Response Signals** displays the original **Estimation Response** signal and the same signal after normalization.

You can see from the graphs that the mean of both normalized signals is approximately 0 and their standard deviation is approximately 1.

- In the Project View, rename the Normalized Estimation Stimulus output to Normalized Stimulus. You can use Normalized Stimulus as the stimulus signal to estimate the model of the motor.
- 8. Rename the **Normalized Estimation Response** output to Normalized Response. You can use **Normalized Response** as the response signal to estimate the model of the motor.
- 9. Select File»Save Project to save the project.

Previous: 4. Splitting Data

Next: 6. Normalizing Validation Data

#### 6. Normalizing Validation Data

In the <u>Splitting Data</u> exercise, you created two sets of stimulus and response data. You can use one set, consisting of <u>Estimation Stimulus</u> and <u>Estimation Response</u>, to estimate the model for the DC motor. You can use the other set, consisting of <u>Validation Stimulus</u> and <u>Validation</u> <u>Response</u>, to validate the model you estimate. If you normalize the data set you use for model estimation, normalize the data set you use for model validation as well.

Complete the following steps to normalize the data you want to use for model validation.

- 1. Press the <Ctrl> key and click and drag the <u>Normalize Data</u> step to the empty white area of the <u>Project View</u> to create a copy of the Normalize Data step.
- In the Project View, click the blue down arrow on the **Normalized Stimulus** input of the new Normalize Data step and select **Validation Stimulus** to specify **Validation Stimulus** as the stimulus signal you want to normalize.

Notice that when you click the blue down arrow, you can choose from a list of ten available inputs. These available inputs are the output signals from the steps above this Normalize Data step in the Project View.

- 3. Click the blue down arrow on the **Normalized Response** input and select **Validation Response** to specify **Validation Response** as the response signal you want to normalize.
- 4. On the Mean Source Configuration tab in the Configuration View, select the Value from another block option in the Stimulus mean source section and verify that Stimulus Mean is selected in the Value from another block pull-down menu. Selecting this option to normalize the stimulus signal ensures that this step uses the same mean for both the estimation data and validation data.
- 5. Select the Value from another block option in the Response mean source section and verify that Response Mean is selected in the Value from another block pull-down menu. Selecting this option to normalize the response signal ensures

that this step uses the same mean for both the estimation data and validation data.

- 6. On the Standard Deviation Source Configuration tab, select the Value from another block option in the Stimulus standard deviation source section and verify that Stimulus Standard Deviation is selected in the Value from another block pull-down menu. Selecting the Stimulus Standard Deviation and Response Standard Deviation to normalize the stimulus and response signals, respectively, ensures that this step uses the same standard deviation for both the estimation data and validation data.
- 7. Select the Value from another block option in the Response standard deviation source section and verify that Response Standard Deviation is selected in the Value from another block pull-down menu.
  - Note You want the System Identification Assistant to use the means and standard deviations of the estimation data to normalize the validation data so that the two sets of data have the same range. Because the System Identification Assistant does not use the actual mean and standard deviation of the validation data to normalize the signals, the normalized signals corresponding to Validation Stimulus and Validation Response may not have a mean of exactly 0 or a standard deviation of exactly 1.
- 8. In the Project View, rename the **Normalized Stimulus 1** output to Test Stimulus. You can use **Test Stimulus** as the stimulus signal to validate the model you estimate.
- 9. Rename the **Normalized Response 1** output to Test Response. You can use **Test Response** as the response signal to validate the model you estimate.
- 10. Select File»Save Project to save the project.

Previous: 5. Normalizing Estimation Data Next: 7. Estimating A Model

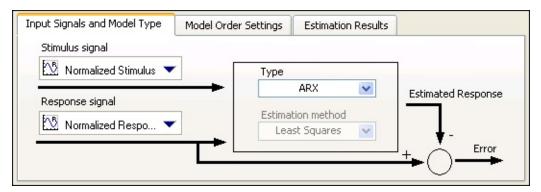
# 7. Estimating a Model

You can use the LabVIEW System Identification Assistant to estimate the parameters of a model for an unknown system. Models enable you to predict the response of a system given only the stimulus.

In the previous exercises, you processed data to make calculations during model estimation and model validation easier. You now can use the processed data to estimate a model for the motor. A model of the motor can estimate the motor speed given a stimulus voltage. The more accurate the model is, the closer the estimated motor speed is to the actual response of the system.

Complete the following steps to estimate the parameters of an ARX model for the motor.

- From the LabVIEW SignalExpress menu bar, select Add Step»System Identification»Model Estimation»Parametric Estimation to add a <u>Parametric Estimation</u> step to the <u>Project</u> <u>View</u>.
- 2. On the **Input Signals and Model Type** tab in the <u>Configuration</u> <u>View</u>, select **Normalized Stimulus** from the **Stimulus signal** pull-down menu to specify **Normalized Stimulus** as the input stimulus signal to use to estimate the model.
  - Note If the System Identification Assistant displays an error message stating The signal length is too small for computation, or the stimulus and response signals are not of the same length., you can ignore this error. This error message displays because the Stimulus signal and Response signal, which you specify in the next step, have different lengths.
- 3. Select **Normalized Response** from the **Response signal** pulldown menu to specify **Normalized Response** as the input response signal to use to estimate the model.
- 4. Verify that ARX is the model type in the Type pull-down menu. This pull-down menu specifies the model type whose parameters you want to estimate by using the selected input signals. The Input Signals and Model Type tab contains a diagram of the model you want to build, as shown in the figure below.



- 5. On the Model Order Settings tab, set both A Order and B order to 2 to set up a second-order model. Setting A Order to 2 configures the System Identification Assistant to determine the value of the response signal based on the two previous values of that signal. Setting B order to 2 configures the System Identification Assistant to determine the value of the stimulus signal based on the two previous values of that signal.
- 6. Verify that 1 is the delay in the **delay** option. A delay of 1 applies a delay of one sample time to the stimulus signal. The following figure shows the correct configuration of the **Model Order Settings** tab.

Model Type	Model Order Set	tings Estima	ation Results		
or orders					
B order	F order	C order	D order	delay	
2	1	1	1	1 🗘	
		or orders Border Forder	or orders	or orders	or orders Border Forder Corder Dorder delay

7. Notice the two graphs in the Configuration View. Both graphs illustrate the accuracy of the model estimation. The **Response Signals** graph displays the original response to the **Normalized Stimulus** signal and the estimated response of the configured ARX model to the **Normalized Stimulus** signal. Recall that **Normalized Response** is the original response to **Normalized Stimulus**. The more the two signals in the **Response Signals** graph overlap, the more accurate the model estimation is. In this example, the signals overlap very closely, which means the estimated model is a close approximation to the system.

Similarly, the **Simulation Error** graph displays the difference between the two signals shown in the **Response Signals** graph. The closer the values of the **Simulation Error** graph are to 0, the more accurate the model estimation is. In this example, the **Simulation Error** is very close to 0, which means the estimated model is a close approximation to the system. Notice, however, that the value of **Simulation Error** asymptotically goes to 0. The model uses data from previous sample times to compute its estimated response. Therefore, at the initial stage of the response, the model does not have sufficient previous data for estimation, and the **Simulation Error** is significantly large.

- 8. On the **Estimation Results** tab, notice that the small values of the **FPE**, **AIC**, and **MDL** also illustrate the accuracy of the model estimation. When these values are close to 0, the estimated model is a close approximation of the system.
- 9. In the Project View, rename the **SysID Model** output to ARX Model.
- 10. Select File»Save Project to save the project.

Previous: 6. Normalizing Validation Data Next: 8. Viewing Model Information

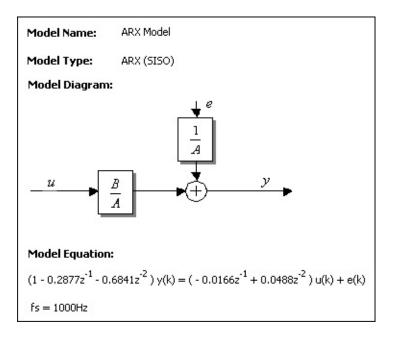
## 8. Viewing Model Information

You can use the LabVIEW System Identification Assistant to display model information in the <u>Data View</u>.

Complete the following steps to view information about the ARX model for the motor.

- 1. Click the **Data View** tab to display the Data View.
- 2. In the <u>Project View</u>, click the **ARX Model** output of the <u>Parametric</u> <u>Estimation</u> step and drag it to the Data View. The Data View now displays information about the **ARX Model**, including its name, type, and structure, as shown in the figure below.

Notice that the model coefficients based on the estimation data appear in the equations in the Data View. Also notice that the orders of the polynomials match the orders you selected when estimating the model.



N

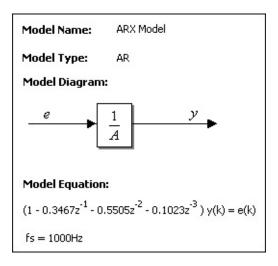
Note You also can display signals and scalar values in the Data View by clicking them in the Project View and dragging them to the Data View. To remove a display from the Data View, right-click the display and select Remove Display or Data View»Remove Display from the shortcut menu.

- 3. Click the **Step Setup** tab to display the <u>Configuration View</u> for the Parametric Estimation step
- 4. On the Model Order Settings tab, set A Order to 3.

Notice in the **Data View** tab that the polynomial order and its coefficients changed. The changes you make in the Configuration View automatically update the information in the Data View.

5. On the **Input Signals and Model Type** tab, select **AR** from the **Type** pull-down menu. In the **Data View** tab, **Model Type** automatically changes from **ARX** to **AR**. The model structure and coefficients also change, as shown in the figure below.

Notice that the **Model Name** no longer matches the **Model Type**. If you change the **Model Type**, be sure to rename the output model accordingly.



Note You can right-click the model display in the Data View and select **Model Diagram**»Never Show from the shortcut menu to hide the **Model Equation**.

You also can right-click the model display in the Data View and select **Show**»**Transfer Function** to display the transfer function of the model, or select **Show**»**Zeros and Poles** to display the poles and zeros of the model.

6. In the Configuration View on the **Step Setup** tab, restore **Type** to **ARX** and **A Order** to 2.

7. Select File»Save Project to save the project.

Previous: 7. Estimating a Model <u>Next: 9. Performing a Pole-Zero</u> <u>Analysis</u>

## 9. Performing a Pole-Zero Analysis

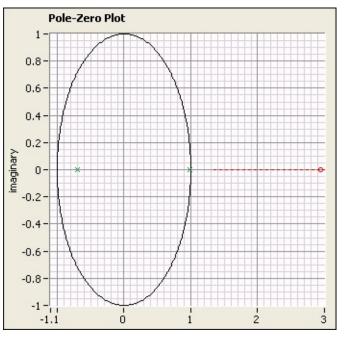
You can use the LabVIEW System Identification Assistant to compute the poles and zeros of the transfer function for one input-output pair of a model. The <u>Pole-Zero Analysis</u> step computes the pole-zero plot for a model.

Complete the following steps to perform a pole-zero analysis of the ARX model.

- In the <u>Project View</u>, right-click the **Parametric Estimation** step and select **Insert After»System Identification»Model Analysis»Pole-Zero Analysis** from the shortcut menu to add a Pole-Zero Analysis step after the <u>Parametric Estimation</u> step.
- On the Input Model tab in the <u>Configuration View</u>, verify that ARX Model is selected in the Model pull-down menu. ARX Model is the only model output from previous steps and therefore appears by default in this menu.
- 3. The **Pole-Zero Plot** displays the poles and zeros of the ARX model. Notice that all the poles of the plot are inside the unit circle, which indicates that the system is stable. A stable system reaches a steady-state value.

Also notice that the poles are on the real axis, which illustrates the monotonic nature of the signal. Poles with imaginary components indicate an oscillating signal.

4. You can set the minimum and maximum values of the x- and yscales of the plot to resize the plot. Click the maximum value on the x-axis and change the value to 3. Notice the zero on the **Pole-Zero Plot**, as shown in the following figure.



5. Select File»Save Project to save the project.

Previous: 8. Viewing Model Information Next: 10. Validating the ARX Model

## **10. Validating the ARX Model**

You can use the LabVIEW System Identification Assistant to perform a kstep-ahead prediction using sample stimulus and response signals. You can validate a model using the data from the k-step-ahead prediction. Model validation applies data from the original stimulus signal to the model to produce a response and compares this predicted response with the original response signal. The closer the predicted response is to the original response, the more accurate the model is.

Complete the following steps to validate the ARX model for the motor.

- In the <u>Project View</u>, right-click the Pole-Zero Analysis step and select Insert After»System Identification»Model Analysis»Model Validation from the shortcut menu to add a <u>Model Validation</u> step after the <u>Pole-Zero Analysis</u> step.
- 2. On the **Input Signals** tab in the <u>Configuration View</u>, verify that **ARX Model** is selected in the **Model** pull-down menu.
- 3. Select **Test Stimulus** from the **Stimulus Signal** pull-down menu to specify **Test Stimulus** as the stimulus signal to use to perform the k-step-ahead prediction.
- 4. Select **Test Response** from the **Response Signal** pull-down menu to specify **Test Response** as the response signal to use to perform the k-step-ahead prediction.

Recall that **Normalized Stimulus** and **Normalized Response** are the signals you used to estimate the ARX model. Now you use **Test Stimulus** and **Test Response**, which are smaller sets of data samples from the same motor, to validate the accuracy of the model.

- 5. Notice that the mean square error (**MSE**) is **0.00075**. On the **Settings** tab, notice that **Prediction Step** is set to 1. This model therefore has a very small error with a small prediction step.
- 6. Set **Prediction Step** to 10. Notice that **MSE** is now **0.00634**. The **MSE** is larger with a larger prediction step. However, the error is less than 1% so you can conclude that the estimated model is a close approximation of the system.
- 7. In the Project View, rename the **SysID Model Prediction** output to ARX Model Prediction. This output signal is the predicted

response from the ARX model to the Test Stimulus stimulus signal.

Notice the two graphs in the Configuration View. Both illustrate the accuracy of the ARX model. The **Response Signals** graph displays Test Response and ARX Model Prediction. Recall that Test Response is the original response to Test Stimulus, whereas **ARX Model Prediction** is the estimated response of the configured ARX model to the **Test Stimulus** signal. The more the two signals in the **Response Signals** graph overlap, the more accurate the model is. In this example, the signals overlap very closely, which means the model is a close approximation of the system.

Similarly, the **Error Signal** graph displays the difference between the two signals shown in the **Response Signals** graph. The closer the values of the Error Signal graph are to 0, the more accurate the model is.

8. Select File»Save Project to save the project.

Zero Analysis

Previous: 9. Performing a Pole- Next: 11. Estimating a Model for **Comparison** 

# **11. Estimating a Model for Comparison**

You estimated the parameters of and validated an ARX model for the DC motor. You also can estimate and validate other models with the same input and validation signals to determine if a different model provides a more accurate estimation for the system.

Complete the following steps to estimate the parameters of an ARMAX model using the sample signals.

- 1. Click the Add Step button to open the Add Step menu.
- 2. Select System Identification»Model Estimation»Parametric Estimation to add a <u>Parametric Estimation</u> step to the <u>Project</u> <u>View</u>. You want to add a new Parametric Estimation step so you can keep **ARX Model** to compare with the ARMAX model.
- 3. On the **Input Signals and Model Type** tab in the <u>Configuration</u> <u>View</u>, select **Normalized Stimulus** from the **Stimulus signal** pull-down menu to specify **Normalized Stimulus** as the input stimulus signal for this Parametric Estimation step.
  - Note If the System Identification Assistant displays an error message stating The signal length is too small for computation, or the stimulus and response signals are not of the same length., you can ignore this error. This error message displays because the Stimulus signal and Response signal, which you specify in the next step, have different lengths.
- 4. Select **Normalized Response** from the **Response signal** pulldown menu to specify **Normalized Response** as the input response signal for this Parametric Estimation step.
- 5. Select **ARMAX** from the **Type** pull-down menu.
- 6. On the **Model Order Settings** tab, set both **A Order** and **B order** to 2 to set up a second-order model.
- Verify that C order is set to 1. An ARMAX model incorporates the effect of previous prediction errors in the current estimation. The C coefficient adjusts the effect of the previous prediction signals. Setting C order to 1 configures the simplest case.
- 8. Set **delay** to 1. Now the ARMAX model has the same input signals, orders, and delay as the ARX model.

- 9. In the Project View, rename the **SysID Model** output to ARMAX Model.
- 10. Select File»Save Project to save the project.

Previous: 10. Validating the ARX Model Next: 12. Validating the New Model

## **12.** Validating the New Model

Now that you have the parameters for an ARMAX model, you can validate the model to determine if the model type provides an accurate estimation of the system.

Complete the following steps to validate the ARMAX model.

- 1. Click the Add Step button to open the Add Step menu.
- 2. Select System Identification»Model Analysis»Model Validation to add a Model Validation step to the Project View.
- 3. On the **Input Signals** tab in the <u>Configuration View</u>, verify that **ARMAX Model** is selected in the **Model** pull-down menu.
- 4. Select **Test Stimulus** from the **Stimulus Signal** pull-down menu to specify **Test Stimulus** as the stimulus signal to use to perform the k-step-ahead prediction.
- 5. Select **Test Response** from the **Response Signal** pull-down menu to specify **Test Response** as the response signal to use to perform the k-step-ahead prediction.
- 6. Notice that the mean square error (**MSE**) is **0.00084**. On the **Settings** tab, notice that **Prediction Step** is set to 1. When the prediction step is small, so is the error.

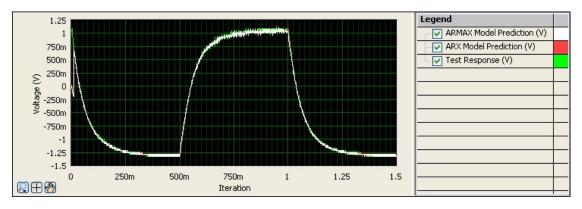
Recall that the **MSE** of the ARX model with one prediction step was **0.00075**. Because the **MSE** of the ARX model is smaller than the **MSE** of the ARMAX model, the ARX model is a slightly more accurate estimation of the system. Nevertheless, both models have very small mean square errors, and therefore they are both accurate models.

- 7. Set **Prediction Step** to 10. Notice that **MSE** is now **0.00644**. When the prediction step is large, so is the **MSE**. However, as with the ARX model, the error is less than 1%. You can conclude that the model estimation with a **Prediction Step** value of 10 is accurate but not as accurate as the model estimation with a **Prediction Step** value of 1.
- 8. In the Project View, rename the **SysID Model Prediction** output to ARMAX Model Prediction. This output signal is the predicted response from the ARMAX model to the **Test Stimulus** signal.
- 9. You visually can compare **ARX Model Prediction** and **ARMAX**

**Model Prediction** by displaying the two signals in the <u>Data View</u>. Click the **Data View** tab to display the Data View.

- 10. In the Project View, click the **ARMAX Model Prediction** output of this Model Validation step and drag it to the Data View.
- 11. In the Project View, click the **ARX Model Prediction** output of the first Model Validation step and drag it to the same graph on the Data View.
- 12. In the Project View, click the **Test Response** input of this Model Validation step and drag it to the same graph on the Data View.
- 13. In the Data View, right-click the graph and select **Visible Items»Legend** to display the graph legend. The graph legend lists every signal displayed in the graph and the corresponding plot color.

Notice that all three signals overlap very closely, as shown below, illustrating the accuracy of both the ARX and ARMAX models.



Previous: 11. Estimating a Model for Comparison

Next: 13. Saving the ARMAX Model

# **13. Saving the ARMAX Model**

You can use the LabVIEW System Identification Assistant to save models to a file.

Complete the following steps to save the ARMAX model.

- 1. Click the **Add Step** button to open the <u>Add Step</u> menu.
- 2. Select System Identification»Import-Export Model»Save System Identification Model to add a <u>Save System</u> Identification Model step to the <u>Project View</u>.
- 3. In the <u>Configuration View</u>, verify that **ARMAX Model** is selected in the **Model** pull-down menu.
- 4. Click the **Browse** button next to the **File Path** text box and navigate to the directory in which you want the System Identification Assistant to save the ARMAX model.
- 5. Enter Sample ARMAX Model in the **File name** text box and click the **OK** button. The System Identification Assistant saves the ARMAX Model as Sample ARMAX Model.sim in this directory only when you run this project. A .sim file is a system identification model file that saves all model coefficients regardless of the model type.
- Run the project once by pressing the <Ctrl-Shift-R> keys or by clicking the down arrow next to the Run button, shown below, on the LabVIEW SignalExpress toolbar and selecting Run Once from the shortcut menu.

🔁 Run 🔻

The steps in a project do not execute until you run the project.

- 7. Verify that Sample ARMAX Model.sim now is in the directory that you specified. You now have a saved, accurate model of the DC motor. You can use the Load System Identification Model step to load this model to a project.
- 8. Select File»Save Project to save the project.

Previous: 12. Validating	Next: Extending System Identification
<u>the New Model</u>	Assistant Projects with LabVIEW

#### Extending System Identification Assistant Projects with LabVIEW

In this help file, you learned how to complete the following procedures:

- Launch the LabVIEW System Identification Assistant in the LabVIEW SignalExpress environment
- Load signals and rebuild missing data
- Split data to use for model estimation and validation
- <u>Normalize data</u> for ease of calculation
- Estimate parameters of a system model
- View model information in the SignalExpress environment
- <u>Compute the pole-zero plot</u> for a model
- <u>Validate</u> the accuracy of a model
- <u>Compare system models</u> through <u>model validation</u>
- Save models to file

#### **Converting to Block Diagrams**

Without prior knowledge about programming in LabVIEW, you can use the System Identification Assistant to develop a model that reflects the behavior of a certain dynamic system. The LabVIEW System Identification Toolkit also provides LabVIEW VIs that you can use to preprocess raw data from a dynamic system and develop a model that reflects the behavior of that system.

You can use LabVIEW to <u>convert System Identification Assistant projects</u> to LabVIEW block diagrams to continue developing an application. You first can use the interactive tools in the System Identification Assistant to analyze and estimate models. You then can convert the sequence into a LabVIEW block diagram to build automated test programs.

The System Identification Assistant converts projects into block diagrams using the VIs that ship with the System Identification Toolkit. The <u>System</u> <u>Identification steps</u> in the <u>Add Step</u> menu do not convert into Express VIs.

Unlike creating a project with the System Identification Assistant, creating a LabVIEW application using the System Identification VIs requires basic knowledge about programming in LabVIEW. Refer to the *LabVIEW Fundamentals* manual and the *Getting Started with LabVIEW* manual for more information about the LabVIEW programming environment. In LabVIEW, access these manuals by clicking the corresponding links in the **Getting Started** window. Refer to the *LabVIEW Help* for more information about LabVIEW VIs and functions. In LabVIEW, access this help file by selecting **Help**.

#### Sweeping

The <u>Sweep</u> step repeatedly performs measurements with different parameter values. The System Identification Assistant converts the Sweep step into a For Loop in a LabVIEW block diagram. The System Identification steps in the **Add Step** menu can exist within a sweep, but they do not provide parameters that you can use within a sweep.

#### **Using User-Defined Steps**

You can <u>build a user-defined VI</u> to use in a <u>Run LabVIEW VI</u> step. The Run LabVIEW VI step runs a VI in the LabVIEW SignalExpress environment. When you import a LabVIEW VI, the System Identification Assistant maps the inputs of the VI as parameters and the outputs of the VI as outputs by default.

#### **Further Examples**

The System Identification Assistant provides several other example projects that demonstrate more capabilities of the System Identification Assistant. These projects are located in the National Instruments\Express Workbench\System Identification Assistant\Examples directory. You can modify an example project to fit an application, or you can copy and paste from one or more examples into a project that you create.

You can find the solution, Sample Project Solution.seproj, to the exercises in this help file, as well as an example of the ARMAX model you created, Sample ARMAX Model Solution.sim, in the National Instruments\Express Workbench\System Identification Assistant\Examples\getting\_started directory.

Previous: 13. Saving the ARMAX Model