System Identification Toolbox Release Notes

Note No significant new features have been introduced for Version 6.1.1 of the System Idenfications Toolbox. The product is essentially unchanged from Version 6.1.

The "System Identification Toolbox 6.1.1 Release Notes" on page 1-1 list major bug fixes since the previous release.

Upgrading from a Version Earlier than Version 6.0

If you are upgrading from a version earlier than Version 6.0, see

- "System Identification Toolbox 6.0 Release Notes" on page 2-1
- "System Identification Toolbox 5.0 Release Notes" on page 3-1

Printing the Release Notes

If you would like to print the Release Notes, you can link to a PDF version.

Contents

1 System Identification Toolbox 6.1.1 Release Notes

2 System Identification Toolbox 6.0 Release Notes

3 System Identification Toolbox 5.0 Release Notes

1

System Identification Toolbox 6.1.1 Release Notes

Major Bug Fixes

The System Identification Toolbox Version 6.1.1 includes important bug fixes made since Version 6.0. You can see a list of major 6.1.1 bug fixes on the MathWorks Web site.

If you are viewing these release notes in PDF form on the MathWorks Web site, please refer to the HTML form of the release notes on the MathWorks Web site and use the link provided.

2

System Identification Toolbox 6.0 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the System Identification Toolbox 6.0.

New Model Object

A new model object, `idproc`, contains simple continuous time process models. This object is characterized by static gain, possible dead time, and dominating time constant(s). A new graphical user interface (GUI) that supports this object is opened from the main Ident GUI.

To gain insight about this object, run the `iddemopr` demo, or try the command

```
m = pem(data,'p1d')
```

Estimation and Validation in the Frequency Domain

The System Identification Toolbox command-line API and the main Ident graphical user interface (GUI) now support estimation and validation using frequency domain data, including the following:

- Inputs and outputs, entered as frequency domain data in the `iddata` object
- Frequency response data, obtained from a frequency analyzer

Frequency response data should be packaged as an `frd` or `idfrd` object. All estimation, simulation, and validation routines accept frequency-domain data and frequency response data as inputs, similar to time-domain data.

With the `fft` and `ifft` commands, you can now transform between the time and frequency domains. You use the `spafdr` command to estimate frequency responses using frequency-dependent resolution.

See

```
help iddata
```

and

```
idprops data
```

for complete descriptions and `iddemofr` for demonstrations.

Storing Continuous-Time Data as Frequency Domain Objects

You can now store continuous-time data as frequency domain data objects. Continuous-time Fourier-transformed data is now stored at a finite number of arbitrary frequencies, enabling direct estimation of continuous-time models. See, for example,

```
help oe
```

Simulink Support of iddata and idmodel Objects

The System Identification Toolbox now provides direct Simulink support of `iddata` and `idmodel` objects.

The command `slident` opens a Simulink block library, which you can then use to simulate any `idmodel` (with or without noise). This library also contains data sources and sinks for `iddata` objects.

Advice Command

A new command `advice` can be applied to any `iddata` and `idmodel` objects provide you with helpful tips and information about the quality, problems and options of the data set or model. See

```
help iddata/advice
```

and

```
help idmodel/advice
```

for more information.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from the System Identification Toolbox 4.0.5 to the System Identification Toolbox 6.0.

Theta Models are no Longer Supported

Theta models (matrices) are no longer supported in the System Identification Toolbox 6.0. Existing code that uses functions such as `th2par` and `th2ss` to access the theta model data will continue to work in the System Identification Toolbox 6.0. However, if you have code that directly indexes into the theta matrix (e.g., `th(1,3)`), that code will no longer work.

3

System Identification Toolbox 5.0 Release Notes

New Features

This section introduces the new features and enhancements in the System Identification Toolbox 5.0.

Object-Based Design

Based on MATLAB object technology, the System Identification Toolbox 5.0 provides functions for creating objects that are directly associated with your models and data. You can run the following commands to see examples:

```
z = iddata(y,u,Ts);
sys = pem(z);
```

The new object-based commands in the System Identification Toolbox 5.0 make it easier to work with Control System Toolbox objects. For example, you can now

- Convert System Identification Toolbox objects to Control System Toolbox objects, and vice versa.
- Apply analysis tools to objects in both the System Identification Toolbox and the Control System Toolbox.

For an overview of the features included in this new object-based approach, type

```
help idhelp
```

You do not need to rewrite any code you wrote using an earlier version of the System Identification Toolbox; the earlier command-line syntax is still supported in the System Identification Toolbox 5.0.

Advanced Feature Enhancements

The System Identification Toolbox 5.0 includes the following enhancements:

- Free parameterization for state-space models is now supported. For example, you can now type

  ```
  m = pem(z,4)
  ```

  to obtain a fourth order state-space model with a well-conditioned parameterization.

- You can now add initial filter conditions, which yields better performance for slow dynamics. See the `'InitialState'` property of `idmodel` objects for more information.
- You can now use the `SearchDirection` and `Advanced` properties of `idmodel` objects to access several variants of iterative search algorithms. For more information, type

  ```
  idprops algorithm
  ```

- You can now focus the model approximation inherent in system identification to various frequency regions by using the `Focus` property. The values for the `Focus` property include `'Prediction'`, `'Simulation'`, or any idmodel or LTI object that uses the frequency weighting of that system.