ModelID User manual

Jørgen K. H. Knudsen

October 13, 2012

2-control Aps, Frimodtsvej 11, DK-2900 Hellerup, Denmark
(e-mail: JoeK@2-control.dk)

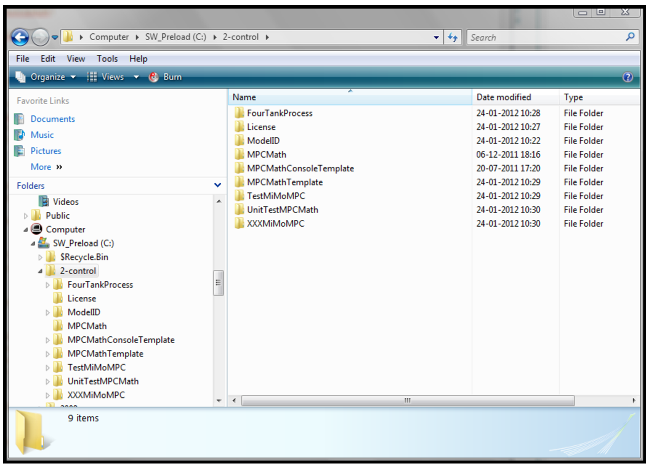
This paper describes how to get *ModelID* up and running on your machine. *ModelID* is still a beta program release, but this version demonstrates the basic ideas and functionalities. If you have comments or questions, please don't hesitate to contact me.

Installing *ModelID*

Unzip the file "2-control 2012-01-24.zip" to c:\creating the folder c:\2-control. (If the file you received has the extension .sip, rename it to .zip).

Copy the license file MPCMath00020.lic to c:\2-control\ License.

The c:\2-control folder should look something like this

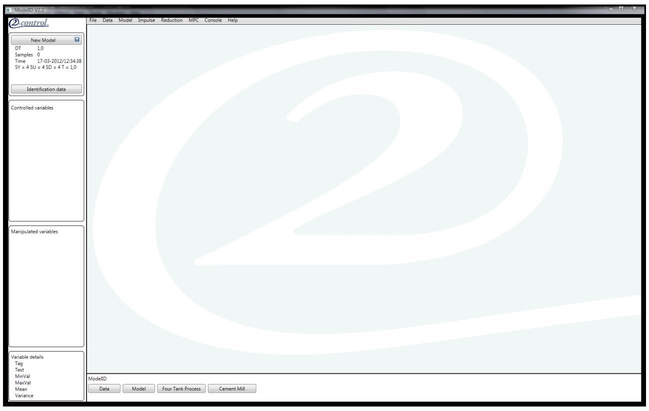


The c:\2-control folder contains *MPCMath* code and the *ModelID* installation files. Open directory ModelID and run setup.exe. Go to the start menu and select "all programs" , ModelID , and start *ModelID* . Eventually you can create a short cut to *ModelID* by right click \send to \Desktop.

Using *ModelID*

Data Tool

The start up picture is



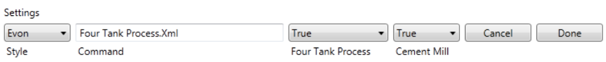
Select one of the following functions:

Data	Create a new model identification package.
Model	Open an existing identification package
Four Tank Process	Four Tank Process demo system
Cement Mill	Cement Mill demo system

The Four tank process and cement mill processes are included in order to have simulated data to demonstrate and test the functionality of *ModelID* .

Settings

Select Menu Data/Settings to show and change the *ModelID* settings.



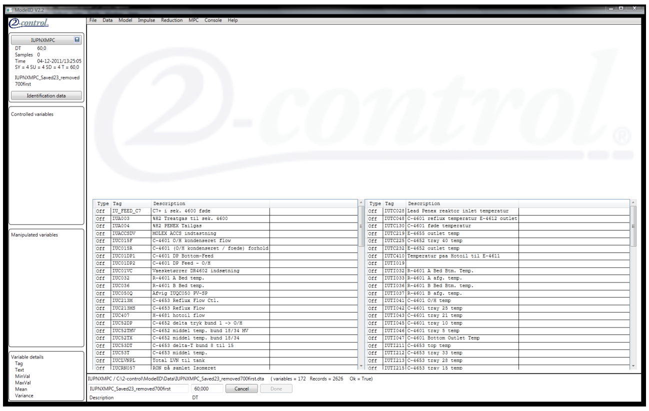
The Settings are

Style	2-control or evon style of *ModelID*
Command	Default data or model file for start up
	Useful when working on the same data for several *ModelID* sessions
Four Tank Process	Display or hide Four Tank Process demo system
Cement Mill	Display or hide Cement Mill demo system

Point Overview and Point selection

Presently ModelId supports three formats for input data. data files wit the extension .dta holds data as generated by Statoil's SEPTIC system, where the data file includes point tags and descriptions. Darafiles with extension .Xml holds data transferred from evon-automation' s [1] XAMControl system.

Data files with the extension *.csv holds pure data with one line per sampling point.The values are separated by ";" character. Presently the '.csv file format does not include any point definition information. It has to be input manually during point selection. The point survey below shows the defined points.



where

Description	Package description
DT	Sampling time for data

Clicking one of the lines with a point definition, brings us to the Point selection phase, where the data is plotted and a point selection menu is displayed in the commend field.

where

Type	Output for a plant Output variable variable
	Input for a plant input (A manipulated or a disturbance variable)
	Off if the variable is to be ignored
DT	Sampling time for data

[1] www.evon-automation.com

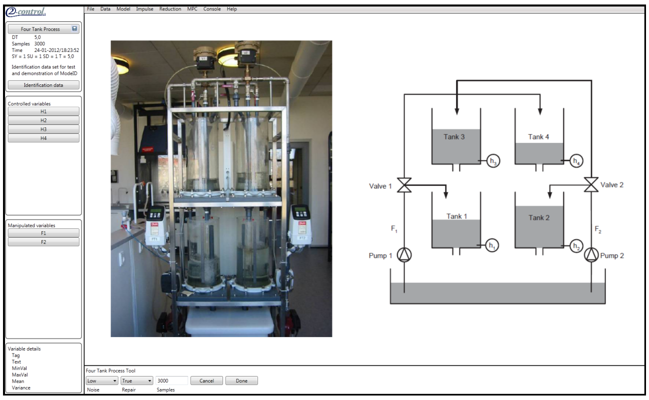
Examples of data.csv files are given in the directory c:\2-control\ ModelID\Data.

At any time during a *ModelID* session, the entered points and parameters parameters can be saved in a *ModelID* package by pressing NPCW17-OE-SO button in the upper left corner of the screen ot using the file menu for "Save as" option.

Model command

Select *ModelID* package saved with the NPCW17-OE-SO button.

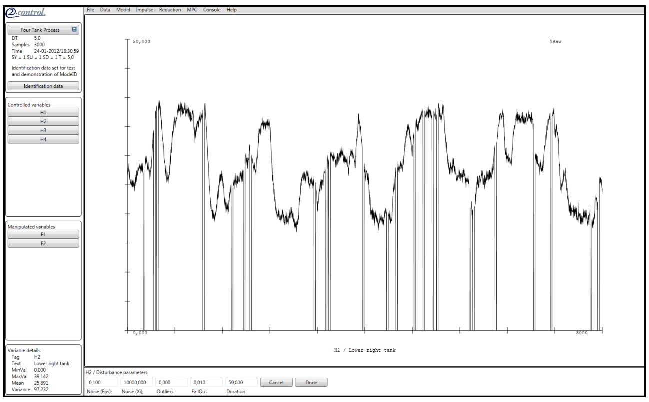
Four Tank Process



where

Noise	Select noise level. (Clean, Low, High)
Repair	Repair spikes and fall outs
Samples	Number samples for identification and verification

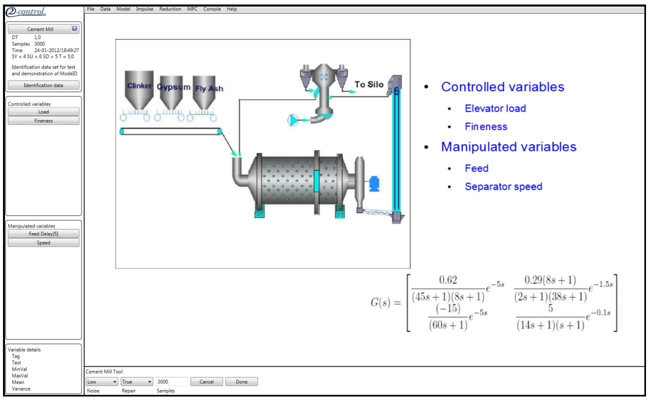
You can set the simulation parameters for the point using the buttons in the left panel.



where

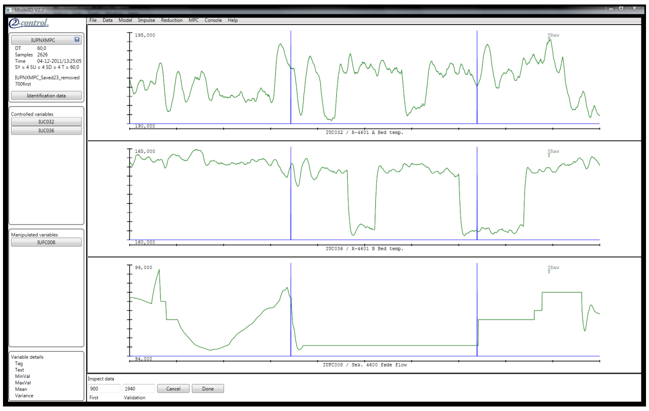
noise(eps)	Measurement noise
noise(xi)	Process noise
Outliers	Outlier probability (0 no spikes)
FallOut	Fall out probability
Duration	Fall out duration

Cement Mill



The cement mill have the same commands and options as the four tank process.

Inspect data



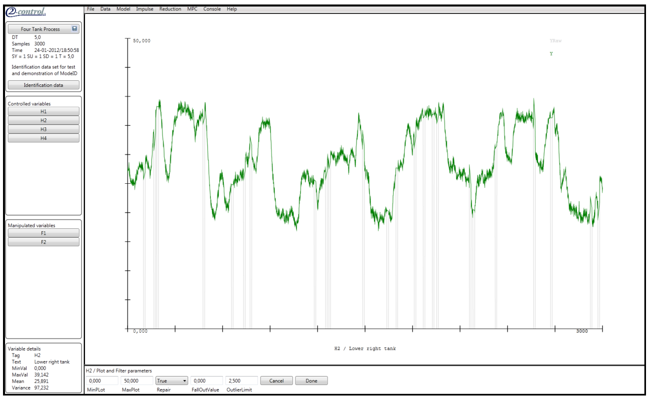
where

First	First data point used for identification
Validation	First data point for validation

The position of First and Validation are shown as vertical blue lines (sorry they cannot be dragged to the correct position, yet)

You can switch between identification and validation data by pressing Identification data button in the upper left panel.

Pressing a variable button displays



where

MinPlot	Minimum plot value
MaxPlot	Maximum plot value
Repair	Repair outliers and fall outs
FallOutValue	Fallout value
OutlierLimit	Threshold for detection of outliers

The plot shows the raw values in grey and the repaired values in green.

Model Tool

The structure of the model to be identified is:

$$Y(t) = G(q)U(t) + H(q)E(t) \tag{1}$$

where

$$G(q) = \frac{B(q)}{A(q)} \qquad H(q) = \frac{\Lambda}{D(q)} \tag{2}$$

The polynomials are

$$A(q) = I - \sum_{j=1}^{sy} A_j q^{-j} \qquad B(q) = \sum_{j=1}^{su} B_j q^{-j} \tag{3}$$

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_{n_y} \end{pmatrix} \tag{4}$$

$$D(q) = I - \sum_{j=1}^{sd} D_j q^{-j} \tag{5}$$

The deterministic one step predictor

$$\hat{Y}(t|t-1) = \sum_{j=1}^{sy} A_j Y(t-j) + \sum_{j=1}^{su} B_j U(t-j) \tag{6}$$

Predictor for the individual output variable

$$\hat{y}_i(t|t-1) = \sum_{j=1}^{sy} a_{i,j} Y(t-j) + \sum_{j=1}^{su} b_{i,j} U(t-j) \tag{7}$$

where $a_{i,j}$ and $b_{i,j}$ are the i rows of A_j and B_j , $1 \leq i < n_y$.
The prediction errors for y_i

$$\epsilon_i(t) = y_i(t) - \hat{y}_i(t) \tag{8}$$

Having n samples of $Y(t)$ and $U(t)$, $0 \leq t < n$, estimated $\hat{A}(q)$ and $\hat{B}(q)$ can be determined minimizing n_y MISO problems

$$V_i = \sum_{t=1}^{n} \ell_i(F_i(q)\epsilon_i(t)) \tag{9}$$

where ℓ_i are suitable norm functions.
Linear low pass filter reducing the effect of high frequency noise signals

$$F(q) = \frac{1-f}{1-fq^{-1}} \qquad 0 \leq f < 1 \tag{10}$$

Having identified the deterministic part $\hat{G}(q)$ as the combined result of n_y MISO identifications, the stochastic noise signal can be estimated

$$W(t) = Y(t) - \hat{Y}(t) = Y(t) - \hat{G}(q)U(t) \tag{11}$$

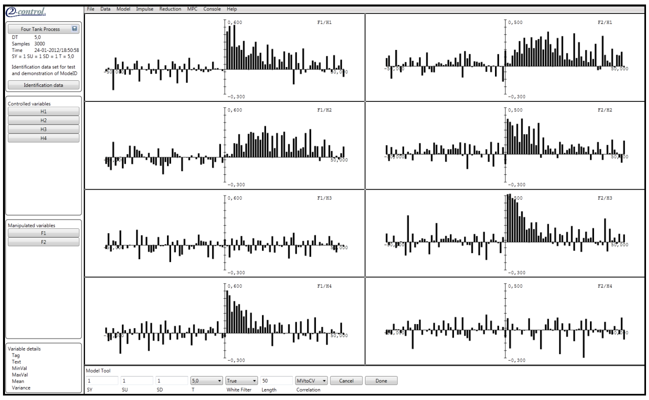
with the one step predictor:

$$\hat{W}(t|t-1) = \sum_{j=1}^{sd} D_j W(t-j) + \Lambda E(t) \tag{12}$$

During the MISO identifications a MISO model for the noise is calculated assuming the D_j matrices to be diagonal matrices. When moving from the the Model tool to the Impulse a MIMO model of the noise is calculated assuming the D_i matrices to be general matrices.

Correlations

In the Model tool the dimensions of the models used for MISO identification are entered, together with delays for Input variables.



where

SY	Dimension of $A(q)$, equation (3)
SU	Dimension of $B(q)$, equation (3)
SD	Dimension of $D(q)$, equation (5)
T	Sample time for identified model
WhiteFilter	Use white filter for correlations
Length	Lenght of correlations
Correlation	Switch between "Input to Output variables" and "Output to Output variables" correlations.

The white filter tries to make the input variable $u(t)$ as white as possible with a whitening filter

The whitening filter $F_{wh}(q)$ is determined by modelling the input variable as an AR-process with dimension 10.

$$F_{wh}(q)u(t) = e(t) \tag{13}$$

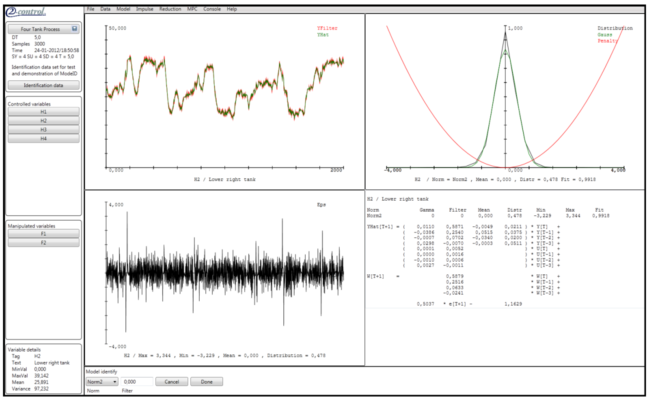
Filtered values

$$u_F(t) = F_{wh}(q)u(t) \tag{14}$$

$$y_F(t) = F_{wh}(q)y(t) \tag{15}$$

Clicking the button of one of the input variables, let you enter a general delay for the variable.

MISO identification



where

Norm	Norm for identification. ℓ_2,ℓ_1,ℓ_∞,ℓ_{Huber} and $IV4$ can be selected.
Filter	Low pass filter for identification.

In the upper left panel the red curve shows the sampled data variables(repaired) and the green curve shows the predicted values.

The lower left panels show the difference ϵ_i between the measured variable and the predicted values

The red curve in the upper right panel shows the penalty function for the MISO identification. The green curve show the normal distribution and the black curve shows the actual distribution for ϵ_i.

Finally does the lower right panel show the calculated MISO model. The first SY lines shows the a_{ij} and the next SU lines shows the b_{ij} of equation (7) finally the last SD lines shows the coefficients of the stochastic MISO model (12).

Use the buttons for the output variables n the left panel to switch between the Ny MISO identifications. When a you are satisfied with all the MISO identification move on to the Impulse tool by pressing the Done button.

Instrumental Variable Methods

Regressions using the the $\ell_2, \ell_1, \ell_\infty$ and ℓ_{Huber} norms delivers unbiased estimates if we are dealing with ARX processes. If the process cannot be properly described as an ARX process, the estimate will be biased.

An example is data from a second order Output Error process with a time constant of 10.0 sec and a damping of 1.5. This process is described by

$$y(t) = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 - a_1 q^{-1} - a_2 q^{-2}} u(t) + \sigma e(t) \tag{16}$$

with $a_1 = 1.7322$, $a_2 = -0.7408$, $b_1 = 0.0045$ and $b_2 = 0.0041$

Regression results with the ℓ_2 norm, using ModelID are shown if Table 1. The parameters for $\sigma^2 = 0.0$ are the correct ones. The results obtained with σ^2= 0.01 and 0.1 are very biased.

Table 1: ℓ_2 norm estimates

σ^2	a_1	a_2	b_1	b_2
0.0	1.7322	-0.7408	0.0045	0,0041
0.01	0.5515	0.3382	0.0028	0,0730
0.1	0.4376	0.4282	0.0262	0,0954

The Instrumental Variable methods are a possible solutions to his problem. ModelID has implemented the IV4 algorithm , which can be selected during the MISO identifications with the Model tool. The results obtained with the IV4 algorithm are shown if Table 2.

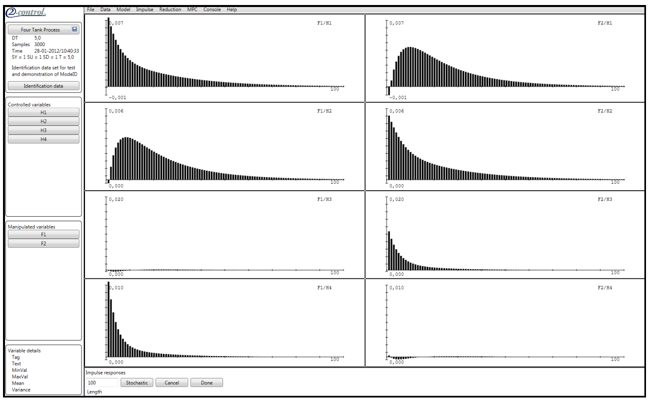
Table 2: IV4 estimates with Filter = 0.5

σ^2	a_1	a_2	b_1	b_2
0.0	1.7322	-0.7408	0.0045	0,0041
0.01	1.8953	-0.9048	0.0126	0.0031
0.1	1.9390	-0.9447	0.0126	0,0069

The IV4 algorithm gives a much better estimate. In some cases the IV4 algorithm gives an unstable predictor, which luckily is very clearly seen. The predictor can be stabilised using the filter option. If stable, the prediction are rather insensitive to the selected value of the filter.

0.1 Impulse tool

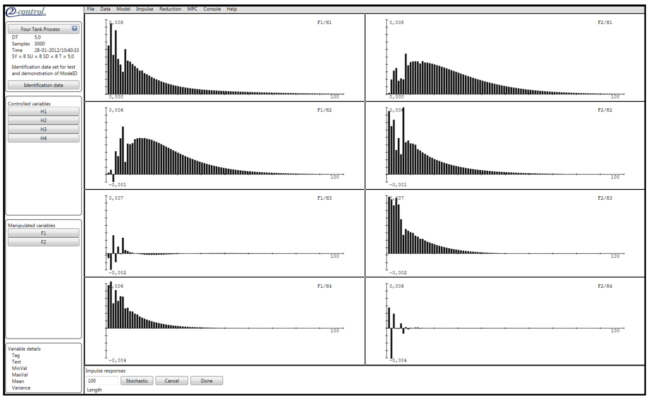
The impulse tool show the Impulse responses



where

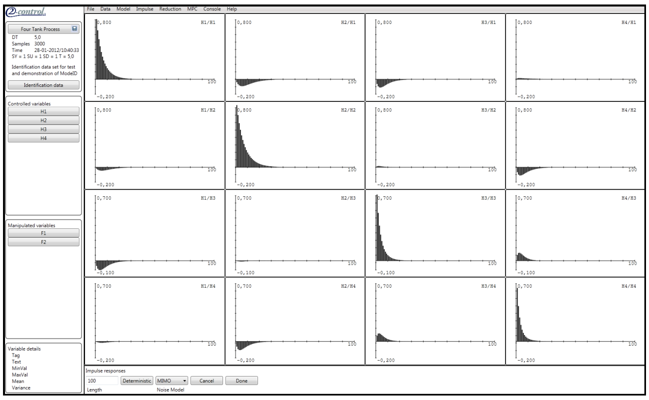
Length	Is the length of the calculated impulse responses. Length should chosen sufficiently high to show the full impulse response.
Stochastic	Switch to stochastic impulse responses

If the dimensions of the system equations are chosen too high the impulse responses will typically have an oscillating initial part as shown below.



Revert to the Model tool by pressing Cancel and enter a lover value.

Selecting Stochastic button displays the impulse responses for the noise model, equation (12).



where

Length	Is the length of the calculated impulse responses.
Noise Model	Select noise model (MIMO, MISO or ARX)
Deterministic	Switch to deterministic impulse responses

If the plant data negligible or zero noise signals you can select ARX noise model assuming that the process can bes described adequately as by an ARX process.

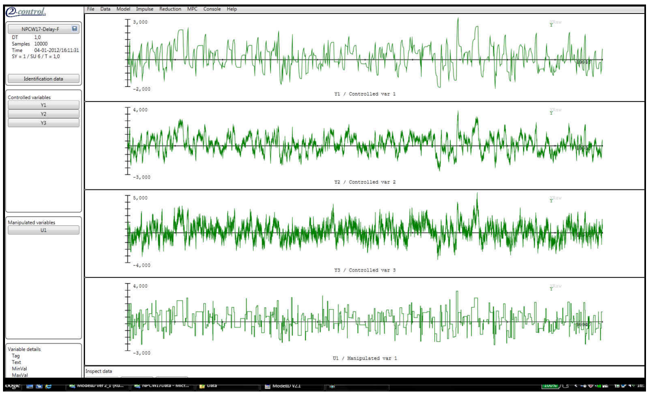
In some cases the MIMO noise model is unstable, the you can select the MIMO or ARX noise model.

Estimating time delays

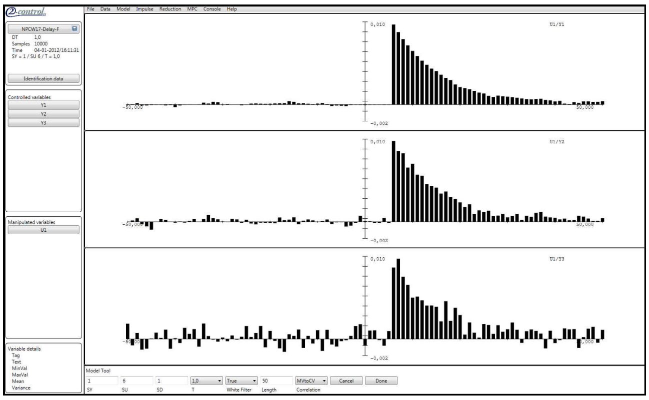
Proper estimation of time delays between the input variables and the output variables is important in order to minimize the dimension of the identified state space model. Data from first order and second order processes with pure time delays will be used to illustrate the problem.

The first order system, with a delay of 5 second and a time constant on 10 seconds, is given by

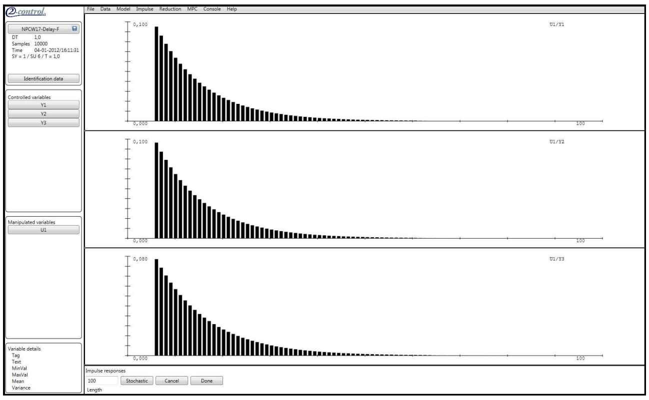
$$y(t) = 0.9048y(t-1) + 0.0952u(t-6) + \sigma * e(t) \tag{17}$$



First order ARX processes signal with increasing noise levels, $\sigma^2 = (0.0, 0.01, 0.1)$.



First order ARX processes cross correlations. The time delay of 5 second is shown on all three responses.

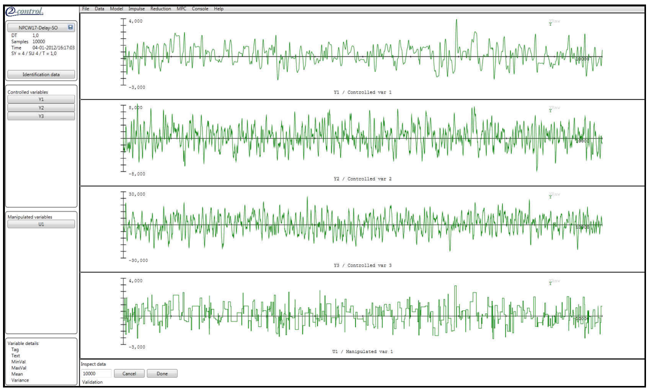


First order ARX processes impulse responses with MISO model dimension $sy = 1$, $su = 6$. All three impulse responses clearly shows the delay.

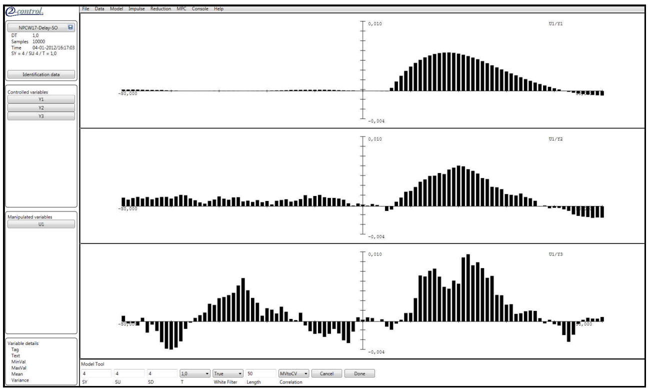
You should return to the Model tool at set $su = 1$ and set a delay of 5 seconds for the input variable.

The second order ARX system, with with a time delay of 5 second, a time constant of 10.0 second and damping of 0.5, is given by

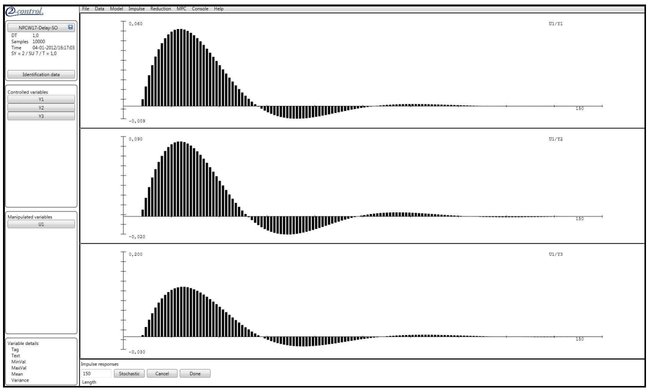
$$y(t) = 1.8953y(t-1) - 0.9048y(t-2) \\ + 0.0047u(t-6) + 0.0047u(t-7) + \sigma e(t) \quad (18)$$



Second order ARX processes signal with increasing noise levels, $\sigma^2 = (0.0, 0.01, 0.1)$



Second order ARX processes cross correlations. With increasing noise level, the proper time delay cannot be determined from the correlation plots



Second order ARX processes impulse responses with $sy = 2$ and $su = 7$. The proper delay is obtained for all three noise levels.

You should return to the model tool at set $su = 2$ and set a delay of 5 seconds for the input variable.

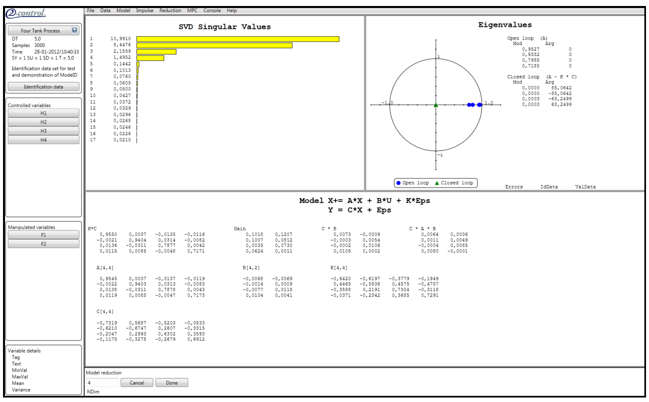
Reduction Tool and State-Space model

In the reduction tool a block-Hankel matrix is constructed from the impulse responses. The Hankel matrix is factorized using a singular value decomposition (SVD)algorithm. The state space model in innovation form can be realized by model reduction in balanced form using the SVD of the Hankel matrix. The state space model in innovation form is

$$X(t+1) = AX(t) + BU(t) + KE(t) \tag{19a}$$
$$Y(t) = CX(t) + E(t) \tag{19b}$$

The rank of the Hankel matrix is equal to the minimal rank for a state-space system representing the process. For a NDim system, the first NDim singular values are non zero and the subsequent singular values close to zero.

The result of the SVD reduction is



The upper left graph shows the singular values from the SVD. The values show that the system can be represented by a state-space model of dimension

NDim = 4. You can set the desired dimension, NDim, and ModelID calculates the state space model as shown in the lower half of the diagram.

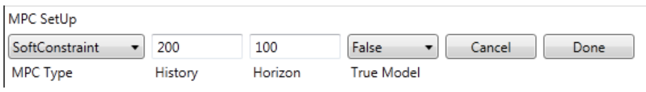
The upper right part of the diagram show the open and closed loop eigenvalues ($A - C * K$ in eq. (19))for the state space model.

If the length of the impulse responses is chosen too short, the singular values will decrease gradually, making the selection of the state-space model dimension, NDim, difficult.

MPC Tool

ModelID includes a MPC module, where tuning of normal MPC and soft constrained MPC for the derived State-Space model can be tested simulating the MPC control loop.

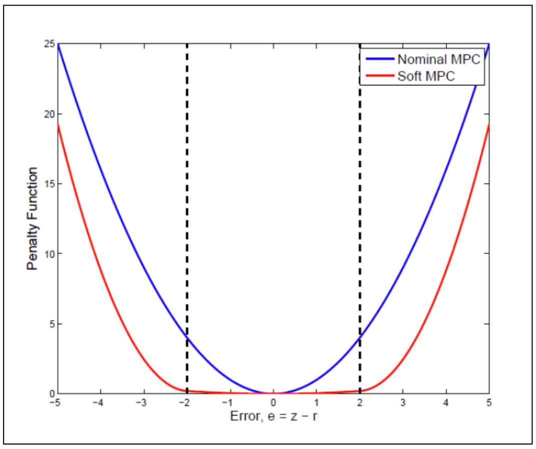
First you have to set up the MPC



where

MPC Type	Normal or Soft Constrained MPC
History	Length of history plot
Horizon	Prediction horizon
True Model	For the Four Tank Process and the Cement mill you can select the true model for the MPC simulations

Model predictive control implements both conventional MPC and Soft Constrained MPC. The conventional MPC has a quadratic penalty function and the Soft Constrained MPC has a dead band zone around the set point where the penalty for not reaching the exact set point is low.



The SoftConstrained MPC minimizes the problem

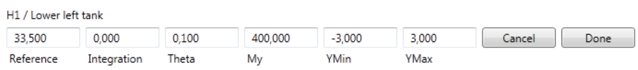
$$\min_{\{z,u,\eta\}} \phi = \frac{1}{2}\sum_{k=0}^{N-1} \|z_{k+1} - r_{k+1}\|^2_{Q_z} + \|\Delta u_k\|^2_S + \sum_{k=1}^{N} \frac{1}{2}\|\eta_k\|^2_{S_\eta} + s'_\eta \eta_k$$

subject to the constraints

$$\begin{array}{ll} z_k = b_k + \sum_{i=1}^{n} H_i u_{k-i} & k = 1, \ldots N \\ u_{\min} \le u_k \le u_{\max} & k = 0, \ldots N-1 \\ \Delta u_{\min} \le \Delta u_k \le \Delta u_{\max} & k = 0, \ldots N-1 \\ z_k \le z_{\max,k} + \eta_k & k = 1, \ldots N \\ z_k \ge z_{\min,k} - \eta_k & k = 1, \ldots N \\ \eta_k \ge 0 & k = 1, \ldots N \end{array}$$

where zk is the plant response, r_k the set-point, ΔU_k the movement of the input variables. The Q_z, S and S_η are weighing matrices for reference error penalty, S movement of input variables penalty and S_η the penalty for coming outside the dead-band zone. Setting S_η to zero result in a conventional MPC controller. The first constraint for z_k is the linear process model.

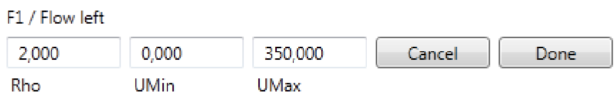
press the output variable buttons in the left panel to set the output variable parameters:



where

Reference	Reference (setpoint)
Integration	Not implemented yet
Theta	Penalty outside soft constraint region
My	Penalty inside soft constraint region
YMin	Low limit for soft constraint region
YMax	High limit for oft constraint region

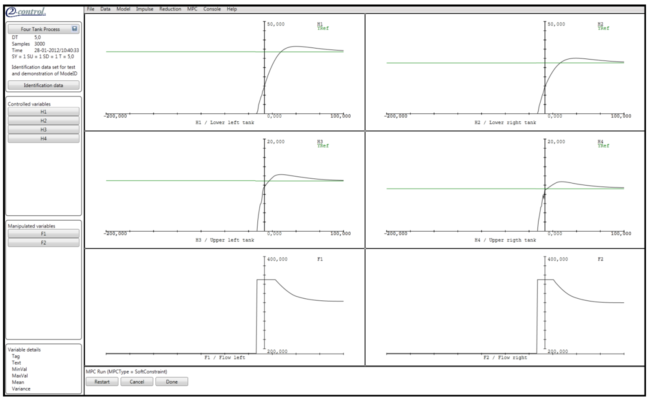
The parameters for the input variables are



where

Rho	Penalty for actuator movement
UMin	Low limit for input variable
UMax	High limit for input variable

Press done to start the MPC controller



Reading the identified Model from a .mdl file

ModelID stores the identified model in a file with the extension *.mdl*. This file contains all selections done during the identification process and *MPCMath StateSpace* object with the model. The readModelFile project in the 2-control directory, demonstrates how to read an *.mld* file and get the *StateSpace* model. The Model property is set to *null* until *ModelID* creates the model in the ModelReduction tool.

The object *Modeldata* and the *Point* objects are documented below. For documentation of the *StateSpace* model see the *MPCMath* user manual, which can be retrieved from the www.2-control.dk website

1 ModelData

Class ModelData

ModelData object for ModelID models saved to file

```
[Serializable]
class ModelData
```

Constructors

`ModelData()`

Properties

CorLength

Correlation lengths

```
int CorLength {set; get;}
```

DataFile

Data file for plant data

```
string DataFile {set; get;}
```

Description

Model desription

```
string Description {set; get;}
```

DT

Sample time

```
double DT {set; get;}
```

FirstRecord

First identification record

```
int FirstRecord {set; get;}
```

History

History lengt for MPC plot

```
int History {set; get;}
```

Horizon

MPC Prediction horizon

```
int Horizon {set; get;}
```

ImpLength

Impulse response lengths

```
int ImpLength {set; get;}
```

LastRecord

Last Validation record

```
int LastRecord {set; get;}
```

Model

State Space model

```
StateSpaceModel Model {set; get;}
```

MPCType

MPC Type

```
MPCType MPCType {set; get;}
```

NDim

Dimension of State Space model

```
int NDim {set; get;}
```

NoiseModel

Noise Model for stochastic Impulse responses

```
NoiseModels NoiseModel {set; get;}
```

Notes

Notes

```
string Notes {set; get;}
```

Points

Point List

```
Point[] Points {set; get;}
```

SD

Dimension of Noise variables

```
int SD {set; get;}
```

SU

Dimension of Manipiulated variables

```
int SU {set; get;}
```

SY

Dimension of Controlled variables

```
int SY {set; get;}
```

T

Sample time for model

```
double T {set; get;}
```

Title

Model Title

```
string Title {set; get;}
```

ValRecord

First validation record

```
int ValRecord {set; get;}
```

2 Point

Class Point

point class, data for Controlled and manipulated variables

```
[Serializable]
class Point : IPoint, INotifyPropertyChanged
```

Constructors

`Point()`

Constructor

```
Point(string Tag)
```

Constructor
Parameters

Tag	Tag name

```
Point(string Tag, PointType Type, double MinPlot, double MaxPlot, string Description)
```

Constructor
Parameters

Tag	Tag name
Type	Point Type
MinPlot	Min plot value
MaxPlot	Max plot value
Desription	variable deswcription

Properties

Col

Column in data file

```
int Col {set; get;}
```

Description

Description

```
string Description {set; get;}
```

Pos

Position in data matrix

```
int Pos {set; get;}
```

PropertyChangedEventHandler

```
event PropertyChangedEventHandler PropertyChanged {}
```

Tag

Tag Unique identificatiom

```
string Tag {set; get;}
```

Enumeration **PointType**

Point type

```
enum PointType
```

Fields

Input	Plant Input Variable
Off	Not selected for identification
Output	Plant Output Variable

Enumeration **PointNorm**

```
enum PointNorm
```

Fields

- Huber
- IV4
- Norm1
- Norm2
- NormInfinity