Akademie věd České republiky
Ústav teorie informace a automatizace

Academy of Sciences of the Czech Republic
Institute of Information Theory and Automation

RESEARCH REPORT

Miroslav Novák, Ludvík Tesař

A Set of Tutorial Experiments for Jobcontrol Environment

System Identification and Controller Design

No. 2151 November 29, 2005

ÚTIA AV ČR, P.O.Box 18, 182 08 Prague, Czech Republic
Tel: (+420)266052422, Fax: (+420)286890378, Url: http://www.utia.cas.cz,
E-mail: utia@utia.cas.cz

Contents

1 Introduction **5**

2 A SISO system control **7**
2.1 Aims of the study . . . 7
2.2 Description of the study . . . 7
2.3 Data . . . 7
2.4 Processing . . . 7
2.5 Description . . . 8
2.5.1 Experiment definition . . . 8
2.5.2 Data description . . . 8
2.5.3 Channels description . . . 8
2.5.4 Prior information . . . 9
2.5.5 Mixture initialization . . . 10
2.5.6 Mixture estimation . . . 10
2.5.7 Mixture validation . . . 10
2.5.8 User ideal . . . 10
2.5.9 Design . . . 11
2.5.10 Verification . . . 11
2.5.11 Original Data Plots . . . 12
2.6 Results . . . 12
2.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation) . . . 12
2.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation) . . . 15
2.6.3 User Ideal Mixture . . . 16
2.6.4 Controller Mixture . . . 16
2.7 Experimental Controller Verification Results . . . 18
2.8 Conclusion . . . 19

3 Disputace **20**
3.1 Aims of the study . . . 20
3.2 Description of the study . . . 20
3.3 Data . . . 20
3.4 Processing . . . 21
3.5 Description . . . 21
3.5.1 Experiment definition . . . 21
3.5.2 Data description . . . 21
3.5.3 Channels description . . . 22
3.5.4 Prior information . . . 23
3.5.5 Mixture initialization . . . 23
3.5.6 Mixture estimation . . . 23
3.5.7 Mixture validation . . . 23
3.5.8 User ideal . . . 24

3.5.9 Design . . . 24
3.5.10 Verification . . . 24
3.5.11 Original Data Plots . . . 25
3.6 Results . . . 25
3.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation) . . . 25
3.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation) . . . 29
3.6.3 User Ideal Mixture . . . 32
3.6.4 Controller Mixture . . . 33
3.7 Experimental Controller Verification Results . . . 33
3.8 Conclusion . . . 34
3.8.1 Experiment with higher process noise . . . 34
3.8.2 Experiment with a non-zero setpoint . . . 35
3.8.3 Experiment with offset change in the middle of the simulation . . . 35

4 Single axis helicopter **37**
4.1 Aims of the study . . . 37
4.2 Description of the study . . . 37
4.3 Data . . . 39
4.4 Processing . . . 39
4.5 Description . . . 39
4.5.1 Experiment definition . . . 39
4.5.2 Data description . . . 39
4.5.3 Channels description . . . 40
4.5.4 Prior information . . . 41
4.5.5 Mixture initialization . . . 41
4.5.6 Mixture estimation . . . 41
4.5.7 Mixture validation . . . 41
4.5.8 User ideal . . . 42
4.5.9 Verification . . . 42
4.5.10 Original Data Plots . . . 43
4.5.11 Results . . . 43
4.5.12 Results of Single Component Identification and Parameter Estimation (and Model Validation) . . . 43
4.6 Designer Controller Tuning . . . 46
4.7 Designer's Controller Prediction Results . . . 48
4.8 Designer's Noncontrolled Prediction Results . . . 49
4.9 Experimental Controller Verification Results . . . 50
4.10 Experimental Noncontrolled Verification Results . . . 51
4.11 Conclusion . . . 52

5 Banana identification experiment **53**
5.1 Aims of the study . . . 53
5.2 Description of the study . . . 53
5.3 Data . . . 53
5.4 Processing . . . 53
5.5 Description . . . 54
5.5.1 Experiment definition . . . 54
5.5.2 Data description . . . 54
5.5.3 Channels description . . . 54
5.5.4 Prior information . . . 55
5.5.5 Mixture initialization . . . 56
5.5.6 Mixture estimation . . . 56

5.5.7 Mixture validation . . . 56
5.5.8 User ideal . . . 56
5.5.9 Design . . . 56
5.5.10 Verification . . . 57
5.5.11 Original Data Plots . . . 57
5.6 Results . . . 57
5.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation) . . . 57
5.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation) . . . 63
5.6.3 User Ideal Mixture . . . 65
5.6.4 Controller Mixture . . . 65
5.7 Conclusion . . . 69

6 Summary **70**

7 Acknowledgements **71**

Chapter 1

Introduction

This report contains a set of tutorial experiments performed in the Jobcontrol environment. It is focused on system identification and controller design. The Jobcontrol is a facade for underlaying toolboxes the Designer and Mixtools, which makes their utilization easier and hides most of the complexity of these toolboxes behind an user friendly interface.

The overall introduction to Jobcontrol is given in the accompanying paper [1], which represents a user's guide and reference for Jobcontrol. It is strongly recommended to read this paper before exploring the experiments given here.

The experiments performed are rather simple as they are intended to provide example on how to use the Jobcontrol practically. The operation of the Jobcontrol environment is illustrated and the description of each experiment is formed by the manually written introduction and conclusion surrounding a machine generated experiment report.

Thanks to the extensive reporting ability of the Jobcontrol, all important information about the experiment calculation is given. It begins from the experiment setting and ends with the results in the form of tables and figures showing the outcome of the particular steps of processing.

Every experiment starts with an introduction section, which contains the overall description of the experiment, its aims, and the origin of the data used for identification. The results of the calculation are summarized in the experiment conclusion, where they are compared with the aims. The introduction and conclusion sections are very important, because they give the useful comments to the results in the generated report.

The given experiments shows the tasks of identification, mixture controller and controller created by the Designer toolbox.

Following chapters contain these experiments:

SISO1T is a simple single SISO system example identified and controlled using the mixture approach

DISPUTACE is another SISO system example used to test the Mixtools toolbox. There is one main experiment presented in the main text. Several modifications are then performed and their results are summarized and compared to the main task in the conclusion. There are following modifications presented:

- different process disturbance noise levels
- zero and non-zero setpoints
- system parameters change in the middle of the simulation

VRT1 is an experiment demonstrating the Designer toolbox. It uses a non-linear Simulink scheme to model a single axis helicopter. The adaptive LQG controller is used to track varying setpoint while keeping the linear model up to date with the changing working point of the Simulink model..

In the conclusion the results are compared with use of a non-adaptive controller for this task.

BANAN is an example of mixture identification ability of the Mixtools toolbox. The example uses so called "banana" function to show the identification in a complicated case.

All these experiments are contained in the Jobcontrol source tree in the directory mixtools/jobcontrol/examples. For instructions how to get and install the Jobcontrol environment see [1].

Chapter 2

A SISO system control

Experiment: `siso1t` - Simple SISO system experiment with Mixtools Target

Author	:	Miroslav Novak
Contact	:	`mira@utia.cas.cz`
Address	:	AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic
Basic references	:	
Source texts	:	`siso1t`
Project	:	Designer

2.1 Aims of the study

Experiment "siso1t" is a simple testing system for the Mixtools toolbox using Target function for creating a controller.

2.2 Description of the study

System used for generating identification data and for verification is:

$$y_t = 1.81y_{t-1} - 0.8187y_{t-2} + 0.00468u_t + 0.00438u_{t-1} + \sqrt{0.001}e_t, \quad e_t \sim N(0,1)$$

2.3 Data

For generating identification data the inputs are

$$u_t \sim N(0,1)$$

2.4 Processing

Whole experiment files are kept under svn in `mixtools/jobcontrol/examples/simple`. Data used for identification are generated by the script `dv_genrawdata_siso1`. To run the experiment:
1) generate the identification data by the `dv_genrawdata_siso1` script, unless it was done before.
2) call "prodini" to initialize mixtools
3) call `jobproceed(siso1t)`, where `siso1t.m` is a function generating the Job description.
4) to generate a nice latex report do `latex siso1t`

2.5 Description

2.5.1 Experiment definition

```
jobname    = 'siso1t';  % name of experiment  ...  no spaces!
% choose short 'jobname' that serves as name root for temporary and saved files
authorname = 'Miroslav Novak';  % author of experiment
email      = 'mira@utia.cas.cz';  % E-mail of author of experiment
address    = 'AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic';  %
   author's address
references = '';  % references to literature
project    = 'ProDaCTool';  % project name
desc       = 'Simple SISO system experiment with Mixtools Target';  % description
   of experiment printed in protocol
debug      = 0;    % debug level determining information during evaluations
seed       = [];  % random seed ([] -> leave the seed, -1 -> randomize by timer)

steps    = [1, 1, 1, 1, 1, 1, 1, 1];  %           % Steps to be performed (1/0)
   = (yes/no)
```

2.5.2 Data description

```
datafile     = 'dv_genrawdata_siso1';  % filename with full path specifying
   the mat file with data
varname      = 'rawdata';  % variable name of data in 'datafile'
transpose    = 0;   % transpose data matrix to have channels to rows (1=yes
   0=no)
rescale_data = 1;   % rescale new data (normaly this is necesary every time
   the new data is given, but it does no harm to do it every time)  (1=yes 0=no)
reset_chns   = 0;   % reset selected channels (normally this is needed only
   once, in the beginning, and if new data does not have anything to do with
   the old data)  (1=yes 0=no)

chns         = [1, 2];  % modelled channels
pr_chns      = [1, 2];  % vector of numbers of channels from which original
   data plots are printed to protocol (-1 means all channels)
pr_merge     = 1;  % whether original data plots in protocol are merged or not
   (0=separated, 1=merged, 2=tiled).
used_data    = -1;   % % At the moment, there are 10000 data samples
```

2.5.3 Channels description

Description of the channel 1

```
chn_name       = 'y';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 0;  % available for control
chn_prty       = 0.5;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-0.0121804, 1.11326];  % expected physical range [min,max]
chn_drange    = [-1;
1];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
```

```
chn_preinfo   = 'olymedian', 'c', 1;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]

type = '';
```

Description of the channel 2

```
chn_name       = 'u';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 1;  % available for control
chn_prty       = 0.5;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [0.207821, 0.784081];  % expected physical range [min,max]
chn_drange    = [-1000;
1000];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 2;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]

type = '';
```

2.5.4 Prior information

```
ncom    = 1;       % the number of components
ord     = 2;       % order of the richest regressor
```

```
diacove = 0.0001;  % diagonal of noise covariancecove
diaCth  = 10000;   % diagonal of par.  covariance
dfm     = 1000;     % degrees of freedom of factors
dfcs    = 1000;  % degrees of freedom of components

doflattening = 0;  % indicates whether the initial mixture should be flattened
```

2.5.5 Mixture initialization

```
SingleOnly  = 0;       % boolean flag whether to skip mixinit and calculate
   just MixSingle
opt     = 'p';    % iterative estimation method (p | q | b | f | Q | P)
niter   = 10;    % maximum number of iterations
frg     = 0.999999;    % forgetting factor
```

2.5.6 Mixture estimation

```
opt         = 'p';       % iterative estimation method (p | q | b | f | Q |
   P)
niter       = 40;           % maximum number of iterations
frg         = 0.999999;           % forgetting factor
frgEstType  = 0;           % estimation type of forgetting factor (0-none, 1-zero
   alt, 2-prev estimate)
frgEstGrid    = [1, 0.99, 0.983362, 0.972317, 0.953941, 0.923366, 0.872495,
   0.787855, 0.647029, 0.412721, 0.022876];  %
% estimation grid of forgetting factor
```

2.5.7 Mixture validation

```
nsteps  = 1;     % the prediction is made nsteps ahead
pchns    = [1, 2];  % channels to be predicted
cchns    = [];  % channels in condition
tstart  = 1;     % starting time determining the part of data used in validation
tend  = 10000;  %   ending time determining the part of data used in validation
epss  = 1;     % (1/0) = (produce/do not produce) encapsulated postscript plots
pauses  = 0;     % pause in seconds after each plot set
plots    = [0, 0, 0, 0];  % [show cluster plot , show time plot, mixture plot,
   histogram], where (1/0)=(y/n)
segments    = 0;     % number of segments for segmentation test (0 means perform
   no segm.  test).  These tests take a very long time.
stoperr     = 1;     % stop on unsuccessful validation.  (1/0)=(y/n)
alt         = 1;     % perform validation using alternative forgetting estimation
   test (takes very long time, 0=do not run, 1=run this test).
```

2.5.8 User ideal

```
method = 't';  % method determining the user target (t | d | z)
% t ...  User defined (initialized by target.m) - this is default option
% d ...  Designer is used
% z ...  User defined (initialized by zeros)
userSetpointEths   = [0, 0];  % user target component Eths
userSetpointCoves  = [1, 1e+06];  % user target component Coves
userSetpointCorrect  = 1;  % do the user target component Correction ?  (0=no,1=yes)
```

```
incremental  = 0;  % use incremental penalization controller ?  (0=no,1=yes)
```

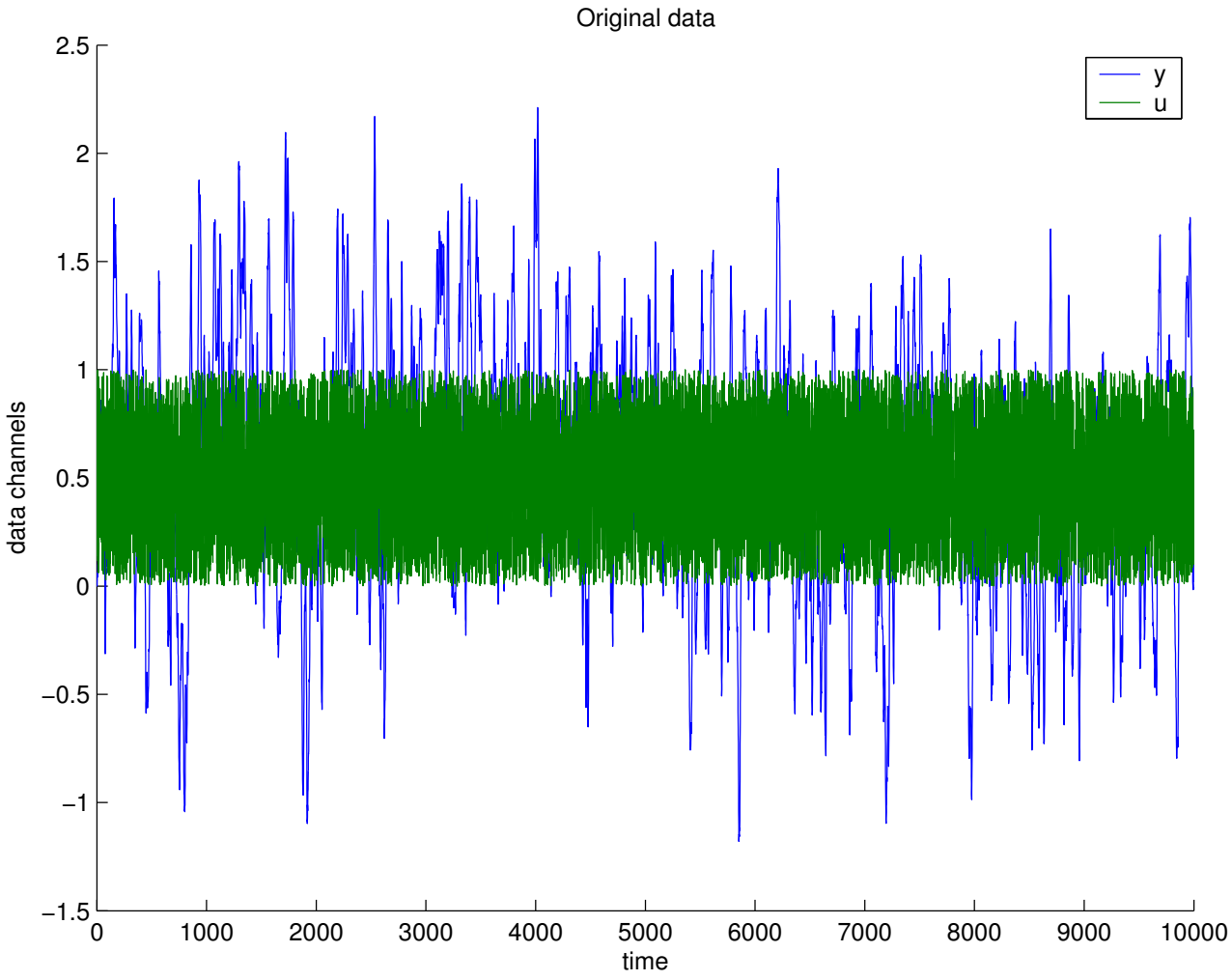
2.5.9 Design

```
designtype = 'i';  % design type (a | i | s)
% a ...  academic design
% i ...  industrial design
% s ...  simultaneous design
horizon    = [50, -1];  % horizon for evaluation of KLD
ufc        = [];  % ufcgen(aMix, aMixu) is used if ufc=[]
```

2.5.10 Verification

```
method = 't';  % verification method (t | d)
% t ...  verification of controller designed by step 7
% d ...  verification of controller designed by UserIdeal with Designer in step
   6
type = 'mixture'; % simulation type (none,simulink,mixture)
MixVer = varload('siso1t_mixver','MixVer'); % verification mixture, enter empty
   matrix to use the identification result
smlsimlength = 1000;     % simulation length for verification
```

2.5.11 Original Data Plots



2.6 Results

2.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	1.959e+03	(the bigger the better)
Test of validity of the model:	1	(1=O.K.,0=bad)
Relative SE of pred.err:	[0.000563259, 0.0100006]	
	(standard error of ep relative to std of data)	
Test of whiteness:	[0.0569913, 0.0524522]	
	(sum of correlations with delayed predictions)	
Test of the condition number:	[4703.63, 38.3829, 5547.12, 130.598, 6284.44, 1220.7, 4285.1, 16.1256, 4986.18, 13.24	
	(proportion of biggest and smallest eigenvalue of data matrix)	

Elementary statistics for the channel y:

	MIN	MAX	MEAN	MEDIAN	STD
data	-1.17939	2.21166	0.550652	0.559995	0.562724
differences	-0.198751	0.228216	1.91239e-05	-0.000612289	0.0557418
predictions	-1.22481	2.24095	0.550465	0.559804	0.561802
errors of prediction	-0.123575	0.116885	0.000186817	0.00047614	0.0316938

Elementary statistics for the channel u:

	MIN	MAX	MEAN	MEDIAN	STD
data	9.01343e-05	0.999918	0.496014	0.495356	0.288124
differences	-0.98473	0.988182	-7.35039e-05	0.00352103	0.406616
predictions	0.48947	0.503043	0.496339	0.496339	0.00269178
errors of prediction	-0.499192	0.508115	-0.000324842	-0.000919476	0.288127

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	0.00332	0.0908	0.184
component 2	0.00295	0.0481	0.139
component 3	0.00288	0.0105	0.0809
component 4	0.0031	0.133	0.136
component 5	0.00304	0.113	0.169
component 6	0.0029	0.0113	0.0769
component 7	0.0027	0.0296	0.0816
component 8	0.00273	0.161	0.132

Mixture Factors

This mixture has 8 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1992.764705, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=0.00332444, dfm=1396.443574

delays →		0	1	2
offset			0.0254656	
1	y		1.81056	-0.818404
2	u	0.0386102	0.00306867	0.00264698

   Factor 2, modelled channnel: 2, called 'u', cove=0.0908168, dfm=643.449676

delays →		0	1	2
offset			-0.84018	
1	y		-0.0248394	
2	u		0.0214779	-0.0253872

2. component, dfcs=1507.208982, factors:
   Factor 3, modelled channnel: 1, called 'y', cove=0.00295473, dfm=1042.004114

delays →		0	1	2
offset				
1	y		1.81278	-0.825892
2	u	0.00549226		

   Factor 4, modelled channnel: 2, called 'u', cove=0.0480899, dfm=538.067667

delays →		0	1
offset			1.17689
1	y		
2	u		

3. component, dfcs=877.848773, factors:
   Factor 5, modelled channnel: 1, called 'y', cove=0.002882, dfm=759.558895

delays →		0	1	2
offset				
1	y		1.82671	-0.835076
2	u		0.00357796	0.00239426

   Factor 6, modelled channnel: 2, called 'u', cove=0.0105385, dfm=588.537443

delays →		0	1
offset			-1.55992
1	y		
2	u		0.00852314

4. component, dfcs=1477.492899, factors:
   Factor 7, modelled channnel: 1, called 'y', cove=0.003096, dfm=730.192481

delays →		0	1	2
offset			-0.0094869	
1	y		1.73629	-0.733612
2	u			-0.00625134

   Factor 8, modelled channnel: 2, called 'u', cove=0.132959, dfm=296.212771

delays →		0	1
offset			0.575572
1	y		
2	u		0.0626245

5. component, dfcs=1835.268888, factors:
   Factor 9, modelled channnel: 1, called 'y', cove=0.0030384, dfm=1034.829765

delays →		0	1	2
offset			-0.00504452	
1	y		1.81475	-0.822106
2	u			

   Factor 10, modelled channnel: 2, called 'u', cove=0.11339, dfm=387.774721

delays →		0	1
offset			-0.238939
1	y		
2	u		

6. component, dfcs=834.370300, factors:
   Factor 11, modelled channnel: 1, called 'y', cove=0.00289895, dfm=732.072393

delays →		0	1	2
offset			-0.0857733	
1	y		1.82453	-0.832945
2	u	0.056043	0.00511369	

   Factor 12, modelled channnel: 2, called 'u', cove=0.0112816, dfm=548.187967

delays →		0	1
offset			1.58209
1	y		
2	u		

7. component, dfcs=884.666363, factors:
   Factor 13, modelled channnel: 1, called 'y', cove=0.00270471, dfm=352.131763

delays →		0	1	2
offset			0.0264342	
1	y		1.81473	-0.825945
2	u	0.0115676		-0.00416591

Factor 14, modelled channnel: 2, called 'u', cove=0.0296442, dfm=203.645471

delays →		0	1	2
offset			-1.22122	
1	y			0.0136029
2	u			0.0120795

8. component, dfcs=1435.531948, factors:
   Factor 15, modelled channnel: 1, called 'y', cove=0.00273428, dfm=575.673990

delays →		0	1	2
offset			0.0186852	
1	y		1.88061	-0.898927
2	u	-0.0254451	0.00211818	0.00469921

   Factor 16, modelled channnel: 2, called 'u', cove=0.160593, dfm=250.840317

delays →		0	1	2
offset			0.440853	
1	y			0.0127503
2	u		-0.035918	

2.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	3.682e+02	(the bigger the better)
Test of validity of the model:	0	(1=O.K.,0=bad)
Relative SE of pred.err:	[0.000563556, 0.0100005]	
	(standard error of ep relative to std of data)	
Test of whiteness:	[0.0549918, 0.0507126]	
	(sum of correlations with delayed predictions)	
Test of the condition number:	[5099.93, 1.00048]	
	(proportion of biggest and smallest eigenvalue of data matrix)	

Elementary statistics for the channel y:

	MIN	MAX	MEAN	MEDIAN	STD
data	-1.17939	2.21166	0.550652	0.559995	0.562724
differences	-0.198751	0.228216	1.91239e-05	-0.000612289	0.0557418
predictions	-1.22471	2.24139	0.550642	0.559758	0.561829
errors of prediction	-0.125308	0.117345	1.0206e-05	0.000114461	0.0317111

Elementary statistics for the channel u:

	MIN	MAX	MEAN	MEDIAN	STD
data	9.01343e-05	0.999918	0.496014	0.495356	0.288124
differences	-0.98473	0.988182	-7.35039e-05	0.00352103	0.406616
predictions	0.496017	0.496017	0.496017	0.496017	6.36745e-14
errors of prediction	-0.495927	0.503901	-2.78884e-06	-0.000660595	0.288124

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	0.00341	1.07	1

Mixture Factors

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=10845.145960, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=0.00341127, dfm=10845.145999

delays →		0	1	2
offset				
1	y		1.81299	-0.821943
2	u			

   Factor 2, modelled channnel: 2, called 'u', cove=1.07418, dfm=10845.145978

delays →		0	1
offset			0.000229105
1	y		
2	u		

2.6.3 User Ideal Mixture

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1.000000, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=0.0107729, dfm=1.000000

delays →		0	1
offset			-0.978354
1	y		
2	u		

   Factor 2, modelled channnel: 2, called 'u', cove=1.29389e+07, dfm=1.000000

delays →		0	1
offset			-1.72128
1	y		
2	u		

2.6.4 Controller Mixture

This mixture has 8 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1992.764705, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=0.00332444, dfm=1396.443574

delays →		0	1	2
offset			0.0254656	
1	y		1.81056	-0.818404
2	u	0.0386102	0.00306867	0.00264698

   Factor 2, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

2. component, dfcs=1507.208982, factors:
   Factor 3, modelled channnel: 1, called 'y', cove=0.00295473, dfm=1042.004114

delays →		0	1	2
offset				
1	y		1.81278	-0.825892
2	u	0.00549226		

   Factor 4, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

3. component, dfcs=877.848773, factors:
   Factor 5, modelled channnel: 1, called 'y', cove=0.002882, dfm=759.558895

delays →		0	1	2
offset				
1	y		1.82671	-0.835076
2	u		0.00357796	0.00239426

   Factor 6, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

4. component, dfcs=1477.492899, factors:
   Factor 7, modelled channnel: 1, called 'y', cove=0.003096, dfm=730.192481

delays →		0	1	2
offset			-0.0094869	
1	y		1.73629	-0.733612
2	u			-0.00625134

   Factor 8, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

5. component, dfcs=1835.268888, factors:
   Factor 9, modelled channnel: 1, called 'y', cove=0.0030384, dfm=1034.829765

delays →		0	1	2
offset			-0.00504452	
1	y		1.81475	-0.822106
2	u			

   Factor 10, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

6. component, dfcs=834.370300, factors:
   Factor 11, modelled channnel: 1, called 'y', cove=0.00289895, dfm=732.072393

delays →		0	1	2
offset			-0.0857733	
1	y		1.82453	-0.832945
2	u	0.056043	0.00511369	

Factor 12, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

7. component, dfcs=884.666363, factors:
   Factor 13, modelled channnel: 1, called 'y', cove=0.00270471, dfm=352.131763

delays →		0	1	2
offset			0.0264342	
1	y		1.81473	-0.825945
2	u	0.0115676		-0.00416591

   Factor 14, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

8. component, dfcs=1435.531948, factors:
   Factor 15, modelled channnel: 1, called 'y', cove=0.00273428, dfm=575.673990

delays →		0	1	2
offset			0.0186852	
1	y		1.88061	-0.898927
2	u	-0.0254451	0.00211818	0.00469921

   Factor 16, modelled channnel: 2, called 'u', cove=0.752432, dfm=1.000000

delays →		0	1	2
offset			-1.97837	
1	y		-19.2124	13.7674
2	u		-0.0824461	-0.00115139

2.7 Experimental Controller Verification Results

Elementary statistics for simulated channels:

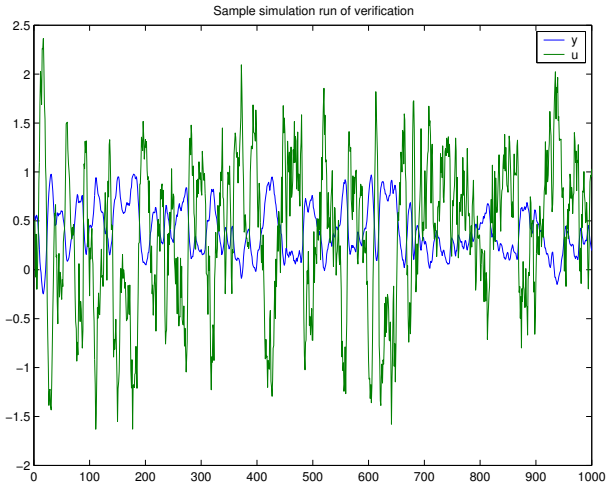
channel	mean	variance	range
1 "y"	0.39619	0.250988	[-0.016648, 0.809028]
Δ 1 "y"	-0.000355525	0.0496036	[-0.15833, 0.169598]
2 "u"	0.369309	0.754411	[-0.871588, 1.6102]
Δ 2 "u"	0.000535235	0.317596	[-1.18996, 1.06915]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

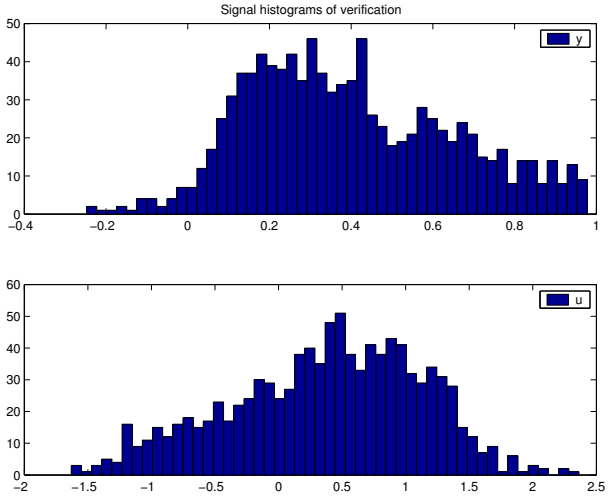
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-1, 1]	[-0.016648, 0.809028]	1
2 "u"	[-1000, 1000]	[-0.871588, 1.6102]	1

Sample simulation signals :



Simulation signals histogram :



2.8 Conclusion

The identification part of processing found a mixture of 8 components. The components are very similar and close to the parameters of the model used for generating of the identification data. The controller designed works well according to the cerification.

Chapter 3

Disputace

Experiment: exp_d0a - Pavel Ettler - Disputace

Author	:	Pavel Ettler X Ludvik Tesar X Mirek Novak
Contact	:	adapt@utia.cas.cz
Address	:	AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic
Basic references	:	
Source texts	:	exp_d0a
Project	:	Designer

3.1 Aims of the study

Experiment "disputace" is a simple testing system created by Pavel Ettler to test the Mixtools toolbox.

This study consist of one main experiment, performed for the situation described in the following text. In the conclusion several modifications of the experiment are discussed without giving the whole report of the computation. Only important differences of the initial settings and final result are given.

The particular modification concerns the different process disturbance in the identification data, non-zero setpoint, and change of the parameters of identified system during the simulation.

3.2 Description of the study

System used for generating identification data and for verification is:

$$y_t = ay_{t-1} + bu_{t-1} + c_1 + c_e e_t, \quad e_t \sim N(0,1)$$

where $a = -0.8$, $b = 0.2$, $c_1 = 5$, and $c_e = 0.001$.

3.3 Data

For generating identification data the inputs are

$$u_t \sim N(c_2, c_u^2)$$

where $c_2 = -24.619$ and $c_u = 0.1$. Number of identification data samples is 100.

3.4 Processing

Whole experiment files are kept under svn in `mixtools/jobcontrol/examples/disputace`. Data used for identification are generated by the script "d0a.m". To run the experiment:
1) generate the identification data by the "d0a.m" script, unless it was done before.
2) call "prodini" to initialize mixtools
3) call `jobproceed(exp_d0a)`, where `exp_d0a.m` is a function generating the Job description.
The other used function is "verifyfun.m" used for simulating model in the verification phase of controller design.

3.5 Description

3.5.1 Experiment definition

```
jobname    = 'exp_d0a';  % name of experiment  ...  no spaces!
% choose short 'jobname' that serves as name root for temporary and saved files
authorname = 'Pavel Ettler X Ludvik Tesar X Mirek Novak';  % author of experiment
email      = 'adapt@utia.cas.cz';  % E-mail of author of experiment
address    = 'AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic';  %
   author's address
references = '';  % references to literature
project    = 'ProDaCTool';  % project name
desc       = 'Pavel Ettler - Disputace';  % description of experiment printed
   in protocol
debug      = 0;    % debug level determining information during evaluations
seed       = [];  % random seed ([] -> leave the seed, -1 -> randomize by timer)

steps    = [1, 1, 1, 1, 1, 1, 1, 1];  %          % Steps to be performed (1/0)
   = (yes/no)
```

3.5.2 Data description

```
datafile     = 'd0a';  % filename with full path specifying the mat file with
   data
varname      = 'DATA';  % variable name of data in 'datafile'
transpose    = 0;   % transpose data matrix to have channels to rows (1=yes
   0=no)
rescale_data = 1;   % rescale new data (normaly this is necesary every time
   the new data is given, but it does no harm to do it every time)  (1=yes 0=no)
reset_chns   = 0;   % reset selected channels (normally this is needed only
   once, in the beginning, and if new data does not have anything to do with
   the old data)  (1=yes 0=no)

chns         = [1, 2];  % modelled channels
pr_chns      = -1;  % vector of numbers of channels from which original data
   plots are printed to protocol (-1 means all channels)
pr_merge     = 1;  % whether original data plots in protocol are merged or not
   (0=separated, 1=merged, 2=tiled).
used_data    = -1;   % % At the moment, there are 100 data samples
```

3.5.3 Channels description

Description of the channel 1

```
chn_name       = 'y';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 0;  % available for control
chn_prty       = 0;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-0.755176, 0.891778];  % expected physical range [min,max]
chn_drange    = [-0.755176;
0.891778];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 1;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]

type = '';
```

Description of the channel 2

```
chn_name       = 'u';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 1;  % available for control
chn_prty       = 0;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-26.8488, -21.9188];  % expected physical range [min,max]
chn_drange    = [-26.8488;
-21.9188];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 2;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
```

```
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = []; % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = []; % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = []; % time constant [uchn, tclow, tchigh]

type = '';
```

3.5.4 Prior information

```
ncom    = 2;       % the number of components
ord     = 1;       % order of the richest regressor
diacove = 0.0001;  % diagonal of noise covariancecove
diaCth  = 10000;   % diagonal of par.  covariance
dfm     = 5;     % degrees of freedom of factors
dfcs    = [5, 5];  % degrees of freedom of components

doflattening = 0;  % indicates whether the initial mixture should be flattened
```

3.5.5 Mixture initialization

```
SingleOnly  = 0;       % boolean flag whether to skip mixinit and calculate
   just MixSingle
opt     = 'p';    % iterative estimation method (p | q | b | f | Q | P)
niter   = 10;    % maximum number of iterations
frg     = 0.999999;    % forgetting factor
```

3.5.6 Mixture estimation

```
opt         = 'p';       % iterative estimation method (p | q | b | f | Q |
   P)
niter       = 40;           % maximum number of iterations
frg         = 0.999999;           % forgetting factor
frgEstType  = 0;           % estimation type of forgetting factor (0-none, 1-zero
   alt, 2-prev estimate)
frgEstGrid    = [1, 0.99, 0.983362, 0.972317, 0.953941, 0.923366, 0.872495,
   0.787855, 0.647029, 0.412721, 0.022876];  %
% estimation grid of forgetting factor
```

3.5.7 Mixture validation

```
nsteps  = 1;     % the prediction is made nsteps ahead
pchns    = [1, 2];  % channels to be predicted
cchns    = [];  % channels in condition
tstart  = 1;     % starting time determining the part of data used in validation
tend  = 100;  %   ending time determining the part of data used in validation
epss  = 1;     % (1/0) = (produce/do not produce) encapsulated postscript plots
pauses  = 0.5;     % pause in seconds after each plot set
plots    = [1, 1, 1, 1];  % [show cluster plot , show time plot, mixture plot,
   histogram], where (1/0)=(y/n)
segments    = 2;     % number of segments for segmentation test (0 means perform
   no segm.  test).  These tests take a very long time.
```

```
stoperr     = 1;     % stop on unsuccessful validation.  (1/0)=(y/n)
alt         = 1;     % perform validation using alternative forgetting estimation
   test (takes very long time, 0=do not run, 1=run this test).
```

3.5.8 User ideal

```
method = 't';  % method determining the user target (t | d | z)
% t ...  User defined (initialized by target.m) - this is default option
% d ...  Designer is used
% z ...  User defined (initialized by zeros)
userSetpointEths   = [0.068301, -24.3838];  % user target component Eths
userSetpointCoves  = [0.678114, 6.07622];  % user target component Coves
userSetpointCorrect  = 1;  % do the user target component Correction ?  (0=no,1=yes)
incremental  = 0;  % use incremental penalization controller ?  (0=no,1=yes)
```

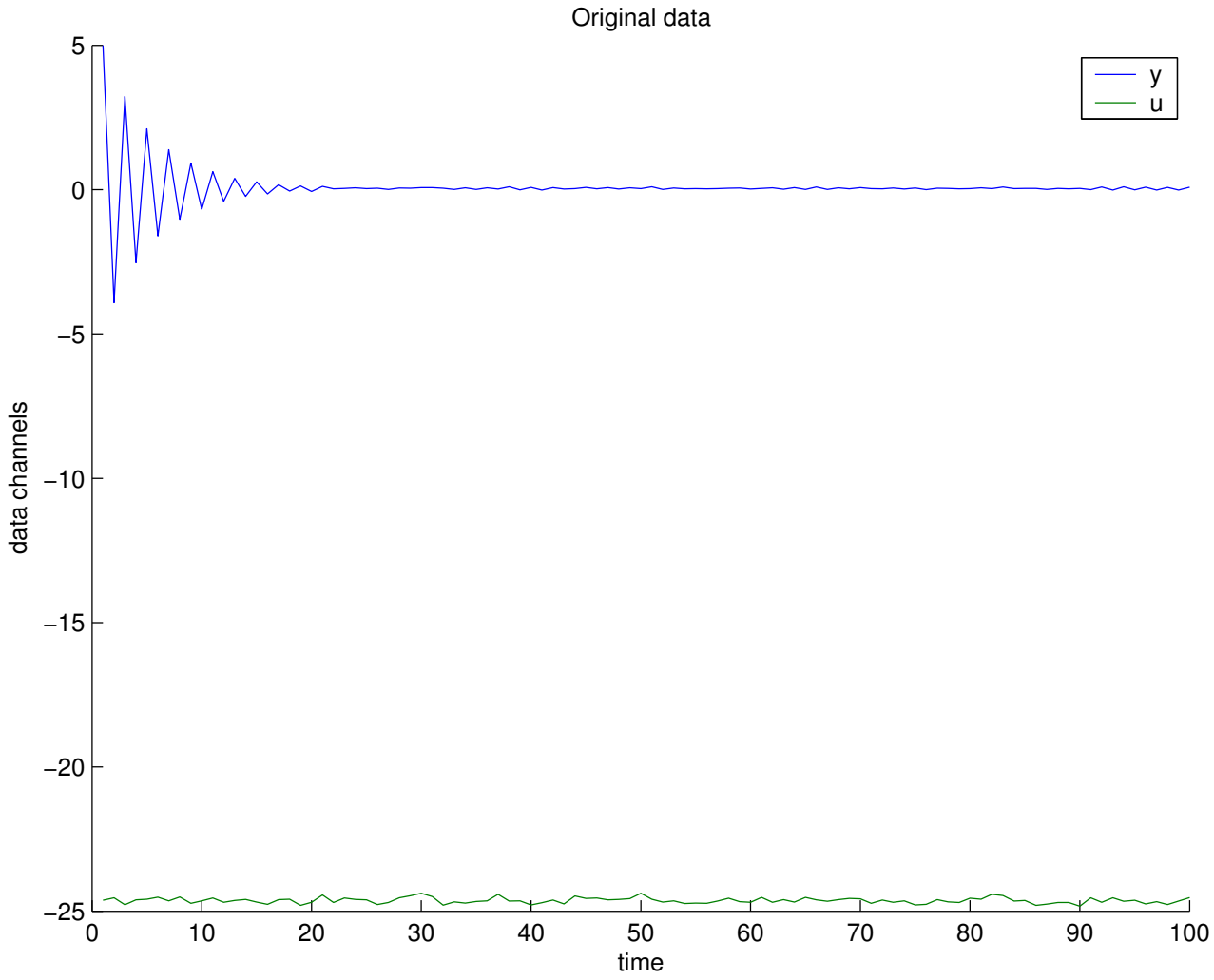
3.5.9 Design

```
designtype = 'i';  % design type (a | i | s)
% a ...  academic design
% i ...  industrial design
% s ...  simultaneous design
horizon    = 100;  % horizon for evaluation of KLD
ufc        = [];  % ufcgen(aMix, aMixu) is used if ufc=[]
```

3.5.10 Verification

```
method = 't';  % verification method (t | d)
% t ...  verification of controller designed by step 7
% d ...  verification of controller designed by UserIdeal with Designer in step
   6
type = 'matlabfunction'; % simulation type (none,simulink,mixture)
matlabfunction = 'verifyfun';  % name of function used for verification
smlsimlength = 100; % simulation length for verification
```

3.5.11 Original Data Plots



3.6 Results

3.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	7.129e+02	(the bigger the better)
Test of validity of the model:	1	(1=O.K.,0=bad)
Relative SE of pred.err:	[0.000138587, 0.0970496]	
	(standard error of ep relative to std of data)	
Test of whiteness:	[0.50254, 0.849859]	
	(sum of correlations with delayed predictions)	
Test of the condition number:	[693766, 17.7015, 4.56933e+06, 1563.17]	
	(proportion of biggest and smallest eigenvalue of data matrix)	

Elementary statistics for the channel y:

	MIN	MAX	MEAN	MEDIAN	STD
data	-3.92324	3.23013	0.0189135	0.0400945	0.675184
differences	-5.76881	7.15337	0.0408952	0.00203752	1.22498
predictions	-3.9236	3.23116	0.0188592	0.0393559	0.675261
errors of prediction	-0.00250489	0.0018847	5.43828e-05	4.26536e-05	0.000934169

Elementary statistics for the channel u:

	MIN	MAX	MEAN	MEDIAN	STD
data	-24.8244	-24.3773	-24.6301	-24.6411	0.0992905
differences	-0.303002	0.291414	6.42917e-05	0.015821	0.128925
predictions	-24.6758	-24.5521	-24.6287	-24.6306	0.0176447
errors of prediction	-0.185144	0.239059	-0.00137233	-0.00347307	0.0963561

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	1.41e-06	0.000938	0.254
component 2	1.02e-06	0.000933	0.746

Mixture Factors

This mixture has 2 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=27.642703, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=1.41382e-06, dfm=20.793658

delays →		0	1
offset			
1	y		-0.800413
2	u	-0.0214245	0.608542

   Factor 2, modelled channnel: 2, called 'u', cove=0.000937939, dfm=18.297114

delays →		0	1
offset			
1	y		
2	u		0.557854

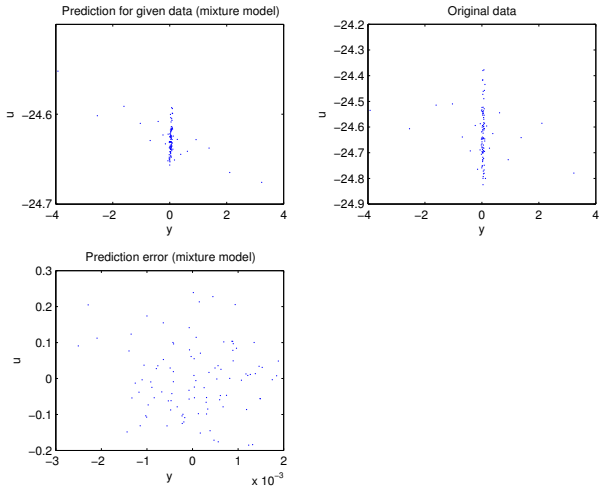
2. component, dfcs=81.342068, factors:
   Factor 3, modelled channnel: 1, called 'y', cove=1.02148e-06, dfm=62.991890

delays →		0	1
offset			
1	y		-0.799778
2	u		0.596059

   Factor 4, modelled channnel: 2, called 'u', cove=0.00093322, dfm=58.655274

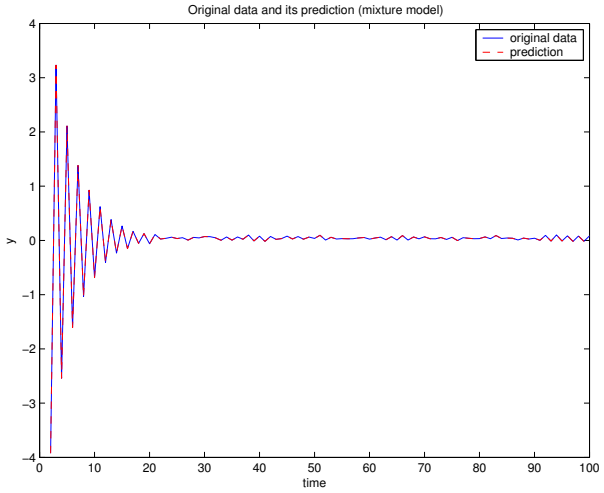
delays →		0	1
offset			-0.114104
1	y		0.00680154
2	u		

Cluster plots:
Graph of prediction, original data, and prediction error for "y" and "u" channels:

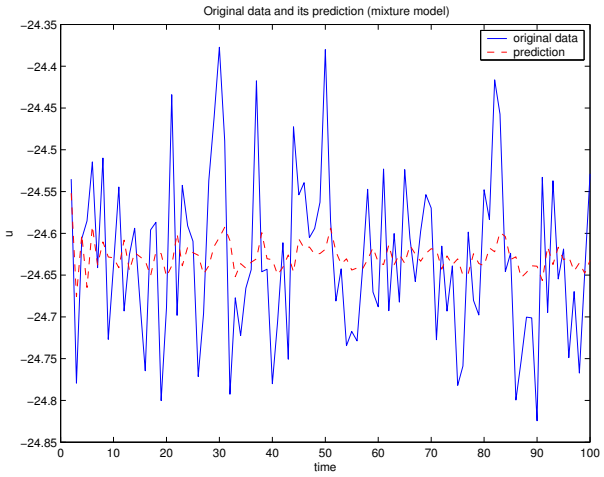


Time plots:

Prediction, original data, and prediction errors plot for channel "y":

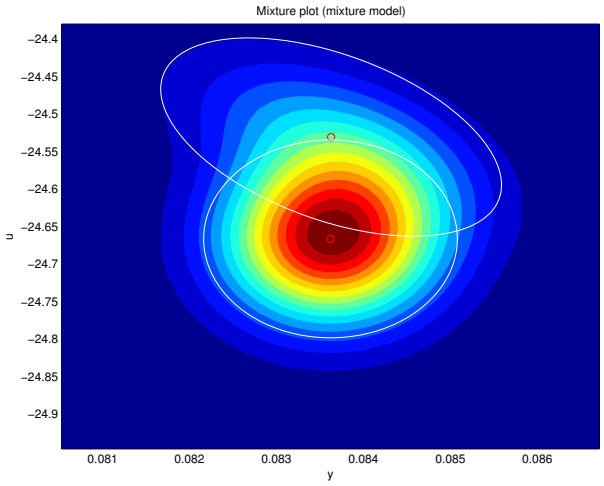


Prediction, original data, and prediction errors plot for channel "u":



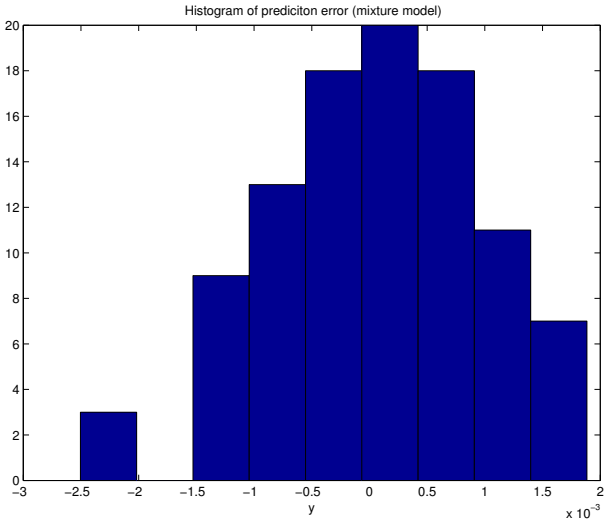
Mixture plots:

Mixture plot for channels "y" and "u":

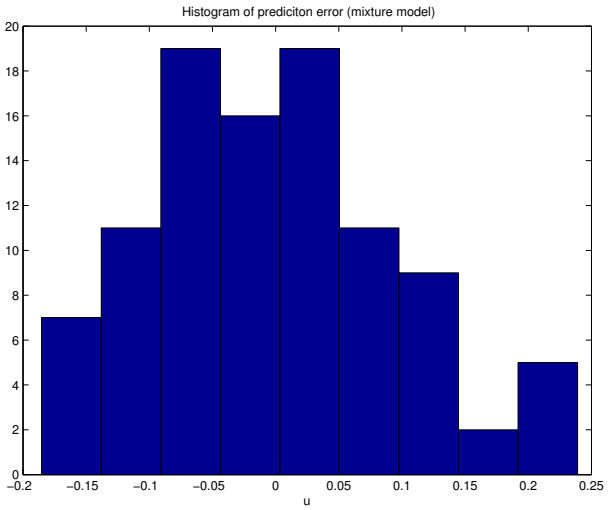


Histgram plots:

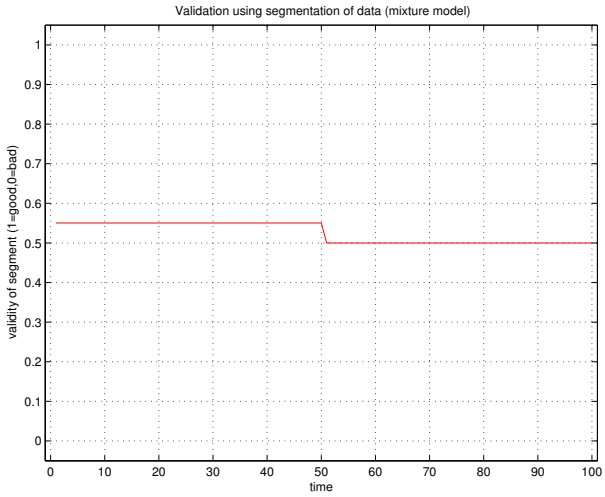
Histogram plot for channel ”y”:



Histogram plot for channel ”u”:



Segment validation (red line: 1=valid, 0=not valid). There are 2 segments, each of the size 50:



3.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	7.080e+02	(the bigger the better)
Test of validity of the model:	1	(1=O.K.,0=bad)
Relative SE of pred.err:	[0.000138036, 0.0999949]	
	(standard error of ep relative to std of data)	
Test of whiteness:	[0.49841, 1.06556]	
	(sum of correlations with delayed predictions)	
Test of the condition number:	[2.67143e+06, 635.138]	
	(proportion of biggest and smallest eigenvalue of data matrix)	

Elementary statistics for the channel y:

	MIN	MAX	MEAN	MEDIAN	STD
data	-3.92324	3.23013	0.0189135	0.0400945	0.675184
differences	-5.76881	7.15337	0.0408952	0.00203752	1.22498
predictions	-3.92313	3.23076	0.0188473	0.0393364	0.675178
errors of prediction	-0.00248606	0.00189673	6.62604e-05	7.59415e-05	0.000929665

Elementary statistics for the channel u:

	MIN	MAX	MEAN	MEDIAN	STD
data	-24.8244	-24.3773	-24.6301	-24.6411	0.0992905
differences	-0.303002	0.291414	6.42917e-05	0.015821	0.128925
predictions	-24.6301	-24.6301	-24.6301	-24.6301	1.7854e-14
errors of prediction	-0.194362	0.252785	3.09209e-07	-0.0110696	0.0992905

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	1.32e-06	0.00167	1

Mixture Factors

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the

first row employs the offset.

1. component, dfcs=108.984761, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=1.32113e-06, dfm=103.985289

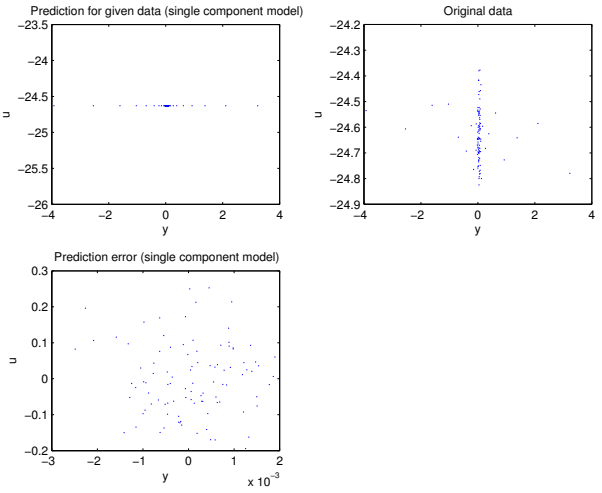
delays →		0	1
offset			
1	y		-0.79984
2	u		0.596338

   Factor 2, modelled channnel: 2, called 'u', cove=0.00167309, dfm=103.985289

delays →		0	1
offset			-0.0999061
1	y		
2	u		

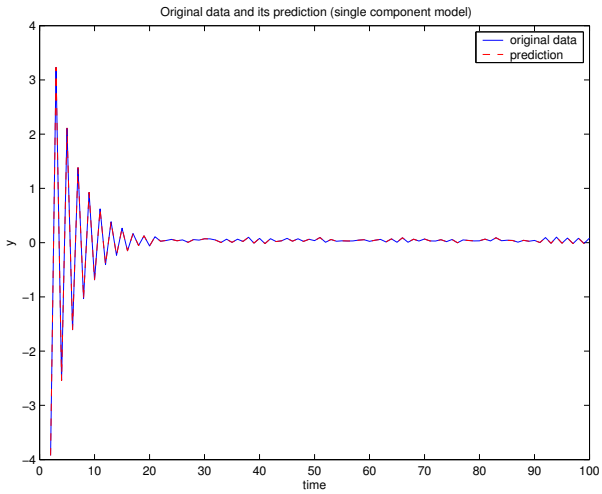
Cluster plots:

Graph of prediction, original data, and prediction error for "y" and "u" channels:

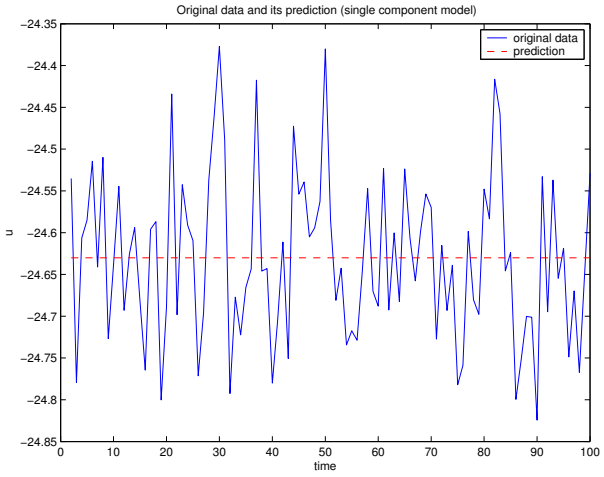


Time plots:

Prediction, original data, and prediction errors plot for channel "y":

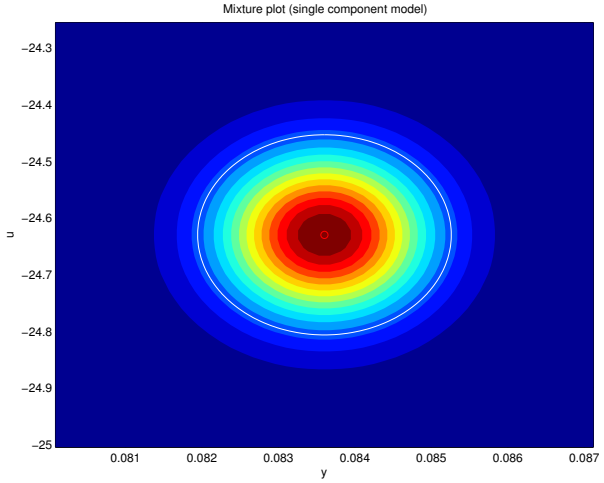


Prediction, original data, and prediction errors plot for channel "u":



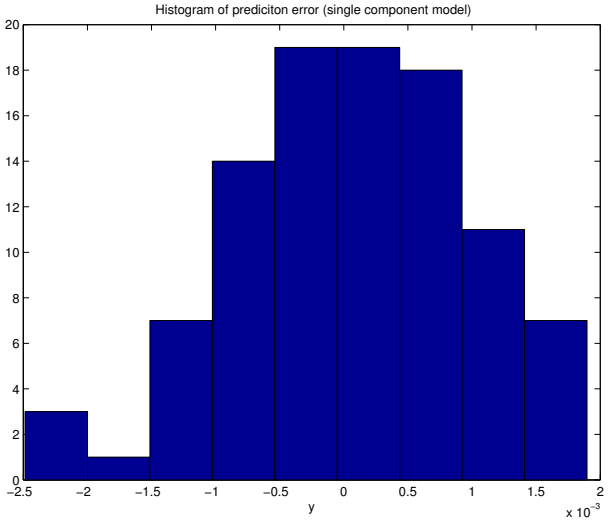
Mixture plots:

Mixture plot for channels ”y” and ”u”:

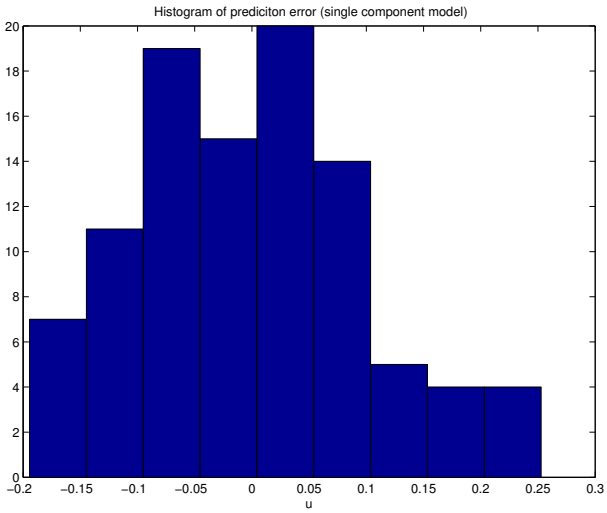


Histgram plots:

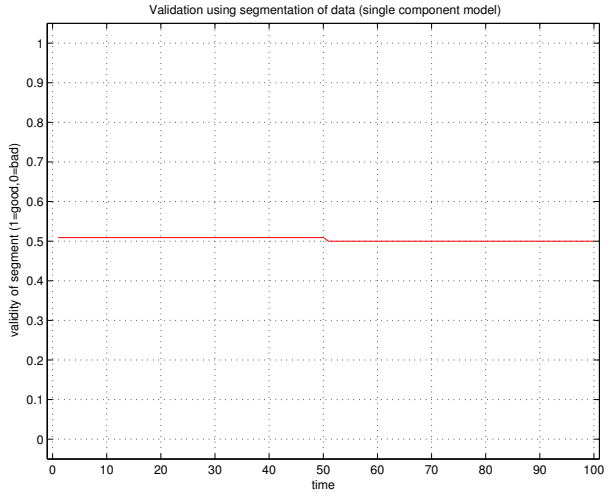
Histogram plot for channel ”y”:



Histogram plot for channel ”u”:



Segment validation (red line: 1=valid, 0=not valid). There are 2 segments, each of the size 50:



3.6.3 User Ideal Mixture

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1.000000, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=1.32113e-06, dfm=1.000000

delays →		0	1
offset			0
1	y		
2	u		

   Factor 2, modelled channnel: 2, called 'u', cove=0.00167309, dfm=1.000000

delays →		0	1
offset			0
1	y		
2	u		

3.6.4 Controller Mixture

This mixture has 2 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=27.642703, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=1.41382e-06, dfm=20.793658

delays →		0	1
offset			
1	y		-0.800413
2	u	-0.0214245	0.608542

   Factor 2, modelled channnel: 2, called 'u', cove=3.66417e-06, dfm=1.000000

delays →		0	1
offset			-0
1	y		-1.06949
2	u		0.801361

2. component, dfcs=81.342068, factors:
   Factor 3, modelled channnel: 1, called 'y', cove=1.02148e-06, dfm=62.991890

delays →		0	1
offset			
1	y		-0.799778
2	u		0.596059

   Factor 4, modelled channnel: 2, called 'u', cove=3.66417e-06, dfm=1.000000

delays →		0	1
offset			-0
1	y		-1.06949
2	u		0.801361

3.7 Experimental Controller Verification Results

Elementary statistics for simulated channels:

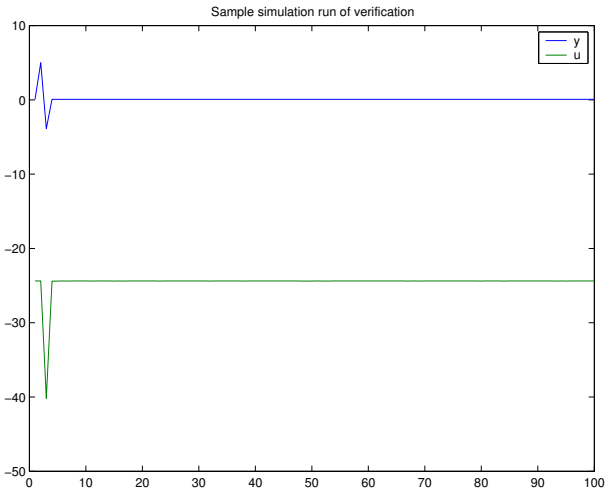
channel	mean	variance	range
1 "y"	0.0781612	0.634776	[-0.965952, 1.12227]
Δ 1 "y"	-9.39204e-18	1.10061	[-8.87845, 4.93281]
2 "u"	-24.5426	1.5791	[-27.14, -21.9453]
Δ 2 "u"	-1.66692e-05	2.25432	[-15.7919, 15.7685]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

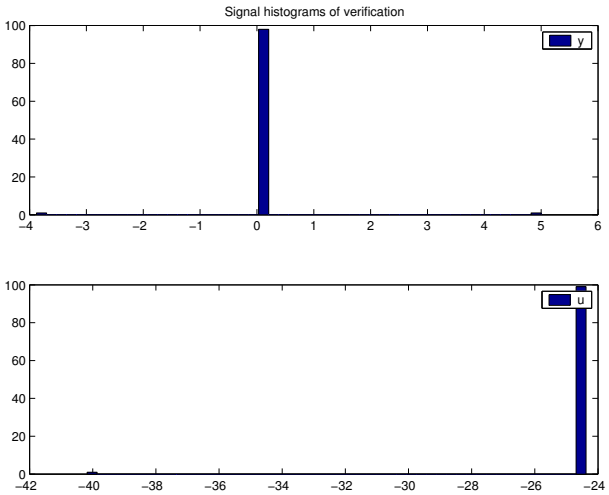
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-0.755176, 0.891778]	[-0.965952, 1.12227]	0.98
2 "u"	[-26.8488, -21.9188]	[-27.14, -21.9453]	0.99

Sample simulation signals :



Simulation signals histogram :



3.8 Conclusion

The system identification found a resulting model as a mixture of two components. The components are very similar in parameters. Thus they behave like one component in the controller design.

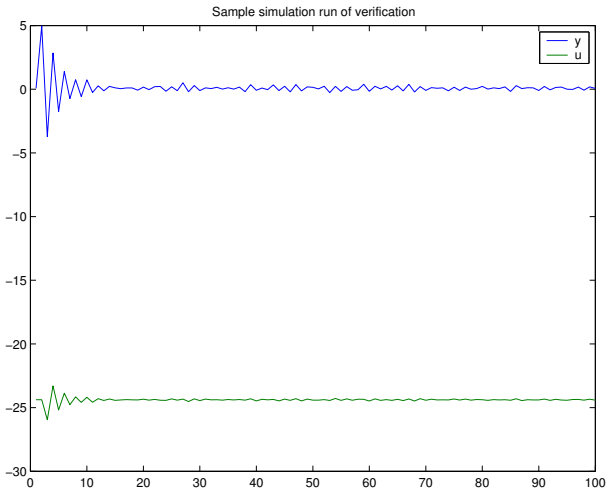
The resulting controller for the given setting, called `exp_d0a`, works almost perfectly up to an initial activity caused by non-stationary initial conditions.

Now the modification are discussed.

3.8.1 Experiment with higher process noise

The `exp_d0a2` experiment, is same as `exp_d0a`, but uses higher system noise of $c_e = 0.1$. The identification correctly identified higher disturbance, and the identified parameters are close to the original case. The designed controller performs well for this case, up more noisy control signals. The verification run is shown in the following figure.

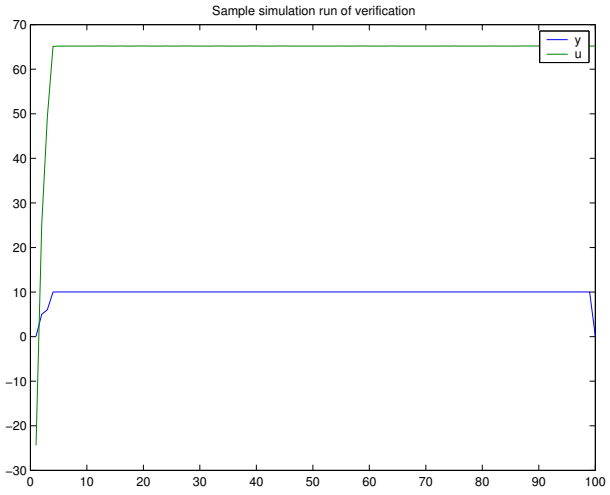
Sample simulation signals :



3.8.2 Experiment with a non-zero setpoint

The `exp_d0a3` experiment, is same as `exp_d0a`, but uses nonzero setpoint on output of value 10. The only difference is in the designed controller, which maintains the setpoint. The simulation of controller verification shown in the following figure confirms the proper operation.

Sample simulation signals :

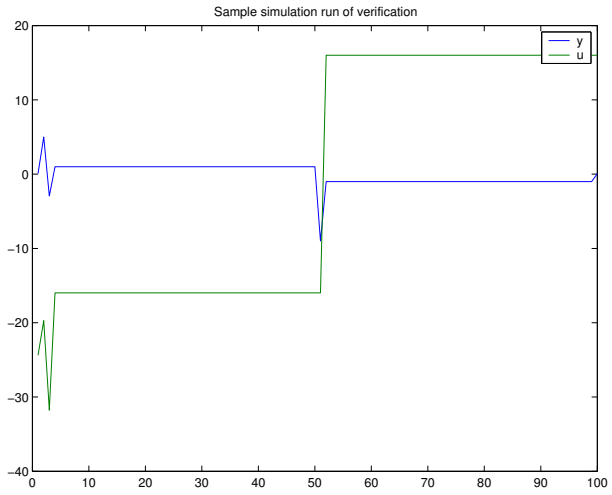


3.8.3 Experiment with offset change in the middle of the simulation

The `exp_d0b` experiment, is same as `exp_d0a`, but the offset in system simulation is changed in half of simulation from $c_1 = 5$ to $c_1 = -1$. The disturbance is turned off by setting $c_e = 0$.

The identification phase found correctly two components distinguished by the offset parameter. The controller in the current phase of the development is controls to the an averaged model from the two components. Therefore the verification simulation exhibits positive and negative offsets in the two parts of simulation divided by the offset change of the simulated system. See figure below.

Sample simulation signals :


Sample simulation run of verification
y
u
20
10
0
−10
−20
−30
−40
0
10
20
30
40
50
60
70
80
90
100

Chapter 4

Single axis helicopter

Experiment: `vrt3` - Single axis helicopter - adaptive with alternative forgetting

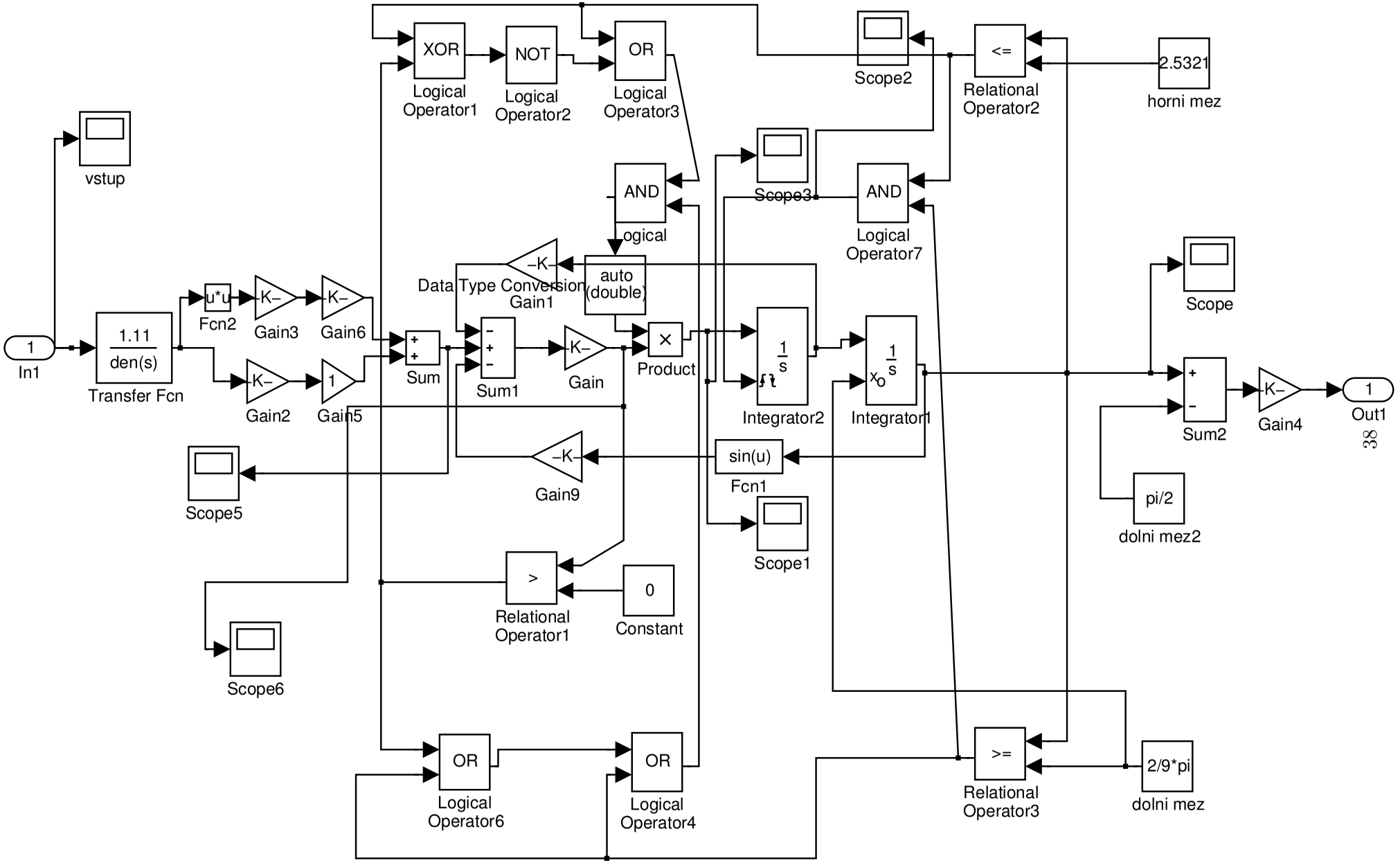
Author	:	Miroslav Novak
Contact	:	`mira@utia.cas.cz`
Address	:	AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic
Basic references	:	
Source texts	:	`vrt3`
Project	:	Designer

4.1 Aims of the study

Experiment `vrt3` is a system modeling a single axis helicopter. The aim of the study is to test designer to create a controller respecting the given constraints placed on the action signal. The experiment "vrt3" uses adaptive controller to track the changing working point of the nonlinear simulink model. For comparison using a nonadaptive controller the result is compared with experiment `vrt1` in the conlcusion of this experiment.

4.2 Description of the study

System used for generating identification data and for verification is contained in a simulink model. The helicopter subsystem is a nonlinear system


XOR
Logical Operator1
NOT
Logical Operator2
OR
Logical Operator3
Scope2
<=
Relational Operator2
2.5321
horni mez
vstup
AND
Logical Operator
Scope3
AND
Logical Operator7
Data Type Conversion Gain1
auto (double)
Scope
1
In1
1.11
den(s)
Transfer Fcn
u*u
Fcn2
Gain3
Gain6
Sum
Sum1
Gain
Product
Integrator2
Integrator1
Sum2
Gain4
1
Out1
Gain2
Gain5
Scope5
Gain9
sin(u)
Fcn1
pi/2
dolni mez2
Scope1
>
Relational Operator1
0
Constant
Scope6
OR
Logical Operator6
OR
Logical Operator4
>=
Relational Operator3
2/9*pi
dolni mez

For more detail see below the simulink file used for data generation.

The task of controller is to follow prescribed step reference trajectory with action signal bounded by interval $[-0.8, 0.8]$ and its increments are bounded by interval $[-0.5, 0.5]$.

4.3 Data

For identification, data are obtained from simulink scheme `dv_genrawdata_vrt_mdl.mdl`. The driving inputs are generated by Band-Limited White Noise generator with power of 0.00025.

4.4 Processing

Whole experiment files are kept under svn in `mixtools/jobcontrol/examples/vrt`. Data used for identification are generated by the script `dv_genrawdata_vrt`. To run the experiment:
1) generate the identification data by the `dv_genrawdata_vrt` script, unless it was done before.
2) call `prodini` to initialize mixtools
3) call `jobproceed(vrt3)`, where `vrt3` is a function generating the Job description.
The other used file is simulink scheme `dv_verify_vrt_sfun.mdl` used for simulating model in the verification phase of controller design.

4.5 Description

4.5.1 Experiment definition

```
jobname    = 'vrt3';  % name of experiment  ...  no spaces!
% choose short 'jobname' that serves as name root for temporary and saved files
authorname = 'Miroslav Novak';  % author of experiment
email      = 'mira@utia.cas.cz';  % E-mail of author of experiment
address    = 'AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic';  %
   author's address
references = '';  % references to literature
project    = 'Designer';  % project name
desc       = 'Single axis helicopter - adaptive with alternative forgetting';
   % description of experiment printed in protocol
debug      = 0;    % debug level determining information during evaluations
seed       = [];  % random seed ([] -> leave the seed, -1 -> randomize by timer)

steps    = [1, 1, 1, 1, 1, 1, 0, 1];  %           % Steps to be performed (1/0)
   = (yes/no)
```

4.5.2 Data description

```
datafile     = 'dv_genrawdata_vrt';  % filename with full path specifying the
   mat file with data
varname      = 'rawdata';  % variable name of data in 'datafile'
transpose    = 1;   % transpose data matrix to have channels to rows (1=yes
   0=no)
rescale_data = 1;   % rescale new data (normaly this is necesary every time
   the new data is given, but it does no harm to do it every time)  (1=yes 0=no)
reset_chns   = 0;   % reset selected channels (normally this is needed only
   once, in the beginning, and if new data does not have anything to do with
   the old data)  (1=yes 0=no)
```

```
chns         = [1, 2];  % modelled channels
pr_chns      = -1;  % vector of numbers of channels from which original data
   plots are printed to protocol (-1 means all channels)
pr_merge     = 1;  % whether original data plots in protocol are merged or not
   (0=separated, 1=merged, 2=tiled).
used_data    = -1;   % % At the moment, there are 301 data samples
```

4.5.3 Channels description

Description of the channel 1

```
chn_name       = 'y';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 0;  % available for control
chn_prty       = 0.5;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-0.160896, 0.031734];  % expected physical range [min,max]
chn_drange    = [-1000;
1000];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 1;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]

type = 'steps';
stepsdata    = [1, -0.25, 50, -0.2, 125, -0.1, 200, 0, 250, 0.1];  % steps definition
   [starting time, value, starting time, value, ....]
```

Description of the channel 2

```
chn_name       = 'u';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 1;  % available for control
chn_prty       = 0.5;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-0.0124424, 0.147368];  % expected physical range [min,max]
chn_drange    = [-0.8;
0.8];  % desired range [min,max]
chn_irange    = [-0.5;
0.5];  % desired range of increments [min,max]
```

```
chn_preinfo   = 'olymedian', 'c', 2;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]

type = '';
```

4.5.4 Prior information

```
ncom    = 1;       % the number of components
ord     = 3;       % order of the richest regressor
diacove = 0.0001;  % diagonal of noise covariancecove
diaCth  = 10000;   % diagonal of par.  covariance
dfm     = 30.1;     % degrees of freedom of factors
dfcs    = 30.1;  % degrees of freedom of components

doflattening = 0;  % indicates whether the initial mixture should be flattened
```

4.5.5 Mixture initialization

```
SingleOnly  = 1;       % boolean flag whether to skip mixinit and calculate
   just MixSingle
opt     = 'p';    % iterative estimation method (p | q | b | f | Q | P)
niter   = 10;    % maximum number of iterations
frg     = 0.999999;    % forgetting factor
```

4.5.6 Mixture estimation

```
opt         = 'p';       % iterative estimation method (p | q | b | f | Q |
   P)
niter       = 40;           % maximum number of iterations
frg         = 0.999999;           % forgetting factor
frgEstType  = 0;           % estimation type of forgetting factor (0-none, 1-zero
   alt, 2-prev estimate)
frgEstGrid    = [1, 0.99, 0.983362, 0.972317, 0.953941, 0.923366, 0.872495,
   0.787855, 0.647029, 0.412721, 0.022876];  %
% estimation grid of forgetting factor
```

4.5.7 Mixture validation

```
nsteps  = 1;     % the prediction is made nsteps ahead
```

```
pchns    = [1, 2];  % channels to be predicted
cchns    = [];  % channels in condition
tstart  = 1;     % starting time determining the part of data used in validation
tend  = 301;  %   ending time determining the part of data used in validation
epss  = 1;     % (1/0) = (produce/do not produce) encapsulated postscript plots
pauses  = 0;     % pause in seconds after each plot set
plots    = [1, 1, 1, 1];  % [show cluster plot , show time plot, mixture plot,
   histogram], where (1/0)=(y/n)
segments    = 0;     % number of segments for segmentation test (0 means perform
   no segm.  test).  These tests take a very long time.
stoperr     = 1;     % stop on unsuccessful validation.  (1/0)=(y/n)
alt         = 1;     % perform validation using alternative forgetting estimation
   test (takes very long time, 0=do not run, 1=run this test).
```

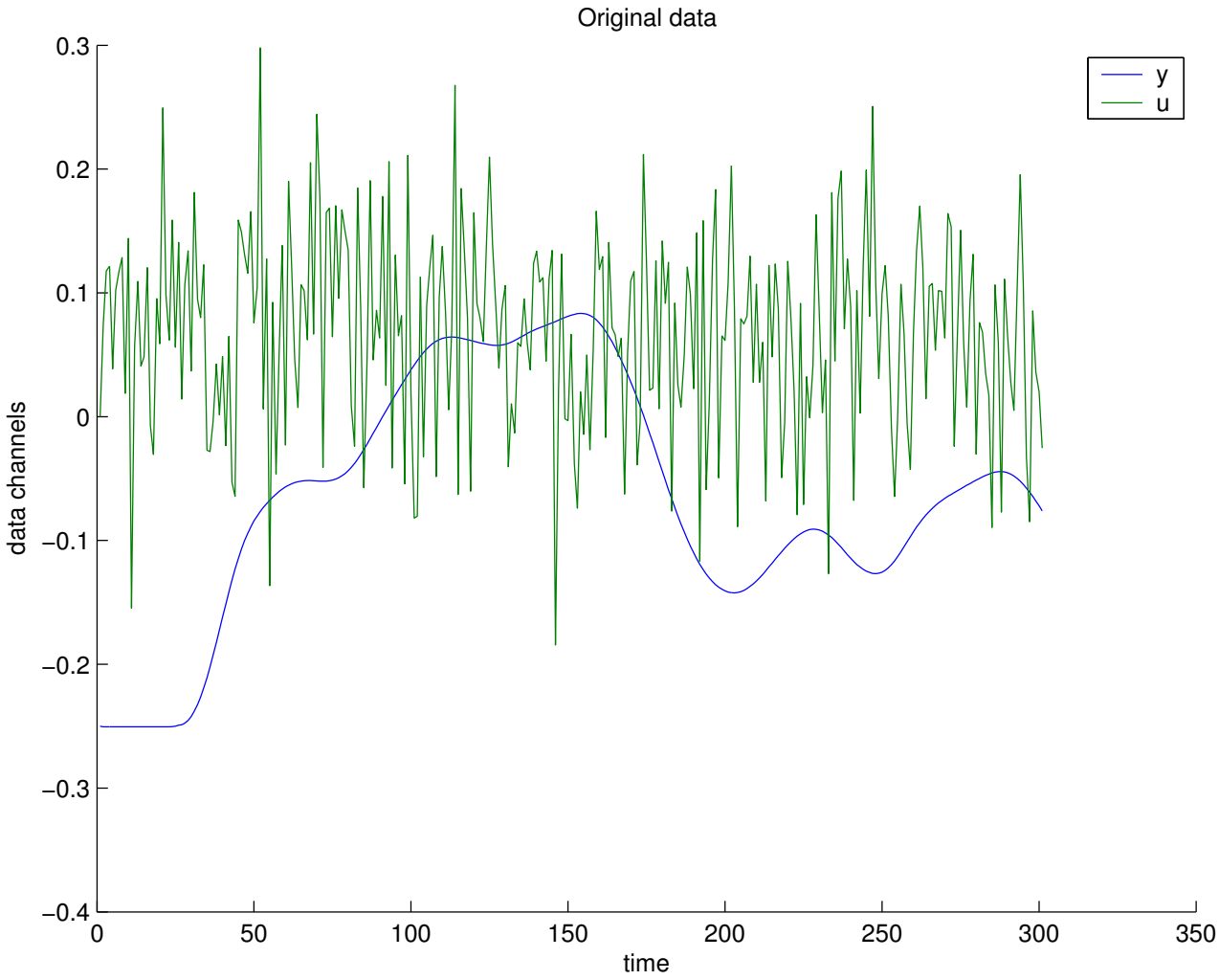
4.5.8 User ideal

```
method = 'd';  % method determining the user target (t | d | z)
% t ...  User defined (initialized by target.m) - this is default option
% d ...  Designer is used
% z ...  User defined (initialized by zeros)
typloss    = 'max'; % (prob/max) probability of constraints
constrtol  = 0.1; % probability tolerance for constraint violation, used only
   if typlos==prob
simlength  = 500; % simulation length for tuning
adaptive   = 2; % adaptive tuning flag, 0 - NONadaptive, 1 - EXP frg.,  2 -
   ALTernative frg.
horizon    = [50, -1];  % horizon with IST
initialData = [-0.25, -0.25, -0.25, -0.25, -0.25;
0, 0, 0, 0, 0];  % initial data for simulation
penal     = [];  % initial penalization, [] = guess automaticaly
penalY     = [];  % custom outputs penalization, [] = use default
```

4.5.9 Verification

```
method = 'd';  % verification method (t | d)
% t ...  verification of controller designed by step 7
% d ...  verification of controller designed by UserIdeal with Designer in step
   6
type = 'simulink'; % simulation type (none,simulink,mixture)
simulink_model = 'dv_verify_vrt_sfun';   % simulink model for verification
smlsimlength = 1000; % simulation length for verification
smlperiode   = 0.04; % sampling periode of simulink scheme for verification
```

4.5.10 Original Data Plots



4.5.11 Results

4.5.12 Results of Single Component Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	1.200e+03	(the bigger the better)
Test of validity of the model:	0	(1=O.K.,0=bad)
Relative SE of pred.err:	[6.07404e-05, 0.0578312]	
	(standard error of ep relative to std of data)	
Test of whiteness:	[0.768068, 0.303228]	
	(sum of correlations with delayed predictions)	
Test of the condition number:	[7.07231e+07, 1.0031]	
	(proportion of biggest and smallest eigenvalue of data matrix)	

Elementary statistics for the channel y:

	MIN	MAX	MEAN	MEDIAN	STD
data	-0.250434	0.0834047	-0.0627114	-0.061111	0.0949653
differences	-0.00765634	0.0103352	0.000587036	0.000822995	0.00350103
predictions	-0.250488	0.0834297	-0.0627132	-0.0611203	0.0949652
errors of prediction	-0.000895219	0.00122141	1.8298e-06	-1.56576e-06	9.97258e-05

Elementary statistics for the channel u:

	MIN	MAX	MEAN	MEDIAN	STD
data	-0.184454	0.297897	0.0674994	0.0733787	0.0801583
differences	-0.330471	0.307715	-0.000494428	-0.00556854	0.115269
predictions	0.0674988	0.0674988	0.0674988	0.0674988	2.64121e-16
errors of prediction	-0.251953	0.230398	5.65478e-07	0.00587991	0.0801583

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	1.15e-06	1.08	1

Mixture Factors

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=327.962201, factors:
   Factor 1, modelled channnel: 1, called 'y', cove=1.14835e-06, dfm=327.962406

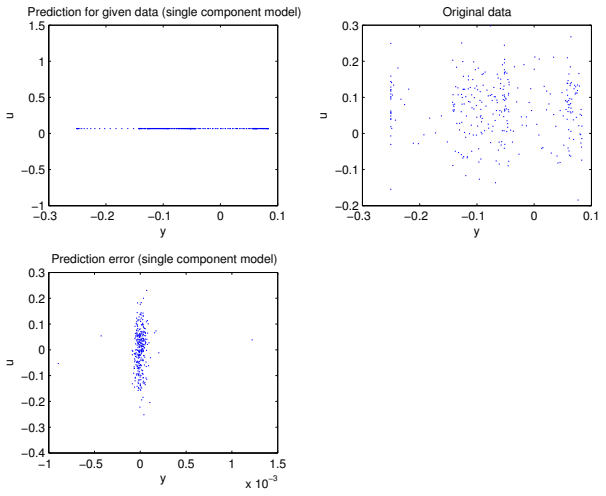
delays →		0	1	2	3
offset					
1	y		2.92574	-2.85979	0.933948
2	u			0.000280427	

   Factor 2, modelled channnel: 2, called 'u', cove=1.07758, dfm=327.962406

delays →		0	1
offset			0.000450346
1	y		
2	u		

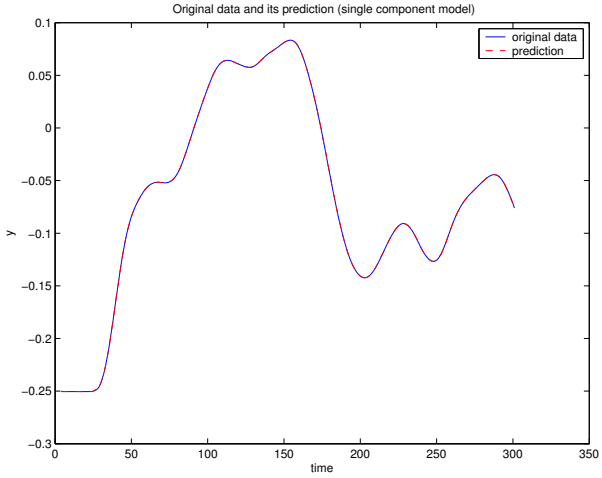
Cluster plots:

Graph of prediction, original data, and prediction error for "y" and "u" channels:

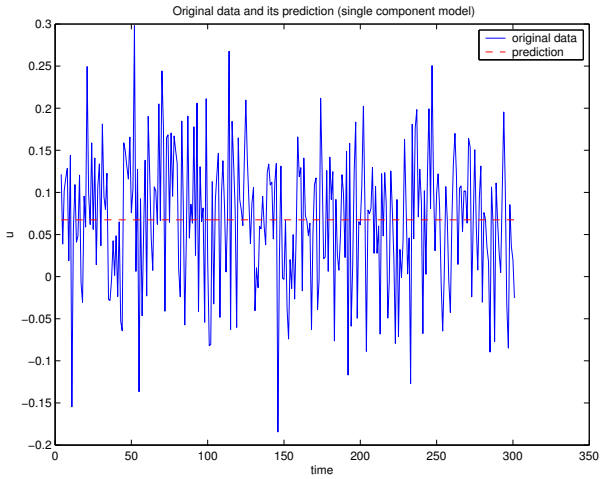


Time plots:

Prediction, original data, and prediction errors plot for channel "y":

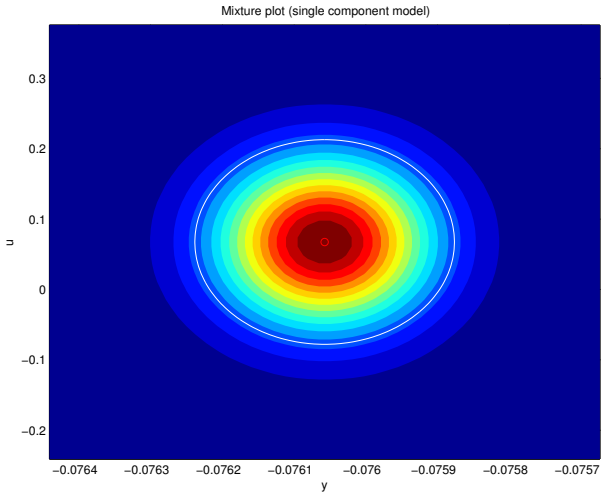


Prediction, original data, and prediction errors plot for channel "u":



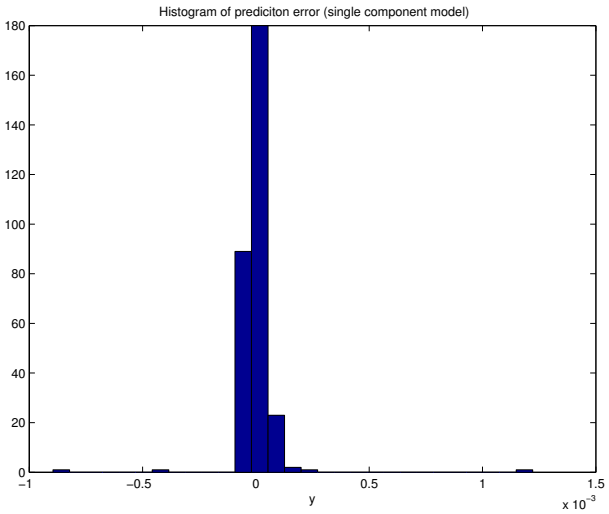
Mixture plots:

Mixture plot for channels "y" and "u":

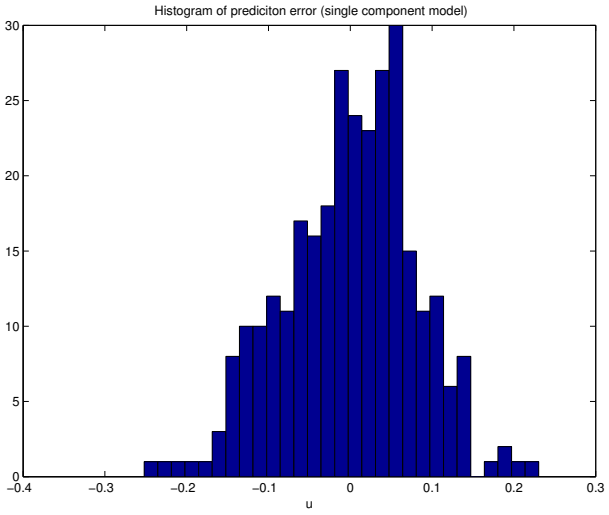


Histgram plots:

Histogram plot for channel "y":



Histogram plot for channel "u":



4.6 Designer Controller Tuning

Final tuning knob value [0.034548 0.013495]

Tuned user ideal model

This mixture has 1 components with 4 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1.000000, factors:
   Factor 1, modelled channnel: 4, called '*channel*4', cove=28.9452, dfm=1.000000

delays →		0	1
offset			
4	*channel*4		1
3	*channel*3		
1	y		
2	u		

   Factor 2, modelled channnel: 3, called '*channel*3', cove=1e+21, dfm=1.000000

delays →		0	1
offset			
4	*channel*4		
3	*channel*3		0
1	y		
2	u		

Factor 3, modelled channnel: 1, called 'y', cove=1, dfm=1.000000

delays →		0	1
offset			
4	*channel*4		
3	*channel*3		1
1	y		
2	u		

Factor 4, modelled channnel: 2, called 'u', cove=74.1028, dfm=1.000000

delays →		0	1
offset			-0.844285
4	*channel*4		
3	*channel*3		
1	y		
2	u		

Optimal controller

This mixture has 1 components with 4 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1.000000, factors:
   Factor 1, modelled channnel: 4, called '*channel*4', cove=0.0001, dfm=1.000000
   (this factor is empty)
   Factor 2, modelled channnel: 3, called '*channel*3', cove=0.0001, dfm=1.000000

delays →		0	1
offset			
4	*channel*4		
3	*channel*3		1
1	y		
2	u		

Factor 3, modelled channnel: 1, called 'y', cove=1.14835e-06, dfm=327.962406

delays →		0	1	2	3
offset					
4	*channel*4				
3	*channel*3				
1	y		2.92574	-2.85979	0.933948
2	u			0.000280427	

Factor 4, modelled channnel: 2, called 'u', cove=13.1601, dfm=1.000000

delays →		0	1	2	3
offset			-0.0501881		
4	*channel*4		0.454655	-0	-0
3	*channel*3		3.59141	-0	-0
1	y		-351.335	656.688	-308.692
2	u		-0.0866379	-0.0926879	-0

4.7 Designer's Controller Prediction Results

Elementary statistics for simulated channels:

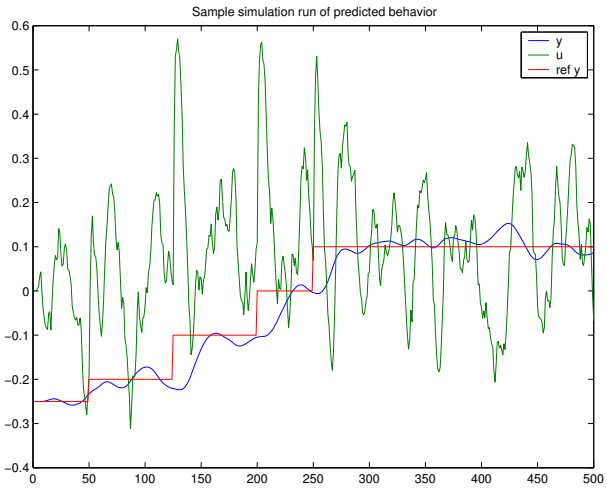
channel	mean	variance	range
1 "y"	-0.0308337	0.142905	[-0.258421, 0.152705]
Δ 1 "y"	0.000676241	0.00235925	[-0.00473306, 0.00775978]
2 "u"	0.0888222	0.155657	[-0.311123, 0.569933]
Δ 2 "u"	-0.000136654	0.0496863	[-0.128173, 0.346803]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

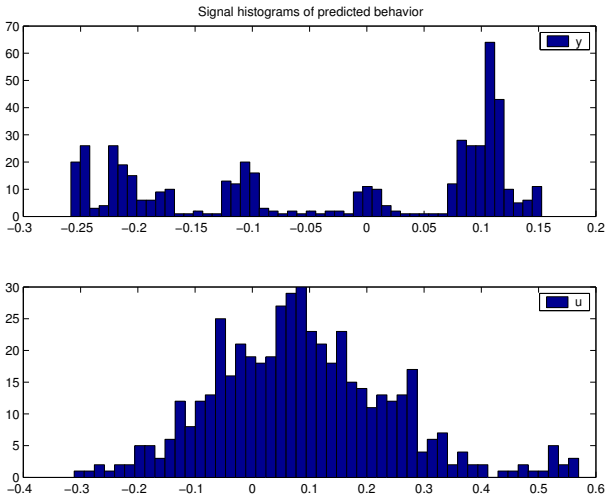
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-1000, 1000]	[-0.258421, 0.152705]	1
2 "u"	[-0.8, 0.8]	[-0.311123, 0.569933]	1
Δ 2 "u"	[-0.5, 0.5]	[-0.128173, 0.346803]	1

Sample simulation signals :



Simulation signals histogram :



4.8 Designer's Noncontrolled Prediction Results

Elementary statistics for simulated channels:

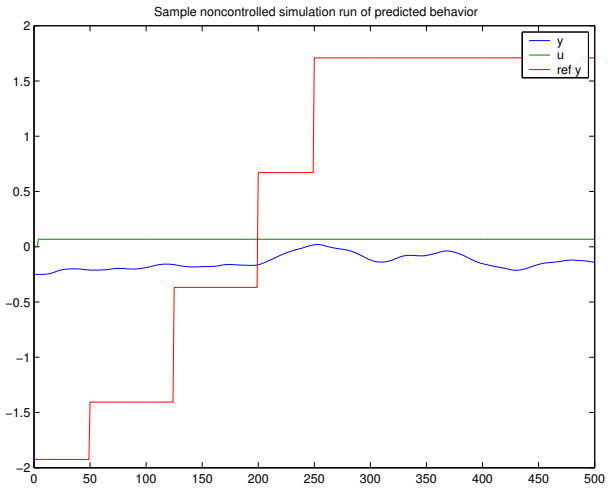
channel	mean	variance	range
1 "y"	-0.136384	0.0668561	[-0.250368, 0.0193115]
Δ 1 "y"	0.000213281	0.00233605	[-0.00527066, 0.00490697]
2 "u"	0.067058	0.00521516	[0, 0.0674628]
Δ 2 "u"	0.000135196	0.00302005	[0, 0.0674628]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

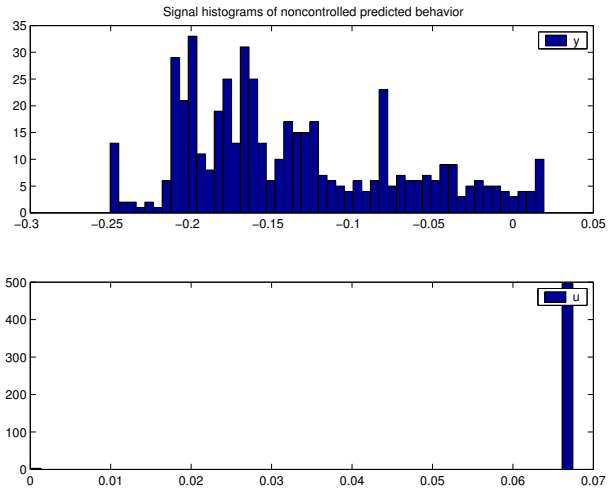
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-1000, 1000]	[-0.250368, 0.0193115]	1
2 "u"	[-0.8, 0.8]	[0, 0.0674628]	1
Δ 2 "u"	[-0.5, 0.5]	[0, 0.0674628]	1

Sample simulation signals :



Simulation signals histogram :



4.9 Experimental Controller Verification Results

Elementary statistics for simulated channels:

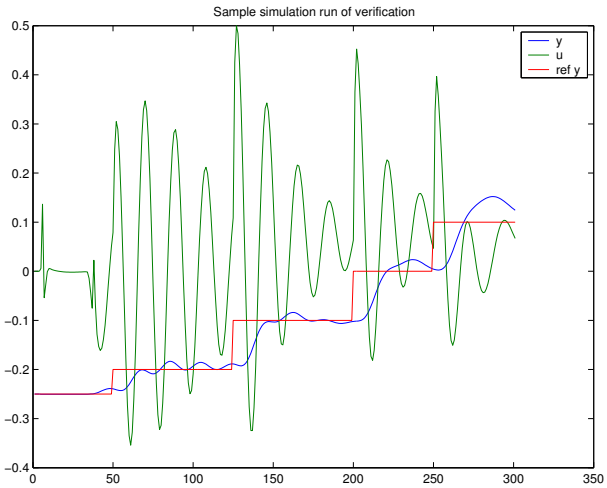
channel	mean	variance	range
1 "y"	-0.106253	0.124201	[-0.250433, 0.152154]
Δ 1 "y"	0.00124743	0.00284903	[-0.00291808, 0.00926728]
2 "u"	0.0399251	0.159137	[-0.354078, 0.5]
Δ 2 "u"	0.000222895	0.0601609	[-0.190487, 0.316587]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

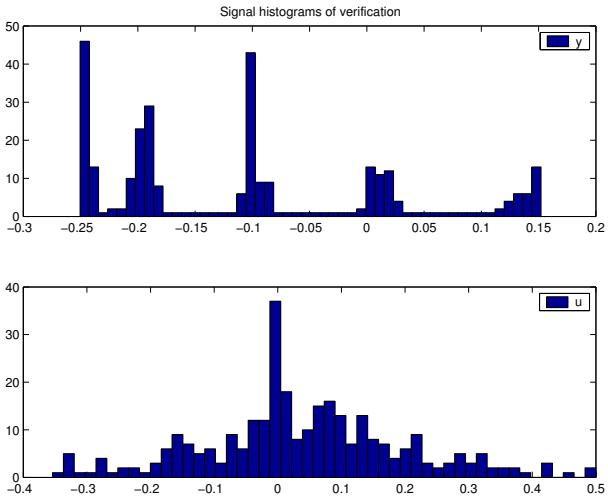
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-1000, 1000]	[-0.250433, 0.152154]	1
2 "u"	[-0.8, 0.8]	[-0.354078, 0.5]	1
Δ 2 "u"	[-0.5, 0.5]	[-0.190487, 0.316587]	1

Sample simulation signals :



Simulation signals histogram :



4.10 Experimental Noncontrolled Verification Results

Elementary statistics for simulated channels:

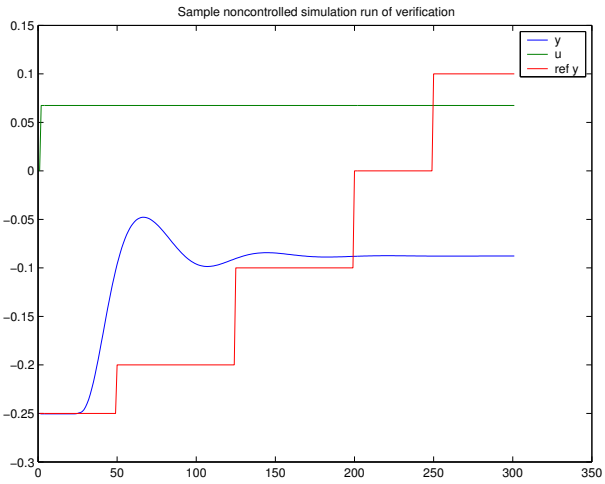
channel	mean	variance	range
1 "y"	-0.106686	0.0546545	[-0.250433, -0.0477858]
Δ 1 "y"	0.000540897	0.00211842	[-0.00201083, 0.00887032]
2 "u"	0.0672387	0.00388849	[0, 0.0674628]
Δ 2 "u"	0.000224876	0.00389497	[0, 0.0674628]

Note: Symbol Δ means increments of the signal, "range" means the minimum a maximum of simulated signal values, "constr.sat." means constraints satisfaction ratio for given channel and constraint described in the Section Channel Description.

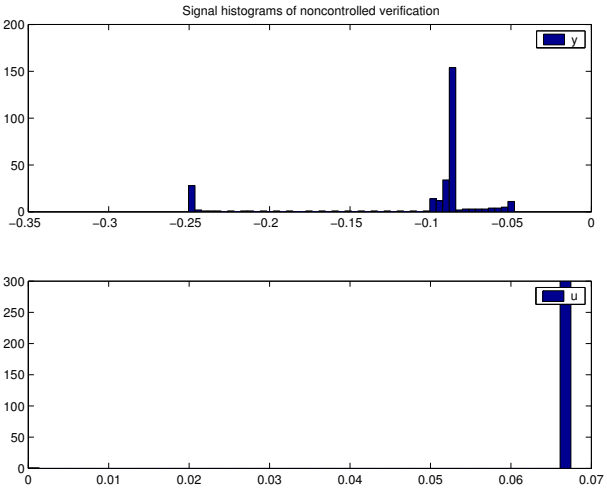
Constraints satisfaction results:

channel	desired range	resulting range	constr.sat.
1 "y"	[-1000, 1000]	[-0.250433, -0.0477858]	1
2 "u"	[-0.8, 0.8]	[0, 0.0674628]	1
Δ 2 "u"	[-0.5, 0.5]	[0, 0.0674628]	1

Sample simulation signals :



Simulation signals histogram :

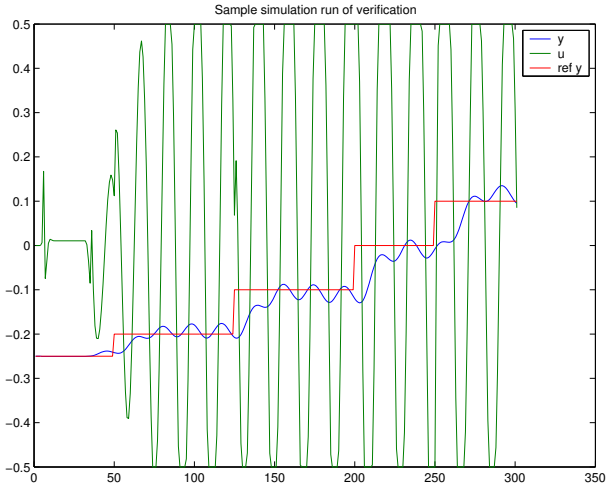


4.11 Conclusion

The designed controller verified that action signal satisfies the prescribed constraints while keeping the output close to the varying reference trajectory. The adaptive property of the controller can be seen in the verification diagram, where the oscilation, caused by change of the working point of the nonlinear simulink model, are being reduced as the model is being adapted for particular working point until it is changed to another value. This behavior demonstrates advantage of the adaptivity.

The verification using a nonadaptive controller, experiment `vrt1`, is shown in the figure below. The oscilation are not decreasing, because of the inaccurate linarized fixed model in different working point.

Sample simulation signals of the nonadaptive controller :



Chapter 5

Banana identification experiment

Experiment: `banan_a` - Banana experiment with two dimensions selected

Author	:	Ludvik Tesar
Contact	:	`tesar at utia dot cas dot cz`
Address	:	AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic
Basic references	:	
Source texts	:	`banan_a`
Project	:	Designer

5.1 Aims of the study

This study performs identification of the mixture of the two "banana-shaped" clusters.

5.2 Description of the study

It is very hard to make proper approximation of two-dimensional "banana-shaped" mixture. Problem of this study is therefore laying in success of identification. To make sure that mixture estimated is static, we set order (`ord`). No control is performed, as it does not give any useful outcome, for static mixture.

5.3 Data

Data are generated by `genrawdata_banan.m` m-file, which uses a big number (30) of component to make reasonable approximation of banana-shaped clusters.

5.4 Processing

Whole experiment files are kept under svn in `mixtools/jobcontrol/examples/acad` together with academic design experiments. Data used for identification are generated by the script `genrawdata_banan`.
To run the experiment:
1) generate the identification data by the `genrawdata_banan` script, unless it was done before.
2) call "prodini" to initialize mixtools
3) call `jobproceed(banan_a)`, where `banan_a.m` is a function generating the Job description.
4) to generate a nice latex report do `latex banan_a`

5.5 Description

5.5.1 Experiment definition

```
jobname    = 'banan_a';  % name of experiment  ...  no spaces!
% choose short 'jobname' that serves as name root for temporary and saved files
authorname = 'Ludvik Tesar';  % author of experiment
email      = 'tesar at utia dot cas dot cz';  % E-mail of author of experiment
address    = 'AS, UTIA, AV CR, POBox 18, 182 08 Prague, Czech Republic';  %
   author's address
references = '';  % references to literature
project    = 'ProDaCTool';  % project name
desc       = 'Banana experiment with two dimensions selected and academic design
   used';  % description of experiment printed in protocol
debug      = 0;    % debug level determining information during evaluations
seed       = [];  % random seed ([] -> leave the seed, -1 -> randomize by timer)

steps    = [1, 1, 1, 1, 1, 1, 1, 1];  %          % Steps to be performed (1/0)
   = (yes/no)
```

5.5.2 Data description

```
datafile     = 'banandata';  % filename with full path specifying the mat file
   with data
varname      = 'DATA';  % variable name of data in 'datafile'
transpose    = 0;   % transpose data matrix to have channels to rows (1=yes
   0=no)
rescale_data = 1;   % rescale new data (normaly this is necesary every time
   the new data is given, but it does no harm to do it every time)  (1=yes 0=no)
reset_chns   = 0;   % reset selected channels (normally this is needed only
   once, in the beginning, and if new data does not have anything to do with
   the old data)  (1=yes 0=no)

chns         = [1, 2];  % modelled channels
pr_chns      = -1;  % vector of numbers of channels from which original data
   plots are printed to protocol (-1 means all channels)
pr_merge     = 1;  % whether original data plots in protocol are merged or not
   (0=separated, 1=merged, 2=tiled).
used_data    = -1;   % % At the moment, there are 1500 data samples
```

5.5.3 Channels description

```
% Description of the channel 1
chn_name       = 'channel 1';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 0;  % available for control
chn_prty       = 0;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [-5.62645, 5.88233];  % expected physical range [min,max]
chn_drange    = [-5.62645;
5.88233];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 1;  % pre-processing information (see help
   preinit)
```

```
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]
type = '';
% Description of the channel 2
chn_name       = 'channel 2';  % name of the channel
chn_oitem      = 1;  % visibility by operator
chn_raction    = 0;  % available for control
chn_prty       = 0;  % presentation priority
chn_type       = 1;  % (1/0) = (continuous/discrete) channel
chn_prange    = [0.459397, 7.34566];  % expected physical range [min,max]
chn_drange    = [0.459397;
7.34566];  % desired range [min,max]
chn_irange    = [];  % desired range of increments [min,max]
chn_preinfo   = 'olymedian', 'c', 2;  % pre-processing information (see help
   preinit)
% Prior informations follow.  You won't get very good documentation for this,
% the best is what you see here or you can find the file guidex.pdf in svn.
% In all cases prior information stacks under each other forming matrices.
chn_gain      = [];  % [uchn, mingain, maxgain] static gain first column is
   index of input channel and then minimum and maximum
chn_stime     = [];  % sampling time is the scalar variable (unit=seconds)
% It provides the time-scale for different prior informations.
% All prior informations that follow need to have sampling time (chn_stime)
   set,
% because they operate on Hertz (you must give the time-scale)
chn_ampl      = [];  % frequency response [uchn, frequency_in_hertz, amplitude_low,
   amplitude_high, phase_in_degrees]
chn_cut       = [];  % cut-off frequency [uchn, frequency_in_hertz]
chn_tc        = [];  % time constant [uchn, tclow, tchigh]
type = '';
```

5.5.4 Prior information

```
ncom    = 5;       % the number of components
ord     = 0;       % order of the richest regressor
diacove = 0.0001;  % diagonal of noise covariancecove
diaCth  = 10000;   % diagonal of par.  covariance
dfm     = 30;     % degrees of freedom of factors
dfcs    = [30, 30, 30, 30, 30];  % degrees of freedom of components

doflattening = 0;  % indicates whether the initial mixture should be flattened
```

5.5.5 Mixture initialization

```
SingleOnly  = 0;       % boolean flag whether to skip mixinit and calculate
   just MixSingle
opt     = 'p';    % iterative estimation method (p | q | b | f | Q | P)
niter   = 10;    % maximum number of iterations
frg     = 0.999999;    % forgetting factor
```

5.5.6 Mixture estimation

```
opt         = 'p';       % iterative estimation method (p | q | b | f | Q |
   P)
niter       = 40;           % maximum number of iterations
frg         = 0.999999;           % forgetting factor
frgEstType  = 0;           % estimation type of forgetting factor (0-none, 1-zero
   alt, 2-prev estimate)
frgEstGrid    = [1, 0.99, 0.983362, 0.972317, 0.953941, 0.923366, 0.872495,
   0.787855, 0.647029, 0.412721, 0.022876];  %
% estimation grid of forgetting factor
```

5.5.7 Mixture validation

```
nsteps  = 1;     % the prediction is made nsteps ahead
pchns    = [1, 2];  % channels to be predicted
cchns    = [];  % channels in condition
tstart  = 1;     % starting time determining the part of data used in validation
tend  = 1500;  %   ending time determining the part of data used in validation
epss  = 1;     % (1/0) = (produce/do not produce) encapsulated postscript plots
pauses  = 0;     % pause in seconds after each plot set
plots    = [1, 0, 1, 0];  % [show cluster plot , show time plot, mixture plot,
   histogram], where (1/0)=(y/n)
segments    = 10;     % number of segments for segmentation test (0 means perform
   no segm.  test).  These tests take a very long time.
alt         = 1;     % perform validation using alternative forgetting estimation
   test (takes very long time, 0=do not run, 1=run this test).
```

5.5.8 User ideal

```
method = 't';  % method determining the user target (t | d | z)
% t ...  User defined (initialized by target.m) - this is default option
% d ...  Designer is used
% z ...  User defined (initialized by zeros)
userSetpointEths   = [0.12794, 3.90253];  % user target component Eths
userSetpointCoves  = [33.113, 11.8552];  % user target component Coves
userSetpointCorrect  = 1;  % do the user target component Correction ?  (0=no,1=yes)
incremental  = 0;  % use incremental penalization controller ?  (0=no,1=yes)
```

5.5.9 Design

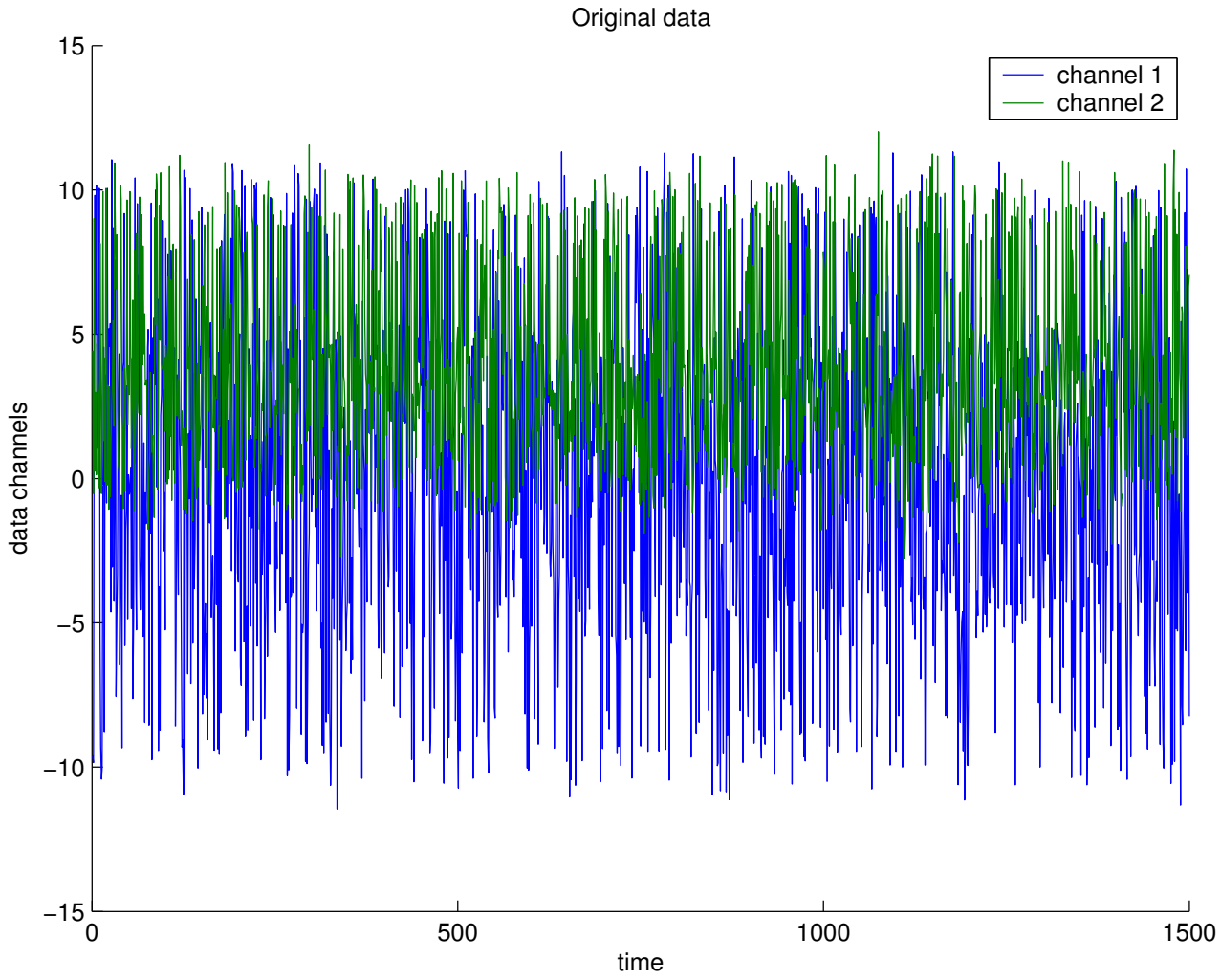
```
designtype = 'a';  % design type (a | i | s)
% a ...  academic design
% i ...  industrial design
% s ...  simultaneous design
```

```
horizon    = [100, -1];  % horizon for evaluation of KLD
ufc        = [];  % ufcgen(aMix, aMixu) is used if ufc=[]
```

5.5.10 Verification

```
method = 't';  % verification method (t | d)
% t ...  verification of controller designed by step 7
% d ...  verification of controller designed by UserIdeal with Designer in step
   6
type = 'none'; % simulation type (none,simulink,mixture)
```

5.5.11 Original Data Plots



5.6 Results

5.6.1 Results of Mixture Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	-2.952e+03	(the bigger the better)
Test of validity of the model: 1		(1=O.K.,0=bad)

Relative SE of pred.err: [0.0258113, 0.0258113] (standard error of ep relative to std of data)
Test of whiteness: [0.320538, 0.238058]
(sum of correlations with delayed predictions)

Elementary statistics for the channel channel 1:

	MIN	MAX	MEAN	MEDIAN	STD
data	-11.4569	11.3183	0.127939	0.312962	5.75439
differences	-21.5957	21.629	0.000911726	-0.170714	8.20672
predictions	0.126708	0.126708	0.126708	0.126708	2.60989e-15
errors of prediction	-11.5836	11.1916	0.00123142	0.186254	5.75439

Elementary statistics for the channel channel 2:

	MIN	MAX	MEAN	MEDIAN	STD
data	-2.78638	12.0178	3.90253	3.28385	3.44313
differences	-12.1044	12.341	0.00175074	-0.0379574	4.88947
predictions	3.90499	3.90499	3.90499	3.90499	7.28549e-14
errors of prediction	-6.69137	8.1128	-0.00246169	-0.621135	3.44313

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	0.0156	0.159	0.13
component 2	0.014	0.143	0.0653
component 3	0.0111	0.0386	0.0453
component 4	0.0347	0.15	0.101
component 5	0.0394	0.0139	0.0412
component 6	0.0197	0.303	0.0883
component 7	0.0718	0.0926	0.0851
component 8	0.0106	0.015	0.016
component 9	5.33e-05	8.41e-06	0.00163
component 10	0.0257	0.136	0.131
component 11	0.0789	0.0332	0.122
component 12	0.00535	0.00864	0.0239
component 13	0.0117	0.0868	0.0424
component 14	0.0163	0.141	0.0671
component 15	0.0701	0.0311	0.0395

Mixture Factors

This mixture has 15 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=213.769128, factors:
   Factor 1, modelled channnel: 1, called '*channel*1', cove=0.0156407, dfm=168.525412

delays →		0	1
offset			0.61451
1	*channel*1		
2	*channel*2	-0.140374	

   Factor 2, modelled channnel: 2, called '*channel*2', cove=0.159098, dfm=168.908655

delays →		0	1
offset			-0.906634
1	*channel*1		
2	*channel*2		

2. component, dfcs=107.460047, factors:
   Factor 3, modelled channnel: 1, called '*channel*1', cove=0.0140134, dfm=101.076474

delays →		0	1
offset			1.70566
1	*channel*1		
2	*channel*2		

   Factor 4, modelled channnel: 2, called '*channel*2', cove=0.142924, dfm=90.001689

delays →		0	1
offset			-0.829721
1	*channel*1		
2	*channel*2		

3. component, dfcs=74.655805, factors:
   Factor 5, modelled channnel: 1, called '*channel*1', cove=0.0110721, dfm=57.894428

delays →		0	1
offset			-0.980559
1	*channel*1		
2	*channel*2	-0.103084	

   Factor 6, modelled channnel: 2, called '*channel*2', cove=0.0385833, dfm=47.563943

delays →		0	1
offset			-1.30767
1	*channel*1		
2	*channel*2		

4. component, dfcs=166.611720, factors:
   Factor 7, modelled channnel: 1, called '*channel*1', cove=0.0346673, dfm=129.061367

delays →		0	1
offset			1.85869
1	*channel*1		
2	*channel*2	-0.752229	

   Factor 8, modelled channnel: 2, called '*channel*2', cove=0.149689, dfm=114.035591

delays →		0	1
offset			1.31343
1	*channel*1		
2	*channel*2		

5. component, dfcs=67.806082, factors:
   Factor 9, modelled channnel: 1, called '*channel*1', cove=0.0394471, dfm=26.887227
   (this factor is empty)
   Factor 10, modelled channnel: 2, called '*channel*2', cove=0.0139318, dfm=23.192514

delays →		0	1
offset			-0.31808
1	*channel*1		
2	*channel*2		

6. component, dfcs=145.401004, factors:
   Factor 11, modelled channnel: 1, called '*channel*1', cove=0.0196963, dfm=116.088495

delays →		0	1
offset			-1.5888
1	*channel*1		
2	*channel*2	0.349975	

   Factor 12, modelled channnel: 2, called '*channel*2', cove=0.303281, dfm=94.965958

delays →		0	1
offset			0.581743
1	*channel*1		
2	*channel*2		

7. component, dfcs=140.169882, factors:
   Factor 13, modelled channnel: 1, called '*channel*1', cove=0.0718158, dfm=68.639390

delays →		0	1
offset			-1.80372
1	*channel*1		
2	*channel*2	0.82696	

   Factor 14, modelled channnel: 2, called '*channel*2', cove=0.0925962, dfm=94.817937

delays →		0	1
offset			1.60232
1	*channel*1		
2	*channel*2		

8. component, dfcs=26.406617, factors:
   Factor 15, modelled channnel: 1, called '*channel*1', cove=0.0105847, dfm=13.390542

delays →		0	1
offset			0.876129
1	*channel*1		
2	*channel*2		

   Factor 16, modelled channnel: 2, called '*channel*2', cove=0.014996, dfm=3.268383

delays →		0	1
offset			-1.35079
1	*channel*1		
2	*channel*2		

9. component, dfcs=2.689100, factors:
   Factor 17, modelled channnel: 1, called '*channel*1', cove=5.33126e-05, dfm=2.184431

delays →		0	1
offset			8.26095
1	*channel*1		
2	*channel*2	4.54187	

   Factor 18, modelled channnel: 2, called '*channel*2', cove=8.41385e-06, dfm=1.807856

delays →		0	1
offset			-1.42949
1	*channel*1		
2	*channel*2		

10. component, dfcs=215.501223, factors:
    Factor 19, modelled channnel: 1, called '*channel*1', cove=0.0257261, dfm=161.658837

delays →		0	1
offset			-0.400646
1	*channel*1		
2	*channel*2	0.513883	

    Factor 20, modelled channnel: 2, called '*channel*2', cove=0.135806, dfm=130.806107

delays →		0	1
offset			-0.375549
1	*channel*1		
2	*channel*2		

11. component, dfcs=201.178403, factors:
Factor 21, modelled channnel: 1, called '*channel*1', cove=0.0788527, dfm=113.780026

delays →		0	1
offset			0.0864997
1	*channel*1		
2	*channel*2		

Factor 22, modelled channnel: 2, called '*channel*2', cove=0.0331939, dfm=128.928740

delays →		0	1
offset			0.00159601
1	*channel*1		
2	*channel*2		

12. component, dfcs=39.333271, factors:
Factor 23, modelled channnel: 1, called '*channel*1', cove=0.00535361, dfm=12.377759

delays →		0	1
offset			-0.773221
1	*channel*1		
2	*channel*2		

Factor 24, modelled channnel: 2, called '*channel*2', cove=0.00863687, dfm=11.649077

delays →		0	1
offset			-0.971525
1	*channel*1		
2	*channel*2		

13. component, dfcs=69.889095, factors:
Factor 25, modelled channnel: 1, called '*channel*1', cove=0.0116679, dfm=47.465021

delays →		0	1
offset			1.48599
1	*channel*1		
2	*channel*2		

Factor 26, modelled channnel: 2, called '*channel*2', cove=0.0867638, dfm=28.519286

delays →		0	1
offset			0.238805
1	*channel*1		
2	*channel*2		

14. component, dfcs=110.542415, factors:
Factor 27, modelled channnel: 1, called '*channel*1', cove=0.0162853, dfm=90.171328

delays →		0	1
offset			-1.71897
1	*channel*1		
2	*channel*2		

Factor 28, modelled channnel: 2, called '*channel*2', cove=0.140959, dfm=84.707381

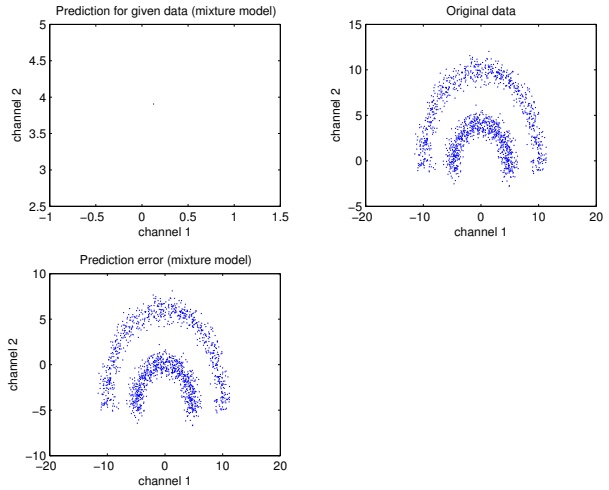
delays →		0	1
offset			-0.804312
1	*channel*1		
2	*channel*2		

15. component, dfcs=65.107341, factors:
Factor 29, modelled channnel: 1, called '*channel*1', cove=0.0700545, dfm=17.554951
(this factor is empty)
Factor 30, modelled channnel: 2, called '*channel*2', cove=0.0310766, dfm=34.091111

delays →		0	1
offset			1.62629
1	*channel*1		
2	*channel*2		

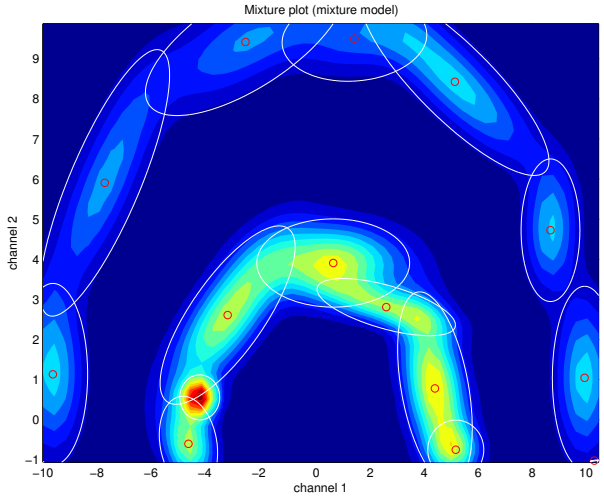
Cluster plots:

Graph of prediction, original data, and prediction error for "channel 1" and "channel 2" channels:

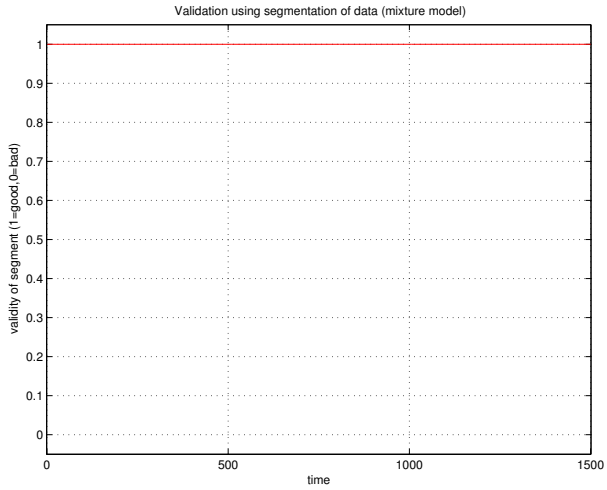


Mixture plots:

Mixture plot for channels "channel 1" and "channel 2":



Segment validation (red line: 1=valid, 0=not valid). There are 10 segments, each of the size 150:



5.6.2 Results of Single Component Identification and Parameter Estimation (and Model Validation)

Comprehensive tests of the model validity:

Value of mixll:	-4.257e+03	(the bigger the better)
Test of validity of the model:	1	(1=O.K.,0=bad)
Relative SE of pred.err:	[0.0258113, 0.0258113]	(standard error of ep relative to std of data)
Test of whiteness:	[0.320538, 0.238058]	
	(sum of correlations with delayed predictions)	

Elementary statistics for the channel channel 1:

	MIN	MAX	MEAN	MEDIAN	STD
data	-11.4569	11.3183	0.127939	0.312962	5.75439
differences	-21.5957	21.629	0.000911726	-0.170714	8.20672
predictions	0.127829	0.127829	0.127829	0.127829	6.38591e-16
errors of prediction	-11.5847	11.1905	0.000110165	0.185133	5.75439

Elementary statistics for the channel channel 2:

	MIN	MAX	MEAN	MEDIAN	STD
data	-2.78638	12.0178	3.90253	3.28385	3.44313
differences	-12.1044	12.341	0.00175074	-0.0379574	4.88947
predictions	3.90256	3.90256	3.90256	3.90256	7.55203e-15
errors of prediction	-6.68894	8.11522	-3.66173e-05	-0.61871	3.44313

Noise noise-variance estimates for individual factors:

	1	2	dfcs
component 1	1.02	1.02	1

Mixture Factors

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1646.519037, factors:
   Factor 1, modelled channnel: 1, called '*channel*1', cove=1.01801, dfm=1526.698693

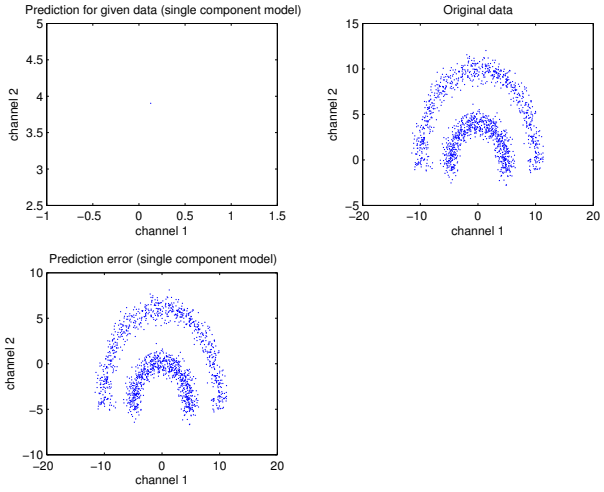
delays →		0	1
offset			-1.92346e-05
1	*channel*1		
2	*channel*2		

Factor 2, modelled channnel: 2, called '*channel*2', cove=1.01803, dfm=1526.698693

delays →		0	1
offset			1.01621e-05
1	*channel*1		
2	*channel*2		

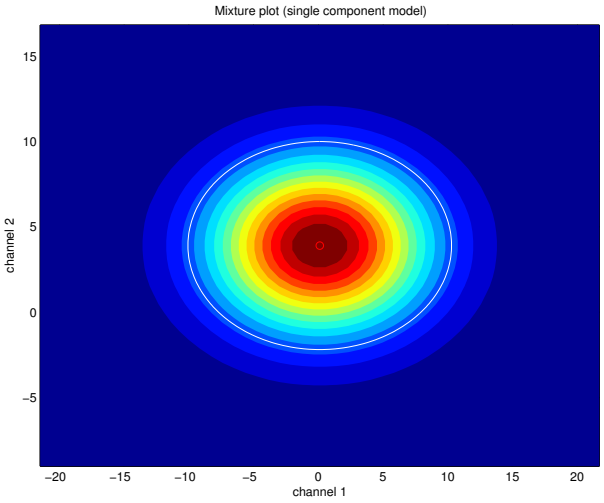
Cluster plots:

Graph of prediction, original data, and prediction error for "channel 1" and "channel 2" channels:

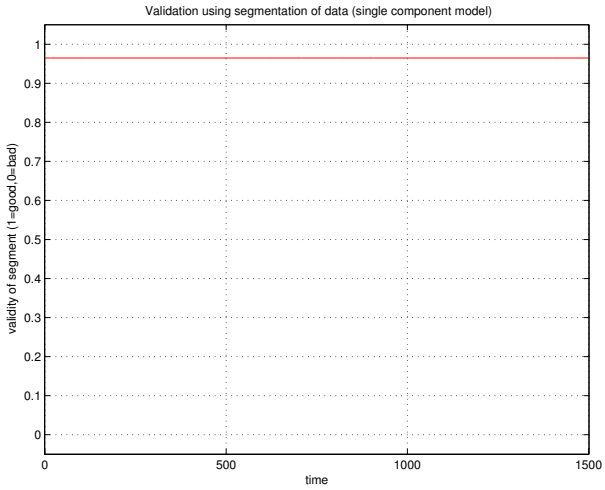


Mixture plots:

Mixture plot for channels "channel 1" and "channel 2":



Segment validation (red line: 1=valid, 0=not valid). There are 10 segments, each of the size 150:



5.6.3 User Ideal Mixture

This mixture has 1 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=1.000000, factors:
   Factor 1, modelled channnel: 1, called '*channel*1', cove=33.7095, dfm=1.000000

delays →		0	1
offset			-1.7378e-07
1	*channel*1		
2	*channel*2		

   Factor 2, modelled channnel: 2, called '*channel*2', cove=12.0688, dfm=1.000000

delays →		0	1
offset			4.3565e-07
1	*channel*1		
2	*channel*2		

5.6.4 Controller Mixture

This mixture has 15 components with 2 factors each. Mixture consists of ARX factors. Interpretation of tables below: Each column correspond to one delay. Each row corresponds to the channel, and the first row employs the offset.

1. component, dfcs=213.769128, factors:
   Factor 1, modelled channnel: 1, called '*channel*1', cove=0.0156407, dfm=168.525412

delays →		0	1
offset			0.61451
1	*channel*1		
2	*channel*2	-0.140374	

   Factor 2, modelled channnel: 2, called '*channel*2', cove=0.159098, dfm=168.908655

delays →		0	1
offset			-0.906634
1	*channel*1		
2	*channel*2		

2. component, dfcs=107.460047, factors:
   Factor 3, modelled channnel: 1, called '*channel*1', cove=0.0140134, dfm=101.076474

delays →		0	1
offset			1.70566
1	*channel*1		
2	*channel*2		

   Factor 4, modelled channnel: 2, called '*channel*2', cove=0.142924, dfm=90.001689

delays →		0	1
offset			-0.829721
1	*channel*1		
2	*channel*2		

3. component, dfcs=74.655805, factors:
   Factor 5, modelled channnel: 1, called '*channel*1', cove=0.0110721, dfm=57.894428

delays →		0	1
offset			-0.980559
1	*channel*1		
2	*channel*2	-0.103084	

   Factor 6, modelled channnel: 2, called '*channel*2', cove=0.0385833, dfm=47.563943

delays →		0	1
offset			-1.30767
1	*channel*1		
2	*channel*2		

4. component, dfcs=166.611720, factors:
   Factor 7, modelled channnel: 1, called '*channel*1', cove=0.0346673, dfm=129.061367

delays →		0	1
offset			1.85869
1	*channel*1		
2	*channel*2	-0.752229	

   Factor 8, modelled channnel: 2, called '*channel*2', cove=0.149689, dfm=114.035591

delays →		0	1
offset			1.31343
1	*channel*1		
2	*channel*2		

5. component, dfcs=67.806082, factors:
   Factor 9, modelled channnel: 1, called '*channel*1', cove=0.0394471, dfm=26.887227
   (this factor is empty)
   Factor 10, modelled channnel: 2, called '*channel*2', cove=0.0139318, dfm=23.192514

delays →		0	1
offset			-0.31808
1	*channel*1		
2	*channel*2		

6. component, dfcs=145.401004, factors:
   Factor 11, modelled channnel: 1, called '*channel*1', cove=0.0196963, dfm=116.088495

delays →		0	1
offset			-1.5888
1	*channel*1		
2	*channel*2	0.349975	

   Factor 12, modelled channnel: 2, called '*channel*2', cove=0.303281, dfm=94.965958

delays →		0	1
offset			0.581743
1	*channel*1		
2	*channel*2		

7. component, dfcs=140.169882, factors:
   Factor 13, modelled channnel: 1, called '*channel*1', cove=0.0718158, dfm=68.639390

delays →		0	1
offset			-1.80372
1	*channel*1		
2	*channel*2	0.82696	

   Factor 14, modelled channnel: 2, called '*channel*2', cove=0.0925962, dfm=94.817937

delays →		0	1
offset			1.60232
1	*channel*1		
2	*channel*2		

8. component, dfcs=26.406617, factors:
   Factor 15, modelled channnel: 1, called '*channel*1', cove=0.0105847, dfm=13.390542

delays →		0	1
offset			0.876129
1	*channel*1		
2	*channel*2		

   Factor 16, modelled channnel: 2, called '*channel*2', cove=0.014996, dfm=3.268383

delays →		0	1
offset			-1.35079
1	*channel*1		
2	*channel*2		

9. component, dfcs=2.689100, factors:
   Factor 17, modelled channnel: 1, called '*channel*1', cove=5.33126e-05, dfm=2.184431

delays →		0	1
offset			8.26095
1	*channel*1		
2	*channel*2	4.54187	

   Factor 18, modelled channnel: 2, called '*channel*2', cove=8.41385e-06, dfm=1.807856

delays →		0	1
offset			-1.42949
1	*channel*1		
2	*channel*2		

10. component, dfcs=215.501223, factors:
    Factor 19, modelled channnel: 1, called '*channel*1', cove=0.0257261, dfm=161.658837

delays →		0	1
offset			-0.400646
1	*channel*1		
2	*channel*2	0.513883	

    Factor 20, modelled channnel: 2, called '*channel*2', cove=0.135806, dfm=130.806107

delays →		0	1
offset			-0.375549
1	*channel*1		
2	*channel*2		

11. component, dfcs=201.178403, factors:
Factor 21, modelled channnel: 1, called '*channel*1', cove=0.0788527, dfm=113.780026

delays →		0	1
offset			0.0864997
1	*channel*1		
2	*channel*2		

Factor 22, modelled channnel: 2, called '*channel*2', cove=0.0331939, dfm=128.928740

delays →		0	1
offset			0.00159601
1	*channel*1		
2	*channel*2		

12. component, dfcs=39.333271, factors:
Factor 23, modelled channnel: 1, called '*channel*1', cove=0.00535361, dfm=12.377759

delays →		0	1
offset			-0.773221
1	*channel*1		
2	*channel*2		

Factor 24, modelled channnel: 2, called '*channel*2', cove=0.00863687, dfm=11.649077

delays →		0	1
offset			-0.971525
1	*channel*1		
2	*channel*2		

13. component, dfcs=69.889095, factors:
Factor 25, modelled channnel: 1, called '*channel*1', cove=0.0116679, dfm=47.465021

delays →		0	1
offset			1.48599
1	*channel*1		
2	*channel*2		

Factor 26, modelled channnel: 2, called '*channel*2', cove=0.0867638, dfm=28.519286

delays →		0	1
offset			0.238805
1	*channel*1		
2	*channel*2		

14. component, dfcs=110.542415, factors:
Factor 27, modelled channnel: 1, called '*channel*1', cove=0.0162853, dfm=90.171328

delays →		0	1
offset			-1.71897
1	*channel*1		
2	*channel*2		

Factor 28, modelled channnel: 2, called '*channel*2', cove=0.140959, dfm=84.707381

delays →		0	1
offset			-0.804312
1	*channel*1		
2	*channel*2		

15. component, dfcs=65.107341, factors:
Factor 29, modelled channnel: 1, called '*channel*1', cove=0.0700545, dfm=17.554951
(this factor is empty)
Factor 30, modelled channnel: 2, called '*channel*2', cove=0.0310766, dfm=34.091111

delays →		0	1
offset			1.62629
1	*channel*1		
2	*channel*2		

5.7 Conclusion

Identification have found a mixture of 15 components, which very closely resembles original banana-shaped mixture. Therefore identification could be considered success. Both segment validation and mixture plots are successful.

Chapter 6

Summary

The presented experiments show the main capabilities of the Jobcontrol environment. Useful comments on possible modification and their result was given in the respective conclusion sections of respective experiments. The reader is encouraged to try his own modification or even make completely new tasks for the Jobcontrol environment. For this purpose we remind again the detailed explanation of the Jobcontrol experiment settings is given in [1].

Chapter 7

Acknowledgements

This report was supported by MŠMT 1M0572 (DAR), project AV ČR 1075351, project GA ČR 102/03/0049, and project GA ČR 102/05/0271.

Bibliography

[1] Ludvík Tesař and Miroslav Novák. Support environment for system identification and controller design: Jobcontrol. Technical Report 2138, Ústav teorie informace a automatizace, AV ČR, Prague, Czech Republic, 2005. research report not submitted to library yet.