Takagi-Sugeno Model Identification Toolbox

March 29, 2021

Automatic static LiP model for an academic example

V1.0

Axel Dürrbaum (`axel.duerrbaum@mrt.uni-kassel.de`)

Department of Measurement and Control (MRT)

Institute for System Analytics and Control (ISAC)

University of Kassel, Germany (`http://www.uni-kassel.de/go/mrt`)

$Id: Static_Acad_auto.m | Fri Feb 26 16:25:05 2021 +0100 | Axel Dürrbaum $

Contents

Example of automatic identification of a static MISO LiP TS model for given multiple inputs u and single output y with minimal requirements.

Determine the MISO LiP TS model

$$y(u) = \sum_{i=1}^{n_v} \phi_i(z) \cdot \left(\sum_{j=1}^{n_u} B_{i,j} \cdot u_j + c_i \right)$$

- for given vectors $u_j, j = 1, \ldots, n_u$ of n_u inputs and
- vector y of single output,
- with FCM membership function

$$\mu_i(x) = \left(\sum_{j=1}^{n_v} \left(\frac{||z - v_i||}{||z - v_j||} \right)^{\frac{2}{\nu - 1}} \right)^{-1}$$

- or Gaussian membership function

$$\mu_i(z) = e^{-\frac{||z - v_i||^2}{2 \cdot \sigma_i^2}}$$

- norm $||z - v_j|| = (z - v_j)^T \cdot A_j \cdot (z - v_j)$
- and fuzzy basis functions

$$\phi_i(z) = \frac{\mu_i(z)}{\sum_{j=1}^{n_v} \mu_j(z)}$$

- with the scheduling variable $z = u$ (for input space clustering) or $z = [u, y]$ (for product space clustering), and
- cluster centers $v_i, i = 1, \ldots, n_v$.

1 Algorithm

1. Search the best TS model with the minimal MSE for $n_v = \{2, 3, 4\}$ and $\nu = \{1.05, 1.1, 1.2, 1.5, 2\}$.
2. Select the TS model with minimal MSE of s multi-start tries for clustering and Least Squares estimation.
3. Optimize the TS model parameters (v_i, B_i, c_i) for each try.

2 Minimal required data

Inputs $u \in \mathbb{R}^{N \times n_u}$ and output $y \in \mathbb{R}^N$, each with N data points

3 Identification data

Given is an academic example as a TS model with

- inputs $u_1, u_2 \in [0, 2]$
- local model matrices

$$B = \begin{pmatrix} -4 & 4 \\ 4 & -2 \\ 2 & 1 \end{pmatrix}, \quad c = \begin{pmatrix} -2 \\ -4 \\ 1 \end{pmatrix}$$

- FCM membership functions ($\nu = 1.2$) with Euclidean norm
- cluster centers

$$v = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 1.5 \\ 1.5 & 1 \end{pmatrix}$$

Load data u, y with $N = 50$ data-points without noise, generated from this model:

```
load( 'Data/AcadEx.mat' )
```

4 Structural parameters

Number of inputs n_u = number of columns in u

```
Par.nu = size( u, 2);
```

Number of clusters n_v = number of local models ($n_v > 1$)

```
Par.nv = [ 2, 3, 4 ];
```

Fuzziness parameter (FCM: $\nu = \{1.05, \ldots, 2\}$, Gauss: σ_i^2)

```
Par.fuzzy = [ 1.05, 1.2 ,2.0 ];
```

5 Optional settings

For more control over the approximation process.

Multi-Start: number of tries s (clustering & LS), default = 10

```
Par.Tries = 10;
```

Clustering: Fuzzy C-Means (FCM) / Gustafson-Kessel (GK) / KMeans (KMeans), default = 'FCM'

```
Par.Clustering = 'FCM';
```

Clustering in product space: u and y (true) or only input space u (false)

```
Par.ProductSpace = true;
```

Norm for clustering: 'Euclidean' or 'Mahalanobis', default = 'Euclidean'

```
Par.Norm = 'Euclidean';
```

Membership functions: 'FCM' or 'Gauss' type clustering

```
Par.MSF = 'FCM';
```

Least Squares estimation of local models: 'local' or 'global', default = 'global'

```
Par.LS = 'global';
```

Optimize TS model parameters: default='both'

- no optimization: 'none',
- only v: 'cluster',
- only local models (B_i, c_i): 'model', or
- both v and B_i, c_i: 'both'

```
Par.ParOpt = 'both';
```

Optimize each try or only best try: default='each'

- each try: 'each',
- best try: 'best' (less computation time)

```
Par.IterOpt = 'each';
```

Plot clusters and residuals: 'none'/'iter'/'final', default='final'

```
Par.Plots = 'final';
```

Debug infos of algorithm progess: (0=none, 1=info, 2=detailed)

```
Par.Debug = 2;
```

6 Estimation of Static TS model parameters

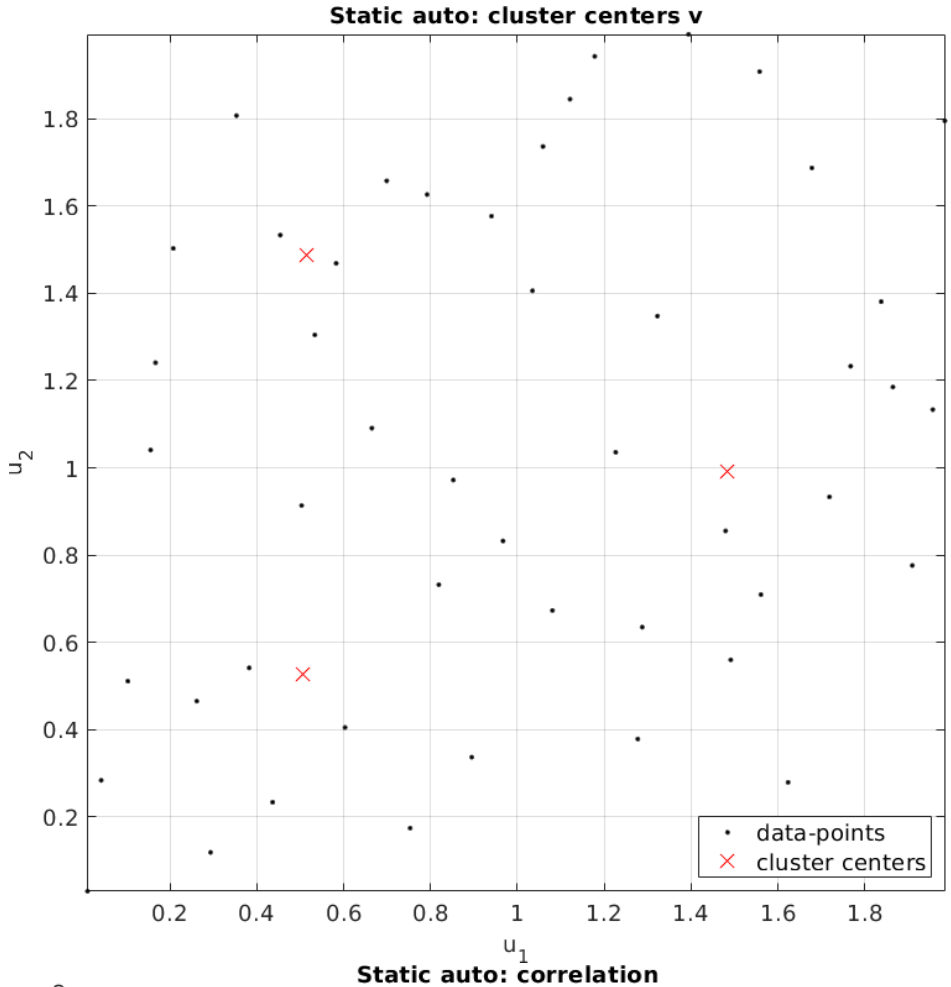
Estimate the TS model with plot of clustering and correlation:

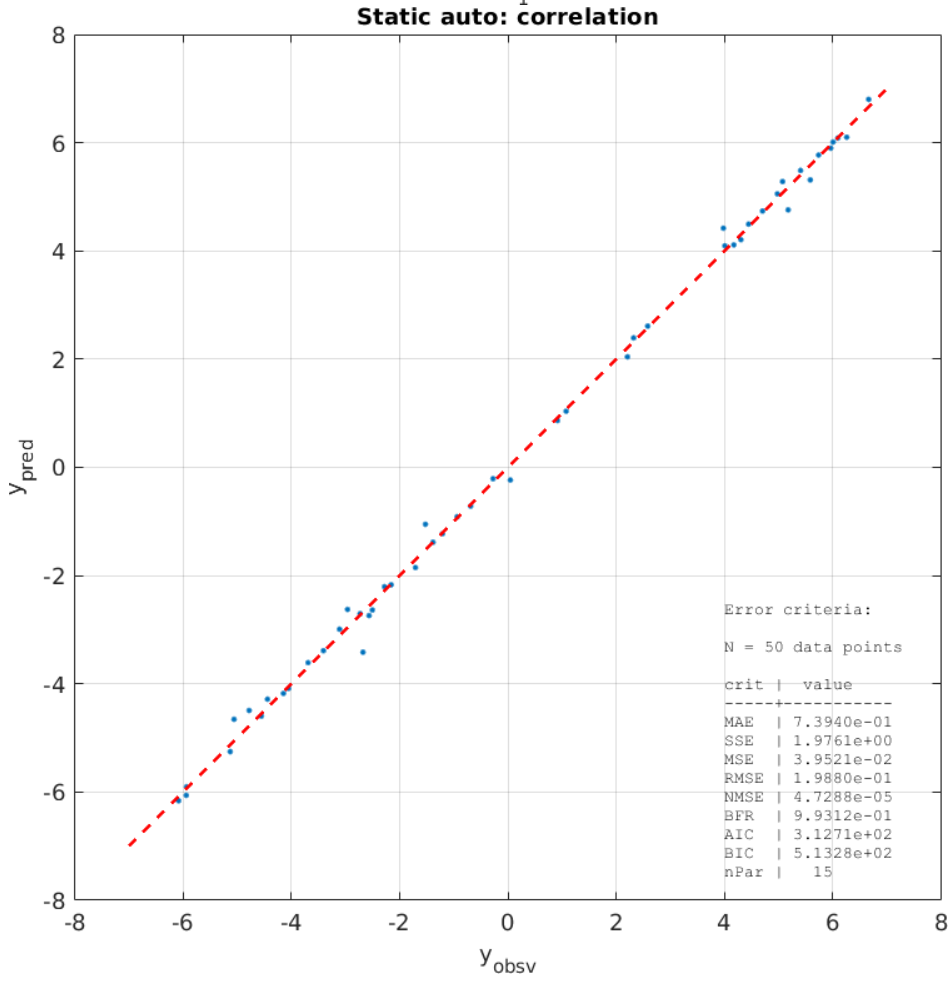
```
model = TSM_Static_auto( u, y, Par );
```

```
Iteration: nv= 2 / fuzzy=1.05
 try  1: mse = 1.4342e+00 / delta = +0.0000e+00 (nv= 2/fuzzy=1.05)
 try  2: mse = 1.4334e+00 / delta = +0.0000e+00 (nv= 2/fuzzy=1.05)
 try  3: mse = 1.4293e+00 / delta = +0.0000e+00 (nv= 2/fuzzy=1.05)
 try  4: mse = 1.4403e+00 / delta = +1.0967e-02 (nv= 2/fuzzy=1.05)
 try  5: mse = 1.4348e+00 / delta = +5.4669e-03 (nv= 2/fuzzy=1.05)
 try  6: mse = 1.4335e+00 / delta = +4.1426e-03 (nv= 2/fuzzy=1.05)
 try  7: mse = 1.4292e+00 / delta = +0.0000e+00 (nv= 2/fuzzy=1.05)
 try  8: mse = 1.4378e+00 / delta = +8.6055e-03 (nv= 2/fuzzy=1.05)
 try  9: mse = 1.4336e+00 / delta = +4.3855e-03 (nv= 2/fuzzy=1.05)
 try 10: mse = 1.4253e+00 / delta = +0.0000e+00 (nv= 2/fuzzy=1.05)
 time = 0.252519 s
Iteration: nv= 2 / fuzzy=1.20
 try  1: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  2: mse = 1.4835e+00 / delta = +5.8163e-02 (nv= 2/fuzzy=1.05)
 try  3: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  4: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  5: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  6: mse = 1.4835e+00 / delta = +5.8163e-02 (nv= 2/fuzzy=1.05)
 try  7: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  8: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try  9: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 try 10: mse = 1.4839e+00 / delta = +5.8539e-02 (nv= 2/fuzzy=1.05)
 time = 0.059033 s
Iteration: nv= 2 / fuzzy=2.00
 try  1: mse = 9.9954e-01 / delta = +0.0000e+00 (nv= 2/fuzzy=   2)
 try  2: mse = 9.9954e-01 / delta = +0.0000e+00 (nv= 2/fuzzy=   2)
 try  3: mse = 9.9954e-01 / delta = +1.2729e-10 (nv= 2/fuzzy=   2)
 try  4: mse = 9.9954e-01 / delta = +1.2596e-10 (nv= 2/fuzzy=   2)
 try  5: mse = 9.9954e-01 / delta = +9.5490e-12 (nv= 2/fuzzy=   2)
 try  6: mse = 9.9954e-01 / delta = +0.0000e+00 (nv= 2/fuzzy=   2)
 try  7: mse = 9.9954e-01 / delta = +1.1290e-11 (nv= 2/fuzzy=   2)
 try  8: mse = 9.9954e-01 / delta = +7.5993e-12 (nv= 2/fuzzy=   2)
 try  9: mse = 9.9954e-01 / delta = +1.3306e-11 (nv= 2/fuzzy=   2)
 try 10: mse = 9.9954e-01 / delta = +1.2980e-10 (nv= 2/fuzzy=   2)
 time = 0.051883 s
Iteration: nv= 3 / fuzzy=1.05
 try  1: mse = 7.5872e-02 / delta = +0.0000e+00 (nv= 3/fuzzy=1.05)
 try  2: mse = 7.6107e-02 / delta = +2.3485e-04 (nv= 3/fuzzy=1.05)
 try  3: mse = 7.5872e-02 / delta = +0.0000e+00 (nv= 3/fuzzy=1.05)
 try  4: mse = 7.6107e-02 / delta = +2.3485e-04 (nv= 3/fuzzy=1.05)
 try  5: mse = 7.5872e-02 / delta = +0.0000e+00 (nv= 3/fuzzy=1.05)
 try  6: mse = 7.5872e-02 / delta = +1.7555e-14 (nv= 3/fuzzy=1.05)
 try  7: mse = 7.5872e-02 / delta = +3.0593e-11 (nv= 3/fuzzy=1.05)
 try  8: mse = 7.5872e-02 / delta = +3.0633e-11 (nv= 3/fuzzy=1.05)
 try  9: mse = 7.5872e-02 / delta = +3.0633e-11 (nv= 3/fuzzy=1.05)
 try 10: mse = 7.5872e-02 / delta = +8.0908e-15 (nv= 3/fuzzy=1.05)
 time = 0.103849 s
Iteration: nv= 3 / fuzzy=1.20
 try  1: mse = 3.9521e-02 / delta = +0.0000e+00 (nv= 3/fuzzy= 1.2)
 try  2: mse = 3.9521e-02 / delta = +2.1647e-13 (nv= 3/fuzzy= 1.2)
 try  3: mse = 3.9521e-02 / delta = +1.6694e-13 (nv= 3/fuzzy= 1.2)
 try  4: mse = 3.9521e-02 / delta = +1.7848e-13 (nv= 3/fuzzy= 1.2)
 try  5: mse = 3.9521e-02 / delta = +2.7951e-13 (nv= 3/fuzzy= 1.2)
```

```
 try  6: mse = 3.9521e-02 / delta = +4.7176e-12 (nv= 3/fuzzy= 1.2)
 try  7: mse = 3.9521e-02 / delta = +4.8312e-12 (nv= 3/fuzzy= 1.2)
 try  8: mse = 3.9521e-02 / delta = +1.3433e-13 (nv= 3/fuzzy= 1.2)
 try  9: mse = 3.9521e-02 / delta = +4.8571e-12 (nv= 3/fuzzy= 1.2)
 try 10: mse = 3.9521e-02 / delta = +4.9304e-12 (nv= 3/fuzzy= 1.2)
 time = 0.034243 s
Iteration: nv= 3 / fuzzy=2.00
 try  1: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  2: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  3: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  4: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  5: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  6: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  7: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  8: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try  9: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 try 10: mse = 5.3089e-01 / delta = +4.9137e-01 (nv= 3/fuzzy= 1.2)
 time = 0.087966 s
Iteration: nv= 4 / fuzzy=1.05
 try  1: mse = 8.3754e-02 / delta = +4.4233e-02 (nv= 3/fuzzy= 1.2)
 try  2: mse = 2.1947e-01 / delta = +1.7995e-01 (nv= 3/fuzzy= 1.2)
 try  3: mse = 1.5795e-01 / delta = +1.1843e-01 (nv= 3/fuzzy= 1.2)
 try  4: mse = 8.0152e-02 / delta = +4.0631e-02 (nv= 3/fuzzy= 1.2)
 try  5: mse = 7.9655e-02 / delta = +4.0134e-02 (nv= 3/fuzzy= 1.2)
 try  6: mse = 1.5795e-01 / delta = +1.1843e-01 (nv= 3/fuzzy= 1.2)
 try  7: mse = 2.1947e-01 / delta = +1.7995e-01 (nv= 3/fuzzy= 1.2)
 try  8: mse = 1.5795e-01 / delta = +1.1843e-01 (nv= 3/fuzzy= 1.2)
 try  9: mse = 1.5795e-01 / delta = +1.1843e-01 (nv= 3/fuzzy= 1.2)
 try 10: mse = 2.1947e-01 / delta = +1.7995e-01 (nv= 3/fuzzy= 1.2)
 time = 0.276368 s
Iteration: nv= 4 / fuzzy=1.20
 try  1: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try  2: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try  3: mse = 3.9728e-02 / delta = +2.0712e-04 (nv= 3/fuzzy= 1.2)
 try  4: mse = 3.9729e-02 / delta = +2.0716e-04 (nv= 3/fuzzy= 1.2)
 try  5: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try  6: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try  7: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try  8: mse = 3.9728e-02 / delta = +2.0712e-04 (nv= 3/fuzzy= 1.2)
 try  9: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 try 10: mse = 3.9728e-02 / delta = +2.0706e-04 (nv= 3/fuzzy= 1.2)
 time = 0.199125 s
Iteration: nv= 4 / fuzzy=2.00
 try  1: mse = 4.8537e-01 / delta = +4.4585e-01 (nv= 3/fuzzy= 1.2)
 try  2: mse = 4.8522e-01 / delta = +4.4570e-01 (nv= 3/fuzzy= 1.2)
 try  3: mse = 4.8493e-01 / delta = +4.4541e-01 (nv= 3/fuzzy= 1.2)
 try  4: mse = 4.8538e-01 / delta = +4.4586e-01 (nv= 3/fuzzy= 1.2)
 try  5: mse = 4.8522e-01 / delta = +4.4570e-01 (nv= 3/fuzzy= 1.2)
 try  6: mse = 4.8538e-01 / delta = +4.4586e-01 (nv= 3/fuzzy= 1.2)
 try  7: mse = 4.8538e-01 / delta = +4.4586e-01 (nv= 3/fuzzy= 1.2)
 try  8: mse = 4.8492e-01 / delta = +4.4540e-01 (nv= 3/fuzzy= 1.2)
 try  9: mse = 4.8522e-01 / delta = +4.4570e-01 (nv= 3/fuzzy= 1.2)
 try 10: mse = 4.8553e-01 / delta = +4.4601e-01 (nv= 3/fuzzy= 1.2)
 time = 0.58829 s

Best model: nv= 3 / fuzzy=1.20 /  mse = 3.9521e-02
```


Static auto: cluster centers v
u2
u1
data-points
cluster centers

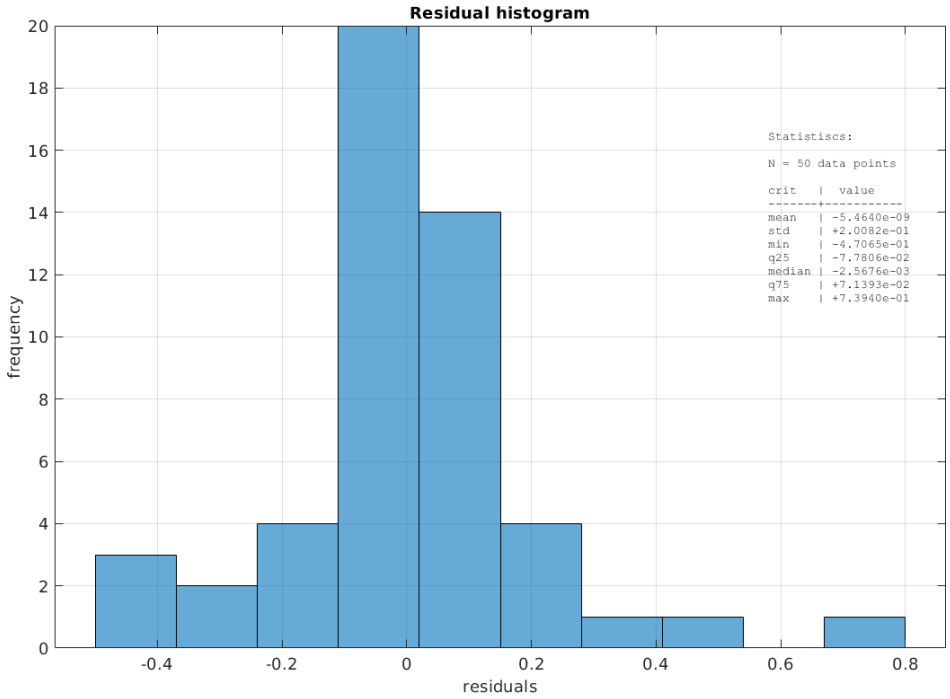

Static auto: correlation
y_pred
y_obsv
Error criteria:
N = 50 data points
crit | value
MAE | 7.3940e-01
SSE | 1.9761e+00
MSE | 3.9521e-02
RMSE | 1.9880e-01
NMSE | 4.7288e-05
BFR | 9.9312e-01
AIC | 3.1271e+02
BIC | 5.1328e+02
nPar | 15

Predict the model output y_{pred} for input u:

```
y_pred = model.predict( u, y );
```

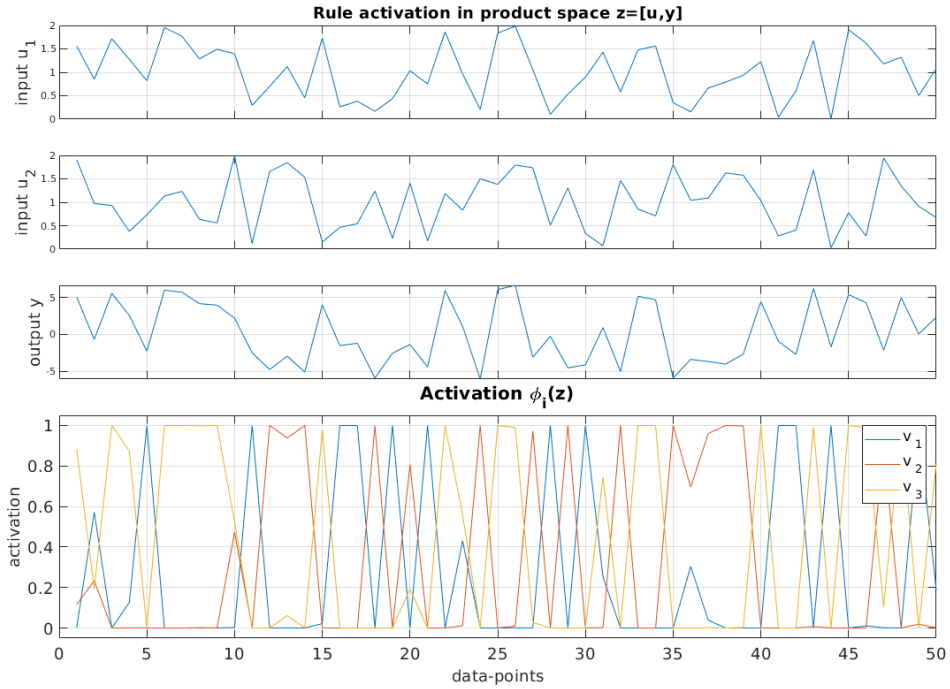
Plot a residual histogram:

```
hr = plotResidualHist( y, y_pred, 'figure', 3 );
```



Plot the rule activation and input/output data:

```
ha = plotRuleActivation( u,y,model, 'figure', 4 );
```



7 Retrieve the parameters of the final TS model

Show the TS model parameters:

```
disp( model )
```

```
TS-Model: Type=Static
Name: 'undefined'
 Type: 'TSModel'
 Date: '29-Mar-2021 15:42:12'
 Comments:
  'created by TSM_static_auto'
 Structural parameters: nu = 2, ny = 1, nv = 3
 Identification data: N=50
 Initial model estimation:
  Clustering: FCM, nue=1.2 norm=Euclidean in product space
 Estimation of local models:
  Initialization of local models: global
  Optimization of model parameters: MF&LM
```

Show the cluster centers v (n_v rows and n_u columns):

```
v = getCluster( model )
```

```
v =
    0.5059    0.5262
    0.5135    1.4882
    1.4815    0.9923
```

Show the local model matrices B_i and c_i (n_v rows and n_u columns):

```
[~,B,c] = getLM( model )
```

```
B =
   -4.0956    4.1970
    3.8744   -1.5237
    1.9182    1.2177
c =
   -1.9318
   -4.6761
    0.8801
```