Takagi-Sugeno Model Identification Toolbox

March 10, 2021

Automatic static LiP model for the 2-dimensional Friedman function.

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_Friedman2D_auto.m | Fri Feb 26 16:25:05 2021 +0100 | Axel Dürrbaum $

Contents

1 Minimal required data

Use the 2-dimensional Friedman function:

$$y = 10 \cdot \sin(\pi \cdot u_1 \cdot u_2)$$

```
nu = 2;
% Choose the fuzziness parameter $\nu = 1.2$
nue = 1.2;
```

Choose the input matrix u as random data with N data-points: $u_1, u2 \in [0, 1]$

```
N = 500;
u = rand( N, nu );
```

Compute the output vector y from the Friedman function:

```
y = Friedman_fct( u, nu );
```

2 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 ($n_v > 1$):

0 = select range $n_v = 2, \ldots, n_{v,\max}$ with $n_{v,\max} = N/(10 * (2 * n_u + 1)$

```
Par.nv = 0;
```

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

```
Par.fuzzy = nue;
```

3 Optional settings

For more control over the approximation process.

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

```
Par.Tries = 3;
```

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: 'Euclidian' or 'Mahalanobis', default = 'Euclidian'

```
Par.Norm = 'Euclidan';
```

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

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

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

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

Optimize 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.Optimize = '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 (0=none, 1=info, 2=detailed)
Par.Debug = 1;
```

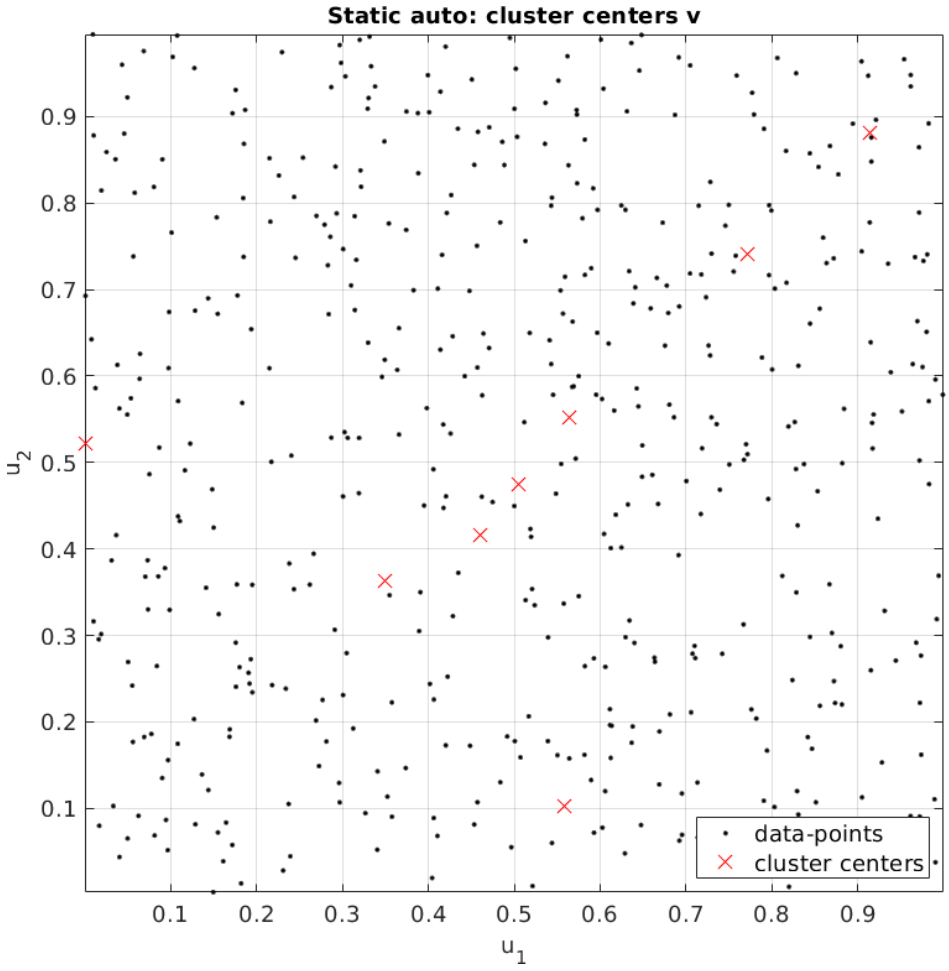
4 Estimation of LiP TS model parameters

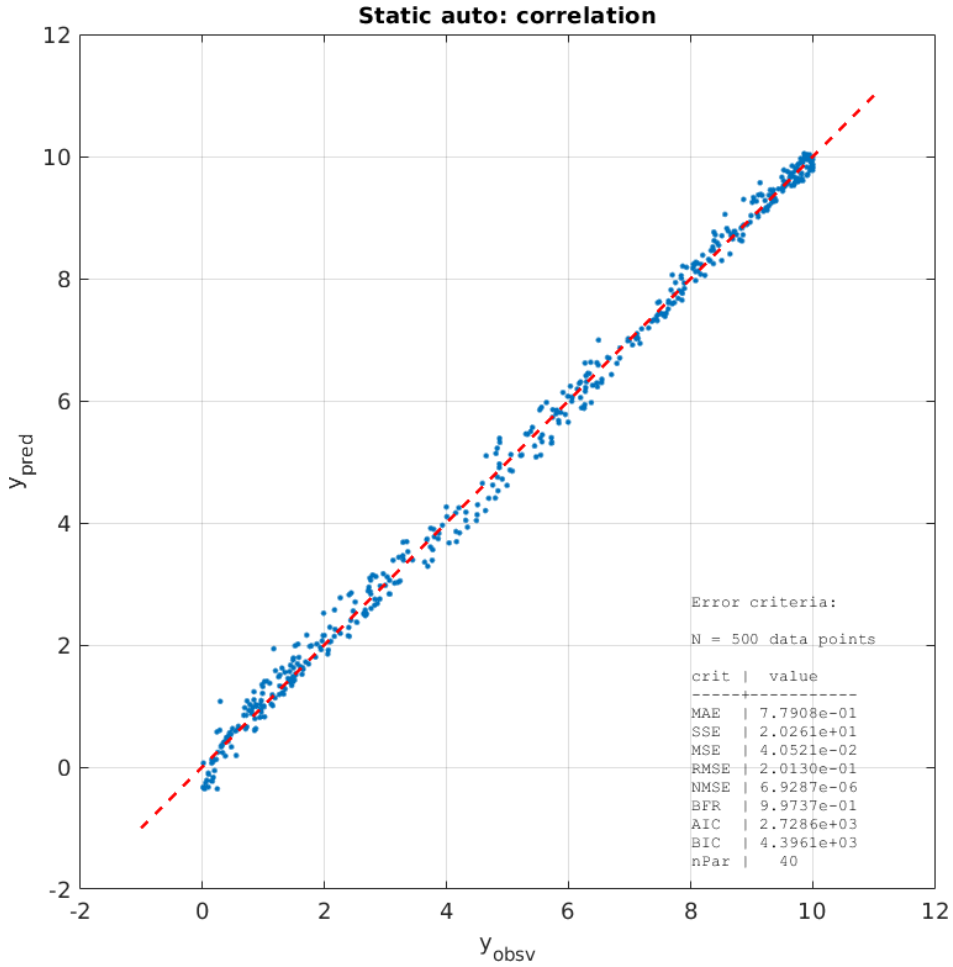
Estimate the TS model with plot of clustering and correlation:

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

```
nv_max choosen as 8 for 6 parameters and 10 data-points/paramter
Iteration: nv= 2 / fuzzy=1.20
 time = 0.451843 s
Iteration: nv= 3 / fuzzy=1.20
 time = 2.82816 s
Iteration: nv= 4 / fuzzy=1.20
 time = 2.83033 s
Iteration: nv= 5 / fuzzy=1.20
 time = 7.67359 s
Iteration: nv= 6 / fuzzy=1.20
 time = 9.06673 s
Iteration: nv= 7 / fuzzy=1.20
 time = 10.0546 s
Iteration: nv= 8 / fuzzy=1.20
 time = 15.6131 s

Best model: nv= 8 / fuzzy=1.20 /  mse = 4.0521e-02
```



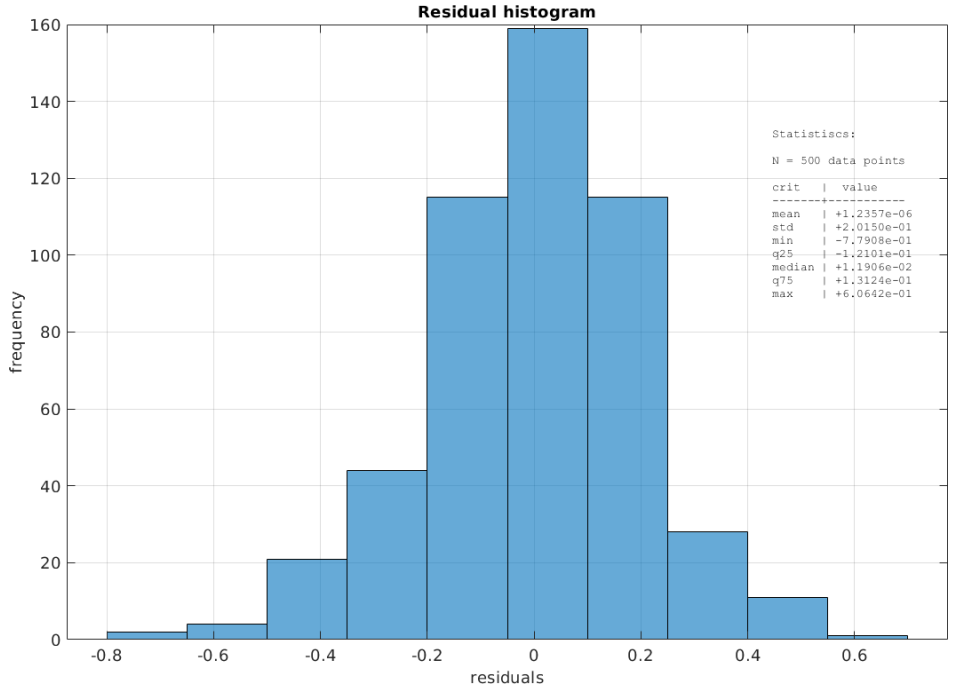


Predict the model output y_{pred} for input u:

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

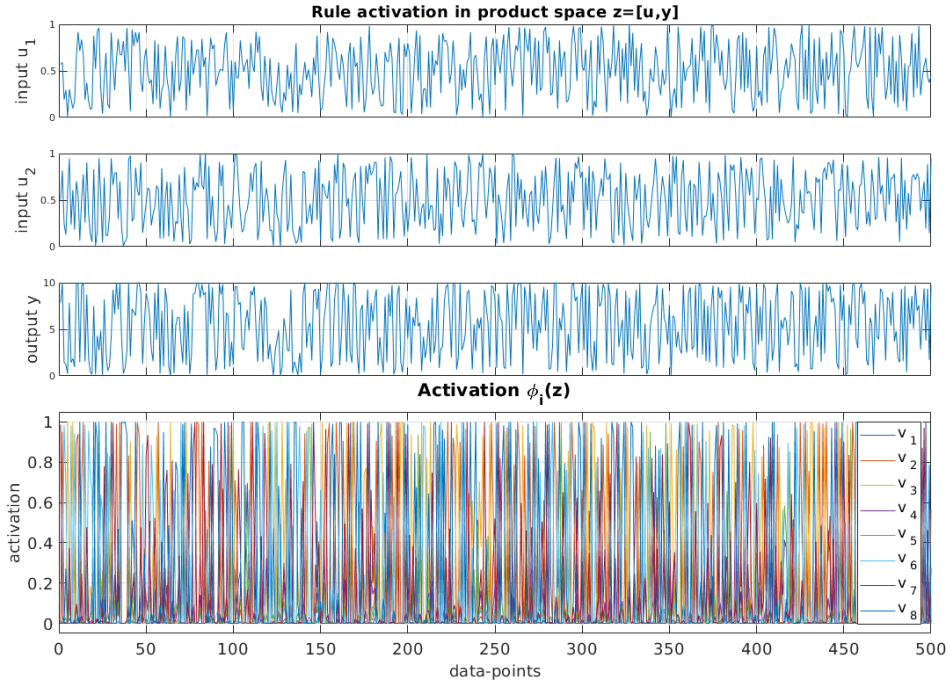
Plot a residual histogram:

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



Plot the rule activation and input/output data:

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



Show the parameter of the resulting TS model:

```
disp( model )
```

```
TS-Model: Type=Static
Name: 'undefined'
 Type: 'TSModel'
 Date: '10-Mar-2021 12:15:39'
 Comments:
  'created by TSM_static_auto'
 Structural parameters: nu = 2, ny = 1, nv = 8
 Identification data: N=500
 Initial model estimation:
  Clustering: FCM, nue=1.2 norm=Euclidian
 Estimation of local models:
  Initialization of local models: global
  Optimization of model parameters: MF&LM
```

5 Validatation of TS model

Choose another N random data-points: $[u_1, u_2]$

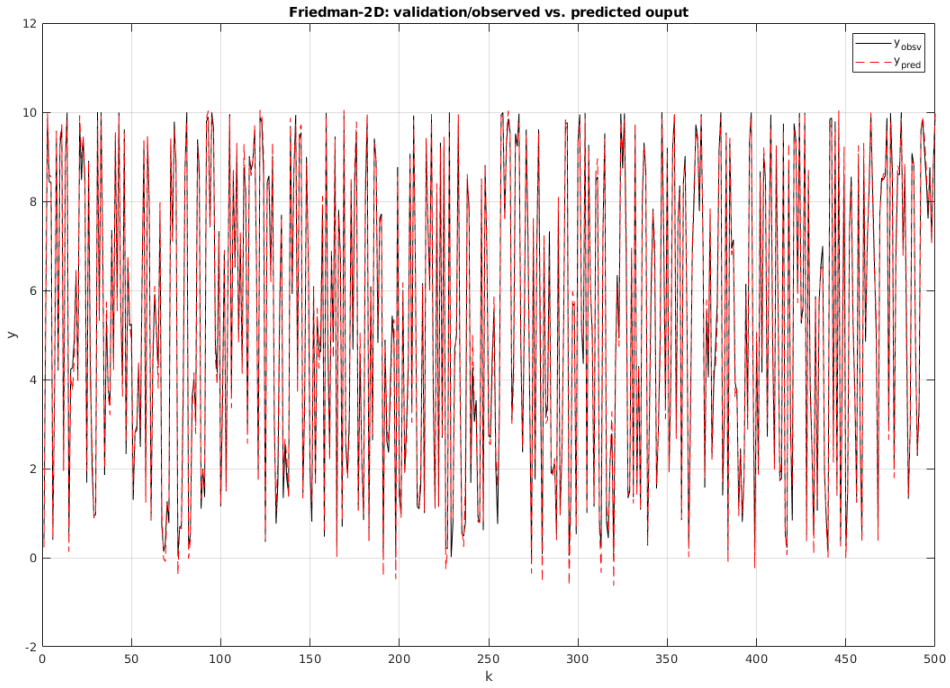
```
u_val = rand( N, nu );
y_val_obsv = Friedman_fct( u_val, nu );%%
```

Compute output vector $y_{val,pred}$

```
y_val_pred = model.predict( u_val );
```

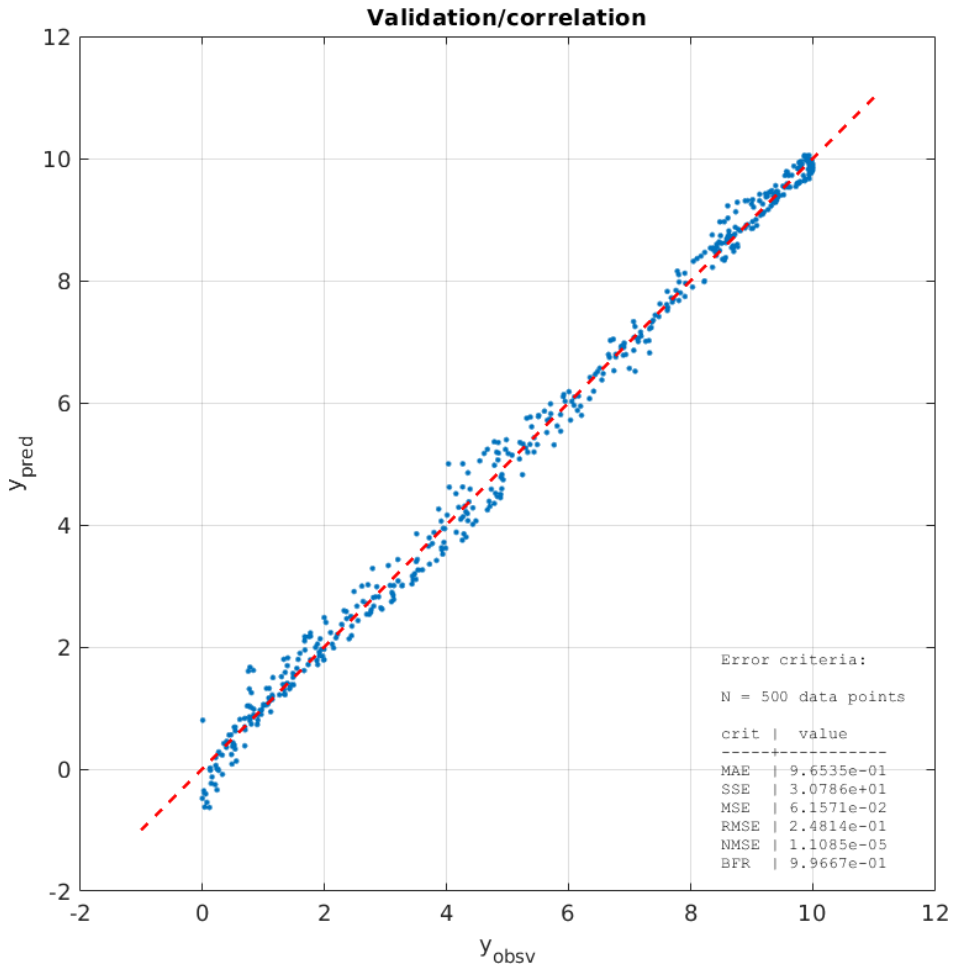
Plot the TS model with the validation data

```
h=figure(3);clf
plot( 1:N, y_val_obsv, 'k-',1:N, y_val_pred, 'r--' )
grid on
xlabel('k')
ylabel('y')
title( 'Friedman-2D: validation/observed vs. predicted ouput' )
legend( 'y_{obsv}','y_{pred}' )
h.WindowState = 'maximized';
```



Plot the correlation for the validation data

```
hr = plotResiduals( y_val_obsv, y_val_pred, 'figure', 4, 'title', 'Validation/correlation' );
hr.WindowState = 'maximized';
```



Plot a residual histogram for the validation data:

```
hv = plotResidualHist( y_val_obsv, y_val_pred, 'figure', 5, 'title', 'Validation/residual histo
hv.WindowState = 'maximized';
```