Takagi-Sugeno Model Identification Toolbox

March 29, 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: $u_1, u_2 \in [0, 1]$

```
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 $n_{v,\max} = N/(10 \cdot (2 \cdot n_u + 1)$

```
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 & Least-Squares), 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: 'Euclidean' or 'Mahalanobis', default = 'Euclidean'

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

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;
```

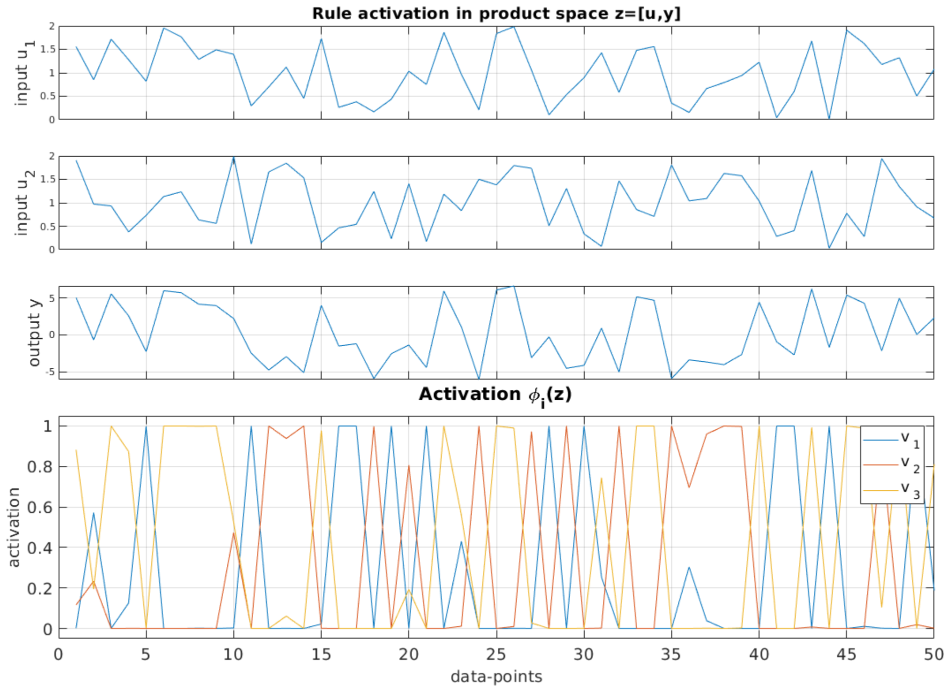
4 Estimation of LiP TS model parameters

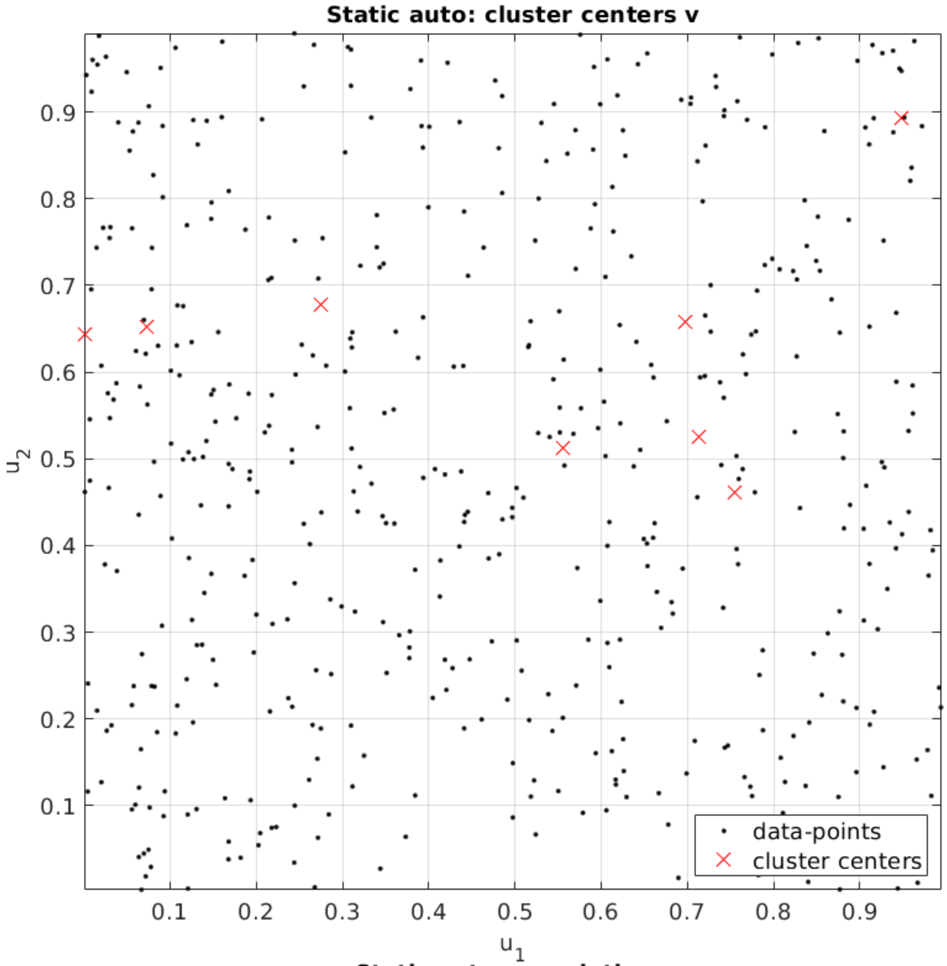
Estimate the TS model with plot of clustering and correlation:

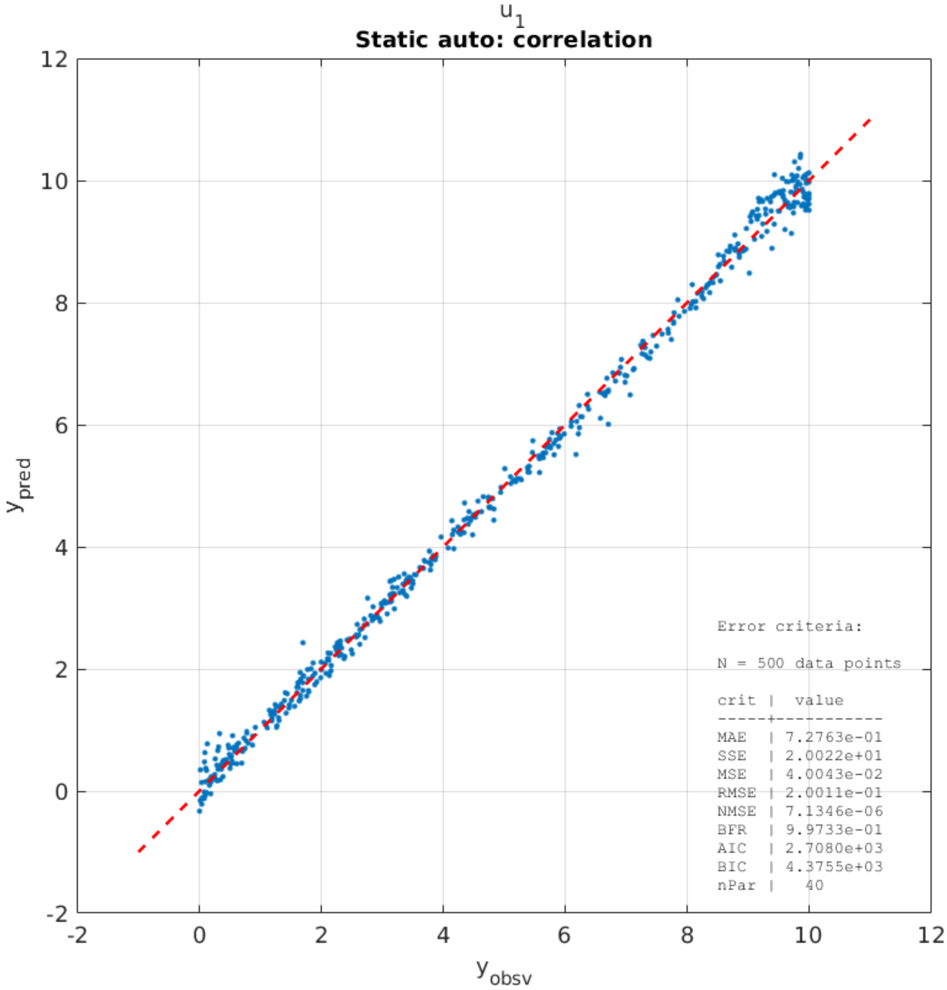
```
model = TSM_Static_auto( u, y, Par );
```

```
nv_max choosen as 8 for 6 parameters and 10 data-points/paramter
Iteration: nv= 2 / fuzzy=1.20
 time = 0.551008 s
Iteration: nv= 3 / fuzzy=1.20
 time = 0.71946 s
Iteration: nv= 4 / fuzzy=1.20
 time = 2.33678 s
Iteration: nv= 5 / fuzzy=1.20
 time = 6.28349 s
Iteration: nv= 6 / fuzzy=1.20
 time = 5.3439 s
Iteration: nv= 7 / fuzzy=1.20
 time = 14.286 s
Iteration: nv= 8 / fuzzy=1.20
 time = 12.8543 s

Best model: nv= 8 / fuzzy=1.20 /  mse = 4.0043e-02
```




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

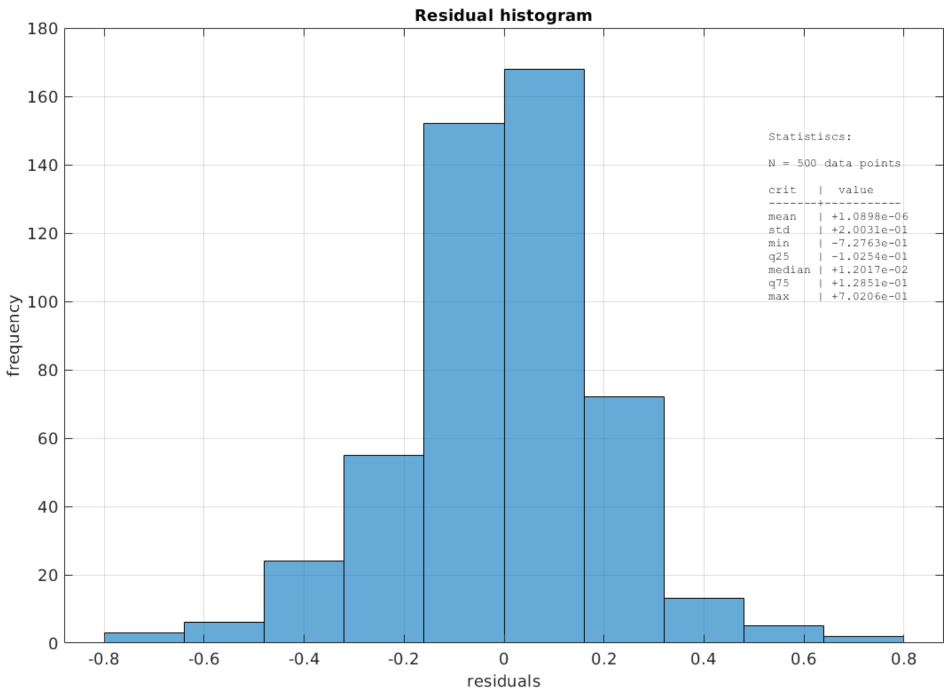

Static auto: correlation
ypred
yobsv
Error criteria:
N = 500 data points
crit | value
MAE | 7.2763e-01
SSE | 2.0022e+01
MSE | 4.0043e-02
RMSE | 2.0011e-01
NMSE | 7.1346e-06
BFR | 9.9733e-01
AIC | 2.7080e+03
BIC | 4.3755e+03
nPar | 40

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

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

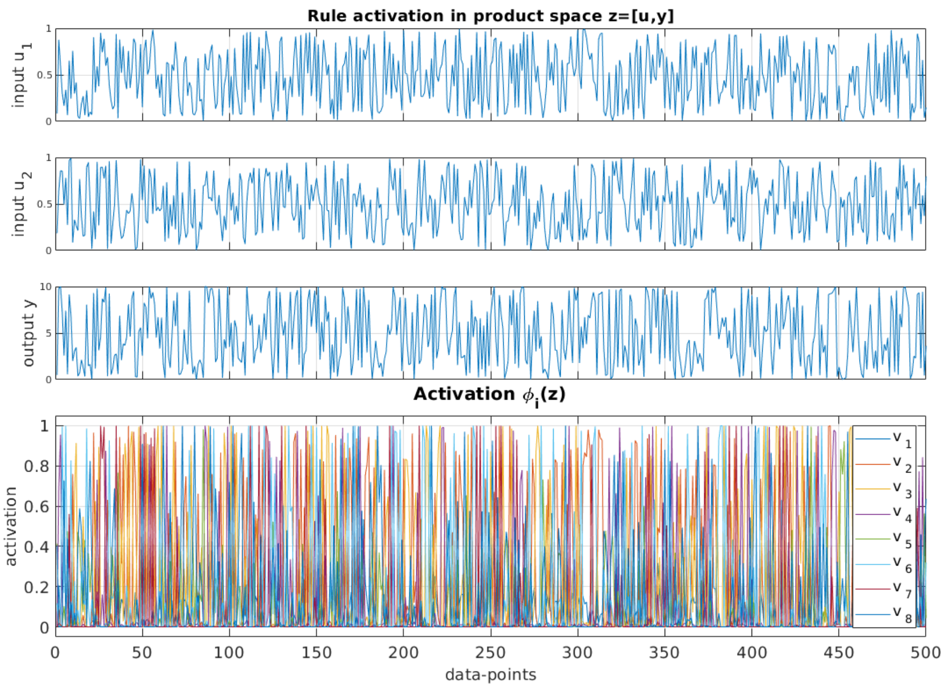
Plot a residual histogram:

```
plotResidualHist( y, y_pred, 'figure', 3 );
set(gcf,'WindowState','maximized');
```



Plot the rule activation and input/output data:

```
plotRuleActivation( u,y,model, 'figure', 4 );
set(gcf,'WindowState','maximized');
```



Show the parameter of the resulting TS model:

```
disp( model )
```

```
TS-Model: Type=Static
Name: 'undefined'
 Type: 'TSModel'
 Date: '29-Mar-2021 15:49:07'
 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=Euclidean in product space
 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: $[u_1, u_2]$

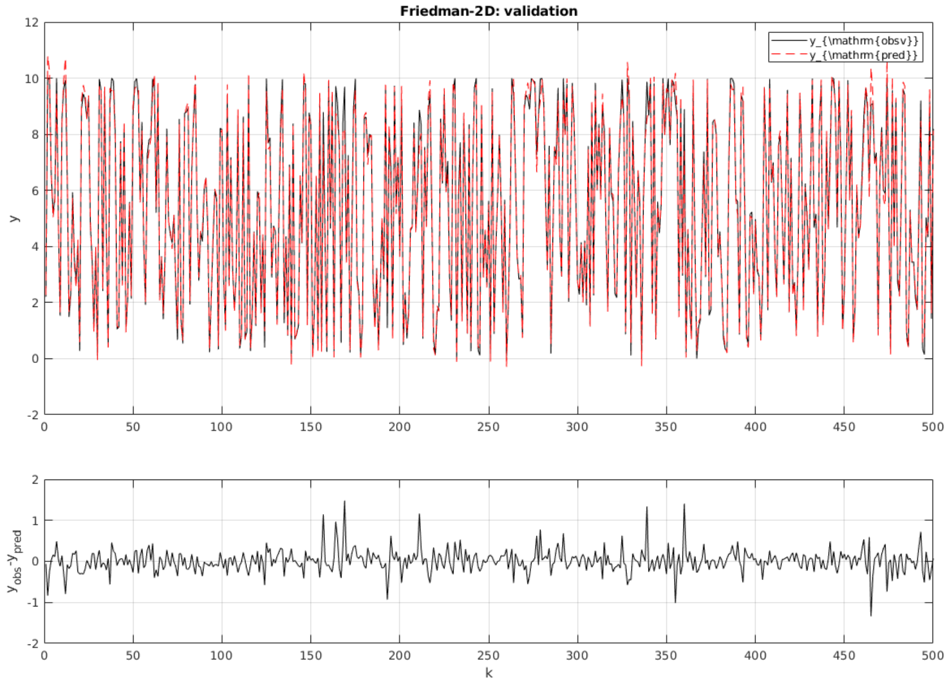
```
u_val = rand( N, nu );
y_val_obsv = Friedman_fct( u_val, nu );
```

Compute output vector $y_{\mathrm{val,pred}}$

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

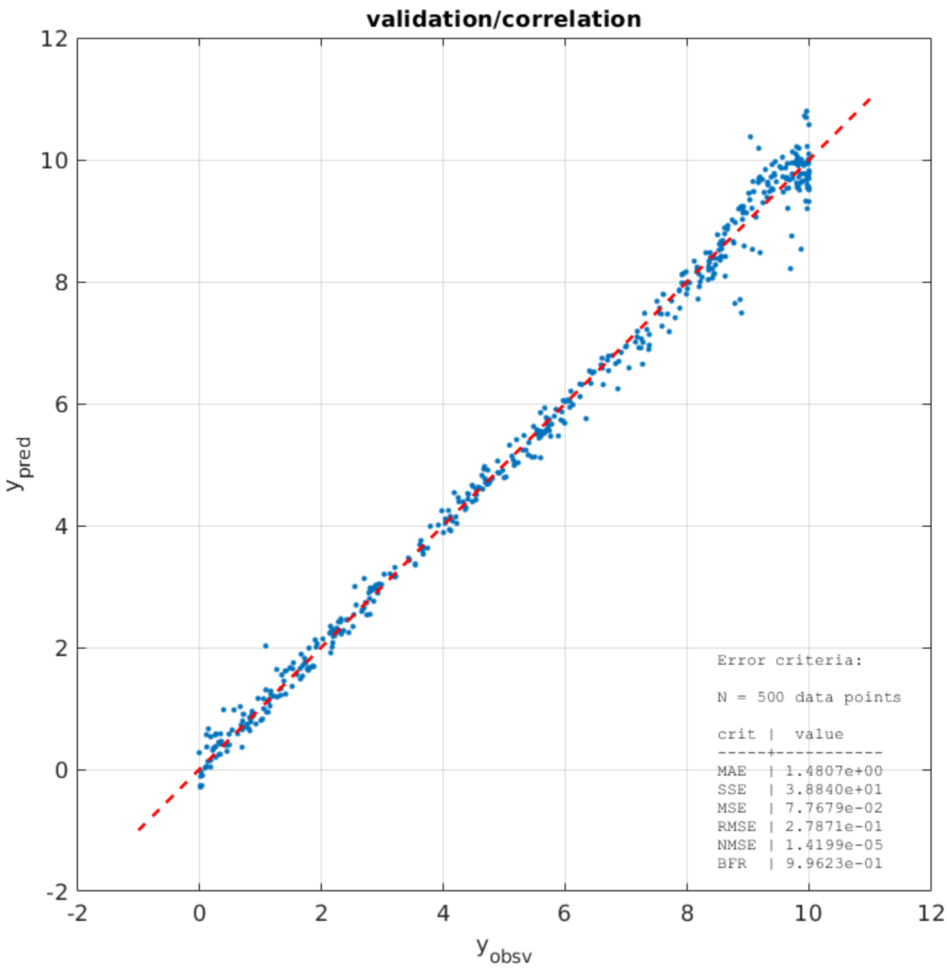
Plot the TS model with the validation data

```
figure(3),clf
subplot(3,1,1:2)
plot( 1:N, y_val_obsv, 'k-',1:N, y_val_pred, 'r--' )
grid on
ylabel('y')
title( 'Friedman-2D: validation' )
legend( 'y_{\mathrm{obsv}}','y_{\mathrm{pred}}' )
subplot(3,1,3)
plot( 1:N, y_val_obsv- y_val_pred, 'k-' )
grid on
ylabel( 'y_{obs}-y_{pred}')
xlabel('k')
set(gcf,'WindowState','maximized');
```



Plot the correlation for the validation data

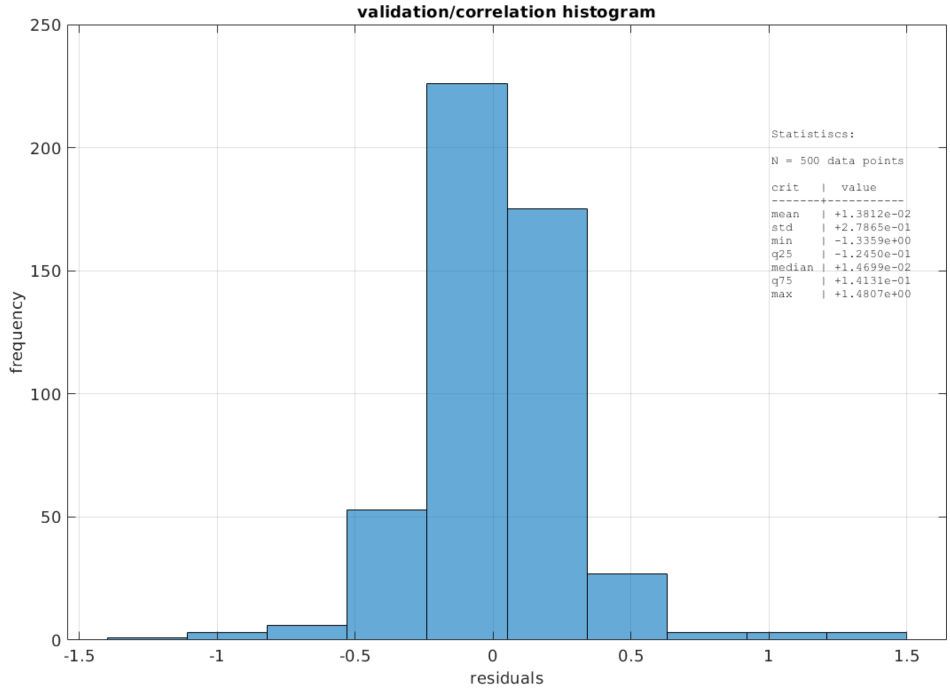
```
plotResiduals( y_val_obsv, y_val_pred, 'figure', 4, ...
     'title', 'validation/correlation' );
set(gcf,'WindowState','maximized');
```



Plot a residual histogram for the validation data:

```
plotResidualHist( y_val_obsv, y_val_pred, 'figure', 5, ...
```

```
    'title', 'validation/correlation histogram' );
set(gcf,'WindowState','maximized');
```



Plot predicted vs. observed y in 3D

```
figure(6),clf
plot3(u_val(:,1),u_val(:,2),y_val_obsv,'k.',...
      u_val(:,1),u_val(:,2),y_val_pred,'r.')
view(50,25)
  grid on
xlabel('u_1')
ylabel('u_2')
zlabel('y')
title( 'validation' )
set(gcf,'WindowState','maximized');
```