Improved Covariance Estimation for Gustafson-Kessel Clustering

R. Babuška[1] P.J. van der Veen[1] U. Kaymak[2]
[1] Delft University of Technology, Faculty ITS, Control Systems Engineering Group
P.O. Box 5031, 2600 GA Delft, the Netherlands, e-mail: R.Babuska@its.tudelft.nl
[2]Faculty of Economics, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR, Rotterdam
the Netherlands,u.kaymak@ieee.org

Abstract - This article presents two techniques to improve the calculation of the fuzzy covariance matrix in the Gustafson-Kessel (GK) clustering algorithm. The first one overcomes problems that occur in the standard GK clustering when the number of data samples is small or when the data within a cluster are linearly correlated. The improvement is achieved by fixing the ratio between the maximal and minimal eigenvalue of the covariance matrix. The second technique is useful when the GK algorithm is employed in the extraction of Takagi–Sugeno fuzzy model from data. It reduces the risk of overfitting when the number of training samples is low in comparison to the number of clusters. This is achieved by adding a scaled unity matrix to the calculated covariance matrix. Numerical examples are presented to demonstrate the benefits of the proposed techniques.

I. Introduction

The Gustafson-Kessel (GK) algorithm [1] is a powerful clustering technique with a large number of applications in various domains including image processing, classification and system identification [2], [3]. Its main feature is the local adaptation of the distance metric to the shape of the cluster by estimating the cluster covariance matrix and adapting the distance-inducing matrix correspondingly.

However, numerical problems frequently occur in the standard GK clustering when the number of data samples (in some clusters) is small or when the data within a cluster are (nearly) linearly correlated. In such a case, the cluster covariance matrix becomes singular and cannot be inverted to compute the norm-inducing matrix. This article presents a method to overcome this singularity problem by fixing the ratio between the maximal and minimal eigenvalue of the covariance matrix. As demonstrated by examples, this simple modification significantly improves the performance of the GK algorithm.

Fuzzy clustering can also be used to extract fuzzy if–then rules from data. The ability of the GK algorithm to estimate local covariance and to partition data into subsets that can be well fitted with linear sub-models makes it useful for the identification of Takagi–Sugeno (TS) models [4], [3]. The second technique proposed in this paper is useful when the GK algorithm is employed in the extraction of TS rules from data. It reduces the risk of overfitting when the number of training samples is low relative to the number of clusters. This is achieved by adding a scaled unity matrix to the calculated covariance matrix. An application example is presented to demonstrate the benefits of the proposed techniques.

II. Gustafson–Kessel Clustering

The Gustafson–Kessel [1] algorithm is based on iterative optimization of an objective functional of the c-means type: [5], [6]:

$$J(\mathbf{Z};\mathbf{U},\mathbf{V},\{\mathbf{A}_i\}) = \sum_{i=1}^{K}\sum_{k=1}^{N}(\mu_{ik})^m D^2_{ik\mathbf{A}_i}\,. \qquad (1)$$

Here, $\mathbf{U} = [\mu_{ik}] \in [0,1]^{K\times N}$ is a fuzzy partition matrix of the data $\mathbf{Z} \in \mathbb{R}^{n\times N}$, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K]$, $\mathbf{v}_i \in \mathbb{R}^n$ is a K-tuple of cluster prototypes and $m \in [1,\infty)$ is a scalar parameter which determines the fuzziness of the resulting clusters. The distance norm $D_{ik\mathbf{A}_i}$ can account for clusters of different geometrical shapes in one data set:

$$D^2_{ik\mathbf{A}_i} = (\mathbf{z}_k - \mathbf{v}_i)^T\mathbf{A}_i(\mathbf{z}_k - \mathbf{v}_i)\,. \qquad (2)$$

The metric of each cluster is defined by a local norm-inducing matrix $\mathbf{A}_i$, which is used as one of the optimization variables in the functional (1). This allows the distance norm to adapt to the local topological structure of the data. The minimization of the GK objective functional is achieved by using the alternating optimization method according to the following well-known algorithm.

Given the data set $\mathbf{Z}$, choose the number of clusters $1 < K < N$, the weighting exponent $m > 1$ (usually 2), the termination tolerance $\epsilon > 0$ (usually 10^{-3}) and the cluster volumes ρ_i (usually 1). Initialize the partition matrix randomly, such that $\mathbf{U}^{(0)} \in M_{fK}$ (i.e., belongs to the fuzzy partitioning space [6]).

0-7803-7280-8/02/$10.00 ©2002 IEEE

Repeat for $l = 1, 2, \ldots$

Step 1: Compute cluster prototypes (means):

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m \mathbf{z}_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \quad 1 \le i \le K\,.$$

Step 2: Compute the cluster covariance matrices:

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m (\mathbf{z}_k - \mathbf{v}_i^{(l)})(\mathbf{z}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \quad 1 \le i \le K\,.$$

Step 3: Compute the distances:

$$D_{ik\mathbf{A}_i}^2 = (\mathbf{z}_k - \mathbf{v}_i^{(l)})^T \left[\rho_i \det(\mathbf{F}_i)^{1/n} \mathbf{F}_i^{-1}\right] (\mathbf{z}_k - \mathbf{v}_i^{(l)}), \quad 1 \le i \le K, \quad 1 \le k \le N\,.$$

Step 4: Update the partition matrix:
for $1 \le k \le N$
if $D_{ik\mathbf{A}_i} > 0$ for $1 \le i \le K$,

$$\mu_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{K} (D_{ik\mathbf{A}_i} / D_{jk\mathbf{A}_j})^{2/(m-1)}},$$

otherwise

$$\mu_{ik}^{(l)} = 0 \text{ if } D_{ik\mathbf{A}_i} > 0, \text{ and } \mu_{ik}^{(l)} \in [0,1] \text{ with} \sum_{i=1}^{K} \mu_{ik}^{(l)} = 1 \text{ otherwise}\,.$$

until $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \epsilon$.

The above-mentioned numerical problems occur in Step 3 of the algorithm where the cluster covariance matrix $\mathbf{F}_i$ is inverted. If the number of data samples is small or when the data within a cluster are linearly correlated, the covariance matrix may become (nearly) singular. The following section presents a simple and effective solution to this problem.

III. Singularity of the covariance matrix

Recall that the eigenvalues and eigenvectors of the covariance matrix describe the shape and orientation of the clusters, see Fig. 1. When an eigenvalue is zero or when the ratio between the maximal and the minimal eigenvalue, i.e., the condition number of $\mathbf{F}$, is very large (say 10^{20}) the matrix is nearly singular. In such a case, the inverse in Step 3 cannot be calculated. Also the normalization to a fixed volume fails, as the determinant (the volume of the covariance matrix) becomes zero and the following formula thus cannot be applied:

$$\det(\mathbf{F}_i)^{1/n} \mathbf{F}_i^{-1}\,. \tag{3}$$

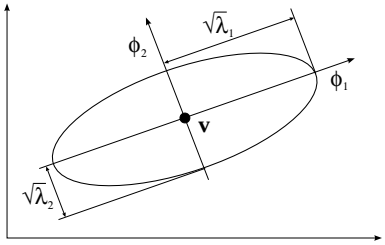


Fig. 1. Equation $(\mathbf{z}-\mathbf{v})^T \mathbf{F}^{-1} (\mathbf{z}-\mathbf{v}) = 1$ defines a hyperellipsoid. The length of the jth axis of this hyperellipsoid is given by $\sqrt{\lambda_j}$ and its direction is spanned by $\boldsymbol{\phi}_j$, where λ_j and $\boldsymbol{\phi}_j$ are the jth eigenvalue and the corresponding eigenvector of $\mathbf{F}$, respectively.

A straightforward way to avoid numerical problems is to constrain the ratio between the maximal and minimal eigenvalue such that it is smaller than some predefined threshold (in our examples, we used 10^{15}). When this threshold is exceeded, the minimal eigenvalue is increased such that the ratio equals to the threshold and the covariance matrix is reconstructed by:

$$\mathbf{F} = \boldsymbol{\Phi} \boldsymbol{\Lambda} \boldsymbol{\Phi}^{-1}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the limited eigenvalues and $\boldsymbol{\Phi}$ is a matrix whose columns are the corresponding eigenvectors.

Figure 2 shows an example of a data set that cannot be clustered with the standard GK algorithm, because of numerical problems. The reason is that the data samples in the three linear segments are completely correlated. By using the above technique, numerical problems are avoided and the improved GK algorithm finds the expected partition into three linear segments. The principal directions of the clusters perfectly coincide with the data. This model then explains 99.85% of the variance in the data.

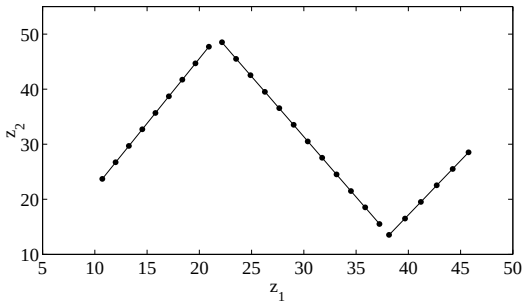


Fig. 2. Example of a data set with linear clusters.

IV. Overfitting problem

The above modification prevents the GK algorithm from running into numerical problems. However, as a result one can get clusters that are extremely long in the direction of the

largest eigenvalues and have little relationship with the real distribution of the data. This can cause overfitting of the data and consequently one obtains a poor model.

This problem occurs mainly when the number of data points in a cluster becomes too low. In such a case, the computed covariance matrix is not a reliable estimate of the underlying data distribution [7]. One way to tackle this problem is to limit the ratio between the maximal and minimal eigenvalues even further than described in the previous section. This will prevent the extreme elongation of the clusters. Another way is to add a scaled identity matrix to the covariance matrix. Tadjudin [7] and Friedman [8] describe several different methods to improve the covariance estimation. Inspired by these methods, we propose the following estimate for the GK algorithm:

$$\mathbf{F}_i^{\text{new}} = (1-\gamma)\mathbf{F}_i + \gamma \det(\mathbf{F}_0)^{1/n}\mathbf{I}, \tag{4}$$

where $\gamma \in [0,1]$ is a tuning parameter and $\mathbf{F}_0$ is the covariance matrix of the whole data set. Depending on the value of γ, the clusters are forced to have a more or less equal shape. When γ is 1, all the covariance matrices are equal and have the same size, which of course limits the possibility of the algorithm to properly identify clusters.

As $\mathbf{F}$ is based on the whole data set, its value does not depend on the number of clusters. The volumes of $\mathbf{F}_i$, however, decrease with the increasing number of clusters. This means that an increase of the number of clusters makes the clusters rounder. The term $\det(\mathbf{F})^{1/n}$ is included to reduce the tuning effort involved. The formula scales with the included volume of the total data set.

A slight disadvantages of this method is the extra tuning parameter γ. However, when using the GK algorithm to extract fuzzy models from off-line data, it is usually not a problem to include an extra parameter and tune it in cross-validation runs. Furthermore, it is expected that the performance of the clustering algorithm will decrease when there are sufficient training data available to construct the covariance matrix.

V. Modified Gustafson-Kessel algorithm

The complete Gustafson-Kessel algorithm including the two above modifications is given below.

Given the data set $\mathbf{Z}$, choose the standard parameters K, m, ϵ, ρ_i, the condition number threshold β and the weighting parameter γ. Initialize the partition matrix and compute the covariance matrix $\mathbf{F}_0$ of the whole data set.

Repeat for $l = 1, 2, \ldots$

Step 1: Compute cluster prototypes (means):

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m \mathbf{z}_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \quad 1 \le i \le K\,.$$

Step 2: Compute the cluster covariance matrices:

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m (\mathbf{z}_k - \mathbf{v}_i^{(l)})(\mathbf{z}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, \quad 1 \le i \le K\,.$$

Add a scaled identity matrix:

$$\mathbf{F}_i := (1-\gamma)\mathbf{F}_i + \gamma \det(\mathbf{F}_0)^{1/n}\mathbf{I},$$

Extract eigenvalues λ_{ij} and eigenvectors ϕ_{ij} from $\mathbf{F}_i$. Find $\lambda_{i\max} = \max_j \lambda_{ij}$ and set:

$$\lambda_{ij} = \lambda_{i\max}/\beta \quad \forall j \text{ for which } \lambda_{i\max}/\lambda_{ij} > \beta$$

Reconstruct $\mathbf{F}_i$ by

$$\mathbf{F}_i = [\phi_{i1}\ \ldots\ \phi_{in}]\,\mathrm{diag}(\lambda_{i1}, \ldots, \lambda_{in})[\phi_{i1}\ \ldots\ \phi_{in}]^{-1}$$

Step 3: Compute the distances:

$$D^2_{ik\mathbf{A}_i} = (\mathbf{z}_k - \mathbf{v}_i^{(l)})^T \left[\rho_i \det(\mathbf{F}_i)^{1/n}\mathbf{F}_i^{-1}\right](\mathbf{z}_k - \mathbf{v}_i^{(l)}), \quad 1 \le i \le K, \quad 1 \le k \le N\,.$$

Step 4: Update the partition matrix:

for $1 \le k \le N$

if $D_{ik\mathbf{A}_i} > 0$ for $1 \le i \le K$,

$$\mu_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{K} (D_{ik\mathbf{A}_i}/D_{jk\mathbf{A}_j})^{2/(m-1)}},$$

otherwise

$$\mu_{ik}^{(l)} = 0 \text{ if } D_{ik\mathbf{A}_i} > 0, \text{ and } \mu_{ik}^{(l)} \in [0,1] \text{ with } \sum_{i=1}^{K} \mu_{ik}^{(l)} = 1 \text{ otherwise}\,.$$

until $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \epsilon$.

In a double-precision floating point implementation (e.g., under MATLAB), the condition number threshold β will typically be set to a large number, such as 10^{15}. The setting of the weighting parameter γ is application dependent and some experimentation may be needed to find the right value. A MATLAB) code of the algorithm is given in the Appendix.

VI. Application example

In this section we use the GK clustering algorithm to construct Tagaki-Sugeno [4] fuzzy models from data by using the method described in [3]. The process under study is an enzymatic Penicillin–G conversion. The fuzzy model describes

how the enzyme kinetics depend on the concentrations of the components involved in the conversion.

We consider the enzymatic conversion (hydrolysis) of Penicillin–G (PenG) to 6–aminopenicillanic acid (APA) and phenyl acetic acid (PhAH) at pH of 8.0 and temperature of 310 K by the enzyme penicillin acylase. It is expected that the conversion rate depends on the concentrations of PenG, APA and PhAH in a nonlinear way and is proportional to the enzyme concentration E:

$$r = \mathrm{E} \cdot f(\mathrm{PenG}, \mathrm{APA}, \mathrm{PhAH})$$

where the nonlinear function f, is unknown. Data from 10 batch experiments started at different initial conditions were used to construct a TS fuzzy model for f. The data were obtained from laboratory experiments performed in a stirred, thermostated laboratory bioreactor with a volume of 1500 cm^3. The laboratory set-up is depicted in Fig. 3. More details about this process can be found in [9].

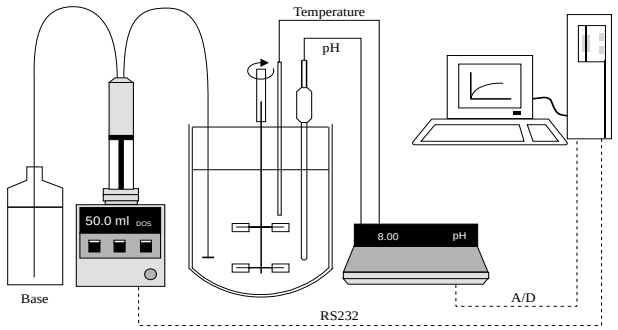


Fig. 3. Experimental setup.

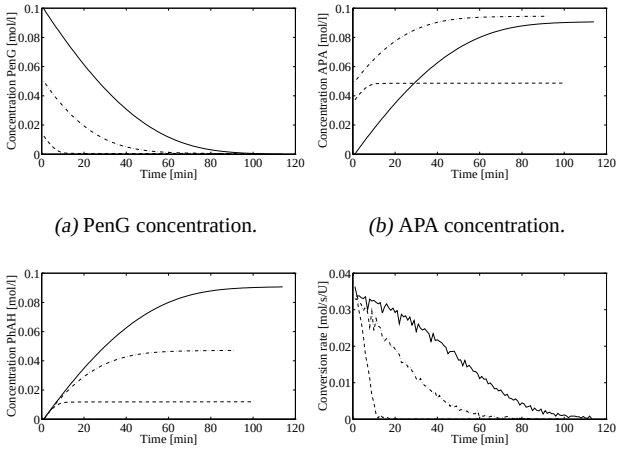
In total, 816 data points were available for identification. Additional experiments were carried out to validate the model. Some typical experiments are shown in Fig. 4.

The TS rules are of the following form:

$$\textbf{If } \mathrm{PenG} \text{ is } A_{i1} \textbf{ and } \mathrm{APA} \text{ is } A_{i2} \textbf{ and } \mathrm{PhAH} \text{ is } A_{i3}$$
$$\textbf{then } r_e = \mathbf{a}_i^T [\mathrm{PenG\ APA\ PhAH}]^T + b_i \,.$$

The membership functions A_i, and the consequent parameters $\mathbf{a}_i$, b_i are found through GK clustering in the Cartesian product space $Z = \mathrm{PenG} \times \mathrm{APA} \times \mathrm{PhAH} \times r_e$.

Note that the identification experiments were not sufficiently exciting the system and the data may be highly correlated in the four-dimensional clustering space Z. This is confirmed by observing the results of the standard GK algorithm, which fails to find clusters for $K > 2$ (see Table I, where '–' means that the clustering failed due to numerical problems). By limiting the maximal ratio of the eigenvalues, the numerical problems are avoided. With more than 3 clusters, however, the performance of the obtained model is decreasing. When the identity matrix is added according to equation



(a) PenG concentration. *(b)* APA concentration.

(c) PhAH concentration. *(d) Conversion rate* r_e.

Fig. 4. Experimental data from some typical batch experiments. The three curves in each graph represent experiments started from different initial conditions.

(4), the performance of the model remains approximately at the same level. The same is true when the eigenvalues are limited.

TABLE I
PERFORMANCE OF THE FUZZY MODEL WITH THE DIFFERENT COVARIANCE ESTIMATIONS METHODS.

	2 clust.	3 clust.	4 clust.	5 clust.
original	95.4 %	–	–	–
cond. number	95.4 %	95.9 %	80.8 %	79.8 %
add diag. matrix	95.1 %	91.9 %	97.3 %	95.9 %
limit eigenvalue	95.2 %	96.5 %	97.7 %	93.6 %

All parameters were kept the same for the different methods (even the random initialization of the algorithm). The model output is calculated using 4-fold cross-validation. The performance of the resulting models is compared by means of the variance accounted for (VAF) index, defined by:

$$\mathrm{VAF} = \left(1 - \frac{\mathrm{var}(y - y_m)}{\mathrm{var}(y)}\right) \cdot 100\%$$

where y is the measured output of the system and y_m is the output of the model. A VAF of 100% means a perfect model prediction.

VII. Conclusions

In this article we proposed a method to avoid numerical problems that may occur when computing the norm-inducing matrix in the Gustafson-Kessel (GK) clustering algorithm.

This solution does not change the performance of the algorithm but guarantees that it is always able to find a partitioning of the data.

When using the GK clustering algorithm to construct Tagaki-Sugeno fuzzy models, a certain degree of overfitting will be experienced for larger numbers of clusters. In such a case, the performance can be improved by further limiting the maximal ratio between the eigenvalues of the covariance matrix or by adding a scaled identity matrix to the covariance matrix. While these latter modifications can improve the performance for small data sets, with a sufficient number of training samples, this restriction in the freedom of the algorithm may have an adverse effect and on the performance. Some experimentation with the weighting parameter γ may thus be needed. GK clustering and fuzzy identification software for MATLAB can be downloaded from http://Lcewww.et.tudelft.nl/~babuska.

Acknowledgement

The authors thank the Kluyver Laboratory for Biotechnology of the Delft University of Technology for providing the data.

APPENDIX: Gustafson-Kessel algorithm

In this appendix we give a MATLAB implementation the modified GK algorithm. Send an e-mail to R.Babuska@its.tudelft.nl to receive a copy of the M-file.

```
function [U,V,F] = gk(Z,U0,m,tol,beta,gamma)
% Numerically robust Gustafson-Kessel algorithm
%
% [U,V,F] = GK(Z,U0,m,tol,beta,gamma)
%-----------------------------------------------
% Input:  Z     ... N by n data matrix
%         U0    ... initial fuzzy partition matrix
%                   or the number of clusters
%         m     ... fuzziness exponent (m > 1)
%         tol   ... termination tolerance
%         beta  ... condition number threshold
%         gamma ... weighting for covariance
%-----------------------------------------------
% Output: U     ... fuzzy partition matrix
%         V     ... cluster means (centers)
%         F     ... cluster covariance matrices

%----------------- prepare matrices --------------
[mz,nz] = size(Z);              % data size
c = size(U0,2);
if c == 1, c = U0; end;         % # of clusters
mZ1 = ones(mz,1);               % aux. variable
nZ1 = ones(nz,1);               % aux. variable
V1c = ones(1,c);                % aux. variable
U = zeros(mz,c);                % partition matr.
d = U;                          % distance matrix
F = zeros(nz,nz,c);             % covariance matr.
f0=eye(nz)*det(cov(Z)).^(1/nz); % "identity" matr.
%----------------- initialize U ------------------
if size(U0,2) == 1,
  minZ = V1c'*min(Z); maxZ = V1c'*max(Z);
  V = minZ + (maxZ-minZ).*rand(c,nz);
  for j = 1 : c,
    ZV = Z - mZ1*V(j,:);
    d(:,j) = sum((ZV.^2)')';
  end;
  d = (d+1e-100).^(-1/(m-1));
  U0 = (d ./ (sum(d')'*V1c));
end;
%----------------- iterate -----------------------
while  max(max(abs(U0-U))) > tol
  U = U0; Um = U.^m; sumU = sum(Um);
  V = (Um'*Z)./(nZ1*sumU)';
  for j = 1 : c,
    ZV = Z - mZ1*V(j,:);
    f = nZ1*Um(:,j)'.*ZV'*ZV/sumU(j);
    f=(1-gamma)*f+gamma*f0;
    if cond(f)>beta;
      [ev,ei]=eig(f); eimax = max(diag(ei));
      ei(beta*ei < eimax) = eimax/beta;
      f=ev*diag(diag(ei))*inv(ev);
    end;
    d(:,j)=sum((ZV*(det(f)^(1/nz)*inv(f)).*ZV)')';
  end;
  d = (d+1e-100).^(-1/(m-1));
  U0 = (d ./ (sum(d')'*V1c));
end
%----------------- create final F and U ----------
Um = U0.^m; sumU = nZ1*sum(Um);
for j = 1 : c,
  ZV = Z - mZ1*V(j,:);
  F(:,:,j) = nZ1*Um(:,j)'.*ZV'*ZV/sumU(1,j);
end;
```

References

[1] D.E. Gustafson and W.C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proc. IEEE CDC*, pages 761–766, San Diego, CA, USA, 1979.

[2] R.N. Dave. Boundary detection through fuzzy clustering. In *IEEE International Conference on Fuzzy Systems*, pages 127–134, San Diego, USA, 1992.

[3] R. Babuška. *Fuzzy Modeling for Control*. Kluwer Academic Publishers, Boston, USA, 1998.

[4] T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[5] J.C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *J. Cybern.*, 3(3):32–57, 1974.

[6] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function*. Plenum Press, New York, 1981.

[7] S. Tadjudin and D.A. Landgrebe. Covariance estimation with limited training samples. *IEEE Transactions on Geoscience and Remote Sensing*, 37(4):2113–2118, July 1999.

[8] J.F. Friedman. Regularize discriminant analysis. *J.R. Statist. Soc.*, 84:17–42, 1989.

[9] R. Babuška, H.B. Verbruggen, and H.J.L. van Can. Fuzzy modeling of enzymatic penicillin–G conversion. *Engineering Applications of Artificial Intelligence*, 12(1):79–92, 1999.