Comparison of Approaches for Identification of Static Cloud-based Evolving Systems

Sašo Blažič * **Igor Škrjanc** ** **Plamen Angelov** ***

* *Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, Ljubljana, Slovenia (e-mail: saso.blazic@fe.uni-lj.si).*
** *Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, Ljubljana, Slovenia (e-mail: igor.skrjanc).*
*** *School of Computing and Communications, Lancaster University, United Kingdom, (e-mail: p.angelov@lancaster.ac.uk).*

Abstract: This paper presents an alternative way of describing local density in the cloud-based evolving systems. The Mahalanobis distance among the data samples is used which leads to the density that is more suitable when the data are scattered around the input-output surface. All the algorithms for the identification of the cloud parameters are given in a recursive form which is necessary for the implementation of the evolving systems. It is also shown that a simple linearised model can be obtained without identification of the consequent parameters. All the proposed algorithms are illustrated on a simple simulation model of a static system.

Keywords: Identification, evolving systems, Mahalanobis distance, Takagi-Sugeno model, clusters

1. INTRODUCTION

Recently, a special type of fuzzy rule-based (FRB) systems with non-parametric antecedents has been proposed by Angelov and Yager (2010). Unlike traditional Mamdani and Takagi-Sugeno FRB systems, the approach does not require an explicit definition of fuzzy sets (and their corresponding membership functions) for each input variable. It introduced the concept of granules in Angelov and Yager (2010) and later clouds (Angelov and Yager, 2011) that rely on relative data density to define antecedents. Data clouds are subsets of previous data samples with common properties. In the original works (Angelov and Yager, 2010, 2011) data closeness has been used as a similarity measure. The approach itself is not limited to any particular similarity measure to classify data into clouds. In identification of dynamical systems it is very important to distinguish among the operating regions that represent different system dynamics. Those regions could be seen as natural clouds. Even if we choose to select the framework of cloud based system identification, there are still a number of subtasks that have to executed. There are also some possible changes that can be introduced to the original method while still keeping the general methodology.

The relative density in the original papers (Angelov and Yager, 2010, 2011) was based on Euclidean distance among the data samples in the cloud although it was stated that any other distance could be used. In the current paper two distance metrics are compared: the original Euclidean distance and Mahalanobis distance where we introduced some versions for calculating actual density.

2. TAKAGI-SUGENO FUZZY MODEL OF A NONLINEAR SYSTEM

A typical Takagi-Sugeno fuzzy model (Takagi and Sugeno, 1985) is given in the form of rules:

$$\textbf{if } z_1 \textbf{ is } \mathbf{A}_{1,k_1} \ \dots \ \textbf{and } z_q \textbf{ is } \mathbf{A}_{q,k_q} \textbf{ then } y = \phi_j(\mathbf{x})$$
$$j = 1, \dots, m \quad k_1 = 1, \dots, f_1 \qquad k_q = 1, \dots, f_q \quad (1)$$

The q-element vector $\mathbf{z}^T = [z_1, ..., z_q]$ denotes the input or variables in the antecedent part of the rules, and variable y is the output of the model. With each variable in the antecedent z_i $(i = 1, \dots, q)$, f_i fuzzy sets $(\mathbf{A}_{i,1}, \dots, \mathbf{A}_{i,f_i})$ are associated, and each fuzzy set $\mathbf{A}_{i,k_i}$ $(k_i = 1, \dots, f_i)$ is associated with a real-valued function $\mu_{A_{i,k_i}}(z_i) : \mathbb{R} \to [0, 1]$, that produces membership grade of the variable z_i with respect to the fuzzy set $\mathbf{A}_{i,k_i}$. To make the list of fuzzy rules complete, all possible variations of fuzzy sets are given in Eq. (1), yielding the number of fuzzy rules $m = f_1 \times f_2 \times \cdots \times f_q$. The variables z_i are not the only inputs of the fuzzy system. Implicitly, the n-element vector $\mathbf{x}^T = [x_1, ..., x_n]$ also represents the input to the system. It is usually referred to as the consequence vector. The functions $\phi_j(\cdot)$ can be arbitrary smooth functions in general, although linear or affine functions are usually used.

The system in Eq. (1) is easily described in the closed form in the case of a product-sum Takagi-Sugeno fuzzy model

$$y = \frac{\sum_{k_1=1}^{f_1} \cdots \sum_{k_q=1}^{f_q} \mu_{A_{1,k_1}}(z_1) \dots \mu_{A_{q,k_q}}(z_q)\ \phi_j(\mathbf{x})}{\sum_{k_1=1}^{f_1} \cdots \sum_{k_q=1}^{f_q} \mu_{A_{1,k_1}}(z_1) \dots \mu_{A_{q,k_q}}(z_q)} \quad (2)$$

Note a slight abuse of notation in Eq. (2) since j is not explicitly defined as a running index. From Eq. (1) it is

evident that each j corresponds to a specific variation of indexes k_i, $i = 1, \ldots, q$.

To simplify Eq. (2), a partition of unity is considered where functions $\beta_j(\mathbf{z})$ defined as

$$\beta_j(\mathbf{z}) = \frac{\mu_{A_{1,k_1}}(z_1) \ldots \mu_{A_{q,k_q}}(z_q)}{\sum_{k_1=1}^{f_1} \cdots \sum_{k_q=1}^{f_q} \mu_{A_{1,k_1}}(z_1) \ldots \mu_{A_{q,k_q}}(z_q)} \quad j = 1, \ldots, m \quad (3)$$

give information about the fulfilment of the respective fuzzy rule in the normalized form. It is obvious that $\sum_{j=1}^{m} \beta_j(\mathbf{z}) = 1$ irrespective of $\mathbf{z}$ as long as the denominator of $\beta_j(\mathbf{z})$ is not equal to zero (this can be easily prevented by stretching the membership functions over the whole potential area of $\mathbf{z}$). Combining Eqs. (2) and (3) and changing summation over k_i by summation over j we arrive to the following equation:

$$y = \sum_{j=1}^{m} \beta_j(\mathbf{z}) \phi_j(\mathbf{x}) \quad (4)$$

From Eq. (4) it is evident that the output of a fuzzy system is a function of the antecedent vector $\mathbf{z}$ (q-dimensional) and the consequence vector $\mathbf{x}$ (n-dimensional). The dimension of the input space d may be and usually is lower than $(q + n)$ since it is very usual to have the same variables present in vectors $\mathbf{z}$ and $\mathbf{x}$.

The class of fuzzy models have the form of linear models, this refers to $\{\beta_j\}$ as a set of basis functions. The use of membership functions in input space with overlapping receptive fields provides interpolation and extrapolation. It is very common to define the output value as a linear combination of consequence variables $\mathbf{x}$

$$\phi_j(\mathbf{x}) = \boldsymbol{\theta}_j^T \mathbf{x}, \quad j = 1, \ldots, m, \quad \boldsymbol{\theta}_j^T = [\theta_{j1}, \ldots, \theta_{jn}] \quad (5)$$

If the matrix of the coefficients for the whole set of rules is denoted as $\boldsymbol{\Theta}^T = [\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_m]$ and the vector of membership values as $\boldsymbol{\beta}^T(\mathbf{z}) = [\beta_1(\mathbf{z}), \ldots, \beta_m(\mathbf{z})]$, then Eq. (4) can be rewritten in the matrix form

$$y = \boldsymbol{\beta}^T(\mathbf{z}) \boldsymbol{\Theta} \mathbf{x} = \sum_{j=1}^{m} \beta_j(\mathbf{z}) \boldsymbol{\theta}_j^T \mathbf{x} \quad (6)$$

A fuzzy model in the form given in Eq. (6) is referred to as an affine Takagi-Sugeno model and can be used to approximate any arbitrary function that maps any compact set $\mathbf{C} \subset \mathbb{R}^d$ from the input space (the input space is the space of the union of variables in $\mathbf{x}$ and $\mathbf{z}$) to $\mathbb{R}$ with any desired degree of accuracy.

3. IDENTIFICATION OF THE ANTECEDENT PART

The local density is defined by a suitable kernel over the distance between the current sample $\mathbf{z}(k)$ and all the previous samples that have already been classified to a particular cloud (j-th in this case) (Angelov and Yager, 2011):

$$\gamma_k^j = \frac{1}{1 + \rho \frac{\sum_{i=1}^{M^j} d_{ki}^j}{M^j}} \quad j = 1, ..., m \quad (7)$$

where d_{ki}^j denotes the square of the distance between the current data sample $\mathbf{z}(k)$ and the i-th sample of the j-th cloud $\mathbf{z}_i^j$, while M^j is the number of input data samples associated with the j-th cloud. Note the factor ρ which is not present in Angelov and Yager (2011) and will be discussed later.

3.1 Density based on Mahalanobis distance

Mahalanobis distance is conceptually different. It is defined between an observation and a group of observations. The latter is characterised with its mean and the corresponding covariance matrix. In our case the distance will be calculated between two samples but taking into account the covariance matrix of the cloud data samples. The mean value of the samples in the j-th cloud will be denoted with $\boldsymbol{\mu}^j$ while the associated covariance matrix will be denoted with $\boldsymbol{\Sigma}^j$. The square of the distance between the current data sample $\mathbf{z}(k)$ and the i-th sample of the j-th cloud ($\mathbf{z}_i^j$) can therefore be computed as

$$d_{ki}^j = (\mathbf{z}(k) - \mathbf{z}_i^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}(k) - \mathbf{z}_i^j) \quad (8)$$

where the lower index in $\boldsymbol{\mu}^j(k)$ and $\boldsymbol{\Sigma}^j(k)$ gives the number of data samples taken into account during their calculation:

$$\begin{aligned} \boldsymbol{\mu}_{M^j}^j &= \frac{1}{M^j} \sum_{i=1}^{M^j} \mathbf{z}_i^j \\ \boldsymbol{\Sigma}_{M^j}^j &= \frac{1}{M^j - 1} \sum_{i=1}^{M^j} (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)(\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)^T \end{aligned} \quad (9)$$

By introducing (8) into (7) we obtain the non-recursive formula for density calculation:

$$\gamma_k^j = \frac{1}{1 + \rho \frac{\sum_{i=1}^{M^j} (\mathbf{z}(k) - \mathbf{z}_i^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}(k) - \mathbf{z}_i^j)}{M^j}} \quad (10)$$

Eq. (10) can be transformed into the recursive form by further developing the summation in it:

$$\begin{aligned} &\sum_{i=1}^{M^j} (\mathbf{z}(k) - \mathbf{z}_i^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}(k) - \mathbf{z}_i^j) = \\ &= \sum_{i=1}^{M^j} ((\mathbf{z}(k) - \boldsymbol{\mu}_{M^j}^j) - (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j))^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} \times \\ &\qquad \times ((\mathbf{z}(k) - \boldsymbol{\mu}_{M^j}^j) - (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)) = \\ &= M^j (\mathbf{z}(k) - \boldsymbol{\mu}_{M^j}^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}(k) - \boldsymbol{\mu}_{M^j}^j) - \\ &- 2 \sum_{i=1}^{M^j} (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}(k) - \boldsymbol{\mu}_{M^j}^j) + \\ &+ \sum_{i=1}^{M^j} (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)^T (\boldsymbol{\Sigma}_{M^j}^j)^{-1} (\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j) \end{aligned} \quad (11)$$

To try to simplify the expression further we need to fulfill the conditions for the covariance matrix inversion. First denote the matrix of all the vectors $(\mathbf{z}_i^j - \boldsymbol{\mu}_{M^j}^j)$ of the j-th cloud in its columns by $\boldsymbol{\Xi}_j$ (dimension $q \times M^j$). The matrix $\boldsymbol{\Sigma}_{M^j}^j$ from (9) is non-singular if and only if $\boldsymbol{\Xi}_j$ has rank q. Since all the rows in $\boldsymbol{\Xi}_j$ have zero mean, at least $q + 1$ columns are needed for the matrix $\boldsymbol{\Xi}_j$ to achieve full rank. If $M^j \geq q + 1$ and all the measurements are independent, the matrix $\boldsymbol{\Sigma}_{M^j}^j$ can be inverted (its inverse

is $(M^j-1)(\Xi_j\Xi_j^T)^{-1}$). Then it is easy to see that the last term in (11) is identical to $(M^j-1)\,\mathrm{trace}(\Xi_j^T(\Xi_j\Xi_j^T)^{-1}\Xi_j)$ which is in turn equal to $(M^j-1)q$ (q is the dimension of the antecedent vector). The term before last is identical to 0 due to the definition of $\boldsymbol{\mu}^j_{M^j}$ in (9). The formula for the relative density (10) therefore takes the form suitable for the recursive implementation:

$$\gamma_k^j = \frac{1}{1+\rho\left[(\mathbf{z}(k)-\boldsymbol{\mu}^j_{M^j})^T(\boldsymbol{\Sigma}^j_{M^j})^{-1}(\mathbf{z}(k)-\boldsymbol{\mu}^j_{M^j})+\frac{(M^j-1)q}{M^j}\right]} \tag{12}$$

The expression in square brackets in Eq. (12) is equivalent to the quadratic form in Eq. (10) and therefore always positive. Now it is properly to discuss the parameter ρ. Small values of ρ have similar effect as wide membership functions in the context of fuzzy systems. By increasing ρ, the membership functions become narrower.

The benefit of using Mahalanobis distance is to describe the ellipsoidally shaped clouds. In fact any cloud stretched in a certain direction can be described easier. The size and the shape of the ellipsoid depends on the covariance matrix of the data in the cloud. While the idea of having such clouds is appealing, it holds a caveat. If a cloud is based on some measurements in a small region of space, the covariance matrix $\boldsymbol{\Sigma}^j_{M^j}$ becomes small and a relatively close measurement may have low density with respect to this particular cloud. The problem lies in the fact that the volume of the cloud (or better of the ellipsoid defined by its covariance matrix) is too small. To prevent this phenomenon, the inverse of the covariance matrix in Eq. (12) is replaced by its normalised version:

$$\gamma_k^j = \frac{1}{1+\rho\left[\frac{(\mathbf{z}(k)-\boldsymbol{\mu}^j_{M^j})^T(\boldsymbol{\Sigma}^j_{M^j})^{-1}(\mathbf{z}(k)-\boldsymbol{\mu}^j_{M^j})}{\sqrt{\det\left((\boldsymbol{\Sigma}^j_{M^j})^{-1}\right)}}+\frac{(M^j-1)q}{M^j}\right]} \tag{13}$$

The normalising factor is the square root of the determinant of the matrix which is proportional to the volume of the ellipsoid. In practice one does not divide by the square root. Instead multiplication with $(\det\boldsymbol{\Sigma}^j_{M^j})^{\frac{1}{2}}$ is used.

It is also possible to only perform normalisation if $(\det\boldsymbol{\Sigma}^j_{M^j})^{\frac{1}{2}}$ falls below a certain threshold. Thus, automatic normalisation of "big" signals is still achieved via the Mahalanobis metric while over-shrinking of clouds is prevented.

Eq. (9) is not suitable for implementation in the recursive identification algorithm. This algorithm can be adapted for our purpose as follows. If a new data sample (say $\mathbf{z}$) is assigned to the j-th cloud, the update of the mean and the covariance matrix can be calculated using the Algorithm 1:

$M^j \leftarrow M^j+1$
$\mathbf{d} \leftarrow \mathbf{z}-\boldsymbol{\mu}^j$
$\boldsymbol{\mu}^j \leftarrow \boldsymbol{\mu}^j+\frac{1}{M^j}\mathbf{d}$
$\mathbf{S}^j \leftarrow \mathbf{S}^j+\mathbf{d}(\mathbf{z}-\boldsymbol{\mu}^j)^T$
$\boldsymbol{\Sigma}^j \leftarrow \frac{1}{M^j-1}\mathbf{S}^j$

All the states (M^j, $\boldsymbol{\mu}^j$, $\mathbf{S}^j$) of this algorithm are initialised with zeros. Note that the lower indexes are omitted in the algorithms due to the nature of the algorithm implementation.

In order to calculate the relative density (12) or (13), one needs the inverse of the covariance matrix. To avoid inverting the matrix in each sampling instant, Woodbury matrix identity is used to obtain the recursive form for the matrix inversion. Last two steps of Algorithm 1 therefore change in Algorithm 2:

$\bar{\mathbf{S}}^j \leftarrow \bar{\mathbf{S}}^j-\bar{\mathbf{S}}^j\mathbf{d}\left[1+(\mathbf{z}-\boldsymbol{\mu}^j)^T\bar{\mathbf{S}}^j\mathbf{d}\right]^{-1}(\mathbf{z}-\boldsymbol{\mu}^j)^T\bar{\mathbf{S}}^j$
$(\boldsymbol{\Sigma}^j)^{-1} \leftarrow (M^j-1)\bar{\mathbf{S}}^j$

Algorithm 2 introduces a new state $\bar{\mathbf{S}}^j$ as an inverse of $\mathbf{S}^j$ from Algorithm 1. It is initialised with a large positive definite matrix, usually a diagonal one. Note that the inverse in Algorithm 2 applies to a (positive) scalar and is not problematic. Note also that Algorithm 1 exactly reproduces the mean and the covariance matrix from (9) while Algorithm 2 also achieves this but there is a slight difference in the initialisation phase (the inversion of zero is undefined). After the full rank of the covariance matrix is achieved, all three algorithms become identical.

But since starting a new cloud with enough initial data is extremely important for robust operation, the above mentioned small difference is irrelevant. Enough initial data means that data are kept in a buffer before a decision for starting a new cloud is taken. Usually, this means that more than $q+1$ measurements are kept. Then, there is no need of initialising the inverse of the covariance matrix with a big positive matrix. Instead, real data from the buffer are used.

3.2 The determination of the input-output mapping

Any nonlinear mapping that maps a compact set from the input space to $\mathbb{R}$ can be approximated by a number of general approximators. One possibility is to use a Takagi-Sugeno model given in Eq. (6). The problem of identifying the model is a very well-known one and has been treated by many authors in the last decades. Most traditional approach is to somehow estimate the parameters $\boldsymbol{\theta}_j$ while simultaneous identification of parameters $\boldsymbol{\theta}_j$ and functions $\beta_j(\cdot)$ has also received quite some attention in the literature.

Here we will try to obtain a very simple and also not so accurate model by only analysing the covariance matrices of the clouds. We will assume that measurement vectors are composed of the input vector $\mathbf{x}$ and the corresponding output y:

$$\mathbf{z}^T = \begin{bmatrix}\mathbf{x}^T & y\end{bmatrix} \tag{14}$$

One possibility to obtain input-output mapping is to deduce it solely by analysing the input part of the FRB. For this purpose, the following idea is used. The data in the input-output space lie along the hyper-surface representing the input-output mapping. Due to disturbances, measurement noise, parasitic disturbances and other sources of errors, the data do not lie exactly on the surface, but are spread in the vicinity of the hyper-surface. Analysing the data in the cloud it turns out that the eigenvectors associated with the dominant eigenvalues lie along the hyper-surface while the smallest eigenvalue is associated with the eigenvector that is perpendicular to the the hyper-surface. For the j-th cloud, this normal vector is denoted by $\mathbf{n}_j$. This vector determines the tangential hyper-plane in the centre of the cloud. In the context of nonlinear

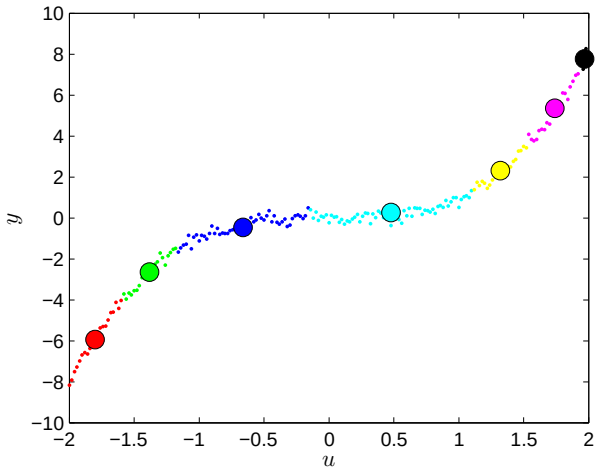

Fig. 1. The clouds obtained with the Euclidean distance in the density calculation

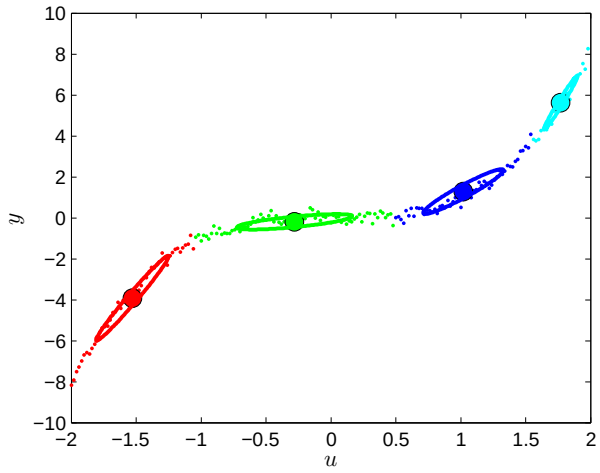

Fig. 2. The clouds obtained with the density given by (13)

systems, the normal vector to the hyper-plane changes from one operating point to another. The normal vector in a certain operating can be obtain by linear combination of individual normal vectors associated with individual clouds. The factors of the linear combination can also be normalised densities associated with the clouds:

$$\mathbf{n}_k = \frac{\sum_{j=1}^{m} \gamma_k^j \mathbf{n}_j}{\sum_{j=1}^{m} \gamma_k^j} \tag{15}$$

When a measurement $\mathbf{z}_k^T = \begin{bmatrix} \mathbf{x}_k^T & y_k \end{bmatrix}$ is obtained, a local linearised model can be obtained:

$$\begin{bmatrix} \mathbf{x}^T - \mathbf{x}_k^T & y - y_k \end{bmatrix} \mathbf{n}_k = 0 \tag{16}$$

This method enables obtaining the local linear model without the need for performing the identification of the consequent part of the FRB. The method also has some drawbacks due to the fact that the data inside a cloud usually do not lie along a hyper-plane and the required normal direction to the surface is contaminated with the direction of the nonlinearity in a certain direction.

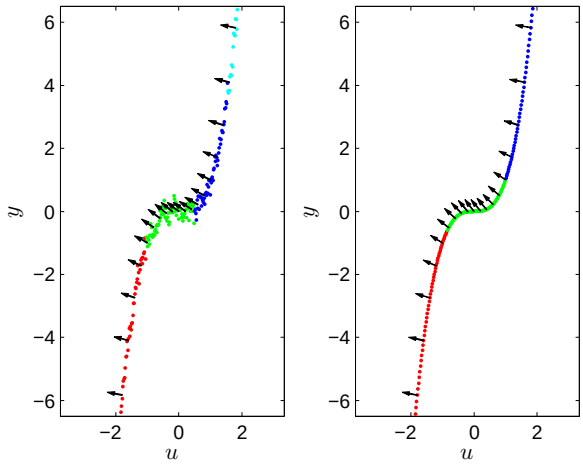

Fig. 3. The data in the clouds and the normal vector calculated from the eigenvectors by Eq. (15)

4. SIMULATION EXAMPLES

First the static system

$$y = u^3 \tag{17}$$

has been treated. The data collected from the system are depicted in Figs. 1 and 2. In Fig. 1 the data clouds obtained by calculating the densities using Euclidean distance are shown in different colours. Fig. 2 shows the results of the example where the density is calculated by the second version of the Mahalanobis distance (Eq. 13). The ellipses show the one standard deviation boundary. All other parameters are the same in both approaches: new cloud is started when the local density falls below 0.4, $\rho = \frac{1}{n} = \frac{1}{2}$. The first thing to note is that lower number of clouds is obtained in Fig. 2 which is understandable because the clouds adapt their shape to the data to a certain extent.

In Fig. 3 the data clouds are depicted together with the normal vectors. The left part of the figure shows the case with high noise (the same data as in Figs. 1 and 2) while in the right part the case with low noise is analysed. As expected, normal vectors are estimated very well when the data lie almost along the hyper-plane. Around nonlinearities and/or in the case of higher noise the estimated normal to the surface becomes less accurate leading to the wrong linearised model.

The input partition of the two models (illustrated in Figs. 1 and 2) was used to design two fuzzy models where the consequent parameters were estimated by the classical least squares method (global optimum of the parameters is searched for in a non-recursive way). Fig. 4 shows the true output of the system with a green colour and the measured one with the black colour. Both outputs of the fuzzy models are also shown. The model that is based on Euclidean distance is shown in red, the one based on Mahalanobis distance is shown in blue. The comparison of errors shows that the proposed method results in the mean square error (MSE) of 0.0172 among the model output and the true output while the MSE of 0.0232 is achieved in the case of Euclidean-distance-based model. This means that the lower error is achieved while lower number of parameters is tuned (4 clouds instead of 7).

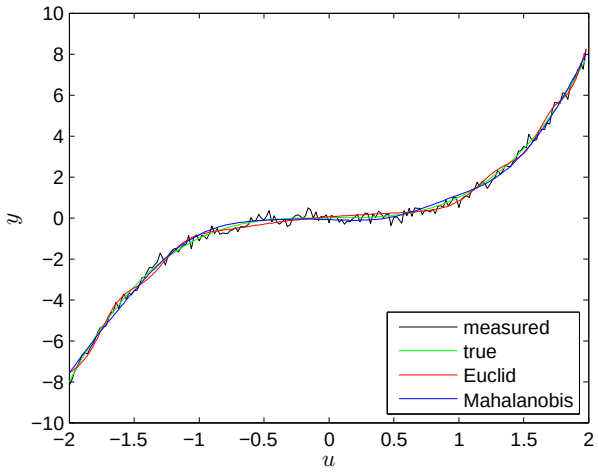

Fig. 4. The comparison of the two simulated outputs with the original one

REFERENCES

Angelov, P. and Yager, R. (2010). A simple fuzzy rule-based system through vector membership and kernel-based granulation. In *5th IEEE International Conference on Intelligent Systems (IS)*, 349–354.

Angelov, P. and Yager, R. (2011). Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density. In *2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems (EAIS)*, 62–69.

Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modelling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15, 116–132.