Fuzzy Identification from a Grey Box Modeling Point of View

P. Lindskog
Linköping University, S-581 83 Linköping

1. Introduction

The design of mathematical models of complex real-world (and typically nonlinear) systems is essential in many fields of science and engineering. The developed models can be used, e.g., to explain the behavior of the underlying system as well as for prediction and control purposes.

A common approach for building mathematical models is so-called black box modeling (Ljung, 1987; Söderström and Stoica, 1989), as opposed to more traditional physical modeling (or white box modeling), where everything is considered known a priori from physics. Strictly speaking, a black box model is designed entirely from data using no physical or verbal insight whatsoever. The structure of the model is chosen from families that are known to be very flexible and successful in past applications. This also means that the model parameters lack physical or verbal significance; they are tuned just to fit the observed data as well as possible.

The term “black box modeling” is sometimes used almost as a synonym to system identification, although a much more convenient definition, and the one often used, is that system identification is the theory of designing mathematical models of dynamical systems from observed data. Hence, by combining the black box approach with physical or verbal modeling in such a way that certain prior knowledge from the system is taken into account, we end up with special identification procedures that commonly are referred to as grey box modeling approaches, see, e.g., (Bohlin, 1991, Hangos, 1995). Two important facts make such methods intuitively appealing.

1. In a real-world modeling situation, we never have complete process knowledge. There are always uncertain factors affecting the system, thus indicating that a complete physical model hardly ever can be constructed. However, uncertain factors can be revealed through experiments and, at least partly, taken care of by employing sufficiently flexible model families.

2. The modeling procedure on the other hand allows us to restrict the flexibility to comply with the prior knowledge. This makes it possible to follow, at least partly, another basic identification principle, namely to only estimate what is still unknown.

Traditional grey box approaches assume that the structure of the model is given directly as a parameterized mathematical function, which (at least partly) is based on physical principles. However, for many real-world systems a great deal of information is provided by human experts, who do not reason in terms of mathematics but instead describe the system verbally through vague or imprecise statements. For example, in case it is hard to design a suitable mathematical model of a heating system, an important part of its behavior can still be characterized, e.g., through

\[
\text{If more energy is supplied to the heater element then the temperature will increase.} \quad (1.1)
\]

Because so much human knowledge and expertise come in terms of verbal rules, a sound engineering approach is to try to integrate such linguistic information into the identification process. A convenient and common way of doing this is to use fuzzy logic concepts in order to cast the verbal knowledge into a conventional mathematical representation (a model structure), which subsequently can be fine-tuned using input-output data. It turns out that the structure so obtained very well can be viewed as a layered network having much in common with an ordinary neural network, see, e.g., (Brown and Harris, 1994; Haykin, 1994; Roger Jang and Sun, 1995; Lin and
Lee, 1996; Chen, 1996). As a matter of fact, the kinship is so evident that many researchers refer to this approach as neuro-fuzzy modeling.

With this in mind, the palpable question is what is conceptually gained by this approach compared to standard black box neural network modeling?

- Firstly and contrary to neural networks, neuro-fuzzy modeling, or just fuzzy modeling, offers a high-level, structured and convenient way of incorporating linguistic prior into the models.

- Secondly, the basic linguistic knowledge entered is of the form “speed(t-1) is HIGH”. In fuzzy modeling, such a proposition is given a precise mathematical meaning through a basis function (membership function) having parameters associated with the property “HIGH”, thus meaning that the parameters can be assigned reasonable initial values. This is important in that the parameter estimation algorithm (which often is iterative) can be started from a point where the risk of getting stuck in an undesired local minimum is reduced compared to if the initial parameters are chosen at random (which often is the case for neural networks).

- Thirdly, physically unsound regions can be avoided. By randomly choosing initial parameter values in a neural network this cannot be guaranteed, and although regularization (see below) is applied in the estimation phase, basis functions corresponding to unsound regions are seldom removed from the final model, which then becomes more complex than necessary.

- A fourth potential advantage comes in terms of extrapolation capabilities. While data can be used to explain certain system features, the linguistic expert knowledge (here the rules) can be employed to pick up other phenomena that are not revealed in the available data.

- Finally, the human expert who supplied the verbal knowledge can always be consulted for model validation.

This contribution concentrates on how to maintain these advantages when fuzzy modeling is complemented with system identification techniques. More precisely, the aim is to provide answers to a number of central grey-box-type of questions:

1. What kind of mathematical rule base interpretation is suited when system identification aspects are also taken into account?

2. What parameter estimation algorithms should be used?

3. How can the knowledge provided by the domain expert, i.e., the meaning of the rule base, be preserved throughout the parameter estimation step?

4. How can different non-structural system features be built into the models? By non-structural knowledge we mean, e.g., that the step response is known to be monotone, or that the steady-state gain curve is monotonic in certain input variables, or some other qualitative property.

To be able to address these issues we first give a brief introduction to the field of parametric system identification, focusing mainly on basic concepts, ideas and algorithms from which the following sections can depart. Sect. 3 addresses various fuzzy modeling matters. It is argued that a Mamdani type of rule base interpretation1 (Mamdani and Assilian, 1975; Roger Jang and Sun, 1995) is suited when the rules are of the form (1.1) and when identification aspects are also accounted for. The remaining main three questions from above are then considered and answered in Sect. 4, whereupon Sect. 5 illustrates the usefulness of the suggested framework on a real-world laboratory-scale application example. Some practical aspects of the proposed modeling approach are thereafter discussed in Sect. 6 and in Sect. 7 we finally put forward some concluding remarks and give a few directions for further research within the fuzzy identification area.

1 In fact the considered model representation turns out to be structurally equivalent to a zero-order Takagi-Sugeno fuzzy structure (Takagi and Sugeno, 1985; Sugeno and Kang, 1988), which is just a special case of the general Takagi-Sugeno fuzzy model family.
2. System identification

2.1 Basic ingredients and notation

System identification deals with the problem of how to infer relationships between past input-output measurements and future outputs (Ljung, 1987; Söderström and Stoica, 1989). In practice, this is a procedure that is highly iterative in nature and is made up from three main ingredients: the data, the model structure and the selection criterion, all of which include choices that are subject to personal judgments.

The data $Z_N$. By the row vector

$$ z(t) = [y(t) \quad u_1(t) \quad \ldots \quad u_m(t)] \in \mathbb{R}^{1+m}, \quad (2.1) $$

we denote one particular data sample at time $t$ collected from a system having one output and $m$ input signals, i.e., we consider a multi input single output (MISO) system. This restriction is mainly for ease of notation and the extension to multi output (MIMO) systems is fairly straightforward, see (Ljung, 1987; Lee, 1990; Wang, 1994). Stacking $N$ consecutive samples on top of each other gives the data matrix

$$ Z_N = [z(1)^T \quad z(2)^T \quad \ldots \quad z(N)^T]^T \in \mathbb{R}^{N \times (1+m)}. \quad (2.2) $$

It is of course crucial that the data reflect the important features of the underlying system. This will typically be the case if the input signals are “sufficiently” exciting and if large “enough” data sets are collected. However, such a situation is unrealistic in many real-world applications, since, firstly, the experimental time is limited and, secondly, many of the inputs are restricted to certain signal classes. Having to live with this reality, it is worth stressing that the problem of having incomplete data very well can be alleviated considerably by building various prior system properties into the models (or rather into the applied model structure).

The model structure $g(\varphi(t), \theta)$. It is generally agreed upon that the single most difficult step in identification is that of model structure selection. Roughly speaking, the problem can be divided into three subproblems. The first one is to specify the type of model set to use. This involves the selection between linear and nonlinear representations, between black box, grey box and physically parameterized approaches, and so forth. The next issue is to decide the size of the model set. This includes the choice of possible variables (inputs and outputs) and combinations of variables to use in the models. It also involves fixing orders and degrees of the chosen model types, often to some intervals. The last item to consider is how to parameterize the model set: what basis functions should be used and how should these be parameterized, etc. With the type already determined here, we will in the sequel focus on the latter two issues.

Mathematically speaking, a quite general MISO predictor family or model structure is

$$ \hat{y}(t|\theta) = g(\varphi(t), \theta) \in \mathbb{R}, \quad (2.3) $$

where $\hat{y}(t|\theta)$ accentuates that the function $g(\cdot, \cdot)$ is a predictor, i.e., it is based on signals that are known at time $t$. The predictor structure is ensured by the regressor $\varphi(t)$, which maps output signals up to index $t-1$ and input signals up to index $t$ to an $r$ dimensional regression vector. This vector is often of the form

$$ \varphi(t) = [y(t-1) \quad \ldots \quad y(t-k) \quad u_1(t) \quad \ldots \quad u_1(t-k_1) \quad \ldots \quad u_m(t) \quad \ldots \quad u_m(t-k_m)]^T, \quad (2.4) $$

although in general its entries can be any at time $t$ known combinations of input-output signals. The mapping $g(\cdot, \cdot)$ (from $\mathbb{R}^r$ to $\mathbb{R}$) is parameterized by $\theta \in \mathcal{D} \subset \mathbb{R}^d$, with the set $\mathcal{D}$ denoting the set of values over which $\theta$ is allowed to range due to parameter restrictions. With this formulation, the work on finding a suitable model structure splits naturally into two subtasks, both possibly being nonlinear in nature.
1. the choice of dynamics, i.e., the choice of regression vector $\varphi(t)$, followed by
2. the choice of static mapping $g(\cdot, \cdot)$.

**The selection criterion** $V_N(Z_N, \theta)$. Measured and model outputs never match perfectly in practice, but differ as

$$
\varepsilon(t|\theta) = y(t) - \hat{y}(t|\theta),
$$

where $\varepsilon(t|\theta)$ is an error term reflecting unmodeled dynamics on one hand and noise on the other. An obvious modeling goal must be that this discrepancy is “small” in some sense. This is achieved by the selection criterion, which ranks different models according to some predetermined cost function. The selection criterion can come in several shapes, although we will here start off with the usual quadratic measure of the fit between measured and predicted values, i.e., with

$$
V_N(Z_N, \theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \varepsilon^2(t|\theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} (y(t) - \hat{y}(t|\theta))^2. \tag{2.5}
$$

Once these three issues are settled we have in principle defined the searched for model. It then “only” remains to estimate the parameters $\theta$ and to decide whether the model is good enough or not. If the model cannot be accepted some or even all of the entities above have to be reconsidered; in the worst conceivable case one must start from the very beginning and collect new data.

**2.2 Nonlinear model structures – The series expansion approach**

In the introduction we stated that fuzzy models have much in common with neural networks. As a matter of fact, these and many other nonlinear modeling approaches can be viewed as series expansions. Adopting the ideas of the comprehensive and unifying work of (Sjöberg et al., 1995), such a function expansion can be written

$$
\hat{y}(t|\theta) = g(\varphi(t), \theta) = \sum_{j=1}^{n} \alpha_j g_j(\varphi(t), \beta_j, \gamma_j), \tag{2.7}
$$

where, sometimes with abuse of notation, we call $g_j(\cdot, \cdot, \cdot)$ a basis function. These are usually rather simple and typically they are all of one single type. The basis functions are also local in the sense that each $g_j(\cdot, \cdot, \cdot)$ essentially covers a certain part of the total regression space. Which part is specified by the parameters $\beta_j$ and $\gamma_j$, where $\beta_j$ is related to the scale or direction of the basis function and $\gamma_j$ specify the position or translation of it. The remaining $\alpha_j$ parameter is a “coordinate” parameter – a weight – giving the basis function its final amplitude shape.

The basic difference from one series expansion approach to another is the choice of basis functions. In principle, there are three fundamentally different ways of generalizing simple univariate basis functions to multi-variate ones:

- **Ridge construction.** A ridge basis function has the form
  $$
g_j(\varphi(t), \beta_j, \gamma_j) = \kappa \left( \beta_j^T \varphi(t) - \gamma_j \right), \tag{2.9}
$$
  where $\kappa(\cdot)$ is a function in one variable having parameters $\beta_j \in \mathbb{R}^r$ and $\gamma_j \in \mathbb{R}$. Notice that the ridge nature has nothing to do with the choice of $\kappa(\cdot)$. It is attributed to that $\kappa(\cdot)$ is constant for all regression vectors in the sub-space where $\beta_j^T \varphi(t)$ is constant, thus forming a ridge along that direction; see Fig. 2.1. With $n$ weighted ridge basis functions the dimension of $\theta$ becomes $n(r + 2)$. Typical examples of this family are, e.g., feed-forward neural networks with one hidden layer (Kung, 1993; Haykin, 1994; Ljung et al., 1996) and hinging hyperplane models (Breiman, 1993; Pucar and Sjöberg, 1995a; Pucar and Sjöberg, 1995b).
Radial construction. Radial basis functions do not show the ridge directional property but have true local support as is illustrated in Fig. 2.1. Such a radial support can be obtained by using basis functions of the form

\[ g_j(\varphi(t), \beta_j, \gamma_j) = \kappa\left(\|\varphi(t) - \gamma_j\|_{\beta_j}^2\right), \]  

(2.10)

where the weighted norm \( \beta_j \) specify the scaling of the basis function. In general, \( \beta_j \) is a positive semi-definite and symmetric matrix of dimension \( r \times r \), although quite often it is chosen to be a scaled identity matrix. This means that the dimension of \( \theta \) is at least \( n(r + 2) \) and at most \( n(r^2 + r + 1) \). Popular choices within this category are kernel estimators (Watson, 1969), radial basis function networks (RBFN) (Poggio and Girosi, 1990; Chen and Billings, 1992) and wavelet networks (Zhang and Benveniste, 1992).

Composition. A composition (tensor product in (Sjöberg et al., 1995)) is obtained whenever ridge and radial constructions are combined when forming the basis functions. A typical example is shown in the rightmost plot of Fig. 2.1. The most extreme composition is

\[ g_j(\varphi(t), \beta_j, \gamma_j) = \prod_{k=1}^{r} g_{j,k}(\varphi_k(t), \beta_{j,k}, \gamma_{j,k}), \]  

(2.11)

where each \( g_{j,k}(\cdot, \cdot, \cdot) \in \mathbb{R} \) is either a ridge or a radial function. In a more general setting, such an element needs not live in \( \mathbb{R} \) but can be defined in any sub-space of \( \mathbb{R}^r \). If all \( n \) basis functions are of the commonly encountered form (2.11), then it is easy to verify that the dimension of \( \theta \) becomes \( n(2r + 1) \). Within this model class we find certain regression tree approaches (Breiman et al., 1984; Strömberg et al., 1990) and, as will be discussed in the following section, the kind of fuzzy identifiers considered in this contribution.

2.3 General parameter estimation techniques

After having determined the type of basis functions to apply, the next step is to use input-output data to estimate what is still unknown. It is here useful to distinguish the estimation needs by the kind of parameters involved in the models. The following three categories can be identified.

Structure estimation. This is the case when the type of basis functions to use have been decided, but where the size, i.e., the number of basis functions \( n \) to employ, is estimated. Selecting the \( r \) “best” regressors out of a set of possible regressors is a typical example in this category. It should be noted that structure estimation often can viewed as a combinatorial optimization problem that in complexity grows exponentially, e.g., with the number of regressors. This means that exhaustive algorithms soon become impractical, which has motivated schemes that provide, if not optimal, then at least good enough solutions. However, another way to reduce the complexity is to use prior structural system knowledge as is the case when a grey box approach is adopted.

Nonlinear-in-the-parameters estimation. Having decided the size of the model structure it remains to find reasonable parameter values \( \theta \). With the scalar loss function (2.6) as the performance
criterion the parameter estimate $\hat{\theta}_N$ is given by

$$\hat{\theta}_N = \arg \min_{\theta \in \mathbb{R}^d} V_N(Z_N, \theta),$$

(2.12)

where “argmin” is the operator that returns the argument that minimizes the loss function. This is a very important and well-known problem formulation leading to prediction error minimization (PEM) methods. The type of PEM algorithm to apply depends on whether the parameters $\theta$ enter the model structure in a linear or a nonlinear way. The latter situation leads to a nonlinear least-squares problem, and appears whenever the model structure (2.7) contains unknown direction $\beta$ or translation $\gamma$ parameters.

**Linear-in-the-parameters estimation.** In case all parameters enter the structure in a linear fashion one usually talks about a linear least-squares problem. For the series expansion (2.7) such an approach is applicable if only coordinate parameters $\alpha_j$ are to be estimated.

It should be emphasized that the complexity of the estimation problem decreases in the listed order, yet at the price of that the amount of prior needed to arrive at a useful model typically increases. With these preliminary observations, we next present some different minimization algorithms, unconstrained as well as constrained ones.

### 2.3.1 Unconstrained linear least-squares algorithms

The parameters of an unconstrained linear least-squares structure (a linear regression) can be estimated efficiently and analytically by solving the normal equations

$$\varphi(t)\varphi(t)^T \hat{\theta}_N = \varphi(t)y(t)$$

(2.13)

for $t = 1, \ldots, N$. The optimal parameter estimate is

$$\hat{\theta}_N = \left[ \sum_{t=1}^{N} \varphi(t)\varphi(t)^T \right]^{-1} \sum_{t=1}^{N} \varphi(t)y(t) = R_N^{-1} f_N,$$

(2.14)

provided that the inverse of the $d \times d$ regression matrix $R_N$ exists. For numerical reasons this inverse is rarely formed, but instead the estimate is computed via so-called QR- or singular value decomposition (SVD) (Golub and Van Loan, 1989; Björk, 1996), which both are able to handle rank deficient regression matrices.

### 2.3.2 Unconstrained nonlinear least-squares algorithms

When the parameters appear in a nonlinear fashion the typical situation is that the minimum of the loss function cannot be computed analytically. Instead we have to resort to certain iterative search routines, most of which can be seen as special cases of Newton’s algorithm (see among many others (Dennis and Schnabel, 1983; Scales, 1985; Fletcher, 1987))

$$\hat{\theta}_N^{(i+1)} = \hat{\theta}_N^{(i)} - \left[ V_N^t(Z_N, \hat{\theta}_N^{(i)}) \right]^{-1} V_N^t(Z_N, \hat{\theta}_N^{(i)}) = \hat{\theta}_N^{(i)} - \Delta_N^{(i)}(Z_N, \hat{\theta}_N^{(i)}),$$

(2.15)

where $\hat{\theta}_N^{(i)} \in \mathbb{R}^d$ is the parameter estimate at the $i$-th iteration, $V_N^t(\cdot, \cdot) \in \mathbb{R}^d$ is the gradient of the loss function and $V_N^{tt}(\cdot, \cdot) \in \mathbb{R}^{d \times d}$ the Hessian of it, both computed with respect to the current parameter vector. More specifically, the gradient is given by

$$V_N^t(Z_N, \hat{\theta}_N^{(i)}) = -\frac{1}{N} \sum_{t=1}^{N} J(t|\hat{\theta}_N^{(i)}) \varepsilon(t|\hat{\theta}_N^{(i)}),$$

(2.16)

with $J(t|\hat{\theta}_N^{(i)}) \in \mathbb{R}^d$ being the **Jacobian** vector

$$J(t|\hat{\theta}_N^{(i)}) = \begin{bmatrix} \frac{\partial \hat{y}(t|\hat{\theta}_N^{(i)})}{\partial \theta_1^{(i)}} & \cdots & \frac{\partial \hat{y}(t|\hat{\theta}_N^{(i)})}{\partial \theta_d^{(i)}} \end{bmatrix}^T.$$

(2.17)

Differentiating the gradient with respect to the parameters yields the **Hessian**
\[ V''_N(Z_N, \hat{\Theta}_N) = \frac{1}{N} \sum_{i=1}^{N} \left( J(t|\hat{\Theta}_N) J(t|\hat{\Theta}_N)^T - \frac{\partial J(t|\hat{\Theta}_N)}{\partial \hat{\Theta}_N} \varepsilon(t|\hat{\Theta}_N) \right), \]  

(2.18)

which thus means that the second derivative of the loss function is needed in (2.15). Simply put, Newton’s algorithm searches for the new parameter vector along a Hessian modified gradient of the current loss function.

The availability of derivatives of the loss function with respect to the parameters is of course of paramount importance in all Newton-based estimation schemes. In case arbitrary (though differentiable) predictor structures are considered these may very well be too hard to obtain analytically or too expensive to compute. One way around this difficulty is to numerically approximate the derivatives by finite differences. The simplest such a method is just to replace each of the \(d\) elements of the Jacobian by the forward difference

\[ J_j(t|\hat{\Theta}_N) = \frac{\partial \hat{y}_j(t|\hat{\Theta}_N)}{\partial \hat{\Theta}_j} \approx \frac{\hat{y}_j(t|\hat{\Theta}_N + h_j e_j) - \hat{y}_j(t|\hat{\Theta}_N - h_j e_j)}{2h_j}, \]

(2.19)

with \(e_j\) being a column vector with a one in the \(j\)-th position and zeros elsewhere, and with \(h_j\) being a small positive scalar perturbation. Because the parameters may differ substantially in magnitude it is here expedient to individually choose these perturbations. A typical choice is \(h_j = \sqrt{e} \max(h_{\text{min}}, |\hat{\Theta}_j|)\), where \(e\) is the relative machine precision and \(h_{\text{min}} > 0\) is the smallest perturbation allowed; consult (Dennis and Schnabel, 1983; Scales, 1985) for further details on this. If a more accurate approximation is deemed necessary one can employ the central difference

\[ J_j(t|\hat{\Theta}_N) = \frac{\partial \hat{y}_j(t|\hat{\Theta}_N)}{\partial \hat{\Theta}_j} \approx \frac{\hat{y}_j(t|\hat{\Theta}_N + h_j e_j) - \hat{y}_j(t|\hat{\Theta}_N - h_j e_j)}{2h_j}, \]

(2.20)

at the cost of \(d\) additional function evaluations.

It now turns out that the Newton update (2.15) has some severe drawbacks, most of which are associated with the computation of the Hessian (2.18). Firstly, it is in general expensive to compute the derivative of the Jacobian. It may also happen that the inverse of the Hessian does not exist, so if further progress towards a minimum is to be made the update vector must be constructed in a different way. Furthermore, even if the inverse exists it is not guaranteed to be positive definite and it may therefore happen that the parameter update vector is such that the loss function actually becomes larger. Finally, although the parameter update vector is a descent one it might be much too large, locating the new parameters at a point with higher loss than what is currently the case. To avoid these problems other search directions than (2.15) are much more common in practice:

**Gradient method.** Simply replace the Hessian by an identity matrix of appropriate size. This, however, does not prevent the update vector from being so large that also \(V_N(\cdot)\) becomes larger. To avoid such a behavior the updating is often complemented with a line search technique

\[ \Delta_N^{(i)}(Z_N, \hat{\Theta}_N) = \mu^{(i)} V_N(Z_N, \hat{\Theta}_N), \]

(2.21)

where \(0 < \mu^{(i)} \leq 1\), thereby giving a **damped gradient algorithm**. The choice of step length \(\mu^{(i)}\) is not critical, and the procedure often used is to start with \(\mu^{(i)} = 1\) and then repeatedly halve it until a lower value of the loss function is obtained.

**Gauss-Newton method.** By neglecting the second derivative term of the Hessian (2.18) and including line search as above we arrive at a **damped Gauss-Newton algorithm** with update vector

\[ \Delta_N^{(i)}(Z_N, \hat{\Theta}_N) = \mu^{(i)} \left[ \sum_{t=1}^{N} J(t|\hat{\Theta}_N) J(t|\hat{\Theta}_N)^T \right]^{-1} \sum_{t=1}^{N} J(t|\hat{\Theta}_N) \varepsilon(t|\hat{\Theta}_N), \]

(2.22)
which is of the same form as the linear least-squares formula (2.14). To cope with a singular or near to singular Hessian approximation the inverse is normally replaced by the so-called pseudo-inverse, which easily can be obtained by computing the SVD of the Jacobian (Golub and Van Loan, 1989).

**Levenberg-Marquardt method.** The Levenberg-Marquardt algorithm handles simultaneously the update step size and the singularity problems through the update

\[
\Delta^{(i)}_{N}(Z_{N}, \theta^{(i)}_{N}) = \left[ \sum_{l=1}^{N} J(t|\theta^{(i)}_{N}) J(t|\theta^{(i)}_{N})^T \right]^{-1} \sum_{l=1}^{N} J(t|\theta^{(i)}_{N}) \varepsilon(t|\theta^{(i)}_{N}),
\]

(2.23)

where the Hessian is guaranteed to be positive definite since \( \mu^{(i)} > 0 \). As is the case for the above procedures, it can be shown that this update is in a descent direction. However, \( \mu^{(i)} \) must be carefully chosen so that the loss function also decreases. The method by Marquardt (Scales, 1985) achieves this by starting with a \( \mu^{(i)} > 0 \), whereupon it is reduced (typically a factor 10) at the beginning of each iteration, thereby aiming at mimicking a Gauss-Newton update step. If this results in an increased loss, then the step \( \mu^{(i)} \) is repeatedly increased (typically a factor 10) until \( V_{N}(Z_{N}, \theta^{(i+1)}_{N}) < V_{N}(Z_{N}, \theta^{(i)}_{N}) \), which means that the update is forced towards a scaled gradient direction. Other and more elaborate choices of \( \mu^{(i)} \) are discussed in, e.g., (Fletcher, 1987).

Although simple, a major drawback with the gradient method is that the convergence rate can be fairly poor close to the minimum. This fact favors the latter two methods, which, especially near the minimum, show similar convergence properties as the full Newton algorithm (Dennis and Schnabel, 1983). For ill-conditioned problems (Dennis and Schnabel, 1983) recommend the Levenberg-Marquardt modification. However, this choice is far less obvious when the pseudo-inverse is used in the Gauss-Newton update. In such a case both methods try to update the parameters that really influence the criterion fit most, whereas the remaining parameters are kept unchanged. This means that so-called *regularization* is built into the algorithms (see below).

A last algorithmic issue to consider here is when to terminate the search. In theory, \( V_{N}(\cdot, \cdot) \) is zero at a minimum, so an obvious practical test is to terminate once \( |V_{N}(\cdot, \cdot)| \) is sufficiently small. Another useful test is to investigate the relative change in parameters from one iteration to another and terminate if this quantity falls below some tolerance level. The algorithms will also terminate when a certain number of maximum iterations has been carried out, or if the line search algorithm fails to decrease the loss function in a predetermined number of iterations.

It is worth stressing that the three schemes above all return estimates that are at least as good as the starting point. Nonetheless, should the algorithm converge to a minimum, then it is important to remember that convergence needs not be to a global minimum but can be to a local one.

**2.3.3 Constrained minimization algorithms.** In a grey box modeling situation the parameters usually have physical or linguistic significance. To really maintain such a property it is necessary to take the corresponding parameter restrictions into account in the estimation procedure, i.e., *constrained optimization* methods are needed.

Therefore assume that there are \( l \) parameter constraints collected in a vector

\[
c(\theta) = [c_1(\theta), c_2(\theta), \ldots, c_l(\theta)]^T \in \mathbb{R}^l,
\]

where each \( c_j(\theta) \) is a well-defined function such that \( c_j(\theta) > 0 \) for \( j = 1, \ldots, l \), hence specifying a feasible parameter region \( D \). There exist quite a few schemes that handles such constraints, see, e.g., (Scales, 1985). An old but simple and versatile idea is to rephrase the original problem into a sequence of unconstrained minimization problems for which a Newton type of method (like the gradient one) can be applied without too much extra coding effort.

This is the basic idea behind the *barrier function* estimation procedure. Algorithmically, the method starts with a feasible parameter vector \( \bar{\theta}^{(0)}_{N} \), whereupon the parameter estimate is itera-
tively obtained by solving (each iteration is started with the estimate of the previous iteration)
\[
\hat{\theta}^{(k+1)}_N = \arg\min_{\theta \in \mathcal{D}} W_N(Z_N, \theta) = \arg\min_{\theta \in \mathcal{D}} \left( V_N(Z_N, \theta) + \rho^{(k)} \sum_{j=1}^l \vartheta(c_j(\theta)) \right),
\]
where, typically, \( \rho^{(k)} = 10^{-k} \) with \( k \) starting from 0 and then increasing by 1 for each iteration until convergence is obtained. In order to maintain a feasible estimate the barrier function \( \vartheta(\cdot) \) is chosen so that an increasingly larger value is added to the objective function \( W_N(\cdot, \cdot) \) as the boundary of the feasibility region \( \mathcal{D} \) is approached from the interior; at the boundary itself this quantity should be infinite. A good choice of barrier function for many kinds of problems seems to be the log barrier function \( \vartheta(c_j(\theta)) = - \ln(c_j(\theta)) \); see (Scales, 1985) for further details on this.

At this stage one may wonder why it is not sufficient to set \( \rho^{(k)} \) to a much smaller value in the beginning. One reason is that if the true minimum is near the boundary, then it could be difficult to minimize the overall cost function because of its rapidly changing curvature near this minimum, thus giving rise to an ill-conditioned problem. One could also argue that the method is too complex as an outer iteration is added. This is only partially true as the inner estimate (essentially at the first few outer iterations) needs not be that accurate. A rule of thumb is to only perform some five iterations in the inner loop. Finally, the outer loop is terminated once the parameter update is sufficiently small or when a number of maximum outer iterations has been carried out.

### 2.4 The bias-variance trade-off

The series expansion approach has been widely used in nonlinear black box identification, where the idea is to employ a parameterization that covers an as broad system class as possible. In practice, however, the typical situation is that merely a fraction of the available flexibility is really needed, i.e., the applied model structures are often over-parameterized. This fact, possibly in combination with an insufficiently informative data set \( Z_N \), leads to ill-conditioning of the Jacobian and the Hessian. This observation also suggests that the parameters should be divided into two sets: the set of spurious parameters, which do not influence the criterion fit that much, and the set of efficient parameters, which do affect the fit. Having such a decomposition it is intuitively reasonable to treat the spurious or redundant parameters as constants that are not estimated. The problem with this is now that it is in general hard to make this decomposition beforehand.

However, using data one can overcome the ill-conditioning problem and automatically unveil an efficient parameterization by incorporating regularization techniques (or trust region techniques) (Dennis and Schnabel, 1983). When such an effect is built into the estimation procedure, as in the Levenberg-Marquardt algorithm, we get so-called implicit regularization, as opposed to explicit regularization, which is obtained by adding a penalty term to the criterion function, e.g., as
\[
W_N(Z_N, \theta) = \left( \frac{1}{N} \sum_{i=1}^N \frac{1}{2} (\xi_i(\theta))^2 \right) + \frac{\mu}{2} |\theta - \theta^d|^2 = V_N(Z_N, \theta) + \frac{\mu}{2} |\theta - \theta^d|^2,
\]
where \( \mu > 0 \) is a small user-tunable parameter ensuring a positive definite Hessian and \( \theta^d \in \mathbb{R}^d \) is some a priori determined parameter vector (possibly representing prior parameter knowledge). Here the important point is that a parameter not affecting the first term that much will be kept close to \( \theta^d \) by the second term. This means that the regularization parameter \( \mu \) can be viewed as a threshold that labels the parameters to be either efficient or spurious (Sjöberg et al., 1995). A large \( \mu \) simply means that the number of efficient parameters \( d^e \) becomes small.

From a system identification viewpoint regularization is a very important means for addressing the ever present bias-variance trade-off, as is emphasized in (Ljung, 1987; Ljung et al., 1996). There it is shown, under fairly general assumptions, that the asymptotic criterion misfit essentially depends on two factors that can be affected by the choice of model structure. First we have the
bias error, which reflects the misfit between the true system and the best possible approximation of it, given a certain model structure. Typically, this error decreases when the number of parameters \( d \) increases. The other term is the parameter variance error, which usually grows with \( d \) but decreases with \( N \). There is thus a clear trade-off between the bias and the variance contributions.

At this point, suppose that a flexible enough model structure has been decided upon. Decreasing the number of parameters that are actually updated (\( d \)) by increasing \( \mu \) is beneficial for the total misfit as long as the decrease in variance error is larger than the increase in bias error. In other words, the purpose of regularization is to decrease the variance error contribution to a level where it balances the bias misfit.

3. Fuzzy modeling framework

The history of methods based on fuzzy concepts is rather short. It all started in the mid 1960s with Zadeh’s pioneering article (Zadeh, 1965), in which a new way of characterizing non-probabilistic uncertainties via so-called fuzzy sets was suggested. Since then, and especially in the last ten or so years, there has been a dramatic growth of sub-disciplines in science and engineering that have adopted fuzzy ideas. To a great extent this development is due to a large number of successful industrial applications, spanning such diverse fields as robotics, consumer electronics, signal processing, bioengineering, image processing, pattern recognition, management and control. See the comprehensive compilations (Marks II, 1994, Chen, 1996).

The fields of fuzzy control and fuzzy identification have been developed largely in parallel. A good first book on fuzzy control is (Driankov et al., 1993), and a shorter but informative overview is given by (Lee, 1990). Various fuzzy identification methods have been proposed by several authors. The work by Sugeno and coworkers (Takagi and Sugeno, 1985, Sugeno and Kang, 1988, Sugeno and Yasukawa, 1993) and by (Wang, 1995) constitute some of the most influential contributions. The merging of fuzzy control and fuzzy identification is discussed, e.g., in (Wang, 1994) and in (Roger Jang and Sun, 1995). Many of the ideas detailed in this section can be found in the latter reference, which is exceptionally well written and highly recommended. With these sources as a basis, the aim of the section is to derive and motivate the use of one particular fuzzy rule base interpretation that is suited for identification purposes.

3.1 Components of a fuzzy model

The basic configuration of a fuzzy model is shown in Fig. 3.1. The model involves six components, of which the four lowermost ones are fuzzy model specific.

Scaling. The physical values of the actual inputs and outputs may differ significantly in magnitude. By mapping these to proper normalized domains via scaling, one can instead work with signals that roughly are of the same magnitude, which is desirable from an estimation point of view. However, the need for scaling is highly problem dependent and therefore not considered any further in here, i.e., from now on we assume that \( \varphi(t) \) is formed directly from \( z(t) \) and that \( \hat{y}_s(t|\theta) = \hat{y}(t|\theta) \).

Regressor generator. The kind of dynamics to include in a fuzzy model is engendered in the regressor generator. The regression vector \( \varphi(t) \) can contain any at time \( t \) known combinations of input-output measurements \( z(t) \), although for such a combination to make sense it ought to have a linguistic interpretation. This is ascribed to that the entries of \( \varphi(t) \) are specified by the linguistic database or, actually, by so-called linguistic variables (see below). Such a typical variable is SPEED\((t - 1) \), which in terms of input-output data may be interpreted as \( z_1(t - 1) \). This also means that the mathematical purpose of the remaining components of a fuzzy model is to provide a static map from \( \varphi(t) \in \mathbb{R}^r \) to \( \hat{y}(t|\theta) \in \mathbb{R} \).

Linguistic database. The linguistic database is the heart of a fuzzy model. The expert knowledge,
which is assumed to be given as a number of if-then rules, is stored in a fuzzy rule base. These rules are subsequently given a precise mathematical meaning through user-supplied definitions of the employed linguistic variables and connectives (and, or, etc.).

**Fuzzifier.** The fuzzifier maps the crisp values of \( \varphi(t) \) into suitable fuzzy sets (discussed below).

**Fuzzy inference engine.** The fuzzy sets provided by the fuzzifier are then interpreted by the fuzzy inference engine, which uses the fuzzy rule base knowledge in order to produce some fuzzy sets in the output \( y \).

**Defuzzifier.** As a last step the defuzzifier converts the output fuzzy sets to a standard crisp signal \( \hat{y}(t|\theta) \in \mathbb{R} \).

From this short description it should be clear that fuzzy sets are vital objects to comprehend in order to understand how a fuzzy model operates. Let us therefore discuss such sets in more detail.

### 3.2 Fuzzy sets and membership functions

An ordinary set is a set with a crisp boundary, i.e., an element can either be or not be a member of that set. A fuzzy set on the other hand does not show this absolute “either-or” membership property. The transition from “belonging to” to “not belonging to” a fuzzy set is instead gradual, where the degree of belonging is characterized by a membership function. Mathematically speaking, the definition is as follows (Driankov et al., 1993).

**Definition 3.1.** If \( u \) is an element in the universe of discourse \( U \), then a fuzzy set \( A \) in \( U \) is the set of ordered pairs

\[
A = \{(u, \mu_A(u)) : u \in U\},
\]

where \( \mu_A(u) \) is a membership function carrying an element from \( U \) into a membership value between 0 (no degree of membership) and 1 (full degree of membership).

**Example 3.1.** Suppose that we want to describe a car traveling at high speed on the motorway. As a first step, let \( u \) denote the speed of any car and introduce \( U = [0,300] \subset \mathbb{R} \), which states
that no car can go faster than 300 km/h. By Nordic standards, a car running at, say, 140 km/h is considered to have a high speed, while this is not the case when the speed is, say, 80 km/h. Moreover, in case the car is running at around 110 km/h most people would say that the speed is neither low nor high. Based on this information, a fuzzy set describing that the speed of a car is high is, e.g.,

$$\text{HIGH} = \left\{ (u, \mu_{\text{HIGH}}(u) = \frac{1}{1 + e^{-0.1(u-110)}}) : u \in [0, 300] \right\}. \quad (3.2)$$

This subjective choice of membership function gives that cars running at 80, 110 and 140 km/h are considered to go at a high speed to a degree of 0.05, 0.90 and 0.95, respectively.

An important point illustrated in this example is that the fuzziness does not emanate from the fuzzy set itself but rather from the vagueness of what it describes. This is manifested by the subjective and non-random nature of the choice of membership function, which may vary considerably depending on who determined it. This is also the main philosophical difference between fuzzy memberships and probabilities (which convey objective information about random phenomena).

As noted above, the membership function (MF) can be any function producing a value between 0 and 1. Here we will focus on three common classes of MFs, all being convex in nature, i.e., the membership functions are of the form “increasing”, “decreasing”, or “bell-shaped”; see (Driankov et al., 1993) for the mathematical definition. First we have what may be called the network-classic MFs, which because of their smoothness are becoming increasingly popular in fuzzy modeling.

**Definition 3.2 (Network-classic MFs).** This class consists of the sigmoidal and the Gaussian membership functions defined as

$$\text{mfsig}(u, \beta, \gamma) : \mu_A(u, \beta, \gamma) = \frac{1}{1 + e^{-\beta(u-\gamma)}},$$

$$\text{mfgauss}(u, \beta, \gamma) : \mu_A(u, \beta, \gamma) = e^{-\frac{1}{2}(\frac{u-\gamma}{\beta})^2},$$

where $\beta$ and $\gamma$ are related to the scale and the position of the membership function, respectively.

The second class, widely used in fuzzy logic theory, was originally suggested by Zadeh, thus meriting the label Zadeh-formed MFs.

**Definition 3.3 (Zadeh-formed MFs).** The Zadeh-formed MFs are the Z-, the S- and the $\pi$-functions (named after their shape) in order defined as ($\gamma_1 \leq \gamma_2 \leq \gamma_3 \leq \gamma_4$)

$$\text{mfz}(u, \gamma_1, \gamma_2) : \mu_A(u, \gamma) = \begin{cases} 1 & u \leq \gamma_1 \\ 1 - 2 \left( \frac{u-\gamma_1}{\gamma_2-\gamma_1} \right)^2 & \gamma_1 < u \leq \frac{\gamma_2 + \gamma_3}{2} \\ 2 \left( \frac{u-\gamma_2}{\gamma_3-\gamma_2} \right)^2 & \frac{\gamma_2 + \gamma_3}{2} < u \leq \gamma_2 \\ 0 & u > \gamma_2 \end{cases},$$

$$\text{mfs}(u, \gamma_1, \gamma_2) : \mu_A(u, \gamma) = 1 - \text{mfz}(u, \gamma_1, \gamma_2),$$

$$\text{mfpi}(u, \gamma_1, \gamma_2, \gamma_3, \gamma_4) : \mu_A(u, \gamma) = \begin{cases} \text{mfs}(u, \gamma_1, \gamma_2) & u \leq \gamma_2 \\ \text{mfz}(u, \gamma_3, \gamma_4) & u > \gamma_3 \end{cases},$$

The last category is piecewise linear MFs, which, primarily because of real-time aspects, have been extensively used in various fuzzy control applications (Driankov et al., 1993).

**Definition 3.4 (Piecewise linear MFs).** The piecewise linear MFs are the open left, the open right, the triangular and the trapezoidal functions ($\gamma_1 \leq \gamma_2 \leq \gamma_3 \leq \gamma_4$):
Notice that with the terminology from the previous section an MF is really nothing but a basis product. Similarly, a fuzzy disjunction (or) is usually defined as a details. The most widely used triangular norms are intersection (the min operator) and algebraic set 

\[ u \land v = \min(u, v) \]

and since it involves one variable \((u)\) only it is of composition type. As will be evident in the following section, fuzzy sets constitute the main building block of a linguistic variable.

### 3.3 Linguistic variables and fuzzy propositions

**Linguistic variables** are fundamental in approximate or fuzzy reasoning. In a generalized form, cf. (Driankov et al., 1993), such a variable is conveniently described by a three-tuple

\[ \langle U, A(U, \theta), D \rangle, \]

where \(U\) is the name of the variable, \(A(\cdot, \cdot)\) is a set of linguistic values, each of which is characterized by a fuzzy set, that can be assigned to \(U\), and \(D\) provides information on how to connect the linguistic domain to the physical measurement domain.

**Example 3.2.** The linguistic variable \(\text{speed}(t - 1)\) of a car on a motorway, is, e.g.,

\[ \langle \text{speed}(t - 1), A(U, \theta) = \{\text{LOW, MEDIUM, HIGH}\}, D : \varphi_1(t) = z_1(t - 1) \rangle \]

where \(\text{LOW} = \{(u, \mu_{\text{LOW}}(u, \gamma)) = \text{mf1}(u, 60, 90) : u \in U\}\)

\(\text{MEDIUM} = \{(u, \mu_{\text{MEDIUM}}(u, \gamma)) = \text{mftrap}(u, 60, 90, 110, 140) : u \in U\}\)

\(\text{HIGH} = \{(u, \mu_{\text{HIGH}}(u, \gamma)) = \text{mfr}(u, 110, 140) : u \in U\}\),

where \(U = [0, 300]\).

The assignment of a linguistic variable to a linguistic value is simply achieved by an atomic fuzzy proposition using the syntax “\(U\) is \(\text{PROP}\)”, e.g., “\(\text{speed}(t - 1)\) is \(\text{LOW}\)”.

Several atomic fuzzy propositions can now be combined using linguistic connectives such as ‘not’, ‘and’ and ‘or’, thus forming more complex propositions as, e.g.,

\[ (U_1 \text{ is not } A_1) \text{ and } (U_2 \text{ is } A_2), \]

where \(A_1\) and \(A_2\) refer to two different fuzzy sets which normally are defined in different universes \(U_1\) and \(U_2\), respectively. While it is mathematically natural to interpret \((U_1 \text{ is not } A_1)\) as the fuzzy set \(1 - \mu_{A_1}(u_1, \cdot)\) with \(\mu_{A_1}(u_1, \cdot)\) being the MF associated with \(A_1\), there are many different ways of interpreting ‘and’ and ‘or’. Often, however, a fuzzy conjunction (and) is defined in terms of a triangular norm \(\ast\), which combines MFs as \(\mu_{A_1}(u_1, \cdot) \ast \mu_{A_2}(u_2, \cdot)\); see (Driankov et al., 1993) for the details. The most widely used triangular norms are intersection (the min operator) and algebraic product (multiplication). Similarly, a fuzzy disjunction (or) is usually defined as a triangular co-norm \(+\), syntactically written \(\mu_{A_1}(u_1, \cdot) + \mu_{A_2}(u_2, \cdot)\). The most commonly encountered co-norms are union (the max operator) and algebraic sum \(\mu_{A_1}(u_1, \cdot) + \mu_{A_2}(u_2, \cdot) - \mu_{A_1}(u_1, \cdot) \cdot \mu_{A_2}(u_2, \cdot)\).

If, in the above operations, \(u_1\) and \(u_2\) are defined in different universes, then a triangular norm or co-norm performs a mapping from \([0, 1] \times [0, 1]\) to \([0, 1]\). Otherwise, the mapping is from \([0, 1]\) to \([0, 1]\). By combining several atomic fuzzy expressions using suitable connectives (others than those above can of course also be defined) it is possible to construct arbitrarily complex fuzzy sets. In doing so the important point is that the result always is a new fuzzy set, although the space in which it is defined is not restricted to one or two dimensions.
3.4 Fuzzy model structure

We are now in a position to discuss the computational units of Fig. 3.1. As with the interpretation of the basic connectives, there is also here a number of choices to be made. However, various grey box identification aspects, e.g., the complexity of computing predictors, Jacobians and possibly Hessians, the approximation capability as well as the interpretability of the estimated models, naturally lead to one particular type of fuzzy model structure, as will be derived next.

Fuzzification. The fuzzification unit is conceptually quite simple. For each value of $\varphi_k(t), k = 1, 2, \ldots, r$, it returns a fuzzy set (a fuzzy fact) denoted $A_k^j$ with MF $\mu_{A_k^j}(u_k) \in \mathbb{U}_k$. Thus, if we are given $N$ measurements and use $r$ regressors, then this implies that $N \cdot r$ fuzzy sets will be generated by the fuzzifier. These sets can in principle be constructed either by a singleton fuzzifier or by a non-singleton ditto. The latter approach may be used to capture noise properties of the regressors (Wang, 1994), though at a rather high computational cost. This fact motivates the use of a singleton fuzzifier, which simply returns a 1 in one single point in $\mathbb{U}_k$:

$$\mu_{A_k^j}(u_k) = \begin{cases} 1 & \text{if } u_k = \varphi_k(t) \\ 0 & \text{otherwise.} \end{cases} \quad (3.5)$$

Fuzzy rule base. A fuzzy rule base $R$ consists of a set of, say, $n$ fuzzy rules. Using a somewhat unorthodox grid-oriented multi-indexing labeling system, such a rule base often takes on the form

$$
\begin{align*}
R_{1,\ldots,1} & : \text{If } (U_1 \text{ is } A_{1,1}) \text{ and } \ldots \text{ and } (U_r \text{ is } A_{1,r}) \text{ then } (Y \text{ is } B_{1,\ldots,1}) \\
R_{1,\ldots,2} & : \text{If } (U_1 \text{ is } A_{1,1}) \text{ and } \ldots \text{ and } (U_r \text{ is } A_{2,r}) \text{ then } (Y \text{ is } B_{1,\ldots,2}) \\
& \vdots \\
R_{n,\ldots,1} & : \text{If } (U_1 \text{ is } A_{n,1}) \text{ and } \ldots \text{ and } (U_r \text{ is } A_{n,r}) \text{ then } (Y \text{ is } B_{n,\ldots,1}) \\
R_{n,\ldots,2} & : \text{If } (U_1 \text{ is } A_{n,1}) \text{ and } \ldots \text{ and } (U_r \text{ is } A_{2,r}) \text{ then } (Y \text{ is } B_{n,\ldots,2}) \\
& \vdots \\
R_{n,\ldots,n_r} & : \text{If } (U_1 \text{ is } A_{n,1}) \text{ and } \ldots \text{ and } (U_r \text{ is } A_{n,r}) \text{ then } (Y \text{ is } B_{n,\ldots,n_r}),
\end{align*}
$$

(3.6)

where $A_{1,1}, \ldots, A_{n,r}$ are the linguistic values that can be assigned to the linguistic variables $U_1, \ldots, U_r$, while $B_{1,\ldots,1}, \ldots, B_{n,\ldots,n}$ denote the linguistic values that can be allotted to the linguistic output $Y$. As usual, the mathematical meaning of any $A_{j_k,k}$ and $B_{j_1,\ldots,j_k}$ are given by suitable membership functions, denoted $\mu_{A_{j_k,k}}(u_k, \cdot)$ and $\mu_{B_{j_1,\ldots,j_k}}(y, \cdot)$, respectively.

With the rule base (3.6) it is worthwhile stressing that each $B_{j_1,\ldots,j_k, j}$ needs not be linguistically unique. On the contrary, the typical situation is that many rules share the same linguistic consequence, yet they have different antecedents. Furthermore, each $U_j$ can be assigned to $n_j$ different linguistic values, which means that the number of rules in a complete fuzzy rule base becomes $n = \prod_{k=1}^{r} n_k$. In case the rule base is incomplete there will be regions (e.g., physically impossible ones) in the overall regression space for which no output can be inferred. Notice also that a rule base with ‘and’ connectives only (3.6) is not that restricted as one might first suspect. The reason is that many other constructs (e.g., ‘not’ and ‘or’) can be logically converted (e.g., by De Morgan’s Law) to the desired form as is shown in (Wang, 1994; Lindskog, 1996).

Interpreting fuzzy if-then rules. Each of the rules of (3.6) describes essentially a relation between $U_1, \ldots, U_r$ on one side and $Y$ on the other side. This suggests that a rule $R_{j_1,\ldots,j}$, should be defined as a fuzzy relation with MF $\mu_{A_{j_k,k}} \times \ldots \times \mu_{A_{j_r,r}} \rightarrow \mu_{B_{j_1,\ldots,j,r}}(u_1, \ldots, u_r, y)$ defined on $U_1 \times \ldots \times U_r \times \mathbb{Y}$. There now exist some 40 different ways of interpreting implication
(Lee, 1990), but striving for computational simplicity we also here choose the simplest and most widely used translation, namely

\[ \mu_{A_j \rightarrow B_j}(u, y, \cdot) = \mu_{A_j}(u, \cdot) \ast \mu_{B_j}(y, \cdot) = \mu_{A_{j,1}}(u_1, \cdot) \ast \ldots \ast \mu_{A_{j,r}}(u_r, \cdot) \ast \mu_{B_j}(y, \cdot), \tag{3.7} \]

which is sometimes referred to as *Mamdani implication*.

**Inference engine.** Faced with some facts from the fuzzifier and a fuzzy rule base \((3.6)\), the fuzzy inference engine is responsible for inferring conclusions in terms of output fuzzy sets. The most generalization of the classical modus ponens rule to the fuzzy domain; see, e.g., (Driankov et al., 1993).

More specifically, the GMP inference scheme takes some fuzzy facts as input, maps it via a fuzzy relation – the rule representation – and returns a fuzzy conclusion:

\[
\begin{align*}
(U_1 \text{ is } A^1_1) \quad \text{and} \quad \ldots \quad \text{and} \quad (U_r \text{ is } A^r_1) \\
\text{If} \ (U_1 \text{ is } A_{j_1,1}) \quad \text{and} \quad \ldots \quad \text{and} \quad (U_r \text{ is } A_{j_r,r}), \quad \text{then} \quad (Y \text{ is } B_{j_1,1}) \quad \ldots \quad \text{and} \quad (Y \text{ is } B_{j_r,r})
\end{align*}
\]

Mathematically speaking (Dubois and Prade, 1992), this scheme first combines the overall MFs of the given fact and the rule. Projecting the so generated MF onto the \(y\)-axis gives a new MF \(\mu_{B_j_1, \ldots, j_r}(y, \cdot) \in \mathbb{Y}\), which can be thought of as the possibility distribution of the fuzzy output given the fuzzy fact. Representing the fuzzy fact by a MF \(\mu_{A_i}(u) = \mu_{A^1_i}(u_1) \ast \ldots \ast \mu_{A^r_i}(u_r)\) defined on \(U_1 \times \ldots \times U_r\), we thus get

\[ \mu_{B_j_1, \ldots, j_r}(y, \cdot) = \text{proj}_{u \in U^r} \left( \mu_{A_i}(u) \text{ AND } \mu_{A_j \rightarrow B_j}(u, y, \cdot) \right) = \sup_{u \in U^r} \left( \mu_{A_i}(u) \ast \mu_{A_j \rightarrow B_j}(u, y, \cdot) \right), \tag{3.8} \]

which is known as *sup-star composition*.

Assuming now that the facts are represented by fuzzy singletons, \(\mu_{A_i}(u)\) will be 1 in one single point in \(U^r\), namely when \(u = \varphi(t)\), and zero elsewhere. Because a triangular norm \(\ast\) returns a 0 membership degree when one of its operands is 0, (3.8) simplifies to

\[ \mu_{B_j}(y, \cdot) = \mu_{B_j_1, \ldots, j_r}(y, \cdot) = \mu_{A_j \rightarrow B_j}(\varphi(t), y, \cdot), \tag{3.9} \]

which means that \(\mu_{B_j}(y, \cdot)\) is obtained by slicing \(\mu_{A_j \rightarrow B_j}(u, y, \cdot)\) along the \(u\)-coordinate specified by \(\varphi(t)\). Compare this with a standard function evaluation. Notice also that the sup-star computation becomes much more involved in case a non-singleton fuzzifier is used.

Finally, by using the Mamdani rule interpretation (3.7), (3.9) can be written

\[ \mu_{B_j}(y, \cdot) = \mu_{A_j}(\varphi(t), \cdot) \ast \mu_{B_j}(y, \cdot) = w_{A_{j,1}} \ast \ldots \ast w_{A_{j,r}} \ast \mu_{B_j}(y, \cdot) = w_{A_j} \mu_{B_j}(y, \cdot), \tag{3.10} \]

where \(w_{A_j} \in [0, 1]\) is a weight known as the *degree of fulfillment*, or *firing strength*, of the rule. Clearly, the higher value on this weight the higher value on \(\mu_{B_j}(y, \cdot)\). In particular, with \(w_{A_j} = 1\) we get the intuitively reasonable result that \(\mu_{B_j}(y, \cdot)\) equals \(\mu_{B_j}(y, \cdot)\).

**Defuzzification.** The last issue to consider is how to aggregate the generated fuzzy sets into a form that can be converted into a crisp output \(\hat{y}(t|\theta) \in \mathbb{R}\). For this purpose we use a defuzzifier, which returns the crisp value that in some sense best corresponds to the possibility distribution of the combined output fuzzy sets. Because there is no universally correct way of doing this quite a few different defuzzification schemes have been suggested in the literature (Driankov et al., 1993).

Dominating in the fuzzy control genre is center of area (COA) or center of sums (COS) defuzzification, which are chosen much because of performance reasons (Lee, 1990). The *center of sums*
method operates on the rules on an individual basis:

\[
\hat{y}(t|\theta) = \frac{\int_y y \cdot \sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \mu_{B_{j_1 \ldots j_r}}(y, \theta) \, dy}{\int_y \sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \mu_{B_{j_1 \ldots j_r}}(y, \theta) \, dy}
\]  

(3.11)

and is here preferred to the COA method since it does not involve a complex rule aggregation part.

**Network representation.** To predict \( \hat{y}(t) \) using (3.11) it now only remains to specify which \(*\)-operator to use. Although max is often employed for \(*\) in control applications (Lee, 1990), it is not that suited from an estimation point of view. The main reason is that it introduces discontinuities, which may lead to problems when computing gradients. Another problem that may occur when updating MF parameters is that certain MFs can affect the overall mapping at one iteration, while at the next iteration they have no impact on the mapping whatsoever. It is not hard to realize that such a behavior can confuse the estimation algorithm quite a bit. Algebraic product (multiplication) on the other hand shows none of these deficiencies, thus making it better suited for identification purposes.

The integrals of (3.11) are in general also rather costly to compute. This is serious as the number of integrals that must be computed is at least the number of parameters \( d \) times the number of iterations needed in the estimation procedure. However, by restricting the MFs of the rule consequences to be fuzzy singletons, i.e., each \( \mu_{B_j}(y, \cdot) \) is 1 in one single, yet unknown, point \( \alpha_j \in \mathbb{Y} \), the predictor structure (3.11) simplifies to the *Mamdani fuzzy model structure*

\[
\hat{y}(t|\theta) = \frac{\sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \prod_{k=1}^{r} \mu_{A_{j_1 \ldots j_r}}(\varphi_k(t), \beta_{j_k}, \gamma_{j_k}, k) \alpha_j w_{A_j}}{\sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \prod_{k=1}^{r} \mu_{A_{j_1 \ldots j_r}}(\varphi_k(t), \beta_{j_k}, \gamma_{j_k}, k) \alpha_j w_{A_j}}
\]  

(3.12)

The corresponding network of (3.12) is reproduced in Fig. 3.2, from which it is evident that it belongs to the composition type of series expansions that was discussed in Sect. 2.2, although here
all parameters have linguistic meanings. The simplicity of structure (3.12) is the main reason for why we focus on it in the following sections. Another and more theoretical reason is the fact that it is actually capable of approximating any real continuous function on a compact (closed and bounded) domain to any degree of accuracy (Kosko, 1992; Wang, 1992; Wang and Mendel, 1992).

To this end, observe that each $\alpha_j$ can be viewed as a local model (a leaf in the model regression tree terminology), which is active to a degree $w_{A_j}$. An obvious generalization is to replace the constants $\alpha_j$ with more complex local models, such as linear regressions. This gives what is known as the Takagi-Sugeno model structure (Takagi and Sugeno, 1985; Sugeno and Kang, 1988; Sugeno and Yasukawa, 1993), which also has been suggested by (Johansen, 1994; Johansen and Foss, 1994) (who suppressed the fuzziness and instead referred to the approach as “operating regime based identification”). The concluding part of each fuzzy rule is thus replaced by a linear-in-the-parameters predictor structure for which it is hard to find a linguistic interpretation.

4. Fuzzy identification based on prior knowledge

This section considers various system identification issues based on the model structure (3.12). In Sect. 4.1 we further discuss and motivate the use of a rule base provided by an expert, as opposed to a pure black box approach where the rule base itself is estimated. The following three subsections constitute the core of the section. Their purpose is to provide answers to three fundamental grey-box-type of questions: “how to estimate the parameters of structure (3.12)?” – Sect. 4.2, “how to preserve the expert knowledge when updating these parameters?” – Sect. 4.3, and lastly “how to ensure certain non-structural system features, in this case how to guarantee a monotone steady-state gain curve?” – Sect. 4.4. Finally, Sect. 4.5 presents three fuzzy hybrid modeling approaches, of which two aim at reducing the number of parameters to estimate, and one aims at modeling dynamics that could not be captured by the fuzzy model.

4.1 Black versus grey box approaches

There are two main ways of determining the regressors, the number of MFs and the number of rules to use in the model structure (3.12). The black box way implies that the structure is estimated from data and the modeling (grey box) way that it is provided by an expert.

Structure estimation. Structure estimation can be split into two separate tasks. The first one is to determine what regressors $\varphi(t)$ to construct from the data $z(t)$. This is clearly not a fuzzy specific problem but is present in all black box approaches. The matter is usually resolved by restricting the regressors to delayed versions of the measurements. What regressors to include can be determined in many different ways, e.g., by statistical screening tests (Draper and Smith, 1981), by certain clustering techniques (Aguirre and Billings, 1995), by trial and error, and so forth. Here the key thing to recognize is that regressor selection is a combinatorial problem, growing rapidly in complexity with the number of possible regressors.

Having chosen $r$ regressors the second task is to find the number of different membership functions associated with each regressor, the number of different centers $\alpha_j$ and the number of rules $n$. As mentioned earlier, the number of rules in a complete rule base is

$$n = \prod_{k=1}^{r} n_k, \quad (4.1)$$

which clearly suffers from the curse of dimensionality problem, i.e., the number of possible rules increases exponentially with $r$. Observe that even moderate values of $r$ and $n_k$ give a large $n$, e.g., $r = n_1 = \ldots = n_r = 4$ result in $n = 256$. Moreover, a large $n$ often leads to many MF parameters to estimate, which in turn easily leads to overfit problems (recall the bias-variance trade-off). For many complex modeling problems the typical situation is that only some of the $n$
rules are important for describing the underlying system. The modeling of glycemic variations in the human body detailed in (Sjöberg et al., 1995), e.g., includes only 64 out of 1620 possible rules. Hence we should keep the $m$ out of $n$ rules that are most influential, i.e., we should again solve a combinatorial optimization problem.

Now, in the light of the fact that iterative search schemes typically must be employed for MF parameter estimation (see the next section) it is highly impractical to apply exhaustive search algorithms for rule base structure estimation. This has motivated methods that try to explore the most promising alternatives in one way or another. It seems that three main classes of structure estimation algorithms have been suggested in the literature: those that successively try to partition the regression space in a tree like manner (Takagi and Sugeno, 1985; Sugeno and Kang, 1988; Higgins and Goodman, 1994; Sun, 1994), those that provide non-optimal but often good enough solutions (simulated annealing, genetic algorithms, etc.) (Kirkpatrick et al., 1983; Ishigami et al., 1993; Lin and Lee, 1996), and those that apply clustering techniques (Sugeno and Yasukawa, 1993; Yoshimari et al., 1993; Babuska and Verbruggen, 1994; Kaymak and Babuska, 1995).

Even if these methods usually are far less computationally demanding than exhaustive search, they are still quite complex chiefly due to that structure estimation in general cannot be separated from the task of adjusting MF parameters. Another problem is that it is not that easy to say how many different output centers $\alpha_j$ to estimate. The fuzzy modeling advantages listed in the first section are also weakened or even lost if a black box approach is followed.

**Modeling.** The modeling approach overcomes these difficulties, at least partly. The computational cost is reduced at the price of the time it takes for the expert to provide the knowledge. At the linguistic level the knowledge required is the name of the linguistic variables, the corresponding linguistic values and the rule base itself. This is later complemented with descriptions of how to create the regressors, the initial shape of the MFs and other numeric information (e.g., parameter restrictions). Because natural language is rather coarse it can here be argued that the rule base often becomes relatively small, and that superfluous parameters therefore can be more easily avoided compared to if a black box approach is adopted.

Perhaps the most important advantage with the modeling approach is due to the curse of dimensionality and comes in terms of extrapolation capabilities. To see this, recall that the regression vector lives in $\mathbb{R}^r$. Even for a moderate $r$ the observations $\varphi(t)$ are by necessity sparse in any bounded region of $\mathbb{R}^r$. For example, filling up the unit cube in $\mathbb{R}^r$ using a grid of granularity 0.1 requires a million measurements. Since such excessive amount of data cannot be collected in practice there will always be regression regions having no real data support. By using a black box method, rules corresponding to such regions are likely to be removed because they do not improve the optimization criterion. This is in contrast to the modeling approach, which allows the combination of data-expert and pure expert explained regression regions. However, in case it is hard to determine the latter regions, it must here be required that only “data supported MF parameters” are updated by the estimation algorithm.

A potential drawback with the modeling approach is now that it can be quite arduous to capture all the important system features, especially in a complex modeling situation. It is then reasonable to condense what is actually linguistically known into a fuzzy rule base, and then try to describe the remaining dynamics within some other model structure, e.g., by letting a black box structure (a standard neural network, etc.) operate in parallel with the fuzzy ditto. Such hybrid approaches will be further investigated in Sect. 4.5. Altogether these facts and possibilities suggest that the strength of fuzzy identification really shows up when the modeling path is followed.

### 4.2 Parameter estimation

The model structure (3.12) is a regular mathematical function with tunable parameters $\theta$ of the form (2.8). By adopting the quadratic performance criterion (2.6) and for a moment neglecting that the parameters have linguistic meanings, this merely reduces to a standard unconstrained
nonlinear least-squares problem, where the nonlinear nature stems from the fact that the scale $\beta$ and the position $\gamma$ parameters enter the predictor in a nonlinear fashion.

The nonlinear least-squares estimation algorithms presented in Sect. 2.3 rely on the assumption that the Jacobian (2.17) can be constructed for any $\theta \in \mathcal{D} \subset \mathbb{R}^d$. A possible complication with the fuzzy predictor (3.12) occurs if $\varphi(t)$ and $\theta$ are such that the denominator of the predictor evaluates to zero. This takes place in the rare situations when also the numerator is zero and is due to that no rule is able to explain the current $\varphi(t)$. The natural way around the difficulty is simply to exclude samples causing an undefined $\hat{y}(t|\theta)$.

The Jacobian can now be constructed if the derivatives of the individual MFs with respect to their parameters exist. This is always the case for the $\alpha_j$ parameters owing to that they enter the predictor linearly, and thus it suffices to investigate the derivatives of the MFs at the input side.

Both network-classic (Definition 3.2) and Zadeh-formed (Definition 3.3) MFs have well-defined and continuous derivatives with respect to any of their parameters, at least if pathological cases are excluded (e.g., $\beta = 0$ of a Gaussian, $\gamma_1 = \gamma_2$ of a Z-formed MF, etc.). This is, however, not the case for piecewise linear MFs (Definition 3.4), and a rather common misunderstanding is therefore that Jacobian based estimation algorithms cannot be used in such a case. To put an end to this misconception we first observe that the only points without well-defined derivatives correspond to data located at the breakpoints (corners) of these MFs. Since the breakpoints are finitely many in any universe of practical interest, it follows that there will be no data points at these positions in the generic case. Nevertheless, in order to really guarantee a well-defined estimation problem we simply adopt the convention that derivatives at the breakpoints are zero, which means that any such data is excluded from the criterion fit. We thus conclude that the Jacobian (2.17) can be formed regardless of which of the mentioned MFs that are used, and that algorithms built on it can be employed to estimate the parameters of the structure (3.12).

Based on these observations a suitable MF parameter estimation procedure is as follows.

1. Fix all $\beta$ and $\gamma$ parameters and estimate $\alpha$ using an unconstrained linear least-squares method (see Sect. 2.3). This gives a rough idea of the quality of the rule base and provides further clues on how to choose the initial values of the $\alpha$ components.

2. Next, let all parameters loose and estimate these using either the Levenberg-Marquardt or the damped pseudo-inverse variant of the Gauss-Newton algorithm, which both are equipped with regularization, thus meaning that only data supported MF parameters are updated (see Sect. 2.3). Besides this extrapolation feature it is worth emphasizing that another decisive reason for using one of these schemes instead of a pure gradient procedure (which is quite common in this area) is due to their superior convergence properties near the minimum (Dennis and Schnabel, 1983). Notice though that a gradient descent method can be warranted in on-line situations where real-time aspects are especially important.

The main problem with this straightforward approach is now that nothing hampers the parameters from being updated in such a way that the original meaning of the parameters is lost. In what follows we show how to avoid such a behavior.

### 4.3 Preserving the meaning of a fuzzy rule base

The MFs and their parameters $\theta$ are directly coupled to certain linguistic values. What is important to recognize is that these values are ordered. Consider, e.g., the linguistic variable $\text{SPEED}(t)$

$$\langle \text{SPEED}(t), \mathcal{A}(\cdot, \cdot) = \{\text{LOW, RATHER LOW, MEDIUM, RATHER HIGH, HIGH}\}, D\rangle,$$  \hspace{1cm} (4.2)

which can be assigned to five different values, here listed in an order representing higher and higher speed. It is of course crucial that this order is reflected by the corresponding MFs. If this is the case for all involved linguistic variables, then we say that the corresponding fuzzy model is *linguistically*
Fig. 4.1. Restrictions on the parameters $\alpha$, $\beta$ and $\gamma$ that are necessary for a linguistic variable (here \texttt{speed}(t)) to have a reasonable linguistic interpretation.

sound. The important question is now how to relate this soundness concept to the parameters of the membership functions.

In case \texttt{speed}(t) is an output linguistic variable, then the MFs characterizing its linguistic values are fuzzy singletons located at positions $\alpha_{j_1}, \ldots, \alpha_{j_n} \in \mathbb{Y}$. Assuming that the order of these centers corresponds to the order of the linguistic values, the interpretation of \texttt{speed}(t) is preserved if and only if

$$\alpha_{j_1} < \alpha_{j_2} < \cdots < \alpha_{j_n}.$$  \hfill (4.3)

See plot a of Fig. 4.1. For $n$ different linguistic values of the output variable this merely generalizes to the parameter restrictions

$$\alpha_{j_1} < \alpha_{j_2} < \cdots < \alpha_{j_n}.$$  \hfill (4.4)

The ordering of the linguistic values is also essential for the linguistic variables involved in the rule antecedents, but then the situation becomes more complicated owing to that the corresponding membership functions are more complex.

Suppose that \texttt{speed}(t) is a linguistic input variable (coupled to the $k$-th regressor) with linguistic values described by network-classic MFs. For these MFs to have a linguistic interpretation a first requirement is that the five position parameters $\gamma_{j_1, k}$ are ordered as

$$\gamma_{1, k} < \gamma_{2, k} < \gamma_{3, k} < \gamma_{4, k} < \gamma_{5, k}.$$  \hfill (4.5)

Furthermore, the scale parameters $\beta_{1, k}, \ldots, \beta_{5, k}$ can be divided into two classes: those that must be positive, $\beta^+$, and those that must be negative, $\beta^-$. The latter category is applicable for MFs
of sigmoidal type and are used to reflect that the degree of membership decreases when the input increases. If, on the other hand, the membership degree should increase with the input or if the MF is a Gaussian, then the scale parameter must be positive. See b of Fig. 4.1. The generalization to \( r \) regressors each with \( n_k \) different linguistic values listed in increasing notion is obvious:

\[
\gamma_{1,k} < \gamma_{2,k} < \ldots < \gamma_{n_k,k}, \quad k = 1, \ldots, r,
\]

\[
0 < -\beta^-, \quad 0 < \beta^+.
\]  

These restrictions still allow some inconsistencies as is also illustrated in b of Fig. 4.1. The problem is that the membership degree of MEDIUM exceeds the membership degree of RATHER HIGH for large input values, which is quite unlogical. To cope with this difficulty one idea is to impose further restrictions on \( \beta \). However, this is not that easy (unless unnecessarily hard restrictions are inflicted) because what can be accepted depends partly on the current universe of discourse and partly on the values of the involved position parameters.

Zadeh-formed or piecewise linear MFs are both able to resolve this dilemma as they do not involve scale parameters. The \( j_k \)-th linguistic value of the \( k \)-th linguistic variable is instead characterized by an MF with two, three or four position parameters \( \gamma_{j_k,k} \). Let the components of \( \gamma_{j_k,k} \) be ordered on \( U_k \) and denote \( \gamma_{j_k,k}^i \) for \( i \in \{a, b, c, d\} \), where \( a \) denotes the point in \( U_k \) from which the membership degree starts to increase, \( b \) is the point where it reaches a full degree of membership, \( c \) is the point where the membership degree starts to decrease (a parameter if and only if \( b \neq c \)), and, finally, \( d \) is the point from which the membership degree is zero. For each individual Zadeh-formed or piecewise linear MF it must thus hold that

\[
\gamma_{j_k,k}^a < \gamma_{j_k,k}^b < \gamma_{j_k,k}^c < \gamma_{j_k,k}^d,
\]

where parameters not present in the current MF are simply removed from (4.8). In order to maintain a reasonable meaning of the linguistic variables we must also guarantee the ordering of their respective MFs. For the linguistic variable SPEED(t) we obtain a language consistent meaning if, besides (4.8), it additionally holds that (see c or d of Fig. 4.1)

\[
\gamma_{2,k}^a < \gamma_{3,k}^a < \gamma_{4,k}^a < \gamma_{5,k}^a
\]

\[
\gamma_{1,k}^a < \gamma_{2,k}^b < \gamma_{3,k}^b < \gamma_{4,k}^b < \gamma_{5,k}^b
\]

\[
\gamma_{1,k}^d < \gamma_{2,k}^d < \gamma_{3,k}^d < \gamma_{4,k}^d.
\]

In case there are \( r \) linguistic variables each with \( n_k, k = 1, \ldots, r \), different and ordered linguistic values this generalizes to

\[
\gamma_{1,k}^a < \gamma_{2,k}^a < \ldots < \gamma_{n_k,k}^a
\]

\[
\gamma_{1,k}^b < \gamma_{1,k}^c < \gamma_{2,k}^b < \gamma_{2,k}^c < \ldots < \gamma_{n_k,k}^b < \gamma_{n_k,k}^c
\]

\[
\gamma_{1,k}^d < \gamma_{2,k}^d < \ldots < \gamma_{n_k,k}^d,
\]

where, as before, parameters not present in the corresponding MFs are removed from (4.12)-(4.14).

By a proper modeling procedure the initial values of \( \alpha, \beta \) and \( \gamma \) will agree with the applicable restrictions from above, i.e., the initial value of \( \theta \) can be assumed to correspond to the feasibility region \( D \). In order to also ensure a feasible parameter estimate when using an unconstrained algorithm it must here be required that the parameter update at any stage is such that the constraints are not violated. Otherwise convergence can be to a (local and undesired) minimum where the parameters cannot be linked to the linguistic domain.

Although this problem does not always occur, it still appears surprisingly often in practice, even for simple static systems as is demonstrated in (Lindskog, 1996). There are several plausible reasons for this. One is that certain regression regions may be reflected by few and noisy data points that actually suggest an infeasible update of the parameter vector. Notice that this is likely to happen
when many parameters are estimated, i.e., when the model structure is too flexible. Another reason is that the initial parameter values and thereby the initial shape of the corresponding membership functions are just too inaccurate.

The only way to really guarantee that the expert knowledge is preserved is to take the parameter restrictions into account in the estimation phase. Since all restrictions considered thus far are inequalities this is a situation that straightforwardly can be handled through the barrier function estimation approach discussed in Sect. 2.3. More specifically, we introduce a number of simple inequality constraints $c_j(\theta)$:

\[
\begin{align*}
\text{if } 0 < \theta_i, & \text{ then let } c_j(\theta) = \theta_i, \\
\text{if } \theta_i < \theta_{i+1}, & \text{ then let } c_j(\theta) = \theta_{i+1} - \theta_i,
\end{align*}
\]

which, when put together, bring about any of the parameter restriction “chains” (4.4), (4.6)–(4.7) or (4.8), (4.12)–(4.14), thereby defining $\mathcal{D}$.

Further parameter restrictions of inequality type can of course also be imposed via barrier functions. We can, e.g., restrict a position parameter to any interval, thus guaranteeing that an MF stays in a position where the corresponding linguistic value is considered valid.

Another possibility is to allow soft parameter knowledge which is more or less consultative in nature, and whose purpose it is to try to balance the expert and the data knowledge by assessing the quality of the initial parameter values. The easiest method to achieve such a behavior is to use explicit regularization as was briefly discussed in Sect. 2.4. As for the barrier function method, the basic idea is to add a penalty term to the objective function of (2.25), i.e., to find parameters by iteratively minimizing something like

\[
\hat{\theta}^{(k+1)}_N = \arg\min_{\theta \in \mathcal{D}} \left( \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} (y(t) - \tilde{y}(\theta(t)))^2 + \rho \sum_{j=1}^{l} \rho(c_j(\theta)) + \sum_{j=1}^{l} \mu_i \left( \theta_i - \theta_i^0 \right)^2 \right),
\]

where $\theta_i^0$ denotes the initial value of the $i$-th parameter, and $\mu_i > 0$ is a user-tunable parameter expressing the relative belief in the value of $\theta_i$. In effect, a large $\mu_i$ (compared to the other $\mu_j$) implies that the cost for moving $\theta_i$ away from $\theta_i^0$ becomes high, thus expressing that its value is believed to be close to $\theta_i^0$.

### 4.4 Guaranteeing certain non-structural system properties

Many dynamic processes are known (from simple physics) to have a steady-state gain curve that is monotonically increasing (decreasing) in the inputs. Consider, e.g., a simple tank system where the inflow is the input and the liquid level the output. Here it is known that a certain constant inflow eventually leads to a “constant” liquid level. Starting from such a steady-state condition it is also known that an increase in the inflow causes the liquid level to increase (in a non-oscillatory manner) and settle at a higher level. This is a non-structural system property that is extremely important to retain in certain applications, e.g., when the model is going to be used in a predictive control arrangement (Koivisto, 1995).

The main problem is now that by applying a flexible nonlinear black box model structure (a neural network, etc.), then it can be quite hard to ensure this monotonicity feature, especially if there are regression regions with few and noisy data. To remedy this, we will here consider a restricted variant of the fuzzy model structure (3.12) that guarantees an increasing (decreasing) function mapping from the regression space $\mathbb{R}^n$ to the output space $\mathbb{R}$. This structure together with a proper choice of regressors $\varphi(t)$ (delayed inputs and outputs only) result in dynamical models showing the desired monotone behavior.

A conceptually simple way to ensure monotonicity is to first restrict the MFs at the input side to correspond to fuzzy partitions (Brown and Harris, 1994; Sjöberg et al., 1995).
Definition 4.1 (fuzzy partition). Suppose that the $k$-th linguistic variable can be assigned to $n_k$ different values each described by a membership function $\mu_{A_{j,k}}(\varphi_k(t), \beta_{j,k}, \gamma_{j,k}) \in U_k$. These MFs form a fuzzy partition if it holds on the entire domain $U_k$ that

$$\sum_{j=1}^{n_k} \mu_{A_{j,k}}(\varphi_k(t), \beta_{j,k}, \gamma_{j,k}) = 1.$$  

(4.18)

By imposing this restriction on all the $r$ linguistic variables (regressors) and additionally assuming that the rule base is complete in the sense that it covers the whole input domain $U^r$, it immediately follows that the model structure (3.12) simplifies to

$$\hat{y}(t|\theta) = \sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \alpha_{j_1, \ldots, j_r} \prod_{k=1}^{r} \mu_{A_{j_k,k}}(\varphi_k(t), \beta_{j_k,k}, \gamma_{j_k,k}) = \sum_{j_1=1}^{n_1} \cdots \sum_{j_r=1}^{n_r} \alpha_j w_{A_j}.$$  

(4.19)

Before proceeding, notice that a fuzzy partition puts certain demands on the MFs and their parameters. For example, we cannot in general use sigmoidal or Gaussian MFs because of their spreading and curvature. Zadeh-formed or piecewise linear MFs on the other hand can readily be parameterized so that a fuzzy partition is obtained. See Fig. 4.2.

Besides simplifying the predictor structure (no normalization is needed) a fuzzy partition always leads to fewer parameters at the input side. The five MFs shown in Fig. 4.2 are, e.g., described by 6 parameters only, whereas a full degree of freedom parameterization of the MFs implies 14 parameters as is shown in Fig. 4.1. Still, of course, the remaining problem is that the complexity of the predictor (4.19) typically increases rapidly with $r$.

Consider now the case with a single input linguistic variable ($r = 1$). Guaranteeing that the predictor is monotonically increasing in $\varphi(t)$ can be done in many different ways (especially if the origin of (4.19) is neglected), but then it can be quite hard to express restrictions on the parameters that ensure that monotonicity is preserved in the estimation step. However, this is a simple task when the input MFs form a fuzzy partition.

To see that this is the case, assume that all input MFs are ordered on the universe $U$ in such a way that $\mu_{A_j}()$ reaches a full degree of membership for a value of $\varphi(t)$ that is lower than what is the case for $\mu_{A_{j+1}}()$. See Fig. 4.2. If the ordered MFs at the input side form a fuzzy partition and the corresponding centers $\alpha_j$ reflecting the output MFs are such that

$$\alpha_1 < \alpha_2 < \ldots < \alpha_m,$$  

(4.20)

then $\hat{y}(t|\theta)$ will show a monotonically increasing behavior. In verifying this, we first notice that at intervals where the $j$-th input MF is fully active then the corresponding output becomes $\alpha_j$. With
fuzzy partitions constructed by Zadeh-formed or piecewise linear MFs we also have that
\[
\hat{y}(t|\theta) = \alpha_j \mu_{A_j}(-) + \alpha_{j+1} \mu_{A_{j+1}}(-) = (\alpha_{j+1} - \alpha_j) \mu_{A_{j+1}}(-) + \alpha_j
\]  
(4.21)

for all intervals \([\gamma_j, \gamma_{j+1}] \subseteq U\) such that \(\mu_{A_j}(-)\) and \(\mu_{A_{j+1}}(-)\) are not always zero. Since \(\alpha_{j+1} > \alpha_j\) (equality gives a constant output on the current interval) and \(\mu_{A_{j+1}}(-)\) is an increasing function on \([\gamma_j, \gamma_{j+1}]\) it follows that also \(\hat{y}(t|\theta)\) is an increasing function on that interval, with values ranging from \(\alpha_j\) to \(\alpha_{j+1}\). These facts altogether give that the overall predictor is a non-decreasing function.

To get a strictly increasing mapping it additionally must be required that all the input MFs lack intervals (flat parts) with a full degree of membership.

Two things are worth emphasizing before considering systems with several input linguistic variables. The first is that Zadeh-formed MFs always result in models with local plateaus at each \(\gamma_j\) position. This is a behavior that is quite unrealistic in certain applications, thus favoring piecewise linear MFs as these do not introduce such plateaus (unless trapezoidal MFs are used).

The second observation is that the restrictions imposed by a fuzzy partition typically reduces the risk for position changes among the parameters \(\alpha\) and \(\gamma\), which in terms of estimation algorithms means that it is often sufficient to use an unconstrained procedure instead of a constrained one.

Now, in order to generalize the above result to predictors having \(r\) regressors it is instructive to first formally define what is meant by a monotonically increasing predictor in \(\varphi(t)\).

**Definition 4.2 (regressor ordering).** Let \(\varphi(t), \tilde{\varphi}(t) \in \mathbb{R}^r\). We say that \(\varphi(t) \geq \tilde{\varphi}(t)\) if \(\varphi_j(t) \geq \tilde{\varphi}_j(t)\) for \(j = 1, \ldots, r\).

**Definition 4.3 (monotonically increasing predictor).** Let \(\varphi(t), \tilde{\varphi}(t) \in \mathbb{R}^r\). We say that a predictor \(g(\varphi(t), \theta)\) is monotonically increasing in the regressors if whenever \(\varphi(t) \geq \tilde{\varphi}(t)\) it holds that \(g(\varphi(t), \theta) \geq g(\tilde{\varphi}(t), \theta)\).

Using the latter definition we now have the following central theorem.

**Theorem 4.1.** Let the model structure be complete and given by (4.19). If, for all \(k = 1, \ldots, r\), it holds that
\[
\sum_{j_k=1}^{n_k} \alpha_{j_1, \ldots, j_k} \mu_{A_{j_k}}(\varphi_k(t), \beta_{j_k, k}, \gamma_{j_k, k})
\]  
(4.22)

are monotonically increasing functions in \(\varphi_k(t)\) on \(U_k\) for all possible combinations of fixed values of \(j_1, \ldots, j_{k-1}, j_{k+1}, \ldots, j_r\), then the predictor (4.19) is monotonically increasing in \(\varphi(t)\).

**Proof.** See (Lindskog, 1996), page 195.

The main point with Theorem 4.1 is that it is sufficient to work with one-dimensional functions. A simple way to ensure increasing functions in all \(\varphi_j(t)\) is therefore to restrict the input MFs to fuzzy partitions and order the corresponding centers as was done in the one-dimensional case.

**Lemma 4.1.** Let the model structure be (4.19) and let \(\varphi_k(t)\) denote one of its regressors. Assume that the ordered (on \(U_k\)) MFs associated with \(\varphi_k(t)\) are either Zadeh-formed or piecewise linear and such that they form a fuzzy partition. If, for all possible combinations of \(j_1, \ldots, j_{k-1}, j_{k+1}, \ldots, j_r\), it holds that
\[
\alpha_{j_1, \ldots, j_k, \ldots, j_r} \leq \alpha_{j_1, \ldots, j_k+1, \ldots, j_r},
\]  
(4.23)

for all \(j_k = 1, \ldots, n_k - 1\), then every
\[
\sum_{j_k=1}^{n_k} \alpha_{j_1, \ldots, j_r} \mu_{A_{j_k}}(\varphi_k(t), \gamma_{j_k, k})
\]  
(4.24)

is a monotonically increasing function in \(\varphi_k(t)\) on \(U_k\).
If the requirements of Lemma 4.2 are fulfilled, then we get a predictor with a monotonically increasing steady-state gain curve in the input. Moreover, starting from a steady-state solution and increasing the input in a stepwise fashion, it follows by simple induction that \( \dot{y}(t|\theta) \) increases monotonically with \( t \). This in particular means that the predictor shows a non-oscillatory step response behavior, which is a restriction but also a property that is valid for many industrial processes (e.g., thermal systems). We will apply fuzzy identification based on (4.19) to one such process in Sect. 5.
4.5 Fuzzy hybrid modeling—Some possibilities

Although a fuzzy model of the form (3.12) is able to theoretically approximate any “well-behaved” system to a desired degree of accuracy, this may require a “too” complex rule base, particularly if \( r \) is large. With the aim of reducing the complexity of the models, while also maintaining (or even enhancing) their performance, it is interesting to marry together fuzzy and other identification approaches. This can be done in many different ways.

**Fuzzy modeling based on physically induced regressors.** A novel first idea is to keep structure (3.12), but apply more involved and physically motivated regressors (linguistic variables) than just delayed in- and outputs. Parts of the important system nonlinearities can then be captured directly in the regressors, thus typically leading to that fewer MFs (parameters) and/or regressors \( r \) are needed in the resulting models. For example, in order to model the power delivered by a heater element (a resistor of some kind), an obvious physically motivated regressor to use would be the squared voltage applied to the heater. In other and more sophisticated modeling situations, suitable regressors can be implicitly given in terms of some dynamic and/or static equations. To then arrive at a set of physically induced regressors requires both symbol and numeric computations as is stressed in (Lindskog and Ljung, 1995, Lindskog, 1996).

**Combining fuzzy and traditional grey box modeling.** Many real-world systems are composed of several subcomponents, some of which are well-described directly in terms of physical principles (conservation laws, etc.) and some of which are better described in linguistic terms. This fact strongly motivates the use of several “small” and interacting fuzzy and grey box model structures, which when combined give the overall predictor. It is our opinion that such a model decomposition always should be considered in a complex modeling situation.

**Combining fuzzy and black box modeling.** Even if the above possibilities are contemplated there can still be important system phenomena that are hard to reflect within the fuzzy framework. As mentioned earlier, it is then appealing to complement the expert determined fuzzy structure by a sufficiently flexible black box ditto (typically a neural network), which is solely responsible for picking up the remaining dynamics. This naturally leads to the setup shown in Fig. 4.4, which in structure is similar to what is discussed and proved useful in (Forssell and Lindskog, 1997).

Notice that the parameters of the fuzzy structure \( \theta_f \) are first estimated based on the measurements \( y(t) \). The residuals \( \varepsilon_f(t|\hat{\theta}_f) = y(t) - \hat{y}_f(t|\hat{\theta}_f) \) are then formed and used for the tuning of the black box parameters \( \theta_b \). The main reason for using this particular scheme is that the obtained fuzzy model gives useful insight into the choice of black box structure, its size, and so forth.
5. Example – Tank level modeling

The objective in this application example is to model how the liquid level \( h(t) \) of a simple laboratory-scale tank system, shown in Fig. 5.1, changes with the inflow that is generated by the voltage \( u(t) \) applied to the pump. We see that the measured estimation data (1000 samples) rather well cover the interesting modeling domain.

To get a feeling for the nonlinearities it is useful to first take a closer look at the simulation behavior of a simple linear regression model, and compare this with experimental (validation) data. One of the best model structures having regressors of the form (2.4) involves three parameters only:

\[
\hat{h}(t|\theta) = \theta_1 h(t-1) + \theta_2 u(t-1) + \theta_3. \tag{5.1}
\]

Simulated outputs from the corresponding linear least-squares fitted model are compared to real tank measurements (1000 new samples) in the left plot of Fig. 5.2. The fit is clearly not that bad, yet the model output is physically impossible since it is sometimes negative. This is of course a nontrivial complication if we are going to use the model to study the behavior of the real system. In fact, all linear regression models with delayed in- and outputs as regressors show this defect.

A simple idea to overcome this difficulty is now to try some semi-physical modeling. The tank level change depends on the difference between in- and outflow (conservation of mass). While the inflow is roughly proportional to \( u(t) \), the outflow can be approximated using Bernoulli’s law, which for a small outlet hole states that the outflow is proportional to \( \sqrt{h(t)} \). By combining these facts, it is pretty straightforward to arrive at the nonlinear model structure (a linear regression)

\[
\hat{h}(t|\theta) = \theta_1 h(t-1) + \theta_2 u(t-1) + \theta_3 + \theta_4 \sqrt{h(t-1)}. \tag{5.2}
\]

By tuning the four parameters of structure (5.2) we obtain a model whose simulation behavior is detailed in the right plot of Fig. 5.2. Compared to the model of the form (5.1), the semi-physical model gives a physically sound response and is seemingly better except for large tank levels. Still, however, there is no guarantee that the model outputs are physically sound for other input values.

Before trying to counteract this it is expedient to list what is actually known about the process.

1. First of all, we know that the more inflow the higher will the liquid level be. The steady-state gain curve of the model should thus be monotonically increasing in \( u(t) = u^* \).

2. The input \( u(t) \) may vary from 3.5 to 7.5 V, which means that there is always a flow across the tank. Even if the estimation data set is of rather high quality it still shows some gaps. The tank is, e.g., never emptied nor is it completely filled up. However, we know for sure that these situations can occur for \( u(t) \in [3.5, 7.5] \) V. A good model should be equipped with these extrapolation capabilities.
where \( z \) and \( c \) chosen so that models describing the level of the tank depicted in Fig. 5.1/2.5/ contain 7 free parameters is reasonable to put further modeling effort into regions where model as the starting point and noticing its good performance at high levels (above 7 cm) it is sensible to describe each linguistic variables with few linguistic values:

\[
\langle \text{LEVEL}(t), \mathbf{A}(\cdot, \cdot) = \{\text{ZERO, VERY LOW, LOW, RATHER LOW, HIGH, MAX}\}, D : \hat{h}(t|\theta) \rangle,
\langle \text{LEVEL}(t-1), \mathbf{A}(\cdot, \cdot) = \{\text{ZERO, LOW, HIGH}\}, D : \varphi_1(t) = h(t-1) = z_1(t-1) \rangle,
\langle \text{VOLTAGE}(t-1), \mathbf{A}(\cdot, \cdot) = \{\text{LOW, HIGH}\}, D : \varphi_2(t) = u(t-1) = z_2(t-1) \rangle,
\]

\[
\hat{h}(t|\theta) \in \mathbb{Y} = [0, 3.5], \quad \varphi_1(t) \in \mathbb{U}_1 = [0, 3.5], \quad \varphi_2(t) \in \mathbb{U}_2 = [3.5, 7.5],
\]

where \( z(t) = [h(t) \quad u(t)] \). The listed system properties can now be guaranteed if the MFs

\[
\mu_{\text{ZERO}}(\hat{h}(t|\theta)) = \alpha_{1,1} = 0, \quad \mu_{\text{LOW}}(\hat{h}(t|\theta)) = \alpha_{2,1}, \quad \mu_{\text{HIGH}}(\hat{h}(t|\theta)) = \alpha_{3,1},
\]

\[
\mu_{\text{VERY LOW}}(\hat{h}(t|\theta)) = \alpha_{1,2}, \quad \mu_{\text{RATHER LOW}}(\hat{h}(t|\theta)) = \alpha_{2,2}, \quad \mu_{\text{MAX}}(\hat{h}(t|\theta)) = \alpha_{3,2} = 35,
\]

and

\[
\mu_{\text{ZERO}}(\varphi_1(t), \gamma_1, \gamma_2) = \mu_{A_{1,1}}(\varphi_1(t), 0, \gamma_2) = \text{mf}_1(\varphi_1(t), 0, \gamma_2),
\]

\[
\mu_{\text{LOW}}(\varphi_1(t), \gamma_1, \gamma_2, \gamma_3) = \mu_{A_{2,1}}(\varphi_1(t), 0, \gamma_2, 35) = \text{mf}_\text{tri}(\varphi_1(t), 0, \gamma_2, 35),
\]

\[
\mu_{\text{HIGH}}(\varphi_1(t), \gamma_1, \gamma_2, \gamma_3) = \mu_{A_{3,1}}(\varphi_1(t), \gamma_2, 35) = \text{mf}_\text{fr}(\varphi_1(t), \gamma_2, 35),
\]

\[
\mu_{\text{LOW}}(\varphi_2(t), \gamma_1, \gamma_2, \gamma_3) = \mu_{A_{1,2}}(\varphi_2(t), \gamma_1, \gamma_2) = \text{mf}_1(\varphi_2(t), \gamma_1, \gamma_2),
\]

\[
\mu_{\text{HIGH}}(\varphi_2(t), \gamma_1, \gamma_2, \gamma_3) = \mu_{A_{2,2}}(\varphi_2(t), \gamma_2, \gamma_2) = \text{mf}_\text{fr}(\varphi_2(t), \gamma_1, \gamma_2),
\]

are used in the fuzzy predictor

\[
\hat{h}(t|\theta) = \sum_{j_1=1}^{3} \sum_{j_2=1}^{2} \alpha_{j_1,j_2} \prod_{k=1}^{2} \mu_{A_{j_1,j_2}}(\varphi_k(t), \gamma_k),
\]

which contains 7 free parameters

\[
\theta = \begin{bmatrix} \alpha_{1,2} & \alpha_{2,1} & \alpha_{3,1} & \gamma_{1,2} & \gamma_{1,2} \end{bmatrix}^T
\]

chosen so that

Fig. 5.2. Simulation behavior (based on validation data) of typical ARX (left) and semi-physical (right) models describing the level of the tank depicted in Fig. 5.1.
Fig. 5.3. Premise (left) and consequence (right) MFs for describing the liquid level of the tank system. Dotted curves show the situation when only the centers $\alpha$ are estimated. Solid curves show the situation after constrained estimation subject to the constraints (5.8).

Fig. 5.4. Simulation (based on validation data) of unconstrained linear least-squares (left) and constrained (right) estimated fuzzy models reflecting the liquid level of the tank from Fig. 5.1.

\[0 = \alpha_{1,1} < \alpha_{1,2} < \alpha_{2,1} < \alpha_{2,2} < \alpha_{3,1} < \alpha_{3,2} = 35,\]
\[0 = \gamma_{1,1} < \gamma_{2,1} < \gamma_{3,1} = 35,\]
\[3.5 < \gamma_{1,2} < \gamma_{2,2} < 7.5.\]  

A graphical representation of the corresponding complete fuzzy rule base is shown in Fig. 5.3. Notice that the MFs associated with each regressor form a fuzzy partition and that this fact together with the restrictions (5.8) guarantee a monotonically increasing predictor in $\varphi(t)$. By Lemma 4.2 we also get a steady-state gain curve that is monotonically increasing in $u(t) = u^*$ (the first property). Furthermore, the extrapolation property is ensured by fixing some of the parameters: $\alpha_{1,1} = \gamma_{1,1} = 0$ and $\alpha_{3,2} = \gamma_{3,1} = 35$, thereby assuring that the predictor is able to return values in the whole output universe $Y$. The third property is finally guaranteed by the use of piecewise linear MFs in accordance with Equations (5.5).

With $\gamma$ fixed according to the left plot of Fig. 5.3 (dotted curves), unconstrained linear least-squares estimation of the four free centers $\alpha$ yields a feasible parameter estimate; see the upper right plot of Fig. 5.3. The simulation detailed to the left of Fig. 5.4 indicates also that this first model is rather good. Starting from this point, it is now true that unconstrained estimation of all seven parameters renders a model with a lower root mean square (RMS) error (0.70 compared to 0.78 for the first model), but then it becomes difficult to linguistically interpret the obtained model.
Fig. 5.5. Function mapping (left) and steady-state gain curve (right) of the fuzzy tank model obtained using constrained estimation.

Resolving this dilemma by performing constrained estimation subject to the constraints (5.8) gives linguistically sound MFs as is shown in Fig. 5.3 (solid curves). On top of that, this final model shows the best simulation performance of all models derived. Compare Figs. 5.2 and 5.4. The built-in increasing nature of the final predictor is now evident from the left plot of Fig. 5.5, and although this mapping is at a first sight quite similar to a linear one, it is from the steady-state gain curve of Fig. 5.5 clear that the model has an important nonlinear behavior in the interesting operating region. Notice also that this steady-state gain curve is monotonically increasing in $u(t) = u^*$.

From this discussion we conclude that the tank system can be accurately described by a fuzzy model having few (7) estimated parameters. A sound physical behavior is guaranteed by applying the model structure (4.19), which allows inclusion of certain extrapolation and steady-state gain monotonicity features. The latter property is especially important to reflect in certain predictive control applications as is stressed in (Koivisto, 1995). Even if models that are good from a loss function point of view are used, it is there illustrated that without such a property (when known from physics) severe stability problems often arise. This behavior is indeed related to the difficulties occurring when performing standard adaptive control based on linear models for which the sign of the first $B(q)$ parameter is incorrect (Åström and Wittenmark, 1995).

6. Practical aspects

This section addresses a number of practical issues that ought to be considered in connection with fuzzy grey box identification.

6.1 Model complexity

We have earlier stressed that the model complexity typically increases rapidly with the number of possible linguistic variables $r$, particularly if these can be assigned to many different linguistic values. While the number of variables can be reduced by a fuzzy hybrid approach (see Sect. 4.5), the number of linguistic values can be kept down by using a coarse description language.

In a way this is in conflict with the experts attempt of pursuing accurate linguistic modeling, yet the use of “few” MFs is often desirable from an estimation point of view, especially in complex modeling situations where the data is sparse in the regression space. The main reason for this is that a “too” dense MF configuration implies that the corresponding parameters are fit to few data, which typically leads to models that perform rather poorly for other data records.

To overcome these difficulties a good practice is often to start with a rather coarse rule base, and, if necessary, successively refine it. This can be accomplished by lumping together similar
linguistic values into one single notion: treat, e.g., VERY LOW, LOW and SOMEWHAT LOW as one linguistic value described by one MF. If the estimated coarse model is not good enough, then introduce new MFs based partly on the expert knowledge and partly on the performance of the coarse model. The use of the coarse model is foremost motivated by the fact that it provides local performance information, i.e., it gives useful data guided refinement information. It may also provide information about phenomena that were overlooked in the modeling phase. This fine-grain procedure is now iterated until (hopefully) a good enough model is found. Although user interaction is good for identifying and avoiding pitfalls, the major drawback with the approach is of course that it is rather time-consuming compared to pure black box modeling.

6.2 Robustness of the identification method

The success of any identification method relies on the descriptive power of the model structure as well as on the quality of the estimation data. Compared to a pure data driven identification approach, it is easier to avoid data caused pitfalls by using an expert determined model structure.

Concerning estimation algorithms, it is worth stressing that the preferred schemes of Sect. 2.3 are all robust in the sense that the fit of the tuned model is at least as good as what is obtained with the initial parameters. With a constrained estimation procedure, we can in addition guarantee that the estimated models are linguistically sound. Notice, though, that this does not imply that a “better” model is obtained, as this only can be assessed after a careful validation procedure.

A distinct advantage with expert modeling is that redundancy in terms of similar MFs as well as physically unsound regions can be avoided. Apart from reducing the model complexity, this also leads to less ill-conditioning problems. To fully handle ill-conditioning we used regularization, having the nice add-on property that it enables extrapolation into more or less expert explained regression regions. Such a regularizing effect is mediated in the algorithms through SVD computations, which are known to be numerically robust to carry out (Golub and Van Loan, 1989).

6.3 Software

An always present and relevant issue in system identification is the availability of software tools. The prototype package used for the above experiments consists of a number of MATLAB (The MathWorks, Inc., 1992) m-files, which can be downloaded from the library

ftp://ftp.control.isy.liu.se/pub/Software/Fuzzy/

Owing to that the model structure as well as the constraints are represented as strings, this package can be used for rather general predictors of the form (2.3), and not just fuzzy ones. For example, it can be applied directly to the fuzzy hybrid approaches suggested in Sect. 4.5.

However, working with strings on a textual basis is a bit awkward and error prone. This is a problem that can be relaxed significantly through a graphical user interface (GUI) of the kind provided by MathWorks’ fuzzy logic toolbox (Roger Jang and Gulley, 1995). The design of such GUI means is an obvious project for the future. Other important software projects for the future include the implementation of more efficient constrained estimation algorithms as well as the development of general and versatile validation procedures.

7. Conclusions and future work

After experiment design and data collection, a typical system identification session involves two main issues: model structure determination followed by parameter estimation. In this contribution we have considered fuzzy grey box identification, which assumes that the former problem is addressed, at least partly, by a human domain expert who indirectly describes the model structure in terms of a number of if-then rules. Taking various fuzzy and identification aspects into account
we arrived at the Mamdani fuzzy model structure (3.12), which, in a more traditional identification setting, is nothing but a series expansion of composition type having much in common with feed-forward neural networks, RBFN networks, model regression trees, etc.

This kinship in particular means that efficient Newton kind of algorithms (the pseudo-inverse Gauss-Newton or the Levenberg-Marquardt procedures) can be applied for MFs parameter estimation. Since these schemes are equipped with regularization it is to some extent possible to preserve expert knowledge having minor data support. However, the series expansions are usually rich in terms of the number of parameters. This fact, especially in combination with few and noisy data, sometimes leads to that the original linguistic interpretation of the rules are lost in the estimation step. To avoid such an undesired behavior it is necessary to impose certain restrictions on the MFs parameters, and then solve the obtained constrained minimization problem.

For some model based control applications it is also extremely important that the applied models reflect certain non-structural system properties, e.g., a monotonically increasing steady-state gain curve and/or a non-oscillatory step response behavior. Whereas such features are in general difficult to guarantee when using neural networks or other flexible series expansions, these can be dealt with by employing the special fuzzy partition based model structure (4.19). Experiments on real-world data – in this case a tank system – as well as other applications (Lindskog, 1996) have demonstrated the feasibility and the usefulness of this approach.

To this end, let us finally point to some extensions and open problems related to the fuzzy identification framework discussed above.

1. Stability and various robustness issues are very important when the models are going to be used in control applications. Then, how does the choice of MFs affect stability? How and to what extent can the linguistic system knowledge be exploited for robust control design? Can we apply modern stability tools stemming from the robust control field (unstructured uncertainties, etc.)? For this problem, (Suykens et al., 1995) has already suggested an interesting method based on a particular neural network model. The basic idea is to view the neural network as a nominal linear model with bounded nonlinear feedback perturbations, and then use a standard robust control design scheme. The obvious question is here if a similar procedure can be devised for fuzzy models as well.

2. To ensure a monotonic steady-state gain curve we restricted the MFs to correspond to fuzzy partitions. What other and perhaps better MF configurations are able to preserve this knowledge? Also, what other kinds of non-structural properties can be captured within the fuzzy framework?

3. A water heating system (Koivisto, 1995) with a known increasing steady-state gain curve behavior is successfully modeled in (Lindskog, 1996) using the fuzzy model structure (4.19). Because of seasonal temperature variations and some other factors this model is only valid under certain operational conditions. To also handle long time seasonal changes there is here a need for fuzzy specific and monotonicity preserving recursive estimation algorithm.

4. The applications considered and mentioned above are rather small. However, it is our belief that the use of linguistic expert knowledge really pays off for more involved processes. To investigate this it is worth looking further into application fields where verbal knowledge is dominating, as, e.g., is the case for many biomedical or biochemical systems.

Bibliography


Forssell, U. and P. Lindskog (1997). Combining semi-physical and neural network modeling: an example of its usefulness. Submitted to the 11th IFAC Symposium on System Identification (SYSID’97) to be held in Fukuoka, Japan, July 1997.


