8-1-2002

Sum Normal Optimization of Fuzzy Membership Functions

Daniel J. Simon
Cleveland State University, d.j.simon@csuohio.edu

SUM NORMAL OPTIMIZATION OF FUZZY MEMBERSHIP FUNCTIONS

DAN SIMON
Cleveland State University
Department of Electrical and Computer Engineering
1960 East 24th Street
Cleveland, Ohio 44115

Given a fuzzy logic system, how can we determine the membership functions that will result in the best performance? If we constrain the membership functions to a certain shape (e.g., triangles or trapezoids) then each membership function can be parameterized by a small number of variables and the membership optimization problem can be reduced to a parameter optimization problem. This is the approach that is typically taken, but it results in membership functions that are not (in general) sum normal. That is, the resulting membership function values do not add up to one at each point in the domain. This optimization approach is modified in this paper so that the resulting membership functions are sum normal. Sum normality is desirable not only for its intuitive appeal but also for computational reasons in the real time implementation of fuzzy logic systems. The sum normal constraint is applied in this paper to both gradient descent optimization and Kalman filter optimization of fuzzy membership functions. The methods are illustrated on a fuzzy automotive cruise controller.

Keywords: Learning; estimation; training; optimization; gradient descent; Kalman filtering; constraints.

1. Introduction

The design of a fuzzy logic system (FLS) includes the design of a rule base, the design of input scale factors, the design of output scale factors, and the design of the membership functions. Input scale factors transform the real inputs into normalized values, and output scale factors transform the normalized outputs into real values.

Some studies have shown that FLS performance is more dependent on membership function design than rule base design[1]. Other studies have discussed rule base design[2,3,4]. The tuning of input and output scale factors is known as context adaptation (because the scale factors are determined by the available data, i.e., the problem context). Some researchers have studied genetic algorithms for context adaptation[5,6]. Others have used genetic algorithms to design the rule base and the scale factors when the normalized membership functions are fixed[7]. Some studies used neural networks for context adaptation[8]. A statistical approach for input scaling has also been proposed[9]. This depends on the Gaussian distribution of the input data. A genetic learning process for the membership function design, coupled with a heuristic method for the rule base design, has been proposed[2]. A fuzzy training process for input scale factors has also been proposed[10].

This paper is restricted to the tuning of membership functions. Researchers have used many different methods over the past decade to optimize fuzzy membership functions. These methods include genetic algorithms[11,12], neural networks[13,14], evolutionary programming[15], geometric methods[16], fuzzy equivalence relations[17], heuristic methods[18], gradient descent[19,11,20], and Kalman filtering[21]. Other methods for membership optimization include the simplex method[22,23], least squares[24,25], and other numerical techniques[26].

Some of these methods use the derivatives of the fuzzy system's performance with respect to the membership function parameters, and some of these methods do not use these derivatives. Derivative-free methods can be desirable in that they do not require the derivative of the objective function with respect to the membership function parameters. They are more robust than derivative-based methods with respect to finding a global minimum and with respect to their applicability to a wide range of objective functions and membership function forms. However, they typically tend to converge more slowly than derivative-based methods. Derivative-based methods have the advantage of fast convergence but they tend to converge to local minima. In addition, due to their dependence on analytical derivatives, they are limited to specific objective functions, specific types of inference, and specific types of membership functions.

In this paper we present a modified form of the gradient descent and Kalman filter methods[27,28,21] for the optimization of asymmetric triangular membership functions. Gradient descent and Kalman filtering are effective for fuzzy membership function optimization but they result in membership functions that are not sum normal. That is, the membership function values do not add up to one at each point in the domain. Sum normal membership functions are desirable for several reasons. First, sum normality is assumed in some approaches to fuzzy decision making[29]. Also, sum normality is desired by many fuzzy system engineers for its aesthetic and intuitive appeal[30]. Some rule base reduction algorithms guarantee that a sum normal set of membership functions will remain sum normal even after rule base reduction[31]. Finally, fuzzy logic software can be written with less code and greater computational efficiency if it can be assumed that the membership functions are sum normal. This last item is simply an example of the general rule that software can be written smaller and faster if its inputs have more constraints and therefore the software requirements can be made less general.

Membership function optimization subject to the constraint of sum normality could also be performed via context adaptation. That is, a set of sum normal membership functions could be defined, and then scaling functions could be tuned under the constraint that the scaled membership functions remain sum normal. An approach similar to this has already been proposed[2]. However, in that paper a genetic algorithm was used for context adaptation. As mentioned above, this derivative-free method has the benefit that it can easily escape from a local minimum. On the other hand, there is no guarantee that the final solution is even locally minimum. The approach we consider in this paper is based on the derivatives of an error function with respect to the membership function parameters. This has the advantage of fast convergence to a local minimum, but some heuristics are needed to escape from a local minimum. This is not to say that one method is superior or inferior to another. The choice of derivative-based or derivative-free optimization must be based on tradeoffs between a wide range of issues, including the fidelity of the initial guess, computational effort, and flexibility with respect to membership function types.

The next section reviews the use of gradient descent and Kalman filtering for membership function optimization. Section 3 shows how those methods can be modified to guarantee sum normality in the resulting membership functions. Section 4 contains some simulation results of a fuzzy automotive cruise controller, and Section 5 contains some concluding remarks. The Appendix contains the derivative formulas that are used in this paper.

2. Fuzzy system optimization via gradient descent and Kalman filtering

We assume that our fuzzy system uses correlation-product inference[32], fit values are combined with the *min* operator, and the input and output membership functions are (possibly asymmetric) triangles. The initial rule base and some initial membership functions are given, perhaps constructed on the basis of experience, or trial and error. The generation of rule bases is a difficult and important task in the construction of fuzzy logic systems but is not discussed in this paper.

Consider the ith fuzzy membership function of the jth input z_j. We will denote its modal point as c_{ij}, its lower half-width as b_{ij}^-, and its upper half width as b_{ij}^+. The membership function attains a value of 1 when the input is c_{ij}. As the input decreases from c_{ij}, the membership function value decreases linearly to 0 at $c_{ij} - b_{ij}^-$, and remains at 0 for all inputs less than $c_{ij} - b_{ij}^-$. As the input increases from c_{ij}, the membership function value decreases linearly to 0 at $c_{ij} + b_{ij}^+$, and remains at 0 for all inputs greater than $c_{ij} + b_{ij}^+$. The degree of membership of the jth crisp input z_j in its ith fuzzy set is therefore given by

$$f_{ij}(z_j) = \begin{cases} 1 + (z_j - c_{ij})/b_{ij}^- & \text{if } -b_{ij}^- \leq (z_j - c_{ij}) \leq 0 \\ 1 - (z_j - c_{ij})/b_{ij}^+ & \text{if } 0 \leq (z_j - c_{ij}) \leq b_{ij}^+ \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We will further assume that our fuzzy system has only one output. This restriction is made only for notational convenience and does not affect the theoretical results presented herein. Suppose there are a total of M rules in the FLS. The consequent of the jth rule is a triangular fuzzy set with modal point γ_j, lower half-width as β_j^-, and upper half width β_j^+. That is, the fuzzy set of the consequent of the jth rule is given as

$$m_j(y) = \begin{cases} 1 + (y - \gamma_j)/\beta_j^- & \text{if } -\beta_j^- \leq (y - \gamma_j) \leq 0 \\ 1 - (y - \gamma_j)/\beta_j^+ & \text{if } 0 \leq (y - \gamma_j) \leq \beta_j^+ \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Suppose that the jth rule is a consequent of z_1 belonging to fuzzy set i and z_2 belonging to fuzzy set k. Then the activation level of the consequent of the jth rule is w_j, which is given as

$$w_j = \min[f_{i1}(z_1), f_{k2}(z_2)]. \tag{3}$$

So the fuzzy output when $z_1 \in$ fuzzy set i and $z_2 \in$ fuzzy set k is given as

$$\bar{m}_j(y) = w_j m_j(y). \tag{4}$$

The overall fuzzy output $m(y)$ takes into account the possibility that each input falls into more than one fuzzy set so more than one rule can be fired at the same time.

$$m(y) = \sum_{j=1}^{M} \bar{m}_j(y). \tag{5}$$

The fuzzy output is mapped to a crisp number $\hat{y}$ using centroid defuzzification[33].

$$\hat{y} = \frac{\sum_{j=1}^{M} w_j \Gamma_j J_j}{\sum_{j=1}^{M} w_j J_j}. \tag{6}$$

Γ_j and J_j are the centroid and area of the jth output fuzzy membership function. The centroid of $m_j(y)$, the jth output fuzzy set, is defined as as

$$\Gamma_j = \frac{\int y m_j(y)\, dy}{\int m_j(y)\, dy}. \tag{7}$$

After substituting (2) into the above equation and working through a couple of pages of straightforward calculus and algebra, we obtain

$$\Gamma_j = \frac{\beta_j^+(3\gamma_j + \beta_j^+) + \beta_j^-(3\gamma_j - \beta_j^-)}{3(\beta_j^+ + \beta_j^-)}. \tag{8}$$

This can easily be extended to the case where there are more than two inputs and one output but the notation becomes cumbersome.

If the fuzzy membership functions are triangles as assumed in this paper, derivative-based methods can be used to optimize the modal points and the half-widths of the input and output membership functions. Consider an error function given by

$$\begin{aligned} E &= \frac{1}{2N} \sum_{n=1}^{N} g_n E_n^2 \\ E_n &= \hat{y}_n - y_n. \end{aligned} \tag{9}$$

where N is the number of training samples, y_n is the target output of the fuzzy system, $\hat{y}_n$ is the actual output of the fuzzy system, and g_n is a time-dependent weighting function. The role of g_n will in illustrated in the example of Section 4. We can minimize E by using the partial derivatives of E with respect to the modal points and half-widths of the input and output fuzzy membership functions. We can obtain expressions for these derivatives using (1)-(6). Then, using the differentiation chain rule on (9), we can obtain expressions for the derivative of the error function with respect to the half-widths and modal points. We can then use those derivatives in an optimization scheme to minimize the error function with respect to the fuzzy membership function parameters. This idea has been previously suggested[33] and later applied to phase-locked loop filter design and motor current estimation[11,28]. The derivative formulas are shown in the Appendix.

2.1. *Gradient descent*

After the partial derivatives are computed as described above, the gradient descent rule can be used to update the independent variables from the kth iteration to the $(k+1)$st iteration as follows.

$$\begin{aligned}
c_{ij}(k+1) &= c_{ij}(k) - \eta \left.\frac{\partial E}{\partial c_{ij}}\right|_{c_{ij}(k)} \\
b_{ij}^{-}(k+1) &= b_{ij}^{-}(k) - \eta \left.\frac{\partial E}{\partial b_{ij}^{-}}\right|_{b_{ij}^{-}(k)} \\
b_{ij}^{+}(k+1) &= b_{ij}^{+}(k) - \eta \left.\frac{\partial E}{\partial b_{ij}^{+}}\right|_{b_{ij}^{+}(k)} \\
\gamma_i(k+1) &= \gamma_i(k) - \eta \left.\frac{\partial E}{\partial \gamma_i}\right|_{\gamma_i(k)} \\
\beta_i^{-}(k+1) &= \beta_i^{-}(k) - \eta \left.\frac{\partial E}{\partial \beta_i^{-}}\right|_{\beta_i^{-}(k)} \\
\beta_i^{+}(k+1) &= \beta_i^{+}(k) - \eta \left.\frac{\partial E}{\partial \beta_i^{+}}\right|_{\beta_i^{+}(k)}
\end{aligned} \tag{10}$$

where η is the gradient descent step size. More generally, a different value of η could be used in each of the six above equations, depending on the sensitivity of the error function to each of the independent variables. Usually some method is used with the gradient descent algorithm to try to avoid convergence to a local minimum. For instance, after a local minimum is found the solution can be randomly perturbed and the gradient descent algorithm can be restarted in an attempt to find a better local minimum.

2.2. *Extended Kalman filtering*

Derivations of the extended Kalman filter are widely available in the literature[34,35]. In this section we briefly outline the algorithm and its application to fuzzy membership function optimization. In general, we will use lower-case letters to refer to scalars, bold-faced lower case letters to refer to column vectors, and upper case letters to refer to matrices. We use the convention that the derivative of an m-element vector $\mathbf{a}$ with respect to a p-element vector $\mathbf{b}$ is defined as

$$\frac{\partial \mathbf{a}}{\partial \mathbf{b}} = \begin{bmatrix} \frac{\partial a_1}{\partial b_1} & \cdots & \frac{\partial a_1}{\partial b_p} \\ \vdots & & \vdots \\ \frac{\partial a_m}{\partial b_1} & \cdots & \frac{\partial a_m}{\partial b_p} \end{bmatrix}. \tag{11}$$

Consider a nonlinear finite dimensional discrete time system of the form

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{f}(\mathbf{x}_n) + \mathbf{w}_n \\ \mathbf{d}_n &= \mathbf{h}(\mathbf{x}_n) + \mathbf{v}_n \end{aligned} \tag{12}$$

where the vector $\mathbf{x}_n$ is the state of the system at time n, $\mathbf{w}_n$ and $\mathbf{v}_n$ are noise, $\mathbf{d}_n$ is the observation vector, and $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are nonlinear vector functions of the state. The problem addressed by the extended Kalman filter is to find an estimate $\hat{\mathbf{x}}_{n+1}$ of $\mathbf{x}_{n+1}$ given $\{\mathbf{d}_0, \ldots, \mathbf{d}_n\}$. It can be shown that the desired estimate $\hat{\mathbf{x}}_n$ can be obtained by the recursive extended Kalman filter

$$\begin{aligned} F_n &= \left.\frac{\partial \mathbf{f}(x)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_n} \\ H_n &= \left.\frac{\partial \mathbf{h}(x)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_n} \\ K_n &= P_n H_n^T (R + H_n P_n H_n^T)^{-1} \\ \hat{\mathbf{x}}_n &= \mathbf{f}(\hat{\mathbf{x}}_{n-1}) + K_n[\mathbf{d}_{n-1} - \mathbf{h}(\hat{\mathbf{x}}_{n-1})] \\ P_{n+1} &= F_n(P_n - K_n H_n P_n)F_n^T + Q \end{aligned} \tag{13}$$

where Q and R are the covariance matrices of $\{\mathbf{w}_n\}$ and $\{\mathbf{v}_n\}$ respectively. It is assumed that $\{\mathbf{w}_n\}$ and $\{\mathbf{v}_n\}$ are independent zero-mean noise processes, although this assumption can be relaxed with modifications of the Kalman filter. K_n is known as the Kalman gain. In the case of a linear system it can be shown that P_n is the covariance matrix of the state estimation error, and the state estimate $\hat{\mathbf{x}}_{n+1}$ is optimal in the sense that it approaches the conditional mean $E[\mathbf{x}_{n+1} \mid (\mathbf{d}_0, \mathbf{d}_1, \cdots, \mathbf{d}_n)]$ for large n. For nonlinear systems the filter is not optimal and the estimates are only approximately conditional means.

We can view the optimization of fuzzy membership functions as a weighted least-squares minimization problem, where the error vector is the difference between the fuzzy system outputs and the target values for those outputs. Consider a fuzzy system that has L outputs. We use $\mathbf{d}_n$ to denote the target vector for the fuzzy system outputs at the the nth time step, and $\mathbf{h}(k)$ to denote the actual outputs at this time step at the kth iteration of the Kalman filter. In order to cast the membership function optimization problem in a form suitable for Kalman filtering, we let the membership function parameters constitute the state of a nonlinear system, and we let the output of the fuzzy system constitute the output of the nonlinear system to which the Kalman filter is applied.

We will consider a two-input, one-output fuzzy system. This restriction is made only for notational convenience and the results in this paper can be (conceptually) easily extended to an unlimited number of inputs and outputs. Consider a fuzzy system that has μ fuzzy sets for the first input, ν fuzzy sets for the second input, and κ fuzzy sets for the output. As before we denote the modal point and half-widths of the ith fuzzy membership function of the jth input by c_{ij}, b_{ij}^-, and b_{ij}^+ respectively. We denote the modal point and half-widths of the ith fuzzy membership function of the output by γ_i, β_i^-, and β_i^+ respectively. The state of the nonlinear system can then be represented as

$$\begin{aligned}\mathbf{x} = [\ & b_{11}^- \quad b_{11}^+ \quad c_{11} \quad \cdots \quad b_{\mu 1}^- \quad b_{\mu 1}^+ \quad c_{\mu 1} \\ & b_{12}^- \quad b_{12}^+ \quad c_{12} \quad \cdots \quad b_{\nu 2}^- \quad b_{\nu 2}^+ \quad c_{\nu 2} \\ & \beta_1^- \quad \beta_1^+ \quad \gamma_1 \quad \cdots \quad \beta_\kappa^- \quad \beta_\kappa^+ \quad \gamma_\kappa\ ]^T .\end{aligned} \tag{14}$$

The vector $\mathbf{x}$ thus consists of all of the fuzzy membership function parameters arranged in a column vector. The nonlinear system model to which the Kalman filter can be applied is

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{w}_n \\ \mathbf{d}_n &= \mathbf{h}(\mathbf{x}_n) + \mathbf{v}_n\end{aligned} \tag{15}$$

where $\mathbf{h}(\mathbf{x}_n)$ is the fuzzy system's nonlinear mapping from the membership function parameters to the single fuzzy system output, and $\mathbf{w}_n$ and $\mathbf{v}_n$ are artificially added noise processes. The addition of these noise processes is a commonly practiced technique in parameter estimation algorithms to increase the stability of the estimator[34,36]. Now we can apply the Kalman recursion (13). $\mathbf{f}(\cdot)$ is the identity mapping, $\mathbf{d}_n$ is the target output of the fuzzy system, and $\mathbf{h}(\hat{\mathbf{x}}_n)$ is the actual output of the fuzzy system given the current membership function parameters. H_n is the partial derivative of the fuzzy output with respect to the membership function parameters (which can be computed as described and referenced earlier in this paper), and F_n is the identity matrix.

The Q and R matrices are tuning parameters which can be considered as the covariance matrices of the artificial noise processes $\mathbf{w}_n$ and $\mathbf{v}_n$ respectively. The determination of Q and R is a difficult task that remains an open research problem[37]. However, some general guidelines can be given. Looking back at (12), we see that $\mathbf{w}_n$ is the noise process that affects the state vector and $\mathbf{v}_n$ is the noise process that affects the measurement. As we increase Q we tell the filter that the state is likely to change more at each time step. This results in a filter that is more responsive to changes in the measurement. As we increase R we tell the filter that our measurement is more noisy. This results in a filter that is less responsive to changes in the measurement.

2.3. *Computational savings*

In order to reduce the computational effort of the gradient descent iteration in Section 2.1, a pseudo-steady-state assumption can be made in (10) that

$$\begin{aligned} \left.\frac{\partial E}{\partial c_{ij}}\right|_{c_{ij}(k)} &\approx \left.\frac{\partial E}{\partial c_{ij}}\right|_{c_{ij}(0)} \\ \left.\frac{\partial E}{\partial b_{ij}^-}\right|_{b_{ij}^-(k)} &\approx \left.\frac{\partial E}{\partial b_{ij}^-}\right|_{b_{ij}^-(0)} \\ \left.\frac{\partial E}{\partial b_{ij}^+}\right|_{b_{ij}^+(k)} &\approx \left.\frac{\partial E}{\partial b_{ij}^+}\right|_{b_{ij}^+(0)} \\ \left.\frac{\partial E}{\partial \gamma_i}\right|_{\gamma_i(k)} &\approx \left.\frac{\partial E}{\partial \gamma_i}\right|_{\gamma_i(0)} \\ \left.\frac{\partial E}{\partial \beta_i^-}\right|_{\beta_i^-(k)} &\approx \left.\frac{\partial E}{\partial \beta_i^-}\right|_{\beta_i^-(0)} \\ \left.\frac{\partial E}{\partial \beta_i^+}\right|_{\beta_i^+(k)} &\approx \left.\frac{\partial E}{\partial \beta_i^+}\right|_{\beta_i^+(0)} . \end{aligned} \tag{16}$$

That is, if we assume that we begin the optimization process close to the optimal membership function values then we can assume that the gradients do not change much during the optimization process. That means we can calculate the partial derivatives only once (at the first iteration), which saves a lot of computational effort.

We can do something similar for the Kalman filter of Section 2.2. We assume in (13) that

$$H_n \approx H_0 = \left.\frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_0} . \tag{17}$$

So the calculation of the partial derivative matrix can be performed only once. This assumption is only valid if the partial derivative of the system output $\mathbf{h}(\cdot)$ with respect to the state estimate $\hat{\mathbf{x}}_n$ does not change much from iteration to iteration[35].

This technique is simply a tradeoff between computational effort and theoretical integrity. In practice it turns out that this tradeoff often results in only a small dropoff in peformance at a fraction of the computational cost.

3. Fuzzy system optimization with sum normal constraints

The optimization proposed in the previous section works well but results in membership functions that are not sum normal. This will be seen in the simulation results presented later in this paper. Sum normality is sometimes desirable in membership functions for several reasons as described in Section 1 of this paper.

At first glance it might be thought that sum normality could be imposed on gradient descent and Kalman filtering by simply optimizing the membership functions with respect to the modal points, and then using the sum normal condition to determine the half-widths. That is, we could optimize with respect to the modal points but not the half-widths. Then the sum-normal constraint could be used to determine the half-widths. This sounds feasible but it does not work either in principle or in practice. When the modal point derivatives are computed apart from the half-width derivatives, and then the half-widths are computed by some other method, the resultant fuzzy logic system does not perform well. This approach is like minimizing a multivariable function with respect to one parameter and then independently changing all the other parameters. The resultant function value will not be minimum and there is no reason to suppose it will even have moved in the right direction. If we independently change all the other parameters then the point at which we are located in function space has changed and our derivative calculation is no longer valid. This section shows that the optimization discussed in the previous section can be modified in a more rigorous way so that the resultant membership functions are optimal under the sum normality constraint.

As above we consider a two-input, one-output fuzzy logic system. The first input has μ fuzzy sets. We denote the modal points and half-widths of the fuzzy membership functions of the first input by c_{i1}, b_{i1}^-, and b_{i1}^+ $(i = 1, \ldots, \mu)$. If the membership functions for the first input are sum normal then the following equalities hold:

$$\begin{aligned}
c_{11} + b_{11}^+ &= c_{21} \\
c_{11} + b_{21}^- &= c_{21} \\
c_{21} + b_{21}^+ &= c_{31} \\
c_{21} + b_{31}^- &= c_{31} \\
\vdots\quad & \quad\vdots \\
c_{\mu-1,1} + b_{\mu-1,1}^+ &= c_{\mu 1} \\
c_{\mu-1,1} + b_{\mu 1}^- &= c_{\mu 1}.
\end{aligned} \tag{18}$$

We have a similar set of equalities for the second input. The fuzzy logic system has ν fuzzy sets for the second input. We denote the modal points and half-widths of the fuzzy membership functions of the second input by c_{i2}, b_{i2}^-, and b_{i2}^+ $(i = 1, \ldots, \nu)$. If the membership functions for the second input are sum normal then the following equalities hold:

$$\begin{aligned}
c_{12} + b_{12}^{+} &= c_{22} \\
c_{12} + b_{22}^{-} &= c_{22} \\
c_{22} + b_{22}^{+} &= c_{32} \\
c_{22} + b_{32}^{-} &= c_{32} \\
\vdots \quad & \quad \vdots \\
c_{\nu-1,2} + b_{\nu-1,2}^{+} &= c_{\nu 2} \\
c_{\nu-1,2} + b_{\nu 2}^{-} &= c_{\nu 2}.
\end{aligned} \tag{19}$$

Finally we have another set of equalities for the output. The fuzzy logic system has κ fuzzy sets for the output. We denote the modal points and half-widths of the fuzzy membership functions of the output by γ_i, β_i^-, and β_i^+ $(i = 1, \ldots, \kappa)$. If the membership functions for the output are sum normal then the following equalities hold:

$$\begin{aligned}
\gamma_1 + \beta_1^{+} &= \gamma_2 \\
\gamma_1 + \beta_2^{-} &= \gamma_2 \\
\gamma_2 + \beta_2^{+} &= \gamma_3 \\
\gamma_2 + \beta_3^{-} &= \gamma_3 \\
\vdots \quad & \quad \vdots \\
\gamma_{\kappa-1} + \beta_{\kappa-1}^{+} &= \gamma_\kappa \\
\gamma_{\kappa-1} + \beta_{\kappa}^{-} &= \gamma_\kappa.
\end{aligned} \tag{20}$$

Equalities (18)-(20) can be written in matrix form as

$$L\mathrm{x} = \mathbf{0} \tag{21}$$

where x is the vector in (14) and L is the block diagonal matrix

$$L = \begin{bmatrix} L_1 & 0 & 0 \\ 0 & L_2 & 0 \\ 0 & 0 & L_3 \end{bmatrix}. \tag{22}$$

The L_i matrices are derived from (18), (19), and (20) respectively. L_1 is a $2(\mu - 1) \times 3\mu$ matrix, L_2 is a $2(\nu - 1) \times 3\nu$ matrix, and L_3 is a $2(\kappa - 1) \times 3\kappa$ matrix. Each L_i matrix is of the form

$$L_i = \begin{bmatrix} M_1 & M_2 & 0_{2\times 3} & \cdots & 0_{2\times 3} \\ 0_{2\times 3} & M_1 & M_2 & \cdots & 0_{2\times 3} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0_{2\times 3} & \cdots & 0_{2\times 3} & M_1 & M_2 \end{bmatrix} \tag{23}$$

where $0_{2\times 3}$ is the 2×3 matrix containing all zeros, and the M_j matrices are given by

$$\begin{aligned} M_1 &= \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \\ M_2 &= \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}. \end{aligned} \tag{24}$$

Therefore, in order to optimize fuzzy membership functions but with the constraint that they remain sum normal, we can project the unconstrained solution onto the constraint surface defined by (21). If we use gradient descent then we want to project the solution at each gradient descent iteration given by (10) onto the constraint surface. If we use Kalman filtering then we want to project the solution at each Kalaman filter iteration given by (13) onto the constraint surface.

This projection problem has previously been explored for general parameter estimation and Kalman filtering problems[38,39]. Suppose that we have a parameter estimate $\hat{\mathbf{x}}$ such as that given by (10) or (13). We desire to find a related parameter estimate $\tilde{\mathbf{x}}$ that is "close" to $\hat{\mathbf{x}}$ in some sense but that satisfies a constraint like (21). That is, we want to find the solution to

$$\min_{\tilde{\mathbf{x}}}(\hat{\mathbf{x}} - \tilde{\mathbf{x}})^T W(\hat{\mathbf{x}} - \tilde{\mathbf{x}}) \text{ such that } L\tilde{\mathbf{x}} = \delta \tag{25}$$

where W is an arbitrary positive-definite weighting matrix and L is full rank. The solution to this problem is given by the following[38,39].

$$\tilde{\mathbf{x}} = \hat{\mathbf{x}} - W^{-1}L^T(LW^{-1}L^T)^{-1}(L\hat{\mathbf{x}} - \delta). \tag{26}$$

It can be see from (21) that in our case $\delta = \mathbf{0}$. It can be seen from (22)-(24) that L is full rank and thus satisfies the premise of (25). Therefore we can carry out a sum normal contrained fuzzy membership optimization algorithm using (10) for gradient descent or (13) for Kalman filtering, augmented with the projection formula (26).

4. Simulation results

In this section we illustrate the use of gradient descent and Kalman filter training for fuzzy membership function parameters, both with and without sum normal constraints. The application is a fuzzy automotive cruise control system[30] . An automobile's acceleration can be stated as a function of the external forces acting on the vehicle: engine force f_e (a function of the throttle position), drag force f_d (a function of velocity), and gravity-induced force f_g (a function of road grade). If we assume that the time constant of the engine is small relative to the time constant of the vehicle, we obtain

$$m\dot{v} = f_e(\theta) - f_d(v) - f_g$$

where m is the vehicle mass, v is the velocity, and θ is the throttle position. The external forces are given by

$$f_e(\theta) \quad = \quad f_i + \gamma\sqrt{\theta}$$

$$
\begin{aligned}
f_d(v) &= \alpha v^2 \text{sign}(v) \\
f_g &= mg\sin(\text{grade})
\end{aligned}
$$

where γ, α, g, and f_i are constants. We will use the values $m = 1000$ kg, $\gamma = 12500$ Newtons, and $\alpha = 4$ N / $(\text{m/s})^2$. f_i is the engine idle force, which we will assume to be 1000 N, and g is the acceleration due to gravity, which is about 9.81 m/s^2.

A 2-input, 1-output fuzzy cruise control can be designed by defining *error* as the reference speed minus the measured speed, and implementing rules such as the following: "If the *error* is small positive, and the *change in error* is zero, then change the throttle position by a *small positive* amount." Another rule might be, "If the *error* is zero, and the *change in error* is large positive, then change the throttle position by a *small positive* amount." A rule base was defined with five membership functions each for the two inputs and the output. So μ, ν, and κ in (14) are all equal to five. The rule base is shown in Table 1.

Table 1. Rule Base for Fuzzy Cruise Controller. NL = Negative Large, NS = Negative Small, Z = Zero, PS = Positive Small, PL = Positive Large.

		Error				
		NL	NS	Z	PS	PL
	NL	NL	NL	NS	NS	NS
Error	NS	NL	NS	Z	Z	Z
Change	Z	NL	NS	Z	PS	PL
	PS	Z	Z	Z	PS	PL
	PL	PS	PS	PS	PL	PL

Since there are a total of three fuzzy variables (two inputs and one output), and each fuzzy variable has five membership functions, the fuzzy cruise control has a total of 15 membership functions. Each membership function is constrained to be triangular so each membership function has three parameters (a modal point and two half-widths). The fuzzy cruise control therefore has a total of 45 parameters to be determined.

Gradient descent can be used to optimize the fuzzy cruise control with respect to these 45 parameters. For the Kalman filter, these 45 parameters are arranged in a vector as shown in (14) and hence comprise the 45-element state of the Kalman filter. If we desire to maintain sum normality in our optimized membership functions, we use the projection equation (26). In this paper we use $W = I$ so that each membership function parameter is given an equal weight in the projection equation. The matrix L in (26) is a 24×45 matrix.

The error function (9) was defined as the reference speed minus the vehicle speed. The fuzzy cruise control was simulated using Matlab for 15 s with a controller update period of 0.25 s, so N in (9) was equal to 60. The weighting function g_n in (9) was set to n/N to give a greater weight to errors at the end of the training interval; in other words, we were more interested in decreasing settling time than in decreasing overshoot.

Gradient descent and Kalman filtering (both with and without sum normal constraints) were implemented in Matlab to optimize the membership functions of the controller inputs and output. The pseudo-steady-state formulation as described in Section 2.3 was used to decrease training time. We tuned the gradient descent and Kalman filter parameters manually for the best convergence results. For gradient descent we obtained $\eta = 10$ (unconstrained) and $\eta = 30$ (constrained). For Kalman filtering we obtained $P_0 = 1E6$, $Q = 4E3$, and $R = 1$ (unconstrained) and $P_0 = 1E18$, $Q = 4E3$, and $R = 1E-8$ (constrained). The training setup consisted of the cruise control operating in steady state on a flat road with a sudden 10% increase in the road grade at time = 0. The reference speed of the cruise control was set at 40 m/s so the objective of the controller was to maintain a 40 m/s velocity even after encountering a sudden 10% increase in road grade.

Figure 1 depicts the progress of training with gradient descent and Kalman filtering (both with and without sum normal constraints). The figure indicates that the Kalman filter methods converge to better solutions than the gradient descent methods. As expected, the unconstrained algorithms converge more quickly and to better solutions than the constrained algorithms.

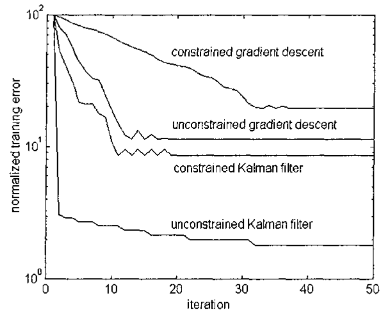

Fig. 1. Training Progress.

The computational requirements of the gradient descent and Kalman filter methods are about the same. Although the Kalman filter equations are more complex than the gradient descent equations, the matrix inversion in (13) involves only a 1×1 matrix (since the dynamic system has only one output). The optimization algorithms were run on a 233 MHz Pentium PC. The computational effort for the two methods was about 7 s at the first iteration for the partial derivative calculations. Each iteration after the first required only about 1.7 s per iteration (since the derivative calculations were skipped). If we had not used the pseudo-steady-state approximation described in Section 2.3 then each method would have required 7 s per iteration. The CPU time required by the optimization algorithms will be highly dependent on the implementation details. The computational effort given here should be used only for relative comparisons.

Now we move from the training scenario to the test scenario. Figure 2 shows a test case comparing the default fuzzy cruise controller with the cruise controller that was optimized *without* sum normal constraints. In this test scenario the automobile encountered a sudden 8% increase in the road grade at time = 0. The optimized cruise controllers were the same as those that were trained with a 10% increase in the road grade.

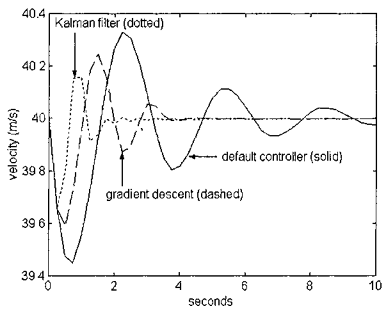

Fig. 2. Test Data Before and After Unconstrained Optimization.

Figure 2 illustrates the cruise controller performance in a scenario other than that for which it was trained. The reference velocity was fixed at 40 m/s so the cruise control attempted to maintain that velocity in the presence of the increased road grade. The reduction in settling time is noticeable for the optimized cruise control. This reflects our choice of g_n in (9) as described above. The optimized membership functions are not sum normal in this case since we did not use the sum normal constraints.

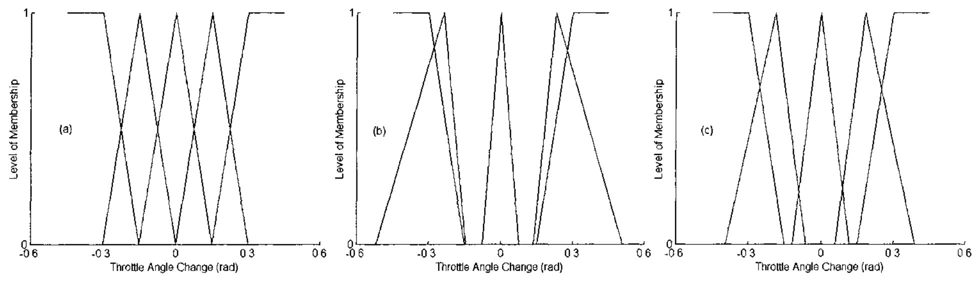

Fig. 3. Output Membership functions. (a) Default; (b) Optimized via Unconstrained Kalman Filtering; (c) Optimized via Unconstrained Gradient Descent.

Figure 3 shows the original membership functions and unconstrained optimized membership functions for the output. (The input membership functions are not shown because they changed only slightly during the optimization process.) The optimized membership functions work well as seen from Figure 2, but they are clearly not sum normal, which may be undesirable. In fact, the optimized membership functions do not even cover the entire range of crisp values. This is nonintuitive, but there is nothing problematic about this from a mathematical point of view. This just means that the crisp output of the fuzzy system will never be equal to the uncovered values.

Figure 4 shows a comparison of the default fuzzy cruise controller with the cruise controller that was optimized *with* sum normal constraints (for the same test case as described above). As above, the reduction in settling time is noticeable for the optimized cruise control. However, a comparison with Figure 2 shows that (as expected) the constrained controller does not perform as well as the unconstrained controller. As seen from Figure 5, the optimized membership functions are indeed sum normal. Comparison with Figure 3 shows what a drastic difference sum normal constraints can make in the resultant membership functions.

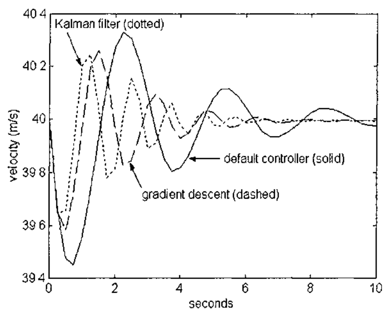

Fig. 4. Test Data Before and After Constrained Optimization.

Table 2 compares the cruise controller's normalized training error as defined by (9) for various membership functions. The table also shows the improvement that is obtained when the algorithm is run without the pseudo-steady-state approximation.

Table 2. Test Case Error Comparison. The initial fuzzy controller in all cases had a normalized training error of 100.

	Normalized Training Error	
Optimization Method	Steady State	Non-Steady-State
Unconstrained Gradient Descent	11.33	11.15
Constrained Gradient Descent	19.46	19.05
Unconstrained Kalman Filtering	1.75	1.48
Constrained Kalman Filtering	8.59	7.87

It is seen that the removal of the pseudo-steady-state approximation results in a decrease of the error function value in all cases, but only by a small amount. In addition, the unconstrained optimization methods result in better performance than the constrained methods. We can also see that Kalman filtering results in better performance than gradient descent. However, this should not be taken as an inviolable law. The performance of gradient descent and Kalman filtering both depend strongly on the initial conditions of the membership functions and the tuning parameters of the optimization algorithm. For gradient descent we need to choose an appropriate value of the scalar η in (10), and it may be best to use different values of η for different parameters. For Kalman filtering we need to choose appropriate values of of the matrices P_0, Q, and R in (13). In general we can get better performance from Kalman filtering simply because we have more parameters to tune. However, gradient descent may be preferred in some instances because its application is simpler and more straightforward. The Matlab code that was used to generate these results can be downloaded from the internet at `academic.csuohio.edu/simond/fuzzyopt/`. These results can then be reproduced by running those Matlab m-files.

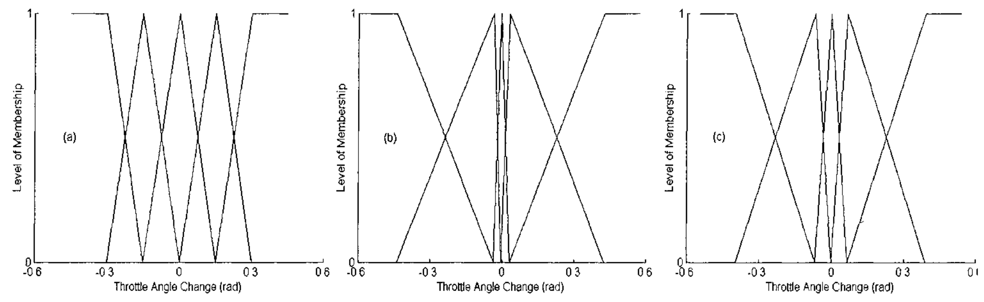

Fig. 5. Output Membership functions. (a) Default; (b) Optimized via Constrained Kalman Filtering; (c) Optimized via Constrained Gradient Descent.

5. Conclusion

We have shown that the membership functions of a fuzzy controller can be optimized via gradient descent and Kalman filtering. In general, these optimization methods result in membership functions that are not sum normal; that is, the membership function values do not add up to one at each point in the domain. We have extended the gradient descent and Kalman filtering algorithms to ensure that the resulting membership functions are sum normal. This results in a fuzzy controller with worse performance that the unconstrained membership functions, but sum normality may be desirable for several reasons (as discussed in Section 1).

The optimization methods presented in this paper were demonstrated on a simulated fuzzy automotive cruise controller. As expected, unconstrained optimization resulted in better performance than constrained optimization. But unconstrained optimization also resulted in non-normal membership functions while constrained optimization resulted in sum normal membership functions. In general, Kalman filtering resulted in better performance than gradient descent. This is to be expected because the Kalman filter has more tunable optimization parameters.

Gradient descent and Kalman filtering are both sensitive to the values of their tunable parameters and to initial conditions. They should be viewed as "fine-tuning" methods rather than as global optimization methods. Initial optimization should be conducted with a more global method, such as one of the derivative-free methods discussed in Section 1. After the global optimization method finds the general neighborhood of the optimal membership function parameters, gradient descent or Kalman filtering can be used to fine-tune the results. Further work in this area could focus on the convergence properties of the Kalman filter in this application, the effect of the tunable parameters of the Kalman filter, the optimization of fuzzy systems with non-triangular membership functions, or the extension of this work to other derivative-based optimization schemes.

References

1. O. Cordon, F. Herrera, and P. Villar, "Analysis and Guidelines to Obtain a Good Uniform Fuzzy Partition Granularity for Fuzzy Rule-Based Systems Using Simulated Annealing," *International Journal of Approximate Reasoning* **25** (2000) 187-216.
2. O. Cordon, F. Herrera, L. Magdalena, and P. Villar, "A Genetic Learning Process for the Scaling Factors, Granularity and Contexts of the Fuzzy Rule-Based System Data Base," *Information Sciences* **136** (2001) 85-107.
3. G. Procyk and E. Mamdani, "A Linguistic Self-Organizing Process Controller," *Automatica* **15** (1979) 15-30.
4. P. Xian-Tu, "Generating Rules for Fuzzy Logic Controllers by Functions," *Fuzzy Sets and Systems* **36** (1990) 83-89.
5. R. Gudwin, F. Gomide, and W. Pedrycz, "Context Adaptation in Fuzzy Processing and Genetic Algorithms," *International Journal of Intelligent Systems* **13** (1998) 929-948.
6. L. Magdalena and F. Monasterio-Huelin, "Fuzzy Logic Controller with Learning through the Evolution of its Knowledge Base," *International Journal of Approximate Reasoning* **16** (1997) 335-358.
7. L. Magdalena, "Adapting the Gain of an FLC with Genetic Algorithms," *International Journal of Approximate Reasoning* **17** (1997) 327-349.
8. W. Pedrycz, R. Gudwin, and F. Gomide, "Nonlinear Context Adaptation in the Calibration of Fuzzy Sets," *Fuzzy Sets and Systems* **88** (1997) 91-97.
9. R. Palm, "Scaling of Fuzzy Controllers Using the Cross-Correlation," *IEEE Transactions on Fuzzy Systems* **3** (1995) 116-123.
10. W. Daugherity, B. Rathakrishnan, and J. Yen, "Performance Evaluation of a Self-Tuning Fuzzy Controller," *IEEE International Conference on Fuzzy Systems*, 1992, pp. 389-397.
11. D. Simon and H. El-Sherief, "Fuzzy Logic for Digital Phase-Locked Loop Filter Design," *IEEE Transactions on Fuzzy Systems* **3** (1995) 211-218.
12. H. Surmann, "Genetic Optimization of a Fuzzy System for Charging Batteries," *IEEE*

Transactions on Industrial Electronics **43** (1996) 541-548.
13. W. Barada and H. Singh, "Generating Optimal Adaptive Fuzzy-Neural Models of Dynamical Systems with Applications to Control," *IEEE Transactions on Systems, Man, and Cybernetics - Part C* **28** (1998) 371-391.
14. M. Figueiredo and F. Gomide, "Design of Fuzzy Systems Using Neurofuzzy Networks," *IEEE Transactions on Neural Networks* **10** (1999) 815-827.
15. J. Goddard, R. Parrazales, I. Lopez, and A. de Luca, "Rule Learning in Fuzzy Systems Using Evolutionary Programs," *IEEE Midwest Symposium on Circuits and Systems*, Ames, Iowa, 1996, pp. 703-709.
16. S. Smith and D. Comer, "Automated Calibration of a Fuzzy Logic Controller Using a Cell State Space Algorithm," *IEEE Control Systems Magazine* **11** (1991) 18-28.
17. R. Wu and S. Chen, "A New Method for Constructing Membership Functions and Fuzzy Rules from Training Examples," *IEEE Transactions on Systems, Man, and Cybernetics - Part B* **29** (1999) 25-40.
18. C. Tao and J. Taur, "Design of Fuzzy Controllers with Adaptive Rule Insertion," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics* **29** (1999) 389-397.
19. B. Demaya, R. Palm, S. Boverie, and A. Titli, "Multilevel Qualitative and Numerical Optimization of Fuzzy Controller," *IEEE International Conference on Fuzzy Systems*, Yokohama, Japan, 1995, pp. 1149-1154.
20. L. Wang and J. Mendel, "Back-propagation of Fuzzy Systems as Nonlinear Dynamic System Identifiers," *IEEE International Conference on Fuzzy Systems*, San Diego, California, 1992, pp. 1409-1418.
21. D. Simon, "Training Fuzzy Systems with the Extended Kalman Filter," *Fuzzy Sets and Systems*, in print, 2002.
22. J. Deskur, R. Muszynski, and D. Sarnowski, "Tuning and Investigation of Combined Fuzzy Controller," *International Conference on Power Electronics and Variable Speed Drives*, London, 1998, pp. 353-357.
23. M. Jacomet, A. Stahel, and R. Walti, "On-Line Optimization of Fuzzy Systems," *Information Sciences* **98** (1997) 301-313.
24. M. Sugeno and K. Tanaka, "Successive Identification of a Fuzzy Model and Its Applications to Prediction of a Complex System," *Fuzzy Sets and Systems* **42** (1991) 315-334.
25. L. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis* (Prentice Hall, Upper Saddle River, New Jersey, 1994).
26. E. Nakamura and N. Kehtarnavez, "Optimization of Fuzzy Membership Function Parameters," *IEEE International Conference on Systems, Man and Cybernetics*, Vancouver, Canada, 1995, pp. 1-6.
27. D. Simon, "Fuzzy Membership Optimization via the Extended Kalman Filter," *North American Fuzzy Information Processing Society Conference*, Atlanta, Georgia, 2000, pp. 311-315.
28. D. Simon, "Design and Rule Base Reduction of a Fuzzy Filter for the Estimation of Motor Currents," *International Journal of Approximate Reasoning* **25** (2000) 145-167.
29. T. Terano, K. Asai, and M. Sugeno, *Fuzzy Systems Theory and Its Applications* (Academic Press, San Diego, California, 1992).
30. J. Yen and R. Langari, *Fuzzy Logic: Intelligence, Control, and Information*(Prentice Hall, Upper Saddle River, New Jerseay, 1999).
31. Y. Yam, P. Baranyi, and C. Yang, "Reduction of Fuzzy Rule Base Via Singular Value Decomposition," *IEEE Transactions on Fuzzy Systems* **7** (1999) 120-132.
32. B. Kosko, *Neural Networks and Fuzzy Systems*, (Prentice Hall, Upper Saddle River, New Jersey, 1992).
33. F. Guély and P. Siarry, "Gradient Descent Method for Optimizing Various Fuzzy Rule

Bases," *IEEE International Conference on Fuzzy Systems*, San Francisco, California, 1993, pp. 1241-1246.
34. B. Anderson and J. Moore, *Optimal Filtering* (Prentice Hall, Upper Saddle River, New Jersey, 1979).
35. A. ·Gelb, *Applied Optimal Estimation* (MIT Press, Cambridge, Massachusetts, 1974).
36. G. Puskorius and L. Feldkamp, "Neurocontrol of Nonlinear Dynamical Systems with Kalman Filter Trained Recurrent Networks," *IEEE Transactions on Neural Networks* **5** (1994) 279-297.
37. G. Leu and R. Baratti, "An extended Kalman filtering approach with a criterion to set its tuning parameters; application to a catalytic reactor," *Computers and Chemical Engineering* **23** (2000) 1839-1849.
38. T. Chia, P. Chow, and H. Chizek, "Recursive parameter identification of constrained systems: An application to electrically stimulated muscle," *IEEE Transactions on Biomedical Engineering* **38** (1991) 429-442.
39. D. Simon and T. Chia, "Kalman Filtering with State Equality Constraints," *IEEE Transactions on Aerospace and Electronic Systems* **39** (2002) 128-136.

Acknowledgements

The comments of the referees were significant in correcting mistakes and strengthening this paper from its original form.

Appendix A

This appendix contains the formulas for the derivatives of the error function with respect to the modal points and half-widths of the input and output fuzzy sets. These derivatives can be obtained by straightforward calculus and algebra.

Input Modal Points

From (1)-(9) we obtain

$$\frac{\partial E}{\partial c_{ij}} = \frac{1}{N}\sum_{n=1}^{N} E_n \frac{\partial \hat{y}_n}{\partial c_{ij}} \tag{A.1}$$

$$\frac{\partial \hat{y}_n}{\partial c_{ij}} = \sum_{k=1}^{M} \frac{\partial \hat{y}_n}{\partial w_k}\frac{\partial w_k}{\partial c_{ij}} \tag{A.2}$$

$$\begin{aligned}\frac{\partial \hat{y}_n}{\partial w_k} &= \frac{\Gamma_k J_k \sum_{j=1}^{M} w_j J_j - J_k \sum_{j=1}^{M} w_j \Gamma_j J_j}{\left(\sum_{j=1}^{M} w_j J_j\right)^2} \\ &= \frac{J_k(\Gamma_k - \hat{y}_n)}{\sum_{j=1}^{M} w_j J_j}.\end{aligned} \tag{A.3}$$

Next we define $r_{i1k} = 1$ if $z_1 \in$ fuzzy set i is a premise of the kth rule and $w_k = f_{i1}(z_1)$, and $r_{i1k} = 0$ otherwise. In other words, $r_{i1k} = 1$ if z_1 determines the activation level of the kth rule because of its membership in the ith fuzzy set. Similarly, $r_{i2k} = 1$ if $z_2 \in$ fuzzy set i is a premise of the kth rule and $w_k = f_{i2}(z_2)$, and $r_{i2k} = 0$ otherwise. With these definitions we can determine

$$\frac{\partial w_k}{\partial c_{i1}} = \frac{\partial f_{i1}(z_1)}{\partial c_{i1}} r_{i1k} \tag{A.4}$$

$$\frac{\partial w_k}{\partial c_{i2}} = \frac{\partial f_{i2}(z_2)}{\partial c_{i2}} r_{i2k}. \tag{A.5}$$

The partials of the input fuzzy sets with respect to their modal points are given as

$$\frac{\partial f_{i1}(z_1)}{\partial c_{i1}} = \begin{cases} -1/b_{i1}^- & \text{if } c_{i1} - b_{i1}^- \le z_1 \le c_{i1} \\ 1/b_{i1}^+ & \text{if } c_{i1} \le z_1 \le c_{i1} + b_{i1}^+ \\ 0 & \text{otherwise} \end{cases} \tag{A.6}$$

$$\frac{\partial f_{i2}(z_2)}{\partial c_{i2}} = \begin{cases} -1/b_{i2}^- & \text{if } c_{i2} - b_{i2}^- \le z_2 \le c_{i2} \\ 1/b_{i2}^+ & \text{if } c_{i2} \le z_2 \le c_{i2} + b_{i2}^+ \\ 0 & \text{otherwise} \end{cases} . \tag{A.7}$$

This completes the presentation of the derivatives of the error function with respect to the modal points of the input fuzzy sets.

Input Half-Widths

Again using (1)-(9) we obtain

$$\frac{\partial E}{\partial b_{ij}^-} = \frac{1}{N}\sum_{n=1}^{N} E_n \frac{\partial \hat{y}_n}{\partial b_{ij}^-} \tag{A.8}$$

$$\frac{\partial \hat{y}_n}{\partial b_{ij}^-} = \sum_{k=1}^{M} \frac{\partial \hat{y}_n}{\partial w_k}\frac{\partial w_k}{\partial b_{ij}^-} \tag{A.9}$$

$$\frac{\partial w_k}{\partial b_{ij}^-} = \frac{\partial f_{ij}(z_j)}{\partial b_{ij}^-} r_{ijk} \tag{A.10}$$

$$\frac{\partial f_{ij}(z_j)}{\partial b_{ij}^-} = \begin{cases} \frac{1-f_{ij}(z_j)}{b_{ij}^-} & \text{if } c_{ij} - b_{ij}^- \le z_j \le c_{ij} \\ 0 & \text{otherwise} \end{cases} . \tag{A.11}$$

These formulas give the partials of the error function with respect to the lower half-widths of the input fuzzy sets. Similarly, we obtain

$$\frac{\partial E}{\partial b_{ij}^+} = \frac{1}{N}\sum_{n=1}^{N} E_n \frac{\partial \hat{y}_n}{\partial b_{ij}^+} \tag{A.12}$$

$$\frac{\partial \hat{y}_n}{\partial b_{ij}^+} = \sum_{k=1}^{M} \frac{\partial \hat{y}_n}{\partial w_k}\frac{\partial w_k}{\partial b_{ij}^+} \tag{A.13}$$

$$\frac{\partial w_k}{\partial b_{ij}^+} = \frac{\partial f_{ij}(z_j)}{\partial b_{ij}^+} r_{ijk} \tag{A.14}$$

$$\frac{\partial f_{ij}(z_j)}{\partial b_{ij}^+} = \begin{cases} \frac{1-f_{ij}(z_j)}{b_{ij}^+} & \text{if } c_{ij} \le z_j \le c_{ij} + b_{ij}^+ \\ 0 & \text{otherwise} \end{cases} . \tag{A.15}$$

These formulas give the partials of the error function with respect to the upper half-widths of the input fuzzy sets.

Output Modal Points

The partials of the error function with respect to the modal points of the output fuzzy sets are given as

$$\frac{\partial E}{\partial \gamma_k} = \frac{1}{M}\sum_{n=1}^{M} E_n \frac{\partial \hat{y}_n}{\partial \gamma_k} \tag{A.16}$$

$$\begin{aligned}\frac{\partial \hat{y}_n}{\partial \gamma_k} &= \frac{\partial \hat{y}_n}{\partial \Gamma_k}\frac{\partial \Gamma_k}{\partial \gamma_k} + \frac{\partial \hat{y}_n}{\partial J_k}\frac{\partial J_k}{\partial \gamma_k} \\ &= \frac{w_k J_k}{\sum_{j=1}^{M} w_j J_j}. \end{aligned} \tag{A.17}$$

Output Half-Widths

The partials of the error function with respect to the upper half-widths of the output fuzzy sets are obtained from the following formulas.

$$\frac{\partial E}{\partial \beta_k^+} = \frac{1}{M}\sum_{n=1}^{M} E_n \frac{\partial \hat{y}_n}{\partial \beta_k^+} \tag{A.18}$$

$$\frac{\partial \hat{y}_n}{\partial \beta_k^+} = \frac{\partial \hat{y}_n}{\partial \Gamma_k}\frac{\partial \Gamma_k}{\partial \beta_k^+} + \frac{\partial \hat{y}_n}{\partial J_k}\frac{\partial J_k}{\partial \beta_k^+} \tag{A.19}$$

$$\frac{\partial \hat{y}_n}{\partial \Gamma_k} = \frac{w_k J_k}{\sum_{j=1}^{M} w_j J_j} \tag{A.20}$$

$$\frac{\partial \Gamma_k}{\partial \beta_k^+} = \frac{1}{3(\beta_k^+ + \beta_k^-)} \tag{A.21}$$

$$\begin{aligned}\frac{\partial \hat{y}_n}{\partial J_k} &= \frac{w_k \Gamma_k \sum_{j=1}^{M} w_j J_j - w_k \sum_{j=1}^{M} w_j \Gamma_j J_j}{\left(\sum_{j=1}^{M} w_j J_j\right)^2} \\ &= \frac{w_k(\Gamma_k - \hat{y}_n)}{\sum_{j=1}^{M} w_j J_j} \end{aligned} \tag{A.22}$$

$$\frac{\partial J_k}{\partial \beta_k^+} = \frac{1}{2}. \tag{A.23}$$

Substituting (A.20)-(A.23) into (A.19) results in

$$\frac{\partial \hat{y}_n}{\partial \beta_k^+} = \frac{w_k}{\sum_{j=1}^{M} w_j J_j}\left[\frac{J_k}{3(\beta_k^+ + \beta_k^-)} + \frac{\Gamma_k - \hat{y}_n}{2}\right]. \tag{A.24}$$

This equation is then substituted into (A.18) to obtain the partials of the error function with respect to the upper half-widths of the output fuzzy sets. Similarly, we obtain the partials with respect to the lower half-widths as

$$\frac{\partial E}{\partial \beta_k^-} = \frac{1}{M}\sum_{n=1}^{M} E_n \frac{\partial \hat{y}_n}{\partial \beta_k^-} \tag{A.25}$$

$$\frac{\partial \hat{y}_n}{\partial \beta_k^-} = \frac{\partial \hat{y}_n}{\partial \Gamma_k}\frac{\partial \Gamma_k}{\partial \beta_k^-} + \frac{\partial \hat{y}_n}{\partial J_k}\frac{\partial J_k}{\partial \beta_k^-}. \tag{A.26}$$

The four terms on the right side of the above equation are given by (A.20), (A.22), and the following two equations.

$$\frac{\partial \Gamma_k}{\partial \beta_k^-} = \frac{-1}{3(\beta_k^+ + \beta_k^-)} \tag{A.27}$$

$$\frac{\partial J_k}{\partial \beta_k^-} = \frac{1}{2}. \tag{A.28}$$

Substituting these equations in (A.26) results in

$$\frac{\partial \hat{y}_n}{\partial \beta_k^-} = \frac{w_k}{\sum_{j=1}^{M} w_j J_j}\left[\frac{-J_k}{3(\beta_k^+ + \beta_k^-)} + \frac{\Gamma_k - \hat{y}_n}{2}\right]. \tag{A.29}$$

This equation is then substituted into (A.25) to obtain the partials of the error function with respect to the lower half-widths of the output fuzzy sets.