H-infinity Estimation for Fuzzy Membership Function Optimization

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

Follow this and additional works at: http://engagedscholarship.csuohio.edu/enece_facpub

Part of the Computer Engineering Commons

Publisher's Statement
NOTICE: this is the author's version of a work that was accepted for publication in International Journal of Approximate Reasoning. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in International Journal of Approximate Reasoning, 40, 3, (11-01-2005); 10.1016/j.ijar.2005.04.002

Original Citation

Repository Citation
http://engagedscholarship.csuohio.edu/enece_facpub/60

This Article is brought to you for free and open access by the Electrical Engineering & Computer Science Department at EngagedScholarship@CSU. It has been accepted for inclusion in Electrical Engineering & Computer Science Faculty Publications by an authorized administrator of EngagedScholarship@CSU. For more information, please contact library.es@csuohio.edu.
$H_\infty$ estimation for fuzzy membership function optimization

Dan Simon *

Cleveland State University, Department of Electrical Engineering, Stilwell Hall Room 332,
2121 Euclid Avenue, 1960 E. 24th Street, Cleveland, OH 44115-2214, United States.

* Tel.: +1 216 687 5407; fax: +1 216 687 5405.
E-mail address: d.j.simon@csuohio.edu
1. Introduction

In this paper we use the term *fuzzy parameters* to refer to the parameters that define the membership functions of a fuzzy logic system. For instance, if we are using triangular membership functions, then the *fuzzy parameters* would be the centers and half-widths of the triangles.

Researchers have used many different methods over the past decade to optimize fuzzy membership functions. The methods can be broadly divided into two types: those that explicitly use the derivatives of the fuzzy system’s performance with respect to the fuzzy parameters, and those that do not use these derivatives. Derivative-free methods include genetic algorithms [1–3], neural networks [4,5], evolutionary programming [6], geometric methods [7], fuzzy equivalence relations [8], and heuristic methods [9]. Derivative-based methods include gradient descent [2,10], Kalman filtering [11], the simplex method [12,13], least squares [14,15], back-propagation [16], and other numerical techniques [17].

Derivative-free methods can be desirable in that they do not require the explicit derivatives of the objective function with respect to the fuzzy 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 an $H_\infty$ filter for fuzzy membership function optimization. This is similar to Kalman filtering methods [11,18]. The use of $H_\infty$ filtering is motivated by the fact that it is often more robust than Kalman filtering in the presence of system noise, modeling errors, and nonlinearities [19]. The application of $H_\infty$ filtering to fuzzy membership function optimization does not involve high levels of noise or modeling errors, but it does involve high levels of nonlinearity. This indicates that $H_\infty$ filtering may be more robust than Kalman filtering for this application.

A straightforward application of $H_\infty$ filtering is effective for fuzzy membership function optimization but it results 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 [20]. Second, sum normality is desired by many fuzzy system engineers for its aesthetic and intuitive appeal [21]. Third, some rule base reduction algorithms guarantee that a sum normal set of membership functions will remain sum normal even after rule base reduction [22]. Fourth, 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 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.) We therefore modify the $H_\infty$ filter used
in this paper in such a way that sum normality is guaranteed in the resulting fuzzy membership functions.

The next section presents the use of $H_\infty$ filtering for membership function optimization. We then modify this method to reduce the computational requirements, and we also modify the method to guarantee sum normality in the resulting membership functions. Section 3 contains some simulation results of a fuzzy automotive cruise controller, including comparisons with Kalman filter based optimization. Section 4 contains a summary and concluding remarks.

2. Fuzzy system optimization via $H_\infty$ filtering

In this paper we assume that our fuzzy system uses correlation product inference [23], which will be described later in this section. We further assume that fitness 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 $i$th fuzzy membership function of the $j$th 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 $j$th crisp input $z_j$ in its $i$th 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 fuzzy system. The consequent of the $j$th rule is a triangular fuzzy set with modal point $\gamma_j$, lower half-width $\beta^-_j$, and upper half-width $\beta^+_j$. That is, the fuzzy set of the consequent of the $j$th 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 $j$th 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 $j$th rule is $w_{jk}$, which is given as
The fuzzy output when \( z_1 \in (\text{fuzzy set } i) \) and \( z_2 \in (\text{fuzzy set } k) \) is given as
\[
\tilde{m}_j(y) = w_j m_j(y).
\]

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.
\[
m(y) = \sum_{j=1}^{M} \tilde{m}_j(y).
\]

The sum aggregation represented by the above equation could result in a membership function value \( m(y) > 1 \) if the membership functions \( \tilde{m}_j(y) \) are not sum normal. This is illustrated later in Fig. 3(b) and (c). A membership function value greater than one is nonintuitive, which is part of the motivation for constrained membership function optimization.

The fuzzy output is mapped to a crisp number \( \hat{y} \) using centroid defuzzification [23].
\[
\hat{y} = \frac{\sum_{j=1}^{M} w_j \Gamma_j J_j}{\sum_{j=1}^{M} w_j J_j}. \tag{6}
\]

\( \Gamma_j \) and \( J_j \) are the centroid and area of the \( j \)th output fuzzy membership function. The centroid of \( m_j(y) \), the \( j \)th 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
\[
\Gamma_j = \frac{\beta_j^y (3\gamma_j + \beta_j^y) + \beta_j^y (3\gamma_j - \beta_j^y)}{3(\beta_j^y + \beta_j^y)}. \tag{8}
\]

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
\[
E = \frac{1}{2N} \sum_{n=1}^{N} g_n E_n^2, \tag{9}
\]
\[
E_n = \hat{y}_n - y_n,
\]
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 weighting function. The role of \( g_n \) will be illustrated in the example of Section 3. We can minimize \( E \) by using the partial derivatives of \( E \) with respect to the modal points and half-widths of the function.
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 was perhaps first suggested in [24] and was later applied to
fuzzy phase-locked loop filter design and motor current estimation [2,18]. The deriv-
ative formulas are given in [25].

2.1. \( H_\infty \) filtering

Various derivations of the \( H_\infty \) filter are available in the literature [26,27]. In this
section we briefly outline the version derived in [28] and show how it can be applied
to fuzzy membership function optimization. We use the convention that the deriva-
tive of an \( m \)-element vector \( a \) with respect to a \( p \)-element vector \( b \) is

\[
\mathbf{\frac{\partial a}{\partial b}} = \begin{bmatrix}
\frac{\partial a_1}{\partial b_1} & \cdots & \frac{\partial a_1}{\partial b_p} \\
\vdots & \ddots & \vdots \\
\frac{\partial a_m}{\partial b_1} & \cdots & \frac{\partial a_m}{\partial b_p}
\end{bmatrix}.
\]  (10)

Consider a nonlinear time-invariant finite dimensional discrete time system of the
form

\[
\begin{align*}
x_{n+1} &= f(x_n) + Bw_n + \delta_n, \\
d_n &= h(x_n) + v_n,
\end{align*}
\]  (11)

where the vector \( x_n \) is the state of the system at time \( n \), \( w_n \) and \( v_n \) are white noise, \( \delta_n \) is
an arbitrary noise sequence, \( d_n \) is the observation vector, and \( f(\cdot) \) and \( h(\cdot) \) are nonli-
ear vector functions of the state. The problem addressed by the \( H_\infty \) filter is to find
an estimate \( \hat{x}_{n+1} \) of \( x_{n+1} \) given \( \{d_0, \ldots, d_n\} \). It is assumed that \( \{w_n\} \) and \( \{v_n\} \) are inde-
pendent unity variance noise process, but the noise sequence \( \{\delta_n\} \) is arbitrary. We
define the augmented noise vector and the estimation error as follows:

\[
\begin{align*}
e_n &= \begin{bmatrix} w_n^T \\
\hat{w}_n^T \end{bmatrix}^T, \\
\hat{x}_n &= x_n - \hat{x}_n.
\end{align*}
\]  (12)

The problem solved by the \( H_\infty \) filter is to find an estimate \( \hat{x}_n \) such that the infinity
norm of the transfer function from the augmented noise vector \( e \) to the estimation
error \( \hat{x} \) is bounded by a user-defined quantity \( \gamma \).

\[
\|G_{ew}\|_\infty < \gamma.
\]  (13)

This means that the maximum steady-state gain from \( e \) to \( \hat{x} \) is less than \( \gamma \). It can be
shown [28] that the desired estimate \( \hat{x}_n \) can be obtained by the following recursive
\( H_\infty \) estimator:
assuming that \( \{Q_n\} \) and \( \{P_n\} \) are nonsingular sequences of matrices. \( K_n \) is known as the \( H_\infty \) gain. In the case of a linear system it can be shown that the covariance of the estimation error is bounded by \( Q_n [28] \).

\[
E[(x_n - \hat{x}_n)(x_n - \hat{x}_n)^T] \leq Q_n.
\]

For nonlinear systems the transfer function \( G_{\infty} \) is undefined (but the maximum gain from \( v \) to \( \hat{x} \) is still generally less than \( \gamma \)), and the covariance bound is not strictly satisfied (but is still approximately satisfied). This is similar to the near optimality of the extended Kalman filter for nonlinear systems. The \( H_\infty \) filter equations do not satisfy the transfer function bound (13) or the covariance bound (15) unless the following inequalities hold at each time step \( n \)

\[
I - Q_n / \gamma^2 > 0,
\]
\[
I + HQ_n H^T > 0.
\]

2.2. Fuzzy system optimization

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. We use \( d_n \) to denote the target vector for the fuzzy system outputs at the \( n \)th time step, and \( h(k) \) to denote the actual outputs at this time step at the \( k \)th iteration of the \( H_\infty \) filter. In order to cast the membership function optimization problem in a form suitable for \( H_\infty \) 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 \( H_\infty \) 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 \( j_1 \) fuzzy sets for the first input, \( j_2 \) fuzzy sets for the second input, and \( \kappa \) fuzzy sets for the output. As before we denote the modal point and half-widths of the \( i \)th fuzzy membership function of the \( j \)th input by \( c_{ij}, b^-_{ij}, \) and \( b^+_{ij} \) respectively. We
The vector \( x \) thus consists of all of the fuzzy membership function parameters arranged in a column vector. The nonlinear system model to which the \( H_\infty \) filter can be applied is

\[
x_{n+1} = x_n + Bw_n + \delta_n, \\
d_n = h(x_n) + v_n,
\]

where \( h(x_n) \) is the fuzzy system's nonlinear mapping from the membership function parameters to the single fuzzy system output, and \( w_n, \delta_n, \) and \( v_n \) 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 [29,30]. Now we can apply the \( H_\infty \) recursion (14). \( f(\cdot) \) is the identity mapping, \( d_n \) is the target output of the fuzzy system, and \( h(\tilde{x}_n) \) 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 \( B \) and \( \gamma \) variables are tuning parameters that can be considered to be proportional to the magnitudes of the artificial noise processes. The determination of \( B \) and \( \gamma \) is a difficult task that remains as an open research problem, similar to the tuning of the covariance matrices of a Kalman filter [31]. However, some general guidelines for determining \( B \) and \( \gamma \) can be given. As we increase \( B \) and \( \gamma \) 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. This can be viewed as an increase in the “bandwidth” of the filter. The \( H_\infty \) optimization is somewhat sensitive to appropriate choices of \( B \) and \( \gamma \). Values of \( B \) and \( \gamma \) that are too small result in slow convergence of the optimization algorithm, and possibly convergence to a local minimum that is larger than that achieved by more appropriate values of \( B \) and \( \gamma \). Values of \( B \) and \( \gamma \) that are too large cause an oversensitivity of the algorithm to local gradients, and result in divergence. In our experiments we found that changes in \( B \) and \( \gamma \) by a factor of two or so did not have much of an effect on the algorithm, but changes by a factor of 10 gave worse results (i.e., divergence, or convergence to poor results) than more appropriate values of \( B \) and \( \gamma \).

### 2.3. Fuzzy system optimization with sum normal constraints

The \( H_\infty \) optimization proposed here 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 the $H_{\infty}$ filter 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 coupled, multivariable function with respect to one parameter and then independently changing 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.

Another way of constraining membership functions to be sum normal is to reduce the number of optimized parameters. For example, if the membership functions are triangular, then the upper half-width of a membership function must be equal to the lower half-width of the next membership function. These two half-widths are then both represented by a single parameter. However, such an optimization method cannot be extended to inequality constrained optimization. Inequality constrained optimization may be desired if we want certain membership functions to have centroids or half-widths that satisfy inequality constraints. We therefore take a more general approach to constrained optimization that can be extended to inequality constraints. Although inequality constrained optimization is not explicitly addressed in this paper, the method that we present can be extended to inequality constraints for applications other than sum normality [32].

As above we consider a two-input, one-output fuzzy logic system. The first input has $\mu_1$ fuzzy sets, and the second input has $\mu_2$ fuzzy sets. We denote the modal points and half-widths of the fuzzy membership functions by $c_{i1}, b_{i1}$, and $b_{i1}^j$ ($i = 1, \ldots, \mu_1$) for the first input, and $c_{i2}, b_{i2}$, and $b_{i2}^j$ ($i = 1, \ldots, \mu_2$) for the second input. If the membership functions for the two inputs are sum normal then the following equalities hold:

$$c_{1j} + b_{1j}^+ = c_{2j}, \quad (j = 1, 2),$$
$$c_{1j} + b_{2j}^- = c_{2j},$$
$$c_{2j} + b_{2j}^+ = c_{3j},$$
$$c_{2j} + b_{3j}^- = c_{3j},$$
$$\vdots$$
$$c_{\mu_j-1,j} + b_{\mu_j-1,j}^+ = c_{\mu_j},$$
$$c_{\mu_j-1,j} + b_{\mu_j}^- = c_{\mu_j}. $$

(19)
We have another set of equalities for the output. The fuzzy logic system has \( \kappa \) fuzzy sets for the output. We denote the modal points and half-widths of the fuzzy membership functions of the output by \( \gamma_i, \beta^+_i \), and \( \beta^-_i \) \( (i = 1, \ldots, \kappa) \). If the membership functions for the output are sum normal then the following equalities hold:

\[
\begin{align*}
\gamma_1 + \beta^+_1 &= \gamma_2, \\
\gamma_1 + \beta^-_2 &= \gamma_2, \\
\gamma_2 + \beta^+_2 &= \gamma_3, \\
\gamma_2 + \beta^-_3 &= \gamma_3, \\
\vdots & \quad \vdots \\
\gamma_{\kappa-1} + \beta^+_{\kappa-1} &= \gamma_\kappa, \\
\gamma_{\kappa-1} + \beta^-_\kappa &= \gamma_\kappa.
\end{align*}
\]  

(20)

Equalities (19) and (20) can be written in matrix form as

\[ Lx = 0, \]  

(21)

where \( x \) is the vector in (17) and \( L \) is the block diagonal matrix

\[
L = \begin{bmatrix}
L_1 & 0 & 0 \\
0 & L_2 & 0 \\
0 & 0 & L_3
\end{bmatrix}.
\]  

(22)

The \( L_i \) matrices are derived from (19) and (20). \( L_1 \) is a \( 2(\mu_1 - 1) \times 3\mu_1 \) matrix, \( L_2 \) is a \( 2(\mu_2 - 1) \times 3\mu_2 \) matrix, and \( L_3 \) is a \( 2(\kappa - 1) \times 3\kappa \) matrix. Each \( L_i \) matrix is of the form

\[
L_i = \begin{bmatrix}
M_1 & M_2 & 0_{2 \times 3} & \cdots & 0_{2 \times 3} \\
0_{2 \times 3} & M_1 & M_2 & \cdots & 0_{2 \times 3} \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0_{2 \times 3} & \cdots & 0_{2 \times 3} & M_1 & M_2
\end{bmatrix}.
\]  

(23)

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

\[
M_1 = \begin{bmatrix}
0 & 1 & 1 \\
0 & 0 & 1
\end{bmatrix},
\]  

\[
M_2 = \begin{bmatrix}
0 & 0 & -1 \\
1 & 0 & -1
\end{bmatrix}.
\]  

(24)

Therefore, in order to optimize fuzzy membership functions with the constraint that they remain sum normal, we can perform \( H_\infty \) filtering under the constraint defined by (21). The sum normal constrained fuzzy system optimization problem therefore reduces to an \( H_\infty \) estimation problem with state equality constraints. This problem has been solved in [32]. The introduction of state constraints changes the \( Q_{n+1} \) and the \( K_n \) equations given in the \( H_\infty \) filter equations (14) to the following:
\[ Q_{n+1} = (I - L^T L)FP_n F^T (I - L^T L) + BB^T, \]
\[ K_n = (I - L^T L)FP_n H^T. \]

The remaining equations in (14) remain unchanged. These changes ensure that the fuzzy logic system defined by the parameters in the \( \hat{x} \) vector is sum normal.

2.4. Computational analysis

The \( H_\infty \) filter given in (14) is dominated by the set of linear equations that must be solved in order to compute \( P_n \). In general, the computational effort required to solve \( k \) linear equations is proportional to \( k^3 \). The incorporation of the equality constraints in (25) is usually a small portion of the total computational effort, because those equations do not involve the simultaneous solutions of linear equations. The computational effort of both unconstrained and constrained Kalman and \( H_\infty \) filtering will therefore be proportional to \( k^3 \), where \( k \) is the total number of fuzzy membership function parameters. This shows that the optimization method presented here does not scale very well for large problems. However, many current efforts are directed towards rule base reduction, hierarchical fuzzy systems, and fuzzy systems whose parameter count grows slower than exponentially with the number of inputs and outputs [33].

In order to reduce the computational effort of the \( H_\infty \) filter, a pseudo-steady-state assumption can be made in (14) that

\[ H_n \approx H_0 = \frac{\partial h(\hat{x})}{\partial \hat{x}}_{\hat{x} = \hat{x}_0}. \]

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 \( h(\cdot) \) with respect to the state estimate \( \hat{x}_n \) does not change much from iteration to iteration [34]. 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 performance at a fraction of the computational cost. However, the computational effort will still grow with \( k^3 \).

Further computational savings can be obtained beyond the pseudo-steady-state assumption. If we monitor the value of the \( K_n \) matrix in the \( H_\infty \) filter, it will eventually reach a steady state value. In this case we can skip the calculation of \( Q_n \) and \( P_n \) and simply use the steady state value of \( K_n \). This method can only be used if the pseudo-steady-state assumption is used. This approximation will reduce the filter equations to the single \( \hat{x}_n \) equation given in (14). This equation has a computational cost that is proportional to \( k\kappa \), where again \( k \) is the total number of fuzzy parameters, and \( \kappa \) is the total number of output membership functions. After \( K_n \) has reached steady state we see that the computational effort is linearly proportional to the number of fuzzy parameters, and the optimization method becomes much more scalable. Again, however, this is a tradeoff between computational effort and theoretical integrity. The use of a steady state \( K_n \) may or may not give good optimization results, depending on the specific problem.
3. Simulation results

In this section we illustrate the use of the $H_\infty$ filter for training fuzzy membership function parameters, both with and without sum normal constraints. The application is a fuzzy automotive cruise control system [21, pp. 186ff]. 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

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

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

$$f_e(\theta) = f_i + \gamma \sqrt{\theta},$$
$$f_d(v) = \alpha v^2 \text{sign}(v),$$
$$f_g = mg \sin(\text{grade}),$$

where $\gamma$, $\alpha$, $g$, and $f_i$ are constants. We will use the values $m = 1000$ kg, $\gamma = 12,500$ N, and $\alpha = 4$ N/(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 two-input, one-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 intuitively with five membership functions each for the two inputs and the output. So $\mu_1$, $\mu_2$, and $\kappa$ in (17) are all equal to five. The rule base is shown in Table 1. This is the same as the rule base that is given for this problem in [21].

<table>
<thead>
<tr>
<th>Error change</th>
<th>Error</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>NL</td>
<td>NL</td>
<td>NS</td>
</tr>
<tr>
<td>NS</td>
<td>NL</td>
<td>NS</td>
</tr>
<tr>
<td>Z</td>
<td>NL</td>
<td>NS</td>
</tr>
<tr>
<td>PS</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>PL</td>
<td>PS</td>
<td>PS</td>
</tr>
</tbody>
</table>

The output of the fuzzy logic system is the change in throttle position. NL = Negative Large, NS = Negative Small, Z = Zero, PS = Positive Small, PL = Positive Large.
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.

A Kalman or $H_\infty$ filter can be used to optimize the fuzzy cruise control with respect to these 45 parameters. These 45 parameters are arranged in a vector as shown in (17) and hence comprise the 45-element state of the Kalman or $H_\infty$ filter. If we are not concerned with sum normality, we use the unconstrained Kalman filter equations [25], or the unconstrained $H_\infty$ filter equations shown in (14). If we desire to maintain sum normality in our optimized membership functions, we use the constrained Kalman filter equations [25], or the constrained $H_\infty$ filter modifications shown in (25). The matrix $L$ in Section 2.3 is a $2[(\mu_1 - 1) + (\mu_2 - 1) + (\kappa - 1)] \times 3(\mu_1 + \mu_2 + \kappa)$ matrix, which for this example is a $24 \times 45$ 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.

$H_\infty$ filtering (both with and without sum normal constraints) was implemented in Matlab to optimize the membership functions of the controller inputs and output. The pseudo-steady-state formulation as described in Section 2.4 was used to decrease training time. We tuned the $H_\infty$ filter parameters manually for the best convergence results. 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.

Fig. 1 depicts the progress of training with $H_\infty$ and Kalman filtering (both with and without sum normal constraints) with a time-varying $H$ matrix. As expected, the unconstrained filters converge more quickly and to a better solution than the constrained filters, and the $H_\infty$ filters exhibit better convergence than the Kalman filters.

The computational effort for the Kalman filter and the $H_\infty$ filter are about the same. The computational effort of the filters with time-varying $H$ matrices was about 20 s per 100 iterations (on a 1.2 GHz PC with 240 MB of RAM). The use of the pseudo-steady-state approximation described in Section 2.4 reduces the computational effort by about 25%. The CPU time required by the Kalman and $H_\infty$ optimization algorithms will be highly dependent on the implementation details. The computational effort given in this paper should be used only for relative comparisons.

Now we move from the training scenario to the test scenario. Fig. 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 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 $g_n$ as described earlier (9). The optimized membership functions are not sum normal in this case since we did not use the sum normal constraints.

Fig. 3 shows the original membership functions and unconstrained optimized membership functions for the output. (The input membership functions are not
shown because they did not change as much during the optimization process.) The optimized membership functions work well as seen from Fig. 2, but they are clearly...
not sum normal, which may be undesirable. In fact, the $H_\infty$ 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.

Fig. 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 Fig. 2 shows that (as expected) the constrained controller does not perform as well as the unconstrained controller. As seen from Fig. 5, the optimized membership functions are indeed sum normal. Comparison of Figs. 3 and 5 shows what a drastic difference sum normal constraints can make in the resultant membership functions.

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.

It is seen from Table 2 that the removal of the pseudo-steady-state approximation generally results in a decrease of the error function value—sometimes by only a small amount, but other times by a large amount. In addition, unconstrained optimization generally results in better performance than constrained optimization. We can also see that $H_\infty$ filtering results in better performance than Kalman filtering. However, this should not be taken as an inviolable law. The performance of $H_\infty$ filtering and Kalman filtering both depend strongly on the initial conditions of the membership functions and the tuning parameters of the optimization algorithm. For $H_\infty$ filtering we need to choose appropriate values of $\gamma$ and $B$ in (14) and (25). For Kalman filtering we need to choose appropriate values of the matrices $P_0$, $Q$, and $R$, as discussed in [25]. In general we can get better performance from $H_\infty$ filtering because the $H_\infty$ filter is inherently more robust to linearization errors than the Kalman filter. The Matlab code that was used to generate these results can be downloaded from the
Fig. 5. Output membership functions: (a) default; (b) optimized via constrained Kalman filtering; (c) optimized via constrained $H_\infty$ filtering.

internet at http://academic.csuohio.edu/simond/fuzzyopt/. These results can then be reproduced by running those Matlab m-files.
The initial fuzzy controller in all cases had a normalized training error of 1000.

4. Conclusion

We have shown that the membership functions of a fuzzy controller can be optimized via $H_\infty$ filtering. In general, this optimization method results 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 therefore extended the $H_\infty$ filtering algorithm to ensure that the resulting membership functions are sum normal. This results in a fuzzy controller with worse performance than the unconstrained membership functions (in general), 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, $H_\infty$ filtering for fuzzy membership function optimization resulted in better performance than Kalman filtering. This is to be expected because the $H_\infty$ filter is more robust to model errors and linearization errors than Kalman filtering.

$H_\infty$ filtering 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 could 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, $H_\infty$ filtering or Kalman filtering could be used to fine-tune the results.

Further work in this area could focus on the convergence properties of the $H_\infty$ filter and the Kalman filter in this application (for example, following the lines of [35]), the effect of the tunable parameters of the filters, the optimization of fuzzy systems with nontriangular membership functions, or the extension of this work to other derivative-based schemes (e.g., unscented filtering [36]) for the optimization of fuzzy membership functions.
References