Fuzzy Membership Optimization via the Extended Kalman Filter

Dan Simon - simon@csvax.csuohio.edu
Cleveland State University
Cleveland, Ohio, USA 44115

Abstract

The generation of membership functions for fuzzy systems is a challenging problem. In this paper we present a new scheme for optimizing Mamdani fuzzy controllers. We optimize the membership functions with an extended Kalman filter. We describe the algorithm and compare it with gradient descent optimization of fuzzy membership functions. The methods discussed in this paper are illustrated on a simple automotive cruise control problem. We demonstrate that the Kalman filter can be an effective tool for improving the performance of a fuzzy controller.

1 Introduction

The performance of a fuzzy system depends on both its rule base and its membership functions. Given a rule base, the membership functions can be optimized in order to obtain the best possible performance from the fuzzy system. This paper does not address the generation of rule bases, but does suggest a method for the generation of membership functions. Many methods (both derivative-based and derivative-free) have been proposed for the optimization of fuzzy membership functions. See [9] for a list of references. This paper extends the results of [9] and demonstrates that the Kalman filter can be effectively used to optimize the membership functions of a Mamdani fuzzy controller.

This paper describes how the extended Kalman filter can be applied to fuzzy system optimization. We demonstrate its performance on a fuzzy controller and compare it with fuzzy controller optimization using gradient descent. It is shown that the Kalman filter converges more quickly, finds a better solution, and requires only slightly more computational effort than gradient descent. A pseudo-steady-state Kalman filter can be used to decrease the computational effort and achieve results on par with the Kalman filter.

2 Optimization

In this paper we will assume that the membership functions are triangular. We denote the centroid, lower half-width, and upper half-width of the ith fuzzy membership function of the jth input by c_{ij}, b_{ij}^-, and b_{ij}^+ respectively. The degree of membership of a crisp input x in the ith category of the jth input is therefore given by

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

In this paper we will restrict our discussion to single output systems for ease of notation and illustration. Conceptually, the results of this paper can be easily extended to multiple output systems, but the notation becomes quite cumbersome.

The fuzzy output is mapped into a crisp numerical value using centroid defuzzification [5]. Centroid defuzzification can be expressed as

$$\text{crisp output} = \frac{\sum_{j=1}^{n} m(\gamma_j)\gamma_j J_j}{\sum_{j=1}^{n} m(\gamma_j) J_j} \tag{2}$$

where γ_j and J_j are the centroid and area of the jth output fuzzy membership function, and n is the

number of fuzzy output sets. For the special case of two fuzzy inputs, the fuzzy output function $m(\gamma)$ is computed as

$$m(\gamma) = \text{fuzzy output function} = \sum_{i,k} m_{ik}(\gamma) \quad (3)$$

where $m_{ik}(\gamma)$ is defined as the consequent fuzzy output function when input 1 is in class i and input 2 is in class k.

$$m_{ik}(\gamma) = w_{ik} m_{oik}(\gamma) \quad (4)$$

$m_{oik}(\gamma)$ is the fuzzy function of the consequent that is activated when input 1 is in class i and input 2 is in class k, and w_{ik} is the activation level of that consequent.

$$w_{ik} = \min[f_{i1}(\text{input } 1), f_{k2}(\text{input } 2)] \quad (5)$$

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

$$E = \frac{1}{2N} \sum_{q=1}^{N} g_q^2 (y_q - \hat{y}_q)^2 \quad (6)$$

where N is the number of training samples, g_q is a weighting function, y_q is the target value of the fuzzy system, and $\hat{y}_q$ is the output of the fuzzy system. We can optimize E by using the partial derivatives of E with respect to the centroids and half-widths of the input and output fuzzy membership functions. We can obtain expressions for these derivatives using (1) and following. Then, using the differentiation chain rule on (6), we can obtain expressions for the derivative of the error function with respect to the half-widths and centroids. 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 first suggested in [3] and was extended in [7, 8]. See those references for detailed derivations and formulas for the derivatives. In this paper we will consider optimization only with respect to the input membership function parameters c_{ij}, b_{ij}^-, and b_{ij}^+.

2.1 The Extended Kalman Filter

Derivations of the extended Kalman filter are widely available in the literature [1, 2]. In this section we briefly outline the algorithm and show how it can be applied to fuzzy membership function optimization. Consider a discrete time system of the form

$$\begin{aligned} x_{n+1} &= f(x_n) + w_n \quad (7) \\ d_n &= h(x_n) + v_n \end{aligned}$$

where the vector x_n is the state of the system at time n, w_n is the process noise, d_n is the observation vector, v_n is the observation noise, and $f(\cdot)$ and $h(\cdot)$ are nonlinear vector functions of the state. Assume that the sequences $\{v_n\}$ and $\{w_n\}$ are zero-mean, and the initial state x_0, $\{v_n\}$, and $\{w_n\}$ are gaussian and independent from each other with

$$\begin{aligned} \mathcal{E}[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^T] &= P_0 \quad (8) \\ \mathcal{E}(w_n w_l^T) &= Q\delta_{nl} \quad (9) \\ \mathcal{E}(v_n v_l^T) &= R\delta_{nl} \quad (10) \end{aligned}$$

where $\mathcal{E}(\cdot)$ is the expectation operator and δ_{nl} is the Kronecker delta. The problem addressed by the extended Kalman filter is to find an estimate $\hat{x}_{n+1}$ of x_{n+1} given d_j $(j = 0, \ldots n)$.

If the nonlinearities in (7) are sufficiently smooth, the system can be approximated as

$$\begin{aligned} x_{n+1} &= F_n x_n + w_n + f(\hat{x}_n) - F_n \hat{x}_n \quad (11) \\ d_n &= H_n^T x_n + v_n + h(\hat{x}_n) - H_n^T \hat{x}_n \end{aligned}$$

where

$$\begin{aligned} F_n &= \left.\frac{\partial f(x)}{\partial x}\right|_{x=\hat{x}_n} \quad (12) \\ H_n^T &= \left.\frac{\partial h(x)}{\partial x}\right|_{x=\hat{x}_n} \end{aligned}$$

It can then be shown that the desired estimate $\hat{x}_n$ can be obtained by the recursion

$$\begin{aligned} K_n &= P_n H_n (R_n + H_n^T P_n H_n)^{-1} \quad (13) \\ \hat{x}_n &= f(\hat{x}_{n-1}) + K_n[d_n - h(\hat{x}_{n-1})] \\ P_{n+1} &= F_n(P_n - K_n H_n^T P_n) F_n^T + Q_n \end{aligned}$$

In order to reduce the computational effort of the Kalman filter, a pseudo-steady-state assumption can be made in (13) that

$$H_n^T \approx H_0^T = \left.\frac{\partial h(x)}{\partial x}\right|_{x=\hat{x}_0} \tag{14}$$

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 [2]. But the extended Kalman filter is only an approximation in the first place because of the linearization discussed above, so if computational effort is any consideration at all, the pseudo-steady-state assumption certainly deserves consideration.

2.2 Application to Fuzzy Control

Inspired by the successful use of the Kalman filter for training neural networks [6], for defuzzification strategies [4], and for training fuzzy filters [8], we can apply a similar technique to the training of fuzzy control systems. 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. In this paper we will optimize the membership function parameters of the inputs but not the output.

We will consider a two-input fuzzy controller. 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 which has μ fuzzy sets for the first input and ν fuzzy sets for the second input. As above we denote the centroid and half-widths of the ith fuzzy membership function of the jth input by c_{ij}, b_{ij}^-, and b_{ij}^+. The state of the nonlinear system can then be represented as

$$\begin{aligned} x = \big[\ & b_{11}^- \quad b_{11}^+ \quad c_{11} \quad \cdots \quad b_{\mu 1}^- \quad b_{\mu 1}^+ \quad c_{\mu 1} \\ & \cdots \quad b_{12}^- \quad b_{12}^+ \quad c_{12} \quad \cdots \quad b_{\nu 2}^- \quad b_{\nu 2}^+ \quad c_{\nu 2}\ \big]^T \end{aligned} \tag{15}$$

The vector x thus consists of all of the fuzzy membership function parameters arranged in a single vector. The nonlinear system model to which the Kalman filter can be applied is

$$\begin{aligned} x_{n+1} &= x_n + w_n \\ d_n &= h(x_n) + v_n \end{aligned} \tag{16}$$

where $h(x_n)$ is the fuzzy system's nonlinear mapping between the membership function parameters and the single output of the fuzzy system, and w_n and v_n are artifically added noise processes that improve the stability of the Kalman filter [6]. Now we can apply the Kalman recursion (13). In Section 2.1, $f(\cdot)$ is the identity mapping, d_n is the target output of the fuzzy system, and $h(\hat{x}_n)$ 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_n and R_n matrices are tuning parameters which can be considered as the covariance matrices of the artificial noise processes w_n and v_n respectively.

3 Simulation Results

In this section we describe and illustrate the use of Kalman filter training for the membership parameters of a fuzzy controller for an automotive cruise control system [10, pp. 186ff.]. An automobile's acceleration can be stated as a function of the external forces acting on the vehicle: engine force, 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

$$m\dot{v} = F_{e1}(\theta) - F_d(v) - F_g \tag{17}$$

where m is the vehicle mass, θ is the throttle position, and F_{e1} and F_d are given by

$$\begin{aligned} F_{e1}(\theta) &= F_i + \gamma\sqrt{\theta} \\ F_d(v) &= \alpha v^2 \mathrm{sign}(v) \end{aligned} \tag{18}$$

where γ, α, and F_i are constants. In this paper we use the values $m = 1000$ kg, $\gamma = 12,500$ Newtons, and $\alpha = 4$ N / (m/s)^2. F_i is the engine idle force, which we will assume to be 1000 N.

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 and ν in (15) are both equal to five and the fuzzy cruise control has a total of 10 membership functions for the inputs. Each membership function is constrained to be triangular, so each membership function has three parameters (a centroid and two half-widths). Thus the fuzzy cruise control has a total of 30 parameters to be determined. Gradient descent can be used to optimize the fuzzy cruise control with respect to these 30 parameters. For the Kalman filter, these 30 parameters are arranged in a vector as shown in (15) and hence comprise the 30-element state of the Kalman filter.

The error function (6) 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 (6) was equal to 60. The weighting function g_q in (6) was set equal to q/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.

Both the gradient descent and Kalman filter methods were implemented in MATLAB to optimize the membership functions of the controller inputs. 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. The reference speed of the cruise control was set at 40 m/s, so the objective of the cruise control was to maintain a 40 m/s velocity even after encountering a 10% increase in road grade.

The Kalman filter parameters were set as follows: the matrix Q in (9) was set to $4000I_{30}$ (where I_{30} is the 30×30 identity matrix); the matrix R in (10) was set to 40 (a scalar, since there is only one measurement for the Kalman filter); and the matrix P_0 in (8) was set to $40000I_{30}$.

Figure 1 depicts the progress of the gradient descent method, the Kalman filter method, and the pseudo-steady-state Kalman Filter as represented in (14) during optimization of the membership functions. It can be seen that although all three methods yield comparable results, the Kalman filter methods converge to better solutions than the gradient descent method. The pseudo-steady-state Kalman Filter converges more quickly than the other two methods, although its final solution is not quite as good as the Kalman filter method.

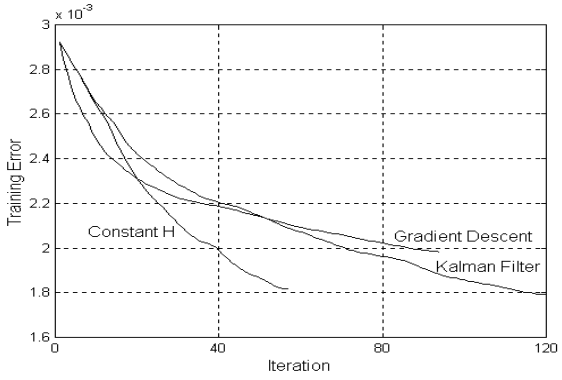

Figure 1: Training Progress.

The computational requirements of the gradient descent and Kalman filter methods are about the same. Although the Kalman filter equations (13) 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 majority of the computational effort for the two methods consists of calculating the derivatives of the objective function with respect to the membership parameters, and this calculation is the same for both optimization methods. The optimization methods were run on a Pentium III 550 MHz PC. The gradient descent method and the Kalman filter method both required about 15 s per iteration. The pseudo-steady-state Kalman Filter method is much faster than the other two methods. Although its first

iteration takes about 15 s, each iteration after the first takes less than 1 s because the partial derivative computation is skipped.

Fig. 2 shows the training data with the nominal fuzzy cruise control, and with the cruise control that has been optimized with the Kalman filter. At $t = 0$ the automobile encounters a sudden 10% increase in the road grade. The cruise control attempts to maintain the 40 m/s velocity in the presence of the increased road grade. It can be seen that the initial oscillations are about the same between the two curves, but the settling time is noticably smaller for the optimized cruise control. This reflects our choice of g_q in (6) to decrease the settling time at the expense of overshoot.

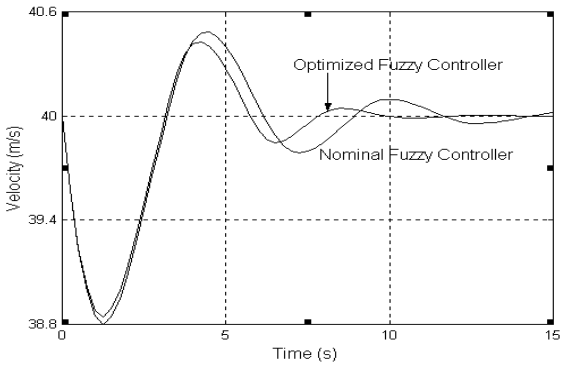

Figure 2: Cruise Control Performance.

4 Conclusion

We have shown that an extended Kalman filter can be used to optimize the membership functions of a fuzzy controller. The method was applied to a fuzzy automotive cruise control. The Kalman filter converges to a better solution than gradient descent, and requires only slightly more computational effort. The computational effort can be drastically decreased by using a pseudo-steady-state Kalman filter. Further work could focus on investigating the effect of the covariance matrices on the convergence of the Kalman filter, training non-Mamdani fuzzy systems, or training fuzzy systems with non-triangular membership functions. MATLAB code that implements the algorithms described in this paper can be downloaded from `http://csaxp.csuohio.edu/~simon/fuzzyopt/`.

References

[1] B. Anderson and J. Moore, *Optimal Filtering*, Prentice Hall: Englewood Cliffs, NJ, 1979.

[2] A. Gelb, Applied Optimal Estimation (MIT Press, Cambridge, Massachusetts, 1974).

[3] F. Guély and P. Siarry, "Gradient Descent Method for Optimizing Various Fuzzy Rule Bases," *IEEE Int. Conf. on Fuzzy Systems*, pp. 1241-1246, March 1993.

[4] T. Jiang and Y. Li, "Generalized Defuzzification Strategies and Their Parameter Learning Procedures," *IEEE Trans. on Fuzzy Systems*, (4)1, pp. 64-71, February 1996.

[5] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice Hall: Englewood Cliffs, NJ, 1992.

[6] G. Puskorius and L. Feldkamp, "Neurocontrol of Nonlinear Dynamical Systems with Kalman Filter Trained Recurrent Networks," *IEEE Trans. on Neural Networks*, (5)2, pp. 279-297, 1994.

[7] D. Simon and H. El-Sherief, "Fuzzy Logic for Digital Phase-Locked Loop Filter Design," *IEEE Trans. on Fuzzy Systems*, (3)2, pp. 211-218, May 1995.

[8] D. Simon, "Design and Rule Base Reduction of a Fuzzy Filter for the Estimation of Motor Currents," *Int. J. of Approximate Reasoning*, submitted for publication.

[9] D. Simon, "Training Fuzzy Systems with the Extended Kalman Filter," *IEEE Trans. on Fuzzy Systems*, submitted for publication.

[10] J. Yen and R. Langari, *Fuzzy Logic: Intelligence, Control, and Information*, Prentice Hall: Englewood Cliffs, NJ, 1999.