Article can be accessed online at http://www.publishingindia.com

Model Predictive Control for Nonlinear Systems in State Space using Fuzzy System Model

S. Ananthi*, G. Venkata Ramu, K. Padmanabhan*****

Abstract

Model predictive (MPC) is a common method used in chemical process industries. Usually, the state space method is applicable for linear systems with a quadratic performance index to find the optimised control law based on a solution of the Riccati equation. However, a nonlinear system can only be modeled by a fuzzy logic based function of the variables. The method of optimal control requires a performance index to be met, which is not necessarily a quadratic type of index but a nonlinear function of the process variables. It could be similarly modeled by another fuzzy inference system. The MPC method for such a fuzzy modeled state space system would be able to provide good predictive control for any nonlinear control system. The evaluation of the control steps by prediction for such a fuzzy model with fuzzy performance index is described in this paper. The optimal control steps are found by iterative search using the Box's Complex search method over a range of control values. Then, the prediction outputs are checked for constraint inequality satisfaction, such as pressure limit for example. Such a control step is applied at the current time step. The paper describes such a technique for a nonlinear process with a nonlinear performance index, also with constraints in the process variables.

Keywords: Fuzzy Control System, Control Law Optimisation, Model Predictive Control, Model Predictive Control

1. Introduction

In model predictive control (Balbis *et al.*, 2005), quite many calculations are made before providing a control step at any instant *t*. The control values are updated in discrete time steps to the model of the plant. The model of the process is used in MPC to evaluate, for a set of control values u, the outputs of the process states. It is used to find how the response will be for various values of control input, given its current measured value. The objective is to obtain a control value for the current time step that will not transgress limiting values in future and also yield an optimal performance value. After obtaining a control step sequence that might be meeting these three requirements at that instant *t*, the control step thus calculated is applied to the process. Then, at the next time step, the calculation for control value is repeated. The MPC technique has a performance index to meet. This can be similar to the quadratic performance involving the errors of the state variables. If we predict the output variables over a period of 5 seconds, the total loss is calculated on the basis of the deviations from the set points of the variables and the power loss due to control action exercised over this 5 second period. In our aim to minimize this loss, control value *u* will be varied. If we adjust the control value over a 2 second period and then leave the control value same for the rest of the 5 second period, then the response of the system during the 5 second time can be calculated. The principle of prediction is just this. This output should be within prescribed values during this time. The reason is that process variables cannot exceed certain limiting values. Also, the control value might have its own limits. Once the values of u are found for discrete time instants during the two second period to satisfy both the optimisation and the constraints, the first step of such a calculated *u* is applied to the system. During the next step, recalculation follows, based upon the measured output at the end of this step.

* Professor & Head, Department of Network Systems and Information Technology, University of Madras, Chennai, India. Email: ananthibabu@yahoo.com

** Instrumentation Department, University of Madras, India

In models of process systems, the state space model is generally used for optimisation. That model becomes nonlinear in the case of actual plant. Models chosen for predictive control are generally assumed to have linear state space relations. J. B. Rawlings describes about model targets and the receding horizon calculation (Box, 1965). Dynamic Matrix Control (DMC) was the first model Predictive Control (MPC) algorithm introduced in early 1980s (Cutler & Ramaker, 1980). Nowadays, DMC is available in commercial industrial distributed control systems and process simulation software packages. This DMC algorithm works with a step response model. The step response of a nonlinear system changes with the value of the step and so this method is not quite good for totally nonlinear processes.

Most of the situations where MPC schemes are in use are for batch processes in chemical industry. These processes are nonlinear and if they are defined by a fuzzy model, of any type, then the state space equations pertaining to the variables will be used for optimisation based control law finding. Jairo J. Espinosa *et al*. (1998) present four algorithms to construct the controllers which are compared using a model of a Continuous Stirred Tank Reactor. A nonlinear model is essentially defined by a fuzzy inference system (FIS). Dhouib, Djemel& Chtourou, (2011) describe a principle where a nonlinear system is divided into a number of linear subsystems. So the linear model based predictive control (MPC) technique is used for each subsystem.

The Takagi Sugeno model alone is capable of fitting a nonlinear function to a FIS (*Huang et al., 2000)*. Babuska *et al.* have also investigated the MPC with such a fuzzy logic defined model. They consider that by linearising a model from time to time (as the response proceeds) by such a FIS, it can help in dealing with the nonlinearity in the process. Their modeling equations are based on the present and future values of state variables and control increments. Instead of just considering the fuzzy model of a state space, they treat the set of all values of state variables for all the discrete time instants together with the control vector values for these instants as one fuzzy functional group. In another paper, Mollov *et al.* (2004) have described a similar strategy for fuzzy inference based handling of the MPC scheme. They have described the same for a pH control scheme and for a distillation column with some detail. Just two membership functions (also trapezium types) have been assumed for this to simplify the calculations.

Pena *et al.* (2009) also discuss a combined fuzzification of the optimisation problem along with the fuzzification of the model of the system. In their equations, they include the derivatives of the optimisation function, which are stated as needed for finding the minimum point in the traverse. The method of calculating the derivatives are given in the paper, which is used to find the minimal point in the course of the predicted set of discrete times.

A novel multiple model control strategy using both fuzzy-based predictive control (FPC) and overall fuzzy-based predictive model (OFPM) has been proposed by Mazinan & Sadati (2010). Concerning the strategy, the system must be modeled through the multiple models over different operating regions.

Pappa *et al.* (2005) have applied linear and fuzzy models in an MPC for a milk pasteurisation control process.

2. Fuzzy Model in Control Systems

It was Dr. Mamdani in London University in the year 1975 who first paved the way for fuzzy model based control scheme to operate a small steam engine.

Figure 1: System Response without Fuzzy Logic Controller

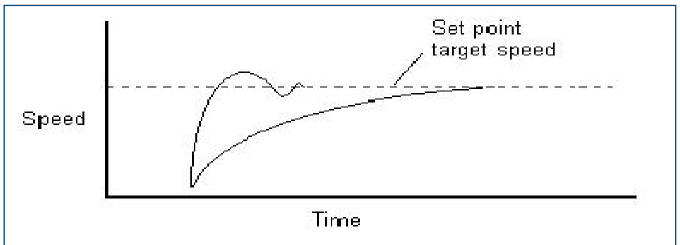


Figure 1. System Response Without Fuzzy Logic Controller

The controller worked well, and better than anything they had done with any other method. The steam engine speed control graph using the fuzzy logic controller appeared as in Figure 2.

The engine speed approached the desired value very quickly, did not overshoot and remained stable. The calculations must have been done on a fuzzy model of the engine and predicted, prior to actually applying the control in discrete time steps. Otherwise, the response shown in Figure 2 would be unrealisable. The name Model Predictive Control was introduced much later, however.

Figure 2: Steam Engine Speed Control Graph Using the Fuzzy Logic Controller

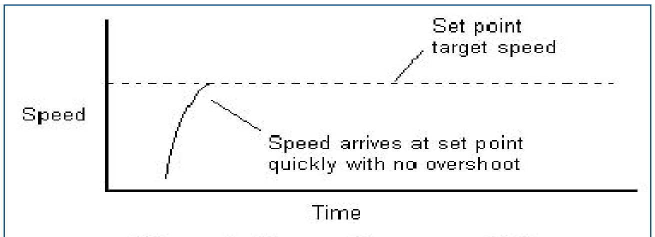


Figure 2. System Response Using Fuzzy Logic Controller

Further, due to the non-linearity of the steam engine operating characteristics, as soon as the speed set-point was changed, the trial and error effort had to be done all over again to arrive at an effective control. This did not occur with the fuzzy logic controller, which they reported as adapting much better to changes, variations and non-linearity in the system.

There were two outputs: control of heat to the boiler and control of the throttle. The outputs operated independently.

3. Analysis Equations of the Model Predictive Control Scheme

MPC is an optimisation based control law, and the performance measure *J* is almost always based on quadratic norms. In constrained finite time optimal control (CITOC) problem, there does not exist any *simple* closed-form expression for the solution. Instead, the first step in MPC is to define a prediction horizon *T* and approximate the performance measure by using a finite horizon. The second idea is to apply only the first control move of the obtained sequence $U_T(x(t))$ to the plant and resolve a new finite horizon problem when we obtain new measurements of the current state *x(t).*

1. Measure x(t) at sampling instance t.
2. Solve the finite optimisation problem and obtain the optimal input sequence $U_T(x(t))$.
3. Apply the first element of the sequence u_0 to the system.
4. Back to step 1.

Due to the computational complexity of solving the optimisation problem in step 2, MPC was limited to systems with long sampling periods, systems with only few state variables and rather short prediction horizons.

Figure 3: Two Cascade Configuration for MPC

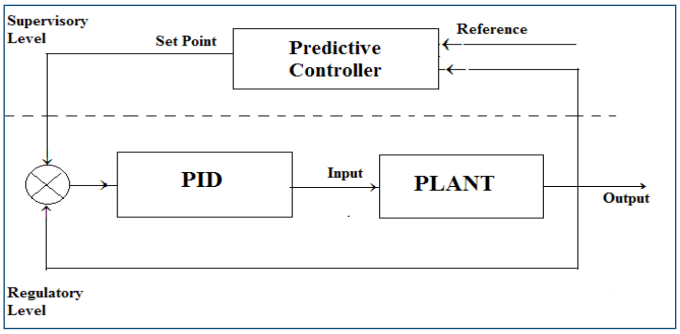


Figure 3 shows a supervisory level controller sitting over the conventional plant cum PID controller. The set point is not the difference between the output and the reference. The set point is provided by the predictive controller as per calculations based on future predicted outputs. This configuration, also known as the Cascade predictive controller (Rawlings, 2000) that has the model predictive block over the existing controller and the closed loop system is intact as usual. The set point only changes with time, so that the output follows the reference with optimum conditions under constraints.

In the scheme shown in Figure 3, the predicted output is compared with the reference and generates the error signal. But the error signal operates not through a PID controller, but through a calculation involving optimisation and evaluates the current value of the control signal *u.*

As an example, if a generator speed is controlled (Figure 4), the control value is calculated based on the model of the system for four time slots. Then, what the process output (a prediction) would be for some more time is also calculated. In this time, the value of the predicted output should lie within certain constraint values. For the four steps of control applied, the power spent in control, the loss estimated due the speed error over the entire time of prediction are also evaluated. The control steps (four numbers) are adjusted so as to minimize this cost. Thus, what should be control step for the current instant is found along with three more steps. But only the first step is applied. Then, the output is read again and the steps repeated.

The model in state space formulation, for two output variables y, z is :

Figure 4: **Control Steps with Time and Sampled Output Values in MPC**

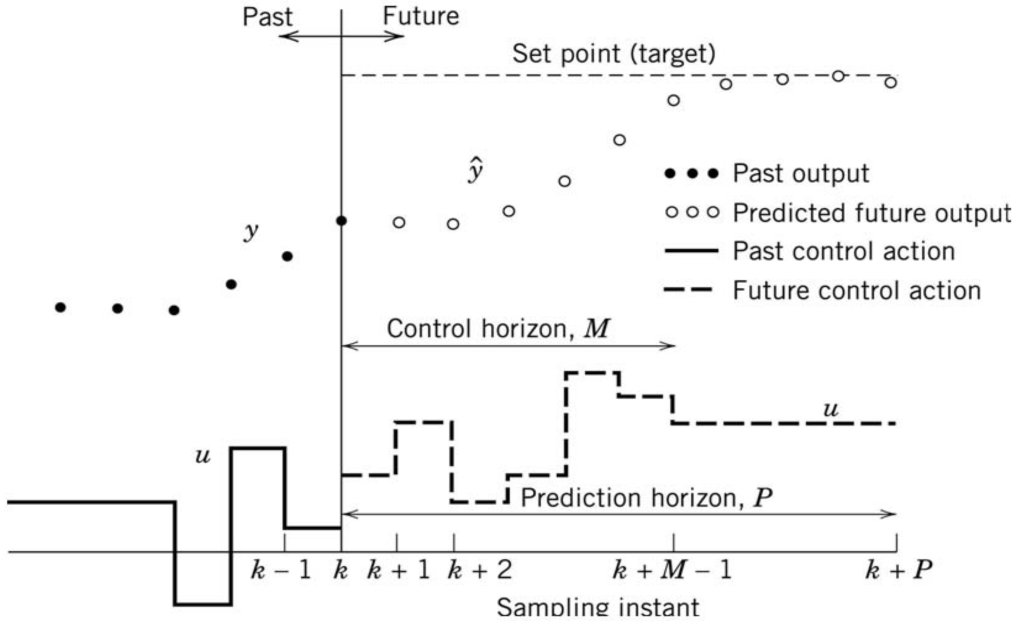
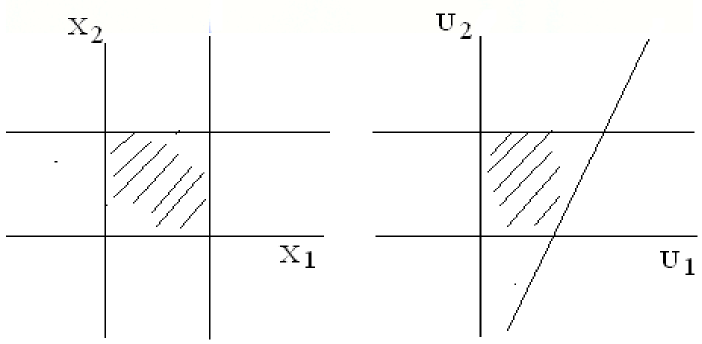


Figure 5: **Constraints in State and Control Variable Space**



$$x' = Ax + Bu$$
$$y = C_y x + Du$$
$$or \qquad (1)$$
$$y = C_y x$$
$$z = C_z x$$

The model predictive control accommodates optimizing a cost or related function as well. In a linear model of this function, the performance index *F* is given in terms of the *Q* and *R* matrices pertaining to cost evaluation based on the state variables *x* and the control variables *u*.

$$F_c = x^T Qx + u^T Ru \qquad (2)$$

Further to this, the control scheme has to meet certain constraints both for the state variables *x* and for the controlling function values *u*. These variables are not to exceed certain ranges, such as, pressure not to exceed 10 bar and flow not to be less than 100 *l/m.*

In normal cases, such constraints can be written as linear inequalities.

$$Hx + Gu \leq 0$$

A graphical picture of these constraints can be as shown in Figure 5.

In discrete time representation, the above equations become:

$$x(k+1) = A\,x(k)$$

$$y(k) = c_y x(k)$$

$z(k) = c_z x(k)$

The *cost* function calculated at time slot *k* will be:

$$V(k) = \sum_i |z(k+i) - r(k+i)|_Q^2 \;---(i = w...p) \\ + \sum_i \{\Delta u(k+i)\}_R^2 \;--- i = 0...(i = w...u-1) \tag{3}$$

In the above equation (3), the first sum is for all values of *i* from the current time *w to p*, where suffix *w* denotes current time slot and *p* denotes the further time slot up to which predicted values are calculated for the variables. *r* is the reference value vector of the variables and *z* is the output vector. The suffices Q and R are relating to performance index.

This first sum thus indicates that there is a cost involved which is due to excess or difference between the required *r* vector and the output vector *z*. The second sigma sum denotes the power cost for the control vector change Δu. (*R* denotes the multiplying factor). The second term is also a squared value indicating power and is summed for values up to the time slot one less than H_u.

To illustrate the calculation process over the predicted time instants, let us take a single variable *x* in the state space and one control variable *u*.

Starting at time slot *1*, we proceed to find out the variable value at time slot 2.

$$x(2) = a\,x(1) + bu(1) \tag{4}$$

The *A* and *B* matrix/vector in Equation (1) has become constants as *a* and *b*.

From the value at slot 2, likewise we can find the value at slot 3.

$$x(3) = a(a\,x(1) + bu(1)) + bu(2) \tag{5}$$

$$= a.a.x_1 + ab\underline{u_1} + bu_1 + b\,\Delta u_2 \tag{6}$$

$$= a^2 x(1) + a^0 bu1 + a^1 bu1 + b\,\Delta u_2 \tag{7}$$

$$= a^2 x(1) + \sum_0^{2-1} a^i bu_1 + b\Delta u_2 \tag{8}$$

Up to the value of time slot H_u-1, if we evaluate the state variable *x*,

$$x_{Hu+1} = A^{Hu} x + \sum_0^{Hu-1} a^i bu + b\Delta u_{Hu} \tag{9}$$

We predict the state variables by pre-calculation from the current time slot up to H_p. We apply control values of *u* from the current time slot up to H_u only, where $H_u < H_{p-}$. Thus, we continue the calculations up to H_p *for x*.

The above illustration for a single state variable can be extended to the general ***x state vector*** in the following matrix product summed equation.

$$\begin{vmatrix} x(k+1) \\ .x(k+2) \\ ..... \\ x(k+H_u) \\ x(k+H_u+1) \\ x(k+H_p) \end{vmatrix} = \begin{vmatrix} A \\ .A^2. \\ ... \\ A^{Hu} \\ A^{Hu+1} \\ A^{Hp} \end{vmatrix} x(k) + \begin{vmatrix} B \\ .AB \\ . \\ \sum_{i=0}^{Hu-1} A^i B \\ \sum_{i=0}^{Hu} A^i B \\ \sum_{i=0}^{Hp-1} A^i B \end{vmatrix} u(k-1)$$

$$+ \begin{vmatrix} B & 0 & 0 & 0 & 0 \\ AB+B & & & & 0 \\ \sum^{Hu-1} A^i B & & & & B \\ \sum^{Hu} A^i B & & & & AB+B \\ \sum^{Hp-1} A^i B & & & & \sum_{i=0}^{Hp-Hu} A^i B \end{vmatrix} . \begin{vmatrix} \Delta u(k) \\ \Delta u(k+1) \\ ...... \\ \Delta u(k+H_u-1) \end{vmatrix} \tag{10}$$

The above vector/matrices can be denoted as Θ, Y and Ψ, so that the condensed equation is:

$$X(k+1) = \Psi\, x(k) + Y\, u(k-1) + \Theta\, \Delta u \tag{11}$$

The above are the equations to calculate the future points and closed loop predictions of *z* are:

$$\begin{vmatrix} z(k=1) \\ .... \\ \\ z(k+H_p) \end{vmatrix} = \begin{vmatrix} C_z & & & \\ & C_z & & \\ & & C_z & \\ & & & C_z \end{vmatrix} \begin{vmatrix} x(k+1) \\ ;; \\ .. \\ x(k+H_P \end{vmatrix} \tag{12}$$

The constrained optimisation problem to be solved is:

$$V(k) = \sum_{i=H_{\acute{E}}}^{H_P} \|\hat{z}(k+i\backslash k) - r(k+i\backslash k)\|_{Q(i)}^2 + \sum_{i=0}^{H_u-1} \|\text{''}\,\hat{u}(k+i\backslash k)\|_{R(i)}^2 \tag{13}$$

This is subject to the inequality with slack variables *f,g,w.* The problem is a Quadratic performance calculation with constraints.

$$E\begin{bmatrix}\Delta u(k)\\1\end{bmatrix}\le 0$$
$$F\begin{bmatrix}u(k)\\1\end{bmatrix}\le 0 \tag{14}$$
$$G\begin{bmatrix}z(k)\\1\end{bmatrix}\le 0$$

The constraints are converted to a single linear inequality of the form;

$$\begin{vmatrix}F_1u(k-1)-f\\-\Gamma\{\Psi\cdot x(k)+Yu(k-1)\}-g\\-w\end{vmatrix}=0 \tag{15}$$

The solution of the eqn. (10) subject to optimisation as per eqn. (13) is to be made at every time step. This is a rather considerable effort in computation. Thus, the straightforward computation method ordinarily done for MPC is subject to computational load which made it feasible for only the slower process control systems. Further, when it comes to a wide range for process variables to be controlled, the nonlinear effects of the process are not included in the eqn. (10) because it considers the state matrices as constants.

4. Fuzzy Inference System Based Model Predictive Control

In this work, a direct approach to state space model using fuzzy logic functions is employed. In formulating the state space model, the usual relations are

$$X' = AX + B.U$$
$$Y = CX \tag{17}$$

where X is the state variable vector, Y, the process output vector, U the control vector and A,B and C are state matrices. The fuzzy model is easily formulated for any nonlinear system using a Sugeno Fuzzy inference method, defined by fuzzy membership functions, rules and output equations to fit any nonlinear surface of the process variables' dynamics. This is given by a fuzzy function *f(X)* in the equations below and as a figure in Figure 6. The control relation *BU* is usually linear.

$$X' = f(X) + B.U$$
$$Y = CX \tag{18}$$

Thus prediction equations step by step, similar to equations (3,4) are :

$$X_{k+1} = f(X_k) + B.U_k$$
$$Y_{k+1} = CX_k \tag{19}$$

The calculation is based on steps applied for k time slots and prediction made for N slots where N>>k. All k time steps are applied and after that time another estimate is made after noting the process values.

The steps applied can be any decreasing monotonous set of values commencing from a large step. The idea is that a large initial step helps in reaching the set value with least delay.

To optimise, suppose we consider the values of U in steps over the possible range within the constraint domain. For example, if the control can be from 0 to 20 in steps of 2, there will be ten such values for U. For each value, the objective function is calculated. Then, the minimal value of the objective function (Figure 7) and its corresponding U can be obtained. That particular U vector is now applied. This needs a step after step fuzzy functional numerical evaluation. After defining a *FIS* for the system, the surfaces are available by calculations (the "*evalfis*" function in MATLAB) made for any one step using the measured process values through the *FIS.*

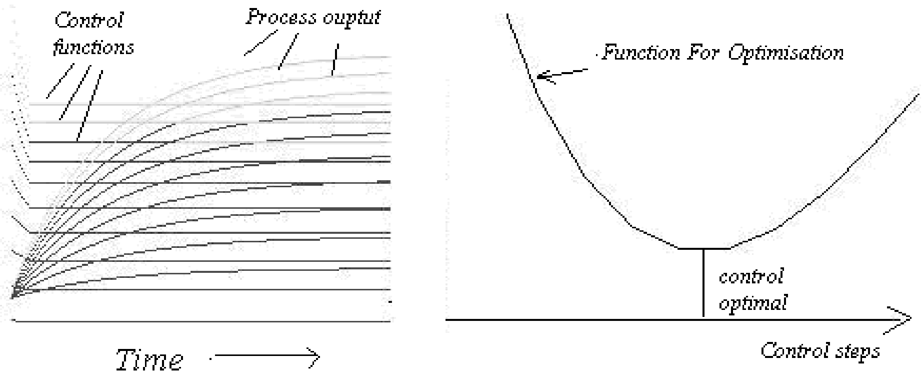
5. Simple Illustration for Nonlinear Model Predictive Control

With a view to elucidate the features of the NMPC for nonlinear optimal control, at first a single input and output system is taken and the method is as follows.

Figure 7 shows starting instant t=0, the calculations of system output y, as exponential-like waveforms. Each curve has a control input u which is also shown. The control input is constant after 10 steps. The control value starts high and decreases in ten steps and further remains same. The light coloured curves indicate that constraints are exceeded, where the control value has exceeded the limit.

For each such control sequence, the cost function is also calculated. There are two components of the cost function. One of the two cost functions relate to the error between the desired value and the actual value. So, the faster the rise to the desired value, the lesser will be cost. Another is due to the control effort u and is a square sum of the

Figure 7: A Typical Set of Response Curves Calculated for Different Decreasing Monotonous Set (up to the Control Horizon) of *U* Values. The Evaluation of the Objective Performance Cost is on the Right, Giving What is the best Control Step



control input values. If high control is applied, there is fast rise and so the first cost decreases, but the control cost increases. So, there is a minimum value of the total cost for a certain control step sequence, which occurs as the fifth one from the bottom in the above calculated curves. The cost function graph is shown on the right. To optimise, we can consider the values of *U* in steps over the possible range within the constraint domain. The steps applied can be any decreasing monotonous set of values commencing from a large step. The idea is that a large initial step helps in reaching the set value with least delay.

The minimal value of the objective function and its corresponding *U* can be obtained. That particular *U* vector is applied.

But instead of searching for the best U in the above sequential ten steps, search algorithms can quickly find the best step. Here, the *U* is a single variable, but in general, *U* vector can have a dimension *n.* For n=2, we have to search over an area, for *n*=3, over a volume and for *n*>3, over a hyper space. Thus, search algorithms such as the Nelder Mead Search (NM) (1965) need to be employed. For this purpose, the NM search is available as a function command 'fminsearch' in MATLAB. The same was modified by Box to include constraints; the program for such is 'fmincon".

There are several variations in the prediction and application steps existing and possible. One or more of these are:

1. The calculation is based on steps applied for k time slots and prediction made for N slots. (*N>k).* Then by optimal choice of control, the first control step alone is applied. The first time step is applied and at the end of that time another estimate is made, noting the process values.
2. The calculation is based on steps applied for k time slots and prediction made for N slots. All k time steps are applied and after that time another estimate is made after noting the process values.

6. Direct Search Fuzzy State Space nonlinear Model Predictive control

In what follows, a method employing the Box's constrained simplex search has been attempted.

1. We first determine the process variable state space formulation as graphical or any other mode of representation. For e.g., if dx/dt = f(x), where f(x) is a nonlinear function of x.

In the state matrix A, we determine which of the terms a_{ij} belong to nonlinear functions, such as, for example, a nonlinear viscous damping in the second order (2 × 2) *A* matrix. Such a nonlinear viscous spring mass damper system is:

$$D^2x + k\, f(Dx) + \omega_n^{\,2}x = \omega_n^{\,2}U \qquad (20)$$

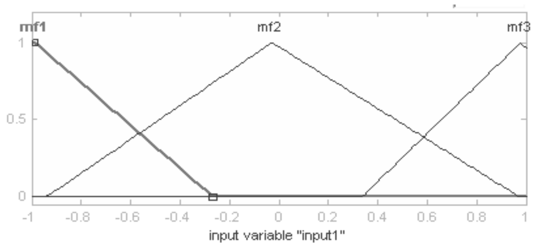
which can be represented in state space form as

$$x' = A\mathbf{x} + Bu$$
$$Y = Cx$$
$$\text{where} \quad \mathbf{x} = [x1\ x2]'$$

The values of *A, B* are based on a comparison with the linear form of the system, which is

$$D^2x + 2\,\xi\,\omega_n Dx + \omega_n^{\,2}x = \omega_n^{\,2}U \qquad (21)$$

Figure 8 a: The choice of Membership functions for the nonlinear damper



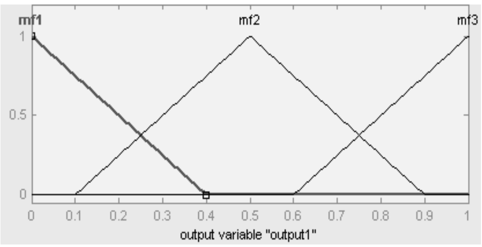
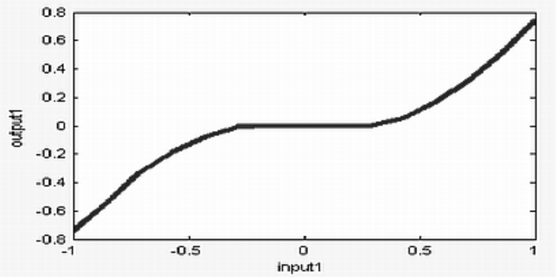


Figure 8 b: The Nonlinear Damping Function Realized by a FIS



The functional relation of the kDx term can be modeled by a fuzzy model relation *f(Dx)*.

All fuzzy models in a FIS will have input and output values usually normalized within the range −1 to +1, and so the factor *k* takes into account the magnitude.

We know that the linear second order equation can be represented in state space form as for example:

$$A = \begin{bmatrix} 0 & 1 \\ -\omega_n{}^2 & -2\xi\omega_n \end{bmatrix} = \begin{Bmatrix} 0 & 1 \\ -16 & -4 \end{Bmatrix} \qquad (22)$$

B = [0 1]’ and C = [1 0]

In a nonlinear system, the matrix

$A = \begin{vmatrix} 0 & 1 \\ -16 & -4 \end{vmatrix}$ becomes partly numbers and partly a function as:

$$Ax = \begin{Bmatrix} 0 & 1 \\ -16 & -f(x_2)/x_{2\max} \end{Bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} \qquad (23)$$

2. We can convert this into a fuzzy model, choosing Membership functions as per choice and determine the rules. The following illustration shows this for a simple nonlinear damper for optimizing a fuzzy performance in fuzzy modeled state space. The *f(x)* which replaces the state equation A*x* is given by a Fuzzy Inference System (FIS), based on a Fuzzy model. The Fuzzy model is prepared as per a Mamdani or Sugeno type, using standard principles in fuzzy logic. The Model and its function *f(x)* may be as given below (Figure 8).

3. Next, we need a **function for the optimisation**. As for the performance index to be met, there are choices: a Quadratic Performance Index (QPI) such as:

i. $J = \int x^T Qx + u^T Ru$ *or*,

ii. $J = \Phi(x) + u^T Ru$, where Φ is a nonlinear relation between performance index and x vector, while the control energy based part is kept as usual, as in *(i)*, *or*

iii. $J = \Phi(x) + \sigma(u)$ having both parts as nonlinear. Note that *u* will contain variables of control [$u1\ u2 \ldots u_c$] vector.

iv. As stated in (ii), the control function part could be taken as a QPI, while the state variable part, which is due to the error in **x** will demand a penalty which might be nonlinear. To illustrate this, it is sketched and modeled by another fuzzy function (Figure 9).

4. Calculate, starting at time step t_0:

i) Assume a control value to start with for *u*, say, u_0.

With this *u*¸ we calculate:

a) The value of d**x**/dt, or incremental dx with the state space relation

d**x**/dt= A{f(**x**)}+*B***u**. For this, one has to do a fuzzy evaluation each time step, where eq.(23) is used.

Figure 9 a: **The penalty function has a nonlinear shape, having small values for low values of the variable x and increasing for large values beyond75%**

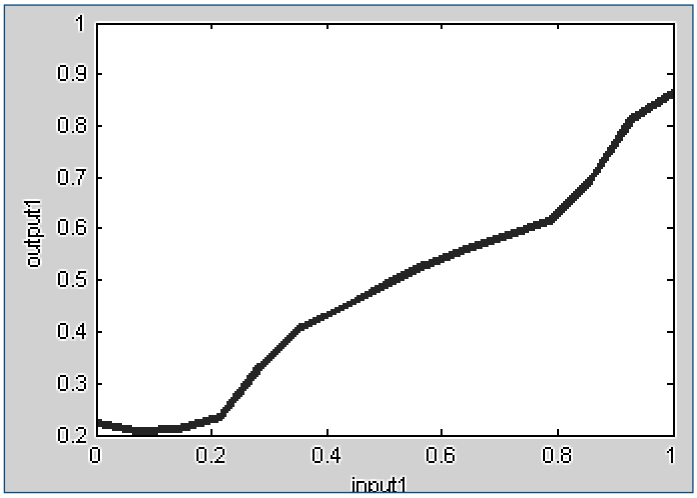


Figure 9 b: **The Fuzzy Membership Functions for x and** *f(x), (input and output)* **Chosen to Meet the Figure9a (I)**

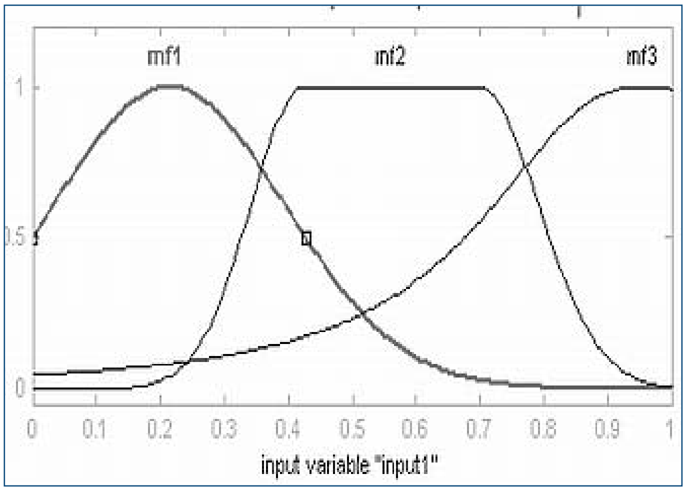
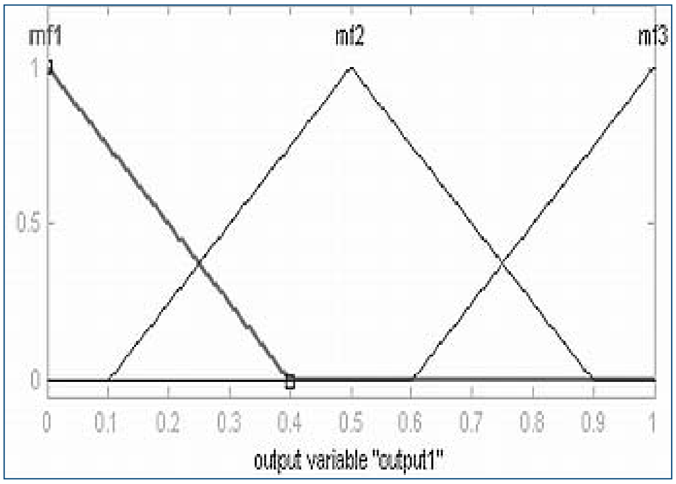


Figure 9 c: **The Fuzzy Membership Functions for x and** *f(x), (input and output)* **Chosen to Meet the Figure9a (II)**



b) Find the new x vector since $\mathbf{x_{new}=x+dx}$.

c) We can calculate the function of performance index, multiply by *dt* and add up to previous sum.

d) Apply a small incremental step to *u* for say, k times, so that u varies by say 35% of initial value in about k steps (k=4, choice made here).

e) Read the actual process value x at each step, use it to determine the dx/dt for each step. Thus repeat calculation *a* to *d* for the k steps.

5. Now we will get a calculated performance Index with this control *u*.

6. We have to now find a ***u*** that minimizes the performance Index by running over the steps from (i) till a minimum in (5) is got.

When it is a two variable x, we employ the Nelder Mead method. This method chooses a simplex of vertices for u; i.e., if we choose a triangle whose vertices are the control vector values. This triangle space is the space of points in the two *u* variables u1, u2. Suppose we would try a 1:10 range for each of the two *u* values, we will have a grid of 100 points. In these hundred points, a Nelder Mead Search finds the point $[u_{1\_opt}, u_{2\_opt}]$, the minimal performance index point.

For example Figure 10 shows how the search points kept changing in the NM method to find the final optimal u vector. The steps are four for prediction period and evaluation time for performance was 100. Then,

$$P.I = \int_0^{100} e^2 f(e)dt + k_{control}u^2 \times 100\Delta t$$

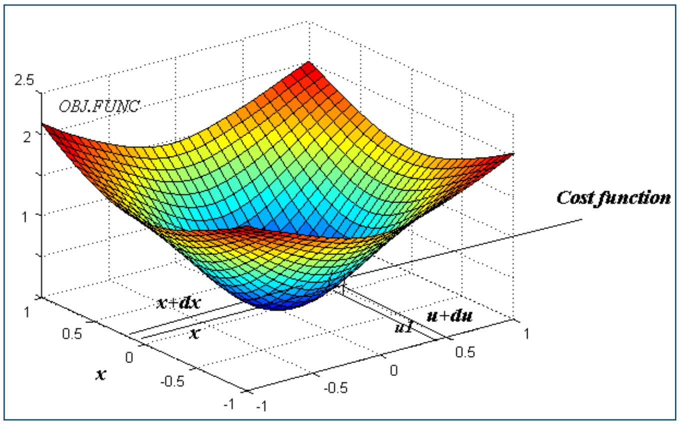
Where *f(e)* is the fuzzy function which is given in the following curve, defined by the fuzzy model *f*. Such an optimisation fuzzy function was given in the text (Figure 9).

The x and y coordinates are the two *u* values, while the z axis shows the calculated Performance index. The actual surface which indicates the minimum point has been found in Figure 10.

It might be noted that, instead of solving a huge set of equations (subject to optimisation and constraints) as given by the eqn. (10-15), the above procedure works out convenient for implementation.

Our method is simple to use and understand and provides a direct solution by sequential integration of the state space equations, even if they are nonlinear. Both fuzzy parameter functions as well as objective functions are handled easily. The programs work based on the models. In an actual plant situation, every step, the x values will be read from the plant. That would take care of disturbances and also dead time delays. These can also be included as a random disturbance to x at each step and using delayed values of x inside the program lines.

Figure 10: The Performance Index Calculated for the Values of U Vector in a 2-D Search Gives a Minimal Point, Which is the Optimal Control Vector



7. Fuzzy S.S., Fuzzy P.I. and Constraints using Box's Method (Matlab's Fmincon Function)

In the procedure for optimal MPC, there are often constraints on state variables which were described in eqn. (16) as inequalities. The Nelder Mead Search will not be able to handle constraints directly. As the search proceeds, one has to examine if the constraints are not violated at each step. The simplex figure which is used in the algorithm can stretch, shrink and move to find the minimal point quickly, but in between the steps, the constraints have to be checked. So, it is convenient to employ the Box's Complex method (Takagi & Sugeno, 1985), as it is known in optimisation techniques. This method has provision to adjust the simplex within the constraint walls of the space. There is a Matlab function called 'fmincon' which provides the implementation. An example of a simple program based on this for the MPC control is included below (Table 1).

In the following programs, we include constraints on the state variables. For this, the Box's modified Search program given by MATLAB's *fmincon* is used. Matlab's Simulink version of MPC cannot be applied with these fuzzy variables and objective functions.

Table 1: Program for Fuzzy modeled System with Fuzzy Performance Index using Box's modification of the Nelder Mead Technique (Matlab 7)

% fa2D is the Dampling Factor of the 2nd order function as fuzzified in Figure 8.

```
subplot(211); subplot(212); clear all;
xinit=[0 ; 0];uinit = 1;
A=[0 1;-16 -3];x=xinit;
for step=1:50
    options=optimset([]);
   LB=[-10; -10]; UB=[12 ; 15]; aineq=[1 ; 1 ]; bineq=[10 ;10];
```

% the following uses the *fmincon* function. The details of use are given in MATLAB.

% the aineq and bineq are the inequalities constraints; the LB and UB are lower and upper bounds. The last 100 denotes the number used for steps in finding performance index using the dd_C1 function. This function calculates the performance at the given set of initial values, doing predictive control using 4 steps and continuing up to 100 steps for evaluating the predicated performance index value.

```
[q1 q2 q3 q4]=fmincon('dd_C1f',uinit,aineq,bineq,[],[],LB,UB,[],options,xinit,100);
tau=50;   xset=10;   q=0;    delta=uinit/10;
predstep=4;
 [k1 k2 k3 y]=dd_C1f(q1,xinit,predstep);
 xinit=k3';     %for next mpc step
 x=k3';u=k2;
% y(t)=x(1);y1(t)=x(2);
%  umean=mean(u(1:10));
    for t=1:predstep
z(t+(step-1)*predstep)=y(t);
w(t+(step-1)*predstep)=u;
    end
end
subplot(211);plot(z);
subplot(212);plot(w)
```

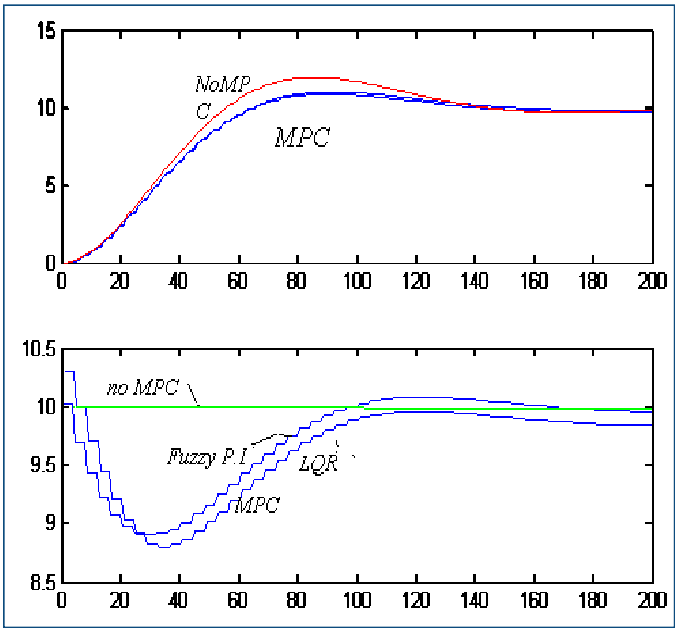
Func dd_c1f:

; the formulation of the function is based on the choice of process variable's fuzzy equations as in Figure9. dd_C1f.m % the fuzzy modeled performance index based function for calculating the response by integrating the s.s. equations and finding the perf. Index figure for the given control value u.

```
function [val,u,x,yf]=dd_C1f(u,x,st)
fa2d=readfis ('fa2d');  %the FIS for the system
fa_PI=readfis('fa_PI');   % the FIS for nonlinear Perf. index
xset=10;K=[1 1];
;A=[0 1; -16 -3];B=[0 1]';A=A-B*K; % To compare with a linear system if required
xa=x;
h=0.01;for t=1:st    % no of steps
xe(1)=(xset-x(1))/(5*xset) ;
xe(2)=x(2)/(5*xset);
%       err(t)=xe(1); err1(t)=xe(2);
y(t)=x(1); y1(t)=x(2);
yf(t)=xa(1);
%  dx=h*(A*xe'+[0 u]')  This is for a linear system
  dg=evalfis((xe),fa2d);  %This uses the Fuzzy Evaluation function for system
  dk=(A(2,1)*x(1) -dg(2)*u)*h;
 dk;
 dq(1)=x(2)*h;
 dq(2)=h*(A(2,1)*x(1) +A(2,2)*x(2));
 dkk=-A(2,2)*x(2)*h;
 dxf=[dq(1) dk]+[0 -A(2,1)*u]*h;
dx=dq+[  0  u*(-A(2,1))]*h;
% dx=A*xe'*h+[0 u]'*h
x=x+dx;    %for linear
xa=x+dxf;  % For Fuzzy system
end
subplot(211)
ya=[y y1]';
plot(y);subplot(212);plot (yf,'r')
% er=xset-yf;s=0;for j=1:st;s=er(j)^2+s;end
```

```
er=xset-yf;s=0;
for j=1:st;
   ers(j)=er(j)*er(j);
        s=evalfis(er(j)/xset,fa_PI)*ers(j)+s;     %Evaluates performance index using Fuzzy inference
end
s=s+u*u/4;
val=s; % This value gives the total performance index for this control value
u;
end
```

Figure11: The Model Predictive Control Step Values Applied in the Bottom Curve and the Response of the Stem to a Step Reference Input of 10



The programs give the idea behind the procedures. The method can be adapted to any system, including interacting MIMO systems also.

8. Conclusion

The programs elucidate the Nelder Mead Search method and the Box's Complex optimisation method, as well as the simple iterative step by step in time evaluation procedure, instead of solving a huge set of equations (subject to optimisation and constraints) as given by the earlier workers. Our method is simple to use and understand and provides a direct solution by sequential integration of the state space equations, even if they are nonlinear. Both fuzzy parameter functions as well as objective functions are handled easily. The programs work based on the models. In an actual plant situation, every step, the x values will be read from the plant. That would take care of disturbances and also dead time delays. These can also be included as a random disturbance to x at each step and using delayed values of ***x*** inside the program lines, but that was not done since there is no point in doing such a blind simulation. Matlab's Simulink version of MPC cannot be applied with these fuzzy variables and objective functions, because it does not support such. The programs give the principles behind the procedures. The method can be adapted to any system, including interacting nonlinear MIMO systems also.

References

Balbis, L., Katebi, R. & Dunia, R. (2005). *Graphical Based Predictive Control Design*. Proceedings of the 2005 IEEE Conference on Control Applications, Toronto, Canada.

Box, M. J. (1965). A new method of constrained optimization and a comparison with other methods. *The Computer Journal*, 8(1), 42-52.

Cutler, C. R. & Ramaker, B. (1980). *Dynamic Matrix Control-A Computer Control Algorithm*. Proceedings of the Joint Automatic Control Conference.

Dhouib, W., Djemel, M., Chtourou, M. (2011). Systems, Signals and Devices (SSD). 8th International Multi-Conference. ISBN: 978-1-4577-0413-0.

Espinosa, J. J., Hadjili, M., Wertz, V. & Vandewalle, J. (1998). *Predictive Control Using Fuzzy Models-Comparative Study*. Proceedings of European Control Conference, Belgium.

Huang, Y. L., Lou, H. H., Gong, J. P. & Edgar, T. F. (2000). *Fuzzy Model Predictive Control*. IEEE Transactions on Fuzzy System, 8(6), 662-668.

Mamdani, E. H. & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-machine Studies*, 7(1), 1-13.

Mazinan, A. H. & Sadati, N. (2010). On the application of fuzzy predictive control based on multiple models strategy to a tubular heat exchanger system. *Transactions of the Institute of Measurement and Control*, August 32(4), 395-418.

Mollov, S., Babuska, R., Abonyi, J. & Verbruggen, H. B. (2004). *Effective Optimization for Fuzzy Model*

Predictive Control. IEEE Transactions on Fuzzy Systems, 12, 661-675.

Nelder, J. A. & Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, **7**(4), 308-313.

Pappa, N., Kaliraj, G. & Shanmugam, J. (2005). Linear and nonlinear model predictive control design for a milk pasteurization plant. *Control and Intelligent Systems*, 33(1), 184-189.

Peña, M., Álvarez, H., Piñón, S. & Carelli, R. (2009). *Fuzzy Model Predictive Control: Simultaneous and Sequential Comparison*. 4th IFAC International Symposium on Intelligent Components and Instruments for Control Applications.

Rawlings, J. B. (2000). *Tutorial Overview of Model Predictive Control*. IEEE Controls Magazine, 28-36.

Roubos, J. A., Mollov, S. R., Babuska, R., & Verbruggen, H. B. (1999). Fuzzy model-based predictive control using Takagi-Sugeno models. *International Journal of Approximate Reasoning*, 22(1/2), 3-30.

Takagi, T. & Sugeno, M. (1985). *Fuzzy Identification of System and its Applications to Modeling and Control*. IEEE Transactions on Systems, Man and Cybernetics, 116-132.