# MULTI-OBJECTIVE GENETIC OPTIMISATION FOR SELF-ORGANISING FUZZY LOGIC CONTROL

M.F. Abbod, M. Mahfouf and D.A. Linkens

University of Sheffield, Sheffield, UK

A multi-objective genetic algorithm is developed for the purpose of optimizing the rule-base of a Self-Organising Fuzzy Logic Control algorithm (SOFLC). The tuning of the SOFLC optimization is based on selection of the best shaped performance index for modifying the rule-base on-line. A comparative study is conducted between various methods of multi-objective genetic optimisation using the SOFLC algorithm on the muscle relaxant anaesthesia system, which includes a severe non-linearity, varying dynamics and time-delay.

Keywords: Fuzzy Logic, Self-organising, Multi-objective optimisation.

## INTRODUCTION

The last decade has seen an upsurge in the development of intelligent control structures over their counterpart model-based control structures due to their success in dealing with complex multivariate uncertain systems without the need of extensive dynamic modelling. The main difficulty in the multivariable case is the interaction between variables and sensitivity to faults in various channels. At the forefront of intelligent control systems technology are Fuzzy Logic Control (FLC), Neural Networks (NN) and Genetic Algorithms (GA) which have all proved to be serious contenders for many other existing forms of control.

In order to map control designs to specific applications, various tuning factors have been appended to these design features, which has a double effect. On the one hand, having a number of tuning factors ('knobs') makes the design attractive to engineers by giving them more flexibility in its application to a wide spectrum of processes. On the other hand, it adds an extra burden of having to find an optimal setting that will reach specific objectives. At this stage, it is worth noting that in most designs, there exist no golden rules for the tuning of such factors. Instead, the user has to rely on his/her intuition and knowledge of the process to find a set of 'good' values necessary to achieve a predefined set of objectives. This task can prove to be tedious and only a compromise solution is adopted whereby an objective is sacrificed in order to satisfy the other objective criteria.

Various synergies are known to exist and as a result have been described in the past between FLC, NN and GA which not only showed that these intelligent structures can interact together but also can make the overall structure more robust against model uncertainties as well as disturbances. For example, the concept of Neuro-Fuzzy Control was shown to work well by producing smoother control than the standard fuzzy control by allowing automatic adjustment of the rule-base and definition of fuzzy sets in terms of widths, peaks, and membership functions. The following sections will attempt to emphasise one such synergy, that between self-organising fuzzy logic control and genetic algorithms by allowing the performance index table to be tuned to an optimal setting using GA techniques which will encompass more than one objective function. It will also be shown that by using this technique, a much reduced size of rule-base can be achieved, in contrast to past experiences where a relatively large number of rules were deemed necessary to achieve an acceptable performance.

## AN INTRODUCTION TO GENETIC ALGORITHMS

Genetic Algorithms (GA) are exploratory search and optimisation methods that were devised on the principles of natural evolution and population genetics. Holland (1) first developed the technique of GA, and several other research studies provided a comprehensive review and introduction of the concept (2). Unlike other optimisation techniques, GA does not require mathematical descriptions of the optimisation problem, but instead relies on a cost-function, in order to assess the fitness of a particular solution to the problem in question. Possible solution candidates are represented by a population of individuals (generation) and each individual is encoded as a binary string containing a well-defined number of chromosomes (1's and 0's). Initially, a population of individuals is generated and the fittest individuals are chosen by ranking them according to an *a priori*-defined fitness-function, which is evaluated for each member of this population. In order to create another better population from the initial one, a mating process is carried out among the fittest individuals in the previous generation, since the relative fitness of each individual is used as a criterion for choice. Hence, the selected individuals are randomly combined in pairs to produce two off-springs by *crossing over* parts of their chromosomes at a randomly chosen position of the string. These new off-springs are supposed to represent a better solution to the problem. In order to provide extra excitation to the process of

generation, randomly chosen bits in the strings are inverted (0's to 1's and 1's to 0's), this mechanism is known as *mutation* and helps to speed up convergence and prevents the population from being predominated by the same individuals. All in all, it ensures that the solution set is never empty. A compromise, however, should be reached between too much excitation and none by choosing a small probability of mutation. There are four well-known reproduction techniques, Generational Replacement (GR), Steady-State (SS), Generational Gap (GG), and Selective Breeding (SB). Only one of these will the subject of this study, i.e. SB, which is described below:

### Selective breeding

Selective breeding reproduction method is designed to overcome some of the deficiencies in the other method. In the steady-state breeding method, a sampling error still occurs in selecting the parents and deletion of individuals from the population, and often good individuals can appear and be deleted without a chance of recombination. Selective breeding introduces determinism in order to eliminate stochastic sampling error in deletion of candidates. The method consists of the following: if the initial population is of '*n*' size, then another population of the same size '*n*' is produced through the mating process. The two populations are combined together to form a population of size '*2n*' which will ranked in the usual manner to produce a population of '*n*' best individuals. It is worth noting that this method has already been found to converge more quickly that most of the other.

## MULTI-OBJECTIVE OPTIMISATION TECHNIQUE

In problems that have multi-objective formulation, objectives are often combined by means of an aggregation function. Combining the objectives to obtain an optimised solution has the advantage of producing a single solution, which requires no interaction with the decision making. However, if the solution found is not acceptable, tuning of the aggregation function is required followed by a new run of the optimiser until a suitable solution is found. The aggregation functions can be as simple as the weighted sum to a target vector. The method functions by generating an initial population which is evaluated to determine the performance of each individual, then an off-spring is generated which in turn is evaluated according to the performance of each individual. The last step is to select the best individual from both generations. Several popular methods exist for producing a single solution to a multi-objective optimisation operation as explained below and their respective performances may differ depending on the problem at hand; these are outlined below:

### Average and distance ranking

The average multi-objective optimisation approach is based on ranking the population according to each objective individually, then a new overall rank can be generated by taking the average of the newly ranked populations. On the other hand, the distance optimisation technique is based on ranking the populations depending on a single objective at a time then taking the square-root of the sum of the squared objective values, and finally ranking the new vector to produce the final generation.

### Pareto ranking

A different approach for multi-objective optimisation is based on ranking according to the actual concept of *pareto optimality* proposed by Goldberg (2). The method guarantees equal probability of reproduction to all non-dominated individuals. If both objectives have the same priority, all the satisfying individuals (the ones which meet their goal) are preferable and have a lower rank than the remaining ones).

## FUZZY LOGIC CONTROLLER (FLC)

Similarly to other control structures such as neural networks, fuzzy logic control has a long history. It stems from the theoretical work of Lotfi Zadeh (3). He proposed the use of fuzzy logic to mimic the human's ability to use imprecise statements to solve complex problems.

The main four components of FLC are fuzzification, knowledge-base, inference engine, and defuzzification. The fuzzification process converts the measured input into a corresponding linguistic value. The knowledge-base comprises the settings of the controller parameters, such as the labels, fuzzy sets shapes and type and number of rules. In this application a Gaussian shape membership function is used for the inputs. Two inputs are considered, the error and the change in error, while the output is calculated using the center of area method. There are 9 control rules which are expressed linguistically in the following form:

***if* error is *x and* change_in_error is *y then* output is *z***

The controller starts with an empty rule-base with constrained inputs and unconstrained outputs. The inputs of the rules are constrained in terms of optimisation of the position and width. The position constraints do not allow a negative labeled rule to be positioned in the positive side, neither do they allow big overlapping of different fuzzy labels. Moreover, the width constraints work by not allowing the fuzzy sets to be too wide or too narrow. The unconstrained output rules allow assignment of the output rules to any label. This has the advantage of giving more flexibility to the controller to generate any shape of control surface.

The learning procedure is to generate the rules and tune them in terms of the input membership function (position and width) and the output of the rules position. Therefore for each rule there are five parameters to be tuned. The membership function of each input and output is defined as follows:

$$\mu(x) = \frac{-\exp(x-c)^2}{2\sigma^2}$$

where c is the peak position and $\sigma$ is the width.

## THE SELF-ORGANISING FUZZY LOGIC CONTROLLER (SOFLC)

The first implementation of a fuzzy controller after Zadeh's seminal paper was followed by the self-organising fuzzy controller (SOFLC) (4) as shown in Figure 1. The controller consists of two levels; the first level is a simple fuzzy controller, while the second level consists of the self-organising mechanism, which acts as a monitor and an evaluator of the controller performance. In the first level, the input signal to the controller is taken at each sampling instant in the form of error and change-in-error. Each signal is mapped to its correspondent discrete level by using the error and change-in-error scaling factors respectively and sent to the Self-Organising Controller (SOC). The SOC, according to control rules issued by the second level, calculates the output with respect to the inputs. The output control signals are scaled to real values using the output scaling factors and sent to the process being controlled. The second level consists of four blocks: the performance index, the process reference model, the rules modifier, and the state buffer. Further details on the design of a SOFLC can be found elsewhere (5) but suffice here to concentrate on the learning part.

The self-organising controller is based on observation of the trajectory of the process to be controlled. Any deviation from the desired trajectory path should be corrected by modifying the rule or rules responsible for the undesired performance.

The performance index functions as an evaluation criterion of the controller performance, In general terms it measures the deviation from the desired trajectory and issues the appropriate correction to the rule that gave the present behaviour. It is derived from linguistic conditional statements by means of using standard fuzzy operations and written in a look-up table form.

As far as the rules modification procedure is concerned, it can be explained assuming that a process has a time-lag of *m* samples, this means that the control action at sample (*nT-mT*) has most contributed to the process performance at the sampling instance *nT*. Thus, if the present instant is *nT*, the modification is made to the controller output *U*, *mT* samples earlier, the rule to be included being:

$$E(nT - mT) \rightarrow CE(nT - mT) \rightarrow U(nT - mT) + P_i(nT)$$

where $P_i(nT)$ is issued by the performance index table, *E* is the error, and *CE* is its derivative.

The key issue with SOFLC is how to select the performance index table. This table is usually selected based on the knowledge of the operator or the expert, but the table is commonly interpreted as a flat surface with curvature on the edges, which ignores the small non-linearities that are located in the middle region of the table. In light of these considerations, the use of GA as a tool for optimising the shape of the table is indeed very attractive. In this work, a GA is used in two ways. 1. To optimise the fuzzy rule-base of a fixed fuzzy *Proportional integral (PI)* controller. 2. To find the best fit for the performance index table by starting with a linear table then repositioning the output of the table with constrained modifications.

## A GENETIC ALGORITHM FOR PARAMETER SELECTION

Coding of the genetic algorithm is based on defining the number in the population and the chromosome length of each one using a concatenated binary mapping. This coding is usually realised by joining segment codes of all the parameters into one composite string. In this study, the GA was set with the following parameters:

*Population size = 30*
*Chromosome = 180*
*Probability of Crossover = 1.0*
*Probability of Mutation = 0.06*
*Number of Generations = 500*
*Fitness Scale = 10 x fitness rank + 100*

The chromosome lengths were selected on the basis of the type of application. For instance, in the case of the SOFLC algorithm, the performance index table includes 25 rules with each rule having only one parameter that need tuning (the output). With 10 bits allocated to each parameter, the performance index rule-base will require a 250-bit chromosome.

As for the control objective, it is defined as the ability to follow the set-point with minimum error. This objective can be expressed in terms of minimisation of the controller performance indices. These include *Integral of Absolute Error* (IAE), *Integral of Square Error* (ISE), and *Integral of Time Absolute Error* (ITAE), as well as minimising the controller effort by calculating the *Integral of controller effort* (ICE). In this study only the IAE and ICE indices are used as will be described below.

## SIMULATION RESULTS

A series of simulations were conducted using GA for optimising the FLC rule-base and the performance index related to the SOFLC algorithm using the optimisation techniques already described in Sections 3.1 and 3.2. As a process test bed we used the muscle relaxation process associated with the drug atracurium (5). The continuous model associated with the drug atracurium is highly nonlinear and is identified to be of the Wiener structure:

$$G(s) = \frac{X_1}{U} = \frac{(1+10.6s).e^{-s}}{(1+4.8s)(1+34.4s)(1+3.1s)}$$

The overall nonlinear model is obtained by combining the above equation with the following Hill equation:

$$E_{eff} = \frac{X_1^{2.98}}{X_1^{2.98} + (0.404)^{2.98}}$$

where $U$ is the drug input, $E_{eff}$ is the actual output (muscle relaxation or paralysis) and $X_1$ is the drug concentration in the blood.

To simulate the above model, a fourth order Runge-Kutta method with fixed step length was used for integration together with a sampling interval period of one minute. A bolus dose of drug was used initially to speed up the response time. Three categories of patients were used depending on their sensitivity to the drug; low, medium and high sensitivity. A training set-point profile of 90% then 80% changed every 70 minutes was used, while a testing profile was chosen to have a set-point change of 95%, 80% and 90% every 70 minutes. The controller used in this series of experiments is of an incremental type (linguistic PI).

The experiment described here used a GA to optimise the performance index table relating to the SOFLC, in an off-line study, using the IAE and ICE as optimising criteria. Figure 2 is a bar chart representing the performance of each algorithm (the non-optimised SOFLC and the optimised SOFLC using the three fitness-ranking methods. Although the *distance ranking method* performed better under the IAE criterion and the *average ranking method* performed better under the ICE criterion, the *Pareto ranking method* was found to lead to a reasonable performance under both objectives. Tables 1 and 2 display the corresponding criteria values under the various regimes for the training and testing set-point profiles.

Finally, Figure 3 shows the performance of the SOFLC when the performance index table was optimised using the fitness *Pareto* ranking method. As seen in Figure 3a the output tracked the output changes efficiently with a reasonable control activity. Moreover, Figure 3c emphasises the nonlinear shape of the control surface.

## CONCLUSIONS

It is widely recognised that for control designs to be flexible, they need to incorporate as many tuning factors ('knobs') as possible to allow them to be tailored to particular applications. Concomitant disadvantages of these tuning factors is the lack of clear guidelines for optimal settings, especially with control designs based on a heuristic approach where stability analyses are either impossible or difficult to carry out. Fuzzy logic control is one of these strategies. One of the adverse effects of this is that a relationship between stability and design tuning factors is not always easy to establish. For instance, it is known that a qualitative rather than a quantitative relationship can be drawn between the parameters of a conventional PID controller and the tuning of a simple PID fuzzy controller. In this paper, we proposed a new method for tuning the performance index table relating to the SOFLC. Future work will include the extension of this work to the multivariable case and the introduction of a fuzzy gain scheduling procedure for selecting the appropriate rule-base based on the initial response of the patient to the initial bolus of drug.

## REFERENCES


1. Holland, J.H., 1973, “Genetic Algorithms and the Optimal Allocation of Trials”, SIAM J. Comput., **2**, pp 89-104

2. Goldberg, D.E. 1989, “Genetic Algorithms in Search, Optimisation and Machine Learning”, Addison-Wesley, Reading, Massachusetts.

3. Zadeh L.A., 1965, “Fuzzy Sets, Information and Control”, **8**, pp 338-353.

4. Procyk, T.J., 1977, “Self-Organising Control for Dynamic Processes”, Unpublished PhD Thesis, Queen Mary College, London.

5. Mahfouf, M. and Abbod, M.F., 1994, “A comparative study between GPC and SOFLC for multivariable anaesthesia”, Chap.4. in Intelligent Control in Biomedicine, D.A. Linkens (Ed.), *Taylor and Francis* Publ., London.

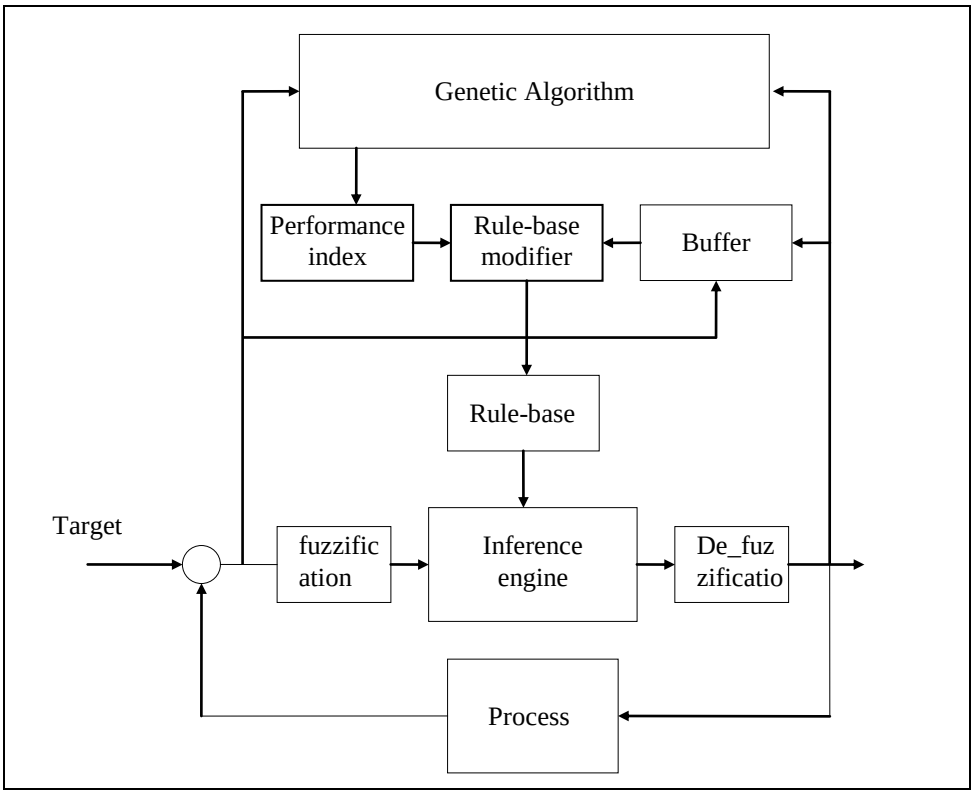


Figure 1: Block diagram of SOFLC with GA learning
FZ: Fuzzification, DFZ: Defuzzification.

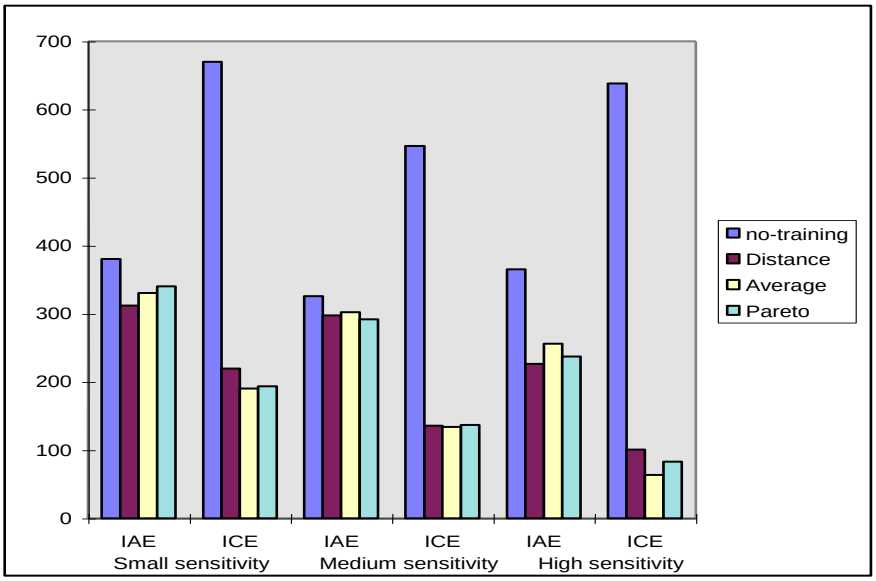


Figure 2: Training error (IAE, ICE) for three patient sensitivities for SOFLC table adjustment

Table 1: IAE and ICE error after training with SOFLC for 3 patient sensitivities.

| Ranking | low sensitivity | | medium sensitivity | | high sensitivity | |
|---|---|---|---|---|---|---|
| | IAE | ICE | IAE | ICE | IAE | ICE |
| no-training | 380.365 | 669.920 | 325.861 | 546.479 | 364.960 | 637.990 |
| Distance | 312.0425 | 219.595 | 297.668 | 135.691 | 226.387 | 100.622 |
| Average | 330.705 | 190.436 | 302.350 | 133.558 | 256.453 | 63.496 |
| Pareto | 340.218 | 193.506 | 291.980 | 136.732 | 237.078 | 83.017 |

Table 2: IAE and ICE error after testing with SOFLC for 3 patient sensitivities.

| Ranking | low sensitivity | | medium sensitivity | | high sensitivity | |
|---|---|---|---|---|---|---|
| | IAE | ICE | IAE | ICE | IAE | ICE |
| no-training | 881.514 | 419.599 | 702.470 | 317.543 | 688.182 | 696.943 |
| Distance | 588.967 | 280.597 | 543.386 | 195.322 | 447.562 | 127.899 |
| Average | 604.4115 | 237.6572 | 524.700 | 171.231 | 454.549 | 101.1101 |
| Pareto | 572.995 | 282.999 | 562.164 | 176.088 | 453.361 | 119.875 |

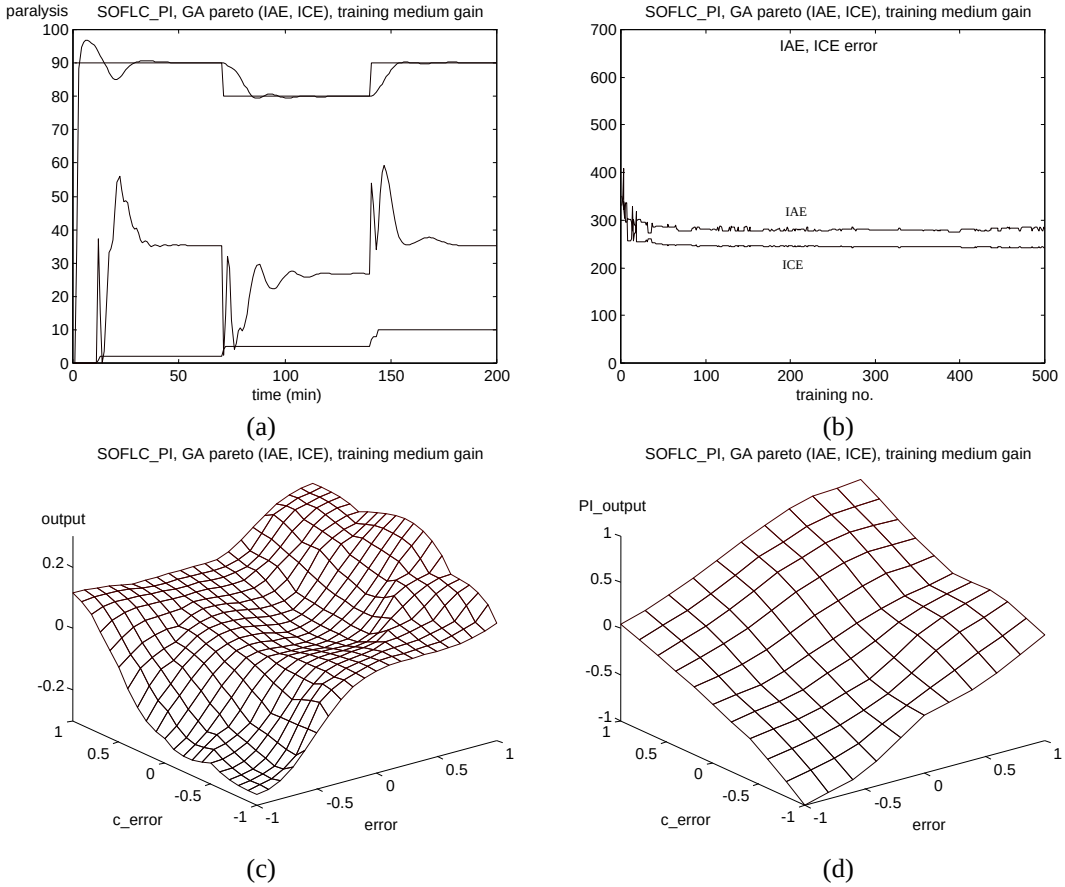


Figure 3: Simulation results of SOFLC using selective breeding and *pareto* multi-objective optimisation (a) simulation of training profile (b) ISE and ICE error minimisation (c) control surface after learning (d) modified performance index