Multi-objective genetic optimisation of GPC and SOFLC tuning parameters using a fuzzy-based ranking method

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

Abstract: A multi-objective genetic algorithm is developed for optimising the tuning parameters relating to the generalised predictive control (GPC) and performance index table of the self-organising fuzzy logic (SOFLC) algorithms, using a multi-objective ranking method based on fuzzy logic theory. A comparative study with more traditional pareto, average and minimum distance ranking methods shows that the proposed method is superior. The study shows that the approach leads to a more effective set of tuning parameters, especially those relating to the important observer polynomial for GPC and to a good reference trajectory for SOFLC. Up to two objective functions were used in the study, although the method can be extended to more objectives. A nonlinear muscle-relaxant anaesthesia model is used as a case study to demonstrate the robustness of the method.

1 Introduction

The number of applications with the popular generalised predictive control (GPC) algorithm [1] has risen considerably since the early years following its introduction. It is reaching a wider audience, including industrialists, academics and clinicians [2]. This is perhaps due to progress in the understanding of the overall issue of design for model-based predictive control (MBPC) in general, followed by an awareness of the criteria governing the choice of the GPC 'tuning parameters' which can be tailored to a specific application.

One of the main issues in earlier work on GPC was that of stability results with respect to the choice of its tuning factors [3]. Such results concerned the minimum output horizon N_1, the maximum output horizon N_2, the control horizon NU and the weighting sequence λ. This was followed by research on the implications of using the CARIMA model, on which GPC is based. In particular, investigations have been made into the problems associated with the selection of the filter polynomial $T(z^{-1})$ (also known as the observer polynomial), which is used to offset the high-filtering characteristics of the Δ operator [4, 5]. Although this filter idea is straightforward, there is another problem associated with it to be overcome; the choice of its order and its cut-off frequency. For example, Robinson and Clarke [4] introduced the notion of stability bounds, with the bound having to be kept as high as possible to ensure robustness.

© IEE, 2000

IEE Proceedings online no. 20000345

DOI: 10.1049/ip-cta:20000345

Paper first received 12th August 1999 and in revised form 20th January 2000

The authors are with the Department of Automatic Control and Systems Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, UK
E-mail: m.mahfouf@sheffield.ac.uk

As GPC is a model-based algorithm, an alternative approach for selecting the above tuning factors may be via various optimisation methods which allow for the search of the best set of parameters suited for the control of the process in question. In this paper, we propose a multi-objective genetic algorithm optimisation method for tuning the maximum output horizon N_2, the control horizon NU and the filter characteristics $T(z^{-1})$. We use two objective functions, which are based on particular characteristics of the process output response, such as the rise time, the ability to reject disturbances effectively, the control activity etc. As there is more than one objective function, an approach based on fuzzy logic is used to rank the various individuals produced between generations in a genetic algorithm (GA) search, which will enable the best individuals to be selected.

The first self-organising fuzzy logic control (SOFLC) scheme was proposed by Procyk and Mamdani [6]. Their scheme includes a policy that can change with respect to the process it is controlling and the environment in which it is operating. An interesting feature of this controller is that it strives to improve its performance until convergence to a predetermined quality. This online improvement in performance is made possible via the specification of a reference trajectory, which uses a set of fuzzy rules written using a qualitative feel for a general monotonic undamped process characterised by a certain amount of overshoot, rise time and settling time. The original rules, formulated by Procyk and Mamdani [6], have proved to be quite robust over a wide range of process dynamics. Their rules have been left practically unchanged, as it is not straightforward to alter them, either individually or in blocks of cells. Similarly to GPC, we propose to optimise these rules via a GA search using two performance indices.

Various synergies between control methods are known to exist. As a result, various schemes have been described where fuzzy logic control (FLC) and GAs have been integrated [7]. This showed that these intelligent structures can interact to form a hybrid system, but also can add more

robustness to the overall control structure in the face of uncertainties, variability of dynamics and disturbances. For example, the concept of genetic-fuzzy control has been shown to work well in producing smoother control than standard fuzzy control, by allowing more flexibility in the automatic adjustment of the rule base and definition of fuzzy sets in terms of widths, peaks, and membership functions [8]. This successful synergy has also been extended to include a scheme where the adaptation mechanism within the SOFLC algorithm is initiated by a model-based predictive control approach, particularly GPC [2].

In this paper, we propose to tune the parameters relating to both popular control algorithms, i.e. GPC and SOFLC, using an optimisation technique based on a multi-objective genetic algorithm. In the optimisation process, two objective functions are used. For ranking the individual candidates, a method based on the theory of fuzzy logic is proposed, which shows its superiority when compared to other commonly used ranking methods such as average ranking, pareto ranking etc.

2 Introduction to GPC and SOFLC

2.1 GPC algorithm

Consider the following locally linearised discrete model in the backward shift operator z^{-1}:

$$A(z^{-1})\Delta y(t) = B(z^{-1})\Delta u(t-1) + C(z^{-1})\zeta(t) \qquad (1)$$

where

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}$$

$$B(z^{-1}) = b_1 + b_2 z^{-1} + b_3 z^{-2} + \ldots + b_m z^{-m+1}$$

$$C(z^{-1}) = c_0 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_p z^{-p}$$

$\zeta(t)$ is an uncorrelated random sequence

$$\Delta = 1 - z^{-1}$$

$u(t)$ represents the control input and $y(t)$ is the measured variable. The controller computes the vector of controls using optimisation of a function of the form

$$J_{GPC} = \sum_{j=N_1}^{N_2}[(P(z^{-1})\hat{y}(t+j) - \omega(t+j))^2] + \sum_{j=1}^{NU}[\lambda(j)(\Delta u(t+j-1))^2] \qquad (2)$$

where N_1 is the minimum costing (output) horizon; N_2 is the maximum costing horizon; NU is the control horizon; ω is the future set point; $\lambda(j)$ is the control weighting sequence; and $P(z^{-1})$ is the inverse model in the model-following context with $P(1) = 1$.

Furthermore, the $C(z^{-1})$ polynomial in eqn. 1 is replaced by a fixed polynomial $T(z^{-1})$, known as the observer polynomial, for the predictions $P(z^{-1})\hat{y}(t+j)$ [1]. As previously mentioned, this allows an offset of the effect of the Δ operator as a high-pass filter on the input-output data.

The minimisation of the cost function described in eqn. 2 leads to the following projected control increment:

$$\Delta u(t) = \tilde{g}^T(W - \Psi), \quad \Psi = [\Psi(t+N_1), \ldots, \Psi(t+N_2)] \qquad (3)$$

where $\tilde{g}^T$ is the first row of the matrix $(G_d^T G_d + \lambda I)^{-1} G_d^T$; $\boldsymbol{G_d}$ is the dynamic (step response) matrix of the form given by Clarke *et al.* [1]; and Ψ is the vector of output predictions up to the horizon N_2.

2.2 SOFLC algorithm

The first implementation of a fuzzy controller [9] after Zadeh's seminal paper [10] was followed by the self-organising fuzzy controller (SOFLC) [6], as shown in Fig. 1. The controller consists of two levels; the first level is a simple fuzzy controller, whereas 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

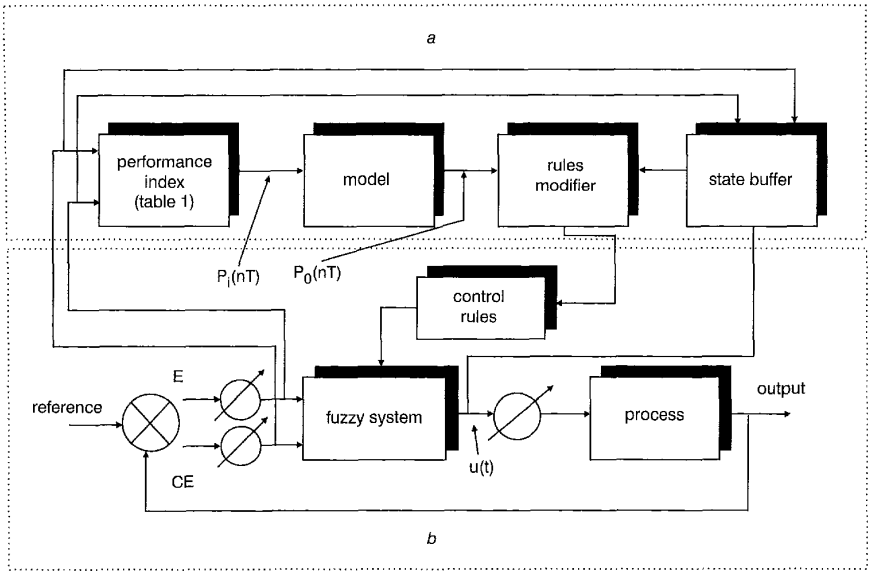

Fig. 1 *Schematic diagram depicting SOFLC*
a Learning part
b Simple fuzzy logic controller part

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). According to control rules issued by the second level, the SOC 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 [11], but here we concentrate on the learning part.

The SOC 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 using standard fuzzy operations and is 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 contributed most 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 is

$$E(nT - mT) \rightarrow CE(nT - mT) \rightarrow U(nT - mT) + P_i(nT) \tag{4}$$

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 (saturated shapes). The table ignores any existing small nonlinearities that are located in the middle region of the table. In light of these considerations, the use of a GA as a tool for optimising the shape of the table is very attractive. In this work, a GA is used to find the best fit for the performance index table, by starting with a linear table and then repositioning the output of the table with constrained modifications. A typical linguistic performance index table is shown in Table 1.

Table 1: Performance index table

Error,	Change in error, CE						
E	NB	NM	NS	ZO	PS	PM	PB
NB	NB	NB	NB	NM	NM	NS	ZO
NM	NB	NB	NM	NM	NS	ZO	PS
NS	NB	NB	NS	NS	ZO	PS	PM
NO	NB	NM	NS	ZO	ZO	PM	PM
PO	NB	NM	ZO	ZO	PS	PM	PB
PS	NM	NS	ZO	PS	PS	PB	PB
PM	NS	ZO	PS	PM	PM	PB	PB
PB	ZO	PS	PM	PM	PB	PB	PB

3 Overview of genetic algorithms

GAs are exploratory search and optimisation methods, devised on the principles of natural evolution and population genetics. Holland [12] first developed the technique of GAs, and other studies have provided a comprehensive review and introduction of the concept [13]. Unlike other optimisation techniques, the GA technique does not require gradients, but instead relies on a function (better known as a 'fitness function') 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 (1s and 0s).

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. 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. Therefore, the selected individuals are randomly combined in pairs to produce an offspring by crossing over parts of their chromosomes at a randomly chosen position of the string. The new offspring is supposed to represent a better solution to the problem.

To provide extra excitation to the process of generation, randomly chosen bits in the strings are inverted (0s to 1s and 1s to 0s). 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.

Thus, for a given population of trials and set of operators, together with procedures for evaluating each trial, a GA proceeds as follows.

(*a*) An initial random population of trials is generated, $\Pi(0) = A_m(0)$, $m = 1, \ldots, M$, where M is the number of trials in the population.

(*b*) For successive sample instances

(i) the performance of each trial, $\mu(A_m(T))$, $T = 0, 1, \ldots$, is evaluated and stored.

(ii) one or more trials are selected, by taking a sample of $\Pi(T)$ using the probability distribution

$$\rho(A_m(T)) = \frac{\mu(A_m(T))}{\sum_{i=1}^{M} \mu(A_i(T))} \tag{5}$$

(iii) one or more genetic operators are applied to the selected trials to produce new offspring, $A_m^o(T)$, $m = 1, \ldots, N$, where N is the number of offspring, which is usually equal to the number of selected trials (parents).

(iv) the next generation of population, $\Pi(T+1)$, is formed by selecting $A_j(T) \in \Pi(T)$, $j = 1, \ldots, N$ to be replaced by the offspring, $A_j^o(T)$: the criterion for selecting which trials should be replaced may be random, on the basis of the least fit or some other fitness basis.

(*c*) The GA process is terminated after a pre-specified number of generations or on the basis of a criterion which determines convergence of the population.

The successful running of a GA involves having to set a number of control parameters, which include population

size, the nature and rates of the recombination operators, crossover, mutation and reproduction. Reproduction is defined as the process through which 'parent structures' are selected to form new offspring, by applying the above genetic operators, which can then replace members of the old generation. The method of selecting an individual to produce offspring (or to be deleted from the population) determines its lifespan and the number of its offspring. For example, if ρ_1 is the probability that an individual $A \in \Pi$ is selected to produce offspring during a sample step, and ρ_2 is the probability that it will be deleted during that sample step, then the expected number of offspring of A is ρ_1/ρ_2 [14].

The most common reproduction techniques are generational replacement (GR), steady-state (SS), generational gap (GG), and selective breeding (SB). Only SB is the subject of study here, as described below.

3.1 Selective breeding reproduction technique

The SB reproduction technique is designed to overcome some of the deficiencies in the other methods. In the SS breeding method, a sampling error still occurs in selecting the parents and deleting individuals from the population, and often good individuals can appear and be deleted without a chance of recombination. SB introduces determinism to eliminate stochastic sampling error in deletion of candidates. The method operates as follows.

(*a*) An initial population $\Pi(0)$ is created in the usual manner.
(*b*) The population is evaluated to determine the performance of each individual, $\mu(A_m, m = 1, \ldots, M)$.
(*c*) For successive generations, thereafter

(i) an entire population of offspring $\Pi^{\circ}(T)$ is produced by selecting parents and applying genetic operators.
(ii) the offspring population is then evaluated.
(iii) the next generation of population is obtained by choosing the best M individuals from both $\Pi(T)$ and $\Pi^{\circ}(T)$.

3.2 Evaluation of trials

Each individual (genotype) in a population is a hypothetical candidate solution to the optimisation problem under consideration. The procedure of evaluating these candidate solutions consists of submitting each to a simulation model and returning an assessment value, according to a given fitness function. A controlled process is defined by a set of state variables $X = \{x_1, x_2, \ldots, x_n\}$, which are controlled by a set of control variables $C = \{c_1, c_2, \ldots, c_m\}$. The genotypes are trial 'control policies' for selecting C as a function of X. The role of the adaptive plan is to derive an optimal policy A_{opt}, which minimises a given performance function J. In the majority of cases, J is determined as a cumulation over time of some instantaneous cost rate:

$$Q[X(t), C(t)], \text{ i.e. } J = \sum_{1}^{T} Q[X(t), C(t)] \qquad (6)$$

For example, the sum of errors over the response trajectory can be directly related to the objective fitness of the trial; this is a measure of the overall worth of a solution and, in the case of this study, that is the performance of GPC or SOFLC algorithms. The configuration of such a learning scheme is shown in Fig. 2.

The control objective is defined as the ability to follow the set point with minimum error. This objective can be

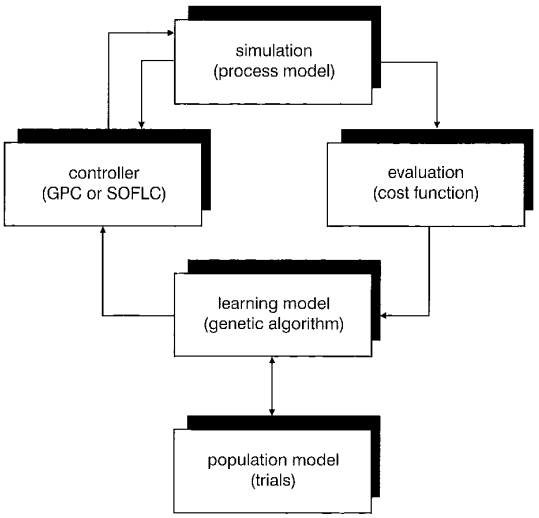

Fig. 2 *Offline learning model*

expressed in terms of minimisation of the controller performance indices. These include integral of absolute error (IAE) plus the integral of absolute value of control effort (ICE), and the integral of time absolute error (ITAE) plus the integral of absolute value of control effort (ICE):

$$IAE + ICE = \int_0^t [\gamma e(t) + |\delta u(t)|]dt \qquad (7)$$

and

$$ITAE + ICE = \int_0^t [\beta |e(t)|t + |\delta u(t)|]dt \qquad (8)$$

$e(t)$ is the error (difference between the measured output and the set point), $\delta u(t)$ is the control increment, and γ, β are arbitrary chosen scaling factors, whose selection is discussed in Section 7. These indices were those used in this study. Note that the idea behind using both objectives at the same time is that the objective function in eqn. 7 tends to minimise just the error and the control effort, especially during the transient period; whereas the sensitivity of objective function in eqn. 8 to errors at a large time is well known, i.e. ineffective rejection of steady-state disturbances is usually heavily penalised by such objective formulation [2].

4 Multi-objective genetic optimisation technique

In problems with 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. An off-spring is then 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, although comparison is only made between the proposed fuzzy ranking method and the pareto ranking method.

4.1 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 one 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.

4.2 Pareto ranking

A different approach for multi-objective optimisation is based on ranking according to the actual concept of pareto optimality proposed by Goldberg [13]. The method guarantees equal probability of reproduction to all individuals. Fig. 3*a* illustrates the ranking of both objectives F_1, F_2 when they have the same priority; all the satisfying individuals, i.e. the ones included in the feasibility area delimited by arbitrary constraints G_1 and G_2, are preferable and have a lower rank than the remaining ones. In the second case (Fig. 3*b*), the second objective is given higher priority i.e. individuals who do not meet the second goal are worse, independent of their performance relating to the first objective.

4.3 Proposed fuzzy population ranking method

As an alternative to the above ranking methods, we propose the following approach for ranking the individuals within a population according to two objective functions.

First, the individuals are all assigned ranks according to their score with respect to each objective function. The ranks, which form the inputs to the fuzzy decision table, are then assigned fuzzy labels ranging from *very low* to *very high*. Five fuzzy labels, *very low, low, average, high, very high* are chosen for each input (each objective), leading therefore to a rule base with 25 rules, as shown in Table 2. A typical rule should read as follows:

> If rank according to objective 1 (eqn. 7) is *'very low'* *and* rank according to objective 2 (eqn. 8) is *'high'* THEN the overall rank is *'low'*.

The number of individuals in a population is chosen to be 34 (Section 5). According to their performance with a particular controller, each individual (solution candidate) will be given a rank from 1 to 34; the lower the ranking, the better this individual will have performed for either control algorithm (GPC or SOFLC). The same set of rules for ranking the individuals is used for both algorithms. The rules are handcrafted and refer to the case study described in Section 5.

Table 2: Fuzzy decision table for final ranking method of individuals within population

		Obj1				
		VL	L	A	H	VH
Obj 2	VL	VL	L	A	H	VH
	L	VL	L	A	H	VH
	A	VL	L	A	H	VH
	H	L	A	H	H	VH
	VH	A	A	H	H	VH

VL = very low, L = low, A = average, H = high, VH = very high,
Obj 1 = rank according to objective function in eqn. 7,
Obj 2 = rank according to objective function in eqn. 8.

A graphical representation of the fuzzy sets relating to the rules of Table 2 is shown in Fig. 4, where the five membership functions are defined to have non-equal spacing to allow for the nonlinear distribution of the fuzzy labels. Furthermore, particular attention has been given when designing the membership function shapes, since the fuzzy sets VL (very low), L (low), and A (Average) have smaller widths than the labels H (High) and VH (Very High). In addition, to accommodate the 50% intersection of two adjacent labels, the membership functions are chosen to be of a Gaussian type with two different widths, i.e. LHS (left-hand side) width and RHS (right-hand side) width. Note also that Gaussian functions allow for smoother inference output than other types of function such as triangular or trapezoidal, and have the advantage of being simply defined using fewer parameters, such as width and peak position. Hence, the membership functions of Fig. 4 are described by

$$\mu(x) = \begin{cases} e^{-\frac{(x-c)}{2\sigma_{LHS}^2}} & \text{if } x \le c \\ e^{-\frac{(x-c)}{2\sigma_{RHS}^2}} & \text{if } x > c \end{cases} \tag{9}$$

c is the peak position and σ is the width of the membership function. Table 3 gives the peak position and width assigned to each label.

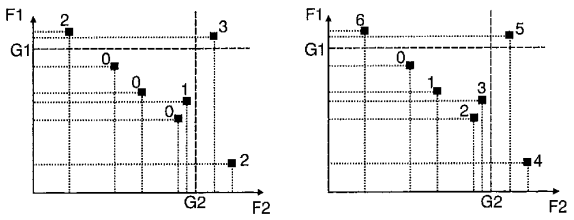

Fig. 3 *Multi-objective pareto ranking with goal values*

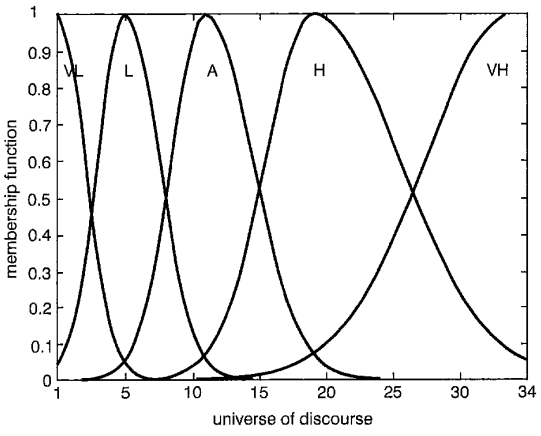

Fig. 4 *Membership functions assignment*

Table 3: Fuzzy labels: width and position

Label	VL	L	A	H	VH
c	1	6	12	20	34
σ_{LHS}	2.5	2.5	3	4	7.5
σ_{RHS}	2.5	3	4	7.5	7.5

VL = very low, L = low, A = average, H = high, VH = very high,
Obj 1 = rank according to objective function in eqn. 7,
Obj 2 = rank according to objective function in eqn. 8.

5 Multi-objective genetic algorithm for GPC tuning factors selection

5.1 Coding of genetic algorithm

Coding of the genetic algorithm is based on defining the number of individuals in the population and the chromosome length of each one, using '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 = 34
chromosome length (bits) = 128
probability of crossover = 0.95
probability of mutation = 0.03
number of generations = 500
fitness scale = 10 × fitness rank + 100

Each individual was then organised into 128 bits, with each block of 32 bits (a lower number of bits can be chosen) representing the following parameters to be optimised: N_2, NU, n and T_c with $T(z^{-1}) = (1 - T_c z^{-1})^n$. Fig. 5 summarises the organisation of the chromosome.

The objective functions in eqns. 7 and 8 were used to evaluate all individuals of the population. However, because the above functions are minimised, the least value returned is the best. Hence, to obtain an objective fitness for the trial evaluations, the orders of magnitude are reversed.

The following minima and maxima were assumed for the GPC tuning parameters:

$$\begin{cases} 6 \leq N_2 \leq 20 \\ 1 \leq NU \leq 5 \\ 1 \leq n \leq 4 \\ 0.30 \leq T_c \leq 0.95 \end{cases}$$

It is worth noting that the coding of the jth parameter in a string of codes represents a value S_{val} between the above predetermined minimum and maximum constraints, which is given by

$$S_{val} = S_{\min} + (S_{\max} - S_{\min}) \frac{\sum_{i=1}^{l_j} \alpha_i 2^{i-1}}{2^{l_j} - 1} \qquad (10)$$

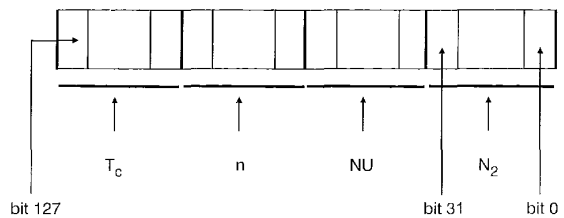

Fig. 5 *Typical coding of GPC tuning factors relating to one individual within population*

α_i are binary code bits, starting with the least significant, and l_j is the code length.

5.2 Simulation results

A series of simulations was conducted for optimising the above tuning factors relating to GPC. As a process case study, we used the muscle relaxation process associated with the drug atracurium [2]. The continuous model associated with the drug atracurium is highly nonlinear and is identified as of the Wiener structure:

$$G(s) = \frac{X}{U} = \frac{(1 + 10.6\,s)e^{-s}}{(1 + 4.8\,s)(1 + 34.4\,s)(1 + 3.1\,s)} \qquad (11)$$

U is the drug input and X is the drug concentration in the blood. The overall nonlinear model is obtained by combining eqn. 11 with the following:

$$E_{eff} = \frac{X^{2.98}}{X^{2.98} + (0.404)^{2.98}} \qquad (12)$$

E_{eff} is the actual output (muscle relaxation).

To simulate the above model represented by eqns. 11 and 12 (which acts as the process block as shown in Fig. 1), a fourth-order Runge-Kutta method with fixed step length was used for integration, together with a sampling interval period of one minute. The GPC algorithm assumed default parameters of $N_1 = 1$, $\lambda = 0$, and $P(z^{-1}) = 1$. For parameter estimation, a UD factorisation version of the well known RLS algorithm was triggered at time 0 on incremental filtered data. An initial covariance matrix of $\text{cov}_i = 100.I$ was used, where I is the identity matrix, with a forgetting factor of $\rho = 0.97$. A third-order model with $3\hat{a}s$ and $3\hat{b}s$ was estimated, with the first $\hat{b}$ parameter initialised at 0. The following training set point profiles were used:

(*a*) 80% for the first 100 samples
(*b*) 95% for the next 100 samples
(*c*) 90% until sample 250
(*d*) 95% until sample 300

A noise sequence of 1% and a disturbance of 5%, in the form of an output change at time 70 min, were added to the system.

The experiment used a GA to optimise the tuning parameters relating to GPC (N_2, NU and $T(z^{-1})$), in an offline study, using the indices in eqns. 7 and 8 as optimising criteria. The average, minimum distance, and pareto ranking methods were compared with the proposed fuzzy ranking method, as explained below. Note that as far as eqns. 7 and 8 are concerned, values of $\gamma = 0.20$, $\beta = 0.40$ were used.

Table 4 shows results from runs conducted on the model of eqns. 11 and 12. All ranking methods led to more or less the same output horizon and control horizon. However, as far as the filter characteristics are concerned, the distance

Table 4: Objective functions and optimised GPC parameters using different ranking methods for multi-objective genetic algorithm

Ranking methods	Objective functions		N_2	NU	n	T_c
	IAE + ICE	ITAE + ICE				
Average	2980	24820	6	1	2	0.72
Distance	2670	25430	8	1	1	0.82
Pareto	2820	25010	7	1	2	0.62
Fuzzy	3260	24450	6	1	2	0.56

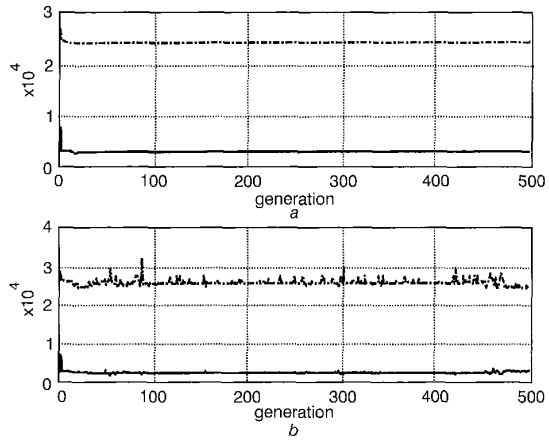

Fig. 6 *Variations of objective criteria with number of generations;*
a Using fuzzy ranking method
b Using average ranking method
—— IAE + ICE
–·–· ITAE + ICE

ranking method failed to agree with previous guidelines on how to select the order and filter cut-off frequency. These guidelines recommend that the order should be at least equal to 2 to hasten the roll-off, and the filter time constant should be fast enough to reject disturbances effectively [2]. The hand-crafted fuzzy rules, which were easy to obtain, have succeeded in reaching a reasonable compromise for the GPC parameters, by producing a fast enough response ($N_2 = 6$) with good filter characteristics ($T(z^{-1}) = (1 - 0.56z^{-1})^2$). Furthermore, the evolution of the objective criteria in time, shown in Fig. 6*a* suggests that the fuzzy strategy led to more settled criteria values than those obtained using the average ranking strategy, where a compromise was being sought from one generation to the next, as seen in Fig. 6*b*.

Fig. 7 shows the result of a run conducted in closed loop using GPC, with the optimised tuning factors obtained using the fuzzy ranking approach. As can be seen, despite the noise and disturbance, the controller displayed reasonable control activity, with the output tracking the targets effectively.

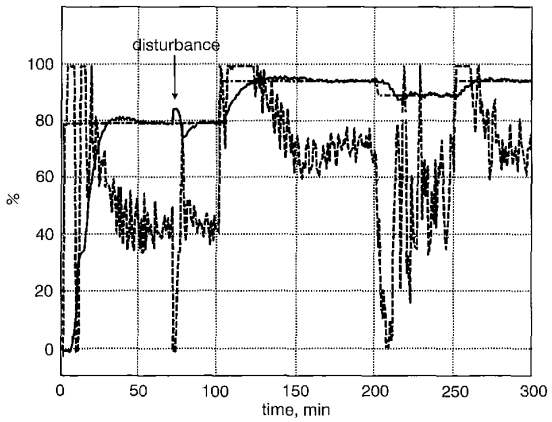

Fig. 7 *Closed-loop control under GPC using multi-objective optimised settings from fuzzy ranking method*
—— output
–·–· target
– – – input

6 Multi-objective genetic algorithm for SOFLC design

6.1 Coding of genetic algorithm

As mentioned above, the fuzzy rules included in the performance index of most SOFLC algorithms are similar to those originally formulated by Procyk and Mamdani [6]; in other words, a rule base containing at least 56 fuzzy rules. In this study, we aim to use a considerably reduced number of rules; 25 rules in total, which incidentally has to be the same number of rules for the low-level simple fuzzy controller: this reduced number of rules will still make the controller retain its adaptive capabilities without loss of performance. Traditionally, the structure of a fuzzy rule is made up of the coding of the fuzzy sets of the linguistic predicates of the rule. As a linguistic variable is defined by its membership function, which is determined by the position of its peak and the width (non-zero membership grades), each linguistic fuzzy variable can be represented by a two-part code: one for the position p and the other for the width w [8]. However, note that, at most, a third part can be used by adding a linguistic value or name l [8].

As far as the performance index (PI) table (1) is concerned, only the consequent part (the output, P_p) needs tuning, since it is this part that plays the role of the adaptive mechanism. Furthermore, because the PI table is configured as a 'look-up table', singleton values are used [2, 11]. Hence, Fig. 8 shows a typical format for coding a population of fuzzy rules within the PI table. Note that the use of a 'look-up table' configuration in SOFLC has the advantage of reducing the computational burden.

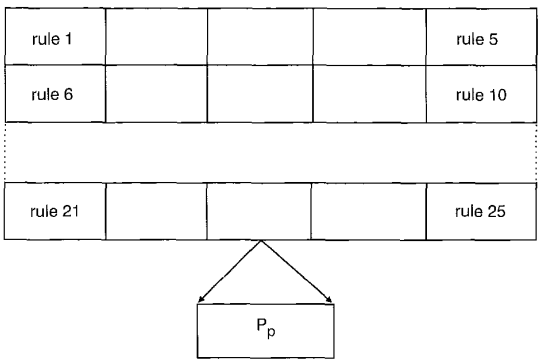

Fig. 8 *Coding of rule base relating to PI table of SOFLC*

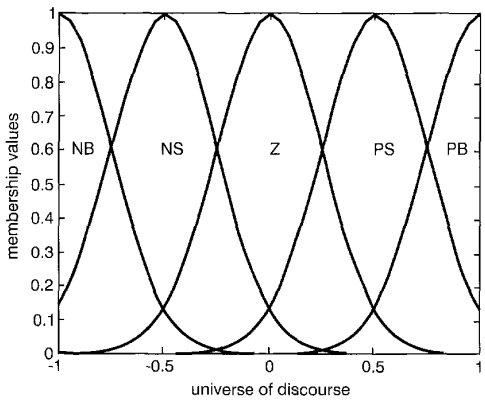

Fig. 9 *Definition of fuzzy sets*

Table 5: (IAE + ICE) and (ITAE + ICE) criteria for testing conditions with SOFLC for three patient sensitivities

Ranking	Low sensitivity		Medium sensitivity		High sensitivity	
methods	IAE + ICE	ITAE + ICE	IAE + ICE	ITAE + ICE	IAE + ICE	ITAE + ICE
No training	844.302	35340.299	715.292	25403.657	603.874	19873.367
Pareto	735.664	25624.442	633.833	21739.281	557.371	19267.719
Fuzzy	682.670	23608.763	567.921	20337.059	536.611	18013.472

For our research, the low-level simple fuzzy controller assumed five fuzzy sets; negative big (NB), negative small (NS), zero (Z), positive small (PS), and positive big (PB), which have been defined in a universe linearly partitioned as shown in Fig. 9.

In addition, the GA was set with the following parameters:

population size = 34
chromosome length (bits) = 250
probability of crossover = 0.95
probability of mutation = 0.06
number of generations = 500
fitness scale = 10 × fitness rank + 100

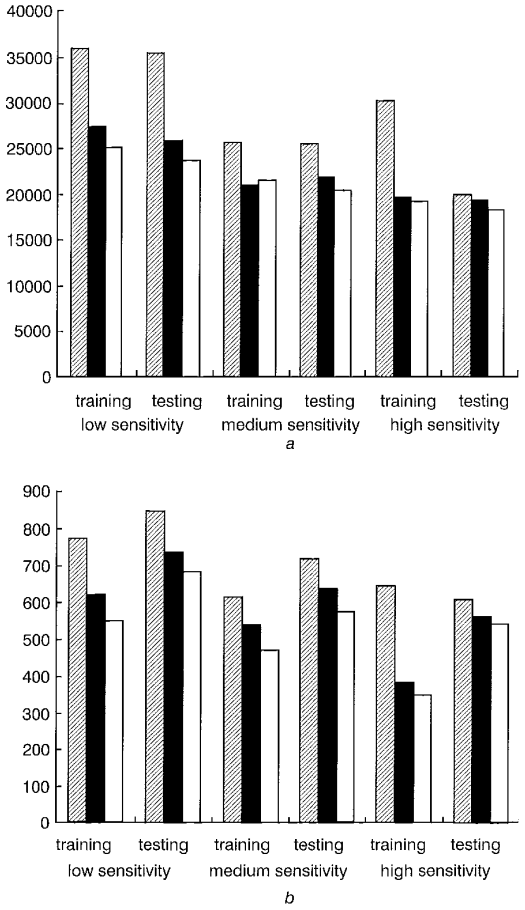

Fig. 10 *Training and testing error criteria for three patient sensitivities for SOFLC PI adjustment*

a ITAE + ICE
b IAE + ICE
▨ no training
■ pareto
□ fuzzy
l = low sensitivity
m = medium sensitivity
h = high sensitivity

The PI table includes 25 rules, with each rule having only one parameter that needs tuning (the output of the table). With 10 bits allocated to each parameter, the PI rule base will require a 250-bit chromosome.

6.2 Simulation results

For this series of experiments, the model is described by eqns. 11 and 12. A bolus dose of the drug was used initially to speed up the response time. Three categories of patient were used, depending on their sensitivity to the drug; low, medium and high sensitivity. A training set point profile of 90%, then 80%, was used and changed every 70 min, and a testing profile was chosen to have a set point change of 95%, 80% and 90% every 70 min. Note that the use of different set point profiles for training and testing

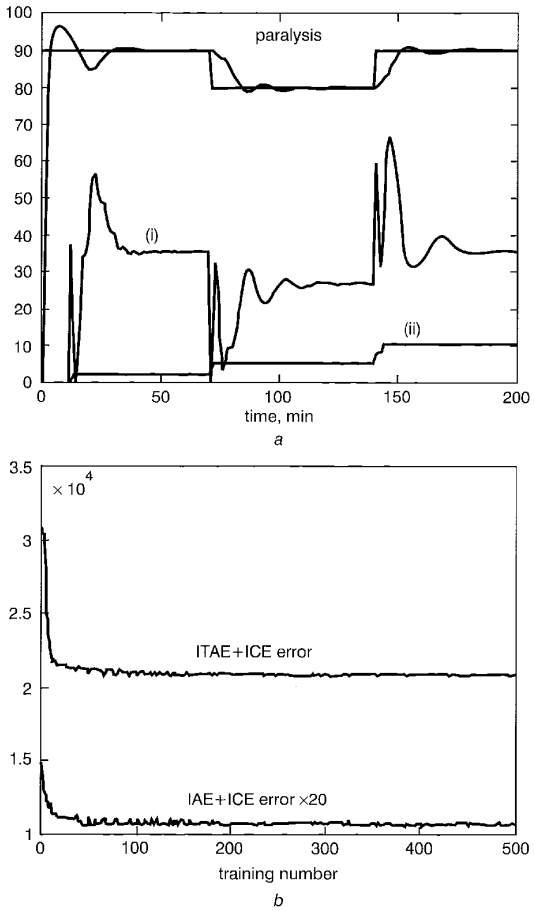

Fig. 11 *Simulation results of SOFLC using multi-objective genetic optimisation with pareto ranking*

a Simulation run using training set point profile
b Objective criteria
(i) drug input, (ii) number of rules

has the advantage of indicating how robust a particular control strategy is. The controller used in this series of experiments was of an incremental type (linguistic PI). We used a GA to optimise the PI table relating to the SOFLC, in an offline study, using the indices in eqns. 7 and 8 as optimising criteria. We used three configurations for the SOFLC algorithm: a SOFLC with a standard PI table [11] (no training); a multi-objective GA-optimised PI table with pareto ranking; and a multi-objective GA-optimised PI table with fuzzy ranking.

Table 5 shows results from runs using three categories of patient, according to their sensitivity to the drug; low sensitivity, medium sensitivity, and high sensitivity.

Figs. 10*a* and *b* illustrate the results for the ITAE + ICE and IAE + ICE criteria. They clearly show that the proposed fuzzy ranking method led to lower values for the above objection functions than the other methods for ranking. Figs. 11–16 show the results of runs for a medium-sensitive patient (medium gain) using the pareto ranking of individuals within a population of 34 and the proposed fuzzy ranking method, respectively.

Comparing Figs. 11*a* and 14*a* for the training set point profile, and Figs. 13 and 16 for the testing set point profile, it can be seen that, although the performances appear to be similar, a close examination reveals that the output response in the fuzzy ranking case tracked the set point with a faster rise time, especially during the time phase 140–200 min.

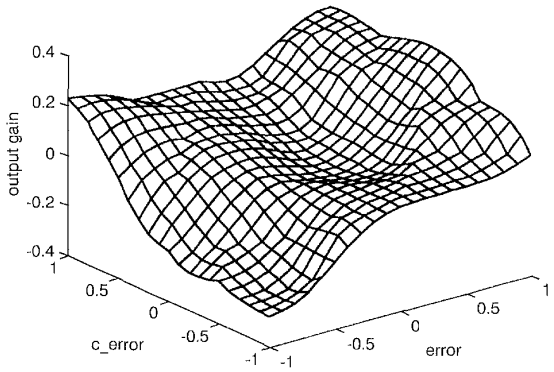

Fig. 12 *Simulation results of SOFLC using multi-objective genetic optimisation with pareto ranking*

a Control surface after learning
b Modified PI surface

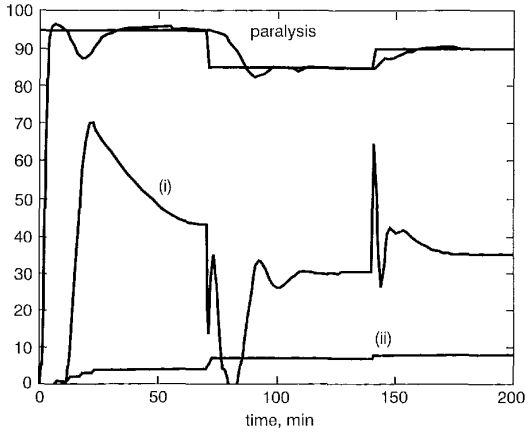

Fig. 13 *Simulation results of SOFLC using multi-objective genetic optimisation with pareto ranking*

Simulation using training set point profile
(i) drug input, (ii) number of rules

Furthermore, Fig. 15 *a* shows a rule base with a flatter surface around the centre than that of Fig. 12*a* (pareto ranking method). A similar pattern was also obtained with low and high-sensitive patients, although this is not shown here.

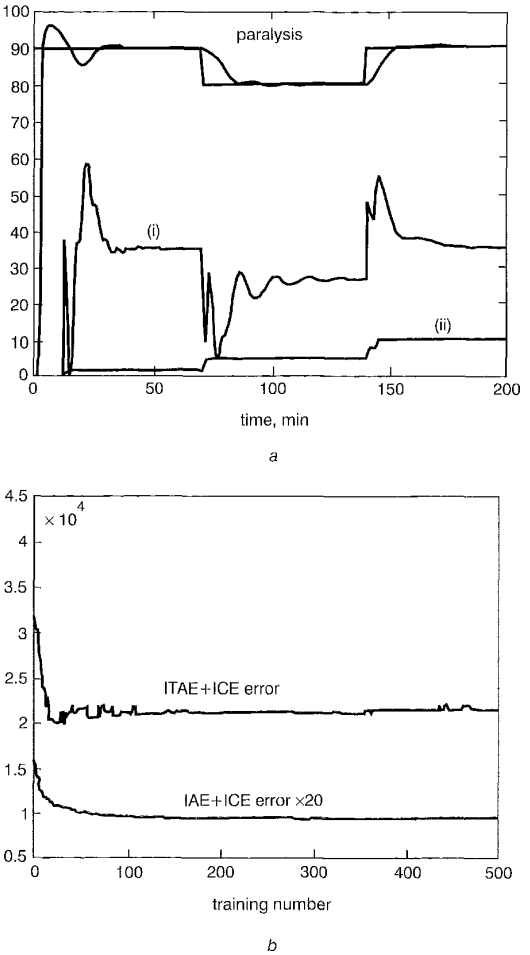

Fig. 14 *Simulation results of SOFLC using multi-objective genetic optimisation with fuzzy ranking*

a Simulation run using training set point profile
b Objective criteria
(i) drug input, (ii) number of rules

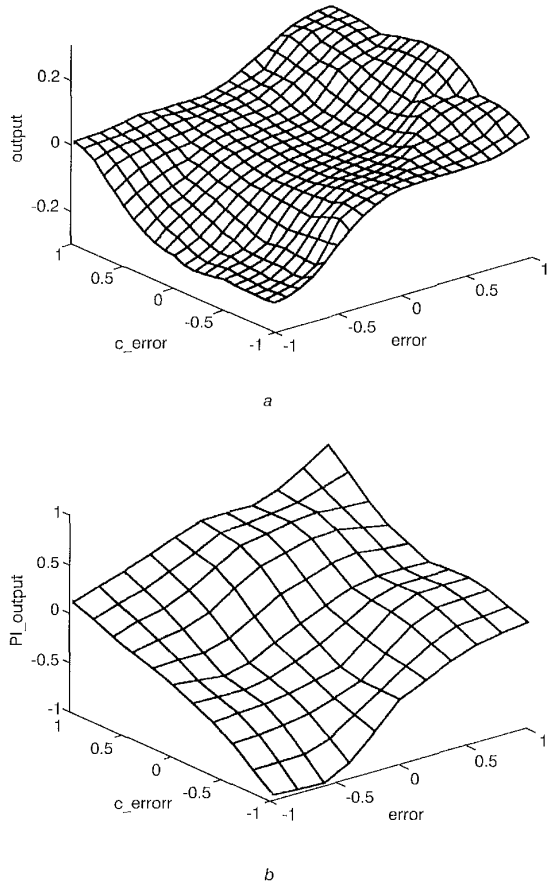

Fig. 15 *Simulation results of SOFLC using multi-objective genetic optimisation with fuzzy ranking*

a Simulation run using training set point profile
b Objective criteria

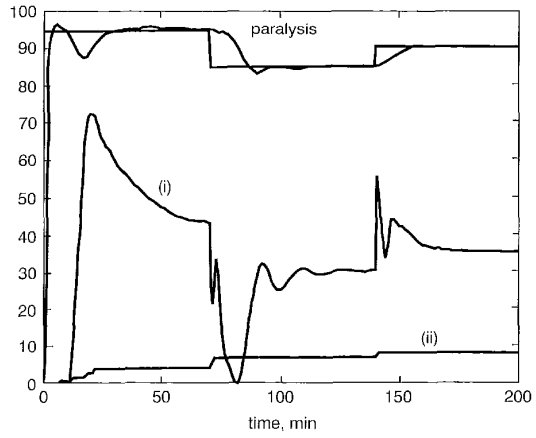

Fig. 16 *Simulation results of SOFLC using multi-objective genetic optimisation with fuzzy ranking*

Simulation using training set point profile

7 Conclusions

It is widely recognised that, for control designs to be flexible, they need to incorporate tuning factors ('knobs') to allow them to be tailored to particular applications. Concomitant disadvantages of a relatively large number of tuning factors are the lack of clear guidelines for optimal settings. In this study, we have looked at two adaptive control schemes, which although similar in philosophy, can be classified as two different paradigms: one is a mathematical model-based technique widely known as the generalised predictive control (GPC) algorithm; the other is a fuzzy linguistic based technique known as the self-organising fuzzy logic control (SOFLC) algorithm. A previous comparative study [2] has highlighted strengths and weaknesses of the two paradigms in a sensitive environment such as muscle relaxation therapy and anaesthesia.

As far as GPC is concerned, stability theorems relating to the choice of these factors have already been reported [3] and they mainly relate to linear systems, whereas in this study we are concerned with highly nonlinear systems. Coupled with these results is the better understanding over the years of their influence on performance and robustness, which make the whole GPC design so attractive to academia and industry alike [2]. In this research, we have proposed a method of tuning the GPC parameters using multi-objective GAs. This study follows other studies with the same agenda but using different tools, such as the analysis via stability margins [3–5]. For SOFLC, we focused on the optimal tuning of the reference trajectory, which is normally represented by a performance index (PI) table. Procyk and Mamdani [6] proposed the original PI table, which was the same table used in many engineering applications. We used the optimisation technique as above to find the optimal settings for the table. The problem is simple in single objective optimisation and becomes more 'tricky' when two or more objectives are specified.

Indeed, in multi-objective optimisation, the concept of optimality becomes relative as a number of potential best candidates are proposed, rather than only one best candidate, for the optimisation problem. As a result, a number of ranking methods have been proposed, including the, pareto, distance and average ranking methods, with one common feature; to decide which candidate is the best. The main difficulty we found with the pareto ranking method was the choice of the constraints boundaries to decide on the feasibility constraints, other than a dry run using another type of ranking method (average ranking method for example) which helps decide on these boundaries. As far as the average and distance ranking methods are concerned, we believe that the final rank they generate is so crisp that the decision seems almost too rigid. Therefore, it does not represent the same judgement that humans would make if presented with the same knowledge of the data.

Hence, we have proposed a method of ranking based on fuzzy reasoning. The idea seems reasonable, especially in the SOFLC case where the control concept is fuzzy anyway. The idea consists of ranking the individual of a population according to each objective function. These ranks are then fuzzified and fed into a fuzzy decision table with rules, which are built using human judgement. Note that this idea can easily be extended to include more than two objective functions using nested *if then* rules. The advantages of this method include various features of ordinary decision-making, i.e. averaging, minimum distance, weighting etc.

One important issue in any optimisation problem is the selection of the fitness function. As far as this study is concerned, we chose the $(IAE+ICE)$ and $(ITAE+ICE)$ criteria, as defined by eqns. 7 and 8. The choice of these two functions is the trade-off between obtaining a relatively fast response with a reasonable control activity. For example, in the GPC case, the control horizon was allowed to

vary up to maximum value of 5, which usually makes the control signal highly active, especially in the stochastic case. Hence, there was a need to use low weighting factors for *IAE* and *ITAE* in eqns. 7 and 8 with respect to the *ICE;* we used 0.20 and 0.40, respectively. Note that we did not find the proposed method sensitive to this choice, as long as the right priorities between *IAE* or *ITAE* and *ICE* in the above objectives are taken into account.

For GPC parameter tuning, we chose the prediction horizon N_2, the control horizon NU and the filter polynomial $T(z^{-1})$. The proposed optimisation tool succeeded in obtaining a compromise set for these parameters. It is surprising, but also encouraging, that the order of the obtained filter is higher than 1, which increases the frequency roll-off; incidentally, such a filter order always leads to a more robust control structure, as far as disturbance rejection and unmodelled dynamics are concerned [4]. Particular attention must be paid to the choice of the lower and higher boundaries of these tuning factors. For example, to avoid GPC reverting to a minimum variance controller ($N_2 = NU$), the higher boundary of NU should be chosen to be less than the lower boundary of N_2. In addition, the multi-objective GA optimiser should be run in stochastic mode, with a possible injection of sudden disturbances, which has the advantage of obtaining a good $T(z^{-1})$ filter polynomial as possible.

As far as SOFLC is concerned, the objective was to use a small number of rules in the PI table, without the loss of the reference trajectory properties of the table itself. The tool optimiser succeeded in reducing the number of rules to 25 from the original 56. Simulation studies on a difficult nonlinear process, i.e. the muscle relaxation process, showed that the performance obtained was still good; in other words, the low-level simple fuzzy logic controller was guided by a good reference trajectory with a much reduced number of rules. Future work will concentrate on the use of this scheme within a multi-input multi-output (MIMO) environment with up to eight objective functions that have to be carefully chosen, in order to reflect a particular performance. Such MIMO environment, in our case, is the realm of anaesthesia, which includes muscle relaxation (the subject of this study), unconsciousness, and analgesia (pain relief) together with eight 'quality' factors to optimise.

8 Acknowledgment

The authors acknowledge financial support for this work from a Leverhulme Trust Research Grant.

9 References

1 CLARKE, D.W., MOHTADI, C., and TUFFS, P.S.: 'Generalised predictive control-part I and II', *Automatica*, 1987, **23**, pp. 149–160
2 MAHFOUF, M., and LINKENS, D.A.: 'Generalised Predictive Control and Bioengineering', (Taylor and Francis Publishers, 1998)
3 CLARKE, D.W., and MOHTADI, C.: 'Properties of generalised predictive control', *Automatica*, 1989, **25**, pp. 859–875
4 ROBINSON, B.D., and CLARKE, D.W.: 'Robustness effects of a prefilter in generalised predictive control', *IEE Proc., Control Theory Appl.*, 1991, **138**, pp. 2–8
5 MCINTOSH, A.R., SHAH, S.L., and FISHER, D.G.: 'Selection of tuning parameters for adaptive generalised predictive control'. Proc. American Control Conf., 1989, Pittsburg, USA, pp. 1828–1833
6 PROCYK, T.J., and MAMDANI, E.H.: 'A linguistic self-organising process controller', *Automatica*, 1979, **15**, pp. 15–30
7 PEDRYCZ, W. (Ed.): 'Fuzzy evolutionary computation' (Kluwer Academic Publishing, Boston, USA, 1997)
8 LINKENS, D.A., and NYONGESA, H.O.: 'Genetic algorithms for fuzzy control', *IEE Proc. Control Theory Appl.*, 1995, **142**, (3), pp. 161–176
9 MAMDANI, E.H.: 'Applications of fuzzy algorithms for control of a simple dynamic process', *IEE Proc. Control Theory Appl.*, 1974, **121**, pp. 1585–1588
10 ZADEH, L.A.: 'Fuzzy sets, Inform. Control' 1965, **8**, pp. 338–353
11 MAHFOUF, M., and ABBOD, M.F.: 'Self-adaptive and self-organising control applied to nonlinear multivariable anaesthesia: a comparative model-based study' *in* 'Intelligent control in biomedicine' LINKENS, D.A. (Ed.) (Taylor and Francis Publishers, 1994) Chap. 4, pp. 79–132
12 HOLLAND, J.H.: 'Genetic algorithms and the optimal allocation of trials', *SIAM J. Comput.*, 1973, **2**, pp. 89–104
13 GOLDBERG, D.E.: 'Genetic algorithms in search, optimization and machine learning' (Addison-Wesley Publishers, 1989) ISBN 0-201-15767-5
14 HOLLAND, J.H.: 'Adaptation in natural and artificial systems' (Addison Wesley Publishers, 1975)