Accelerating Multi-Objective Control System Design Using a Neuro-Genetic Approach

N. M. Duarte, A. E. Ruano and C.M. Fonseca
Unit of Exact Sciences & Humanities
University of Algarve, Portugal

P. J. Fleming
Department of Automatic Control and Systems Engineering
University of Sheffield, Sheffield, UK

Abstract – Designing control systems using multiobjective genetic algorithms can lead to a substantial computational load as a result of the repeated evaluation of the multiple objectives and the population-based nature of the search. Here, a neural network approach, based on radial basis functions, is introduced to alleviate this problem by providing computationally inexpensive estimates of objective values during the search. A straightforward example demonstrates the utility of the approach.

1 Introduction

Many problems arising in control and systems engineering require the simultaneous optimisation of multiple, often conflicting, design criteria, such as performance, reliability, and cost (Fig. 1). Unlike in single-objective optimisation, the global solution to such problems is seldom a single point, but a family of compromise solutions known as the Pareto-optimal set, such as illustrated by the trade-off surface in Fig. 1. These solutions are optimal in the sense that improvement in any objective can only be achieved at the expense of degradation in at least one of the remaining objectives.

Fonseca and Fleming (1993) proposed a multiobjective genetic algorithm approach to solving this problem and presented a detailed account of its development and application in Fonseca and Fleming (1998a, 1998b). This approach is outlined in this paper in Section 2. Inevitably, owing to the existence of multiple objectives and the population-based nature of the search, its application can result in a considerable computational burden for complex problems. In Section 3, a neural network-based approach is proposed to alleviate this burden through efficient estimation of objectives. This approach is demonstrated in Section 4 on a simple control system example and shown to be effective.

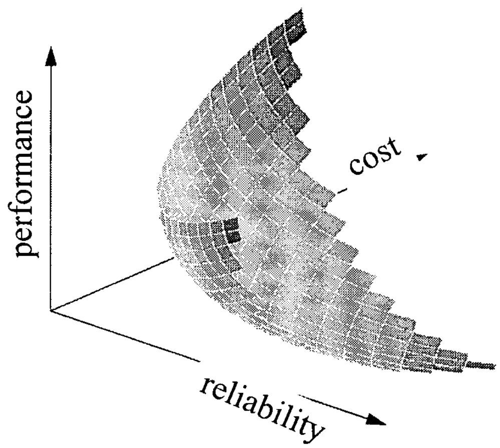

Fig. 1 Trade-off surface depicting competing system performance objectives

2 MultiObjective Genetic Algorithm (MOGA)

2.1 Multiobjective optimisation.

Consider the following multiobjective optimisation (MO) design problem:

$$\min \mathbf{F}(\mathbf{p}) \qquad \ldots(1)$$
$$\mathbf{p} \in \Omega$$

where $\mathbf{p}=[p_1, p_2, \ldots, p_q]$, Ω defines the set of q free variables, **p**, subject to any constraints and $\mathbf{F}(\mathbf{p}) = [f_1(\mathbf{p}), f_2(\mathbf{p}), \ldots, f_n(\mathbf{p})]$ are the design objectives to be minimised.

Clearly, for this set of functions, **F(p)**, it can be seen that there is no one ideal `optimal' solution, rather a set of Pareto-optimal solutions for which an improvement in one of the design objectives will lead to a degradation in one or more of the remaining objectives. In Fig.2 there are two objectives, f_1 and f_2, to be simultaneously minimised. These objectives are competing with one another such that there is no single solution. Candidate solution point A has a lower value of f_2, but a higher value of f_1, than candidate

0-7803-6375-2/00/$10.00

solution point B. Thus, it is not possible to state that one point on the trade-off curve shown in Fig. 2 is better or worse than another. Such solutions are known as Pareto-optimal solutions (alternatively as *non-inferior* or *non-dominated* solutions) to the multiobjective optimisation problem.

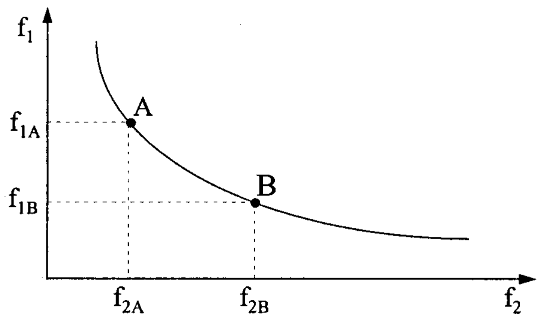

Fig. 2 Pareto-optimal set of solutions for 2-objective problem

Hitherto, members of the Pareto-optimal solution set have been sought through solution of an appropriately formulated non-linear programming (NP) problem. A number of approaches are currently employed including the ε-constraint, weighted-sum and goal attainment methods (Hwang and Masud, 1979). However, such approaches require precise expression of a, usually not well understood, set of weights and goals.

If the trade-off surface between the design objectives is to be better understood, repeated application of such methods is necessary. In addition, NP methods cannot handle multimodality and discontinuities in function space well and can thus only be expected to produce local solutions.

The population-based nature of genetic algorithms (GAs) enables the evolution of a Pareto-optimal set of solutions. Also, because of the stochastic nature of the search mechanism, GAs are capable of searching the entire solution space with more likelihood of finding the global optimum than conventional optimisation methods. Indeed, conventional methods usually require the objective function to be well behaved, whereas the generational nature of GAs can tolerate noisy, discontinuous and time-varying function evaluations, and, as is the case in this paper – *estimates* of the objectives.

The MOGA approach proposed by Fonseca and Fleming (1993) uses a rank-based fitness assignment, where the rank of a certain individual x_i at generation t is related to the number of individuals $p_i(t)$ in the current population by which it is dominated. This is given by

$$rank(x_i, t) = p_i(t). \qquad \ldots(2)$$

All non-dominated individuals are assigned rank 0 and remaining individuals are penalised according to Eqn. (2).

Fitness is assigned by interpolating from the best individual (rank=0) to the worst, and then the fitness assigned to individuals with the same rank is averaged where the global population fitness is kept constant. However, such fitness assignment tends to produce premature convergence due to the fact that all non-dominated (best rank) points are considered equally fit (Fig. 3). In order to overcome this deficiency, Fonseca and Fleming have used a niche induction method to promote the distribution of the population over the Pareto-optimal front in order to maintain diversity. This is achieved by a method of fitness sharing which encourages the reproduction of isolated individuals and favours diversification.

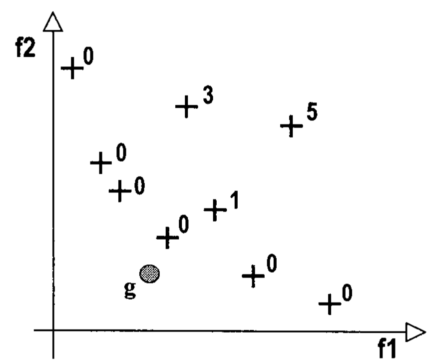

Fig. 3. Pareto-ranking without preference information

2.2 Preference information

Preference information is also introduced in the form of a goal vector, g, which provides a means of evolving only a specific region of the search space. This allows the decision-maker to focus on a region of the Pareto front by providing external information to the selection algorithm. A typical set of design trade-offs resulting from a MOGA design exercise is shown in Fig. 4. This figure illustrates a representation which deals with more than two objectives (eight objectives, in fact, for this flight control example). In this "parallel co-ordinates representation", each line in the graph connects the performance objectives achieved by an individual member of the population and represents a potential solution to the design problem. All solutions illustrated in Fig.4 are both non-dominant **and** satisfy the prescribed goals as represented by the "×" marks. The decision-maker (DM) must select a suitable compromise from this set of solutions. DM may interact with the

MOGA as it evolves, to "tighten" or "slacken" the goals, in order to target a specific compromise solution.

3 Estimation of Objectives Using Neural Networks

A typical control system design problem might be posed as follows. Given a system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t)$, where $\mathbf{x}$ and $\mathbf{u}$ are the system state and control vectors and $\mathbf{f}$ is a vector non-linear function, find a controller $\mathbf{u}$ such that the design specifications,

$$f_i(\mathbf{x}, \mathbf{u}, t) \leq g_i, i = 1, \ldots, m, \tag{1}$$

are satisfied, where g_i are the design goals. Zakian and Al-Naib (1973) proposed a method for obtaining a control vector, $\mathbf{u}$, of prespecified structure, which satisfied the design specifications/constraints eqn. (1).

For the application of MOGA we formulate this problem as a MO problem where f_i are the objectives to be optimised and g_i are components of the goal vector used for preference information. This solution method is superior to that of Zakian and Al-Naib for a number of reasons which include the fact that:

a) MOGA will obtain a family of solutions,

b) these solutions will be optimal in some sense (Pareto-optimal), and

c) MOGA will not fail if g_i are unattainable.

However, use of multiobjective optimisation for system design can often result in a substantial computational load arising from the repeated evaluation of the multiple objectives, especially in cases where the objective function evaluation is costly, for example, when the value is obtained following a system simulation. While evolutionary computing methods have proved effective in obtaining solutions to multiobjective optimisation problems, the population-based nature of the search can exacerbate this computational load difficulty.

To overcome this drawback, an approximation approach is proposed whereby MOGA works with estimates of the objectives, rather than the actual values. (GAs and MOGAs are robust under these conditions). In the first few MOGA generations, actual values of objective functions will be calculated for a representative set of points in decision variable space. Neural networks can then be trained on these points to act as function approximators for these objective functions. In the following Section, a simple six-objective control system design problem illustrates the approach.

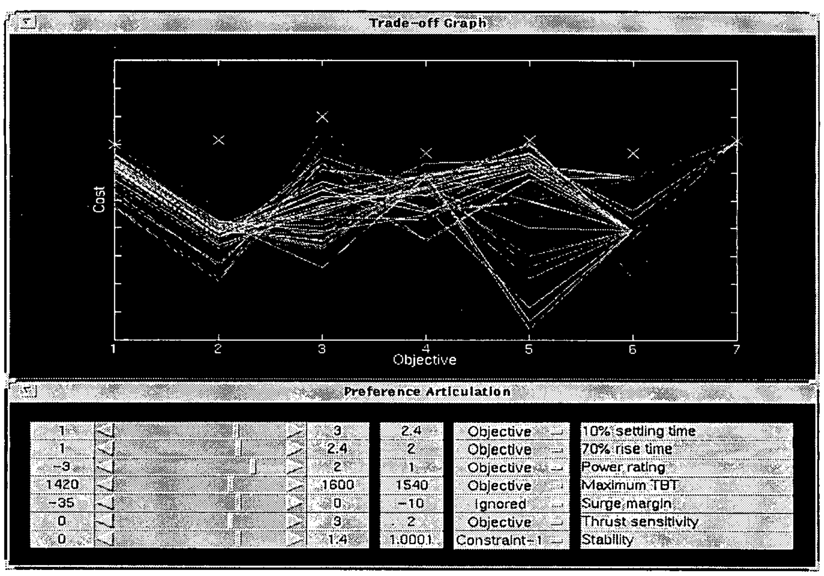

Fig. 4 Parallel Co-ordinates Representation: Design Objective Trade-Offs

4. Design Example

This simple problem is to design a cascade compensator r, $G_c(s)$, for a unity-gain feedback control system with plant transfer function, $G_p(s)$, where

$$G_p(s) = \frac{1}{s(1 + s/2)} \quad \text{and} \quad G_c(s) = K\frac{1 + \tau_1 s}{1 + \tau_2 s},$$

in an attempt to satisfy the specifications outlined in Table 1. (E_{SS} is the steady-state error due to a unit ramp input, T_S is the 2% settling time, T_P is the time to maximum overshoot (peak time), $\%O_S$ is the percentage overshoot to a unit step input and BW is the bandwidth.)

These specifications represent the objectives, f_i, eqn. (1), to be minimised in the MO problem with respect to K, τ_1, and τ_2, i.e.

minimise $\mathbf{f} = [f_1, \ldots, f_6,]^T$ w.r.t. K, τ_1, and τ_2.

With the exception of E_{ss}, the objectives were obtained by simulation and analysis in the time and frequency domain, using Simulink (MathWorks, 1993). Since this was computationally costly, neural networks were trained to provide estimates of the objectives.

4.1 Neural network RBF-based function approximators

Five radial basis function networks (RBFs) were used to map the compensator parameters (K, τ_1, τ_2) to the design objectives. One additional RBF was trained simply to classify between acceptable and non-acceptable (unstable or near unstable) systems. In the subsequent MOGA design process, if an unacceptable solution is identified then there is no evaluation of objectives and the associated individual is simply assigned a low value of fitness in all objectives. All these RBFs were trained prior to running the MOGA, although the first few MOGA generations could have provided the training data. RBFs were preferred to Multilayer Perceptrons (MLPs) for this work. While the approximation capabilities of the two neural network schemes are similar, the hybrid training methods for RBFs are faster than the supervised training methods for MLPs (Haykin, 1994):

The Gaussian radial basis function with the general form $\phi(r) = e^{-\left(\frac{r^2}{\beta}\right)}$ was chosen as the activation function of the hidden functions, where β is the parameter specifying the width of the basis function. The neural networks training process was in two stages - the first to obtain the optimal parameters of the hidden functions (with the use of the K-means clustering algorithm (Bishop, 1995)) - and the second to obtain the linear layer weights. The compensator parameters used in the training belonged to the following range of values: $1 < K < 200$ and $0 < \tau_1, \tau_2 < 1$. All the networks were trained with 246 records and 40 hidden neurons were employed. A test set with 63 records was employed to verify whether the networks generalise well.

Table 2 illustrates the mean errors arising from training non-training data test sets (I - VI). Comparing these results, it can be seen that the networks were not over-trained and that a good generalisation was achieved.

E_{SS}	T_S	T_P	$\%O_S$	BW @ -3dB gain	BW @ -40dB gain
≤ 0.02	≤ 0.3s	≤ 0.1s	$\leq 20\%$	≥ 25 rad/s ≤ 60 rad/s	≤ 300 rad/s

Table 1 Design specifications

% Error	I	II	III	IV	V	VI
Training data	4.4029	4.2198	7.0152	8.7516	9.4866	2.8150
Non-training data	6.3566	4.9558	8.8053	10.2318	9.0713	3.4723

Table 2 Mean errors (%) obtained from training and non-training data

4.2 Compensator design results

Two design exercises are compared:

Design I - uses MOGA with direct evaluation of the design objectives, via simulation and analysis, and

Design II - uses MOGA combined with RBF NNs which estimate the design objectives.

In both Design cases, the MOGA process was run for 30 generations with a population of 75 individuals and identical GA operators and parameters. Fig. 5 shows the normalised design objective values (costs) achieved by each individual member of the resulting set of non-dominant solutions for Design I. Twenty non-dominated solutions are shown in the Figure, where each line represents a solution and the design specification goals are indicated by "X". Ideally, solutions should satisfy all of the design specifications, i.e. pass below all the points (X). It will be observed that this is not the case – demonstrating that an infeasible set of design objectives was postulated. Nonetheless, the resulting Pareto-optimal (non-dominated) set contains the best set of solutions possible.

Fig. 6 shows the corresponding set of solutions arising out of Design II (56 non-dominated solutions in this case) – where the design objective values were estimated by the RBF NNs. Comparing Figs. 5 and 6, it is apparent that similar solutions have been obtained, albeit almost three times more non-dominated solutions for Design II. Remember, also, that the stochastic nature of the GA process means that no two runs will produce identical results. One candidate solution from the set of solutions arising from Design II is selected for Table 2 and the estimated design values are compared with the actual design values for this solution. Reassuringly, there is a very close match between the estimated and actual values.

4.3 Computational effort

There is a very great saving in computational effort using the neuro-genetic approach (Design II) which requires 30 times fewer FLOPs (floating-point operations) than Design I. This does not take into account the computational effort required in training the NNs, but does promise considerable potential for significant savings in design time for more complex design problems. Since the MOGA approach is intended as a high-level decision-making tool requiring designer interaction, this has considerable importance for the viability of the method for large-scale problems.

5. Concluding Remarks

MOGAs are a powerful decision-making aid for the control system designer. It is possible to search for many Pareto-optimal solutions concurrently, while concentrating on relevant regions of the Pareto set. Also, a human decision-maker may interactively supply preference information to the algorithm as it runs. Applications have included the design of controllers for flight dynamics, gas turbine engines and active magnetic bearings. Design problem characteristics have included non-linear system descriptions, incorporation of H-infinity approaches and on-line use of the MOGA tool. Examples of the use of the method may be found in Fonseca & Fleming (1998a; 1998b), Dakev *et al.* (1997), Chipperfield & Fleming (1996) and Schroder *et al.* (1998).

The main computational burden arises from the evaluation of the multiple objectives. Inevitably, in some cases, due to the complexity of designs and the population-based search approach, this burden can prove excessive. It has been shown that using RBF NN function approximators to estimate the values of the objectives can alleviate this load. Currently, an alternative approach that uses Response Surface Models is also under investigation as a means of reducing the computational load.

Design parameters	K	τ_1	τ_2	Design values	E_{SS}	T_s	T_P	$\%O_S$	BW -3dB	BW -40dB
	37.4	0.47	0.037	*Estimated*	0.02	0.29	0.10	22.0	42.0	307
				Actual	0.02	0.29	0.11	22.1	41.9	309

Table 3. Candidate solution – Design II.

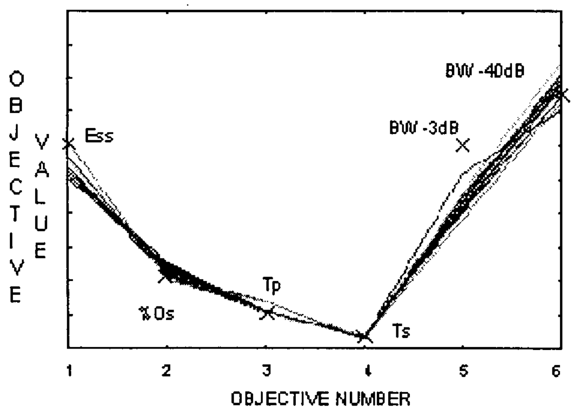

Fig.5 Design I: Pareto-optimal solution set

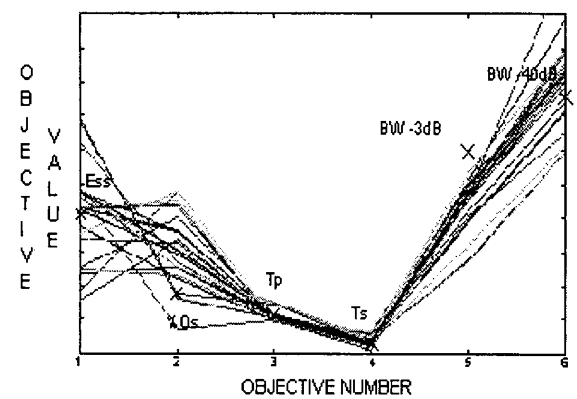

Fig. 6 Design II: Pareto-optimal solution set

References

Bishop CM (1995): Neural networks for pattern recognition, Oxford Clarendon Press.

Chipperfield AJ and Fleming PJ (1996): Multiobjective gas turbine engine controller design using genetic algorithms, *IEEE Transactions on Industrial Electronics*, vol 43, no 5, pp 583-589.

Dakev, NV, Whidborne JF, Chipperfield AJ and Fleming PJ (1997): Evolutionary H_∞ design of an EMS control system for a Maglev vehicle, *Proc Inst Mech Engrs, Part I: J of Sys and Contr. Eng*, vol 211, pp345-355.

Fonseca CM and Fleming PJ (1993): Genetic algorithms for multiobjective optimization: formulation, discussion and generalization, *Proc Int Conf Genetic Algorithms*, Illinois, pp 416-423.

Fonseca CM and Fleming PJ (1998a): Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part 1: A unified formulation, *IEEE Trans Systems, Man and Cybernetics*, vol 28, no 1, pp26-37.

Fonseca CM and Fleming PJ (1998b): Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part II: Application example, *IEEE Trans Systems, Man and Cybernetics*, vol 28, no 1, pp38-47.

Haykin S (1994): "Neural Networks: A Comprehensive Foundation", MacMillan.

Hwang C-L and. Masud ASM (1979): Multiple Objective Decision Making - Methods and Applications, Vol. 164 of *Lecture Notes in Economics and Mathematical Systems*, Springer-Verlag, Berlin.

MathWorks (1993), "Simulink Users Manual", *MathWorks*.

Schroder P, Green B, Grum N and Fleming PJ (1998): On-line auto-tuning of mixed H_2/H_∞ optimal magnetic bearing controllers, *Proc UKACC International Conference on CONTROL '98*, Swansea, UK, pp 1123-1128.

Zakian V and Al-Naib U (1973): Design of dynamical control systems by the method of inequalities, *Proc. IEE*, Vol 120, pp1421-1427.