# Selection of Controller Parameters using Genetic Algorithms

Paulo Oliveira João Sequeira João Sentieiro
LRPI/CAPS and Instituto Superior Técnico
Av.Rovisco Pais, 1, 1096 Lisboa Codex, Portugal
e-mail: D3093@BETA.IST.RCCN.PT Fax: +351 1 352 3014

## 1 Abstract

In this paper the problem of setting the initial values for the parameters of a class of controllers is approached using genetic algorithms. The controller parameters, computed **off-line** via the optimization of a quadratic error performance index, can be used to initialize the controller incorporated in the real plant. During the **on-line** control stage and to increase the overall performance, other tuning mechanisms can be considered [5].

In the application described in this paper, the population strings are build juxtaposing 8 bits substrings quantifying the controllers' parameters, covering the search subspace. Due to the nature of the control problem (reference following) the quadratic functional is chosen as the fitness function.

PID controllers are applied to a significant set of simulated plants and results are presented. A comparison is made between the results obtained for an LQ controller, synthesized solving a Ricatti equation and the results obtained with the GAs optimization procedure.

## 2 Introduction

In the design of a control strategy, the initial values of a set of significant parameters may have to be chosen, in order to guarantee a reasonable performance (essentially to avoid initial instability). This is particularly critical when the system to be controlled is poorly modelled. These parameters can be controllers' parameters, adaptation parameters, supervisors' parameters or others.

Genetic algorithms [2] are based in natural genetics. In a set of individuals (strings) belonging to generation $(t)$, those better fitted to the environment will have a higher survival probability in the next generation $(t+1)$ (**reproduction**). Among those, a **crossover** may be considered, based on the idea that if two individuals reveal to be well fitted a combination of them will potentially show a good (or even better) fitness as well. Also, with small probability, some **mutations** may occur, adding new individuals to the population and therefore increasing the points of the parameter space to be explored.

Genetic algorithms (GAs) are used to find the initial values for the parameters of a class of controllers, by optimizing a quadratic error performance index over time. For this kind of control application, genetic algorithms have shown to be very time consumable, in particular due to the simulation of the close loop control necessary in order to evaluate the fitness function of the strings in the population. In the approach considered in this paper a complementary mechanism is added to diminish this evaluation time. The idea underlying the introduction of this mechanism is supported on the fact that strings evaluated at a particular generation, and that have not been crossedover or mutated during propagation, should exhibit the same fitness (deterministic environment) or a similar one (non-deterministic environment).

When GAs are used and no constraints are imposed on the time horizon, convergence to global optimum with probability 1 has already been proved [3,4].

During the execution of the Genetic Algorithms, only the fitness function of the strings is evaluated. No derivatives, gradient calculations or other environment knowledge is necessary.

# 3 Basic Genetic Algorithms Concepts

Genetic Algorithms (GAs) were first introduced by John Holland as search algorithms based in natural selection principles [2,7].

Initially, a population of size $n$ composed by a set of artificial creatures (strings) is randomly generated [2]. Each string (usually binary) codes a point in the parameter space. A set of operators based on natural genetics can be used to guide the population in the search for the maximal (minimal) fitness. In this paper only three basic operators will be used:

- **Reproduction** – In generation $t$ each string survival probability is proportional to the actual fitness values. The reproduced strings are found by a random roulette wheel selection procedure.
- **Crossover** – Given the selected members of population, generated *via* reproduction, and assuming that if two individuals exhibit good fitness then a combination of them, will potentially show a good (or even better) fitness, the strings are mated (parents). With a given probability ($xProb$) the couples will crossover, generating two new strings (children) or will remain unchangeable. In the case of an effective crossover, the break point in the mated strings will be also randomly chosen (the site occurs between 1 and $length - 1$).

  Example of crossover:

$$\begin{array}{cccc} & Parents & Children & \\ Mate\ 1 \left\{ \begin{array}{c} 0011|11 \\ 0111|01 \end{array} \right. & & \left. \begin{array}{c} 0011|01 \\ 0111|11 \end{array} \right\} effective \end{array}$$

- **Mutation** – Finally, to obtain the $t+1$ generation, some mutations with a small probability ($mProb$) are performed on each bit of all the strings of the population previously reproduced and crossedover. In the binary case, mutation consists on interchanging '0' and '1' symbols in the population strings.

GAs are different from other optimization strategies since the search is performed, over a set of points in the parameter space (each one represented by a string in population), not guided by gradients or other environment information.

In order to reduce the computational burden associated with the application of GAs to control problems, a complementary mechanism is added. It consists of the following two steps:

(i) identification of the strings that propagate from generation ($t$) to generation ($t+1$), without change. This means that the string is reproduced but is not crossedover or mutated. The reduction time could be computed proportionally to:

$$P_{unchanged} = \left[(1 - xProb) + \left(1 - \frac{f(s_j)}{\overline{f}} . \frac{f(s_k)}{\overline{f}}\right) .xProb\right] .(1 - mProb)^l \quad (1)$$

where:

$\overline{f}$ – average fitness of population, in generation $t$.

$xProb$, $mProb$ – crossover and mutation probabilities.

$l$ – total length of string.

(ii) evaluation of the new individuals in population.

Due to the non deterministic properties of the operators, it is important to know how information improves in the population. Schemata, consisting in similarity templates in strings at any position, will be used. A schema, composed by the symbols $\{0, 1, *\}$ (where $*$ equals the symbols '0' or '1'), maps the strings in the population into the blocks of information that are improved during GAs execution. The **Schema Theorem** or the **Fundamental Theorem of Genetic Algorithms** [2] is a well-known result, which gives a lower bound on the number of copies of a schema $H$ present in generation $t+1$ (due to the reproduction, crossover and mutation operators), given the number of copies on generation $t$:

$$m(H, t+1) \geq m(H, t) \cdot \frac{f(H)}{\overline{f}} \left[1 - xProb \cdot \frac{\delta(H)}{l-1} - o(H) \cdot mProb\right] \quad (2)$$

where:

$m(H, x)$ – number of copies of schema H, in generation $x$.

$f(H)$ – fitness function of string with schema $H$.

$\delta(H)$, $o(H)$ – length and number of fixed positions of the schema.

It follows from the Schema Theorem that the schema $H$ grows or decays depending on whether the schema is above or bellow the population average and whether the schema has respectively long or short length. Here mutation is ignored, due to the smaller probability associated, compared with crossover probability ($mProb \ll xProb$). Reproduction involves a local optimization procedure (*specialization*) and the crossover and mutation operators lead to the exploration of new points (*diversification*).

Using a Markov Chain formulation the following Lemma, proving convergence to the optimum, with probability 1 is proved in [3]:

**Lemma:**

$$Pr\{T(0) < \infty\} = 1. \text{ In Fact, } E[T(0)] \leq N^N.$$

where

$T(0)$ – elapsed number of generations until global optimum is reached.

$N$ – search space dimension.

# 4 Implementation and Results

The problem under consideration regards the selection of the initial parameters of a PID controller, implemented as:

$$u(t) = K_P \left( error(t) + K_I \sum_{k=t0}^{t} error(k) + K_D \Delta error(i) \right)$$

minimizing the functional

$$J = \sum_t (reference_t - output_t)^2$$

over a period of time (300 sampling instants).

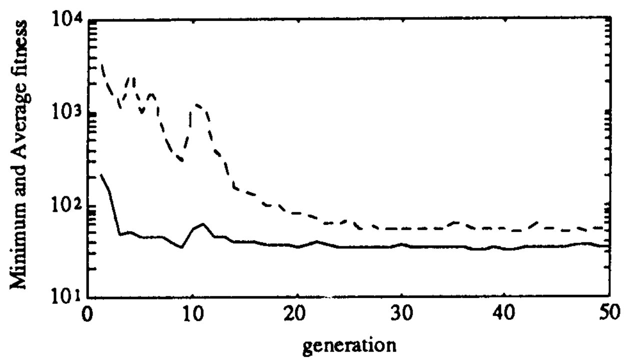
To analyse the results of this optimization procedure, several simple dynamic systems were simulated:

- A non-minimum phase system.
- An unstable system.
- A minimum phase system, with noise.
- A minimum phase system, with delay.

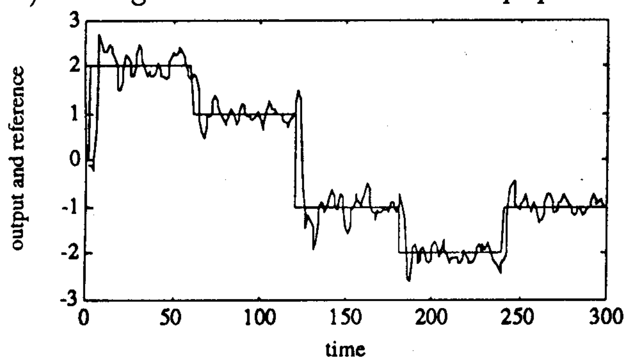
Each set of PID parameters is coded in a 24 bits string, where $K_P$ is limited to $[-10, +10]$, $K_I$ to $[-1, +1]$ and $K_D$ to $[-5, +5]$. As the GAs perform a maximization and the problem under consideration involves a minimization of the functional $J$, the inverse of $J$ is used as the fitness function.

In Figures 1, 2, 3 and 4 the results obtained for the simulated systems are presented. In each figure the following information is displayed:

- Average and minimal value of the functional of strings in population, over the optimization procedure.
- Output and reference signals using the best string in the last generation.
- Characteristics of the simulated system and respective difference equation.
- A set of configuration parameters for the GAs and the reduction time observed.




a) Average and minimum fitness in population

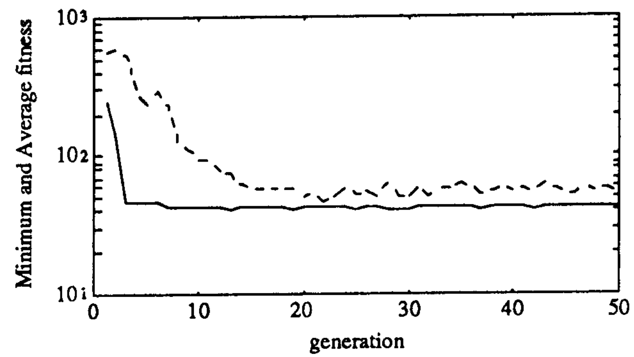



b) Output and reference for the best string in generation 50

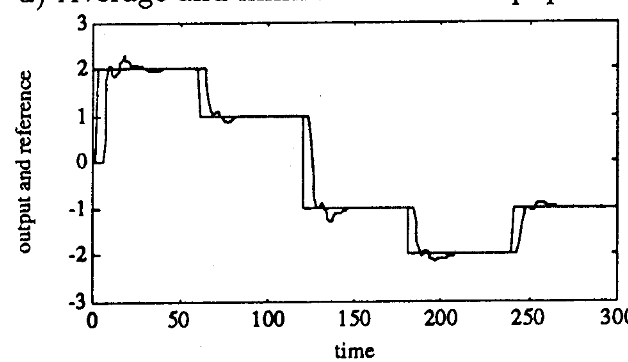
Figure 3: Results for the minimum phase system, with noise

$$\begin{aligned} y(t) &= 1.06y(t-1) - 0.22y(t-2) + \\ &\quad 1.99 \cdot 10^{-2}u(t-1) + 1.99 \cdot 10^{-2}u(t-2) + 0.1e(t) \end{aligned}$$

| Generations | Population | mProb | xProb | Red. Time |
|---|---|---|---|---|
| 50 | 50 | 0.6 | 0.05 | 12.5% |




a) Average and minimum fitness in population

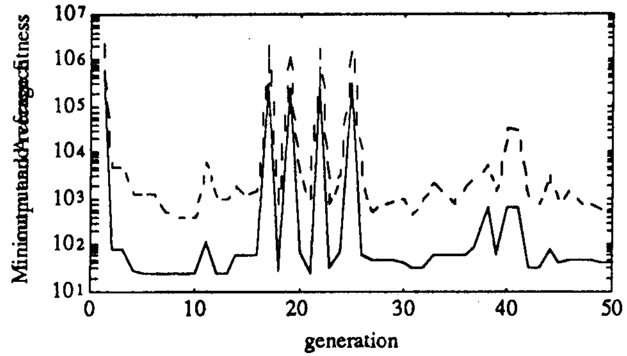



b) Output and reference for the best string in generation 50

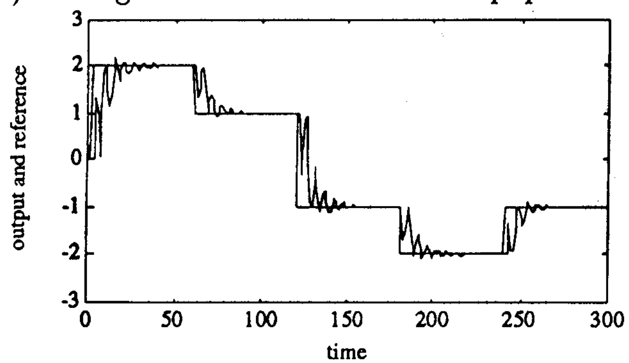
Figure 4: Results for the minimum phase system, with delay

$$\begin{aligned} y(t) &= 1.06y(t-1) - 0.22y(t-2) + \\ &\quad 1.99 \cdot 10^{-2}u(t-3) + 1.99 \cdot 10^{-2}u(t-4) \end{aligned}$$

| Generations | Population | mProb | xProb | Red. Time |
|---|---|---|---|---|
| 50 | 50 | 0.6 | 0.05 | 11.5% |




a) Average and minimum fitness in population

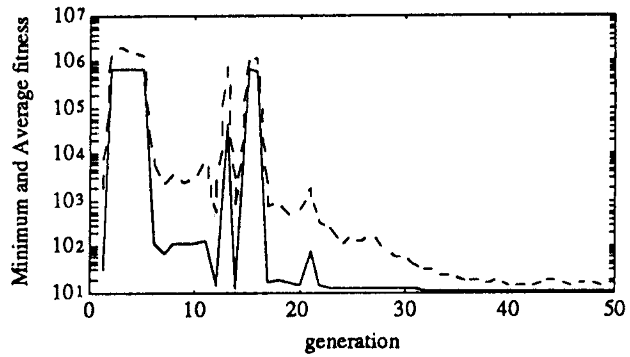



b) Output and reference for the best string in generation 50

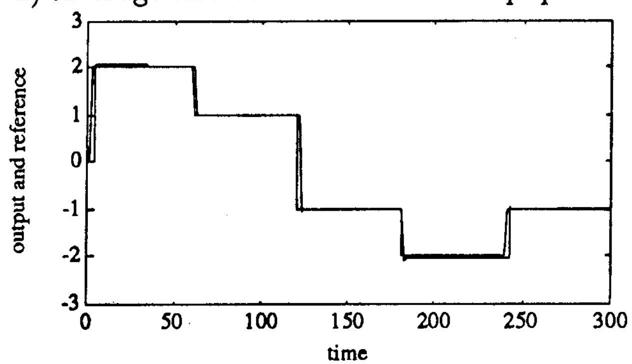
Figure 1: Results for the non-minimum phase system:

$$y(t) = 1.2y(t-1) - 0.35y(t-2) - u(t) + 1.05u(t-1)$$

| Generations | Population | mProb | xProb | Red. Time |
|---|---|---|---|---|
| 50 | 50 | 0.6 | 0.05 | 11% |




a) Average and minimum fitness in population




b) Output and reference for the best string in generation 50

Figure 2: Results for the unstable system

$$y(t) = y(t-1) + u(t-1)$$

| Generations | Population | mProb | xProb | Red. Time |
|---|---|---|---|---|
| 50 | 50 | 0.6 | 0.05 | 11% |

# 5 Selection of Parameters using GAs in a LQ Problem

The optimizations procedures presented in the previous section raise the problem of discussing for a given controller structure, how far apart the optima found by GAs and the reachable optima are.

In a general set-up, the optimum calculation is very laborious or even analytically impossible. A new problem can be formulated, as an LQ problem, where the controller parameters are computed using the solution of a Ricatti equation, that minimizes the functional:

$$J = \sum_t \left( (reference_t - output_t)^2 + \rho control_t^2 \right).$$

The LQ optimal regulator solution gives the smallest $J$ achievable by any LTI regulator that stabilizes the system [1]. The results obtained by the LQ and the GAs approaches are plotted in Figure 5 for a range of $\rho$ values.

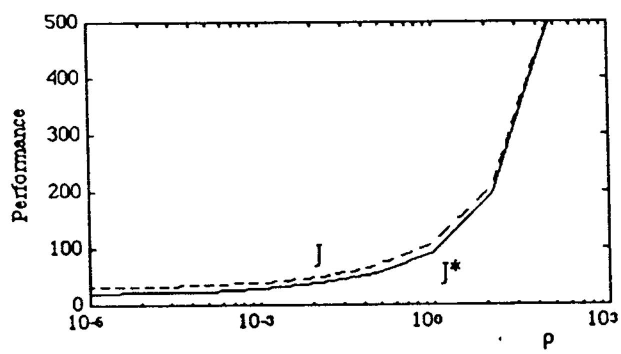



Figure 5: Performance obtained with LQ gains (J*) and with GAs optimization (J).Results for the minimum phase system:

$$\begin{aligned} y(t) &= 1.06y(t-1) - 0.22y(t-2) + \\ & \quad 1.99 \cdot 10^{-2}u(t-1) + 1.99 \cdot 10^{-2}u(t-2) \end{aligned}$$

| Generations | Population | mProb | xProb |
|---|---|---|---|
| 50 | 50 | 0.6 | 0.05 |

The results reveal the good performance of the GAs optimization.

# 6 Conclusions

Genetic Algorithms were used to find the initial parameter values both for a PID controller and for a controller with a structure similar to the one obtained using an LQ approach.

The first set of results, leads to stable closed loop systems, what identifies the proposed tool as a good off-line optimization procedure. The second set of results reveals that the performance obtained using GAs is close to the optimal eventhough only 0.015% of the search space was evaluated.

# References


[1] Boyd, S. and Norman, S. (1990). *Linear Controller Design: Limits of Performance via Convex Optimization.* Proceedings of the IEEE, vol. 78, n 3, March.

[2] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Eds. Addison-Wesley.

[3] Karlin, S. and Taylor, H (1975). *A First Course in Stochastic Processes.* Eds. Academic Press.

[4] Moed, M. C. (1991). *A Connecionist / Symbolic Model for Planning Robotic Tasks.* Ph. D. Thesis, RPI.

[5] Oliveira, P.; Lima, P.; Sentieiro, J. (1990). *Fuzzy Supervision of Direct Controllers.* Proceedings of the V International Symposium on Intelligent Control.

[6] Oliveira, P., Lima, P. and Sentieiro, J. (1991). *Fuzzy Supervision on Intelligent Control Systems.* To be presented on the 2th. European Control Conference.

[7] Schaffer,J. (1989). *Proceedings of the Third International Conference on Genetic Algorithms.* Eds. Morgan Kaufmann.