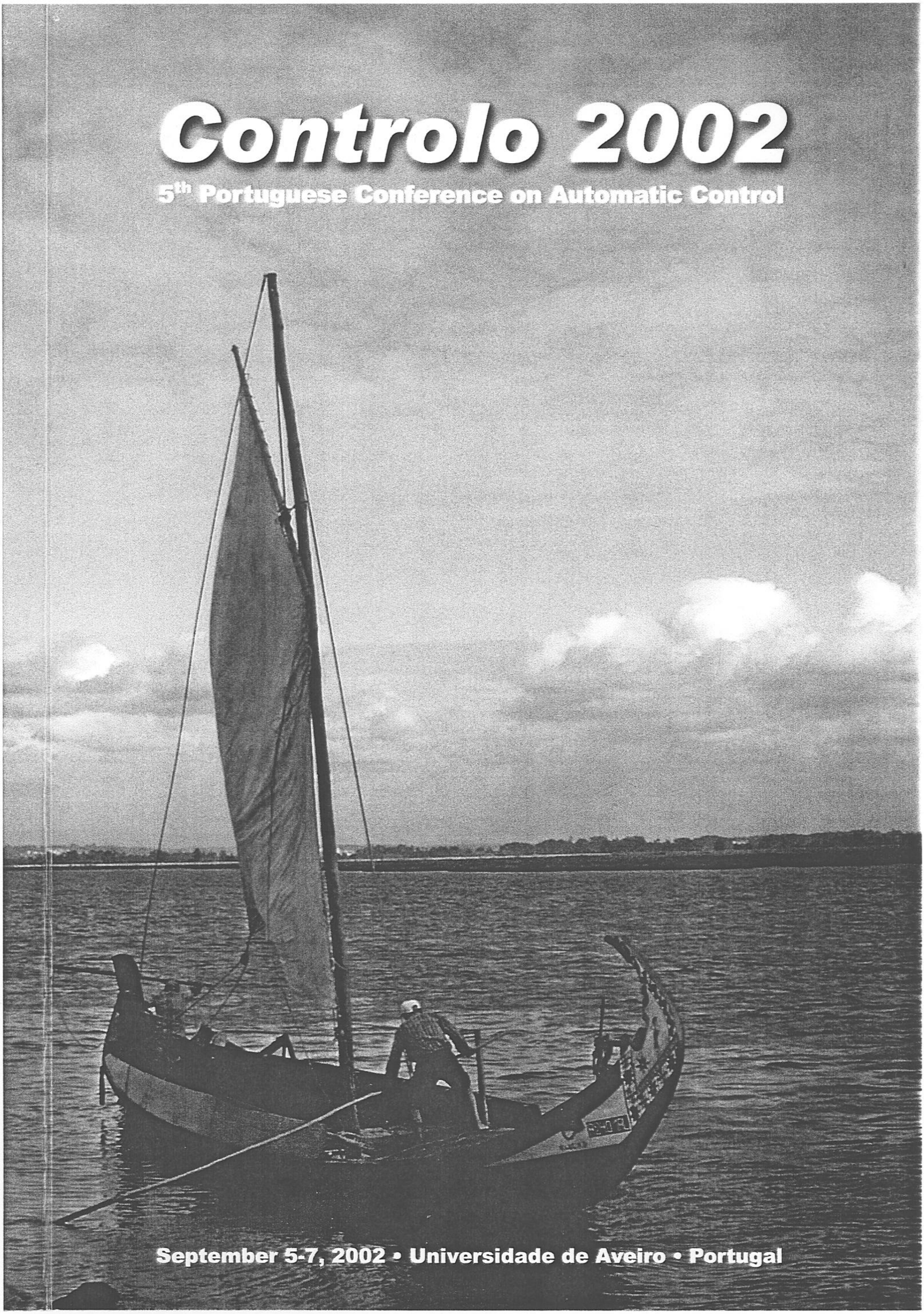
Controlo 2002

5th Portuguese Conference on Automatic Control



September 5-7, 2002 • Universidade de Aveiro • Portugal

Proceedings of Controlo 2002
5th Portuguese Conference on Automatic Control
Aveiro, Portugal, September 5-7, 2002

NON-LINEAR CONCENTRATION CONTROL SYSTEM DESIGN USING A NEW ADAPTIVE PARTICLE SWARM OPTIMISER

J. P. Coelho[a], P.B. De Moura Oliveira[b] and J. Boaventura Cunha[b]

a. Instituto de Trás-os-Montes e Alto Douro para a Investigação e Desenvolvimento Agro-Industrial, 5000 Vila Real, Portugal.
b. Universidade de Trás-os-Montes e Alto Douro, Dep. Engenharias 5001-911 Vila Real, Portugal
e-mail: jpcoelho@pol.pt

Abstract: Designing a linear controller that performs satisfactorily within a non-linear operating envelope is a difficult control engineering problem. In this paper, a new adaptive particle swarm optimisation algorithm is proposed, that can track difficult dynamic search landscapes. The proposed adaptive particle swarm algorithm is applied to design linear discrete PI regulators to control a classical non-linear concentration system, in which the operating point conditions change considerably with the input flow change. Simulation results are presented that show the effectiveness of the proposed evolutionary based design technique.

Keywords: PID Control, Tuning based on Optimisation, Non-Linear Control, Adaptive Particle Swarm Optimisation.

1. INTRODUCTION

It is well known that a great deal of the control systems design is done by using nominal operating point locally linearised plant models. Classical control techniques, such as PID control, have been successfully applied to this problem by using standard tuning rules (Ziegler and Nichols, 1942; Cohen and Coon, 1953; Lopez *et al.*, 1969; Luyben 1996), or automatic tuning techniques (Åström, 1982; Åström and Hägglund 1984; Nishikawa *et al.*, 1994). However, when the objective is to design a non-linear control system, performing well not only around a linearised nominal plant operating point, but in the entire envelope, fixed gains controllers fail. To avoid the tedious system design in different operating conditions of the search space, a gain scheduling controller (Åström and Wittenmark, 1995) can be used in which the gains are scheduled using either a function or a table. Gain-scheduling design can be a difficult task, depending on the complexity of the non-linear system. This motivates the research on developing alternative control techniques that can automatically compute the controller gains to different operating conditions.

Evolutionary algorithms constitutes a sound alternative technique in control system design for both off-line design and analysis and on-line adaptation and tuning. In off-line applications the evolutionary algorithm can be employed as a search and optimisation mechanism to look for optimal parameter settings for a particular controller structure (Porter and Jones, 1992) (Fonseca and Fleming, 1994) (Oliveira and Pires, 1999). On-line adaptation can be used as a learning mechanism for adaptive controller tuning (Salami and Cain, 1992).

Although more classical evolutionary techniques such as genetic algorithms (GAs) (Holland, 1975; Goldberg, 1989) has been widely referred in the literature as a suitable tool for optimisation in control system design, there has been considerable research effort into developing new optimisation algorithms

Inspired on natural concepts like the particle swarm optimisation algorithm (PSO) proposed by Kennedy and Eberhart. This technique has already been applied to design both linear and non-linear PID controllers by (Oliveira *et al.*, 2002) and predictive controllers (Coelho *et al.*, 2002). This paper reports the use of a new adaptive particle swarm optimisation algorithm for adaptive design and tune linear PI controllers in the discrete time domain.

2. PARTICLE SWARM OPTIMISATION

2.1 Particle Swarm Optimisation Algorithm for Static Landscapes

Particle Swarm Optimisation (PSO) is an algorithm inspired on the social behaviour of organisms (Kennedy and Eberhart 1995). The algorithm searches a multidimensional space by adjusting swarm individuals trajectories toward the positions of their own previous best performance and the best previous performance of their neighbours, i.e. a particle change its position using knowledge acquired from individuals in a neighbourhood and based on its own experience.

Particles change their position until a relatively steady state has been achieved, or until user defined limits are exceeded. The PSO algorithm presented in figure 1 represents an improved version of the original one and was proposed by Shi and Eberhart (1998).

$k = 0$

$X(k) \leftarrow$ *Initialise* ( )

REPEAT until stop criterion=TRUE

$X^{\#}, X_i^{*} \leftarrow Evaluate\big(X(k)\big)$

$CO_i(k) = \phi_c\left[X_i^{*} - X_i(k-1)\right]$

$SO_i(k) = \phi_s\left[X^{\#} - X_i(k-1)\right]$

$V_i(k) = \omega V_i(k-1) + CO_i + SO_i$

$X_i(k) = X_i(k-1) + V_i(k)$

$k = k+1$

END REPEAT

k – Epoch counter
$X^{\#}$ - Global best position achieved
X_i^{*} - Individual i best position achieved
$X_i(k)$ – Position of individual i at epoch k
$V_i(k)$ – Velocity of individual i at epoch k
ω – Inertial weight
CO_i – Cognition only segment of individual i
SO_i – Social only segment of individual i
ϕ_c,ϕ_s – Cognitive and Social weights

Fig. 1. Improved particle swarm optimisation algorithm (Shi and Eberhart, 1998).

Particle position is initialised randomly, starting to move in the search space with a predefined initial velocity. Velocity and position adjustments are carried out by simple algebraic equations and usually they are bounded to prevent the algorithm to prospect out of the defined search space. The cognitive and social parameters, ϕ_c and ϕ_s, are positive constants defined randomly in the interval $[0,1]$, and the inertial weight ω is usually selected to have a decrescent linear distribution over the evolution time. A larger inertia weight has a tendency to promote global exploration while a smaller one tends to improve local search leading to a fine-tuning of the solution.

2.2 Particle Swarm Optimisation Algorithm for Dynamic Landscapes

The basic PSO algorithm described in the last section presumes the existence of a static search space, i.e. the original algorithm is unable to track variations in global optima. This issue can be a serious drawback in many engineering optimisation problems in which the search space is dynamic, meaning that the global optimum can change with time.

A simple and minor modification that can be done, so the PSO algorithm can follow changes in the location of the optima, is to change the velocity update equation to:

$$V_i(k) = \omega V_i(k-1) + SO_i(k) \qquad (1)$$

with

$$SO_i(k) = \phi_s\left[X^{\#}(k-1) - X_i(k-1)\right] \qquad (2)$$

The social only equation is different from the one presented in figure 1. In this case the global best position is the best particle of the swarm at each epoch rather than the global best achieved trough all evolution. Notice that in this case the inertia weights should be chosen with values close to unity, to prevent premature convergence. Indeed, this modification can track dynamic search spaces, but is limited to cases where there are no local optima with sufficient energy to elude the swarm. This strategy has showed a good performance tracking the sphere objective function (Kennedy and Eberhart, 1995) with time varying centre, but it was unable to follow changes of more complex functions as the one presented in section 4.

To bypass the multimodal constraint, Clerc (2000) proposes a modification to the original algorithm that he called *Cheap-PSO*. This strategy has a high level of adaptation in the parameters and in the swarm size. In this algorithm the velocity and position are governed by equations (3) and (4), respectively.

$$V_i(k) = \alpha V_i(k-1) + b\left[X^{\#}(k-1) - X_i(k-1)\right] \quad (3)$$

$$X_i(k) = X_i(k-1) + V_i(k) \quad (4)$$

Parameters α and b are epoch varying and are evaluated by equations (5) and (6).

$$\alpha = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\frac{e^m - 1}{e^m + 1} \quad (5)$$

$$b = \frac{b_{\min} + b_{\max}}{2} + \frac{(b_{\max} - b_{\min})(e^{-m} - 1)}{2(e^{-m} + 1)} \quad (6)$$

in which m is obtained by,

$$m = \frac{f(X^{\#}(k-1)) - f(X_i(k-1))}{f(X^{\#}(k-1)) + f(X_i(k-1))} \quad (7)$$

and: $\alpha_{\min} \approx 0.5$, $b_{\min} > \alpha_{\min}$, $\alpha_{\max} \approx b_{\max}$. Parameter $\alpha_{\max}$ is selected to be smaller than but close to 1. Notice that in this method the objective function f is supposed to be always positive or zero. Complementary the swarm size can change with evolutionary time by adding or removing particles. This decision in taken by evaluating a «gradient» Δf for each element (Clerc, 2000).

3. SPLIT ADAPTIVE PSO ALGORITHM

In this paper a split adaptive particle swarm optimisation algorithm (SAPSO) is proposed, which is a new alternative to the original particle swarm optimisation algorithm. This approach splits the swarm population in two sub-populations: a social subset S and a cognition subset C. The recurrent equations for adjusting particles trajectories are given by:

$$V_s(k) = \omega V_s(k-1) + \phi_s\left[X^{\#}(k-1) - X_s(k-1)\right] \quad (8)$$

$$V_c(k) = \omega V_s(k-1) + \phi_c\left[X_c^{*} - X_c(k-1)\right] \quad (9)$$

$$X_s(k) = X_s(k-1) + V_s(k) \quad (10)$$

$$X_c(k) = X_c(k-1) + V_c(k) \quad (11)$$

Each swarm subset is adjusted almost independently except for equation (9) where the sub-velocity V_c is a function of the «social» velocity V_s. Each epoch, the variables ω, ϕ_s and ϕ_c are chosen randomly with different values for each problem dimension. Inertia weights are generated randomly and incorporated in a vector ω with values between 0.9 and 0.4, and ϕ_s and ϕ_c are random vectors with values between 0 and 2. Additionally, if the previous best individual position X_c^{*} is better than the current individual position X_c, solution X_c^{*} is re-evaluated.

4. SAPSO VERSUS CHEAP-PSO

In order to evaluate and compare the performance of the previously described methods, a multimodal dynamic landscape was established. The function selected was a variation of the *peaks* Matlab® function and it is described by the following equations:

$$\begin{aligned} & f(x,y) = \left|f_1(x,y) + f_2(x,y) + f_3(x,y)\right|, \text{ were} \\ & f_1(x,y) = \xi(1-x)^2 e^{(-x^2-(y+1)^2)} \\ & f_2(x,y) = -10\left(\frac{x}{5} - x^3 - y^5\right)e^{(-x^2-y^2)} \\ & f_3(x,y) = -\frac{1}{3}e^{(-(x+1)^2-y^2)} \end{aligned} \quad (12)$$

In which $x \in [-3,3]$, $y \in [-3,3]$ and ξ represents a parameter that is changed to modify the global maximum. In the first 150 epochs $\xi = 3$ and for the remainder 150 epochs $\xi = 6$. Figures 2 and 3 show the landscape generated with this function for $\xi = 3$ and $\xi = 6$, respectively. Additionally a dot is plotted to indicate were the global maximum for each cost function is located.

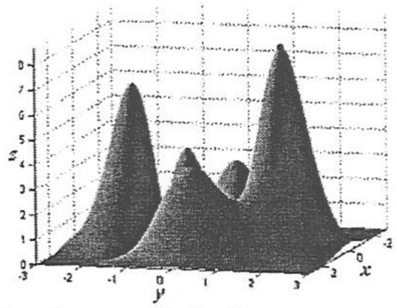


Fig. 2. Search space with $\xi = 3$.

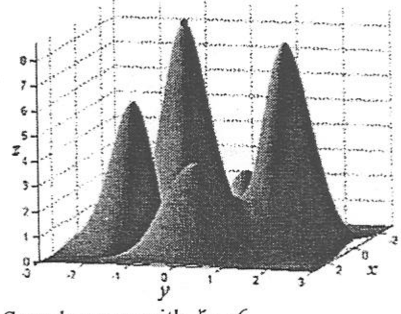


Fig. 3. Search space with $\xi = 6$.

Matlab® is a registered trademark of The MathWorks, Inc.

A relative measurement Δ of the distance between optima was established. This measurement indicates how hard is for the algorithm to escape when the swarm converges for the first maximum. The relative distance is computed as:

$$\Delta(\%) = \frac{d_{sol}}{d_{max}} \times 100 \approx 28.4\% \qquad (13)$$

where d_{sol} is the Euclidean distance between optima and d_{max} is the larger diagonal distance of the search space.

Both algorithms were run 100 times for 300 epochs each time. Conceptually only a velocity bound were incorporated in both algorithms with a value of half the search space for each dimension. Figure 4 shows a convergence plot of both algorithms for this problem.

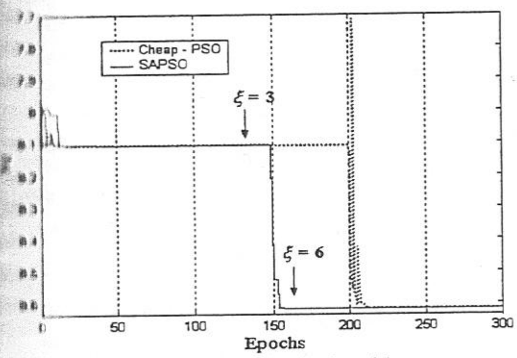


Fig. 4. Convergence rate of both algorithms.

The above plot indicates the best convergence achieved for both algorithms over 100 test iterations. As it can be observed, both algorithms were able to track optima variations, but the proposed algorithm has a faster convergence rate. In table 1, the percentage of successful convergences in 100 tests of both algorithms to the solution are showed.

Table 1 Convergence percentage for the Cheap-PSO and SAPSO algorithms

Algorithm	Convergence (%)
Cheap-PSO	14 %
SAPSO	88 %

As it can be observed, the proposed algorithm out-performs *Cheap-PSO* algorithm for multimodal dynamic objective functions.

5. PROBLEM STATEMENT

In order to demonstrate the usefulness of the adaptive particle swarm optimisation algorithm to design proportional-integral (PI) controllers for non-linear time varying processes, an example of a two tanks concentration control system is examined (Åström and Wittenmark, 1995). In this problem, the process dynamics change when the production rate changes, so a well-tuned controller for one production rate will not necessarily perform well for other rates. Although this kind of problems can be avoided with a gain scheduling strategy (Åström and Wittenmark, 1995), in this paper is purposed the tuning of a PI controller using an adaptive PSO technique. The use of this algorithm avoids using variable sampling rates for the different operating conditions.

Consider a concentration control for a fluid that flows through a pipe to a buffer tank with no blending, and then to a tank with perfect mixing. A schematic diagram of the process is shown in figure 5.

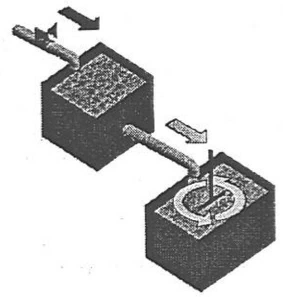


Fig. 5. Schematic diagram of a concentration control system.

Let u be the fluid concentration at the pipe admission, y the concentration in the mixing tank and in the pipe output. For this process, a pipe volume V_d and a tank volume V_m is defined. Moreover it is also available a measurement of the fluid flow q.

Using T_s as the sampling period, the discrete process model is described by the following model.

$$y(kT_s) = \alpha y(kT_s - 1) + (1-\alpha)u(kT_s - \tau) \qquad (14)$$

in which $\tau = (\text{int})\frac{V_d}{qT_s}$, $\alpha = e^{-\left(\frac{V_d}{\tau V m}\right)}$ and $k \in \square^{+}$.

By analysing equation (17) it is possible to conclude that the model dynamics are dependent of the fluid flow (q). If the flow changes, then there will be significant variations reflected in the system response. The purpose is then to tune an "*on-line*" PI controller in order to be able to follow the dynamics in the variations of the system response caused by fluctuation on the fluid flow.

The PI controller is governed in the discrete time domain by the following incremental equation.

$$\Delta u(kT_s) = K_p\left[\Delta e(kT_s) + K_i T_s e(kT_s)\right] \qquad (15)$$

in which, $e(kT_s) = v(kT_s) - y(kT_s)$ is the error signal, $v(kT_s)$ is the reference signal, K_p and K_i represent respectively the proportional and integral gains. In order to use the PSO algorithm to optimise a control system with a PI controller it is necessary to encode the tuning parameters. Thus a vector $\left[K_p \quad K_i\right]$ using a real-based coding scheme represents each particle in the population. For the present problem, the particles are evolved from randomly initialised positions.

The PI controller design is accomplished by optimising the system response to a unit set-point change by minimising the discrete Integral Time of the Absolute Error (ITAE) performance criterion.

$$J\left(K_p, K_i\right) = \sum_{k=1}^{N} k\left|e(kT_s)\right| \tag{16}$$

in which N is the simulation horizon.

6. SIMULATION RESULTS

In order to solve the concentration control problem described in section 5 by means of the SAPSO algorithm, the test conditions described in table 2 and the PSO tuning parameters expressed in table 3 were used.

Table 2 Simulation conditions

Variable	Value
Tank Volume (V_m)	10
Pipe Volume (V_d)	1
Sampling Time (T_s)	0.3
Fluid Flow (q)	[0.4,2]

Table 3 SAPSO tuning parameters

Variable	Value
Swarm Size	100
Inertia Weight	[0.4,0.9]
ϕ_c	[0,2]
ϕ_s	[0,2]

Table 4 shows the experimental set established for the different flow (q) operating condition and the results for the PI controller gains at minimum ITAE.

Table 4 Obtained results with SAPSO.

	Solution		
Flow (q)	K_p	K_i	$J\left(K_p, K_i\right)$
0.4	5.3964	0.0420	579.658
0.67	5.0140	0.0673	325.089
0.94	4.8369	0.0844	253.024
1.21	4.9129	0.1130	186.725
2	4.5329	0.1709	125.836

The flow was set to change every 50-evolution epochs, which was set deliberately to a small number to evaluate the adaptation capabilities of the proposed algorithm. For every flow case the algorithm converged to the global minimum. This can be observed in Figures 6 and 7 that illustrate the landscape obtained for the extreme flow cases of q=0.4 and q=2, respectively.

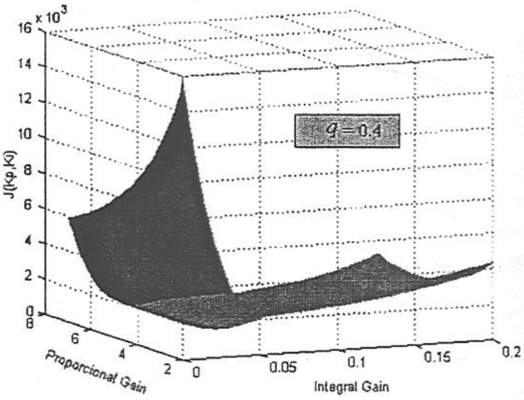


Fig. 6. ITAE for q=0.4.

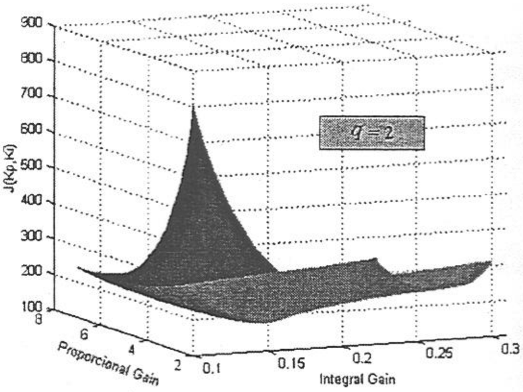


Fig. 7. ITAE for q=2.

The unit set-point responses are shown in figure 8, where it can be seen a good tracking capability to all the different operating point conditions. It is important to notice that in this case the sampling time is constant (T_s=0.3) for all the studied cases, and that no identification of the changing plant dynamics was feedback to the design algorithm.

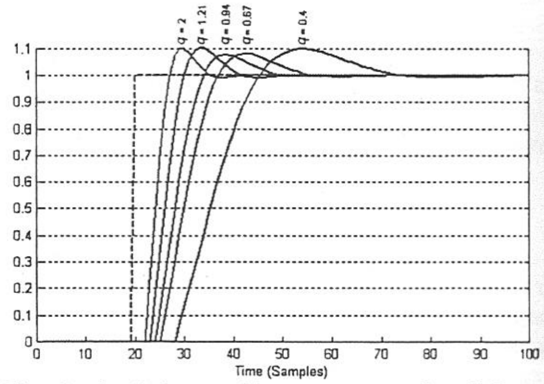


Fig. 8. Set-Point tracking responses for different values for the flow (q).

7. CONCLUSION

In this paper, a new adaptive particle swarm algorithm was presented in order to track variations of global optima in dynamic landscapes. The proposed adaptive evolutionary algorithm was applied to design PI controllers that can adapt to changes in the operating conditions of a non-linear concentration system. Simulation results indicate that the converged PI controller settings provide good set-point tracking. The proposed adaptive algorithm does not require random population re-initialisation to avoid dynamic global optima changes. Despite the good results obtained, further work will be endorsed to study the algorithm full potential with dynamic subpopulations sizes.

REFERENCES

Åström K. J. (1982). Ziegler-Nichols Auto-Tuners, Report TFRT-3167, *Dep. of Automatic Control, Lund Institute of Technology*, Lund, Sweden.

Åström K. J. and Hägglund T. (1984). Automatic Tuning of Simple Regulators with Specifications on Phase and Amplitude Margins. *Automatica*, **20**, **5**, 645-651.

Åström K. J. and Wittenmark B. (1995). *Adaptive Control*, Addison Wesley, ISBN 0-201-55866-1, 2nd edition.

Clerc M. (2000). The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization, *presented at International Congress on Evolutionary Computation*, San Diego, California.

Coelho J. P., De Moura Oliveira P. B and Cunha J. B. (2002). Greenhouse Air Temperature Control using the Particle Swarm Optimisation Algorithm, Accepted for Publication in the *15th IFAC World Congress*, July, Barcelona, Spain

Cohen G. H. and Coon G. A. (1953). Theoretical Consideration of Retarded Control, Transactions of the ASME, **75**, 827-834.

De Moura Oliveira P. B and Solteiro Pires E. J. (1999). Projecto de Controladores PID Discretos Nao-Lineares Usando PBIL, *XX Jornadas de Automatica*, Salamanca, Spain, pp.229-234.

De Moura Oliveira P. B, Cunha J. B. and Coelho J. P. (2002). Design of PID Controllers using the Particle Swarm Algorithm, *Proc. of IASTED MIC'2002*, February, Innsbruck, Austria, pp. 263-268.

Fonseca, M. and Fleming, J. (1994). Multiobjective Optimal Controller Design With Genetic Algorithms, *Proc. IEE Control '94*, pp. 745-749.

Goldberg E. D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Adison Wesley P.C.

Holland J. H. (1975). *Adaptation in Natural and Artificial Systems*, 1st MIT Press ed.

Kennedy J. and Eberhart R. C. (1995). Particle Swarm Optimization, *Proceedings of the 1995 IEEE Int. Conference on Neural Networks*, Perth, Australia, pp. 1942-1948.

Lopez A, Murril P. and Smith C. (1969). An Advanced Tuning Method. *British Chemical Eng.*, Vol. 14 (11), pp. 1553.

Luyben W. L. (1996). Tuning Proportional-Integral-Derivative Controllers for Integrator/Deadtime Processes. *Industrial Engineering Chemical Research*, Vol. 35, pp. 3480-3483.

Nishikawa Y., Sannomiya N., Otha T. and Tanaka H. (1994). A Method for Auto-tuning of PID Control Parameters, *Automatica*, Vol. 20, No 3, pp. 321-332.

Porter, B. and A. H. Jones (1992). Genetic Tuning of Digital PID Controllers, *Electronics Letters*, Vol. 28, No. 9, pp. 843-844

Salami, M. and Cain, G. (1994). An Adaptive PID controller based on genetic algorithm processor. *Proc. of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems*, Sheffield (UK), pp. 88-93.

Shi Y. and Eberhart R. C. (1998). Parameter Selection in Particle Swarm Optimisation, Evolutionary Programming VII, *Lecture Notes in Computer Science 1447*, pp. 591-600, Springer Verlag.

Ziegler J. G. and Nichols N. B., (1942). Optimum Settings for Automatic Controllers, *Transactions of the ASME*, **64**, pp. 759-768.