Expert Systems with Applications 36 (2009) 9159–9167

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Evolutionary algorithms based design of multivariable PID controller

M. Willjuice Iruthayarajan *, S. Baskar

Department of EEE, Thiagarajar College of Engineering, Madurai 625 015, Tamilnadu, India

ARTICLE INFO

Keywords:
PID Control
Evolutionary algorithm
MIMO system
On-line tuning
Off-line tuning

ABSTRACT

In this paper, performance comparison of evolutionary algorithms (EAs) such as real coded genetic algorithm (RGA), modified particle swarm optimization (MPSO), covariance matrix adaptation evolution strategy (CMAES) and differential evolution (DE) on optimal design of multivariable PID controller design is considered. Decoupled multivariable PI and PID controller structure for Binary distillation column plant described by Wood and Berry, having 2 inputs and 2 outputs is taken. EAs simulations are carried with minimization of IAE as objective using two types of stopping criteria, namely, maximum number of functional evaluations (Fevalmax) and Fevalmax along with tolerance of PID parameters and IAE. To compare the performances of various EAs, statistical measures like best, mean, standard deviation of results and average computation time, over 20 independent trials are considered. Results obtained by various EAs are compared with previously reported results using BLT and GA with multi-crossover approach. Results clearly indicate the better performance of CMAES and MPSO designed PI/PID controller on multivariable system. Simulations also reveal that all the four algorithms considered are suitable for off-line tuning of PID controller. However, only CMAES and MPSO algorithms are suitable for on-line tuning of PID due to their better consistency and minimum computation time.

1. Introduction

Proportional-integral-derivative (PID) control offers the simplest and yet most efficient solution to many real-world control problems. Three-term functionality of PID controller covers treatment of both transient and steady state responses. The popularity of PID control has grown tremendously, since the invention of PID control in 1910 and the Ziegler–Nichol's straight forward tuning method in 1942. With the advances in digital technology, the science of automatic control now offers a wide spectrum of choices for control schemes such as adaptive control (Astrom & Wittenmark, 1995), neural network control (Fukuda & Shibata, 1992) and fuzzy logic control (Lee, 1990). However more than 90% of industrial controllers are still implemented based around PID control algorithms, as no other controllers match the simplicity, clear functionality, applicability and ease of use offered by the PID controllers (Ang, Chang, & Li, 2005).

Several approaches have been reported in literature for tuning the parameters of PID controllers. Ziegler–Nichols and Cohen–Coon are the most commonly used conventional methods for tuning PID controllers and neural network, fuzzy based approach, neuro-fuzzy approach and evolutionary computation techniques are the recent methods (Astrom & Hagglund, 1995).

Many researches have already reported the optimal design of PID controller parameters using various EAs such as GA (Chen, Cheng, & Lee, 1995), MPSO (Gaing, 2004; Ghoshal, 2004; Mukherjee & Ghoshal, 2007; Wang, Zhang, & Wang, 2006) and DE (Bingul, 2004) for SISO system. In general, EAs are robust search and optimization methodology, able to cope with ill-defined problem domain such as multimodality, discontinuity, time-variance, randomness and noise. GA approach for tuning of PID controllers for multi-input multi-output (MIMO) process is also reported (Chang, 2007; Zuo, 1995).

In Chang (2007), decoupled multivariable PI controller tuning using GA with multi-parent crossover approach was presented. Simple three-parent differential crossover and uniform mutation operators have been employed. The better performance of three-parent crossover RGA over BLT and traditional two-parent crossover based RGA was demonstrated in the paper.

Recently, several modifications are carried out in crossover and mutation mechanisms of RGA such as SBX crossover, PCX crossover and non-uniform polynomial mutation to improve the performance of RGA. Self-adaptive simulated binary crossover (SBX) based RGA was successfully applied to various engineering optimization problems (Deb, 2001). SBX crossover is self-adaptive in nature which creates children solutions in proportion to the difference in parent solutions. The near parent solutions are monotonically more likely to be chosen as offspring than solutions distant from parents.

* Corresponding author. Tel.: +91 94434 87093.
E-mail address: willjuice@tce.edu (M.W. Iruthayarajan).

0957-4174/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2008.12.033

Another EA, namely, covariance matrix adaptation evolution strategy (CMAES) with the ability of learning of correlations between parameters and the use of the correlations to accelerate the convergence of the algorithm is recently proposed. Due to the learning process, the CMAES algorithm performs the search independent of the coordinate system, reliably adapts topologies of arbitrary functions, and significantly improves convergence rate especially on non-separable and/or badly scaled objective functions. CMAES algorithm has been successfully applied in varieties of engineering optimization problems (Baskar, Alphones, Suganthan, Ngo, & Zheng, 2005).This algorithm outperforms all other similar classes of learning algorithms on the benchmark multimodal functions (Kern et al., 2004).

Covariance matrix adaptation evolution strategy algorithm and also recent modifications in other EAs were not applied for the tuning of PID controllers. Also, all the reported papers for EA based PID controller design, have considered one or two algorithms for the purpose of comparison.

This paper focuses mainly on the performance evaluation of various EAs such as Self-adaptive RGA, MPSO, DE and CMAES on optimum design of multivariable PI and PID controllers for binary distillation column plant described by Wood and Berry (Chang, 2007). The essence of the paper lies in the determination of suitable EA method for the tuning of PID controller for MIMO system.

The remaining part of the paper is organized as follows. Section 2 introduces PID controller structure for SISO and MIMO systems. Section 3 describes the various EAs methods. Section 4 introduces the MIMO system considered for PID controller tuning. Section 5 presents the implementation of EA based multivariable PID controller design. Section 6 reveals the simulation results. Finally, conclusions are given in Section 7.

2. PID controller structure

A standard PID controller structure is also known as the "three-term" controller, whose transfer function is generally written in the ideal form in (1) or in the parallel form in (2)

$$G(s) = K_P\left(1 + \frac{1}{T_I s} + T_D s\right) \quad (1)$$

$$G(s) = K_P + \frac{K_I}{s} + K_D s \quad (2)$$

where K_P is the proportional gain, T_I is the integral time constant, T_D is the derivative time constant, $K_I = K_P/T_I$ is the integral gain and $K_D = K_P T_D$ is the derivative gain.

The "three-term" functionalities are highlighted below.

- The proportional term – providing an overall control action proportional to the error signal through the all pass gain factor.
- The integral term – reducing steady state errors through low frequency compensation by an integrator.
- The derivative term – improving transient response through high frequency compensation by a differentiator.

For optimum performance, K_P, K_I (or T_I) and K_D (or T_D) are tuned by EAs by minimizing the performance measures such as IAE, ISE and ITAE.

2.1. PID controller for MIMO system

Consider a multivariable PID control structure as in Fig. 1, where, desired output vector: $Y_d = [y_{d1}, y_{d2}, \ldots, y_{dn}]^T$;

Actual output vector: $Y = [y_1, y_2, \ldots, y_n]^T$;

Error vector: $E = Y_d - Y = [y_{d1} - y_1, y_{d2} - y_2, \ldots, y_{dn} - y_n]^T = [e_1, e_2, \ldots, e_n]^T$;

Control input vector: $U = [u_1, u_2, \ldots, u_n]^T$;

$n \times n$ Multivariable processes:

$$G(s) = \begin{bmatrix} g_{11}(s) & \cdots & g_{1n}(s) \\ \vdots & \ddots & \vdots \\ g_{n1}(s) & \cdots & g_{nn}(s) \end{bmatrix} \quad (3)$$

$n \times n$ Multivariable PID controller:

$$K(s) = \begin{bmatrix} k_{11}(s) & \cdots & k_{1n}(s) \\ \vdots & \ddots & \vdots \\ k_{n1}(s) & \cdots & k_{nn}(s) \end{bmatrix} \quad (4)$$

In this work, decoupled multivariable PID controller is considered. So $K(s)$ becomes

$$K(s) = \begin{bmatrix} k_1(s) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & k_n(s) \end{bmatrix} \quad (5)$$

The form of $k_i(s)$ is either in (1) or (2). In this work, "parallel form" of PID controller in (2) is used and can be rewritten as

$$k_i(s) = k_{P_i} + \frac{k_{I_i}}{s} + k_{D_i} s, \quad (6)$$

For convenience, let $\theta_i = [k_{P_i}, k_{I_i}, k_{D_i}]$, represents the gains vector of ith diagonal sub PID controller in $K(s)$ .For multivariable PI controller, $k_i(s)$ in (6) can be rewritten as

$$k_i(s) = k_{P_i} + \frac{k_{I_i}}{s} \quad (7)$$

$\theta_i = [k_{P_i}, k_{I_i}]$ represents the gains vector of the ith diagonal sub PI controller in $K(s)$.

2.2. Performance index

In the design of PID controller, the performance criterion or objective function is first defined based on the desired specifications such as time-domain specifications, frequency domain specifications and time-integral performance. The commonly used time-integral performance indexes are integral of the square error (ISE), integral of the absolute value of the error (IAE) and integral of the time-weighted absolute error (ITAE). Minimization of IAE as given in (8) is considered as the objective of this paper

$$\text{IAE} = \int_0^\infty (|e_1(t)| + |e_2(t)| + \cdots + |e_n(t)|)\,\text{d}t \quad (8)$$

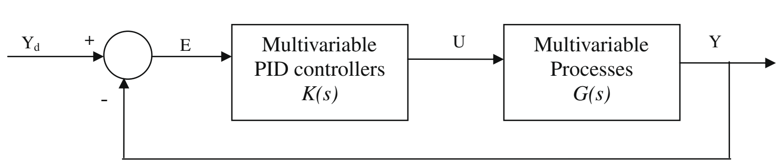

Fig. 1. A multivariable PID control system.

3. Evolutionary algorithms

Evolutionary algorithms differ from the traditional optimization techniques in that EAs make use of a population of solutions, not a single point solution. EAs are inherently parallel, because they simultaneously evaluate many points in the parameter space (search space). Considering many points in the search space, EA has a reduced chance of converging to the local optimum and would be more likely to converge to the global optimum. An iteration of EA involves a competitive selection that weeds out poor solutions and offspring generation mechanism. Several evolutionary search algorithms like GA, EP, MPSO, DE and CMAES were developed independently. These algorithms differ in selection, offspring generation and replacement mechanisms. For solving global functional optimization problems, RGA, MPSO DE and CMAES algorithms are normally used and hence these algorithms are employed in this paper. Over the years, several variants of these algorithms were proposed. Some of the recent variants of these algorithms are briefly explained in this section.

3.1. Real coded genetic algorithm with SBX crossover

In general, a genetic algorithm has five components as follows:

1. A genetic representation of solutions to the problem.
2. A way to create an initial population of solutions.
3. An evaluation function rating solution in terms of its fitness.
4. Parent selection mechanism and genetic operators that alter the genetic composition of children during reproduction.
5. Values for the parameter of genetic algorithm.

Real-number encoding is best used for function optimization problems. It has been widely confirmed that real-number encoding performs better than binary or gray encoding for constrained optimization. Owing to the adaptive capability, SBX crossover and polynomial mutation operators are employed. Tournament selection is used as selection mechanism in order to avoid premature convergence. Simulated binary crossover (SBX) and polynomial mutation are briefly explained below.

3.1.1. Simulated binary crossover

In SBX crossover (Deb, 2001), two children solutions are created from two parents as follows:

Choose a random number $u_i, \in [0,1]$ and calculate β_{qi} as given in (9)

$$\beta_{qi} = \begin{cases} (2u_i)^{\frac{1}{\eta_c+1}}, & u_i \leqslant 0.5 \\ \left(\frac{1}{2(1-u_i)}\right)^{\frac{1}{\eta_c+1}}, & \text{otherwise} \end{cases} \tag{9}$$

A spread factor β_{qi} is defined as the ratio of the absolute difference in offspring values to that of the parents. η_c is the crossover index.

Then compute the offspring $x_i^{(1,t+1)}$ & $x_i^{(2,t+1)}$ as

$$\begin{aligned} x_i^{(1,t+1)} &= 0.5\left[(1+\beta_{qi})x_i^{(1,t)} + (1-\beta_{qi})x_i^{(2,t)}\right] \\ x_i^{(2,t+1)} &= 0.5\left[(1-\beta_{qi})x_i^{(1,t)} + (1+\beta_{qi})x_i^{(2,t)}\right]. \end{aligned} \tag{10}$$

3.1.2. Polynomial mutation

Newly generated offspring undergoes polynomial mutation operation. Like in the SBX operator, the probability distribution can also be a polynomial function, instead of a normal distribution. The new offspring $y_i^{(1,t+1)}$ is determined as follows:

$$y_i^{(1,t+1)} = x_i^{(1,t+1)} + (x_i^U - x_i^L)\overline{\delta_i}. \tag{11}$$

x_i^U and x_i^L are the upper and lower limit values.where the parameter $\overline{\delta_i}$ is calculated from the polynomial probability distribution

$$P(\delta) = 0.5(\eta_m+1)(1-|\delta|)^{\eta_m}$$

$$\overline{\delta_i} = \begin{cases} (2r_i)^{1/(\eta_m+1)} - 1, & \text{if} \quad r_i < 0.5 \\ 1 - [2(1-r_i)]^{1/(\eta_m+1)}, & \text{if} \quad r_i \geqslant 0.5 \end{cases} \tag{12}$$

where η_m is the mutation index.

Newly generated individuals replace their parents and form the parents for the next generation and the above procedure is repeated until a maximum number of function evaluations are completed.

3.2. Modified particle swarm optimization (MPSO)

Particle swarm optimization provides a population-based search procedure in which individuals called particles, change their positions (states) with time. In a PSO system, particles fly around in a multidimensional search space. During flight, each particle adjusts its position according to its own experience, and the experience of neighboring particles, making use of the best position encountered by itself and its neighbors. The swarm direction of a particle is defined by the set of particles neighboring the particle and its history of experience.

Let X_i and V_i represent ith particle position and its corresponding velocity in a search space. The best previous position of ith particle is recorded and represented as $Pbest_i$. The best particle among all the particles in the group is represented as $Gbest$. The updated velocity of individual i is given in (13) (Shi & Eberhart, 1998)

$$\begin{aligned} V_i^{k+1} = \omega^k V_i^k &+ c_1\ rand_1\left(Pbest_i^k - X_i^k\right) \\ &+ c_2\ rand_2\left(Gbest^k - X_i^k\right) \end{aligned} \tag{13}$$

where

V_i^k	Velocity of ith individual at iteration k
ω^k	Inertia weight at iteration k
c_1, c_2	Acceleration factors
$rand_1, rand_2$	Uniform random numbers between 0 and 1
X_i^k	Position of ith individual at iteration k
$Pbest_i^k$	Best position of ith individual at iteration k
$Gbest^k$	Best position of the group until iteration k

Each individual moves from the current position to the next one with the modified velocity as

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{14}$$

The above procedure is repeated until a maximum number of functional evaluations have been reached.

3.3. Differential evolution

Differential Evolution is a simple population-based, stochastic parallel search evolutionary algorithm for global optimization (Storm & Price, 1997). The initial population is chosen randomly and should cover the entire parameter space. At each generation, DE employs both mutation and crossover (recombination) to produce one trial vector $U_{i,J+1}$ for each target vector $X_{i,J}$. Then, a selection phase takes place, where each trial vector is compared with the corresponding target vector; the better one will enter the population of the next generation. For each target vector $X_{i,J}$, a mutant vector is generated using Rand/Rand strategy as

$$U_{i,J+1} = X_{r3,J} + F(X_{r1,J} - X_{r2,J}) \tag{15}$$

where

$U_{i,J+1}$	ith mutated individual for the next iteration
X	Population set
F	Mutation constant [0,2]
J	Current iteration
$J+1$	Next iteration
$X_{r1,J}\ldots X_{r3,J}$	Randomly selected individuals from the population of the current iteration.

To increase the diversity of the perturbed parameter vectors crossover is performed after mutation

$$V_{i,J+1} = X_{i,j}(1 - \mathrm{CR}) + U_{i,J+1}\mathrm{CR} \tag{16}$$

where, CR = crossover probability constant from interval [0,1]

The parents for the next iteration are selected as follows:

$$X_{i,J+1} = \begin{cases} V_{i,J+1} & \text{if } f(V_{i,J+1}) > f(X_{i,J+1}) \\ X_{i,J+1} & \text{if } f(X_{i,J+1}) > f(V_{i,J+1}) \end{cases} \tag{17}$$

where $f(V_{i,J+1})$ is the fitness function value of the ith individual of the population to which the mutation and crossover operators are applied and $f(X_{i,J+1})$ is the fitness function value of the ith individual in the original population. The loss of the best individuals in the following iteration is avoided by this selection mechanism, as the worst individuals are replaced by the best individuals. This process continues until the maximum function evaluation is reached.

3.4. Covariance matrix adaptation evolution strategy (CMAES)

Covariance matrix adaptation evolution strategy is a class of continuous EA that generates new population members by sampling from a probability distribution that is constructed during the optimization process. One of the key concepts of this algorithm involves the learning of correlations between parameters and the use of the correlations to accelerate the convergence of the algorithm. The adaptation mechanism of CMAES consists of two parts, (1) the adaptation of the covariance matrix **C** and (2) the adaptation of the global step size. The covariance matrix **C** is adapted by the evolution path and difference vectors between the μ best individuals in the current and previous generation. The detailed CMAES algorithm is presented in Kern et al. (2004).

3.4.1. CMAES algorithm

Step 1: Generate an initial random solution.

Step 2: The offspring at $g + 1$ generation x_k^{g+1} are sampled from a Gaussian distribution using covariance matrix and global step size at generation g

$$x_k^{(g+1)} = z_k,\ z_k = N\left(\langle x\rangle_\mu^{(g)}, \sigma^{(g)^2}\mathbf{C}^{(g)}\right) \quad k = 1, \ldots, \lambda \tag{18}$$

where $\langle x\rangle_\mu^{(g)} = \sum_{i=1}^{\mu} x_i^{(g)}$ with μ being the selected best individuals from the population.

The parameters c_c, c_{cov}, c_σ and d required for further computations are by default given in terms of the number of decision variables (n) and μ as follows:

$$\begin{aligned} &c_c = \frac{4}{n+4}, \quad c_\sigma = \frac{10}{n+20}, \quad d = \max\left(1, \frac{3\mu}{n+10}\right) + c_\sigma, \\ &c_{cov} = \frac{1}{\mu}\frac{2}{(n+\sqrt{2})^2} + \left(1 - \frac{1}{\mu}\right)\min\left(1, \frac{2\mu - 1}{(n+2)^2 + \mu}\right) \end{aligned} \tag{19}$$

The parameters c_σ and c_{cov} control independently the adaptation time scales for the global step size and the covariance matrix.

Note that if $\mu \gg n,d$ is large and the change in σ is negligible compared to that of **C**. The initial values are $\mathbf{P}_\sigma^{(0)} = \mathbf{P}_c^{(0)} = 0$ and $\mathbf{C}^{(0)} = \mathbf{I}$.

Step 3: The evolution path $\mathbf{P}_c^{(g+1)}$ is computed as follows:

$$\mathbf{P}_c^{(g+1)} = (1 - c_c)\cdot\mathbf{P}_c^{(g)} + \sqrt{c_c(2 - c_c)}\cdot\frac{\sqrt{\mu}}{\sigma^{(g)}}\left(\langle x\rangle_\mu^{(g+1)} - \langle x\rangle_\mu^{g}\right) \tag{20}$$

$$\begin{aligned} \mathbf{C}^{(g+1)} = {} & (1 - c_{cov})\cdot\mathbf{C}^{(g)} + c_{cov} \\ & \cdot\left(\frac{1}{\mu}\mathbf{P}_c^{(g+1)}\left(\mathbf{P}_c^{(g+1)}\right)^{\mathrm{T}} + \left(1 - \frac{1}{\mu}\right)\frac{1}{\mu}\sum_{i=1}^{\mu}\frac{1}{\sigma^{(g)^2}}\left(x_i^{(g+1)} - \langle x\rangle_\mu^{(g)}\right)\right. \\ & \left.\left(x_i^{(g+1)} - \langle x\rangle_\mu^{(g)}\right)^{\mathrm{T}}\right) \end{aligned} \tag{21}$$

The strategy parameter $c_{cov} \in [0, 1]$ determines the rate of change of the covariance matrix **C**.

Step 4: Adaptation of global step size $\sigma^{(g+1)}$ is based on a conjugate evolution path $\mathbf{P}_\sigma^{(g+1)}$

$$\begin{aligned} \mathbf{P}_\sigma^{(g+1)} = {} & (1 - c_\sigma)\cdot\mathbf{P}_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)}\cdot\mathbf{B}^{(g)}(\mathbf{D}^{(g)})^{-1}(\mathbf{B}^{(g)})^{-1} \\ & \times\frac{\sqrt{\mu}}{\sigma^{(g)}}\left(\langle x\rangle_\mu^{(g+2)} - \langle x\rangle_\mu^{g}\right) \end{aligned} \tag{22}$$

the matrices $\mathbf{B}^{(g)}$ and $\mathbf{D}^{(g)}$ are obtained through a principal component analysis:

$$\mathbf{C}^{(g)} = \mathbf{B}^{(g)}(\mathbf{D}^{(g)})^2(\mathbf{B}^{(g)})^{\mathrm{T}} \tag{23}$$

where the columns of $\mathbf{B}^{(g)}$ are the normalized eigen vectors of $\mathbf{C}^{(g)}$ and $\mathbf{D}^{(g)}$ is the diagonal matrix of the square roots of the given eigen values of $\mathbf{C}^{(g)}$. The global step size $\sigma^{(g+1)}$ is determined by

$$\sigma^{(g+1)} = \sigma^{(g)}\exp\left(\frac{c_\sigma}{d}\left(\frac{\|\mathbf{P}_\sigma^{(g+1)}\|}{E(\|N(0,\mathbf{I})\|)}\right) - 1\right) \tag{24}$$

Step 5: Repeat Steps 2–4 until the stopping criteria are satisfied.

4. MIMO system

Most of the industrial processes belong to the category of MIMO system, which requires MIMO control techniques to improve performance, even though they are naturally more difficult to exploit than SISO system. Binary distillation column plant described by Wood and Berry (Chang, 2007; Wang, Zou, Lee, & Qiang, 1997) is considered. The transfer function of the above process is given in (25)

$$G(s) = \begin{bmatrix} \frac{12.8e^{-s}}{1+16.7s} & \frac{-18.9e^{-3s}}{1+21s} \\ \frac{6.6e^{-7s}}{1+10.9s} & \frac{-19.4e^{-3s}}{1+14.4s} \end{bmatrix} \tag{25}$$

The transfer function concerned with multivariable process has first order dynamics and significant time delays and it has a strong interaction between inputs and outputs. In this paper, multivariable controller with PI and PID structures are used for optimizing the IAE performance for set point regulation using EAs.

5. EA implementation of multivariable PID controller

System described by (25), has two inputs and two outputs. The decoupled multivariable PID controller $K(s)$ for this system is given in (26)

$$K(s) = \begin{bmatrix} k_1(s) & 0 \\ 0 & k_2(s) \end{bmatrix} \tag{26}$$

In order to obtain the optimum performance, the parameters of $K(s)$ i.e., $[\theta_1, \theta_2] = [k_{P_1}, k_{I_1}, k_{D_1}, k_{P_2}, k_{I_2}, k_{D_2}]$ are optimized by optimization algorithms. The chromosome/particle representation is given in Fig. 2.

First three elements are the parameters of $k_1(s)$ and next three elements are the parameters of $k_2(s)$. For multivariable PI controller, the structure of the chromosome is given in Fig. 3. First two

k_{P_1}	k_{I_1}	k_{D_1}	k_{P_2}	k_{I_2}	k_{D_2}

Fig. 2. Chromosome/particle of multivariable PID controller.

k_{P_1}	k_{I_1}	k_{P_2}	k_{I_2}

Fig. 3. Chromosome/particle of multivariable PI controller.

elements are the parameters of $k_1(s)$ and the next two elements are the parameters of $k_2(s)$.

All the elements of chromosomes/particles of population are randomly initialized within the search space specified by their lower and upper bounds of individual parameters as given in (Chang, 2007). The inequality conditions for the parameter ranges of PI and PID controllers are given in (27) and (28), respectively

$$-1 \leqslant K_{P_1} \leqslant 1, \quad -1 \leqslant K_{I_1} \leqslant 1, \quad -1 \leqslant K_{P_2} \leqslant 1, \quad -1 \leqslant K_{I_2} \leqslant 1 \tag{27}$$

$$\begin{aligned} &-1 \leqslant K_{P_1} \leqslant 1, \quad -1 \leqslant K_{I_1} \leqslant 1, \quad -1 \leqslant K_{D_1} \leqslant 1, \\ &-1 \leqslant K_{P_2} \leqslant 1, \quad -1 \leqslant K_{I_2} \leqslant 1, \quad -1 \leqslant K_{D_2} \leqslant 1. \end{aligned} \tag{28}$$

6. Simulation results

In this work, two experiments namely design of multivariable PI and PID controller for binary distillation column plant described by Wood and Berry, using various EAs such as RGA, MPSO, DE and CMAES are conducted. For simulating Binary distillation column plant, MATLAB-SIMULINK software is employed. Simulations are carried out using Core 2 Duo Processor 2.2 GHz, 2GB RAM PC. IAE is determined for step response over 150 min time period. EAs simulations are carried out using two types of stopping criteria namely, Fevalmax, Fevalmax with tolerance of design variables and tolerance of objective function. In this work, the Fevalmax is set at 6000 functional evaluations and tolerance is fixed as 10^{-5} for the last 20 generations. Owing to the randomness of the EAs, many trials with independent population initializations should be made to acquire a useful conclusion of the performance of the algorithm. Hence, best, mean, standard deviation of IAE measure and average computation in 20 independent trials of various EAs are reported and compared with the already reported results (Chang, 2007).

6.1. Parameter tuning

Optimal parameter combinations for different EA methods are experimentally determined by conducting a series of experiments with different parameter settings before conducting actual runs to collect the results. The parameters actually used in the simulations are summarized in the Table 1.

6.2. Tuning of multivariable PI controller

Best PI parameters and the corresponding IAE values for the 20 trials of multivariable PI controllers using different EAs with and without tolerance in stopping criteria are reported in Tables 2 and 3, respectively. For the purpose of comparison, already reported values obtained by conventional BLT method and GA with multi-crossover approach are directly taken from (Chang, 2007) and given in the Tables.

Table 1
Parameter selection.

Evolutionary algorithms	Parameter
RGA	$P_c = 0.8$
	$P_m = 1/n$
	$\eta_c = 5$
	$\eta_m = 20$
MPSO	$C_1 = 1$
	$C_2 = 1$
	$V_{max} = 0.1$
	ω is linearly decreased from 0.9 to 0.2 over the iterations
DE	CR = 0.5
	$F = 0.8$
CMAES	Parameters are fixed automatically by the algorithm

Table 2
Optimum parameters of multivariable PI controller – without tolerance.

Method	Optimum parameters of multivariable PI controller				IAE
	k_{P_1}	k_{I_1}	k_{P_2}	k_{I_2}	
BLT*	0.3750	0.0452	−0.0750	−0.0032	23.5568
RGA-multi-crossover*	0.9971	0.0031	−0.0141	−0.0071	10.5778
RGA-SBX crossover	0.8433	0.0026	−0.0127	−0.0069	10.4395
MPSO	0.8485	0.0026	−0.0132	−0.0069	10.4378
DE	0.8485	0.0026	−0.0132	−0.0069	10.4378
CMAES	0.8485	0.0026	−0.0132	−0.0069	10.4378

* Data taken from Chang (2007).

Table 3
Optimum parameters of multivariable PI controller – with tolerance.

Method	Optimum parameters of multivariable PI controller				IAE
	k_{P_1}	k_{I_1}	k_{P_2}	k_{I_2}	
BLT*	0.3750	0.0452	−0.0750	−0.0032	23.5568
RGA-multi-crossover*	0.9971	0.0031	−0.0141	−0.0071	10.5778
RGA-SBX crossover	0.8622	0.0026	−0.0135	−0.0069	10.44
MPSO	0.8485	0.0026	−0.0132	−0.0069	10.4378
DE	0.8485	0.0026	−0.0132	−0.0069	10.4379
CMAES	0.8485	0.0026	−0.0132	−0.0069	10.4378

* Data taken from Chang (2007).

Table 4
Statistical performance of EAs of multivariable PI controller – without tolerance.

Method	Best value	Mean value	Standard deviation	Average computation time (s)
RGA-SBX crossover	10.4395	10.9366	1.6638	137.8698
MPSO	10.4378	10.8157	1.6570	135.2763
DE	10.4378	10.4379	4.5272E-5	137.9376
CMAES	10.4378	10.4378	0	142.5264

Table 5
Statistical performance of EAs of multivariable PI controller – tolerance.

Method	Best value	Mean value	Standard deviation	Average computation time (s)	Average functional evaluations
RGA-SBX crossover	10.44	10.4961	0.0810	134.8504	5840
MPSO	10.4378	10.5394	0.3410	110.3138	4805
DE	10.4379	11.2228	1.5740	107.3965	4676
CMAES	10.4378	10.4378	0	84.8255	3572

Almost all the EAs are giving equal performance with respect to the best IAE. The statistical performances such as best, mean, standard deviation of IAE and average computation time (ACT) of 20 trials using various EAs with and without considering tolerance are given in Tables 4 and 5, respectively. Without considering tolerance in stopping criteria, CMAES and DE are the same with respect to mean value but CMAES is computationally slightly expensive due to complex mathematical manipulations. While considering tolerance, CMAES algorithm gives better performance in consistently achieving good results as compared to all other algorithms and also it requires less number of average functional evaluations (AFeval). MPSO algorithm gives better performance than DE and RGA.

Fig. 4 shows the convergence characteristics of various EAs considering tolerance. Higher value of IAE during the initial generations/iterations indicates unstable PID control settings. For

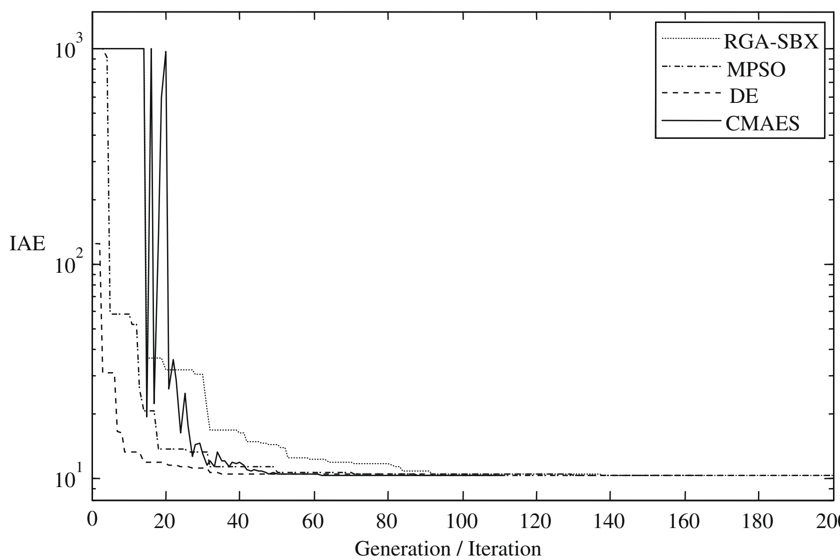

Fig. 4. Convergence characteristics of EAs – multivariable PI controller.

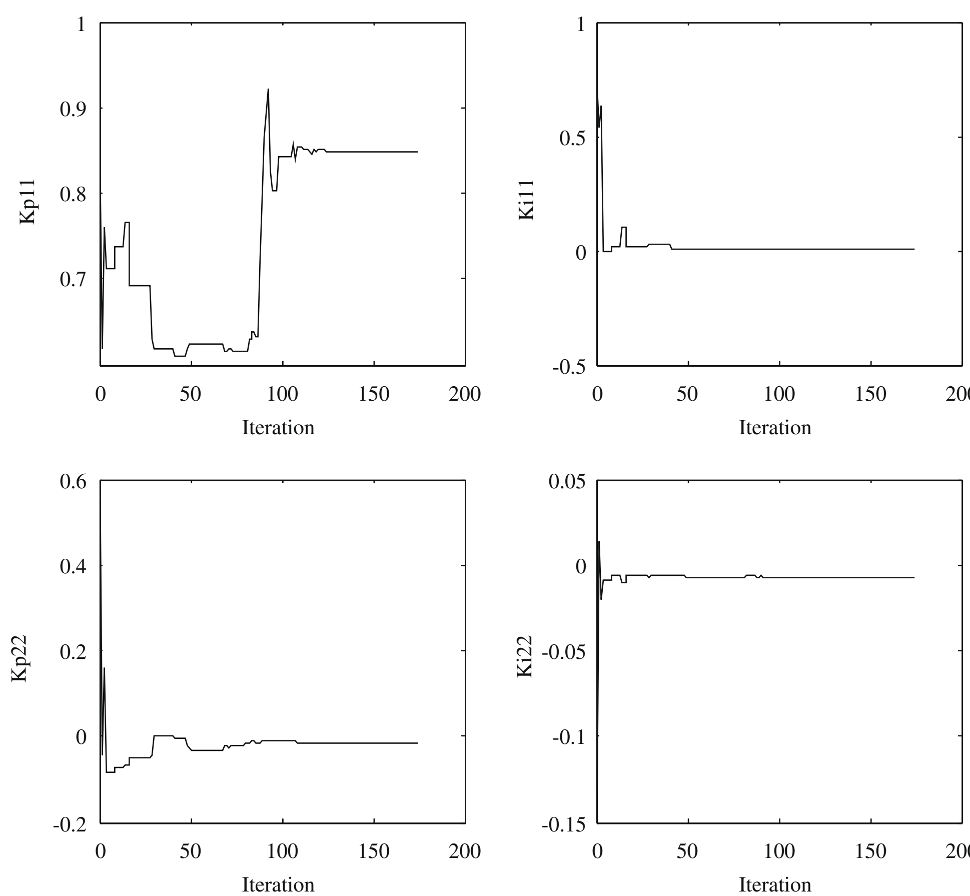

Fig. 5. Convergence characteristics of PID parameters using MPSO.

convenience, IAE for the unstable response is limited to 1000. Other than CMAES algorithm, the convergence characteristics of all algorithms are smooth. Due to the self-learning behavior of CMAES algorithm, convergence characteristics show large variations during the initial search. Fig. 5 shows the convergence characteristics of multivariable PI parameters obtained by MPSO algorithm with tolerance in stopping criteria.

Output responses y_1 and y_2 of the MIMO system with multivariable PI controller using best PI parameter obtained out of the 20 trials by using CMAES/MPSO algorithm is shown in Figs. 6 and 7, respectively. For comparison purpose, output responses using RGA with multi-crossover approach are also given in the same figure. PI controller simulation results show the better performance of CMAES and MPSO algorithms as compared to the previously reported results obtained using GA with multi-crossover approach (Chang, 2007) and conventional BLT method. Time-responses specifications for the designed PI controllers such as peak overshoot in%, rise time (min) and settling time for ±5% tolerance (min) for output responses y_1 and y_2 are summarized in Table 6. Peak overshoot of system response with multivariable PI controller designed by CMAES/MPSO is approximately 50% lesser than GA with multi-crossover approach with a slight increase in rise time and settling time.

6.3. Tuning of multivariable PID controller

Best PID parameters and the corresponding IAE values for the 20 trials of multivariable PID controllers using various EAs with and without considering tolerance in stopping criteria are reported in Tables 7 and 8, respectively. The CMAES and MPSO algorithms are almost giving equal performance with respect to the best IAE. The statistical performances such as best, mean, standard deviation of IAE, ACT and AFeval of 20 trials using various EAs with and without considering tolerance are given in Tables 9 and 10, respectively. CMAES, MPSO and RGA algorithms give better performance in consistently achieving good results as compared to DE. But AFeval required and ACT for RGA is larger than CMAES and MPSO.

For clarity, transient portion (25 min) of output responses y_1 and y_2 of the MIMO system with multivariable PID controller using best PID parameter obtained using CMAES/MPSO algorithm is shown in Figs. 8 and 9, respectively. For comparison purpose, output responses using RGA with multi-crossover approach and multivariable PI controller designed by CMAES are also given in the same figure. PID controller simulation results show the better performance of CMAES and MPSO algorithms as compared to the previously reported results obtained using GA with multi-crossover approach (Chang, 2007) and multivariable PI controller designed

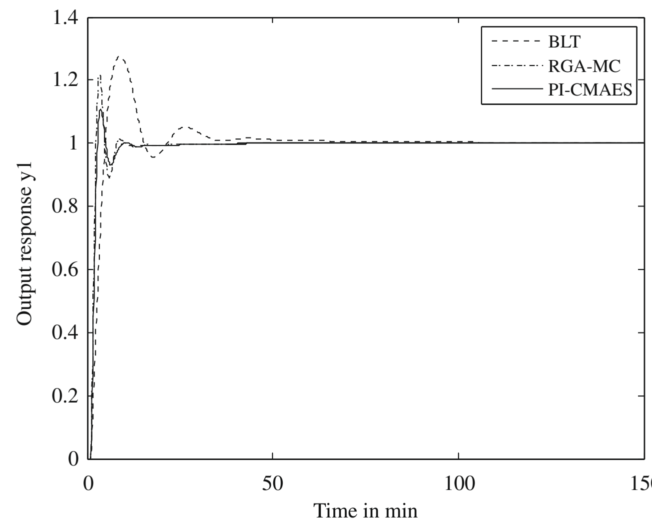

Fig. 6. Output response y_1.

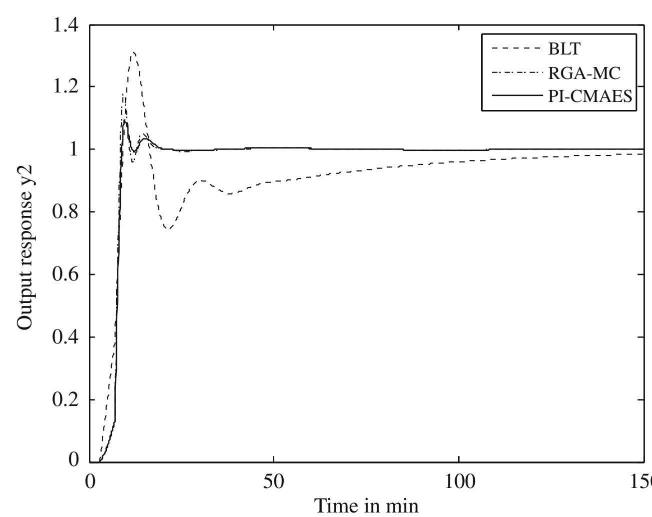

Fig. 7. Output response y_2.

Table 6
Time-response specifications of output responses – multivariable PI controller.

Method	y_1			y_2		
	%Mp	Rise time (min)	Settling time 5% (min)	%Mp	Rise time (min)	Settling time 5% (min)
BLT	27.4365	4.9	32.2	30.9972	9.2	88.2
RGA with multi-crossover	21.9866	2.5	7.1	17.8161	8.5	14.9
CMAES	10.8390	2.8	7.3	9.2414	8.9	10.6

Table 7
Optimum parameters of multivariable PID controller – without tolerance.

Method	Optimum PID parameters						IAE
	k_{P_1}	k_{I_1}	k_{D_1}	k_{P_2}	k_{I_2}	k_{D_2}	
RGA-SBX crossover	1.0	0.0025	0.3892	−0.0317	−0.0072	−0.0885	9.6859
PID-MPSO	1.0	0.0025	0.3872	−0.0332	−0.0073	−0.0909	9.6824
DE	0.9945	0.0026	0.4021	−0.0289	−0.0071	−0.0709	9.7108
CMAES	1.0	0.0025	0.3872	−0.0332	−0.0073	−0.0909	9.6824

Table 8
Optimum parameters of multivariable PID controller – with tolerance.

Method	Optimum PID parameters						IAE
	k_{P_1}	k_{I_1}	k_{D_1}	k_{P_2}	k_{I_2}	k_{D_2}	
RGA-SBX crossover	1.0	0.0025	0.3860	−0.0358	−0.0073	−0.0991	9.6871
PID-MPSO	1.0	0.0025	0.3872	−0.0332	−0.0073	−0.0909	9.6824
DE	0.9825	0.0026	0.4239	−0.0563	−0.0082	−0.1563	9.9251
CMAES	1.0	0.0025	0.3872	−0.0332	−0.0073	−0.0909	9.6824

Table 9
Statistical performance of EAs of multivariable PID controller – without tolerance.

Method	Best value	Mean value	Standard deviation	Average computation time
RGA-SBX crossover	9.6859	10.2370	0.6968	136.6054
MPSO	9.6824	10.0619	0.4304	136.6904
DE	9.7479	9.8855	0.1135	138.9478
CMAES	9.6824	10.4451	0.7116	146.4358

Table 10
Statistical performance of EAs of multivariable PID controller – with tolerance.

Method	Best value	Mean value	Standard deviation	Average computation time (s)	Average functional evaluations
RGA-SBX crossover	9.6871	10.0954	0.3128	139.4493	5943
MPSO	9.6824	10.1992	0.6073	133.0369	5705
DE	9.9251	46.4502	85.6846	71.8283	3075
CMAES	9.6824	10.3186	0.7618	138.6396	5677

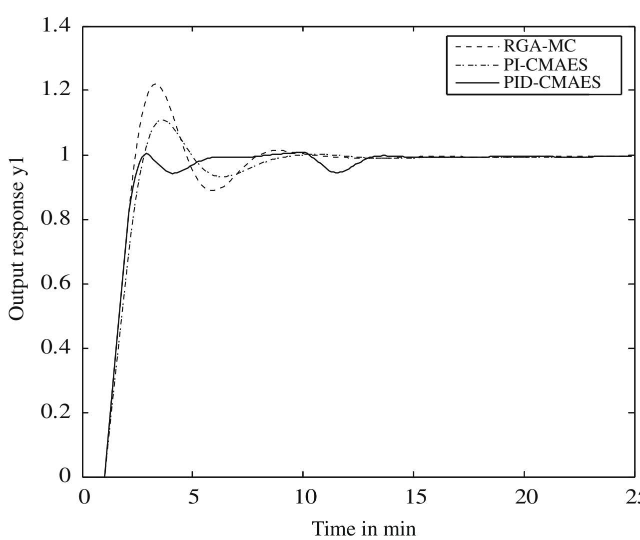

Fig. 8. Transient portion of output response y_1.

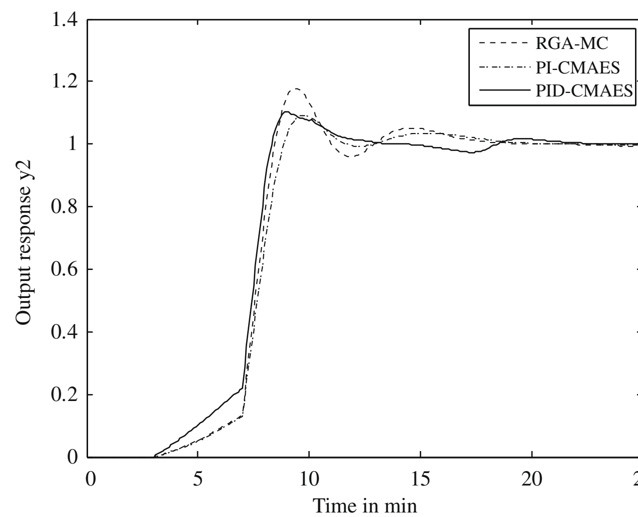

Fig. 9. Transient portion of output response y_2.

Table 11
Time-response specifications of output responses – multivariable PID controller.

Method	y_1			y_2		
	%Mp	Rise time (min)	Settling time 5% (min)	%Mp	Rise time (min)	Settling time 5% (min)
RGA-SBX	0.9602	2.8	12	11.9161	8.3	10.9
DE	2.1241	5.1	12.3	20.7696	8.1	11.7
CMAES/MPSO	0.8059	2.9	11.9	10.4336	8.4	10.6

by CMAES. Time responses of y_1 and y_2 using best PID parameters using various EAs are summarized in Table 11. Results show the better performance of CMAES/MPSO in terms of very less peak overshoot and settling time of response y_1 and y_2 with slight increase in rise time.

7. Conclusions

In this paper, performance evaluation of EAs such as RGA with SBX crossover, MPSO, DE and CMAES on the optimal design of multivariable PI and PID controller for the binary distillation column plant is conducted. EAs simulations are carried out using two types of stopping criteria, namely, Fevalmax and Fevalmax along with tolerance of PID parameters and IAE. Multivariable PI/PID controllers are designed by minimizing IAE and the results are compared with those of the already reported in literature namely, BLT and GA with multi-crossover approach. Simulation results are summarized as follows:

(i) The better performance of evolutionary designed multivariable PI controller over the already reported results. Also, multivariable PID controllers designed for the same system by various EAs are better than multivariable PI controller.
(ii) Without tolerance, the best IAE obtained in 20 independent trials by all EAs is almost equal. This reveals that all EAs are equally applicable to off-line PID controller tuning.
(iii) Considering the tolerance of PID parameters and IAE, CMAES and MPSO algorithms are more suitable for on-line tuning of PID controller due to their better consistency and minimum computation time. Also, MPSO is much more suitable for on-line tuning of PID controller due to the reduced computation time.

Acknowledgments

The authors gratefully acknowledge the Management of the Thiagarajar College of Engineering, Madurai 625 015, Tamilnadu, India, for their continued support, encouragement, and the extensive facilities provided to carry out this research work. They also gratefully acknowledge the support of Dr. M. Chidambaram, Director, NIT, Trichy.

References

Ang, K. H., Chang, G., & Li, Yun (2005). PID control system analysis, design and technology. *IEEE Transaction on Control System Technology, 13*(4), 559–577.

Astrom, K. J., & Wittenmark, B. (1995). *Adaptive control* (2nd ed.). Addison Wesley.

Astrom, K. J., & Hagglund, T. (1995). *PID controllers: theory, design, and tuning* (2nd ed.). Instrument society of America.

Baskar, S., Alphones, A., Suganthan, P. N., Ngo, N. Q., & Zheng, R. T. (2005). Design of optimal length low-dispersion FBG filter using covariance matrix adapted evolution. *IEEE Photonics Technology Letters, 17*(10), 2119–2121.

Bingul, Z. (2004). A new PID tuning technique using differential evolution for unstable and integrating processes with time delay. *ICONIP, Proceedings Lecture Notes in Computer Science, 3316*, 254–260.

Chang, W. D. (2007). A multi-crossover genetic approach to multivariable PID controllers tuning. *Expert Systems with Applications, 33*, 620–626.

Chen, B. S., Cheng, Y. M., & Lee, C. H. (1995). A genetic approach to mixed H_2/H_∞ Optimal PID control. *IEEE Control Systems, 15*(5), 51–60.

Deb, K. (2001). *Multiobjective optimization using evolutionary algorithms*. Chichester, UK: Wiley.

Fukuda, T., & Shibata, T. (1992). Theory and application of neural networks for industrial control systems. *IEEE Transactions on Industrial Electronics, 39*(6), 472–489.

Gaing, Z. L. (2004). A particle swarm optimization approach for optimum design of PID controller in AVR system. *IEEE Transactions on Energy Conversion, 19*(2), 384–391.

Ghoshal, S. P. (2004). Optimizations of PID gains by particle swarm optimizations in fuzzy based automatic generation control. *Electric Power Systems Research, 72*, 203–212.

Kern, S., Müller, S. D., Hansen, N., Büche, D., Ocenasek, J., & Koumoutsakos, P. (2004). Learning probability distributions in continuous evolutionary algorithms – A comparative review. *Natural Computation, 3*(1), 77–112.

Lee, C. C. (1990). Fuzzy logic in control systems: Fuzzy logic controller – Part I and II. *IEEE Transactions on Systems Man and Cybernetics, 20*(2), 404–435.

Mukherjee, V., & Ghoshal, S. P. (2007). Intelligent particle swarm optimized fuzzy PID controller for AVR system. *Electric Power Systems Research, 77*(12), 1689–1698.

Shi, Y., & Eberhart, R. C. (1998). A modified particle swarm optimizer. *Proceedings of IEEE International Conference on Evolutionary Computation*, 69–73. Anchorage, AK.

Storm, R., & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*, 341–359.

Wang, J. S., Zhang, Y., & Wang, W. (2006). Optimal design of PI/PD controller for non-minimum phase system. *Transactions of the Institute of Measurement and Control, 28*(1), 27–35.

Wang, Q. G., Zou, B., Lee, T. H., & Qiang, B. (1997). Auto-tuning of multivariable PID controllers from decentralized relay feedback. *Automatica, 33*(3), 319–330.

Zuo, W. (1995). Multivariable adaptive control for a space station using genetic algorithms. *IEE Proceedings – Control Theory and Applications, 142*(2), 81–87.