Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A closed-loop particle swarm optimizer for multivariable process controller design

Kai HAN[†], Jun ZHAO[†‡], Zu-hua XU, Ji-xin QIAN

(*State Key Lab of Industrial Control Technology, Institute of Industrial Process Control, Zhejiang University, Hangzhou 310027, China*)

[†]E-mail: khan@iipc.zju.edu.cn; jzhao@iipc.zju.edu.cn

Received Nov. 25, 2007; revision accepted Apr. 23, 2008

Abstract: Design of general multivariable process controllers is an attractive and practical alternative to optimizing design by evolutionary algorithms (EAs) since it can be formulated as an optimization problem. A closed-loop particle swarm optimization (CLPSO) algorithm is proposed by mapping PSO elements into the closed-loop system based on control theories. At each time step, a proportional integral (PI) controller is used to calculate an updated inertia weight for each particle in swarms from its last fitness. With this modification, limitations caused by a uniform inertia weight for the whole population are avoided, and the particles have enough diversity. After the effectiveness, efficiency and robustness are tested by benchmark functions, CLPSO is applied to design a multivariable proportional-integral-derivative (PID) controller for a solvent dehydration tower in a chemical plant and has improved its performances.

Key words: Multivariable process control, Proportional-integral-derivative (PID) control, Model predictive control (MPC), Particle swarm optimization (PSO), Closed-loop system

doi:10.1631/jzus.A0720081 **Document code:** A **CLC number:** TP273

INTRODUCTION

Most industrial processes in reality are classified as the multiple-input-multiple-output (MIMO) systems. The control of MIMO systems can be accomplished by either centralized multivariable controllers or a set of independent single-input-single-output (SISO) controllers. Though their superior performance, the centralized multivariable controllers have not been applied as much as expected. The majority of industrial process control applications are still based on decentralized single loop controllers. This is partly caused by the simplified design and well-developed tuning methods of single loop controllers (Campo and Morari, 1994). As competition increases, the manufacturers are forced to improve the performances of controllers for higher efficiency and productivity. High performance multivariable controller design for MIMO systems becomes an urgent topic in process control (Wang *et al*., 2003).

Great efforts have been made and can be found in the literature. Proportional-integral-derivative (PID) controllers are the most popular ones in industry. Wang *et al*.(1997) and Wang (2007) extended much work on PID controllers in SISO systems to the MIMO cases and developed useful rules of tuning multivariable PID controllers. Nordfeldt and Hagglund (2006) introduced a decoupler into the closed-loop system and facilitated the design of PID controllers.

Model predictive control (MPC), a significant multivariable technique developed rapidly in the past two decades, has an increasing acceptance in engineering field recently. However, a major problem that engineers have to face when applying MPC is how to select a suitable set of parameters to maximize the performance, which has been studied extensively (Shridhar and Cooper, 1997; Trierweiler and Farina, 2003; Wojsznis *et al*., 2003).

[‡] Corresponding author

Generally, design of multivariable process controllers can be described as follows:

$$\min f(\boldsymbol{x}) \tag{1}$$
$$\text{s.t.} \quad g_i(\boldsymbol{x}) \le 0, \quad i=1,2,\cdots,n,$$
$$h_j(\boldsymbol{x}) = 0, \quad j=1,2,\cdots,p,$$

where $\boldsymbol{x}=[x_1, x_2, ..., x_d]^{\mathrm{T}}$ denotes the adjustable parameters in controllers, n is the number of inequality constraints, and p is the number of equality constraints. The objective function $f(\boldsymbol{x})$ is defined based on users' desired specifications. Typical output specifications in the time domain are peak overshooting, rise time, settling time, and steady-state error. The integral absolute error (IAE) performance index described in Eq.(2) is usually considered as the objective function,

$$IAE = \int_0^{\infty} \sum_{i=1}^{n} |e_i(t)| \mathrm{d}t. \tag{2}$$

Recent improvements in computing power have expanded evolutionary algorithms (EAs) to a wide variety of application areas. Attempts have been made to employ EAs to optimize the design. Chang (2007) utilized the genetic algorithm (GA) to determine parameters of PID controllers and gained comparatively satisfactory performances.

Particle swarm optimization (PSO) algorithm, developed by Eberhart and Kennedy (1995a; 1995b), has drawn much attention recently due to its simplicity in implementation and reliability in search ability. However, Angeline (1998) pointed out that the basic PSO lacks the ability of balancing the global investigation of the search space and the fine search around a local optimum, which may lead to local optima in some complex cases.

Various attempts have been made to improve the performance of the basic PSO. Niu *et al*.(2007) introduced a bio-inspired mechanism to make the basic PSO more intelligent by separating the population in PSO into two parts: a master swarm and several slave swarms. A hybrid PSO model has been reported by Lovbjerg *et al*.(2001), in which the velocity and position update relations employ the concepts of breeding and subpopulation.

Inertia weight in the velocity update equations has a crucial effect on exploration and exploitation in the process of search (Eberhart and Shi, 2001b). Changing inertia weight linearly (Shi and Eberhart, 1998) or randomly (Eberhart and Shi, 2001a) are straightforward strategies which work well in simple cases. Chatterjee and Siarry (2006) modified the linear changing strategy into a non-linear one to improve the performance of PSO. However, a limit exists, i.e., the inertia weight is only determined by the history of itself and iteration regardless of the current information of each individual (called 'particle') in swarms.

In this work, a new technique for tuning inertia weight is proposed by introducing the concept of 'closed-loop control system', a basic knowledge in control theory. In the proposed closed-loop PSO (CLPSO), each particle and its movement are mapped into the closed-loop system. The dynamic unit denotes a single particle; the output of the dynamic unit is formulated as a function of the particle's fitness in every iteration; the input is the inertia weight; and the controller is designed by a proportional-integral (PI) controller because of its simplicity in structure, robustness in performance and little burden on computation. At each time step, particles adjust their inertia weight based on their own situations, rather than sharing a unified one. In such a way a suitable diversity in the population is guaranteed and the risk of converging to local sub-optima is reduced. Simulation results demonstrate superior performance of the proposed CLPSO in solving complex optimization problems compared with other PSO variants.

The rest of the paper is organized as follows. In Section 2 some basics and recent work of the PSO algorithm are provided. Section 3 explains the details of the proposed CLPSO. The superiority of CLPSO over other variants is demonstrated by two well-known benchmark test functions in Section 4. Section 5 provides the applications of CLPSO in multivariable controller design. The paper ends up with conclusions and the future work in Section 6.

PSO ALGORITHM

Basics of PSO

PSO is an evolutionary algorithm which maintains a swarm of candidate solutions, referred to as 'particles'. The best solution (fitness) each particle has achieved so far is called '*pbest*', and '*gbest*' is the overall best solution obtained so far by any particle in the population. PSO relies on the exchange of information between particles. At each time step, each particle adjusts its trajectory towards its *pbest* and the *gbest* locations by modifying its velocity.

Let x_i denote the current position of the *i*th particle, v_i the rate of the velocity for the *i*th particle, and $V_i^{\max}$ the upper bound on the absolute value of the velocity of the particle. In PSO, particles are manipulated according to the following equations:

$$v_i = wv_i(t) + c_1 r_1 (x_i^{pbest} - x_i) + c_2 r_2 (x^{gbest} - x_i), \quad (3)$$

$$x_i(t+1) = x_i(t) + v_i(t), \quad (4)$$

where c_1 and c_2 are positive constants, representing the cognitive and social parameter, respectively; r_1 and r_2 are random numbers uniformly distributed in the range [0, 1]; w is the inertia weight to balance the global and local search ability.

Previous work

There has been a lot of research in determining the inertia weight. The linear decreasing weight (LDW) was presented by Shi and Eberhart (1998) and described as

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter, \quad (5)$$

where $iter_{\max}$ is the maximum number of iterations, *iter* is the current number of iterations, and $w_{\max}$=0.9, $w_{\min}$=0.4. This version of PSO is referred to as 'standard PSO' (SPSO) in this paper.

Eberhart and Shi (2001a) proposed the PSO with a random inertia weight factor (RPSO), where the inertia weight w changes according to

$$w = 0.5 - \text{rand}()/2, \quad (6)$$

where rand() is a uniformly distributed random number within the range [0, 1].

Recently there has been a trend of employing non-linear techniques in determining the inertia weight. Chatterjee and Siarry (2006) proposed a non-linear variant of inertia weight as follows:

$$\begin{aligned} w_{iter} &= f(iter) \\ &= \frac{(iter_{\max} - iter)^n}{(iter_{\max})^n}(w_{\text{initial}} - w_{\text{final}}) + w_{\text{final}}, \end{aligned} \quad (7)$$

where w_{initial} is the initial inertia weight at the start of a given run, and w_{final} is the final inertia weight at the end of a given run when $iter$=$iter_{\max}$. $iter_{\max}$ is the maximum number of iterations in a given run, and *iter* is the number of iterations at the present time step. w_{iter} is the inertia weight at the present time step and n is the non-linear modulation index. This method is referred to as 'non-linear inertia weight PSO' (NLPSO).

CLOSED-LOOP PSO

Mechanisms of CLPSO

In control theory, the outputs of a dynamic unit are referred to as 'controlled variables' (CVs), and the inputs of a dynamic unit, called 'manipulated variables' (MVs), are used to drive CVs to desired values, or reference inputs. Fig.1 shows the functional block diagram of an open-loop control system. In Fig.1, according to the reference input R, the MV is input into the dynamic unit, and a change occurs in the CV; however, the CV is not necessary to approach the reference input R by reason of no information of CV sent back to the MV. To improve the control performance, the information of CVs is introduced back into the system, which forms a closed loop, as shown in Fig.2. At each sampling time, the difference between R and CV is compared, and the controller calculates a suitable MV based on the current error, along with the previous information.

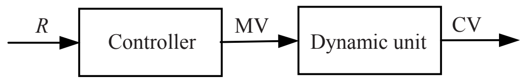


Fig.1 Block diagram of an open-loop control system

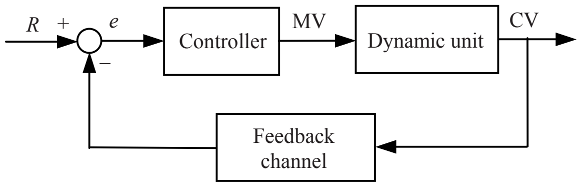


Fig.2 Block diagram of a closed-loop control system

During the run, each particle in PSO is acting as a dynamic unit. The dynamics includes the current position and fitness, the velocity in the last step and the best fitness ever achieved. In some variants of PSO, such as SPSO, RPSO and NLPSO, a unified and predefined inertia weight for population is used. In these algorithms, at the beginning of a search a relatively large value is assigned to the inertia weight to encourage quick movements and global explorations which might, however, drive particles with a better fitness to jump too fast to find the area worthy of a fine search; while at the end of the search, a smaller inertia weight aiming at facilitating local exploitation may obstruct particles with a worse fitness to catch up with others or even make new discoveries. In brief, a uniform inertia weight cannot fit all individual particles with different dynamics. To overcome such a drawback, in the proposed strategy the mechanisms of feedback and closed loop are introduced. At each time step, the output of particles, i.e., the fitness in the last step, is sent back to the loop as a feedback signal. A controller, based on the signal, calculates a proper inertia weight which is the counterpart of MV in the closed-loop control system. Subsequently applying the new inertia weight, particles make a movement, reach a new position and get a new fitness. With respect to the controller, PID is employed in this work due to its simple structure and strong robustness, and extra burden on computation can be neglected.

Formulations in CLPSO

The structure of CLPSO is illustrated in Fig.3. Assume the population of PSO consists of n particles. P_i denotes the ith particle (i=1, 2, ..., n), and the output y_i is defined as $y_i=\varphi_i$, where φ_i is the fitness of the ith particle in the previous step. φ_i is sent back through a feedback channel. The feedback channel H_i and the channel output f_i are respectively defined as

$$H_i=\sigma|1/\psi|, \tag{8}$$

$$f_i=H_i\cdot|y_i|=\sigma|\varphi_i/\psi|, \tag{9}$$

where ψ is the mean of all particles' fitness, σ is an adjustable factor and is fixed at 1 for most cases, and f_i is a measurement of current achievement. In minimization problems f_i with a value greater than 1 implies that the ith particle is in a predicament, while with a value smaller than 1, f_i indicates a good situation for the ith particle. All the problems in this paper are assumed to be minimization problems. The results and conclusions can be easily extended to the maximization problems.

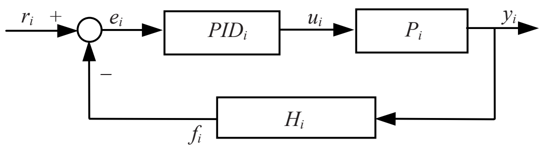


Fig.3 Block diagram of the closed-loop PSO

The same as in the classical control theory, the error signal e_i is defined as

$$e_i=r_i-f_i=1-\sigma|\varphi_i/\psi|. \tag{10}$$

As mentioned above, the inertia weight for the ith particle is selected as the MV, or $u_i=w_i$.

Basics of PID controllers

The ability of PID controllers to compensate most practical industrial processes has led to their wide acceptance in industrial applications. The ideal continuous time domain PID controller for a SISO process is expressed in the Laplace domain as follows:

$$G_c(s)=K_p\left(1+\frac{1}{T_i s}+T_d s\right), \tag{11}$$

where K_p is the proportional gain, T_i the integral time constant or reset time, and T_d the derivative time constant.

Parameterization of controllers in CLPSO

Let $\boldsymbol{\theta}_i=[K_{p,i}\ \ T_{i,i}\ \ T_{d,i}]^T$ represent the sub-PID controller gain vector for the ith particle. Determination of these three parameters is described in this subsection.

In PID controllers, the proportional term is the basic term, causing a corrective control actuation proportional to the error. Chatterjee and Siarry (2006) showed that a higher value of the inertia weight implied larger incremental changes in velocity per time step, which means exploration of new search areas in pursuit of a better solution; whereas a smaller inertia weight means less variation in velocity which slows down the updating rate for fine tuning a local search. It is inferred that a large inertia weight avails particles with large fitness trapping in predicament, while a small inertia weight assists particles out of predicament, in a fine local search. Based on these analyses, a direct acting controller ($K_{p,i}<0$), in which output increases as input increases, is required. In Eq.(9), ψ has a further function as a scaling factor to φ_i; as a result, generally $K_{p,i}$ can be set to −1.

It should be noted that proportional action reduces but does not eliminate the error, and thus an offset between the CV and reference input will normally exist. The integral term gives a correction proportional to the integral of the error. This has the positive feature of ultimately ensuring that sufficient control efforts are made to reduce the offset to zero. Integral time constant is also called ‘reset time’ since it specifies how long it takes for the integral action to produce the same control action as the proportional action does. In other words, the MV immediately steps as an error occurs due to the proportional action. The magnitude of the step up is K_pe. The integral action then causes the MV again to increase by the same magnitude over the period time 0 to time T_i. In a search of PSO with the maximum number of generations fixed at 1000, an intuitional expectation for particles with a fitness larger than ψ is that they should catch up with the mean level of the whole population within about 50 time steps. Therefore one twentieth of the maximum generation is a reasonable choice for $T_{i,i}$.

The derivative action gives a predictive capability yielding a control action proportional to the rate of change of the error. This tends to produce a better response than PI controllers in complex cases but often leads to extra work in parameter determination and system stabilization. Thus a PI controller is applied here, correspondingly in $\boldsymbol{\theta}_i$, $T_{d,i}$=0.

In summary, the PID gain vector $\boldsymbol{\theta}_i$ for the ith particle is specified as

$$\boldsymbol{\theta}_i=[K_{p,i}\ T_{i,i}\ T_{d,i}]^T=[-1\ iter_{max}/20\ 0]^T, \quad (12)$$

where $iter_{max}$ is the maximum number of generations in a run.

Satisfactory results can be obtained with parameters specified in Eq.(12) normally. However, a user in a given problem may attempt to arrive at an even improved performance by further fine tuning the parameters.

Framework of CLPSO

Fig.4 illustrates the framework of CLPSO. The range of inertia weight for each particle is specified as [0.4, 0.9] as in SPSO. All inertia weights are initialized at 0.9 for a quick movement is encouraged at the beginning.

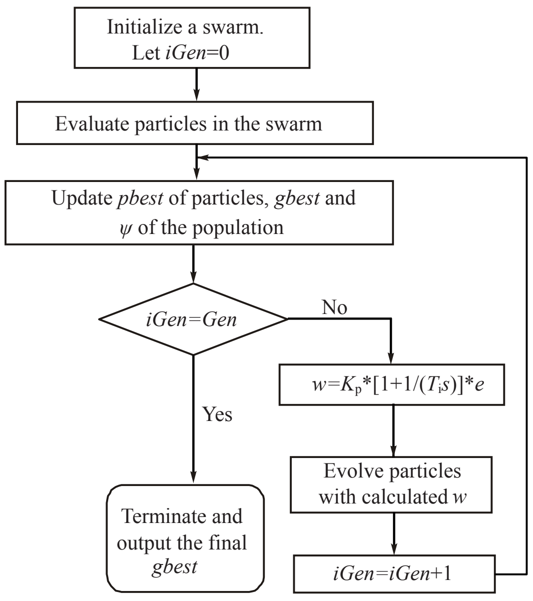

Initialize a swarm. Let iGen=0
Evaluate particles in the swarm
Update pbest of particles, gbest and ψ of the population
iGen=Gen
No
Yes
w=Kp*[1+1/(Tis)]*e
Evolve particles with calculated w
Terminate and output the final gbest
iGen=iGen+1

Fig.4 Flow chart of the closed-loop PSO

Anti-windup scheme and stability of CLPSO

It is well known that the behavior of linear time-invariant (LTI) systems is subject to actuator saturation. Therefore an anti-windup scheme is required to modify the control action when the saturation occurs. In CLPSO, a simple and popular effective anti-windup scheme is applied, i.e., the integral action will stop when the inertia weights reach the given bounds. This scheme prevents a wrong integral action from causing the inertia weight to stay at the boundary all the time.

In control theory, stability has to be taken into account when the closed loop is formed. An improperly designed controller will give wrong inputs, driving the output of the process far away from the set point. With respect to the stability in CLPSO, the inertia weight, the ‘input’ of particles, is only a coefficient in the velocity update Eq.(3); in addition, the region of inertia weight and the search space for particles are predefined. Therefore the output of particles (i.e., φ_i) will converge, which means the stability is guaranteed.

BENCHMARK TEST FUNCTIONS

Two well-known benchmarks that are commonly used in testing the performance of the optimization algorithms are employed in this work to evaluate the proposed CLPSO's performance before applying it in multivariable process controllers tuning. Sphere function is a unimodal minimizing optimization problem and Rastrigrin function is a multimodal one,

Sphere function: $$f_1(x)=\sum_{i=1}^{n} x_i^2. \tag{13}$$

Rastrigrin function:

$$f_2(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]. \tag{14}$$

Their configurations are listed in Table 1.

Table 1 Parameters of the two benchmark functions

Function	Dimension	Minimum value	Search range	Range of initialization
f_1	30	0	$[-100, 100]^n$	$[50, 100]^n$
f_2	30	0	$[-5.12, 5.12]^n$	$[2.56, 5.12]^n$

f_1: Sphere function; f_2: Rastrigrin function

The performance of CLPSO is compared with that of the aforementioned influential variants SPSO, RPSO and NLPSO. For fair comparison, in all algorithms the population size is fixed at 50 and the maximum number of generations at 1000, and a widely accepted setting for the maximum allowable velocity $V_{\max}$ is adopted, i.e., a half value of the upper bound of the search space. The variants are configured according to recommendations presented by (Shi and Eberhart, 1998; Eberhart and Shi, 2001a; Chatterjee and Siarry, 2006). The acceleration constants c_1 and c_2 for SPSO, NLPSO and CLPSO are assigned at 2.0. In RPSO, let $c_1=c_2=1.494$. As for the strategy of determining the inertia weight, Eqs.(5)~(7) are used for SPSO, RPSO and NLPSO, respectively. In CLPSO, the gain vector $\boldsymbol{\theta}_i=[-1\ 50\ 0]^{\mathrm{T}}$ is adopted. In order to eliminate stochastic discrepancy, each algorithm is implemented 50 times independently.

The trendlines of the averaged *gbest* over 50 runs produced by SPSO, RPSO, NLPSO and the proposed CLPSO on the Sphere function and the Rastrigrin function are shown in Figs.5a and 5b, respectively. Since Sphere function is unimodal and relatively simple, all algorithms succeed in achieving the global optimum. However, the differences in terms of convergence rate can be observed.

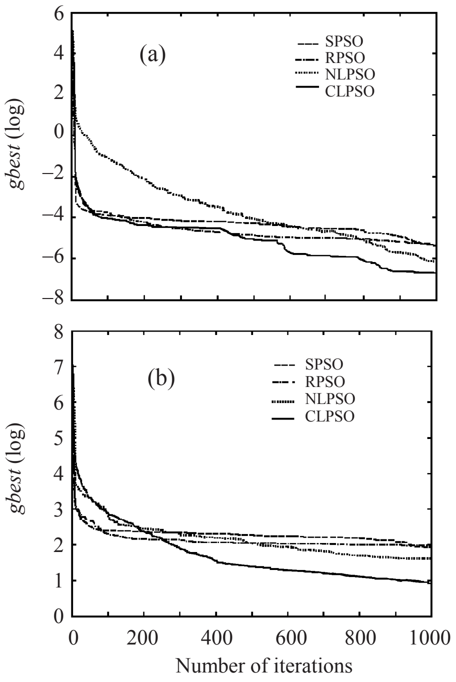


Fig.5 Trendlines of the mean *gbest* on Sphere function (a) and on Rastrigrin function (b)

The superiority of CLPSO becomes apparent in solving the Rastrigrin function. After about 200 generations, no significant improvements are made by SPSO, RPSO and NLPSO, while CLPSO can manage to find way to better solutions.

These results show that the schemes of linear decreasing weight (adopted in SPSO) and random inertia weight (adopted in RPSO) often fail in helping PSO to find the global optimum. Improvement is achieved by employing nonlinear inertia weight, but sluggish converging cannot be neglected. In CLPSO the maximum diversity is maintained, which is a key point in searching for the global optimum. From

Table 2, it can be seen that the best solution found by CLPSO is also the best one in all algorithms. The comparisons of average performance and standard deviation (Std. Dev.) indicate that the proposed approach is powerful and reliable.

Figs.6a and 6b show two representative tracks of particles' inertia weights in a run. Both tracks started at 0.9. In Fig.6a, the inertia weight decreased as the particle ran at the top of the population, but when the particle was surpassed, the inertia weight increased again to try pushing it out of the predicament as quickly as possible. At the end of the search, the particle kept ahead and was conscious that the global optimum was nearby. So the particle occupied itself in a fine search with the help of the smallest inertia weight. The particle in Fig.6b had a different ending. High values of inertia weight indicate that the particle dropped behind most of the time, and thus exploration was its main job. Not like the ones in SPSO, RPSO and NLPSO following a predefined trajectory, the inertia weight in CLPSO has a different trajectory for each particle.

Through these benchmark tests, CLPSO can be identified as a highly consistent strategy in finding the global optimum and removing additional anxieties of employing this approach in tuning parameters of multivariable process controllers.

SIMULATION RESULTS AND AN INDUSTRIAL EXAMPLE

In this section, numerical simulations are carried out on the tuning of multivariable PID controllers and MPC controllers respectively to investigate the performances of the proposed CLPSO. In addition, a multivariable PID controller determined by CLPSO is tested in a real process.

For each testing problem, the parameter settings of CLPSO in Section 4 are still adopted.

Introduction of multivariable PID controllers

An $n\times n$ MIMO system can be described as

$$\boldsymbol{G}(s)=\begin{bmatrix} g_{11}(s) & \cdots & g_{1n}(s) \\ \vdots & & \vdots \\ g_{n1}(s) & \cdots & g_{nn}(s) \end{bmatrix}. \tag{15}$$

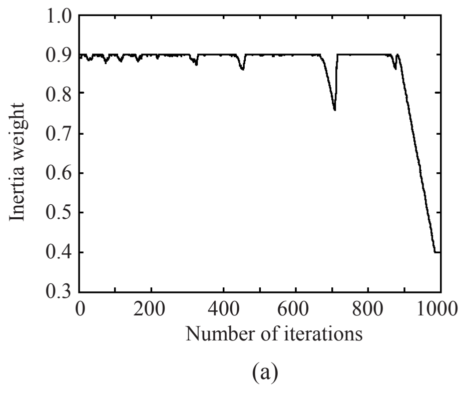
And an $n\times n$ multivariable PID controller is

$$\boldsymbol{G}_{\mathrm{c}}(s)=\begin{bmatrix} G_{11}(s) & \cdots & G_{1n}(s) \\ \vdots & & \vdots \\ G_{n1}(s) & \cdots & G_{nn}(s) \end{bmatrix}, \tag{16}$$

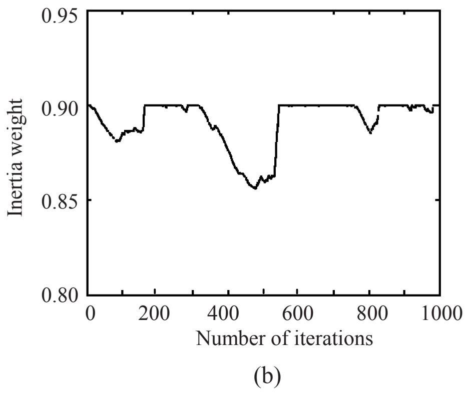
where $G_{ij}(s)$ for $i, j\in\{1, 2, ..., n\}$ are given by

Table 2 Results for all algorithms on the two benchmark functions

Algorithm	Sphere function				Rastrigrin function			
	Best	Worst	Mean	Std. Dev.	Best	Worst	Mean	Std. Dev.
SPSO	9.5013e-07	2.3119e-04	4.1932e-06	2.2770e-04	4.4471e-01	93.4057	78.5948	46.7408
RPSO	6.0681e-08	4.8607e-04	4.2689e-06	4.8208e-04	9.3559e-01	97.9169	74.0561	48.6733
NLPSO	8.9130e-10	8.7622e-05	7.1128e-07	8.6948e-05	4.1034e-03	49.3693	39.9253	24.6481
CLPSO	4.5654e-13	2.2149e-05	1.9433e-07	2.1965e-05	3.5296e-06	13.3299	8.6375	6.9560



(a)



(b)

Fig.6 Track of the inertia weight of a 'good' particle (a) and a 'bad' particle (b)

$$G_{ij}(s)=K_{p,ij}\left(1+\frac{1}{T_{i,ij}s}+T_{d,ij}s\right), \tag{17}$$

where $K_{p,ij}$ is the proportional gain, $T_{i,ij}$ the integral time constant, and $T_{d,ij}$ the derivative time constant of the ijth sub-PID controller.

Simulation results for designing multivariable PID controllers

Consider the following binary distillation column plant described by Wood and Berry (1973):

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{12.8\mathrm{e}^{-s}}{1+16.7s} & \dfrac{-18.9\mathrm{e}^{-3s}}{1+21s}\\ \dfrac{6.6\mathrm{e}^{-s}}{1+10.9s} & \dfrac{-19.4\mathrm{e}^{-3s}}{1+14.4s}\end{bmatrix}. \tag{18}$$

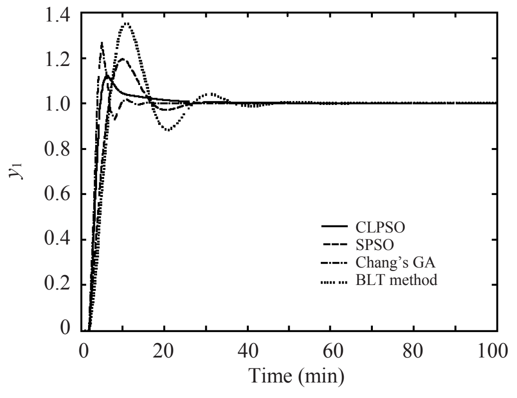
This is a two-input-two-output (TITO) system with strong interaction between the input and output variables. The control objective is to make both outputs have good performances in set point tracking.

This process was also studied by Wang *et al.*(1997) and a multivariable PID controller was provided according to their presented BLT method

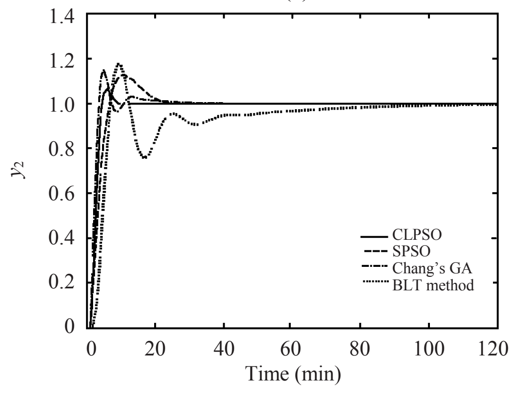
$$\boldsymbol{G}_{\mathrm{c}}(s)=\begin{bmatrix}0.3750\left(1+\dfrac{1}{8.2965s}\right) & 0\\ 0 & -0.0750\left(1+\dfrac{1}{23.4375s}\right)\end{bmatrix}. \tag{19}$$

Chang (2007) took the *IAE* in Eq.(2) as the objective function and solved the PID controller design problem by a modified GA. SPSO was also tested in this example and the configurations for SPSO in Section 4 were still adopted here. The evolution was repeated for 50 runs independently. For fair comparison, the performance criterion *IAE* was also used as the objective function in SPSO and CLPSO.

Table 3 lists the resulting parameters of the PID controller and the *IAE* values derived from the four different methods. The corresponding step responses of both outputs y_1 and y_2 are shown in Figs.7a and 7b, respectively. It is observed that the step responses of the BLT method are oscillatory and have long settling time, and that the interactions between inputs and outputs are not well handled. Chang (2007)'s GA found a set of controller gains which can produce rapid step responses and obtain a smaller *IAE* value; however, the aggressive control actions lead to large overshoots in both outputs, which is not desirable in practice. SPSO reduces the overshoots, but the response of y_2 appears sluggish. The smallest *IAE* value is obtained by CLPSO. The responses of both outputs have short settling time, and the resulting overshoots are acceptable.



(a)



(b)

Fig.7 Step responses of outputs y_1 (a) and y_2 (b)

Table 3 PID parameters and *IAE* values of different methods

Method	Best solution					Mean *IAE*
	$K_{p,11}$	$T_{i,11}$	$K_{p,22}$	$T_{i,22}$	*IAE*	
BLT method	0.3750	8.2965	−0.0750	23.4375	23.5619	
Chang's GA	0.9971	321.6452	−0.0141	1.9859	10.5759	11.8534
SPSO	0.6321	51.1667	−0.1427	37.1642	16.9374	17.9130
CLPSO	0.8980	158.3968	−0.3183	5.8383	8.9445	9.6917

An industrial example for designing multivariable PID controllers

A multivariable PID controller designed by CLPSO has been applied in a solvent dehydration tower in the chemical plant PTA equipment of Sinopec Yangzi Petrochemical Company in China. The main role of the solvent dehydration tower is to purify acetic acid solvent and to remove the H_2O produced in oxidation reaction parts. The control objective is to restrict the tower bottom H_2O content within a specified range 7%~8.5% in normal operating conditions, and to lower the acid content in the tower top to reduce the cost. Therefore, the outputs of the process are the acid content in top flow (y_1) and the H_2O content in bottom flow (y_2). And the inputs are chosen as reflux flow rate (u_1) and the temperature set point in tower bottom (u_2). There is a cascade control loop to control the bottom temperature and it works well. So it is not discussed here. The schematic of the process is depicted in Fig.8.

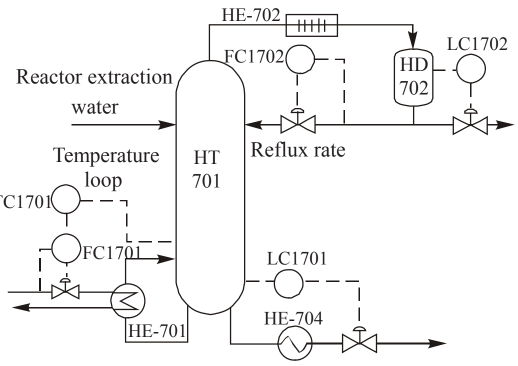


Fig.8 Schematic of the solvent dehydration tower

The identified transfer function matrix of the process is given as

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{-0.0956\mathrm{e}^{-16s}}{1+26.7s} & \dfrac{0.0525\mathrm{e}^{-21s}}{1+50s}\\ \dfrac{0.15\mathrm{e}^{-25s}}{1+50s} & \dfrac{-1.54\mathrm{e}^{-13.3s}}{1+10s}\end{bmatrix}. \tag{20}$$

The distributed control system (DCS) in the plant is Honeywell TDC 3000, which provides several forms of PID controllers. The one used here is referred to as an interactive form and can be described as

$$G_c(s)=K_p\frac{1+T_is}{T_is}\frac{1+T_ds}{1+aT_ds}, \tag{21}$$

where a is set to be 0.1.

With the same setting as in the previous example and the objective function in Eq.(2), CLPSO found a PID controller. The controller gains, together with the original ones, are listed in Table 4.

Table 4 Original and proposed PID controller gains

Controller	$K_{p,11}$	$T_{i,11}$	$T_{d,11}$	$K_{p,22}$	$T_{i,22}$	$T_{d,22}$
Original	−1	10	1	−0.8	20	2
Proposed	−0.749	4.082	1.239	−0.213	5.172	1.377

Fig.9, taken from the universal station of the DCS, illustrates the on-line control performances. It is observed that with large integral gains, the original controller acts weakly on disturbance rejection, resulting in large fluctuation ranges of both outputs. In contrast, disturbances are rejected rapidly by the proposed controller. With the first objective improved, the tower has potential for lowering the acid content, which means a reduction of the cost.

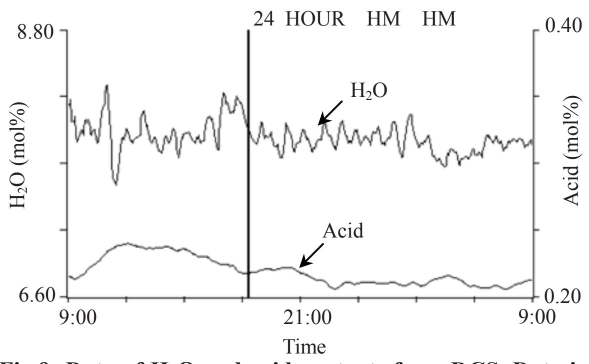


Fig.9 Data of H_2O and acid contents from DCS. Data in the left hand side of the vertical line were produced by the original controller and in the right hand side by the proposed controller

Introduction of MPC parameter tuning strategy

Among the different MPC formulations available today, DMC (dynamic matrix control) law presented by Cutler (1983) is the most popular and represents the chemical process industry's standard for MPC. For this reason, DMC is considered in this work, and the result can be easily extended to other forms. In DMC, there is a least-square optimization problem:

$$\min J(k)=\sum_{i=0}^{P}\|\boldsymbol{y}((k+i)\mid k)-\boldsymbol{y}_{\mathrm{r}}\|_{\boldsymbol{Q}}^{2}+\sum_{i=0}^{M-1}\|\Delta u(k+i)\|_{\boldsymbol{R}}^{2}, \tag{22}$$

where P is the length of prediction horizon, M the control horizon, $\boldsymbol{Q}$ the matrix of controlled variable weights, and $\boldsymbol{R}$ the matrix of move suppression coefficients. $\boldsymbol{R}$ and $\boldsymbol{Q}$ are defined as

$$\boldsymbol{R}=\begin{bmatrix} r_1\boldsymbol{I}_{M\times M} & \cdots & \boldsymbol{0} \\ \vdots & & \vdots \\ \boldsymbol{0} & \cdots & r_m\boldsymbol{I}_{M\times M}\end{bmatrix}_{Mm\times Mm}, \tag{23}$$

$$\boldsymbol{Q}=\begin{bmatrix} q_1\boldsymbol{I}_{P\times P} & \cdots & \boldsymbol{0} \\ \vdots & & \vdots \\ \boldsymbol{0} & \cdots & q_l\boldsymbol{I}_{P\times P}\end{bmatrix}_{Pl\times Pl}. \tag{24}$$

For the unconstrained MIMO DMC law, the future input trajectory is determined by

$$\Delta\boldsymbol{u}(k)=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{A}+\boldsymbol{R})^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Q}[\boldsymbol{y}_{\mathrm{r}}(k)-\tilde{\boldsymbol{y}}_0(k)], \tag{25}$$

where $\boldsymbol{y}_{\mathrm{r}}(k)$ represents the reference trajectory, $\boldsymbol{y}_0(k)$ the free response vector of the plant over P, and $\boldsymbol{A}$ the multivariable dynamic matrix.

The adjustable parameters in MPC technique include P, M, $\boldsymbol{Q}$ and $\boldsymbol{R}$. In practice, generally P and M are specified based on dynamics of the process. The relationship between $\boldsymbol{Q}$ and $\boldsymbol{R}$ has a significant impact on the control action of MPC controllers. Therefore, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are selected to be tuned in the simulation.

Simulation results for tuning MPC controllers

The non-minimum phase continuous stirred-tank reactor (CSTR) proposed by Engell and Klatt (1993) is an ill-conditioned control system, which is studied as a benchmark problem in the literature. The linear state space model analyzed here is given by

$$\boldsymbol{A}=\begin{bmatrix} -86 & 0 & -4.2 & 0 \\ 50.6 & -69.4 & 1 & 0 \\ 172 & 198 & -36.8 & 30.8 \\ 0 & 0 & 86.7 & -86.7\end{bmatrix},$$

$$\boldsymbol{B}=\begin{bmatrix} 3.87 & 0 \\ -0.9 & 0 \\ -4.13 & 0 \\ 0 & 1\end{bmatrix},\ \boldsymbol{C}=\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0\end{bmatrix},\ \boldsymbol{D}=\begin{bmatrix} 0 & 0 \\ 0 & 0\end{bmatrix}.$$

The time scale of the system dynamics is in hours. Based on the dynamics, for the MPC controller, the sampling time is fixed at 8.8 s, and P=16, M=4. The simulation is carried out by changing the set point of y_1 while the set point of y_2 kept no change.

Still take the *IAE* in Eq.(2) as the objective function. SPSO is employed for comparison. Both SPSO and CLPSO are repeated for 30 runs independently. The well-known RPN-MPC tuning methodology proposed by Trierweiler and Farina (2003) is also considered in this example.

Table 5 lists the results for different methods, and the output responses are shown in Fig.10. SPSO leads to a controller with sluggish control actions. The controller tuned by RPN drives y_1 to the new set point faster than SPSO, but at the expense of y_2. The controller by CLPSO behaves better in decoupling. The response of y_2 is improved, and simultaneously a similar response of y_1 to RPN's is obtained.

Table 5 Comparison of results by different methods

Method	Best solution					Mean *IAE*
	q_1	q_2	r_1	r_2	*IAE*	
RPN	1.520	0.063	1.3e-2	3.0e-6	14.408	
SPSO	29.085	28.782	3.3e-1	7.9e-2	20.328	25.174
CLPSO	69.183	48.093	3.4e-3	1.3e-4	9.270	11.694

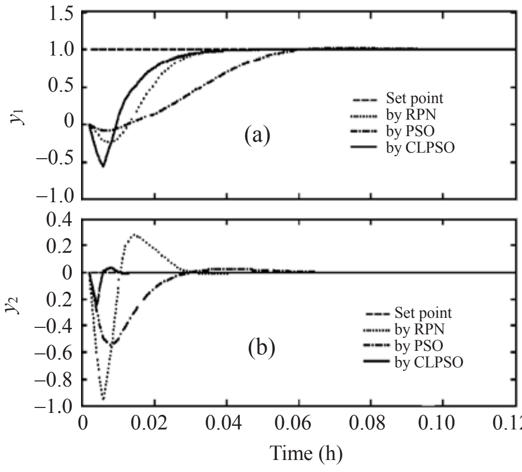


Fig.10 Responses of outputs y_1 (a) and y_2 (b)

CONCLUSION

Integrated with classical control techniques which have been well developed and proven in industry for years, a CLPSO has been proposed in this paper to improve the performance of the ordinary PSO in optimal design of multivariable process controllers. In ordinary PSO, there is only one uniform inertia weight for the whole population, regardless of the progress of each individual particle. Applying the principle of feedback control, CLPSO adjusts inertia weight for particles according to their own information, which guarantees the ample diversity in population. The proposed method exhibits superior effectiveness, efficiency and robustness through well-known simulations in both academic and engineering fields. Additionally a real industrial process test demonstrates the satisfactory performance of CLPSO in the applications. Although CLPSO takes longer time than the deterministic tuning methods, the tuning procedure considered in the work is implemented off-line, so the performance rather than the time is a key problem.

To achieve better performances and wider applications, future work will focus on the optimization of the PI controller in CLPSO.

References

Angeline, P.J., 1998. Evolutionary Optimization versus Particle Swarm Optimization and Philosophy and Performance Difference. Proc. 7th Annual Conf. on Evolutionary Programming, San Diego, USA, p.601-610. [doi:10.1007/BFb0040753]

Campo, P., Morari, M., 1994. Achievable closed-loop properties of systems under decentralized control: conditions involving the steady-state gain. *IEEE Trans. on Automatic Control*, **39**(5):932-943. [doi:10.1109/9.284869]

Chang, W.D., 2007. A multi-crossover genetic approach to multivariable PID controllers tuning. *Expert Syst. Appl.*, **33**(3):620-626. [doi:10.1016/j.eswa.2006.06.003]

Chatterjee, A., Siarry, P., 2006. Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. *Comput. Oper. Res.*, **33**(3):859-871. [doi:10.1016/j.cor.2004.08.012]

Cutler, C.R., 1983. Dynamic Matrix Control—An Optimal Multivariable Control Algorithm with Constraints. Ph.D Thesis, University of Houston, Houston, TX.

Eberhart, R.C., Kennedy, J., 1995a. Particle Swarm Optimization. Proc. IEEE Int. Conf. on Neural Networks, Perth, Australia, p.1942-1948. [doi:10.1109/ICNN.1995.488968]

Eberhart, R.C., Kennedy, J., 1995b. A New Optimizer Using Particle Swarm Theory. Proc. Int. Symp. on Micro Machine and Human Science, Nagoya, Japan, p.29-43. [doi:10.1109/MHS.1995.494215]

Eberhart, R.C., Shi, Y., 2001a. Tracking and Optimizing Dynamic Systems with Particle Swarms. Proc. IEEE Congress on Evolutionary Computation, Seoul, Korea, p.94-97. [doi:10.1109/CEC.2001.934376]

Eberhart, R.C., Shi, Y., 2001b. Particle Swarm Optimization: Development, Applications and Resources. Proc. IEEE on Evolutionary Computation, Seoul, Korea, p.81-86.

Engell, S., Klatt, K., 1993. Nonlinear Control of a Non-minimum-phase CSTR. Proc. American Control Conf., Los Angeles, p.2041-2045.

Lovbjerg, M., Rasmussen, T.K., Krink, T., 2001. Hybrid Particle Swarm Optimizer with Breeding and Subpopulations. Proc. Third Genetic and Evolutionary Computation Congress, San Diego, USA, p.469-476.

Niu, B., Zhu, Y.L., He, X.X., Zeng, X.P., 2007. MCPSO: a multi-swarm cooperative particle swarm optimizer. *Appl. Math. Comput.*, **185**(2):1050-1062. [doi:10.1016/j.amc.2006.07.026]

Nordfeldt, P., Hagglund, T., 2006. Decoupler and PID controller design of TITO systems. *J. Process Control*, **16**(9):923-936. [doi:10.1016/j.jprocont.2006.06.002]

Shi, Y., Eberhart, R.C., 1998. A Modified Particle Swarm Optimizer. Proc. IEEE Int. Conf. on Evolutionary Computation, Anchorage, Alaska, p.69-73. [doi:10.1109/ICEC.1998.699146]

Shridhar, R., Cooper, D.J., 1997. A novel tuning strategy for multivariable model predictive control. *ISA Trans.*, **36**(4):273-280. [doi:10.1016/S0019-0578(97)00036-0]

Trierweiler, J.O., Farina, L.A., 2003. RPN tuning strategy for model predictive control. *J. Process Control*, **13**(7):591-598. [doi:10.1016/S0959-1524(02)00093-8]

Wang, Q.G., 2007. A quasi-LMI approach to computing stabilizing parameter ranges of multi-loop PID controllers. *J. Process Control*, **17**(1):59-72. [doi:10.1016/j.jprocont.2006.08.006]

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. [doi:10.1016/S0005-1098(96)00177-X]

Wang, Q.G., Zhang, Y., Chiu, M.S., 2003. Non-interacting control design for multivariable industrial processes. *J. Process Control*, **13**(3):253-265. [doi:10.1016/S0959-1524(02)00028-8]

Wojsznis, W., Gudaz, J., Blevins, T., Mehta, A., 2003. Practical approach to tuning MPC. *ISA Trans.*, **42**(1):149-162. [doi:10.1016/S0019-0578(07)60121-9]

Wood, R.K., Berry, M.W., 1973. Terminal composition control of a binary distillation column. *Chem. Eng. Sci.*, **28**(9):1707-1717.