

Run-to-run Optimization for Fed-batch Fermentation Process with Swarm Energy Conservation Particle Swarm Optimization Algorithm^{*}

WANG Jianlin (王建林)^{**}, XUE Yaoyu (薛尧予), YU Tao (于涛) and ZHAO Liqiang (赵利强)

College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China

Abstract An iterative optimization strategy for fed-batch fermentation process is presented by combining a run-to-run optimization with swarm energy conservation particle swarm optimization (SEC-PSO). SEC-PSO, which is designed with the concept of energy conservation, can solve the problem of premature convergence frequently appeared in standard PSO algorithm by partitioning its population into several sub-swarms according to the energy of the swarm and is used in the optimization strategy for parameter identification and operation condition optimization. The run-to-run optimization exploits the repetitive nature of fed-batch processes in order to deal with the optimal problems of fed-batch fermentation process with inaccurate process model and unsteady process state. The kinetic model parameters, used in the operation condition optimization of the next run, are adjusted by calculating time-series data obtained from real fed-batch process in the run-to-run optimization. The simulation results show that the strategy can adjust its kinetic model dynamically and overcome the instability of fed-batch process effectively. Run-to-run strategy with SEC-PSO provides an effective method for optimization of fed-batch fermentation process.

Keywords run-to-run optimization, fed-batch process, particle swarm optimization, swarm energy conservation particle swarm optimization

1 INTRODUCTION

Fed-batch processes are suitable for manufacture of high value added products, such as special polymers, special chemicals, and pharmaceuticals [1], since these processes allow the substrate concentration in the culture broth to be adjusted to a desired level in order to regulate the specific growth rate and product formation [2]. The optimal control strategy of a fermentation process is to maximize the final product yield. Previous studies mostly concentrate on the determination of an optimal operation strategy of fed-batch fermentation processes. Herrera and Zhang [3] presented an optimal control strategy for batch processes using PSO. Kawohl *et al.* [4] implemented a model based optimal control strategy. Ronen *et al.* [5] designed an optimal feeding profile by means of an evolutionary algorithm. Li *et al.* [6] described a strategy of phosphorus feeding to improve the industrial fermentation of penicillin G. These strategies are limited to a single run of fed-batch process, so that the useful information from one fed-batch run is not made use of in the next run. It is logical to exploit the information from the previous run to improve the operation of current fed-batch run.

The particle swarm optimization (PSO) is an evolutionary computation technique developed by Kennedy and Eberhart based on the simulation of a simplified social model [7]. Due to the simple concept, easy implementation and quick convergence, PSO has received much attention and has many applications in process control fields [3, 8]. However, the performance of standard PSO depends on its parameters and it often suffers the problem of being trapped in local opti-

mal solutions [9]. Many studies have been carried out to balance the exploration and exploitation so as to prevent premature convergence [10].

In this study, a run-to-run optimization strategy of fed-batch process with swarm energy conservation PSO (SEC-PSO) is presented. SEC-PSO is used in the run-to-run strategy to identify kinetic model parameters and optimize operating conditions. The run-to-run optimization strategy exploits the repetitive nature of fed-batch processes to deal with the optimal problems of the process with inaccurate process model and unsteady process state.

2 SWARM ENERGY CONSERVATION PARTICLE SWARM OPTIMIZATION

2.1 Standard particle swarm optimization

PSO is a population based optimization tool, where the system is initialized with a population of random particles and the algorithm searches for optima by updating generations. If there are m particles located in the D -dimensional space to form a community, the position of the i th particle is represented as $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{iD}]^T$, $i = 1, 2, \dots, m$, and its velocity is represented as $\mathbf{v}_i = [v_{i1}, v_{i2}, \dots, v_{iD}]^T$. The best position in the searching history of the i th particle is represented as $\mathbf{x}_i^*$ and the best position of the particle among the population is $\mathbf{x}^*$. Put $\mathbf{x}_i$ into the objective function and calculate the fitness value. In the $(k+1)$ th iteration each particle updates its velocity and

Received 2010-02-03, accepted 2010-06-29.

^{*} Supported by the National Natural Science Foundation of China (20676013).

^{**} To whom correspondence should be addressed. E-mail: wangjl@mail.buct.edu.cn

position according to the following equations

$$\mathbf{v}_i^{k+1} = \omega \mathbf{v}_i^k + c_1 r_1 (\mathbf{x}_i^* - \mathbf{x}_i^k) + c_2 r_2 (\mathbf{x}^* - \mathbf{x}_i^k) \quad (1)$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1} \quad (2)$$

where ω is the inertia weight that determines how much a particle holds its current velocity in the next iteration. Suitable selection of inertia weight can provide the particles with a balance between the ability of exploitation and exploration. c_1 and c_2 are positive constants, named learning factors or acceleration coefficients, which control the maximum step size. r_1 and r_2 are independently uniformly distributed random variables with range $[0, 1]$. $\mathbf{v}_i^{k+1}$ is the vector sum of $\mathbf{v}_i^k$, $\mathbf{x}_i^* - \mathbf{x}_i^k$ and $\mathbf{x}^* - \mathbf{x}_i^k$. The implementing steps of PSO are similar to other evolutionary algorithms, which include initialization, fitness evaluation, update of velocity and position, and testing of stop criterion [11].

2.2 Swarm energy conservation particle swarm optimization

In the standard PSO, a single particle gains experience and information not only from itself but also from the population during its motion. The two types of information are come from the best particle. We attempt to improve the velocity equation of the standard PSO and make use of failure experience of the worst particle. In other words, the worst positions of individual and population are put into the algorithm. In this way, the algorithm updates its velocity equation according to the best value and the worst value of individual and the population. Thus the particle may change its status with more information.

Equations (1) and (2) become

$$\mathbf{v}_i^{k+1} = \omega \mathbf{v}_i^k + c_1 r_1 (\mathbf{x}_i^* - \mathbf{x}_i^k) - u_1 r_1 (\mathbf{x}_i^o - \mathbf{x}_i^k) + c_2 r_2 (\mathbf{x}^* - \mathbf{x}_i^k) - u_2 r_2 (\mathbf{x}^o - \mathbf{x}_i^k) \quad (3)$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1} \quad (4)$$

Compared with the standard PSO, the modified strategy has two additional parameters. They are impact factors u_1 and u_2 , reflecting the effect of the worst value of individual and population in the updating of the velocity. The worst position of individual is represented as $\mathbf{x}_i^o$ and the worst position of population is $\mathbf{x}^o$. Other parameters are the same as those in the standard PSO.

With the worst value, the modified algorithm has some advantages, such as rapid convergence and good global convergence. First, when we analyze the algorithm from the perspective of the amount of information, each particle in the new algorithm is moving with the experience of the worst value of itself and the population, which makes it move to the opposite direction of the worst position and speed up the conver-

gence. Second, the swarm searches for not only the best value but also the worst value. It is beneficial for the diversity and global convergence of the algorithm.

In order to make the worst particle play greater role in the algorithm, we use a punitive measure on the worst particle to ensure the algorithm reliability. An initial velocity will be given to the particle in the initialization of the algorithm. During the motion, most of the particles will fall into local or global optimal solution. Their velocity will be zero or very low, which means that the particles will lose their energy. If we consider the particle swarm as a system, its energy will be conserved. The particles that does not fall into optimum will get much more kinetic energy. Then the probability of obtaining the optimal solutions will increase.

Since the velocity of the i th particle is represented as $\mathbf{v}_i$ and the energy of a single particle is represented as $g_i = \mathbf{v}_i^T \mathbf{v}_i$, the energy of the swarm is

$$G = \sum_{i=1}^m g_i = \sum_{i=1}^m \mathbf{v}_i^T \mathbf{v}_i \quad (5)$$

Suppose that the swarm energy is conserved during the motion of particles. After some of the particles fall into optimum, most of the swarm energy concentrates on the bad particles. Then the swarm is divided into two sub-swarms: one contains good particles and the other contains bad particles. We consider the velocity increase of bad particles as a punishment and call them penalty particles. This idea of swarm separation is different from previous literatures. First, the two sub-swarms do not move independently. They gain experience from each other when they update their velocity and positions. In other words, they are not separated essentially. Second, the swarm separation is a dynamic status. Some particles move from good sub-swarm to bad sub-swarm after each update, while some of bad particles become good ones.

We set a swarming coefficient in the algorithm and represent it as $f \in [0, 1]$. The number of good particles is defined as $f \times m$, and that of bad particles is $(1-f) \times m$. The particles are arranged according to the fitness value by ascending order. $f \times m$ particles in the front of the sequence form the good sub-swarm and the rest are in bad sub-swarm.

In SEC-PSO, good particles update their velocity and position according to Eqs. (3) and (4). After the update, the energy of good sub-swarm is

$$G_1 = \sum_{i=1}^{f \times m} \mathbf{v}_i^T \mathbf{v}_i \quad (6)$$

Bad particles update their velocity and position according to following equations

$$\mathbf{v}_i^{k+1} = \omega \mathbf{v}_i^k + c_1 r_1 (\mathbf{x}_i^* - \mathbf{x}_i^k) + c_2 r_2 (\mathbf{x}^* - \mathbf{x}_i^k) \quad (7)$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \lambda \mathbf{v}_i^{k+1} \quad (8)$$

$$\lambda \mathbf{v}_{i+1}^T \mathbf{v}_{i+1} = \frac{G - G_1}{m(1-f)} \quad (9)$$

where λ is the penalty coefficient. It indicates the effect acting on the bad particles uniformly, which is generated by the difference between the swarm energy G and the good sub-swarm energy G_1 .

In SEC-PSO, the swarm separation is a dynamic state. The velocity of the penalty particles will be zero, which become good particles when they find optimal position. Because of the constant energy of the swarm, some particles will keep moving in the search space. The aim of this improvement is to prevent the algorithm from premature convergence.

SEC-PSO can be implemented according to the following steps:

Step 1 Initialize the impact factors (c_1 , c_2 , u_1 and u_2), the inertia weight (ω) and the upper limit of evolutionary generations as MaxDT. Initialize population with m particles randomly in the solution space. Initialize the velocity of each particle as $\mathbf{v}$ and compute the swarm energy G .

Step 2 Evaluate population $\mathbf{x}^k$ and compute the fitness of each particle in each dimension.

Step 3 Divide the swarm into two sub-swarms according to the order of the fitness value.

Step 4 Operate to the good population. Update the velocity and the position of the particle according Eqs. (3) and (4). Generate new population $\mathbf{x}^{k+1}$ and compute the swarm energy. Compare the fitness of the particle with its optimal value. If the present value is better than the optimal value of $\mathbf{x}_i^*$, set the optimal value as the present value and set $\mathbf{x}_i^*$ as the best position. Compare the fitness of the particle with its worst value. If the present value is worse than the worst value of $\mathbf{x}_i^o$, set the worst value as the present value and set $\mathbf{x}_i^o$ as the worst position.

Step 5 Compare the fitness value with the best and worst values of the population. If the present value is better than the global optimal value, set $\mathbf{x}^*$ as the present value. If the present value is worse than the global worst value, set $\mathbf{x}^o$ as the present value.

Step 6 Operate to the bad population. Punish the bad particles according to the losing energy of good population and update them according to Eqs. (7)–(9).

Step 7 Find the history optimal value and the global optimal value of the population.

Step 8 Check the termination condition and end the loop if the condition is satisfied; or set $k = k + 1$ and go to Step 2. The termination condition is the number of evolutionary generations reaches the upper limit of evolutionary generations (MaxDT), or the fitness value is less than the specified value ϵ .

3 RUN-TO-RUN OPTIMIZATION STRATEGY

In a fed-batch process, the feeding of substrates makes it possible to maintain substrate concentrations

at a desired level in order to regulate the specific growth rate and product formation. In general, the feeding is controlled by operating recipes. Such recipes are often based on process knowledge and previous experience, but mathematical models may be applied in recipe development and adaptation [2]. Because of the complexity of fermentation processes, inaccuracy of process model and non-linearity of governing dynamic equations, it is difficult to determine the optimal feed rate to produce the maximum production. Therefore, a dynamic and accurate optimization strategy is needed to deal with this difficulty.

3.1 Run-to-run optimization design

In optimization of fed-batch process, there are two major approaches: optimization based on model refinement and optimization with direct input update [12]. Bonvin *et al.* [13] have further classified these optimization methods into four approaches: real-time explicit optimization, run-to-run explicit optimization, real-time implicit optimization, and run-to-run implicit optimization. These strategies are classified according to whether a process model is used for implementation (horizontal division). A second dimension for classification is the type of adaptation, which is either online or run-to-run (vertical division).

As shown in Fig. 1, the run-to-run strategy presented is to overcome the inaccuracy of process model and the instability of fed-batch process. In the strategy, there are two operation objects, which are fed-batch fermentation process (Object 1) and the process kinetic model (Object 2), and two optimizations, which are model parameter identification and operation condition optimization.

In the parameter identification, the kinetic model is optimized with SEC-PSO, and the model fitting curves will approach the real data curves. That is to say, the optimization objective is to minimize the integration of difference-values between real time-series data and fitted time-series data. The optimization objects are the identified parameters, which will be used in the operation condition optimization of current run. The optimization problem can be described as

$$\min J_1 = \int_0^{t_f} \sum_{i=1}^k (R_i^2(t) - M_i^2(t)) dt$$

$$\text{subject to: } R_i(t) \geq 0; M_i(t) \geq 0 \quad (i = 1, 2, \dots, k) \quad (10)$$

where J_1 is the object to be optimized, k is the number of output types, t denotes the run time, t_f denotes the final time, $R(t)$ denotes the run-time outputs of real fed-batch process, and $M(t)$ denotes the run-time outputs of model simulation process.

The operation condition optimization is a model-based offline optimization method by using SEC-PSO, and the model is updated in the parameter identification. The optimization objective is to maximize the integration of model simulation outputs, and the optimization object is the control trajectory, which will be

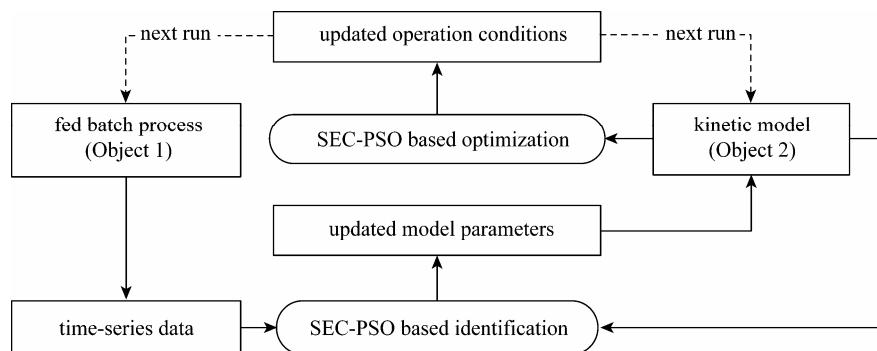


Figure 1 Run-to-run optimization strategy based on SEC-PSO

used in the next run. The optimization problem can be described as

$$\begin{aligned} \max J_2 &= \int_0^{t_f} M^2(t) dt \\ \text{subject to: } M(t) &\geq 0 \end{aligned} \quad (11)$$

where J_2 is the object to be optimized.

3.2 Run-to-run optimization implementation

Run-to-run strategy can be described as follows:

Step 1 Select a specific curve as the initial operation condition. In order to obtain a group of time-series data, the fed-batch process (Object 1) is run with this curve.

Step 2 Identify parameters of the kinetic model (Object 2) with time-series data obtained from Step 1. The operation condition of kinetic model is set the same as the initial operation condition in Step 1.

Step 3 Update kinetic model with new parameters obtained from Step 2.

Step 4 Optimize the operation conditions with the updated kinetic model.

Step 5 Update the operation conditions and go to Step 1.

Because the optimization of conditions is executed in every run, the strategy can adjust itself dynamically and overcome the instability of fed-batch process. Additionally, the model parameters are identified in every run, so that the problem of the inaccuracy of process model overcomes to some extent.

4 SIMULATION AND DISCUSSION

A group of simulation experiments are carried out to verify the effectiveness of the run-to-run optimization strategy. Simulation 1 is model parameter identification with SEC-PSO. Simulation 2 is model-based optimization of operation conditions by using SEC-PSO. Simulation 3 is comprehensive simulation experiment of run-to-run optimization strategy.

Additionally, we choose a process model to describe fed-batch process (Object 1). Meanwhile, we construct a simplified model as kinetic model (Object 2) by considering the key state variables (*i.e.* culture

volume, the concentrations of substrate, biomass, and product) and the key control variables (*i.e.* substrate feed) of the complete fed-batch process model (Object 1). The simplified model maintains most dynamic characteristics of fermentation processes.

4.1 Case studies

Several considerations are involved in the selection of process model (Object 1). First, the model should be nonlinear enough for adequate testing of the methodology. Second, the process should be industrially relevant to present the importance of this research. Third, enough process data are available to guarantee the calculation of the fitness value. Since the nonlinearity of model makes the minimization of the objective function more difficult, we must consider [14]: (1) the size of the parameter space, (2) the existence of local minima, (3) the continuity of the objective function, and (4) the sensitivity of the objective function to each of the model parameters. With these considerations, we choose industrial yeast fermentation process as the study object.

The dynamic model of industrial yeast fermentation process is based on mass balance equations described by volume and concentrations, such as biomass, ethanol, glucose, oxygen and carbon dioxide concentrations. The model is constructed by completing the kinetics with substrate limitation and product inhibition. It comprises the following equations [15–18].

$$\frac{dV}{dt} = F \quad (12)$$

$$\frac{dC_X}{dt} = \mu C_X - \frac{F}{V} C_X \quad (13)$$

$$\frac{dC_e}{dt} = (Q_{e,pr} - Q_{e,ox}) C_X - \frac{F}{V} C_e \quad (14)$$

$$\frac{dC_S}{dt} = \frac{F}{V} (S_0 - C_S) - \left(\frac{\mu}{Y_{X/S}^{OX}} + \frac{Q_{e,pr}}{Y_{e/S}} + Q_m \right) C_X \quad (15)$$

$$\frac{dC_O}{dt} = -Q_O C_X + k_L a_O (C_O^* - C_O) - \frac{F}{V} C_O \quad (16)$$

Table 1 Values of parameters in the fed-batch yeast fermentation model

$K_e/\text{g}\cdot\text{L}^{-1}$	$K_o/\text{g}\cdot\text{L}^{-1}$	$K_s/\text{g}\cdot\text{L}^{-1}$	$K_{\text{S}}/\text{g}\cdot\text{L}^{-1}$	$Y_{\text{X/S}}^{\text{OX}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{X/S}}^{\text{red}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{C/S}}^{\text{OX}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{C/S}}^{\text{red}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{O/S}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{O/e}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{e/O}}/\text{g}\cdot\text{g}^{-1}$	$K_{\text{e/S}}/\text{g}\cdot\text{g}^{-1}$
0.1	9.6×10^{-5}	3.5	0.612	0.585	0.05	0.5744	0.462	0.3857	0.8896	1.1236	0.4859
$Y_{\text{C/e}}/\text{g}\cdot\text{g}^{-1}$	$Y_{\text{X/e}}/\text{g}\cdot\text{g}^{-1}$	$\frac{Q_{\text{S,max}}}{\text{g}\cdot\text{g}^{-1}\cdot\text{h}^{-1}}$	$\frac{Q_{\text{O,max}}}{\text{g}\cdot\text{g}^{-1}\cdot\text{h}^{-1}}$	$\frac{Q_{\text{e,max}}}{\text{g}\cdot\text{g}^{-1}\cdot\text{h}^{-1}}$	$Q_{\text{m}}/\text{g}\cdot\text{g}^{-1}\cdot\text{h}^{-1}$	$\mu_{\text{cr}}/\text{h}^{-1}$	$S_0/\text{g}\cdot\text{h}^{-1}$	$C_0^*/\text{g}\cdot\text{h}^{-1}$	$C_{\text{C}}^*/\text{g}\cdot\text{h}^{-1}$	$k_{\text{LaO}}/\text{h}^{-1}$	$k_{\text{LaC}}/\text{h}^{-1}$
0.645	0.7187	0.2943	0.255	0.238	0.03	0.21	325	0.007712	0.00044	600	470.4

$$\frac{dC_{\text{C}}}{dt} = Q_{\text{C}}C_{\text{X}} + k_{\text{LaC}}(C_{\text{C}}^* - C_{\text{C}}) - \frac{F}{V}C_{\text{C}} \quad (17)$$

Glucose uptake rate

$$Q_{\text{S}} = Q_{\text{S,max}} \frac{C_{\text{S}}}{K_{\text{S}} + C_{\text{S}}} (1 - e^{-t/2}) \quad (18)$$

Specific growth rate limit

$$Q_{\text{S,lim}} = \frac{\mu_{\text{cr}}}{Y_{\text{X/S}}^{\text{OX}}} \quad (19)$$

Oxidation capacity

$$Q_{\text{O,lim}} = Q_{\text{O,max}} \frac{C_{\text{O}}}{K_{\text{O}} + C_{\text{O}}} \frac{K_{\text{i}}}{K_{\text{i}} + C_{\text{e}}} \quad (20)$$

Oxidative glucose metabolism

$$Q_{\text{S,OX}} = \min \left(\begin{array}{c} Q_{\text{S}} \\ Q_{\text{S,lim}} \\ Q_{\text{O,lim}}/Y_{\text{O/S}} \end{array} \right) \quad (21)$$

Reductive glucose metabolism

$$Q_{\text{S,red}} = Q_{\text{S}} - Q_{\text{S,OX}} \quad (22)$$

Ethanol uptake rate

$$Q_{\text{e,up}} = Q_{\text{e,max}} \frac{C_{\text{e}}}{K_{\text{e}} + C_{\text{e}}} \frac{K_{\text{i}}}{K_{\text{i}} + C_{\text{e}}} \quad (23)$$

Oxidative ethanol metabolism

$$Q_{\text{e,OX}} = \min \left(\begin{array}{c} Q_{\text{e,up}} \\ (Q_{\text{O,lim}} - Q_{\text{S,OX}}Y_{\text{O/S}})Y_{\text{e/O}} \end{array} \right) \quad (24)$$

Total specific growth rate

$$\mu = Q_{\text{S,OX}}Y_{\text{X/S}}^{\text{OX}} + Q_{\text{S,red}}Y_{\text{X/S}}^{\text{red}} + Q_{\text{e,OX}}Y_{\text{X/e}} \quad (25)$$

Ethanol production rate

$$Q_{\text{e,pr}} = Q_{\text{S,red}}Y_{\text{e/S}} \quad (26)$$

Oxygen consumption rate

$$Q_{\text{O}} = Q_{\text{S,OX}}Y_{\text{O/S}} + Q_{\text{e,OX}}Y_{\text{O/e}} \quad (27)$$

Carbon dioxide production rate

$$Q_{\text{C}} = Q_{\text{S,OX}}Y_{\text{C/S}}^{\text{OX}} + Q_{\text{S,red}}Y_{\text{C/S}}^{\text{red}} + Q_{\text{e,OX}}Y_{\text{C/e}} \quad (28)$$

The total fermentation time for simulation is 16.5 hours. The length of time interval is 10 minutes. Thus the samples during the total fermentation time are 100 from 0 to 16.5 hours. The following nonlinear exponential feed flow rate F is used for the data generation

for the simulation.

$$F(t) = F_0 e^{0.05t} \quad (29)$$

where F_0 is the initial value of feed flow rate, 500 $\text{L}\cdot\text{h}^{-1}$ for the simulation. The model parameters and initial conditions used for dynamic simulations are listed in Tables 1 and 2.

Table 2 Initial conditions for dynamic simulation

V/L	$C_{\text{X}}/\text{g}\cdot\text{L}^{-1}$	$C_{\text{e}}/\text{g}\cdot\text{L}^{-1}$	$C_{\text{S}}/\text{g}\cdot\text{L}^{-1}$	$C_{\text{O}}/\text{g}\cdot\text{L}^{-1}$	$C_{\text{C}}/\text{g}\cdot\text{L}^{-1}$
50000	0.5	0	7	0.007712	0

The kinetic model (Object 2) is simplified from the process model mentioned above. The simplified model includes 4 variables, volume of fermentation broth (V), biomass concentration (C_{X}), ethanol concentration (C_{e}) and glucose concentration (C_{S}).

Balance equations are

$$\begin{cases} \frac{dV}{dt} = F \\ \frac{dC_{\text{X}}}{dt} = \mu_{\text{X}}C_{\text{X}} - \frac{F}{V}C_{\text{X}} \\ \frac{dC_{\text{e}}}{dt} = \mu_{\text{p}}C_{\text{X}} - \frac{F}{V}C_{\text{e}} \\ \frac{dC_{\text{S}}}{dt} = \frac{F}{V}(S_0 - C_{\text{S}}) - \left(\frac{\mu_{\text{X}}}{Y_{\text{X/S}}} + \frac{\mu_{\text{p}}}{Y_{\text{e/S}}} + Q_{\text{m}} \right) C_{\text{X}} \end{cases} \quad (30)$$

In simulation experiment, the balance equations should be discretized. The discretization method adopted here is Euler method. Step length is set as s . Eq. (30) changes to

$$\begin{cases} V(k) = V(k-1) + s \times F \\ C_{\text{X}}(k) = C_{\text{X}}(k-1) + s \times \left(\mu_{\text{X}}C_{\text{X}}(k-1) - \frac{F}{V}C_{\text{X}}(k-1) \right) \\ C_{\text{e}}(k) = C_{\text{e}}(k-1) + s \times \left(\mu_{\text{p}}C_{\text{X}}(k-1) - \frac{F}{V}C_{\text{e}}(k-1) \right) \\ C_{\text{S}}(k) = C_{\text{S}}(k-1) + s \times \left(\frac{F}{V}(S_0 - C_{\text{S}}(k-1)) - \left(\frac{\mu_{\text{X}}}{Y_{\text{X/S}}} + \frac{\mu_{\text{p}}}{Y_{\text{e/S}}} + Q_{\text{m}} \right) C_{\text{X}}(k-1) \right) \end{cases} \quad (31)$$

where $s = 10$ (minutes).

Rate equations are

$$\begin{cases} \mu_X = \mu_{X\max} \frac{C_S(k-1)}{K_S + C_S(k-1)} \\ \mu_P = \mu_{P\max} \frac{C_S(k-1)}{K_i + C_S(k-1) + C_S^2(k-1)/K_e} \end{cases} \quad (32)$$

4.2 Parameter identification

The objective function of parameter identification is calculated as variance according to the following equation

$$T_{\text{fitness}} = \sum_{i=1}^n \left[(X_i - X'_i)^2 + (S_i - S'_i)^2 + (P_i - P'_i)^2 + (V_{Li} - V'_{Li})^2 \right] \quad (33)$$

where X_i is the actual measurement value and X'_i is the model value of biomass concentration, S_i is the measured value and S'_i is the model value of substrate concentration, P_i is the measured value and P'_i is the model value of production concentration, V_{Li} is the measured value and V'_{Li} is the model value of volume, and n is sample size.

One popular choice for PSO parameters is $\omega = 0.7298$, $c_1 = c_2 = 1.4962$ [19]. The ratio between good particles and bad particles is decided by the swarming coefficient f . If the value of f is close to 0, the bad par-

ticles will be in the majority and the value of penalty coefficient λ will be small. The improvement for the standard PSO is not obvious. On the contrary, if the value of f is close to 1, the number of bad particles is limited and the value of λ will be large, which also leads to worse performance of SEC-PSO. The value of swarming coefficient f can be obtained from experiences and experimental simulations. The effect of the worst particle of individual and population in the updating of the velocity is controlled by the impact factors u_1 and u_2 . If their values are overlarge, the worst information will disturb the effect of the best information. However, if their values are very small, the effect of worst information will not be significant. The appropriate setting for u_1 and u_2 is increasing their values linearly with the increase of iteration number k from 1 to MaxDT, where MaxDT is the maximum iteration number. According to the experimental simulations, we set $f = 0.5$ and $u_1 = u_2 = \frac{0.001k}{\text{MaxDT}}$.

The sample size is set to $n = 200$. Fermentation period is set to $t = 16.5$ h. Initial conditions is set to $[X(0), S(0), P(0), V_L(0)] = [0.5, 7, 0, 50000]$. Identify the simplified model parameters with SEC-PSO according to Eqs. (31) and (32) and time-series data obtained from the process model (Object 1). Table 3 shows the results. Fig. 2 shows the comparison of actual data curves and model fitting curves. The model

Table 3 Parameter identification results

$K_S/\text{g}\cdot\text{L}^{-1}$	$K_e/\text{g}\cdot\text{L}^{-1}$	$K_i/\text{g}\cdot\text{L}^{-1}$	$\mu_{P\max}/\text{h}^{-1}$	$\mu_{X\max}/\text{h}^{-1}$	$Y_{X/S}/\text{g}\cdot\text{g}^{-1}$	$Y_{e/S}/\text{g}\cdot\text{g}^{-1}$	$Q_m/\text{g}\cdot\text{g}^{-1}\cdot\text{h}^{-1}$
0.34075	1.0984	5.8948	6.0824	2.0235	0.2476	0.74314	0.070687

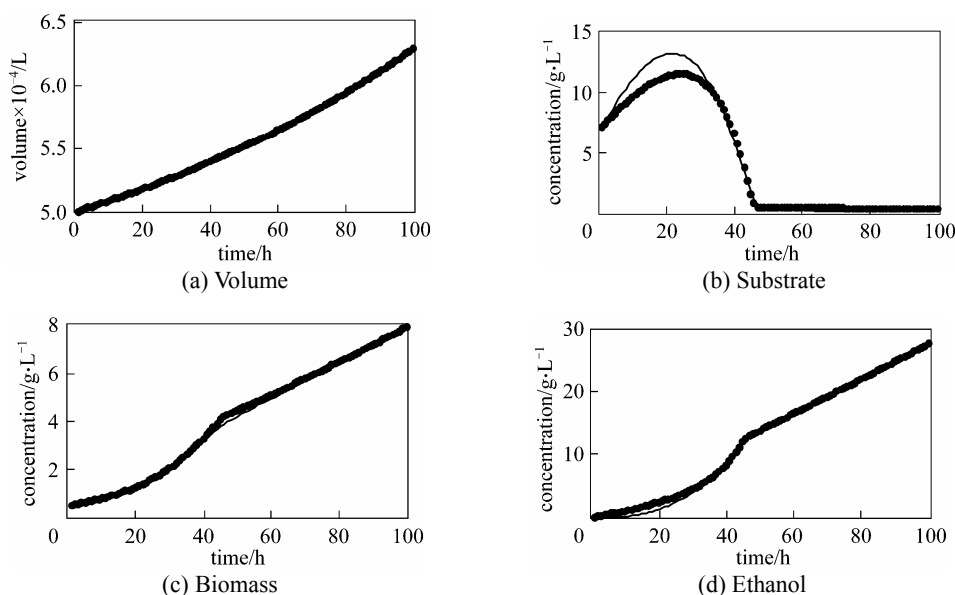


Figure 2 Comparison of hybrid model values with simulator values for yeast fermentation

● fitting curve; — original curve

fitting curves are very close to the actual data curves. The fermentation process can be described by kinetic model whose parameters are identified by using the improved PSO. Therefore, SEC-PSO is effective for parameter identification.

4.3 Operation condition optimization

The optimization objective is set as integration of time-series biomass concentration (X_i). The objective function of operation condition optimization is calculated according to the following equation

$$T_{\text{fitness}} = \sum_{i=1}^n X_i^2 \quad (34)$$

Parameters of SEC-PSO are the same as those in Section 4.2. Simplified model parameters are obtained from the results of parameter identification. Initial operation condition is set as $F1 = 2000$.

In Fig. 3 (a), the operation curve $F2$ is obtained from the optimization. After the simulation of kinetic model (Object 2), we obtain biomass concentration curves shown in Fig. 3 (b). The biomass concentration curves with $F1$ and $F2$ show the improvement of fed-batch process production. Therefore, SEC-PSO is effective for operation condition optimization.

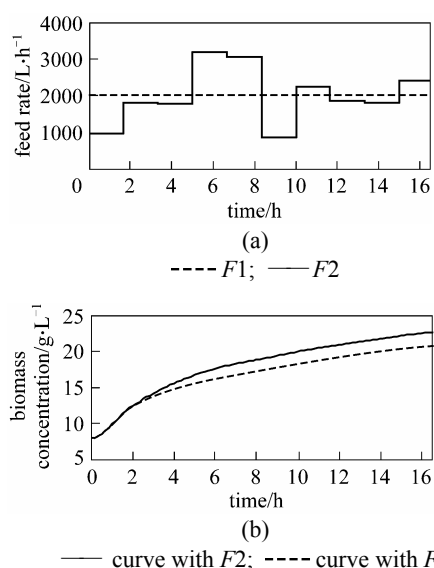


Figure 3 Feed rate and biomass concentration

4.4 Run-to-run strategy simulation

Two simulation experiments described above are prepared for the simulation experiment of run-to-run optimization strategy. Initial operation condition in the first run is set as a linear incremental curve. The fed-batch process is described by the process model (Object 1) and the kinetic model is described by simplified model (Object 2). These models are mentioned in Section 4.1.

The simulation is taken according to run-to-run

optimization strategy. The value of T_{fitness} stops increasing in the 35th runs. The experimental data from the first run to the 35th run are shown in Figs. 4–7. The dash lines show the changing trend of feed rate, biomass concentration, ethanol concentration and glucose concentration. Fig. 5 shows the improvement of biomass concentration from the first run to the 35th run obviously. These results verify the effectiveness of the run-to-run optimization strategy.

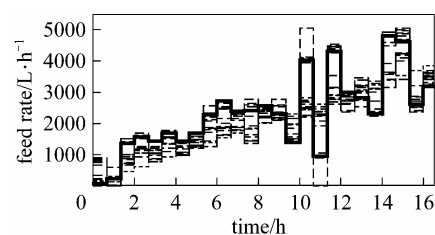


Figure 4 Feed rate of run-to-run optimization
— 1st run; — 35th run; ---- 2nd–34th runs

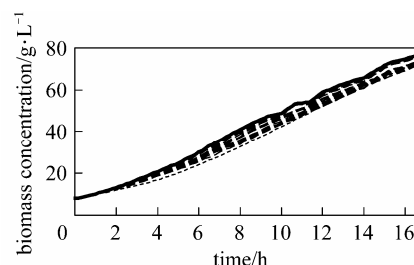


Figure 5 Biomass concentration of run-to-run optimization
— 1st run; — 35th run; ---- 2nd–34th runs

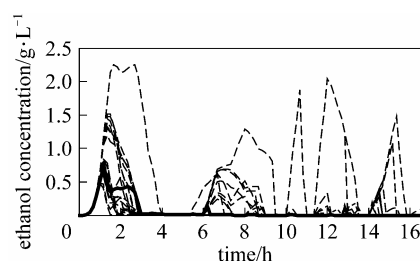


Figure 6 Ethanol concentration of run-to-run optimization
— 1st run; — 35th run; ---- 2nd–34th runs

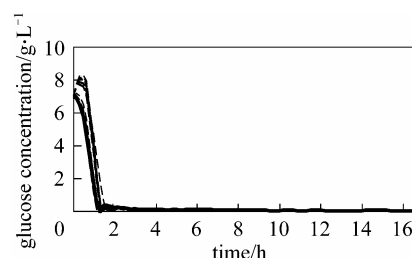


Figure 7 Glucose concentration of run-to-run optimization
— 1st run; — 35th run; ---- 2nd–34th runs

5 CONCLUSIONS

A run-to-run optimization strategy for fed-batch

process is introduced to optimize the feed rate of fed-batch process. An improved particle swarm optimization algorithm, SEC-PSO, is used in the optimization strategy to deal with two optimization problems. One is parameter identification of the kinetic model and the other is operation condition optimization of fed-batch process. Simulation results show that the strategy can improve the productivity of fed-batch process. It is effective, feasible, easy to implement with low computational complexity. Therefore, this run-to-run optimization strategy with SEC-PSO provides an effective method for optimal control of fed-batch process.

NOMENCLATURE

C_i	concentration of component i , $\text{g} \cdot \text{L}^{-1}$
c_1, c_2	learning factor or acceleration coefficient of SEC-PSO
F	feed flow rate, $\text{L} \cdot \text{h}^{-1}$
f	swarming coefficient of SEC-PSO
G	energy of swarm of SEC-PSO
g_i	energy of a single particle of SEC-PSO
J_1, J_2	object to be optimized
K_e	saturation constant for ethanol, $\text{g} \cdot \text{L}^{-1}$
K_i	inhibition constant, $\text{g} \cdot \text{L}^{-1}$
K_O	saturation constant for oxygen, $\text{g} \cdot \text{L}^{-1}$
K_S	saturation constant for substrate (glucose), $\text{g} \cdot \text{L}^{-1}$
k_{LaO}	volumetric mass transfer coefficient for oxygen, h^{-1}
k_{LaC}	volumetric mass transfer coefficient for carbon dioxide, h^{-1}
$M(t)$	run-time outputs of model simulation process
m	population size of SEC-PSO
Q_i	specific consumption or production rate, $\text{g} \cdot \text{g}^{-1} \cdot \text{h}^{-1}$
Q_m	substrate maintenance coefficient, $\text{g} \cdot \text{g}^{-1} \cdot \text{h}^{-1}$
$R_i(t)$	run-time outputs of real fed-batch process
r_1, r_2	independently uniformly distributed random variable
S_0	substrate concentration of feed, $\text{g} \cdot \text{L}^{-1}$
T_{fitness}	fitness value
t	time, h
t_f	final time
u_1, u_2	impact factors of SEC-PSO
V	volume, L
v_i	velocity of the i th particle
X	biomass concentration
x^*	best position of particle among the population of SEC-PSO
x^0	worst position of population of SEC-PSO
x_i	i th particle of SEC-PSO
x_i^*	best position in searching history of the i th particle of SEC-PSO
x_i^0	worst position of individual of SEC-PSO
Y_{ij}	yield of component i on j , $\text{g} \cdot \text{g}^{-1}$
λ	penalty coefficient of SEC-PSO
μ	specific growth rate, h^{-1}
μ_{max}	maximum specific growth rate, h^{-1}
μ_p	specific rate of production, h^{-1}
$\mu_{p\text{max}}$	maximum specific rate of production, h^{-1}
ω	inertia weight of SEC-PSO

Subscripts

C	carbon dioxide
cr	critic
e	ethanol
lim	limitation

max	maximum
O	dissolved oxygen
OX	oxidative
pr	production
red	reductive
S	substrate (glucose)
up	uptake
X	biomass

REFERENCES

- Xiong, Z.H., Zhang, J., Dong, J., "Optimal iterative learning control for batch processes based on linear time-varying perturbation model", *Chin. J. Chem. Eng.*, **16** (2), 235–240 (2008).
- Ko, C.L., Wang, F.S., "Run-to-run fed-batch optimization for protein production using recombinant *Escherichia coli*", *Biochem. Eng. J.*, **30** (3), 279–285 (2006).
- Herrera, F., Zhang, J., "Optimal control of batch processes using particle swarm optimisation with stacked neural network models", *Comput. Chem. Eng.*, **33** (10), 1593–1601 (2009).
- Kawohl, M., Heine, T., King, R., "Model based estimation and optimal control of fed-batch fermentation processes for the production of antibiotics", *Chem. Eng. Process.*, **46** (11), 1223–1241 (2007).
- Ronen, M., Shabtai, Y., Guterman, H., "Optimization of feeding profile for a fed-batch bioreactor by an evolutionary algorithm", *J. Biotechnol.*, **97** (3), 253–263 (2002).
- Li, X.B., Zhao, G.R., Yuan, Y.J., "A strategy of phosphorus feeding for repeated fed-batch fermentation of penicillin G", *Biochem. Eng. J.*, **27** (1), 53–58 (2006).
- Kennedy, J., Eberhart, R., "Particle swarm optimization", In: Proceedings of the IEEE International Conference on Neural Networks, IEEE, Perth, Australia, 1942–1948 (1995).
- Zhou, J.Z., Fang, R.C., Li, Y.H., Zhang, Y.C., Peng, B., "Parameter optimization of nonlinear grey Bernoulli model using particle swarm optimization", *Appl. Math. Comput.*, **207** (2), 292–299 (2009).
- Jiang, Y., Hu, T.S., Huang, C.C., "An improved particle swarm optimization algorithm", *Appl. Math. Comput.*, **193** (1), 231–239 (2007).
- Liu, B., Wang, L., Jin, Y.H., Tang, F., Huang, D.X., "Improved particle swarm optimization combined with chaos", *Chaos Soliton. Fract.*, **25** (5), 1261–1271 (2005).
- Luo, Y.Q., Yuan, X.G., "Global optimization for the synthesis of integrated water systems with particle swarm optimization algorithm", *Chin. J. Chem. Eng.*, **13** (4), 535–541 (2005).
- Francois, G., Srinivasan, B., Bonvin, D., "Use of measurements for enforcing the necessary conditions of optimality in the presence of constraints and uncertainty", *J. Process Control*, **15** (6), 701–712 (2005).
- Bonvin, D., Srinivasan, B., Hunkeler, D., "Control and optimization of batch processes—Improvement of process operation in the production of specialty chemicals", *IEEE Contr. Syst. Mag.*, **26** (6), 34–45 (2006).
- Fut, P.C., Barford, J.P., "A hybrid neural network-first principles approach for modeling of cell metabolism", *Comput. Chem. Eng.*, **20** (6/7), 951–958 (1996).
- Pertev, C., Turker, M., Berber, R., "Dynamic modeling, sensitivity, analysis and parameter identification of industrial yeast fermenters", *Comput. Chem. Eng.*, **21** (Suppl.), S739–S744 (1997).
- Karakuzu, C., Turker, M., Ozturk, S., "Modelling, on-line state estimation and fuzzy control of production scale fed-batch baker's yeast fermentation", *Control Eng. Pract.*, **14** (8), 959–974 (2006).
- Yuzgec, U., Turker, M., Hocalar, A., "On-line evolutionary optimization of an industrial fed-batch yeast fermentation process", *ISA T.*, **48**, 79–92 (2009).
- Chen, L.Z., Nguang, S.K., Chen, X.D., Li, X.M., "Modelling and optimization of fed-batch fermentation processes using dynamic neural networks and genetic algorithms", *Biochem. Eng. J.*, **22** (1), 51–61 (2004).
- Eberhart, R.C., Shi, Y., "Comparing inertia weights and constriction factors in particle swarm optimization", In: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, San Diego, USA, 84–88 (2000).