Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

An Improved APSO-SQP with Adaptive Transition Strategy for Dynamic Optimization ⋆

Fang Dong * **Jianming Zhang** * **Lei Xie** * **Huining Zhao** *
Xiongxiong He **

* *State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, PR China (e-mail: jmzhang@iipc.zju.edu.cn).*
** *College of Information Engineering, Zhejiang University of Technology, Hangzhou, 310032 PR China, (e-mail: hxx@zjut.edu.cn).*

Abstract: This paper employs a logistical regression classifier for combining particle swarm optimizer and local optimizer to solve dynamic optimization problems. It achieves the ability of automatic transition from global optimizer to local optimizer according to the state of solutions. The particle swarm optimizer is used to globally search for a good approximation of the solution and then SQP is applied as the local optimizer for quickly converging to the optima. Experimental results on two complex chemical processes demonstrate that the proposed method can get comparable and even better results than the existed methods in precision and especially in function evaluations.

Keywords: Optimal Control, Adaptive Particle Swarm Optimization, Local Search, Machine Learning.

1. INTRODUCTION

Dynamic optimization is a hot topic in the research areas of chemical engineering. Due to the nonlinear, multimodal, and constrained nature in processes, it is challenging to solve dynamic optimization problems. The evolutionary algorithm has been shown to be a simple effective algorithm for nonlinear optimization. Because of its ability to find global optima, a large number of researchers have applied evolutionary algorithm to optimize engineering processes.

Rajesh et al. (2001) applied the ant colony algorithm (AC) to the dynamic optimization of chemical processes combined with control vector parameterization method. Six classical chemical process problems were successfully solved and the results were comparable with other traditional derivative methods. Zhang et al. (2005) proposed an iterative ant colony algorithm and applied it to the dynamic optimization of chemical process. Babu and Angira (2006) applied the differential evolution algorithm (DE) to optimal control problems of nonlinear stirred tank reactors. Castellani et al. (2012) modified the bee algorithm and implemented it into dynamic optimization. Shelokar et al. (2008) combined the power of multicanonical sampling with the beneficial features of simulated annealing and tabu list to solve complex chemical engineering processes. Padhiyar et al. (2006) employed DE into the problem of optimal grade transition in polymerization reactors.

Evolutionary algorithm is a type of stochastic methods that can avoid being trapped in local optima to some degree. However, the searching speed is relatively slow and the solution they get in the same problem may be different from each running. To improve searching efficiency and solution precision, hybrid methods have become popular in recent years. Chen et al. (2013) combined particle swarm optimization (PSO) with quasi-Newton method for four classical dynamic processes and a high dimensional continuous stirred tank reactor (CSTR) problem. Egea et al. (2009) employed enhanced scatter search method combined with local search to the optimal control of biological reactors. Schlter et al. (2009) proposed a framework of combining ant colony optimization and local mixed integral sequential quadratic programming for nonconvex mixed integral nonlinear programming problems. Most of them adopt a two-step optimization strategy: the evolutionary algorithm searches the good initial solutions for local search and then the local optimizer converges to the local minima. However, the transition timepoint from evolutionary algorithm to local optimizer is always user defined and problem dependent. In this paper, a logistic regression classifier is applied to determine the transition from PSO to SQP according to the state of solutions. The logistic regression classifier is trained by the data from classical dynamic optimization problems. Based on this adaptive transition model, an improved adaptive particle swarm optimizer combined with SQP by logistic regression (APSO-LR) is proposed for dynamic optimization. Two complex chemical processes are used to test the proposed method.

⋆ This work was supported in part by National Natural Science Foundation of P.R. China (61374121)

2. PROBLEM STATEMENT

A typical dynamic optimization problem can be described as follows:

$$\max_{u(t)} \ J = \psi(z(t_f)) \tag{1}$$

$$\frac{dx(t)}{dt} = f(z(t), u(t), t) \tag{2}$$

$$g(z(t), u(t), t) = 0 \tag{3}$$

$$h(z(t), u(t), t) \leq 0 \tag{4}$$

Here J is the performance index of the system, $z(t)$ is the state vector, $u(t)$ is the control vector and t_f is the fixed terminal time. The task of dynamic optimization is to find the optimal control $u^*(t)$ to minimize (or maximize) the performance index J that is subject to the dynamic system (2), equality constraints (3) and inequality constraints (4) with the initial condition $z(t_0) = z_0$, where t_0 is the initial time and z_0 is the initial state.

To solve the dynamic optimization problems using evolutionary algorithm, a discretization method is needed. Sequential (also known as Control Vector Parameterization, CVP) and simultaneous methods are two typical strategies for discretization. More discretization methods and details can be found in a comprehensive review by Biegler (2007). As sequential methods only discretize the control variables, the dimension of the transformed dynamic optimization problem is relatively small compared with simultaneous method. Evolutionary algorithms are suitable to be combined with CVP, because higher dimensions will cause the curse of dimensionality for evolutionary algorithm.

Piecewise-constant control approximation is a simple and common method in control vector parameterization. It can be written as follows:

$$u(t) \approx u^p(t|\xi) = \sum_{k=1}^{n_p} \xi_k \delta_{[\tau_{k-1}, \tau_k)}(t), t \in [t_0, t_f] \tag{5}$$

where $[\tau_{k-1}, \tau_k)$ is the kth control subinterval, ξ_k is the control value on the kth subinterval, n_p is the number of control subintervals and $\delta_{[\tau_{k-1}, \tau_k)}$ is the characteristic function defined as

$$\delta_{[\tau_{k-1}, \tau_k)}(t) = \begin{cases} 1, & t \in [\tau_{k-1}, \tau_k) \\ 0, & t \notin [\tau_{k-1}, \tau_k) \end{cases} \tag{6}$$

In addition, $\tau_k, k = 1, 2, ..., n_p$, are knot points satisfying the following conditions

$$t_0 = \tau_0 \leq \tau_1 \leq \tau_2 \leq \cdots \leq \tau_{n_p - 1} \leq \tau_{n_p} = t_f \tag{7}$$

Substitute (5) into (1)-(4) yields

$$\max_{\xi_k, \tau_k, k=1,...,n_p} \ J = J^p(\xi, \tau) \tag{8}$$

$$\dot{z}(t) = f(t, z(t), \xi_k), \ t \in [\tau_{k-1}, \tau_k) \tag{9}$$

$$g^p(t, z(t), \xi_k) = 0, \ t \in [\tau_{k-1}, \tau_k) \tag{10}$$

$$h^p(t, z(t), \xi_k) \leq 0, \ t \in [\tau_{k-1}, \tau_k) \tag{11}$$

The original problem and transformed problem are equivalent in the limit as n_p approaches infinity, which has been proved by Lin et al. (2012). After control vector parameterization, the original dynamic optimization has been transformed into a nonlinear programming problems where optimization parameters are ξ_k and $\tau_k, k = 1, 2, ..., n_p$. This nonlinear programming problem can be solved by evolutionary algorithms.

3. ADAPTIVE PARTICLE SWARM OPTIMIZER WITH LOCAL SEARCH

3.1 Adaptive Particle Swarm Optimizer

Particle swarm optimization (PSO), proposed by Kennedy and Eberhart (1995), is one kind of swarm intelligence-based stochastic optimization algorithms. In PSO, a crowd of particles are considered as potential solutions of the optimization problem. Each particle i has velocity $\mathbf{v}_i = [v_i^1, v_i^2, ..., v_i^D]$ and position $\mathbf{x}_i = [x_i^1, x_i^2, ..., x_i^D]$ describing their states, where D is the dimension of problem. A particle searches for the solution depending on its historical personal best position stored by itself and the information shared among other particles. Each particle i updates its velocity $\mathbf{v}_i$ and position $\mathbf{x}_i$ as follows:

$$\begin{aligned} v_i^d(q+1) = \ & w \cdot v_i^d(q) + c_1 \cdot r_1(x_{pbest_i}^d(q) - x_i^d(q)) \\ & + c_2 \cdot r_2(x_{gbest}^d(q) - x_i^d(q)) \end{aligned} \tag{12}$$

$$x_i^d(q+1) = x_i^d(q) + v_i^d(q+1) \tag{13}$$

where $x_i^d(q)$ and $v_i^d(q)$ represent the position and velocity of particle i in the dth dimension respectively; q is the iteration number of the evolutionary process. $\mathbf{x}_{pbest_i}$ is the personal best position found by particle i, and $\mathbf{x}_{gbest}$ is the global best position found by the whole population so far. In addition, $w \in (0, 1)$ is an inertia weight used to balance the exploitation and exploration ability of PSO proposed by Shi and Eberhart (1998). c_1, c_2 are acceleration constants. r_1 and r_2 are random numbers generated according to the uniform distribution in $(0, 1)$.

Based on the standard particle swarm optimizer, adaptive particle swarm optimizer (APSO) was proposed by Zhan et al. (2009). It automatically adjusts the parameters in PSO according to the position distribution of the swarm. In addition, a novel elite learning strategy was also proposed in APSO to enhance the swarm diversity in optimization. Because of its good performance in singular and multimodal functions, it is applied in our method in this paper. Due to the space limit, more information about APSO can be found in the reference of Zhan et al. (2009).

3.2 APSO with Local Optimizer for Dynamic Optimization

In order to solve the dynamic optimization problems by APSO more effectively at the beginning stage of optimization, constraints are preprocessed. In the discretized problem described by (7)-(11), there are two types of constraints. Constraints like (10) and (11) are called path constraints. Path constraint is the constraint where relevant state variables and control parameters must satisfy each other from the initial time to the terminal time. When dealing with these kinds of constraints, a penalty method is adopted by adding constraints into the objective functions with some penalty coefficients as described in Teo et al. (1993). A mapping function is used to change the solutions generated by APSO into the solutions satisfy

the constraints in (7). The structure of solutions in APSO for dynamic optimization are as follows

$$X = [x_1, x_2..., x_{n_p}, \xi_1, \xi_2, ..., \xi_{n_p}] \tag{14}$$

where $\xi_k, k = 1, 2, ..., n_p$ is the control variable on the kth subinterval in (5), the time point variable in (7) is calculated as follows

$$\tau_k = \tau_{n_p} \cdot \sum_{i=1}^{k} \frac{|x_i|}{\sum_{j=1}^{n_p-1} |x_j|} + \tau_0, k = 1, 2, ..., n_p - 1 \tag{15}$$

With this mapping function the solution of the dynamic optimization problem can be constructed as follows

$$S = [\tau_0, \tau_1, \tau_2..., \tau_{n_p-1}, \tau_{n_p}, \xi_1, \xi_2, ..., \xi_{n_p}] \tag{16}$$

where $\tau_0 = t_0, \tau_{n_p} = t_f$. With the control profile in (5), initial time τ_0 and terminal time τ_{n_p} the state vector $z(t)$ can be calculated through solving the differential equation (9) by solvers such as Runge-Kutta or any other ordinary differential equation solvers. In addition, the objective function can be calculated by (1). Therefore, the original dynamic optimization problem is changed into a normal nonlinear optimization problem without constraints, which can be solved by evolutionary algorithms such as APSO.

Depend on the solution found by APSO in the first stage, sequential quadratic programming, SQP (Boggs and Tolle (1995)) is then used as the local searcher to accelerate the searching process and improve the solution precision. As SQP is a gradient based solver, the gradient of objective with respect to the decision variables can be effectively calculated through solving a sensitivity differential equation as shown in Loxton et al. (2008). The convergence of APSO-LR is determined by the local search SQP, and other gradient based algorithms can also be applied.

3.3 Transition from APSO to Local Search

In our two stage optimizer, APSO is used to conduct a preliminary search of the feasible region, which offers a good initial feasible solution for SQP, and the SQP finally makes solutions converge to the local minima. The method for transition from APSO to SQP is very important. Most of algorithms simply use a threshold in function evaluations to control the transition from APSO to local search. In this paper, a logical regression classifier in John et al. (1996) is used to determine the transition according to the state of solutions, because it is simple to implement and high efficiency in training process, which will not increase too much complexities compared with the original algorithm. Logical regression is one kind of binary classifier, the data are labeled with 0/1, where label 0 stands for making transition and label 1 stands for keeping current state.

Six classical dynamic optimization problems in Rajesh et al. (2001) are selected to generate the training set for logistic regression, including an unconstrained mathematical program, nonlinear unconstrained mathematical program, tubular reactor parallel reaction problem, bath reactor consecutive reaction problem, plug flow reactor catalyst blend and one reactor problem. They are all unconstrained, small type dynamic optimization problems with only one control variable. They are solved by APSO and 9 features are recorded as training set for logistic regression, among which three features relevant with global best particle are calculated as follows

$$fe = \frac{d_g - d_{\min}}{d_{\max} - d_{\min}} \tag{17}$$

$$d_i = \frac{1}{N_{sp} - 1} \sum_{j=1, i \neq i}^{N_{sp}} \sqrt{\sum_{k=1}^{D} (x_i^k - x_j^k)^2} \tag{18}$$

$$Ss = \frac{1}{D} \sum_{i=1}^{D} \left(\frac{1}{N_{sp} - 1} \sqrt{\sum_{j=1}^{N_{sp}} (\hat{x}_j^i - \bar{\hat{x}}^i)^2}\right) \tag{19}$$

$$Oi = \frac{|f_g(q) - f_g(q-1))|}{|f_g(q)| + \varepsilon} \cdot 100\% \tag{20}$$

where q is the number of iterations, N_{sp} is the number of swarm in APSO, D is the dimension of transformed optimization problem, $g \in \{1, 2, ..., N_{sp}\}$ stands for the best particle's index in the current swarm, $\hat{x}_j^i$ is the normalized solution of the jth particle in ith dimension, $\bar{\hat{x}}^i$ is the average of normalized solutions among the swarm in the ith dimension and $f_g(q)$ is the global best particles fitness value (objective value) in qth iteration. fe is the factor which represents the distribution of particles in relation to the global best particle in the current swarm. Ss reflects the variance of swarm. Oi is the increase rate of objective's global best value in current iteration. $\varepsilon = 0.00001$ is used to avoid zero division.

Apart from these features another six features are supplemented. They are iteration count q, number of function evaluations fes, mean value of each particle's normalized solution value in each dimension Sm, mean distances between each particle and other particles dm, average of each particle's historically accumulative stagnation count $Staym$, and the global best particle's continuous stagnation count $Stayg$. The stagnation is defined as the statement that one particle's personal historical best solution can't get improved within one iteration. Among them Sm and dm reflects the distribution of swarm, which are calculated as follows

$$Sm = \frac{1}{D} \sum_{i=1}^{D} \left(\frac{1}{N_{sp}} \sum_{j=1}^{N_{sp}} \hat{x}_j^i\right) \tag{21}$$

$$dm = \frac{1}{N_{sp}} \sum_{i=1}^{N_{sp}} d_i \tag{22}$$

Therefore, training set for logistic regression is defined as $X_{fe,i} = [l_i, fes_i, fe_i, Ss_i, Sm_i, Oi_i, dm_i, Staym_i, Stayg_i, 1]$ with label $class_i$, where i is the index of training sample and the last term 1 in $X_{fe,i}$ is added to correspond with the constant term in logistic regression model. Samples are labeled as follows, if current global best solution is within the domain of global optimal, it is labeled as class 1, otherwise it is labeled as class 0. As the optimal solutions of each classical optimization problem for benchmarks are known, we can label sample as class 1 if it satisfies $\frac{|f(x) - f(x_{opt})|}{|f(x_{opt})|} \leq \upsilon$, otherwise it is labeled as class 0. υ is 0.05 according to experiments.

3.4 Training for Logistic Regression

All six dynamic optimization problems are solved 5 times separately and the number of particles and maximal number of iterations in APSO is set to be 30 and 100 separately. In addition, the number of subintervals n_p in (15) is set to be 5 in each problem. These parameters are set through experiments to make sure that APSO can find the optimal regions of all problems. As a result, 1400 feature data are uniformly selected during optimization, among which 700 are class 1 and 700 are class 0. The logistic model is trained through quasi-Newton method introduced in Gilbert and Lemarechal (2006) combined with 20 cross-validations. The average training error is 0.898. The trained model is described as follows

$$P_{tran} = \frac{1}{e^{-(W \cdot X_{fe}^T)} + 1} \tag{23}$$

$$y = \begin{cases} 1, & P_{tran} \geq \eta \\ 0, & P_{tran} < \eta \end{cases} \tag{24}$$

where $W = [0.1955, 0.0188, -0.8075, 23.4985, -0.8823, -2.9365, -0.032, -0.8082, -0.1962, -7.337]$. P_{tran} is the probability for transition. η is usually set to be 0.5, while in our algorithm η is set to be 0.4 to introduce some bias into classification. This bias is shown to perform well in experiments. y is the flag of transition from APSO to local search. That is when y becomes 1 the search of APSO is stopped and the local searcher is activated to converge to local optimal of problem. In terms of machine learning, the pattern we tried to learn through logistic regression classifier is current solution's degree of closeness to the optima, which is reflected by the features abstract from solutions. Based on the generation ability of logistic regression classifier, the similar pattern in other problems can also be found by this trained classifier to some extent. However, more researches on the generation ability of APSO-LR in solving dynamic optimization problems will be our future work.

4. EXPERIMENTAL STUDIES

APSO is a parameter adaptive particle swarm optimizer. There are three user defined parameters besides the core module. They are maximum of iteration number q_{max}, maximum of function evaluations FES_{max} and number of particle swarms N_{sp}. In this paper, $q_{max} = 100$, $FES_{max} = 3000$, $N_{sp} = 30$ in all experiments. The number of subintervals in dynamic optimization n_p is set according to the problem.

Two complex dynamic optimization problems are used to test our algorithm compared with other algorithms from the literatures. All these problems are run 10 times separately solved by APSO-LR. The best, mean and worst results of objective values and the number of function evaluations of each problem are shown in Table 1 and Table 2, respectively.

4.1 Problem 1

The first problem is about the optimal production of secreted protein in a fed-batch reactor, which was introduced by Park and Fred Ramirez (1988). It has been studied in Luus and Hennessy (1999), Upreti (2004), Chen et al. (2013), Angira and Santosh (2007). The mathematical model is as follows

$$\begin{aligned} &Max\ J(t_f) = z_1(t_f)z_5(t_f) \\ &s.t. \begin{cases} \dot{z}_1 = g_1(z_2 - z_1) - \dfrac{u}{z_5}z_1 \\ \dot{z}_2 = g_2 z_3 - \dfrac{u}{z_5}z_2 \\ \dot{z}_3 = g_3 z_3 - \dfrac{u}{z_5}z_3 \\ \dot{z}_4 = -7.3 g_3 z_3 + \dfrac{u}{z_5}(20 - z_4) \\ \dot{z}_5 = u \end{cases} \\ &g_1 = \frac{4.75 g_3}{0.12 + g_3}, g_2 = \frac{z_4}{0.1 + z_4}e^{(-5.0 z_4)} \\ &g_3 = \frac{21.87 z_4}{(z_4 + 0.4)(z_4 + 62.5)} \\ &0 \leq u \leq 2, t_f = 15, z(0) = (0, 0, 1, 5, 1) \end{aligned} \tag{25}$$

where z_1 and z_2 are the concentrations of secreted and total protein, respectively. z_3 and z_4 are the concentrations of cell and glucose, respectively, z_5 is the volume of reactor. g_1, g_2, g_3 are kinetic parameters. The control variable u is the feed rate to the reactor, and the objective is to maximize the product of concentration of protein z_1 and the volume z_5 at final time t_f.

Comparison results with other results from literatures are shown in Table 1, and the optimal control profile of problem 1 is shown in Fig. 1, which is plotted according to the best results of APSO-LR among 10 runs. Symbol $'/'$ in Table 1 stands for the data that are not mentioned in the literature.

Table 1. Comparison Results of Problem 1

Reference	Algorithm	n_p	N_{sp}	Best	Mean	FES
This work	APSO-LR	**15**	30	**32.69296**	32.68401	**879**
Chen et al. (2013)	HGPSO	31	62	32.61308	32.60809	30037
		45	90	32.68604	32.65554	42594
Luus and Hennessy (1999)	Direct Search	31	/	32.61339	/	/
		45	/	32.68687	/	/
Upreti (2004)	GA	45	/	32.59277	/	/
Angira and Santosh (2007)	TDE	45	30	32.68434	/	/

From Table 1, APSO-LR performs better than other algorithms on both precision and function evaluations. The number of function evaluations is largely reduced compared with other algorithms, which is the construction of gradient calculating method in local search SQP and the transition model trained by logistic regression. In APSO-LR, the gradients of objective with respect to control variables are calculated by solving the sensitivity differential equation (Loxton et al. (2008)) instead of finite difference method which needs lots of function evaluations. The transition model trained by logistic regression determines the appropriate time at which APSO is switched to SQP, which reduces lots of unnecessary number of function evaluations. In this problem, we set the number of subintervals to be 15. According to Table 1, small number of subintervals in the proposed algorithm can get the compared or even better results than big ones in other algorithms.

4.2 Problem 2

The second problem is feeding-rate optimization of foreign protein production that is studied by Lee and Ramirez (1994) and Roubos et al. (1999). The model contains

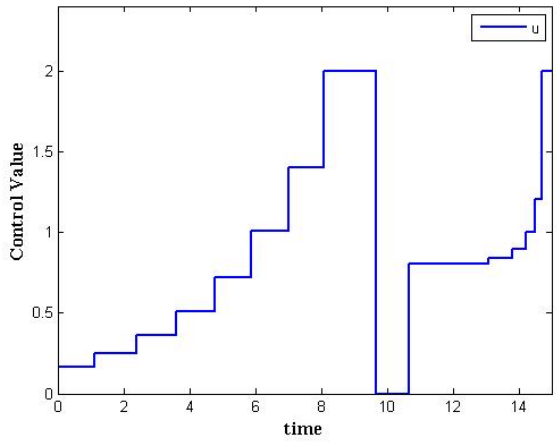

Fig. 1. Optimal Control u for Problem 1

seven state variables and two control variables, and can be described as follows

$$\begin{aligned}
\frac{dx_1}{dt} &= u_1 + u_2 \\
\frac{dx_2}{dt} &= \mu z_2 - \frac{u_1+u_2}{z_1} z_2 \\
\frac{dx_3}{dt} &= \frac{u_1}{z_1} C_{nf} - \frac{u_1+u_2}{z_1} z_3 - Y^{-1}\mu z_2 \\
\frac{dx_4}{dt} &= R_{fp} z_2 - \frac{u_1+u_2}{z_1} z_4 \\
\frac{dx_5}{dt} &= \frac{u_2}{z_1} C_{if} - \frac{u_1+u_2}{z_1} z_5 \\
\frac{dx_6}{dt} &= -k_1 z_6 \\
\frac{dx_7}{dt} &= k_2(1-z_7) \\
\mu &= \frac{0.407 z_3}{0.108 + z_3 + \frac{z_3^2}{14814.8}} (z_6 + z_7 \frac{0.22}{0.22+z_5}), C_{if} = 4.0 \\
R_{fp} &= \frac{0.095 z_3}{0.108 + z_3 + \frac{z_3^2}{14814.8}} (\frac{0.0005+z_5}{0.22+z_5}), C_{nf} = 100.0 \\
k_1 = k_2 &= \frac{0.09 z_5}{0.034 + z_5}, \; Y = 0.51
\end{aligned} \tag{26}$$

The state variables are reaction volume $z_1(L)$, cell density $z_2(g/L)$, nutrient concentration $z_3(g/L)$, foreign protein concentration $z_4(g/L)$, inducer concentration $z_5(g/L)$, inducer shock factor on the cell growth rate z_6 and inducer recovery factor on the cell growth rate z_7. The two control variables are glucose feed rate $u_1(L/h)$ and inducer feed rate $u_2(L/h)$, respectively. $Y, C_{nf}, C_{if}, \mu, R_{fp}, k_1, k_2$ are growth yield coefficient, nutrient concentration in inducer feed, specific growth rate, foreign production rate, shock and recovery parameters, respectively. The objective is to find the optimal control profile u_1 and u_2 to maximize $J(u_1, u_2)$, which is given by the following equation

$$J(u_1, u_2) = z_4(t_f) z_1(t_f) - Q \int_0^{t_f} u_2(t) dt \tag{27}$$

where $Q = 5$ is the price ratio of inducer to product. $t_f = 10$h and the initial conditions for dynamic system are $z(0) = [1, 0.1, 40, 0, 0, 1, 0]^T$. Boundary constants for u_1 and u_2 are within the range of $[0, 2]L/h$.

Table 2. Comparison Results of Problem 2

Reference	Algorithm	n_p	N_{sp}	Best	Mean	FES
This work	APSO-LR	5	30	0.8165	0.8146	**533**
Tholudur and Ramirez (1996)	ANN	/	/	0.8030	/	/
Roubos et al. (1999)	GA	/	/	0.8149	/	1000000
Mekarapiruk and Luus (1997)	IDP	/	/	0.8164	/	/
Jayaraman et al. (2001)	CACO	/	/	0.8095	/	26763
Sarkar and Modak (2003)	ANNSA	/	/	**0.8166**	/	1729601
Zhang et al. (2005)	SACA	/	/	0.8158	/	120000

Comparison results with other results from the literature are shown in Table 2, and the optimal control profile of u_2 is shown in Fig. 2. The optimal control of u_1 is zero. In this problem, APSO-LR gets the results comparable with the best result from Sarkar and Modak (2003). However, APSO-LR reduces the number of function evaluations greatly.

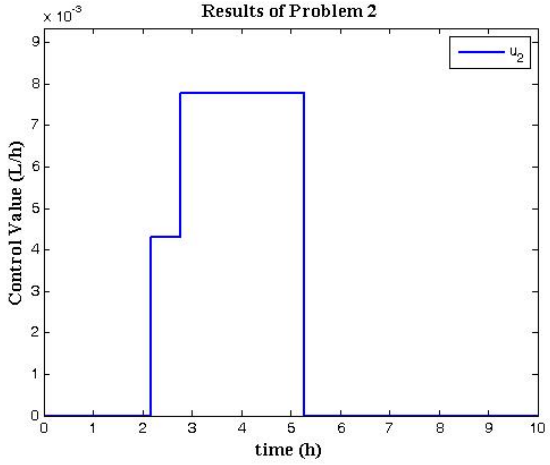

Fig. 2. Optimal Control u_2 for Problem 2

Summary results of two dynamic optimization problems are listed in Table 3 where the effect of logistic regression can be shown. fes_{APSO} and fes_m represent the mean function evaluations for APSO and the total function evaluations in APSO-LR, respectively. fes_{APSO} is determined by the logistic regression depending on the state of current solutions automatically.

Table 3. Summary of Results from APSO-LR

No.	*best*	*worst*	*mean*	*std*	fes_{APSO}	fes_m
1	32.69296	32.67269	32.68401	8.26E-03	727	879
2	0.81648	0.80725	0.81462	3.88E-03	462	533

5. CONCLUSIONS

This paper combines a powerful adaptive particle swarm optimizer with SQP as the local search optimizer to solve the dynamic optimization problem. A logistic regression model is used to control the transition from APSO to SQP. As APSO and our transition model are both adaptive, our method APSO-LR is an adaptive algorithm for dynamic optimization, which needs no extra parameters besides the basic parameters for evolutionary algorithm and dynamic optimization. Two complex dynamic optimization problems are used to test APSO-LR and the results are compared with other algorithms from the literatures. All results show that APSO-LR performs comparable or even better than other algorithms while greatly reducing the

number of function evaluations. In addition, the combination framework used in APSO-LR can be used in particle swarm optimizer which combined with other local optimizers.

REFERENCES

Angira, R. and Santosh, A. (2007). Optimization of dynamic systems: A trigonometric differential evolution approach. *Computers and Chemical Engineering*, 31(9), 1055 – 1063.

Babu, B. and Angira, R. (2006). Modified differential evolution (mde) for optimization of non-linear chemical processes. *Computers and Chemical Engineering*, 30(67), 989 – 1002.

Biegler, L.T. (2007). An overview of simultaneous strategies for dynamic optimization. *Chemical Engineering and Processing: Process Intensification*, 46(11), 1043 – 1053.

Boggs, P.T. and Tolle, J.W. (1995). Sequential quadratic programming. *Acta Numerica*, 4, 1–51.

Castellani, M., Pham, Q.T., and Pham, D.T. (2012). Dynamic optimisation by a modified bees algorithm. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 226(7), 956–971.

Chen, X., Du, W.L., Qi, R.B., Qian, F., and Tianfield, H. (2013). Hybrid gradient particle swarm optimization for dynamic optimization problems of chemical processes. *Asia-Pacific Journal of Chemical Engineering*, 1–15.

Egea, J.A., Balsa-Canto, E., Garcia, M.S.G., and Banga, J.R. (2009). Dynamic optimization of nonlinear processes with an enhanced scatter search method. *Industrial and Engineering Chemistry Research*, 48(9), 4388–4401.

Gilbert, J.C. and Lemarechal, C. (2006). *Numerical optimization: theoretical and practical aspects*. Springer.

Jayaraman, V.K., Kulkarni, B.D., Gupta, K., Rajesh, J., and Kusumaker, H.S. (2001). Dynamic optimization of fed-batch bioreactors using the ant algorithm. *Biotechnology Progress*, 17(1), 81–88.

John, N., William, W., H., K.M., et al. (1996). *Applied linear statistical models*, volume 4. Irwin Chicago.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE International Conference in Neural Networks*, 1942–1948.

Lee, J. and Ramirez, W.F. (1994). Optimal fed-batch control of induced foreign protein production by recombinant bacteria. *AIChE Journal*, 40(5), 899–907.

Lin, Q., Loxton, R., Teo, K.L., and Wu, Y.H. (2012). Optimal control computation for nonlinear systems with state-dependent stopping criteria. *Automatica*, 48(9), 2116 – 2129.

Loxton, R.C., Teo, K.L., and Rehbock, V. (2008). Optimal control problems with multiple characteristic time points in the objective and constraints. *Automatica*, 44(11), 2923 – 2929.

Luus, R. and Hennessy, D. (1999). Optimization of fed-batch reactors by the luus-jaakola optimization procedure. *Industrial and Engineering Chemistry Research*, 38(5), 1948–1955.

Mekarapiruk, W. and Luus, R. (1997). Optimal control of inequality state constrained systems. *Industrial and Engineering Chemistry Research*, 36(5), 1686–1694.

Padhiyar, N., Bhartiya, S., and Gudi, R.D. (2006). Optimal grade transition in polymerization reactors: a comparative case study. *Industrial and Engineering Chemistry Research*, 45(10), 3583–3592.

Park, S. and Fred Ramirez, W. (1988). Optimal production of secreted protein in fed-batch reactors. *AIChE Journal*, 34(9), 1550–1558.

Rajesh, J., Gupta, K., Kusumakar, H.S., Jayaraman, V.K., and Kulkarni, B.D. (2001). Dynamic optimization of chemical processes using ant colony framework. *Computers and Chemistry*, 25(6), 583 – 595.

Roubos, J., Straten, G.V., and Boxtel, A.V. (1999). An evolutionary strategy for fed-batch bioreactor optimization:concepts and performance. *Journal of Biotechnology*, 67(23), 173 – 187.

Sarkar, D. and Modak, J.M. (2003). Annsa: a hybrid artificial neural network/simulated annealing algorithm for optimal control problems. *Chemical Engineering Science*, 58(14), 3131 – 3142.

Schlter, M., Egea, J.A., and Banga, J.R. (2009). Extended ant colony optimization for non-convex mixed integer nonlinear programming. *Computers and Operations Research*, 36(7), 2217 – 2229.

Shelokar, P.S., Jayaraman, V.K., and Kulkarni, B.D. (2008). Multicanonical jump walk annealing assisted by tabu for dynamic optimization of chemical engineering processes. *European Journal of Operational Research*, 185(3), 1213 – 1229.

Shi, Y. and Eberhart, R. (1998). A modified particle swarm optimizer. In *The 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence*, 69–73.

Teo, K., Rehbock, V., and Jennings, L. (1993). A new computational algorithm for functional inequality constrained optimization problems. *Automatica*, 29(3), 789 – 792.

Tholudur, A. and Ramirez, W.F. (1996). Optimization of fed-batch bioreactors using neural network parameter function models. *Biotechnology Progress*, 12(3), 302–309.

Upreti, S.R. (2004). A new robust technique for optimal control of chemical engineering processes. *Computers and Chemical Engineering*, 28(8), 1325 – 1336.

Zhan, Z.H., Zhang, J., Li, Y., and H.S.-H., C. (2009). Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,*, 39(6), 1362–1381.

Zhang, B., Chen, D., and Zhao, W. (2005). Iterative ant-colony algorithm and its application to dynamic optimization of chemical process. *Computers and Chemical Engineering*, 29(10), 2078 – 2086.