Iranian Journal of Mathematical Sciences and Informatics
Vol. 12, No. 1 (2017), pp 47-67
DOI: 10.7508/ijmsi.2017.01.005

Integrating Differential Evolution Algorithm with Modified Hybrid GA for Solving Nonlinear Optimal Control Problems

Saeed Nezhadhosein[a,*], Aghile Heydari[a], Reza Ghanbari[b]

[a]Department of Applied Mathematics, Payame Noor University, Tehran, Iran.
[b]Department of Applied Mathematics, Faculty of Mathematical science, Ferdowsi University of Mashhad, Mashhad, Iran.

E-mail: s_nejhadhosein@pnu.ac.ir
E-mail: a_heidari@pnu.ac.ir

Abstract. Here, we give a two-phase algorithm based on integrating differential evolution (DE) algorithm with modified hybrid genetic algorithm (MHGA) for solving the associated nonlinear programming problem of a nonlinear optimal control problem. In the first phase, DE starts with a completely random initial population where each individual, or solution, is a random matrix of control input values in time steps. After phase 1, to achieve more accurate solutions, we increase the number of time steps. The values of the associated new control inputs are estimated by linear or spline interpolations using the curves computed in the phase 1. In addition, to maintain the diversity in the population, some additional individuals are added randomly. Next, in the second phase, MHGA starts by the new population constructed by the above procedure and tries to improve the obtained solutions at the end of phase 1. The numerical results showed that the proposed algorithm will find almost better solution than other proposed algorithms.

Keywords: Nonlinear optimal control problem, Differential evolution, Modified hybrid genetic algorithm, Successive quadratic programming.

2000 Mathematics subject classification: 49J15.

*Corresponding Author

Received 26 June 2014; Accepted 03 May 2015

1. Introduction

Nonlinear optimal control problems (NOCP) are dynamic optimization problems with many applications in industrial processes such as airplane [19], robotic arm [8], bio-process system, sequencing batch reactor [21], biomedicine [24], chemical processes [25, 34], electric power systems [28] and etc., [16].
NOCPs are formulated as optimization problems by the performance index as the objective function and differentiate equations as constraints that called dynamic optimizations. There are different types of these problems e.g. tracking problem, terminal control problem and minimize time problem [16]. We consider bounded continuous-time continuous control problem in which a control function, u, is exerted over the planning horizon $[t_0, t_f]$. The particular problem considered is that of finding the control input $u(.) \in \mathbb{R}^m$ that minimize the cost functional:

$$min \quad J(u(t)) = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt \tag{1.1}$$

subject to:

$$\dot{x}(t) = f(x(t), u(t), t), \qquad t \in [t_0, t_f] \tag{1.2}$$

$$c_i(x(t), u(t), t) = 0, \quad i = 1, \ldots, n_c, \quad t \in [t_0, t_f] \tag{1.3}$$

$$d_i(x(t), u(t), t) \leq 0, \quad i = 1, \ldots, n_d, \quad t \in [t_0, t_f] \tag{1.4}$$

$$x(t_0) = x_0, \tag{1.5}$$

$$\psi_i(x(t_f), t_f) = 0, \quad i = 1, \ldots, n_\psi. \tag{1.6}$$

where $x(.) \in \mathbb{R}^n$ denotes the state variables for the system. The performance index (1.1) must be minimized subject to dynamic (1.2), control and state equality constraints (1.3) and control and state inequality constraints (1.4), the initial condition (1.5) and the final state constrains (1.6).
Many methods for solving NOCPs, eqns (1.1)-(1.6), either direct or indirect, rely upon gradient information and therefore may converge to a local optimum [1]. In direct methods [16, 29], the original continuous-time problem is approximated by a finite-dimensional nonlinear programming problem using discrete states and control variables. The major drawback of these methods is the lack of accuracy. In indirect approaches, the problem through the use of the Pontryagins minimum principle (PMP), is converted to two boundary value problems (TBVP) that can be solved by numerical methods such as shooting method [16]. These methods have two major disadvantages. First, they may converge to a local optimum. Next, they require good initial guesses that lie within the domain of convergence.
Metaheuristics as the global optimization methods can overcome these problems. They differ from classic methods. They don't really need good initial guesses and deterministic rules. Some of these methods are; Genetic Algorithm

(GA), see [1, 30, 31], Genetic Programming (GP), see [17], Particle Swarm Optimization (PSO), see [3, 4, 23], Ant Colony Optimization (ACO), see [36] and Differential Evolution (DE), see [7, 18, 37].

Many authors proposed many types of metaheuristics for solving NOCPs. Wang and Chiou [37] proposed a DE to solve NOCPs described by differential-algebraic systems with nonlinear constraints. Lee et al. [18] used a modified DE algorithm for dynamic optimization of a continuous polymer reactor. Sim et al. [31] combined a GA and the shooting method for solving optimal control problems. Cruz et al. [7] used efficient DE algorithms for solving multi-modal NOCPs. Firstly they used DE to approximate the global minimum. Next, a classical local optimization algorithm was used to compute the global optimum as accurate as possible. Arumugam and Rao [4] considered the popular GA operators, cross-over and root mean square variants into PSO algorithm to make a faster convergence. Arumugam et al. [3] used various optimization algorithms, including PSO, with time varying inertia weight methods, and PSO with globally and locally tuned parameters to solve the NOCPs for steel annealing processes. van Ast et al. [36] proposed a novel ACO approach to solve NOCP. Kumar and Balasubramaniam [17], using GA, solved NOCP for a linear system with quadratic performance.

Shi et al. [30] presented an improved GA with variable population-size inspired by the natural features of the variable size of the population used to continuous optimization problems. Recently, Ghosh et al. [14] used an ecologically inspired optimization technique for solving NOCPs. They used Bézier curves to parameterize the control functions. Abo-Hammour et al. [1] applied the continuous GA for solving NOCPs. They used smooth genetic operators to solve NOCPs. Modares and Naghibi-Sistani [23] proposed a hybrid algorithm by integrating an improved PSO with SQP [6]. Li et al. [20] used a PSO based method to obtain the time-optimal bang-bang control law for both linear and nonlinear systems.

Based on the success of the metaheuristics for solving NOCPs, mentioned above, we propose a two-phase algorithm based on integrating DE with MHGA. At first, to solve an NOCP, the time interval is uniformly divided by using a constant number of time steps. Next, in each of these time steps, the control variable is approximated by a scaler matrix of control input values. Thus, an infinite dimensional NOCP is changed to a finite dimensional nonlinear programming (NLP). Now, we encounter two conflict situations: the quality of the global solution and the needed computational time. In other words, when the number of time steps is increased then we expect the quality of the global solution is also increased but we know that in this situation the computational time is increased dramatically. In other situation, we consider less number of time steps, then the computational time is decreased but we may find a poor local solution. To conquer these problems, a two-phase algorithm is proposed.

In the first phase of the proposed algorithm (exploration phase), to decrease the computational time and to find a promising solution in the search space, DE algorithm is applied with a less number of time steps. After phase 1, to increase the quality of solutions obtained from phase 1, the number of time steps is increased. Using the obtained solution in the phase 1, the values of the new control inputs are estimated by linear or spline interpolations. Next, in the second phase, MHGA (exploitation phase) is applied which uses the solutions constructed by the above procedure, as an initial population.
In this algorithm, we give a MHGA which combines GA with SQP, see [6]. SQP is an iterative algorithm for solving NLP and uses gradient information. Also, it can be used for solving NOCPs, see [12, 15, 35]. MHGA decreases the number of iterations in GA, also may increase the quality of the obtained solutions. Moreover, for decreasing the running time in the early generations (iterations) of MHGA, a less number of iterations for SQP was used and then, when the promising region of search space was found, we increase the number of iterations of SQP gradually.
The paper is organized as follows: in Section 2, a description of DE algorithm and our proposed MHGA are presented. In Section 3, we introduce an algorithm for solving NOCP. In Section 4, we provide more than 25 NOCPs to examine our proposed algorithm. Results are compared with some numerical and heuristic methods. Also, a statistical approach is done for the proposed algorithm. We conclude in Section 5.

2. Overview of DE and MHGA

Here, we introduce DE algorithm and our proposed MHGA as subprocedures for the main algorithm. First, the control variables are parametrized. Next, NOCP is changed into a finite dimensional NLP, (See [12]).

2.1. **DE algorithm.**

DE algorithm introduced by Storn and Price in 1996 [33], is a population based algorithm which has three main operators; mutation, crossover and selection [7].
The underlying DE has the following steps:
Initialization: The time interval is divided into $N_t - 1$ equidistant subintervals with time steps $t_0 < t_1 < \ldots < t_{N_t-1} = t_f$ and then control input values are computed (or selected) randomly, by the following stages:

(1) Let $t_k = t_0 + kh$, where $h = \frac{t_f - t_0}{N_t - 1}, k = 0, 1, \ldots, N_t - 1$ and t_0 and t_f are the initial and final times, respectively.
(2) The corresponding control input value at each time step t_k is an $m \times 1$ vector, u_k. So, each individual of the population is an $m \times N_t$ matrix, with following components:

$$u_{ij} = u_{left,i} + (u_{right,i} - u_{left,i}).r_{ij}, \quad i = 1, 2, \ldots, m, j = 1, \ldots, N_t \qquad (2.1)$$

where r_{ij} is a random number in $[0,1]$ with a uniform distribution and $u_{left}, u_{right} \in R^m$ are the lower and the upper bound vectors of control input values, which can be given by the problem's definition or the user (e.g. see the NOCPs no. 5 and 6 in Appendix).
Next, we let $U^{(k)} = [u_0, u_1, \ldots, u_{N_t-1}] = (u_{ij})_{m\times N_t}, k = 1, 2, \ldots, N_p$ as the k-th individual of the population, control input matrix, which N_p is the size of the population.

Evaluation: The corresponding state matrix with the position matrix, $U^{(k)}, k = 1, 2, \ldots, N_p$, is an $n \times N_t$ matrix, $X^{(k)} = [x_0, x_1, \ldots, x_{N_t-1}]$, where x_j, $j = 0, 1, \ldots, N_t - 1$, is an $n \times 1$ vector as the $(j+1)$-th column of state matrix, and can approximately be computed by the forth Runge-Kutta method on dynamic system (1.2) with the initial condition (1.5). Then, the performance index, $J(U^{(k)})$, is approximated by a numerical method, $\tilde{J}$. If NOCP includes equality or inequality constraints (1.3) or (1.4), then we add some penalty terms to the corresponding fitness value of the solution. Finally, we assign $I(U^{(k)})$ to $U^{(k)}$ as the fitness value as follows:

$$I(U^{(k)}) = \tilde{J} + \sum_{i=1}^{n_d}\sum_{j=0}^{N_t-1} M_{1i} max\{0, d_i(x_j, u_j, t_j)\} + \sum_{i=1}^{n_c}\sum_{j=0}^{N_t-1} M_{2i} c_i^2(x_j, u_j, t_j) + \sum_{i=1}^{n_\psi} M_{3i}\psi_i^2(x_{N_t-1}, t_{N_t-1}) \tag{2.2}$$

where M_{1i}, M_{2i} and M_{3i} are big numbers, and $c_i(.,.)$, $i = 1, 2, \ldots, n_c$, $d_i(.,.)$, $i = 1, 2, \ldots, n_d$ and $\psi_i(.,.)$, $i = 1, 2, \ldots, n_\psi$ are defined in (1.3), (1.4) and (1.6), respectively.
Mutation: The mutation operation of DE, which expands the search space, applies the vector differentials between the existing population members for determining both the degree and direction of perturbation applied to the individual. The mutation process at each generation begins by randomly selecting three individuals in the population and then generate a new solution as following:

$$\bar{U} = U^{(\alpha)} + F.(U^{(\beta)} - U^{(\gamma)}) \tag{2.3}$$

where α, β, $\gamma \in \{1, 2, \ldots, N_p\}$ are integer distinct random numbers and mutation factor $F \in [0, 2]$ is a real constant parameter which affects differential variation between two vectors, proposed by Storn and Price [32].
Crossover: Using a parent solution, called target matrix, $U^{(l)}, l \in \{1, 2, \ldots, N_p\}$, and previous perturbed individual, $\bar{U}$, the new individual matrix, of, called trial matrix, is generated by the following components:

$$(of)_{ij} = \begin{cases} (\bar{U})_{ij}, & r_j < CR \quad \text{or} \quad j = R \\ (U^{(l)})_{ij}, & r_j > CR \quad \text{and} \quad j \neq R \end{cases} \tag{2.4}$$

where $i = 1, 2, \ldots, m, j = 1, 2, \ldots, N_t$, R is a random chosen index in $\{1, 2, \ldots, N_t\}$, $CR \in [0, 1]$ is the crossover constant as a parameter, which increases the diversity of the individuals in the population, and r_j is a random number in $[0, 1]$.
Selection: The better individual between target matrix and trial matrix is replaced by the worst individual in the population.
Stopping criteria: The algorithm is terminate when the number of iteration is equal to $Maxiter$ or running time is equal to $CPUTIME$.
DE algorithm is given in Algorithm 1.

Algorithm 1 DE algorithm

{Initialization} Input the number of time steps, N_t, the size of population, N_p, the maximum number of iteration $Maxiter$, the mutation factor, F, the crossover constant, CR and the maximum running time $CPUTIME$.
Let $iter = 0$.
while stopping criteria are not satisfied **do**
Let $iter := iter + 1$.
for $i = 1$ to N_p **do**
{Mutation} Perturb the current individual to generate $\bar{U}$ by (2.3).
{Crossover} Generate the trial matrix, of, using (2.4).
{Selection} Select the better individual between target and trial matrices for next generation and replace it by the worst individual in the current population.
end for
end while
Return the best individual in the final population as an approximate solution of NOCP.

2.2. **GA.** GAs introduced by Holland in 1975, are a class of heuristics and probabilistic methods. These algorithms start with an initial population of solutions. This population is evaluated by using genetic operators that include selection, crossover and mutation. In the following, we introduce GA operators.

2.2.1. *GA operators.* Here, in MHGA, the underling GA has the following steps:
Initialization: The initial population is sequence random input matrices, similar to initialization in DE, from previous Section.
Evaluation: The fitness of each individual is calculated similar to (2.2).
Selection: To select two parents, we use a tournament operator with size 8 [10].
Crossover: When two parents $U^{(1)}$ and $U^{(2)}$ are selected, we use the following stages to construct an offspring:

(1) Select the following numbers

$$\lambda_1 \in [0,1], \lambda_2 \in [-\lambda_{max}, 0], \lambda_3 \in [1, 1+\lambda_{max}] \tag{2.5}$$

randomly, where λ_{max} is a random number in $[0,1]$.

(2) Let

$$of^k = \lambda_k U^{(1)} + (1-\lambda_k)U^{(2)}, \quad k = 1,2,3 \tag{2.6}$$

where $\lambda_k, k = 1,2,3$ is defined in (2.5). For $i = 1,2,\ldots,m$ and $j = 1,\ldots,N_t$, if $(of^k)_{ij} > u_{right,i}$, then let $(of^k)_{ij} = u_{right,i}$ and if $(of^k)_{ij} < u_{left,i}$, then let $(of^k)_{ij} = u_{left,i}$.

(3) Let $of = of^*$, where of^* is the best $of^i, i = 1,2,3$ constructed by (2.6).

Mutation: We apply a perturbation on each component of the offspring as follows:

$$(of)_{ij} = (of)_{ij} + r_{ij}.\alpha, \quad i = 1,2,\ldots,m, j = 1,2,\ldots,N_t \tag{2.7}$$

where r_{ij} is selected randomly in $\{-1,1\}$ and α is a random number in $[0,1]$. If $(of)_{ij} > u_{right,i}$, then let $(of)_{ij} = u_{right,i}$ and if $(of)_{ij} < u_{left,i}$, then let $(of)_{ij} = u_{left,i}$.

Replacement: Here, in the underling GA, we use a traditional replacement strategy. The replacement is done, if the new offspring has two properties: First, it is better than the worst person in the population. Second, it isn't very similar to a person in the population.

Stopping criteria: Underlying GA is terminated when at least one of the following conditions is occurred: over a specified number of generations, N_i, we don't have any improvement (the best individual is not changed), the maximum number of generations, N_g, is reached, or a predefined running time, $CPUTIME$, is achieved.

2.3. **MHGA.** In MHGA, GA uses a local search method to improve solutions. Here, we use SQP as a local search [6]. Using SQP as a local search in the hybrid metaheuristic is common for example see [23].

In the beginning of our MHGA, a less number of iterations for SQP was used. Then, when the promising regions of search space were found, we increase the number of iterations of SQP gradually. Using this approach, we may decrease the needed running time (in [5] the philosophy of this approach is discussed). Finally, we give our MHGA, in Algorithm 2.

3. Our Proposed Algorithm

Here, we give a two-phase algorithm, which is a direct approach, based on integrating DE with MHGA, for solving NOCPs, defined in Section 1. The main idea of our algorithm is to find promising regions of search space with a few number of time steps, using DE. Then, after finding good solutions, we

Algorithm 2 MHGA

{Initialization} Input the number of time steps N_t, the size of population N_p, the maximum number of generations without improvement N_i, the maximum number of generations N_g, the maximum running time $CPUTIME$, the mutation implementation probability P_m, the initial value of the maximum number of iterations in SQP, *sqpmaxiter* and an initial population.
{Evaluation} Evaluate the fitness of each individual by (2.2).
{Local search} Perform SQP on each individual of the population when the maximum number of iteration is *sqpmaxiter*.
while stopping criteria are not satisfied **do**
{Selection} Select two parents $U^{(1)}$ and $U^{(2)}$ by using an eight tournament from the population.
{Crossover} Construct a new offspring, of, from $U^{(1)}$ and $U^{(2)}$ by using (2.5) and (2.6).
{Mutation} Apply (2.7) on of with probability P_m.
{Local search} Perform SQP on of when the maximum number of iteration is *sqpmaxiter*.
{Replacement}
if replacement conditions are satisfied (see Section 2.2.1) **then** replace of with the worst individual of the population.
end if
Let: $sqpmaxiter := sqpmaxiter + 1$
end while
Return the best individual in the final population as an approximate solution of NOCP.

increase the number of time steps to improve the approximation of the optimal solution.
In the first phase, we perform DE (Algorithm 1) with a completely random initial population constructed by (2.1). Since the main goal in the first phase is to find the promising regions of the search space in a less running time, we use a few numbers of time steps, here. Also, to have a faster converged DE, the size of the population in the first phase is usually less than the size of the population in the second phase.
After phase 1, to maintain the property of individuals in the last population of the phase 1 and to increase the accurately of solutions, we add some additional time steps. Thus, we increase time steps from N_{t_1} in the phase 1 to N_{t_2} in the phase 2. The corresponding control input values of the new time steps are added to individuals. To use the information of the obtained solutions from phase 1 in the construction of the initial population of the phase 2, we use either linear or spline interpolations to estimate the value of the control inputs in the

new time steps in each individual of the last population of phase 1. Moreover, to maintain the diversity in the initial population of the phase 2, we add new random individuals to the population using (2.1). In the second phase, MHGA (Algorithm 2) starts with the new population.
Finally, our proposed algorithm is given in Algorithm 3.

Algorithm 3 Our proposed algorithm

{**Initialization**} Input $CPUTIME$ and let $CPUTTIME_1 := \frac{CPUTIME}{2}$
{**Phase 1**} Perform DE (Algorithm 1) with a random population and $N_{t_1},\ N_{p_1},\ Maxiter,\ F,\ CR$ and $CPUTIME_1$.
{**Construction of the initial population of the phase 2**} Increase time steps uniformly to N_{t_2} and estimate the corresponding control input values of the new time steps in each individual obtained from phase 1, using either linear or spline interpolations.
Create $N_{p_2} - N_{p_1}$ new different individuals with N_{t_2} time steps, randomly.
Let $CPUTIME_2 := CPUTIME$ - the running time of the **Phase 1**.
{ **Phase 2** } Perform MHGA (Algorithm 2) with the constructed random population and $N_{t_2},\ N_{p_2},\ N_i,\ N_g, CPUTIME_2, P_m$ and $sqpmaxiter$.

4. Numerical Experiments

In this Section, 26 well-known NOCPs are considered to show the feasibility and the efficiency of the proposed algorithm. Firstly, in subsection 4.1, a chemical process which described by an NOCP, is considered, as a real world problem. In subsection 4.2, we perform our proposed algorithm on a high dimensional problem. Next, in subsection 4.3, in order to compare the proposed algorithm with other methods, 24 benchmarks are solved. Finally, in subsection 4.4, a statistical approach is done for the proposed algorithm. To set the parameters of our proposed algorithm, we ran them with different values of parameters and selected the best of them. Also, because of the stochastic nature of the proposed algorithm, 100 different runs were made and the results, contain best and mean, are reported.
The numerical behaviour of the algorithms can be studied from three points of view: the performance index, J, the final state constraints, $\psi = [\psi_1, \ldots, \psi_{n_\psi}]^T$, defined in (1.6), and the running time, $Time$.
The algorithm was implemented in Matlab R2011a environment on a Notebook with Windows 7 Ultimate, CPU 2.53 GHz and 4.00 GB RAM. Furthermore, to implement SQP in our proposed algorithm, we used 'fmincon' in Matlab when the 'Algorithm' was set to 'SQP'.

Remark 4.1. We use the following abbreviations to show the used interpolation method in our proposed algorithm:

(1) LI: linear interpolation.
(2) SI: spline interpolation.

Remark 4.2. In Algorithm 3, we let $CPUTIME = 400$ s. Also, in the first phase of Algorithm 3, in DE algorithm (Algorithm 1), we let $Maxiter = 1000$, CR and F are selected randomly, separately in each problem. In second phase, in MHGA, we let $sqpmaxiter = 4$, $P_m = 0.8$. Besides, we use the composite Simpson method to approximate (1.1).

4.1. **TCCR problem** [34]**.** In this Section, the chemical process of Temperature Control for Consecutive Reaction, TCCR, is considered, which is an unconstrained two-state variable mathematical system. The objective is to obtain the optimal temperature profile that maximizes the yield of the temperature product B at the end of operation in a batch reactor, where the reaction $A \to B \to C$ is occurred. The state variables, x_1 and x_2, are the concentration of A and B, respectively, and the control variable u is the temperature. The mathematical model of TCCR problem is

$$\begin{aligned} \max \quad & J(u) = x_2(t_f) \\ s.t: \quad & \dot{x}_1 = -4000exp(\frac{-2500}{u}x_1^2), \\ & \dot{x}_2 = 4000exp(\frac{-2500}{u})x_1^2 - 620000exp(\frac{-5000}{u})x_2 \end{aligned}$$

where $x(t_0) = [1, 0]^T$ and $t_f = 1$. The problem solved by HIGA [34], which was more accurate than ACO [27] and iterative ACO [40]. The optimal value TCCR problem, for HIGA, was obtained as 0.61046. We used the proposed algorithm for the problem with the parameters as $N_{p_1} = 12,\ N_{p_2} = 15,\ N_{t_1} = 11,\ N_{t_2} = 15,\ N_g = 1000$ and $N_i = 600$. In addition, from the definition of model, u_{left} and u_{right} are 298 and 398, respectively. The best obtained values for J are 0.61067 and 0.61078, with the required running time 76.34 s and 87.65 s, for LI and SI respectively, which is better than HIGA. The optimal control signal and the state trajectories, for SI method, are shown in Figure 1.

4.2. **High dimensional problem.** In this Section to check the proposed algorithm for a special issue of high dimensional problem, we applied the algorithm on a time-invariant, linear-quadratic dynamic system [39], which $n = 1006$ (the number of state variables) and $m = 1$ (the number of state variables). The problem is

$$\begin{aligned} \min \quad & J(u) = \int_0^1 u^2(t)dt \\ s.t. \quad & \dot{x} = Ax(t) + Bu(t), \\ & x(0) = 1 \end{aligned}$$

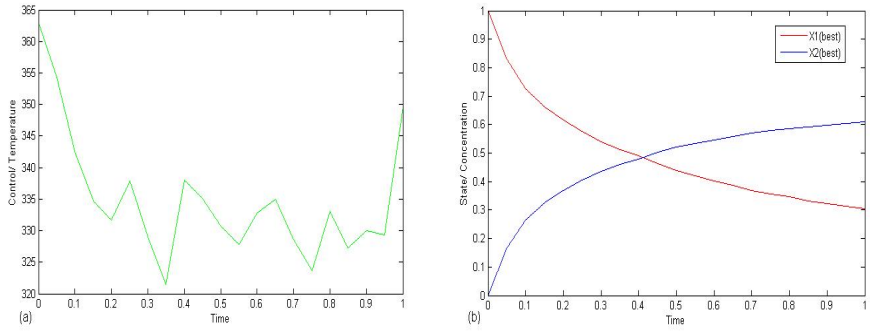

FIGURE 1. The optimal control signal, (a), and the optimal trajectories, x_1 and x_2, (b), for the TCCR problem, using SI method.

with the following matrices, where, $e_i \in \mathbb{R}^i$ is a vector each entry equal to 1:

$$A = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & A_3 & \\ & & & A_4 \end{bmatrix}, A_1 = \begin{bmatrix} -1 & 100 \\ -100 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -1 & 200 \\ -200 & -1 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} -1 & 400 \\ -400 & -1 \end{bmatrix}, A_4 = -diag\{1, 2, \ldots, 1000\}, \quad B = \begin{bmatrix} 10e_6 \\ e_{1000} \end{bmatrix}$$

The system was selected from [26], as a large dynamical system. The problem was solved by the proposed algorithm with the parameters as $N_{p_1} = 12$, $N_{p_2} = 15$, $N_{t_1} = 11$, $N_{t_2} = 51$, $N_g = 1000$ and $N_i = 600$. The best and the mean of performance index are 8.08×10^{-17} and 1.53×10^{-17}, respectively, using SI method. Also, these values for $Time$ are 253.73 s and 303.12 s, respectively.

4.3. **Comparison with some algorithms.** Here, 24 NOCPs, which are described in Appendix in terms of eqns (1.1)-(1.6), are considered. These NOCPs are selected with single control signal and multi control signals, which will be demonstrated in a general manner.

In Table 1, comparisons are made with some metaheuristic algorithms and some numerical methods. The numerical results for the NOCPs no. 1-3, in Appendix, are compared with a continuous GA, CGA, proposed in [1]. For the NOCPs no. 4-5, the results are compared with IPSO, proposed in [23], as metaheuristics. Similarly, for the NOCPs no. 6-9 the results are compared with some numerical methods. Moreover, for the NOCPs no. 10-24, in Appendix, the numerical results of LI and SI methods are compared with two numerical methods contain: SQP and SUMT, proposed in [12].

The notation φ_f, in Table 1, shows the norm of error in the final state constraints, i.e. $\phi_f = \|\psi\|_2$. To have a more careful comparison, we computed the gap between the value of the performance index of the algorithms and the best value of the obtained performance index. In other words, let J be the obtained value of the performance index of an algorithm. Now, similar to [38], we define the *Gap* as follows:

$$Gap(J) = |\frac{J - J^*}{J^*}| \tag{4.1}$$

where J^* is the best value of the obtained performance index. The applied parameters of the proposed algorithm are reported in Table 2, for each problem. Table 1, shows the best numerical results (the cost function, J, the norm of error in the final state constraints, φ_f, the gap between performance indexes, *Gap*, and the required running time, $Time$), in 100 independent runs (The mean numerical results are reported in Table 3). The best value of each column is shown in bold.

Table 1: The best numerical results for NOCPs described in Appendix, in 100 different runs.

Problem	**Algorithm**	J	φ_f	$Time$	Gap
VDPO	CGA [1]	1.7404	$\mathbf{2.67E-11}$	501.28	0.0346
	LI	**1.6822**	$2.71E-10$	**270.97**	**0**
	SI	**1.6822**	$5.30E-10$	316.38	**0**
CRP	CGA [1]	0.0163	$7.57E-10$	84.13	0.0724
	LI	**0.0152**	$\mathbf{3.59E-10}$	**34.71**	**0**
	SI	**0.0152**	$2.01E-9$	43.60	**0**
FFRP	CGA [1]	83.63	$4.65E-3$	1413	4.1370
	LI	**16.28**	$\mathbf{2.70E-5}$	151.61	**0**
	SI	16.42	$5.06E-4$	**142.33**	0.0086
CSTCR	IPSO[23]	0.1354	—	NR[a]	0.0360
	[2]	0.135	—	NR	0.0329
	[7]	0.1358	—	NR	0.0390
	LI	**0.1307**	—	**30.61**	**0**
	SI	**0.1307**	—	54.98	**0**
MSNIC	IPSO[23]	0.1727	—	NR	0.0135
	[15]	0.1816	—	NR	0.0657
	[22]	0.1769	—	NR	0.0381
	LI	**0.1704**	—	39.87	**0**
	SI	**0.1704**	—	**32.76**	**0**
No. 6	[41]	0.0266	—	NR	0.7272
	LI	**0.0154**	—	33.05	**0**
	SI	**0.0154**	—	**27.51**	**0**

Continued on next page

Table 1 – *Continued from previous page*

Problem	**Algorithm**	J	φ_f	$Time$	Gap
No. 7	[13]	-5.3898	—	NR	0.0439
	LI	$\mathbf{-5.6370}$	—	64.78	**0**
	SI	-5.6319	—	**59.35**	0.0009
No. 8	[13]	0.1713	—	NR	0.0012
	LI	**0.1711**	—	34.73	**0**
	SI	0.1725	—	**32.21**	0.0082
No. 9	HPM [9]	0.2353	$4.20-6$	NR	0.2042
	LI	**0.1954**	$\mathbf{9.09E-13}$	60.97	**0**
	SI	**0.1954**	$1.91E-12$	**47.92**	**0**
No. 10	SUMT [12]	$5.15E-6$	—	NR	$1.23E9$
	SQP [12]	$6.57E-6$	—	NR	$1.33E9$
	LI	$5.46E-15$	—	**14.02**	0.3
	SI	$\mathbf{4.20E-15}$	—	14.49	**0**
No. 11	SUMT [12]	1.7980	—	NR	0.1180
	SQP [12]	1.7950	—	NR	0.1162
	LI	**1.6082**	—	**74.01**	**0**
	SI	1.6083	—	91.16	$6.21E-5$
No. 12	SUMT [12]	0.1703	—	NR	0.2121
	SQP [12]	0.2163	—	NR	0.5395
	LI	0.1406	—	**27.48**	0.0007
	SI	**0.1405**	—	29.42	**0**
No. 13	SUMT [12]	**3.2500**	NR	NR	**0**
	SQP [12]	**3.2500**	NR	NR	**0**
	LI	**3.2500**	$2.40E-9$	**258.50**	**0**
	SI	**3.2500**	$\mathbf{1.41E-9}$	279.44	**0**
No. 14	SUMT [12]	-0.2490	NR	NR	0.0036
	SQP [12]	-0.2490	NR	NR	0.0036
	LI	-0.2498	$5.31E-10$	173.45	0.0004
	SI	$\mathbf{-0.2499}$	$\mathbf{8.85E-14}$	**132.86**	**0**
No. 15	SUMT [12]	0.0167	NR	NR	0.2462
	SQP [12]	0.0168	NR	NR	0.2537
	LI	**0.0134**	$4.04E-8$	72.52	**0**
	SI	**0.0134**	$\mathbf{9.48E-10}$	**50.04**	**0**
No. 16	SUMT [12]	3.7700	NR	NR	0.1406
	SQP [12]	3.7220	NR	NR	0.1261
	LI	**3.3052**	$\mathbf{7.05E-8}$	176.63	**0**
	SI	**3.3052**	$1.19E-7$	**165.36**	**0**
No. 17	SUMT [12]	$9.29E-4$	NR	NR	0.0153
	SQP [12]	$1.01E-3$	NR	NR	0.1038

Continued on next page

Table 1 – *Continued from previous page*

Problem	**Algorithm**	J	φ_f	$Time$	Gap
	LI	$\mathbf{9.55E-4}$	$1.20E-9$	112.58	**0**
	SI	$9.56E-4$	$\mathbf{2.73E-10}$	**93.92**	0.0120
No. 18	SUMT [12]	2.2080	NR	NR	0.0725
	SQP [12]	2.2120	NR	NR	0.0745
	LI	**2.0587**	$\mathbf{5.91E-11}$	69.24	**0**
	SI	**2.0587**	$7.13E-11$	**28.65**	**0**
No. 19	SUMT [12]	-8.8690	NR	NR	$2.25-5$
	SQP [12]	-8.8690	NR	NR	$2.25E-5$
	LI	$\mathbf{-8.8692}$	$\mathbf{8.80E-10}$	**45.89**	**0**
	SI	$\mathbf{-8.8692}$	$1.28E-9$	54.84	**0**
No. 20	SUMT [12]	0.0368	—	NR	0.0575
	SQP [12]	0.0368	—	NR	0.0575
	LI	**0.0348**	—	**85.50**	**0**
	SI	**0.0348**	—	86.70	**0**
No. 21	SUMT [12]	76.83	NR	NR	2.0344
	SQP [12]	77.52	NR	NR	2.0616
	LI	25.34	$\mathbf{1.20E-5}$	**398.98**	0.0008
	SI	**25.32**	$1.40E-5$	402.20	**0**
No. 22	SUMT [12]	0.3428	NR	NR	2.6160
	SQP [12]	0.3439	NR	NR	2.6276
	LI	**0.0948**	$\mathbf{8.82E-4}$	131.46	**0**
	SI	0.1179	$2.67E-3$	**128.43**	0.2436
No. 23	SUMT [12]	$5.27E-3$	NR	NR	7.5831
	SQP [12]	$5.19E-3$	NR	NR	7.4528
	LI	$1.59E-3$	**0.8356**	238.08	1.5896
	SI	$\mathbf{6.14E-4}$	0.8688	**220.61**	**0**
No. 24	SUMT [12]	$5.22E-3$	NR	NR	0.0419
	SQP [12]	$5.19E-3$	NR	NR	0.0359
	LI	$5.12E-3$	0.2885	261.86	0.0219
	SI	$\mathbf{5.01E-3}$	**0.2876**	**234.87**	**0**

[a] Not reported.

From Table 1, the associated values of *Gap* for LI and SI methods are less than other methods, for all test problems. It shows that the proposed algorithm provides robust solutions with respect to other methods. The mean values of *Gap* for the LI, SI, SQP and SUMT methods, on NOCPs no. 10-24, are 0.1276, 0.0170, $8.87e+7$ and $8.20e+7$, respectively. Thus, it is obvious that, the proposed algorithm gave much better solutions in comparison with SQP and SUMT. To compare with the CGA, on NOCPs no. 1-3, (as a global search algorithm), as shown in Table 1, we see that the mean values of the *Gap* for

CGA, LI and SI are 1.4147, 0 and 0.0029. Thus, we can see the LI and SI methods are 100 percent better than CGA from *Gap* perspective. The result shows that the proposed algorithm's estimations of global minimal is better than CGA's estimation. Therefore, based on these numerical study, we can conclude that the proposed algorithm outperforms CGA.
The mean values of violation of the norm of the final state constraints, ϕ_f, are 0.750 and 0.0773, for the LI and SI methods, respectively. Therefore, it is evident that our method is more robust. In addition, the mean value of ϕ_f for CGA, LI and SI methods are 0.0016, 9.0×10^{-6} and 1.69×10^{-4}, respectively, on NOCPs no. 1-3. Thus, we can say that the feasibility of the solutions given by the proposed algorithm and CGA are competitive. Therefore, it is seen that LI and SI methods could provide very suitable solutions with respect to the optimality and feasibility criteria.
The required running time, $Time$, of the proposed algorithm, is reported in Table 1, for each NOCP, separately. The mean values of the running time for LI and SI methods are 119.23 and 115.42, on all NOCPs, respectively. To compare with CGA, the mean of $Time$ for LI, SI and CGA are 152.43, 167.43 and 666.14, respectively. Therefore, the required running time for the proposed algorithm is less than CGA. The computational time of the proposed algorithm is given in details as shown Table 3. We will discuss it in Section 4.3.

4.4. **Comparing LI and SI.** In this Section, a statistical analysis, based on the one-way analysis of variance (ANOVA), used for comparing LI and SI methods. It is done based on the statistical software IBM SPSS version 21. For this purpose, initially, the absolute errors for J and ϕ_f are defined

$$E_J = |J - J^*|, \quad E_\psi = |\phi_f - \phi_f^*| \tag{4.2}$$

where J^* and $\phi_f^* = \|\psi^*\|_2$ are the best obtained solutions in different runs. In order to investigate the algorithm more precisely, we now present a new criterion, called factor, defined as follows:

$$K_\psi = E_J + E_\psi \tag{4.3}$$

where E_J and E_ψ are defined in (4.2). Note that K_ψ shows the summation of two important errors. Thus, based on K_ψ, we can study the behaviour of LI and SI methods on the quality and feasibility of given solutions, simultaneously. Table 3 shows the mean values of the numerical results, containing J, K_ψ and $Time$, for NOCPs in Appendix, in 100 different runs for LI and SI methods, separately. Table 4 summarizes the statistical data, including the test statistics (F) and p-values, of ANOVA tests. As shown in Table 4, the significant level or p-value, for J, K_ψ and $Time$ are equal to 0.754, 0.508 and 0.920, respectively, which are greater than 0.05. Thus, considering ANOVA, LI and SI methods have no significant difference on the outputs.

TABLE 2. The parameters of the proposed algorithm for the NOCPs in Appendix.

Problem	Parameters							
	N_{p_1}	N_{p_2}	N_{t_1}	N_{t_2}	N_g	N_i	u_{left}	u_{right}
1	12	15	91	171	8000	6500	-0.5	2
2	9	12	21	31	5000	3000	-1.5	2
3	9	15	11	15	5000	2600	-15	10
4	12	15	31	171	5000	3700	-7	7
5	12	15	41	51	3000	2200	-20	20
6	12	15	41	51	2000	1200	-1	1
7	12	15	51	191	2000	1200	-2	2
8	11	15	31	131	3000	2200	-5	15
9	9	12	31	51	2000	1000	0	1
10	9	15	21	51	5000	3600	0	2
11	9	15	51	71	5000	3600	-1	1
12	12	15	91	111	5000	3600	-20	20
13	9	15	31	91	5000	3600	-3	3
14	12	15	51	71	5000	3600	-1	1
15	11	15	31	71	5000	3600	-1.5	2
16	9	12	21	51	8000	6700	$-\pi$	π
17	11	15	9	11	5000	3600	-1	1
18	9	15	31	91	7000	5700	-3	3
19	9	15	31	131	7000	5700	-30	30
20	9	15	31	91	7000	5700	-3	3
21	9	12	11	15	5000	2600	-15	10
22	9	15	11	15	5000	2600	-2	2
23	9	12	15	21	5000	3600	$\begin{bmatrix}-2.8\\-0.8\end{bmatrix}$	$\begin{bmatrix}2.8\\0.7\end{bmatrix}$
24	9	15	11	21	6000	4600	$\begin{bmatrix}-2.8\\-0.8\end{bmatrix}$	$\begin{bmatrix}2.8\\0.7\end{bmatrix}$

5. CONCLUSIONS

In this paper, we gave a two-phase algorithm based on integrating DE algorithm with MHGA for solving the associated nonlinear programming problem of an NOCP. In the first phase, DE started with a completely random initial population where each individual, or solution, is a random matrix of control input values in time steps. After phase 1, to achieve more accurate solutions, we increased the number of time steps. The values of the associated new control inputs were estimated by linear or spline interpolations using the curves computed in the phase 1. In addition, to maintain the diversity in the population,

TABLE 3. The mean values of numerical results for NOCPs in Appendix, using LI and SI methods, in 100 different runs.

	SI			LI		
Problem	J	K_ψ	$Time$	J	K_ψ	$Time$
VDPO	1.6822	$2.58-7$	321.92	1.6822	$2.29E-7$	324.50
CRP	0.0152	$2.05E-8$	45.89	0.0152	$2.14E-8$	28.12
FFRP	32.30	15.88	154.89	35.03	18.75	159.32
CSTCR	0.1307	0.0488	41.25	0.1307	0.0471	31.50
MSNIC	0.1704	$1.98E-6$	32.17	0.1671	$2.52E-6$	31.78
No. 6	0.0221	0.0066	11.62	0.0219	0.0064	12.32
No. 7	-5.6132	0.0238	66.46	-5.6370	0.0187	62.18
No. 8	0.1720	0.0005	33.42	0.1712	0.0001	35.02
No. 9	0.1954	$1.55E-9$	79.74	0.1954	$2.05E-9$	84.37
No. 10	$1.38E-14$	$9.60E-15$	15.30	$1.30E-14$	$7.57E-15$	14.70
No. 11	1.6092	0.0010	86.33	1.6091	0.0008	83.78
No. 12	0.1431	0.0026	27.56	0.1429	0.0024	34.74
No. 13	2.8540	$2.28E-6$	272.97	2.8540	$2.42E-6$	280.24
No. 14	-0.2491	0.0007	127.17	-0.2490	0.0008	130.14
No. 15	0.0134	$5.23E-6$	69.11	0.0134	$6.35E-6$	70.82
No. 16	3.3362	0.0310	179.01	3.3944	0.0893	179.19
No. 17	$9.68E-4$	$1.22E-5$	88.70	$9.68E-4$	$1.31E-5$	93.41
No. 18	2.0587	$5.21E-7$	92.95	2.0587	$5.61E-7$	90.41
No. 19	-8.8692	$3.85RE-8$	50.64	-8.8692	$2.20E-8$	53.10
No. 20	0.0348	0.0001	101.61	0.0348	0.0002	99.50
No. 21	44.30	18.97	407.87	54.15	28.81	411.76
No. 22	0.3541	0.2389	124.65	0.3375	0.2462	129.23
No. 23	$5.29E-4$	0.0293	204.65	$5.46E-4$	0.0599	201.99
No. 24	$5.21E-3$	0.0052	225.16	$5.19E-3$	0.0083	227.63

TABLE 4. Summary of statistical data of ANOVA test for data in Table 3.

	J	K_ψ	$Time$
Test statistic (F)	0.099	0.445	0.01
p-value	0.754	0.508	0.920

some additional individuals were added randomly. Next, in the second phase, MHGA started by the new population constructed by the above procedure and tries to improve the obtained solutions at the end of phase 1. Our proposed MHGA combined a GA with a SQP, as a local search. In MHGA, to decrease the running time in the early iterations, a less number of iterations of SQP was used. Then, after finding the promising regions of the search space, we increased the number of iterations for SQP gradually.

We implemented our proposed algorithm on more than 25 well-known NOCPs. The numerical results showed the proposed algorithm could find almost better solutions than other proposed algorithms.

ACKNOWLEDGEMENTS

Authors are grateful to there anonymous referee and editor for their constructive comments.

References

1. Z.S. Abo-Hammour, A.Gh. Asasfeh, A.M. Al-Smadi, Othman M.K. Alsmadi, A novel continuous genetic algorithm for the solution of optimal control problems,*Optimal Control Applications and Methods* **32**(4), (2011), 414-432.
2. M.M. Ali, C. Storey, A. Törn, Application of stochastic global optimization algorithms to practical problems,*Journal of Optimization Theory and Applications*, **95**(3), (1997), 545-563.
3. M. Senthil Arumugam, G. Ramana Murthy, C. K. Loo, On the optimal control of the steel annealing processes as a two stage hybrid systems via PSO algorithms, *International Journal Bio-Inspired Computing*, **1**(3), (2009), 198-209.
4. M. Senthil Arumugam, M. V. C. Rao, On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems, *Application Soft Computing*, **8**(1), (2008), 324–336.
5. S. Babaie-Kafaki, R. Ghanbari, N. Mahdavi-Amiri, Two effective hybrid metaheuristic algorithms for minimization of multimodal functions, *International Journal Computing Mathematics*, **88**(11), (2011), 2415–2428.
6. J.J.F. Bonnans, J.C. Gilbert, C. Lemaréchal, C.A. Sagastizábal, *Numerical optimization: Theoretical and practical aspects*, Springer London, Limited, 2006.
7. I.L. Lopez Cruz, L.G. Van Willigenburg, G. Van Straten, Efficient differential evolution algorithms for multimodal optimal control problems, *Applied Soft Computing*, **3**(2), (2003), 97-122.
8. F. Dong, O. Petzold, W. Heinemann, R. Kasper, Time-optimal guidance control for an agricultural robot with orientation constraints, *Computers and Electronics in Agriculture*, **99**(0), (2013), 124-131.
9. S. Effati, H. Saberi Nik, Solving a class of linear and non-linear optimal control problems by homotopy perturbation method, *IMA Journal of Mathematical Control and Information*, **28**(4), (2011), 539-553.
10. A.P. Engelbrecht, *Computational intelligence: An introduction*, Wiley, 2007.
11. B.C. Fabien, Numerical solution of constrained optimal control problems with parameters,*Applied Mathematics and Computation*, **80**(1), (1996), 43-62.
12. B.C. Fabien, Some tools for the direct solution of optimal control problems, *Advances Engineering Software*, **29**(1), (1998), 45-61.
13. F. Ghomanjani, M.H Farahi, M Gachpazan, Bézier control points method to solve constrained quadratic optimal control of time varying linear systems, *Computational and Applied Mathematics*, **31**, (2012), 433-456.
14. A. Ghosh, S. Das, A. Chowdhury, R. Giri, An ecologically inspired direct search method for solving optimal control problems with Bézier parameterization, *Engineering Applications of Artificial Intelligence*, **24**(7), (2011), 1195-1203.
15. C.J. Goh, K.L. Teo, Control parametrization: A unified approach to optimal control problems with general constraints, *Automatica*, **24**(1), (1988), 3-18.
16. D.E. Kirk, *Optimal control theory: An introduction*, Dover Publications, 2004.
17. A. Vincent Antony Kumar, P. Balasubramaniam, Optimal control for linear system using genetic programming, *Optimal Control Applications and Methods* **30**(1), (2009), 47-60.
18. Moo Ho Lee, Ch. Han, K.S. Chang, Dynamic optimization of a continuous polymer reactor using a modified differential evolution algorithm, *Industrial and Engineering Chemistry Research*, **38**(12), (1999), 4825-4831.
19. R. Li, Y.J. Shi, The fuel optimal control problem of a hypersonic aircraft with periodic cruising mode, *Mathematical and Computer Modelling*, **55**(11-12), (2012), 2141-2150.

20. Y. Li, X. Zhang, Y. Chen, H. Zhou, Particle swarm optimization for time-optimal control design, *Journal of Control Theory and Applications*, **10**(3), (2012), 365–370.
21. J. Antonio Delgado San Martn, M. Nicols Cruz Bournazou, P. Neubauer, T. Barz, Mixed integer optimal control of an intermittently aerated sequencing batch reactor for wastewater treatment, *Computers & Chemical Engineering*, **71**(0), (2014), 298-306.
22. W. Mekarapiruk, R. Luus, Optimal control of inequality state constrained systems, *Industrial and Engineering Chemistry Research*, **36**(5), (1997), 1686-1694.
23. H. Modares, M.B. Naghibi-Sistani, Solving nonlinear optimal control problems using a hybrid IPSO - SQP algorithm, *Engineering Applications of Artificial Intelligence*, **24**(3), (2011), 476-484.
24. H. Moradi, Gh. Vossoughi, H. Salarieh, Optimal robust control of drug delivery in cancer chemotherapy: A comparison between three control approaches, *Computer Methods and Programs in Biomedicine*, **112**(1), (2013), 69-83.
25. G.M. Ostrovsky, N.N. Ziyatdinov, T.V. Lapteva, Optimal design of chemical processes with chance constraints, *Computers & Chemical Engineering*, **59**(0), (2013), 74-88.
26. Th. Penzl, Algorithms for model reduction of large dynamical systems, *Linear Algebra and its Applications*, **415**(2-3), (2006), 322-343.
27. J. Rajesh, K. Gupta, H.Sh. Kusumakar, V.K. Jayaraman, B.D. Kulkarni, Dynamic optimization of chemical processes using ant colony framework, *Computers & Chemistry*, **25**(6), (2001), 583-595.
28. A.F. Ribeiro, M.C.M. Guedes, G.V. Smirnov, S. Vilela, On the optimal control of a cascade of hydro-electric power stations, *Electric Power Systems Research*, **88**(0), (2012), 121-129.
29. S. Sedaghat, Y. Ordokhani, Stability and numerical solution of time variant linear systems with delay in both the state and control, *Iranian Journal of Mathematical Sciences and Informatics*, **7**(1), (2012), 43-57.
30. X. H. Shi, L. M. Wan, H. P. Lee, X. W. Yang, L. M. Wang, Y. C. Liang, An improved genetic algorithm with variable population-size and a PSO-GA based hybrid evolutionary algorithm, *Machine Learning and Cybernetics, 2003 International Conference on*, **3**, (2003), 1735-1740.
31. Y. C. Sim, S. B. Leng, V. Subramaniam, A combined genetic algorithms-shooting method approach to solving optimal control problems, *International Journal of Systems Science*, **31**(1), (2000), 83-89.
32. R. Storn, K. Price, Minimizing the real functions of the icec'96 contest by differential evolution, *Evolutionary Computation, Proceedings of IEEE International Conference on*, (1996), 842-844.
33. R. Storn, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, **11**(4), (1997), 341-359.
34. F. SUN, W. DU, R. QI, F. Qian, W. Zhong, A hybrid improved genetic algorithm and its application in dynamic optimization problems of chemical processes, *Chinese Journal of Chemical Engineering*, **21**(2), (2013), 144-154.
35. K.L. Teo, C.J. Goh, K.H. Wong, A unified computational approach to optimal control problems, *Pitman monographs and surveys in pure and applied mathematics*, Longman Scientific and Technical, 1991.
36. J. M. van Ast, R. Babuška, B. De Schutter, Novel ant colony optimization approach to optimal control, *International Journal of Intelligent Computing and Cybernetics*, **2**(3), (2009), 414-434.

37. F.Sh. Wang, J.P. Chiou, Optimal control and optimal time location problems of differential-algebraic systems by differential evolution, *Industrial and Engineering Chemistry Research*, **36**(12), (1997), 5348-5357.
38. M. Yaghini, M. Karimi, M. Rahbar, A hybrid metaheuristic approach for the capacitated p-median problem, *Applied Soft Computing*, **13**(9), (2013), 3922-3930.
39. M. Yarahmadi, S. M. Karbassi, Design of robust controller by neuro-fuzzy system in a prescribed region via state feedback, *Iranian Journal of Mathematical Sciences and Informatics*, **4**(1), (2009), 1-16.
40. B. Zhang, D. Chen, W. Zhao, Iterative ant-colony algorithm and its application to dynamic optimization of chemical process, *Computers & Chemical Engineering* **29**(10), (2005), 2078-2086.
41. W. Zhang, H.p. Ma, Chebyshev-legendre method for discretizing optimal control problems, *Journal of Shanghai University*, **13**(2), (2009), 113-118.

Appendix

In this Section 24 NOCPs, as test problems, are considered. These problems are represented as abbreviations form, eqns (1.1)-(1.6).

(1) [1, 11] (Van Der Pol oscillator problem (VDPO)) $g = (x_1^2 + x_2^2 + u^2)/2, t_0 = 0, t_f = 5, f = [x_2, -x_2 + (1 - x_1^2)x_2 + u]^T, x_0 = [1, 0]^T, \psi = x_1 - x_2 + 1$.

(2) [1, 16] (Chemical reactor problem (CRP)) $g = (x_1^2+x_2^2+0.1u^2)/2, t_0 = 0, t_f = 0.78, f = [x_1-2(x_1+0.25)+(x_2+0.5)exp(25x_1/(x_1+2))-(x_1+0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

(3) [1, 12] (Free floating robot problem (FFRP)) $g = (u_1^2 + u_2^2 + u_3^2 + u_4^2)/2, t_0 = 0, t_f = 5, f = [x_2, ((u_1+u_2)\cos x_5-(u_2+u_4)\sin x_5)/M,\ x_4, ((u_1+u_3)\sin x_5+(u_2+u_4)\cos x_5)/M, x_6, (D(u_1+u_3)-L_e(u_2+u_4))/I]^T, x_0 = [0, 0, 0, 0, 0, 0]^T, \psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5, x_6]^T, M = 10, D = 5, I = 12, L_e = 5$.

(4) [23] (Continuous stirred-tank chemical reactor (CSTCR)) $g = x_1^2+x_2^2+0.1u^2, t_0 = 0, t_f = 0.78, f = [-(2+u)(x_1+0.25)+(x_2+0.5)exp(25x_1/(x_1+2)), 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.09, 0.09]^T$.

(5) [23] (Mathematical system with nonlinear inequality constraint (MSNIC)) $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u, x_1^2 + x_2^2 + 0.005u^2]^T, d = [-(20 - u)(20 + u), x_2 + 0.5 - 8(t - 0.5)^2]^T, x_0 = [0, -1, 0]^T$.

(6) [41] $g = 0.39(x_1^2 + x_2^2 + 0.1u^2), t_0 = -1, t_f = 1, f = [0.39(-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u), 0.39(0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2)))]^T, x_0 = [0.05, 0]^T$.

(7) [13] $g = 2x_1, t_0 = 0, t_f = 3, f = [x_2, u]^T, d = [-(2 - u)(2 + u), -6 - x_1]^T, x_0 = [2, 0]^T$.

(8) [13] $g = x_1^2 + x_2^2 + 0.005u^2, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u]^T, d = x_2 + 0.5 - 8(t - 0.5)^2, x_0 = [0, -1]^T$.

(9) [9] $g = u^2, t_0 = 0, t_f = 1, f = \frac{1}{2}x^2 \sin x + u,\ x_0 = 0, \psi = x - 0.5$.

(10) [12] $g = x^2 \cos^2 u, t_0 = 0, t_f = \pi, f = \sin u/2, x_0 = \pi/2$.

(11) [12] $g = (x_1^2 + x_2^2 + u^2)/2, t_0 = 0, t_f = 5, f = [x_2, -x_1 + (1 - x_1^2)x_2 + u]^T, d = -(x_2 + 0.25), x_0 = [1, 0]^T$.

(12) [12] $g = x_1^2 + x_2^2 + 0.005u^2, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u]^T, d = [-(20 - u)(20 + u), 0.5 + x_2 - (8(t - 0.5)^2]^T, x_0 = [0, -1]^T$.

(13) [12] $g = u^2/2, t_0 = 0, t_f = 2, f = [x_2, u]^T, x_0 = [1, 1]^T, \psi = [x_1, x_2]^T$.

(14) [12] $g = -x_2, t_0 = 0, t_f = 1, f = [x_2, u]^T, d = -(1 - u)(1 + u), x_0 = [0, 0]^T, \psi = x_2$.

(15) [12] $g = (x_1^2+x_2^2+0.1u^2)/2, t_0 = 0, t_f = 0.78, f = [-2(x_1+0.25)+(x_2+0.5)exp(25x_1/(x_1+2))-(x_1+0.25)u, 0.5-x_2-(x_2+0.5)exp(25x_1/(x_1+2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

(16) [12] $g = u^2/2, t_0 = 0, t_f = 10, f = [\cos u - x_2, \sin u]^T, d = -(\pi - u)(\pi + u), x_0 = [3.66, -1.86]^T, \psi = [x_1, x_2]^T$.

(17) [12] $g = (x_1^2 + x_2^2)/2, t_0 = 0, t_f = 0.78, f = [-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1+2))-(x_1+0.25)u, 0.5-x_2-(x_2+0.5)exp(25x_1/(x_1+2))]^T, d = -(1 - u)(1 + u), x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

(18) [12] $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, u, u^2/2]^T, d = x_1 - 1.9, x_0 = [0, 0, 0]^T, \psi = [x_1, x_2 + 1]^T$.

(19) [12] $\phi = -x_3, t_0 = 0, t_f = 5, f = [x_2, -2 + u/x_3, -0.01u]^T, d = -(30 - u)(30 + u), x_0 = [10, -2, 10]^T, \psi = [x_1, x_2]^T$.

(20) [12] $\phi = (x_1 - 1)^2 + x_2^2 + x_3^2,\ g = \frac{1}{2}u^2,\ t_0 = 0, t_f = 5,\ f = [x_3 \cos u, x_3 \sin u, \sin u]^T, x_0 = [0, 0, 0]^T$.

(21) [12] $g = (u_1^2+u_2^2+u_3^2+u_4^2)/2, t_0 = 0, t_f = 5, f = [x_2, ((u_1+u_3)\cos x_5 - (u_2+u_4)\sin x_5)/M, x_4, ((u_1+u_3)\sin x_5+(u_2+u_4)\cos x_5)/M, x_6, (D(u_1+u_3) - L_e(u_2 + u_4))/I]^T, x_0 = [0, 0, 0, 0, 0, 0]^T, \psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5 - \pi/4, x_6]^T, M = 10, D = 5, I = 12, L_e = 5$.

(22) [12] $g = 4.5(x_3^2+x_6^2)+0.5(u_1^2+u_2^2), t_0 = 0, t_f = 1, f = [9x_4, 9x_5, 9x_6, 9(u_1+17.25x_3), 9u_2, -9(u_1-27.0756x_3+2x_5x_6)/x_2]^T, x_0 = [0, 22, 0, 0, -1, 0]^T, \psi = [x_1 - 10, x_2 - 14, x_3, x_4 - 2.5, x_5, x_6]^T$.

(23) [12] Same as problem 21 with $d = [-(2.83374-u_1)(2.83374+u_1), -(0.71265-u_2)(0.80865 + u_2)]^T$.

(24) [12] Same as problem 21 with $d = [-(2.83374-u_1)(2.83374+u_1), -(0.71265-u_2)(0.80865 + u_2), -(2.5 - x_4)(2.5 + x_4), -(1 - x_5)(1 + x_5)]^T$.