SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

December 2024, Vol. 67, Iss. 12, 222208:1–222208:9
https://doi.org/10.1007/s11432-024-4207-5

Optimization methods rooted in optimal control

Huanshui ZHANG[1,2*], Hongxia WANG[1], Yeming XU[1] & Ziyuan GUO[1]

[1]*School of Electrical and Automation Engineering, Shandong University of Science and Technology, Qingdao* 266590, *China;*
[2]*School of Control Science and Engineering, Shandong University, Jinan* 250061, *China*

Received 10 January 2024/Revised 6 September 2024/Accepted 30 October 2024/Published online 11 December 2024

Abstract In the paper, we investigate the optimization problem (OP) by applying the optimal control method. The optimization problem is reformulated as an optimal control problem (OCP) where the controller (iteration updating) is designed to minimize the sum of costs in the future time instant, which thus theoretically generates the "optimal algorithm" (fastest and most stable). By adopting the maximum principle and linearization with Taylor expansion, new algorithms are proposed. It is shown that the proposed algorithms have a superlinear convergence rate and thus converge more rapidly than the gradient descent; meanwhile, they are superior to Newton's method because they are not divergent in general and can be applied in the case of a singular or indefinite Hessian matrix. More importantly, the OCP method contains the gradient descent and the Newton's method as special cases, which discovers the theoretical basis of gradient descent and Newton's method and reveals how far these algorithms are from the optimal algorithm. The merits of the proposed optimization algorithm are illustrated by numerical experiments.

Keywords optimal control, optimization methods, optimization algorithm, maximum principle, superlinear convergence

1 Introduction

The research history of optimization problems (OPs) is long and rich, spanning several centuries and evolving with the development of mathematics, engineering, operations research, and computer science. This originates from that OPs are widespread in the modeling of real-world systems for a very broad range of applications. Such applications include economies of scale, fixed charges, finance, allocation and location problems, structural optimization, engineering design, network and transportation problems, chip design and database problems, mechanical design, chemical engineering design and control, molecular biology, and several other combinatorial OPs such as integer programming and related graph problems. With the development and implementation of practical optimization algorithms, more and more scientists in diverse disciplines have been using optimization techniques to solve problems [1–4].

Predominant optimization methods, such as the gradient descent [5] and Newton's methods [6], are proposed based on the function approximation. Despite being powerful, favorable, and widely used optimization algorithms, they also have limitations and potential drawbacks. The learning rate is a crucial hyperparameter in gradient descent [7]. Gradient descent's performance is susceptible to the learning rate (if the learning rate is too small, the algorithm may converge slowly, while if it is too large, the algorithm may overshoot the minimum and fail to converge [8]) while the hyperparameters tuning is challenging [9]. Newton's method may diverge if the Hessian matrix is singular or the algorithm encounters numerical instability (when the Hessian matrix is ill-conditioned). Both gradient descent and Newton's methods are sensitive to the choice of the initial value. Suppose the initial guess is far from the actual root or minimum. In this case, Newton's method might converge to a different solution or fail to converge altogether and gradient descent may get stuck in local minima or saddle points. To mitigate some of the aforementioned issues, a body of variations and enhancements, such as stochastic gradient descent [10], momentum [11], adaptive learning rate methods [12–14], quasi-Newton method [15], inexact

* Corresponding author (email: hszhang@sdu.edu.cn)

Newton method [16], and modified Newton method [17] have been developed. However, it still seems somewhat blind to tune parameters in diverse methods.

The paper attempts to propose solving OPs and understanding the existing optimization methods from the optimal control view, while primarily facing two difficulties. The first difficulty is to break through the structure of classical methods including gradient descent and Newton's method because it is known that gradient descent and Newton's method have been well-recognized for hundreds of years. Moreover, we aim to derive an algorithm possessing several advantages of classical optimization algorithms. More precisely, the algorithm should hold not only the stability of gradient descent but also the fast convergence of Newton's method. The second difficulty lies in implementing the optimal control method because it is costly to calculate. To address this issue, we employ first-order Taylor expansions and introduce some mild assumptions.

It should be emphasized that in this paper, we do not apply the results of optimal control mechanically, but propose new optimization algorithms based on the optimal control and make a convergence analysis. Every parameter or variable in the optimal control problem (OCP) has a clear physical meaning whereby it is easy to tune the parameters to alter the interested variables [18]. Accordingly, the convergence rate of algorithms based on OCP can be altered by adjusting those parameters inherited from OCP. The algorithms feature converging more rapidly than gradient descent, meanwhile, they are superior to Newton's method because they can be applied in the case of a singular Hessian matrix. We also point out that the convergence rate of the proposed algorithms can be accelerated by decreasing the magnitude of the control weight matrix inherited from OCP.

The remainder is arranged as follows. In Section 2, we propose an update relation, an optimization method, for OP by the solution to OCP. We explore the relationship between the existing (gradient descent and Newton's method) and proposed methods in Section 3. Section 4 is devoted to the convergence analysis of explicitly implementing the proposed method. Some conclusion is achieved in Section 5.

Notation. Throughout the paper, the superscript T stands for matrix transposition; R^n denotes the n-dimensional real vector space; I_n is n-dimensional unit matrix. For matrix M, $M > 0$ means that it is positive definite. Let $f'(x)$ and $f''(x)$ denote the gradient and the Hessian matrix of $f(x)$; $|\cdot|$ and $||\cdot||$ denote appropriate vector norm and matrix norm, respectively.

2 Resolving OP by using optimal control method

Assume that $f(x) : R^n \mapsto R^1$ is twice continuously differentiable. In most practical settings, one would almost surely need to resort to numerical techniques to address the OP:

$$\min_x f(x). \tag{1}$$

The traditional numerical method solving (1), for instance, gradient descent or Newton's method, directly finds zeros of $f'(x)$ by an update recursion

$$x_{k+1} = \Phi(x_k) \tag{2}$$

with $\Phi(\cdot)$ being a designed function.

Instead, we will propose an updated relation for (1) with the aid of the following OCP:

$$\min_u \sum_{k=0}^{N} \left[f(x_k) + \frac{1}{2} u_k^{\mathrm{T}} R u_k \right] + f(x_{N+1}), \tag{3}$$

$$\text{subject to } x_{k+1} = x_k + u_k, \tag{4}$$

where $x_k \in R^n$ and $u_k \in R^n$ are the state and control of system (4), respectively, integer $N > 0$ is the control terminal time, positive definite matrix R is the control weight matrix, and Eq. (3) is the cost functional of OCP, closely related to OP (1).

Remark 1. It can be seen from (3) that the optimization algorithm based on OCP will be one of the fastest algorithms because every u_k in (4) is chosen to minimize $\sum_{i=k+1}^{N+1} f(x_i)$.

Remark 2. Because the control energy $\sum_{k=0}^{N} u_k^{\mathrm{T}} R u_k$ is involved in the cost function (3), it can be concluded that the algorithm minimizing (3) would be stable and always convergent.

Lemma 1. The optimal control strategy of OCP (3) and (4) admits

$$u_k = -R^{-1} \sum_{i=k+1}^{N+1} f'(x_i), \tag{5}$$

and the update iteration for OP (1) is given as Algorithm I.

Algorithm I.

$$x_{k+1} = x_k - R^{-1} \sum_{i=k+1}^{N+1} f'(x_i). \tag{6}$$

Proof. The derivation of (5) can refer to the proof of Theorem 1 in [19]. From (4) and (5), it is immediate to get (6). The proof is completed.

Remark 3. Let $N = k$, Eq. (6) can be reduced as $x_{k+1} = x_k - R^{-1}f'(x_{k+1})$. Let $f'(x_k) \approx f'(x_{k+1})$, then we can obtain the gradient descent. Additionally, according to the first-order Taylar expansion of $f'(x_{k+1})$ at x_k, we have $x_{k+1} = x_k - R^{-1}\big[f'(x_k) + f''(x_k)(x_{k+1} - x_k)\big]$, which can be rearranged as $x_{k+1} = x_k - (R + f''(x_k))^{-1}f'(x_k)$. In particular, it can be recovered as Newton's method when $R = 0$.

Although the relation (6) is implicit, some explicit approximation implementations are possible. We will give an explicit approximation recursion later.

It should be stressed that the proposal of a new update scheme is not our ultimate goal. We are also interested in revealing the relationship between existing methods and our methods.

3 The proposed algorithm and convergence analysis

In spite of the special nonlinear form of (6), it can be implemented by solving for (x_k, u_k, λ_k) such that there hold nonlinear Euler-Lagrange equations consisting of (4) and

$$Ru_k + \lambda_k = 0, \tag{7}$$

$$\lambda_{k-1} = \lambda_k + f'(x_k),\ \lambda_N = f'(x_{N+1}) \tag{8}$$

with the aid of the method of successive approximations (MSA) [19]. In order to simplify the calculation, a simple algorithm will be presented in this section by Taylor expansion. Firstly, we have Lemma 2.

Lemma 2. The update relation (6) can be approximated by

$$x_{k+1} = x_k - g_k, \tag{9}$$

$$g_k = (R + f''(x_k))^{-1}\left(f'(x_k) + \sum_{i=k+1}^{N} (f'(x_i) - f''(x_i)g_i)\right),\ g_N = (R + f''(x_N))^{-1}f'(x_N). \tag{10}$$

Proof. It can be derived by using induction over k and first-order Taylor expansion of $f'(x_i)$ at point x_{i-1} for all $i > k$ in (6).

First, let $k = N$. Make the first-order Taylor expansion of $f'(x_{N+1})$ at point x_N in (6) with $k = N$. One has

$$\begin{aligned} x_{N+1} =& x_N - R^{-1}f'(x_{N+1}) \\ =& x_N - R^{-1}[f'(x_N) + f''(x_N)(x_{N+1} - x_N)]. \end{aligned} \tag{11}$$

Rearranging the above equality yields

$$x_{N+1} = x_N - (R + f''(x_N))^{-1}f'(x_N), \tag{12}$$

which means that Eq. (6) can be approximated by (9) for $k = N$.

Second, assume that Eq. (9) with (10) can approximate (6) for any $k = s + 1, \ldots, N$.

Third, we will show that Eq. (9) with (10) can approximate (6) for any $k = s$. It follows from (6) and Taylor expansions that

$$x_{s+1} = x_s - R^{-1} \sum_{i=s+1}^{N+1} [f'(x_{i-1}) + f''(x_{i-1})(x_i - x_{i-1})]$$

$$=x_s - R^{-1}[f'(x_s) + f''(x_s)(x_{s+1} - x_s)] - R^{-1}\sum_{i=s+2}^{N+1}[f'(x_{i-1}) - f''(x_{i-1})g_{i-1}], \tag{13}$$

where we have used the assumption that Eq. (9) with (10) can approximate (6) for any $k = s+1, \ldots, N$ in the second equality and the relation $x_i - x_{i-1} = g_{i-1}$.

According to (13), it is easy to get that

$$x_{s+1} = x_s - (R + f''(x_s))^{-1}\left[f'(x_s) + \sum_{i=s+2}^{N+1}(f'(x_{i-1}) - f''(x_{i-1})g_{i-1})\right]. \tag{14}$$

It means Eq. (9) with (10) can approximate (6) for any $k = 0, \ldots, N$. The proof is completed.

It is noted that Eq. (9) with (10) is still implicit, which is difficult to implement in practice. To facilitate iteration, we rewrite (9) and (10) by replacing all x_i for $i > k$ in the right-hand side of (9) with x_k and obtain the following.

Algorithm II.

$$x_{k+1} = x_k - \bar{g}_k(x_k),\ k = 0, \ldots, N, \tag{15}$$

$$\bar{g}_k(x_k) = (R + f''(x_k))^{-1}(f'(x_k) + R\bar{g}_{k+1}(x_k)),\quad \bar{g}_N(x_k) = (R + f''(x_k))^{-1}f'(x_k). \tag{16}$$

Obviously, algorithm (15) with (16) is explicit and implementable. More importantly, it will be shown that the above algorithm holds superlinear convergence, which is proved in Lemma 3.

Lemma 3. Assume $f'(x_*) = 0$ and $f''(x_*) > 0$. For some integer $k_0 > 0$, $\{x_k\}$ generated by (15) and (16) admits

$$|x_{k+1} - x_*| \leqslant ||(R + f''(x_*))^{-1}R||^{N+1-k}|x_k - x_*|,\ k > k_0, \tag{17}$$

i.e., Algorithm II in (15) and (16) is superlinearly convergent as $N \to +\infty$.

To make the proof of Lemma 3 more clear, we prepare the following.

Lemma 4. Assume $f'(x_*) = 0$ and $f''(x_*) > 0$. Consider function sequence $\bar{g}_k(x)$ given by

$$\bar{g}_k(x) = (R + f''(x))^{-1}[f'(x) + R\bar{g}_{k+1}(x)],\ \bar{g}_N(x) = (R + f''(x))^{-1}f'(x). \tag{18}$$

Then there holds $\bar{g}_k'(x_*) = I_n - [(R + f''(x_*))^{-1}R]^{N+1-k}$.

Proof. According to (18),

$$\bar{g}_k'(x) = (R + f''(x))^{-1}(f''(x) + R\bar{g}_{k+1}'(x)) - (R + f''(x))^{-1}(\bar{g}_{k+1}^{\mathrm{T}}(x) \otimes I_n)f'''(x), \tag{19}$$

$$\bar{g}_N'(x) = (R + f''(x))^{-1}f''(x) - (R + f''(x))^{-1}(\bar{g}_N^{\mathrm{T}}(x) \otimes I_n)f'''(x), \tag{20}$$

where $\otimes$ denote the Kronecker product, $f'''(x) = \frac{\mathrm{d}(\mathrm{vec}(f''(x)))}{\mathrm{d}x^{\mathrm{T}}}$ and $f''(x)$ is vectorized as $\mathrm{vec}(f''(x))$.

Eq. (18) implies that every $\bar{g}_i(x)$ contains $f'(x)$ as a factor. Accordingly, if $f'(x_*) = 0$, then

$$\bar{g}_i(x_*) = 0,\quad i = 0, 1, \ldots, N, \tag{21}$$

which together with (19) and (20) yields

$$\bar{g}_k'(x_*) = (R + f''(x_*))^{-1}(f''(x_*) + R\bar{g}_{k+1}'(x_*)), \tag{22}$$

$$\bar{g}_N'(x_*) = (R + f''(x_*))^{-1}f''(x_*). \tag{23}$$

Based on (22) and (23), $\bar{g}_i'(x_*) = I_n - [(R + f''(x_*))^{-1}R]^{N+1-i}$ can be acquired by using backward induction technique.

First, in the case of $i = N$, it is direct that

$$\bar{g}_N'(x_*) = (R + f''(x_*))^{-1}f''(x_*) = I_n - (R + f''(x_*))^{-1}R. \tag{24}$$

Second, assume for all $i \geqslant k+1$, there always holds

$$\bar{g}_i'(x_*) = I_n - [(R + f''(x_*))^{-1}R]^{N+1-i}. \tag{25}$$

Third, we will show that in the case of $i = k$, Eq. (25) also holds.

$$\begin{aligned}\bar{g}_k'(x_*) =&(R + f''(x_*))^{-1}(f''(x_*) + R\bar{g}_{k+1}'(x_*))\\ =&(R + f''(x_*))^{-1}[f''(x_*) + R(I_n - ((R + f''(x_*))^{-1}R)^{N-k})]\\ =&I_n - [(R + f''(x_*))^{-1}R]^{N+1-k}.\end{aligned} \tag{26}$$

The proof is completed.

With the preparation in Lemma 4, we turn to prove Lemma 3.

Proof. Note that $f'(x)$ is a factor of every $\bar{g}_k(x)$. Accordingly, $f'(x_*) = 0$ always results in $\bar{g}_k(x_*) = 0$. From Lemma 4 that $\bar{g}_k'(x_*) = I_n - [(R + f''(x_*))^{-1}R]^{N+1-k}$. Now it can be known that for x_k generated by (15) and (16), there holds

$$\begin{aligned}&x_{k+1} - x_*\\ &= x_k - x_* - \bar{g}_k(x_k)\\ &= x_k - x_* - [\bar{g}_k(x_*) + \bar{g}_k'(x_*)(x_k - x_*) + o(|x_k - x_*|)]\\ &\approx x_k - x_* - \bar{g}_k'(x_*)(x_k - x_*)\\ &= (I_n - \bar{g}_k'(x_*))(x_k - x_*)\\ &= [(R + f''(x_*))^{-1}R]^{N+1-k}(x_k - x_*),\end{aligned} \tag{27}$$

where the third equality has used the first-order Taylor expansion of $\bar{g}_k(x_k)$ at x_*. From the fact that $||(R + f''(x_*))^{-1}R|| < 1$ when $f''(x_*) > 0$, the proof of Lemma 3 is completed.

Corollary 1. Assume $f'(x_*) = 0$, $f''(x_*) > 0$. In the case of that $f(x)$ is quartic differentiable and x is scalar. $\{x_k\}$ generated by (15) and (16) admits

$$|x_{k+1} - x_*| \leqslant ||(R + f''(x_*))^{-1}R||^{N+1-k}|x_k - x_*| + ||\bar{g}_k''(x_*)||\,|x_k - x_*|^2 \tag{28}$$

with $||(R + f''(x_*))^{-1}R|| < 1$ and $||\bar{g}_k''(x_*)|| < |f'''(x_*)/f''(x_*)|$, i.e., Algorithm II may converge more rapidly than Newton's method as $N \to +\infty$.

Proof. Because of $f''(x_*) > 0$ and $R > 0$, $||(R + f''(x_*))^{-1}R|| < 1$. From (15), we have

$$\begin{aligned}&|x_{k+1} - x_*|\\ &= |x_k - x_* - \bar{g}_k(x_k)|\\ &= |x_k - x_* - [\bar{g}_k(x_*) + \bar{g}_k'(x_*)(x_k - x_*) + \bar{g}_k''(x_*)(x_k - x_*)^2 + o(|x_k - x_*|^2)]|\\ &\leqslant ||(I_n - \bar{g}_k'(x_*))||\,|x_k - x_*| + ||\bar{g}_k''(x_*)||\,|x_k - x_*|^2 + o(|x_k - x_*|^2)\\ &= ||((R + f''(x_*))^{-1}R)^{N+1-k}||\,|x_k - x_*| + ||\bar{g}_k''(x_*)||\,|x_k - x_*|^2 + o(|x_k - x_*|^2)\\ &\leqslant ||(R + f''(x_*))^{-1}R||^{N+1-k}|x_k - x_*| + ||\bar{g}_k''(x_*)||\,|x_k - x_*|^2 + o(|x_k - x_*|^2).\end{aligned} \tag{29}$$

In the above, the second equality has used the second-order Taylor expansion of $\bar{g}_k(x_k)$ at x_*.

Next, we derive $\bar{g}_k''(x_*)$. It is evident from (18) that $\bar{g}_k'(x)$ admits

$$\bar{g}_k'(x) = (R + f''(x))^{-1}(f''(x) + R\bar{g}_{k+1}'(x)) - (R + f''(x))^{-2}f'''(x)(f'(x) + R\bar{g}_{k+1}(x)), \tag{30}$$

$$\bar{g}_N'(x) = (R + f''(x))^{-1}f''(x) - (R + f''(x))^{-2}f'''(x)f'(x). \tag{31}$$

According to (30) and (31), $\bar{g}_k''(x)$ satisfies

$$\begin{aligned}\bar{g}_k''(x) =&(R + f''(x))^{-1}(f'''(x) + R\bar{g}_{k+1}''(x)) - (R + f''(x))^{-2}f'''(x)(f''(x) + R\bar{g}_{k+1}'(x))\\ &+ 2(R + f''(x))^{-3}(f'''(x))^2(f'(x) + R\bar{g}_{k+1}(x))\\ &- (R + f''(x))^{-2}[f''''(x)(f'(x) + R\bar{g}_{k+1}(x)) + f'''(x)(f''(x) + R\bar{g}_{k+1}'(x))],\end{aligned} \tag{32}$$

$$\begin{aligned}\bar{g}_N''(x) =&(R + f''(x))^{-1}f'''(x) - (R + f''(x))^{-2}f''(x)f'''(x)\\ &+ 2(R + f''(x))^{-3}(f'''(x))^2f'(x)\\ &- (R + f''(x))^{-2}[f''''(x)f'(x) + f'''(x)f''(x)].\end{aligned} \tag{33}$$

Let $x = x_*$ in (32) and (33). Recall that $f'(x_*) = 0$ and $\bar{g}_k(x_*) = 0$. We then have

$$\bar{g}_k''(x_*) = (R + f''(x_*))^{-1}(f'''(x_*) + R\bar{g}_{k+1}''(x_*)) - 2(R + f''(x_*))^{-2}f'''(x_*)(f''(x_*) + R\bar{g}_{k+1}'(x_*)), \quad (34)$$

$$\bar{g}_N''(x_*) = (R + f''(x_*))^{-1}f'''(x_*) - 2(R + f''(x_*))^{-2}f''(x_*)f'''(x_*). \quad (35)$$

In view of (34) and (35), by the induction derivation, we can acquire

$$\begin{aligned}\bar{g}_k''(x_*) =& f'''(x_*)(f''(x_*))^{-1}[(2N - 2k + 3)R^{N-k+1}f''(x_*) + R^{N-k+2} \\ & - (R + f''(x_*))^{N-k+2}]/(R + f''(x_*))^{N-k+2}. \end{aligned} \quad (36)$$

On account of the fact $|(2N-2k+3)R^{N-k+1}f''(x_*)+R^{N-k+2}-(R+f''(x_*))^{N-k+2}| < |(R+f''(x_*))^{N-k+2}|$, we have $|\bar{g}_k''(x_*)| < |f'''(x_*)/f''(x_*)|$. Now by associating it with (29), it is immediate to get (28).

Lemma 4 means that when $f''(x_*) > 0$, x_k converges to the minimal x_* of $f(x)$. Yet, if the initial guess x_0 is far away from x_* and $||R + f''(x_0)||$ is very small, $||(R + f''(x_0))^{-1}R||^{N+1}$ may be very large and even blow up so that the resultant x_1 is far away from x_*. For this, we propose Algorithm III.

Algorithm III.

$$x_{k+1} = x_k - \hat{g}_k(x_k), \quad (37)$$

$$\hat{g}_k(x_k) = (R + f''(x_k))^{-1}[f'(x_k) + R\hat{g}_{k-1}(x_k)],\ \hat{g}_0(x_k) = (R + f''(x_k))^{-1}f'(x_k). \quad (38)$$

The convergence of algorithm (37) and (38) is given below.

Theorem 1. Assume $f'(x_*) = 0$ and $f''(x_*) > 0$. For some integer $k_0 > 0$, $\{x_k\}$ generated by (37) and (38) admits

$$|x_{k+1} - x_*| \leqslant ||(R + f''(x_*))^{-1}R||^{k+1}|x_k - x_*|,\ k > k_0, \quad (39)$$

i.e., Algorithm III is superlinearly convergent.

Proof. Note that $f'(x)$ is also a factor of every $\hat{g}_k(x)$. Accordingly, $f'(x_*) = 0$ always results in $\hat{g}_k(x_*) = 0$. Instead of the backward induction in the proof of Lemma 4, we can show that $\hat{g}_k'(x_*) = I_n - [(R + f''(x_*))^{-1}R]^{k+1}$ by foward induction. Now it can be known that for x_k generated by (37) and (38), there holds

$$\begin{aligned} & x_{k+1} - x_* \\ & = x_k - x_* - \hat{g}_k(x_k) \\ & = x_k - x_* - [\hat{g}_k(x_*) + g_k'(x_*)(x_k - x_*) + o(|x_k - x_*|)] \\ & \approx x_k - x_* - \hat{g}_k'(x_*)(x_k - x_*) \\ & = (I_n - \hat{g}_k'(x_*))(x_k - x_*) \\ & = [(R + f''(x_*))^{-1}R]^{k+1}(x_k - x_*). \end{aligned} \quad (40)$$

In view of the fact that $||(R + f''(x_*))^{-1}R|| < 1$ when $f''(x_*) > 0$, the proof is completed.

Remark 4. According to (39) in Theorem 1 and $||(R+f''(x_*))^{-1}R|| < 1$, the smaller $||(R+f''(x_*))^{-1}R||$, the more rapidly Algorithm III in (37) and (38) converges.

Remark 5. Algorithm III in (37) and (38) has varying $\hat{g}_k(x_k)$ at each iteration. From (39), Algorithm III holds a greater $\hat{g}_k(x_k)$ in the initial stages and a smaller one in the latter stages, which avoids oscillatory occurrence.

Remark 6. It should be pointed out that, unlike Newton's method, Algorithm III in (37) and (38) still works even if there occasionally holds $f''(x) = 0$ because of introducing the positive definite matrix R.

4 Numerical experiment

We will illustrate some dramatic merits of the proposed algorithms based on the OCP method by some testing functions. All our experiments are executed on the same computer with a 3.2 GHz Intel Core i5 processor with Matlab R2020a. We do our best to guarantee that all algorithms run under the same environment and conditions. For the following gradient descent, we always choose the favorable step size to ensure rapid convergence as possibly as it can. Algorithm I is implemented using the MSA [19].

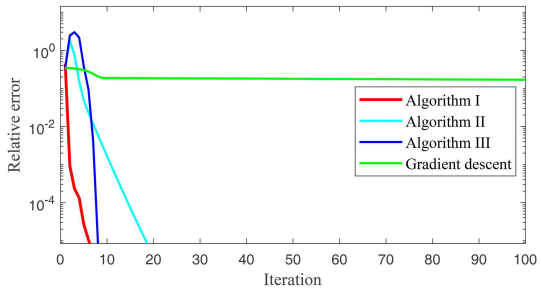

Figure 1 (Color online) Relative error $\|(x_k, y_k) - (x^*, y^*)\| / \|(x^*, y^*)\|$ for $f_1(x, y)$.

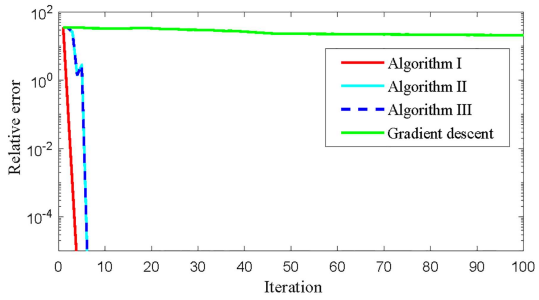

Figure 2 (Color online) Relative error $\|(x_k, y_k) - (x^*, y^*)\| / \|(x^*, y^*)\|$ for $f_2(x, y)$.

4.1 Fast convergence

Consider the Freudenstein and Roth function

$$f_1(x, y) = (-13 + x + ((5 - y)y - 2)y)^2 + \ (-29 + x + ((y + 1)y - 14)y)^2.$$

It has a unique minimum point at $(x^*, y^*) = (5, 4)$. Let $(x_0, y_0) = (6, 2)$ and $N = 100$. We select an initial step size $\eta = 0.0004$ for gradient descent, and set $R = 0.01 \times I_2$, $R = \text{diag}\{400, 35000\}$, and $R = \text{diag}\{1, 500\}$ for Algorithms I–III, respectively. It can be seen from Figure 1 that the optimization algorithms based on optimal control converge more rapidly than gradient descent. In some cases, it is difficult for gradient descent to converge unless with the appropriate step size (specific details can be found in Subsection 4.4).

Consider the Rosenbrock's Parabolic Valley function

$$f_2(x, y) = 100(y - x^2)^2 + \ (1 - x)^2.$$

This function has a unique minimum point at $(x^*, y^*) = (1, 1)$. Let the initial value be$(x_0, y_0) = (-7, 50)$ and $N = 100$. We select the step size $\eta = 0.0001$ for gradient descent and $R = 0.001 \times I_2$, $R = 0.0001 \times I_2$, and $R = 0.0001 \times I_2$ for Algorithms I–III, respectively. As shown in Figure 2, the gradient descent converges slowly, whereas our algorithms converge within only several iterations. Our algorithms offer a more favorable convergence behavior.

4.2 Wide application

Compared to Newton's method, Algorithms I–III still work even if the indefinite Hessian matrix appears during the iterating process. Consider the function

$$f_3(x) = \frac{1}{4}x^4 - \frac{1}{2}x^2 - 3x.$$

This function has a unique minimum point at $x^* = 1.6717$ and we initialize $x_0 = 0$, $N = 100$. Set $R = 0.01$, $R = 150$, and $R = 10$ for Algorithms I–III, respectively. Figure 3 shows that Newton's method diverges in this case, while our algorithms converge with a small relative error.

Consider the Beale's function

$$f_4(x, y) = (1.5 - x(1 - y))^2 + (2.25 - x(1 - y^2))^2 + \ (2.625 - x(1 - y^3))^2.$$

This function has a unique minimum point at $(x^*, y^*) = (3, 0.5)$ and we initialize $(x_0, y_0) = (4, 0)$, $N = 100$. We choose $R = 10^{-6} \times I_2$, $R = \text{diag}\{100, 1000\}$ and $R = \text{diag}\{0.1, 10\}$ for Algorithm I–III, respectively. Figure 4 shows that our algorithms converged and Newton's method did not converge to the minimum point. It should be pointed out that Newton's method finds a saddle point in this case.

4.3 Application to non-convex functions

Using Algorithm I in (6) to solve OP (1), each local minimum point can be associated with a different input weight matrix R. Consider the non-convex function

$$f_5(x) = (x - 1)(x + 1)(x + 0.5)(x + 1.5)(x - 0.5)(x - 1.5)$$

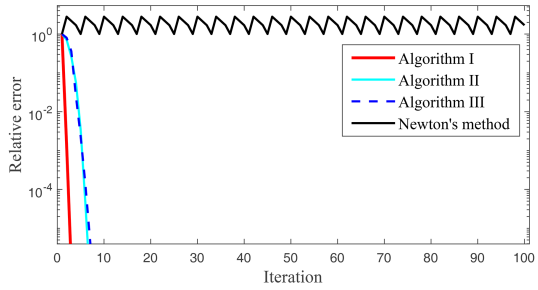

Figure 3 (Color online) Relative error $\|x_k - x^*\| / \|x^*\|$ for $f_3(x)$.

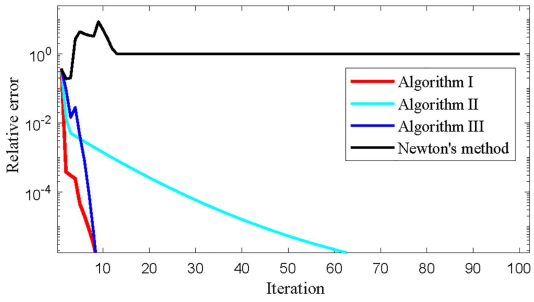

Figure 4 (Color online) Relative error $\|(x_k, y_k) - (x^*, y^*)\| / \|(x^*, y^*)\|$ for $f_4(x, y)$.

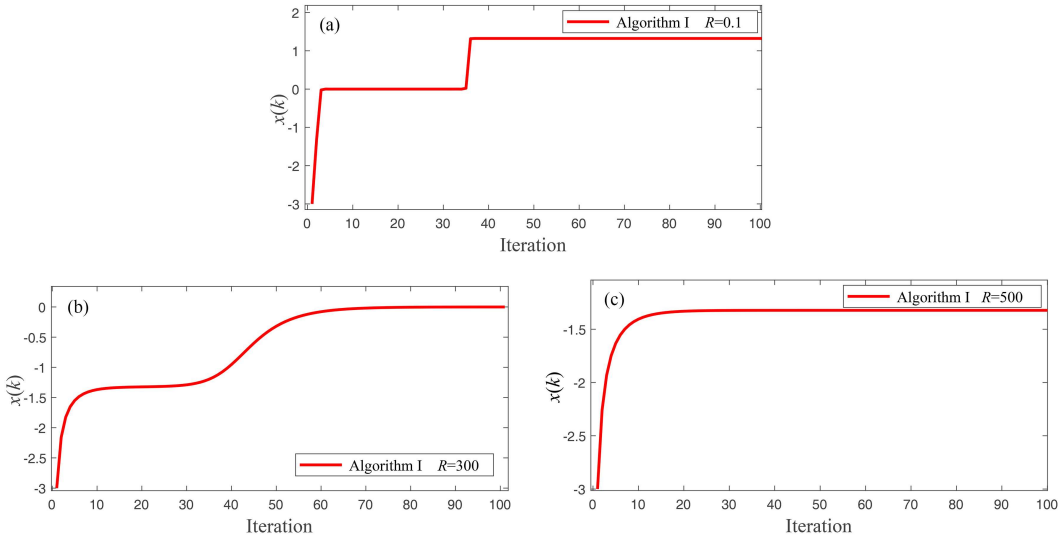

Figure 5 (Color online) Iteration trajectories of x_k for $f_5(x)$ with (a) $R = 0.1$, (b) $R = 300$, and (c) $R = 500$ using Algorithm I.

with three local minimum points $x_1^* = 1.323$, $x_2^* = 0$, $x_3^* = -1.323$. we initialize $x_0 = -3$, $N = 100$. When R is set to 0.1, 300, and 500, the sequence $\{x_k\}$ convergence to x_1^*, x_2^*, x_3^* respectively. The corresponding iteration trajectories are shown in Figure 5. Based on the experimental results, it is clear that different input weight matrices R could make x_k converge to different local minimum points.

4.4 Quantitative comparison

Since Algorithm III is straightforward to implement and demonstrates good convergence, it is our preferred choice. We quantificationally compare the performance of Algorithm III in Table 1 from the average runtime and the number of iterations.

It is shown in Table 1 that the average runtime of our Algorithm III is less than gradient descent when the same accuracy is achieved. Particularly, to minimize $f_1(x, y)$, the relative error of gradient descent reaches only 10^{-12}. It is almost impossible to reduce the relative error further by iterating. It can be noticed that the average running time of our method is not less than Newton's method when used to solve $\min f_1(x, y)$ and $\min f_2(x, y)$. Yet, Newton's method fails to work for $\min f_3(x)$, $\min f_4(x, y)$.

5 Conclusion

In the paper, several novel algorithms have been developed based on OCP. The algorithms converge more rapidly than gradient descent. Meanwhile, unlike Newton's method, they still work even if the singular or indefinite Hessian matrix appears during the iterating process. We also pointed out that Newton's method is recovered when the control weight matrix disappears in the developed Algorithms II and III. The convergence rate of the algorithms depends on an adjustable input weight matrix. Because of the

Table 1 Comparison average over 20 trials for solving OP

Function	Method	Average runtime (s)	Relative error	Iterations
$f_1(x,y)$ $(x_0,y_0)=(-100,10)$	Algorithm III	0.000632	$<10^{-16}$	13
	Gradient descent	1.511929	10^{-12}	$>10^6$
	Newton's method	0.000174	10^{-16}	13
$f_2(x,y)$ $(x_0,y_0)=(-7,50)$	Algorithm III	0.000233	10^{-7}	7
	Gradient descent	0.451101	10^{-7}	$>5\times10^5$
	Newton's method	0.000076	10^{-7}	7
$f_3(x)$ $x_0=0$	Algorithm III	0.000025	10^{-7}	12
	Gradient descent	0.000079	10^{-7}	38
	Newton's method	–	–	–
$f_4(x,y)$ $(x_0,y_0)=(4,0)$	Algorithm III	0.000379	10^{-7}	9
	Gradient descent	0.001396	10^{-7}	1200
	Newton's method	–	–	–

clear physical meanings inherited from OCP, it is very convenient to tune it to speed up the convergence rate.

Acknowledgements This work was supported by Original Exploratory Program Project of National Natural Science Foundation of China (Grant No. 62450004), High-Level Talent Team Project of Qingdao West Coast New Area (Grant No. RCTD-JC-2019-05), Joint Funds of the National Natural Science Foundation of China (Grant No. U23A20325), Major Basic Research of Natural Science Foundation of Shandong Province (Grant No. ZR2021ZD14), and Natural Science Foundation of Shandong Province (Grant No. ZR2024MF045).

References

1 Rao S S. Engineering Optimization: Theory and Practice. Hoboken: John Wiley & Sons, 2019
2 Belegundu A D, Chandrupatla T R. Optimization Concepts and Applications in Engineering. Cambridge: Cambridge University Press, 2019
3 Banichuk N V. Introduction to Optimization of Structures. New York: Springer Science & Business Media, 2013
4 Chong E K P, Lu W S, Zak S H. An Introduction to Optimization: with Applications to Machine Learning. Hoboken: John Wiley & Sons, 2023
5 Faires J D, Burden R L. Numerical Methods. Boston: Cengage Learning, 2003
6 Boyd S P, Vandenberghe L. Convex Optimization. Cambridge: Cambridge University Press, 2004
7 Baydin A G, Cornish R, Rubio D M, et al. Online learning rate adaptation with hypergradient descent. 2017. ArXiv:1703.04782
8 Fei Z, Wu Z, Xiao Y, et al. A new short-arc fitting method with high precision using Adam optimization algorithm. Optik, 2020, 212: 164788
9 Ruder S. An overview of gradient descent optimization algorithms. 2016. ArXiv:1609.04747
10 Bottou L. Large-scale machine learning with stochastic gradient descent. In: Proceedings of the 19th International Conference on Computational Statistics, Paris, 2010. 177–186
11 Qian N. On the momentum term in gradient descent learning algorithms. Neural Netws, 1999, 12: 145–151
12 Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res, 2011, 12: 2161–2180
13 O'Donoghue B, Candés E. Adaptive restart for accelerated gradient schemes. Found Comput Math, 2015, 15: 715–732
14 Zeiler M D. Adadelta: an adaptive learning rate method. 2012. ArXiv:1212.5701
15 Broyden C G. Quasi-Newton methods and their application to function minimisation. Math Comp, 1967, 21: 368–381
16 Dembo R S, Eisenstat S C, Steihaug T. Inexact Newton methods. SIAM J Numer Anal, 1982, 19: 400–408
17 Fletcher R, Freeman T L. A modified Newton method for minimization. J Optim Theor Appl, 1977, 23: 357–372
18 Lewis F L, Vrabie D, Syrmos V L. Optimal Control. Hoboken: John Wiley & Sons, 2012
19 Xu Y M, Guo Z Y, Wang H X, et al. Optimization method based on optimal control. 2023. ArXiv:2309.05280