Optimization Methods and Software

ISSN: 1055-6788 (Print) 1029-4937 (Online) Journal homepage: http://www.tandfonline.com/loi/goms20

An augmented Lagrangian trust region method for equality constrained optimization

Xiao Wang & Yaxiang Yuan

To cite this article: Xiao Wang & Yaxiang Yuan (2015) An augmented Lagrangian trust region method for equality constrained optimization, Optimization Methods and Software, 30:3, 559-582, DOI: 10.1080/10556788.2014.940947

To link to this article: http://dx.doi.org/10.1080/10556788.2014.940947

Accepted author version posted online: 08 Aug 2014.
Published online: 01 Sep 2014.

Optimization Methods & Software, 2015
Vol. 30, No. 3, 559–582, http://dx.doi.org/10.1080/10556788.2014.940947

An augmented Lagrangian trust region method for equality constrained optimization

Xiao Wang[a*] and Yaxiang Yuan[b]

[a]*School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, People's Republic of China;* [b]*State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, People's Republic of China*

(*Received 15 December 2013; accepted 25 June 2014*)

In this paper we propose an augmented Lagrangian trust region method for equality constrained optimization. Different from standard augmented Lagrangian methods which minimize the augmented Lagrangian function for fixed Lagrange multiplier and penalty parameter at each iteration, the proposed method tries to minimize its second-order approximation function. We propose a new strategy for adjusting the penalty parameter. With adaptive update of Lagrange multipliers, we prove the global convergence of the proposed method. Numerical results on test problems from the CUTEr collection are also reported.

Keywords: equality constraints; augmented Lagrangian function; trust region; Lagrange multiplier; penalty parameter; convergence

AMS Subject Classification: 90C30; 65K05

1. Introduction

In this paper, we consider the following equality constrained optimization:

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & f(\boldsymbol{x}) \\ \text{s. t.} \quad & \boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}, \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is Lipschitz continuously differentiable and $\boldsymbol{c} = (c^{(1)}, \ldots, c^{(m)})^{\mathsf{T}}$ with $c^{(i)} : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, Lipschitz continuously differentiable.

In the past decades, quantities of methods have been studied for solving (1). One of the most effective methods is the sequential quadratic programming (SQP) method, which dates back to Wilson [31] and later is developed by Han [20], Powell [27], etc. Its basic idea is to transform the original problem into a sequence of quadratic programming (QP) subproblems. At each iteration, a trial step solves the following QP subproblem:

$$\begin{aligned} \min_{\boldsymbol{d}\in\mathbb{R}^n} \quad & \boldsymbol{g}_k^{\mathsf{T}}\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^{\mathsf{T}}\mathbf{B}_k\boldsymbol{d} \\ \text{s. t.} \quad & \boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d} = \boldsymbol{0}, \end{aligned} \tag{2}$$

*Corresponding author. Email: wangxiao@ucas.ac.cn

where k is referred as the iteration number, $\mathbf{A}_k$ is the Jacobian of the constraints and $\mathbf{B}_k$ denotes the exact Hessian of the Lagrange function $f(\boldsymbol{x}) - \boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{c}(\boldsymbol{x})$ or its approximation evaluated at the current iterate $\boldsymbol{x}_k$. Here, $\boldsymbol{\lambda} \in \mathbb{R}^m$ is a vector of multipliers. To be practical, specific strategies need to be applied to globalize the SQP method. One strategy is to carry out line search techniques. However, in order to avoid unbounded solutions, line search methods normally require $\mathbf{B}_k$ to be positive definite in the null space of $\mathbf{A}_k$, a condition which may fail when $\mathbf{B}_k$ is set as the exact Hessian of the Lagrangian function. Another strategy is to apply the trust region technique. It restricts the length of the trial step by adding the trust region constraint $\|\boldsymbol{d}\| \leq \Delta_k$. It allows indefinite $\mathbf{B}_k$, so that $\mathbf{B}_k$ can be chosen as the exact Hessian of the Lagrange function. However, complications may arise, as the inclusion of the trust region bound might cause infeasibility of the linearized constraints. Thus, it is necessary to relax the constraints (see, e.g. [6,28]), which however complicates the algorithm. Besides, it is very important to choose an effective merit function to ensure global convergence. Several kinds of merit functions have been studied, for example, l_1 penalty function, l_∞ penalty function and augmented Lagrangian function. However, the effectiveness of all these merit functions depends on, and may be sensitive to, the choices of some parameters, which need very careful consideration.

To avoid the possible difficulties arising from infeasible subproblems, two kinds of quadratic-related subproblems based on exact penalty functions have been proposed. Two resultant methods are the sequential l_1 quadratic programming (Sl_1QP) method [15] and the sequential l_∞ quadratic programming (Sl_∞QP) method [32], respectively. Their common idea is penalizing the linearized constraints in the objective of (2) in terms of an l_1 penalty term or an l_∞ penalty term, and integrating trust region techniques. And the corresponding l_1 or l_∞ penalty function is applied as the merit function. However, one major concern associated with these two methods is the nonsmoothness. As the merit function is nonsmooth, *Maratos effect* may happen [22]. It prevents the superlinear trial step from being accepted due to an increase in the merit function. Another difficulty of using a nonsmooth exact penalty function is that the corresponding subproblem is a nonsmooth problem, which may be not easy to solve. Motivated by these, we consider to build up some kind of smooth and feasible subproblems.

Penalty methods for nonlinear programming have been greatly developed in the past years. The key idea of such methods is moving the original constraints into the objective in (1). Then, at each iteration a much easier optimization problem, for example, unconstrained optimization or simple-bounded optimization problem, is to be solved. One of the most famous penalty methods is the augmented Lagrangian method. This method was first studied by Hestenes [21], Powell [26] and Rockfellar [30], and then was adapted by Conn *et al.* [8]. Since then, it has attracted much attention, and many augmented Lagrangian function-based variants have sprung up, such as [2,3,13,14]. It is fully applied in other fields, such as compressed sensing [5], semidefinite programming [34] and matrix completion [16]. Let us take (1) for example. At each iteration, the augmented Lagrangian function with fixed Lagrange multiplier $\boldsymbol{\lambda}_k$ and penalty parameter σ_k is minimized:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) - \boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}(\boldsymbol{x}) + \frac{\sigma_k}{2}\|\boldsymbol{c}(\boldsymbol{x})\|_2^2. \tag{3}$$

By introducing the explicit Lagrange multiplier, this method reduces the possibility of ill conditioning arising from simple penalty methods, like the Courant penalty method or log-barrier approach (see [25] for reference). Moreover, it does not introduce any nonsmoothness, different from Sl_1QP and Sl_∞QP methods (see, e.g. [25]). Thus, implementations can be constructed from standard methods and software for unconstrained optimization. One influential work on practical augmented Lagrangian method was the paper by Conn *et al.* [8], based on which a well-known package LANCELOT [9] has been released. We will compare our method with it in the numerical experiments presented below. In [8], subproblem (3) is solved until its first-order criticality measure within the precision w_k, where the sequence $\{w_k\}$ is reduced to zero. However, it is

questionable whether such an approach is the best. Firstly, when w_k is very small, if f is highly nonlinear, it will be very costly to solve (3) to achieve the required precision w_k. Secondly, when Lagrange multiplier $\boldsymbol{\lambda}_k$ is far away from the Lagrange multiplier associated with the solution, it seems unnecessary to make too much endeavour to solve (3) with high precision. Therefore, it is desirable to construct subproblems which can be easily solved to reduce the computational cost at each iteration.

A useful approach was given by Niu and Yuan [24]. They propose a filter-typed method, where the subproblem is to minimize a quadratic approximation to the augmented Lagrangian function in the trust region, thus different from the standard augmented Lagrangian methods where (3) to be solved. An advantage of Niu and Yuan's method is that the trust region subproblem is a relatively easier problem, which can be efficiently solved by many algorithms. Some satisfactory numerical results are reported in [24]. However, no convergence results are given in that paper. Recently, Curtis *et al.* [11] apply similar trust region subproblems and propose an adaptive method for large-scale constrained optimization. Nice numerical results are presented. However, the first-order criticality of the accumulation points of iterates could not be obtained, if there are infinite number of penalty parameter updates and the constraint violation converges to zero. Motivated by the nice numerical properties of trust region subproblems applied in aforementioned two papers, we adopt the same design of subproblems in our algorithm.

Contributions. Our contributions in this paper lie in several folds. Firstly, we propose a new strategy to update the penalty parameters, which not only depends on constraint violations, but also relies on the model improvement. This new strategy normally prevents the iteration from being trapped around some local minimizer of the augmented Lagrangian function with fixed Lagrange multiplier and penalty parameter. Secondly, for this proposed method, we show its global convergence when penalty parameters are either bounded or unbounded. Moreover, the relaxed constant positive linear dependence (RCPLD) constraint qualification (will be defined later) is applied to analyse the global convergence. This RCPLD condition is recently proposed by Andreani *et al.* [4] and has been shown weaker than many other traditional constraint qualification conditions.

Notations. Throughout this paper, we use the notation $\mathbb{R}^n$ to represent the n dimensional real vector space, and $\mathbb{R}_+$ as the set of nonnegative real numbers. $\mathbb{N}$ denotes the set of all non-negative integers. The superscript $^{(i)}$ refers to the ith element of a vector, while the subscript $_k$ refers to the iteration number in an algorithm. Without specification, $\|\cdot\|$ represents the Euclidean norm $\|\cdot\|_2$ in $\mathbb{R}^n$. We denote $g(\boldsymbol{x})$ as the gradient of f and $\mathbf{A}(\boldsymbol{x})$ as the Jacobian of $\boldsymbol{c}$ at $\boldsymbol{x}$, namely,

$$g(\boldsymbol{x}) = \nabla f(\boldsymbol{x}), \quad \mathbf{A}(\boldsymbol{x}) = (\nabla c^{(1)}(\boldsymbol{x}), \ldots, \nabla c^{(m)}(\boldsymbol{x}))^{\mathsf{T}}.$$

We denote $\mathcal{N}_k$ as the null space of $\mathbf{A}(\boldsymbol{x}_k)$ and the operator P_Ω as the Euclidean projection into a closed and convex set Ω. For convenience, we abbreviate $g(\boldsymbol{x}_k)$ to $\boldsymbol{g}_k$, and similar abbreviations f_k, $\boldsymbol{c}_k$ and $\mathbf{A}_k$ are also used.

Organizations. The rest of this paper is organized as follows. In Section 2, we propose an augmented Lagrangian trust region (ALTR) method for equality constrained optimization (1). In Section 3, we study the global convergence of the proposed method. In Section 4, preliminary numerical results are reported on majority of the equality constrained optimization problems from CUTEr. Finally, we draw some conclusions in Section 5.

2. Algorithm description

In this section, we propose an ALTR method for equality constrained optimization (1). Firstly, we give two widely used optimality conditions characterizing the feasible points of (1) (see, e.g. [32]).

DEFINITION 2.1 *A feasible point x_* satisfies the Fritz-John conditions, if there exist a scalar λ_0 and a vector $\boldsymbol{\lambda} \in \mathbb{R}^m$, such that $\lambda_0^2 + \|\boldsymbol{\lambda}\|^2 \neq 0$ and $\lambda_0 g(x_*) = \boldsymbol{A}(x_*)^{\mathsf{T}}\boldsymbol{\lambda}$. Then x_* is called a Fritz-John point of (1).*

Specifically, when $\lambda_0 \neq 0$, we obtain the definition of Karush-Kuhn-Tucker (KKT) points.

DEFINITION 2.2 *A feasible point x_* satisfies the KKT conditions, if there exists a vector $\boldsymbol{\lambda}_* \in \mathbb{R}^m$ such that*

$$g(x_*) = \boldsymbol{A}(x_*)^{\mathsf{T}}\boldsymbol{\lambda}_*. \tag{4}$$

Then x_* is called a KKT point of (1).

Constraint qualifications are normally used to describe the relations among the gradients of constraints. One of the most widely used constraint qualification conditions in nonlinear programming is the so-called linear independence constraint qualification (LICQ) (see, e.g. [25]). With respect to (1), we give its definition as follows.

DEFINITION 2.3 *Given a feasible point x_* of* (1)*, we say that the LICQ condition holds at x_*, if the gradients of all the equality constraints $\nabla c^{(i)}(x)$, $i = 1, \ldots, m$, are linearly independent.*

Recently, another constraint qualification condition has attracted much attention [1–3,29], which is called constant positive linear dependence (CPLD) condition. It has been shown that the CPLD condition is weaker than LICQ condition. Later Andreani *et al.* [4] propose a RCPLD constraint qualification, which is proved strictly weaker than CPLD. As in this paper the problem (1) has only equality constraints, we state the definition of RCPLD with respect to equality constrained optimization.

DEFINITION 2.4 *Given a feasible point x_* of* (1)*, we say that the RCPLD condition holds at x_*, if there exists a neighbourhood of x_* such that $\{\nabla c^{(i)}(x)\}_{i=1}^m$ has the same rank for any x in this neighbourhood.*

We next discuss our algorithm in details, including the issues associated with the trust region subproblems, and the update strategy of Lagrange multipliers and penalty parameters.

2.1 *Trust region subproblems*

Firstly, let us recall classic augmented Lagrangian methods. The augmented Lagrangian function with respect to $(x, \boldsymbol{\lambda}, \sigma)$ is defined as

$$L(x; \boldsymbol{\lambda}, \sigma) = f(x) - \boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{c}(x) + \frac{\sigma}{2}\|\boldsymbol{c}(x)\|^2, \tag{5}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^m$ refers to the vector of multipliers and $\sigma \in \mathbb{R}_+$ is denoted as the penalty parameter. The reason for introducing the augmented Lagrangian function (5) is based on the following theoretical result. If x_* is a local solution of (1) and $\boldsymbol{\lambda}_*$ is the Lagrange multiplier associated with x_*, under some regularity assumptions, there exists a threshold σ_*, such that, for all $\sigma > \sigma_*$, x_* is a strict local minimizer of $L(x; \boldsymbol{\lambda}_*, \sigma)$. Interested readers are referred to [25, Theorem 17.5] for further details. Based on this result, even though in practice the exact values of $\boldsymbol{\lambda}_*$ and σ_* are not known, a good estimate of x_* can still be obtained by minimizing $L(x; \boldsymbol{\lambda}, \sigma)$ when σ is

not particularly large and $\boldsymbol{\lambda}$ is close to $\boldsymbol{\lambda}_*$. Therefore, given the current iterate $\boldsymbol{x}_k$, $\boldsymbol{x}_{k+1}$ can be obtained by solving the following subproblem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k). \tag{6}$$

The good news is that there is no need to solve (6) exactly. Given a tolerance w_k, Conn *et al.* [8] suggest to apply an algorithm solving (6) until finding $\boldsymbol{x}_{k+1}$ such that

$$\|\nabla_{\boldsymbol{x}} L(\boldsymbol{x}_{k+1};\boldsymbol{\lambda}_k,\sigma_k)\| \le w_k.$$

Then by forcing $w_k \to 0$ together with adaptive update of $\boldsymbol{\lambda}_k$ and σ_k, the algorithm will find KKT points or Fritz-John points. However, in practice, as $f(\boldsymbol{x})$ might be highly nonlinear, this procedure may be too expensive, especially when w_k is small.

Instead of minimizing the nonlinear function $L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k)$, Niu and Yuan [24] consider its quadratic approximation. Such an approximation to $L(\boldsymbol{x}_k+\boldsymbol{d};\boldsymbol{\lambda}_k,\sigma_k)$ can be defined as

$$\Phi_k(\boldsymbol{d}) := L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k) + \boldsymbol{g}_k^{\mathsf{T}}\boldsymbol{d} - \boldsymbol{\lambda}_k^{\mathsf{T}}\mathbf{A}_k\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^{\mathsf{T}}\mathbf{B}_k\boldsymbol{d} + \frac{\sigma_k}{2}\|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2, \tag{7}$$

where $\mathbf{B}_k$ is the Hessian of Lagrange function $f(\boldsymbol{x}) - \boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}(\boldsymbol{x})$, or its approximation, and the constraints in the penalty term are approximated by their linearizations. To ensure that $\Phi_k(\boldsymbol{d})$ can be trusted as an adequate approximation, the search region is restricted around the current iterate by applying the trust region technique, which gives the following trust region subproblem:

$$\begin{aligned} \min_{\boldsymbol{d}\in\mathbb{R}^n} \quad & q_k(\boldsymbol{d}) := \boldsymbol{g}_k^{\mathsf{T}}\boldsymbol{d} - \boldsymbol{\lambda}_k^{\mathsf{T}}\mathbf{A}_k\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^{\mathsf{T}}\mathbf{B}_k\boldsymbol{d} + \frac{\sigma_k}{2}\|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2 \\ \text{s. t.} \quad & \|\boldsymbol{d}\| \le \Delta_k. \end{aligned} \tag{8}$$

In (8), choices of approximate Hessian $\mathbf{B}_k$ are possible. For example, $\mathbf{B}_k$ can be generated through update strategies, such as damped BFGS update (see, e.g. [25]), which only depends on the first-order derivatives of f and $\boldsymbol{c}$.

Subproblem (8) minimizes a quadratic function within an l_2-norm trust region. This kind of problem has the strong advantage that there are very efficient methods for solving it, such as *gqtpar* subroutine [23], sequential subspace method [18] and so on. Actually, it just needs at most a polynomial (in n) number of operations (see, e.g. [10]). Moreover, there is no need to solve (8) exactly or nearly exactly. It is enough to find an inexact solution $\boldsymbol{s}_k$ satisfying

$$q_k(\mathbf{0}) - q_k(\boldsymbol{s}_k) \ge \bar{\beta}\cdot\left[q_k(\mathbf{0}) - \min_{\|\boldsymbol{d}\|\le\Delta_k} q_k(\boldsymbol{d})\right] \tag{9}$$

for some positive constant $\bar{\beta}\in(0,1)$. For example, the truncated CG method always satisfies (9) for $\bar{\beta}=0.5$, if the Hessian of $q_k(\boldsymbol{d})$ is positive definite (see, e.g. [33]). We believe that (9) admits many inexact solutions of (8). In the following analysis, we will assume that (9) holds for all k.

After obtaining the trial step $\boldsymbol{s}_k$, we should decide whether to accept it or not. Similar to the classic strategy in trust region methods, the ratio between the actual reduction and the predicted reduction:

$$\rho_k := \frac{\text{Ared}_k}{\text{Pred}_k} = \frac{L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k) - L(\boldsymbol{x}_k+\boldsymbol{s}_k;\boldsymbol{\lambda}_k,\sigma_k)}{q_k(\mathbf{0}) - q_k(\boldsymbol{s}_k)} \tag{10}$$

is computed. If ρ_k is far from 1, it implies that the approximation of $L(\boldsymbol{x}_k+\boldsymbol{d};\boldsymbol{\lambda}_k,\sigma_k)$ is not adequate in the current trust region. Then we reduce Δ_k. We expect ρ_k to be close to one when Δ_k is sufficiently small.

However, a delicate situation may arise. It may happen that $\boldsymbol{x}_k$ is a local minimizer of $L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k)$ for some $\boldsymbol{\lambda}_k$ and σ_k, so the equality $\nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k)=0$ holds. Then although the least value of $L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k)$ is reached at $\boldsymbol{x}=\boldsymbol{x}_k$, and although the gradient of $q_k(\boldsymbol{d})$ at $\boldsymbol{d}=\boldsymbol{0}$ is zero, $q_k(\boldsymbol{d})$ may take negative values near $\boldsymbol{d}=\boldsymbol{0}$ due to negative curvature, because $\mathbf{B}_k$ could be a large negative multiple of the unit matrix. Then Ared_k and Pred_k are negative and positive, respectively, for every positive trust region radius Δ_k, giving $\rho_k<0$, which may cause an infinite loop. As a matter of fact, in such circumstance, two cases are possible. One case arises when $\boldsymbol{x}_k$ is feasible. Then the relations

$$\nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k)=\boldsymbol{g}_k-\mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k+\sigma_k\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k=\boldsymbol{0}\quad\text{and}\quad\|\boldsymbol{c}_k\|=0 \tag{11}$$

indicate that $\boldsymbol{x}_k$ is a KKT point as $\boldsymbol{g}_k-\mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k=0$. In this case, the algorithm terminates finitely. The other case happens when $\boldsymbol{x}_k$ is infeasible, giving

$$\nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k)=\boldsymbol{0}\quad\text{and}\quad\|\boldsymbol{c}_k\|>0. \tag{12}$$

In order to make the algorithm escape from this situation, we break the first equality in (12) by increasing σ_k. If $\nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k)\neq\boldsymbol{0}$ for a new larger σ_k, it can go into the minimization phase of (8). While, if for all large $\sigma\geq\sigma_k$, $\nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma)$ is zero vector and $\|\boldsymbol{c}_k\|>0$, it will be proved that $\boldsymbol{x}_k$ is an infeasible stationary point of minimizing $\|\boldsymbol{c}(\boldsymbol{x})\|^2$ in Theorem 3.2. Besides, normally increasing σ_k is helpful for reducing constraint violations.

Let us look back to subproblem (8) again. Compared with classic penalty methods, it just needs to minimize a quadratic approximation of the augmented Lagrangian function at each iteration. Therefore, it can be regarded as applying one iteration step of a trust region method to solve (6). This strategy avoids some trouble in trying to solve the equations $\nabla_x L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k)=\boldsymbol{0}$ approximately. Thus we can expect a substantial reduction in the cost of computation at each iteration.

As we have mentioned already, standard trust region SQP methods may encounter infeasible subproblems. Although Sl_1QP and Sl_∞QP are designed to settle this situation, these two approaches still involve difficulties arisen from nonsmoothness. However, subproblem (8) can be seen as applying the augmented Lagrangian function directly for (2), with a trust region constraint. This not only solves the difficult infeasibility issues, but also avoids the disadvantages of nonsmooth penalty functions.

2.2 *Update of penalty parameters and vectors of multipliers*

In augmented Lagrangian methods, the strategies to update penalty parameters and Lagrange multipliers play an important role. Different updating schemes normally have different influences on the efficiency of algorithms.

Firstly, we decide how to update σ_k effectively in our algorithm. Niu and Yuan [24] suggest

$$\sigma_{k+1}=\begin{cases}\max\{2\sigma_k,2\|\boldsymbol{\lambda}_{k+1}\|\} & \text{if } \|\boldsymbol{c}(\boldsymbol{x}_k+\boldsymbol{s}_k)\|\geq 0.5\|\boldsymbol{c}_k\|,\\ \max\{\sigma_k,2\|\boldsymbol{\lambda}_{k+1}\|\}, & \text{otherwise.}\end{cases} \tag{13}$$

The motivation for (13) is the inequality $\sigma_{k+1}\geq 2\|\boldsymbol{\lambda}_{k+1}\|$, a condition that Niu and Yuan [24] use to analyse the exactness of the augmented Lagrangian function. However, $\sigma_*\geq 2\|\boldsymbol{\lambda}_*\|$ is not a necessary condition for $L(\boldsymbol{x};\boldsymbol{\lambda}_*,\sigma_*)$ to be an exact penalty function of (1). Thus, in our method, we do not use (13) to update penalty parameters.

As we have discussed in the subsection above, in order not to be trapped around local minimizers of $L(\boldsymbol{x};\boldsymbol{\lambda}_k,\sigma_k)$, we should increase σ_k as long as (12) holds. We next assume that the

subproblem (8) generates a trial step $\boldsymbol{s}_k$. Most updating techniques for penalty parameters depend on the improvement of constraint violations, such as (13). In general, the penalty parameter is increased if sufficient improvement is not obtained. To be more exact, if the constraint violation $\|\boldsymbol{c}(\boldsymbol{x}_k + \boldsymbol{s}_k)\|$ is not improved much compared with $\|\boldsymbol{c}_k\|$, e.g.

$$\|\boldsymbol{c}(\boldsymbol{x}_k + \boldsymbol{s}_k)\| \geq \tau \|\boldsymbol{c}_k\|$$

for some constant $0 < \tau < 1$ independent of k, σ_k will be increased.

However, it is inadequate to adopt such kind of strategies in our method. We see that in the subproblem (8), the penalty term involving σ_k also plays an important role in the objective, which consequently affects the computation of the trial step $\boldsymbol{s}_k$ as well as the predicted reduction Pred_k. Conversely, the important contribution from σ_k to Pred_k inspires us to use Pred_k in the choice of σ_{k+1}. When updating σ_k, we pay attention to the relation among the predicted reduction Pred_k, the constraint violation $\|\boldsymbol{c}_k\|$ and the trust region radius Δ_k. Our novel way of updating σ_k tests the inequality:

$$\text{Pred}_k < \delta_k \sigma_k \min\{\Delta_k \|\boldsymbol{c}_k\|, \|\boldsymbol{c}_k\|^2\}, \tag{14}$$

where $\delta_k \in \mathbb{R}_+$ is an auxiliary parameter. If (14) holds, we think that the model (8) has not given enough attention to $\|\boldsymbol{c}_k\|$, so we focus on reducing $\|\boldsymbol{c}_k\|$ by increasing the penalty parameter σ_k. It is necessary for $\sigma_k \delta_k$ to have the property

$$\sigma_k \delta_k \to 0,$$

if σ_k increases to infinity. Once σ_k increases to a very large number, $\|\boldsymbol{c}_k\|$ becomes very small. Then $\boldsymbol{x}_k$ is close to the feasible region, so it is not necessary to increase σ_k unless Pred_k is very small. The condition (14) is different from that in [32] due to the l_2-norm penalty term of the augmented Lagrangian function.

Our next major consideration is how to update the multiplier $\boldsymbol{\lambda}_k$. Whenever the iterates are relatively far away from the feasible region, or the constraint violation is not improved much, it seems urgent to reduce the constraint violation and drive the iterates to the feasible region instead of updating $\boldsymbol{\lambda}_k$. We set a switch condition

$$\|\boldsymbol{c}_{k+1}\| \leq R_k, \tag{15}$$

where $\{R_k\}$ is a sequence of nonincreasing controlling factors. It can be regarded as some indication on the progress of the algorithm. If (15) holds, we think that the iterates are approaching the feasible region. In this case, we choose to update the multiplier with the latest algorithmic information. Otherwise, the new iterate is thought to be relatively far away from the feasible region. Then, we pay our attention to obtaining a point with lower constraint violation, rather than giving priority to updating the multiplier.

In augmented Lagrangian methods, $\boldsymbol{\lambda}_k$ plays a very important role in guiding the algorithm to the solution of (1). A good update strategy will speed up the algorithm a lot. In standard augmented Lagrangian methods, as the new iterate $\boldsymbol{x}_{k+1}$ obtained by solving (6) satisfies the equation

$$\boldsymbol{g}_{k+1} - \mathbf{A}_{k+1}^{\mathsf{T}} \boldsymbol{\lambda}_k + \sigma_k \mathbf{A}_{k+1}^{\mathsf{T}} \boldsymbol{c}_{k+1} = \mathbf{0}, \tag{16}$$

(4) suggests that a good estimate of $\boldsymbol{\lambda}_{k+1}$ is $\boldsymbol{\lambda}_k - \sigma_k \boldsymbol{c}_{k+1}$. However, this estimate is unsuitable in our method, because $\boldsymbol{x}_{k+1}$ is obtained from the approximate subproblem (8), which does not imply (16). As the standard update $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k - \sigma_k \boldsymbol{c}_{k+1}$ can not be used here, we obtain $\boldsymbol{\lambda}_{k+1}$ through minimizing $\|\boldsymbol{g}_{k+1} - \mathbf{A}_{k+1}^{\mathsf{T}} \boldsymbol{\lambda}\|$.

In addition, Lagrange multipliers are normally required to be bounded or not to grow too fast compared with the penalty parameter σ_k (see, e.g. [8, Lemma 4.1–4.2]). A similar requirement is

needed in our method. To ensure global convergence, the ratio $\boldsymbol{\lambda}_k/\sigma_k$ needs to approach zero vector when σ_k increases to infinity. However, if we simply define $\boldsymbol{\lambda}_k = \arg\min_{\boldsymbol{\lambda}\in\mathbb{R}^m} \|\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}\|^2$, we can not guarantee that $\tilde{\boldsymbol{\lambda}}_k/\sigma_k$ will converge to zero as σ_k goes to infinity. So we introduce an auxiliary vector $\tilde{\boldsymbol{\lambda}}_k$ which is defined as

$$\tilde{\boldsymbol{\lambda}}_k = \arg\min_{\boldsymbol{\lambda}\in\mathbb{R}^m} \|\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}\|^2. \tag{17}$$

The easiest way to obtain an acceptable $\boldsymbol{\lambda}_k$ is to set a boundedness condition on it. Therefore, after obtaining $\tilde{\boldsymbol{\lambda}}_k$, we project it onto a compact box $[\boldsymbol{\lambda}_{\min}, \boldsymbol{\lambda}_{\max}]$ to get $\boldsymbol{\lambda}_k$, namely,

$$\boldsymbol{\lambda}_k = P_{[\boldsymbol{\lambda}_{\min},\boldsymbol{\lambda}_{\max}]}\tilde{\boldsymbol{\lambda}}_k. \tag{18}$$

This projection seems reasonable, as a bounded optimal multiplier $\boldsymbol{\lambda}_*$ exists at the solution $\boldsymbol{x}_*$ of the original problem (1). For a globally convergent method with $\boldsymbol{\lambda}_k$ converging to $\boldsymbol{\lambda}_*$, the boundedness of $\boldsymbol{\lambda}_k$ is natural. Although we choose $\tilde{\boldsymbol{\lambda}}_k$ as defined in (17) in the following ALTR algorithm, any update strategy for $\tilde{\boldsymbol{\lambda}}_k$ will not destroy the global convergence property presented in the following section, provided that $\boldsymbol{\lambda}_k$ stays bounded.

2.3 *Algorithm framework*

The above discussions are summarized in the following ALTR algorithm for solving the equality constrained optimization problem (1).

In ALTR algorithm, we introduce an update of penalty parameters (19) to move the iterates away from infeasible local minimizers of the augmented Lagrangian function. The following lemma shows a property of $\boldsymbol{x}_k$ if the loop (19) is an infinite loop at $\boldsymbol{x}_k$.

LEMMA 2.5 *Suppose that $\boldsymbol{x}_k$ is an iterate satisfying $\nabla_x L(\boldsymbol{x}_k; \boldsymbol{\lambda}_k, \sigma_k^j) = 0$ with σ_k^j updated by (19) in an infinite loop. Then $\boldsymbol{x}_k$ is an infeasible KKT point of*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{c}(\boldsymbol{x})\|^2. \tag{21}$$

Proof The KKT point x of (21) is characterized by $\mathbf{A}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{c}(\boldsymbol{x}) = \mathbf{0}$. As j increases to ∞, $\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k + \sigma_k^j\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k = \nabla_x L(\boldsymbol{x}_k; \boldsymbol{\lambda}_k, \sigma_k^j) = \mathbf{0}$. Since $\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k$ is bounded, it implies that $\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k = \mathbf{0}$. Therefore, due to $\|\boldsymbol{c}_k\| > 0$, $\boldsymbol{x}_k$ is an infeasible KKT point of (21). ■

Theoretically, (19) may be trapped in an infinite loop at each iteration. However, in practical computation, we could not allow this situation to happen. So we usually set a maximum number of j to test whether $\nabla_x L(\boldsymbol{x}_k; \boldsymbol{\lambda}_k, \sigma_k^j) = \mathbf{0}$ or not. If it fails, we return $\boldsymbol{x}_k$ as an approximate solution. We thus in the following context assume that (19) finishes in a finite loop at each iteration. Once this loop terminates finitely, we have $\nabla_x L(\boldsymbol{x}_k; \boldsymbol{\lambda}_k, \sigma_k) \neq \mathbf{0}$. Then the subproblem (8) is solved. It is easy to see that (8) always generates a trial step $\boldsymbol{s}_k$ such that $\rho_k \geq \eta$, provided that the trust region radius Δ_k is sufficiently small (see, e.g. [10]). Therefore, the loop between Steps 2 and 3 in ALTR algorithm will stop in finite number of inner iterations. This implies that ALTR algorithm is well-posed.

3. Global convergence

We assume that ALTR algorithm generates an infinite sequence of iterates $\{\boldsymbol{x}_k\}$. In this section, we study the convergence properties of ALTR algorithm. We first give the following assumptions required throughout this paper.

Algorithm 1 AUGMENTED LAGRANGIAN TRUST REGION METHOD (ALTR)

Step 0: Initialization. Given constants $\beta \in (0,1)$, $\epsilon > 0$, $\theta > 1$, $\boldsymbol{\lambda}_{min} < \boldsymbol{\lambda}_{max}$ and

$$0 < \eta < \eta_1 < \frac{1}{2}, \quad R_0 = \max\{\|\boldsymbol{c}(\boldsymbol{x}_0)\|, 1\}.$$

Given $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\mathbf{B}_0 \in \mathbb{R}^{n\times n}$, $\boldsymbol{\lambda}_0 \in \mathbb{R}^m$, $\sigma_0 > 1$, $\delta_0 > 0$, $\Delta_0 > 0$; Set $j := 0$, $k := 0$.

Step 1: Termination Test. If $\|\boldsymbol{c}_k\| = 0$ and $P_{\mathcal{N}_k}(\boldsymbol{g}_k) = 0$, stop (return $\boldsymbol{x}_k$ as solution).

Step 2: Computing Trial Steps. Set $\sigma_k^0 := \sigma_k$ and $j := 0$;

While $\nabla_x L(\boldsymbol{x}_k; \boldsymbol{\lambda}_k, \sigma_k^j) = 0$ and $\|\boldsymbol{c}_k\| > 0$,

$$\sigma_k^{j+1} := \theta\sigma_k^j;\ j := j+1 \tag{19}$$

Endwhile; Set $\sigma_k = \sigma_k^j$; Solve (8) obtaining $\boldsymbol{s}_k$;

Step 3: Update of Iterates. Compute the ratio ρ_k in (10); If $\rho_k \geq \eta$, go to Step 4;

$\Delta_{k+1} = \|\boldsymbol{s}_k\|/4$; $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$; $k := k+1$; go to Step 2;

Step 4: Update of Penalty Parameters and Multipliers. If (14) is satisfied, then set

$$\sigma_{k+1} = 2\sigma_k; \quad \delta_{k+1} = \delta_k/4; \tag{20}$$

else $\sigma_{k+1} = \sigma_k$, $\delta_{k+1} = \delta_k$;

If $\|\boldsymbol{c}_{k+1}\| \leq R_k$, then compute $\tilde{\boldsymbol{\lambda}}_{k+1}$ through (18) and set $R_{k+1} = \beta R_k$;

otherwise, set $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k$ and $R_{k+1} = R_k$;

Step 5: Update of Trust Region Radii. Set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$ and

$$\Delta_{k+1} = \begin{cases} \max\{\Delta_k, 1.5\|\boldsymbol{s}_k\|\}, & \text{if} \rho_k \in [1-\eta_1, +\infty), \\ \Delta_k, & \text{if } \rho_k \in [\eta_1, 1-\eta_1), \\ \max\{0.5\Delta_k, 0.75\|\boldsymbol{s}_k\|\}, & \text{if } \rho_k \in [\eta, \eta_1); \end{cases}$$

Calculate f_{k+1}, $\boldsymbol{g}_{k+1}$, $\boldsymbol{c}_{k+1}$ and $\mathbf{A}_{k+1}$; Generate $\mathbf{B}_{k+1}$;

$k := k+1$, then go to Step 1.

AS.1 *$f(\boldsymbol{x})$ and $\boldsymbol{c}(\boldsymbol{x})$ are Lipschitz continuously differentiable.*

AS.2 *$\{\boldsymbol{x}_k\}$ and $\{\mathbf{B}_k\}$ are bounded.*

It is reasonable to assume the boundedness of $\{\boldsymbol{x}_k\}$, as otherwise we are allowing the possibility that the original problem has no bounded solution. For $\{\mathbf{B}_k\}$, we can ensure its boundedness unless a numerical overflow happens.

From the construction of ALTR algorithm, we could not exclude the case that the penalty parameter σ_k increases to infinity. Thus the following analysis is separated into two parts. Firstly, we analyse the theoretical properties of ALTR algorithm in the case when σ_k increases to infinity, while the case with bounded σ_k is addressed later.

3.1 *Convergence properties with unbounded penalty parameters*

When the penalty parameter σ_k increases to infinity, we find that the sequence of constraint violations $\{\|\boldsymbol{c}_k\|\}$ is convergent, as stated and proved in the following lemma.

LEMMA 3.1 *Under assumptions* AS.1–AS.2, *if* $\sigma_k \to \infty$, *then* $\lim_{k\to\infty} \|\boldsymbol{c}_k\|$ *exists.*

Proof The penalty parameters σ_k, $k = 1, 2, 3, \ldots$, are positive and increase monotonically. Therefore the numbers $\sigma_i^{-1} - \sigma_{i+1}^{-1}$, $i \geq 1$, are all nonnegative, and, for all integers p and q that satisfy $0 < p < q$, their sum has the property

$$\sum_{i=p}^{q-1} (\sigma_i^{-1} - \sigma_{i+1}^{-1}) = \sigma_p^{-1} - \sigma_q^{-1}. \tag{22}$$

Thus we deduce the bound

$$\begin{aligned}\sum_{i=p}^{q} \sigma_i^{-1}[f(\boldsymbol{x}_i) - f(\boldsymbol{x}_{i+1})] &= \sigma_p^{-1} f(\boldsymbol{x}_p) + \sum_{i=p}^{q-1} (-\sigma_i^{-1} + \sigma_{i+1}^{-1}) f(\boldsymbol{x}_{i+1}) - \sigma_q^{-1} f(\boldsymbol{x}_{q+1}) \\ &\leq \sigma_p^{-1} f_{\max} + (\sigma_p^{-1} - \sigma_q^{-1}) f_{\max} + \sigma_q^{-1} f_{\max} = 2\sigma_p^{-1} f_{\max}, \end{aligned} \tag{23}$$

where $f_{\max}$ is an upper bound on $|f(\boldsymbol{x}_k)|$, $k = 1, 2, 3, \ldots$.

We also require a bound on the sum

$$\begin{aligned}\sum_{i=p}^{q} \sigma_i^{-1}[\boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1}) - \boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_i)] &= -\sigma_p^{-1} \boldsymbol{\lambda}_p^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_p) + \sigma_q^{-1} \boldsymbol{\lambda}_q^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{q+1}) \\ &\quad + \sum_{i=p}^{q-1} \{\sigma_i^{-1} \boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1}) - \sigma_{i+1}^{-1} \boldsymbol{\lambda}_{i+1}^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1})\}. \end{aligned} \tag{24}$$

We give special attention to the integers i such that $\boldsymbol{\lambda}_{i+1} \neq \boldsymbol{\lambda}_i$. We let $\mathcal{I}(p, q)$ be the set of such integers in the interval $[p, q-1]$, but this set may be empty.

The algorithm provides the condition

$$\begin{aligned}\sum_{i \in \mathcal{I}(p,q)} \{\sigma_i^{-1} \boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1}) - \sigma_{i+1}^{-1} \boldsymbol{\lambda}_{i+1}^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1})\} &\leq \sum_{i \in \mathcal{I}(p,q)} \{\sigma_i^{-1} \|\boldsymbol{\lambda}_i\| + \sigma_{i+1}^{-1} \|\boldsymbol{\lambda}_{i+1}\|\} \|\boldsymbol{c}(\boldsymbol{x}_{i+1})\| \\ &\leq 2\sigma_p^{-1} \|\boldsymbol{\lambda}_{\max}\| \sum_{i \in \mathcal{I}(p,q)} \|\boldsymbol{c}(\boldsymbol{x}_{i+1})\| \\ &\leq 2\sigma_p^{-1} \|\boldsymbol{\lambda}_{\max}\| \frac{R_0}{(1-\beta)}, \end{aligned}$$

where $\|\boldsymbol{\lambda}_{\max}\|$ is an upper bound on every $\|\boldsymbol{\lambda}_k\|$. For all the other terms in the second line of expression (24), the vectors $\boldsymbol{\lambda}_i$ and $\boldsymbol{\lambda}_{i+1}$ are the same, which gives the inequality

$$\sigma_i^{-1} \boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1}) - \sigma_{i+1}^{-1} \boldsymbol{\lambda}_{i+1}^{\mathsf{T}} c(\boldsymbol{x}_{i+1}) \leq (\sigma_i^{-1} - \sigma_{i+1}^{-1}) \|\boldsymbol{\lambda}_{\max}\| \, \|\boldsymbol{c}_{\max}\|. \tag{25}$$

Therefore, with the help of Equations (22) and (24), we find the bound

$$\begin{aligned}\sum_{i=p}^{q} \sigma_i^{-1}[\boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_{i+1}) - \boldsymbol{\lambda}_i^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{x}_i)] &\leq \sigma_p^{-1} \|\boldsymbol{\lambda}_{\max}\| \, \|\boldsymbol{c}_{\max}\| + \sigma_q^{-1} \|\boldsymbol{\lambda}_{\max}\| \, \|\boldsymbol{c}_{\max}\| \\ &\quad + 2\sigma_p^{-1} \|\boldsymbol{\lambda}_{\max}\| \frac{R_0}{(1-\beta)} + (\sigma_p^{-1} - \sigma_q^{-1}) \|\boldsymbol{\lambda}_{\max}\| \, \|\boldsymbol{c}_{\max}\| \\ &= 2\sigma_p^{-1} \|\boldsymbol{\lambda}_{\max}\| \left\{ \|\boldsymbol{c}_{\max}\| + \frac{R_0}{(1-\beta)} \right\}. \end{aligned} \tag{26}$$

Now, the algorithm provides $L(\boldsymbol{x}_{k+1}; \boldsymbol{\lambda}_i, \sigma_i) \leq L(\boldsymbol{x}_k; \boldsymbol{\lambda}_i, \sigma_i)$ on every iteration, because $\boldsymbol{x}_{k+1}$ is different from $\boldsymbol{x}_k$ only if Ared_k is positive. Inequalities (23) and (26) with the definition (5) imply

the property

$$\sum_{i=p}^{q} \sigma_i^{-1}\{L(\boldsymbol{x}_i;\boldsymbol{\lambda}_i,\sigma_i) - L(\boldsymbol{x}_{i+1};\boldsymbol{\lambda}_i,\sigma_i)\} \le \frac{1}{2}\|c(\boldsymbol{x}_p)\|^2 - \frac{1}{2}\|\boldsymbol{c}(\boldsymbol{x}_{q+1})\|^2 + M_0\sigma_p^{-1}, \tag{27}$$

where M_0 is the constant

$$M_0 = 2f_{\max} + 2\|\boldsymbol{\lambda}_{\max}\|\left\{\|\boldsymbol{c}_{\max}\| + \frac{R_0}{(1-\beta)}\right\}.$$

We see that the sum on the left-hand side of inequality (27) is bounded above by $\frac{1}{2}\|\boldsymbol{c}(\boldsymbol{x}_p)\|^2 + M_0\sigma_p^{-1}$. By letting q become infinite for any fixed p, it follows that the sum of the products $\sigma_k^{-1}\{L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k) - L(\boldsymbol{x}_{k+1};\boldsymbol{\lambda}_k,\sigma_k)\}$, $k = 1, 2, 3, \ldots$, is absolutely convergent. Furthermore, the nonnegativity of the left-hand side of inequality (27) supplies the condition

$$\|\boldsymbol{c}(\boldsymbol{x}_{q+1})\|^2 \le \|\boldsymbol{c}(\boldsymbol{x}_p)\|^2 + 2M_0\sigma_p^{-1}, \quad 0 < p < q. \tag{28}$$

Therefore, by letting q become infinite again with p fixed, we find that the sequence $\|\boldsymbol{c}(\boldsymbol{x}_k)\|^2$, $k = 1, 2, 3, \ldots$, is bounded.

Let its liminf be $\|\boldsymbol{c}_\infty\|^2$, and, for any $\varepsilon > 0$, let p satisfy $\|\boldsymbol{c}(\boldsymbol{x}_p)\|^2 < \|\boldsymbol{c}_\infty\|^2 + \varepsilon$. Because of the assumption $\sigma_k \to \infty$, the choice of p can also satisfy $2M_0\sigma_p^{-1} < \varepsilon$. It follows that condition (28) gives the bound

$$\|\boldsymbol{c}(\boldsymbol{x}_{q+1})\|^2 < \|\boldsymbol{c}_\infty\|^2 + 2\varepsilon, \quad q > p,$$

showing that limsup of the sequence $\|\boldsymbol{c}(\boldsymbol{x}_k)\|^2$ as $k \to \infty$ is at most $\|\boldsymbol{c}_\infty\|^2 + 2\varepsilon$. Thus, because the positive number ε can be arbitrarily small, the lim inf and lim sup limits of $\|\boldsymbol{c}(\boldsymbol{x}_k)\|^2$, $k = 1, 2, 3, \ldots$, are the same. The proof is complete. ■

Lemma 3.1 ensures the convergence of $\{\|\boldsymbol{c}_k\|\}$, as $\sigma_k \to \infty$. Hence, two cases arise: either all the accumulation points of $\{\boldsymbol{x}_k\}$ are infeasible, or all the accumulation points are feasible. We welcome methods that can find feasible accumulation points, but that is impossible if the original problem is 'naturally' infeasible, for example, $\boldsymbol{c}(\boldsymbol{x}) \neq 0$ for any $\boldsymbol{x} \in \mathbb{R}^n$. Therefore, it is necessary to study how the algorithm behaves in terms of infeasibility.

THEOREM 3.2 *Under assumptions* AS.1–AS.2, *if* $\lim_{k\to\infty} \sigma_k = \infty$ *and* $\lim_{k\to\infty} \|\boldsymbol{c}_k\| > 0$, *then any accumulation point of* $\{\boldsymbol{x}_k\}$ *is a KKT point of* (21).

Proof To prove the theorem, we have to establish $\lim_{k\to\infty} \|\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k\| = 0$. Firstly we prove

$$\liminf_{k\to\infty} \|\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k\| = 0. \tag{29}$$

By contradiction, we assume that (29) were not true. Then there exists a constant $\mu > 0$ such that

$$\liminf_{k\to\infty} \|\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k\| \ge 2\mu. \tag{30}$$

Denote $\bar{\boldsymbol{g}}_k = \nabla_x L(\boldsymbol{x}_k;\boldsymbol{\lambda}_k,\sigma_k)$. Due to $\sigma_k \to \infty$ and (30), there exists k_0 such that the following property holds:

$$\|\bar{\boldsymbol{g}}_k\| \ge \sigma_k\|\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k\| - \|\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k\| \ge \mu\sigma_k, \quad \text{for all } k \ge k_0. \tag{31}$$

Therefore, (19) occurs at most finite number of iterations. Without loss of generality, we assume that (19) never happens. Then $\sigma_k \to \infty$ is only caused by (20) happening infinitely many times.

Note that the subproblem (8) can be rewritten as

$$\begin{aligned} \min_{\boldsymbol{d}\in\mathbb{R}^n} \quad & q_k(\boldsymbol{d}) = \bar{\boldsymbol{g}}_k^{\mathsf{T}}\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^{\mathsf{T}}(\mathbf{B}_k + \sigma_k\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k)\boldsymbol{d} + \frac{\sigma_k}{2}\|\boldsymbol{c}_k\|^2 \\ \text{s. t.} \quad & \|\boldsymbol{d}\| \le \Delta_k. \end{aligned} \tag{32}$$

Then, the predicted reduction achieved at the exact solution of (32) satisfies (see, e.g. Chapter 4 in [25]) the following inequality

$$q_k(\mathbf{0}) - \min_{\|\boldsymbol{d}\|\le\Delta_k} q_k(\boldsymbol{d}) \ge \frac{1}{2}\|\bar{\boldsymbol{g}}_k\| \min\left\{\frac{\|\bar{\boldsymbol{g}}_k\|}{\|\mathbf{B}_k + \sigma_k\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k\|}, \Delta_k\right\}. \tag{33}$$

It follows from AS.1 and AS.2 that there exists a positive constant M such that $\|\mathbf{B}_k + \sigma_k\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k\| \le M\sigma_k$. Then (33) gives the lower bound

$$q_k(\mathbf{0}) - \min_{\|\boldsymbol{d}\|\le\Delta_k} q_k(\boldsymbol{d}) \ge \frac{1}{2}\|\bar{\boldsymbol{g}}_k\| \min\left\{\frac{\|\bar{\boldsymbol{g}}_k\|}{M\sigma_k}, \Delta_k\right\} \ge \frac{\mu\sigma_k}{2}\min\left\{\frac{\mu}{M}, \Delta_k\right\},$$

where the second inequality is due to (31). Then (9) implies that the predicted reduction Pred_k along $\boldsymbol{s}_k$ satisfies the following lower bound:

$$\mathrm{Pred}_k \ge \frac{\bar{\beta}\mu\sigma_k}{2}\min\left\{\frac{\mu}{M}, \Delta_k\right\}, \quad \text{for all } k \ge k_0. \tag{34}$$

However, $\sigma_k \to \infty$ provides the conditions that $\delta_k\sigma_k \to 0$ and $\mathrm{Pred}_k < \delta_k\sigma_k \min\{\Delta_k\|\boldsymbol{c}_k\|, \|\boldsymbol{c}_k\|^2\}$ holds for infinitely many k. This contradicts (34). Hence, (29) holds.

We now assume that there exist an accumulation point $\bar{\boldsymbol{x}}$ of $\{\boldsymbol{x}_k\}$ and a subset $\mathcal{K}$ such that $\{\boldsymbol{x}_k\}_{\mathcal{K}} \to \bar{\boldsymbol{x}}$ and $\|\mathbf{A}(\bar{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{c}(\bar{\boldsymbol{x}})\| \ne 0$. Then a constant $\eta > 0$ is existent such that

$$\min_{\|d\|\le 1} \|\boldsymbol{c}(\bar{\boldsymbol{x}}) + \mathbf{A}(\bar{\boldsymbol{x}})\boldsymbol{d}\|^2 \le \|\boldsymbol{c}(\bar{\boldsymbol{x}})\|^2 - \eta.$$

Therefore, there exists $\epsilon \in (0, 1)$ such that, for all $\boldsymbol{x}_k$ satisfying $\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}\| \le \epsilon$

$$\min_{\|\boldsymbol{d}\|\le 1} \|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2 \le \|\boldsymbol{c}_k\|^2 - 0.9\eta,$$

which gives the inequality

$$\min_{\|\boldsymbol{d}\|\le\Delta_k} \|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2 \le \|\boldsymbol{c}_k\|^2 - 0.9\min(1, \Delta_k)\eta \tag{35}$$

due to the convexity of $\|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2$. Denote $\boldsymbol{d}_k$ as the minimizer of $\min\{\|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}\|^2 \mid \|\boldsymbol{d}\| \le \Delta_k\}$. Then the predicted reduction has the lower bound

$$\begin{aligned} \mathrm{Pred}_k &\ge \bar{\beta}[q_k(\mathbf{0}) - q_k(\boldsymbol{d}_k)] \\ &= \bar{\beta}\left[-(\boldsymbol{g}_k^{\mathsf{T}}\boldsymbol{d}_k - (\mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k)^{\mathsf{T}}\boldsymbol{d}_k + \frac{1}{2}\boldsymbol{d}_k^{\mathsf{T}}\mathbf{B}_k\boldsymbol{d}_k) + \frac{\sigma_k}{2}(\|\boldsymbol{c}_k\|^2 - \|\boldsymbol{c}_k + \mathbf{A}_k\boldsymbol{d}_k\|^2)\right] \\ &\ge \mathrm{O}(\Delta_k) + \frac{\beta\sigma_k}{2}0.9\min(1, \Delta_k)\eta \\ &\ge \frac{\beta\sigma_k}{4}\eta\min(1, \Delta_k), \quad \text{for all large } k \in \mathcal{K}, \end{aligned}$$

as $\sigma_k \to \infty$. (35) also indicates that $\{\Delta_k\}_{\mathcal{K}} \to 0$ due to the fact that $\lim_{k\to\infty}\|\boldsymbol{c}_k\|$ exists, which implies that $\{\rho_k\}_{\mathcal{K}} \to 1$, as $k \to \infty$. Hence, there are infinitely many $k \in \mathcal{K}$ such that $\Delta_k \le \epsilon$

and $\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}\| \le \frac{1}{4}\epsilon$. For those k, the following property holds:

$$\|\boldsymbol{c}_k + \mathbf{A}_k \boldsymbol{s}_k\|^2 \le \|\boldsymbol{c}_k\|^2 - 0.8\eta\Delta_k,$$

which shows the relation between $\|\boldsymbol{c}_k\|$ and $\|\boldsymbol{c}_{k+1}\|$:

$$\|\boldsymbol{c}_{k+1}\|^2 \le \|\boldsymbol{c}_k\|^2 - 0.7\eta\Delta_k. \tag{36}$$

We now choose an arbitrary $\hat{k} \in \mathcal{K}$ such that $\|\boldsymbol{x}_{\hat{k}} - \bar{\boldsymbol{x}}\| \le \epsilon/8$ and $\Delta_{\hat{k}} \le \epsilon$. Let p be the smallest nonnegative integer such that $\|\boldsymbol{x}_{\hat{k}+p+1} - \bar{\boldsymbol{x}}\| \ge \epsilon/4$. Thus, for $i = 0, \ldots, p$, we have $\|\boldsymbol{x}_{\hat{k}+i} - \bar{\boldsymbol{x}}\| \le \epsilon/4$, which indicates that

$$\|\boldsymbol{s}_{\hat{k}+i}\| \le \|\boldsymbol{x}_{\hat{k}+i+1} - \bar{\boldsymbol{x}}\| + \|\boldsymbol{x}_{\hat{k}+i} - \bar{\boldsymbol{x}}\| \le \frac{1}{2}\epsilon, \quad \text{for } 0 \le i \le p-1.$$

The above bound and the update rule of trust region radii show that

$$\Delta_{\hat{k}+i} \le 1.5\|\boldsymbol{s}_{\hat{k}+i}\| \le \epsilon, \quad \text{for } 0 \le i \le p.$$

Hence, (36) holds for $k = \hat{k} + i$, $(i = 0, \ldots, p)$. Thus, we obtain

$$\begin{aligned}
\|\boldsymbol{c}_{\hat{k}+i+1}\| &\le \|\boldsymbol{c}_{\hat{k}}\|^2 - 0.7\eta \sum_{i=0}^{p} \Delta_{\hat{k}+i} \\
&\le \|\boldsymbol{c}_{\hat{k}}\|^2 - 0.7\eta \sum_{i=0}^{p} \|\boldsymbol{s}_{\hat{k}+i}\| \\
&\le \|\boldsymbol{c}_{\hat{k}}\|^2 - 0.7\eta \|\boldsymbol{x}_{\hat{k}+p+1} - \boldsymbol{x}_{\hat{k}}\| \\
&\le \|\boldsymbol{c}_{\hat{k}}\|^2 - 0.7\eta [\|\boldsymbol{x}_{\hat{k}+p+1} - \bar{\boldsymbol{x}}\| - \|\boldsymbol{x}_{\hat{k}} - \bar{\boldsymbol{x}}\|] \\
&\le \|\boldsymbol{c}_{\hat{k}}\|^2 - 0.7\eta \cdot \frac{1}{8}\epsilon,
\end{aligned}$$

which contradicts Lemma 3.1 as there are infinitely many such $\hat{k}$. Therefore, $\lim_{k\to\infty} \|\mathbf{A}_k^{\mathrm{T}} \boldsymbol{c}_k\| = 0$, which completes the proof. ■

We now give the convergence property of ALTR algorithm when $\sigma_k \to \infty$ and $\|\boldsymbol{c}_k\| \to 0$.

THEOREM 3.3 *Let assumptions* AS.1–AS.2 *hold. If* $\lim_{k\to\infty} \sigma_k = \infty$ *and* $\lim_{k\to\infty} \|\boldsymbol{c}_k\| = 0$, *then the sequence of iterates* $\{\boldsymbol{x}_k\}$ *is not bounded away from KKT points of* (1), *or its Fritz-John points at which the RCPLD condition fails to hold.*

Proof Let $\bar{\boldsymbol{x}}$ be any accumulation point of $\{\boldsymbol{x}_k\}$. Then $\boldsymbol{c}(\bar{\boldsymbol{x}}) = 0$. If the RCPLD condition fails to hold at $\bar{\boldsymbol{x}}$, then all the gradients of constraints at $\bar{\boldsymbol{x}}$ are linearly dependent. Thus according to the definition of the RCPLD condition, we obtain that $\bar{\boldsymbol{x}}$ is a Fritz-John point. We next study the case when the RCPLD condition holds at accumulation points.

As $\sigma_k \to \infty$ and the increase of σ_k is due to (19) or (and) (20), two possible cases may happen.

Case 1 Update (19) occurs in only finitely many iterations. Then for all large k, only update (20) happens in the kth iteration. Without loss of generality, we assume that (19) never happens.

Firstly, we prove that for any $\varepsilon > 0$, there exists $k = k(\varepsilon)$, such that

$$\|\boldsymbol{c}_k\| < \varepsilon \quad \text{and} \quad \|P_{\mathcal{N}_k}(\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k)\| < \varepsilon. \tag{37}$$

Suppose (37) were not true. Then, due to $\lim_{k\to\infty}\|\boldsymbol{c}_k\| = 0$, there exists $\bar{\varepsilon} > 0$ such that for all sufficiently large k,

$$\|\boldsymbol{c}_k\| < \bar{\varepsilon} \quad \text{and} \quad \|P_{\mathcal{N}_k}(\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_{\boldsymbol{k}})\| \geq \bar{\varepsilon}.$$

Hence, the predicted reduction satisfies the lower bound

$$\begin{aligned}\text{Pred}_k &\geq \bar{\beta}\xi_1\|P_{\mathcal{N}_k}[\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k]\|\min\{\xi_2\|P_{\mathcal{N}_k}[\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k]\|, \Delta_k\}\\ &\geq \bar{\beta}\xi_1\bar{\varepsilon}\min\{\xi_2\bar{\varepsilon}, \Delta_k\} \geq \bar{\nu}\min\{\Delta_k\|\boldsymbol{c}_k\|, \|\boldsymbol{c}_k\|^2\}\end{aligned}$$

with some positive constant $\bar{\nu}$. However, $\sigma_k \to \infty$ indicates that

$$\text{Pred}_k \leq \sigma_k\delta_k\min\{\Delta_k\|\boldsymbol{c}_k\|, \|\boldsymbol{c}_k\|^2\}$$

holds for infinitely many k with $\sigma_k\delta_k \to 0$ as $k \to \infty$. This is a contradiction. As a result, for any $\varepsilon > 0$, there exists k such that (37) holds. Consequently, by forcing $\varepsilon \to 0$ we obtain a subsequence $\mathcal{K}$ such that

$$\lim_{\substack{k\to\infty\\ k\in\mathcal{K}}}\|\boldsymbol{c}_k\| = 0 \quad \text{and} \quad \lim_{\substack{k\to\infty\\ k\in\mathcal{K}}}\|P_{\mathcal{N}_k}(\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k)\| = 0.$$

Let $\boldsymbol{x}_*$ be any accumulation point of $\{\boldsymbol{x}_k\}_{\mathcal{K}}$. Then there exists a subset $\mathcal{K}_1 \subseteq \mathcal{K}$ such that $\{\boldsymbol{x}_k\}_{\mathcal{K}_1} \to \boldsymbol{x}_*$. Assume that the RCPLD condition holds at $\boldsymbol{x}_*$. We next prove that $\boldsymbol{x}_*$ is a KKT point of (1).

Denote $\{\nabla c^{(i)}(\boldsymbol{x}_*)\}_{i\in I}$ as the maximal set of linearly independent vectors, among all the gradients of constraints at $\boldsymbol{x}_*$. Then by RCPLD and continuity of constraint functions, there exists a neighbourhood of $\boldsymbol{x}_*$ such that for any point $\boldsymbol{x}$ in this neighbourhood, $\{\nabla c^{(i)}(\boldsymbol{x})\}_{i\in I}$ are linearly independent and

$$\text{span}\{\nabla c^{(i)}(\boldsymbol{x}),\ i = 1, \ldots, m\} = \text{span}\{\nabla c^{(i)}(\boldsymbol{x}),\ i \in I\}.$$

Then an index k_0 is existent such that for all $k_0 < k \in \mathcal{K}$, $\{\nabla c^{(i)}(\boldsymbol{x}_k)\}_{i\in I}$ are linearly independent and

$$\text{Range}(\mathbf{A}_k^{\mathsf{T}}) = \text{span}\{\nabla c^{(i)}(\boldsymbol{x}_k),\ i = 1, \ldots, m\} = \text{span}\{\nabla c_{(i)}(\boldsymbol{x}_k),\ i \in I\}. \tag{38}$$

We now introduce a new matrix $\bar{\mathbf{A}}_k^{\mathsf{T}}$, whose columns consist of the vectors $\nabla c^{(i)}(\boldsymbol{x}_k)$, $i \in I$. We denote it as $\bar{\mathbf{A}}_k^{\mathsf{T}} = (\nabla c_{(i)}(\boldsymbol{x}_k))_{i\in I}$. Similarly, we denote $\bar{\mathbf{A}}_*^{\mathsf{T}}$ as $\bar{\mathbf{A}}_*^{\mathsf{T}} = (\nabla c_{(i)}(\boldsymbol{x}_*))_{i\in I}$. Because $\bar{\mathbf{A}}_*^{\mathsf{T}}$ has full column rank and $\{\boldsymbol{x}_k\}_{\mathcal{K}_1} \to \boldsymbol{x}_*$, there exist $k_1 \geq k_0$ and M such that $\bar{\mathbf{A}}_k^{\mathsf{T}}$ has full column rank and

$$\|(\bar{\mathbf{A}}_k^{\mathsf{T}})^+\| \leq M, \quad \forall\, k_1 \leq k \in \mathcal{K}_1. \tag{39}$$

Since for any k, there exist $\boldsymbol{\mu}_k$ and $\boldsymbol{y}_k$ such that $\boldsymbol{g}_k - \boldsymbol{\mu}_k = \bar{\mathbf{A}}_k^{\mathsf{T}}\boldsymbol{y}_k$, (39) implies that $\boldsymbol{\mu}_k \to \mathbf{0}$ as $k \in \mathcal{K}_1$, $k \to \infty$. Due to $\boldsymbol{g}_k \to \boldsymbol{g}_*$ and $\bar{\mathbf{A}}_k \to \bar{\mathbf{A}}_*$ as $k \in \mathcal{K}_1$, $k \to \infty$, it follows that $\boldsymbol{y}_*$ is existent such that $\boldsymbol{y}_k \to \boldsymbol{y}_*$ as $k \in \mathcal{K}_1$, $k \to \infty$. Therefore, we obtain $\boldsymbol{g}_* = \bar{\mathbf{A}}_*^{\mathsf{T}}\boldsymbol{y}_*$. This shows that $\boldsymbol{g}_* \in \text{Span}\{\nabla c_{(i)}(\boldsymbol{x}_*), i \in \mathcal{I}\}$, which indicates that $\boldsymbol{g}_* \in \text{Range}(A_*^{\mathsf{T}})$. Therefore, it follows from $\boldsymbol{c}_* = \mathbf{0}$ that $\boldsymbol{x}_*$ is a KKT point of (1).

Case 2 Update (19) occurs in infinitely many iterations.

In this case, from the update strategy (19), we know that there exist subsequences $\{\boldsymbol{x}_k\}_{\mathcal{K}}$ and $\{\bar{\sigma}_k\}_{\mathcal{K}}$ such that

$$\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k + \bar{\sigma}_k \mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k = \mathbf{0}, \quad k \in \mathcal{K}. \tag{40}$$

Suppose $\boldsymbol{x}_*$ is an accumulation point of $\{\boldsymbol{x}_k\}_{\mathcal{K}}$ and $\{\boldsymbol{x}_k\}_{\mathcal{K}_1} \to \boldsymbol{x}_*$ where $\mathcal{K}_1 \subseteq \mathcal{K}$. If the RCPLD condition holds at $\boldsymbol{x} = \boldsymbol{x}_*$, for any sufficiently large $k \in \mathcal{K}_1$, similar to the discussion in Case 1, there exists $\bar{\mathbf{A}}_k$ which has full row rank such that

$$\text{Range}(\mathbf{A}_k^{\mathsf{T}}) = \text{Range}(\bar{\mathbf{A}}_k^{\mathsf{T}}).$$

As (40) indicates that $\boldsymbol{g}_k \in \text{Range}(\mathbf{A}_k^{\mathsf{T}})$, consequently $\boldsymbol{g}_k \in \text{Range}(\bar{\mathbf{A}}_k^{\mathsf{T}})$. Therefore, $\boldsymbol{g}_* \in \text{Range}(\bar{\mathbf{A}}_*^{\mathsf{T}})$ due to $\{\boldsymbol{x}_k\}_{\mathcal{K}_1} \to \boldsymbol{x}_*$. It follows from $\boldsymbol{c}_* = \mathbf{0}$ that $\boldsymbol{x}_*$ is a KKT point. ■

3.2 *Convergence properties with bounded penalty parameters*

We now investigate the theoretical properties of ALTR algorithm when all the penalty parameters are bounded. In this case, σ_k keeps unchanged for all large k, equivalently, neither (19) nor (20) happens. Without loss of generality, we assume that

$$\sigma_k = \sigma \quad \text{for all } k. \tag{41}$$

Firstly, we show that all the accumulation points are feasible if $\{\sigma_k\}$ is bounded.

LEMMA 3.4 *Let* AS.1–AS.2 *hold. If* $\{\sigma_k\}$ *is bounded, then there must have*

$$\lim_{k\to\infty} \|\boldsymbol{c}_k\| = 0. \tag{42}$$

Proof Due to (41), it follows from Step 4 in ALAS that Pred_k has the lower bound

$$\text{Pred}_k \geq \delta\sigma \min\{\Delta_k\|\boldsymbol{c}_k\|, \|\boldsymbol{c}_k\|^2\}. \tag{43}$$

The update scheme of $\boldsymbol{\lambda}_k$ in Step 4 and the assumption AS.2 indicate that the sum of all $-\boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_k + \boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_{k+1}$ is bounded because

$$\sum_{k=0}^{\infty}(-\boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_k + \boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_{k+1}) \leq 2\|\boldsymbol{\lambda}_{\max}\|\left(\boldsymbol{c}_{\max} + \frac{1}{1-\beta}R_0\right) < \infty.$$

Thus, the sum of all Ared_k is bounded as well:

$$\sum_{k=0}^{\infty}\text{Ared}_k = \sum_{k=0}^{\infty}(f_k - f_{k+1}) + \sum_{k=0}^{\infty}(-\boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_k + \boldsymbol{\lambda}_k^{\mathsf{T}}\boldsymbol{c}_{k+1}) + \frac{\sigma}{2}\sum_{k=0}^{\infty}(\|\boldsymbol{c}_k\|^2 - \|\boldsymbol{c}_{k+1}\|^2) \leq \bar{M} \tag{44}$$

for some positive constant $\bar{M}$ due to AS.1–AS.2. In order to prove (42), we first need to show that

$$\liminf_{k\to\infty} \|\boldsymbol{c}_k\| = 0. \tag{45}$$

By contradiction, let us assume that (45) were not true. Then there exists a constant $\tau > 0$ such that for all large k, $\|\boldsymbol{c}_k\| \geq \tau$. In this case, (43) indicates $\text{Pred}_k \geq \delta\sigma \min\{\Delta_k\tau, \tau^2\}$. Denote $\bar{\mathcal{S}}$ as

the set of all indices corresponding to successful iterations, namely,

$$\bar{\mathcal{S}} = \{k \in \mathbb{N} : \rho_k \geq \eta\}.$$

With the help of (43) and (44), we have that $\{\Delta_k\}_{\bar{\mathcal{S}}} \to 0$. Then the update rule of trust region radii indicates that for all k

$$\Delta_k \to 0, \quad \text{as } k \to \infty, \tag{46}$$

which deduces that for all large k

$$|\rho_k - 1| = \frac{|\text{Ared}_k - \text{Pred}_k|}{\text{Pred}_k} \leq \frac{M\Delta_k^2}{\delta\sigma \min\{\Delta_k\tau, \tau^2\}} \to 0,$$

where M is a finite positive constant. It implies that $\Delta_{k+1} \geq \Delta_k$ for all large k, which contradicts Equation (46). Therefore Equation (45) holds.

We now prove Equation (42) by the way of contradiction. Assume that Equation (42) were not true. Then there exists an infinite indices set $\{m_i\} \subset \bar{\mathcal{S}}$ and a positive number ν such that

$$\|\boldsymbol{c}_{m_i}\| \geq 2\nu. \tag{47}$$

Moreover, Equation (45) suggests the existence of a subsequence $\{n_i\} \subset \bar{\mathcal{S}}$ such that

$$\|\boldsymbol{c}_k\| \geq \nu \ (m_i \leq k < n_i) \quad \text{and} \quad \|\boldsymbol{c}_{n_i}\| < \nu. \tag{48}$$

Define the set

$$\mathcal{K} := \bigcup_i \{k \in \bar{\mathcal{S}} : m_i \leq k < n_i\}.$$

Then, Ared_k with $k \in \mathcal{K}$ has the lower bound

$$\text{Ared}_k \geq \eta\delta\sigma \min\{\Delta_k\nu, \nu^2\} \geq \xi\Delta_k, \quad k \in \mathcal{K}$$

for some positive constant ξ. From this and Equation (44) we know that $\{\Delta_k\}_{\mathcal{K}} \to 0$. Then for all sufficiently large i,

$$\|\boldsymbol{x}_{m_i} - \boldsymbol{x}_{n_i}\| \leq \sum_{k=m_i}^{n_i-1} \|\boldsymbol{x}_k - \boldsymbol{x}_{k+1}\| \leq \sum_{\substack{k=m_i \\ k\in\bar{\mathcal{S}}}}^{n_i-1} \Delta_k \leq \frac{1}{\xi} \sum_{\substack{k=m_i \\ k\in\bar{\mathcal{S}}}}^{n_i-1} \text{Ared}_k.$$

The boundedness of $\sum_{k=0}^{\infty} \text{Ared}_k$ in Equation (44) implies that $\|\boldsymbol{x}_{m_i} - \boldsymbol{x}_{n_i}\| \to 0$ as $i \to \infty$, which further indicates that $\|\boldsymbol{c}_{m_i} - \boldsymbol{c}_{n_i}\| \to 0$ as $i \to \infty$. Therefore, with the help of Equation (48), we know that for all sufficiently large i, $\|\boldsymbol{c}_{m_i}\| < 2\nu$ holds, which contradicts Equation (47). The proof is complete. ■

We are now ready to give the main convergence result of ALTR algorithm with bounded penalty parameters.

THEOREM 3.5 *Let* AS.1–AS.2 *hold. If* $\{\sigma_k\}$ *is bounded, then* $\{\boldsymbol{x}_k\}$ *generated by ALTR algorithm is not bounded away from KKT points of* (1).

Proof Due to (41), the subproblem (8) now turns into

$$\begin{aligned} \min_{\boldsymbol{d}\in\mathbb{R}^n} \quad & q_k(\boldsymbol{d}) = \bar{\boldsymbol{g}}_k^{\mathsf{T}}\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^{\mathsf{T}}(\mathbf{B}_k + \sigma\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k)\boldsymbol{d} + \frac{\sigma}{2}\|\boldsymbol{c}_k\|^2 \\ \text{s. t.} \quad & \|\boldsymbol{d}\| \le \Delta_k \end{aligned}$$

with $\bar{\boldsymbol{g}}_k = \boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k + \sigma\mathbf{A}_k^{\mathsf{T}}\boldsymbol{c}_k$. Then through simple calculations, the predicted reduction at each iteration satisfies

$$\begin{aligned} q_k(\mathbf{0}) - q_k(\boldsymbol{s}_k) &\ge \bar{\beta}\frac{\|\bar{\boldsymbol{g}}_k\|}{2}\min\left\{\frac{\|\bar{\boldsymbol{g}}_k\|}{\|\mathbf{B}_k + \sigma\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k\|}, \Delta_k\right\} \\ &\ge \bar{\beta}\zeta_1\|\bar{\boldsymbol{g}}_k\|\min\{\zeta_2\|\bar{\boldsymbol{g}}_k\|, \Delta_k\}, \end{aligned} \tag{49}$$

with two positive constants ζ_1 and ζ_2 (see [10] for reference). Then (44) indicates that

$$\liminf_{k\to\infty} \|\bar{\boldsymbol{g}}_k\| = 0. \tag{50}$$

It follows from Lemma 3.4 that

$$\liminf_{k\to\infty} \|\boldsymbol{g}_k - \mathbf{A}_k^{\mathsf{T}}\boldsymbol{\lambda}_k\| = 0.$$

As $\boldsymbol{\lambda}_k$ is bounded for all k, there exist an accumulation point $\boldsymbol{x}_*$ of $\{\boldsymbol{x}_k\}$ and an accumulation point $\boldsymbol{\lambda}_*$ of $\{\boldsymbol{\lambda}_k\}$ such that

$$\boldsymbol{c}(\boldsymbol{x}_*) = \mathbf{0}, \quad \nabla f(\boldsymbol{x}_*) = \mathbf{A}(\boldsymbol{x}_*)^{\mathsf{T}}\boldsymbol{\lambda}_*.$$

Therefore, $\boldsymbol{x}_*$ is a KKT point of (1). ■

Note that we do not assume any constraint qualification in the convergence result of ALTR algorithm when penalty parameters are bounded.

4. Numerical experiments

In this section we examine the numerical behaviour of ALTR algorithm. Our implementation is executed in Matlab 7.6.0 (R2008a) on a PC with a 1.86 GHz Pentium Dual-Core microprocessor and 1GB of memory running Fedora 8.0. We test 136 equality constrained optimization problems from CUTEr [17]. The test set comprises both medium and large-scale problems. The number of variables varies from 2 to 4499, while the number of equality constraints ranges from 1 to 2998.

In our implementations, we use the default starting point x_0 for each problem. Moreover, as CUTEr can provide the exact Hessian of functions, we set $\mathbf{B}_k = \nabla^2_{\boldsymbol{xx}} l(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ at each iteration. We believe that it is useful during the early stages of code development. Many solvers have been proposed for the trust region subproblem (8) (see, e.g. [7,19,23]). Among them, the routine *gqtpar* proposed by Moré and Sorensen [23] is very effective and stable. It relies on a convenient characterization of the exact solution and usually performs very well for relatively small problems. But it was designed initially for general purpose without consideration of any specific characteristic of each problem, especially for large-scale problems. However, for large-scale problems, normally there exist some special structures, such as sparsity and low rank property. Take problems AUG2DC and DTOC3 for examples. We draw the distribution curve of nonzero elements of the Hessian $\mathbf{B}_k + \sigma_k\mathbf{A}_k^{\mathsf{T}}\mathbf{A}_k$ for these two problems, respectively, in Figure 1. The percentage for nonzero numbers are both around 0.1%, which shows that both problems are very sparse

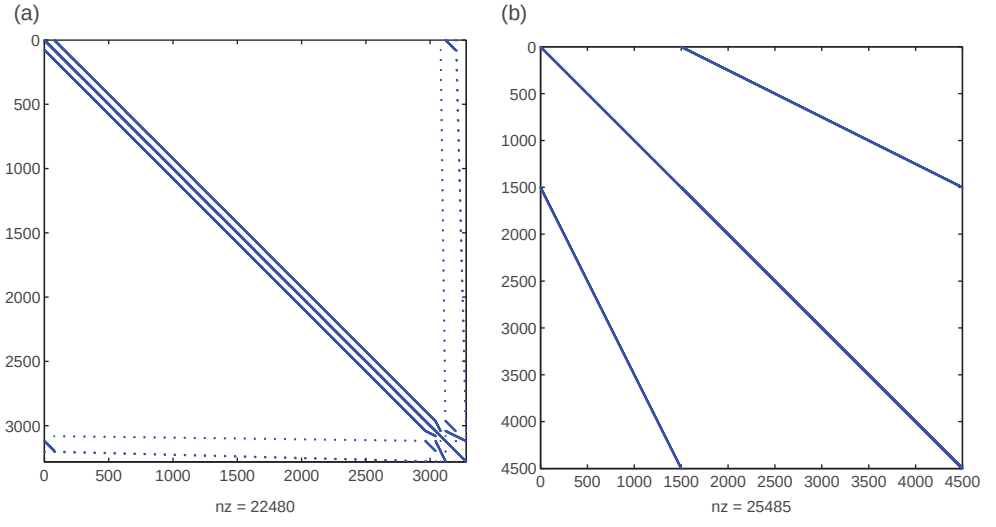

Figure 1. Dimension and distribution curve of nonzero elements of the Hessian $\mathbf{B}_k + \sigma_k \mathbf{A}_k^{\mathrm{T}} \mathbf{A}_k$. (a) AUG2DC ($3280 \times 3280$) and (b) DTOC3 ($4499 \times 4499$).

although their dimensions are large. So the performance is believed to be improved a lot if the algorithm can make use of structures of the test problems. For large-scale problems, the main cost of *gqtpar* is spent on Cholesky factorizations. Actually, this cost can be reduced if sparsity is considered. Motivated by this, we rewrite *gqtpar* in Matlab and combine some techniques to deal with sparsity.

We compare ALTR with the famous code LANCELOT [8], which is written in Fortran. It seems fair that we should compare our method with some existing approach which is also written in Matlab. So we choose to compare with the subroutine *fmincon* in Matlab Optimization Toolbox. To meet different users' demands, some parameters in LANCELOT and *fmincon* are optional. In numerical experiments we adopt the following settings to make the comparisons as fair as possible.

LANCELOT:
BEGIN
gradient-accuracy-required 1e-5
exact-second-derivatives-used
trust-region-radius 1.0
maximum-number-of-iterations 1000
two-norm-trust-region-used
END
fmincon:
options = optimset('Algorithm', 'Active-set Algorithm', 'Hessian', 'on', 'InitTrustRegionRadius', '1', 'MaxFunEvals', '1000', 'TolCon', 1e-5, 'TolFun', '1e-5', 'TolX', 1e-15).

In ALTR, we also set the initial trust region radius as 1. And the maximum number of function evaluations is set as 1000. The practical termination condition adopted here is

$$\|\boldsymbol{c}_k\| < 10^{-5}, \quad \|P_{\mathcal{N}_k}(\boldsymbol{g}_k)\| < 10^{-5}. \tag{51}$$

Except for (51) we terminate ALTR when

$$\|\boldsymbol{s}_k\| \leq 10^{-15} \tag{52}$$

Table 1. Results on small- and medium-scale problems.

	Prob. Dim.		LANCELOT		*fmincon*		ALTR	
Problem	n	m	n_f	n_g	n_f	n_g	n_f	n_g
AIRCRFTA	8	5	4	5	3	2	3	3
ARGTRIG	200	200	19	17	3	2	4	4
ARTIF	102	100	34	25	12	10	11	11
BDVALUE	100	100	1	2	2	1	2	2
BDVALUES	100	100	28	29	14	13	33	33
BOOTH	2	2	3	4	2	1	4	4
BROWNALE	200	200	6	7	14	7	9	9
BROYDN3D	500	500	6	7	5	4	7	7
BT1	2	1	23	20	*F*	*F*	7	7
BT2	3	1	27	27	16	15	21	21
BT3	5	3	10	11	9	8	22	22
BT4	3	2	20	21	13	11	7	7
BT5	3	2	17	17	9	7	6	6
BT6	5	2	21	20	33	26	13	11
BT7	5	3	48	46	154	38	122	120
BT8	5	2	27	25	11	10	14	14
BT9	4	2	20	21	*F*	*F*	23	21
BT10	2	2	17	18	8	7	16	16
BT11	5	3	18	19	13	10	18	18
BT12	5	3	22	21	7	6	21	18
BYRDSPHR	3	2	35	22	166	14	16	15
CATENA	33	10	53	53	*F*	*F*	133	111
CATENARY	33	10	81	79	*F*	*F*	277	254
CHAIN	800	401	*F*	*F*	4	3	*F*	*F*
CHANDHEU	100	100	14	15	10	9	13	13
CHNRSBNE	50	98	74	61	*F*	*F*	46	40
CLUSTER	2	2	11	11	8	7	9	9
CUBENE	2	2	46	40	3	2	19	15
DECONVNE	61	40	57	46	2	1	24	15
DRCAVTY3	196	100	45	37	*F*	*F*	20	12
DTOC2	298	198	31	31	114	52	82	71
EIGENA2	6	3	5	6	3	2	5	5
EIGENACO	110	55	19	20	3	1	13	13
EIGENAU	110	110	20	20	2	1	12	12
EIGENB2	6	3	10	10	3	1	19	19
EIGENB	110	110	185	151	*F*	*F*	86	69
EIGENBCO	6	3	18	16	3	1	10	9
EIGENC2	30	15	44	40	2	1	10	10
EIGENCCO	462	231	204	169	*F*	*F*	211	198
ELEC	75	25	48	42	667	211	27	22
FLOSP2TH	323	323	*F*	*F*	*F*	*F*	230	158
GENHS28	10	8	6	7	8	7	8	8
GOTTFR	2	2	27	24	8	5	10	6
HATFLDF	3	3	24	22	*F*	*F*	9	7
HATFLDG	25	25	15	14	18	6	8	8
HEART6	6	6	*F*	*F*	*F*	*F*	490	481
HEART8	8	8	599	520	*F*	*F*	40	35
HIMMELBA	2	2	3	4	2	1	5	5
HIMMELBC	2	2	7	7	7	5	6	6
HIMMELBE	3	3	5	6	3	2	5	5
HS100LNP	7	2	39	38	*F*	*F*	9	7
HS111LNP	10	3	55	52	42	41	12	12
HS26	3	1	31	29	8	4	16	16
HS27	3	1	12	13	631	96	13	11
HS28	3	1	3	4	8	6	6	6
HS39	4	2	20	21	*F*	*F*	23	21
HS40	4	3	10	11	7	6	6	6
HS42	4	2	10	11	10	9	8	8
HS46	5	2	26	20	15	11	17	16
HS47	5	3	22	22	61	27	16	15

(*Continued*)

Table 1. Continued

Problem	Prob. Dim.		LANCELOT		*fmincon*		ALTR	
	n	m	n_f	n_g	n_f	n_g	n_f	n_g
HS48	5	2	3	4	8	6	6	6
HS49	5	2	15	16	22	18	15	15
HS50	5	3	10	11	19	10	14	14
HS51	5	3	2	3	7	5	10	10
HS52	5	3	7	8	6	5	17	17
HS56	7	4	13	13	10	12	10	8
HS61	3	2	18	18	*F*	*F*	11	10
HS6	2	1	53	48	14	7	14	12
HS77	5	2	23	22	24	21	13	10
HS78	5	3	13	12	10	9	7	7
HS79	5	3	14	13	11	10	6	6
HS7	2	1	17	17	18	10	8	8
HS8	2	2	10	10	6	5	6	6
HS9	2	1	5	6	11	6	6	6
HYDCAR20	99	99	*F*	*F*	10	8	759	753
HYDCAR6	29	29	*F*	*F*	6	5	76	71
HYPCIR	2	2	5	6	6	4	5	5
INTEGREQ	102	100	3	4	3	2	3	3
JUNKTURN	510	350	72	68	3	2	*F*	*F*
LCH	150	1	35	34	*F*	*F*	29	17
MARATOS	2	1	7	8	4	3	8	6
MWRIGHT	5	3	18	18	19	9	8	8
METHANB8	31	31	243	244	3	2	4	4
METHANL8	31	31	592	584	5	4	18	18
MSQRTA	100	100	18	17	7	5	12	11
MSQRTB	100	100	19	17	7	4	9	9
OPTCTRL3	299	200	82	83	*F*	*F*	31	28
OPTCTRL6	299	200	82	83	*F*	*F*	28	26
ORTHRDM2	203	100	300	260	8	6	8	8
ORTHRDS2	503	250	852	738	*F*	*F*	614	614
ORTHREGB	27	6	74	66	6	5	48	47
ORTHRGDS	503	250	908	790	43	32	*F*	*F*
POWELLBS	2	2	47	42	23	11	61	56
POWELLSQ	2	2	16	14	*F*	*F*	22	19
RECIPE	3	3	16	17	23	11	14	14
RSNBRNE	2	2	32	28	5	2	14	12
S316-322	2	1	26	27	*F*	*F*	14	14
SINVALNE	2	2	37	33	5	2	21	18
SPMSQRT	499	829	15	13	*F*	*F*	10	10
TRIGGER	7	6	22	20	29	11	12	10
YATP1SQ	120	120	181	161	11	5	77	70
YATP2SQ	120	120	993	905	*F*	*F*	20	20
YFITNE	3	17	95	81	*F*	*F*	39	38
ZANGWIL3	3	3	7	8	3	1	11	11

or

$$\|\boldsymbol{c}_k\| \le 10^{-5} \quad \text{and} \quad \|\boldsymbol{s}_k\| \le 10^{-5}. \tag{53}$$

According to Theorem 3.2, it may occur that the sequence $\{\boldsymbol{x}_k\}$ is not bounded away from the infeasible stationary points of $\min \|\boldsymbol{c}(\boldsymbol{x})\|^2$, so it is reasonable to terminate the algorithm when the trial step is very short such as (52). In addition, once the iterate is close to the feasible region, it is acceptable to terminate the algorithm when the trial step is short, as required in (53).

We separate all the test problems into two classes. One class insists of small- and medium-scale problems, while the other is of large scale. We present the numerical results on all the aforementioned problems in Tables 1 and 2, respectively. For each problem, 'n' and 'm' denote

Table 2. Results on large-scale problems.

	Prob. Dim.		LANCELOT			*fmincon*			ALTR		
Problem	*n*	*m*	*nf*	*ng*	CPU(s)	*nf*	*ng*	CPU(s)	*nf*	*ng*	CPU(s)
AUG2DC	3280	1600	58	59	1.99	3	1	462.65	29	29	3.77
BRATU2D	1024	900	4	5	0.11			>10 min	7	7	0.44
BRATU2DT	1024	900	8	9	0.26			>10 min	9	9	0.56
BROYDN3D	1000	1000	6	7	0.02	9	4	20.964	8	8	0.23
CBRATU2D	3200	2888	5	6	0.55			>10 min	8	8	1.6
CBRATU3D	3456	2000	6	7	0.15			>10 min	7	7	3.14
DRCAVTY1	961	961	46	40	14.26	F	F	F	29	24	7.03
DRCAVTY2	961	961	104	85	24.06	F	F	F	64	59	16.5
DTOC1L	745	490	14	15	0.11	17	8	25.248	11	11	0.37
DTOC3	4499	2998	57	58	0.87	F	F	F	30	30	1.92
DTOC4	4499	2998	33	34	0.84	F	F	F	28	28	1.52
DTOC5	1999	999	37	38	0.3	38	11	385.4	26	26	0.52
DTOC6	1000	500	120	118	0.98			>10 min	128	197	1.46
EIGENC	462	462	299	247	9.83	F	F	F	45	35	23.03
FLOSP2TL	867	803			>10 min	9	4	336	21	21	6.6
FLOSP2TM	867	803			>10 min	19	9	461.1	66	66	27.54
GRIDNETB	3444	1764	32	33	3.11	F	F	F	22	22	4.18
HAGER1	2001	1000	11	12	0.14			>10 min	14	14	0.37
HAGER2	2001	1000	12	13	0.12	7	3	142.46	15	15	0.40
HAGER3	2001	1000	9	10	0.15	13	7	134.1	13	13	0.49
LUKVLE10	1000	998	47	37	0.2			>10 min	24	19	0.47
LUKVLE11	998	664	34	30	0.14	21	4	18.884	26	26	0.43
LUKVLE13	998	664	101	93	0.29			>10 min	121	116	1.56
LUKVLE16	997	747	59	50	0.17			>10 min	42	36	0.86
LUKVLE1	1000	998	20	20	0.14	43	15	98.29	30	30	0.53
LUKVLE3	1000	2	26	26	0.08			>10 min	16	16	0.27
LUKVLE6	999	499	41	42	0.31			>10 min	20	20	0.53
LUKVLE7	1000	4	92	80	0.21			>10 min	29	20	0.33
ORTHREGA	2053	1024	177	173	1.8			>10 min	26	23	11.71
ORTHREGC	1005	500	50	44	0.27	46	21	91.14	17	11	1.61
ORTHREGD	1003	500	515	443	3.55			>10 min	14	12	1.07
ORTHRGDM	4003	2000	158	141	2.96	F	F	F	12	12	27.19

the number of variables and the number of equality constraints separately. In addition, the number of function evaluations 'n_f' and the number of gradient evaluations 'n_g' are given. Actually, as the function value and constraint value can be obtained simultaneously by calling the subroutine 'cfn' in CUTEr, 'n_f' denotes the number of 'cfn' calls. Similarly, 'n_g' denotes the number of 'cgr' calls to compute both the gradient of objective function and gradient of constraints simultaneously. The entry 'F' means that corresponding algorithm terminates unsuccessfully because the CPU time exceeds 10 min, or the number of function evaluations exceeds the maximum number.

As the problems in Table 1 are of small and medium scale, they can be solved very fast by all of these three algorithms, so we do not report the CPU time on them here. We only record the CPU time on large-scale problems in Table 2. Note that *fmincon* is much slower than other two algorithms. One possible reason is that the computation of the trial step at each iteration costs too much. And although ALTR algorithm is written in Matlab, the CPU time on large problems are quite satisfactory, compared with LANCELOT.

To make a vivid description of algorithms, efficiency comparisons are shown in Figures 2 and 3, using the performance profiles introduced by Dolan and Moré [12]. The performance profiles are generated by executing solvers $\mathcal{S}$ on the test set $\mathcal{P}$. Considering the measure of interest, e.g. number of function evaluations, for each problem $p \in \mathcal{P}$ and each solvers from the set $s \in \mathcal{S}$, define $n_{p,s}$ as the number of function evaluations to solve problem by solver s. To compare the performance on problem p by solver s with the best performance on p by any solver,

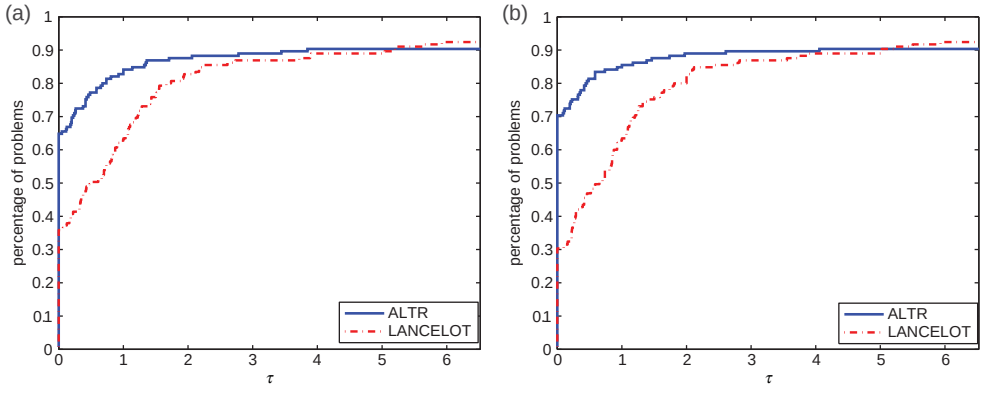

Figure 2. Performance profiles of ALTR and LANCELOT. (a) Function evaluations and (b) gradient evaluations.

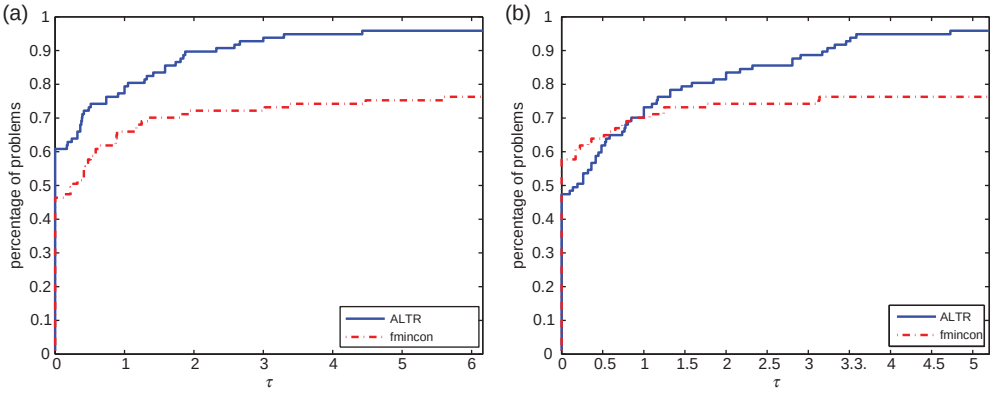

Figure 3. Performance profiles of ALTR and *fmincon*. (a) Function evaluations and (b) gradient evaluations.

define the performance ratio

$$r_{p,s} = \frac{n_{p,s}}{\min\{n_{p,s} : 1 \le s \le n_s\}},$$

where n_s is the number of solvers. Whenever the solver s does not solve problem p successfully, set $r_{p,s} = r_M$. Here, r_M is a very large preset positive constant. To obtain the overall performance of solver s on the test set $\mathcal{P}$, define

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : \log_2 r_{p,s} \le \tau\}.$$

Equivalently, $\rho_s(\tau)$ represents the probability that the performance ratio $r_{p,s}$ is within the factor 2^τ. It is easy to see that $\rho_s(0)$ is the probability that the solver s wins over the rest of solvers.

Observing Figure 2, we can see that ALTR performs slightly better than LANCELOT on both function evaluations and gradient evaluations. It shows that the probability that ALTR is faster on the function evaluations is near 65%(obtained from $\rho_s(0)$) and on the gradient evaluations is 70%. One can also compare the results when τ increases, by the definition of $\rho_s(\tau)$. So on the test problems our algorithm is comparable with the famous solver LANCELOT. Similar analysis can be made to Figure 3, which indicates that ALTR is more effective and efficient than *fmincon*.

5. Conclusions

In this paper we propose an ALTR method for equality constrained optimization. At each iteration, we minimize a second-order approximation of the augmented Lagrangian function. We introduce a new technique for updating penalty parameters which depends on results of trust region subproblems. With Lagrange multipliers adjusted adaptively, we establish the global convergence of our algorithm under mild conditions, no matter penalty parameters are bounded or unbounded. Numerical tests are reported on majority of equality constrained optimization problems from CUTEr. The numerical results reveal that our method is promising.

Acknowledgements

The authors would like to thank Prof. M.J.D. Powell for his great suggestions that improved the presentation of the paper, particularly for his providing the proof of Lemma 3.1. The authors are grateful to Prof. Yin Zhang and Prof. Zaiwen Wen for their kind advice on the numerical implementations in this paper and to Prof. Hongchao Zhang for his comments on an earlier version of this paper. They would like to thank two anonymous referees, the associate editor and Prof. Stefan Ulbrich for their detailed and valuable comments and suggestions.

Research of the first author is partially supported by Postdoc Grant 119103S175, UCAS President Grant Y35101AY00 and NSFC grant 11301505. Research of the second author is partially supported by NSFC Grant 11331012.

References

[1] R. Andreani, J.M. Martínez, and M.L. Schuverdt, *On the relation between constant positive linear dependence condition and quasinormality constraint qualification*, J. Optim. Theory Appl. 125 (2005), pp. 473-485.
[2] R. Andreani, E.G. Birgin, J.M. Martínez, and M.L. Schuverdt, *On augmented Lagrangian methods with general lower-level constraints*, SIAM J. Optim. 18 (2007), pp. 1286–1309.
[3] R. Andreani, E.G. Birgin, J.M. Martínez, and M.L. Schuverdt, *Augmented Lagrangian methods under the constant positive linear dependence constraint qualification*, Math. Program. 111 (2008), pp. 5–32.
[4] R. Andreani, G. Haeser, M.L. Schuverdt, and P.J.S. Silva, *A relaxed constant positive linear dependence constraint qualification and applications*, Math. Program. 135 (2012), pp. 255–273.
[5] N.S. Aybat and G. Iyengar, *A first-order augmented Lagrangian method for compressed sensing*, SIAM J. Optim. 22 (2012), pp. 429–459.
[6] R.H. Byrd, R.B. Schnabel, and G.A. Schultz, *A trust region algorithm for nonlinearly constrained optimization*, SIAM J. Numer. Anal. 24 (1987), pp. 1152–1170.
[7] R.H. Byrd, R.B. Schnabel, and G.A. Schultz, *Approximate solution of the trust region problem by minimization over two-dimensional subspaces*, Math. Program. 40 (1988), pp. 247–263.
[8] A.R. Conn, N.I.M. Gould, and Ph.L. Toint, *A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds*, SIAM J. Numer. Anal. 28 (1991), pp. 545–572.
[9] A.R. Conn, N.I.M. Gould, and Ph.L. Toint, *LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization(Release A)*, Springer, New York, 1992.
[10] A.R. Conn, N.I.M. Gould, and Ph.L. Toint, *Trust-region Methods*, MPS-SIAM Series on Optimization, Vol. 1 SIAM, Philadelphia, PA, 2000.
[11] F.E. Curtis, H. Jiang, and D.P. Robinson, *An adaptive augmented Lagrangian method for large-scale constrained optimization*, Math. Program., 2014, doi: 10.1007/s10107-014-0784-y.
[12] E.D. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. 91 (2002), pp. 201–213.
[13] Z. Dostál, *Semi-monotonic inexact augmented lagrangians for quadratic programming with equality constraints*, Optim. Methods Softw. 20 (2005), pp. 715–727.
[14] Z. Dostál, A. Friedlander, and S.A. Santos, *Augmented Lagrangians with adaptive precision control for quadratic programming with simple bounds and equality constraints*, SIAM J. Optim. 13 (2003), pp. 1120–1140.
[15] R. Fletcher, *Numerical experiments with an l_1 exact penalty function method*, in *Nonlinear Programming*, O.L. Mangasarian, R.R. Meyer and S.M. Robinson, eds., Vol. 4, Academic Press, New York, 1981, pp. 99–129.
[16] D. Goldfarb, S. Ma, and Z. Wen (eds.), *Solving Low-Rank Matrix Completion Problems Efficiently*, Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, USA, 2009.
[17] N.I.M. Gould, D. Orban, and Ph.L. Toint, *CUTEr, a constrained and unconstrained testing environment (revisited)*, ACM Trans. Math. Softw 29 (2003), pp. 373–394.
[18] W.W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim. 12 (2001), pp. 188–208.
[19] W.W. Hager and S. Park, *Global convergenct of SSM for minimizing a quadratic over a sphere*, Math. Comput. 74 (2004), pp. 1413–1423.

[20] S.P. Han, *Superlinearly convergent variable metric algorithms for general nonlinear programming problems*, Math. Program. 11 (1976), pp. 263–282.
[21] M.R. Hestenes, *Multiplier and gradient method*, J. Optim. Theory Appl. 4 (1969), pp. 303–320.
[22] N. Maratos, *Exact penalty function algorithms for finite dimensional and control optimization problems*, Ph.D. diss., Imperial College Sci. Tech. University of London, 1978.
[23] J.J. Moré and D.C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Comput. 4 (1983), pp. 553–572.
[24] L.F. Niu and Y. Yuan, *A new trust region algorithm for nonlinear constrained optimization*, J. Comput. Math. 28 (2010), pp. 72–86.
[25] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, New York, 2006.
[26] M.J.D. Powell, *A method for nonlinear constraints in minimization problems*, in *Optimization*, R. Fletcher, ed., Academic Press, London, 1969, pp. 283–298.
[27] M.J.D. Powell, *Algorithms for nonlinear constraints that use Lagrangian functions*, Math. Program. 14 (1978), pp. 224–248.
[28] M.J.D. Powell and Y. Yuan, *A trust region algorithm for linearly constrained optimization calculations*, Math. Program. 49 (1991), pp. 189–211.
[29] L. Qi and Z. Wei, *On the constant positive linear dependence condition and its application to SQP methods*, SIAM J. Optim. 10 (2000), pp. 963–981.
[30] R.T. Rockfellar, *The multiplier method of Hestenes and Powell applied to convex programming*, J. Optim. Theory Appl. 12 (1973), pp. 555–562.
[31] R.B. Wilson, *A simplicial algorithm for concave programming*, Ph.D. diss., Harvard Business School, Boston, 1963.
[32] Y. Yuan, *On the convergence of a new trust region algorithm*, Numer. Math. 70 (1995), pp. 515–539.
[33] Y. Yuan, *On the truncated conjugate gradient method*, Math. Program. 87 (2000), pp. 561–571.
[34] X.Y. Zhao, D. Sun, and K.C. Toh, *A newton-CG augmented Lagrangian method for semidefinite programming*, SIAM J. Optim. 20 (2010), pp. 1737–1765.