An augmented Lagrangian filter method

Sven Leyffer & Charlie Vanaret

ONLINE FIRST

Mathematical Methods of Operations Research

ISSN 1432-2994

Math Meth Oper Res
DOI 10.1007/s00186-020-00713-x

Springer

Mathematical Methods of Operations Research
https://doi.org/10.1007/s00186-020-00713-x

ORIGINAL ARTICLE

An augmented Lagrangian filter method

Sven Leyffer[1] · Charlie Vanaret[2]

Received: 23 October 2017 / Revised: 13 April 2020

Abstract
We introduce a filter mechanism to enforce convergence for augmented Lagrangian methods for nonlinear programming. In contrast to traditional augmented Lagrangian methods, our approach does not require the use of forcing sequences that drive the first-order error to zero. Instead, we employ a filter to drive the optimality measures to zero. Our algorithm is flexible in the sense that it allows for equality-constrained quadratic programming steps to accelerate local convergence. We also include a feasibility restoration phase that allows fast detection of infeasible problems. We provide a convergence proof that shows that our algorithm converges to first-order stationary points. We provide preliminary numerical results that demonstrate the effectiveness of our proposed method.

Keywords Augmented Lagrangian · Filter methods · Nonlinear optimization

Mathematics Subject Classification 90C30

1 Introduction

Nonlinearly constrained optimization is one of the most fundamental problems in scientific computing with a broad range of engineering, scientific, and operational applications. Examples include nonlinear power flow (Bautista et al. 2007; Donde et al. 2005; Momoh et al. 1997; Penfield et al. 1970; Sheble and Fahd 1994), gas transmission networks (Klaus and Steinbach Marc 2005; Martin et al. 2006; Bonnans and André 2009), the coordination of hydroelectric energy (Castro and Gonzalez 2004; Borghetti et al. 2003; Rabinowitz et al. 1988), and finance (Cornuejols and Tütüncü 2007),

✉ Sven Leyffer
leyffer@anl.gov

Charlie Vanaret
charlie.vanaret@itwm.fraunhofer.de

1 Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL 60439, USA

2 Fraunhofer ITWM, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany

Published online: 24 June 2020

including portfolio allocation (Konno and Yamazaki 1991; Ghaoui et al. 2003; Womersley and Lau 1996) and volatility estimation (Coleman et al. 1999; Altay-Salih et al. 2003). Chemical engineering has traditionally been at the forefront of developing new applications and algorithms for nonlinear optimization; see the surveys (Biegler and Grossmann 2004a, b). Applications in chemical engineering include process flowsheet design, mixing, blending, and equilibrium models. Another area with a rich set of applications is optimal control (Betts 2001); optimal control applications include the control of chemical reactions, the shuttle re-entry problem (Bonnard et al. 2003; Betts 2001), and the control of multiple airplanes (Arrieta-Camacho et al. 2007). More importantly, nonlinear optimization is a basic building block of more complex design and optimization paradigms, such as mixed-integer nonlinear optimization (Abhishek et al. 2010; Fletcher and Leyffer 1994; Goux and Leyffer 2002; Leyffer 2001; Bonami et al. 2015; Belotti et al. 2013) and optimization problems with complementarity constraints (Leyffer et al. 2006; Raghunathan and Biegler 2005; Leyffer and Munson 2007).

Nonlinearly constrained optimization has been studied intensely for more than 50 years, resulting in a wide range of algorithms, theory, and implementations. Current methods fall into two competing classes, both Newton-like schemes: active-set methods (Gill et al. 1997, 2002; Fletcher and Leyffer 1998; Byrd et al. 2004; Chin and Fletcher 2003; Fletcher and Sainz de la Maza 1989) and interior-point methods (Forsgren et al. 2002; Kawayir et al. 2009; Wächter and Biegler 2005a, b; Benson et al. 2002; Vanderbei and Shanno 1999; Byrd et al. 1999, 2006). While both have their relative merits, interior-point methods have emerged as the computational leader for large-scale problems.

The Achilles' heel of interior-point methods is the lack of efficient warm-start strategies. Despite significant recent advances (Gondzio and Grothey 2002; Benson and Shanno 2007, 2008), interior-point methods cannot compete with active-set approaches when solving mixed-integer nonlinear programs (Bonami et al. 2011). This deficiency is at odds with the rise of complex optimization paradigms, such as nonlinear integer optimization that require the solution of thousands of closely related nonlinear problems and drive the demand for efficient warm-start techniques. On the other hand, active-set methods exhibit an excellent warm-starting potential. Unfortunately, current active-set methods rely on pivoting approaches and do not readily scale to multicore architectures (though some successful parallel approaches to linear programming (LP) active-set solvers can be found in the series of papers (Huangfu and Hall 2013; Smith and Hall 2012; Lubin et al. 2013)). To overcome this challenge, we study augmented Lagrangian methods, which combine better parallel scalability potential with good warm-starting capabilities.

We consider solving the following nonlinear program (NLP):

$$\begin{aligned} \underset{x}{\text{minimize}} \;\; & f(x) \\ \text{subject to} \;\; & c(x) = 0 \\ & l \le x \le u \end{aligned} \tag{NLP}$$

where $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$, $c : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable. We use superscripts $\cdot^{(k)}$ to indicate iterates, such as $x^{(k)}$, and evaluation of nonlinear functions, such as $f^{(k)} := f(x^{(k)})$ and $\nabla c^{(k)} = \nabla c(x^{(k)})$. The Lagrangian of (NLP) is defined as

$$\mathcal{L}(x, y) = f(x) - y^T c(x), \tag{1.1}$$

where $y \in \mathbb{R}^m$ is a vector of Lagrange multipliers of $c(x) = 0$.

The first-order optimality conditions of (NLP) can be written as

$$\min\{x - l, \max\{x - u, \nabla_x \mathcal{L}(x, y)\}\} = 0 \tag{1.2a}$$
$$c(x) = 0, \tag{1.2b}$$

where the min and max are taken componentwise. It can be shown that (1.2a) is equivalent to the standard Karush–Kuhn–Tucker (KKT) conditions for (NLP). Introducing Lagrange multipliers z for the simple bounds, we obtain the KKT conditions

$$\nabla\mathcal{L}(x, y) - z = 0, \quad c(x) = 0, \quad l \le x \le u \;\perp\; z,$$

where $\perp$ represents complementarity and means that $z_i = 0$ if $l_i < x_i < u_i$, and that $z_i \ge 0$ and $z_i \le 0$ if $x_i = l_i$ and $x_i = u_i$, respectively. This complementarity condition is equivalent to $\min\{x - l, \max\{x - u, z\}\} = 0$, and hence the KKT conditions are equivalent to (1.2a).

1.1 Augmented Lagrangian methods

The augmented Lagrangian is defined as

$$\begin{aligned}\mathcal{L}_\rho(x, y) &= f(x) - y^T c(x) + \frac{1}{2}\rho\|c(x)\|^2 \\ &= \mathcal{L}_0(x, y) + \frac{1}{2}\rho\|c(x)\|^2,\end{aligned} \tag{1.3}$$

for a given penalty parameter ρ. The Lagrangian (1.1) is therefore given by $\mathcal{L}_0(x, y)$, that is (1.3) with $\rho = 0$. Augmented Lagrangian methods have been studied by Bertsekas (1982), Powell (1978), Murtagh and Saunders (1982). Recently, researchers have expressed renewed interest in augmented Lagrangian methods because of their good scalability properties, which had already been observed in Conn et al. (1992). The key computational step in bound-constrained augmented Lagrangian methods, such as LANCELOT (Conn et al. 1992) and ALGENCAN (Birgin and Martínez 2012, 2014), is minimization of $\mathcal{L}_{\rho_k}(x, y^{(k)})$ over x for given $\rho_k > 0$ and $y^{(k)} \in \mathbb{R}^m$, giving rise to the bound-constrained Lagrangian problem

$$\begin{aligned}&\underset{x}{\text{minimize}}\ \mathcal{L}_{\rho_k}(x, y^{(k)}) \\ &\text{subject to } l \le x \le u,\end{aligned} \tag{BCL$_k$}$$

whose solution we denote by $x^{(k+1)}$. A basic augmented Lagrangian method solves (BCL$_k$) approximately and updates the multipliers using the so-called first-order multiplier update:

$$y^{(k+1)} = y^{(k)} - \rho_k c(x^{(k+1)}). \tag{1.4}$$

or keeps $y^{(k)}$ and increases ρ_k. Traditionally, augmented Lagrangian methods have used two forcing sequences $\eta_k \searrow 0$ and $\omega_k \searrow 0$ to control the infeasibility and first-order error, and enforce global convergence. Sophisticated update schemes for η, ω can be found in Conn et al. (2000). Motivated by the KKT conditions (1.2a), we define the primal and dual infeasibility as

$$\eta(x) := \|c(x)\|. \tag{1.5a}$$

$$\omega_\rho(x, y) := \|\min\left\{x - l, \max\left\{x - u, \nabla_x \mathcal{L}_\rho(x, y)\right\}\right\}\| \tag{1.5b}$$

and observe that $\omega_0(x, y)$ is the dual feasibility error of (NLP). Moreover, the first-order multiplier update implies that

$$\begin{aligned}\nabla_x \mathcal{L}_0(x^{(k+1)}, y^{(k+1)}) &= \nabla f(x^{(k+1)}) - \nabla c(x^{(k+1)})^T y^{(k)} + \rho_k \nabla c(x^{(k+1)})^T c(x^{(k+1)}) \\ &= \nabla \mathcal{L}_{\rho_k}(x^{(k+1)}, y^{(k)}).\end{aligned}$$

It follows that

$$\omega_0(x^{(k+1)}, y^{(k+1)}) = \omega_{\rho_k}(x^{(k+1)}, y^{(k)}),$$

which is the dual feasibility error of (BCL_k). Hence, we can monitor the dual infeasibility error of (NLP) while solving (BCL_k).

A rough outline of an augmented Lagrangian method is given in Algorithm 1; we use a double-loop representation to simplify the comparison to our proposed filter method.

```
Given sequences η_k ↘ 0 and ω_k ↘ 0, an initial point (x^(0), y^(0)) and ρ_0, set
k ← 0 ;
while (x^(k), y^(k)) not optimal do
    Set j ← 0 and initialize x̂^(j) ← x^(k) ;
    Set up the augmented Lagrangian subproblem (BCL_k) ;
    while ω_{ρ_k}(x̂^(j), y^(k)) > ω_k and η(x̂^(j)) > η_k (not acceptable) do
        x̂^(j+1) ← approximate argmin_{l≤x≤u} L_{ρ_k}(x, y^(k)) from initial point x̂^(j);
        if ω_{ρ_k}(x̂^(j), y^(k)) ≤ ω_k but η(x̂^(j)) > η_k then
            Increase penalty parameter ρ_k ← 2ρ_k
        else
            Update multipliers: ŷ^(j+1) ← y^(k) − ρ_k c(x̂^(j+1)) ;
        Set j ← j + 1 ;
    Set (x^(k+1), y^(k+1)) ← (x̂^(j), ŷ^(j)), update ρ_{k+1} ← ρ_k, and increase
    k ← k + 1 ;
```

Algorithm 1: Classical Bound-Constrained Augmented Lagrangian Method.

Our goal is to improve traditional augmented Lagrangian methods in three ways, extending the augmented Lagrangian filter methods developed in Friedlander and Leyffer (2008) for quadratic programs to general NLPs:

1. Replace the forcing sequences (η_k, ω_k) by a less restrictive algorithmic construct, namely a filter (defined in Sect. 2);
2. Introduce a second-order step to promote fast local convergence, similar to sequential linear quadratic programming (SLQP) methods (Byrd et al. 2004; Chin and Fletcher 2003; Fletcher and Sainz de la Maza 1989);
3. Equip the augmented Lagrangian method with a fast and robust detection of infeasibility of (NLP), see, e.g. Fletcher and Leyffer (2003).

In Birgin and Martinez (2008), the authors study a related approach in which the augmented Lagrangian algorithm is used to find an approximate minimizer (e.g. to a tolerance of 10^{-4}), and then a crossover is performed to an interior-point method or a Newton method on the active constraints. In contrast, we propose a method that more naturally integrates second-order steps within the augmented Lagrangian framework.

Our paper is organized as follows. The next section defines the filter for augmented Lagrangians, and outlines our method. Section 3 presents the detailed algorithm and its components, and Sect. 4 presents the global convergence proof. In Sect. 5, we present some promising numerical results. We close the paper with some conclusions and outlooks.

2 An augmented Lagrangian filter

This section defines the basic concepts of our augmented Lagrangian filter algorithm. We start by defining a suitable filter and related step acceptance conditions. We then provide an outline of the algorithm that is described in more detail in the next section.

The new augmented Lagrangian filter is defined by using the residual of the first-order conditions (1.2a), defined in (1.5). Augmented Lagrangian methods use forcing sequences (ω_k, η_k) to drive $\omega_0(x, y)$ and $\eta(x)$ to zero. Here, we instead use the filter mechanism (Fletcher et al. 2002; Fletcher and Leyffer 2002) to achieve convergence to first-order points. A filter is formally defined as follows.

Definition 1 (*Augmented Lagrangian Filter and Acceptance*) A filter $\mathcal{F}$ is a list of pairs $(\eta_l, \omega_l) := \left(\eta(x^{(l)}), \omega_0(x^{(l)}, y^{(l)})\right)$ such that no pair dominates another pair, i.e. there exists no pairs $(\eta_l, \omega_l), (\eta_k, \omega_k), l \neq k$ such that $\eta_l \leq \eta_k$ and $\omega_l \leq \omega_k$. A point $(x^{(k)}, y^{(k)})$ is acceptable to the filter $\mathcal{F}$ if and only if

$$\eta_k := \eta(x^{(k)}) \leq \beta \eta_l \quad \text{or} \quad \omega_k := \omega_0(x^{(k)}, y^{(k)}) \leq \omega_l - \gamma \eta(x^{(k)}), \quad \forall (\eta_l, \omega_l) \in \mathcal{F}. \tag{2.1}$$

where $0 < \gamma, \beta < 1$ are constants.

At iteration k of our algorithm, we have a filter $\mathcal{F}_k$ with the property that $\eta_l > 0$ for all $l \in \mathcal{F}_k$. The fact that $(\eta(x), \omega_0(x, y)) \geq 0$ implies that we have an automatic upper bound on $\eta(x)$ for all points that are acceptable:

$$\eta(x) \leq U := \max\left(\omega_{\min}/\gamma, \eta_{\min}\right), \tag{2.2}$$

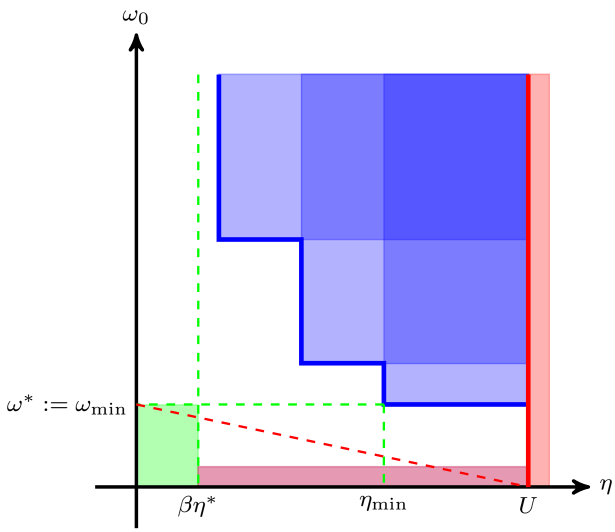

Fig. 1 Example of an augmented Lagrangian filter $\mathcal{F}$ with three entries. The filter is in blue, the dashed green line shows the envelope in η, and the upper bound U (red line) is implied by the sloping envelope condition (2.1) and $\omega_0 \geq 0$. Values above and to the right of the filter are not acceptable. The ideal point is (η^*, ω^*), and the green area shows the set of filter entries that are guaranteed to be acceptable. The shaded purple area is the set of entries that trigger the switch to restoration. (Color figure online)

where $\omega_{\min}$ is the smallest first-order error of any filter entry, that is $\omega_{\min} := \min\{\omega_l : (\eta_l, \omega_l) \in \mathcal{F}_k\}$, and $\eta_{\min}$ is the η-value corresponding to $\omega_{\min}$, see Fig. 1. The point (η^*, ω^*) is the ideal filter entry.

Our filter is based on the Lagrangian and not on the augmented Lagrangian. This choice is deliberate: one can show that the gradient of the Lagrangian after the first-order multiplier update (1.4) equals the gradient of the augmented Lagrangian, namely:

$$\nabla_x \mathcal{L}_0(x^{(k)}, y^{(k)}) = \nabla_x \mathcal{L}_{\rho_k}(x^{(k)}, y^{(k-1)}). \tag{2.3}$$

Thus, by using the Lagrangian, we ensure that filter-acceptable points remain acceptable after the first-order multiplier update. Moreover, (2.3) shows that the filter acceptance can be readily checked during minimization of the augmented Lagrangian, in which the multiplier is fixed and we iterate over x only.

The filter envelope defined by β and γ ensures that iterates cannot accumulate at points where $\eta > 0$, and it promotes convergence (see Lemma 5). A benefit of the filter approach is that we do not need to assume that the multipliers remain bounded or that the iterates remain in a compact set, although we assume later that there exists no feasible points at infinity. We outline the main algorithmic ideas in Algorithm 2; in the next section we provide a detailed description of the algorithm and its main components.

Given $(x^{(0)}, y^{(0)})$ and ρ_0, set $\omega_0 \leftarrow \omega(x^{(0)}, y^{(0)})$, $\eta_0 \leftarrow \eta(x^{(0)})$, $\mathcal{F}_0 \leftarrow \{(\eta_0, \omega_0)\}$, and $k \leftarrow 0$;
while $(x^{(k)}, y^{(k)})$ *not optimal* **do**
    Set $j \leftarrow 0$, and initialize $\hat{x}^{(j)} \leftarrow x^{(k)}$;
    while $(\hat{\eta}_j, \hat{\omega}_j)$ not *acceptable to* $\mathcal{F}_k$ **do**
        $\hat{x}^{(j+1)} \leftarrow$ *approximate* $\underset{l \le x \le u}{\operatorname{argmin}} \mathcal{L}_{\rho_k}(x, y^{(k)})$ from initial point $\hat{x}^{(j)}$;
        if restoration switching condition holds **then**
            Increase penalty: $\rho_{k+1} \leftarrow 2\rho_k$;
            Switch to restoration to find acceptable $(\hat{\eta}_j, \hat{\omega}_j)$;
        Update multipliers: $\hat{y}^{(j+1)} \leftarrow \hat{y}^{(j)} - \rho_k c(\hat{x}^{(j+1)})$;
        Set $j \leftarrow j + 1$;
    Set $(x^{(k+1)}, y^{(k+1)}) \leftarrow (\hat{x}^{(j)}, \hat{y}^{(j)})$;
    if $\eta_{k+1} > 0$ **then**
        Add $(\eta_{k+1}, \omega_{k+1})$ to $\mathcal{F}_k$ (only points with $\eta_{k+1} > 0$ are added);
    Set $k \leftarrow k + 1$;

Algorithm 2: Outline of Augmented Lagrangian Filter Method

Algorithm 2 has an inner iteration in which we minimize the augmented Lagrangian until a filter-acceptable point is found. Inner iterates are distinguished by a "hat", that is $\hat{x}^{(j)}$. Outer iterates are denoted by $x^{(k)}$. A restoration phase is invoked if the iterates fail to make progress toward feasibility. The outline of our algorithm is deliberately vague to convey the main ideas. Details of the conditions of switching to restoration, termination of the inner iteration, and increase of the penalty parameter are developed in the next section. The algorithm supports an optional penalty increase condition, which triggers a heuristic to estimate the penalty parameter. In addition, our algorithm implements an optional second-order step on the set of active constraints. Our analysis, however, concentrates on the plain augmented Lagrangian approach.

Most of the effort of Algorithm 2 lies in the approximate minimization of the augmented Lagrangian, for which efficient methods exist, such as bound-constrained projected-gradient conjugate-gradient methods, see, e.g. Moré and Toraldo (1991), Calamai and Moré (1987)

3 Detailed algorithm statement

We start by describing the four algorithmic components not presented in our outline: the penalty update, the restoration switching condition, the termination condition for the inner iteration, and the second-order step. We then discuss the complete algorithm.

3.1 Optional penalty update heuristic

Augmented Lagrangian methods can be shown to converge provided that the penalty parameter is sufficiently large and the multiplier estimate is sufficiently close to the optimal multiplier; see, for example, Bertsekas (1982). Here, we extend the penalty estimate from Friedlander and Leyffer (2008) to nonlinear functions. We stress that this step of the algorithm is not needed for global convergence, although it has been shown that these steps improve the behavior of our method in the context of QPs (Friedlander and Leyffer 2008). We show in Sect. 4 that the penalty update is bounded, so that our heuristic does not harm the algorithm.

Consider the Hessian of the augmented Lagrangian $\mathcal{L}_\rho(x, y)$:

$$\nabla^2 \mathcal{L}_\rho = \nabla^2 \mathcal{L}_0 + \rho \nabla c \nabla c^T + \rho \sum_{i=1}^{m} c_i \nabla^2 c_i, \tag{3.1}$$

which includes the usual Lagrangian Hessian, $\nabla^2 \mathcal{L}_0(x, y)$, and the last two terms that represent the Hessian of the penalty term, $\frac{\rho}{2}\|c(x)\|_2^2$. Ideally, we would want to ensure $\nabla^2 \mathcal{L}_\rho \succeq 0$ at the solution. Instead, we drop the $\nabla^2 c_i$ terms and consider

$$\nabla^2 \mathcal{L}_\rho \approx \nabla^2 \mathcal{L}_0 + \rho \nabla c \nabla c^T. \tag{3.2}$$

Now, we use the same ideas as in Friedlander and Leyffer (2008) to develop a penalty estimate that ensures that the augmented Lagrangian is positive definite on the null space of the active inequality constraints. We define the active and inactive sets as

$$\mathcal{A}^k := \mathcal{A}(x^{(k)}) := \left\{ i : x_i^{(k)} = l_i \text{ or } x_i^{(k)} = u_i \right\} \text{ and } \mathcal{I}^k := \{1, 2, \ldots, n\} - \mathcal{A}^k, \tag{3.3}$$

respectively.

Next, we define reduced Hessian and Jacobian matrices. For a set of row and column indices $\mathcal{R}, \mathcal{C}$ and a matrix M, we define the submatrix $M_{\mathcal{R},\mathcal{C}}$ as the matrix with entries M_{ij} for all $(i, j) \in \mathcal{R} \times \mathcal{C}$ (we also use the Matlab notation ":" to indicate that all entries on a dimension are taken). In particular, for the Hessian $\nabla_x^2 \mathcal{L}_{\rho_k}(x^{(k)}, y^{(k)})$, and the Jacobian $\nabla c(x^{(k)})^T$, we define the reduced Hessian and Jacobian as

$$H_k := \left[\nabla_x^2 \mathcal{L}_{\rho_k}(x^{(k)}, y^{(k)})\right]_{\mathcal{I}_k, \mathcal{I}_k} \quad \text{and} \quad A_k := \left[\nabla c_1(x^{(k)}) : \cdots : \nabla c_m(x^{(k)})\right]_{\mathcal{I}_k, :} \in \mathbb{R}^{|\mathcal{I}_k| \times m}. \tag{3.4}$$

We can show that a sufficient condition for $\nabla^2 \mathcal{L}_\rho \succeq 0$ on the active set is

$$\rho \geq \rho_{\min}(\mathcal{A}^k) := \frac{\max\{0, -\lambda_{\min}(H_k)\}}{\sigma_{\min}(A_k)^2}, \tag{3.5}$$

where $\lambda_{\min}(\cdot)$ and $\sigma_{\min}(\cdot)$ denote the smallest eigenvalue and singular value, respectively. Computing (3.5) directly would be prohibitive for large-scale problems, and we use the following estimate instead:

$$\tilde{\rho}_{\min}(\mathcal{A}^k) := \max \left\{ 1, \frac{\|H_k\|_1}{\max\left\{ \frac{1}{\sqrt{|\mathcal{I}_k|}} \|A_k\|_\infty, \frac{1}{\sqrt{m}} \|A_k\|_1 \right\}} \right\}, \tag{3.6}$$

where $|\mathcal{I}^k|$ is the number of free variables and m is the number of general equality constraints. If $\rho_k < \tilde{\rho}_{\min}(\mathcal{A}^k)$, then we increase the penalty parameter to $\rho_{k+1} = 2\tilde{\rho}_{\min}(\mathcal{A}^k)$. We could further improve this estimate by taking the terms $\rho c_i \nabla^2 c_i$ into account, which would change the numerator in (3.6).

An alternative adaptive penalty update is proposed in Curtis et al. (2015) to mitigate any initial poor choices of penalty parameter during early iterations.

3.2 Switching to restoration phase

In practice, many NLPs are not feasible; this situation happens frequently, for example during the resolution of MINLPs. In this case, it is important that the NLP solver quickly and reliably find a minimum of the constraint violation $\eta(x)^2$. To converge quickly to such a point, we have to drive the penalty parameter to infinity or switch to minimizing $\eta(x)$. We prefer the latter approach because it provides an easy escape if we determine that the NLP appears to be feasible after all. Unlike in linear programming (LP), there does not exist a phase I/phase II approach for NLPs, because even once feasibility is achieved, subsequent steps cannot be guaranteed to maintain feasibility for general NLP, unlike for LP, where we only need to establish feasibility once in phase I.

We define a set of implementable criteria that force the algorithm to switch to the feasibility restoration phase that minimizes the constraint violation. Recall that the augmented Lagrangian filter implies the existence of an upper bound $U = \max\{\omega_{\min}/\gamma, \eta_{\min}\}$ from (2.2). Thus any inner iteration that generates

$$\hat{\eta}_{j+1} = \eta(\hat{x}^{(j+1)}) \geq \beta U \tag{3.7}$$

triggers the restoration phase. The second test that triggers the restoration phase is related to the minimum constraint violation $\eta_{\min}$ of filter entries. In particular, if it appears that the augmented Lagrangian is converging to a stationary point while the constraint violation is still large, then we switch to the restoration phase, because we take this situation as an indication that the penalty parameter is too small, illustrated by the purple area in Fig. 1. This observation motivates the following condition:

$$\omega_{\rho_k}(\hat{x}^{(j+1)}, y^{(k)}) \leq \epsilon \;\text{ and }\; \eta(\hat{x}^{(j+1)}) \geq \beta \eta_{\min}, \tag{3.8}$$

where $\epsilon > 0$ is a constant and $\eta_{\min}$ is the smallest constraint violation of any filter entry, namely $\eta_{\min} := \min\{\eta_l : (\eta_l, \omega_l) \in \mathcal{F}_k\} > 0$, which is positive because we only ever add entries with positive constraint violation to the filter. In our algorithm, we switch to restoration if (3.7) or (3.8) holds.

Each time we switch to restoration, we increase the penalty parameter and start a new major iteration. The outcome of the restoration phase is either a (local) minimum of the infeasibility or a new point that is filter-acceptable. The (approximate) first-order condition for a minimum of the constraint violation $\eta(x)^2$ at $\hat{x}^{(j)}$ is

$$\left\| \min\left(\hat{x}^{(j)} - l, \max\left(\hat{x}^{(j)} - u, 2\nabla c(\hat{x}^{(j)})^T c(\hat{x}^{(j)})\right)\right)\right\| \leq \epsilon \;\text{ and }\; \eta(\hat{x}^{(j)}) > \epsilon, \tag{3.9}$$

where $\epsilon > 0$ is a constant that represents the optimality tolerance. The mechanism of the algorithm ensures that we either terminate at a first-order point of the constraint violation, or find a point that is acceptable to the filter, because $\eta_{\min} > 0$, which is formalized in the following lemma.

Lemma 1 *Either the restoration phase converges to a minimum of the constraint violation, or it finds a point $x^{(k+1)}$ that is acceptable to the filter in a finite number of steps.*

Proof The restoration phase minimizes $\eta(x)^2$ and hence either converges to a local minimum of the constraint violation or generates a sequence of iterates $x^{(j)}$ with $\eta(x^{(j)}) \to 0$. Because we only add points with $\eta_l > 0$ to the filter, it follows that $\eta_l > 0$ for all $(\eta_l, \omega_l) \in \mathcal{F}_k$ (defined in Algorithm 2), and hence that we must find a filter-acceptable point in a finite number of iterations in the latter case. □

Whenever we switch to the restoration phase, we assume that the algorithm generates a new primal-dual iterate, $(x^{(k)}, y^{(k)})$ that is acceptable to the filter. We can achieve this, for example by minimizing $\eta(x)$, and performing a first-order multiplier update.

3.3 Termination of inner minimization

The filter introduced in Sect. 2 ensures convergence only to *feasible* limit points; see Lemma 5. Thus, we need an additional condition that ensures that the limit points are also first-order optimal. We introduce a sufficient reduction condition that will ensure that the iterates are stationary. A sufficient reduction condition is more natural (since it corresponds to a Cauchy-type condition, which holds for all reasonable optimization routines) than is a condition that explicitly links the progress in first-order optimality ω_k to progress toward feasibility η_k.

In particular, we require that the following condition be satisfied at each inner iteration:

$$\Delta\mathcal{L}_{\rho_k}^{(j)} := \mathcal{L}_{\rho_k}(\hat{x}^{(j)}, y^{(k)}) - \mathcal{L}_{\rho_k}(\hat{x}^{(j+1)}, y^{(k)}) \geq \sigma\hat{\omega}_j, \tag{3.10}$$

where $\sigma > 0$ is a constant. This condition can be satisfied, for example, by requiring Cauchy decrease on the augmented Lagrangian for fixed ρ_k and $y^{(k)}$. We note that the right-hand side of (3.10) is the dual infeasibility error of the augmented Lagrangian at $\hat{x}^{(j)}$, which corresponds to the dual infeasibility error of (NLP) after the first-order multiplier update.

We will show that this sufficient reduction condition of the inner iterates in turn implies a sufficient reduction condition of the outer iterates as we approach feasibility; see (4.2). This outer sufficient reduction leads to a global convergence result. To the best of our knowledge, this is the first time that a more readily implementable sufficient reduction condition has been used in the context of augmented Lagrangians.

3.4 Optional second-order (EQP) step

Our algorithm allows for an additional second-order step. The idea is to use the approximate minimizers $x^{(k)}$ of the augmented Lagrangian to identify the active inequality constraints $x_i^{(k)} = l_i$ or $x_i^{(k)} = u_i$, and then solve an equality-constrained QP (EQP) on those active constraints, similarly to popular SLQP approaches. Given sets of active and inactive constraints (3.3), our goal is to solve an EQP with $x_i^{(k)} = l_i$, or $x_i^{(k)} = u_i$, $\forall i \in \mathcal{A}^k$. Provided that the EQP is convex, its solution can be obtained by solving an augmented linear system.

Using the notation introduced in (3.3) and (3.4), the convex EQP is equivalent to the following augmented system,

$$\begin{bmatrix} H_k & -A_k \\ -A_k^T & \end{bmatrix} \begin{pmatrix} \Delta x_{\mathcal{I}^k} \\ \Delta y \end{pmatrix} = \begin{pmatrix} -(\nabla \mathcal{L}_{\rho_k}^{(k+1)})_{\mathcal{I}^k} \\ c(x^{(k+1)}) \end{pmatrix}, \tag{3.11}$$

and $\Delta x_{\mathcal{A}^k} = 0$. In general, we cannot expect that the solution $x^{(k+1)} + \Delta x$ is acceptable to the filter (or may not be a descent direction for the augmented Lagrangian). Hence, we add a backtracking line search to our algorithm to find an acceptable point. We note that because $(x^{(k+1)}, y^{(k+1)})$ is known to be acceptable, we can terminate the line search if the step size is less that some $\alpha_{\min} > 0$ and instead accept $(x^{(k+1)}, y^{(k+1)})$.

3.5 Complete algorithm

A complete description of the method is given in Algorithm 3. It has an inner loop that minimizes the augmented Lagrangian with fixed penalty parameter ρ_k and multipliers $y^{(k)}$ until a filter-acceptable point is found. Quantities associated with the inner loop are indexed by j and have a "hat." The outer loop corresponds to major iterates and may update the penalty parameter. The inner iteration also terminates when we switch to the restoration phase. Any method for minimizing $\eta(x)^2$ (or any measure of constraint infeasibility) can be used in this phase. Note that the penalty parameter is also increased every time we switch to the restoration phase, although we could use a more sophisticated penalty update in that case, too.

We note that Algorithm 3 uses a flag *RestFlag* to indicate whether the restoration phase was entered or not. If the restoration phase is entered, the penalty parameter is increased in the outer loop iterates k. Two possible outcomes for the restoration phase exist: either we find a nonzero (local) minimizer of the constraint violation indicating that problem (NLP) is infeasible, or we find a filter-acceptable point and exit the inner iteration. In the latter case, *RestFlag* = *true* ensures that we do not update the penalty parameter using (3.6), which does not make sense in this situation.

4 Convergence proof

This section establishes a global convergence result for Algorithm 3, without the second-order step for the sake of simplicity. We make the following assumptions throughout this section.

Assumption 4.1 Consider problem (NLP), and assume that the following hold:

A1 The problem functions f, c are twice continuously differentiable.
A2 The constraint norm satisfies $\|c(x)\| \to \infty$ as $\|x\| \to \infty$.

```
Given (x^(0), y^(0)) and ρ_0, set ω_0 ← ω(x^(0), y^(0)), η_0 ← η(x^(0)), F_0 ← {(η_0, ω_0)}, and k ← 0 ;
while (x^(k), y^(k)) not optimal do
    Set j ← 0, RestFlag ← false, and initialize x̂^(0) ← x^(k) ;
    while (η̂_j, ω̂_j) not acceptable to F_k do
        Approximately minimize the augmented Lagrangian for x̂^(j+1) starting at x̂^(j):

            minimize_{l≤x≤u}  L_{ρ_k}(x, y^(k)) = f(x) − y^(k)T c(x) + ½ ρ_k ‖c(x)‖²

        such that the sufficient reduction condition (3.10) holds.
        if restoration switching condition (3.7) or (3.8) holds then
            Set RestFlag = true ;
            Increase penalty parameter: ρ_{k+1} ← 2ρ_k ;
            Switch to restoration phase to find (x̂^(j+1), ŷ^(j+1)) acceptable to F or find an infeasible
            point that minimizes ‖c(x)‖² subject to l ≤ x ≤ u;
        else
            Provisionally update multipliers: ŷ^(j+1) ← y^(k) − ρ_k c(x̂^(j+1));
            Compute ω̂_{j+1} ← ω_0(x̂^(j+1), ŷ^(j+1)) and η̂_{j+1} ← η(x̂^(j+1)).
        Set j ← j + 1;
    Set (x^(k+1), y^(k+1)) ← (x̂^(j), ŷ^(j)) ;
    optional EQP step
        Solve EQP (3.11) for (Δx^(k+1), Δy^(k+1))
        Line-search: Find α_k ∈ {0} ∪ [α_min, 1] such that

            (x^(k+1), y^(k+1)) = (x̂^(k+1), ŷ^(k+1)) + α_k(Δx^(k+1), Δy^(k+1)) acceptable to F_k

    Compute ω_{k+1} ← ω_0(x^(k+1), y^(k+1)), η_{k+1} ← η(x^(k+1));
    if η_{k+1} > 0 then
        Add (η_{k+1}, ω_{k+1}) to filter: F_{k+1} ← F_k ∪ {(η_{k+1}, ω_{k+1})} (ensuring η_l > 0 ∀l ∈ F_{k+1})
    if not RestFlag and ρ_k < ρ̃_min(A^k) see (3.6) then
        Increase penalty: ρ_{k+1} ← 2ρ̃_min(A^k);
    else
        Leave penalty parameter unchanged: ρ_{k+1} ← ρ_k;
    Set k ← k + 1;
```

Algorithm 3: Augmented Lagrangian Filter Method with optional EQP steps.

Assumption **A1** is standard. Assumption **A2** implies that our iterates remain in a compact set (see Lemma 2). This assumption could be replaced by an assumption that we optimize over finite bounds $l \leq x \leq u$. Both assumptions together imply that $f(x)$ and $c(x)$ and their derivatives are bounded for all iterates.

Algorithm 3 has three distinct outcomes.

1. There exists an infinite sequence of restoration phase iterates $x^{(k_l)}$, indexed by $\mathcal{R} := \{k_1, k_2, \ldots\}$, whose limit point $x^* := \lim x^{(k_l)}$ minimizes the constraint violation, satisfying $\eta(x^*) > 0$;
2. There exists an infinite sequence of successful major iterates $x^{(k_l)}$, indexed by $\mathcal{S} := \{k_1, k_2, \ldots\}$, and the linear independence constraint qualification (LICQ) fails to hold at the limit $x^* := \lim x^{(k_l)}$, which is a Fritz-John (FJ) point of (NLP);
3. There exists an infinite sequence of successful major iterates $x^{(k_l)}$, indexed by $\mathcal{S} := \{k_1, k_2, \ldots\}$, and LICQ holds at the limit $x^* := \lim x^{(k_l)}$, which is a Karush–Kuhn–Tucker point of (NLP).

Outcomes 1 and 3 are normal outcomes of NLP solvers in the sense that we cannot exclude the possibility that (NLP) is infeasible without making restrictive assumptions such as Slater's constraint qualification. Outcome 2 corresponds to the situation where a constraint qualification fails to hold at a limit point.

Outline of Convergence Proof. We start by showing that all iterates remain in a compact set. Next, we show that the algorithm is well defined by proving that the inner iteration is finite, which implies the existence of an infinite sequence of outer iterates $x^{(k)}$, unless the restoration phase fails or the algorithm converges finitely. We then show that the limit points are feasible and stationary. Finally, we show that the penalty estimate (3.6) is bounded.

We first show that all iterates remain in a compact set.

Lemma 2 *All major and minor iterates, $x^{(k)}$ and $\hat{x}^{(j)}$, remain in a compact set C.*

Proof The upper bound U on $\eta(x)$ implies that $\|c(x^{(k)})\| \le U$ for all k. The switching condition (3.7) implies that $\|c(\hat{x}^{(j)})\| \le U$ for all j. The feasibility restoration minimizes $\eta(x)$, implying that all its iterates in turn satisfy $\|c(x^{(k)})\| \le U$. Assumptions **A1** and **A2** now imply that the iterates remain in a bounded set C. □

The next lemma shows that the mechanism of the filter ensures that there exists a neighborhood of the origin in the filter that does not contain any filter points, as illustrated in Fig. 1.

Lemma 3 *There exists a neighborhood of* $(\eta, \omega) = (0, 0)$ *that does not contain any filter entries.*

Proof The mechanism of the algorithm ensures that $\eta_l > 0,\ \forall(\eta_l, \omega_l) \in \mathcal{F}_k$. First, assume that $\omega_{\min} := \min\{\omega_l : (\eta_l, \omega_l) \in \mathcal{F}_k\} > 0$. Then it follows that there exist no filter entries in the quadrilateral bounded by $(0, 0)$, $(0, \omega_{\min})$, $(\beta\eta_{\min}, \omega_{\min} - \gamma\beta\eta_{\min})$, $(\beta\eta_{\min}, 0)$, illustrated by the green area in Fig. 1. Next, if there exists a filter entry with $\omega_l = 0$, then define $\omega_{\min} := \min\{\omega_l > 0 : (\eta_l, \omega_l) \in \mathcal{F}_k\} > 0$, and observe that the quadrilateral $(0, 0)$, $(0, \omega_{\min})$, $(\beta\eta_{\min}, \omega_{\min})$, $(\beta\eta_{\min}, 0)$ contains no filter entries. In both cases, the area is nonempty, thus proving that there exists a neighborhood of $(0, 0)$ with filter-acceptable points. □

Next, we show that the inner iteration is finite and the algorithm is well defined.

Lemma 4 *The inner iteration is finite.*

Proof If the inner iteration finitely terminates with a filter-acceptable point or switches to the restoration phase, there is nothing to prove. Otherwise, there exists an infinite sequence of inner iterates $\hat{x}^{(j)}$ with $\hat{\eta}_j \leq \beta U$. Lemma 2 implies that this sequence has a limit point $x^* = \lim \hat{x}^{(j)}$. Because the penalty parameter and the multipliers are fixed during the inner iteration, we consider the sequence $\mathcal{L}_\rho(\hat{x}^{(j)}, y)$ for fixed $\rho = \rho_k$ and $y = y^{(k)}$. The sufficient reduction condition (3.10) implies that

$$\Delta\mathcal{L}_\rho^{(j)} := \mathcal{L}_\rho(\hat{x}^{(j)}, y) - \mathcal{L}_\rho(\hat{x}^{(j+1)}, y) \geq \sigma\hat{\omega}_j.$$

If the first-order error $\hat{\omega}_j \geq \overline{\omega} > 0$ is bounded away from zero, then this condition implies that $\mathcal{L}_\rho(x, y^{(k)})$ is unbounded below, which contradicts the fact that $f(x), \|c(x)\|$ are bounded by Assumption **A1** and Lemma 2. Thus, it follows that $\hat{\omega}_j \to 0$. If in addition $\hat{\eta}_j \to \hat{\eta} < \beta\eta_{\min}$, we must find a filter-acceptable point in the green region of Fig. 1, and terminate finitely. Otherwise, $\hat{\omega}_j \to 0$ and $\hat{\eta}_j \geq \beta\eta_{\min}$, which triggers the restoration phase after a finite number of steps. In either case, we exit the inner iteration according to Lemma 3. □

The next lemma shows that all limit points of the outer iteration are feasible.

Lemma 5 *Assume that there exist an infinite number of outer iterations. Then* $\eta(x^{(k)}) \to 0$.

Proof Every outer iteration for which $\eta_k > 0$ adds an entry to the filter. The proof follows directly from Chin and Fletcher (2003, Lemma 1). □

The next two lemmas show that the first-order error ω_k also converges to zero. We split the argument into two parts depending on whether the penalty parameter remains bounded or not.

Lemma 6 *Assume that the penalty parameter is bounded,* $\rho_k \leq \bar{\rho} < \infty$*, and consider an infinite sequence of outer iterations. Then* $\omega(x^{(k)}) \to 0$.

Proof Because the penalty parameter is bounded, it is updated only finitely often. Hence, we consider the tail of the sequence $x^{(k)}$ for which the penalty parameter has settled down, namely $\rho_k = \bar{\rho}$. We assume that $\omega_k \geq \bar{\omega} > 0$ and seek a contradiction. The sufficient reduction condition of the inner iteration (3.10) implies that

$$\Delta\mathcal{L}_{\bar{\rho},k}^{\text{in}} := \mathcal{L}_{\bar{\rho}}(x^{(k)}, y^{(k)}) - \mathcal{L}_{\bar{\rho}}(x^{(k+1)}, y^{(k)}) \geq \sigma\omega_k \geq \sigma\bar{\omega} > 0. \tag{4.1}$$

We now show that this "inner" sufficient reduction (for fixed $y^{(k)}$) implies an "outer" sufficient reduction. We combine (4.1) with the first-order multiplier update (1.4) and obtain

$$\Delta\mathcal{L}_{\bar{\rho},k}^{\text{out}} := \mathcal{L}_{\bar{\rho}}(x^{(k)}, y^{(k)}) - \mathcal{L}_{\bar{\rho}}(x^{(k+1)}, y^{(k+1)}) = \Delta\mathcal{L}_{\bar{\rho},k}^{\text{in}} - \bar{\rho}\|c(x^{(k+1)})\|_2^2 \geq \sigma\bar{\omega} - \bar{\rho}\eta_{k+1}^2. \tag{4.2}$$

Lemma 5 implies that $\eta_k \to 0$; hence, as soon as $\eta_{k+1} \leq \sigma\frac{\bar{\omega}}{2\bar{\rho}}$ for all k sufficiently large, we obtain the following sufficient reduction condition for the outer iteration:

$$\Delta\mathcal{L}_{\bar{\rho},k}^{\text{out}} \geq \sigma\frac{\bar{\omega}}{2},$$

for all k sufficiently large. Thus, if $\omega_k \geq \bar{\omega} > 0$ is bounded away from zero, it follows that the augmented Lagrangian must be unbounded below. However, because all $x^{(k)} \in C$ remain in a compact set, it follows from Assumption **A1** that $f(x)$ and $\|c(x)\|$ are bounded below and hence that $\mathcal{L}_{\bar{\rho}}(x, y)$ can be unbounded below only if $-y^T c(x)$ is unbounded below.

We now show by construction that there exists a constant $M > 0$ such that $c(x^{(k)})^T y^{(k)} \leq M$ for all major iterates. The first-order multiplier update implies that $y^{(k)} = y^{(0)} - \bar{\rho} \sum c^{(l)}$ and hence that

$$c(x^{(k)})^T y^{(k)} = \left(y^{(0)} - \bar{\rho} \sum_{l=1}^{k} c^{(l)} \right)^T c^{(k)} \leq \left(\|y^{(0)}\| + \bar{\rho} \sum_{l=1}^{k} \|c^{(l)}\| \right) \|c^{(k)}\|$$
$$= \left(y_0 + \bar{\rho} \sum_{l=1}^{k} \eta_l \right) \eta_k \, , \tag{4.3}$$

where $y_0 = \|y^{(0)}\|$, and we have assumed without loss of generality that $\bar{\rho}$ is fixed for the whole sequence. Now define $E_k := \max_{l \geq k} \eta_l$ and observe that $E_k \to 0$ from Lemma 5. The definition of the filter then implies that $E_{k+1} \leq \beta E_k$, and we obtain from (4.3) that

$$c(x^{(k)})^T y^{(k)} \leq \left(y_0 + \bar{\rho} \sum_{l=1}^{k} E_l \right) E_k = \left(y_0 + \bar{\rho} \sum_{l=1}^{k} \beta^l E_0 \right) \beta^k E_0$$
$$= \left(y_0 + \bar{\rho}\beta \frac{1 - \beta^k}{1 - \beta} E_0 \right) \beta^k E_0 < M.$$

Moreover, because $E_0 < \infty$, $\bar{\rho} < \infty$ and $0 < \beta < 1$, it follows that this expression is uniformly bounded as $k \to \infty$. Hence $c(x^{(k)})^T y^{(k)} \leq M$ for all k, and $\mathcal{L}_{\bar{\rho}}(x, y)$ must be bounded below, which contradicts the assumption that $\omega_k \geq \bar{\omega} > 0$ is bounded away from zero. It follows that $\omega_k \to 0$. □

We now consider the case where $\rho_k \to \infty$. In this case, we must assume that LICQ holds at every limit point. If LICQ fails at a limit point, then we cannot guarantee that the limit is a KKT point; it may be a Fritz-John point instead. The following lemma formalizes this result.

Lemma 7 *Consider the situation where $\rho_k \to \infty$. Then any limit point $x^{(k)} \to x^*$ is a Fritz-John point. If in addition LICQ holds at x^*, then it is a KKT point, and $\omega_k \to 0$.*

Proof Lemma 5 ensures that the limit point is feasible. Hence, it is trivially a Fritz-John point. Now assume that LICQ holds at x^*. We use standard augmented Lagrangian theory to show that this limit point also satisfies $\omega(x^*) = 0$. Following Theorem 2.5 of Friedlander (2002), we need to show that, for all restoration iterations $\mathcal{R} := \{k_1, k_2, k_3, \ldots\}$ on which we increase the penalty parameter, that the quantity

$$\sum_{l=1}^{\infty} \eta_{k_\nu + l}$$

remains bounded as $\nu \to \infty$. Similarly to the proof of Lemma 6, the filter acceptance ensures that $\eta_{k_\nu+l} \leq \beta^l \eta_{k_\nu}$, which gives the desired result. Thus, we can invoke Theorem 2.5 of Friedlander (2002), which shows that the limit point is a KKT point. □

The preceding lemmas are summarized in the following result.

Theorem 1 *Under Assumptions* **A1** *and* **A2**, *either Algorithm 3 terminates after a finite number of iterations at a KKT point, that is, for some finite k, $x^{(k)}$ is a first-order stationary point with $\eta(x^{(k)}) = 0$ and $\omega(x^{(k)}) = 0$, or there exists an infinite sequence of iterates $x^{(k)}$ and any limit point $x^{(k)} \to x^*$ that satisfy one of the following:*

1. *The penalty parameter is updated finitely often, and x^* is a KKT point;*
2. *There exists an infinite sequence of restoration steps at which the penalty parameter is updated. If x^* satisfies LICQ, it is a KKT point. Otherwise, it is an FJ point;*
3. *The restoration phase converges to a minimum of the constraint violation.*

Remark 1 We seem to be able to show that the limit point is a KKT point without assuming a constraint qualification, as long as the penalty parameter remains bounded. On the other hand, without a constraint qualification, we would expect the penalty parameter to be unbounded. It would be interesting to test these results in the context of mathematical programs with equilibrium constraints (MPECs). We suspect that MPECs that satisfy a strong-stationarity condition would have a bounded penalty, but those that do not have strongly stationary points would require the penalty to be unbounded.

Remark 2 The careful reader may wonder whether Algorithm 3 can cycle, because we do not add iterates to the filter for which $\eta_k = 0$. We can show, however, that this situation cannot happen. If we have an infinite sequence of iterates for which $\eta_k = 0$, the sufficient reduction condition (3.10) implies that we must converge to a stationary point, similarly to the arguments in Lemma 6. If we have a sequence that alternates between iterates for which $\eta_k = 0$ and iterates for which $\eta_k > 0$, we can never revisit any iterates for which $\eta_k > 0$ because those iterates have been added to the filter. By Lemma 5, any limit point is feasible. Thus, if LICQ holds, the limit is a KKT point; otherwise, it may be an FJ point. We observe that these conclusions are consistent with Theorem 1.

5 Numerical results

We have implemented a preliminary version of filter-al (Algorithm 3) in `C++`, using L-BFGS-B 3.0 (Zhu et al. 1997) to minimize the bound-constrained augmented Lagrangian and BQPD (Fletcher and Leyffer 1998) to solve the EQP step corresponding to solving the reduced KKT system. All experiments are run on a Lenovo Thinkpad X1 Carbon with an Intel Core i7 processor running at 2.6GHz and 16Gb RAM under the Ubuntu 18.04.2 LTS operating system. The convergence tolerance is $\epsilon = 10^{-6}$.

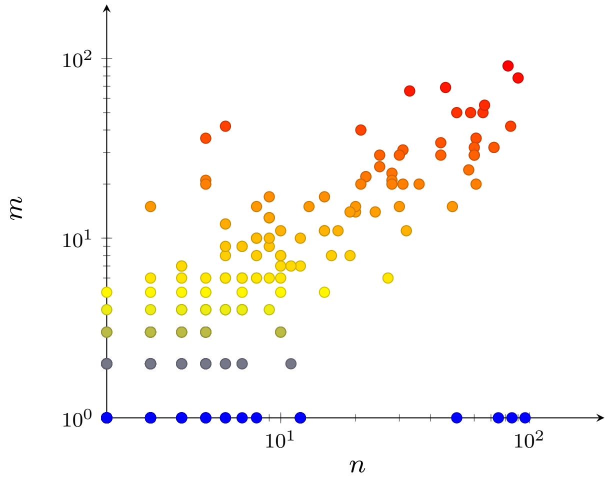

Fig. 2 Distribution of CUTEst test problems (n variables, m constraints)

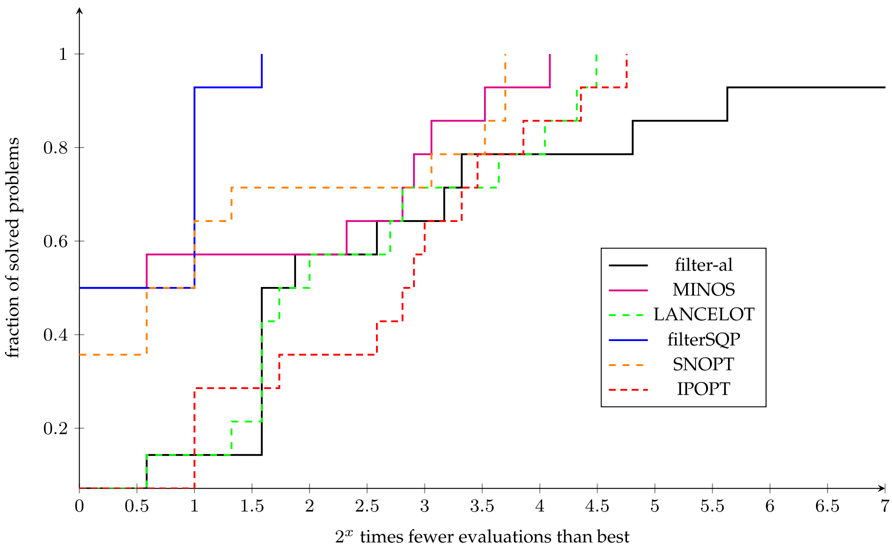

Fig. 3 Performance profile comparing number of objective evaluations of different NLP solvers for 14 small linear problems from CUTEst

We have chosen 429 small test problems from the CUTEst test set (Gould et al. 2015) that have up to 100 variables and/or constraints: 14 linear problems, 72 quadratic problems and 343 nonlinear problems. The distribution of the problem sizes is shown in Fig. 2.

We compare the performance of our filter augmented Lagrangian method, referred to as filter-al, with five other state-of-the-art NLP solvers:

1. FilterSQP (Fletcher and Leyffer 2002) is a filter SQP solver endowed with a trust-region mechanism to enforce convergence;

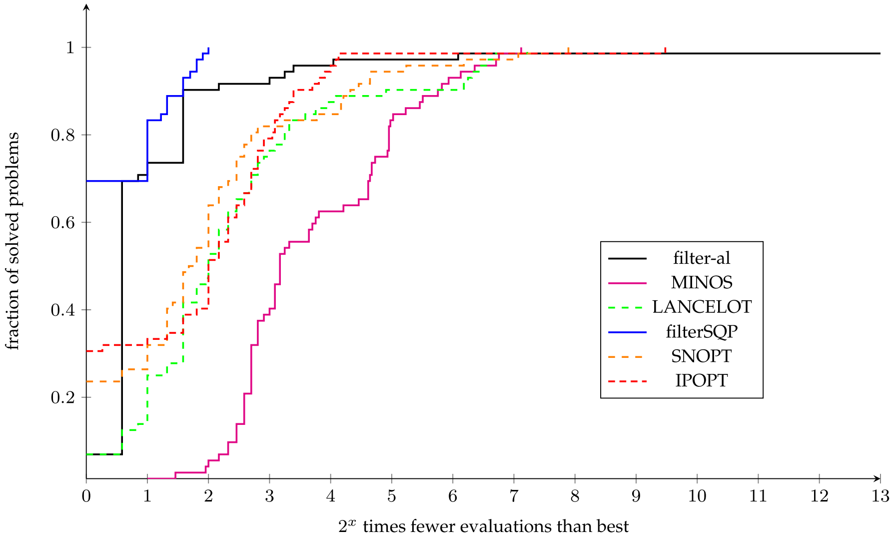

Fig. 4 Performance profile comparing number of objective evaluations of different NLP solvers for 72 small quadratic problems from CUTEst

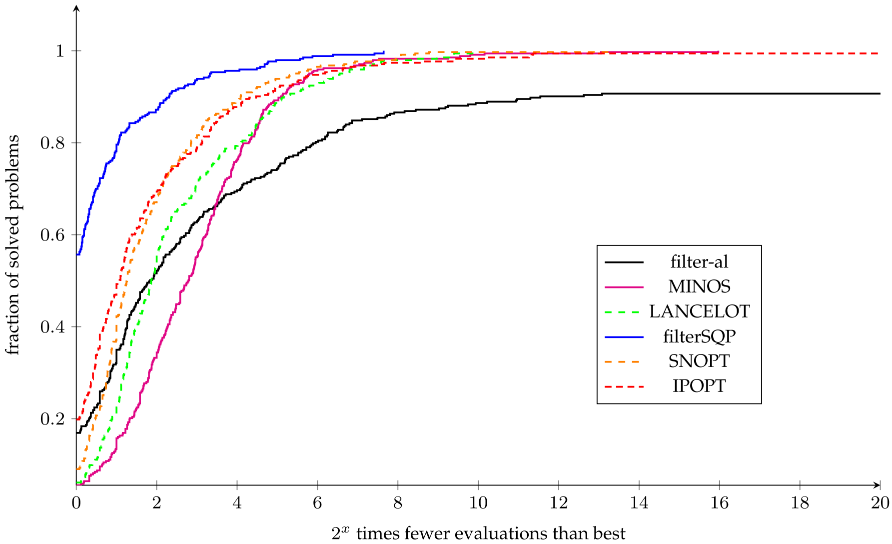

Fig. 5 Performance profile comparing number of objective evaluations of different NLP solvers for 343 small nonlinear (not QPs) problems from CUTEst

2. SNOPT (Gill et al. 2005) is an SQP method using limited-memory quasi-Newton approximations to the Hessian of the Lagrangian with an augmented Lagrangian merit function;
3. MINOS (Murtagh and Saunders 1993) implements a linearly-constrained augmented Lagrangian method with a line-search mechanism;

Table 1 Set of CUTEst problems for which filter-al does not converge

Problems	Termination/failure status
vion2, core1, cresc4, discs, dualc8, heart6, heart8, himmelbk, hs100mod, hs101, hs103, hs106, hs116, hs99exp, launch, minmaxbd, model, optmass, snake, spanhyd	max iterations reached by L-BFGS-B (1000)
chebyqad, palmer1a, palmer1b, palmer2a palmer2b, palmer3a, palmer3b, palmer4a palmer4b, palmer7a	IEEE error in evaluating gradient
palmer2, palmer3, palmer4	IEEE error in evaluating objective
polak3	IEEE error in evaluating constraint

4. IPOPT (Wächter and Biegler 2006) implements a filter interior-point method with a line-search mechanism;
5. LANCELOT (Conn et al. 1992) is a bound-constrained augmented Lagrangian method.

Because the test problems are small and our implementation is still preliminary, we only compare the number of function evaluations to solve a problem. This statistic is a good surrogate for the number of major iterations. Detailed results for LPs, QPs and NLPs are shown in Tables 2, 3 and 4, respectively, in the "Appendix". IPOPT fails to converge on the nonlinear problems argauss and lewispol.

In Figs. 3, 4 and 5, we summarize our numerical results using a performance profile (Dolan and Moré 2002). We observe that filter-al is competitive with the two SQP solvers, FilterSQP and SNOPT, which typically require the smallest number of iterations. This result is very encouraging, because while filter-al can in principle be parallelized by using parallel subproblem solvers, parallelizing an SQP method is significantly harder. Moreover, our new solver, filter-al, is also competitive with the two augmented Lagrangian methods, MINOS and LANCELOT, even though our implementation suffers from some large iteration counts of L-BFGS-B, which increases the number of function evaluations. This behavior indicates that the use of a filter provides a fast convergence mechanism, reducing the number of iterations.

Our preliminary implementation, filter-al, fails to converge on 34 test problems. We provide detailed comments on the type of failure in Table 1: 14 problems failed due to an IEEE exception during the function or gradient evaluation, which is outside the control of the solver, and 20 problems failed due to an error in the subproblem solver, L-BFGS-B.

6 Conclusions

We have introduced a new filter strategy for augmented Lagrangian methods that removes the need for the traditional forcing sequences. We prove convergence of our method to first-order stationary points of nonlinear programs under mild conditions, and we present a heuristic for adjusting the penalty parameter based on matrix-norm estimates. We show that second-order steps are readily integrated into our method to accelerate local convergence.

The proposed method is closely related to Newton's method in the case of equality constraints only. If no inequality constraints exist, that is if x is unrestricted in (NLP), then our algorithm reverts to standard Newton/SQP for equality constrained optimization with a line-search safeguard. In this case, we only need to compute the Cauchy point to the augmented Lagrangian step that is acceptable to the filter. Of course, a more direct implementation would be preferable.

Our proof leaves open a number of questions. We did not show second-order convergence, but we believe that such a proof follows directly if we use second-order correction steps as suggested in Wächter and Biegler (2005a), or if we employ a local non-monotone filter similar to Shen et al. (2012). We have presented preliminary numerical results on 429 small CUTEst test problems that show that our new augmented Lagrangian filter method outperforms other augmented Lagrangian solvers, and is competitive with SQP methods in terms of major iterations.

Acknowledgements We are grateful to two anonymous referees and an associate editor for their careful and detailed refereeing work that greatly helped us improve the final manuscript. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357. This work was also supported by the U.S. Department of Energy through Grant DE-FG02-05ER25694.

Appendix

See Tables 2, 3 and 4.

Table 2 Number of evaluations of nonlinear solvers on a subset of linear CUTEst problems

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
booth	3	2	2	2	1	3
degenlpa	99	2	26	29	15	25
degenlpb	20	2	26	41	23	45
extrasim	3	2	1	6	0	3
goffin	11	3	25	10	25	10
himmelba	3	2	2	2	0	3
linspanh	56	2	5	54	14	13
makela4	9	3	1	8	1	20
model		2	23	15	34	33
res	1	2	0	10	5	1
simpllpa	6	2	3	14	3	5
simpllpb	6	2	1	11	1	4
supersim	3	2	1	2	1	7
zangwil3	3	2	3	2	2	3

Table 3 Number of evaluations of nonlinear solvers on a subset of quadratic CUTEst problems

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
3pk	3	7	41	12	210	47
arglinb	3	2	1	3	9	2
arglinc	3	2	1	3	9	2
avgasa	9	2	9	10	20	12
avgasb	21	2	9	13	16	11
biggsc4	16	2	13	35	17	15
bqp1var	2	2	1	6	6	3
bqpgabim	3	2	36	21	108	6
bqpgasim	3	2	40	21	122	6
bt3	3	2	5	2	13	6
deconvb	8	30	302	5705	31	31
dixon3dq	3	2	10	2	44	4
dual1	3	2	475	17	278	9
dual2	3	2	269	14	216	9
dual4	3	2	145	14	209	8
dualc1	19	2	7	30	25	17
dualc2	33	2	5	28	15	19
dualc5	136	2	7	11	25	8
dualc8		2	9	13	17	24
fccu	3	4	19	2	49	13
genhs28	3	2	11	2	13	5
hatfldc	9	5	16	6	20	9
hatfldh	3	2	3	19	9	13
hilberta	3	2	2	2	13	60
hilbertb	3	2	50	2	113	5
hs003	3	2	2	5	13	16
hs021	3	2	1	9	9	3
hs028	3	2	4	2	13	4
hs035	3	2	5	8	13	8
hs044	3	2	2	20	11	11
hs048	3	2	6	2	17	3
hs051	3	2	6	2	13	10
hs052	3	2	5	2	12	6
hs053	3	2	2	7	13	6
hs054	3	2	5	8	17	9
hs076	3	2	4	8	14	9
hs118	3	3	21	12	42	19

Table 3 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
hs21mod	3	2	1	17	9	3
hs268	6	2	6	17	37	27
hs35mod	3	2	1	16	7	3
hs3mod	3	2	5	6	14	4
hs44new	3	2	4	14	11	10
lotschd	3	3	8	15	6	9
lsqfit	3	2	3	8	12	7
maratosb	8	13	4	33	11	8
nasty	3	2	1	2	7	5
obstclal	6	2	37	15	91	7
obstclbl	6	2	44	13	88	3
obstclbu	6	2	36	13	75	2
oslbqp	3	2	6	15	12	3
palmer1c	3	7	8	2	62	145
palmer1d	3	6	7	2	51	34
palmer2c	3	5	8	2	63	298
palmer3c	3	5	8	2	62	206
palmer4c	3	6	8	2	62	176
palmer5c	3	4	6	2	27	2
palmer5d	3	5	4	2	26	2
palmer6c	3	6	8	2	61	159
palmer7c	3	8	8	2	65	189
palmer8c	3	7	8	2	62	152
portfl1	3	2	12	10	49	20
portfl2	3	2	12	9	51	14
portfl3	3	2	13	11	53	14
portfl4	3	2	11	10	50	19
portfl6	3	2	11	9	49	20
qudlin	2	2	11	26	19	2
sim2bqp	3	2	2	8	10	3
simbqp	3	2	2	8	11	2
tame	3	2	1	6	9	2
tointqor	3	2	50	2	164	8
zangwil2	3	2	2	2	12	4
zecevic2	6	2	2	9	10	7

Table 4 Number of evaluations of nonlinear solvers on a subset of nonlinear CUTEst problems

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
aircrfta	8	4	4	4	8	10
aircrftb	13	21	58	19	66	27
airport	50	13	58	16	528	69
aljazzaf	6	15	145	82	65	24
allinitc	64	24	105	44	56	76
allinit	15	11	17	19	30	13
allinitu	15	12	14	15	21	15
alsotame	3	5	6	9	12	11
argauss	3	1	9		17	13
avion2		19	19	143	18	787
bard	42	11	23	9	37	15
batch	293	9	33	34	380	1000
beale	29	10	15	19	25	21
biggs3	12	11	24	28	32	40
biggs5	48	50	107	36	39	64
biggs6	62	83	120	50	120	103
box2	3	9	10	9	11	20
box3	12	8	24	15	16	31
brkmcc	6	4	10	4	14	7
brownal	15	8	21	8	78	24
brownbs	57	48	32	8	36	7
brownden	21	9	40	9	42	9
bt10	11	7	23	7	11	20
bt11	16	7	12	9	57	22
bt12	12	5	10	5	60	11
bt13	74	48	33	25	207	1001
bt1	62	1	12	15	17	19
bt2	93	13	18	13	386	36
bt4	23	11	10	10	48	25
bt5	27	9	11	8	173	20
bt6	118	12	14	18	118	25
bt7	634	19	36	30	86	49
bt8	92	12	14	52	22	30
bt9	36	23	28	14	78	22
byrdsphr	189	11	200	19	97	43
camel6	8	8	19	11	25	8
cantilvr	29	16	27	12	176	27
catena	254	13	145	7	418	56

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
cb2	25	7	12	9	78	18
cb3	16	7	16	10	72	18
chaconn1	33	5	12	7	55	12
chaconn2	9	5	7	7	55	11
chebyqad		50	3	168	6	62
chnrosnb	40	59	170	92	593	68
cliff	26	28	28	24	47	28
cluster	18	10	9	10	22	45
concon	100	5	8	10	1	676
congigmz	131	4	23	33	43	30
coolhans	15	3	12	10	7	281
core1		6	12	105	151	1001
coshfun	1274	303	233	1039	2508	154
cresc4		52	93	269	865	1001
csfi1	317	18	48	12	10	155
csfi2	57	8	84	86	10	180
cube	10	41	42	58	67	52
dallass	586	56	109	29	141	1001
deconvc	85	58	81	99	88	43
deconvu	45	971	152	687	31	69
demymalo	45	8	18	12	95	28
denschna	9	7	12	7	24	13
denschnb	9	10	10	25	18	11
denschnc	9	11	21	11	31	13
denschnd	29	43	77	27	112	65
denschne	15	11	44	25	35	16
denschnf	9	7	12	7	22	8
dipigri	1508	13	23	22	130	63
disc2	243	25	522	48	706	20
discs		40	4893	186	629	422
dixchlng	404	12	31	11	1582	44
djtl	84	29	1345	861	89	100
dnieper	9	4	13	31	37	75
eg1	6	8	9	8	15	9
eigencco	45	29	34	14	159	17
eigmaxc	80	7	11	7	373	21
eigminc	30	7	10	8	120	11
engval2	48	20	34	33	67	30

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
errinros	22	53	267	70	553	76
expfita	36	13	24	31	29	54
expfit	30	13	18	9	29	11
extrosnb	1	2	2	1	6	1
fletcher	46	1	2	28	9	28
genhumps	63	188	77	321	170	134
gigomez1	143	8	17	19	141	33
gottfr	16	13	8	9	10	35
gridnetg	24	4	22	11	43	21
gridneth	14	5	36	7	115	20
gridneti	19	5	47	14	112	22
growthls	49	106	184	171	259	178
growth	49	106	187	171	259	178
gulf	33	26	66	44	696	63
hadamals	21	13	19	128	267	20
haifas	728	13	23	10	144	27
hairy	17	84	35	96	60	102
haldmads	23536	41	12	23	129	45
hart6	8	11	16	14	37	9
hatflda	30	15	30	11	45	47
hatfldb	27	11	28	11	33	28
hatfldd	18	23	29	27	49	66
hatflde	18	26	31	32	64	57
hatfldf	246	15	4148	1335	12	113
hatfldg	829	15	17	20	11	29
heart6ls	20	1001	1351	1588	22799	1001
heart6		16	401	3002	129	1001
heart8ls	445	217	1313	189	475	238
heart8		12	133	40	157	359
helix	9	19	28	25	54	18
himmelbb	12	25	8	12	21	11
himmelbc	10	8	7	9	9	11
himmelbd	198	4	27	79	64	52
himmelbe	15	2	2	3	3	8
himmelbf	14	8	53	11	48	29
himmelbg	10	10	12	14	18	16
himmelbh	6	8	10	24	11	7
himmelbk		6	11	19	172	206
himmelp1	12	9	19	12	21	29

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
himmelp2	230	9	32	19	152	275
himmelp3	690	5	8	13	120	870
himmelp4	1204	5	8	25	116	737
himmelp5	2680	12	44	543	76	273
himmelp6	208	2	2	12	6	2
hong	6	5	4	13	15	6
hs001	8	36	48	53	10	41
hs002	29	9	15	17	13	7
hs004	2	3	4	6	7	2
hs005	9	11	9	9	14	9
hs006	117	3	9	7	91	63
hs007	24	13	30	28	65	26
hs008	14	6	6	6	8	13
hs009	30	5	10	6	12	22
hs010	31	10	31	13	59	18
hs011	12	6	15	9	47	16
hs012	72	8	11	9	159	26
hs013	79	34	17	79	54	60
hs014	10	6	10	8	10	13
hs015	28	7	11	21	86	47
hs016	19	5	5	23	10	19
hs017	42	8	19	18	12	20
hs018	295	7	32	27	94	117
hs019	39	7	9	16	57	45
hs020	12	5	5	7	9	23
hs022	9	2	7	7	48	10
hs023	370	7	7	12	54	51
hs024	6	3	8	13	9	14
hs025	1	27	2	44	6	1
hs026	230	18	27	26	77	41
hs027	18	8	21	143	137	31
hs029	77	8	14	9	174	18
hs030	6	2	5	26	30	8
hs031	10	6	11	8	29	12
hs032	6	2	5	20	15	7
hs033	9	5	9	16	39	9
hs034	15	8	7	10	33	21
hs036	6	3	10	13	9	7
hs037	9	6	10	13	17	13
hs038	7	54	101	78	89	56

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
hs039	36	23	28	14	78	22
hs040	10	5	9	4	22	11
hs041	6	2	7	12	6	7
hs042	9	6	10	7	22	13
hs043	37	11	11	10	103	25
hs045	1	2	2	48	6	1
hs046	53	19	32	20	122	28
hs047	335	21	28	21	126	29
hs049	30	17	34	20	62	38
hs050	15	9	20	10	25	11
hs055	14	2	2	4	6	7
hs056	22	19	52	40	53	12
hs057	9	5	62	28	29	2
hs059	1000	11	22	72	128	340
hs060	35	7	13	8	121	18
hs061	24	1	39	10	71	19
hs062	53	10	16	9	19	37
hs063	180	1	18	8	120	21
hs064	30	13	28	18	99	38
hs065	169	5	11	91	320	42
hs066	9	14	6	8	21	11
hs067	6234	12	32	12	61	287
hs070	149	42	34	36	67	39
hs071	21	6	8	9	56	16
hs072	30	15	35	17	153	65
hs073	8	4	7	9	16	18
hs074	1274	6	15	10	28	13
hs075	12	5	12	10	19	110
hs077	54	14	16	13	124	27
hs078	27	5	7	5	58	12
hs079	40	5	12	5	55	14
hs080	92	8	9	7	49	13
hs081	91	38	11	8	56	17
hs083	42	5	8	15	12	16
hs084	339	6	58	12	46	47
hs085	48	1	9904	127	893	1001
hs086	20	5	16	11	14	15
hs087	107	7	11	18	51	32
hs088	32	19	59	18	67	56

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
hs089	29	31	85	38	198	61
hs090	35	2	55	28	93	58
hs091	39	337	73	15	216	62
hs092	35	2	56	25	111	58
hs093	4	2	33	10	46	4
hs095	1956	3	2	18	4	24
hs096	1213	3	2	24	4	23
hs097	74	7	36	24	106	19
hs098	255	7	36	21	87	19
hs099	63	9	19	7	61	997
hs100lnp	406	14	33	21	134	32
hs100mod		14	32	27	124	137
hs100	1508	13	23	22	130	63
hs101		34	530	273	5495	1001
hs102	3669	42	238	36	981	1001
hs103		28	177	64	1419	1001
hs104	239	23	29	11	86	80
hs105	688	9	89	31	115	1001
hs106		17	13	15	504	1001
hs107	317	6	14	12	21	26
hs108	286	36	152	17	165	43
hs109	6020	7	349	44	354	1000
hs110	6	5	11	7	44	5
hs111lnp	52	31	64	16	389	57
hs111	72	31	70	16	389	46
hs112	140	12	35	18	93	47
hs113	2086	6	28	12	147	97
hs114	1194	1	9	73	5	664
hs116		14	22	26	97	1003
hs117	224	6	20	23	158	66
hs119	19	7	22	15	30	28
hs99exp		12	42	30	213	1001
hubfit	3	2	8	9	12	8
humps	207	1001	257	571	194	1001
hypcir	9	8	5	8	8	10
jensmp	73	11	36	10	56	10
kiwcresc	45	11	17	11	81	23
kowosb	18	18	33	23	40	24
lakes	18	63	39	20	1045	1001

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
launch		1	246	673	50	1001
lewispol	6307	1	6		38	22
loadbal	48	8	58	18	131	62
loghairy	42	1001	249	4173	403	1001
logros	82	50	109	358	148	66
lootsma	9	5	9	16	39	9
lsnnodoc	9	7	8	15	8	11
madsen	106	25	12	25	72	26
makela1	36	15	8	19	69	19
makela2	54	5	26	8	59	40
makela3	205	25	50	17	238	125
maratos	12	10	9	5	21	9
matrix2	26	12	14	21	66	13
maxlika	688	9	89	31	115	1001
mconcon	100	5	8	10	1	676
mdhole	12	56	70	106	117	68
methanb8	80	47	306	9	176	221
methanl8	6000	96	499	82	670	640
mexhat	6	10	37	5	19	4
meyer3	36	280	6159	494	776	559
mifflin1	12	23	8	7	36	18
mifflin2	145	10	17	16	124	50
minmaxbd		9	71	78	356	592
minmaxrb	173	3	8	11	112	81
minsurf	9	10	24	21	197	15
mistake	96	18	19	15	160	30
mwright	12	12	12	11	38	19
nonmsqrt	73	716	7413	83890	1491	167
nuffield_continuum	6	5	9	7	16	15
odfits	22	7	17	11	29	49
optcntrl	1149	4	5	134	16	352
optmass		18	2	23	332	1001
optprloc	388	6	12	19	686	438
orthregb	5969	2	9	3	263	64
orthrege	190	180	31	77	858	795
osbornea	17	998	120	152	128	57
osborneb	2222	20	82	25	128	45
palmer1a		51	205	71	12	102
palmer1b		21	87	26	12	55

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
palmer1e	30	74	186	122	150	353
palmer1	33	33	30	1854	40	28
palmer2a		68	115	392	198	211
palmer2b		16	61	34	8	77
palmer2e	268	86	191	52	346	133
palmer2		33	44	63	8	34
palmer3a		82	136	199	8	201
palmer3b		21	54	15	17	36
palmer3e	57	117	293	133	427	1001
palmer3		12	13	553	12	58
palmer4a		52	109	133	8	93
palmer4b		21	52	31	17	64
palmer4e	175	24	123	38	211	123
palmer4		12	14	1182	12	142
palmer5a	5	1001	1303	12804	321079	1001
palmer5b	5	855	1343	208	3729	961
palmer5e	6	3	1313	7933	25502	8
palmer6a	5	137	202	263	271	277
palmer6e	5	38	198	59	260	47
palmer7a		991	1299	8073	21	1001
palmer7e	5	1001	1315	12230	925	39
palmer8a	9	50	127	102	104	61
palmer8e	55	29	92	31	122	86
pentagon	27	12	15	20	23	48
pfit1ls	22	566	480	678	718	453
pfit1	22	566	480	678	718	453
pfit2ls	21	210	175	184	238	217
pfit2	21	210	175	184	238	217
pfit3ls	33	139	289	337	475	272
pfit3	33	139	289	337	475	272
pfit4ls	32	126	470	548	755	410
pfit4	32	126	470	548	755	410
polak1	89	8	17	7	86	37
polak2	124	10	225	15	350	321
polak3		24	208	4862	88	234
polak4	33	5	6	10	76	16
polak5	21	45	58	33	68	7
polak6	2592	29	108	301	248	644

Table 4 continued

Problem	filter-al	filterSQP	SNOPT	IPOPT	MINOS	LANCELOT
powellbs	30	16	15	12	15	106
powellsq	76	4	217	109	52	23
prodpl0	21	9	14	16	41	35
prodpl1	15	7	10	17	70	27
pspdoc	10	7	15	15	26	10
recipe	11	2	2	3	3	12
rk23	18015	9	11	12	58	54
robot	286	45	18	10	378	33
rosenbr	44	29	45	45	10	36
rosenmmx	1522	37	41	22	167	226
s365mod	835	86	31	43	540	91
sineali	12	1001	1351	8025	7334	1001
sineval	51	62	94	110	124	75
sisser	9	19	15	21	18	38
snake		3	6	14	21	1001
spanhyd		11	13	24	61	26
spiral	245	152	107	64	392	96
ssnlbeam	1032	5	36	22	125	39
stancmin	14	2	5	11	6	9
swopf	355	6	24	17	137	290
synthes1	17	5	10	10	22	13
try-b	18	8	10	20	11	13
twobars	15	8	11	10	32	13
vanderm4	2514	1	57	51	34	36
watson	17	21	172	14	118	44
weeds	25965	39	51	32	9	3
womflet	412	9	27	12	135	97
yfit	6	48	95	185	131	103
yfitu	27901	48	95	69	131	103
zecevic3	19	9	11	22	39	19
zecevic4	15	6	8	10	23	12
zigzag	287	11	23	23	219	43
zy2	9	5	9	10	44	9

References

Abhishek K, Leyffer S, Linderoth JT (2010) FilMINT: an outer-approximation-based solver for nonlinear mixed integer programs. INFORMS J Comput 22:555–567

Altay-Salih A, Pinar MC, Leyffer S (2003) Constrained nonlinear programming for volatility estimation with GARCH models. SIAM Rev 45(3):485–503

Arrieta-Camacho JJ, Biegler LT, Subramanian D (2007) NMPC-based real-time coordination of multiple aircraft. In: Allgower RF, Biegler LT (eds) Assessment and future directions of nonlinear model predictive control. Springer, Heidelberg, pp 629–639

Bautista G, Anjos MF, Vannelli A (2007) Formulation of oligopolistic competition in AC power networks: an NLP approach. IEEE Trans Power Syst 22(1):105–115

Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed-integer nonlinear optimization. Acta Numer 22(5):1–131

Benson H, Shanno D (2007) An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. Comput Optim Appl 38:371–399. https://doi.org/10.1007/s10589-007-9048-6

Benson HY, Shanno DF (2008) Interior-point methods for nonconvex nonlinear programming: regularization and warmstarts. Comput Optim Appl 40(2):143–189

Benson HY, Shanno DF, Vanderbei RJ (2002) Interior-point methods for nonconvex nonlinear programming: filter-methods and merit functions. Comput Optim Appl 23(2):257–272

Bertsekas DP (1982) Constrained optimization and lagrange multiplier methods. Athena Scientific, New York

Betts JT (2001) Practical methods for optimal control using nonlinear programming. Advances in design and control. SIAM, Philadelphia

Biegler LT, Grossmann IE (2004a) Challenges and research issues for product and process design optimization. In: Floudas CA, Agrawal R (eds) Proceedings of foundations of computer aided process design. CACHE Corporation, Austin

Biegler LT, Grossmann IE (2004b) Part I: retrospective on optimization. Comput Chem Eng 28(8):1169–1192

Birgin EG, Martinez JM (2008) Improving ultimate convergence of an augmented Lagrangian method. Optim Methods Softw 23(2):177–195

Birgin EG, Martínez JM (2012) Augmented lagrangian method with nonmonotone penalty parameters for constrained optimization. Comput Optim Appl 51(3):941–965

Birgin EG, Martínez JM (2014) Practical augmented Lagrangian methods for constrained optimization, vol 10. SIAM, Philadelphia

Bonami P, Biegler LT, Conn AR, Cornuejols G, Grossmann IE, Laird CD, Lee J, Lodi A, Margot F, Sawaya N, Wähter A (2005) An algorithmic framework for convex mixed integer nonlinear programs. Research report RC 23771, IBM T. J. Watson Research Center, Yorktown, NY

Bonami P, Lee J, Leyffer S, Waechter A (2011) More branch-and-bound experiments in convex nonlinear integer programming. Technical report ANL/MCS-P1949-0911, Argonne National Laboratory, Mathematics and Computer Science Division, Lemont, IL

Bonnans JF, André J (2009) Optimal structure of gas transmission trunklines. Technical report 6791, INRIA Saclay, 91893 ORSAY Cedex, France

Bonnard B, Faubourg L, Launay G, Trélat E (2003) Optimal control with state constraints and the space shuttle re-entry problem. J Dyn Control Syst 9(2):155–199

Borghetti A, Frangioni A, Lacalandra F, Nucci CA (2003) Lagrangian heuristics based on disaggregated bundle methods for hydrothermal unit commitment. IEEE Trans Power Syst 18:1–10

Byrd RH, Hribar ME, Nocedal J (1999) An interior point algorithm for large scale nonlinear programming. SIAM J Optim 9(4):877–900

Byrd RH, Gould NIM, Nocedal J, Waltz RA (2004) An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. Math Progr Ser B 100(1):27–48

Byrd RH, Nocedal J, Waltz RA (2006) KNITRO: an integrated package for nonlinear optimization. In: di Pillo G, Roma M (eds) Large-scale nonlinear optimization. Springer, New York, pp 35–59

Calamai PH, Moré JJ (1987) Projected gradient methods for linearly constrained problems. Math Progr 39:93–116

Castro J, Gonzalez JA (2004) A nonlinear optimization package for long-term hydrothermal coordination. Eur J Oper Res 154(1):641–658

Chin CM, Fletcher R (2003) On the global convergence of an SLP-filter algorithm that takes EQP steps. Math Progr 96(1):161–177

Coleman TF, Li Y, Verman A (1999) Reconstructing the unknown volatility function. J Comput Finance 2(3):77–102

Conn AR, Gould NIM, Toint PL (1992) LANCELOT: a Fortran package for large-scale nonlinear optimization (release A). Springer, Heidelberg

Conn AR, Gould NIM, Toint PL (2000) Trust-region methods. MPS-SIAM series on optimization. SIAM, Philadelphia

Cornuejols G, Tütüncü R (2007) Optimization methods in finance. Cambridge University Press, Cambridge

Curtis FE, Jiang H, Robinson DP (2015) An adaptive augmented Lagrangian method for large-scale constrained optimization. Math Progr 152(1–2):201–245

Dolan ED, Moré J (2002) Benchmarking optimization software with performance profiles. Math Progr 91(2):201–213

Donde V, Lopez V, Lesieutre B, Pinar A, Yang C, Meza J (2005) Identification of severe multiple contingencies in electric power networks. In: Proceedings 37th North American power symposium, LBNL-57994

Klaus E, Steinbach Marc C (2005) Nonlinear optimization in gas networks. In: Bock HG, Kostina E, Phu HX, Ranacher R (eds) Modeling, simulation and optimization of complex processes. Springer, Berlin, pp 139–148

Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. Math Progr 66:327–349

Fletcher R, Leyffer S (1998) User manual for filterSQP. Numerical analysis report NA/181, University of Dundee

Fletcher R, Leyffer S (2002) Nonlinear programming without a penalty function. Math Progr 91:239–270

Fletcher R, Leyffer S (2003) Filter-type algorithms for solving systems of algebraic equations and inequalities. In: di Pillo G, Murli A (eds) High performance algorithms and software for nonlinear optimization. Kluwer, Dordrecht, pp 259–278

Fletcher R, Sainz de la Maza E (1989) Nonlinear programming and nonsmooth optimization by successive linear programming. Math Progr 43:235–256

Fletcher R, Leyffer S, Toint PL (2002) On the global convergence of a filter-SQP algorithm. SIAM J Optim 13(1):44–59

Forsgren A, Gill PE, Wright MH (2002) Interior methods for nonlinear optimization. SIAM Rev 4(4):525–597

Friedlander MP (2002) A globally convergent linearly constrained Lagrangian method for nonlinear optimization. Ph.D. thesis, Stanford University

Friedlander MP, Leyffer S (2008) Global and finite termination of a two-phase augmented Lagrangian filter method for general quadratic programs. SIAM J Sci Comput 30(4):1706–1729

Ghaoui LE, Oks M, Oustry F (2003) Worst-case value-at-risk and robust portfolio optimization: a conic programming approach. Oper Res 51(4):509–679

Gill PE, Murray W, Saunders MA (1997) User's guide for SQOPT 5.3: a Fortran package for large-scale linear and quadratic programming. Technical report NA 97-4, Department of Mathematics, University of California, San Diego

Gill PE, Murray W, Saunders MA (2002) SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM J Optim 12(4):979–1006

Gill PE, Murray W, Saunders MA (2005) SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM Rev 47(1):99–131

Gondzio J, Grothey A (2002) Reoptimization with the primal-dual interior point method. SIAM J Optim 13(3):842–864

Gould NIM, Orban D, Toint PL (2015) Cutest: a constrained and unconstrained testing environment with safe threads for mathematical optimization. Comput Optim Appl 60(3):545–557

Goux J-P, Leyffer S (2002) Solving large MINLPs on computational grids. Optim Eng 3:327–346

Huangfu Q, Hall JAJ (2013) Novel update techniques for the revised simplex method. Technical report ERGO-13-001, The School of Mathematics, University of Edinburgh, Edinburgh, UK

Kawayir Y, Laird C, Wächter A (2009) Introduction to Ipopt: a tutorial for downloading, installing, and using Ipopt. Manual. IBM T.J. Watson Research Center, Yorktown Heights

Konno H, Yamazaki H (1991) Mean-absolute deviation portfolio optimization model and its application to Tokyo stock market. Manag Sci 37:519–531

Leyffer S (2001) Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput Optim Appl 18:295–309

Leyffer S, Munson TS (2007) A globally convergent filter method for MPECs. Preprint ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division

Leyffer S, Lopez-Calva G, Nocedal J (2006) Interior methods for mathematical programs with complementarity constraints. SIAM J Optim 17(1):52–77

Lubin M, Hall JAJ, Petra CG, Anitescu M (2013) Parallel distributed-memory simplex for large-scale stochastic LP problems. Comput Optim Appl 55(3):571–596

Martin A, Möller M, Moritz S (2006) Mixed integer models for the stationary case of gas network optimization. Math Progr 105:563–582

Momoh JA, Koessler RJ, Bond MS, Stott B, Sun D, Papalexopoulos A, Ristanovic P (1997) Challenges to optimal power flow. IEEE Trans Power Syst 12:444–455

Moré JJ, Toraldo G (1991) On the solution of quadratic programming problems with bound constraints. SIAM J Optim 1(1):93–113

Murtagh BA, Saunders MA (1993) MINOS 5.4 user's guide. Report SOL 83-20R, Department of Operations Research, Stanford University

Murtagh BA, Saunders MA (1982) A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. In: Buckley AG, Goffin J-L (eds) Algorithms for constrained minimization of smooth nonlinear functions, vol 16. Mathematical programming studies, vol 16. Springer, Berlin, pp 84–117

Penfield P, Spence R, Duinker S (1970) Tellegen's theorem and electrical networks. MIT Press, Cambridge

Powell MJD (1978) Algorithms for nonlinear constraints that use Lagrangian functions. Math Progr 14(1):224–248

Rabinowitz G, Mehrez A, Oron G (1988) A nonlinear optimization model of water allocation for hydroelectric energy production and irrigation. Manag Sci 34(8):973–990

Raghunathan A, Biegler LT (2005) An interior point method for mathematical programs with complementarity constraints (MPCCs). SIAM J Optim 15(3):720–750

Sheble GB, Fahd GN (1994) Unit commitment literature synopsis. IEEE Trans Power Syst 9:128–135

Shen C, Leyffer S, Fletcher R (2012) A nonmonotone filter method for nonlinear optimization. Comput Optim Appl 52(3):583–607

Smith E, Hall JAJ (2012) A high performance primal revised simplex solver for row-linked block angular linear programming problems. Technical report ERGO-12-003, The School of Mathematics, University of Edinburgh, Edinburgh, UK

Vanderbei RJ, Shanno DF (1999) An interior point algorithm for nonconvex nonlinear programming. COAP 13:231–252

Wächter A, Biegler LT (2005a) Line search filter methods for nonlinear programming: local convergence. SIAM J Optim 16(1):32–48

Wächter A, Biegler LT (2005b) Line search filter methods for nonlinear programming: motivation and global convergence. SIAM J Optim 16(1):1–31

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Progr 106(1):25–57

Womersley RS, Lau K (1996) Portfolio optimisation problems. In: May RL, Easton AK (eds) Computational techniques and applications: CTAC95. World Scientific Publishing Co., Singapore, pp 795–802

Zhu C, Byrd RH, Lu P, Nocedal J (1997) Algorithm 778: L-BFGS-B—Fortran subroutines for large-scale bound-constrained optimization. ACM Trans Math Softw (TOMS) 23(4):550–560

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.