# An Augmented Lagrangian Filter Method[*]

Sven Leyffer[†] Charlie Vanaret[‡]

July 29, 2019

**Abstract**

**Changes to previous version are marked in red.**

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. R.Ed

**Keywords: Augmented Lagrangian, filter methods, nonlinear optimization.**

**AMS-MSC2000: 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 [4, 31, 57, 61, 65], gas transmission networks [33, 56, 16], the coordination of hydroelectric energy [23, 18, 63], and finance [28], including portfolio allocation [51, 43, 72] and volatility estimation [25, 2]. Chemical engineering has traditionally been at the forefront of developing new applications and algorithms for nonlinear optimization; see the surveys [11, 12]. Applications in chemical engineering include process flowsheet design, mixing, blending, and equilibrium models. Another area with a rich set of applications is optimal control [10]; optimal control applications include the control of chemical reactions, the shuttle re-entry problem [17, 10], and the control of multiple airplanes [3]. More importantly, nonlinear optimization is a basic building block of more complex design and optimization paradigms, such as mixed-integer R.1
nonlinear optimization [1, 34, 48, 52, 14, 5] and optimization problems with complementarity constraints [53, 64, 54].

[*] Preprint ANL/MCS-P6082-1116

[†] Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL 60439, USA, leyffer@mcs.anl.gov.

[‡] Fraunhofer ITWM, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany, charlie.vanaret@itwm.fraunhofer.de

Nonlinearly constrained optimization has been studied intensely for more than 30 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 [44, 45, 35, 19, 24, 39] and interior-point methods [40, 50, 70, 69, 8, 68, 20, 21]. 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 [46, 6, 7], interior-point methods cannot compete with active-set approaches when solving mixed-integer nonlinear programs [15]. 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 R.1
programming (LP) active-set solvers can be found in the series of papers [49, 67, 55]). 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):

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

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, the KKT conditions are

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

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 $\rho$. 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 [9, 62, 60]. Recently, re- R.2.2
searchers have expressed renewed interest in augmented Lagrangian methods because of their
good scalability properties, which had already been observed in [26]. The key computational step
in bound-constrained augmented Lagrangian methods, such as LANCELOT, [26], is the mini- R.2.3
mization of $x \mapsto \mathcal{L}_{\rho_k}(x, y^{(k)})$ for given $\rho_k > 0$ and $y^{(k)} \in \mathbb{R}^m$, giving rise to the bound-constrained R.2.3
Lagrangian (BCL($y^{(k)}, \rho_k$)) problem:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \mathcal{L}_{\rho_k}(x, y^{(k)}) \\ \text{subject to} & l \le x \le u. \end{array} \qquad (\text{BCL}(y^{(k)}, \rho_k))$$

We denote the solution of (BCL($y^{(k)}, \rho_k$)) by $x^{(k+1)}$. A basic augmented Lagrangian method solves (BCL($y^{(k)}, \rho_k$)) approximately and updates the multipliers using the so-called first-order multiplier update:

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

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 $\eta, \omega$ can be found in [27]. Motivated by the KKT conditions (1.2a), we define the primal and dual infeasibility as

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

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

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

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

Hence, 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($y^{(k)}, \rho_k$)). Hence, we can monitor the dual infeasibility error of (NLP) whilst solving (BCL($y^{(k)}, \rho_k$)).

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 $\eta_k \searrow 0$ and $\omega_k \searrow 0$, an initial point $(x^{(0)}, y^{(0)})$ and $\rho_0$, set $k \leftarrow 0$
while $(x^{(k)}, y^{(k)})$ not optimal do
    Set $j \leftarrow 0$ and initialize $\hat{x}^{(j)} \leftarrow x^{(k)}$
    Set up the augmented Lagrangian subproblem (BCL($y^{(k)}, \rho_k$))
    while $\omega_{\rho_k}(\hat{x}^{(j)}, y^{(k)}) > \omega_k$ and $\eta(\hat{x}^{(j)}) > \eta_k$ (not acceptable) 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 $\omega_{\rho_k}(\hat{x}^{(j)}, y^{(k)}) \le \omega_k$ but $\eta(\hat{x}^{(j)}) > \eta_k$ then
            Increase penalty parameter $\rho_k \leftarrow 2\rho_k$
        else
            Update multipliers: $\hat{y}^{(j+1)} \leftarrow y^{(k)} - \rho_k c(\hat{x}^{(j+1)})$
        Set $j \leftarrow j + 1$
    Set $(x^{(k+1)}, y^{(k+1)}) \leftarrow (\hat{x}^{(j)}, \hat{y}^{(j)})$, update $\rho_{k+1} \leftarrow \rho_k$, and increase $k \leftarrow k + 1$
```

**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 [42] for quadratic programs to general NLPs:

1. Replace the forcing sequences $(\eta_k, \omega_k)$ by a less restrictive algorithmic construct, namely a filter (defined in Section 2);

2. Introduce a second-order step to promote fast local convergence, similar to recent sequential linear quadratic programming (SLQP) methods [19, 24, 39];

3. Equip the augmented Lagrangian method with a fast and robust detection of infeasible subproblems [37].

In [13], 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.

This 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 Section 4 presents the global convergence proof. In Section 5, we present some promising numerical results. We close the paper with some conclusions and outlooks. R.Ed

## 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 $(\omega_k, \eta_k)$ to drive $\omega_0(x, y)$ and $\eta(x)$ to zero. Here, we instead use the filter mechanism [38, 36] to achieve R.1 convergence to first-order points. A filter is formally defined as follows.

**Definition 2.1** (Augmented Lagrangian Filter and Acceptance). *A filter $\mathcal{F}$ is a list of pairs $(\eta_l, \omega_l) := (\eta(x^{(l)}), \omega_0(x^{(l)}, y^{(l)}))$ 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*

$$\eta_k := \eta(x^{(k)}) \leq \beta \eta_l \quad \text{or} \quad \omega_k := \omega_0(x^{(k)}, y^{(k)}) \leq \omega_l - \gamma \eta(x^{(k)}), \quad \forall (\eta_l, \omega_l) \in \mathcal{F}. \tag{2.6}$$

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

In our algorithm, we maintain a filter, $\mathcal{F}_k$, at iteration $k$, 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 $\eta(x)$ R.2.4 for all points that are acceptable:

$$\eta(x) \leq U := \max\left(\omega_{\min}/\gamma, \beta \eta_{\min}\right), \tag{2.7}$$

where $\omega_{\min}$ is the smallest first-order error of any filter entry, that is $\omega_{\min} := \min\{\omega_l : (\eta_l, \omega_l) \in \mathcal{F}\}$, and the smallest infeasibility is $\eta_{\min} := \min\{\eta_l : (\eta_l, \omega_l) \in \mathcal{F} \text{ and } \eta_l > 0\}$, see Figure 1. The point $(\eta_{\min}, \omega_{\min})$ is the ideal filter entry.

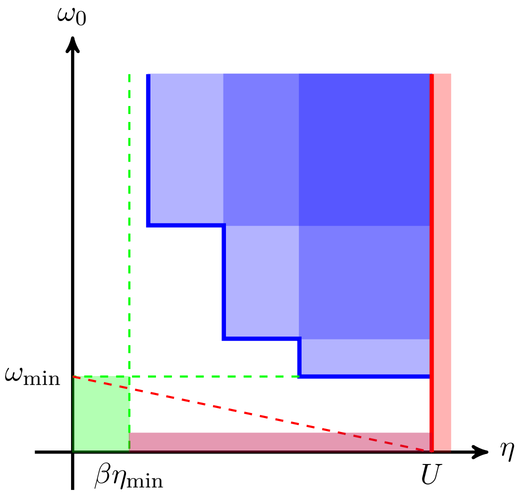

Figure 1: Example of an augmented Lagrangian filter $\mathcal{F}$ with three entries. The filter is in blue, the dashed green line shows the envelope in $\eta$, and the upper bound $U$ (red line) is implied by the sloping envelope condition (2.6) and $\omega_0 \geq 0$. Values above and to the right of the filter are not acceptable. The green area shows the set of filter entries that are guaranteed to be acceptable, and the shaded purple area is the set of entries that trigger the switch to restoration.

We note that 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:

$$\nabla_x \mathcal{L}_0(x^{(k)}, y^{(k)}) = \nabla_x \mathcal{L}_{\rho_k}(x^{(k)}, y^{(k-1)}). \tag{2.8}$$

Thus, by using the Lagrangian, we ensure that filter-acceptable points remain acceptable after the first-order multiplier update. Moreover, (2.8) shows that the filter acceptance can be readily checked during the minimization of the augmented Lagrangian, in which the multiplier is fixed and we iterate over $x$ only.

The filter envelope defined by $\beta$ and $\gamma$ ensures that iterates cannot accumulate at points where $\eta > 0$, and it promotes convergence (see Lemma 4.4). 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.

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 R.1 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.

We note, that most of the effort of Algorthm 2 is in the solution of the approximate minimization of the augmented Lagrangian, for which efficient methods exist, such as bound-constrained

```
Given $(x^{(0)}, y^{(0)})$ and $\rho_0$, 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

projected-gradient conjugate-gradient methods, see, e.g. [58, 22] R.2.6

## 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, [9]. Here, we extend the penalty estimate from [42] 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 [42]. We will show in Section 4 that the penalty update is bounded, so that our heuristic does not harm the algorithm.

Consider the Hessian of the augmented Lagrangian, $\mathcal{L}_\rho(x, y)$: R.1

$$\nabla^2 \mathcal{L}_\rho = \nabla^2 \mathcal{L}_0 + \rho \nabla c \nabla c^T + \rho \sum_{i=1}^{m} c_i \nabla^2 c_i, \tag{3.9}$$

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 R.2.7

$$\nabla^2 \mathcal{L}_\rho \approx \nabla^2 \mathcal{L}_0 + \rho \nabla c \nabla c^T. \tag{3.10}$$

Now, we use the same ideas as in [42] 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 set and the inactive set as

$$\mathcal{A}^k := \mathcal{A}(x^{(k)}) := \left\{ i : x_i^{(k)} = l_i \text{ or } x_i^{(k)} = u_i \right\} \text{ and } \mathcal{I}^k := \{1, 2, \ldots, n\} - \mathcal{A}^k, \tag{3.11}$$

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

$$H_k := \left[ \nabla_x^2 \mathcal{L}_{\rho_k}(x^{(k)}, y^{(k)}) \right]_{\mathcal{I}_k, \mathcal{I}_k} \quad \text{and} \quad A_k := \left[ \nabla c(x^{(k)})^T \right]_{:, \mathcal{I}_k}. \tag{3.12}$$

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

$$\rho \geq \rho_{\min}(\mathcal{A}^k) := \frac{\max\{0, -\lambda_{\min}(H_k)\}}{\sigma_{\min}(A_k)^2}, \tag{3.13}$$

where $\lambda_{\min}(\cdot)$ and $\sigma_{\min}(\cdot)$ denote the smallest eigenvalue and singular value, respectively. Computing (3.13) directly would be prohibitive for large-scale problems, and we use the following estimate instead: R.2.15

$$\tilde{\rho}_{\min}(\mathcal{A}^k) := \max \left\{ 1, \frac{\|H_k\|_1}{\max\left\{ \frac{1}{\sqrt{|\mathcal{I}_k|}} \|A_k\|_\infty , \frac{1}{\sqrt{m}} \|A_k\|_1 \right\}} \right\}, \tag{3.14}$$

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 (3.14).

An alternative adaptive penalty update is proposed in [29]. The authors propose an adaptive penalty parameter update 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 in particular when solving 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 either drive the R.1
penalty parameter to infinity or switch to the minimization of $\eta(x)$. We prefer the latter approach because it provides an easy escape if we determine that the NLP appears to be feasible after all. We note, that unlike linear programming (LP), there does not exist a phase I/phase II approach for NLPs, because even once we become feasible, 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. R.2.9

We define a set of implementable criteria that force the algorithm to switch to the minimization and R.1
of the constraint violation. Recall that the augmented Lagrangian filter implies the existence of an upper bound $U = \max\{\omega_{\min}/\gamma, \beta\eta_{\min}\}$ from (2.7). Thus any inner iteration that generates

$$\hat{\eta}_{j+1} = \eta(\hat{x}^{(j+1)}) \geq \beta U \tag{3.15}$$

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 Figure 1. This observation motivates the following condition:

$$\omega_{\rho_k}(\hat{x}^{(j+1)}, y^{(k)}) \le \epsilon \text{ and } \eta(\hat{x}^{(j+1)}) \ge \beta\eta_{\min}, \tag{3.16}$$

where $\epsilon > 0$ is a constant and $\eta_{\min}$ is the smallest constraint violation of any filter entry, namely $\eta_{\min} := \min\{\eta_l : (\eta_l, \omega_l) \in \mathcal{F}\} > 0$, which is positive because we only ever add entries with positive constraint violation to the filter. In our algorithm, we switch to restoration if (3.15) or (3.16) holds. R.2.10

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:

$$\left\| \min\left(\hat{x}^{(j)} - l, \max\left(\hat{x}^{(j)} - u, 2\nabla c(\hat{x}^{(j)})^T c(\hat{x}^{(j)})\right)\right)\right\| \le \epsilon \text{ and } \eta(\hat{x}^{(j)}) > \epsilon, \tag{3.17}$$

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 3.1.** *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 R.2.11
Algorithm 2), and hence that we must find a filter-acceptable point in a finite number of iterations R.2.12
in the latter case. □

### 3.3 Termination of Inner Minimization

The filter introduced in Section 2 ensures convergence only to *feasible* limit points; see Lemma 4.4. 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 $\omega_k$ to progress toward feasibility $\eta_k$.

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

$$\Delta\mathcal{L}_{\rho_k}^{(j)} := \mathcal{L}_{\rho_k}(\hat{x}^{(j)}, y^{(k)}) - \mathcal{L}_{\rho_k}(\hat{x}^{(j+1)}, y^{(k)}) \ge \sigma\hat{\omega}_j, \tag{3.18}$$

where $\sigma > 0$ is a constant. This condition can be satisfied, for example, by requiring Cauchy decrease on the augmented Lagrangian for fixed $\rho_k$ and $y^{(k)}$. We note that the right-hand side of (3.18) 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 (4.21). 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 $x_i^{(k)} = 0$, and then solve an equality-constrained QP (EQP) on those active constraints, similarly to popular SLQP approaches. Given sets of active and inactive constraints (3.11), our goal is to solve an EQP with $x_i^{(k)} = l_i$, or $x^{(k)} = u_i$, $\forall i \in \mathcal{A}^k$. Provided that the EQP is convex, its solution can be obtained R.2.14
by solving an augmented linear system.

Using the notation introduced in (3.11) and (3.12), the convex EQP is equivalent to the following augmented system,

$$\begin{bmatrix} H_k & A_k^T \\ A_k & \end{bmatrix} \begin{pmatrix} \Delta x_{\mathcal{I}} \\ \Delta y \end{pmatrix} = \begin{pmatrix} -\nabla f_{\mathcal{I}}^{(k+1)} \\ -c(x^{(k+1)}) \end{pmatrix}, \tag{3.19}$$

and $\Delta x_{\mathcal{A}} = 0$. We note that, 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 R.2.16
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 for fixed penalty parameter, $\rho_k$, and multipliers, $y^{(k)}$ until a filter- R.1
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 we entered the restoration phase or not. If we enter the restoration phase, we increase the penalty parameter 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 (3.14), 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:*

```
Given $(x^{(0)}, y^{(0)})$ and $\rho_0$, 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 $j \leftarrow 0$, RestFlag $\leftarrow$ false, and initialize $\hat{x}^{(0)} \leftarrow x^{(k)}$
    while $(\hat{\eta}_j, \hat{\omega}_j)$ not acceptable to $\mathcal{F}_k$ do
        Approximately minimize the augmented Lagrangian for $\hat{x}^{(j+1)}$ starting at $\hat{x}^{(j)}$:

            $$\underset{l \le x \le u}{\text{minimize}} \ \mathcal{L}_{\rho_k}(x, y^{(k)}) = f(x) - {y^{(k)}}^T c(x) + \frac{1}{2}\rho_k \|c(x)\|^2$$

         such that the sufficient reduction condition (3.18) holds.
        if restoration switching condition (3.15) or (3.16) holds then
            Set RestFlag = true
            Increase penalty parameter: $\rho_{k+1} \leftarrow 2\rho_k$
            Switch to restoration phase to find $(\hat{x}^{(j+1)}, \hat{y}^{(j+1)})$ acceptable to $\mathcal{F}$ or find an
              infeasible point that minimizes $\|c(x)\|^2$ subject to $l \le x \le u$
        else
            Provisionally update multipliers: $\hat{y}^{(j+1)} \leftarrow y^{(k)} - \rho_k c(\hat{x}^{(j+1)})$
            Compute $\hat{\omega}_{j+1} \leftarrow \omega_0(\hat{x}^{(j+1)}, \hat{y}^{(j+1)})$ and $\hat{\eta}_{j+1} \leftarrow \eta(\hat{x}^{(j+1)})$.
        Set $j \leftarrow j + 1$
    Set $(\hat{x}^{(k+1)}, \hat{y}^{(k+1)}) \leftarrow (\hat{x}^{(j)}, \hat{y}^{(j)})$
    Solve EQP (3.19) for $(\Delta x^{(k+1)}, \Delta y^{(k+1)})$
    Line-search: Find $\alpha_k \in \{0\} \cup [\alpha_{\min}, 1]$ such that

        $$(x^{(k+1)}, y^{(k+1)}) = (\hat{x}^{(k+1)}, \hat{y}^{(k+1)}) + \alpha_k(\Delta x^{(k+1)}, \Delta y^{(k+1)}) \text{ acceptable to } \mathcal{F}_k$$

    Compute $\omega_{k+1} \leftarrow \omega_0(x^{(k+1)}, y^{(k+1)})$, $\eta_{k+1} \leftarrow \eta(x^{(k+1)})$
    if $\eta_{k+1} > 0$ then
        Add $(\eta_{k+1}, \omega_{k+1})$ to filter: $\mathcal{F}_{k+1} \leftarrow \mathcal{F}_k \cup \{(\eta_{k+1}, \omega_{k+1})\}$ (ensuring $\eta_l > 0 \ \forall l \in \mathcal{F}_{k+1}$)
    if not RestFlag and $\rho_k < \tilde{\rho}_{\min}(\mathcal{A}^k)$ see (3.14) then
        Increase penalty: $\rho_{k+1} \leftarrow 2\tilde{\rho}_{\min}(\mathcal{A}^k)$
    else
        Leave penalty parameter unchanged: $\rho_{k+1} \leftarrow \rho_k$
    Set $k \leftarrow k + 1$
```

**Algorithm 3:** Augmented Lagrangian Filter Method.

**A1** *The problem functions $f, c$ are twice continuously differentiable.*

**A2** *The constraint norm satisfies $\|c(x)\| \to \infty$ as $\|x\| \to \infty$.*

Assumption **A1** is standard. Assumption **A2** implies that our iterates remain in a compact set (see Lemma 4.1). This assumption could be replaced by an assumption that we optimize over finite bounds $l \le x \le u$. 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 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 the linear independence constraint qualification 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 (3.14) is bounded.

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

**Lemma 4.1.** *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 $\|c(x^{(k)})\| \leq U$ for all $k$. The switching condition (3.15) implies that $\|c(\hat{x}^{(j)})\| \leq U$ for all $j$. The feasibility restoration minimizes $\eta(x)$, implying that all its iterates in turn satisfy $\|c(x^{(k)})\| \leq U$. 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 Figure 1 (right).

**Lemma 4.2.** *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 Figure 1 (right). 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 4.3.** *The inner iteration is finite.*

**Proof.** If the inner iteration finitely terminates with a filter-acceptable point or switches to the restoration phase, then 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 4.1 implies that this sequence has a limit point $x^* = \lim \hat{x}^{(j)}$. We note that the penalty parameter and the multipliers are fixed during the inner iteration, and we consider the sequence $\mathcal{L}_\rho(\hat{x}^{(j)}, y)$ for fixed $\rho = \rho_k$ and $y = y^{(k)}$. The sufficient reduction condition (3.18) 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 4.1. Thus, it follows that $\hat{\omega}_j \to 0$. If in addition, $\hat{\eta}_j \to \hat{\eta} < \beta\eta_{\min}$, then we must find a filter-acceptable point in the green region of Figure 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 4.2. □

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

**Lemma 4.4.** *Assume that there exist an infinite number of outer iterations. It follows that* $\eta(x^{(k)}) \to 0$. R.1

**Proof.** Every outer iteration for which $\eta_k > 0$ adds an entry to the filter. The proof follows directly from [24, Lemma 1]. □

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

**Lemma 4.5.** *Assume that the penalty parameter is bounded,* $\rho_k \leq \bar{\rho} < \infty$, *and consider an infinite sequence of outer iterations. Then it follows that* $\omega(x^{(k)}) \to 0$.

**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 (3.18) implies that

$$\Delta\mathcal{L}^{\text{in}}_{\bar{\rho},k} := \mathcal{L}_{\bar{\rho}}(x^{(k)}, y^{(k)}) - \mathcal{L}_{\bar{\rho}}(x^{(k+1)}, y^{(k)}) \geq \sigma\omega_k \geq \sigma\bar{\omega} > 0. \tag{4.20}$$

We now show that this "inner" sufficient reduction (for fixed $y^{(k)}$) implies an "outer" sufficient reduction. We combine (4.20) with the first-order multiplier update (1.4) and obtain

$$\Delta\mathcal{L}^{\text{out}}_{\bar{\rho},k} := \mathcal{L}_{\bar{\rho}}(x^{(k)}, y^{(k)}) - \mathcal{L}_{\bar{\rho}}(x^{(k+1)}, y^{(k+1)}) \;=\; \Delta\mathcal{L}^{\text{in}}_{\bar{\rho},k} - \bar{\rho}\|c(x^{(k+1)})\|_2^2 \;\geq\; \sigma\bar{\omega} - \bar{\rho}\eta_{k+1}^2. \tag{4.21}$$

Lemma 4.4 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:

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

for all $k$ sufficiently large. Thus, if $\omega_k \geq \bar{\omega} > 0$ is bounded away from zero, then 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 $c(x^{(k)})^T y^{(k)} \leq M$ R.1 and
for all major iterates. The first-order multiplier update implies that $y^{(k)} = y^{(0)} - \bar{\rho}\sum c^{(l)}$ and hence R.2.19
that

$$c(x^{(k)})^T y^{(k)} = \left( y^{(0)} - \bar{\rho}\sum_{l=1}^{k} c^{(l)} \right)^T c^{(k)} \leq \left( \|y^{(0)}\| + \bar{\rho}\sum_{l=1}^{k} \|c^{(l)}\| \right) \|c^{(k)}\| = \left( y_0 + \bar{\rho}\sum_{l=1}^{k} \eta_l \right) \eta_k \, , \tag{4.22}$$

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 4.4. The definition of the filter then implies that $E_{k+1} \leq \beta E_k$, and we obtain from (4.22) 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 the linear independence constraint qualification (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 4.6.** *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 4.4 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 [41], 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, it follows that the quantity

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

remains bounded as $\nu \to \infty$. Similarly to the proof of Lemma 4.5, 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 [41], which shows that the limit point is a KKT point. □

The preceding lemmas are summarized in the following result.

**Theorem 4.1.** *Assume that Assumptions* **A1** *and* **A2** *hold. Then 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, then it is a KKT point. Otherwise, it is an FJ point;*
3. *The restoration phase converges to a minimum of the constraint violation.*

**Remark 4.1.** *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 that those that do not have strongly stationary points would require the penalty to be unbounded.*

**Remark 4.2.** *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 $\eta_k = 0$, then the sufficient reduction condition (3.18) implies that we must converge to a stationary point, similarly to the arguments in Lemma 4.5. If we have a sequence that alternates between iterates for which $\eta_k = 0$ and iterates for which $\eta_k > 0$, then we can never revisit any iterates for which $\eta_k > 0$ because those iterates have been added to the filter. By Lemma 4.4, any limit point is feasible. Thus, if LICQ holds, then the limit is a KKT point; otherwise, it may be an FJ point. We note that these conclusions are consistent with Theorem 4.1.*

## 5 Numerical Results

We have implemented a preliminary version of Algorithm 3 in `C++`, using L-BFGS-B 3.0, see [73], to minimize the bound-constrained augmented Lagrangian and MA57, see [32], to solve 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. R.Ed

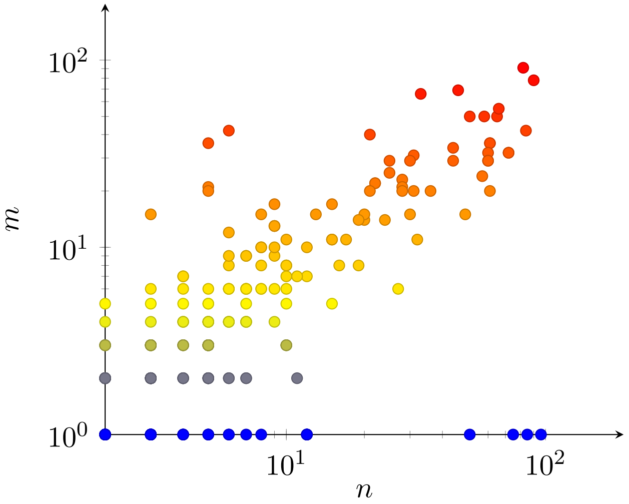

Figure 2: Distribution of CUTEst test problems based on number of variables, $n$, and number of constraints, $m$.

We have chosen 429 small test problems from the CUTEst test set [47] that have up to 100 variables and/or constraints. The distribution of the problem sizes is shown in Figure 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:

- FilterSQP [36] is a filter SQP solver endowed with a trust-region mechanism to enforce convergence;
- SNOPT [45] is an SQP method using limited-memory quasi-Newton approximations to the Hessian of the Lagrangian with an augmented Lagrangian merit function;
- MINOS [59] implements a linearly-constrained augmented Lagrangian method with a line-search mechanism;
- IPOPT [71] implements a filter interior-point method with a line-search mechanism; and,

- LANCELOT [26] 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 gradient evaluation to solve a problem. This statistic is a good surrogate for the number of major iterations. Detailed results are shown in Table 1 in the appendix. We note, that MINOS and SNOPT do not report gradient evaluations for some problems such as QPs, which results in empty entries in the table. In these cases, we set the number of gradient evaluations to "1", which favors these solvers slightly, but does not change our overall conclusions.

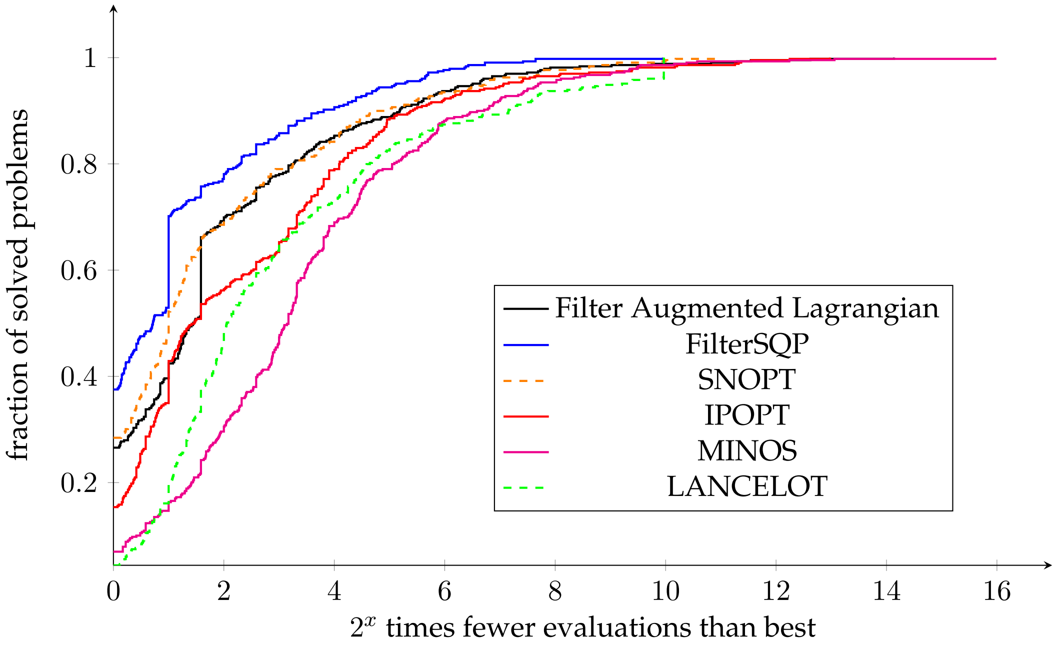

Figure 3: Performance profile comparing number of gradient evaluations of different NLP solvers for 429 small CUTEst problems.

We summarize our numerical results using a performance profile, [30], in Figure 3. 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 whilst 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, clearly outperforms the two augmented Lagrangian methods, MINOS and LANCELOT, indicating that the use of a filter provides a faster convergence mechanism, reducing the number of iterations.

## 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. First, 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 [69], or if we employ a local non-monotone filter similar to [66]. A second open question is how the proposed method performs in practice. We have some promising experience with large-scale quadratic programs [42], and we are working on an implementation for nonlinear programs.

We have presented preliminary numerical results on 429 small CUTEst test problems that showed that our new augmented Lagrangian filter method outperforms other augmented Lagrangian solvers, and is competitive with SQP methods in terms of major iterations.

## Acknowledgments

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.

## References

[1] Kumar Abhishek, Sven Leyffer, and Jeffrey T. Linderoth. FilMINT: An outer-approximation-based solver for nonlinear mixed integer programs. *INFORMS Journal on Computing*, 22:555–567, 2010.

[2] A. Altay-Salih, M.C. Pinar, and S. Leyffer. Constrained nonlinear programming for volatility estimation with GARCH models. *SIAM Review*, 45(3):485–503, 2003.

[3] J.J. Arrieta-Camacho, L.T. Biegler, and D. Subramanian. NMPC-based real-time coordination of multiple aircraft. In R.F. Allgower and L.T. Biegler, editors, *Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 629–639. Springer, Heidelberg, 2007.

[4] Guillermo Bautista, Miguel F. Anjos, and Anthony Vannelli. Formulation of oligopolistic competition in AC power networks: An NLP approach. *IEEE Transaction on Power Systems*, 22(1):105–115, 2007.

[5] Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, James Luedtke, and Ashutosh Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 5 2013.

[6] Hande Benson and David Shanno. An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. *Computational Optimization and Applications*, 38:371–399, 2007. 10.1007/s10589-007-9048-6.

[7] Hande Y. Benson and David F. Shanno. Interior-point methods for nonconvex nonlinear programming: regularization and warmstarts. *Computational Optimization and Applications*, 40(2):143–189, 2008.

[8] H.Y. Benson, D.F. Shanno, and R.J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: filter-methods and merit functions. *Computational Optimization and Applications*, 23(2):257–272, 2002.

[9] D.P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, New York, 1982.

[10] J.T. Betts. *Practical Methods for Optimal Control Using Nonlinear Programming*. Advances in Design and Control. SIAM, Philadelphia, 2001.

[11] L.T. Biegler and I.E. Grossmann. Challenges and research issues for product and process design optimization. In C.A. Floudas and R. Agrawal, editors, *Proceedings of Foundations of Computer Aided Process Design*. CACHE Corporation, Austin, TX, 2004.

[12] L.T. Biegler and I.E. Grossmann. Part I: Retrospective on optimization. *Computers and Chemical Engineering*, 28(8):1169–1192, 2004.

[13] E.G. Birgin and J.M. Martinez. Improving ultimate convergence of an augmented Lagrangian method. *Optimization Methods and Software*, 23(2):177–195, 2008.

[14] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuejols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wähter. An algorithmic framework for convex mixed integer nonlinear programs. Research Report RC 23771, IBM T. J. Watson Research Center, Yorktown, NY, 2005.

[15] Pierre Bonami, Jon Lee, Sven Leyffer, and Andreas Waechter. 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, September 2011.

[16] J.F. Bonnans and J. André. Optimal structure of gas transmission trunklines. Technical Report 6791, INRIA Saclay, 91893 ORSAY Cedex, France, 2009.

[17] B. Bonnard, L. Faubourg, G. Launay, and E. Trélat. Optimal control with state constraints and the space shuttle re-entry problem. *Journal of Dynamical and Control Systems*, 9(2):155–199, 2003.

[18] A. Borghetti, A. Frangioni, F. Lacalandra, and C.A. Nucci. Lagrangian heuristics based on disaggregated bundle methods for hydrothermal unit commitment. *IEEE Transactions on Power Systems*, 18:1–10, 2003.

[19] R.H. Byrd, N.I.M. Gould, J. Nocedal, and R.A. Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming, Series B*, 100(1):27–48, 2004.

[20] R.H. Byrd, M.E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM J. Optimization*, 9(4):877–900, 1999.

[21] R.H. Byrd, J. Nocedal, and R.A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer-Verlag, New York, 2006.

[22] Calamai, P.H. and Moré, J.J. Projected gradient methods for linearly constraint problems. *Mathematical Programming*, 39:93–116, 1987.

[23] J. Castro and J.A. Gonzalez. A nonlinear optimization package for long-term hydrothermal coordination. *European Journal of Operational Research*, 154(1):641–658, 2004.

[24] C.M. Chin and R. Fletcher. On the global convergence of an SLP-filter algorithm that takes EQP steps. *Mathematical Programming*, 96(1):161–177, 2003.

[25] T.F. Coleman, Y. Li, and A. Verman. Reconstructing the unknown volatility function. *Journal of Computational Finance*, 2(3):77–102, 1999.

[26] A.R. Conn, N.I.M. Gould, and Ph.L. Toint. *LANCELOT: A Fortran package for large-scale nonlinear optimization (Release A)*. Springer Verlag, Heidelberg, New York, 1992.

[27] A.R. Conn, N.I.M. Gould, and Ph.L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, 2000.

[28] G. Cornuejols and R. Tütüncü. *Optimization Methods in Finance*. Cambridge University Press, 2007.

[29] Frank E Curtis, Hao Jiang, and Daniel P Robinson. An adaptive augmented Lagrangian method for large-scale constrained optimization. *Mathematical Programming*, 152(1-2):201–245, 2015.

[30] Elizabeth D. Dolan and Jorge Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[31] V. Donde, V. Lopez, B. Lesieutre, A. Pinar, C. Yang, and J. Meza. Identification of severe multiple contingencies in electric power networks. In *Proceedings 37th North American Power Symposium*, 2005. LBNL-57994.

[32] Iain S Duff. Ma57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Transactions on Mathematical Software (TOMS)*, 30(2):118–144, 2004.

[33] Klaus Ehrhardt and Marc C. Steinbach. Nonlinear optimization in gas networks. In H.G. Bock, E. Kostina, H.X. Phu, and R. Ranacher, editors, *Modeling, Simulation and Optimization of Complex Processes*, pages 139–148. Springer, Berlin, 2005.

[34] R. Fletcher and S. Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66:327–349, 1994.

[35] R. Fletcher and S. Leyffer. User manual for filterSQP. Numerical Analysis Report NA/181, University of Dundee, April 1998.

[36] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91:239–270, 2002.

[37] R. Fletcher and S. Leyffer. Filter-type algorithms for solving systems of algebraic equations and inequalities. In G. di Pillo and A. Murli, editors, *High Performance Algorithms and Software for Nonlinear Optimization*, pages 259–278. Kluwer, Dordrecht, 2003.

[38] R. Fletcher, S. Leyffer, and Ph.L. Toint. On the global convergence of a filter-SQP algorithm. *SIAM J. Optimization*, 13(1):44–59, 2002.

[39] Fletcher, R. and Sainz de la Maza, E. Nonlinear programming and nonsmooth optimization by successive linear programming. *Mathematical Programming*, 43:235–256, 1989.

[40] A. Forsgren, P.E. Gill, and M.H. Wright. Interior methods for nonlinear optimization. *SIAM Review*, 4(4):525–597, 2002.

[41] Michael P. Friedlander. *A Globally Convergent Linearly Constrained Lagrangian Method for Nonlinear Optimization*. Ph.D. thesis, Stanford University, August 2002.

[42] M.P. Friedlander and S. Leyffer. Global and finite termination of a two-phase augmented Lagrangian filter method for general quadratic programs. *SIAM J. Scientific Computing*, 30(4):1706–1729, 2008.

[43] L.E. Ghaoui, M. Oks, and F. Oustry. Worst-case value-at-risk and robust portfolio optimization: A conic programming approach. *Operations Research*, 51(4), 2003.

[44] P.E. Gill, W. Murray, and M.A. Saunders. User's guide for SQOPT 5.3: A Fortran package for large-scale linear and quadratic programming. Tech. Rep. NA 97-4, Department of Mathematics, University of California, San Diego, 1997.

[45] P.E. Gill, W. Murray, and M.A. Saunders. SNOPT: An SQP algorithm for large–scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[46] Jacek Gondzio and Andreas Grothey. Reoptimization with the primal-dual interior point method. *SIAM Journal on Optimization*, 13(3):842–864, 2002.

[47] Nicholas IM Gould, Dominique Orban, and Philippe L Toint. Cutest: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Computational Optimization and Applications*, 60(3):545–557, 2015.

[48] J.-P. Goux and S. Leyffer. Solving large MINLPs on computational grids. *Optimization and Engineering*, 3:327–346, 2002.

[49] Q. Huangfu and J.A.J. Hall. Novel update techniques for the revised simplex method. Technical Report ERGO-13-001, The School of Mathematics, University of Edinburgh, Edinburgh, UK, 2013.

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

[51] H. Konno and H. Yamazaki. Mean-absolute deviation portfolio optimization model and its application to Tokyo stock market. *Management Science*, 37:519–531, 1991.

[52] S. Leyffer. Integrating SQP and branch-and-bound for mixed integer nonlinear programming. *Computational Optimization & Applications*, 18:295–309, 2001.

[53] S. Leyffer, G. Lopez-Calva, and J. Nocedal. Interior methods for mathematical programs with complementarity constraints. *SIAM Journal on Optimization*, 17(1):52–77, 2006.

[54] S. Leyffer and T.S. Munson. A globally convergent filter method for MPECs. Preprint ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division, September 2007.

[55] Miles Lubin, J.A.J. Hall, Cosmin G. Petra, and Mihai Anitescu. Parallel distributed-memory simplex for large-scale stochastic LP problems. *Computational Optimization and Applications*, 55(3):571–596, 2013.

[56] A. Martin, M. Möller, and S. Moritz. Mixed integer models for the stationary case of gas network optimization. *Mathematical Programming*, 105:563–582, 2006.

[57] J.A. Momoh, R.J. Koessler, M.S. Bond, B. Stott, D. Sun, A. Papalexopoulos, and P. Ristanovic. Challenges to optimal power flow. *IEEE Transaction on Power Systems*, 12:444–455, 1997.

[58] J.J. Moré and G. Toraldo. On the solution of quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

[59] B.A. Murtagh and M.A. Saunders. MINOS 5.4 user's guide. Report SOL 83-20R, Department of Operations Research, Stanford University, 1993.

[60] Bruce A. Murtagh and Michael A. Saunders. A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. In A.G. Buckley and J.-L. Goffin, editors, *Algorithms for Constrained Minimization of Smooth Nonlinear Functions*, volume 16 of *Mathematical Programming Studies*, pages 84–117. Springer Berlin Heidelberg, 1982.

[61] P. Penfield, R. Spence, and S. Duinker. *Tellegen's theorem and electrical networks*. MIT Press, Cambridge, London, 1970.

[62] M.J.D. Powell. Algorithms for nonlinear constraints that use Lagrangian functions. *Mathematical Programming*, 14(1):224–248, 1978.

[63] Gad Rabinowitz, Abraham Mehrez, and Gideon Oron. A nonlinear optimization model of water allocation for hydroelectric energy production and irrigation. *Management Science*, 34(8):973–990, 1988.

[64] A. Raghunathan and L. T. Biegler. An interior point method for mathematical programs with complementarity constraints (MPCCs). *SIAM Journal on Optimization*, 15(3):720–750, 2005.

[65] G.B. Sheble and G.N. Fahd. Unit commitment literature synopsis. *IEEE Transactions on Power Systems*, 9:128–135, 1994.

[66] Chungen Shen, Sven Leyffer, and Roger Fletcher. A nonmonotone filter method for nonlinear optimization. *Computational Optimization and Applications*, 52(3):583–607, 2012.

[67] E. Smith and J.A.J. Hall. 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, 2012.

[68] R.J. Vanderbei and D.F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *COAP*, 13:231–252, 1999.

[69] A. Wächter and L.T. Biegler. Line search filter methods for nonlinear programming: Local convergence. *SIAM Journal on Optimization*, 16(1):32–48, 2005.

[70] A. Wächter and L.T. Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005.

[71] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[72] R.S. Womersley and K. Lau. Portfolio optimisation problems. In R.L. May and A.K. Easton, editors, *Computational Techniques and Applications: CTAC95*, pages 795–802. World Scientific Publishing Co., 1996.

[73] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.

# A Appendix

Table 1: Number of evaluations of nonlinear solvers on a subset of CUTEr problems.

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| 3pk | 3 | 7 |  | 12 | 210 | 47 |
| 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 |
| arglinb | 3 | 2 |  | 3 | 9 | 2 |
| arglinc | 3 | 2 |  | 3 | 9 | 2 |
| avgasa | 9 | 2 |  | 10 | 20 | 12 |
| avgasb | 21 | 2 |  | 13 | 16 | 11 |
| avion2 |  | 19 | 19 | 143 | 18 | 787 |
| bard | 42 | 11 | 23 | 9 | 37 | 15 |
| batch |  | 9 | 33 | 34 | 380 | 1000 |
| beale |  | 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 |
| biggsc4 | 16 | 2 |  | 35 | 17 | 15 |
| booth | 3 | 2 |  | 2 |  | 3 |
| box2 | 3 | 9 | 10 | 9 | 11 | 20 |
| box3 | 12 | 8 | 24 | 15 | 16 | 31 |
| bqp1var | 2 | 2 |  | 6 | 6 | 3 |
| bqpgabim | 3 | 2 |  | 21 | 108 | 6 |
| bqpgasim | 3 | 2 |  | 21 | 122 | 6 |
| brkmcc | 6 | 4 | 10 | 4 | 14 | 7 |
| brownal | 15 | 8 | 21 | 8 | 78 | 24 |
| brownbs | 57 | 48 | 32 | 8 | 36 | 7 |
| browniden | 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 |
| bt3 | 3 | 2 |  | 2 | 13 | 6 |
| bt4 | 23 | 11 | 10 | 10 | 48 | 25 |
| bt5 | 27 | 9 | 11 | 8 | 173 | 20 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| 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 |
| 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 |
| deconvb | 8 | 30 | 103 | 5705 | 31 | 31 |
| deconvc | 85 | 58 | 81 | 99 | 88 | 43 |
| deconvu | 45 | 971 | 152 | 687 | 31 | 69 |
| degenlpa |  | 2 |  | 29 |  | 25 |
| degenlpb |  | 2 |  | 41 |  | 45 |
| 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 |
| dixon3dq | 3 | 2 |  | 2 | 44 | 4 |
| djtl | 84 | 29 | 1345 | 861 | 89 | 100 |
| dnieper | 9 | 4 | 13 | 31 | 37 | 75 |
| dual1 | 3 | 2 | 221 | 17 | 278 | 9 |
| dual2 | 3 | 2 | 116 | 14 | 216 | 9 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| dual4 | 3 | 2 | 31 | 14 | 209 | 8 |
| dualc1 | | 2 | | 30 | 25 | 17 |
| dualc2 | | 2 | | 28 | 15 | 19 |
| dualc5 | 136 | 2 | | 11 | 25 | 8 |
| dualc8 | | 2 | | 13 | 17 | 24 |
| 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 |
| errinros | 22 | 53 | 267 | 70 | 553 | 76 |
| expfita | 36 | 13 | 24 | 31 | 29 | 54 |
| expfit | | 13 | 18 | 9 | 29 | 11 |
| extrasim | 3 | 2 | | 6 | | 3 |
| extrosnb | 1 | 2 | 2 | 1 | 6 | 1 |
| fccu | 3 | 4 | | 2 | 49 | 13 |
| fletcher | 46 | 1 | 2 | 28 | 9 | 28 |
| genhs28 | 3 | 2 | | 2 | 13 | 5 |
| genhumps | 63 | 188 | 77 | 321 | 170 | 134 |
| gigomez1 | 143 | 8 | 17 | 19 | 141 | 33 |
| goffin | 11 | 3 | | 10 | | 10 |
| 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 |
| hatfldc | 9 | 5 | 14 | 6 | 20 | 9 |
| 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 |
| hatfldh | 3 | 2 | | 19 | 9 | 13 |
| 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 |
| hilberta | 3 | 2 | | 2 | 13 | 60 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| hilbertb | 3 | 2 |  | 2 | 113 | 5 |
| himmelba | 3 | 2 |  | 2 |  | 3 |
| himmelbb | 12 | 25 | 8 | 12 | 21 | 11 |
| himmelbc | 10 | 8 | 7 | 9 | 9 | 11 |
| himmelbd |  | 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 |
| 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 |  | 9 | 15 | 17 | 13 | 7 |
| hs003 | 3 | 2 |  | 5 | 13 | 16 |
| 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 |
| hs021 | 3 | 2 |  | 9 | 9 | 3 |
| 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 |
| hs028 | 3 | 2 |  | 2 | 13 | 4 |
| hs029 | 77 | 8 | 14 | 9 | 174 | 18 |
| hs030 | 6 | 2 | 5 | 26 | 30 | 8 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| 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 |
| hs035 | 3 | 2 |  | 8 | 13 | 8 |
| hs036 | 6 | 3 | 10 | 13 | 9 | 7 |
| hs037 | 9 | 6 | 10 | 13 | 17 | 13 |
| hs038 | 7 | 54 | 101 | 78 | 89 | 56 |
| 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 |
| hs044 | 3 | 2 |  | 20 | 11 | 11 |
| hs045 | 1 | 2 | 2 | 48 | 6 | 1 |
| hs046 | 53 | 19 | 32 | 20 | 122 | 28 |
| hs047 | 335 | 21 | 28 | 21 | 126 | 29 |
| hs048 | 3 | 2 |  | 2 | 17 | 3 |
| hs049 | 30 | 17 | 34 | 20 | 62 | 38 |
| hs050 | 15 | 9 | 20 | 10 | 25 | 11 |
| hs051 | 3 | 2 |  | 2 | 13 | 10 |
| hs052 | 3 | 2 |  | 2 | 12 | 6 |
| hs053 | 3 | 2 |  | 7 | 13 | 6 |
| hs054 | 3 | 2 |  | 8 | 17 | 9 |
| 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 |
| hs076 | 3 | 2 |  | 8 | 14 | 9 |
| 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 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| hs081 | 91 | 38 | 11 | 8 | 56 | 17 |
| hs083 | 42 | 5 | 8 | 15 | 12 | 16 |
| hs084 |  | 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 |
| 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 |  | 3 | 2 | 18 | 4 | 24 |
| hs096 |  | 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 |  | 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 |  | 1 | 9 | 73 | 5 | 664 |
| hs116 |  | 14 | 22 | 26 | 97 | 1003 |
| hs117 | 224 | 6 | 20 | 23 | 158 | 66 |
| hs118 | 3 | 3 |  | 12 | 42 | 19 |
| hs119 | 19 | 7 | 22 | 15 | 30 | 28 |
| hs21mod | 3 | 2 |  | 17 | 9 | 3 |
| hs268 | 6 | 2 |  | 17 | 37 | 27 |
| hs35mod | 3 | 2 |  | 16 | 7 | 3 |
| hs3mod | 3 | 2 |  | 6 | 14 | 4 |
| hs44new | 3 | 2 |  | 14 | 11 | 10 |
| hs99exp |  | 12 | 42 | 30 | 213 | 1001 |
| hubfit | 3 | 2 | 8 | 9 | 12 | 8 |
| humps | 207 | 1001 | 257 | 571 | 194 | 1001 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| hypcir | 9 | 8 | 5 | 8 | 8 | 10 |
| jensmp |  | 11 | 36 | 10 | 56 | 10 |
| kiwcresc | 45 | 11 | 17 | 11 | 81 | 23 |
| kowosb | 18 | 18 | 33 | 23 | 40 | 24 |
| lakes |  | 63 | 39 | 20 | 1045 | 1001 |
| launch |  | 1 | 246 | 673 | 50 | 1001 |
| lewispol | 6307 | 1 | 6 |  | 38 | 22 |
| linspanh | 56 | 2 |  | 54 |  | 13 |
| 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 |
| lotschd | 3 | 3 |  | 15 | 6 | 9 |
| lsnnodoc | 9 | 7 | 8 | 15 | 8 | 11 |
| lsqfit | 3 | 2 |  | 8 | 12 | 7 |
| 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 |
| makela4 | 9 | 3 |  | 8 |  | 20 |
| maratosb | 8 | 13 | 7 | 33 | 11 | 8 |
| maratos | 12 | 10 | 9 | 5 | 21 | 9 |
| matrix2 | 26 | 12 | 14 | 21 | 66 | 13 |
| maxlika |  | 9 | 89 | 31 | 115 | 1001 |
| mconcon | 100 | 5 | 8 | 10 | 1 | 676 |
| mdhole | 12 | 56 | 70 | 106 | 117 | 68 |
| methanb8 |  | 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 |
| model |  | 2 |  | 15 |  | 33 |
| mwright | 12 | 12 | 12 | 11 | 38 | 19 |
| nasty | 3 | 2 |  | 2 | 7 | 5 |
| nonmsqrt | 73 | 716 | 7413 | 83890 | 1491 | 167 |
| nuffield_continuum | 6 | 5 | 9 | 7 | 16 | 15 |
| obstclal | 6 | 2 |  | 15 | 91 | 7 |
| obstclbl | 6 | 2 |  | 13 | 88 | 3 |
| obstclbu | 6 | 2 |  | 13 | 75 | 2 |
| odfits | 22 | 7 | 17 | 11 | 29 | 49 |
| optcntrl |  | 4 | 5 | 134 | 16 | 352 |
| optmass |  | 18 | 2 | 23 | 332 | 1001 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| optprloc | | 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 | | 20 | 82 | 25 | 128 | 45 |
| oslbqp | 3 | 2 | | 15 | 12 | 3 |
| palmer1a | | 51 | 205 | 71 | 12 | 102 |
| palmer1b | | 21 | 87 | 26 | 12 | 55 |
| palmer1c | 3 | 7 | | 2 | 62 | 145 |
| palmer1d | 3 | 6 | | 2 | 51 | 34 |
| 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 |
| palmer2c | 3 | 5 | | 2 | 63 | 298 |
| 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 |
| palmer3c | 3 | 5 | | 2 | 62 | 206 |
| 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 |
| palmer4c | 3 | 6 | | 2 | 62 | 176 |
| 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 |
| palmer5c | 3 | 4 | | 2 | 27 | 2 |
| palmer5d | 3 | 5 | | 2 | 26 | 2 |
| palmer5e | 6 | 3 | 1313 | 7933 | 25502 | 8 |
| palmer6a | 5 | 137 | 202 | 263 | 271 | 277 |
| palmer6c | 3 | 6 | | 2 | 61 | 159 |
| palmer6e | 5 | 38 | 198 | 59 | 260 | 47 |
| palmer7a | | 991 | 1299 | 8073 | 21 | 1001 |
| palmer7c | 3 | 8 | | 2 | 65 | 189 |
| palmer7e | 5 | 1001 | 1315 | 12230 | 925 | 39 |
| palmer8a | 9 | 50 | 127 | 102 | 104 | 61 |
| palmer8c | 3 | 7 | | 2 | 62 | 152 |
| 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 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| 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 |
| portfl1 | 3 | 2 |  | 10 | 49 | 20 |
| portfl2 | 3 | 2 |  | 9 | 51 | 14 |
| portfl3 | 3 | 2 |  | 11 | 53 | 14 |
| portfl4 | 3 | 2 |  | 10 | 50 | 19 |
| portfl6 | 3 | 2 |  | 9 | 49 | 20 |
| 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 |
| qudlin | 2 | 2 |  | 26 | 19 | 2 |
| recipe | 11 | 2 | 2 | 3 | 3 | 12 |
| res | 1 | 2 |  | 10 |  | 1 |
| 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 |  | 86 | 31 | 43 | 540 | 91 |
| sim2bqp | 3 | 2 |  | 8 | 10 | 3 |
| simbqp | 3 | 2 |  | 8 | 11 | 2 |
| simpllpa | 6 | 2 |  | 14 |  | 5 |
| simpllpb | 6 | 2 |  | 11 |  | 4 |
| 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 |
| supersim | 3 | 2 |  | 2 |  | 7 |
| swopf | 355 | 6 | 24 | 17 | 137 | 290 |
| synthes1 | 17 | 5 | 10 | 10 | 22 | 13 |
| tame | 3 | 2 |  | 6 | 9 | 2 |
| tointqor | 3 | 2 |  | 2 | 164 | 8 |
| try-b | 18 | 8 | 10 | 20 | 11 | 13 |
| twobars | 15 | 8 | 11 | 10 | 32 | 13 |
| vanderm4 | 2514 | 1 | 57 | 51 | 34 | 36 |

| Problem | filter-al | filterSQP | snopt | ipopt | minos | lancelot |
|---|---|---|---|---|---|---|
| watson | 17 | 21 | 172 | 14 | 118 | 44 |
| weeds | | 39 | 51 | 32 | 9 | 3 |
| womflet | 412 | 9 | 27 | 12 | 135 | 97 |
| yfit | 6 | 48 | 95 | 185 | 131 | 103 |
| yfitu | | 48 | 95 | 69 | 131 | 103 |
| zangwil2 | 3 | 2 | | 2 | 12 | 4 |
| zangwil3 | 3 | 2 | | 2 | | 3 |
| zecevic2 | 6 | 2 | | 9 | 10 | 7 |
| 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 |

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (Argonne). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. http://energy.gov/downloads/doe-public-access-plan.