IMA Journal of Numerical Analysis (2008) **28**, 749–769
doi:10.1093/imanum/drn003
Advance Access publication on March 14, 2008

Flexible penalty functions for nonlinear constrained optimization

FRANK E. CURTIS†
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208-3118, USA

AND

JORGE NOCEDAL
Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208-3118, USA

[Received on 11 April 2007; revised on 5 January 2008]

Dedicated to Prof. M. J. D. Powell, pioneer of nonlinear optimization, on occasion of his 70th birthday.

We propose a globalization strategy for nonlinear constrained optimization. The method employs a 'flexible' penalty function to promote convergence, where during each iteration the penalty parameter can be chosen as any number within a prescribed interval, rather than a fixed value. This increased flexibility in the step acceptance procedure is designed to promote long productive steps for fast convergence. An analysis of the global convergence properties of the approach in the context of a line search sequential quadratic programming method and numerical results for the KNITRO software package are presented.

Keywords: nonlinear programming; constrained optimization; sequential quadratic programming; penalty functions; global convergence.

1. Introduction

In this paper, we consider step acceptance mechanisms for nonlinear constrained optimization. For simplicity, we frame our discussion in the context of the equality-constrained problem

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \ f(x) \\ &\text{s.t. } c(x) = 0, \end{aligned} \tag{1.1}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$ and $c\colon \mathbb{R}^n \to \mathbb{R}^t$ are smooth functions, but consider ways in which our methods can be applied to problems with inequality constraints in Section 5. The main purpose of this paper is to develop a globalization strategy designed to promote long productive steps and fast convergence, supported by convergence guarantees to first-order optimal points.

Most globally convergent iterative algorithms for problem (1.1) have the following general form. At a given iterate x_k, a step is computed in either the primal or primal–dual space based on local and/or historical information of the problem functions. The step is then either accepted or rejected based on the reductions attained in the nonlinear objective $f(x)$, constraint infeasibility $\|c(x)\|$ or some combination

†Email: fecurt@gmail.com

of both measures. Here, $\|\cdot\|$ denotes a norm on $\mathbb{R}^t$. The manner in which these reductions are quantified and evaluated may have a significant impact on the types of steps accepted and the speed with which the algorithm converges to a solution.

We motivate our proposed globalization strategy, i.e. step acceptance method, by outlining two popular tools used for this purpose: exact penalty functions and filter mechanisms. The exact penalty function we consider in this paper combines the objective and a constraint infeasibility measure into a function of the form

$$\phi_\pi(x) \triangleq f(x) + \pi\|c(x)\|, \tag{1.2}$$

where $\pi > 0$ is a penalty parameter. During iteration k, a step is deemed acceptable only if a sufficient reduction in ϕ_{π_k} is attained for a suitable value of the penalty parameter. In contemporary algorithms, the value for π_k is chosen upon completion of the step computation procedure and the sequence $\{\pi_k\}$ is typically monotonically increasing throughout the run of the algorithm. Figure 1 illustrates the region of acceptable points from $p_k = (\|c(x_k)\|, f(x_k))$, corresponding to the current iterate x_k, in $\|c\|$–f space. A step d_k is acceptable if the resulting point $\bar{x} = x_k + d_k$ yields a pair $(\|c(\bar{x})\|, f(\bar{x}))$ lying sufficiently below the solid line through p_k, where the slope of the line is defined by the current value of the penalty parameter π_k. The global convergence properties of such an approach were first shown by Han (1977) and Powell (1978).

A filter mechanism, proposed by Fletcher & Leyffer (2002), avoids the definition of a parameter to balance reductions in the objective with reductions in the constraints. In the spirit of multiobjective optimization, a filter considers pairs of values $(\|c(x)\|, f(x))$ obtained by evaluating the functions $\|c\|$ and f at all or some iterates preceding the current one. A pair $(\|c(x_i)\|, f(x_i))$ is said to dominate another pair $(\|c(x_j)\|, f(x_j))$ if and only if both $\|c(x_i)\| \leqslant \|c(x_j)\|$ and $f(x_i) \leqslant f(x_j)$. The filter $\mathcal{F}$ is then defined to be an index set corresponding to a list of pairs such that no pair dominates any other. A step d_k from x_k is considered acceptable if the resulting point $\bar{x}$ corresponds to a pair $(\|c(\bar{x})\|, f(\bar{x}))$

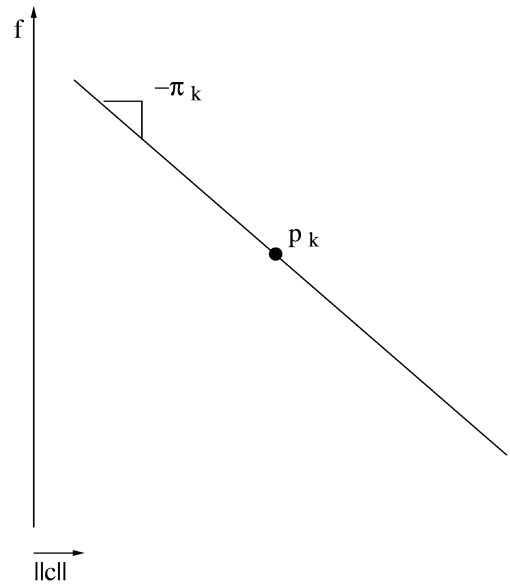

FIG. 1. Boundary of the region of acceptable points from p_k for the penalty function ϕ_{π_k}.

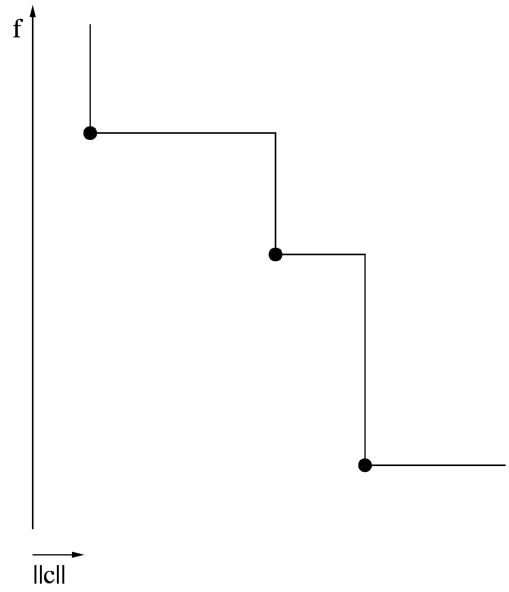

FIG. 2. Boundary of the region of acceptable points for a filter with three entries.

such that either

$$\|c(\bar{x})\| \ll \|c(x_i)\| \quad \text{or} \quad f(\bar{x}) \ll f(x_i) \tag{1.3}$$

for all $i \in \mathcal{F}$, where by '$\ll$' we mean that the value is less with respect to some appropriate margin. Upon the acceptance of such a step, the pair $(\|c(\bar{x})\|, f(\bar{x}))$ may be added to the filter, in which case all points dominated by this pair are removed from $\mathcal{F}$. Figure 2 illustrates the region of acceptable points for a filter with three entries as that lying sufficiently below and to the left of the piecewise linear function. The global convergence guarantees of such an approach have been shown when paired with certain types of step computation methods; e.g. see Fletcher *et al.* (1998), Gonzaga *et al.* (2003) and Wächter & Biegler (2005a,b).

Penalty functions and filter mechanisms both have their own advantages and disadvantages. One disadvantage of a penalty function relates to the monotonicity required when updating the penalty parameter π during the solution process. Nonmonotone updates for the penalty parameter that maintain global convergence guarantees are available, but such methods often rely on *ad hoc* heuristics that eventually fall back on the convergence properties of monotone strategies, and so we do not discuss them here. Depending on the specific update strategy used, π may at some point be set to an excessively large value, even at a point that is relatively far from a solution. As a result, a large priority will be placed on computing steps that produce sufficient reductions in constraint infeasibility, effectively 'blocking' steps that move away from the feasible region. This can be detrimental as empirical evidence has shown that accepting steps that temporarily increase infeasibility can often lead to fast convergence. Figure 3 illustrates this blocking behaviour of a penalty function, where we highlight the region of points that would be rejected despite the fact that each corresponding step would have provided a reduction in the objective f (and so may have been acceptable to a filter).

We note that a second disadvantage of a penalty function is that a low value of π may block steps that improve feasibility but increase f. However, modern step acceptance strategies effectively deal with this

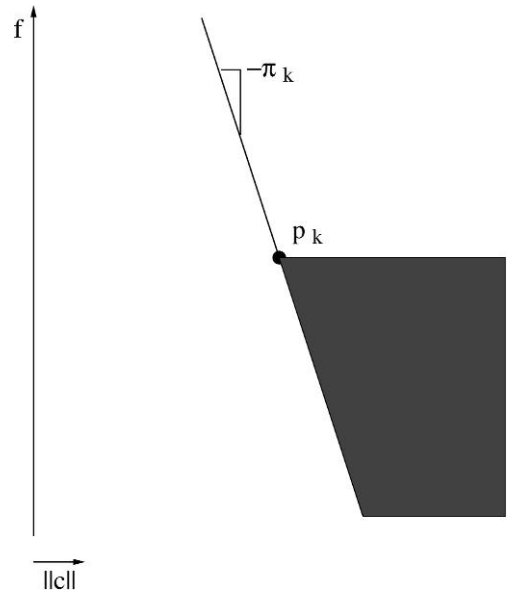

FIG. 3. A region of points blocked by ϕ_{π_k}.

problem by defining local models of ϕ_π (as will be seen in Section 3), with which an adequately large value of π can be determined to avoid excessive blocking. Thus, our view is that the main weakness of penalty-based strategies is the blocking effect illustrated in Fig. 3, which can be particularly detrimental when $\|c_k\|$ is zero, or at least small, while x_k is far from optimal.

One disadvantage of a filter mechanism is that a step can be blocked by a filter entry, i.e. historical information of the problem functions, when in fact the step is a productive move towards a solution in a local region of the search space. This is particularly worrisome when steps are blocked that would amount to a sufficient reduction in constraint infeasibility. Figure 4 depicts a filter with the single entry a where the point $p_k = (\|c(x_k)\|, f(x_k))$, corresponding to the current iterate x_k, is shown as the isolated point with an objective value sufficiently less than the filter entry. The shaded portion illustrates one region of points that are blocked by the filter, despite the fact that a step into this region would correspond to a reduction in constraint infeasibility from the current iterate (and so may be acceptable for a penalty function approach with parameter π_k).

In an extreme example, consider the case where the filter entry a in Fig. 4 is a Pareto optimal solution to the multiobjective optimization problem of minimizing the pair $(\|c(x)\|, f(x))$ over all $x \in \mathbb{R}^n$. A point is Pareto optimal if it cannot be dominated by any other point. Thus, if the current iterate again corresponds to the point p_k in Fig. 4, then all paths from p_k to the feasible region must pass through a region of points dominated by a in $\|c\|-f$ space. Feasibility can only be attained if a single computed step were to fall beyond the region dominated by the filter entry or if a backup mechanism, such as a feasibility restoration phase, were implemented.

In summary, both penalty functions and filters can be shown to block different types of productive steps. A penalty function may suffer from high priority being placed on improving feasibility and convergence can be slowed by forcing the algorithm to hug the feasible region. A filter mechanism, on the other hand, may suffer from handling problem (1.1) too much like a multiobjective optimization

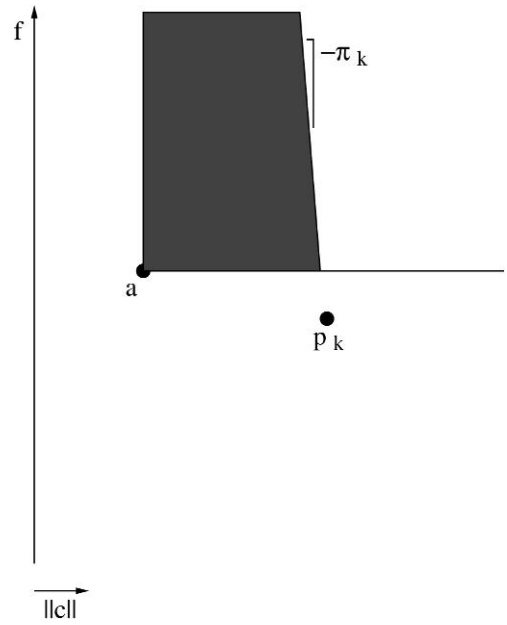

FIG. 4. A region of points blocked by a filter with entry a.

problem, when in fact a certain priority on converging to the feasible region may be appropriate, especially as the algorithm progresses.

2. Flexible penalty functions

In this section, we define a new step acceptance mechanism for nonlinear programming algorithms. By observing the strengths and weaknesses of penalty functions and filters, we hope to emulate some of the step acceptance behaviours of both methods while attempting to avoid any blocking of productive steps.

During early iterations, the filter mechanism has the benefit that a variety of steps are considered acceptable. For example, for a one-element filter, i.e. a filter containing only an entry corresponding to the current iterate, a step will be accepted as long as a sufficient reduction in the objective or constraint infeasibility is attained. This may be of use to promote long steps during early iterations when an appropriate value for the penalty parameter may not yet be known. However, during later iterations, it may be reasonable to assume that an appropriate value for the penalty parameter may be determinable based on information computed throughout the run of the algorithm, which can be used to correctly block steps from increasing constraint infeasibility. The use of a penalty function in later iterations may also avoid the risk of blocking steps in the manner illustrated in Fig. 4.

In an attempt to define a single mechanism that will capture all of these characteristics, and given that the penalty function approach appears to be more flexible than a filter in that it permits a reweighting of objective and constraint infeasibility measures, we present an improvement of the penalty strategies.

Our method can be motivated by observing the iterative nature of the penalty parameter update implemented in some current algorithms; e.g. see Waltz *et al.* (2006). At the start of iteration k, a specific value π_{k-1} of the penalty parameter is carried over from the previous iteration. If the algorithm were to maintain this value, then only a step corresponding to a move into the region sufficiently below the solid

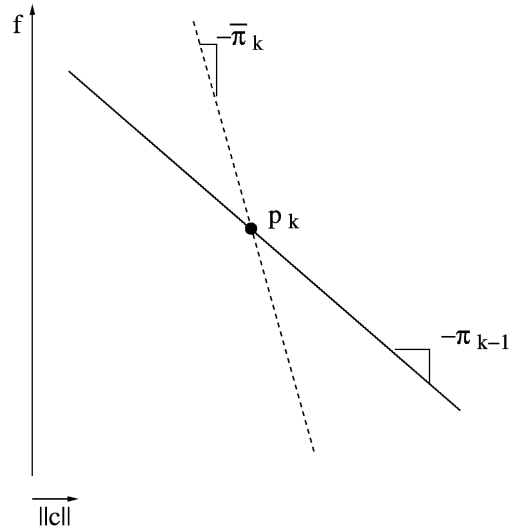

FIG. 5. Illustration of the iterative nature of penalty parameter updates.

line in Fig. 5 would be acceptable. However, upon the calculation of d_k, the algorithm may determine that an increase of the penalty parameter to some value $\bar{\pi}_k > \pi_{k-1}$ may be appropriate, in which case only a step corresponding to a move into the region sufficiently below the dashed line in Fig. 5 would be acceptable. Rather than automatically setting $\pi_k \leftarrow \bar{\pi}_k$, a simple heuristic that maintains the global convergence properties of the algorithm is to first compute the function values for $\bar{x} = x_k + d_k$, namely, $\|c(\bar{x})\|$ and $f(\bar{x})$. If $(\|c(\bar{x})\|, f(\bar{x}))$ lies sufficiently below the dashed line in Fig. 5, then we may accept the step and indeed set $\pi_k \leftarrow \bar{\pi}_k$. However, if $(\|c(\bar{x})\|, f(\bar{x}))$ lies sufficiently below the solid line, then the step could be considered acceptable for setting $\pi_k \leftarrow \pi_{k-1}$, effectively avoiding an increase in the penalty parameter. In summary, such a strategy does not consider a single value of π at x_k, but rather may select from a pair of values depending on the actual reductions attained by the step. Thus, we can view the region of acceptable points as that lying below the solid *or* dashed line in Fig. 5.

An extension of this idea forms the basis of the method we now propose. Consider the collection of penalty functions

$$\phi_\pi(x) \triangleq f(x) + \pi \|c(x)\|, \quad \pi \in [\pi^l, \pi^u], \tag{2.1}$$

for $0 < \pi^l \leqslant \pi^u$. We define a step to be acceptable if a sufficient reduction in ϕ_π has been attained for *at least one* $\pi \in [\pi^l, \pi^u]$. Clearly, if π^l is always chosen to equal π^u, then this approach is equivalent to using a penalty function with a fixed π during each iteration. Alternatively, if $\pi^l \approx 0$ while π^u is very large, then this approach has the form of a one-element filter. In general, the region of acceptable points is that given by the region down and to the left of the piecewise linear function illustrated in Fig. 6, where the kink in the function always occurs at $p_k = (\|c(x_k)\|, f(x_k))$, corresponding to the current iterate x_k. As the penalty parameter π is allowed to fluctuate in the interval $[\pi^l, \pi^u]$, we refer to (2.1) as a 'flexible' penalty function.

Let us expound further on the relationship between our approach and some techniques that employ a filter by saying that the region of acceptable points in Fig. 6 has features similar to the 'slanting envelope' around a filter entry proposed by Chin & Fletcher (2003) and considered later in a paper by Li (2006). However, despite the fact that the shape of the acceptable regions is similar in some areas of the

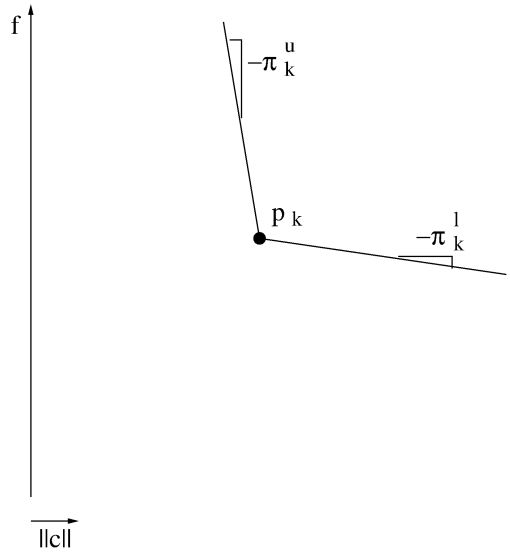

FIG. 6. Boundary of the region of acceptable points from p_k for a flexible penalty function over $[\pi_k^l, \pi_k^u]$.

$\|c\|-f$ plane, the important difference between our flexible penalty function and these and other filter mechanisms is that we do not maintain a collection of previous infeasibility measure/objective value pairs. The step acceptance criteria we propose for a flexible penalty function depend only on π_k^l, π_k^u and constraint and objective information at the current iterate x_k.

The practical behaviour of standard penalty function techniques depends heavily on the update strategy for the single parameter π. For a flexible penalty function, we need to only consider the update strategies for two parameters: π^l and π^u. As different requirements in terms of convergence guarantees are necessary for each of these boundary values, and as they have significantly different practical effects, we have the ability to design their updates in a manner suitable for accepting long productive steps.

We present a concrete strategy for updating π^l and π^u in Section 3 as certain details are better described once features of the chosen step computation procedure are outlined.

Notation. In the remainder of our discussion, we drop functional notation once values are clear from the context and delimit iteration number information for functions as with variables; i.e. we denote $f_k \triangleq f(x_k)$ and similarly for other quantities. We define $\|\cdot\|$ to be any fixed norm.

3. A line search sequential quadratic programming framework

In this section, we describe a precise globalization strategy for problem (1.1) based on the flexible penalty function (2.1) in the context of a line search sequential quadratic programming (SQP) method.

Let us begin by formalizing a basic SQP method. The Lagrangian function for problem (1.1) is

$$\mathcal{L}(x, \lambda) \triangleq f(x) + \lambda^{\mathrm{T}} c(x) \tag{3.1}$$

and the first-order optimality conditions are

$$\nabla \mathcal{L}(x, \lambda) = \begin{bmatrix} g(x) + A(x)^{\mathrm{T}} \lambda \\ c(x) \end{bmatrix} = 0, \tag{3.2}$$

where $g(x) \triangleq \nabla f(x)$, $A(x)$ is the Jacobian of $c(x)$ and $\lambda \in \mathbb{R}^t$ are Lagrange multipliers. The line search SQP methodology applied to problem (1.1) defines an appropriate displacement d_k in the primal space from an iterate x_k as the minimizer of a quadratic model of the objective subject to a linearization of the constraints. The quadratic program has the form

$$\min_{d \in \mathbb{R}^n} f(x_k) + g(x_k)^{\mathrm{T}} d + \tfrac{1}{2} d^{\mathrm{T}} W(x_k, \lambda_k) d \tag{3.3a}$$

$$\text{s.t. } c(x_k) + A(x_k) d = 0, \tag{3.3b}$$

where

$$W(x, \lambda) \approx \nabla^2_{xx} \mathcal{L}(x, \lambda) = \nabla^2_{xx} f(x) + \sum_{i=1}^{t} \lambda^i \nabla^2_{xx} c^i(x)$$

is equal to, or is a symmetric approximation for, the Hessian of the Lagrangian. Here, $c^i(x)$ and λ^i denote the ith constraint function and its corresponding dual variable, respectively. If the constraint Jacobian $A(x_k)$ has full row rank and $W(x_k, \lambda_k)$ is positive definite on the null space of $A(x_k)$, then a solution d_k to (3.3) is well defined and can be obtained via the solution of the linear system (see Nocedal & Wright, 2006):

$$\begin{bmatrix} W(x_k, \lambda_k) & A(x_k)^{\mathrm{T}} \\ A(x_k) & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \delta_k \end{bmatrix} = - \begin{bmatrix} g(x_k) + A(x_k)^{\mathrm{T}} \lambda_k \\ c(x_k) \end{bmatrix}. \tag{3.4}$$

The new iterate is then given by

$$x_{k+1} \leftarrow x_k + \alpha_k d_k,$$

where the step-length coefficient $\alpha_k \in (0, 1]$ is given by a globalization procedure. Here, we intend to employ the flexible penalty function (2.1), requiring appropriate update strategies for π^l and π^u. In the following discussion, let us assume that $\|c_k\| \neq 0$ for each k. We comment on suitable updates for π^l_k and π^u_k in the special case of $\|c_k\| = 0$ at the end of this section.

First, consider the parameter π^u. A large value of π^u indicates that the algorithm considers almost any step that provides a sufficiently large reduction in constraint infeasibility to be acceptable. Thus, as approaching the feasible region is a necessity for any algorithm for solving problem (1.1), we may choose to initialize π^u to a large value and increase it only when necessary. This can be done by updating π^u in a manner currently used for setting π in some contemporary penalty function approaches. The technique we have in mind makes decisions based on a model m_π of the penalty function ϕ_π and in effect will increase π (or, in our case, π^u) if and only if the computed step indicates that a large increase in the objective is likely to result from a reduction in constraint infeasibility.

Let us define a local model of ϕ_π around the current iterate x_k as

$$m_\pi(d) = f_k + g_k^{\mathrm{T}} d + \frac{\omega(d)}{2} d^{\mathrm{T}} W_k d + \pi \|c_k + A_k d\|,$$

where

$$\omega(d) \triangleq \begin{cases} 1 & \text{if } d^{\mathrm{T}} W_k d \geqslant 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

(e.g. see El-Hallabi, 1999, Byrd *et al.*, 1999, Omojokun, 1989, and Waltz *et al.*, 2006). Notice that m_π contains a linear or quadratic model of the objective f and a linear approximation of constraint infeasibility. With this approximation, we can estimate the reduction in ϕ_π attained by d_k by evaluating

$$\begin{aligned} m\mathrm{red}_\pi(d_k) &\triangleq m_\pi(0) - m_\pi(d_k) \\ &= \left[-g_k^{\mathrm{T}} d_k - \frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k\right] + \pi \|c_k\|. \end{aligned} \tag{3.6}$$

As the step d_k satisfies the linearized constraints in problem (3.3), it follows that the model predicts no increase in constraint infeasibility, as evidenced by the non-negative contribution of the last term in (3.6). Our model of the objective, however, may indicate that an increase or decrease in f (corresponding to a negative or positive value, respectively, of the term in square brackets in (3.6)) is likely to occur along d_k. Overall, we consider the reduction in the model m_π attained by d_k to be sufficiently large if

$$m\mathrm{red}_\pi(d_k) \geqslant \sigma \pi \|c_k\|, \tag{3.7}$$

for some $0 < \sigma < 1$, which can be seen to hold if

$$\pi \geqslant \frac{g_k^{\mathrm{T}} d_k + \frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k}{(1-\sigma)\|c_k\|} \triangleq \chi_k. \tag{3.8}$$

Various algorithms will in fact enforce inequality (3.7) and so will set π according to (3.8) for all k.

It turns out that our desired properties of π^u can also be achieved by constructing an update around the term χ_k. In particular, we propose a scheme of the form

$$\pi_k^u \leftarrow \begin{cases} \pi_{k-1}^u & \text{if } \pi_{k-1}^u \geqslant \chi_k, \\ \chi_k + \varepsilon & \text{otherwise,} \end{cases} \tag{3.9}$$

where $\varepsilon > 0$ is a small constant. In this manner, π^u will be increased during an iteration if and only if an increase in the model objective, reflected by a positive numerator in (3.8), indicates that an increase in f is likely to occur in conjunction with a move towards the feasible region, implied by the fact that the step satisfies the linearized constraints in (3.3).

By using the model m_π to set a value for the penalty parameter, the resulting sequence of values can be shown to remain bounded under common assumptions due to certain desirable properties of the quadratic subproblem (3.3). (This phenomenon, which remains important for our flexible penalty function approach in the context of π^u, can be observed more precisely in our proof of Lemma 3.8 in Section 3.1.) A drawback of this technique, however, is that such a model may not always accurately reflect changes in the objective and constraint values. For example, $m\mathrm{red}_\pi$ may suggest that a move along d_k corresponds to a decrease in constraint infeasibility and an increase in the objective, when in fact the opposite may occur if one were to take the full step d_k. As such, the penalty parameter may be set to a large value that results in excessive blocking in later iterations. Further, motivation for incorporating a flexible penalty function, therefore, results from the fact that an excessively large value

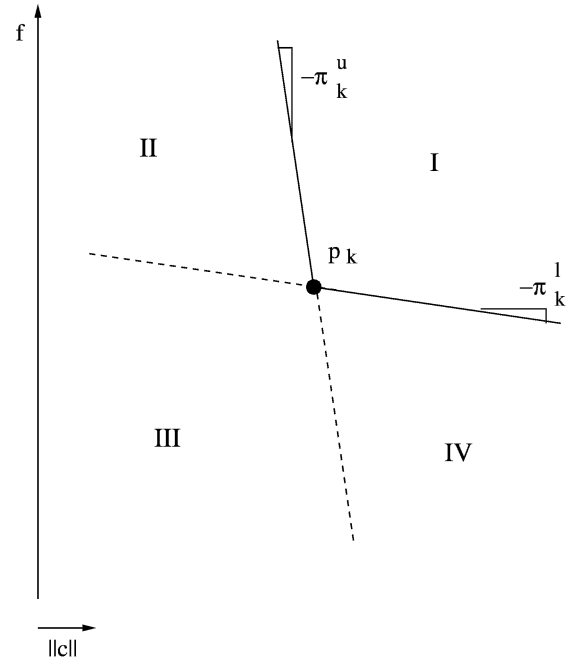

FIG. 7. Distinct regions defined by the current state of a flexible penalty function.

for π_k^u is less of a concern if the penalty parameter is able to fluctuate over an interval $[\pi_k^l, \pi_k^u]$ during the line search—especially if the mechanism for choosing π_k^l is not based on local models of the functions at all.

The method we propose for setting π^l is such a technique. In particular, we choose to have π_k^l set in a manner that reflects the actual reductions in f and $\|c\|$ attained during the *previous* iteration $k-1$ (where π_0^l is provided as a small initial value).

To motivate the details of the scheme we propose, consider the numbered regions illustrated in Fig. 7, where the position and shape of each portion depend on the parameters π_k^l (set during iteration $k-1$) and π_k^u, and the location of the point $p_k = (\|c(x_k)\|, f(x_k))$. A step into region I would not be acceptable to the flexible penalty function (2.1), as opposed to a step into region II, III or IV, which would be acceptable. Our strategy for setting π_{k+1}^l will depend on the region in $\|c\|$–f space to which the step $\alpha_k d_k$ moved upon the conclusion of the line search. If a sufficient reduction in $\phi_{\pi_k^l}$ was obtained (i.e. the step was into region III or IV), then we say that the reductions in f and/or $\|c\|$ are sufficient for the current state of the flexible penalty function and so we set $\pi_{k+1}^l \leftarrow \pi_k^l$. Otherwise, i.e. if the step was into region II, π^l will be increased. This has the logical interpretation that we only become more restrictive by blocking steps that increase infeasibility when the algorithm is confronted with steps that indicate that *actual* moves towards the feasible region correspond to *actual* increases in the objective (thus freeing ourselves from being bound by parameters set based on models or other local information). The precise update after a step into region II is given by

$$\pi_{k+1}^l \leftarrow \min\{\pi_k^u, \pi_k^l + \max\{0.1(v - \pi_k^l), \varepsilon^l\}\}, \tag{3.10}$$

where $\varepsilon^l > 0$ is some small constant and

$$\nu = \frac{f(x_k + \alpha_k d_k) - f(x_k)}{\|c(x_k)\| - \|c(x_k + \alpha_k d_k)\|}. \tag{3.11}$$

Here, the definition of ν ensures that the value for π^l_{k+1} depends on the actual reductions in the objective and constraint infeasibility attained by $\alpha_k d_k$, where it can be seen that $\nu \in [\pi^l_k, \pi^u_k]$ after a step into region II. We introduce the damping factor 0.1 so that the value for π^l will increase only gradually, thus blocking as few future steps as possible while still ensuring convergence.

Our procedures for updating the state of the flexible penalty function (2.1) are now set. Before presenting the algorithm in detail, however, let us remark on an important detail of the line search procedure for computing α_k. With $D\phi_\pi(d_k)$ denoted as the directional derivative of ϕ_π along d_k, we require that α_k satisfy the Armijo condition

$$\phi_\pi(x_k + \alpha_k d_k) \leqslant \phi_\pi(x_k) + \eta\alpha_k D\phi_{\pi^m_k}(d_k), \quad \text{for some } \pi \in [\pi^l_k, \pi^u_k], \tag{3.12}$$

where $0 < \eta < 1$ and $\pi^m_k \in [\pi^l_k, \pi^u_k]$. Note that we have defined a parameter π^m_k for calculating a single value of the directional derivative, which must be chosen to ensure that this term is sufficiently negative for each k. This could be achieved by choosing $\pi^m_k = \pi^u_k$ for all k (see Lemma 3.7). However, as seen in Theorem 18.2 of Nocedal & Wright (2006), the directional derivative is given by

$$D\phi_{\pi^m_k}(d_k) = g_k^{\mathrm{T}} d_k - \pi^m_k \|c_k\| \tag{3.13}$$

and so larger values of π^m_k will make this term more negative. As fewer values of α_k will satisfy (3.12) for more negative values of $D\phi_{\pi^m_k}(d_k)$, we would like to choose π^m_k in the interval $[\pi^l_k, \pi^u_k]$ so that this term is negative enough to ensure sufficient descent, while also being as close to zero as possible so as to allow the largest number of acceptable step-lengths. We use

$$\pi^m_k \leftarrow \max\{\pi^l_k, \chi_k\}, \tag{3.14}$$

which, along with (3.9) and the fact that $\pi^l_k \leqslant \pi^u_{k-1}$ (see (3.10)), ensures $\pi^m_k \geqslant \chi_k$ and $\pi^m_k \in [\pi^l_k, \pi^u_k]$.

Overall, we have described the following algorithm.

ALGORITHM 3.1 **Line search SQP method with a flexible penalty function**

Initialize x_0, λ_0, $0 < \pi^l_0 \leqslant \pi^u_{-1}$, $0 < \varepsilon, \varepsilon^l$, and $0 < \eta, \sigma < 1$
for $k = 0, 1, 2, \ldots,$ until a convergence test for problem (1.1) is satisfied
  Compute f_k, g_k, c_k, W_k, and A_k and set $\alpha_k \leftarrow 1$
  Compute (d_k, δ_k) via (3.4)
  If $c_k \neq 0$, set π^u_k according to (3.9) and π^m_k by (3.14); else, set $\pi^u_k \leftarrow \pi^u_{k-1}$ and $\pi^m_k \leftarrow \pi^m_{k-1}$
  until the Armijo condition (3.12) holds for some $\pi \in [\pi^l_k, \pi^u_k]$, set $\alpha_k \leftarrow \alpha_k/2$
  If the Armijo condition (3.12) holds for $\pi = \pi^l_k$, set $\pi^l_{k+1} \leftarrow \pi^l_k$; else, set π^l_{k+1} by (3.10)
  Set $(x_{k+1}, \lambda_{k+1}) \leftarrow (x_k, \lambda_k) + \alpha_k(d_k, \delta_k)$
endfor

A practical implementation of the line search procedure of Algorithm 3.1 is attained by the observation that, during iteration k, the Armijo condition (3.12) is satisfied for $\pi \in [\pi^l_k, \pi^u_k]$ if and only if it is satisfied for either $\pi = \pi^l_k$ or $\pi = \pi^u_k$. Thus, the line search for a given step d_k can be performed

simply by evaluating the reductions attained in $\phi_{\pi_k^l}$ and $\phi_{\pi_k^u}$. We also note that in the special case of $\|c_k\| = 0$ during iteration k, we maintain $\pi_k^u \leftarrow \pi_{k-1}^u$ as in this case the directional derivative $D\phi_\pi(d_k)$ is independent of π (see (3.13)). We can also trivially set $\pi_k^m \leftarrow \pi_k^l$ and maintain $\pi_{k+1}^l \leftarrow \pi_k^l$ since in this setting region II of Fig. 7 is empty.

3.1 *Global analysis*

In this section, we explore the global convergence properties of Algorithm 3.1 under the following assumptions.

ASSUMPTION 3.2 The sequence $\{x_k, \lambda_k\}$ generated by Algorithm 3.1 is contained in a convex set Ω and the following properties hold:

(a) The functions f and c and their first and second derivatives are bounded on Ω.
(b) The constraint Jacobians A_k have full row rank and their smallest singular values are bounded below by a positive constant.
(c) The sequence $\{W_k\}$ is bounded.
(d) There exists a constant $\mu > 0$ such that over all k and for any $u \in \mathbb{R}^n$ with $u \neq 0$ and $A_k u = 0$, we have $u^T W_k u \geqslant \mu \|u\|^2$.

These assumptions are fairly standard for a line search method; e.g. see Han (1977) and Powell (1983). Assumption 3.2(b), however, is strong, but we use it to simplify the analysis in order to focus on the issues related to the incorporation of a flexible penalty function. Assuming that W_k is positive definite on the null space of the constraints is natural for line search algorithms, for otherwise there would be no guarantee of descent.

Our analysis hinges on our ability to show that the algorithm will eventually compute an infinite sequence of steps that sufficiently reduce the penalty function ϕ_{π^l} for a fixed $\pi^l > 0$, which we achieve by following the approach taken in Byrd *et al.* (2008) for an inexact SQP method. In particular, we consider the decomposition

$$d_k = u_k + v_k, \tag{3.15}$$

where the tangential component u_k lies in the null space of the constraint Jacobian A_k and the normal component v_k lies in the range space of A_k^T. The components are not to be computed explicitly; the decomposition is only for analytical purposes. We refer to u_k, which by definition satisfies $A_k u_k = 0$, as the tangential component and v_k as the normal component.

We first present a result related to the length of the primal step d_k and the sequence of Lagrange multiplier estimates $\{\lambda_k\}$.

LEMMA 3.3 For all k, the primal step d_k is bounded in norm. Moreover, the sequence of Lagrange multipliers $\{\lambda_k\}$ is bounded.

Proof. Under Assumption 3.2, it can be shown that the primal–dual matrix in (3.4) is nonsingular and that its inverse is bounded in norm over all k (e.g. see Nocedal & Wright, 2006). Thus, the relation

$$\begin{bmatrix} d_k \\ \lambda_k + \delta_k \end{bmatrix} = -\begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix}^{-1} \begin{bmatrix} g_k \\ c_k \end{bmatrix}$$

implies that $\|(d_k, \lambda_k + \delta_k)\| \leqslant \gamma \|(g_k, c_k)\|$ holds over all k for some constant $\gamma > 0$. The results then follow from $\alpha_k \leqslant 1$ and the fact that Assumption 3.2(a) implies that $\|(g_k, c_k)\|$ is bounded over all k. □

The next result ensures a precise bound on the length of the normal component v_k with respect to the current value of the infeasibility measure.

LEMMA 3.4 There exists $\gamma_1 > 0$ such that, for all k,

$$\|v_k\|^2 \leqslant \gamma_1 \|c_k\|. \tag{3.16}$$

Proof. From $A_k v_k = A_k d_k = -c_k$ and the fact that v_k lies in the range space of A_k^{T}, it follows that

$$v_k = -A_k^{\mathrm{T}}(A_k A_k^{\mathrm{T}})^{-1} c_k$$

and so

$$\|v_k\| \leqslant \|A_k^{\mathrm{T}}(A_k A_k^{\mathrm{T}})^{-1}\| \|c_k\|.$$

The result follows from the facts that Assumption 3.2(a) states that $\|c_k\|$ is bounded and Assumptions 3.2(a) and (b) imply that $\|A_k^{\mathrm{T}}(A_k A_k^{\mathrm{T}})^{-1}\|$ is bounded. □

We now turn to the following result concerning an important property of the tangential steps.

LEMMA 3.5 There exists a constant $\gamma_2 > 0$ such that, over all k, if $\|u_k\|^2 \geqslant \gamma_2 \|v_k\|^2$, then $\frac{1}{2} d_k^{\mathrm{T}} W_k d_k \geqslant \frac{\mu}{4} \|u_k\|^2$.

Proof. Assumption 3.2(d) implies that for any $\gamma_2 > 0$ such that $\|u_k\|^2 \geqslant \gamma_2 \|v_k\|^2$, we have

$$\begin{aligned} \tfrac{1}{2} d_k^{\mathrm{T}} W_k d_k &= \tfrac{1}{2} u_k^{\mathrm{T}} W_k u_k + u_k^{\mathrm{T}} W_k v_k + \tfrac{1}{2} v_k^{\mathrm{T}} W_k u_k \\ &\geqslant \frac{\mu}{2} \|u_k\|^2 - \|u_k\| \|W_k\| \|v_k\| - \frac{1}{2} \|W_k\| \|v_k\|^2 \\ &\geqslant \left(\frac{\mu}{2} - \frac{\|W_k\|}{\sqrt{\gamma_2}} - \frac{\|W_k\|}{2\gamma_2} \right) \|u_k\|^2. \end{aligned}$$

Thus, with Assumption 3.2(c) we have that the result holds for some sufficiently large $\gamma_2 > 0$. □

With the above results, we can now identify two types of iterations. Let $\gamma_2 > 0$ be chosen large enough as described in Lemma 3.5 and consider the sets of indices

$$K_1 \triangleq \{k : \|u_k\|^2 \geqslant \gamma_2 \|v_k\|^2\} \quad \text{and} \quad K_2 \triangleq \{k : \|u_k\|^2 < \gamma_2 \|v_k\|^2\}.$$

Our remaining analysis will be dependent on these sets and the corresponding quantity

$$\Theta_k \triangleq \begin{cases} \|u_k\|^2 + \|c_k\| & k \in K_1, \\ \|c_k\| & k \in K_2. \end{cases}$$

The quantity Θ_k will help us form a common bound for the length of the primal step and the quantity $D\phi_{\pi_k^m}(d_k)$.

LEMMA 3.6 There exists $\gamma_3 > 1$ such that, for all k,

$$\|d_k\|^2 \leqslant \gamma_3 \Theta_k$$

and hence

$$\|d_k\|^2 + \|c_k\| \leqslant 2\gamma_3 \Theta_k. \tag{3.17}$$

Proof. For $k \in K_1$, Lemma 3.4 implies

$$\|d_k\|^2 = \|u_k\|^2 + \|v_k\|^2 \leqslant \|u_k\|^2 + \gamma_1 \|c_k\|.$$

Similarly, Lemma 3.4 implies that for $k \in K_2$

$$\|d_k\|^2 = \|u_k\|^2 + \|v_k\|^2 < (\gamma_2 + 1)\|v_k\|^2 \leqslant (\gamma_2 + 1)\gamma_1 \|c_k\|.$$

To establish (3.17), we note that $\Theta_k + \|c_k\| \leqslant 2\Theta_k$ for all k. □

The next result bounds the quantity $D\phi_{\pi_k^m}(d_k)$, where π_k^m is defined by (3.14).

LEMMA 3.7 There exists $\gamma_4 > 0$ such that, for all k,

$$D\phi_{\pi_k^m}(d_k) \leqslant -\gamma_4 \Theta_k.$$

Proof. Recall that by Theorem 18.2 in Nocedal & Wright (2006), we have

$$D\phi_{\pi_k^m}(d_k) = g_k^{\mathrm{T}} d_k - \pi_k^m \|c_k\|. \tag{3.18}$$

If $\|c_k\| = 0$, then (3.4) yields

$$D\phi_{\pi_k^m}(d_k) = g_k^{\mathrm{T}} d_k = -d_k^{\mathrm{T}} W_k d_k.$$

Lemmas 3.4 and 3.5 then imply $\|v_k\| = 0$ and $k \in K_1$, and so

$$D\phi_{\pi_k^m}(d_k) = -d_k^{\mathrm{T}} W_k d_k \leqslant -\frac{\mu}{2}\|u_k\|^2$$

and the result holds for $\gamma_4 = \frac{\mu}{2}$.

Now suppose $\|c_k\| \neq 0$. Here, (3.8), (3.18) and the fact that (3.14) implies $\pi_k^m \geqslant \chi_k$ yield

$$D\phi_{\pi_k^m}(d_k) \leqslant -\frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k - \sigma \pi_k^m \|c_k\|. \tag{3.19}$$

By Lemma 3.5 and (3.5), we have that $\omega_k = 1$ for $k \in K_1$ and thus

$$D\phi_{\pi_k^m}(d_k) \leqslant -\frac{\mu}{4}\|u_k\|^2 - \sigma \pi_k^m \|c_k\|.$$

Similarly, for $k \in K_2$ we have from (3.5) and (3.19) that

$$D\phi_{\pi_k^m}(d_k) \leqslant -\sigma \pi_k^m \|c_k\|.$$

The result holds for $\gamma_4 = \min\{\frac{\mu}{4}, \sigma\pi_k^m\}$, which is positive as $\pi_k^m \geqslant \pi_k^l \geqslant \pi_0^l > 0$ for all k. □

An important property of Algorithm 3.1 is that under Assumption 3.2 the sequence $\{\pi_k^u\}$ remains bounded. We prove this result next.

LEMMA 3.8 The sequence $\{\pi_k^u\}$ is bounded above and π_k^u remains constant for all sufficiently large k.

Proof. Recall that π_k^u is set during iteration k of Algorithm 3.1 to satisfy (3.8), which is equivalent to saying that (3.7) will be satisfied, as in

$$-g_k^{\mathrm{T}} d_k - \frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k + (1-\sigma)\pi_k^u \|c_k\| \geqslant 0. \tag{3.20}$$

If $d_k^{\mathrm{T}} W_k d_k \geqslant 0$, then $\omega_k = 1$ by (3.5) and so (3.4) and Lemma 3.3 imply that there exists $\gamma_5 > 0$ such that

$$-g_k^{\mathrm{T}} d_k - \tfrac{1}{2} d_k^{\mathrm{T}} W_k d_k = \tfrac{1}{2} d_k^{\mathrm{T}} W_k d_k - c_k^{\mathrm{T}}(\lambda_k + \delta_k) \geqslant -\gamma_5 \|c_k\|.$$

Similarly, if $d_k^{\mathrm{T}} W_k d_k < 0$, then $\omega_k = 0$, $k \in K_2$ and $\|d_k\|^2 \leqslant \gamma_3 \|c_k\|$ by Lemma 3.6. Then, Assumption 3.2, (3.4) and Lemma 3.3 imply that there exists $\gamma_6, \gamma_6' > 0$ such that

$$-g_k^{\mathrm{T}} d_k - \frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k = d_k^{\mathrm{T}} W_k d_k - c_k^{\mathrm{T}}(\lambda_k + \delta_k) \geqslant -\gamma_6(\|d_k\|^2 + \|c_k\|) \geqslant -\gamma_6' \|c_k\|.$$

These results together imply that for all k,

$$-g_k^{\mathrm{T}} d_k - \frac{\omega_k}{2} d_k^{\mathrm{T}} W_k d_k \geqslant -\max\{\gamma_5, \gamma_6'\} \|c_k\|$$

and so (3.20), and equivalently (3.7), is always satisfied if

$$\pi_k^u \geqslant \max\{\gamma_5, \gamma_6'\}/(1-\sigma).$$

Therefore, if $\pi_{\bar{k}}^u \geqslant \max\{\gamma_5, \gamma_6'\}/(1-\sigma)$ for some iteration number $\bar{k} \geqslant 0$, then $\pi_k^u = \pi_{\bar{k}}^u$ for all $k \geqslant \bar{k}$. This, together with the fact that whenever Algorithm 3.1 increases π^u it does so by at least a positive finite amount, proves the result. □

A similar result can be shown for the parameter π^l.

COROLLARY 3.9 $\{\pi_k^l\}$ is bounded above and π_k^l remains constant for all sufficiently large k.

Proof. By Lemma 3.8, π_k^u is constant for all sufficiently large k. Then, we have by (3.10) that if π^l is increased, then it is done so by at least a finite constant amount or it is set equal to π^u. Thus, the result follows from (3.10) and the fact that there can only be a finite number of increases of π^l. □

The previous lemmas can be used to bound the sequence of step-length coefficients.

LEMMA 3.10 The sequence $\{\alpha_k\}$ is bounded below by a positive constant.

Proof. Let us rewrite the Armijo condition (3.12) for convenience as

$$\phi_\pi(x_k + \alpha_k d_k) - \phi_\pi(x_k) \leqslant \eta \alpha_k D\phi_{\pi_k^m}(d_k) \tag{3.21}$$

for $\pi \in [\pi_k^l, \pi_k^u]$. Suppose that the line search fails for some $\bar{\alpha} > 0$, which means that (3.21) does not hold for any $\pi \in [\pi_k^l, \pi_k^u]$. In particular,

$$\phi_{\pi_k^m}(x_k + \bar{\alpha} d_k) - \phi_{\pi_k^m}(x_k) > \eta \bar{\alpha} D\phi_{\pi_k^m}(d_k),$$

where we recall that $\pi_k^m \in [\pi_k^l, \pi_k^u]$. As seen in Nocedal & Wright (2006, p. 541), it can be shown under Assumption 3.2 that for some $\gamma_7 > 0$, we have

$$\phi_{\pi_k^m}(x_k + \bar{\alpha} d_k) - \phi_{\pi_k^m}(x_k) \leqslant \bar{\alpha} D\phi_{\pi_k^m}(d_k) + \bar{\alpha}^2 \gamma_7 \pi_k^m \|d_k\|^2,$$

so

$$(\eta - 1) D\phi_{\pi_k^m}(d_k) < \bar{\alpha} \gamma_7 \pi_k^m \|d_k\|^2.$$

Lemmas 3.6 and 3.7 then yield

$$(1-\eta)\gamma_4\Theta_k < \bar{\alpha}\gamma_3\gamma_7\pi_k^m\Theta_k,$$

so

$$\bar{\alpha} > (1-\eta)\gamma_4/(\gamma_3\gamma_7\pi_k^m) > (1-\eta)\gamma_4/(\gamma_3\gamma_7\pi_k^u).$$

Thus, α_k is never set below $(1-\eta)\gamma_4/(2\gamma_3\gamma_7\pi_k^u)$, which is bounded below and away from zero by Lemma 3.8, in order to satisfy the Armijo condition (3.12) for some $\pi \in [\pi_k^l, \pi_k^u]$. □

We are now ready to present the main result of this section.

THEOREM 3.11 Algorithm 3.1 yields

$$\lim_{k\to\infty} \left\| \begin{bmatrix} g_k + A_k^{\mathrm{T}}\lambda_k \\ c_k \end{bmatrix} \right\| = 0.$$

Proof. By Corollary 3.9, the algorithm eventually computes, during a certain iteration $k^* \geqslant 0$, a finite value π^* beyond which the value of the parameter π^l will never be increased. This means that for all sufficiently large k, the Armijo condition (3.12) is satisfied for $\pi^l = \pi^*$ or else π^l would be increased (see the second-to-last line of Algorithm 3.1). From Lemmas 3.7 and 3.10, we then have that for all $k \geqslant k^*$

$$\phi_{\pi^*}(x_k) - \phi_{\pi^*}(x_k + \alpha_k d_k) \geqslant \gamma_8\Theta_k$$

for some $\gamma_8 > 0$. Therefore, (3.17) implies

$$\begin{aligned} \phi_{\pi^*}(x_{k^*}) - \phi_{\pi^*}(x_k) &= \sum_{j=k^*}^{k-1} (\phi_{\pi^*}(x_j) - \phi_{\pi^*}(x_{j+1})) \\ &\geqslant \gamma_8 \sum_{j=k^*}^{k-1} \Theta_j \\ &\geqslant \frac{\gamma_8}{2\gamma_3} \sum_{j=k^*}^{k-1} (\|d_j\|^2 + \|c_j\|) \end{aligned}$$

and so

$$\lim_{k\to\infty} \|d_k\| = 0 \quad \text{and} \quad \lim_{k\to\infty} \|c_k\| = 0 \tag{3.22}$$

follow from the fact that Assumption 3.2(a) implies ϕ_{π^*} is bounded below. Finally, the first block equation of (3.4), Assumption 3.2(c) and Lemma 3.3 imply

$$\begin{aligned} \|g_{k+1} + A_{k+1}^{\mathrm{T}}\lambda_{k+1}\| &= \|g_k + A_k^{\mathrm{T}}\lambda_{k+1} + (g_{k+1} - g_k) + (A_{k+1} - A_k)^{\mathrm{T}}\lambda_{k+1}\| \\ &= \|(1-\alpha_k)(g_k + A_k^{\mathrm{T}}\lambda_k) - \alpha_k W_k d_k + (g_{k+1} - g_k) + (A_{k+1} - A_k)^{\mathrm{T}}\lambda_{k+1}\| \\ &\leqslant (1-\alpha_k)\|g_k + A_k^{\mathrm{T}}\lambda_k\| + \mathrm{O}(\|d_k\|) \end{aligned}$$

and so

$$\lim_{k\to\infty} \|g_k + A_k^{\mathrm{T}}\lambda_k\| = 0$$

follows from (3.22), the fact that $\alpha_k \leqslant 1$ and Lemma 3.10. □

4. Numerical results

In this section, we present numerical results for a particular implementation of Algorithm 3.1 incorporated into the KNITRO-Direct algorithm in the KNITRO 5.0 software package; see Waltz & Plantenga (2006) for details. We tested the code using a set of 85 equality-constrained problems from the CUTEr (see Bongartz *et al.*, 1995, and Gould *et al.*, 2003) and COPS (see Dolan *et al.*, 2004) collections. From these sets, we chose problems for which AMPL models were readily available. The default KNITRO-Direct algorithm may revert to a trust region iteration to handle negative curvature and to ensure global convergence. In our tests, we enabled internal options to prevent this from occurring. Instead, the algorithm modifies W_k if necessary to ensure that the resulting matrix is positive definite on the null space of A_k—to ensure that our implementation performs as a pure line search algorithm.

As the globalization strategy described in this paper incurs little computational cost and is designed to promote long steps for fast convergence, we propose that the numbers of iterations and function evaluations required to find a solution are appropriate measures for comparison with other methods. We compare the results of an algorithm using the default penalty function approach in KNITRO-Direct, call it pi_default, with the results using a flexible penalty function. The penalty parameter update strategy in KNITRO-Direct corresponds to the case when (3.10) is replaced by $\pi_{k+1}^l \leftarrow \pi_k^u$. For pi_default and the algorithm with a flexible penalty function, we initialize π and π^l, respectively, to 10^{-8}. We consider the four initial values 1, 10, 100 and 1000 for π^u, which correspond to the algorithms we refer to as pi_flex_1, pi_flex_10, pi_flex_100 and pi_flex_1000, respectively. Table 1 contains a complete listing of the input parameters for our implementation of Algorithm 3.1.

The results for the five algorithms are summarized in Figs 8 and 9 in terms of logarithmic performance profiles, as described in Dolan & Moré (2002). Here, the leftmost values indicate the proportion of times each algorithm solves a given problem using the least value of the given measure; i.e. number of iterations or of function evaluations. The values fail to add to 1 as ties are present. The rightmost function values illustrate the robustness of each approach; i.e. the percentage of times that a given problem is solved.

The results are encouraging. An algorithm with a flexible penalty function approach often not only requires slightly fewer iterations to find a solution but also a considerable amount of savings is often experienced in terms of function evaluations. This can be understood as the line search procedure generally

TABLE 1 *Input values for Algorithm* 3.1

Parameter	Value
π_0^l	10^{-8}
π_{-1}^u	{1, 10, 100, 1000}
ε	10^{-4}
ε^l	10^{-4}
η	10^{-8}
σ	10^{-1}

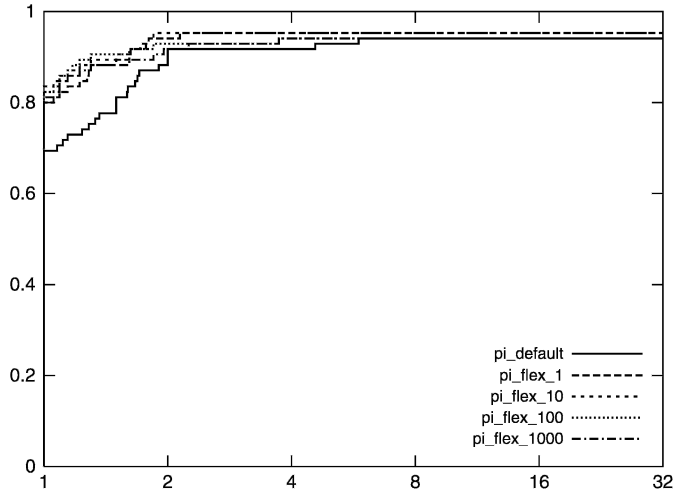

FIG. 8. Performance profile for iterations.

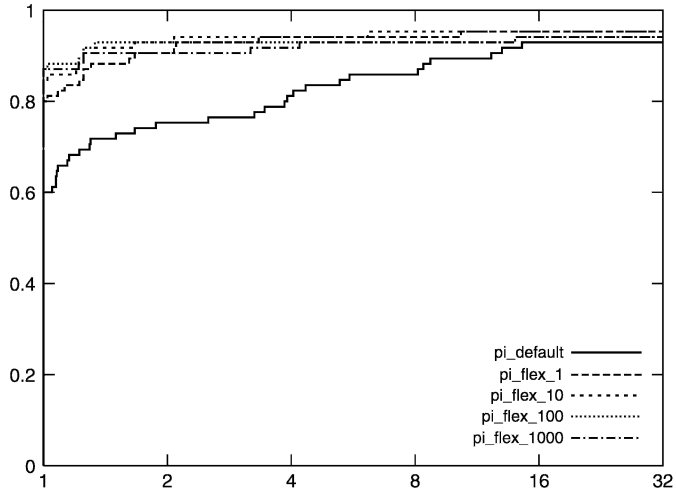

FIG. 9. Performance profile for function evaluations.

has to perform fewer backtracks for a given step, leading to longer steps and a higher percentage of unit step-lengths (i.e. full Newton steps). We also observe that the plots for pi_flex_1, pi_flex_10, pi_flex_100 and pi_flex_1000 are nearly indistinguishable throughout much of Figs 8 and 9. This suggests that the initial value for π^u is inconsequential compared to the effect that separate updating strategies for π^l and π^u have on the practical performance of the approach.

5. Final remarks

In this paper, we have proposed and analysed a new globalization strategy for equality-constrained optimization problems. Our flexible penalty function not only allows for relatively unrestricted movement

during early iterations but also automatically tightens itself to forcefully guide convergence when necessary, thus manipulating the search appropriately throughout a run of the algorithm. An example of a particular implementation of the mechanism was presented in the context of a line search SQP method, after which the global behaviour was analysed and successful numerical results were outlined.

We close by describing how the ideas of this paper might be extended to generally constrained problems of the form

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \ f(x) \\ &\text{s.t. } c^E(x) = 0, \\ &\quad c^I(x) \leqslant 0, \end{aligned} \tag{5.1}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$, $c^E\colon \mathbb{R}^n \to \mathbb{R}^{t^E}$ and $c^I\colon \mathbb{R}^n \to \mathbb{R}^{t^I}$ are smooth functions. One of the leading classes of methods for solving problem (5.1) are interior-point approaches. Some algorithms of this type begin by introducing a log-barrier term with parameter $\mu > 0$ for the inequalities into the objective to form the perturbed problem

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \ f(x) - \mu \sum_{i\in I} \ln s^i \\ &\text{s.t. } c^E(x) = 0, \\ &\quad c^I(x) + s = 0. \end{aligned} \tag{5.2}$$

A solution for problem (5.1) is then found via the (approximate) solution of a sequence of problems of the form (5.2) for $\mu \to 0$, where throughout the process the vector of slack variables $s = (s^1, \ldots, s^{t^I}) \in \mathbb{R}^{t^I}$ is forced to be positive. Thus, for each given $\mu > 0$, we can define the flexible penalty function associated with the barrier subproblem (5.2) as

$$\varphi_\pi(x) \triangleq f(x) - \mu \sum_{i\in I} \ln s^i + \pi \left\| \begin{bmatrix} c^E(x) \\ c^I(x) + s \end{bmatrix} \right\|, \quad \pi \in [\pi^l, \pi^u],$$

where $0 \leqslant \pi^l \leqslant \pi^u$, and a line search algorithm similar to Algorithm 3.1 can be applied. (The discussion here refers to a generic algorithm; to obtain practical methods with global convergence guarantees, various safeguards or modifications must be added. One such modification is the penalty function regularization described in Chen & Goldfarb (2006).)

A similar approach can be used in a trust region algorithm. Here, a step d_k from x_k is typically accepted if and only if the actual reduction in a penalty function ϕ_π, defined by

$$\phi\text{red}_\pi(d_k) \triangleq \phi_\pi(x_k) - \phi_\pi(x_k + d_k),$$

is large with respect to the reduction obtained in a model such as m_π (see Section 3). This condition can be written as

$$\frac{\phi\text{red}_\pi(d_k)}{m\text{red}_\pi(d_k)} \geqslant \eta$$

for some $0 < \eta < 1$, where it should be observed that we may now have $\|c_k + A_k d_k\| > 0$. Rather than restricting the step acceptance criteria to this inequality with a fixed $\pi > 0$ during each iteration k, we

claim that an effect similar to that expressed in this paper can be achieved if instead a step is considered acceptable if

$$\frac{\phi\text{red}_{\pi_k^l}(d_k)}{m\text{red}_{\pi_k^m}(d_k)} \geqslant \eta \quad \text{or} \quad \frac{\phi\text{red}_{\pi_k^u}(d_k)}{m\text{red}_{\pi_k^m}(d_k)} \geqslant \eta,$$

where $[\pi_k^l, \pi_k^u]$ is a prescribed interval and $\pi_k^m \in [\pi_k^l, \pi_k^u]$ is chosen carefully so that $m\text{red}_{\pi_k^m}(d_k)$ is sufficiently positive. All of the quantities π_k^l, π_k^u and π_k^m can be defined and updated in a manner similar to that described in this paper.

The previous discussion outlines ways in which our flexible penalty function can be employed in the context of constrained optimization. We note, however, that in order to obtain practical algorithms with global convergence guarantees, various algorithmic components must be added to the methods described above.

Acknowledgements

The authors are grateful to Richard Byrd and Richard Waltz for a number of productive discussions on this work. They also acknowledge a useful discussion with Sven Leyffer, Philippe Toint and Andreas Wächter about filter mechanisms and penalty functions.

Funding

Department of Energy (DE-FG02-87ER25047-A004 to F.E.C.); National Science Foundation (CCF-0514772) and Intel Corporation to J.N.

References

BONGARTZ, I., CONN, A. R., GOULD, N. I. M. & TOINT, P. L. (1995) CUTE: constrained and unconstrained testing environment. *ACM Trans. Math. Softw.*, **21**, 123–160.

BYRD, R. H., CURTIS, F. E. & NOCEDAL, J. (2008) An inexact SQP methods for equality constrained optimization. *SIAM J. Optim.* (to appear).

BYRD, R. H., HRIBAR, M. E. & NOCEDAL, J. (1999) An interior point algorithm for large scale nonlinear programming. *SIAM J. Optim.*, **9**, 877–900.

CHEN, L. & GOLDFARB, D. (2006) Interior-point l_2 penalty methods for nonlinear programming with strong global convergence properties. *Math. Program.*, **108**, 1–36.

CHIN, C. M. & FLETCHER, R. (2003) On the global convergence of an SLP-filter algorithm that takes EQP steps. *Math. Program.*, **91**, 161–177.

DOLAN, E. D. & MORÉ, J. J. (2002) Benchmarking optimization software with performance profiles. *Math. Program. Ser. A*, **91**, 201–213.

DOLAN, E. D., MORÉ, J. J. & MUNSON, T. S. (2004) Benchmarking optimization software with COPS 3.0. *Technical Report ANL/MCS-TM-273*. Argonne, IL, USA: Argonne National Laboratory.

EL-HALLABI, M. (1999) A hybrid algorithm for nonlinear equality constrained optimization problems: global and local convergence theory. *Technical Report TR4-99*. Rabat, Morocco: Mathematics and Computer Science Department, Institut National des Postes et Télécommunications.

FLETCHER, R. & LEYFFER, S. (2002) Nonlinear programming without a penalty function. *Math. Program. Ser. A*, **91**, 239–269.

FLETCHER, R., LEYFFER, S. & TOINT, P. L. (1998) On the global convergence of an SLP-filter algorithm. *Technical Report 98/13*. Namur, Belgium: Department of Mathematics, University of Namur.

GONZAGA, C. C., KARAS, E. & VANTI, M. (2003) A globally convergent filter method for nonlinear programming. *SIAM J. Optim.*, **14**, 646–669.

GOULD, N. I. M., ORBAN, D. & TOINT, P. L. (2003) CUTEr and sifdec: a constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Softw.*, **29**, 373–394.

HAN, S. P. (1977) A globally convergent method for nonlinear programming. *J. Optim. Theory Appl.*, **22**, 297–309.

LI, D. Q. (2006) A new SQP-filter method for solving nonlinear programming problems. *J. Comput. Math.*, **24**, 609–634.

NOCEDAL, J. & WRIGHT, S. J. (2006) *Numerical Optimization*, 2nd edn. Springer Series in Operations Research. T. V. Mikosch, S. I. Resnick & S. M. Robinson eds). New York: Springer.

OMOJOKUN, E. O. (1989) Trust region algorithms for optimization with nonlinear equality and inequality constraints. *Ph.D. Thesis*, University of Colorado, Boulder, CO, USA.

POWELL, M. J. D. (1978) A fast algorithm for nonlinearly constrained optimization calculations. *Numerical Analysis, Dundee 1977* (G. A. Watson ed.). Lecture Notes in Mathematics, vol. 630. Berlin: Springer, pp. 144–157.

POWELL, M. J. D. (1983) Variable metric methods for constrained optimization. *Mathematical Programming: The State of the Art* (A. Bachem, M. Grötschel & B. Korte eds). Bonn: Springer, pp. 288–311.

WÄCHTER, A. & BIEGLER, L. T. (2005a) Line search filter methods for nonlinear programming: motivation and global convergence. *SIAM J. Optim.*, **16**, 1–31.

WÄCHTER, A. & BIEGLER, L. T. (2005b) Line search filter methods for nonlinear programming: local convergence. *SIAM J. Optim.*, **16**, 32–48.

WALTZ, R. A., MORALES, J. L., NOCEDAL, J. & ORBAN, D. (2006) An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Math. Program. Ser. A*, **107**, 391–408.

WALTZ, R. A. & PLANTENGA, T. (2006) *KNITRO User's Manual*. Evanston, IL: Ziena Optimization, Inc.