Adaptive Barrier Update Strategies for Nonlinear Interior Methods

Jorge Nocedal* Andreas Wächter† Richard A. Waltz*

January 12, 2006

Abstract

This paper considers strategies for selecting the barrier parameter at every iteration of an interior-point method for nonlinear programming. Numerical experiments suggest that adaptive choices, such as Mehrotra's probing procedure, outperform static strategies that hold the barrier parameter fixed until a barrier optimality test is satisfied. A new adaptive strategy is proposed based on the minimization of a *quality function*. The paper also proposes a globalization framework that ensures the convergence of adaptive interior methods. The barrier update strategies proposed in this paper are applicable to a wide class of interior methods and are tested in the two distinct algorithmic frameworks provided by the IPOPT and KNITRO software packages.

1 Introduction

In this paper we describe interior methods for nonlinear programming that update the barrier parameter adaptively, as the iteration progresses. The goal is to design algorithms that are scale invariant and efficient in practice, and that enjoy global convergence guarantees. The adaptive strategies studied in this paper allow the barrier parameter to increase or decrease at every iteration and provide an alternative to the so-called Fiacco-McCormick approach that fixes the barrier parameter until an approximate solution of the barrier problem is computed. Our motivation for this work stems from our belief that robust interior methods for nonlinear programming must be able to react swiftly to changes of scale in the problem and to correct overly aggressive decreases in the barrier parameter.

*Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, 60208-3118, USA. These authors were supported by National Science Foundation grants CCR-0219438 and DMI-0422132, and Department of Energy grant DE-FG02-87ER25047-A004.

†Department of Mathematical Sciences, IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA.

Adaptive barrier update strategies are well established in interior methods for linear and convex quadratic programming. The most popular approach of this type is Mehrotra's predictor-corrector method [18]. It computes, at every iteration, a probing (affine scaling) step that determines a target value of the barrier parameter, and then takes a primal-dual step using this target value. A corrector step is added to better follow the trajectory to the solution. Mehrotra's method has proved to be very effective for linear and convex quadratic programming, but is not supported by global convergence guarantees. Indeed, as we show in Section 7 its reliability is heavily dependent upon an appropriate choice of the starting point.

When solving nonlinear nonconvex programming problems, much caution must be exercised to prevent the iteration from failing. Non-minimizing stationary points can attract the iteration, and aggressive decreases in the barrier parameter can lead to failure. Our numerical experience shows that the direct extension of Mehrotra's predictor-corrector method to nonlinear programming does not result in a robust method. As we discuss below, the main source of instability is the corrector step. Other adaptive barrier update strategies designed specifically for nonlinear programming include [1, 8, 11, 19, 20, 21].

In this paper we propose a new strategy for updating the barrier parameter that is effective in practice and is supported by a global convergence analysis. To show the generality of our technique, we implement it in the two different algorithmic contexts provided by the IPOPT [23] and KNITRO [24] software packages.

The global convergence properties of interior methods for nonlinear programming have recently received much attention [3, 8, 13, 14, 16, 19, 20, 22, 28]. Some of these studies focus on the effects of merit functions or filters, and on regularization techniques. With the exception of [8, 19, 20], however, these papers do not consider the numerical or theoretical properties of adaptive barrier update techniques.

Notation. For any vector z, we denote by Z the diagonal matrix whose diagonal entries are given by z. We let e denote the vector of ones, of appropriate dimension, that is, $e = (1, 1, \cdots, 1)^T$.

2 Primal-Dual Nonlinear Interior Methods

The problem under consideration will be written as

$$\min_x \quad f(x) \tag{2.1a}$$
$$\text{s.t.} \quad c(x) = 0 \tag{2.1b}$$
$$x \geq 0, \tag{2.1c}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable functions. For conciseness we will refer to interior-point methods for nonlinear programming as "nonlinear interior methods." A variety of these methods have been proposed in the last 10 years; they differ mainly in some aspects of the step computation and in the globalization scheme. Most of the nonlinear interior methods are related to the simple primal-dual iteration described next; our discussion of barrier parameter choices will be phrased in the context of this iteration.

We associate with the nonlinear program (2.1) the barrier problem

$$\min_x \quad \varphi_\mu(x) \equiv f(x) - \mu \sum_{i=1}^{n} \ln x_i \tag{2.2a}$$

$$\text{s.t.} \quad c(x) = 0, \tag{2.2b}$$

where $\mu > 0$ is the barrier parameter. As is well known, the KKT conditions of the barrier problem (2.2) can be written as

$$\nabla f(x) - A(x)^T y - z = 0 \tag{2.3a}$$

$$Xz - \mu e = 0 \tag{2.3b}$$

$$c(x) = 0, \tag{2.3c}$$

together with

$$x \geq 0, \qquad z \geq 0. \tag{2.4}$$

Here $A(x)$ denotes the Jacobian matrix of the constraint function $c(x)$.

Applying Newton's method to (2.3), in the variables (x, y, z), gives the *primal-dual* system

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & -A(x)^T & -I \\ Z & 0 & X \\ A(x) & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A(x)^T y - z \\ Xz - \mu e \\ c(x) \end{bmatrix}, \tag{2.5}$$

where $\mathcal{L}$ denotes the Lagrangian of the nonlinear program, that is,

$$\mathcal{L}(x, y, z) = f(x) - y^T c(x) - z^T x. \tag{2.6}$$

After the step $\Delta = (\Delta x, \Delta y, \Delta z)$ has been determined, we compute primal and dual steplengths, α_p and α_d, and define the new iterate (x^+, y^+, z^+) as

$$x^+ = x + \alpha_p \Delta x, \quad y^+ = y + \alpha_d \Delta y, \quad z^+ = z + \alpha_d \Delta z. \tag{2.7}$$

The steplengths are computed in two stages. First we compute

$$\alpha_x^{\max} = \max\{\alpha \in (0, 1] : x + \alpha \Delta x \geq (1 - \tau)x\} \tag{2.8a}$$

$$\alpha_z^{\max} = \max\{\alpha \in (0, 1] : z + \alpha \Delta z \geq (1 - \tau)z\}, \tag{2.8b}$$

with $\tau \in (0, 1)$ (e.g. $\tau = 0.995$). Next, we perform a backtracking line search to compute the final steplengths

$$\alpha_p \in (0, \alpha_x^{\max}], \qquad \alpha_d \in (0, \alpha_z^{\max}], \tag{2.9}$$

which provide sufficient decrease of a merit function or ensure acceptability by a filter.

The other major ingredient in this simple primal-dual iteration is the procedure for choosing the barrier parameter μ. Two types of barrier update strategies have been studied in the literature: adaptive and static. Adaptive strategies [8, 11, 20, 21] allow changes in the barrier parameter at every iteration, and often achieve scale invariance, but as already mentioned, they generally do not enjoy global convergence properties. (The analyses presented in [8, 20] provide certain convergence results to stationary points, but these methods do not explicitly aim to decrease the objective function—they only enforce reduction of a measure of stationarity.)

The most important static strategy is the so-called Fiacco-McCormick approach that fixes the barrier parameter until an approximate solution of the barrier problem is computed. It has been employed in various nonlinear interior algorithms [2, 4, 10, 12, 23, 25, 27] and has been implemented, for example, in the IPOPT and KNITRO software packages. The Fiacco-McCormick strategy provides a framework for establishing global convergence [3, 22], but suffers from important limitations. It can be very sensitive to the choice of the initial point, the initial value of the barrier parameter and the scaling of the problem, and it is often unable to recover when the iterates approach the boundary of the feasible region prematurely. The numerical experience with IPOPT and KNITRO reported below suggests that more dynamic update strategies are needed to improve the speed of nonlinear interior methods.

3 Choosing the Barrier Parameter

In this section we discuss two adaptive barrier strategies proposed in the literature and compare them numerically with the static Fiacco-McCormick approach. These numerical results motivate the discussion of the following sections.

Given an iterate (x, y, z), consider an interior method that computes primal-dual search directions by (2.5). The most common approach for choosing the barrier parameter μ is to make it proportional to the current complementarity value, that is,

$$\mu = \sigma \frac{x^T z}{n}, \tag{3.1}$$

where $\sigma > 0$ is a *centering parameter* and n denotes the number of variables. Mehrotra's predictor-corrector (MPC) method [18] for linear programming determines the value of σ using a preliminary step computation (an affine scaling step). We now describe a direct extension of Mehrotra's strategy to the nonlinear programming case.

First, we calculate an affine scaling step

$$(\Delta x^{\text{aff}}, \Delta y^{\text{aff}}, \Delta z^{\text{aff}}) \tag{3.2}$$

by setting $\mu = 0$ in (2.5), that is,

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & -A(x)^T & -I \\ Z & 0 & X \\ A(x) & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x^{\text{aff}} \\ \Delta y^{\text{aff}} \\ \Delta z^{\text{aff}} \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A(x)^T y - z \\ Xz \\ c(x) \end{bmatrix}. \tag{3.3}$$

We then compute α_x^{aff} and α_z^{aff} to be the longest steplengths in $(0, 1]$ that can be taken along the direction (3.2) before violating the non-negativity conditions $(x, z) \geq 0$. Explicit formulae for these values are given by (2.8) with $\tau = 1$.

Next, we define μ^{aff} to be the value of complementarity that would be obtained by a full step to the boundary, that is,

$$\mu^{\text{aff}} = (x + \alpha_x^{\text{aff}} \Delta x^{\text{aff}})^T (z + \alpha_z^{\text{aff}} \Delta z^{\text{aff}})/n, \tag{3.4}$$

and set the centering parameter to be

$$\sigma = \left(\frac{\mu^{\text{aff}}}{x^T z/n} \right)^3. \tag{3.5}$$

This heuristic choice of σ is based on experimentation with linear programming problems, and has proved to be effective for convex quadratic programming as well. Note that when good progress is made along the affine scaling direction, we have $\mu^{\text{aff}} \ll x^T z/n$, so the σ obtained from this formula is small. In other cases, σ may be chosen to be greater than 1.

Mehrotra's algorithm also computes a *corrector* step, but we take the view that the corrector is not part of the selection of the barrier parameter, and is simply a mechanism for improving the quality of the step. In Section 7 we study the complete MPC algorithm including the corrector step.

Other adaptive procedures of the form (3.1) have been proposed specifically for nonlinear interior methods [8, 11, 20, 21]. The strategy employed in the LOQO software package [21] is particularly noteworthy because of its success in practice. It defines σ as

$$\sigma = 0.1 \min \left(0.05 \frac{1-\xi}{\xi}, 2 \right)^3, \quad \text{where} \quad \xi = \frac{\min_i \{x_i z_i\}}{x^T z/n}. \tag{3.6}$$

Note that ξ measures the deviation of the smallest complementarity product $x_i z_i$ from the average. When $\xi = 1$ (all individual products are equal to their average) we have that $\sigma = 0$ and the algorithm takes an aggressive step. The rule (3.6) always chooses $\sigma \leq 0.8$, so that even though the value of μ may increase from one iteration to the next, it will never be chosen to be larger than the current complementarity value $x^T z/n$.

Let us compare the numerical performance of these two adaptive strategies with the static Fiacco-McCormick approach. To better measure the impact of the choice of barrier parameter, that is, to try to distinguish it from other algorithmic features, we use both the IPOPT and KNITRO software packages in our tests. These codes implement significantly different variations of the simple primal-dual iteration (2.5). IPOPT [23] was implemented for these tests without a line search; only the fraction to the boundary rule (2.8) was imposed on the primal and dual steplengths. For KNITRO we used the Interior/Direct option (we will refer to this version as KNITRO-DIRECT henceforth), which invokes a line search approach that is occasionally safeguarded by a trust region iteration (for example, when negative curvature is encountered) [25].

The barrier parameter strategies tested in our experiments are as follows:

- LOQO *rule*. The barrier parameter is chosen by (3.1) and (3.6).
- *Mehrotra probing*. At every iteration, the barrier parameter μ is given by (3.1) and (3.5). Since this requires the computation of the affine scaling step (3.2), this strategy

is more expensive than the LOQO rule. For KNITRO-DIRECT, in the iterations in which the safeguarding trust region algorithm is invoked the barrier parameter is computed by the LOQO rule instead of Mehrotra probing. This is done because Mehrotra probing is expensive to implement in the trust region algorithm, which uses a conjugate gradient iteration.

- *MPC.* The complete Mehrotra predictor-corrector algorithm as described in Section 7. As in the Mehrotra probing rule, when KNITRO-DIRECT falls back on the safeguarded trust region algorithm, the barrier parameter is computed using the LOQO rule for efficiency, and no corrector step is used.

- *Monotone.* (Also known as the Fiacco-McCormick approach.) The barrier parameter is fixed, and a series of primal-dual steps is computed, until the optimality conditions for the barrier problem are satisfied to some accuracy. At this point the barrier parameter is decreased. IPOPT and KNITRO implement somewhat different variations of this monotone approach; see [23, 25] for details about the initial value of μ, the rule for decreasing μ, and the form of the barrier stop tests.

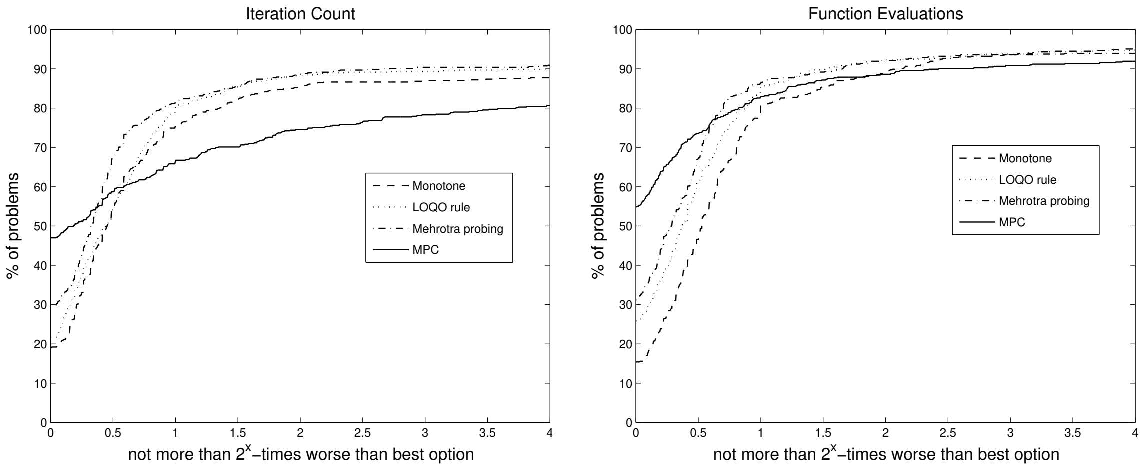
For the numerical comparison, we select all the nonlinear programming problems in the CUTEr test set that contain at least one general inequality or bound constraint. We exclude those problems that seem infeasible, unbounded, or are given with initial points at which the model functions cannot be evaluated; see [23]. This gives a total of 599 problems. Figure 1(a) reports the total number of iterations for IPOPT, and Figure 1(b) shows the total number of function evaluations for KNITRO, both comparing the performance of the four barrier strategies. Measuring iterations and function evaluations give similar profiles and we provide both for greater variety. All the plots in the paper use the logarithmic performance profiles proposed by Dolan and Moré [7]. To account for the fact that different local solutions might be computed, problems with significantly different final objective function values for successful runs were excluded.

Note that Mehrotra probing appears to be the most successful in terms of iterations or function evaluations. The complete MPC algorithm is very fast on some problems, but is not sufficiently robust. The latter is clear in Figure 1(a) which reports the performance of the pure MPC strategy implemented in IPOPT; it is less evident in Figure 1(b), but this is due to the safeguarding approach implemented in KNITRO-DIRECT mentioned above. The reason for the lack of robustness of the MPC strategy will be discussed in Section 7.

Motivated by these results, we give further attention to adaptive choices of the barrier parameter.

4 Quality Functions

The Mehrotra and LOQO rules rely on the heuristic parameters (3.5) and (3.6). We now consider an approach in which μ is selected using a clear-cut objective, formulated in terms of a *quality function* to be minimized. As before, we assume that $\mu = \sigma \frac{x^T z}{n}$, where the centering parameter $\sigma \geq 0$ is to be determined, and define $\Delta(\sigma)$ to be the solution of the



(a) IPOPT (b) KNITRO

Figure 1: Results for four barrier parameter updating strategies.

primal-dual equations (2.5) as a function of σ. We also let $\alpha_x^{\max}(\sigma), \alpha_z^{\max}(\sigma)$ denote the steplengths satisfying the fraction to the boundary rule (2.8) for the step $\Delta = \Delta(\sigma)$, and we define the *probing points*

$$x(\sigma) = x + \alpha_x^{\max}(\sigma)\Delta x(\sigma),$$

$$y(\sigma) = y + \alpha_z^{\max}(\sigma)\Delta y(\sigma), \quad z(\sigma) = z + \alpha_z^{\max}(\sigma)\Delta z(\sigma).$$

Our goal is to choose the value of σ that provides significant improvement toward the solution of the nonlinear program (2.1). For example, we could choose σ so as to minimize the following nonlinear quality function based on the KKT error:

$$\begin{aligned} q_N(\sigma) &= \|\nabla f(x(\sigma)) - A(x(\sigma))^T y(\sigma) - z(\sigma)\|^2 + \|c(x(\sigma))\|^2 \\ &\quad + \|Z(\sigma)X(\sigma)e\|^2. \end{aligned} \tag{4.1}$$

The evaluation of q_N is, however, expensive since it requires the evaluation of the problem functions and derivatives for every value of σ. We can avoid this expense by using a *linear quality function.* If we assume that f and c are linear functions, we have that (4.1) can be expressed as

$$\begin{aligned} q_L(\sigma) &= (1 - \alpha_z^{\max}(\sigma))^2 \|\nabla f(x) - A(x)^T y - z\|^2 + (1 - \alpha_x^{\max}(\sigma))^2 \|c(x)\|^2 \\ &\quad + \|(X + \alpha_x^{\max}(\sigma)\Delta X(\sigma))(Z + \alpha_z^{\max}(\sigma)\Delta Z(\sigma))e\|^2, \end{aligned} \tag{4.2}$$

where $\Delta X(\sigma)$ is the diagonal matrix with $\Delta x(\sigma)$ on the diagonal, and similarly for $\Delta Z(\sigma)$.

Note that $\Delta(\sigma) = \Delta(0) + \sigma(\Delta(1) - \Delta(0))$. Therefore, $\Delta(\sigma)$ can be computed easily for any value of σ once the linear system (2.5) has been solved twice to obtain $\Delta(0)$ and $\Delta(1)$.

Having computed $\Delta(\sigma)$, the dominant cost in the evaluation of q_L lies in the computation of the maximal steplengths $\alpha_x^{\max}(\sigma), \alpha_z^{\max}(\sigma)$ and the last term in (4.2), which requires a few vector operations.

For linear programming problems, the function q_L measures the KKT error exactly at the probing points $(x(\sigma), y(\sigma), z(\sigma))$. We have defined the quality function q_L using squared norms to severely penalize any large components in the KKT error. Note that $q_L(\sigma)$ is not a convex function of σ, in general. Moreover, due to the complicated dependence of the steplengths $\alpha_x^{\max}(\sigma), \alpha_z^{\max}(\sigma)$ on the parameter σ, it does not seem possible to obtain an analytic expression of the minimizers of q_L. Nevertheless, we have observed that, in practice, this function is usually unimodal.

Therefore, we implement a one-dimensional search scheme to compute an approximate minimizer of q_L. It uses a golden bisection procedure (see, e.g. [17]), and ignores the fact that q_L may not necessarily be unimodal. We first choose $\sigma^{\min}$ and $\sigma^{\max}$, which define a minimum and maximum limit on the σ value, and define the two intervals $[\sigma^{\min}, 1]$ and $[1, \sigma^{\max}]$. In our implementation, the value $\sigma^{\min} = \max(\gamma, \mu^{\min} n / x^T z)$, where $\mu^{\min}$ ($= 10^{-9}$ in our implementation defines a minimal permissible value of the barrier parameter, γ is some small number (say, 10^{-6} or 10^{-8})) and $\sigma^{\max} = 1000$. We first evaluate the quality function for $\sigma = 1$ and for some σ value slightly less than 1 (say 0.99). If $q_L(0.99) \leq q_L(1)$, then we perform our golden bisection procedure in the interval $[\sigma^{\min}, 1]$, otherwise we search in the interval $[1, \sigma^{\max}]$. (It is important that σ be allowed to take on values greater than one so that the algorithm can recover from overly aggressive reductions of the barrier parameter.) Our bisection procedure terminates if either 12 evaluations of the quality functions are performed, or if the search interval $[a, b]$ becomes smaller than $b \times 10^{-2}$.

More implementation details are given in Section 6. Before presenting our numerical results with the quality function, we study how to guarantee the global convergence of nonlinear interior methods that choose the barrier parameter adaptively.

5 A Globalization Method

The adaptive strategies described in Section 3 can be seen from the numerical results in that section to be quite robust. (We show in the next section this is also the case with the quality function approach.) Yet, since the barrier parameter is allowed to change at every iteration in these algorithms, there is no mechanism that enforces global convergence of the iterates. In contrast, the monotone barrier strategy employed in the Fiacco-McCormick approach allows us to establish global convergence results by combining two mechanisms. First, the algorithms that minimize a given barrier problem (2.2) use a line search or trust region to enforce a decrease in a merit function (as in KNITRO) or to guarantee acceptability by a filter (as in IPOPT). This ensures that an optimality test for the barrier function is eventually satisfied to some tolerance ϵ. Second, by repeating this minimization process for decreasing values of μ and ϵ that converge to zero, one can establish global convergence results [3, 9] to stationary points of the nonlinear programming problem (2.1).

We now propose two globalization schemes that monitor the performance of the iterations in reference to a mechanism that enforces global convergence. As long as the adaptive primal-dual steps make sufficient progress towards the solution, the algorithm is free to choose a new value for the barrier parameter at every iteration. We call this the *free mode.* However, if the iteration fails to maintain progress, then the algorithm reverts to a *monotone mode*, in which a Fiacco-McCormick strategy is applied. Here, the value of the barrier parameter remains fixed, and a robust globalization technique (e.g., based on a merit function or a filter) is employed to ensure progress for the corresponding barrier problem. Once the barrier problem is approximately minimized, the barrier parameter is decreased. The monotone mode continues until an iterate is generated that makes sufficient progress for the original problem, at which point the free mode resumes.

There are various ways to measure whether steps in the free mode make sustained progress toward the solution of the nonlinear program (2.1). We have developed two mechanisms, one based on a measure of KKT error and the other using a filter based on the value of the objective (2.1a) and a measure of the constraint violation. Both aim to interfere with adaptive steps as little as possible so as not to slow down convergence.

5.1 Nonmontone Decrease of the KKT Error

In our first globalization scheme, we monitor the KKT error of the original nonlinear program,

$$\Phi(x, y, z) = \|\nabla f(x) - A(x)^T y - z\|^2 + \|c(x)\|^2 + \|ZXe\|^2. \tag{5.1}$$

We require that this measure be reduced by a factor of $\kappa \in (0, 1)$ over at most a fixed number $l^{\max}$ of iterations, when the algorithm is in the free mode.

KKT-Error Based Globalization Framework

Given (x_0, y_0, z_0) with $(x_0, z_0) > 0$, a constant $\kappa \in (0, 1)$ and an integer $l^{\max} \geq 0$.
Set $k \leftarrow 0$.
Repeat
  Choose a target value of the barrier parameter μ_k, based on any rule.
  Compute the primal dual search direction Δ from (2.5).
  Determine step sizes $\alpha_p \in (0, \alpha_x^{\max}]$ and $\alpha_d \in (0, \alpha_z^{\max}]$.
  Compute the new trial iterate $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$ from (2.7).
  Compute the KKT error $\tilde{\Phi}_{k+1} \equiv \Phi(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$.
  Set $M_k = \max\{\Phi_{k-l}, \Phi_{k-l+1}, \ldots, \Phi_k\}$ with $l = \min\{k, l^{\max}\}$.
  If $\tilde{\Phi}_{k+1} \leq \kappa M_k$
    Accept $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$ as the new iterate, and set $\Phi_{k+1} \leftarrow \tilde{\Phi}_{k+1}$.
    Set $k \leftarrow k + 1$ and return to the beginning of the loop.
  else
    Start Monotone Mode:
    Starting from $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$, and for an initial value $\bar{\mu}$, solve a sequence of barrier problems with a monotonically decreasing sequence of barrier parameters to obtain a new iterate

$(x_{k+1}, y_{k+1}, z_{k+1})$ such that

$\Phi_{k+1} \equiv \Phi(x_{k+1}, y_{k+1}, z_{k+1}) \leq \kappa M_k$.

Set $k \leftarrow k+1$ and resume the free mode at the beginning of the loop.

end if

End (repeat).

Since this framework ensures that the optimality measure Φ is reduced by a factor of $\kappa < 1$ in at most every $l^{\max}$ iterations, it is clear that $\Phi_k \to 0$. Consequently, since the interior method maintains non-negative estimates for z_k and x_k, all limit points of the sequence of iterates satisfy the first order optimality conditions of the nonlinear program (2.1). An important issue when switching to the monotone mode is the initialization of the barrier parameter $\bar{\mu}$. This can be chosen, for example, to be some fraction of the current complementarity value.

In the monotone mode, it is not required to solve each barrier problem to the specified tolerance before checking whether the method can revert to the free mode. Instead, we compute the optimality error $\Phi(x, y, z)$ for all intermediate iterates in the monotone mode, and return to the free mode, as soon as $\Phi(x, y, z) \leq \kappa M_k$.

Note that also in the free mode we might want to choose steplengths α_p, α_d that are shorter than the maximal step sizes $\alpha_x^{\max}, \alpha_z^{\max}$. In our implementations, we perform a line search to enforce progress in a merit function or a filter, both of which are defined with respect to the barrier problem (2.2) corresponding to the current value μ_k. In this way, we force the algorithm to consider the objective function when determining a new trial, and not only the norm of the optimality conditions.

5.2 Two-Dimensional Filter

In the second globalization method, we make use of a filter that accepts trial points if they provide sufficient progress in terms of constraint violation $\theta(x) = \|c(x)\|$ *or* the objective function $f(x)$, compared to the previous iterates generated in the free mode. We let $\mathcal{F}_k \subseteq \{(f, \theta) \in \mathbb{R}^2 : \theta \geq 0\}$ denote the (f, θ) pairs that are not acceptable at the current iteration k. The concept of acceptabilty by the filter is made precise below.

Filter Based Globalization Framework

Given (x_0, y_0, z_0) with $(x_0, z_0) > 0$, and constants $\kappa_1, \kappa_2 > 0$; initialize the filter $\mathcal{F}_0 = \emptyset$.
Set $k \leftarrow 0$.

Repeat

Choose a target value of the barrier parameter μ_k, based on any rule.
Compute the primal dual search direction Δ from (2.5).
Determine step sizes $\alpha_p \in (0, \alpha_x^{\max}]$ and $\alpha_d \in (0, \alpha_z^{\max}]$.
Compute the new trial iterate $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$ from (2.7).
Compute the filter margin $\delta_k = \kappa_1 \min\{\kappa_2, \Phi(x_k, y_k, z_k)\}$

If $(f(\tilde{x}_{k+1}) + \delta_k, \|c(\tilde{x}_{k+1})\| + \delta_k) \notin \mathcal{F}_k$

Accept $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$ as the new iterate.

Update the filter $\mathcal{F}_{k+1} = \mathcal{F}_k \cup \{(f,\theta) : f \geq f(\tilde{x}_{k+1}) \text{ and } \theta \geq \|c(\tilde{x}_{k+1})\|\}$.
Set $k \leftarrow k+1$ and return to the beginning of the loop.

else

Start Monotone Mode:
Starting from $(\tilde{x}_{k+1}, \tilde{y}_{k+1}, \tilde{z}_{k+1})$, and for an initial value $\bar{\mu}$, solve a sequence of barrier problems with a monotonically decreasing sequence of barrier parameters to obtain a new iterate $(x_{k+1}, y_{k+1}, z_{k+1})$ such that

$$(f(x_{k+1}) + \delta_k, \|c(x_{k+1})\| + \delta_k) \notin \mathcal{F}_k.$$

Augment the filter:

$$\mathcal{F}_{k+1} = \mathcal{F}_k \cup \{(f,\theta) : f \geq f(x_{k+1}) \text{ and } \theta \geq \|c(x_{k+1})\|\}.$$

Set $k \leftarrow k+1$ and resume the free mode at the beginning of the loop.

end if

End (repeat).

Similar to the KKT-error based globalization scheme, the monotone mode is terminated as soon as an iterate is encountered that is acceptable to the filter. As in Section 5.1, we perform a line search to obtain the step sizes α_p and α_d in the free mode.

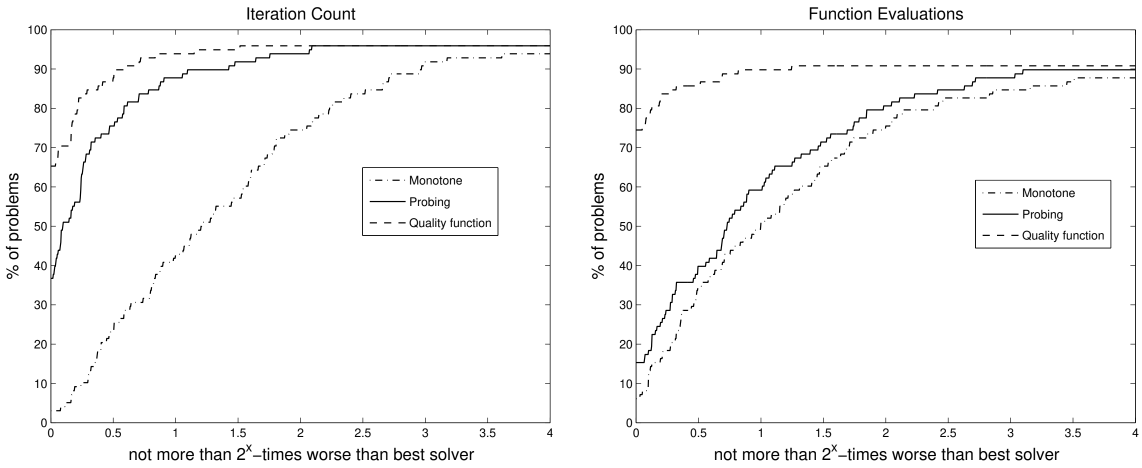
Under the assumption that $\{f(x_k)\}$ is bounded below, and that $\{c(x_k)\}$ is bounded, if there are infinitely many iterations in the free mode, we have that $\Phi(x_k, y_k, z_k)$ converges to zero. This can easily be shown by noting that the filter margin δ_k is defined in terms of $\Phi(x_k, y_k, z_k)$. On the other hand, if there are only finitely many free iterations, the algorithm eventually becomes identical with the Fiacco-McCormick algorithm. Therefore, the above scheme ensures global convergence of the overall method. In IPOPT the boundedness of $\{c(x_k)\}$ is guaranteed, since the regular filter in IPOPT enforces such an upper bound on the constraint violation.

We have tested the KKT and Filter globalization approaches using IPOPT and KNITRO, and found both to be effective in practice. For the sake of uniformity, in the next section we report results only for the Filter globalization approach. We set $\kappa_1 = 10^{-5}$ and $\kappa_2 = 1$ for these tests.

6 Numerical Results

We first discuss the choice of norms in the quality function (4.2) and in the optimality measure (5.1). (For consistency, we use the same norms and scaling factors in (4.2) and (5.1).) In IPOPT we use the 2-norm, and each of the three terms is divided by the number of elements in the vectors whose norms are being computed. In KNITRO, we choose the norm and scaling factors to be similar to the terms used in the KNITRO termination test: The first two terms in (4.2) and (5.1) use the infinity-norm, the complementarity term uses the 1-norm divided by n, and we scale these terms using the factors described in [25].

The tests involving IPOPT were run on a Dual-Pentium III, 1GHz machine running Linux. The KNITRO tests were run on a machine with an AMD Athlon XP 3200+ 2.2GHz processor running Linux. For both codes, the maximum number of iterations was set to



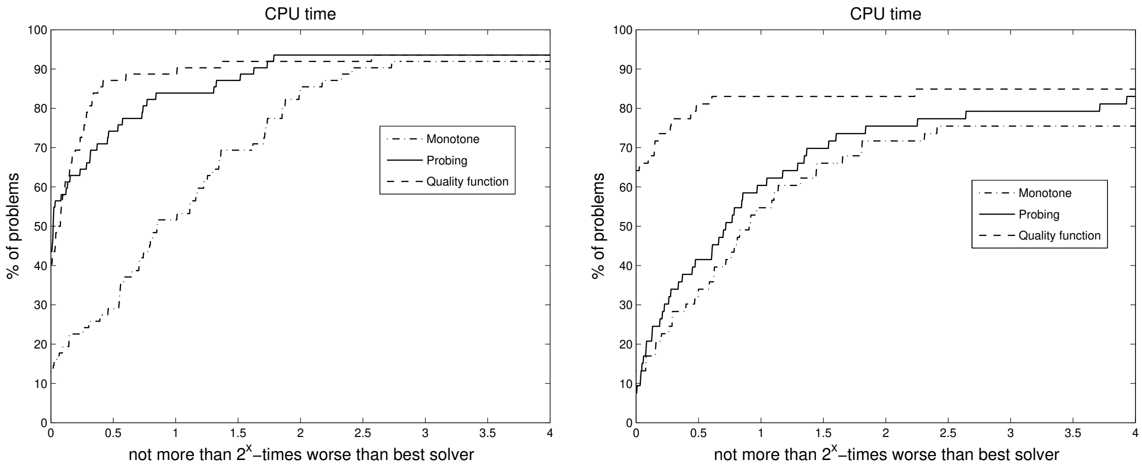
(a) IPOPT (b) KNITRO

Figure 2: Iteration and function evaluation comparison for the NETLIB test set.

3000 and the time limit was set to 1800 CPU seconds. The tests were run using the latest development versions of IPOPT and KNITRO as of October 2005.

The first results are for the linear programming problems in the NETLIB collection, as specified in the CUTEr test set [15]. No preprocessing was performed, and no initial point strategy was employed (i.e., the default starting point $x_0 = (0, \dots, 0)$ was used). Figure 2 compares the performance of the quality function approach, in terms of iteration and function evalution count, with two of the strategies described in Section 3, namely the monotone method and the Mehrotra probing heuristic. Since we are primarily interested in methods with globally convergent frameworks we use the filter based globalization framework described in Section 5 for both the Mehrotra probing approach and the quality function approach. The monotone approach is globally convergent on its own. Even though our focus is on nonlinear optimization, linear programming problems are of interest since they allow us to assess the effectiveness of the quality function in a context in which they exactly predict the KKT error. It is apparent from Figure 2 that the quality function approach is very effective on the NETLIB test set. We note, however, that the quality function approach requires extra work, and hence the discrepancy in terms of CPU time is less pronounced, see Figure 3.

The performance of the three barrier update strategies on nonlinear programming problems with at least one inequality or bound constraint from the CUTEr collection is reported in Figure 4. The quality function approach again performs significantly better than the monotone method; it also outperforms the Mehrotra probing strategy, which had given the best results in the experiments reported in Section 3, in terms of function evaluations. The differences are less clearly pronounced when comparing CPU performance, see Figure 5. These observations are of practical importance because the monotone method is currently



(a) IPOPT

(b) KNITRO

Figure 3: CPU time comparison for the NETLIB test set.

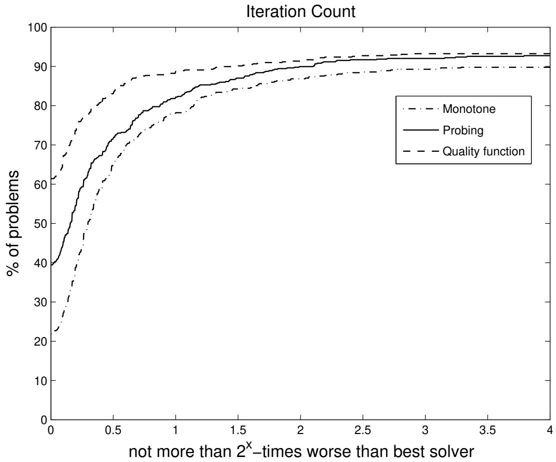
the default strategy in both IPOPT and KNITRO; our tests suggest that significant improvements can be expected by using adaptive strategies, particularly in applications where the function evaluation is the dominant cost. We believe that the quality function approach shows potential for future advances.

7 Corrector Steps

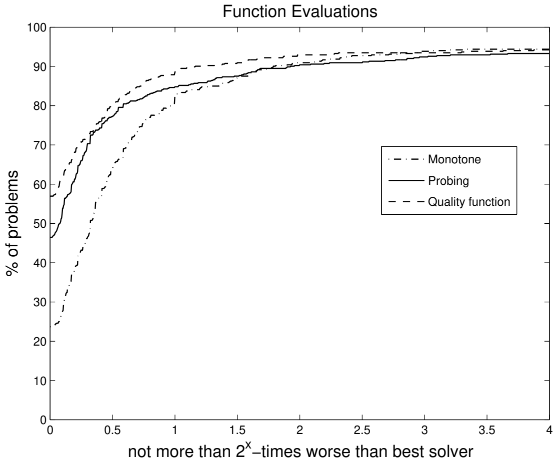
The numerical results of Section 3 indicate that, when solving nonlinear problems, including the corrector step in Mehrotra's method (the MPC method) is often not beneficial. This is in stark contrast with the experience in linear programming and convex quadratic programming, where the corrector step is known to accelerate the interior-point iteration without degrading its robustness. In this section we study the effect of the corrector step and find that it can also be harmful in the linear programming and quadratic programming cases *if* an initial point strategy is not used. These observations are relevant because in nonlinear programming it is much more difficult to find a good starting point.

Let us begin by considering the linear programming case. There are several ways of viewing the MPC method in this context. One is to consider the step computation as taking place in three stages (see, e.g., [26]). First, the algorithm computes the affine scaling step (3.2) and uses it to determine the target value of the barrier parameter $\mu = \sigma\frac{x^T z}{n}$, where σ is given by (3.5). Next, the algorithm computes a primal-dual step, say $\Delta^{\rm pd}$, from (2.5) using that value of μ. Finally, a corrector step $\Delta^{\rm corr}$ is computed by solving (2.5) with the right hand side given by

$$-(0, \Delta X^{\rm aff}\Delta Z^{\rm aff} e, 0)^T, \tag{7.1}$$

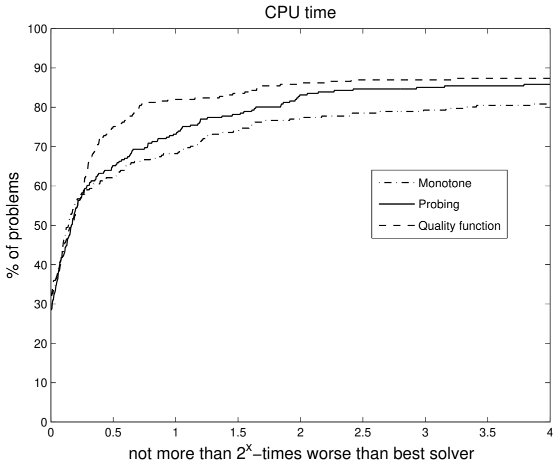


(a) IPOPT

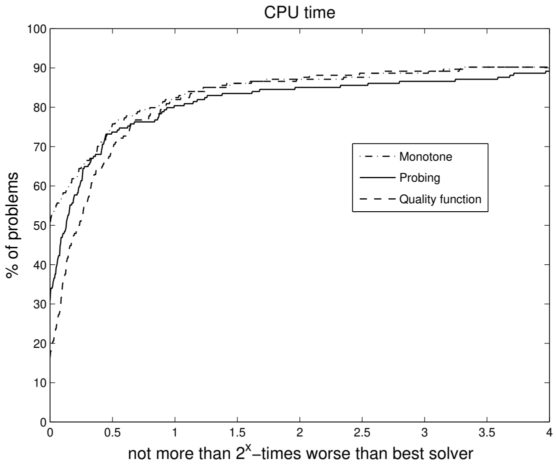


(b) KNITRO

Figure 4: Iteration and function evaluation comparison for CUTEr test set.



(a) IPOPT



(b) KNITRO

Figure 5: CPU time comparison for the CUTEr test set.

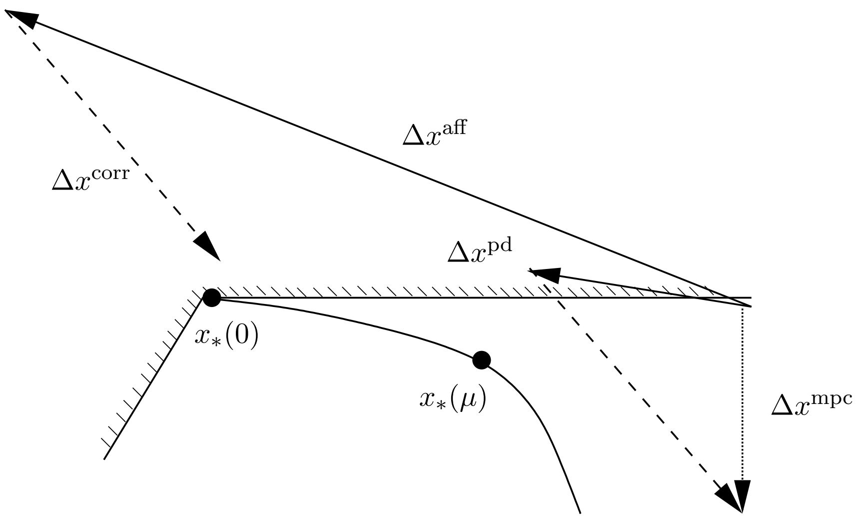


Figure 6: An unfavorable corrector step.

where ΔX^{aff} is the diagonal matrix with diagonal entries given by Δx^{aff}, and similarly for ΔZ^{aff}. The complete MPC step is the sum of the primal-dual and corrector steps. We can compute it by adding the right hand sides and solving the following system:

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & -A^T(x) & -I \\ Z & 0 & X \\ A(x) & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x^{\mathrm{mpc}} \\ \Delta y^{\mathrm{mpc}} \\ \Delta z^{\mathrm{mpc}} \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A^T(x)y - z \\ Xz - \mu e + \Delta X^{\mathrm{aff}} \Delta Z^{\mathrm{aff}} e \\ c(x) \end{bmatrix}. \tag{7.2}$$

The new iterate (x^+, y^+, z^+) of the MPC method is given by (2.7)-(2.8) with $\Delta = (\Delta x^{\mathrm{mpc}}, \Delta y^{\mathrm{mpc}}, \Delta z^{\mathrm{mpc}})$.

Alternative views of the MPC method are possible by the linearity of the step computation: We can group the right hand side in (7.2) in different ways and thereby interpret the step as the sum of different components. Yet all these views point out the following inconsistency in the MPC approach.

In the linear programming case, primal and dual feasibility are linear functions and hence vanish at the full affine scaling point, defined by

$$(x, y, z) + (\Delta x^{\mathrm{aff}}, \Delta y^{\mathrm{aff}}, \Delta z^{\mathrm{aff}}). \tag{7.3}$$

The complementarity term takes on the value

$$(X + \Delta X^{\mathrm{aff}})(Z + \Delta Z^{\mathrm{aff}}) = \Delta X^{\mathrm{aff}} \Delta Z^{\mathrm{aff}}.$$

Therefore the value of the right hand side vector in (2.5) at the full affine scaling step (7.3) is given by (7.1). Thus the corrector step can be viewed as a modified Newton step taken from the point (7.3) and using the primal-dual matrix evaluated at the current iterate (x, y, z).

Iter	Primal Obj	Dual Obj	PriInf	DualInf	$\alpha_x^{\max}$	$\alpha_z^{\max}$	$\log(\frac{x^Tz}{n})$	$\|\Delta^{\text{aff}}\|$	$\|\Delta^{\text{mpc}}\|$
0	9.0515e+01	-4.8813e+06	1.0e-00	1.9e+00	0.0e+00	0.0e+00	0.00	0.0e+00	0.0e+00
1	9.0216e+01	-1.3664e+08	1.0e-00	1.9e+00	8.6e-13	5.0e-12	0.08	5.6e+06	1.2e+13
2	9.0403e+01	-3.3916e+08	1.0e-00	1.9e+00	7.3e-13	4.8e-13	0.16	9.1e+07	1.9e+14
3	9.0769e+01	-1.1343e+10	1.0e-00	1.9e+00	4.0e-12	1.2e-11	1.18	2.2e+08	3.9e+14
4	9.0860e+01	-1.8010e+11	1.0e-00	1.9e+00	1.5e-12	5.0e-12	2.35	8.0e+09	1.4e+16
5	9.1312e+01	-2.9307e+12	1.0e-00	1.9e+00	4.3e-12	5.1e-12	3.56	1.3e+11	2.2e+17
6	9.1710e+01	-8.2787e+13	1.0e-00	1.9e+00	6.0e-12	9.1e-12	5.01	2.1e+12	3.6e+18
7	9.2036e+01	-1.5505e+15	1.0e-00	1.9e+00	7.5e-12	6.0e-12	6.28	5.9e+13	1.0e+20
8	9.2282e+01	-6.8149e+16	1.0e-00	1.9e+00	7.0e-12	1.4e-11	7.93	1.1e+15	1.9e+21
9	9.2279e+01	-4.4155e+18	1.0e-00	1.9e+00	9.2e-12	2.1e-11	9.74	4.8e+16	8.3e+22
10	9.2244e+01	-2.8697e+20	1.0e-00	1.9e+00	6.8e-12	2.1e-11	11.55	3.1e+18	5.4e+24
11	9.2381e+01	-3.1118e+22	1.0e-00	1.9e+00	1.1e-11	3.6e-11	13.58	2.0e+20	3.5e+26
12	9.2462e+01	-7.0471e+24	1.0e-00	2.2e+01	6.2e-12	7.6e-11	15.94	2.2e+22	3.8e+28
13	9.2523e+01	-9.9820e+26	1.0e-00	2.8e+03	1.4e-11	4.7e-11	18.09	5.0e+24	8.6e+30
14	9.2605e+01	-1.1959e+30	1.0e-00	2.2e+01	2.1e-11	4.0e-10	21.17	7.1e+26	1.2e+33

Table 1: Output for NETLIB problem Forplan for default PCx with bad starting point

The inconsistency in the MPC approach arises because the corrector step, which is designed to improve the full affine scaling step, is applied at the primal-dual point; see Figure 6. In some circumstances, this mismatch can cause poor steps. In particular, we have observed that if the affine scaling step is very long, in the sense that the steplengths (2.8) are very small, and if the corrector step is even larger, then the addition of the corrector step to the primal-dual step (2.7) can significantly increase the complementarity value $x^T z$. This behaviour can be sustained and lead to very slow convergence or failure, as shown in Table 1. The results in this table were obtained using PCx [6], an interior-point code for linear programming that implements the MPC method, applied to problem Forplan from the NETLIB collection. Practical implementations of the MPC use a procedure for choosing a favorable starting point described by Mehrotra [18]. We disabled this initial point strategy of PCx and set the initial point to $x = e, z = e$. Note from Table 1 that the affine scaling and corrector steps appear to grow without bound, and examination of the results shows that the dual variables diverge.

To provide further support to the claim that the corrector step can be harmful we ran the complete set of test problems (94 in all) in the NETLIB collection. Using the default settings, which includes a strategy for computing a good starting point, PCx solved 90 problems, and terminated very close to the solution in the remaining 4 cases. Next we disabled the initial point strategy and set the initial point to $x = e, z = e$. PCx was now able to solve only 28 problems (and in only 3 additional cases terminated very close to the solution).

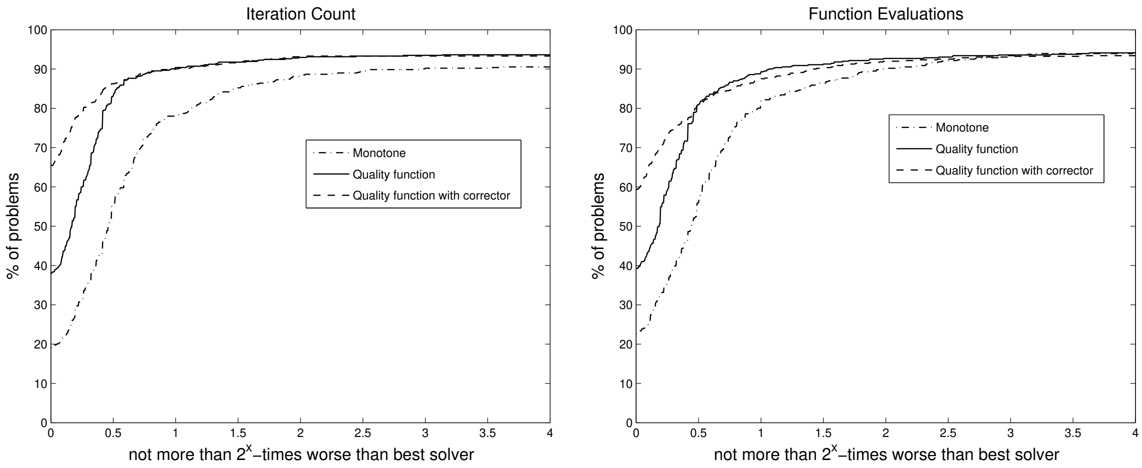
We repeated the experiment, using the initial point $x = e, z = e$, but this time removing the corrector step; this corresponds to the algorithm called *Mehrotra probing* in Section 3. We also tested a variant that we call *conditional MPC* in which the corrector step is employed in the MPC method only if it does not result in an increase of complementarity by a factor larger than 2. The results, in terms of iterations, are reported in Figure 7. Note the

Figure 7: Results on the NETLIB test set for three corrector step strategies implemented in PCx. The initial point was set to $x = e, z = e$.

dramatic increase in robustness of both strategies, compared with the MPC algorithm. The conditional MPC strategy is motivated by the observation that harmful effects of the corrector steps manifest themselves in a significant increase in complementarity. The failure of convergence of the MPC method has also been analyzed by Cartis [5].

Finally we compare the monotone and quality function approaches described in Section 3 with the conditional MPC approach on the nonlinear programming problems used in that section. The conditional MPC method is now implemented so as to reject corrector steps that increase complementarity (this more conservative approach appears to be more suitable in the nonlinear case). Furthermore, if the conditional MPC step does not pass the merit function or filter acceptance test for the current barrier problem, the corrector step is also rejected, and the backtracking line search for the regular primal-dual step is executed. Finally, no corrector step is computed while the algorithm is in the monotone mode. The results, given in Figure 8, indicate that this conditional MPC method requires fewer iterations and function evaluations, and is not less robust, than the other strategies.

Acknowledgment. We would like to thank Richard Byrd for his many valuable comments and suggestions during the course of this work.



(a) IPOPT (b) KNITRO

Figure 8: Results for safeguarded corrector steps.

References

[1] P. Armand, J. Benoist, and D. Orban. Interpretation of interior point methods as damped newton methods. Technical report, Laboratoire LACO, Universite de Limoges, Limoges, France, 2005.

[2] J. Betts, S. K. Eldersveld, P. D. Frank, and J. G. Lewis. An interior-point nonlinear programming algorithm for large scale optimization. In O. Ghattas, M. Heinkenschloss, D. Keyes, L. T. Biegler, and B. van Bloemen Waanders, editors, *Large-Scale PDE-Constrained Optimization*, Lecture Notes in Computational Science and Engineering, pages 184–198. Springer Verlag, 2003.

[3] R. H. Byrd, J.-Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.

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

[5] C. Cartis. Some disadvantages of a Mehrotra-type primal-dual corector interior point algorithms for linear programming. Technical Report NA-04/27, Oxford Computing Laboratory, Oxford, UK, 2004.

[6] Czyzyk, J., S. Mehrotra, and S. J. Wright. PCx User Guide. Technical report, Argonne National Laboratory, Argonne, IL, USA, 1996.

[7] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming, Series A*, 91:201–213, 2002.

[8] A. S. El-Bakry, R. A. Tapia, T. Tsuchiya, and Y. Zhang. On the formulation and theory of the Newton interior-point method for nonlinear programming. *Journal of Optimization Theory and Applications*, 89(3):507–541, June 1996.

[9] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. J. Wiley and Sons, Chichester, England, 1968. Reprinted as *Classics in Applied Mathematics 4*, SIAM, Philadelphia, USA, 1990.

[10] A. Forsgren and P. E. Gill. Primal-dual interior methods for nonconvex nonlinear programming. *SIAM Journal on Optimization*, 8(4):1132–1152, 1998.

[11] D. M. Gay, M. L. Overton, and M. H. Wright. A primal-dual interior method for nonconvex nonlinear programming. In Y. Yuan, editor, *Advances in Nonlinear Programming (Beijing, 1996)*, pages 31–56, Dordrecht, The Netherlands, 1998. Kluwer Academic Publishers.

[12] E. M. Gertz and P. E. Gill. A primal-dual trust region algortihm for nonlinear programming. *Mathematical Programming*, 100:49–94, 2004.

[13] D. Goldfarb and L. Chen. Interior-point ℓ_2 penalty methods for nonlinear programming with strong global convergence properties. Technical report, IEOR Dept, Columbia University, New York, NY 10027, 2004. To appear in Mathematical Programming.

[14] N. I. M. Gould, D. Orban, and Ph. Toint. An interior-point l1-penalty method for nonlinear optimization. Technical Report RAL-TR-2003-022, Rutherford Appleton Laboratory Chilton, Oxfordshire, UK, 2003.

[15] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and sifdec: A Constrained and Unconstrained Testing Environment, revisited. *ACM Trans. Math. Softw.*, 29(4):373–394, 2003.

[16] I. Griva, D.F. Shanno, and R.J. Vanderbei. Convergence Analysis of a Primal-Dual Method for Nonlinear Programming. Report, Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ, USA, 2004.

[17] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, Reading, Massachusetts, USA, second edition, 1984.

[18] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[19] A. L. Tits, A. Wächter, S. Bakhtiari, T. J. Urban, and C. T. Lawrence. A primal-dual interior-point method for nonlinear programming with strong global and local convergence properties. *SIAM Journal on Optimization*, 14(1):173–199, 2003.

[20] M. Ulbrich, S. Ulbrich, and L. Vicente. A globally convergent primal-dual interior point filter method for nonconvex nonlinear programming. *Mathematical Programming*, 2(100):379–410, 2004.

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

[22] 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.

[23] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[24] R. A. Waltz. KNITRO 4.0 User's Manual. Technical report, Ziena Optimization, Inc., Evanston, IL, USA, October 2004.

[25] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. Technical Report 2003-6, Optimization Technology Center, Northwestern University, Evanston, IL, USA, June 2003. To appear in Mathematical Programming A.

[26] S. Wright. *Primal-dual interior-point methods.* Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[27] H. Yamashita. A globally convergent primal-dual interior-point method for constrained optimization. *Optimization Methods and Software*, 10(2):443–469, 1998.

[28] H. Yamashita, H. Yabe, and T. Tanabe. A globally and superlinearly convergent primal-dual interor point trust region method for large scale constrained optimization. *Mathematical Programming, Series A*, 102(1):111–120, 2005.