Available online at www.sciencedirect.com
SCIENCE DIRECT®
Applied Mathematics Letters 17 (2004) 945–952
Applied Mathematics Letters
www.elsevier.com/locate/aml

A Starting Point Strategy for Nonlinear Interior Methods

M. GERTZ*
Computer Science Department, University of Wisconsin at Madison
Madison, WI 53706, U.S.A.

J. NOCEDAL†
Department of Electrical and Computer Engineering
Northwestern University, Evanston, IL 60208-3118, U.S.A.

A. SARTENAER‡
Department of Mathematics, Facultés Universitaires Notre-Dame de la Paix
61, rue de Bruxelles, B-5000 Namur, Belgium

(Received and accepted September 2003)

Communicated by P. T. Boggs

Abstract—This paper presents a strategy for choosing the initial point, slacks, and multipliers in interior methods for nonlinear programming. It consists of first computing a Newton-like step to estimate the magnitude of these three variables and then shifting the slacks and multipliers so that they are sufficiently positive. The new strategy has the option of respecting the initial estimate of the solution given by the user, and attempts to avoid the introduction of artificial nonconvexities. Numerical experiments on a large test set illustrate the performance of the strategy. © 2004 Elsevier Ltd. All rights reserved.

1. INTRODUCTION

It is well known that interior methods for linear and quadratic programming perform poorly (and can even fail) if the starting point is unfavorable. To overcome this problem, it is common to employ heuristics for choosing an initial value for the variables, slacks, and multipliers (see, e.g., [1–3]). These heuristics have proved to be generally successful in practice and have been incorporated into commercial linear programming packages. In this paper, we study initial point strategies for nonlinear programming. This topic has not received much attention in spite of the fact that nonlinear interior methods can be as sensitive as their linear counterparts to a poor initial guess.

The heuristics developed for linear and quadratic programming cannot be extended directly

We would like to thank J. L. Morales for providing the MATLAB code IPM which served as the basis for Algorithm 5.1, and R. Waltz for implementing the initial point strategy in KNITRO 3.0.
*This author supported by National Science Foundtion Grants CCR-9896198 and DMS-9973276.
†This author was supported by National Science Foundation Grants CCR-9987818, ATM-0086579, and CCR-021943 and Department of Energy Grant DE-FG02-87ER25047-A004.
‡This author was supported by the Belgian National Fund for Scientific Research.

0893-9659/04/$ - see front matter Typeset by AMS-TEX
doi:10.1016/j.aml.2003.09.005

to nonlinear problems. First of all, in linear programming an initial estimate of the solution is typically not provided by the user. Moreover, since the objective function and constraints are defined everywhere, there is great freedom in selecting initial values, and some of the most popular strategies often choose very large values for the variables, slacks, and (possibly) multipliers; see [4] and the references therein.

In contrast, nonlinear programming algorithms compute only local minimizers and accept user-supplied initial estimates that often lie in the vicinity of a minimizer of interest. Therefore, initial point strategies should either respect the user-supplied estimate or compute one that is not too distant from it. Even large initial values of the multipliers should be avoided since they may introduce unnecessary nonconvexities in the problem, as we discuss later on. The initial point heuristics presented in this paper aim to preserve user-supplied information, are readily computable, and allow interior methods to perform efficiently on a wide range of problems.

2. INTERIOR POINT FRAMEWORK

We will consider the solution of nonlinear programming problems of the form

$$\begin{aligned} &\text{minimize} && f(x), \\ &\text{subject to} && h(x) = 0, \\ & && g(x) \geq 0, \end{aligned} \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^q$, and $g : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable. Introducing a vector of slack variables, s, we can restate (2.1) as

$$\begin{aligned} &\text{minimize} && f(x), \\ &\text{subject to} && h(x) = 0, \\ & && g(x) - s = 0, \\ & && s \geq 0. \end{aligned} \tag{2.2}$$

The first-order optimality conditions of (2.2) can be written as

$$r(x, y, z, s, \lambda) = 0, \qquad s^\top \lambda = 0, \qquad s, \lambda \geq 0,$$

where r is a vector function whose components are

$$r_x(x, y, z) = \nabla f(x) - A(x)^\top y - C(x)^\top z, \tag{2.3a}$$

$$r_y(x) = h(x), \tag{2.3b}$$

$$r_z(x, s) = g(x) - s, \tag{2.3c}$$

$$r_s(z, \lambda) = z - \lambda. \tag{2.3d}$$

Here $A(x)$ and $C(x)$ denote the Jacobian matrices of h and g, respectively. Rather than using equation (2.3d) to eliminate λ, as is often done, we will keep this equation and maintain separate values of z and λ so as to have more flexibility in choosing an initial value for z. In Section 3, we describe how to apply the initial point strategy to interior methods that do not use this formulation.

A primal-dual interior method computes a displacement by applying Newton's method to the system

$$r(x, y, z, s, \lambda) = 0, \tag{2.4a}$$

$$S\lambda = \mu e, \tag{2.4b}$$

where the scalar $\mu > 0$ is the barrier parameter, S is a diagonal matrix whose diagonal entries are given by the components of s, and e is the vector of all ones. The displacement produced by this Newton iteration solves the system

$$\begin{aligned} H(x,y,z)\Delta x - A(x)^\top \Delta y - C(x)^\top \Delta z &= -\nabla f(x) + A(x)^\top y + C(x)^\top z, \\ A(x)\Delta x &= -h(x), \\ C(x)\Delta x - \Delta s &= -g(x) + s, \\ \Delta z - \Delta\lambda &= -z + \lambda, \\ \Lambda\Delta s + S\Delta\lambda &= -\Lambda s + \mu e. \end{aligned} \tag{2.5}$$

Here $H(x,y,z) = \nabla_x^2 L(x,y,z)$, where $L(x,y,z) = f(x) - y^\top h(x) - z^\top g(x)$, and Λ is a diagonal matrix whose diagonal entries are given by the components of λ. The new iterate is given by

$$(x^+, y^+, z^+, s^+, \lambda^+) = (x,y,z,s,\lambda) + \alpha(\Delta x, \Delta y, \Delta z, \Delta s, \Delta\lambda), \tag{2.6}$$

where $\alpha > 0$ is a steplength that ensures decrease of a merit function and positivity of the variables s, λ.

This simple formulation provides the conceptual framework for many nonlinear interior algorithms, but modifications or reformulation are required to deal with nonconvexity and singularities; see, e.g., [5]. Since we wish to present the new strategies in the most general framework, we will initially assume that the iterates are computed by an algorithm of the form (2.5),(2.6).

3. AN INITIAL POINT STRATEGY FOR NONLINEAR OPTIMIZATION

Interior methods can be very sensitive to the initial choice of the variables because, in unfavorable circumstances, the primal-dual direction (2.5) is drastically shortened by the nonnegativity requirement $s, \lambda > 0$ and produces negligible progress toward the solution. This behavior can be sustained for many iterations, rendering the solution process inefficient. In this section, we present a heuristic that we have found often produces a good starting point for both nonlinear programming and for the simpler classes of linear and quadratic programming problems.

We assume that *preliminary* values $x_0, y_0, z_0, s_0, \lambda_0$ are assigned to all the variables. The initial estimate of the solution, x_0, is either provided by the user or is set to a default value (such as $x_0 = 0$), and we assume that the interior method computes the multiplier estimates y_0, z_0. Vectors s_0 and λ_0 will be set to a constant value $\delta > 0$ in our tests.

Using these preliminary values, we compute an affine-scaling step, Δv_A, by setting $\mu = 0$ in the primal-dual system (2.5). Once $\Delta v_A = (\Delta x_A, \Delta y_A, \Delta z_A, \Delta s_A, \Delta\lambda_A)$ has been computed, we define the vectors

$$u = (s_0 + \Delta s_A)^-, \qquad w = (\lambda_0 + \Delta\lambda_A)^-,$$

where $x^- = \max\{0, -x\}$. Vectors u and w represent the violations of the primal and dual nonnegativity constraints caused by the full affine scaling step. Then, given scalars $\beta_1 > 0$ and $\beta_2 \geq 1$, we compute initial values, s_1 and λ_1, by one of the following two rules:

$$\begin{aligned} &\text{Rule 1:} && s_1^{(i)} = \max\left(\beta_1, \left|s_0^{(i)} + \Delta s_A^{(i)}\right|\right), && \lambda_1^{(i)} = \max\left(\beta_1, \left|\lambda_0^{(i)} + \Delta\lambda_A^{(i)}\right|\right). \\ &\text{Rule 2:} && s_1 = s_0 + \Delta s_A + \beta_1 + \beta_2 u, && \lambda_1 = \lambda_0 + \Delta\lambda_A + \beta_1 + \beta_2 w. \end{aligned}$$

Whichever of these rules is chosen, we define $(x_1, y_1, z_1) = (x_0, y_0, z_0)$ and set the initial value of the barrier parameter to $\mu_1 = s_1^\top \lambda_1 / m$.

As a practical matter, many interior point implementations do not maintain separate values of z and λ, i.e., they do not include equation (2.3d) in the statement of the optimality conditions,

and λ is not defined. We recommend that these codes compute z_1 by one of the rules mentioned above, with z playing the role of λ. We also recommend that the Hessian for the primal-dual step (2.5) at the initial point v_1 be defined using z_0 and not z_1, i.e.,

$$H_1 = H(x_0, y_0, z_0).$$

There are three reasons for making this choice. An examination of the primal-dual system (2.3) reveals that if H_1 does not depend on z_1, then neither does the step $(\Delta x_1, \Delta y_1, \Delta z_1, \Delta s_1, \Delta\lambda_1)$. Thus, with the choice $H_1 = H_0$, all algorithms will compute the same steps in the primal and dual slacks, whether or not they maintain z and λ as separate variables. The second reason is that z_1 could be very large and introduce an undesirable distortion in the quadratic model used by Newton's method. In particular, if one of the components, say z_1^i, is large and the corresponding Hessian term $\nabla^2 c_i(x_1)$ is indefinite, the Hessian H_1 can become indefinite, slowing down the iteration (we have often observed this phenomenon in practice). Finally, the cost of evaluating $H(x, y, z)$ is saved.

We summarize our interior point strategy in the following pseudocode.

ALGORITHM 3.1. START STRATEGY.

Choose constants $\delta > 0$, $\beta_1 > 0$, and $\beta_2 \geq 1$.
x_0 is provided by the user; otherwise it is set to a default value such as $x_0 = 0$.
Compute initial values of the multipliers y_0 and z_0.
Evaluate the functions f, h, g and their derivatives at x_0.
Let $s_0^{(i)} \leftarrow \delta$ and $\lambda_0^{(i)} \leftarrow \delta$.
Compute the affine scaling step Δv_A by solving system (2.5) with $\mu = 0$.
Let $(x_1, y_1) \leftarrow (x_0, y_0)$.
Choose s_1 and λ_1 by one of the two rules given above. Define $\mu_1 = s_1^T \lambda_1 / m$.
if the algorithm maintains separate values for z and λ **then**
  Let $z_1 \leftarrow z_0$. Start the interior algorithm from $v_1 = (x_1, y_1, z_1, s_1, \lambda_1)$.
else
  Let $z_1 \leftarrow \lambda_1$. Start the interior algorithm from $v_1 = (x_1, y_1, z_1, s_1)$.
end if
Define the initial Hessian matrix $H_1 = H(x_0, y_0, z_0)$.

Note that, when the algorithm maintains both z and λ, we can partition the variables of the problem into two classes. The first consists of the variables x, y, and z which are unrestricted in sign and which are needed to evaluate the problem functions, including the Lagrangian $L(x, y, z)$. The second class consists of the variables s and λ which are subject to nonnegativity constraints; these are the only variables reset by the initial point strategy.

4. MOTIVATION

The interior point method forms a quadratic model of the nonlinear program at the preliminary point (x_0, y_0, z_0). Because this quadratic model represents all the information available at the start, it makes sense to consider what values of $v_1 = (x_1, y_1, z_1, s_1, \lambda_1)$ would constitute a good initial point for minimizing the quadratic model. Thus, our initial point strategy is motivated by the theory and practice of interior methods for convex quadratic programming.

Many interior methods for quadratic programming can be seen as path-following methods. One way of characterizing the central path is by means of the neighborhood

$$\mathcal{N}_{-\infty}(\beta, \delta) = \left\{ v \;\middle|\; \frac{\|r(v)\|}{\gamma} \leq \beta \left[\frac{\|r(v_1)\|}{\gamma_1} \right],\ (s, \lambda) > 0,\ s^{(i)}\lambda^{(i)} \geq \delta\gamma \right\},$$

where $\beta \geq 1$, $\delta \in (0, 1)$, and $\gamma = s^T\lambda/m$. It has been shown [4] that for certain primal-dual methods, if the step α_k is restricted at each iteration so that $v_{k+1} \in \mathcal{N}_{-\infty}(\beta, \delta)$, then there is the

sequence of steplengths $\{\alpha_k\}$ that is bounded away from zero. Moreover, for a fairly wide range of initial points, the sequence $\{\|r(v_k)\|/\gamma_k\}$ will remain bounded, the sequences $s_k^{(i)}\lambda_k^{(i)}/\gamma_k$ will remain bounded away from zero for every i and k and the iteration will converge. Thus, for a good choice of initial point, the iteration stays in $\mathcal{N}_{-\infty}(\beta,\delta)$, for some β and δ.

Typical start strategies for linear and quadratic programming [1,2,6] attempt to choose a point which is close to the central path, in the metric suggested by the definition of $\mathcal{N}_{-\infty}(\beta,\delta)$. Thus, they attempt to find a point for which $\|r(v_1)\|/\gamma_1$ is small and for which none of the products $s_1^{(i)}\lambda_1^{(i)}$ is much smaller than γ_1. Although these start strategies are heuristic, they are effective in practice. They tend to prevent the steplength α_1 from being small and tend to bound $\{\|r(v_k)\|/\gamma_k\}$ by a small number, which is often $\|r(v_1)\|/\gamma_1$.

Rules 1 and 2 place a nonnegative lower bound on λ and s, which in turn places a lower bound on each of their pairwise products. Rule 1 is based on the observation that the affine scaling step often captures the scale of the variables and so chooses a perturbed point with the same scale. Rule 2 is based on the observation that $\|r(v)\|$ grows linearly in $\|(s,\lambda)\|$ but γ grows quadratically, and so shifting all the variables will tend to increase γ_1 faster than it increases $r(v_1)$. In Rule 2, the scale of the affine scaling step is incorporated into the shift.

However, it is important to observe that for the primal-dual system (2.5) for quadratic programs, neither Δs nor $\Delta\lambda$ depends on the value of x, y, or z. Moreover, the maximum step α for which $s+\alpha\Delta s$ and $\lambda+\alpha\Delta\lambda$ are nonnegative does not depend on x, y, and z. It is not difficult to see that for most quadratic programming algorithms, the entire sequence of iterates $\{(s_k,\lambda_k)\}$ does not depend at all on (x_1,y_1,z_1). Thus, the size of $\|r(v_1)\|/\gamma_1$ is not important in an absolute sense. It is only important that for the chosen (s_1,λ_1), there exists a choice of (x_1,y_1,z_1) that makes the ratio $\|r(v_1)\|/\gamma_1$ small.

In nonlinear programming, it is wise to choose $(x_1,y_1,z_1)=(x_0,y_0,z_0)$, rather than using the affine scaling values, for the following reasons. First, we preserve the user-supplied starting point x_0, and we avoid the risk that the problem functions may not be defined at the value given by the affine scaling step. Moreover, our heuristics are based on a particular quadratic model of the problem. Changing (x,y,z) alters this quadratic model, which puts into question the usefulness of these heuristics.

5. PRACTICAL IMPLEMENTATION

We will test our initial point strategies using KNITRO [7,8], a software package that implements a nonlinear interior method. We will use KNITRODIRECT, the version that computes the step using direct linear algebra.

To demonstrate the general applicability of our approach, we also test it using a simple interior algorithm designed specifically for this study. Algorithm 5.1 outlines this primal-dual iteration. To measure progress, we employ the merit function

$$\phi_\rho(x,s) = f(x) - \sum \log s^{(i)} + \rho\,\|[r_y(x), r_z(x,s)]\|_2\,, \tag{5.1}$$

where $\rho>0$ is the penalty parameter.

ALGORITHM 5.1. DAMPED SHORT-STEP PATH-FOLLOWING METHOD.

Choose tolerances $\gamma_{\text{tol}}, r_{\text{tol}}, \tau$.
Set $k \leftarrow 1$ and initialize ρ.
Compute $v_1=(x_1,y_1,z_1,s_1,\lambda_1)$ and μ_1 using Algorithm 3.1.
Compute $\gamma_1 = s_1^T\lambda_1/m$ and $r_1 = r(v_1)$, where r is defined in (2.3).
while $\gamma_k \geq \gamma_{\text{tol}}$ **or** $\|r(v_k)\|_\infty \geq r_{\text{tol}}$ **do**
  Solve the primal-dual system (2.5) for Δv_k.
  Compute $\bar{\alpha}_k \leftarrow \max\{\alpha\in(0,1] \mid s_k+\alpha\Delta s_k \geq \tau s_k \text{ and } \lambda_k+\alpha\Delta\lambda_k \geq \tau\lambda_k\}$.
  Update the penalty parameter ρ if necessary.

Use a line-search procedure to choose v_{k+1}.
Compute $\gamma_{k+1} = s_{k+1}^T \lambda_{k+1}/m$ and $r_{k+1} = r(v_{k+1})$.
if $\|r_{k+1}\|_\infty < \mu_k$ **and** $\gamma_{k+1} < 10\mu_k$ **then**
  $\mu_{k+1} \leftarrow \max\{\gamma_{k+1}/11, 10^{-2} \times \gamma_{\text{tol}}\}$.
else
  $\mu_{k+1} \leftarrow \mu_k$.
end if
$k \leftarrow k+1$.
end do

The algorithm omits several key steps needed to ensure global convergence. In particular, it can only be applied to convex problems since it does not include a feature for handling negative curvature. Nonetheless, we find that this simple algorithm is useful for illustrating the effectiveness of the initial point heuristics; nonconvex problems will be tested with KNITRO.

6. NUMERICAL EXPERIMENTS

All the results in this section will be presented using the logarithmic performance profiles described in [9]. In the figures, the y-axis plots $\pi_s(t)$, which is defined as

$$\pi_s(t) = \frac{\text{no. of problems where } \log_2(r_{p,s}) \leq t}{\text{total no. of problems}}, \qquad t \geq 0, \tag{6.1}$$

where $r_{p,s}$ is the ratio between the time to solve problem p by solver s over the lowest time required by any of the solvers. The x-axis plots t.

We first test Algorithm 5.1 on the Maros-Mezaros set of quadratic programs; see `http://cuter.rl.ac.uk/cuter-www/Problems/marmes.html`. In Figure 1, we compare the number of iterations required by Algorithm 5.1 using Rules 1 and 2, and using no initial point strategy (NoInit). Algorithm 5.1 has been implemented in MATLAB and uses the HSL routine `MA27` [10] to solve the primal-dual system. If negative curvature is detected, the algorithm stops. The parameters of Algorithm 5.1 were set as $[\gamma_{\text{tol}}, r_{\text{tol}}, \tau] = [10^{-6}, 10^{-6}, 0.005]$.

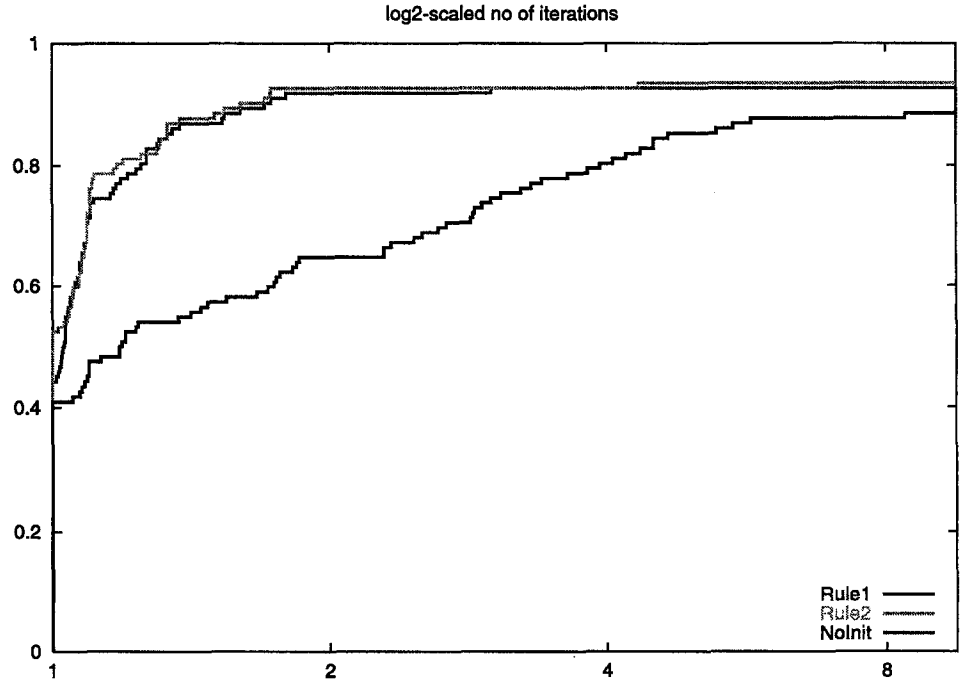

Figure 1. Number of iterations for Algorithm 5.1 on quadratic programs.

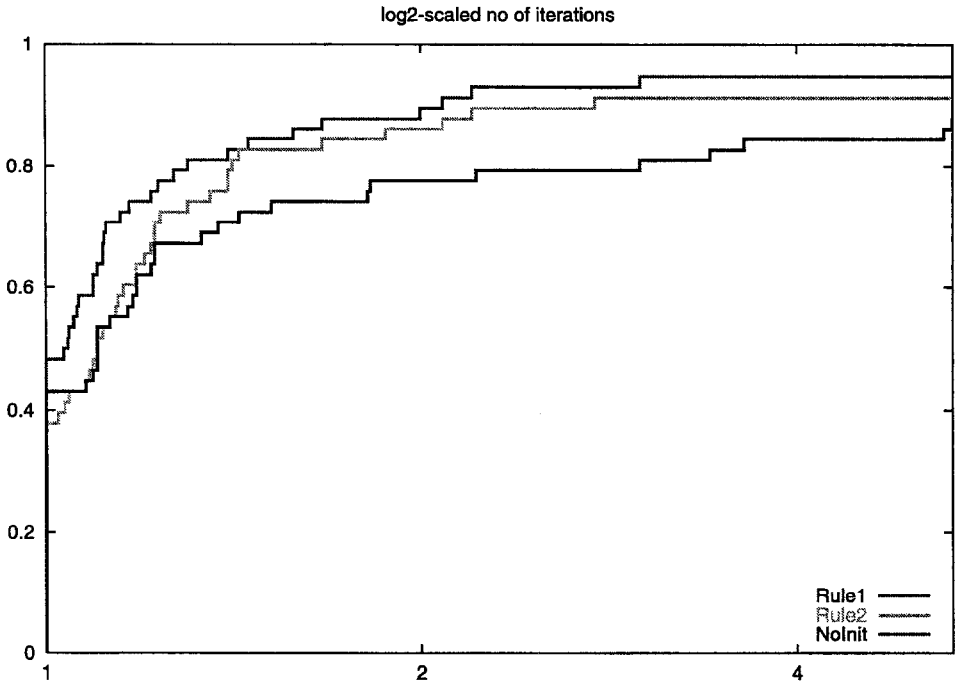

Figure 2. Number of iterations for Algorithm 5.1 on CUTEr problems.

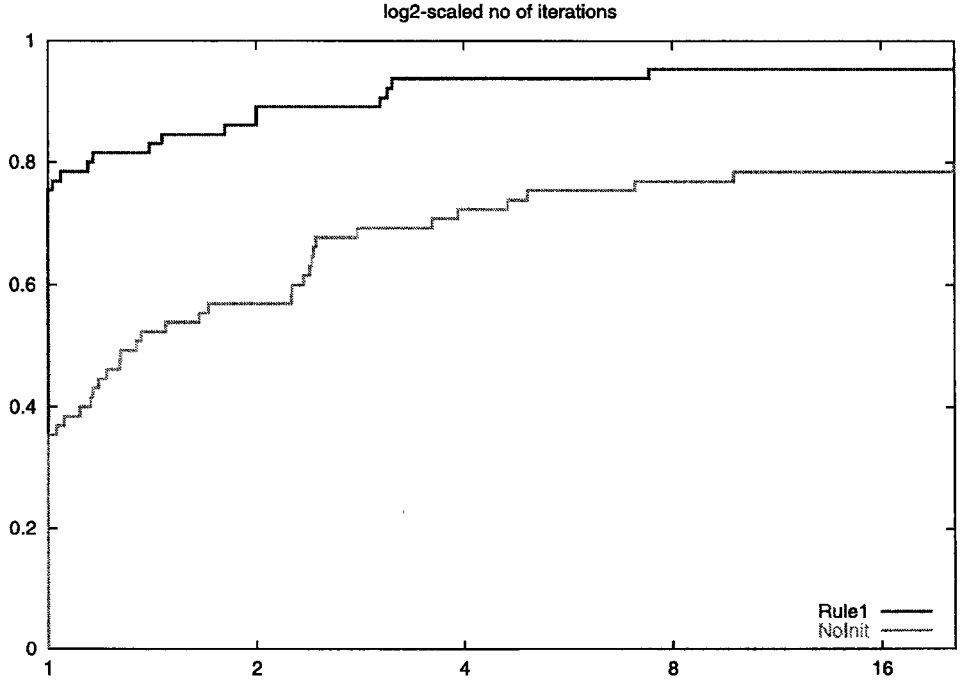

Figure 3. Number of iterations for KNITRO.

Next, we test Algorithm 5.1 on 58 problems from the CUTEr collection [11] that were selected as follows. We identified all problems with inequality constraints that could be run in less than 30 minutes. Then, we removed all problems that could be solved by all strategies in less than ten iterations, as well as problems for which the number of variables plus constraints is less than 50. We also removed all problems in which negative curvature was detected. The performance of

Algorithm 5.1 on this test set are presented in Figure 2. Even though this test set contains many problems that can be solved quickly using the default starting point, we note that the initial point strategies yield a slight improvement in robustness and efficiency.

Finally, we test KNITRO 3.0 on 66 challenging problems selected specifically for this study. The set consists of most of the Brunel problems from the Maros-Mezaros test set, various difficult problems we have identified, and some problems from the CUTER test set.

In Figure 3, we report results for Rule 1 (which gives the best results for KNITRO) and for the default that uses no initial point strategy (NoInit). We set $\beta_1 = 1$ in Rule 1. Note the dramatic improvement in performance provided by the initial point strategy. As a result of this testing, we recommend this as an option in nonlinear interior methods.

REFERENCES

1. S. Mehrotra, On the implementation of a primal-dual interior point method, *SIAM Journal on Optimization* **2** (4), 575–601, (1992).
2. E.D. Andersen, J. Gondzio, C. Mészáros and X. Xu, Implementation of interior point methods for large scale linear programming, In *Interior Point Methods in Mathematical Programming*, (Edited by T. Terlaky), pp. 189–252, Kluwer Academic, Dordrecht, The Netherlands, (1996).
3. R.J. Vanderbei and D.F. Shanno, An interior point algorithm for nonconvex nonlinear programming, *Computational Optimization and Applications* **13**, 231–252, (1999).
4. S.J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, PA, (1997).
5. A. Forsgren, P.E. Gill and M.H. Wright, Interior methods for nonlinear optimization, *SIAM Review* **44** (4), 525–598, (2002).
6. E.M. Gertz and S.J. Wright, Object-oriented software for quadratic programming, Technical Report, Computer Sciences Department, University of Wisconsin, (2001).
7. 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).
8. R.A. Waltz and J. Nocedal, KNITRO User's Manual, Technical Report OTC 2003/05, Optimization Technology Center, Northwestern University, Evanston, IL, (April 2003).
9. E.D. Dolan and J.J. Moré, Benchmarking optimization software with performance profiles, *Mathematical Programming, Series A* **91**, 201–213, (2002).
10. Harwell Subroutine Library, *A Catalogue of Subroutines (HSL 2000)*, AEA Technology, Harwell, Oxfordshire, England, (2002).
11. I. Bongartz, A.R. Conn, N.I.M. Gould and Ph.L. Toint, CUTE: Constrained and Unconstrained Testing Environment, *ACM Transactions on Mathematical Software* **21** (1), 123–160, (1995).