Finite Termination and Superlinear Convergence in Primal-Dual Methods

Sanjay Mehrotra†

Department of Industrial Engineering and Management Sciences,
Northwestern University, Evanston, IL 60208-3119, U.S.A.

Technical Report 91-13

July 1991

Abstract

This paper studies two questions important for developing implementations of primal-dual interior point methods. (i) When to jump to an optimal solution from a near optimal solution? (ii) What is the role of asymptotic superlinear convergence in these methods?

The empirical evidence reported in this paper suggests that typically we detect superlinear convergence after the optimal face and a point on it could have been identified. However, superlinear convergence property serves well as an indicator for deciding when to try for the optimal face.

† Supported in part by a grant from the GTE Laboratories and by the grant CCR-9019469 from the National Science Foundation.

Introduction

It is well known that an optimal solution of a linear program can be obtained in polynomial time, if the problem data is rational. Recently, Mehrotra and Ye[16] showed that the optimal face of a linear program can be identified in polynomial time using an interior point method. Furthermore, a point in the interior of this face can be generated efficiently.

In addition to being an interesting combinatorial problem, the ability to identify optimal face has several direct uses in the context of interior methods. First, it could serve as a termination criterion with in the framework of the implementations of these methods. In addition, the sensitivity analysis proposed by Adler and Monteiro [1] assumes its knowledge.

The Mehrotra and Ye approach uses an indicator to partition variables and generates a point (optimal solution) in the interior of the optimal face. Computational results given in [16] showed the practical usefulness of identifying optimal face.

A question that remained unanswered in their work is: when to partition variables in practice?

In a class of interior point methods, namely the primal-dual methods, practical implementations indicate that asymptotically the variables converge superlinearly. A theory explaining the superlinear convergence behavior was developed by Zhang, Tapia and Dennis [25]. More recently, Zhang and Tapia [27] have developed a polynomial time primal-dual method which is shown to be superlinearly convergent under certain conditions. However, the importance of superlinear convergence is often questioned for polynomial algorithms since they also have finite termination property.

The natural question to ask is: in practice do we observe superlinear convergence before identifying the optimal face?

This paper addresses the two questions asked above. It provides computational evidence indicating that typically a point on the optimal face can be generated before we begin to observe superlinear convergence. This indicates that the superlinear convergence behavior observed in primal-dual methods is a consequence of a solution being close to the optimal face. However, if we do not use the correctly identified partition to stop, and let the algorithm run, it begins to converge superlinearly almost immediately.

The latter suggests a non-traditional use of superlinear convergence property. We can use superlinear convergence as an indicator for deciding when to partition variables. We give computational evidence on this.

This paper is organized as follows. In the next section we state several results on the optimal face identification problem and the superlinear convergence properties in primal-dual methods. In Section 2 we outline the algorithm we implemented for our experiments. Section 3 presents and discusses computational results on the face identification problem and the superlinear convergence observed in practice.

1. Optimal Facet and Superlinear Convergence

We are interested in solving a linear program (LP):

$$\begin{array}{ll} \min & c^T x \\ s.t. & Ax = b, \ x \geq 0, \end{array}$$

and its dual (LD):

$$\begin{array}{ll} \max & b^T y \\ s.t. & A^T y + s = c, \ s \geq 0, \end{array}$$

where $A \in \mathbf{R}^{\mathbf{m} \times \mathbf{n}}$, $c \in \mathbf{R}^{\mathbf{n}}$, and $b \in \mathbf{R}^{\mathbf{m}}$. Feasible solutions x^* and (y^*, s^*) respectively for (LP) and (LD) are optimal, if and only if,

$$x_j^* s_j^* = 0 \ \text{ for } \ j = 1, 2, ..., n.$$

1.1 A Generic First Order Primal-Dual Method

We first describe a generic primal-dual method (e.g., see Kojima et al. [10], Lustig, Marsten and Shanno [11], McShane, Monma and Shanno [12], Mehrotra [13], Mizuno, Todd and Ye [17], Monteiro and Adler [18], and Zhang, Tapia and Dennis [25]). Several convergence results are stated for this method later in this section.

Let us assume that at the beginning of iteration k a feasible solution $x^k > 0$, y^k, $s^k > 0$ is available. For simplicity we define $x(1) = x^k$, $y(1) = y^k$, $s(1) = s^k$. Let $X(1) = \text{diag}(x(1))$. We follow similar notation to define diagonal matrices for other variables. Let $\beta^k \in [0, 1)$ and $\tau^k \in (0, 1)$. Let $\mu^k = \beta^k x^k s^k / n$ and e be a vector of all ones.

We define a system of non-linear equations as

$$\begin{array}{c} X(\alpha)s(\alpha) = \alpha X(1)s(1) + (1 - \alpha)\mu^k e, \\ A^T y(\alpha) + s(\alpha) = c, \\ Ax(\alpha) = b. \end{array} \tag{1.1}$$

The first derivative of (1.1) at $\alpha = 1$ is used as a search direction. Differentiating (1.1) with respect to α at $\alpha = 1$ (for simplicity α is suppressed from the argument) gives

$$\begin{bmatrix} S & X & 0 \\ & I & A^T \\ A & 0 & 0 \end{bmatrix} \begin{pmatrix} \dot{x} \\ \dot{s} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} X^k s^k - \mu^k e \\ 0 \\ 0 \end{pmatrix}. \tag{1.2}$$

After the first derivative is computed using (1.2) a new solution is obtained as follows

$$\begin{array}{l} x^{k+1} \leftarrow x^k - \tau^k \alpha_x^k \dot{x}, \\ s^{k+1} \leftarrow s^k - \tau^k \alpha_s^k \dot{s}, \\ y^{k+1} \leftarrow y^k - \tau^k \alpha_s^k \dot{y}, \end{array}$$

where

$$\alpha_x^k = \min(1, \min(x_i^k/\dot{x}_i |\ \dot{x}_i > 0)),$$
$$\alpha_s^k = \min(1, \min(s_i^k/\dot{s}_i |\ \dot{s}_i > 0)).$$

1.2 Results on the Optimal Face Identification Problem

Let $\sigma(x)$ represent the index set of positive components in $x \geq 0$, that is,

$$\sigma(x) = \{i : x_i > 0\}.$$

Among all the optimal solutions for (LP) and (LD), there exists at least one optimal solution pair (x^*, s^*) which is strictly complementary, that is,

$$\sigma(x^*) \cap \sigma(s^*) = \emptyset \ \text{ and } \ \sigma(x^*) \cup \sigma(s^*) = \{1, 2, \ldots, n\}. \tag{1.3}$$

for every complementary solution (x^*, s^*). Moreover, $\sigma(x^*)$ and $\sigma(s^*)$ remain invariant for every strictly complementary solution (x^*, s^*). Hence, we can denote $\sigma(x^*)$ by σ^* for (LP) and let $\bar{\sigma}^* = \{1, \ldots, n\} \setminus \sigma^*$. One can further show that

$$\sigma(x^*) \subset \sigma^* \ \text{ and } \ \sigma(s^*) \subset \bar{\sigma}^*$$

for every complementary solution (x^*, s^*). Thus, the optimal face for the primal is

$$\Theta_p = \{x : Ax = b,\ x \geq 0,\ \ x_j = 0, \text{ for } j \in \bar{\sigma}^*\},$$

and one for the dual is

$$\Theta_d = \{y : A^T y + s = c, \quad s_j = 0, \text{ for } j \in \sigma^*\}.$$

This property has been known since the early days of linear programming (Goldman and Tucker [7]). A proof can be found in the recent book by Schrijver ([20], pp. 95-96). The following results were proved by Mehrotra and Ye [16].

Theorem 1. *Let $\{(x^k, s^k)\}$ be generated in an interior point method. At iteration k, let*

$$\sigma^k = \{j : x_j^k \geq s_j^k\}. \tag{1.4}$$

Let the data in (LP) be rational, and L be its input length. For all the algorithms satisfying the results of Güler and Ye [9], if $(x^k)^T s^k \leq 2^{-3L}$, then

$$\sigma^k = \sigma^*.$$

Theorem 2. *Let us assume that the sequence $(x^k, s^k) \to (x^*, s^*)$ is generated in a primal-dual algorithm. Define*

$$\sigma^k = \{j : |x_j^{k+1} - x_j^k|/x_j^k \leq |s_j^{k+1} - s_j^k|/s_j^k\}. \tag{1.5}$$

Then for all the algorithms satisfying the results of Güler and Ye [9], there exists a K such that for all $k \geq K$

$$\sigma^k = \sigma^*.$$

The following criterion, which is derived from (1.4) and (1.5), was used to partition variables for the computational results in [16]:

$$\sigma^k = \{j : s_j^k \le 10^{-14} \text{ or } |x_j^{k+1} - x_j^k|/x_j^k \le |s_j^{k+1} - s_j^k|/s_j^k\}. \tag{1.6}$$

The use of relative change in variables as an indicator is similar to Tapia's indicator (see El-Bakry, Tapia and Zhang [3]). Mehrotra and Ye also gave the following approach for finding a solution in the interior of Θ_p and Θ_d. Solutions obtained from this approach are used to verify optimality of a partition.

For simplicity, let those columns in A corresponding to σ^k form matrix B and the remaining columns form matrix N. Let us represent the corresponding variables by x_B and x_N, respectively. Note that we have not made any assumptions on B. To find a point in the interior of Θ_p, we solve system of linear equations:

$$B\Delta x_B = b - Bx_B^k = Nx_N^k \tag{1.7}$$

for Δx_B. Linearly dependent rows and/or columns in (1.7) are deleted during the Gaussian elimination. The elements of Δx_B corresponding to the linearly dependent columns are set to zero. A solution is then generated as $x_B^* = x_B^k + \Delta x_B$, $x_N^* = 0$. A point in the interior of Θ_d can be found by considering the problem

$$B^T\Delta y = c_B - B^T y^k = s_B^k,$$

and using $y^* = y^k + \Delta y$ for computing $s = c - A^T y^*$. Linearly dependent rows and/or columns of B^T are dropped as they are identified. The components of Δy corresponding to the linearly dependent columns are set to zero.

1.3 Results on Superlinear Convergence

Zhang, Tapia and Dennis [25] proved the following result (see also Zhang, Tapia and Potra [26]).

Theorem 3. *Let $\{(x^k, s^k)\}$ be generated by the generic primal-dual algorithm and $(x^k, s^k) \to (x^*, s^*)$. If the following assumptions hold,*

(i) strict complementarity,

(ii) the sequence $x^{k^T} s^k/(n\min(x_i^k s_i^k))$ is bounded,

(iii) $\tau^k \to 1$ and $\beta^k \to 1$,

then (x^*, s^*) solves (LP) and the sequence $\{x^k s^k\}$ converges to zero $Q-$superlinearly.

Recently, Zhang and Tapia [27] showed that it is possible to develop a primal-dual algorithm for which assumptions (ii) and (iii) in Theorem 3 are satisfied for appropriate choices of parameters.

2. A Practical Interior Point Method

In this section we describe a practical interior point method which we used for our experiments. This method is discussed in more details in Mehrotra [13, 14]. The description given here is for completeness. This description assumes problems in the form (LP). For our handling of bounds and free variables see [14] and [15] respectively. The method developed here uses a second order approximation of an appropriately defined trajectory. This method was used for all the results reported in this paper.

Let us assume that at iteration k a solution $x(1) = x^k > 0$, $y(1) = y^k$, $s(1) = s^k > 0$ is available. This solution may be infeasible. Let $\xi_x \equiv Ax(1) - b$ and $\xi_s \equiv A^T\pi(1) + s(1) - c$. Now consider the trajectory defined by

$$\begin{aligned} X(\alpha)s(\alpha) &= \alpha X^k s^k + \alpha(1-\alpha)^2 \mu e, \\ A^T\pi(\alpha) + s(\alpha) &= c + \alpha\xi_s, \\ Ax(\alpha) &= b + \alpha\xi_x. \end{aligned} \tag{2.1}$$

The first derivative of this trajectory at $\alpha = 1$ (value of α is suppressed from the argument) is obtained by solving

$$H\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \equiv \begin{bmatrix} -X^{-1}S & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} \xi_s - s^k \\ \xi_x \end{pmatrix}, \tag{2.2}$$

and computing $\dot{s} = \xi_s - A^T\dot{y}$. This derivative is used to compute the centering parameter μ. To compute μ, we first find

$$\begin{aligned} \alpha_x^1 &\equiv \min(1, \min\{x_i(1)/\dot{x}_i \mid \dot{x}_i > 0\}), \\ \alpha_s^1 &\equiv \min(1, \min\{s_i(1)/\dot{s}_i \mid \dot{s}_i > 0\}), \end{aligned}$$

and then compute

$$\beta \equiv (x^k - \alpha_x^1\dot{x})^T(s^k - \alpha_s^1\dot{s})/{x^k}^T s^k.$$

We then take

$$\mu = \begin{cases} \beta^3 {x^k}^T s^k/n & \text{if } (\|\dot{x}\|^2 + \|\dot{s}\|^2)/{x^k}^T s^k < 1.1; \\ \beta^3 {x^k}^T s^k/n \min(\alpha_x^1, \alpha_s^1) & \text{otherwise.} \end{cases} \tag{2.3}$$

The centering parameter μ computed from (2.3) is used in (2.2) for computing the second derivative. This derivative is obtained by solving

$$H\begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = \begin{pmatrix} 2\mu {X^k}^{-1} e - 2\dot{X}\dot{s} \\ 0 \end{pmatrix},$$

and $\ddot{s}(1) = -A^T\ddot{y}(1)$. The computations of first and second derivative require us to factor H once and use these factors to solve for two different right hand sides. The approach we used to compute the factors of H is described in Fourer and Mehrotra [4].

First and second derivatives are used in a Taylor polynomial to generate a step direction. The maximum step in the Taylor polynomial is first found by using

$$\begin{aligned} \alpha_x &= \min(1, \max\{\alpha \mid x(1) - \alpha\dot{x} + .5\alpha^2\ddot{x} \geq 0\}), \\ \alpha_s &= \min(1, \max\{\alpha \mid s(1) - \alpha\dot{s} + .5\alpha^2\ddot{s} \geq 0\}). \end{aligned}$$

The step direction d_x, d_y, d_s is then computed from

$$\begin{aligned} d_x &= \alpha_x\dot{x} - \alpha_x^2\ddot{x}, \\ d_s &= \alpha_s\dot{s} - \alpha_s^2\ddot{s}, \\ d_y &= \alpha_s\dot{y} - \alpha_s^2\ddot{y}. \end{aligned}$$

The step length is computed by finding

$$\begin{aligned} l_x &\equiv \text{argmin}\{x_i^k/(d_x)_i \mid (d_x)_i > 0, i = 1, 2, \ldots n\}, \\ l_s &\equiv \text{argmin}\{s_i^k/(d_s)_i \mid (d_s)_i > 0, i = 1, 2, \ldots n\}, \end{aligned}$$

first, and then computing f_x, f_s such that

$$\begin{aligned} (x_{l_x}^k - f_x * (d_x)_{l_x})(s_{l_x}^k - f_s * (d_s)_{l_s}) = (x^k - d_x)^T(s^k - d_s)/n\gamma_a, \\ (x_{l_x}^k - f_x * (d_x)_{l_x})(s_{l_x}^k - f_s * (d_s)_{l_s}) = (x^k - d_x)^T(s^k - d_s)/n\gamma_a. \end{aligned}$$

Now we take

$$\begin{aligned} f_x &:= \max(f_x, \gamma_f), \\ f_s &:= \max(f_s, \gamma_f). \end{aligned}$$

$\gamma_f = .9$ and $\gamma_a = 1/(1 - \gamma_f)$ is used for all problems. The new iterate is obtained as

$$\begin{aligned} x^{k+1} &\leftarrow x^k - f_x d_x, \\ s^{k+1} &\leftarrow s^k - f_s d_s, \\ y^{k+1} &\leftarrow y^k - f_s d_y. \end{aligned}$$

Primal and dual starting solutions are generated by using the solutions of the least squares problems:

$$\begin{aligned} \min \quad & \|x\| \\ \text{s.t.} \quad & Ax = b, \end{aligned} \tag{2.4}$$

and

$$\begin{aligned} \min \quad & \|s\| \\ \text{s.t.} \quad & A^T\pi + s = c. \end{aligned} \tag{2.5}$$

Let $\tilde{x}$, and $\tilde{\pi}$, $\tilde{s}$ be the solution of (2.4) and (2.5) respectively. We take $\delta_x = \max(-1.5 * \min\{\tilde{x}_i\}, 0)$ and $\delta_s = \max(-1.5 * \min\{\tilde{s}_i\}, 0)$. Now we compute

$$\begin{aligned} \tilde{\delta}_x &= \delta_x + .5 * \frac{(\tilde{x}+\delta_x e)^T(\tilde{s}+\delta_s e)}{\sum_{i=1}^n (\tilde{s}_i+\delta_s)}, \\ \tilde{\delta}_s &= \delta_s + .5 * \frac{(\tilde{x}+\delta_x e)^T(\tilde{s}+\delta_s e)}{\sum_{i=1}^n (\tilde{x}_i+\delta_s)}, \end{aligned}$$

and take $\tilde{x} + \delta_x e, \tilde{\pi}, \tilde{s} + \delta_s e$ as primal and dual starting points respectively. The least-squares problems (2.4) and (2.5) can be solved by using the augmented matrix H.

3. Computational Results

For our experiments we used a full implementation (including problems with bounds) of the basic algorithm described in the previous section. The research code we used is written in FORTRAN 77. All computational tests were performed on a SPARC-1 Sun workstation. Computational tests were performed on 86 **netlib** problems which we solved successfully. Other remaining problems (*Pilot, Pilot87, Dfl001*) were sufficiently large for our current computing environment. Table 3.1 provides information on the size of these problems. Problems with bounds and ranges are marked with B and R respectively. The optimal objective values for simplex method given here are those reported by Bixby [2]. For all the runs terminated with an optimal face this value was matched to all digits by both primal and dual solutions.

A complementary solution pair x^* and s^* is declared optimal if it satisfies

$$\begin{array}{c} |c^T x^* - b^T y^*|/(1 + |b^T y^*|) \leq \epsilon^* \\ \|Ax^* - b\|/(1 + \|b\|) \leq \epsilon_x^* \\ \|A^T y^* + s^* - c\|/(1 + \|c\|) \leq \epsilon_s^*. \end{array} \tag{3.1}$$

$\epsilon^* = \epsilon_x^* = \epsilon_s^* = 10^{-12}$ was used for all the problems in all the runs.

Table 3.2 gives computational results that we have obtained under various settings. These results are discussed later in this section. The second column of this table gives iteration number at which we could identify (and verify) the optimal partition for the first time. For this run we generated a partition using (1.6) at every iteration and tried to generate a feasible interior point on the primal and dual faces defined by this partition.

In Mehrotra and Ye [16] after (3.1) was satisfied for $\epsilon^* = \epsilon_x^* = \epsilon_s^* = 10^{-8}$ we tried to find an optimal face at each iteration. Results from this approach are reported in Column 3. This column gives the iteration at which the optimal partition was found, and in () the number of attempts required to identify optimal partition using this approach.

Another run was made to record the superlinear convergence behavior. For this run we masked the option of identifying the optimal face. We recorded several quantities to observe the behavior of the algorithm. This included the step for the second order polynomial (α_x and α_s), and the duality gap at each iteration. In this case the algorithm was terminated if $|c^T x^* - b^T y^*|/(1+|b^T y^*|) \leq 10^{-14}$ was satisfied, or if the number of iterations was one more than the number of iteration at which the optimal face would have been identified for the first time. Columns 5 to 8 of Table 3.2 give $|c^T x^k - b^T y^k|/(1 + |b^T y^k|)$ recorded at four successive iterations. These are two iterations prior and one iteration after the iteration at which the optimal face can be identified for the first time. Actually, we ran the algorithm for an additional few iterations, but the results for several problems were contaminated with numerical errors, therefore we do not present them here.

Finally, Column 4 gives results with a criterion that tries to make use of the observed superlinear convergence behavior in deciding the iteration for identifying the optimal partition. For this purpose we generated a partition (and verified its optimality) if at iteration k the basic algorithm of previous section satisfied

$$\begin{array}{c} \alpha_x \geq .95, \\ \alpha_s \geq .95, \\ {x^{k+1}}^T s^{k+1} / {x^k}^T s^k \leq .01. \end{array} \tag{3.2}$$

In practice we observe $\alpha_x \to 1$ and $\alpha_s \to 1$ and ${x^{k+1}}^T s^{k+1} / {x^k}^T s^k \to 0$. This is similar to the observation of Zhang, Tapia and Dennis [25], which led to the basic assumptions for their theory.

We now discuss the computational results in Table 3.2.

The results clearly show that for several problems there is no clear evidence of superlinear (or fast linear) convergence till the iteration at which the optimal partition could be identified and verified for the first time. For example the average ratio of best-1/best-2 is 0.27. This hardly indicates superlinear convergence.

However, the average ratio best/best-1 is 0.05, and best+1/best is 0.02. The average ratio best+1/best is 0.0068 if we exclude problems *finnis, greenbea and nesm.* Clearly, at iteration best or best+1 algorithm has begun to converge quickly. This improvement in convergence rate is reflected by individual problems as well.

From these results we infer that in practice the superlinear convergence is a consequence of iterates having come sufficiently close to the optimal face.

Checking for an optimal partition at each iteration is not practical. In practice we must use some indicators to decide when to partition the variables. Interestingly, the criterion based on (3.1) has worked well on the large number of problems in *netlib*. However, the choice of eight digits of precision is arbitrary. For several problems there is no need to wait for eight digits of precision, while for others (e.g, *etamacro, nesm, pilot.we, scsd6*) eight digits of accuracy is not sufficient to verify optimality.

The results obtained from using (3.2) show a marked improvement in the number of attempts for optimal face for problems *etamacro* and *nesm*, and required one less iteration to identify the optimal partition for 15 problems. These are *boeing2, cycle, grow22, sc105, sc50a, sc50b, scorpion, scsd1, ship04l, ship08s, stair, standata, vtp.base, wood1p, woodw*. However it took an extra iteration for seven problems. These are *80bau3b, bnl1, d2q06c, etamacro, greenbeb, pilot.we, scsd6*. It took extra two iterations for *finnis* and *nesm*. More importantly it always identified the optimal partition within two iteration of the "best" iteration. The results indicate that (3.2) generally performs just as well or better than the "eight digit relative precision criterion" based on (3.1).

If we used (3.2), in most cases the optimal partition was identified and verified at the first attempt. However, for several problems we needed two and for *scsd6* we needed three attempts. This can happen because the algorithm may make large progress at solutions far from the optimal face.

Conclusions: Computational results show that fast convergence in primal-dual methods occur after solutions have come sufficiently close to the optimal face. As a result, we can jump to a point in the interior of the optimal face before benefiting from the fast convergence behavior. However, we have shown that a criterion based on fast convergence behavior can be developed to effectively decide when to partition variables.

Name	Rows	Cols	Nonzeros	BR	Objective
25fv47	822	1571	11127		5.5018458883e+3
80bau3b	2263	9799	29063	B	9.8722419241e+5
adlittle	57	97	465		2.2549496316e+5
afiro	28	32	88		-4.6475314286e+2
agg	489	163	2541		-3.5991767287e+7
agg2	517	302	4515		-2.0239252356e+7
agg3	517	302	4531		1.0312115935e+7
bandm	306	472	2659		-1.5862801845e+2
beaconfd	174	262	3476		3.3592485807e+4
blend	75	83	521		-3.0812149846e+1
bnl1	644	1175	6129		1.9776295615e+3
bnl2	2325	3489	16124		1.8112365404e+3
boeing1	351	384	3865	BR	-3.3521356751e+2
boeing2	167	143	1339	BR	-3.1501872802e+2
bore3d	234	315	1525	B	1.3730803942e+3
brandy	221	249	2150		1.5185098965e+3
capri	272	353	1786	B	2.6900129138e+3
cycle	1904	2857	21322	B	-5.2263930249e+0
czprob	930	3523	14173	B	2.1851966989e+6
d2q06c	2172	5167	35674		1.2278421081e+5
degen2	445	534	4449		-1.4351780000e+3
degen3	1504	1818	26230		-9.8729400000e+2
e226	224	282	2767		-1.8751929066e+1
etamacro	401	688	2489	B	-7.5571523337e+2
fffff800	525	854	6235		5.5567956482e+5
finnis	498	614	2714	B	1.7279106560e+5
fit1d	25	1026	14430	B	-9.1463780924e+3
fit1p	628	1677	10894	B	9.1463780924e+3
fit2d	26	10500	138018	B	-6.8464293294e+4
fit2p	3001	13525	60784	B	6.8464293294e+4
forplan	162	421	4916	BR	-6.6421896127e+2
ganges	1310	1681	7021	B	-1.0958573613e+2
gfrd-pnc	617	1092	3467	B	6.9022359995e+6
greenbea	2393	5405	31499	B	-7.2555248130e+6
greenbeb	2393	5405	31499	B	-4.3022602612e+6
grow15	301	645	5665	B	-1.0687094129e+8
grow22	441	946	8318	B	-1.6083433648e+8
grow7	141	301	2633	B	-4.7787811815e+7
israel	175	142	2358		-8.9664482186e+5
kb2	44	41	291	B	-1.7499001299e+3
lotfi	154	308	1086		-2.5264706062e+1
maros	847	1443	10006	B	-5.8063743701e+4
nesm	663	2923	13988	BR	1.4076036488e+7

Table 3.1: Problem Data Summary (A-N)

Name	Rows	Cols	Nonzeros	BR	Objective
perold	626	1376	6026	B	-9.3807552782e+3
pilot.ja	941	1988	14706	B	-6.1131364656e+3
pilot.we	723	2789	9218	B	-2.7201075328e+6
pilot4	411	1000	5145	B	-2.5811392589e+3
pilotnov	976	2172	13129	B	-4.4972761882e+3
recipe	92	180	752	B	-2.6661600000e+2
sc105	106	103	281		-5.2202061212e+1
sc205	206	203	552		-5.2202061212e+1
sc50a	51	48	131		-6.4575077059e+1
sc50b	51	48	119		-7.0000000000e+1
scagr25	472	500	2029		-1.4753433061e+7
scagr7	130	140	553		-2.3313898243e+7
scfxm1	331	457	2612		1.8416759028e+4
scfxm2	661	914	5229		3.6660261565e+4
scfxm3	991	1371	7846		5.4901254550e+4
scorpion	389	358	1708		1.8781248227e+3
scrs8	491	1169	4029		9.0429695380e+2
scsd1	78	760	3148		8.6666666743e+1
scsd6	148	1350	5666		5.0500000078e+1
scsd8	398	2750	11334		9.0499999993e+2
sctap1	301	480	2052		1.4122500000e+3
sctap2	1091	1880	8124		1.7248071429e+3
sctap3	1481	2480	10734		1.4240000000e+3
seba	516	1028	4874	BR	1.5711600000e+4
share1b	118	225	1182		-7.6589318579e+4
share2b	97	79	730		-4.1573224074e+2
shell	537	1775	4900	B	1.2088253460e+9
ship04l	403	2118	8450		1.7933245380e+7
ship04s	403	1458	5810		1.7987147004e+6
ship08l	779	4283	17085		1.9090552114e+6
ship08s	779	2387	9501		1.9200982105e+6
ship12l	1152	5427	21597		1.4701879193e+6
ship12s	1152	2763	10941		1.4892361344e+6
sierra	1228	2036	9252	B	1.5394362184e+7
stair	357	467	3857	B	-2.5126695119e+2
standata	360	1075	3038	B	1.2576995000e+3
standmps	468	1075	3686	B	1.4060175000e+3
stocfor1	118	111	474		-4.1131976219e+4
stocfor2	2158	2031	9492		-3.9024408538e+4
tuff	334	587	4523	B	2.9214776509e-1
vtp.base	199	203	914	B	1.2983146246e+5
wood1p	245	2594	70216		1.4429024116e+0
woodw	1099	8405	37478		1.3044763331e+0

Table 3.1: Problem Data Summary (P-W)

Name	best	8(t)	sup(t)	best-2	best-1	best	best+1
25fv47	25	26(1)	26(1)	7.5e-05	2.1e-05	4.5e-07	4.2e-12
80bau3b	45	45(2)	46(1)	1.0e-08	7.9e-10	2.4e-11	5.4e-15
Adlittle	9	10(1)	10(1)	1.5e-03	2.2e-04	7.7e-06	8.1e-10
Afiro	5	7(1)	7(1)	7.0e-01	1.9e-01	3.5e-02	6.3e-04
Agg	25	26(1)	26(1)	2.3e-04	1.7e-05	6.8e-07	2.1e-11
Agg2	22	23(1)	23(1)	9.3e-05	1.8e-05	9.4e-07	1.3e-10
Agg3	21	21(2)	21(2)	8.5e-06	2.2e-09	4.3e-15	
Bandm	18	18(2)	18(2)	6.1e-07	3.1e-11	2.4e-14	
Beaconfd	6	7(1)	7(1)	2.8e-03	4.6e-04	3.2e-05	6.2e-09
Blend	9	10(1)	10(1)	5.8e-03	6.6e-04	8.9e-06	9.0e-12
Bnl1	29	29(1)	30(1)	4.7e-06	4.6e-07	8.8e-09	8.2e-12
Bnl2	36	37(1)	37(1)	6.3e-06	5.1e-07	3.1e-08	1.4e-10
Boeing1	25	26(1)	26(1)	2.3e-05	3.3e-06	5.3e-08	3.9e-13
Boeing2	17	19(1)	18(1)	7.7e-03	5.0e-04	3.9e-05	5.3e-08
Bore3d	17	18(1)	18(1)	1.2e-02	9.1e-04	3.5e-05	2.9e-09
Brandy	18	19(1)	19(1)	7.1e-05	1.1e-05	1.2e-07	3.9e-13
Capri	18	19(1)	19(1)	1.7e-03	2.5e-04	5.4e-06	1.2e-11
Cycle	29	30(1)	29(1)	2.3e-03	2.1e-04	7.2e-07	2.2e-14
Czprob	35	35(1)	35(1)	2.5e-06	5.1e-08	6.5e-14	1.2e-15
D2q06c	29	30(1)	31(1)	6.9e-06	1.5e-06	2.1e-07	5.4e-09
Degen2	11	12(1)	12(1)	5.1e-04	1.4e-04	1.8e-06	5.4e-12
Degen3	15	16(1)	16(1)	2.1e-04	6.7e-05	2.0e-06	4.0e-10
E226	20	20(1)	20(1)	3.4e-06	4.6e-07	5.7e-12	4.6e-15
Etamacro	33	33(4)	34(1)	3.0e-09	1.1e-09	1.3e-10	7.5e-14
Fffff800	38	38(1)	38(1)	1.3e-07	4.5e-08	3.1e-10	2.7e-15
Finnis	26	26(2)	28(1)	1.4e-08	3.4e-09	8.1e-10	2.2e-10
Fit1d	18	18(1)	18(1)	5.4e-06	1.3e-06	1.6e-09	1.1e-14
Fit1p	18	18(1)	18(1)	1.7e-06	2.1e-07	1.6e-11	5.6e-14
Fit2d	22	23(1)	23(1)	3.2e-05	3.5e-06	5.9e-07	3.9e-09
Fit2p	20	20(1)	20(1)	1.4e-06	4.3e-07	2.7e-09	1.9e-15
Forplan	22	24(1)	24(1)	5.6e-04	2.1e-05	4.4e-06	6.5e-08
Ganges	18	18(1)	18(1)	1.3e-06	3.3e-07	5.7e-10	7.9e-12
Gfrd-pnc	15	16(1)	16(1)	6.2e-04	1.7e-04	2.8e-06	4.4e-11
Greenbea	41	41(1)	41(1)	8.3e-08	1.5e-08	3.2e-10	2.0e-10
Greenbeb	42	42(2)	43(1)	9.5e-10	6.5e-11	1.3e-14	
Grow15	10	11(1)	11(1)	1.4e-03	2.5e-04	1.6e-05	5.2e-10
Grow22	12	13(1)	12(1)	2.2e-04	9.6e-05	3.6e-07	8.8e-15
Grow7	10	11(1)	11(1)	1.7e-03	1.7e-04	5.6e-06	2.9e-10
Israel	25	25(2)	25(1)	7.3e-08	2.0e-09	4.0e-13	1.4e-15
Kb2	17	19(1)	19(1)	2.7e-04	2.8e-04	4.4e-05	2.4e-06
Lotfi	13	14(1)	14(1)	7.7e-03	5.8e-04	1.2e-05	3.5e-12
Maros	26	27(1)	27(1)	1.0e-05	1.5e-06	2.2e-08	2.0e-12
Nesm	36	36(5)	38(1)	7.9e-11	2.6e-11	7.6e-12	6.1e-12

Table 3.2: Results for facet finding problem (A-N)

Name	best	8(t)	sup(t)	best-2	best-1	best	best+1
Perold	30	32(1)	32(1)	2.1e-05	3.5e-06	3.8e-07	1.1e-08
Pilot.ja	47	48(1)	48(1)	3.9e-07	1.0e-07	1.8e-08	2.4e-11
Pilotnov	24	24(1)	24(1)	2.7e-05	2.7e-07	6.0e-13	4.0e-15
Pilot.we	51	51(3)	52(1)	8.5e-09	9.0e-11	2.5e-11	8.7e-14
Pilot4	43	43(2)	43(1)	3.3e-08	3.5e-09	3.3e-13	*
Recipe	10	11(1)	11(1)	2.5e-03	7.5e-04	2.5e-05	4.0e-10
Sc105	7	9(1)	8(1)	2.8e-02	4.8e-03	3.1e-04	1.5e-06
Sc205	10	11(1)	11(1)	2.9e-03	1.5e-03	6.7e-05	3.3e-09
Sc50a	5	8(1)	7(1)	1.9e-01	4.7e-02	1.5e-02	6.1e-04
Sc50b	4	7(1)	6(1)	6.3e-01	2.6e-01	5.0e-02	1.1e-03
Scagr25	16	17(1)	17(1)	7.5e-05	1.2e-05	3.8e-07	7.2e-12
Scagr7	13	13(1)	13(1)	4.5e-06	1.0e-07	2.0e-11	3.9e-16
Scfxm1	16	18(1)	18(1)	4.9e-04	1.2e-04	1.5e-05	2.9e-07
Scfxm2	19	20(1)	20(1)	1.4e-04	2.5e-05	1.9e-06	3.6e-10
Scfxm3	19	20(1)	20(1)	1.6e-04	3.0e-05	2.7e-06	2.6e-09
Scorpion	10	12(1)	11(1)	2.3e-03	7.8e-04	1.2e-04	7.2e-07
Scrs8	20	21(1)	21(1)	3.3e-04	2.2e-05	3.3e-07	4.6e-13
Scsd1	6	8(1)	7(1)	2.0e-01	7.3e-02	3.9e-03	7.0e-07
Scsd6	12	12(3)	13(3)	2.5e-10	1.9e-10	8.8e-12	5.9e-15
Scsd8	8	9(1)	9(1)	4.7e-03	1.8e-04	3.2e-09	7.7e-15
Sctap1	15	15(1)	15(2)	2.8e-04	3.3e-08	3.0e-15	
Sctap2	14	14(2)	14(2)	5.2e-05	1.4e-11	9.2e-16	
Sctap3	14	14(1)	14(2)	7.4e-03	2.0e-05	4.4e-13	2.2e-15
Seba	17	18(1)	18(1)	7.8e-03	6.1e-04	1.8e-05	1.8e-10
Share1b	22	22(1)	22(2)	4.7e-06	1.0e-08	1.8e-12	1.8e-16
Share2b	11	12(1)	12(1)	6.2e-04	8.5e-04	1.1e-05	1.5e-10
Shell	19	20(1)	20(1)	1.4e-04	2.6e-05	2.1e-06	1.3e-10
Ship04l	10	12(1)	11(1)	1.4e-03	3.0e-04	5.4e-05	1.2e-07
Ship04s	13	13(1)	13(1)	5.7e-06	6.5e-07	5.6e-11	6.3e-12
Ship08l	14	15(1)	15(1)	3.8e-05	1.6e-06	3.1e-08	3.3e-13
Ship08s	13	14(1)	13(1)	1.8e-04	1.1e-05	4.7e-08	5.4e-15
Ship12l	16	18(1)	18(1)	2.0e-04	7.1e-05	7.8e-06	9.7e-07
Ship12s	15	16(1)	16(1)	2.0e-04	5.7e-05	3.3e-06	1.5e-09
Sierra	19	20(1)	20(1)	1.2e-05	2.2e-06	2.9e-07	8.8e-10
Stair	15	16(1)	15(1)	2.5e-04	7.3e-05	5.8e-07	4.7e-10
Standata	12	14(1)	13(1)	3.9e-02	3.1e-04	1.1e-10	2.1e-15
Standmps	21	22(1)	21(1)	1.1e-03	4.0e-04	4.8e-06	2.8e-12
Stocfor1	15	16(1)	16(1)	4.7e-03	1.2e-03	2.4e-05	1.0e-10
Stocfor2	23	24(1)	24(1)	3.3e-04	8.0e-05	1.3e-06	1.4e-11
Tuff	20	20(1)	20(2)	1.2e-06	1.8e-08	3.7e-12	4.4e-14
Vtp.base	24	25(1)	24(1)	1.4e-02	2.9e-03	1.0e-05	2.7e-13
Wood1p	30	31(1)	30(1)	6.3e-03	6.5e-05	1.5e-08	2.3e-12
Woodw	39	40(1)	39(1)	2.3e-05	4.2e-06	1.6e-08	3.8e-12

* results not given because of significant numerical errors.

Table 3.2: Results for facet finding problem (P-W)

References

[1] I. Adler and R. C. Monteiro (1989). "A geometric view of parametric linear programming," manuscript, Department of Industrial Engineering and Operations Research, University of California, Berkeley, California.

[2] R. Bixby (1990). "Implementing the Simplex Method: The Initial Basis," Technical Report, TR90-32, Rice University, Huston, Texas.

[3] A. S. El-Bakry, R. A. Tapia and Y. Zhang (1991). "A study of indicators for identifying zero variables in interior-point methods," TR 91-15, Department of Mathematical Sciences, Rice University (Houston, TX, 1991).

[4] R. Fourer and S. Mehrotra (1991). "Performance of an Augmented System Approach for Solving Least-Squares Problems in an Interior-Point Method for Linear Programming," Technical Report (in preparation), Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208.

[5] R. M. Freund, R. Roundy and M. J. Todd (1985). "Identifying the set of always active constraints in a system of linear inequalities by a single linear program," Technical Report, Sloan W.P. No. 1674-85, MIT, MA.

[6] D. M. Gay (1985). "Electronic mail distribution of linear programming test problems," Mathematical Programming Society News Letter, 10-12.

[7] A. J. Goldman and A. W. Tucker (1956). "Theory of linear programming," in: *Linear Inequalities and Related Systems*, H. W. Kuhn and A. W. Tucker, eds., Princeton University Press, Princeton, New Jersey, pp. 53-97.

[8] C. C. Gonzaga (1989). "An algorithm for solving linear programming problems in $O(n^3L)$ operations," in: N. Megiddo, ed., *Progress in Mathematical Programming - Interior Point and Related Methods* (Springer-Verlag, New York, 1989) pp. 1-28.

[9] O. Güler and Y. Ye (1991). "Convergence behavior of some interior-point algorithms," Working Paper 91-4, The College of Business Administration, The University of Iowa (Iowa City, IA, 1991).

[10] M. Kojima, S. Mizuno and A. Yoshise (1989). "A polynomial-time algorithm for a class of linear complementarity problems," *Mathematical Programming* **44** 1-26.

[11] I.J. Lustig, R.E. Marsten and D.F. Shanno (1989). "Computational Experience with a Primal-Dual Interior Point Method for Linear Programming," TR J-89-11, Industrial and Systems Engineering Report Series, Gerogia Institute of Technology, Atlanta, GA 30332.

[12] K. A. McShane, C. L. Monma and D. F. Shanno (1989). "An implementation of a primal-dual interior point method for linear programming," *ORSA Journal on Computing* **1** 70-83.

[13] S. Mehrotra (1990). "In the implementation of a (primal-dual) interior point method," TR90-03, Dept. of IE/MS, Northwestern University, Evanston, IL 60208.

[14] S. Mehrotra (1990). "Higher order methods and their Performance," TR90-16R1, Dept. of IE/MS, Northwestern University, Evanston, IL 60208.

[15] S. Mehrotra (1991). "Handling free variables in interior methods." Technical Report 91-06, Department of Industrial Engineering and Management Sciences, Northwestern University.

[16] S. Mehrotra and Y. Ye (1991). "On Finding the Optimal Facet of Linear Programs." Technical Report 91-10, Department of Industrial Engineering and Management Sciences, Northwestern University.

[17] S. Mizuno, M. J. Todd, and Y. Ye (1990). "On adaptive-step primal-dual interior-point algorithms for linear programming," Technical Report No. 944, School of ORIE, Cornell University, Ithaca, NY.

[18] R. C. Monteiro and I. Adler (1989). "Interior path following primal-dual algorithms. Part I: Linear programming," *Mathematical Programming* **44** 27-41.

[19] J. Renegar (1988). "A polynomial-time algorithm, based on Newton's method, for linear programming," *Mathematical Programming* **40** 59-93.

[20] A. Schrijver (1986). *Theory of Linear and Integer Programming*, John Wiley & Sons, New York.

[21] M. J. Todd (1990). "A low complexity interior-point algorithm for linear programming," Technical Report No. 903, School of ORIE, Cornell University, Ithaca, NY.

[22] P. M. Vaidya (1990). "An algorithm for linear programming which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations," *Mathematical Programming* **47** 175-202.

[23] Y. Ye (1988). "An $O(n^3L)$ potential reduction algorithm for linear programming," manuscript to appear in *Mathematical Programming*.

[24] Y. Ye (1991). "On the finite convergence of interior-point algorithms for linear programming," Working Paper 91-5, The College of Business Administration, The University of Iowa (Iowa City, IA, 1991).

[25] Y. Zhang, R. A. Tapia and J. E. Dennis (1990). "On the superlinear and quadratic convergence of primal-dual interior-point linear programming algorithms," TR90-6, Department of Mathematical Sciences, Rice University, Houston, TX.

[26] Y. Zhang, R. A. Tapia and F Potra (1990). "On the superlinear and quadratic convergence of interior point algorithms for a general class of problems," TR90-9, Department of Mathematical Sciences, Rice University, Houston, TX.

[27] Y. Zhang, R. A. Tapia (1990). "A Quadratically Convergent Polynomial Primal-Dual Interior-Point Algorithm for Linear Programming," TR90-40, Department of Mathematical Sciences, Rice University, Houston, TX.