matrix consumes the majority of the CPU time. Thus our goal is to use the factorization to compute several independent directions, and to determine the "best" possible combination based on these directions. Viewed in this way, our strategy is similar in spirit to the "predictor-corrector" methods of Mehrŏtra [Meh89].

Our multidimensional algorithm for LP can be thought of as optimizing over a low-dimensional subspace at each major iteration. We formalize it as follows.

Multidimensional Algorithm

1. Compute a strictly feasible point, u^0; set i=0.
2. At u^i, generate q independent directions, s^j, $j = 1, \dots, q$.
3. Form and solve the subproblem

$$\begin{aligned} \min_{\zeta} c^{\mathsf{T}} &\left(u^i + \sum_{j=1}^{q} \zeta_j s^j \right) \\ \text{subject to: } A &\left(u^i + \sum_{j=1}^{q} \zeta_j s^j \right) \le b \end{aligned}$$

   to obtain ζ^*.

4. Set $u^{i+1} := u^i + \alpha \sum_{j=1}^{q} \zeta_j^* s^j$ where α is the steplength.
5. If convergence then
   Stop;
   Else, set $i := i + 1$; Go to 2.

Note that the subproblem in Step 3 is equivalent to

$$\begin{aligned} &\min_{\zeta} \sum_{j=1}^{q} \zeta_j c^{\mathsf{T}} s^j \\ \text{subject to: } &\sum_{j=1}^{q} \zeta_j A s^j \le b - Au^i. \end{aligned}$$

The quantities $c^{\mathsf{T}} s^j$, As^j, and $b - Au^i$ need only be computed once and thus this low-dimensional problem can be solved efficiently. We comment further on its solution in §4. The steplength α in step 4 is fixed at .99, a value in accordance with much of the work in interior-point methods for LP. (See §4 for details.)

The extension to QP of this general procedure is obvious. The subproblem objective function includes the quadratic term, i.e.,

$$\begin{aligned} &\min_{\zeta} \tilde{c}^{\mathsf{T}} \zeta + \tfrac{1}{2} \zeta^{\mathsf{T}} \tilde{Q}_q \zeta \\ &\text{subject to: } \; Au \le b \end{aligned} \tag{2.1}$$

where

$$\tilde{c}_j = c^{\mathsf{T}} s^j + (u^i)^{\mathsf{T}} Q s^j$$

and $\tilde{Q}_q \in \Re^{q \times q}$ is defined analogously.

In our work to date, our algorithm optimizes over a 3-dimensional subspace, and hence we designate the method by *O3D*. The subproblem solution in Step 3 has three variables and m constraints. In the LP case, we can efficiently solve this problem using a specialized, revised dual simplex procedure. In the QP case, this does not appear to be a viable option, and the QP subproblem is solved by a simplified interior-point method, the details of which are in §4.

3. Directions

The efficacy of the multidimensional algorithm depends critically on the directions that generate the subproblem. In this section, we derive our directions from the method of centers. To do this, we first present some notation.

Define the residuals for the constraints to be

$$r_k(u, t) = b_k - A_k u, \qquad k = 1, \dots, m$$

where A_k is the kth row of A. Define the residual for the objective function to be

$$r_0(u, t) = t - c^{\mathsf{T}} u - \tfrac{1}{2} u^{\mathsf{T}} Q u$$

where t is a scalar whose value is determined as follows. Let u^0 be strictly feasible and let t^0 be the value of the objective function at u^0. Note that $r_k(u^0, t^0) > 0$, $k = 1, \dots, m$ and that $r_0(u^0, t) > 0$ for $t > t^0$. Thus

$$\prod_{k=0}^{m} r_k(u, t^0)$$

is positive in the interior of the feasible region that corresponds to points where the objective function is less than t^0. The *center* can then be defined as the maximum of this product, or, equivalently,

$$\min_u L(u, t^0) = \min_u \sum_{k=0}^{m} \log r_k(u, t^0) \tag{3.1}$$

which defines $L(u,t)$. The *method of centers* is to solve (3.1) to obtain u^1, set t^1 equal to the objective function value at u^1, and repeat. It can be shown that this yields a sequence of points that converges to an optimal solution.

In [DBRW91] we show that if the constraint on the objective function is moved continuously then a trajectory is formed that approaches an optimal solution. More importantly, by a slight generalization, it can be shown that there exists a trajectory connecting every strictly feasible point to an optimal solution. The vector field defined by these directions is given by

$$s^1 = \nabla_{uu}L(u,t)^{-1}\nabla_{ut}L(u,t).$$

Our first direction, therefore, is s^1, known in the literature as the *dual-affine* direction [ARV89]. After some algebraic simplifications we have that s^1 is

$$s^1 = \beta_1 H^{-1}(c + Qu) \tag{3.2}$$

where

$$H = A^{\mathsf{T}} D^2 A + \frac{Q}{r_0(u,t)},$$

D is a diagonal matrix with kth diagonal entry $1/r_k(u,t)$, and β_1 is a scalar.

It is likewise possible to compute the so-called *recentering* direction, i.e., the Newton direction for solving (3.1) given by

$$\nabla_{uu}L(u,t)^{-1}\nabla_{u}L(u,t).$$

This direction is a combination of two directions, the first of which is s^1 and the second of which is our second direction, s^2, given by

$$s^2 = \beta_2 H^{-1} A^{\mathsf{T}} De \tag{3.3}$$

where e is the vector of all ones, and β_2 is a scalar.

The third direction is derived from considering the rank one effect on the Hessian matrix, H, of the first constraint, whose index is denoted by $\tilde{k}$, encountered in the s^1 direction. It is easily shown that this change is dominated by the vector $A_{\tilde{k}}$, and that the resulting direction is

$$s^3 = H^{-1} A_{\tilde{k}}^{\mathsf{T}}. \tag{3.4}$$

Finally, using the notation of factorable functions [JM86], we obtain our fourth direction, a third order correction to s^1, namely

$$s^4 = H^{-1} \sum_{k=1}^{m} A_k^{\mathsf{T}} \left[\frac{1}{r_k(u,t)}\right]^3 (A_k s^1)^2. \tag{3.5}$$

and the quadratic programming (QP) is

$$\begin{array}{l} \min\limits_{u} c^{\mathsf{T}} u + \tfrac{1}{2} u^{\mathsf{T}} Q u \\ \text{subject to: } \; Au \le b \end{array} \tag{1.2}$$

where $u, c \in \Re^n$, $Q \in \Re^{n \times n}$, $A \in \Re^{m \times n}$, and $b \in \Re^m$. We assume that Q is symmetric and positive semi-definite, i.e., we allow the possibility that some variables in the quadratic program enter only linearly. In addition we make the standard assumptions that the problems have a finite optimal solution and that A has full column rank. We do *not* assume that the problem has a full dimensional interior, nor do we assume that the problem is nondegenerate.

Interior point methods have been extended to the quadratic programming problem by several authors. For example, Shanno, *et al.* [Sha91] have been working on a primal-dual method, and Anstreicher has suggested a log barrier approach [AdHRT90] and a related dual version [Ans90] that is similar to our approach, although his work is primarily theoretical. Our work is based on the method of centers originally proposed by Huard [Hua67]. We believe that this framework provides some advantages over the primal-dual approach. In particular, as will be shown below, the Hessian matrix that is required involves the sum of Q and a matrix of the form $A^{\mathsf{T}}A$ whereas in the primal-dual framework the Hessian involves a matrix of the form $A^{\mathsf{T}}Q^{-1}A$. Clearly, even if Q is sparse, the inverse will generally not be, and thus this latter form is much more likely to be dense. Also, since the method of centers was originally proposed for general convex programs, the directions we compute are natural extensions of those for the linear programming case and can be computed at a low relative cost.

In §2 we present a brief description of the multi-dimensional method that forms the basis both for our linear programming work and for the quadratic extensions. In §3 we discuss the generation of the directions, in §4 we describe the details of our implementation, and in §5 we give a summary of our preliminary numerical results. These results indicate that the proposed algorithm can solve the quadratic programming problems in approximately the same time as that required for the linear problems. Finally, in §6 we note some directions for future research.

2. Multidimensional Methods

Our strategy for solving linear programming problems by interior-point methods is based on the observation that the formation and factorization of the Hessian

An Interior-Point Method for Linear and Quadratic Programming Problems *

Paul T. Boggs [†] Paul D. Domich [‡] Janet E. Rogers [§]
Christoph Witzgall [¶]

April 1, 1991

1. Introduction

We have been working on a particular class of interior point methods for solving linear programming problems for several years. (See, e.g., [DBRW91].) Our methods combine several search directions that are readily computed at each iteration. The final step is then calculated by computing the step that solves the original problem restricted to the subspace spanned by these search directions. In this paper we propose an extension of these ideas to the case of convex quadratic programming.

The linear programming (LP) problem that we consider is

$$\begin{array}{l} \min\limits_u c^{\mathsf{T}} u \\ \text{subject to:} \quad Au \leq b \end{array} \qquad (1.1)$$

*Contribution of the National Institute of Standards and Technology and not subject to copyright in the United States. This research was supported in part by ONR Contract N-0014-87-F0053.

[†]Applied and Computational Mathematics Division, National Institute of Standards and Technology, Gaithersburg, MD 20899. INTERNET: boggs@cam.nist.gov

[‡]Applied and Computational Mathematics Division, National Institute of Standards and Technology, Boulder, CO 80303-3328. INTERNET: domich@cam.nist.gov

[§]Applied and Computational Mathematics Division, National Institute of Standards and Technology, Boulder, CO 80303-3328. INTERNET: jrogers@cam.nist.gov

[¶]Applied and Computational Mathematics Division, National Institute of Standards and Technology, Gaithersburg, MD 20899. INTERNET: witzgall@cam.nist.gov

The details for all of these directions may be found in [DBRW91]. Our strategy, outlined below, is to choose three of these four directions at each iteration.

The only difference between the formulas here for the quadratic programming problem and those for the linear programming problem is the appearance of the Q term. One can readily observe that the sparsity of H is decreased by the addition of Q, but not catastrophically, as it is in methods such as those mentioned in §1. The work per iteration, therefore, is approximately the same for the quadratic program as for the linear program.

4. Implementation Details

In the results reported here, we use the basic *O3D* algorithm presented in §2 with the three directions, s^1, s^4, and either s^2 or s^3, to specify the subproblem. The selection of the third direction is based on the proximity to the optimal vertex. The implementation uses s^2 in early iterations, and s^3 in the final iterations. The "switch-over" is performed when the duality gap (see below) is less than or equal to 10^{-4} or the residual on the objective function (c.f., §3) is less than or equal to $1/m$. The main procedure continues until it satisfies at least one of three convergence criteria: (a) the relative change in the objective function, (b) the relative difference between the primal and dual objective values (see, e.g., [ARV89]), and (c) the steplength.

Note that computation of the dual variables is theoretically more complicated in the QP case. Our preliminary work, however, simply extends the techniques we used for the LP case, using $y = D^2A(A^{\mathsf{T}}D^2A+Q/r_0)^{-1}(c+Qu)$ to approximate a dual feasible solution, provided that $y \geq 0$. It can be shown that this is guaranteed to yield a dual feasible point in the limit, and this appears to be working well in practice.

The subproblem is solved by using an interior point approach on the three dimensional subspace. At each iteration of the subproblem, a dual affine direction (3.2) is computed and a step is taken either a fixed percentage of the distance to the boundary, or a distance which minimizes the objective function in that direction, whichever is smaller.

Problem scaling, starting values and the phase 1 procedure are exactly the same as used for our earlier LP subproblem work. In particular, both A and the subproblem constraint matrix defined in (1.2) are scaled. Our algorithm is initialized by setting $u_0 = 0$, where 0 denotes the 0-vector of the appropriate dimension, and then taking a single recentering step using a quadratic model in the steepest descent direction, $A^T De$. When necessary, an initial feasible solution is obtained using a big-M Phase 1 procedure (see, e.g., [BJ77] and [DBRW91]).

The A matrix is stored in sparse format using the XMP experimental mathematical programming data structures described in [Mar81]. The Hessian matrix $(A^T D^2 A + Q/r_0)$ is stored using the data structures from the Yale Sparse Matrix Package SMPAK [SMP85]. A minimum-local-fill ordering procedure [VS83] is invoked only once at the beginning of the procedure to find a permutation of the rows and columns that reduces fill-in. The Hessian is then factored and solved using the Yale Sparse Matrix Package SMPAK [SMP85]. Constraints that are sufficiently far from the current point u, are explicitly removed from the computations.

The methods reported here are implemented in Fortran 77 and executed in double precision on an IBM RISC System/6000 Model 520 workstation running at 20MHz using the *xlf* compiler with the -O option.

5. Results

We create a set of QP test problems by augmenting 83 of the linear programming problems (publicly available through Netlib [Gay85]) with a quadratic term using $Q = I$. This allows an easy comparison of the work required to solve the QP problems with that required for the LP problems. Also, in the sense that any strictly convex QP can be transformed into one with $Q = I$, this can be regarded as a general formulation.

The convergence conditions and the corresponding total CPU time required for these QP problems are nearly identical to the LP results reported in [DBRW91]. For the 83 problems, 40 terminated due to the convergence criterion based on the relative difference between the primal and dual objective values. Of the remaining 43 problems that terminated due to the objective function convergence criterion, all but 3 found a point reasonably close (but not close enough for duality gap convergence) to the dual objective to indicate that the problem had solved correctly. Because the "true" objective function values are not readily available for these QP problems, however, we did not verify that the correct objective value was found. The total CPU time for the 83 QP problems is 2637 seconds, only 50 seconds more than the time required for our best LP results.

	Augmented		*Normal*	
problem	it	time	it	time
25fv47	26	259.84	26	164.66
80bau3b	48	814.16	46	444.44
adlittle	10	1.20	10	.82
afiro	7	.29	7	.23
agg	25	15.12	25	33.83
agg2	22	101.61	22	71.15
agg3	20	90.35	20	77.88
bandm	17	11.53	17	8.36
beaconfd	7	4.04	7	2.55
blend	10	1.53	10	1.38
bnl1	29	82.94	29	54.89
bnl2	36	1092.99	36	1113.57
boeing1	25	32.51	26	23.93
boeing2	19	8.26	19	6.84
bore3d	17	3.76	17	4.10
brandy	20	14.25	20	7.99
capri	20	16.11	31	22.54
cycle	31	354.44	24	370.79
czprob	36	73.93	35	42.83
d2q06c	30	1770.20	30	2967.94
degen2	12	57.93	12	34.70
degen3	16	1056.70	16	634.72
e226	20	19.21	20	10.08
etamacro	30	64.41	32	60.12
fffff800	37	104.11	38	87.80
finnis	26	24.87	25	23.07
fit1d	18	36.30	21	23.58
fit1p	18	33.58		∞
fit2d	23	447.15	24	266.52
fit2p	22	260.21		∞
forplan	24	22.68	23	16.03
ganges	18	60.73	19	92.23
gfrd-pnc	16	16.19	17	9.98
greenbea	40	439.75	39	(6)
greenbeb	40	385.11	36	(5)
grow15	12	21.59	12	13.82
grow22	13	37.60	14	21.92
grow7	12	9.81	12	6.27
israel	24	15.02	24	57.06
kb2	20	1.87	20	1.39
lotfi	14	4.54	14	3.43
nesm	34	170.54	35	152.00
perold	32	216.78	27	(3)

	Augmented		*Normal*	
problem	it	time	it	time
pilot	25	369.50	25	350.62
pilot.ja	44	665.84	34	(3)
pilot.we	51	225.83	45	(3)
pilot4	43	95.50	32	(3)
pilotnov	44	4505.81	43	3942.65
recipe	10	2.12	11	1.53
sc105	9	1.31	9	.89
sc205	11	3.00	11	2.15
sc50a	8	.60	8	.41
sc50b	6	.38	6	.33
scagr25	17	9.15	17	7.39
scagr7	13	1.97	13	1.55
scfxm1	18	16.14	18	10.84
scfxm2	20	36.26	20	24.49
scfxm3	20	52.48	20	38.29
scorpion	12	9.51	12	4.17
scrs8	21	24.10	21	16.69
scsd1	8	5.35	8	2.97
scsd6	10	11.92	10	6.10
scsd8	9	21.45	9	12.07
sctap1	15	9.55	15	5.83
sctap2	13	45.56	13	33.10
sctap3	14	55.92	14	42.28
seba	18	16.16	22	493.91
share1b	22	6.34	22	3.88
share2b	12	2.40	12	1.70
shell	20	24.90	20	15.57
ship04l	12	19.71	12	11.02
ship04s	13	18.89	13	7.77
ship08l	14	38.63	14	20.89
ship08s	14	21.39	14	10.94
ship12l	18	63.93	18	33.18
ship12s	16	27.83	16	15.14
sierra	21	125.67	22	55.41
stair	16	30.17	12	(2)
standata	13	12.73	14	8.86
standmps	20	23.72	26	16.58
stocfor1	16	2.49	16	1.92
stocfor2	24	93.51	24	77.79
tuff	20	31.39	19	23.39
vtp.base	18	2.66	19	2.51
wood1p	29	289.03	29	175.23
woodw	29	287.30	29	187.71

Iteration counts and timings for the augmented system and normal equation approaches. In the column for normal equation timings, ∞ indicates that the equations were too large and dense to solve on the computer used; (k) means that the algorithm failed after k digits of accuracy were achieved.

[7] I.S. Duff, N.I.M. Gould, J.K. Reid, J.A. Scott and K. Turner, "The factorization of sparse symmetric indefinite matrices." Report RAL-90-066, Rutherford Appleton Laboratory, Oxon (1990).

[8] I.S. Duff and J.K. Reid, "The multifrontal solution of indefinite sparse symmetric linear equations." *ACM Transactions on Mathematical Software* **9** (1983) 302–325.

[9] P.E. Gill, W. Murray, M.A. Saunders, J.A. Tomlin and M.H. Wright, "On projected Newton barrier methods for linear programming." *Mathematical Programming* **36** (1986) 183–209.

[10] M. Kojima, S. Mizuno and A. Yoshise, "A $O(\sqrt{n}L)$ potential reduction algorithm for linear complementarity problems." Research Report B-217, Department of Information Sciences, Tokyo Institute of Technology (1988).

[11] I. Lustig, R.E. Marsten and D.F. Shanno, "On Implementing Mehrotra's Predictor-Corrector Interior Point Method for Linear Programming." Technical Report SOR 90-03, Department of Civil Engineering and Operations Research, Princeton University (1990).

[12] I. Lustig, J.M. Mulvey and T.J. Carpenter, "Formulating Stochastic programs for interior point methods." Technical Report SOR 89-16, Department of Civil Engineering and Operations Research, Princeton University (1989).

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

[14] S. Mehrotra, "On the implementation of a (primal-dual) interior point method." Technical Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University (1990).

[15] S. Mehrotra, "Performance of higher order methods." Technical Report 90-16R, Department of Industrial Engineering and Management Sciences, Northwestern University (1990).

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

[17] R.C. Monteiro and I. Adler, "An $O(n^3L)$ primal-dual interior point algorithm for linear programming." *Mathematical Programming* **44** (1989) 43–66.

[18] R.D. Skeel, "Scaling for numerical stability in Gaussian elimination." *Journal of the Association for Computing Machinery* **26** (1979) 494–526.

[19] R.J. Vanderbei, "Splitting dense columns in sparse linear systems." Manuscript, AT&T Bell Laboratories, Murray Hill, NJ (1990).

[20] R.J. Vanderbei, "ALPO: another linear program solver." Manuscript, AT&T Bell Laboratories, Murray Hill, NJ (1990).

6. Future Directions

Our preliminary study demonstrates the feasibility of extending our multidimensional method, *O3D*, to the QP case. We intend to explore these ideas further by first attempting the problem set with a general positive semi-definite matrix Q. In these tests, we will not transform to $Q = I$ since we believe that transforming the problem to $Q = I$ will destroy too much of the sparsity. Next, although our subproblem solver has performed adequately in these preliminary tests, we think that some improvements are possible. Finally, we will investigate the use of this procedure in a sequential quadratic programming (SQP) algorithm for general large-scale nonlinear programming problems.

References

[AdHRT90] K. M. Anstreicher, D. den Hertog, C. Roos, and T. Terlaky. A long step barrier method for convex quadratic programming. Technical report, Delft University of Technology, Delft, Netherlands, 1990.

[Ans90] Kurt M. Anstreicher. On long step path following and sumt for linear and quadratic programming. Manuscript, Yale School of Organization and Management, August 1990.

[ARV89] Ilan Adler, Mauricio G. C. Resende, and Geraldo Veiga. An implementation of Karmarkar's algorithm for linear programming. *Mathematical Programming*, 44(3):297–335, 1989.

[BJ77] Mokhtar S. Bazaraa and John J. Jarvis. *Linear Programming and Network Flows*. John Wiley and Sons, Inc., New York, 1977.

[DBRW91] Paul D. Domich, Paul T. Boggs, Janet E. Rogers, and Christoph Witzgall. Optimizing over three-dimensional subspaces in an interior-point method for linear programming. *Linear Algebra and its Applications*, 152, July 1991.

[Gay85] David M. Gay. Electronic mail distribution of linear programming test problems. Mathematical Programming Society COAL Newsletter 13, December 1985.

[Hua67] Pierre Huard. Resolution of mathematical programming with nonlinear constraints by the method of centres. In J. Abadie, editor, *Nonlinear Programming*, pages 209–219. North Holland, Amsterdam, 1967.

[JM86] Richard H. F. Jackson and Garth P. McCormick. The polyadic structure of factorable function tensors with applications to higher-order minimization techniques. *Journal of Optimization Theory and Applications*, 51(1):63–93, 1986.

[Mar81] Roy E. Marsten. The design of the XMP linear programming library. *ACM Transactions on Mathematical Software*, 7(4):481–497, December 1981.

[Meh89] Sanjay Mehrotra. Implementations of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate gradient methods. Technical Report 89-04, Northwestern University, August 1989.

[Sha91] David F. Shanno, 1991. Personal communication.

[SMP85] *SMPAK User's Guide Version 1.0*, 1985.

[VS83] Jiri Vlach and Kishore Singhal. *Computer Methods for Circuit Analysis and Design.* Van Nostrand Reinhold Company, New York, 1983.

of digits of accuracy in the solution at that stage (in place of the timing). We believe that these failures are primarily due to a well-known numerical difficulty associated with the treatment of free variables (see, for example, [20]). Indeed, the augmented system version succeeded on problems *greenbea* and *greenbeb* only after a small number of free variables, which had been modeled as differences of nonnegative variables, were identified explicitly to the code.

From the standpoint of computational time, the augmented system approach achieves a predictably great advantage in solving those problems (*fit1p, fit2p, israel, seba*) that are well known to have dense columns. We also observe substantial improvement in another group of problems (*agg, capri, d2q06c, ganges*) whose dense columns are not so readily identified. For other problems the normal equations are acceptably sparse, and hence the augmented system is more expensive to solve; even so, performance is rarely worse by more than a factor of 2.

These preliminary results suggest that the augmented system approach offers a significant advantage in robustness of implementation for interior point methods, at an acceptable cost in speed for the easier problems.

Acknowledgements

This work has been supported in part by: National Science Foundation grants DDM-8908818 (Fourer) and CCR-8810107 (Mehrotra); and grants from GTE Laboratories (Mehrotra).

References

[1] I. Adler, N. Karmarkar, M.G.C. Resende and G. Veiga, "An implementation of Karmarkar's algorithm for linear programming." *Mathematical Programming* **44** (1989) 297–336.

[2] M. Arioli, I.S. Duff and P.P.M. de Rijk, "On the augmented system approach to sparse least-squares problems." *Numerische Mathematik* **55** (1989) 667–684.

[3] P.T. Boggs, P.D. Domich, J.R. Donaldson and C. Witzgall, "Optimal 3-dimensional methods for linear programming." MISTIR 89-4225, Center for Computing and Applied Mathematics, National Institute of Standards and Technology, Gaithersburg, MD (1989).

[4] A.A. Björck, "Iterative refinement of linear least-squares solutions." BIT **7** (1967) 257–278.

[5] J.R. Bunch and B.N. Parlett, "Direct methods for solving symmetric indefinite systems of linear equations." *SIAM Journal on Numerical Analysis* **8** (1971) 639–655.

[6] I.C. Choi, C.L. Monma and D.F. Shanno, "Further Development of a Primal-Dual Interior Point Method." *ORSA Journal on Computing* **2** (1990) 304–311.

Following the "Markowitz-b" approach of [7], we let n_i be the number of nonzero rows in column i of C (the *degree* of column i) and let $\hat{n}_{ij} = \min(n_i + n_j - 4, k - 2)$. For each kind of 2×2 pivot we define a "cost" as follows:

$$\begin{array}{ll} \textit{oxo} & (n_i - 1)(n_j - 1) \\ \textit{tile} & (n_i - 1)(\hat{n}_{ij} + 1) \quad \text{if } c_{ii} = 0 \\ & (n_j - 1)(\hat{n}_{ij} + 1) \quad \text{if } c_{jj} = 0 \\ \textit{full} & (\hat{n}_{ij})(\hat{n}_{ij}) \end{array}$$

We seek to find, in an approximate sense, the "sparsest" pivot that is numerically acceptable. For each $r = 1, 2, \ldots$, we first consider all 1×1 pivots of degree r, and then all 2×2 pivots of the form (2.2) whose cost is less than a certain threshold value. This threshold is taken to be $(r-1)^2$, $(r-1)(2r-3)$, or $(2r-4)^2$ for an *oxo*, *tile*, or *full* pivot, respectively.

Once a pivot order has been determined in this way, we try to re-use it at subsequent iterations of the interior point method. If it becomes unstable, then we generate a new pivot sequence.

Stability test

A 1×1 pivot c_{ii} is accepted if it satisfies the usual partial-pivoting condition,

$$|c_{ii}^{-1}| \cdot \|c_i\|_\infty \le \beta^{-1},$$

where $\|c_i\|_\infty$ denotes the largest element in row (or column) i of C. A 2×2 pivot (2.2) is accepted if it satisfies an analogous condition,

$$\left| \begin{pmatrix} c_{ii} & c_{ij} \\ c_{ji} & c_{jj} \end{pmatrix}^{-1} \right| \begin{pmatrix} \|c_i\|_\infty \\ \|c_j\|_\infty \end{pmatrix} \le \begin{pmatrix} \beta^{-1} \\ \beta^{-1} \end{pmatrix}.$$

In our tests, we initially take $\beta = 16^{-6}$, a value much lower than that used in [7]. If at any iteration the current β is inadequate for a numerically stable elimination, it is increased by a factor of 16.

3. Computational Results

Our tests use the primal-dual path-following algorithm [10, 17, 13, 11] as in the implementation described by Mehrotra [14], with the addition of facilities for bounded variables [15, Algorithm 2.1].

Timings are reported for both the original normal equations version of the implementation, and for a new version that incorporates the augmented system approach as described above. For the normal equations the pivot order is chosen by a minimum degree heuristic, which is the 1×1 analogue of the Markowitz-b approach. All times are in seconds, using the same Fortran compiler on a Sun-4/110.

The accompanying table gives iterations and timings for linear programs from the `netlib` test set. Whereas the augmented system approach achieves eight digits of accuracy in all cases, the normal equations approach sometimes fails; failures are indicated in the table by the iteration at which failure occurs, and the number

INTERIOR METHOD VS SIMPLEX METHOD: BEYOND NETLIB

Irvin J. Lustig
Department of Civil Engineering and Operations Research
Princeton University
Princeton, NJ 08544

Roy E. Marsten
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332

David F. Shanno
Rutgers Center for Operations Research
Rutgers University
New Brunswick, NJ 08903

March, 1991

abstract

An up-to-date comparison of the simplex and interior point methods for linear programming. The comparison is between the OSL simplex code and the OB1 interior point code, both run on an IBM RISC System/6000, model 530. The comparison uses a test set containing a few of the larger NETLIB models, and several large proprietary models.

The last few years have been a time of intense competition between the classical simplex method for linear programming and the newer interior point method. The simplex method, which only a few years ago seemed to be a mature and stable technology, has been dramatically improved. The OSL(simplex) code by John Forrest [11] is representative of the current state-of-the-art. The implementation of interior point methods has been developed from scratch by a number of researchers, with major improvements almost every month. See for example [1,2,3,6,7,9,10]. For a general introduction to interior point methods, see [8].

Most of the computational results that have appeared in the literature so far have used the NETLIB [4] test set of publically available LP models. While the NETLIB set has been an invaluable aid to code development and testing, the models it contains are of small to medium size and do not reveal the dramatic superiority of the interior method for large models. We would like to take this opportunity to begin reporting results that go beyond NETLIB, into the realm of truly large scale industrial LP models. The trend in the comparison between the simplex and interior methods as model size increases is unmistakeable.

Table 1 contains the dimensions of the models that we will present results for. We include some of the larger NETLIB models to provide a point of reference with earlier results. The models are ordered according to the number of non-zeros in the coefficient matrix, nz(A). We also include the number of non-zeros in the Cholesky factor, L, of A*A_transpose, as this is crucial to the performance of the interior method.

PILOT, PILOT87, 25FV47, 80BAU3B, and DFL001 are from NETLIB. The AMOCO and FORD models come from those companies. PLAN01, MFMPTE08, and MFMPTE16 come from the Shell Oil Company. The MFMPTE models are multi-period, single refinery models with 8 and 16 time periods, respectively. NCM and TRIP are from the U. S. Department of Energy. NCM is the National Coal Model and TRIP is the Transportation and Refining of International Petroleum model. DIAGMA is from the Diagma Corp. (France), REG530 is from the Interet Corp., PIMS2 is from Bechtel, and MARKAL is a global energy model sent to us by Haverly Systems. RAJA is from Prof. Roger Glassey of the University of California at Berkeley. OILCO is from an oil company that we are not free to identify at this time.

TABLE 1. TEST PROBLEMS

name	rows	columns	nz(A)	nz(L)
AMOCO	586	1,338	6,220	25,391
PLAN01	1,518	1,464	7,236	13,434
RAJA	1,400	1,500	7,811	46,995
25FV47	821	1,571	10,400	33,353
REG530	574	780	10,530	70,754
REG530(a)				13,026
PIMS2	2,601	2,366	16,082	88,612
80BAU3B	2,262	9,799	21,002	40,340
DIAGMA	4,287	8,290	25,459	142,844
AMOCO2	1,612	3,969	27,476	236,821
DFL001	6,071	12,230	35,632	1,634,257
PILOT	1,441	3,652	43,167	202,338
MFMPTE08	3,275	6,874	50,659	148,886
MARKAL	5,831	7,689	59,435	152,674
PILOT87	2,030	4,883	73,152	430,314
MFMPTE16	6,507	13,610	100,523	297,149
NCM	3,352	16,298	128,057	48,836
TRIP	8,335	21,200	161,160	452,848
FORD	43,387	106,908	189,864	188,149
OILCO	18,800	38,540	219,880	1,623,918

(a) The 35 columns of A with more than 100 non-zeros were removed, to give a sparser Cholesky factor, L.

The codes that we compared are OSL(simplex), Release 1.0, a product of the IBM Corporation [11], and the March, 1991, version of OB1, a product of XMP Software, Inc. The current version of OB1 differs

handles dense columns naturally as part of the elimination of the augmented system. As a bonus, this approach also provides for a natural and stable handling of free variables as proposed by Mehrotra [16].

2. The Augmented System Approach

Problem (1.1) can also be solved by considering the augmented system

$$C_\alpha \begin{pmatrix} u \\ w \end{pmatrix} \equiv \begin{pmatrix} \alpha I & DA^T \\ AD & 0 \end{pmatrix} \begin{pmatrix} u \\ w \end{pmatrix} = \begin{pmatrix} v \\ 0 \end{pmatrix}, \tag{2.1}$$

where I is an $n \times n$ identity matrix and α is the scaling constant recommended by Björck [4] to improve the condition number of C_α. In particular Skeel's condition number [18], $\kappa_S(C_\alpha) = \| \, |C_\alpha^{-1}| \, |C_\alpha| \, \|$, where $|C_\alpha|$ is a matrix of entries $|(C_\alpha)_{ij}|$, is minimized by some $\alpha \in [0, \|AD\|_\infty]$ (see [2]). Our tests use $\alpha = 16^{-2}\|AD\|_\infty$.

If the first n rows and columns are eliminated from C_α, then the above system reduces to the normal equations (1.2); a somewhat different ordering gives the so-called Schur complement modification [6]. One can hope to obtain a sparser and more stable factorization, however, by applying a more general elimination method to C_α. Since C_α is a symmetric indefinite matrix, it can be symmetrically factored by using 1×1 and 2×2 block pivots [5]; but in contrast to the positive definite case, a tentative pivot sequence obtained from the nonzero pattern of C_α may not be numerically stable.

Mehrotra [16] shows that, if one of the diagonal entries of αI in C_α is taken as zero rather than α, then the associated variable is free rather than nonnegative in the underlying linear program. Since C_α remains symmetric and positive semi-definite, this adjustment offers a natural approach to handling free variables that has no direct analogue in the normal equations or Schur complement approaches.

Duff *et al.* [7] have recently reported very encouraging results for solving sparse symmetric indefinite systems of the kinds just described. They consider a variety of strategies for promoting sparsity in a series of 1×1 and 2×2 pivots, and propose a new stability test. Although their work is still in an experimental stage, initial computational results show significant improvements over the earlier Harwell multifrontal code, MA27 [8].

The computational results that we present here were obtained by using our own implementation of the Duff *et al.* approach, adapted for use within an interior point code. We next briefly describe the selection criterion and stability test under this approach, as we have implemented them.

Selection criterion

Assume that at some stage of the elimination we have the reduced (symmetric) matrix $C \in \Re^{k \times k}$. We must next select either a 1×1 pivot $c_{ii} \neq 0$, or a 2×2 pivot

$$\begin{pmatrix} c_{ii} & c_{ij} \\ c_{ji} & c_{jj} \end{pmatrix} \tag{2.2}$$

such that $c_{ij} \neq 0$. Following the terminology in Duff *et al.* [7] we call a 2×2 pivot *oxo*, if $c_{ii} = c_{jj} = 0$; *tile*, if either c_{ii} or $c_{jj} = 0$, but not both; or *full*, if $c_{ii} \neq 0$ and $c_{jj} \neq 0$.

Performance of an Augmented System Approach for Solving Least-Squares Problems in an Interior-Point Method for Linear Programming

Robert Fourer and Sanjay Mehrotra

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

1. Introduction

Interior point methods for linear programming solve a weighted least-squares problem,

$$\min_w \|DAw - v\|, \tag{1.1}$$

to compute the search direction. Specifically, an iteration first generates a new weight matrix D, and solves (1.1) for one or more right hand sides v. The obtained solutions are combined to construct a search direction, and a step is taken to generate a new iterate and a new D.

It is well understood that solution of the least-squares problem represents a large part of the work of interior point methods. Many prior implementations [1, 3, 9, 11, 13, 14] first form the associated normal equations,

$$AD^2A^Tw = ADv, \tag{1.2}$$

and then compute the Cholesky factorization $LL^T = AD^2A^T$ by elimination. A major advantage of this approach is that all elimination pivot orders for finding L are stable. A computationally suitable pivot order can be determined (for example, by using a minimum degree or minimum local fill heuristic) at the outset, and can then be used for all iterations of the algorithm.

In this note we document our experience with an alternative, augmented system approach for solving the least-squares problem. Although this approach has been well studied in other contexts [2, 7], its suitability in the context of interior point methods is, to the best of our knowledge, still an open question. Our objective here is to quickly disseminate our results; a subsequent report will give detailed findings.

Our interest in the augmented system approach is motivated by the difficulties encountered in solving the least squares problem when A contains "dense" columns. Because the matrix AD^2A^T is then much denser than A, the normal equations approach must be modified in some way for this case. One idea is to remove the dense columns from A, and then to correct for their absence by an iterative method [1], a direct method [6], or some combination of the two [11]. All of these modifications suffer from numerical instability, which may be traced to singularity or near-singularity of AD^2A^T after the dense columns have been removed. A more numerically sound alternative is based on an augmented linear system formed by "splitting" dense columns [12, 19]; we do not yet have firm evidence as to the efficiency of this kind of modification, however.

All of the above approaches have the drawback of requiring that dense columns be identified in A in some arbitrary way. The approach that we describe below

from that reported in [7] primarily in the new super-nodal Cholesky factorization [5]. All of the computational testing was done on an IBM RISC System/6000, model 530, with 32 Mbytes of memory. Both OSL and OB1 are written in FORTRAN. The times reported do not include the time to read the MPS input file. OSL times include time for pre-solve, crash, and solution. Unless otherwise noted, all OSL runs were made with pre-solve=3 and crash=2. We have had the best results on most problems with crash=2. OB1 times include time for pre-solve, min-degree ordering, and solve. Except for problem REG530, all OB1 runs were made with the default control parameters. REG530 was run with the defaults, and again with dense=100, to illustrate the effect of dense columns.

TABLE 2. OSL(simplex) vs OB1(interior)

	OSL simplex		OB1 interior		OSL/OB1
name	iter	time(sec.)	iter	time(sec.)	time
AMOCO	900	7.04	26	9.45	0.74
PLAN01	770	7.52	22	5.78	1.30
RAJA	3,247	51.16	19	12.60	4.06
25FV47	2,999	36.71	24	12.41	2.96
REG530	885	11.80	15	40.36	0.29
REG530(a)			15	9.15	1.29
PIMS2	1,494	38.75	25	30.06	1.29
80BAU3B	8,117	160.34	39	44.19	3.63
DIAGMA	2,485	82.71	23	66.22	1.25
AMOCO2	1,929	79.32	33	164.00	0.48
DFL001	66,703	5,012.06	47	4,230.45	1.18
DFL001(b)	83,870	5,949.51			1.41
DFL001(c)	89,560	12,983.96			3.07
PILOT	6,992	376.10	29	146.22	2.57
PILOT(d)	5,526	302.39			2.07
MFMPTE08	9,135	311.06	29	64.13	4.85
MARKAL	16,041	664.22	41	132.03	5.03
PILOT87	11,578	1,459.42	42	621.15	2.35
PILOT87(d)	9,211	1,142.00			1.84
MFMPTE16	27,257	2,017.97	30	133.76	15.09
MFMPTE16(b)	16,221	1,066.46			7.97
MFMPTE16(c)	17,052	1,141.62			8.53
NCM	8,938	309.06	44	69.14	4.47
TRIP	104,687	12,417.22	41	418.97	29.64
TRIP(b)	113,371	14,033.78			33.50
FORD	19,935	7,430.95	56	632.20	11.75
OILCO	95,857	19,889.51	64	3,688.76	5.39
OILCO(b)	86,531	17,174.98			4.66

(a) OB1 run with dense=100, so that dense columns are handled by the Schur complement approach.
(b) OSL run with crash=4.
(c) OSL run with crash=3.
(d) OSL run with scaling.

REFERENCES

1. I. Adler, N. Karmarkar, M.G.C. Resende, and G. Veiga, 1989. "An Implementation of Karmarkar's Algorithm for Linear Programming," MATHEMATICAL PROGRAMMING 44:3, 297-336.

2. I. Adler, N. Karmarkar, M.G.C. Resende, and G. Veiga, 1989."Data Structures and Programming Techniques for the Implementation of Karmarkar's Algorithm," ORSA JOURNAL ON COMPUTING 1:2, 84-106.

3. I. C. Choi, C. L. Monma, and D. F. Shanno, 1990. "Further Development of a Primal-Dual Interior Point Method," ORSA JOURNAL ON COMPUTING 2:4, 304-311.

4. D. M. Gay, 1985. "Electronic Mail Distribution of Linear Programming Test Problems," Mathematical Programming Society COAL Newsletter, December.

5. A. Gupta and E. Rothberg, 1990. "Efficient Sparse Matrix Factorization on High-Performance Workstations - Exploiting the Memory Hierarchy," Department of Computer Science, Stanford University, Stanford, CA, 94305.

6. I. J. Lustig, R. E. Marsten, and D. F. Shanno, 1989. "Computational Experience with a Primal-Dual Interior Point Method for Linear Programming," Technical Report SOR 89-17, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

7. I. J. Lustig, R. E. Marsten, and D. F. Shanno, 1990. "On Implementing Mehrotra's Predictor-Corrector Interior Point Method for Linear Programming," Technical Report SOR 90-03, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

8. R. E. Marsten, R. Subramanian, M. J. Saltzman, I. J. Lustig, and D. F. Shanno, 1990. "Interior Point Algorithms for Linear Programming: Just Call Newton, Lagrange, and Fiacco and McCormick!," INTERFACES 20:4, 105-116.

9. 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 1:2, 70-83.

10. S. Mehrotra, 1990. "On the Implementation of a (Primal-Dual) Interior Point Method," Tech. Report 90-03, Dept. of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

11. "Optimization Subroutine Library: Guide and Reference," Publication SC23-0519-1, IBM Corporation, Second Edition, August, 1990.

[16] C.B. Moler, J. Little, S. Bangert and S. Kleiman, Pro-Matlab User's Guide MathWorks, Sherborn, MA, 1987.

[17] C. Monma and A. Morton, "Computational experience with a dual affine variant of Karmarkar's method for linear programming," *Operations Research Letters* 6 (1987), 261-267.

[18] C. Roos and J.-P. Vial, "A polynomial method of approximate centers for linear programming," Report, Delft University of Technology (1988), to appear in *Mathematical Programming*.

[19] M.J. Todd, "The effects of degeneracy and null and unbounded variables on variants of Karmarkar's linear programming algorithm," in: Large-Scale Numerical Optimization, T.F. Coleman and Y. Li, eds., *SIAM*, Philadelphia, 1990, 81-91.

[20] M.J. Todd, "A low complexity interior point algorithm for linear programming," Technical Report 903, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1990.

[21] M.J. Todd and J.-P. Vial, "Todd's low-complexity algorithm is a predictor-corrector path-following method," Technical Report No. 952, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1990.

[22] M.J. Todd and Y. Wang, "On combined phase 1 - phase 2 projective methods for linear programming," Technical Report No. 877, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1989, to appear in Algorithmica.

[23] T. Tsuchiya, "Global convergence of the affine scaling methods for degenerate linear programming problems," Technical Report, The Institute of Statistical Mathematics, Tokyo, 1990.

[24] D. Xiao and D. Goldfarb, "A path-following projective interior point method for linear programming," Manuscript, Department of IE/OR, Columbia University, New York, 1990.

[25] Y. Ye, "An $O(n^3L)$ potential reduction algorithm for linear programming," manuscript, Department of Management Science, University of Iowa, Iowa City, IA, 1988, to appear in *Mathematical Programming*.

References

[1] I. Adler, M.G.C. Resende, G. Veiga and N. Karmarkar, "An implementation of Karmarkar's algorithm for linear programming," *Mathematical Programming* 44 (1989), 297-335.

[2] K. Anstreicher, "Strict monotonicity and improved complexity in the standard form projective algorithm for linear programming," CORE Discussion Paper 9035, CORE, Université Catholique de Louvain (Louvain-la-Neuve, Belgium), 1990.

[3] K.M. Anstreicher and P. Watteyne, "A family of search directions for Karmarkar's algorithm," CORE Discussion Paper 9030, CORE, Université Catholique de Louvain (Louvain-la-Neuve, Belgium), 1990.

[4] E.R. Barnes, "A variation on Karmarkar's algorithm for solving linear programming problems," *Mathematical Programming* 36 (1986), 174-182.

[5] I.C. Choi, C.L. Monma and D.F. Shanno, "Further development of a primal-dual interior point method," *ORSA Journal on Computing* 2 (1990), 304-311.

[6] I.I. Dikin, "Iterative solution of problems of linear and quadratic programming," *Doklady Academiia Nauk SSSR* 174 (1967), 747-748.

[7] R.M. Freund, "Polynomial-time algorithms for linear programming based only on primal scaling and projected gradients of a potential function," manuscript, Sloan School of Management, M.I.T., Cambridge, Massachusetts, 1988.

[8] D.M. Gay, "Electronic mail distribution of linear programming test problems," *COAL Newsletter* 13 (1985), 10-12.

[9] C. Gonzaga, "Polynomial affine algorithms for linear programming," *Mathematical Programming* 49 (1990), 7-21.

[10] N. Karmarkar, "A new polynomial time algorithm for linear programming," *Combinatorica* 4 (1984), 373-395.

[11] I.J. Lustig, R.E. Marsten and D.F. Shanno, "Computational experience with a primal-dual interior-point method for linear programming," Technical Report SOR 89-17, Department of Civil Engineering and Operations Research, Princeton University, 1989, to appear in *Linear Algebra and its Applications.*

[12] I.J. Lustig, R.E. Marston and D.F. Shanno, "The primal-dual interior point method on the Cray supercomputer," in: Large-Scale Numerical Optimization, T.F. Coleman and Y. Li, eds., *SIAM*, Philadelphia, 1990, 70-80.

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

[14] N. Megiddo and M. Shub, "Boundary behavior of interior point algorithms in linear programming," *Mathematics of Operations Research* 14 (1989), 97-146.

[15] S. Mehrotra, "On the implementation of a (primal-dual) interior point method," Technical Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1990.

Interior-Point Algorithms for Solving Nonlinear Optimization Problems

Panos M. Pardalos[1], Chi-Geun Han[1], and Yinyu Ye[2]

Abstract

In this note, we briefly describe our recent computational work on interior-point algorithms for solving nonlinear optimization problems. For these interior-point algorithms, a potential function is constructed using the primal-dual formulation of the problem and an optimum solution is obtained by reducing the potential function. Numerical experience indicates that theses algorithms are suitable for some application problems, including obstacle problems, elastic-plastic torsion problems, entropy optimization problems, and several classes of linear complementarity problems.

1. Interior-Point Algorithms for Large-Scale Quadratic Problems

Convex quadratic programming (QP) has been shown to be polynomially solvable by using the ellipsoid algorithm [7]. After Karmarkar's new polynomial algorithm [5] for linear programming many researchers have developed different polynomial-time interior-point algorithms for convex QP problems. The nonconvex QP problem is known to be an NP-hard problem [9]. An interior-point algorithm that is based on solving a sequence of quadratic problems over an ellipsoid was also proposed [12].

In this note, we report only our computational experience of using a potential reduction algorithm to solve some convex QP problems. We emphasize that this is not a survey paper. Many important developments on QP due to other researchers are not included here. For these developments, we refer the reader to the survey paper [12] for a thorough literature search.

Consider the QP problem with box constraints:

$$\min f(x) = \frac{1}{2}x^T Qx + c^T x \tag{1.1}$$
$$\text{s.t. } 0 \le x \le e,$$

where $c \in R^n$, Q is an $n \times n$ positive semi-definite matrix, and $e \in R^n$ is a vector of 1's. Many algorithms have been proposed for the solution of (1.1), including conjugate gradient and projected gradient methods and active set algorithms. More references and applications can be found in [3].

Next, we outline a primal-dual potential reduction algorithm, **PRA**, for solving (1.1). The original primal-dual potential reduction algorithm was proposed for linear complementarity problems by Kojima et al. [6]. We then extended and implemented the algorithm for QP problems. We

[1]Computer Science Department, The Pennsylvania State University, University Park, PA 16802
[2]Department of Management Sciences, The University of Iowa, Iowa City, Iowa 52242

found that the practical performance of the algorithm depends on appropriate choice of the step length and potential parameter [3][10].

Problem (1.1) can be transformed to the standard form

$$\min f(x) = \frac{1}{2}x^TQx + c^Tx \tag{1.2}$$

$$\text{s.t. } x + z = e \text{ and } x, z \geq 0,$$

where $z \in R^n$. The dual of (1.2) can be written as

$$\max e^Ty - \frac{1}{2}x^TQx \tag{1.3}$$

$$\text{s.t. } s_x = Qx + c - y \geq 0 \text{ and } s_z = -y \geq 0,$$

where $y \in R^n$. Also note that the duality gap of (1.2) and (1.3) is given by

$$\delta = \frac{1}{2}x^TQx + c^Tx - (e^Ty - \frac{1}{2}x^TQx) = x^Ts_x + z^Ts_z \geq 0.$$

We use the primal-dual potential function to characterize the interior points x, z, s_x, and s_z,

$$\phi(x, z, s_x, s_z) = \rho \ln(x^Ts_x + z^Ts_z) - \sum_{i=1}^{n} \ln(x_i(s_x)_i) - \sum_{i=1}^{n} \ln(z_i(s_z)_i), \tag{1.4}$$

$$\text{where } \rho \geq 2n + \sqrt{2n}.$$

It is easily verified that

$$\delta \leq \exp(\frac{\phi(x, z, s_x, s_z) - 2n \ln 2n}{(\rho - 2n)}). \tag{1.5}$$

Hence, if

$$\phi(x, z, s_x, s_z) \leq -O((\rho - 2n)L)$$

then

$$\delta < 2^{-O(L)}.$$

In order to minimize the duality gap, we reduce the potential function over the feasible domain. It can be shown that the Newton direction of the following nonlinear system (1.6)-(1.10) with fixed δ guarantees a constant reduction in the potential function. The system is given by

$$XS_xe = \frac{\delta}{\rho}e, \tag{1.6}$$

$$ZS_ze = \frac{\delta}{\rho}e, \tag{1.7}$$

where the slack vectors are

$$z = e - x, \tag{1.8}$$

$$s_x = Qx + c - y, \tag{1.9}$$

$$s_z = -y, \tag{1.10}$$

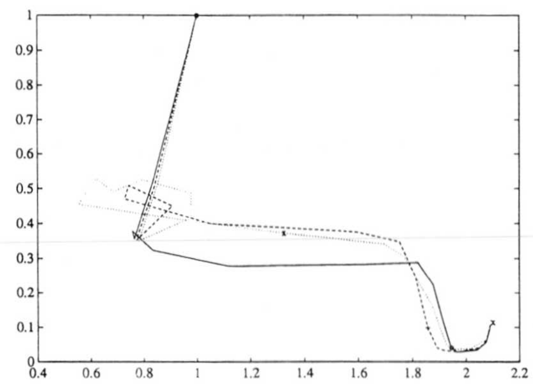

Figure 3(a)

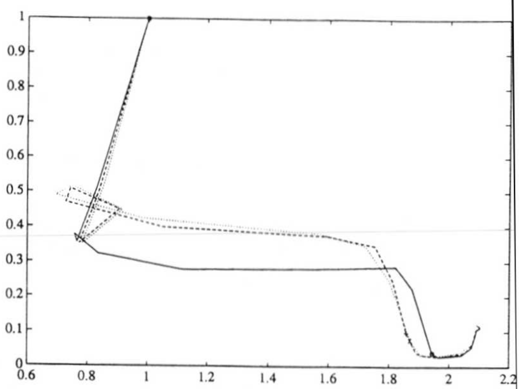

Figure 3(b)

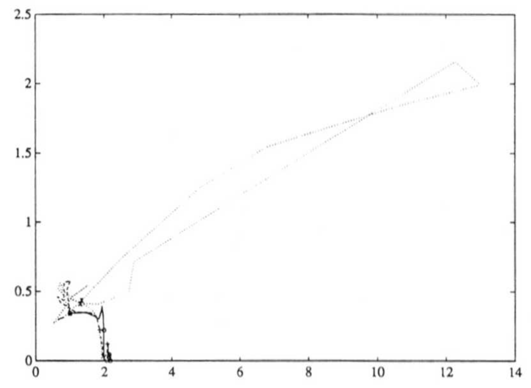

Figure 3(c)

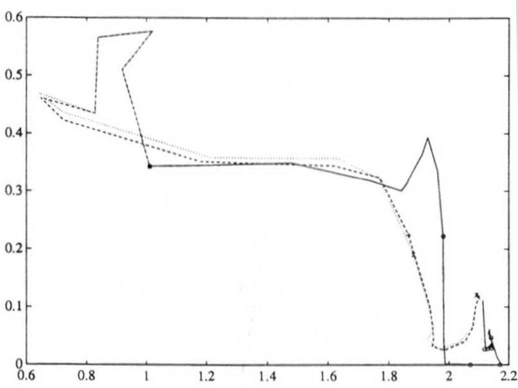

Figure 3(d)

4. Problems without initial feasible point

We conclude by making some brief comments about our results where an initial feasible point is not available. We generate such problems as in section 3, but with $b = A\tilde{x}$ where each component of $\tilde{x}$ is generated as the absolute value of an independent standard Gaussian random variable. Now we add an artificial variable with column $b - Ae$ and cost 10^{10} to get a new problem with an extra variable. We modified the line search to go all the way to the boundary if this eliminated the artificial variable.

Both variants 2 and 3 had difficulty with such problems if they were initialized with lower bounds of $-\infty$. The iterates grew very large (usually to the order of 10^8) before returning to the neighborhood of the optimal solution, as in the experiments in [19]. We therefore initialized the lower bound to -10^5, as in [19]. In this case, variant 2 was usually comparable to the affine-scaling algorithm and two to six iterations faster than variant 3. Average iteration counts varied from 13.4 to 28.0. Typical trajectories for a 300×599 problem are shown in figure 3(a). Here the dashed line, marked with a "+" every ten iterations, shows variant 2; the solid line, with "o" 's, shows the affine-scaling method and the dotted line, with "x" 's, illustrates variant 3. The algorithms require 19, 18 and 22 iterations respectively. If we modified variant 3 to make it monotonic ($\bar{d} = \bar{d}_\zeta$ if $\zeta > \alpha$, else $\bar{d} = \bar{d}_\alpha$), its behavior becomes almost identical to that of variant 2 -- see figure 3(b); 19 iterations are now required.

Finally, we examined the behavior of these methods when started at the feasible solution $\tilde{x}$, so that an artificial variable is not needed. The behavior of variant 2 (now with initial bound $-\infty$) was comparable to its behavior on the artificial problem (with initial bound -10^5). The affine-scaling method performed very poorly, partly because of its difficulty in generating lower bounds, but also because some variables became prematurely very small. Variant 3 needed either an initial lower bound of -10^5 or monotonic directions to perform adequately. In the latter case it was again comparable to variant 2. These results are illustrated by the trajectories for the same problem as before. (Here the trajectories start near (1,.3).) Figure 3(c) shows variant 2 (dashed line, 20 iterations), the affine-scaling algorithm (solid line, 85 iterations), and variant 3 with initial bound -10^5 (dotted line, 36 iterations). Notice how the latter initially has variables increasing substantially -- in fact, some reach the order of 10^3, while the initial and final solutions are of order 1. Figure 3(d) shows the first two algorithms as above, and variant 3 with monotonic directions (dotted line, 20 iterations). Here we see the poor behavior of the affine-scaling method -- one variable becomes much too small and impedes progress.

Overall, the results suggest that the affine-scaling algorithm can perform very badly when the initial solution is far from centered (compare with the analytic results of Megiddo and Shub [14]), and that it is perhaps preferable to introduce an artificial variable even if a feasible solution is known, or to take some initial centering steps.

and the upper-case letter (X) designates the diagonal matrix of the vector (x) in lower-case.

We follow the Newton direction but use a different step length, which we describe below. Given the kth iterate x^k and y^k, that are interior feasible points for the primal (1.2) and the dual (1.3), that is

$$x^k > 0, z^k = e - x^k > 0 \text{ and } x^k + z^k = e,$$

and

$$s_x^k = Qx^k + c - y^k > 0 \text{ and } s_z^k = -y^k > 0,$$

we solve the following system of linear equations to obtain the search direction $\{\Delta x, \Delta z, \Delta y\}$.

$$X^k \Delta s_x + S_x{}^k \Delta x = \frac{\delta^k}{\rho} e - X^k S_x{}^k e, \tag{1.11}$$

$$Z^k \Delta s_z + S_z{}^k \Delta z = \frac{\delta^k}{\rho} e - Z^k S_z{}^k e, \tag{1.12}$$

$$\Delta x = -\Delta z, \tag{1.13}$$

$$\Delta s_x = Q\Delta x - \Delta y, \tag{1.14}$$

$$\Delta s_z = -\Delta y. \tag{1.15}$$

Using (1.13)-(1.15), we can write (1.11) and (1.12) as

$$\begin{pmatrix} X^k Q + S_x^k & -X^k \\ -S_z^k & -Z^k \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \frac{\delta^k}{\rho} e - X^k S_x^k e \\ \frac{\delta^k}{\rho} e - Z^k S_z^k e \end{pmatrix}. \tag{1.16}$$

The system (1.16) can be further decomposed as

$$(X^k Q + S_x^k + X^k S_z^k (Z^k)^{-1})\Delta x = \frac{\delta^k}{\rho}(e - X^k (Z^k)^{-1} e) - X^k (s_x^k - s_z^k) \tag{1.17}$$

and

$$-Z^k \Delta y = S_z^k \Delta x + (\frac{\delta^k}{\rho} e - Z^k s_z^k). \tag{1.18}$$

The matrices in (1.17) and (1.18) may become increasingly ill-conditioned as x approaches the boundary. For ill-conditioned matrices, special care may be taken. Define the diagonal matrix $\bar{D}$ whose diagonal coefficients are given by:

$$\bar{D}_{ii} = \sqrt{x_i^k z_i^k} \text{ for } i = 1, \cdots, n.$$

Then, we can use $\bar{D}$ to transform (1.17) and (1.18) to

$$(\bar{D} Q \bar{D} + S_x^k Z^k + S_z^k X^k)(\bar{D}^{-1} \Delta x) = \frac{\delta^k}{\rho}(\bar{D}(X^k)^{-1} e - \bar{D}(Z^k)^{-1} e) - \bar{D}(s_x^k - s_z^k) \tag{1.19}$$

and

$$-\bar{D}\Delta y = S_z^k X^k (\bar{D}^{-1} \Delta x) + (\frac{\delta^k}{\rho} \bar{D}(Z^k)^{-1} e - \bar{D} s_z^k). \tag{1.20}$$

After obtaining the direction, we can choose the step length $\bar{\theta}$ by performing a line search to minimize the potential function (1.4), and generate the $(k+1)$th iterate

$$x^{k+1} = x^k + \bar{\theta}\Delta x \text{ and } y^{k+1} = y^k + \bar{\theta}\Delta y.$$

In practice, we observed that for ρ large enough (usually larger than $\Omega(n)$) $\bar{\theta}$ can be simply chosen as

$$\bar{\theta} = \beta \max\{\theta \ : \ x^k + \theta\Delta x \geq 0, e - (x^k + \theta\Delta x) \geq 0, -y^k - \theta\Delta y \geq 0, \\ \text{and } {s_x}^k + \theta(Q\Delta x - \Delta y) \geq 0\} \text{ for some } 0 < \beta < 1. \qquad (1.21)$$

Still, x^{k+1} and y^{k+1} guarantee a constant reduction in the potential function. This suggests that the use of the line-search is unnecessary and can be replaced by (1.21), which is used in our implementation.

Now, we are ready to summarize the primal-dual potential reduction algorithm:

Algorithm PRA

```
Choose $x^0 = .5e$ and $\alpha = \max\{1, 3 \parallel Qx^0 + c \parallel - \frac{e^T}{2n}(Qx^0 + c)\}$;
Set $z^0 = e - x^0$, $y^0 = -\alpha e$, ${s_x}^0 = Qx^0 + c - y^0$, and ${s_z}^0 = -y^0$;
Let $\delta^0 = (x^0)^T {s_x}^0 + (z^0)^T {s_z}^0$ and $k = 0$;
while $\delta^k \geq 2^{-L}$ do
begin
    Compute $\Delta x$ and $\Delta y$ from (1.19) and (1.20);
    Let $\bar{\theta}$ be given by (1.21);
    $x^{k+1} = x^k + \bar{\theta}\Delta x$;
    $z^{k+1} = e - x^{k+1}$ ;
    $y^{k+1} = y^k + \bar{\theta}\Delta y$;
    ${s_x}^{k+1} = Qx^{k+1} + c - y^{k+1}$;
    ${s_z}^{k+1} = -y^{k+1}$;
    $\delta^{k+1} = (x^{k+1})^T {s_x}^{k+1} + (z^{k+1})^T {s_z}^{k+1}$;
    $k = k + 1$;
end.
```

The following theorem presents the theoretical complexity of **PRA**, where we assume that the coefficients of Q and c are integers.

Theorem 2.1. *Let $\rho \geq 2n + \sqrt{2n}$ and $L = 2n^2 + [\log|P|]$ where P is the product of the nonzero integer coefficients appearing in Q and c. Then,* **PRA** *terminates in $O((\rho - 2n)L)$ iterations and each iteration uses $O(n^3)$ arithmetic operations.*

The table below gives the average number of iterations required for five random problems and the three algrithms discussed above. For the affine-scaling algorithm, the average proportion of the way to the boundary was (of course) .95. For variant 2, it ranged from .96 to .99, and for variant 3, from .88 to .99. The advantage of the latter algorithms is that their use of a potential function allows further flexibility: short steps will be taken if necessary, longer steps if safe.

	100×200	200×400	300×600	400×800
affine-scaling	11.8	12.8	13.8	14.4
variant 2	12.2	13.6	13.8	14.4
variant 3	13.6	16.0	18.2	19.2

We see that in an algorithm with a polynomial-time guarantee it generally pays to choose a direction between that of the pure potential reduction algorithm and that of the affine-scaling algorithm, and that the penalty of using variant 2 instead of the latter is very slight in terms of number of iterations. (Unfortunately, variant 2 needs an extra projection at each iteration.)

Finally, all the runs with variant 2, and to a lesser extent, the other methods, show a strong symmetry between the objective function values and the lower bounds generated -- see Figure 2 for a typical example, where the solid line shows the progress of the objective and the dotted line that of the bound; at each iteration, both have roughly the same accuracy.

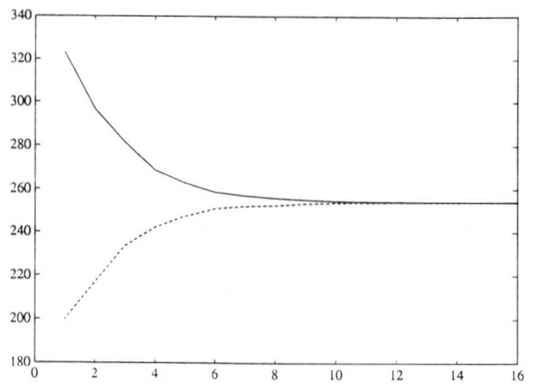

Figure 2.

Notice how the modified BLCA reaches the vicinity of the central path in only 3 iterations, and after 10 is further along the path than the BLCA after 30 iterations. Both these methods follow very smooth paths. (Experiments with modifications taking step sizes of .8 or .95 along the affine-scaling direction lead to similar results, with fewer iterations and still a very small number of centering steps.) We must conclude that interior-point methods whose step is of fixed (or uniformly bounded by one) Euclidean length in the scaled space seem to be very short-step methods with a strong resemblance to path-following methods. Such methods include the original projective-scaling algorithm of Karmarkar [10] (although a line search is usually employed) and the affine-scaling method with the step size analyzed by Dikin [6], Barnes [4], and Tsuchiya [23] (in practice, a step a certain proportion of the way to the boundary is typically used). See also Xiao and Goldfarb [24] and Anstreicher [2].

We also remark that the efficient method (the solid line) also appears to follow the central path quite closely after its initial few steps.

3. Problems with e feasible

Here we briefly discuss experiments on problems generated so that the vector of ones, e, yields an initial feasible solution. (Each entry of A was generated as an independent standard Gaussian random variable; b was set to Ae; and c was set to $A^Ty + s$, where y and s were generated as was A, and then s replaced by |s|, the vector of absolute values of the components of s. This was also the way the problems of the previous section were generated.) We tested variant 2 of the BLCA (with $q = 2n$) against the affine-scaling algorithm, with step size .95 of the way to the boundary of the feasible region, and against another variant of the BLCA. We call this variant 3; it only differs from variant 2 in its choice of direction, which is $\bar{d}_\zeta$ (see section 4 of [20]). This is very close to the potential reduction methods of Gonzaga [9], Ye [25] and Freund [7]. The direction chosen by variant 2 is usually half-way between $\bar{d}_\zeta$ and the affine-scaling direction. With $q = 2n$, both variants require O(nt) iterations to obtain t digits of accuracy.

The termination criterion in all cases was that the relative error in the objective function be no more than 10^{-4}. (In the line search, we checked whether a step all the way to the boundary would satisfy this criterion, and, if so, took such a step.) Here the relative error is with respect to the lower bound. Such a lower bound is generated in variants 2 and 3, using lemma 5 of [20]. We also found that the same procedure generated lower bounds for the affine-scaling algorithm. The latter is normally terminated when the relative improvement in the objective function falls below a certain level. Especially if a dual solution is required, it might be worth adding a further test at this stage, by trying to compute a lower bound using lemma 5 of [20]. If successful, this provides a guarantee of the quality of the solution as well as a near-optimal dual solution, and our results indicate that success is likely for well-scaled problems with near-central initial solutions. The additional cost required is a further least-squares solution, to obtain e_p.

2. Computational Results

We implemented the primal-dual potential reduction algorithm, **PRA**, for convex QP problems with box constraints using vectorization on an IBM 3090-600S computer with Vector Facilities. The VS Fortran compiler was used for the computations and all numerical results were obtained in double precision. The ESSL (Engineering and Scientific Subroutine Library) was used for vector and matrix operations.

All CPU times reported in the Tables are given in seconds and we used the DSDCG subroutine of the ESSL that uses the conjugate gradient method for solving linear systems (1.19). This method takes advantage of the sparsity and the special nonzero pattern of the matrix Q.

Two parameters, β and ρ, are used in the algorithm. In order to find the parameter values with which the algorithm converges to the optimum objective function value fast, we performed experimental computations by fixing one parameter value and changing another parameter value. We found that as β approached 1, **PRA** converged very fast and the average number of iterations is very large for small values of ρ, decreases rapidly as ρ increases up to the value of $n^{1.5}$, and then slightly increases. When ρ is about $n^{1.5}$, we have the minimum average number of iterations. We conclude that a good combination of parameter values is given by β=0.99 and ρ=$n^{1.5}$.

The obstacle problem [8], the elastic-plastic torsion problem [1], and the journal-bearing problem can be formulated as (1.1). Using instances of the obstacle problem and the elastic-plastic torsion problem. we tested the algorithm **PRA**. The obstacle problems and the elastic-plastic torsion problems are explained in detail in [4].

For these problems. Q is a special block tridiagonal matrix. The superdiagonal and subdiagonal blocks of Q are $-I \in R^{m\times m}$, where I is an identity matrix, and diagonal blocks are also tridiagonal matrices whose diagonal elements are 4 and their superdiagonal and subdiagonal elements are -1. All blocks of Q have the same size m by m, where $m^2 = n$. The vector c of the obstacle problems is given by

$$c = -h^2 \quad \text{where} \quad h = \frac{1}{m+1}.$$

We used two different bounds for the obstacle problems. For problem I, the upper bound (u) and lower bound (l) are given by

$$l_i = (\sin(9.2\alpha_i) \times \sin(9.3\gamma_i))^3,$$

$$u_i = (\sin(9.2\alpha_i) \times \sin(9.3\gamma_i))^2 + 0.02,$$

and for problem II

$$l_i = \sin(3.2\alpha_i) \times \sin(3.3\gamma_i),$$

$$u_i = 2000,$$

where $\alpha_i = (i - \lfloor \frac{i-1}{m} \rfloor \times m) \times h$ and $\gamma_i = \lceil \frac{i}{m} \rceil \times h$ for $i = 1, \cdots, n$. For the elastic-plastic torsion problems.

$$|x_i| \leq \text{distance to the boundary of the grid}$$

and $c = -th^2$, where $t = 5, 10$, and 20.

$\Delta^k \leq 10^{-6}$ was used for the stopping criterion of the potential reduction algorithm and the quantity $\| Ax - b \| / \| x \| \leq 10^{-9}$ was used as a stopping criterion for solving the linear system of the form $Ax = b$. To check the accuracy of the solution obtained by the potential reduction algorithm, we calculated $\tau = \| \nabla f(x) \| / \| \nabla f(x^0) \|$ (see [8]), where

$$(\nabla(f(x)))_i = \begin{cases} \partial_i f(x) & \text{if } x_i \in (l_i, u_i) \\ \min\{\partial_i f(x), 0\} & \text{if } x_i = l_i \\ \max\{\partial_i f(x), 0\} & \text{if } x_i = u_i. \end{cases}$$

$x_i^0 = (u_i + l_i)/2$ and $x_i^0 = l_i + 1$ are initial points for problem I and II of the obstacle problems, respectively. For the elastic-plastic torsion problems, origin and upper bound are used as initial points for the problems of the form (1.1). Tables 1, 2, and 3 show the computational results. Figure 1 shows the final solution of the obstacle problem I. For the elastic-plastic torsion problems, it is observed that when the starting point is upper bound and t has a large value, less CPU time is needed to solve the problems. Figure 2 shows the final solution of the elastic-plastic torsion problem when $t = 20$.

3. Remarks

In this note, we outline an interior-point algorithm (**PRA**) for solving quadratic programming problems with box constraints and present our computational results. It is found that **PRA** can be a good solver for practical engineering problems that are very large and sparse.

The proposed **PRA** can be used for the solution of the general nonconvex quadratic optimization problem. Many algorithms for such problems (see [9]) solve a convex subproblem at each iteration. These subproblems can be solved efficiently using **PRA**. An important problem that can be formulated as a quadratic problem is the linear complementarity problem (LCP). In the general case, the LCP is a nonconvex problem. Interior-point algorithms for solving such LCPs are described in detail in [11].

When the matrix Q has a special structure as observed in the obstacle problem and the elastic-plastic torsion problem, we can take advantage of parallelism by using a parallel linear system solver (e.g. cyclic reduction algorithm). Parallelization of **PRA** is one subject of our current research.

Recently, we used an interior-point algorithm for solving the entropy optimization problem. The problem is given by:

$$\min \ f(x) = c^T x + \sum_{j=1}^{n} w_j x_j \ln(x_j) \qquad (3.1)$$

$$\text{s.t. } x \in \Omega_p = \{x \in R^n : Ax = b, x \geq 0\}$$

where $c, w \in R^n$ and $w \geq 0, A \in R^{m \times n}$ and $b \in R^m$. This problem can be solved by the potential reduction algorithm, which generates a solution x^k

$$f(x^k) - f(x^*) \leq \epsilon$$

in $O(\sqrt{n}|\ln \epsilon|)$ iterations, where x^* is the optimal solution. Computational results with some test problems can be found in [2].

central path (e.g. [18]). Indeed, Todd and Vial [21] recently showed that $\|\bar{d}_\alpha\|$ is exactly the measure of centrality of Roos and Vial [18]. Further, they showed that after some initial centering steps, each affine-scaling step is followed by at most three centering steps before another affine-scaling step. Hence the BLCA generates iterates close to the central path.

What only emerged from computational testing is how astonishingly close the iterates remain. Indeed, after the initial centering phase, no further centering steps were made in several runs on random standard-form problems of dimensions 50×100 up to 300×600, and in fact $\|\bar{d}_\alpha\|$ continued to decrease. This encouraged us to try modifications. Suppose we perform a line search on the logarithmic barrier function along the constant-cost centering direction if $\|\bar{d}_\alpha\| > 9/10$. We take a Newton step in this direction if $\|\bar{d}_\alpha\|$ lies between 1/16 and 9/10. Finally, if $\|\bar{d}_\alpha\|$ is smaller than 1/16, we take a step of length .4 (rather than .2) in the affine-scaling direction. It can be shown [21] that, after the initial centering phase, each affine-scaling step is followed by at most two centering steps; but in practice, a single centering step is only needed every 10-20 iterations.

In Figure 1, we show the trajectories of these algorithms on a 50×100 problem. Figure 1(a) plots the 31st component of x against the 30th, while 1(b) plots the 34th against the 33th. The dashed line corresponds to the BLCA, which requires 347 iterations; every tenth iterate is marked with a "+". The dotted line is the modification described above, needing only 170 iterations; every tenth iterate is marked with a "x." Finally, the solid line gives the progress of the most efficient variant of the BLCA, called variant 2 in [20], with q = 2n; this needs only 11 iterations. All trajectories start at (1,1).

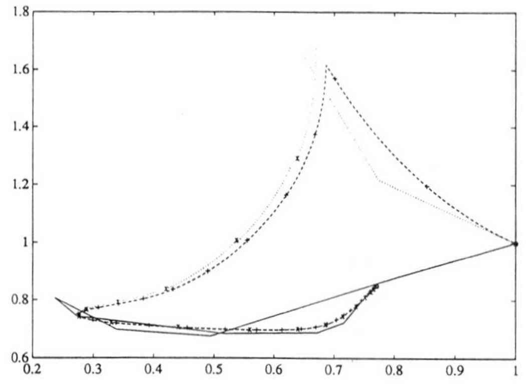
Figure 1(a)

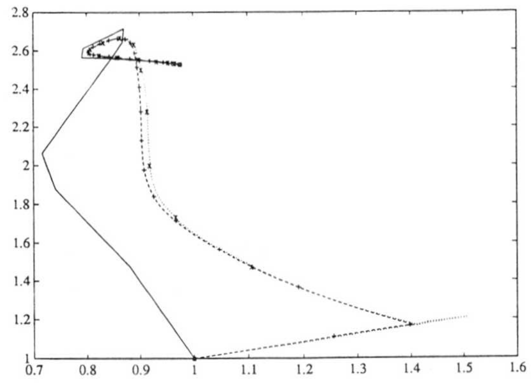
Figure 1(b)

Here cbar ($\bar{c}$) and Abar ($\bar{A}$) are the scaled cost vector and coefficient matrix, ones(cbar) is the vector (e) of ones of the same dimension as cbar, and Abar$'$ \ [cbar ones(cbar)] yields the matrix $[y_c \; y_e]$, where y_c is the least-squares solution to $\bar{A}^T y_c \approx \bar{c}$ and y_e that to $\bar{A}^T y_e \approx e$. MATLAB computations are carried out in a numerically stable way, are remarkably efficient for dense matrices, and require, as is evident from the example above, very little programming time.

In section 2 I will discuss using MATLAB to elucidate the relationship between the low-complexity algorithm in [20] and path-following methods. Section 3 compares a more efficient version of the low-complexity algorithm to a variant which bears a strong resemblance to the methods of Gonzaga [9], Ye [25] and Freund [7] and also to the affine-scaling algorithm. Finally, in section 4 I consider solving problems where an initial strictly feasible solution is not at hand.

2. Following the central path

The basic low-complexity algorithm (BLCA) in [20] solves the standard-form linear programming problem

$$(P) \qquad \begin{aligned} &\min c^T x \\ &Ax = b \\ &x \geq 0 \end{aligned}$$

as follows. Given a strictly positive feasible solution $\hat{x}$, let $\bar{A} = A\hat{X}$ and $\bar{c} = \hat{X}c$, where $\hat{X} = \mathrm{diag}(\hat{x})$. These are the transformed data in the scaled problem

$$(\bar{P}) \qquad \begin{aligned} &\min \bar{c}^T \bar{x} \\ &\bar{A}\bar{x} = b \\ &\bar{x} \geq 0 \end{aligned}$$

in terms of the scaled variables $\bar{x} = \hat{X}^{-1}x$. Note that the current iterate $\hat{x}$ corresponds to $\bar{x} = e$, the vector of ones.

The algorithm first computes the projections $\bar{c}_p$ and e_p of $\bar{c}$ and e onto the null space of $\bar{A}$, and sets $\bar{d}_\alpha := -\alpha\bar{c}_p + e_p$ where $\alpha := \bar{c}_p^T e/\bar{c}_p^T\bar{c}_p$. Then the search direction is $\bar{d} = \bar{d}_\alpha/\|\bar{d}_\alpha\|$ if $\|\bar{d}_\alpha\| \geq .3$ and $\bar{d} = -\bar{c}_p/\|\bar{c}_p\|$ otherwise. The new point is $\bar{x}_+ = e + .2\bar{d}$ in the scaled space, and $x_+ = \hat{X}\bar{x}_+$ in the original space. Perhaps surprisingly, this algorithm achieves the best known complexity in terms of the number of iterations: given a suitable starting point, it gets within 2^{-t} of the optimal value of (P) within $O(\sqrt{n}\; t)$ iterations, where n is the number of variables in (P).

The BLCA takes a step in the direction $\bar{d}_\alpha/\|\bar{d}_\alpha\|$ (a constant-cost centering step) if $\|\bar{d}_\alpha\|$ is large, and a step in the direction $-\bar{c}_p/\|\bar{c}_p\|$ (the affine-scaling direction of Dikin) if $\|\bar{d}_\alpha\|$ is small. In the latter case, $\bar{c}_p$ and e_p are in some sense close to being collinear, which characterizes the

References

[1] R. Glowinski. Numerical methods for nonlinear variational problems. Heidelberg Berlin New York, Springer (1984).

[2] C.-G. Han, P.M. Pardalos, and Y. Ye. On potential reduction algorithms for some entropy optimization problems. Technical report CS-91-02, Computer Science Department, The Pennsylvania State University (1991).

[3] C.-G. Han, P.M. Pardalos, and Y. Ye. Computational aspects of an interior point algorithm for quadratic programming problems with box constraints. *Large-Scale Numerical Optimization* (Eds: T.F. Coleman and Y. Li), pp. 92-112, SIAM (1990).

[4] C.-G. Han, P.M. Pardalos, and Y. Ye. Solving some engineering problems using an interior-point algorithm. Technical report CS-91-04, Computer Science Department, The Pennsylvania State University (1991).

[5] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica* 4 (1984), pp. 373-395.

[6] M. Kojima, S. Mizuno, and A. Yoshise. An $O(\sqrt{n}L)$ iteration potential reduction algorithm for linear complementarity problems. Research Report on Information Sciences B-217, Tokyo Institute of Technology, Japan, to appear in *Mathematical Programming.*

[7] M.K. Kozlov, S.P. Tarasov, and L.G. Khachiyan. Polynomial solvability of convex quadratic programming. *Soviet Math. Dokl.* Vol. 20 (1979), pp. 1108-1111.

[8] J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. Technical report MCS-P77-0589 (1989), Argonne National Laboratory, Argonne, Il.

[9] P.M. Pardalos and J.B. Rosen. *Constrained global optimization: algorithms and applications.* Springer-Verlag, Lecture Notes in Computer Science 268 (1987).

[10] P.M. Pardalos, Y. Ye, and C.-G. Han. Algorithms for the solution of quadratic knapsack problems. To appear in *Linear Algebra and its Applications*, Special issue on Interior-Point Methods (1991).

[11] P. M. Pardalos, Y. Ye, C.-G. Han, and J. A. Kaliski. Solution of P_0-matrix linear complementarity problems using a potential reduction algorithm. To appear in *SIAM J. on Matrix Analysis and Applications* (1991).

[12] Y. Ye. Interior-point algorithms for quadratic programming. Working Paper No. 89-29, College of Business Administration, The University of Iowa (1989), to appear in *Recent Developments in Mathematical Programming* (S. Kumar ed.), Gordon & Breach Scientific Publishers (1991).

Appendix

	I			II		
n	$Itr.$	Time	τ	$Itr.$	Time	τ
10000	15	16.3	5.2E-5	15	25.4	6.2E-5
40000	17	131.1	1.0E-5	17	203.9	5.4E-6
90000	18	437.6	5.6E-6	18	699.9	2.3E-6
115600	19	700.3	2.9E-6	18	1018.7	4.0E-6
160000	19	1035.8	6.0E-6	18	1534.7	1.9E-4
250000	20	2110.5	5.1E-6	19	3141.9	6.7E-5
360000	22	4090.3	4.4E-6	19	5312.4	2.2E-6
490000	28	8977.8	3.4E-6	20	8966.0	8.1E-5

Table 1: Obstacle problems. $\rho=n^{1.5}$, $\beta=0.99$.

	$t=5$			$t=10$			$t=20$		
n	$Itr.$	CPU	τ	$Itr.$	CPU	τ	$Itr.$	CPU	τ
10000	15	13.9	2.5E-4	15	11.1	3.5E-4	15	8.8	1.9E-4
40000	16	107.5	4.5E-4	17	91.1	8.5E-5	16	65.9	1.4E-4
90000	19	428.9	1.4E-4	18	307.5	2.0E-4	18	229.6	1.9E-4
115600	18	564.2	1.4E-4	18	442.2	2.7E-4	34	646.9	1.9E-4
160000	23	1232.3	2.1E-4	28	1172.1	2.2E-4	*	*	*

Table 2: Elastic-plastic torsion problems, $x^0 = origin$,*: The algorithm does not converge within 100 iterations.

PLAYING WITH INTERIOR POINTS

by

Michael J. Todd
School of Operations Research
and Industrial Engineering
229 Engineering & Theory Center Bldg.
Cornell University
Ithaca, NY 14853

Abstract. We describe some insights that have been obtained from experimenting with various interior-point algorithms on random linear programming problems.

1. Introduction

Most published computational results for interior-point approaches to linear programming have been concerned with state-of-the-art implementations of various algorithms to suites of large sparse test problems, usually including but not limited to the NETLIB suite (Gay [8]). Among such studies are those of Adler, Karmarkar, Resende and Veiga [1] and Monma and Morton [17] for the dual affine-scaling method; and those of McShane, Monma and Shanno [13], Choi, Monma and Shanno [5], Lustig, Marsten and Shanno [11,12], and Mehrotra [15] for primal-dual methods. However, if one is interested primarily in how the algorithms deal with certain features of the problems or how various families of algorithms compare, it often suffices to apply simple implementations to relatively small dense problems. Examples include Anstreicher and Watteyne [3], Todd and Wang [22], and Todd [19,20].

Here I do not wish to present detailed comparisons of various algorithms. Instead I will give some qualitative conclusions, and also stress the insights that can be obtained by observing the performance of different methods.

A delightfully painless environment for making such observations is provided by MATLAB [16], a high-performance interactive software package for performing matrix computations and displaying results. I will illustrate the ease of use of this package by displaying the MATLAB code for computing the projections of the scaled cost vector and the vector of ones onto the null space of the scaled coefficient matrix:

```
function [cp, ep] = projc(Abar, cbar)
%       computes the projections of cbar and  e  into the null
%       space of Abar.
cpep = [cbar ones(cbar)] - Abar' * (Abar' \ [cbar ones(cbar)]);
cp   = cpep(:,1);
ep   = cpep(:,2);
return
```

Research supported in part by NSF, AFORS and ONR through NSF grant DMS-8920550.

[4] C.C. Gonzaga, "An algorithm for solving linear programming problems in $O(n^3L)$ operations", Memo No. UCB/ERL M87/10, College of Engineering, University of California, Berkeley, CA, (1987).

[5] N. Karmarkar, "A new polynomial-time algorithm for linear programming", *Combinatorica* 4 (1984) 373–395.

[6] M. Kojima, N. Megiddo, S. Mizuno, "Theoretical convergence of large-step primal-dual interior point algorithms for linear programming", Report RJ 7872 (72532) 12/7/90, IBM Research Division, Yorktown Heights, NY 10598.

[7] M. Kojima, S. Mizuno, A. Yoshise, "A primal-dual interior point algorithm for linear programming", in N. Megiddo, ed., *Progress in Mathematical Programming: Interior-Point and Related Methods*, Springer-Verlag, New York (1989) pp. 29–47.

[8] I.J. Lustig, R.E. Marsten, D.F. Shanno, "On implementing Mehrotra's predictor-corrector interior-point method for linear programming", Report SOR 90-03, Dept. of Civil Eng. and OR, Princeton University, Princeton, NJ 08544 (1990).

[9] N. Megiddo, "Pathways to the optimal set in linear programming", in N. Megiddo, ed., *Progress in Mathematical Programming: Interior-Point and Related Methods*, Springer-Verlag, New York (1989) pp. 131–158.

[10] S. Mehrotra, "On the implementation of a (primal-dual) interior point method", Report 90-03, Dept. of Ind. Engineering and Management Sciences, Northwestern University, Evanston, IL (1990).

[11] S. Mizuno, "An $O(n^3L)$ algorithm using a sequence for a linear complementarity problem", *J. Operations Research Soc. of Japan* 32 (1989).

[12] S. Mizuno, M. Todd, "An $O(n^3L)$ adaptive path following algorithm for a linear complementarity problem", Report (1989), to appear in *Mathematical Programming.*

[13] S. Mizuno, M. Todd, Y. Ye, "Anticipated behavior of long-step algorithms for linear programming", Report No. 24, Department of Industrial Engineering and Management, Tokyo Institute of Technology, Oh-Okayama, Meguro-ku, Tokyo 152, Japan, (1989).

[14] S. Mizuno, A. Yoshise, T. Kikuchi, "Practical polynomial time algorithms for linear complementarity problems", *Journal of the Operations Research Society of Japan*, Vol. 32, (1989).

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

[16] C. Roos, J.P. Vial, "Long-steps with the logarithmic penalty barrier function in linear programming", Report 89-44, Delft University of Technology, The Netherlands (1989).

[17] G. Sonnevend, "An "analytical centre" for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming", *Lecture Notes of Control and Information Sciences* 84 (1986) 866–878.

[18] G. Sonnevend, J. Stoer, G. Zhao, "On the complexity of following the central path by linear extrapolation in linear programs", to appear in U. Rieder and P. Kleinschmidt eds., *Proc. 14 Symp. on Operations Research* (Ulm 1989).

[19] P.M. Vaidya, "An algorithm for solving linear programs which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations", preprint, AT&T Bell Laboratories, Murray Hill, NJ (1987).

[20] Y. Ye, "Further development on the interior algorithm for convex quadratic programming", Report, Stanford University, Stanford, CA (1987).

	$t = 5$			$t = 10$			$t = 20$		
n	*Itr.*	CPU	τ	*Itr.*	CPU	τ	*Itr.*	CPU	τ
10000	17	12.4	7.5E-5	15	8.8	1.7E-5	12	5.4	3.2E-4
40000	21	104.9	1.2E-5	18	71.0	1.1E-5	15	43.0	3.0E-5
90000	23	354.5	2.1E-5	20	236.7	1.1E-5	17	146.1	1.0E-5
115600	25	551.5	9.4E-6	21	355.8	3.9E-6	18	222.2	2.1E-6
160000	26	911.5	1.3E-5	22	597.5	2.9E-6	19	381.1	2.2E-6
250000	28	1810.5	9.1E-6	23	1128.6	7.5E-6	20	722.5	1.2E-6
360000	30	3338.4	5.2E-6	25	2141.1	1.5E-6	21	1280.7	1.4E-6
490000	32	5528.3	2.5E-6	26	3405.8	1.9E-6	22	2052.4	1.1E-6

Table 3: Elastic-plastic torsion problems, $x^0 =$ *upper bound.*

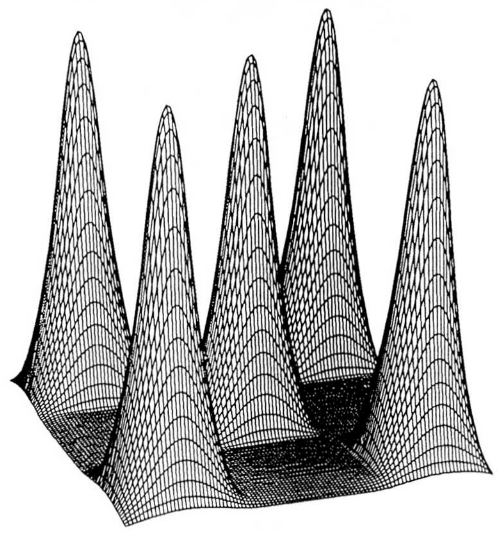
Figure 1: Obstacle problem (I). Final solution.

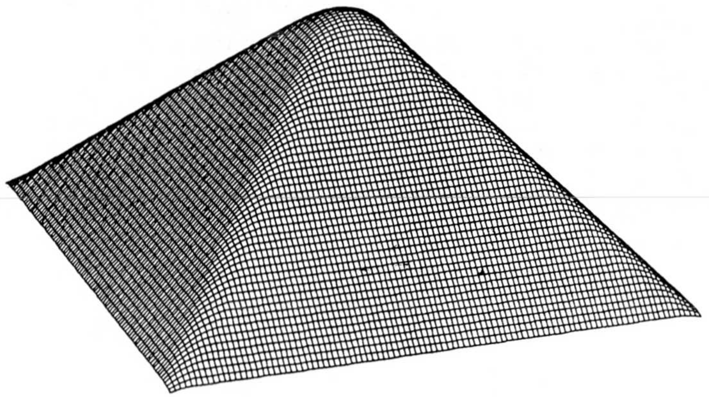
Figure 2: Elastic-plastic torsion problem with $t = 20$. Final solution.

2. Same as above, only in the third inequality we use $\|b\|_\infty \le \|b\|_2$ instead of $\|b\|_\infty \le \frac{\kappa}{\sqrt{n}}\|b\|_2$. ∎

By definition of v_i and by (2.8) and (2.10), it follows that $\|v_1\|_2, \|v_2\|_2 \le \sqrt{\mu/2}$ and $\|v_3\|_2, \|v_4\|_2 \le \sqrt{2n/\mu}$. Further, inequality (2.13) (with $v_1 = \tilde{\Delta} x$ and $v_2 = \tilde{\Delta} z$) states that $v_1^2 v_2^2 \le \mu^2/\beta^4 = \mu^2/16$. From these estimates and the lemma we obtain

$$a_0 \le -\frac{3}{16}\mu^2 < 0, \qquad a_1 \le 2\mu(\frac{\kappa}{2} + \frac{\kappa}{2} + \frac{1}{4}) = (2\kappa + \frac{1}{2})\mu,$$

$$a_2 \le 6\kappa^2 - \frac{1}{4}, \qquad a_3 \le 8\kappa^2\frac{\sqrt{n}}{\mu}, \qquad a_4 \le 4\kappa^2\frac{n}{\mu^2}.$$

In this case we can verify that $p(1/(16\kappa\sqrt[4]{n})) < 0$, so that in the "best case" of the adaptive algorithm we may expect a steplength of (at least) $O(1/\sqrt[4]{n})$, while the worst-case bound of $O(1/\sqrt{n})$ on the steplength is maintained regardless.

The factor $\sqrt[4]{n}$ remains because the iterates stay close to the path of centers (in the 2 norm). If we knew in advance that all projections were well balanced, then the whole analysis could be carried out in terms of the infinity norm as indicated earlier, and the rate of convergence would be independent of n.

4. CONCLUSIONS

The short-step interior-point methods that allow "nice" polynomial-time proofs of convergence for linear programming turn out to be much too slow for practical algorithms. Thus a number of "long-step" methods have been analyzed to date, most of which are aimed at proving the correctness of existing numerical implementations; see in particular [6].

In Section 3.1, however, we have presented a worst-case example in which the iterates x, y, z are not able to "follow" a long step in the reduction of the complementarity parameter μ. The possibility of such worst cases is responsible for the weak proofs of convergence for long-step methods. These proofs not only fail to explain the fast convergence of the implementations that has been observed for *all* numerical examples, but also exhibit a worse theoretical complexity than even the short-step methods.

The adaptive method presented here is intended to close the gap between theoretical and practical complexity. If we were able to show that some percentage of the projections of the vectors s^k are well balanced over a sequence of iterations, then we could indeed reduce the theoretical complexity. This is the goal of future work.

References

[1] I. Adler, R.D.C. Monteiro, "The limiting behavior of the affine scaling trajectories for linear programming problems", Report ESRC 88-9, University of California, Berkeley, CA (1988).

[2] D. den Hertog, C. Roos, "A survey of search directions in interior point methods for linear programming", Report 89-65, Delft University of Technology, The Netherlands (1989).

[3] D. den Hertog, C. Roos, T. Terlaky, "A potential reduction variant of Renegar's short-step path-following method for linear programming", Report 90-14, Delft University of Technology, The Netherlands (1990).

where

$$\begin{aligned} a_0 &= v_1^2 v_2^2 - \mu^2/4, \\ a_1 &= 2(v_1 v_3 v_2^2 + v_2 v_4 v_1^2 + \mu/4), \\ a_2 &= v_2^2 v_3^2 + v_1^2 v_4^2 + 4 v_1 v_2 v_3 v_4 - 1/4, \\ a_3 &= 2(v_1 v_3 v_4^2 + v_2 v_4 v_3^2), \\ a_4 &= v_3^2 v_4^2, \end{aligned} \tag{3.7}$$

and each "product" of four vectors "*abcd*" stands for the sum $\sum_{j=1}^n (a_j b_j c_j d_j)$.

The maximal permissible steplength δ is therefore given by the largest positive zero of p that is less than one[5]. (This $\delta = \delta^k$ can be computed in constant time.) Thus we obtain a simple modification of the model algorithm by changing Step 2 to

2′. $\quad \mu^{k+1} := \mu^k(1 - \delta^k)$.

3.3. Achievable Gains

The analysis of the model algorithm in Section 2.3 has shown that a steplength of $\delta = \mu/(4\sqrt{n})$ is possible in the worst case, i.e. that $p(\delta) < 0$ for $\delta \in (0, \mu/(4\sqrt{n}))$. We now examine how long the step might be on average. We will call a vector v *well balanced* if

$$\|v\|_\infty \le \frac{\kappa}{\sqrt{n}} \|v\|_2$$

for some small constant $\kappa > 1$. This excludes the case that v has only a few large components.

Let us observe first that the vector s^k is well balanced, since $1/\sqrt{2} \le s_j^k \le \sqrt{2}$ from the estimates (2.8) and (2.9).

To find how large we may expect δ to be in the "best case" we assume that the projections $v_3 = \Pi_N s^k$ and $v_4 = s^k - v_3$ are also well balanced, which essentially excludes the case that some of the components of s^k are magnified by more than a constant factor during the orthogonal projection Π_N. This allows a tight bound on the size of the coefficients of $p(\delta)$ by the following Lemma.

Lemma

1. If a and b are well balanced, then $abcd \le \frac{\kappa^2}{n} \|a\|_2 \|b\|_2 \|c\|_2 \|d\|_2$.
2. If a is well balanced, then $abcd \le \frac{\kappa}{\sqrt{n}} \|a\|_2 \|b\|_2 \|c\|_2 \|d\|_2$.

Proof

1. $\sum a_j b_j c_j d_j \le \max |a_j| \max |b_j| \sum c_j d_j \le \max |a_j| \max |b_j| \; \|c\|_2 \|d\|_2 \le \frac{\kappa^2}{n} \|a\|_2 \|b\|_2 \|c\|_2 \|d\|_2$. (The second inequality is the Cauchy-Schwarz inequality, and the third uses that a and b are well balanced.)

[5]It is easy to see that if p is nonnegative in $[0,1]$ we may choose $\delta = 1$ and thus find the solution of (P) in one step. Unfortunately, this will not happen in general, and so we assume that p has a zero in $(0,1)$.

An Interior Trust Region Method for Linearly Constrained Optimization

Clovis C. Gonzaga
Dept. of Systems Engineering and Computer Sciences
COPPE-Federal University of Rio de Janeiro
Cx. Postal 68511
21945 Rio de Janeiro, RJ
Brazil

April, 1991

Abstract

Trust region methods for constrained optimization usually work on spherical regions intersected with the feasible set. In this paper we explore the shape of the trust regions to generate ellipsoidal regions adapted to the shape of the feasible set, departing from the experience gained in interior point methods for linear programming. The resulting algorithms generate sequences of points in the interior of the feasible set.

Keywords: Trust region methods, Nonlinear programming, Interior point methods.

1 Introduction

In this short paper we study trust regions for linearly constrained minimization problems, with no convergence proofs. The paper has two purposes: first, to describe the construction of the regions and the manipulation of their shapes and sizes; second, to show how the approach leads to the constructions of Karush-Kuhn-Tucker multiplier estimates that may have interest in themselves. The complete results will appear in a forthcoming paper.

Let $f : I\!R^n \mapsto I\!R$ be a function of class C^2. The linearly constrained nonlinear programming problem is stated as:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax \;\le\; b \end{array} \tag{1}$$

where $b \in I\!R^m$, A is an $m \times n$ full-rank matrix.

We do not require that $m \geq n$ (in contrast with the usual linear programming practice). We assume that an initial strictly feasible solution x^o is at hand, and that the *effective feasible set*

$$S = \{x \in I\!R^n \mid Ax \leq b, f(x) \leq f(x^o)\}$$

is bounded, with interior $S^o \neq \emptyset$.

An interior trust region method is an algorithm in which each iteration starts with a point $x^k \in S^o$ and generates

(i) A quadratic (or linear) model $q_k(\cdot)$ for the function variations around x^k

$$h \in I\!R^n \mapsto q_k(h) \approx f(x^k + h) - f(x^k),$$

(ii) A trust region Ω_k around the origin, in which the model $q_k(\cdot)$ is believed to be reliable,

(iii) A step given by an optimal solution of the *trust region problem*

$$h^k \in \operatorname*{argmin}_{h \in \Omega_k} q_k(h),$$

such that $x^k + h^k \in S^o$.

The reliability of the trust region is measured by the ratio

$$\rho(h^k) = \frac{f(x^k + h^k) - f(x^k)}{q(h^k)}. \tag{2}$$

The region is acceptable if $\rho(h^k) \geq 0.25$.

The core of the algorithm is the construction of an acceptable region Ω_k, as large as possible, such that $x^k + h^k \in S^o$. This is done by an internal iterative

the path of centers. This implies that the second derivative of $x(\mu)$ must be large near μ^k.

Similar considerations also hold for "non-central" points $Xz = \mu w$ where $w \neq e$ is a positive weight vector. For estimates about the curvature of $x(\mu)$ we refer to [18].

3.2. An Adaptive Method

If the path of centers is "well behaved" we do not anticipate the worst case—that the infinity norm of the residual is increased by a factor of $O(\sqrt{n})$—in each step (see [13]). The hope that the estimates used in the model algorithm are too pessimistic in the average case suggests the following *adaptive* choice of the steplength.

Let us again take $\beta = \frac{1}{2}$ as in the model algorithm. Given x^k, y^k, z^k and $\mu = \mu^k$ such that

$$\|r^k\|_2 = \|X^k z^k - \mu e\|_2 \leq \frac{\mu}{2},$$

we wish to determine $\bar{\mu} := \mu^{k+1}$ as small as possible such that the residual r^{k+1} following the Newton step for finding $x(\bar{\mu}), y(\bar{\mu}), z(\bar{\mu})$ satisfies the analogous bound

$$\|r^{k+1}\|_2 = \|X^{k+1} z^{k+1} - \bar{\mu} e\|_2 \leq \frac{\bar{\mu}}{2}, \tag{3.3}$$

where $x^{k+1} = x^k + \Delta x^k$ and $z^{k+1} = z^k + \Delta z^k$ are obtained from x^k and z^k by (2.2) with $r = \bar{r}^k := X^k z^k - \bar{\mu} e$. If we set $\bar{\mu} := \mu(1 - \delta)$ (for some $\delta = \delta^k > 0$), then it follows that $\bar{r}^k = r^k + \delta \mu e$. The component of $\Delta x^k = D\Pi_N D X^{-1} \bar{r}^k$ resulting from r^k is often referred to as the *centering direction* (it brings the iterate closer to the center $x(\mu)$), while the component resulting from δe is aimed at reducing the duality gap and is usually referred to as the *affine scaling direction.* By (2.7) the next residual is given by $r^{k+1} = \tilde{\Delta X}^k \tilde{\Delta z}^k$, where

$$\tilde{\Delta x}^k = \Pi_N D X^{-1} \bar{r}^k, \qquad \tilde{\Delta z}^k = \Pi_R D X^{-1} \bar{r}^k, \tag{3.4}$$

(where $x = x^k$, $z = z^k$, $D^2 = XZ^{-1}$). If we define $q^k = DX^{-1} r^k$, $s^k = DX^{-1} e$ and

$$v_1 = \Pi_N q^k, \qquad v_2 = q^k - v_1, \qquad v_3 = \Pi_N s^k, \qquad v_4 = s^k - v_3,$$

then

$$\tilde{\Delta x}^k = v_1 + \delta v_3, \qquad \tilde{\Delta z}^k = v_2 + \delta v_4.$$

This shows that (3.3) is equivalent to

$$\|r^{k+1}\|_2^2 = \|(V_1 + \delta V_3)(v_2 + \delta v_4)\|_2^2 \leq \frac{\bar{\mu}^2}{4} = \frac{\mu^2 - 2\delta\mu + \delta^2}{4}, \tag{3.5}$$

which in turn is equivalent to

$$p(\delta) := a_0 + a_1 \delta + a_2 \delta^2 + a_3 \delta^3 + a_4 \delta^4 \leq 0, \tag{3.6}$$

3.1. Understanding the Analysis

The purpose of the following analysis is to shed some light on this apparent "incompatibility" of norms. Assume for the moment that for some $\mu = \mu^k$ the exact center x, y, z is given. (A "central" analysis is particularly interesting since it has been observed (see e.g. [2]) that at the center all the search directions suggested for interior-point algorithms since Karmarkar [5] are equivalent.) At the center, D, r and q reduce to

$$D^2 = XZ^{-1} = \frac{1}{\mu}X^2, \qquad r = (\mu^k - \mu^{k+1})e =: \delta\mu e, \qquad q = \delta\sqrt{\mu}e, \tag{3.1}$$

where $\delta \in (0,1)$. Using the Newton step for finding $x(\mu^{k+1}), y(\mu^{k+1}), z(\mu^{k+1})$ via (2.2) we obtain (using (2.5))

$$\Delta x = -\delta X \Pi_N e. \tag{3.2}$$

What is the largest possible steplength δ that keeps the next iterate strictly feasible? Clearly, $x + \Delta x > 0$ is maintained iff $\delta \Pi_N e < e$. As mentioned in (2.12′), the orthogonal projection Π_N may increase some components of e by $O(\sqrt{n})$. A simple example to illustrate this worst case is $A = (-\sqrt{n}, 1, \ldots, 1) \in I\!R^{1\times(n+1)}$, $x = z = c = e = (1, \ldots, 1)^T \in I\!R^{n+1}$, $y = 0$ and $b = n - \sqrt{n}$. These vectors satisfy (2.1) with $\mu = 1$, i.e. x, y, z form a center, and $\Pi_R e = \frac{n-\sqrt{n}}{2n}(-\sqrt{n}, 1, \ldots, 1)^T$, and hence $\Pi_N e = \frac{n+\sqrt{n}}{2n}(\sqrt{n}, 1, \ldots, 1)^T$. The infinity norms of $\Pi_R e$ and $\Pi_N e$ are larger than $\|e\|_\infty$ by a factor of (nearly) $\sqrt{n}/2$.

We emphasize that in this situation the search direction, even when secured by a linesearch, does not allow a "long" step[4]! This is particularly interesting since all interior-point methods (also the "long-step" methods) generate the same search direction at this point. To guarantee feasibility of $x + \Delta x$ in the worst case for a fixed steplength δ we therefore need $\delta < 2/\sqrt{n}$, almost as in the model algorithm. Thus the factor $\sqrt{n}$ can be interpreted as the "magnification" of certain components of a vector during its orthogonal projection, rather than a result of "incompatibility" of the norms or a result of our inability to prove the best possible bound.

At this point we may ask for a geometrical interpretation of the orthogonal projection and of the fact that it sometimes magnifies certain components of the vector e, i.e. that $\|\Pi_N e\|_\infty = O(\sqrt{n}) \gg 1 = \|e\|_\infty$. Given a point x, y, z on the path of centers, it is well known that $-X\Pi_N e$ is the direction of the tangent (in the primal space) to $x(\mu)$. (To see this, simply differentiate (2.1) with respect to μ and observe that the resulting equation is exactly (2.2) with $r = -e$ and with the derivatives x', z' in place of Δx, Δz.) Now, if some components of $\Pi_N e$ are $\gg 1$ (the case $\ll -1$ is similar with z in place of x) then the linear approximation $x(\mu(1-\delta)) \approx x - \delta\Delta x$ to the path of centers has a short "trust region", since even for small δ (any $\delta > 1/\|\Pi_N e\|_\infty$) the points $x - \delta\Delta x$ are not feasible, i.e. not near

[4]The conventional notion of short and long steps is somewhat confusing. Of course one may decrease the complementarity parameter μ by a long step at this point (and this is the general understanding of a long step). What this example really shows is that the variables x, y, z may not be able to follow such a long step; i.e. a damped Newton step is required, and the actual complementarity gap is only reduced by a small amount.

process, in which both the shape and the size of the region are adjusted. Note that although we referred to the method as "interior trust region", Ω_k does not have to be contained in S, as long as the result of the trust region step belongs to S^o. Figure 1 below shows a family of such ellipsoids.

There is a vast literature on trust region methods for unconstrained problems, surveyed in Moré [18] and in the textbooks by Fletcher [10] and by Dennis and Schnabel [15]. This last reference, as well as Moré and Sorensen [17] have detailed studies of the solution of the trust region problem, and global convergence for a wide variety of variants is well established (see for instance the cited textbooks and Shultz, Schnabel and Byrd [20]).

A sample of the main approaches for problems with nonlinear equality constraints is found in the papers by Powell and Yuan [19], by Celis, Dennis and Tapia [6] and by Byrd, Schnabel and Shultz [5]. Problems with inequality constraints are treated in Gay [11], Conn, Gould and Toint [7], Moré [16] and in Burke, Moré and Toraldo [4].

All methods in these references use at each iteration a spherical trust region, and adjust its radius with basis on the ratio (2). Ellipsoids can be used instead of spheres by rescaling the problem, but this fact is not used explicitly in the algorithms; convergence proofs allow ellipsoidal trust regions as long as the ellipsoids do not degenerate.

In methods for equality constrained problems, each iteration starts at a non feasible point and generates a ball Ω_k. The actual trust region is the intersection of $x^k + \Omega_k$ and a linearization of the feasible set, usually associated with a sequential quadratic programming algorithm. Then $q_k(\cdot)$ is a model for a Lagrangean function.

Methods for inequality constraints start each iteration at a feasible point usually in the boundary of S, and use as effective trust region the intersection of S and a ball $x^k + \Omega_k$. The trust region problem in such a region is very difficult. The computation of a step becomes manageable for linearly constrained problems if one renounces to the minimization of a quadratic model and resorts to projected gradients, as introduced by Bertsekas [3] for problems with simple bounds. This is the approach followed by Conn, Gould and Toint [7] (for simple bounds), Moré [16] and Burke, Moré and Toraldo [4] (for linear and convex constrains). When the constraints are not simple bounds, the projected gradient method requires the computation of minimum distance problems.

In all these methods, only the size of the trust regions is manipulated. The shape of Ω_k is not dependent on the shape of the feasible set. We now review briefly the affine-scaling algorithm, where the dominant characteristic of Ω_k is the shape.

Remark on notation: In the study of one iteration of the algorithm, we drop the iteration index k wherever this is not confusing, denoting the resulting direction by $\hat{h}$. The slack vector is $z = b - Ax^k$, and we define $Z = \operatorname{diag}(z_1, \ldots, z_m)$. The letter e will be reserved for the vector of ones $e = [1 \cdots 1]^T$ with dimension determined by the context.

The affine-scaling method for linear and convex programming

This method was developed by Dikin [8,9], who proved its convergence for linear programming under a primal non-degeneracy hypothesis. Most studies of the affine-scaling method are made for problems stated with linear equality constraints and non-negative variables, but an equivalent formulation can be made for inequality constrained problems, as we shall do now. This is usually called "dual formulation", and was first used by Adler, Resende, Veiga and Karmarkar [1].

Consider problem (1) with a linear objective $f(x) \equiv c^T x$. Let $x \in S^o$ be given, with slacks $z = b - Ax > 0$. A step h from x is feasible if $x + h \in S$, or equivalently if $Ah \leq z$, or still, if $Z^{-1}Ah \leq e$. It follows that the (maybe unbounded) ellipsoidal trust region

$$\Omega = \{h \in I\!R^n \mid \|Z^{-1}Ah\| \leq 1\} \tag{3}$$

is entirely contained in S.

Dikin's algorithm is precisely the usual trust region algorithm applied to the linear programming problem with this trust region. Each iteration starts at an interior point x^k, computes

$$h^k = \operatorname*{argmin}_{h \in I\!R^n}\{c^T h \mid \|Z^{-1}Ah\| \leq 1\} \tag{4}$$

and sets $x^{k+1} := x^k + h^k$.

The algorithm converges globally to an optimal solution for problems satisfying the primal non-degeneracy assumption, which requires that the active constraint gradients are linearly independent at all boundary points. Dikin proves that if the algorithm finds a boundary point at any iteration then it is optimal and the method can be stopped. Dikin's proof was reworked by Vanderbei and Lagarias [21], who divide it in a sequence of clear steps.

More efficient steps can be taken by increasing the radius of the trust region by setting $\Omega = \{h \in I\!R^n \mid \|Z^{-1}Ah\| \leq \alpha\}$, with $\alpha \geq 1$. It is easy to see that for a linear objective this is equivalent to a line search along the direction described above, and leads to large steps. Efficient steps are constructed by stretching h^k to a fixed ratio (usually around 99%) of the maximum step to the boundary. The algorithm with large steps was independently developed by Barnes [2] and by Vanderbei, Meketon and Freedman [22], who proved its convergence both primal and dual non-degeneracy assumptions. Gonzaga [13] followed the methodology in Vanderbei and Lagarias [21] to find a convergence proof for the algorithm with arbitrarily large steps and primal non-degeneracy.

The affine-scaling algorithm for linear equality constrained convex problems with non-negative variables takes $c = \nabla f(x^k)$ at each iteration and determines the step length by a line search on $f(x^k + \lambda h^k)$. Its global convergence was proved by Gonzaga and Carlos [14].

Hence, the relative error $\|Xz - \mu e\|_2/\mu = \|r\|_2/\mu$ is squared after each iteration of Newton's method[2].

2.4. A Model Algorithm

The above analysis allows us to formulate the following classical model algorithm. Let $x^0 > 0, y^0, z^0 > 0$ and μ^0 be given such that $Ax^0 = b$, $A^T y^0 + z^0 = c$, $X^0 z^0 - \mu^0 e = r^0$ and $\|r^0\|_2/\mu^0 \leq \frac{1}{2}$. (Again this assumes that the initial point is moderately close to the path of centers.) Further, let some desired accuracy ϵ be given. Set $k = 0$.

1. Perform one Newton step via (2.2) to obtain x^{k+1}, y^{k+1}, z^{k+1}.
2. Decrease μ^k to $\mu^{k+1} := \mu^k(1 - \frac{1}{4\sqrt{n}})$.
3. Set $k = k + 1$.
4. If $\mu^k \leq \epsilon/n$ then stop, else go to Step 1.

Since $\|r^{k+1}\|_2 \leq \|\tilde{r}^k\|_2 + \frac{1}{4\sqrt{n}}\|e\|_2 \leq \frac{1}{4} + \frac{\sqrt{n}}{4\sqrt{n}} \leq \frac{1}{2}$ (with $\tilde{r}^k$ as in (2.13)) it follows by induction that all residuals r^k fulfill (2.9) with $\beta = \frac{1}{2}$, and hence (by the results of Section 2.2) the method is well defined. Since the complementarity parameter μ converges linearly to zero (Step 2), the method terminates after $O(\sqrt{n}\log\frac{n\mu^0}{\epsilon})$ iterations. This result is well known [7].

3. IMPROVEMENTS

We shall now examine possible improvements of the convergence analysis. For this purpose, it is useful to analyze which of the estimates leading to (2.13) are sharp, and which ones can possibly be tightened.

Note that the relations (2.7), (2.8) and (2.12) are exact. However the bounds (2.10) and (2.11) could also be established under the weaker assumption

$$\|r\|_\infty \leq \beta\mu. \tag{2.9'}$$

(It is interesting to note that for any feasible x, y, z one can always find some $\mu > 0$ such that (2.9') is satisfied with $\beta = 1$.) Also, the bound (2.13) is based on the inequality $\|St\|_2 \leq \|s\|_2\|t\|_2$, which also holds in terms of the infinity norm: $\|St\|_\infty \leq \|s\|_\infty\|t\|_\infty$. Finally, to guarantee feasibility of $x + \Delta x, z + \Delta z$, the inequalities $\|\tilde{\Delta x}\|_\infty$, $\|\tilde{\Delta x}\|_\infty \leq \tilde{\beta}$ would suffice in place of (2.12)[3]. Unfortunately, an estimate corresponding to (2.12) in terms of the infinity norm only holds in the form

$$\|\tilde{\Delta x}\|_\infty \leq \sqrt{n}\tilde{\beta}\cos\theta, \qquad \|\tilde{\Delta z}\|_\infty \leq \sqrt{n}\tilde{\beta}\sin\theta. \tag{2.12'}$$

We "lose" a factor of $\sqrt{n}$ both times. Stating (2.13) in terms of the infinity norm would thus "lose" a factor of n.

[2]Note that (2.13) and the first equation in (2.7) imply that $(X + \Delta X)(z + \Delta z) > 0$. One way to verify strict feasibility (i.e. $x + \Delta x > 0$ and $z + \Delta z > 0$) is to show that $\|X^{-1}\Delta x\|_2 \leq 1$. By (2.12), $\|\tilde{\Delta x}\| \leq \tilde{\beta}$, and so $\|X^{-1}\Delta x\|_2 = \|X^{-1}D\tilde{\Delta x}\|_2 = \|(\sqrt{R + \mu I})^{-1}\tilde{\Delta x}\|_2 \leq \tilde{\beta}/\sqrt{\mu(1-\beta)} \leq 1$.

[3]An analysis in terms of the infinity norm would be particularly interesting, since by the update in Step 2 of the model algorithm the residual r^{k+1} is obtained from $\tilde{r}^k$ by $r^{k+1} = \tilde{r}^k + (\mu^k - \mu^{k+1})e$, and the infinity norm of e is smaller than its 2 norm: $\|e\|_\infty = \frac{1}{\sqrt{n}}\|e\|_2$.

This becomes obvious as we observe that (2.3) is equivalent to (2.4)–(2.6). First,

$$DA^T\Delta y = \Pi_R q \tag{2.4}$$

(since DA^T has maximal rank), where Π_R is the orthogonal projection onto the range $R(DA^T) = \{y \mid y = DA^T w \text{ for some } w \in I\!R^m\}$ of DA^T. Let Π_N denote the orthogonal projection onto the null space $N(AD)$ of AD. Since $N(AD)$ is the orthogonal complement to $R(DA^T)$, i.e. $\Pi_R + \Pi_N = I$, it follows that

$$\Delta x = D(\Pi_R q - q) = -D\Pi_N q, \tag{2.5}$$

and finally,

$$\Delta z = -D^{-1}(q + D^{-1}\Delta x) = -D^{-1}\Pi_R q. \tag{2.6}$$

It is straightforward to verify that (2.4)–(2.6) satisfy (2.2).

2.3. Analysis of the Newton Step

The residual after executing the above Newton step (without damping[1]) is given by

$$\begin{aligned} \tilde{r} &= (X + \Delta X)(z + \Delta z) - \mu e = Xz + X\Delta z + Z\Delta x + \Delta X \Delta z - \mu e \\ &= Xz - r + \Delta X \Delta z - \mu e = \Delta X \Delta z = \tilde{\Delta X}\tilde{\Delta z}, \end{aligned} \tag{2.7}$$

where $\tilde{\Delta x} = -D^{-1}\Delta x = \Pi_N q$ and $\tilde{\Delta z} = -D\Delta z = \Pi_R q$. Note that

$$DX^{-1} = \sqrt{X^{-1}Z^{-1}} = (\sqrt{R + \mu I})^{-1} \tag{2.8}$$

(using $Xz = r + \mu e$). From (2.7)–(2.8) and the definition of q in (2.3) we can readily derive the classical convergence results about the primal-dual method as given in [7] or [16]. Assume that $x > 0, z > 0$ and y are given such that we can find a positive μ for which

$$\|r\|_2 \le \beta\mu \tag{2.9}$$

for some $\beta \in [0, \frac{1}{2}]$. (This means we are assuming that x, y, z is "moderately close" to the path of centers.) It follows that

$$\|(R + \mu I)^{-1}\|_2 \le \frac{1}{\mu(1-\beta)}, \tag{2.10}$$

so that

$$\|q\|_2 \le \beta\sqrt{\mu/(1-\beta)} =: \tilde{\beta}, \tag{2.11}$$

and

$$\|\tilde{\Delta x}\|_2 = \tilde{\beta}\cos\theta, \qquad \|\tilde{\Delta z}\|_2 = \tilde{\beta}\sin\theta, \tag{2.12}$$

where θ is the angle between q and $\Pi_N q$. Finally by (2.7), (2.12) and (2.11),

$$\|\tilde{r}\|_2 = \|\tilde{\Delta X}\tilde{\Delta z}\|_2 \le \tilde{\beta}^2 \cos\theta \sin\theta \le \frac{1}{2(1-\beta)}\mu\beta^2 \le \mu\beta^2. \tag{2.13}$$

[1]Analyzing full Newton steps only is not a severe restriction, since a damped Newton step can be viewed as a full Newton step for a linearly perturbed system of equations (the original system of nonlinear equations to which a linear perturbation is added).

2 Karush-Kuhn-Tucker multiplier estimates

In this section we show how multiplier estimates can be associated to a point $x \in S^o$ in connection with the trust region (4). Multiplier estimates are studied in many references, usually in association with a choice of active set, with augmented Lagrangeans or penalty methods. For a nice reference see for instance Gill and Murray [12]. Our multiplier estimates follow the basic approach of choosing a minimum norm multiplier vector among a set of candidates.

Multiplier estimates are constructed from a linearization of the problem (1) about a point $x^k \in S^o$. They will be dual solutions for the linearized problems. Let c be either equal to $\nabla f(x^k)$ or an approximation to this gradient, and consider the linear programming problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \;\le\; b \end{array} \tag{5}$$

Here we shall assume that $A \in I\!R^{m \times n}$ has rank n. This assumption was not made for problem (1), and will be discussed below.

A vector $w \in I\!R^m$ is a dual solution if

$$A^T w = -c.$$

It will be called a feasible dual solution if it also satisfies $w \ge 0$. It is an optimal dual solution if it is feasible and satisfies the complementarity condition

$$Zw = 0.$$

A multiplier estimate must be a dual solution. Our hypothesis that A has rank n ensures that dual solutions exist. A method for choosing estimates among the available dual solutions must also guarantee that the estimates approach the other conditions (feasibility and complementarity) near an optimal solution. On the other hand, multiplier estimates are useful in the algorithms for providing stopping rules or for choosing active sets (not discussed in this paper). They will be essential in convergence proofs: if a method generates multiplier estimates that converge to optimal dual solutions, then the sequences generated by the method converge to points satisfying optimality conditions.

Among the vectors that satisfy $A^T w = -c$, we shall choose the vector that minimizes some norm. Any vector $w \in I\!R^m$ can be decomposed as

$$w = Av + \gamma,$$

where $v \in I\!R^n$ and $\gamma \in \mathcal{N}(A^T)$, the null space of A^T. It follows that

$$-c \;=\; A^T w \;=\; A^T Av.$$

Since A has rank n, $A^T A$ is non-singular. Hence $v = -(A^T A)^{-1}c$ and

$$w = -A(A^T A)^{-1}c \tag{6}$$

provides a minimum-norm dual solution using the Euclidean norm.

Scaling

Different choices are given by scaling the problem constraints and then taking the minimum-norm dual solution. Let D be a positive diagonal matrix. We shall scale the slack variables by rewriting the problem as

$$\begin{array}{lrcl} \text{minimize} & c^T x & & \\ \text{subject to} & DAx & \leq & Db \end{array} \tag{7}$$

Now $\tilde{w}$ is a dual solution for (7) if

$$A^T D \tilde{w} = -c$$

and then $w = D\tilde{w}$ is a dual solution for (5). Using (6), $\tilde{w} = -DA(A^T D^2 A)^{-1} c$ and hence

$$w = -D^2 A (A^T D^2 A)^{-1} c. \tag{8}$$

Our choice for D at an interior point x with slacks $z > 0$ will be $D = Z^{-1}$, and the resulting dual solution is

$$w = -Z^{-2} A (A^T Z^{-2} A)^{-1} c. \tag{9}$$

We conclude that the expression above associates a dual solution to each interior point in a problem with a rank n constraint matrix. These dual estimates have the nice property that for $c = \nabla f(x)$, if x approaches a non-degenerate optimal solution of problem (1), then w approaches the unique optimal Karush-Kuhn-Tucker multipliers at the optimal solution. This qualifies our choice of w as legitimate multiplier estimates.

It is apparent that these multiplier estimates are related to the affine-scaling trust regions described in the previous section, but the discussion of this relationship will have to wait for the presentation of the actual trust regions to be used.

3 The trust regions

In this section we show how to construct trust regions for problem (1) (without the assumption that the rank of A is equal to n). Consider as usual problem (1), an interior point x^k with slacks z. Our trust regions will take into account two requirements:

(i) **Shape and feasibility:** Large regions are obtained as in the affine-scaling method seen in (3) by choosing

$$\Omega_\alpha = \{h \in I\!R^n \mid \|Z^{-1} A h\| \leq \alpha\},$$

with $\alpha \geq 1$. For $\alpha = 1$ the region is contained in S, and has a shape that is well adapted to the feasible set.

2.1. Some Known Theory

Following the motivation given in [8], the "penalized" Lagrangian corresponding to (P) with a logarithmic barrier term for the inequality constraints is given by

$$L_\mu(x,y) = c^T x - y^T(Ax - b) - \mu \sum_{j=1}^{n} \ln x_j,$$

where $\mu > 0$. The necessary (and sufficient) conditions for a stationary point of L_μ are

$$F_\mu(x,y,z) := \begin{pmatrix} Ax - b \\ A^T y + z - c \\ Xz - \mu e \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad x, z > 0, \tag{2.1}$$

where $X = \operatorname{diag}(x_1, \ldots, x_n)$, $e = (1, \ldots, 1)^T \in I\!R^n$, and z is an auxiliary variable. (In the following a small letter and the same letter capitalized—like x and X—will always stand for a vector and its corresponding diagonal matrix.) The unique solution $x(\mu), y(\mu), z(\mu)$ of (2.1) is the *analytic center* as defined in [17]. It is obvious that $x(\mu)$ is primal feasible and $y(\mu)$ is dual feasible. From (2.1) it immediately follows that the duality gap between $x = x(\mu)$ and $y = y(\mu)$ is bounded by $n\mu$:

$$\begin{aligned} n\mu = \|\mu e\|_1 = \|Xz\|_1 = \|X(c - A^T y)\|_1 \\ \leq x^T(c - A^T y) = c^T x - (Ax)^T y = c^T x - b^T y. \end{aligned}$$

As first proved in [9], $x(\mu)$ and $y(\mu)$ converge to primal and dual optimal solutions as μ tends to zero. (See also [1].)

2.2. The Newton Step for Finding the Center

Suppose that $x > 0$, $z > 0$ and y are given such that $Ax = b$ and $A^T y + z = c$. For $\mu > 0$ we define the residual

$$r := Xz - \mu e,$$

so that $F_\mu(x,y,z) = (0, 0, r^T)^T$. The Newton step for finding the analytic center $x(\mu)$, $y(\mu)$, $z(\mu)$ is then given by

$$\begin{aligned} A\Delta x &= 0, \\ A^T \Delta y + \Delta z &= 0, \\ Z\Delta x + X\Delta z &= -r. \end{aligned} \tag{2.2}$$

Note that if we were able to solve the (nonlinear) system (2.2′) where in the last equation of (2.2) the term $\Delta X \Delta z$ is added on the left-hand side, then we could find the exact center as $(x + \Delta x,\ y + \Delta y,\ z + \Delta z)$, since (2.2′) is then equivalent to (2.1).

Define the positive diagonal matrix D by $D^2 := XZ^{-1}$. Then (2.2) can be solved via

$$\begin{aligned} q &= DX^{-1} r, \\ \Delta y &= (AD^2 A^T)^{-1} ADq, \\ \Delta x &= D^2 A^T \Delta y - Dq, \\ \Delta z &= -D^{-1} q - D^{-2} \Delta x. \end{aligned} \tag{2.3}$$

implementations of interior-point algorithms that yield very fast convergence for (most) numerical examples. In allowing long steps, however, the theoretical complexity of these investigations degrades. This apparently paradoxical behavior is due to the fact that for large steps the Newton iterates may no longer be feasible, and therefore the method has to use damped Newton iterations with a damping factor $\alpha \in (0,1)$. Loosely speaking, the iterates may not be able to "follow" such a long step δ in the parameter μ. The goal of the present paper is to improve the theoretical complexity in terms of the number of iterations. We will consider an adaptive choice of the steplength δ such that the full Newton iterate may always be used. Similar choices of an adaptive steplength have been suggested before, see e.g. [20, 14, 12], but to our knowledge the resulting improvement in complexity has not yet been analyzed. In [14], Mizuno *et al.* report numerical experiments for their adaptive algorithm (B'). The experiments indicate that the number of iterations can indeed be reduced significantly by an adaptive choice of the steplength.

By a thorough analysis of the integral over the curvature of the path of centers it was shown in [18] that for certain subclasses of linear programs the theoretical complexity could be reduced below $O(\sqrt{n}\log\frac{1}{\epsilon})$ iterations, and in [13] a probabilistic analysis showed that the "anticipated" number of iterations could be reduced below $O(\sqrt{n}\log\frac{1}{\epsilon})$, but in a general worst-case analysis this complexity is (still) the state of the art. The analysis presented here is particularly simple. It also illuminates why the theoretical complexity could not be improved so far by any analysis that focuses on the worst case in a single iteration, rather than examining a sequence of iterations.

In Sections 2.1–2.4 we present a proof of convergence for the method of centers. The result is well known (see e.g. [7]), but the analysis is new and allows us to explain in Section 3.1 where the factor $\sqrt{n}$ comes from, and to analyze a new adaptive method in Section 3.2 that automatically chooses large steps if Newton's method for finding the center converges "well", and takes short steps otherwise.

2. THE PROBLEM AND A SIMPLE METHOD

The problem under consideration is the linear program

$$\min_{x\in S} c^Tx, \quad S = \{x \geq 0 \mid Ax = b\}, \tag{P}$$

where $A \in I\!R^{m\times n}$, $x \in I\!R^n$, $b \in I\!R^m$. For brevity we assume that the relative interior of the feasible set S is nonempty and bounded, and that the rows of A are linearly independent. The dual problem to (P) is given by

$$\max_{y\in I\!R^m} \{b^Ty \mid A^Ty \leq c\}. \tag{D}$$

(ii) **Size and model reliability:** as in all trust region models, the model $q(\cdot)$ adopted for the variations of $f(\cdot)$ around x^k can only be trusted in a region defined by some ball

$$\Omega_\Delta = \{h \in I\!R^n \mid \|h\| \leq \Delta\}.$$

Regions like $\Omega_\alpha \cap \Omega_\Delta$ might be very good, but these are not ellipsoids, and the trust region problem would be too difficult. Ellipsoidal regions are given by our choice of format

$$\Omega(\alpha,\Delta) = \{h \in I\!R^n \mid \frac{1}{\Delta^2}\|h\|^2 + \|Z^{-1}Ah\|^2 \leq \alpha^2\}, \tag{10}$$

where $\alpha \geq 1$ and $\Delta > 0$.

Figure 1 shows $\Omega(\alpha,\Delta)$ for a problem in which A has rank n and S is a triangle (the two extra constraints in the figure will be explained below). The region grows from a small sphere for a small Δ and $\alpha = 1$ while changing its shape (ellipsoids a,b in the figure), reaches the affine-scaling trust region for $\Delta = +\infty$ and $\alpha = 1$ (ellipsoid c), and then expands isometrically for $\Delta = +\infty$ and $\alpha > 1$ (ellipsoids d,e).

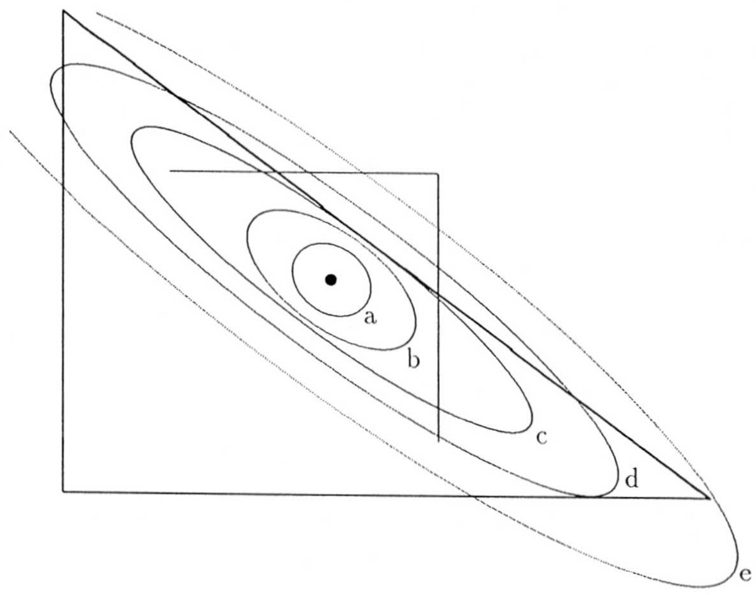

Figure 1: Trust regions and dummy constraints.

The values α and Δ should be managed as we illustrated in the figure. Small values of Δ may occur for $\alpha = 1$: in this case the model reliability predominates in the limitation of the trust region; $\alpha > 1$ should only be used for large values of Δ: in this case feasibility is the main determinant of the region size.

If A has rank n, then it is convenient to set $\Delta = +\infty$ whenever $\alpha > 1$. Otherwise, the affine-scaling region is unbounded, and a finite value of Δ (say $\Delta > 1$ for $\alpha > 1$) must be used.

The trust region definition can be rewritten by defining

$$\bar{A} = \begin{bmatrix} A \\ I \end{bmatrix} \quad , \quad \bar{Z} = \begin{bmatrix} Z & 0 \\ 0 & \Delta I \end{bmatrix} \quad , \quad \bar{z} = \begin{bmatrix} z \\ \Delta e \end{bmatrix} \quad , \tag{11}$$

where I is the $n \times n$ identity matrix and $e \in I\!R^n$. As we shall see in a moment, these entities provide data for a modified problem associated to x. Now,

$$\Omega(\alpha, \Delta) = \{h \in I\!R^n \mid \|\bar{Z}^{-1}\bar{A}h\| \leq \alpha\}, \tag{12}$$

or equivalently,

$$\Omega(\alpha, \Delta) = \{h \in I\!R^n \mid h^T\bar{A}^T\bar{Z}^{-2}\bar{A}h \leq \alpha^2\}. \tag{13}$$

Note that $\bar{Z}^{-1}\bar{A}$ has rank n for any $\Delta \in (0, +\infty)$.

The algorithm: the algorithm follows the scheme described in the introduction. Here we simply assume that an internal iterative process determines $\alpha \geq 1$ and $\Delta > 0$ such that the resulting step $\hat{h}$ satisfies $x^k + \hat{h} \in S^o$ and the region is acceptable, i.e., the ratio (2) satisfies $\rho(\hat{h}) \geq 0.25$. We must also guarantee that Δ is not too small, and this can be done by growing the region at least until either $\Delta > 1$ or $\Omega(1, 2\Delta)$ is not acceptable.

Optimality conditions for the trust region problem: Let the model function around x^k be given by

$$h \in I\!R^n \mapsto q(h) = g^Th + \frac{1}{2}h^TBh, \tag{14}$$

where $g = \nabla f(x^k)$ and B is either the Hessian matrix at x^k or a quasi-Newton approximation. The trust region problem is

$$\text{minimize } \{q(h) \mid h^T\bar{A}^T\bar{Z}^{-2}\bar{A}h \leq \alpha^2\}.$$

The optimality conditions for this problem imply that at an optimal solution $\hat{h}$

$$g + B\hat{h} + \mu\bar{A}^T\bar{Z}^{-2}\bar{A}\hat{h} = 0. \tag{15}$$

The case where $\mu = 0$ corresponds to an interior Newton iteration, and is easy. Let us analyse the case $\mu > 0$. In this case $\hat{h}$ is in the boundary of the trust region. Assume that the trust region problem has been solved, and set

$$c = g + B\hat{h},$$

AN ADAPTIVE PRIMAL-DUAL METHOD FOR LINEAR PROGRAMMING*

Florian Jarre[†] and Michael A. Saunders[‡]

June 12, 1991

Abstract

A simple analysis for an adaptive primal-dual method for linear programming is given. Starting from a pair of primal and dual feasible points near the path of centers, the method maintains a worst-case complexity of $O(\sqrt{n}\log\frac{1}{\epsilon})$ iterations to reduce the initial duality gap by a factor of ϵ. In contrast to other interior-point algorithms that share the same complexity ([15, 4] and many others), the algorithm proposed here allows an acceleration of the rate of convergence (up to a complexity of $O(\sqrt[4]{n}\log\frac{1}{\epsilon})$) if the problem is "well behaved".

Key words: linear programming, interior-point method, theoretical complexity, orthogonal projection

1. INTRODUCTION

We discuss interior-point algorithms that apply one iteration of Newton's method to a set of nonlinear equations involving a parameter $\mu > 0$, and then reduce μ by a certain amount before repeating the process. For $\mu = 0$ the solution of the set of nonlinear equations coincides with the solution of a given linear program. The reduction of μ is of the form $\mu := \mu(1-\delta)$ for some "steplength" $\delta \in (0, 1)$, where the focus is on the size of δ. The method of centers (e.g. [15]) is typical of "short-step" methods in which $\delta \leq 1/\sqrt{n}$ at every iteration, where n is the number of variables. Roughly speaking, the shortness of the steplength $\delta \leq 1/\sqrt{n}$ allows a proof that the Newton iterates remain strictly feasible, but for numerical implementations this rate of convergence is too slow when n is large. A "long-step" method is one for which $\delta > 1/\sqrt{n}$. (Typically δ is a constant independent of n.)

In the recent past a number of investigations have been presented that analyze long-step interior-point methods for linear programming (see [3, 6, 11, 16]). Their goal is to generalize the existing proofs of convergence of certain interior-point algorithms to wider classes of algorithms and to justify the correctness of existing

*This work was supported by a research grant from the Deutsche Forschungsgemeinschaft, and by the U.S. National Science Foundation Grant DDM-8715153 and the Office of Naval Research Grant N00014-90-J-1242.

[†]On leave from Institut für Angewandte Mathematik, University of Würzburg, 8700 Würzburg, (West) Germany.

[‡]Dept. of Operations Research, Stanford University, Stanford, CA 94305-4022

$S_1, \dots, S_\nu$, $S_j T_1, \dots, S_j T_{N-1}, j = 1, \dots, N$ form the rotational group for P. (cf [5, Chapter 7]). Denote these matrices by $U_1, \dots, U_m$. Under the action of these matrices each vertex of P is carried into every other vertex exactly ν times. It therefore follows from multiplying each side of (3.7) on the left by U_i and on the right by U_i^T, and summing over i, that

$$\begin{aligned} \frac{m}{n} I &= \sum_{j=1}^{N} p_j \sum_{i=1}^{m} U_i (x_j - c)(x_j - c)^T U_i^T \qquad (3.9) \\ &= \nu \sum_{j=1}^{N} p_j \sum_{i=1}^{N} (x_i - c)(x_i - c)^T = \nu \sum_{i=1}^{N} (x_i - c)(x_i - c)^T \end{aligned}$$

If we divide this equation by ν we obtain (3.7) with $p_j = \frac{1}{N}$. A similar argument gives $p_j = \frac{1}{N}$ in (3.8). This completes the proof of the theorem for tetrahedra, octahedra, and icosahedra. By taking $\nu = 1$ in (3.9) we also obtain the proof for polyhedra in $\Re^2$. Since the cube and the octahedron have the same rotational groups as the tetrahedron and the icosahedron, respectively, the proof is complete for all polyhedra in $\Re^2$ and $\Re^3$.

Theorem 2 has an interesting interpretation for the mechanics of solids. Clearly the first and second moments of inertia of a thin circular hoop of mass M about a line through the center of the hoop is independent of the direction of the line. Theorem 2 says that this remains true for the system obtained by concentrating the mass at N equally spaced points along the hoop. The theorem in $\Re^3$ gives an analogous result for the moments of inertia of a thin hollow sphere.

References

[1] F. John, "Extremal problems with inequalities as subsidiary conditions," in : Studies and Essays, presented to Richard Courant on his 60th birthday, January 8, 1948, *Interscience*, New York, 1948.

[2] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, Vol. 4, 1984, pp. 373-395.

[3] J. Kiefer and J. Wolfowitz, "The equivalence of two extremum problems," *Canadian Journal of Mathematics*, Vol. 12, pp. 363-366, 1960.

[4] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley and Sons, New York, 1986.

[5] V. I. Smirnov, *Linear Algebra and Group Theory*, Dover Publications, Inc., New York, 1961.

an approximation for $\nabla f(x + \hat{h})$. It follows from (15) that

$$c = -\mu \bar{A}^T \bar{Z}^{-2} \bar{A} \hat{h}, \qquad (16)$$

We must now relate the solution of the trust region problem to multiplier estimates.

A modified problem

Consider the following problem associated to point x^k:

$$\begin{array}{rrcl} \text{minimize} & f(x) & & \\ \text{subject to} & Ax & \leq & b \\ & x & \leq & x^k + \Delta e. \end{array} \qquad (17)$$

The constraint matrix and the slack vector at x^k for this problem are respectively $\bar{A}$ and $\bar{z}$, so that the affine-scaling ellipsoid is precisely $\Omega(1, \Delta)$. The extra restrictions can be seen as dummy constraints, since they are satisfied for any point $x + h$ such that $||h|| \leq \Delta$. Figure 1 shows these constraints associated to ellipsoid b.

Multiplier estimates: the constraint matrix $\bar{A}$ for problem (17) has rank n for any finite $\Delta > 0$. A multiplier estimate can be computed by expression (9):

$$\bar{w} = -\bar{Z}^{-2} \bar{A} (\bar{A}^T \bar{Z}^{-2} \bar{A})^{-1} c, \qquad (18)$$

where $c = \nabla f(x)$ or an approximated gradient.

The main results

Expressions (16) and (18) provide our main results relating trust region directions and multiplier estimates for the transformed problem. Using the fact that $||\bar{Z}^{-1} \bar{A} \hat{h} = \alpha||$ because $\hat{h}$ is in the boundary of the trust region (12), we obtain:

Substituting (16) into (18) and manipulating,

$$\bar{w} = \mu \bar{Z}^{-2} \bar{A} \hat{h}. \qquad (19)$$

Multiplying by $\bar{Z}$ and taking norms,

$$||\bar{Z} \bar{w}|| = \mu \alpha.$$

Multiplying (16) by $\hat{h}^T$,

$$c^T \hat{h} = \mu \alpha^2.$$

Merging these two expressions,

$$c^T \hat{h} = -\alpha ||\bar{Z} \bar{w}||, \qquad (20)$$

Setting $\bar{w}^T = [w^T\ u^T]$, where w and u are respectively the multiplier vectors for the original and dummy constraints, using (11), and restoring the iteration index k,

$$((c^k)^T h^k)^2 = \alpha_k^2(\|Z_k w^k\|^2 + \Delta_k^2\|u^k\|^2). \tag{21}$$

Finally, directly from (19),

$$Ah^k = \frac{1}{\mu_k} Z_k^2 w^k. \tag{22}$$

These last two expressions carry much information, and are the basis for the following convergence results:

• One must have $(c^k)^T h^k \to 0$, otherwise it is easy to show that $f(x^k) \to -\infty$.

• The case where $\Delta_k \to 0$ can only occur in non-convex problems, and is analysed as in unconstrained trust region methods. An accumulation point where this occurs must satisfy first and second order necessary optimality conditions.

• With these facts, (21) implies that $Z_k w^k \to 0$, showing that the complementarity condition is satisfied at any accumulation point of the sequence (x^k). Besides this, the multipliers u^k for the dummy constraints converge to zero.

• The greatest difficulties are in proving that the multipliers w^k have feasible (non negative) accumulation points. With the primal non-degeneracy hypothesis, w^k converge along any convergent subsequence of (x^k). The easiest case is the following:
If (x^k) converges to a point $\hat{x}$, then (w^k) converges to an optimal multiplier $\hat{w}$. The proof is immediate from (22): if any component w_i converges to $\hat{w}_i < 0$, then for large k we must have $A_i x^{k+1} = A_i x^k + A_i h^k < A_i x^k$, contradicting the fact that the constraint $A_i x^k \leq b_i$ must be active at the limit by complementarity.

We have not succeeded in finding a global convergence proof for general non-convex functions, but some results are already proved: any accumulation point satisfies optimality conditions if $f(\cdot)$ is convex; an accumulation point x^* satisfying a strong complementarity condition satisfies optimality conditions if either $\nabla^2 f(x^*)$ is positive definite in the tangent space at x^* or if it has a negative eigenvalue in this space. Thus the method can in principle find a non-optimal accumulation point, but then the sequence cannot be convergent, $f(\cdot)$ must be non-convex with a positive semidefinite Hessian on the tangent space at the point (under strong complementarity). In this unfortunate case, multiplier estimators are available and some of its components must be negative, thus detecting the failure of the method.

hyperplane through the centroid of P and let $d_1, \ldots d_N$ denote the signed distances from the vertices of P to H. Then

$$\sum_{j=1}^{N} d_j = 0$$

and

$$\sum_{j=1}^{N} d_j^2 = \frac{Nr^2}{n}.$$

Proof. Let $x_1, \ldots, x_N$ denote the vertices of P. Clearly the ellipsoid of least volume containing P is the inscribing ball for P. Thus in this case R in (2.2) is the identity matrix and we have

$$\frac{1}{n} I = \sum_{j=1}^{N} p_j (x_j - c)(x_j - c)^T \tag{3.7}$$

for some $c \in \Re^n$. Let u be a unit normal to the hyperplane H. Then

$$d_j = \pm\, r u^T (x_j - c), j = 1, \ldots, N$$

and it follows from Theorem 1 that

$$\sum_{j=1}^{N} p_j d_j = 0 \tag{3.8}$$

If we multiply each side of (3.7) on the left by ru^T and on the right by ru we obtain

$$\frac{r^2}{n} = \sum p_j d_j^2.$$

To complete the proof of the theorem we need only show that we can take each $p_j = \frac{1}{N}$ in (3.7) and (3.8).

Consider the rotations of P which carry P into itself. In $\Re^2$ there are N such rotations corresponding to rotations through angles of $0, \frac{2\pi}{N}, \frac{4\pi}{N}, \ldots, \frac{2(N-1)\pi}{N}$ radians. Each vertex of P is left fixed by exactly one of these rotations, namely the rotation through 0 radians.

Assume now that $P \subset \Re^3$. P is then a tetrahedron, a cube, an octahedron, a dodecahedron or an icosohedron, for these are the only regular polytopes in $\Re^3$. If P is a tetrahedron, an octahedron or an icosahedron, fix a vertex A of P and draw a line ℓ through A and the centroid of P. Let ν denote the number of rotations of P about ℓ that carry P into itself. $\nu = 3$ for the tetrahedron, 4 for the octahedron and 5 for the icosahedron. Let $S_1, \ldots, S_\nu$ be unitary matrices corresponding to these rotations. Let $T_1, \ldots, T_{N-1}$ denote unitary matrices corresponding to the rotations about c which carry P into itself and vertex A to the remaining $N-1$ vertices of P. The $m = \nu N$ matrices is a solution of (2.3). This proves the second conclusion in Theorem 1 if c is taken to be the center of the ellipsoid of least volume containing P. To obtain the first conclusion observe that the determinant $\det Q$ cannot be decreased by varying c. Therefore the gradient of this determinant with respect to c is 0. That is

$$\sum_{j=1}^{N} p_j(x_j - c) = 0$$

as claimed.

3 Moments of Inertia

We are now ready to make some observations about regular polytopes in $\Re^2$ and $\Re^3$. First consider an equilateral triangle P in $\Re^2$ with a line drawn through its centroid parallel to one of its sides. Let d_1, d_2, d_3 denote the signed distances from the vertices to the line as shown in Figure 1(a). Since the centroid divides each median in the ratio 1:2 we have

$$d_1 + d_2 + d_3 = 0. \tag{3.5}$$

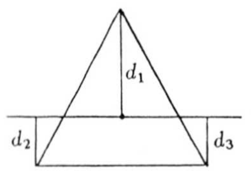

Figure 1(a)

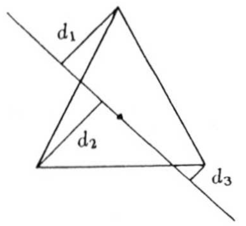

Figure 1(b)

Let r denote the radius of the inscribing circle for P. In Figure 1(a) we have $d_1 = \pm r$ and $d_2 = d_3 = \mp\frac{1}{2}r$. Therefore

$$d_1^2 + d_2^2 + d_3^2 = \frac{3}{2}r^2 \tag{3.6}$$

The following theorem gives a generalization of (3.5) and (3.6) to all regular polytopes in $\Re^2$ and $\Re^3$.

Theorem 2. Let P be any regular polytope in $\Re^n$, $n = 2$ or 3. Let N denote the number of vertices of P and let r denote the radius of the ball inscribing P. Let H be any

References

[1] I. Adler, M. Resende, G. Veiga, and N. Karmarkar. An implementation of Karmarkar's algorithm for linear programming. *Mathematical Programming*, 44, 1989.

[2] E. R. Barnes. A variation on Karmarkar's algorithm for solving linear programming problems. *Mathematical Programming*, 36:174–182, 1986.

[3] D. P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM Journal of Control and Optimization*, 20(2):221–246, 1982.

[4] J. Burke, J. Moré, and G. Toraldo. *Convergence Properties of Trust Region Methods for Linear and Convex Constraints*. Report ANL/MCS-TM-116, Argonne National Laboratory, Argonne, Illinois, 1988.

[5] R. Byrd, R. Schnabel, and G. Shultz. *A Trust Region Algorithm for Nonlinearly Constrained Optimization*. Report CU-CS-313-85, Dept. of Computer Science, University of Colorado at Boulder, Boulder, Colorado, 1985.

[6] M. R. Celis, J. Dennis, and R. Tapia. A trust region strategy for nonlinear equality constrained optimization. In P. Boggs, R. Byrd, and R. Schnabel, editors, *Numerical Optimization 1984*, pages 71–82, SIAM, Philadelphia, 1985.

[7] A. Conn, N. Gould, and Ph. Toint. *Global Convergence of a Class of Trust Region Algorithms for Optimization Problems with Simple Bounds*. Report, Dept. of Computer Sciences, University of Waterloo, Waterloo, Canada, 1988.

[8] I. I. Dikin. Iterative solution of problems of linear and quadratic programming. *Soviet Mathematics Doklady*, 8:674–675, 1967.

[9] I. I. Dikin. On the speed of an iterative process. *Upravlyaemye Sistemi*, 12:54–60, 1974.

[10] R. Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, New York, N.Y., second edition, 1987.

[11] D. M. Gay. A trust region approach to linearly constrained optimization. In D. F. Griffiths, editor, *Numerical Analysis: Proceedings Dundee 1983, Lecture Notes in Mathematics 1066*, Springer Verlag, Berlin, 1984.

[12] P. Gill and W. Murray. The computation of lagrange-multiplier estimates for constrained minimization. *Mathematical Programming*, 17:32–60, 1979.

and

$$R = \frac{1}{n}\Big\{\sum_{j=1}^{N} p_j(x_j - c)(x_j - c)^T\Big\}^{-1}. \tag{2.2}$$

Proof. Clearly the ellipsoid of least volume containing P is the ellipsoid of least volume containing the vertices of P. The volume of the ellipsoid (2.1) is proportional to $\det R^{-\frac{1}{2}}$. Therefore R is the solution of the problem

$$\text{minimize } \det S^{-1} \tag{2.3}$$

$$\text{subject to } (x_j - c)^T S(x_j - c) \le 1, j = 1, \ldots, N,$$

where $S \in \Re^{n\times n}$ is positive definite and $c \in \Re^n$.

Let $\rho_1, \ldots, \rho_N$ be nonnegative numbers satisfying $\sum_{j=1}^{N} \rho_j = 1$ and let $S \in \Re^{n\times n}$ and $c \in \Re^n$ be feasible for (2.3). Define

$$Q = \sum_{j=1}^{N} \rho_j(x_j - c)(x_j - c)^T.$$

Then

$$\begin{aligned}\{\det S^{\frac{1}{2}}QS^{\frac{1}{2}}\}^{\frac{1}{n}} &\le \frac{1}{n}\sum_{j=1}^{N} \rho_j \text{ trace } S^{\frac{1}{2}}(x_j - c)(x_j - c)^T S^{\frac{1}{2}} \\ &= \frac{1}{n}\sum_{j=1}^{N} \rho_j(x_j - c)^T S(x_j - c) \le \frac{1}{n}.\end{aligned}$$

The first inequality here is just the arithmetic-geometric means inequality applied to the eigenvalues of $S^{\frac{1}{2}}QS^{\frac{1}{2}}$. It follows that

$$\det Q \le \det(nS)^{-1}.$$

This suggests that we consider the optimization problem

$$\text{maximize } \det \sum_{j=1}^{N} \rho_j(x_j - c)(x_j - c)^T \tag{2.4}$$

$$\text{subject to } \sum_{j=1}^{N} \rho_j = 1, \rho_j \ge 0, j = 1, \ldots, N.$$

Kiefer and Wolfowitz [3] have shown that for c fixed problems (2.3) and (2.4) are duals of each other and that if $p_1, \ldots, p_N$ is a solution of (2.4),

$$R = \frac{1}{n}\Big\{\sum_{j=1}^{N} p_j(x_j - c)(x_j - c)^T\Big\}^{-1}$$

Minimum Containing Ellipsoids and Regular Polyhedra

Earl R. Barnes [1]
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205

Abstract

Let P be a full dimensional polytope in $\Re^n$ and let E denote the ellipsoid of least volume containing P. A theorem due to F. John [1] states that $\frac{1}{n}E \subset P$. $\frac{1}{n}E$ denotes the ellipsoid obtained by shrinking E by a factor of n about its center. In this note we point out some other implications of F. John's theorem for the geometry of polytopes.

1 Introduction

Minimum containing ellipsoids for polytopes arise in the analysis of Khachian's ellipsoid algorithm for linear programming [cf 2, Chapter 3], and in Karmarkar's projective scaling algorithm [2]. In Karmarkar's case the polytope is a simplex and the minimum containing ellipsoid is a ball. In both references critical use is made of John's result stated in our abstract. In this note we give some new applications of John's work to regular polytopes in $\Re^2$ and $\Re^3$. When properly interpreted these results give the first and second moments of inertia of certain regularly spaced particles in $\Re^3$ about lines and planes.

2 Minimum Containing Ellipsoids

The following theorem follows from John's optimality conditions in [1]. We have simplified the proof.

Theorem 1. Let P be an n-dimensional polytope in $\Re^n$. Let $R \in \Re^{n\times n}$ and $c \in \Re^n$ be chosen such that R is positive defnite and the set

$$E = \{x|(x-c)^T R(x-c) \leq 1\} \tag{2.1}$$

is the ellipsoid of least volume containing P. Let $x_1, \ldots, x_N$ denote the vertices of P. Then there exists nonnegative numbers $p_1, \ldots, p_N$ satisfying $p_1 + \ldots + p_N = 1$ such that

$$c = \sum_{j=1}^{N} p_j x_j$$

[1]This work was supported by the National Science Foundation grant ECS 9014823.

NOTES FROM THE U.S. CO-EDITOR

During the past decade, the Committee on Algorithms and the COAL Newsletter played a significant role in the advancement of computational mathematical programming by sponsoring conferences, rapidly disseminating research results, and establishing standards for computational studies. In this same period, many new journals were created that publish theoretical and empirical results on the interplay between optimization and computing. Consequently, the need for the COAL Newsletter as a vehicle for transmitting noteworthy preliminary research findings to a receptive audience has been somewhat mitigated. Indeed, the last technical articles appeared in the December 1987 issue.

This special theme issue on interior point methods is designed to expand the scope of the Newsletter into a forum for the rapid dissemination of ideas and philosophies on the present and future role of computational sciences in the next decade. The topic of this theme issue recognizes the significant impact on computing that interior methods have enjoyed. We particularly encourage the submission, for future issues, of thought provoking articles that report on ideas which have not been fully developed for formal communication, but which the authors are willing to share in order to stimulate research and discussion. We are also interested in receiving articles on the teaching of optimization algorithms and on the impact of new computer architecture and languages on mathematical programming.

It is fair to say that readers of the Newsletter are primarily interested in important theoretical and practical issues that push the boundaries of computing in the directions of faster solution times and larger problem sizes. The subject of this theme issue has done more to stimulate research activity and draw attention to our field than, perhaps, any other development in the past twenty years. The breadth and depth of our current understanding of interior point methods are well represented in the invited articles by some of the major contributors in the area. Future theme issues are planned whose subjects will be chosen to reflect reader interest. Your suggestions and contributions are always welcome.

Faiz A. Al-Khayyal

25.8.91

Dear MPS-member,

The COAL Newsletter has during the last 6 years been published with larger and larger intervals mainly due to the steadily decreasing number of contributions from the readership.

A first step towards a reenvigoration of the newsletter is the outstanding special issue on interior point methods consisting of 8 invited contributions you just received.

At the MPS Symposium held in Amsterdam 5. - 9.8. COAL decided to propose to the MPS council the continuation of this approach of theme-oriented newsletters as a means of rapid dissemination of results on algorithms and computation in the field of mathematical programming. Publishing a small number of short invited and contributed papers on an important subject in each issue ensures that issues are published regularly (and not cancelled due to lack of material), and we believe this will have a positive effect on the number of submitted contributions.

In connection with the discussion it was decided to change the name of the Newsletter to the COAL Bulletin to reflect the contents: technical rather than news of different kinds.

Consequently, we aim at publishing the COAL Bulletin two times a year in March and September with deadlines for contributions two months before, i.e. January 1. resp. July 1. If our plan is approved by the MPS Council the next issue of the Bulletin will be published in March 1992 with a deadline for contributions of January 1, 1992.

Papers should be submited to one of the editors in one copy (or by e-mail as a LaTeX or Postscript file), should not exceed 5 pages and should be suitable for photographic reproduction in reduced size. The decision of acceptance is made by the editors thus eliminating the often timeconsuming refereeing process of regular journals.

Faiz A. Al-Khayyal — Jens Clausen

Chairman,
US Editor — European Editor

COAL OBJECTIVES

The Committee on Algorithms is involved in computational developments in mathematical programming. There are three major goals: (1) ensuring a suitable basis for comparing algorithms, (2) action as a focal point for computer programs that are available for general calculations and for test problems, and (3) encouraging those who distribute programs to meet certain standards of portability, testing, ease of use and documentation.

NEWSLETTER OBJECTIVES

The Newsletter's primary objective is to provide a vehicle for the rapid dissemination of new results in computational mathematical programming. To date, our profession has not developed a clear understanding of the issues of how computational tests should be carried out, how the results of these tests should be presented in the literature, or how mathematical programming algorithms should be properly evaluated and compared. These issues will be addressed in the newsletter.

Mathematical Programming Society Committee on Algorithms Newsletter

NO. 19
AUGUST 1991

JENS CLAUSEN
FAIZ A. AL-KHAYYAL
EDITORS

Special Issue on Interior Point Methods

Contents