Chapter 1

RECENT DEVELOPMENTS IN INTERIOR-POINT METHODS

Stephen J. Wright

Abstract The modern era of interior-point methods dates to 1984, when Karmarkar proposed his algorithm for linear programming. In the years since then, algorithms and software for linear programming have become quite sophisticated, while extensions to more general classes of problems, such as convex quadratic programming, semidefinite programming, and nonconvex and nonlinear problems, have reached varying levels of maturity. Interior-point methodology has been used as part of the solution strategy in many other optimization contexts as well, including analytic center methods and column-generation algorithms for large linear programs. We review some core developments in the area and discuss their impact on these other problem areas.

Keywords: optimization, interior-point methods

1 INTRODUCTION

Interior-point methods have been a topic of intense scrutiny by the optimization community during the past 15 years. Although methods of this type had been proposed in the 1950s, and investigated quite extensively during the 1960s [Fiacco and McCormick, 1968], it was the announcement of an algorithm with intriguing complexity results and good practical performance by Karmarkar [Karmarkar, 1984] that ushered in the modern era. This work placed interior-point methods at the top of the agenda for a large and diverse body of researchers and led to a series of remarkable advances in various areas of convex optimization.

Today, interior-point methods for linear programming have become quite mature both in theory and in practice, and several high-quality codes are available. For the rival algorithm, the simplex method, the sudden appearance of credible competition spurred significant improve-

ments in the software, resulting in a quantum advance in the state of the art in computational linear programming since 1988.

The theory of interior-point methods in other areas of convex programming and monotone complementarity also appears to have reached a fairly advanced stage. The computational picture is less clear than for general linear programming, however. In some areas, such as semidefinite programming, there is no apparent alternative algorithm whose practical efficiency is comparable to the interior-point approach, while in others, such as quadratic programming, active-set methods (which descend from the simplex method for linear programming) provide strong competition.

Investigation of the use of interior-point methods in various areas of nonconvex optimization, including discrete optimization, is in a much less advanced stage. The eventual prospects are still unclear, though early results in some areas (for example, nonlinear programming) show distinct promise. A thread common to many approaches is the use of interior-point methods to find inexact solutions of convex subproblems that arise during the course of the larger algorithm.

We start in Section 2 by outlining the state of the art of interior-point methods in linear programming, discussing the pedigree of the most important algorithms, computational issues, and customization of the approach to structured problems. In Section 3, we discuss the straightforward extensions to quadratic programming and linear complementarity, and compare the resulting algorithms with active-set methods. The extension to semidefinite programming is discussed in Section 4, along with the theoretical work on self-concordant functionals and self-scaled cones that forms the underpinning of some of this work. Finally, we present some conclusions in Section 5.

A great deal of literature is available to the reader interested in delving further into this area. A number of recent books [Wright, 1997], [Ye, 1997], [Roos et al., 1997] give overviews of the area, from first principles to new results and practical considerations. Theoretical background on self-concordant functionals and related developments is described in [Nesterov and Nemirovskii, 1994] and [Renegar, 1999]. Technical reports from the past five years can be obtained from the Interior-Point Methods Online Web site at `http://www.mcs.anl.gov/otc/InteriorPoint`.

For lack of space, we have omitted discussion of many interesting areas in which interior-point approaches are making an impact. Convex programming problems of the form

$$\min_x \ f(x) \ \text{ s.t. } \ g_i(x) \leq 0, \ i = 1, 2, \ldots, m,$$

where f and g_i, $i = 1, 2, \ldots, m$, are convex functions, can be solved by extensions of the primal-dual approach of Section 3; see, for example, [Ralph and Wright, 1996]. Interestingly, it is possible to prove superlinear convergence of the resulting algorithms without assuming linear independence of the active constraints at the solution. This observation prompted recent work on improving the convergence properties of other algorithms, notably sequential quadratic programming. A number of researchers have used interior-point methods in algorithms for combinatorial and integer programming problems. (In some cases, the interior-point method is used to find an inexact solution of related problems in which the integrality constraints are relaxed.) Recent computational results are presented in [Mitchell, 1997], and a comprehensive survey appears in [Mitchell et al., 1998]. In decomposition methods for large linear and convex problems, such as Dantzig-Wolfe/column generation and Benders' decomposition, interior-point methods have been used to find inexact solutions of the large master problems, or to approximately solve analytic center subproblems to generate test points. Approaches such as these are described in [Gondzio and Sarkissian, 1996], [Gondzio and Kouwenberg, 1999], and the survey paper [Goffin and Vial, 1999]. Additionally, application of interior-point methodology to nonconvex nonlinear programming has occupied many researchers for some time now. The methods that have been proposed to date contain many ingredients, including primal-dual steps, barrier and merit functions, and scaled trust regions. Recent reports in this area include [Byrd et al., 1997], [Conn et al., 1999], [Gay et al., 1997], and [Forsgren and Gill, 1998].

2 LINEAR PROGRAMMING

We consider first the linear programming problem, which we state in standard form:

$$\min_x \ c^T x \ \text{ s.t. } Ax = b, \ x \geq 0, \tag{1.1}$$

where $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m \times n}$. We assume that this problem has a strict interior, that is, the set

$$\mathcal{F}^\circ \stackrel{\text{def}}{=} \{x \mid Ax = b, \, x > 0\}$$

is nonempty, and that the objective function is bounded below on the set of feasible points. Under these assumptions, (1.1) has a (not necessarily unique) solution.

By using a logarithmic barrier function to account for the bounds $x \geq 0$, we obtain the parametrized optimization problem

$$\min_x \; f(x;\mu) \stackrel{\text{def}}{=} \frac{1}{\mu} c^T x - \sum_{i=1}^{n} \log x_i, \;\; \text{s.t. } Ax = b, \tag{1.2}$$

where log denotes the natural logarithm, and $\mu > 0$ denotes the barrier parameter. Because the logarithmic function requires its arguments to be positive, the solution $x(\mu)$ of (1.2) must belong to $\mathcal{F}^\circ$. It is well known (see, for example, [Wright, 1992, Theorem 5]) that for any sequence $\{\mu_k\}$ with $\mu_k \downarrow 0$, all limit points of $\{x(\mu_k)\}$ are solutions of (1.1).

The traditional SUMT approach [Fiacco and McCormick, 1968] accounts for equality constraints by including a quadratic penalty term in the objective. When the constraints are linear, as in (1.1), it is simpler and more appropriate to handle them explicitly. By doing so, we devise a *primal barrier algorithm* in which a projected Newton method is used to find an approximate solution of (1.2) for a certain value of μ, and then μ is decreased. The projected Newton step Δx from a point x satisfies the following system:

$$\begin{bmatrix} -\mu X^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \lambda^+ \end{bmatrix} = - \begin{bmatrix} c + \mu X^{-1} e \\ Ax - b \end{bmatrix}, \tag{1.3}$$

where $X = \text{diag}(x_1, x_2, \ldots, x_n)$ and $e = (1, 1, \ldots, 1)^T$. Note that

$$\nabla^2_{xx} f(x;\mu) = -X^{-2}, \quad \nabla_x f(x;\mu) = (1/\mu)c + X^{-1}e,$$

so that the equations (1.3) are the same as those that arise from a sequential quadratic programming algorithm applied to (1.2), modulo the scaling by μ in the first line of (1.3). A line search can be performed along Δx to find a new iterate $x + \alpha \Delta x$, where $\alpha > 0$ is the step length.

The prototype primal barrier algorithm can be specified as follows:

primal barrier algorithm
Given $x^0 \in \mathcal{F}^\circ$ and $\mu_0 > 0$;
Set $k \leftarrow 0$;
repeat
    Obtain x^{k+1} by performing one or more Newton steps (1.3),
        starting at $x = x^k$, and fixing $\mu = \mu_k$;
    Choose $\mu_{k+1} \in (0, \mu_k)$; $k \leftarrow k + 1$;
until some termination test is satisfied.

A *short-step* version of this algorithm takes a single Newton step at each iteration, with step length $\alpha = 1$, and sets

$$\mu_{k+1} = \mu_k / \left(1 + \frac{1}{8\sqrt{n}}\right). \tag{1.4}$$

It is known (see, for instance, [Renegar, 1999, Section 2.4]) that if the feasible region of (1.1) is bounded, and x^0 is sufficiently close to $x(\mu_0)$ in a certain sense, then we obtain a point x^k whose objective value $c^T x^k$ is within ϵ of the optimal value after

$$O\left(\sqrt{n}\log\frac{n\mu_0}{\epsilon}\right) \text{ iterations}, \tag{1.5}$$

where the constant factor disguised by the $O(\cdot)$ depends on the properties of (1.1) but is independent of n and ϵ.

The rate of decrease of μ in short-step methods is too slow to allow good practical behavior, so *long-step* variants were proposed that decreased μ more rapidly, while possibly taking more than one Newton step for each μ_k and also using a line search. Although long-step algorithms have better practical behavior, the complexity estimates associated with them typically are no better than the estimate (1.5) for the short-step approach; see [Renegar, 1999, Section 2.4], [Gonzaga, 1991]. In fact, a recurring theme of worst-case complexity estimates for linear programming algorithms is that no useful relationship exists between the estimate and the practical behavior of the algorithm.

Better practical algorithms are obtained from the primal-dual framework. These methods recognize the importance of the path of solutions $x(\mu)$ to (1.2) in the design of algorithms, but differ from the approach above in that they treat the dual variables explicitly in the problem, rather than as adjuncts to the calculation of the primal iterates.

The dual problem for (1.1) is

$$\max_{(\lambda,s)} b^T\lambda \text{ s.t. } A^T\lambda + s = c,\ s \geq 0, \tag{1.6}$$

where $s \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}^m$, and the optimality conditions for x^* to be a solution of (1.1) and (λ^*, s^*) to be a solution of (1.6) are that $(x, \lambda, s) = (x^*, \lambda^*, s^*)$ satisfies

$$\begin{aligned} Ax &= b, && \text{(1.7a)}\\ A^T\lambda + s &= c, && \text{(1.7b)}\\ XSe &= 0, && \text{(1.7c)}\\ (x, s) &\geq 0, && \text{(1.7d)} \end{aligned}$$

where $X = \text{diag}(x_1, x_2, \ldots, x_n)$ and $S = \text{diag}(s_1, s_2, \ldots, s_n)$. Primal-dual methods solve (1.1) and (1.6) simultaneously by generating a sequence of iterates (x^k, λ^k, s^k) that in the limit satisfies the conditions (1.7). As mentioned above, the *central path* defined by the following

perturbed variant of (1.7) plays an imporant role in algorithm design:

$$\begin{aligned} Ax &= b, && \text{(1.8a)} \\ A^T\lambda + s &= c, && \text{(1.8b)} \\ XSe &= \mu e, && \text{(1.8c)} \\ (x, s) &> 0, && \text{(1.8d)} \end{aligned}$$

where $\mu > 0$ parametrizes the path. Note that these conditions are simply the optimality conditions for the problem (1.2): If $(x(\mu), \lambda(\mu), s(\mu))$ satisfies (1.8), then $x(\mu)$ is a solution of (1.2). We have from (1.8c) that a key feature of the central path is that

$$x_i s_i = \mu, \quad \text{for all } i = 1, 2, \ldots, n, \tag{1.9}$$

that is, the pairwide products $x_i s_i$ are identical for all i.

In primal-dual algorithms, steps are generated by applying a perturbed Newton method to the three equalities in (1.8), which form a nonlinear system in which the number of equations equals the number of unknowns. We constrain all iterates (x^k, λ^k, s^k) to have $(x^k, s^k) > 0$, so that the matrices X and S remain positive diagonal throughout, ensuring that the perturbed Newton steps are well defined. Supposing that we are at a point (x, λ, s) with $(x, s) > 0$ and the feasibility conditions $Ax = b$ and $A^T\lambda + s = c$ are satisfied, the primal-dual step $(\Delta x, \Delta\lambda, \Delta s)$ is obtained from following system:

$$\begin{bmatrix} 0 & A & 0 \\ A^T & 0 & I \\ 0 & S & X \end{bmatrix} \begin{bmatrix} \Delta\lambda \\ \Delta x \\ \Delta s \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ XSe - \sigma\mu e + r \end{bmatrix}, \tag{1.10}$$

where $\mu = x^T s/n$, $\sigma \in [0, 1]$, and r is a perturbation term, possibly chosen to incorporate higher-order information about the system (1.8), or additional terms to improve proximity to the central path. Using the general step (1.10), we can state the basic framework for primal-dual methods as follows:

primal-dual algorithm
Given (x^0, λ^0, s^0) with $(x^0, s^0) > 0$;
Set $k \leftarrow 0$ and $\mu_0 = (x^0)^T s^0/n$;
repeat
    Choose σ_k and r^k;
    Solve (1.10) with $\mu = \mu_k$, $\sigma = \sigma_k$ and $r = r^k$
        to obtain $(\Delta x^k, \Delta\lambda^k, \Delta s^k)$;
    Choose step length $\alpha_k \in (0, 1]$ and set

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) \leftarrow (x^k, \lambda^k, s^k) + \alpha_k(\Delta x^k, \Delta\lambda^k, \Delta s^k);$$

Set $\mu_{k+1} \leftarrow (x^{k+1})^T s^{k+1}/n$; $k \leftarrow k+1$;

until some termination test is satisfied.

The various algorithms that use this framework differ in the way that they choose the starting point, the centering parameter σ_k, the perturbation vector r^k, and the step α_k. The simplest algorithm—a short-step path-following method similar to the primal algorithm described above—sets

$$r^k = 0, \quad \sigma_k \equiv 1 - \frac{0.4}{\sqrt{n}}, \quad \alpha_k \equiv 1,$$

and, for suitable choice of starting point, achieves convergence to a feasible point (x, λ, s) with $x^T s/n \leq \epsilon$ for a given ϵ in

$$O\left(\sqrt{n}\log\frac{\mu_0}{\epsilon}\right) \quad \text{iterations.} \tag{1.11}$$

Note the similarity of both the algorithm and its complexity estimate to the corresponding primal algorithm. As in that case, algorithms with better practical performance but not necessarily better complexity estimates can be obtained through more aggressive, adaptive choices of the centering parameter (that is, σ_k closer to zero). They use a line search to maintain proximity to the central path. The proximity requirement dictates, implicitly or explicitly, that while the condition (1.9) may be violated, the pairwise products must not be too different from each other. Many such algorithms, including path-following, potential-reduction, and predictor-corrector algorithms, are discussed in [Wright, 1997].

Most interior-point software for linear programming is based on Mehrotra's predictor-corrector algorithm [Mehrotra, 1992], often with the higher-order enhancements described in [Gondzio, 1996]. This approach uses an adaptive choice of σ_k, selected by first solving for the pure Newton step (i.e., setting $r = 0$ and $\sigma = 0$ in (1.10)). If this step makes good progress in reducing μ, we choose σ_k small so that the step actually taken is quite close to this pure Newton step. Otherwise, we enforce more centering and calculate a conservative direction by setting σ_k closer to 1. The perturbation vector r^k is chosen to improve the similarity between the system (1.10) and the original system (1.8) that it approximates. Gondzio's technique further enhances r^k by performing further solves of the system (1.10) with a variety of right-hand sides, where each solve reuses the factorization of the matrix and is therefore not too expensive to perform.

To turn this basic algorithmic approach into a useful piece of software, we must address many issues. These include problem formulation, presolving to reduce the problem size, choice of the step length, linear algebra techniques for solving (1.10), and user interfaces and input formats.

Possibly the most interesting issues are associated with the linear algebra. Most codes deal with a partially eliminated form of (1.10), either eliminating Δs to obtain

$$\begin{bmatrix} 0 & A \\ A^T & -X^{-1}S \end{bmatrix} \begin{bmatrix} \Delta\lambda \\ \Delta x \end{bmatrix} = - \begin{bmatrix} 0 \\ -X^{-1}(XSe - \sigma\mu e + r) \end{bmatrix}, \quad (1.12)$$

or eliminating both Δs and Δx to obtain a system of the form

$$A(S^{-1}X)A^T\Delta\lambda = t, \quad (1.13)$$

to which a sparse Cholesky algorithm is applied. A modified version of the latter form is used when dense columns are present in A. These columns may be treated as a low-rank update and handled via the Sherman-Morrison-Woodbury formula or, equivalently, via a Schur complement strategy applied to a system intermediate between (1.12) and (1.13). In many problems, the matrix in (1.13) becomes increasingly ill-conditioned as the iterates progress, eventually causing the Cholesky process to break down as negative pivot elements are encountered. A number of simple (and in some cases counterintuitive) patches have been proposed for overcoming this difficulty while still producing useful approximate solutions of (1.13) efficiently; see, for example, [Andersen, 1996] and [Wright, 1999].

Despite many attempts, iterative solvers have not shown much promise as means to solve (1.13), at least for general linear programs. A possible reason is that, besides its poor conditioning, the matrix lacks the regular spectral properties of matrices obtained from discretizations of continuous operators. Some codes do, however, use preconditioned conjugate gradient as an alternative to iterative refinement for improving the accuracy, when the direct approach for solving (1.13) fails to produce a solution of sufficient accuracy. The preconditioner used in this case is simply the computed factorization of the matrix $A(S^{-1}X)A^T$.

A number of interior-point linear programming codes are now available, both commercially and free of charge. Information can be obtained from the World-Wide Web via the URL mentioned earlier. It is difficult to make blanket statements about the relative efficiency of interior-point and simplex methods for linear programming, as improvements to the implementations of both techniques continue to be made. Interior-point

methods tend to be faster on large problems and can better exploit multiprocessor platforms, because the expensive operations such as Cholesky factorization of (1.13) can be parallelized to some extent. They are not able to exploit "warm start" information—a good prior estimate of the solution, for instance—to the same extent as simplex methods. For this reason, they are not well suited for use in contexts such as branch-and-bound or branch-and-cut algorithms for integer programming, which solve many closely related linear programs.

Several researchers have devised special interior-point algorithms for special cases of (1.1) that exploit the special properties of these cases in solving the linear systems at each iteration. For network flow problems, Mehrotra and Wang consider preconditioned conjugate-gradient methods for solving (1.13), in which the preconditioner is built from a spanning tree for the underlying network [Mehrotra and Wang, 1995]. For multicommodity flow problems, Castro describes an algorithm for solving a version of (1.13) in which the block-diagonal part of the matrix is used to eliminate many of the variables, and a preconditioned conjugate-gradient method is applied to the remaining Schur complement [Castro, 1998]. Techniques for stochastic programming (two-stage linear problems with recourse) are described in [Birge and Qi, 1988] and [Birge and Loueaux, 1997, Section 5.6]

3 SIMPLE EXTENSIONS OF THE PRIMAL-DUAL APPROACH

The primal-dual algorithms of the preceding section are readily extended to convex quadratic programming (QP) and monotone linear complementarity (LCP), both classes being generalizations of linear programming. Indeed, many of the convergence and complexity properties of primal-dual algorithms were first elucidated in the literature with regard to monotone LCP.

We state the convex QP as

$$\min_x \; c^T x + \tfrac{1}{2} x^T Q x \;\; \text{s.t. } Ax = b,\; x \geq 0, \tag{1.14}$$

where Q is a positive semidefinite matrix. The monotone LCP is defined by square matrices M and N and a vector q, where M and N satisfy a monotonicity property: all vectors y and z that satisfy $My + Nz = 0$ have $y^T z \geq 0$. This problem requires us to identify vectors y and z such that

$$My + Nz = q, \;\; (y, z) \geq 0, \;\; y^T z = 0. \tag{1.15}$$

With some transformations, we can express the optimality conditions (1.7) for linear programming, and also the optimality conditions for (1.14), as a monotone LCP. Other problems fit under the LCP umbrella as well, including bimatrix games and equilibrium problems. The central path for this problem is defined by the following system, parametrized as in (1.8) by the positive scalar μ:

$$\begin{aligned} My + Nz &= q, && \text{(1.16a)} \\ YZe &= \mu e, && \text{(1.16b)} \\ (y, z) &> 0, && \text{(1.16c)} \end{aligned}$$

and a search direction from a point (y, z) satisfying (1.16a) and (1.16c) is obtained by solving a system of the form

$$\begin{bmatrix} M & N \\ Z & Y \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta z \end{bmatrix} = - \begin{bmatrix} 0 \\ YZe - \sigma\mu e + r \end{bmatrix}, \qquad (1.17)$$

where $\mu = y^T z/n$, $\sigma \in [0, 1]$, and, as before, r is a perturbation term. The corresponding search direction system for the quadratic program (1.14) is identical to (1.10) except that the $(2, 2)$ block in the coefficient matrix is replaced by Q. The primal-dual algorithmic framework and the many variations within this framework are identical to the case of linear programming with the minor difference that the step length should be the same for all variables. (In linear programming, different step lengths can be, and often are, taken for the primal variable x and the dual variables (λ, s).)

Complexity results are also similar to those obtained for the corresponding linear programming algorithm. For an appropriately chosen starting point (y^0, z^0) with $\mu_0 = (y^0)^T z^0/n$, we obtain convergence to a point with $mu \leq \epsilon$ in

$$O\left(n^\tau \log \frac{\mu_0}{\epsilon}\right) \quad \text{iterations},$$

where $\tau = 1/2$, 1, or 2, depending on the algorithm. Fast local convergence results typically require an additional strict complementarity assumption that is not necessary in the case of linear programming (see [Monteiro and Wright, 1994]), although some authors have proposed superlinear algorithms that do not require this assumption. Algorithms of the latter type require accurate identification of the set of degenerate indices before the fast convergence becomes effective. This property makes them of limited interest, since by the time the degenerate set has been identified, the problem is essentially solved.

The LCP algorithms can, in fact, be extended to a wider class of problems involving so-called sufficient matrices. Instead of requiring M and N to satisfy the monotonicity property defined above, we require that there exist a nonnegative constant κ such that

$$y^T z \geq -4\kappa \sum_{i \mid y_i z_i > 0} y_i z_i, \text{ for all } y, z \text{ with } My + Nz = 0.$$

The complexity estimate for interior-point methods applied to such problems depends on the parameter κ; that is, the complexity is not polynomial on the whole class of sufficient matrices.

Primal-dual methods have been applied to many practical applications of (1.14) and (1.15). For example, an application to Markowitz's formulation of the portfolio optimization problem is presented in [Takehara, 1993]; applications to optimal control and model predictive control are described in [Wright, 1993] and [Rao et al., 1998]; an application to ℓ_1 regression is described in [Portnoy and Koenker, 1997].

The interior-point approach has a number of advantages over the active-set approach from a computational point of view. It is difficult for an active-set algorithm to exploit any structure inherent in both Q and A, without redesigning most of the complex operations that make up this algorithm (adding a constraint to the active set, deleting a constraint, evaluating Lagrange multiplier estimates, calculating the search direction, and so on). In the interior-point approach, on the other hand, the only complex operation is solution of the linear system (1.17)—and this operation is fairly straightforward by comparison with the operations in an active-set method. Since the structure and dimension of the linear system remain the same at all iterations, the routines for solving the linear systems can be designed to fully exploit the properties of the systems arising from each problem class. In fact, the algorithm can be implemented to high efficiency using an object-oriented approach, in which the programmer of each new problem class needs to supply only code for the factorization and solution of the systems (1.17), optimized for the structure of the new class, along with a number of simple operations such as inner-product calculations. Code that implements upper-level decisions (choice of parameter σ, vector r, steplength α) remains efficient across the gamut of applications of (1.15) and can simply be reused by all applications.

We note, however, that active-set methods may still require much less execution time than interior-point methods in many contexts, especially when "warm start" information is available and when the problem is generic enough that not much benefit is gained by exploiting its structure.

The extension of primal-dual algorithms from linear programming to convex quadratic programming is so straightforward that a number of the interior-point linear programming codes have recently been extended to handle problems in the class (1.14) as well. In their linear algebra calculations, these codes treat both Q and A as general sparse matrices, and hence are efficient across a wide range of applications. By contrast, as noted in [Gould and Toint, 1999, Section 4], implementations of active-set methods for (1.14) that are capable of handling even moderately sized problems have not been widely available.

4 SEMIDEFINITE PROGRAMMING

Here we discuss extensions of interior-point techniques to broad classes of problems that include semidefinite programming (SDP) and second-order cone programming. The SDP problem can be stated as

$$\min_X C \bullet X, \text{ s.t. } X \succeq 0, \ A_i \bullet X = b_i, \ i = 1, 2, \ldots, m, \tag{1.18}$$

where X, C, and A_i, $i = 1, 2, \ldots, m$, are $n \times n$ symmetric matrices $\mathcal{S}\mathbb{R}^{n \times n}$, $X \succeq 0$ denotes the constraint that X be positive definite, and "$\bullet$" denotes the inner product $P \bullet Q = \sum_{i,j} P_{ij} Q_{ij}$. By further restricting X, C, and A_i all to be diagonal, we recover the linear programming problem (1.1). The class (1.18) has been studied intensively during the past seven years, in part because of its importance in applications to control systems and because many combinatorial problems have powerful SDP relaxations. The second-order cone programming problem is

$$\begin{aligned} &\min_{x_1, t_1, \ldots x_N, t_N} \textstyle\sum_{i=1}^N c_i^T x_i + \beta_i t_i, \text{ s.t.} \\ &\textstyle\sum_{i=1}^N A_i x_i + d_i t_i = b, \ \|x_i\|_2 \leq t_i, \ i = 1, 2, \ldots, N, \end{aligned} \tag{1.19}$$

where each x_i is a vector of length $n_i \geq 1$, and each t_i is a scalar. Convex quadratically constrained quadratic programs can be posed in the form (1.19), along with sum-of-norms problems and many other applications (see [Lobo et al., 1998]).

The key to extending efficient interior-point algorithms to these and other convex problems was provided in [Nesterov and Nemirovskii, 1994]. The authors explored the properties of self-concordant functions and showed that algorithms with polynomial complexity could be constructed by using barrier functions of this type for the inequality constraint and then applying a projected Newton's method to the resulting linearly constrained problem.

The fundamental property of self-concordant functions is that their third derivative can be bounded by some expression involving their second derivative at each point in their domain. This property implies that

the second derivative does not fluctuate too rapidly in a relative sense, so that the function does not deviate too much from the second-order approximation on which Newton's method is based. Hence, we can expect Newton's method to perform reasonably well on such a function.

Given a finite-dimensional real vector space $\mathcal{V}$, an open, nonempty convex set $\mathcal{S} \subset \mathcal{V}$, and a closed convex set $\mathcal{T} \subset \mathcal{V}$ with nonempty interior, we have the following formal definition.

Definition 1 *The function $F : \mathcal{S} \to \mathbb{R}$ is* self-concordant *if it is convex and if the following inequality holds for all $x \in \mathcal{S}$ and all $h \in \mathcal{V}$:*

$$\left|D^3F(x)[h,h,h]\right| \leq 2\left(D^2F(x)[h,h]\right)^{3/2}, \tag{1.20}$$

where $D^kF[h_1, h_2, \ldots, h_k]$ denotes the kth differential of F along the directions $h_1, h_2, \ldots, h_k$.

F is called strongly self-concordant *if $F(x_i) \to \infty$ for all sequences $x_i \in \mathcal{S}$ that converge to a point on the boundary of $\mathcal{S}$.*

*F is a ϑ-*self-concordant barrier *for $\mathcal{T}$ if it is a strongly self-concordant function for* $\mathrm{int}\mathcal{T}$*, and the parameter*

$$\vartheta \stackrel{\text{def}}{=} \sup_{x \in \mathrm{int}\mathcal{T}} F'(x)^T \left[F''(x)\right]^{-1} F'(x) \tag{1.21}$$

is finite.

Note that the exponent 3/2 on the right-hand side of (1.20) makes the condition independent of the scaling of h. It is shown in [Nesterov and Nemirovskii, 1994, Corollary 2.3.3] that if $\mathcal{T} \neq \mathcal{V}$, then the parameter ϑ is no smaller than 1.

It is easy to show that log-barrier function of Section 2 is an n-self-concordant barrier for the positive orthant $\mathbb{R}^n_+$ if we take

$$\mathcal{V} = \mathbf{R}^n, \quad \mathcal{S} = \mathbf{R}^n_{++}, \quad F(x) = -\sum_{i=1}^{n} \log x_i.$$

where $\mathbb{R}^n_{++}$ denotes the strictly positive orthant. Another interesting case is the second-order cone (or "ice-cream cone"), for which we have

$$\mathcal{V} = \mathbb{R}^{n+1}, \quad \mathcal{S} = \{(x,t) \mid \|x\|_2 \leq t\}, \quad F(x,t) = -\log\left(t^2 - \|x\|^2\right), \tag{1.22}$$

where $t \in \mathbf{R}$ and $x \in \mathbb{R}^n$. In this case, F is an 2-self-concordant barrier and is appropriate for the inequality constraints in (1.19). A third important case is the cone of positive semidefinite matrices, for which we

have

$$\begin{aligned} \mathcal{V} &= n \times n \text{ symmetric matrices} \\ \mathcal{S} &= n \times n \text{ symmetric positive semidefinite matrices} \\ F(X) &= -\log\det X, \end{aligned}$$

where F is an n-self-concordant barrier. This barrier function can be used to model the constraint $X \succeq 0$ in (1.18).

Self-concordant barrier functions allow us to generalize the primal barrier method of Section 2 to problems of the form

$$\min \langle c, x\rangle \text{ s.t. } Ax = b,\ x \in \mathcal{T}, \tag{1.23}$$

where $\mathcal{T}$ is a closed convex set, $\langle c, x\rangle$ denotes a linear functional on the underlying vector space $\mathcal{V}$, and A is a linear operator. Similarly to (1.2), we define the barrier subproblem to be

$$\min_x\ f(x;\mu) \stackrel{\text{def}}{=} \frac{1}{\mu}\langle c, x\rangle + F(x), \text{ s.t. } Ax = b, \tag{1.24}$$

where $F(x)$ is a self-concordant barrier and $\mu > 0$ is the barrier parameter. Note that by the Definition 1, $f(x;\mu)$ is also a strongly self-concordant function. The primal barrier algorithm for (1.23) based on (1.24) is as follows:

primal barrier algorithm
Given $x^0 \in \text{int}\mathcal{T}$ and $\mu_0 > 0$;
Set $k \leftarrow 0$;
repeat
    Obtain $x^{k+1} \in \text{int}\mathcal{T}$ by performing one or more projected Newton steps
        for $f(\cdot;\mu_k)$, starting at $x = x^k$;
    Choose $\mu_{k+1} \in (0, \mu_k)$;
until some termination test is satisfied.

Remarkably, the worst-case complexity of algorithms of this type depends on the parameter ϑ associated with F, but not on any properties of the data that defines the problem instance. For example, we can define a short-step method in which a single full Newton step is taken for each value of k, and μ is decreased according to

$$\mu_{k+1} = \mu_k / \left(1 + \frac{1}{8\sqrt{\vartheta}}\right).$$

Given a starting point with appropriate properties, we obtain an iterate x^k whose objective $\langle c, x^k\rangle$ is within ϵ of the optimum in

$$O\left(\sqrt{\vartheta}\log\frac{\vartheta\mu_0}{\epsilon}\right) \text{ iterations.}$$

Long-step variants are discussed in [Nesterov and Nemirovskii, 1994]. The practical behavior of the methods does, of course, depend strongly on the properties of the particular problem instance.

The primal-dual algorithms of Section 2 can also be extended to more general problems by means of the theory of self-scaled cones developed by Nesterov and Todd (see [Nesterov and Todd, 1997, Nesterov and Todd, 1998]). The basic problem considered is the conic programming problem

$$\min \langle c, x \rangle \text{ s.t. } Ax = b,\ x \in K, \tag{1.25}$$

where $K \subset \mathbf{R}^n$ is a closed convex cone, that is, a closed convex set for which $x \in K \Rightarrow tx \in K$ for all nonnegative scalars t, and A denotes a linear operator from $\mathbf{R}^n$ to $\mathbb{R}^m$. The dual cone for K is denoted by K^* and defined as

$$K^* \stackrel{\text{def}}{=} \{s \mid \langle s, x \rangle \geq 0 \text{ for all } x \in K\},$$

and we can write the dual instance of (1.25) as

$$\max \langle b, \lambda \rangle \text{ s.t. } A^*\lambda + s = c,\ s \in K^*, \tag{1.26}$$

where A^* denotes the adjoint of A. The duality relationships between (1.25) and (1.26) are more complex than in linear programming, but if either problem has a feasible point that lies in the interior of K or K^*, respectively, the strong duality property holds. That is, if the optimal value of either (1.25) or (1.26) is finite, then both problems have finite optimal values, and these values are the same.

K is a self-scaled cone when its interior $\text{int}K$ is the domain of a self-concordant barrier function F with certain strong properties that allow us to define algorithms in which the primal and dual variables are treated in a perfectly symmetric fashion and play interchangeable roles. The full elucidation of these properties is quite complicated. It suffices to note here that the three cones mentioned above—the positive orthant $\mathbf{R}^n_+$, the second-order cone (1.22), and the cone of positive semidefinite symmetric matrices—are the most interesting self-scaled cones, and their associated barrier functions are the logarithmic functions already mentioned.

To build algorithms from the properties of self-scaled cones and their barrier functions, the Nesterov-Todd theory defines a *scaling point* for a given pair $x \in \text{int}K$, $s \in \text{int}K^*$ to be the unique point w such that $H(w)x = s$, where $H(\cdot)$ is the Hessian of the barrier function. In the case of linear programming, it is easy to verify that w is the vector in $\mathbb{R}^n$ whose elements are $\sqrt{s_i/x_i}$. The Nesterov-Todd search directions are obtained as projected steepest descent direction for the primal and dual barrier subproblems (that is, (1.24) and its dual counterpart), where a

weighted inner product involving the matrix $H(w)$ is used to define the projections onto the spaces defined by the linear constraints $Ax = b$ and $A^*\lambda + s = c$, respectively. The resulting directions satisfy the following linear system:

$$\begin{bmatrix} 0 & A & 0 \\ A^* & 0 & I \\ 0 & H(w) & I \end{bmatrix} \begin{bmatrix} \Delta\lambda \\ \Delta x \\ \Delta s \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ s + \sigma\mu\nabla F(x) \end{bmatrix}, \tag{1.27}$$

where $\mu = \langle x, s\rangle/\vartheta$. (The correspondence with (1.10) is complete if we choose the perturbation term to be $r = 0$.) By choosing the starting point appropriately, and designing schemes for choosing the parameters σ and step lengths to take along these directions, we obtain polynomial algorithms for this general setting.

Primal-dual algorithms for (1.25), where K is a self-scaled cone, are also studied by [Faybusovich, 1997], who takes the viewpoint of differential geometry and, in particular, uses a Jordan algebra framework.

In the important case of semidefinite programming (1.18), the Nesterov-Todd framework is far from the only means for devising primal-dual methods. Many algorithms proposed before and since this framework was described do not fall under its umbrella, yet have strong theoretical properties and, in some cases, much better practical behavior. To outline a few of these methods, we write the dual of (1.18) as

$$\max_{y,S} b^T\lambda \ \text{ s.t. } \ \sum_{i=1}^m \lambda_i A_i + S = C, \ S \succeq 0, \tag{1.28}$$

where $S \in \mathcal{S}\mathbb{R}^{n\times n}$ and $\lambda \in \mathbf{R}^m$. Points on the central path for (1.18), (1.28) are defined by the following parametrized system:

$$\sum_{i=1}^m \lambda_i A_i + S = C, \tag{1.29a}$$

$$A_i \bullet X = b_i, \ \ i = 1, 2, \ldots, m, \tag{1.29b}$$

$$XS = \mu I, \tag{1.29c}$$

$$X \succeq 0, \qquad S \succeq 0, \tag{1.29d}$$

where as usual μ is the positive parameter. Unlike the corresponding equations for linear programming, the system (1.29a), (1.29b), (1.29c) is not quite "square," since the variables reside in the space $\mathcal{S}\mathbb{R}^{n\times n} \times \mathbb{R}^m \times \mathcal{S}\mathbb{R}^{n\times n}$ while the range space of the equations is $\mathcal{S}\mathbb{R}^{n\times n} \times \mathbb{R}^m \times \mathbb{R}^{n\times n}$. In particular, the product of two symmetric matrices (see (1.29c)) is not necessarily symmetric. Before Newton's method can be applied to (1.29a)—the fundamental operation in primal-dual algorithms—the

domain and range have to match. The different primal-dual algorithms differ in the ways that they reconcile the domain and range of these equations.

The paper [Todd, 1999] is witness to the intensity of research in SDP interior-point methods: It describes twenty techniques for obtaining search directions for SDP. In many of these, the equation (1.29c) is replaced by one whose range lies in $\mathcal{S}\mathbf{R}^{n\times n}$. That is, it is "symmetrized" and replaced with an mapping

$$\Theta(X,S) = 0, \tag{1.30}$$

whose linearization is

$$\mathcal{E}\Delta X + \mathcal{F}\Delta S = -\Theta(X,S). \tag{1.31}$$

Primal-dual methods are then derived as perturbed Newton's methods applied to (1.29a), (1.29b), (1.30). The heuristics associated with linear programming algorithms—Mehrotra and Gondzio corrections, step length determination, and so on—translate in a fairly straightforward way to this setting. The implementations are much more complex, however, since the linear problem to be solved at each iteration has a much more complicated structure than that of (1.10). It is noted in [Haeberly et al., 1999] that the benefits of higher-order corrections in the SDP context are even more pronounced than in linear programming, since the cost of factoring the coefficient matrix relative to the cost of solving for a different right-hand side is much greater for SDP.

Examples of symmetrizations (1.30) include the Monteiro-Zhang family, in which

$$\Theta(X,S) = \frac{1}{2}\left(P(XS)P^{-1} + P^{-T}(XS)^T P^T\right) - \mu I,$$

for some nonsingular P. The Alizadeh-Haeberly-Overton direction [Alizadeh et al., 1998], which appears to be the most promising one from a practical point of view, is obtained by setting $P = I$, while the Nesterov-Todd direction is obtained from

$$P^2 = S^{1/2}(S^{1/2}XS^{1/2})^{-1/2}S^{1/2}.$$

A survey of the applications of SDP, ranging across eigenvalue optimization, structural optimization, control and systems theory, statistics, and combinatorial optimization, is given in [Vandenberghe and Boyd, 1996]. The main use of SDP in combinatorial optimization is in finding SDP relaxations (that is, problems of the form (1.18) that contain all the feasible points of the underlying combinatorial problem in their feasible

sets) that yield high quality approximate solutions to the combinatorial problem. We illustrate the technique with possibly the most famous instance to date: the technique of Goemans and Williamson, which yields an approximate solution whose value is within 13% of optimality for the MAX CUT problem [Goemans and Williamson, 1995].

In MAX CUT, we are presented with an undirected graph with N whose edges w_{ij} have nonnegative weights. The problem is choose a subset $\mathcal{S} \subset \{1, 2, \ldots, N\}$ so that the sum of weights of the edges that cross from $\mathcal{S}$ to its complement is minimized. In other words, we aim to choose $\mathcal{S}$ to minimize the objective

$$w(\mathcal{S}) \stackrel{\text{def}}{=} \sum_{i \in \mathcal{S}, j \notin \mathcal{S}} w_{ij}.$$

This problem can be restated as an integer quadratic program by introducing variables y_i, $i = 1, 2, \ldots, N$, such that $y_i = 1$ for $i \in \mathcal{S}$ and $y_i = -1$ for $i \notin \mathcal{S}$. We then have

$$\max_y \ \tfrac{1}{2} \sum_{i<j} w_{ij}(1 - y_i y_j) \ \text{ subject to } y_i \in \{-1, 1\} \text{ for all } i = 1, 2, \ldots, N. \tag{1.32}$$

This problem is NP-complete. Goemans and Williamson replace the variables $y_i \in \mathbb{R}$ by vectors $v_i \in \mathbb{R}^N$ and consider instead the problem

$$\max_{v_1, v_2, \ldots, v_N} \ \tfrac{1}{2} \sum_{i<j} w_{ij}(1 - v_i^T v_j), \ \text{ subject to } \|v_i\| = 1 \text{ for all } i = 1, 2, \ldots, N. \tag{1.33}$$

This problem is a relaxation of (1.32) because any feasible point y for (1.32) corresponds to a feasible point

$$v_i = (y_i, 0, 0, \ldots, 0)^T, \ \ i = 1, 2, \ldots, N,$$

for (1.33). The problem (1.33) can be formulated as an SDP by changing variables $v_1, v_2, \ldots, v_N$ to a matrix $Y \in \mathbb{R}^{N \times N}$, such that

$$Y = V^T V, \ \text{ where } \ V = [v_1, v_2, \ldots, v_N].$$

The constraints $\|v_i\| = 1$ can be expressed simply as $Y_{ii} = 1$, and since $Y = V^T V$, we must have Y semidefinite. The transformed version of (1.33) is then

$$\max \ \tfrac{1}{2} \sum_{i<j} w_{ij}(1 - Y_{ij}) \ \text{ subject to } Y_{ii} = 1, \, i = 1, 2, \ldots, N \text{ and } Y \succeq 0,$$

which has the form (1.18) for appropriate definitions of C and A_i, $i = 1, 2, \ldots, N$. We can recover V from Y by performing a Cholesky

factorization. The final step of recovering an approximate solution to the original problem (1.32) is performed by choosing a random vector $r \in \mathbf{R}^N$, and setting

$$y_i = \begin{cases} 1, & \text{if } r^T v_i > 0, \\ -1 & \text{if } r^T v_i \leq 0. \end{cases}$$

A fairly simple geometric argument shows that the expected value of the solution so obtained has objective value at least .87856 of the optimal solution to (1.32).

Similar relaxations have been obtained for many other combinatorial problems, showing that is possible to find good approximate solutions to many NP-complete problems by using polynomial algorithms. Such relaxations are also useful if we seek *exact* solutions of the combinatorial problem by means of a branch-and-bound or branch-and-cut strategy. Relaxations can be solved at each node of the tree (in which some of the degrees of freedom are eliminated and some additional constraints are introduced) to obtain both a bound on the optimal solution and in some cases a candidate feasible solution for the original problem. Since the relaxations to be solved at adjacent nodes of the tree are similar, it is desirable to use solution information at one node to "warm start" the SDP algorithm at a child node. Mitchell discusses an efficient strategy along these lines for the branch-and-cut strategy [Mitchell, 1999].

5 CONCLUSIONS

Interior-point methods remains an active and fruitful area of research, although the frenetic pace that has characterized the area has slowed in recent years. Linear programming codes have become mainstream and continue to undergo development, although they face continuing stiff competition from the simplex method. Semidefinite programming has proved to be an area of major impact. Applications to quadratic programming show considerable promise, because of the superior ability of the interior-point approach to exploit problem structure efficiently. The influence on nonlinear programming theory and practice has yet to be determined, even though substantial research has already been devoted to this topic. Use of the interior-point approach in decomposition methods appears promising, though no rigorous comparative studies with alternative approaches have been performed. Applications to integer programming problems have been tried by a number of researchers, but the interior-point approach is hamstrung here by competition from the simplex method with its superior warm-start capabilities.

Acknowledgments

I thank M. J. D. Powell and the other organizers of the IFIP '99 conference for arranging a most enjoyable and interesting meeting.

THis work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

References

[Alizadeh et al., 1998] Alizadeh, F., Haeberly, J.-P. A., and Overton, M. L. (1998). Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability, and numerical results. *SIAM Journal on Optimization*, 8:746–768.

[Andersen, 1996] Andersen, K. D. (1996). A modified Schur complement method for handling dense columns in interior-point methods for linear programming. *ACM Transaction on Mathematical Software*, 22(3):348–356.

[Birge and Loueaux, 1997] Birge, J. R. and Loueaux, F. (1997). *Introduction to Stochastic Programming.* Springer Series in Operations Research. Springer.

[Birge and Qi, 1988] Birge, J. R. and Qi, L. (1988). Computing block-angular karmarkar projections with applications to stochastic programming. *Management Science*, 34:1472–1479.

[Byrd et al., 1997] Byrd, R. H., Hribar, M., and Nocedal, J. (1997). An interior point algorithm for large scale nonlinear programming. OTC Technical Report 97/05, Optimization Technology Center.

[Castro, 1998] Castro, J. (1998). A specialized interior-point algorithms for multicommodity network flows. Technical report, Statistics and Operations Research, Universitat Rovira i Virgili, Tarragona, Spain.

[Conn et al., 1999] Conn, A. R., Gould, N. I. M., D., O., and Toint, P. (1999). A primal-dual trust-region algorithm for minimizing a non-convex function subject to general inequality and linear equality constraints. Technical Report RAL-TR-1999-054, Computational Sciences and Engineering Department, Atlas Center, Rutherford Appleton Laboratory.

[Faybusovich, 1997] Faybusovich, L. (1997). Linear systems in Jordan algebras and primal-dual interior-point algorithms. *Journal of Computational and Applied Mathematics*, 86:149–175.

[Fiacco and McCormick, 1968] Fiacco, A. V. and McCormick, G. P. (1968). *Nonlinear Programming: Sequential Unconstrained Minimiza-*

tion Techniques. Wiley, New York. Reprinted by SIAM Publications, 1990.

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

[Gay et al., 1997] Gay, D. M., Overton, M. L., and Wright, M. H. (1997). A primal-dual interior method for nonconvex nonlinear programming. Technical Report 97-4-08, Computing Sciences Research, Bell Laboratories, Murray Hill, NJ.

[Goemans and Williamson, 1995] Goemans, M. X. and Williamson, D. P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Association for Computing Machinery*, 42(6):1115–1145.

[Goffin and Vial, 1999] Goffin, J. and Vial, J. (1999). Convex nondifferentiable optimization: A survey based on the analytic center cutting plane method. Technical Report 99.02, Logilab, HEC, Section of Management Studies, University of Geneva.

[Gondzio, 1996] Gondzio, J. (1996). Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications*, 6:137–156.

[Gondzio and Kouwenberg, 1999] Gondzio, J. and Kouwenberg, R. (1999). High performace computing for asset liability management. Technical Report MS-99-004, Department of Mathematics and Statistics, The University of Edinburgh.

[Gondzio and Sarkissian, 1996] Gondzio, J. and Sarkissian, R. (1996). Column generation with a primal-dual method. Technical Report 96.9, Logilab, HEC, Section of Management Studies, University of Geneva. Revised October, 1997.

[Gonzaga, 1991] Gonzaga, C. (1991). Large-step path-following methods for linear programming. *SIAM Journal on Optimization*, 1:268–279.

[Gould and Toint, 1999] Gould, N. I. M. and Toint, P. L. (1999). SQP methods for large-scale nonlinear programming. Technical Report RAL-TR-1999-055, Computational Science and Engineering Department, Atlas Center, Rutherford Appleton Laboratory.

[Haeberly et al., 1999] Haeberly, J., Nayakkankuppam, M. V., and Overton, M. L. (1999). Extending Mehrotra and Gondzio higher order methods to mixed semidefinite-quadratic-linear programming. To appear in Optimization Methods and Software.

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

[Lobo et al., 1998] Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second-order cone programming. *Linear Algebra and Its Applications*, 248:193–228.

[Mehrotra, 1992] Mehrotra, S. (1992). Asymptotic convergence in a generalized predictor-corrector method. Technical Report, Dept. of Industrial Engineering and Management Science, Northwestern University, Evanston, Ill.

[Mehrotra and Wang, 1995] Mehrotra, S. and Wang, J.-S. (1995). Conjugate gradient based implementation of interior point methods for network flow problems. Technical Report 95–70, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Ill.

[Mitchell, 1997] Mitchell, J. E. (1997). Computational experience with an interior-point cutting plane algorithm. Technical report, Mathematical Sciences Department, Rensselaer Polytechnic Institute. Revised March 1999.

[Mitchell, 1999] Mitchell, J. E. (1999). Restarting after branching in the SDP approach to MAX-CUT and similar combinatorial optimization problems. Technical report, Mathematical Sciences Department, Rensselaer Polytechnic Institute, Troy, NY.

[Mitchell et al., 1998] Mitchell, J. E., Pardalos, P. M., and Resende, M. (1998). Interior-point methods for combinatorial optimization. In Du, D. and Pardalos, P. M., editors, *Handbook of Combinatorial Optimization*, volume 1. Kluwer Academic Publishers.

[Monteiro and Wright, 1994] Monteiro, R. D. C. and Wright, S. J. (1994). Local convergence of interior-point algorithms for degenerate monotone LCP. *Computational Optimization and Applications*, 3:131–155.

[Nesterov and Nemirovskii, 1994] Nesterov, Y. E. and Nemirovskii, A. S. (1994). *Interior Point Polynomial Methods in Convex Programming*. SIAM Publications, Philadelphia.

[Nesterov and Todd, 1997] Nesterov, Y. E. and Todd, M. J. (1997). Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research*, 22:1–42.

[Nesterov and Todd, 1998] Nesterov, Y. E. and Todd, M. J. (1998). Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8:324–362.

[Portnoy and Koenker, 1997] Portnoy, S. and Koenker, R. (1997). The Gaussian hare and the Laplacian tortoise: Computability of squared-error vs. absolute-error estimators. *Statistical Science*, 12:279–300.

[Ralph and Wright, 1996] Ralph, D. and Wright, S. J. (1996). Superlinear convergence of an interior-point method despite dependent constraints. Preprint ANL.MCS-P622-1196, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill.

[Rao et al., 1998] Rao, C. V., Wright, S. J., and Rawlings, J. B. (1998). Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, 99:723–757.

[Renegar, 1999] Renegar, J. (1999). A mathematical view of interior-point methods in convex optimization. Unpublished notes.

[Roos et al., 1997] Roos, C., Vial, J.-P., and Terlaky, T. (1997). *Theory and Algorithms for Linear Optimization : An Interior Point Approach*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons.

[Takehara, 1993] Takehara, H. (1993). An interior-point algorithm for large-scale portfolio optimization. *Annals of Operations Research*, 45:373–386.

[Todd, 1999] Todd, M. J. (1999). A study of search directions in primal-dual interior-point methods for semidefinite programming. Technical report, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY.

[Vandenberghe and Boyd, 1996] Vandenberghe, L. and Boyd, S. (1996). Semidefinite programming. *SIAM Review*, 38:49–95.

[Wright, 1992] Wright, M. H. (1992). Interior methods for constrained optimization. In *Acta Numerica 1992*, pages 341–407. Cambridge University Press, Cambridge.

[Wright, 1993] Wright, S. J. (1993). Interior point methods for optimal control of discrete-time systems. *Journal of Optimization Theory and Applications*, 77:161–187.

[Wright, 1997] Wright, S. J. (1997). *Primal-Dual Interior-Point Methods*. SIAM Publications, Philadelphia.

[Wright, 1999] Wright, S. J. (1999). Modified Cholesky factorizations in interior-point algorithms for linear programming. *SIAM Journal on Optimization*, 9:1159–1191.

[Ye, 1997] Ye, Y. (1997). *Interior Point Algorithms : Theory and Analysis*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons.