ABSTRACT

Title of dissertation: PRIMAL-DUAL INTERIOR-POINT ALGORITHMS FOR LINEAR PROGRAMS WITH MANY INEQUALITY CONSTRAINTS

Luke Michael Blohm Winternitz, 2010

Dissertation directed by: Professor André L. Tits
Department of Electrical
and Computer Engineering

Linear programs (LPs) are one of the most basic and important classes of constrained optimization problems, involving the optimization of linear objective functions over sets defined by linear equality and inequality constraints. LPs have applications to a broad range of problems in engineering and operations research, and often arise as subproblems for algorithms that solve more complex optimization problems.

"Unbalanced" inequality-constrained LPs with many more inequality constraints than variables are an important subclass of LPs. Under a basic nondegeneracy assumption, only a small number of the constraints can be active at the solution–it is only this active set that is critical to the problem description. On the other hand, the additional constraints make the problem harder to solve. While

modern "interior-point" algorithms have become recognized as some of the best methods for solving large-scale LPs, they may not be recommended for unbalanced problems, because their per-iteration work does not scale well with the number of constraints.

In this dissertation, we investigate "constraint-reduced" interior-point algorithms designed to efficiently solve unbalanced LPs. At each iteration, these methods construct search directions based only on a small working set of constraints, while ignoring the rest. In this way, they significantly reduce their per-iteration work and, hopefully, their overall running time.

In particular, we focus on constraint-reduction methods for the highly efficient primal-dual interior-point (PDIP) algorithms. We propose and analyze a convergent constraint-reduced variant of Mehrotra's predictor-corrector PDIP algorithm, the algorithm implemented in virtually every interior-point software package for linear (and convex-conic) programming. We prove global and local quadratic convergence of this algorithm under a very general class of constraint selection rules and under minimal assumptions. We also propose and analyze two regularized constraint-reduced PDIP algorithms (with similar convergence properties) designed to deal directly with a type of degeneracy that constraint-reduced interior-point algorithms are often subject to. Prior schemes for dealing with this degeneracy could end up negating the benefit of constraint-reduction. Finally, we investigate the performance of our algorithms by applying them to several test and application problems, and show that our algorithms often outperform alternative approaches.

PRIMAL-DUAL INTERIOR-POINT ALGORITHMS FOR LINEAR PROGRAMS WITH MANY INEQUALITY CONSTRAINTS

by

Luke Michael Blohm Winternitz

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

Advisory Committee:
Professor André L. Tits, Chair/Advisor
Professor Benjamin Kedem
Professor Steven I. Marcus
Assistant Professor Nuno Martins
Professor Dianne P. O'Leary
Associate Professor Adrian Papamarcou

Dedication

For Lei.

Acknowledgments

First and foremost I want to thank my friend and advisor André Tits. André is not only one of the most generous and kind people I know, but he is also one of the most intelligent and technically apt, and it has been a pleasure to work with him on this research program. Although, at times I have found this effort to be extremely challenging, both technically and personally, my iteraction with André has been nothing but positive. I have, saved in my inbox, hundreds of email threads where André and I were working together to tackle technical issues both small and large; many times he would email me early in the morning with ideas that came to him in his sleep! His passion for problem solving was infectious, and it was often in these efforts that I was able to let go and truely enjoy my research. I know that André and I will remain friends and I hope to continue our technical interaction in the future.

Next, I would like to thank Dr. Dianne O'Leary for serving on my committee and for her helpful feedback and discussions regarding my work and our collaborative work together throughout my Ph.D. experience. Dr. O'Leary's ideas often helped me to see things in a different light, and her technical comments were always valuable.

I would like to thank the rest of my dissertation committee: Dr. Kedem, Dr. Marcus, Dr. Martins and Dr. Papamarcou, for serving on my commitee. I truely appreciate their careful reading of my thesis and their helpful questions and suggestions, and I thank them for making my defense an enjoyable experience.

I would like to further thank Dr. Kedem and Dr. Papamarcou for their outstanding courses in the statistical theory of estimation/detection and information, fields that are still of great interest to me, and of great relevance to my work at NASA. They are both incredible teachers and I want to say that their effort is greatly appreciated.

I would also like to thank Dr. Krishnaprasad for all of the time that he gave me

during my early graduate career. I have great admiration for Dr. Krishnaprasad's depth of knowledge and passion for science, and found him to be an inspirational figure in my academic career.

I need to thank NASA and Goddard Space Flight Center, my workplace since 2001, for its outstanding academic programs that I have taken full advantage of over the past eight years. These programs have allowed me to pursue graduate work, something I wanted and needed to do, while developing a career in a field I truely love. I want to thank the adminsitrators of the Part-Time Graduate Study Program and the Study Fellowship Program, and especially my supervisors in code 596 for their support throughout my extended graduate school career. I owe NASA and Goddard a debt of gratitude for this opportunity, and I intend to pay it back with a career focused on furthering the NASA mission.

I am very fortunate to have been blessed with a loving family who have been supportive of every effort I have undertaken. During my graduate school experience, my parents and siblings were always there to provide support and encouragement when I needed it, and for that, and for everything else I want to thank them deeply. It is very hard to express in words what my family means to me, so I will just reiterate that I am truely blessed to have them.

Finally, I want to thank my best friend and wife Lei for her support over the past ten years. I know that it has been a very long, and at times difficult, process for her, and I truely appreciate the sacrifices she has made to allow me to do this. I am looking forward so much to spending more time with her (and Peanut) and our future family together.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Background

1.1 The problem of interest

The fundamental problem dealt with in the field of mathematical programming is the constrained optimization problem

$$\begin{aligned} &\min_{x \in \mathcal{X}} f(x) \\ &\text{s.t. } x \in \mathcal{X}_0, \end{aligned} \tag{1.1}$$

where s.t. stands for "subject to", and the constraint set $\mathcal{X}_0$ is some subset of the space of variables $\mathcal{X}$. In most practical applications, the abstract constraint set $\mathcal{X}_0$ can be expressed in terms of a number of equality and inequality constraints

$$\begin{aligned} &\min_{x \in \mathcal{X}} f(x) \\ &\text{s.t. } g_i(x) = 0 \quad i \in \mathcal{E}, \\ &\qquad h_j(x) \leq 0 \quad j \in \mathcal{I}. \end{aligned} \tag{1.2}$$

where $\mathcal{E}$ and $\mathcal{I}$ are the equality and inequality constraint index sets respectively. This formulation has enormous modeling power; a vast array of problems arising in all fields of science, engineering, and operations research can be put into this framework. Unfortunately, without further restriction on the objective and constraint functions (and the space of variables and constraint indices), this problem is utterly intractable. That is, there is no implementable algorithm that can solve every instance, even in an approximate sense, with a reasonable amount of theoretical or practical effort. To render it tractable we must narrow the scope by restricting the problem data.

There are a number of useful ways to restrict the problem data. First, a basic assumption is to restrict attention to problems where $\mathcal{X}$ is a finite dimensional vector space, and where $\mathcal{I}$ and $\mathcal{E}$ are finite sets. Unfortunately, even after such restriction, problem (1.2) is still intractable.[1] To achieve tractability, the restriction that is really needed is *convexity* of the objective and feasible set. The key advantage of the convexity assumption is that it guarantees that all local minima are also global minima. Convex optimization is still a very powerful modeling paradigm, and the modern theory and algorithms for convex optimization are becoming quite satisfactory. The first two chapters of [Nes03] provide further and clearer motivation for the convexity assumption.

Another type of restriction is to assume "smoothness" of the problem data. This allows the use of the derivatives and associated local approximations of the functions f, g_i, and h_j in algorithms for solving (1.2). Even within the class of smooth-convex optimization problems there is a fundamental trade-off between the restrictiveness of the class of problems that a particular algorithm addresses and the performance of that algorithm. The most efficient algorithms address fairly specific classes of problems and exploit their structure to the fullest extent pos-

[1]For example, hard combinatorial optimization problems can be recovered even after this assumption by using nonconvex constraints.

sible. Important further restricted classes of smooth-convex problems include (in decreasing nested order) semidefinite programs (SDP), second-order-cone programs (SOCP), convex quadratic programs (QP), and finally linear programs (LP), where the functions in (1.2) are all affine.

While LPs are the most basic class of constrained smooth convex optimization problems, they are also one of the most important. They have applications in a broad range of problems in engineering and science, and especially in operations research. Furthermore, LPs often arise as subproblems in algorithms that solve more complex classes of optimization problems, particularly hard combinatorial optimization problems. See, for example, [BT97, Van96, Lue84, PS82] for more motivation for the importance of LP.

In this dissertation, we focus our attention on linear programs with the further structure that in (1.2), with m the dimension of $\mathcal{X}$, $|\mathcal{I}|$ is very large relative to $m-|\mathcal{E}|$, that is, the number of inequality constraints is large relative to the dimension of the affine manifold containing the feasible set. For linear programs in dual standard form (see (1.3) below), we have $\mathcal{X} = \mathbb{R}^m$, $|\mathcal{I}| = n$, and $|\mathcal{E}| = 0$, so that in this case, we are interested in problems with $n \gg m$. Figure 1.1 gives a schematic diagram showing where this class of problems lives among all optimization problems. We will attempt to modify a class of algorithms to improve their efficiency on these "unbalanced" linear programs. Our algorithm class of interest is the primal-dual interior-point methods (PDIPMs). PDIPMs have emerged as some of the most efficient algorithms, both practically and theoretically, for solving linear programs as well as other smooth-convex, and even nonconvex optimization problems. While PDIPMs are (arguably) becoming the algorithms of choice for most large-scale linear programming applications, alternative algorithms may have the advantage when $n \gg m$ for reasons that we will cover in detail in this chapter. Our goal in this dissertation is to attempt to improve the performance of the PDIPMs on our problem

class of interest, linear programs with many inequality constraints.

In the rest of the first chapter, we give an overview of traditional methods for solving LPs with many inequality constraints. We then introduce interior-point methods (IPMs) for LP, and introduce some notions of "constraint-reduction" that attempt to help IPMs deal with many inequality constraints by allowing them to, in some sense, ignore most of the constraints most of the time. We also give a comprehensive review of past research on constraint-reduced interior-point methods. In Chapter 2 and 3 we propose and analyze three constraint-reduced PDIPMs. The first algorithm, the topic of Chapter 2, is a constraint-reduced variant of Mehrotra's predictor-corrector, the PDIPM implemented in virtually all interior-point software packages for LP, while the two algorithms proposed in Chapter 3 are PDIPMs designed to deal with a certain type of degeneracy that constraint-reduced IPMs are subject to. In Chapter 4 we develop some specific constraint selection rules and test our algorithms on a few example problems. We also compare one of our algorithms to alternative constraint-reduced IPMs. In Chapter 5 we look more closely at some real-world applications from the area of digital filter design, and apply our algorithms to LPs arising there. Chapter 6 gives a more in-depth summary of the content and contributions of the dissertation (to which we now refer the interested reader for a more detailed overview). Finally, Chapter 7 proposes and briefly investigates some possible future lines of research.

Remark 1.1.1. *Although our focus in this dissertation will be on linear programming, we believe that a good deal of what can be said regarding LPs with many constraints applies to convex (and possibly even nonconvex) optimization problems with many constraints, so throughout this introduction and later in Chapter 7, we keep in mind the more general case.*

Remark 1.1.2. *In some sections of this chapter, we go into great detail in our description of some prior approaches for problems with many constraints (particularly,*

sections 1.3.1, 1.8, and 1.9). Such sections can be lightly skimmed without loss of continuity.

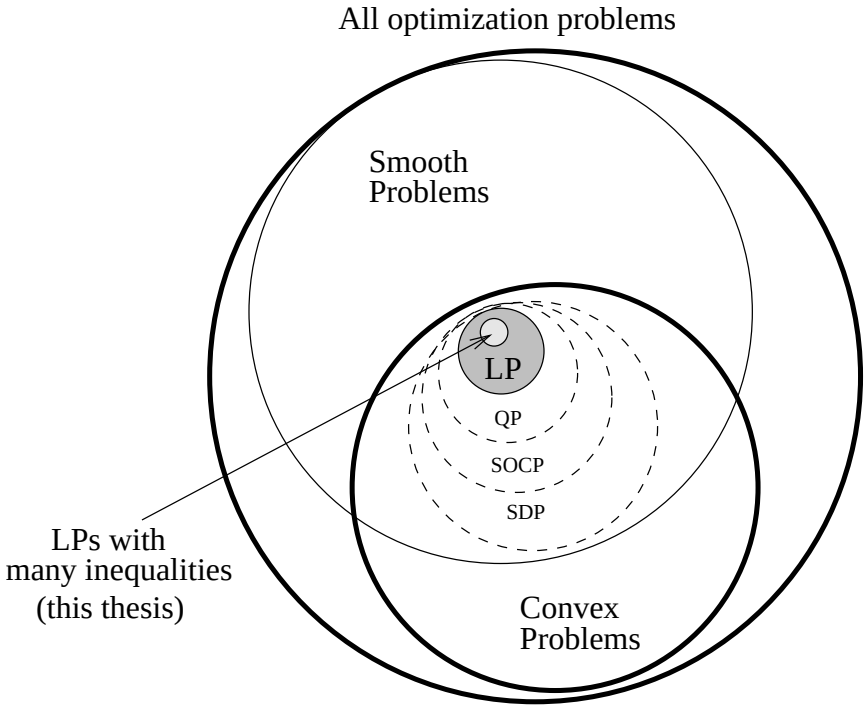

Figure 1.1: Our problem class of interest and its place among all optimization problems.

1.2 Dealing with large numbers of inequality constraints

For most algorithms for LP (as well as for more general problems), the larger $|\mathcal{I}|$ is, the more effort will be required to solve the problem. Under certain "constraint qualifications", only very few, usually no more than m (the number of variables), of these constraints can be active at the solution, and it is only these active constraints that are critical to the description of the feasible region. The remaining inactive inequalities are in some sense irrelevant to the problem, and possibly even redundant (e.g., Figure 1.2 below). Intuitively, there should be a way to ignore most of these

"irrelevant" constraints to reduce our workload. This will be the simple underlying motivation driving many of the ideas in this dissertation. In the rest of this chapter, we will address a number of traditional past approaches to problems with many constraints. The approach of the first class of methods is based on trying to guess and verify the set of active constraints as efficiently as possible.

1.3 Active set methods.

A popular approach to solving mathematical programs, the active set methods, intelligently guess which constraints are active at the solution, and then solve a smaller *equality* constrained problem with the "active" inequality constraints replaced by equality constraints, and the remaining inequality constraints deleted. The solution to this smaller problem is then checked for optimality in the original problem; if it is optimal, then the algorithm terminates, otherwise the guess at the active set is updated, and the algorithm iterates.

Active set methods deal naturally with the difficulty of many constraints. These methods generally manage to do a modest amount of work at each iteration. Another advantage is that finite termination for these methods can often be proven if the equality constrained subproblems can be solved. This is because these methods intentionally seek to exploit the combinatorial aspect of constrained optimization problems: there are only finitely many active sets to check. The main disadvantage of the active set methods is that they may require a very large number of iterations: there can be a huge number of possible active sets. The most famous active set method is the simplex method for linear programming, introduced by Dantzig in the late 1940's. Even with seemingly intelligent "pivoting" rules for updating the active set, examples were constructed, namely the famous Klee-Minty cubes [KM72], on which the simplex method would check all possible feasible active sets, i.e., vertices

of the feasible region, before finding the correct one, showing the simplex method to be theoretically inefficient, and inspiring research that ultimately lead to the rediscovery of interior-point methods and the proof of their efficiency.

1.3.1 Linear programming: revised simplex method

In this section, and through much of this dissertation, we will mainly consider the primal and dual standard forms of linear programming:

$$\begin{array}{c} \min c^{\mathrm{T}} x \\ \text{s.t. } Ax = b, \\ x \geq 0, \end{array} \qquad \text{and} \qquad \begin{array}{c} \max b^{\mathrm{T}} y \\ \text{s.t. } A^{\mathrm{T}} y \leq c, \end{array} \tag{1.3}$$

where A is an $m \times n$ matrix with $n \gg m$, that is, the dual problem has many more inequality constraints than variables. We assume $b \neq 0$.[2] The dual problem can alternatively be written in the form (with slack variable s)

$$\begin{array}{c} \max b^{\mathrm{T}} y \\ \text{s.t. } A^{\mathrm{T}} y + s = c, \\ s \geq 0. \end{array} \tag{1.4}$$

The first highly successful algorithm for LP was Dantzig's simplex method. As originally posed, the simplex method requires more computation and storage than necessary, and is not set up to deal well with large numbers of constraints. Specifically, one iteration of the original (primal) simplex method consists of updating an $(m+1) \times (n+1)$ matrix, the "simplex tableaux", which always requires $\mathcal{O}(mn)$ work and memory since the tableaux is generally dense. The *revised simplex method* improves this situation by maintaining only the critical pieces of data, namely, the

[2]This assumption is benign, since if $b = 0$ the problem at hand is readily solved: any dual feasible point y^0 (assumed available for the algorithm analyzed here) is dual optimal and $x = 0$ is primal optimal.

current $m \times m$ basis matrix (possibly in inverted or factored form), and the list of basic indices. If A is sparse and has only $\mathcal{O}(m^2)$ nonzeros, then the per-iteration memory requirement is reduced to $\mathcal{O}(m^2)$. The per-iteration computational cost is still $\mathcal{O}(mn)$ in the worst-case, although this can often be reduced to $\mathcal{O}(m^2)$ by "partial pricing" or column generation techniques (see discussion in section 1.4 below and [BT97]). We now give a more detailed account of the revised simplex method which will lead to the first methods attempting to deal with large numbers of dual inequality constraints.

An overview of the primal and dual simplex methods [BT97, Van96, Lue84, PS82]

The simplex method is based on the fundamental fact about linear programming that, if the feasible region is nonempty and contains a vertex[3] (equivalently an extreme point or basic feasible solution), then, unless the optimal set is empty, at least one of these extreme points is optimal. Therefore, when seeking optimal solutions to LPs, it is sufficient to restrict attention to the vertices. The simplex method starts at some vertex and moves to an adjacent vertex (one that shares an edge) with improved objective value, until an optimal vertex is reached.

There are two main flavors: the primal simplex algorithm and the dual simplex algorithm. Each moves around vertices of their respective feasible regions. Specifically, both algorithms maintain a basis partition, i.e., a partition (B, N) of the set $\mathbf{n} := \{1, 2, \ldots n\}$, with the "basis matrix" A_B (i.e., the matrix formed from the columns of A with indices in B) invertible. It will be convenient to assume a fixed, but arbitrary, ordering for the sets B and N during each iteration. We will use the notation $B(j)$ to denote the jth element of B, and the notation $(x_B)_j := x_{B(j)}$. The current iterate, or basic solution, is given by (x, y, s) where $x_N := 0$ and $s_B := 0$,

[3]The feasible region of a primal standard form LP, if nonempty, always contains a vertex when A is of full rank.

enforcing complementarity (the condition that $x_i s_i = 0$ for all $i \in \mathbf{n}$), and the remaining components are set to satisfy the equality portion of the primal and dual feasibility conditions

$$Ax = b,$$

and

$$A^{\mathrm{T}} y + s = c. \tag{1.5}$$

This is achieved by setting

$$x_B = A_B^{-1} b,$$

and

$$s_N = c_N - A_N^{\mathrm{T}} y,$$

with

$$y = (A_B^{-1})^{\mathrm{T}} c_B.$$

The difference between the primal and dual algorithms is that the primal algorithm maintains primal feasibility

$$x_B \geq 0,$$

moving from vertex to adjacent vertex of the primal polyhedron $\{x \mid Ax = b,\ x \geq 0\}$ in a way such that the primal objective decreases, while the dual algorithm maintains dual feasibility

$$s_N \geq 0,$$

moving from vertex to adjacent vertex of the dual polyhedron $\{y \mid A^{\mathrm{T}} y \leq c\}$ in a way such that the dual objective increases. In the primal algorithm, dual feasibility is achieved only when a primal optimal vertex is reached, while in the dual algorithm, primal feasibility is achieved only at a dual optimal vertex.

We now discuss in detail the primal algorithm, which will be seen to be more

efficient for problems with $n \gg m$. In what follows, we essentially follow the discussion in [Van96, chap. 4], but make the notation and terminology more compatible with the standard notation of interior-point methods. Starting from a primal basic feasible solution and the corresponding dual basic solution, the algorithm checks the current solution for optimality, i.e., checks if $s_N \geq 0$. The component s_N is commonly referred to as the "reduced cost" and the process of computing s_N is referred to as "pricing". If the current vertex is not optimal, then the algorithm moves along a feasible edge to a new vertex with improved cost. This can be viewed as a continuous process where a single component of x_N, say the jth, is allowed to increase from zero. Here, a "+" superscript is used to denote the new situation with $x_N^+ = te_j$, for some $t > 0$, where e_j is a vector of appropriate dimension whose jth component is one and the rest are zeros. To maintain $Ax = b$ when $x_N^+ = te_j$ increases from zero, x_B^+ must become

$$x_B^+ = A_B^{-1}b - A_B^{-1}A_N x_N^+ = x_B - tA_B^{-1}A_N e_j = x_B + t\Delta x_B,$$

with $\Delta x_B := -A_B^{-1}A_N e_j$. Plugging this into the objective, gives

$$\begin{aligned} c^{\mathrm{T}}x^+ &= c_B^{\mathrm{T}}x_B^+ + c_N^{\mathrm{T}}x_N^+ \\ &= c_B^{\mathrm{T}}(x_B + t\Delta x_B) + tc_N^{\mathrm{T}}e_j \\ &= c_B^{\mathrm{T}}(x_B - tA_B^{-1}A_N e_j) + tc_N^{\mathrm{T}}e_j \\ &= c_B^{\mathrm{T}}x_B + t(c_N - (A_B^{-1}A_N)^{\mathrm{T}}c_B)^{\mathrm{T}}e_j \\ &= c_B^{\mathrm{T}}x_B + t(c_N - A_N^{\mathrm{T}}y)^{\mathrm{T}}e_j \\ &= c_B^{\mathrm{T}}x_B + t(s_N)^{\mathrm{T}}e_j = c_B^{\mathrm{T}}x_B + t(s_N)_j. \end{aligned}$$

Thus, taking j, the column "entering the basis", to be such that $(s_N)_j < 0$, will cause the objective to improve for small enough t (barring degeneracy, see e.g.,

[BT97]). So, t is increased until one of the components of x_B becomes zero, and thus "leaves the basis". This never happens if $\Delta x_B \geq 0$, which indicates that the primal objective is not bounded from below on the primal feasible region, and hence that the dual is infeasible. Otherwise, it happens when

$$t = t^* := \min \left\{ \frac{(x_B)_k}{-(\Delta x_B)_k} \;\middle|\; k \text{ s.t. } (\Delta x_B)_k < 0 \right\}.$$

Let i denote an index that achieves the minimum. Then, in the terminology of the method, with j the column entering the basis and i the column leaving the basis, the iterate moves to the new vertex by updating

$$\begin{aligned} t^* &= \frac{(x_B)_i}{-(\Delta x_B)_i}, \\ x_B^+ &= x_B + t^* \Delta x_B, \\ x_N^+ &= t^* e_j, \end{aligned} \tag{1.6}$$

and update the basis

$$\begin{aligned} B^+ &= (B \cup \{N(j)\}) \setminus \{B(i)\}, \\ N^+ &= \mathbf{n} \setminus B^+. \end{aligned}$$

Alternatively, once the new basis is known, the update can be computed anew as

$$\begin{aligned} x_B^+ &= (A_{B^+})^{-1} b, \\ x_N^+ &= 0, \end{aligned} \tag{1.7}$$

which must agree with (1.6).

The dual iterate can be computed with the new basis via[4]

$$\begin{aligned} y^+ &= (A_{B^+}^{\mathrm{T}})^{-1} c_{B^+}, \\ s_{N^+}^+ &= c_{N^+} - A_{N^+}^{\mathrm{T}} y^+, \\ s_{B^+}^+ &= 0. \end{aligned}$$

Alternatively, the dual update can be viewed as a continuous process. As pointed out in [Van96], that view reveals a remarkable symmetry between the primal and dual algorithms. In the continuous view, the dual variables move along a direction $(\Delta y, \Delta s)$,

$$(y(\tau), s(\tau)) = (y, s) + \tau(\Delta y, \Delta s),$$

while preserving the equality constraint $s(\tau) = c - A^{\mathrm{T}} y(\tau)$. This requirement forces $\Delta s = -A^{\mathrm{T}} \Delta y$. Complementary slackness at the updated iterate only allows $(s_B)_i$ to increase from zero, so that $\Delta s_B := e_i$ can be taken as a definition. This completely determines the dual step since then it must hold that $\Delta y = -A_B^{-1} e_i$ and $\Delta s = -A^{\mathrm{T}} A_B^{-1} e_i$. Finally the step length τ^* is determined by $(s_N^+)_j = (s_N)_j + \tau^* (\Delta s_N)_j = 0$, which is required by complementary slackness of (x^+, s^+). In summary (noting that Δy is really ancillary to the process) the dual step is given by

$$\begin{aligned} \tau^* &= \frac{(s_N)_j}{-(\Delta s_N)_j}, \\ \Delta s_N &= -A_N^{\mathrm{T}} A_B^{-1} e_i, \\ s_B^+ &= \tau^* e_i, \\ s_N^+ &= s_N + \tau^* \Delta s_N. \end{aligned} \tag{1.8}$$

In summary, the (revised) primal simplex algorithm:

[4] In practice, this can and should be done efficiently with low-rank updates.

Algorithm 1: Revised Primal Simplex Algorithm

Input: LP data A, b, c; Initial iterate: x a primal basic feasible solution, with basis partition (B, N); Parameters: none;

Output: Exact primal-dual solution (x, y, s)

while *There exists* j *such that* $(s_N)_j < 0$ **do**

Choose one such j (entering column);

Compute $\Delta x_B = -(A_B^{-1} A_N) e_j$;

Set $t = \min \left\{ \frac{(x_B)_k}{-(\Delta x_B)_k} \mid (\Delta x_B)_k < 0 \right\}$,

with index i achieving the minimum (leaving column);

Compute $\Delta s_N = (A_B^{-1} A_N)^{\mathrm{T}} e_i$;

Let $\tau = \frac{(s_N)_j}{-(\Delta s_N)_j}$;

Set

$$x_N^+ = t e_j, \quad x_B^+ = x_B + t \Delta x_B$$

$$s_B^+ = \tau e_i, \quad s_N^+ = s_N + \tau \Delta s_N$$

Set $B^+ = (B \cup \{N(j)\}) \setminus \{B(i)\}$;

end

The symmetry between the primal and dual simplex methods is striking when Δy is eliminated from the dual algorithm. The idea is exactly the same: start from a (now dual) basic feasible solution, choose one active constraint from $s_B = 0$ (rather than $x_N = 0$) to release that will help to improve the objective, and then make sure to preserve the equality constraint and complementary slackness. We omit a detailed description of the dual simplex algorithm, but list it for comparison.

Algorithm 2: Dual Simplex Algorithm

Input: LP data A, b, c; Initial iterate: s corresponding to a dual basic feasible solution y, with basis partition (B, N); Parameters: none;

Output: Exact primal-dual solution (x, y, s)

while *There exists j such that $(x_B)_j < 0$* **do**

Choose one such j (leaving column);

Compute $\Delta s_N = (A_B^{-1} A_N)^{\mathrm{T}} e_j$;

Set $t = \min \left\{ \frac{(s_N)_k}{-(\Delta s_N)_k} \mid (\Delta s_N)_k < 0 \right\}$,

with index i achieving the minimum (entering column);

Compute $\Delta x_B = -(A_B^{-1} A_N) e_i$;

Let $\tau = \frac{(x_B)_j}{-(\Delta x_B)_j}$;

Set

$$s_B^+ = t e_i, \quad s_N^+ = s_N + t \Delta s_N$$

$$x_N^+ = \tau e_j, \quad x_B^+ = x_B + \tau \Delta x_B$$

Set $B^+ = (B \cup \{N(i)\}) \setminus \{B(j)\}$;

end

1.4 Column generation and cutting-plane methods for linear programs

In this section we discuss some variations on the basic revised simplex method, that are more efficient for unbalanced problems with $n \gg m$. A good general reference for this material of this section, which we draw heavily on, is [BT97].

1.4.1 Column generation

The computational cost of the major tasks required in each iteration of the revised primal simplex method are listed in Table 1.1.[5]

Operation	flops
Solve $A_B \Delta x_B = -A_N e_j$	$\mathcal{O}(m^2)$
Solve $A_B^{\mathrm{T}} \Delta y = -A_N e_i$	$\mathcal{O}(m^2)$
Compute $\Delta s_N = -A_N^{\mathrm{T}} \Delta y$	$\mathcal{O}(mn)$
Update the factorization of A_B	$\mathcal{O}(m^2)$

Table 1.1: Revised primal simplex algorithm dominant costs per iteration (assuming dense matrices).

To complete an iteration of the revised primal simplex method, one needs only the current basis and a method to choose a new column to enter the basis that will lead to an improvement in the objective. The usual way to do this is to fully compute s_N and choose a negative component according some "pivoting" rule; the most common method chooses the column with the most negative s_j. However, notice in Table 1.1, that computing the update for s_N is the only operation whose cost depends on n, and when $n \gg m$ this could easily be the most expensive task. However, we reiterate the fact that the entire s_N vector does not need to be computed in the iteration; all that is needed is a way of generating a column of A with negative reduced cost, hence the name "column generation". Thus, one may try to compute only part of s_N, a technique called "partial pricing". In some cases, it may be possible to guess, e.g., using prior information on the problem's structure, an index j for which $s_j < 0$, or alternatively, one could just compute them one by one in some order, or in random order, until a negative s_j is identified. These methods are typically very effective in early iterations and can reduce the cost of an iteration to $\mathcal{O}(m^2)$ if intelligent updating schemes are used. In later iterations, as

[5]The first two rows of the table suggest that an $m \times m$ linear system can be solved with $\mathcal{O}(m^2)$ work rather than $\mathcal{O}(m^3)$ work; this can be done if efficient low-rank factorization updating schemes are used (since the basis matrix A_B only changes in two columns at each iteration).

fewer and fewer columns have negative reduced cost, it is likely that more and more work will be needed to identify one. In the end, the minimal component of s_j must be computed to check for optimality. If this is done by computing s_N completely, then the iterations necessarily cost $\mathcal{O}(mn)$ eventually.

Note that these ideas do not work for the dual simplex algorithm. This is because the corresponding "pricing" operation is cheap in the dual algorithm, while the computation of t, which cannot be avoided, becomes expensive: it is now an $O(n)$ operation.

In some cases, it may not be necessary to explicitly compute $s_j = c_j - a_j^{\mathrm{T}} y$ for any j, if instead there is available a "column generator" subroutine that takes the current iterate as input and returns a column with negative reduced cost. One prominent example of this is in the Dantzig-Wolfe decomposition [DW60] method for problems with "multi-commodity flow" structure (i.e., the constraint matrix has a block diagonal structure plus a group of linking constraints). There, the original problem is converted to an equivalent LP with fewer equality constraints, but exponentially many variables. This would normally be a bad idea, but as it turns out, the converted problem has available an efficient column generator that involves the solution of a handful of smaller LPs. See [BT97] or [Lue84] for more detail.

A systematic class of methods that use these ideas are as follows. The revised simplex method described above can be thought of as consisting of an outer iteration and of a (trivial) inner iteration. The inner iteration consists of solving the "restricted master LP"

$$\begin{aligned} &\min c_B^{\mathrm{T}} x_B \\ &\text{s.t. } A_B x_B = b, \\ &x_B \geq 0, \end{aligned} \tag{1.9}$$

which only requires a basis inversion. The outer iteration works toward solving the

original LP (the "master problem") by, at each iteration, updating the working set of constraints (the basis B), adding a single constraint from the master problem with negative reduced cost and removing a single constraint from B. Termination occurs when no constraint in the master problem has negative reduced cost; then the optimal point for the restricted problem is optimal for the master problem. This can be generalized to allow for a larger working set of constraints W in the restricted problem. The inner iteration then consists of solving the restricted master problem

$$\begin{aligned} &\min c_W^{\mathrm{T}} x_W \\ &\text{s.t. } A_W x_W = b, \\ &x_W \geq 0, \end{aligned} \tag{1.10}$$

and the outer iteration consists of updating W by adding some constraints from the master problem in $\mathbf{n} \setminus W$ with negative reduced cost, and possibly adding and deleting others from W. If this is not possible then the master problem is solved. Taking this to the other extreme, by setting $W = \mathbf{n}$, the outer iteration becomes trivial.

These column generation methods with appropriate updating (pivot selection) rules inherit the finite termination property of the simplex method, and as mentioned above, they often perform very well in early iterations but tend to slow down, or "tail off" in later iterations.

1.4.2 Cutting-plane methods for linear programs

From the dual point of view, the column generation method can be seen as a constraint generation, or cutting-plane method. The duals of the restricted master programs are "relaxed dual problems"—polyhedral outer approximations to the master LP. From this point of view, the relaxed LP is solved in the inner iteration and check

for dual feasibility, that is check if $s \geq 0$ (no negative reduced costs). If the solution to the relaxed problem is feasible, then it is also optimal for the master problem; otherwise the constraint set is updated by adding some violated constraints and adding/deleting others and then iterating.

With these methods in hand, there is no requirement that the the restricted master or relaxed dual problems be solved by the primal simplex method. One can, in fact, use any method for solving LPs, and many have been tried. However, it turns out that since successive restricted problems are generally similar to one another, differing in only a few columns of the constraint matrix, the new problem can be efficiently "reoptimized" using the primal simplex method. Specifically, if the optimal basis from the previous restricted problem is kept in the new problem, then the previous optimal point will be basic for the updated problem. Thus, the primal simplex method can be "warm-started" from this point and can often solve the new problem in only a handful of iterations. This is one of the strengths of simplex methods that have kept them competitive in the interior-point era.

1.5 Cutting-plane methods for general (smooth or non-smooth) convex optimization

The main idea of the cutting-plane method applies more broadly than just to LP. Consider the convex feasibility problem

$$\begin{aligned} &\text{find } x \\ &\text{s.t. } x \in \mathcal{X}. \end{aligned} \tag{1.11}$$

This is quite general since $\mathcal{X}$ can, for example, represent the optimal set of (1.1). The basic cutting-plane method finds a point in $\mathcal{X}$ by creating a sequence of improving polyhedral outer approximations P^k, such that $P^{k+1} \subset P^k$, $P^{k+1} \neq P^k$, and an

associated sequence of "query" points $y^k \in P^k$. If the volume (other measures of "size" are possible) of P^k shrinks at a constant rate, and $\mathcal{X}$ has positive volume, then then the algorithm must terminate finitely. Under the assumption that P^0 is contained in a ball of radius R and $\mathcal{X}$ contains a ball of radius r, and if the ratios of volumes of successive approximations satisfy

$$\frac{\mathrm{vol}(P^{k+1})}{\mathrm{vol}(P^k)} \leq \beta < 1,$$

then the algorithm requires at most

$$\left\lceil \frac{n \log(R/r)}{-\log(\beta)} \right\rceil$$

iterations.

These algorithms use a *separation oracle*, that is, a subroutine that takes, as input, the query point $y^k \in P^k$, and either determines $y^k \in \mathcal{X}$, and hence, that the problem is solved, or declares $y^k \notin \mathcal{X}$, and returns a separating hyperplane defined by a vector u, i.e., u is such that $u^{\mathrm{T}} y^k \leq \alpha \leq u^{\mathrm{T}} x$ for all $x \in \mathcal{X}$. The half-space containing $\mathcal{X}$ is then intersected with P^k to generate P^{k+1}. If the separating hyperplane satisfies $u^{\mathrm{T}} y^k < \alpha$, then it is called a *deep cut*, otherwise it is called a *neutral cut* [BV07].

These methods are of interest in this dissertation because $\mathcal{X}$ may involve many linear and/or nonlinear inequality constraints, while P^k is generally much simpler, and the work required to update the iterate, i.e., computing y^{k+1}, is primarily determined by the complexity of P^k. (Of course, the work required by the separation oracle should also be taken into account.)

A variation of this basic method allows for a linear objective function:

$$\begin{aligned} &\min\ c^{\mathrm{T}}x \\ &\text{s.t. } x \in \mathcal{X}. \end{aligned} \tag{1.12}$$

The cutting-plane method for LP, described in section 1.4.2 as the dual view of the column generation method, fits this framework. There, the target set $\mathcal{X}$ is the optimal set of the master LP. The sets $P^{k+1} \subset P^k$ correspond to the feasible regions of the relaxed dual problems, and the query points are the optimal solutions to relaxed problems. The separation oracle here works by identifying a violated constraint from the master problem and returning this constraint as the cutting plane. This method is one instance of the Kelley cutting-plane (KCP) algorithm [Kel60]. One of its benefits, in the LP case, is that it allows efficient reoptimization using the simplex method. A more general version of KCP allows for a nonlinear objective and, in addition to the polyhedral "model" of the feasible region, a polyhedral (piecewise linear) model of the objective is built-up during the iteration. The minimizers of this model function over the P^k are taken as the query points y^k. The KCP method for general convex problems has been shown to be theoretically (very) inefficient, e.g., [Nes03], but some "stabilizations" of KCP, namely bundle and level methods, are much more efficient [Nes03]. In the context of LP, the convergence of this method has been observed to be quite slow, particularly in later iterations; this is the so called "tailing off" effect.

We now describe some specific cutting-plane methods for (1.11). Different cutting-plane methods use different methods for selecting y^k and u^k. It is intuitively clear that for good performance y^k should be as deeply in the interior of P^k as possible.[6] Some well known centers [Nes03, BVS] used to define y^k are as follows.

[6] An intuitive reason for why the KCP method tails off is just this: its query points are extreme points of the feasible region, not interior-points at all. Of course, KCP generates deep cuts, but as the iteration approaches optimality, it is generally the case that the cuts get less and less deep.

- **Center of gravity.** The center of gravity of P^k is defined as

$$y^k := \frac{\int_{P^k} x dx}{\int_{P^k} dx}.$$

The corresponding cutting-plane method—regardless of how the hyperplane is chosen!—has $\beta \leq 1 - 1/e \simeq 0.63$, which is nearly optimal. This is not trivial to prove, but it is not hard to see that $\beta \leq \frac{1}{2}$. Remarkably, this β is completely independent of the problem data, including the dimension of $\mathcal{X}$. Thus the complexity in terms of calls to the oracle is $\mathcal{O}(n \log \frac{R}{r})$, which is optimal in the sense that, up to a multiplicative factor, this bound coincides with certain lower complexity bounds for the problem class; see [Nes03] for further detail on optimal methods. Unfortunately, computing the center of gravity of a general convex set $\mathcal{X}$ is at least as hard as solving (1.11), so this choice of y^k is only of theoretical interest.

- **Chebyshev center.** Elzinga and Moore [EM75] suggested generating the cuts from the *Chebyshev center* of P^k, giving the Elzinga-Moore cutting-plane (EMCP) method. The Chebyshev center is defined to be the center of the ball of largest radius that fits inside a polyhedron. It can be found as the solution of a dual standard form LP with $m+1$ variables and n constraints, where n is the number of constraints defining P^k. Therefore, the work required to compute the Chebyshev center is essentially the same as solving the subproblem in the KCP method. Reoptimization by simplex methods for EMCP may not be as effective as in the KCP method however, because successive optima are not basic feasible solutions for the successive LP subproblems, and so the simplex method cannot be easily "warm-started".

- **Center of maximum volume inscribed ellipsoid.** Another possible center is the center of the maximum volume ellipsoid inscribed in P^k. This center can

be obtained as the solution to a convex optimization problem [BV04, KT93]. It can be shown that $\beta = 1 - 1/n$, and hence, that its efficiency in terms of calls to the oracle is $\mathcal{O}(n^2 \log \frac{R}{r})$.

- **Analytic center.** Yet another way to define the deepest point of a polyhedron, or any convex set $\{x \mid f_i(x) \leq 0,\ i \in I\}$, defined by a system of inequalities, is as the *analytic center*, that is, the minimizer of the log-barrier function

$$-\sum_{i \in I} \log(-f_i(x)). \tag{1.13}$$

  The centering subproblem here is not an LP, as was the case for the KCP and EMCP methods, and so simplex methods do not apply. However, the problem is efficiently solved by Newton's method, and, in fact, this centering problem is the fundamental subproblem solved by many polynomial-time interior-point algorithms. The analytic center cutting-plane (ACCP) idea generated a large amount of research in the 1990's (presumably because of its connection to IPMs) and it is still an active area of research. Several variants of this method when applied to convex programs have been shown to have overall polynomial complexity, e.g., [Ye92, Nes95, Nes03, BVS]. Furthermore, the performance of this method seems to be quite good in practice, see, e.g., [BVS].

- **Center of minimum volume containing ellipsoid.** This center can also be obtained as the solution to a convex optimization problem [BV04] and, presumably, a corresponding cutting-plane method can be developed. The author has not seen such an algorithm though. Of course, this center is used by Khachiyan's famous ellipsoid method [Kha79] for LP, the first polynomial time algorithm for linear programming. The ellipsoid method is not a cutting-plane method, but the basic idea is the same. Rather than P^k being polytopes, they are ellipsoids. This allows for particularly efficient computation of successive

centers. It is possible that the ellipsoid method could be used to efficiently solve problems with many constraints.

1.6 Interior-point methods

Interior-point methods (IPMs) flip the workload as compared to active set and cutting plane methods. Iteration counts are both theoretically and practically much lower than in the active set methods. For LP, for a fixed accuracy, theoretically, $\mathcal{O}(\sqrt{n})$ iterations suffice, while practically it is commonly believed that only $\mathcal{O}(\log n)$ iterations are needed. The downside to IPMs is that the work per iteration is generally much higher.

In this section, we again focus primarily on linear programming problem (1.3), repeated here for convenience:

$$\begin{array}{c} \min c^{\mathrm{T}} x \\ \text{s.t. } Ax = b, \\ x \geq 0, \end{array} \qquad \text{and} \qquad \begin{array}{c} \max b^{\mathrm{T}} y \\ \text{s.t. } A^{\mathrm{T}} y \leq c. \end{array} \tag{1.14}$$

Although there are many different interior-point-algorithms for LP, in most of them, if direct methods of linear algebra are used, the dominant task per iteration is the formation and solution of a system of "normal equations", which have the form

$$ADA^{\mathrm{T}} u = f, \tag{1.15}$$

with A the constraint matrix of the LP and D some diagonal matrix. When A is dense, the cost of forming the normal matrix is $\mathcal{O}(nm^2)$ floating-point operations, and the cost of solving the resulting system, typically by Cholesky factorization and back substitution, is $\mathcal{O}(m^3)$.

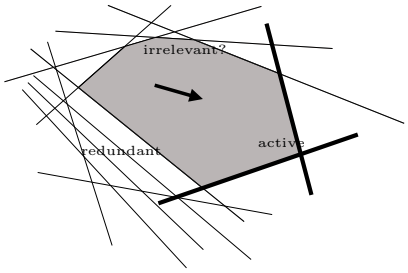

Figure 1.2: A view of the y space when $m = 2$ and $n = 12$. The arrow indicates the direction of vector b. The two active constraints are critical and define the solution, while the others are redundant or perhaps not very relevant for the formation of good search directions.

When $n \gg m$, the cost of forming the normal matrix dominates the work per-iteration. This requires computing the sum

$$ADA^{\mathrm{T}} = \sum_{i=1}^{n} d_i a_i a_i^{\mathrm{T}}, \tag{1.16}$$

where a_i is the ith column of A and d_i is the ith diagonal entry of the diagonal matrix D. Each term of the sum corresponds to a particular constraint in the dual problem (or variable in the primal). On the other hand, as argued before, we might expect most of the n constraints to be redundant, or not very relevant for the formation of a good search direction (see Figure 1.2). We would like to reduce the required work by ignoring some or most of them. For example, one possible approach relies on the fact that the coefficient d_i is often related to "nearness to activity" of the ith constraint,[7] being large for constraints close to the current iterate, and small for constraints far away from it. In (1.16), if a small set of $q < n$ "important" constraints, perhaps related to the size of the corresponding d_i, were selected and then only the corresponding partial sum computed, the dominant term in the cost-per-iteration would be reduced to $\mathcal{O}(qm^2)$ operations. This approach is precisely that taken in the paper of Tits et al. [TAW06], which we will discuss

[7] At least if we assume A has its columns normalized.

in more detail below. Similar possibilities arise in other interior-point methods: by somehow ignoring most of the constraints, one may hope that a "good" step can still be computed, at significantly reduced cost. (Such a step may even be *better*: see [DNPT06] for evidence of the potential harm caused by redundant constraints.)

1.7 Prior work on constraint-reduction in IPMs for LP

In this section we review, in some detail, prior work on constraint-reduction for interior-point methods in the context of linear programming. There are essentially three classes of interior-point algorithms:

1. Affine-scaling,
2. Potential-reduction,
3. Path-following.

It turns out that these three classes are closely related, in that almost all variants use a linear combination of two fundamental steps or search directions: the affine-scaling and centering direction [Her92]. IPMs can further be classified as primal, dual, or primal-dual, depending on the prevalence of the primal vs. dual variables in the iteration. This notion too can be somewhat blurry at times. We will see that there has been work on all three classes attempting to reduce the cost of an iteration, and in some cases the iteration complexity as well, using some notion of constraint-reduction. The first constraint-reduced IPM was proposed in an unpublished technical report [DY91], by two of the most important figures in linear programming.

1.8 Dantzig and Ye's build-up dual affine-scaling method

The idea of constraint-reduction for interior-point algorithms goes back at least as far as Dantzig and Ye [DY91], who proposed a "build-up" variant of a dual affine-scaling algorithm (DAS). We will first review the standard dual affine-scaling algorithm, a dual variant of Dikin's famous algorithm [Dik74].

1.8.1 The dual affine-scaling algorithm

Given a strictly feasible point (y, s) for the dual problem in (1.14), i.e., $s := c - A^{\mathrm{T}}y,\ s > 0$,[8] the standard dual affine-scaling step or search direction is defined as the solution to the elementary ellipsoid (sometimes called the Dikin ellipsoid) constrained optimization problem

$$\begin{aligned} &\max\ b^{\mathrm{T}}\Delta y, \\ &\text{s.t. } \|S^{-1}A^{\mathrm{T}}\Delta y\| \leq \beta, \end{aligned} \tag{1.17}$$

where the S is defined as the diagonal matrix with components $S_{ii} = s_i,\ i = 1, 2 \ldots n$. (Throughout this chapter, and this dissertation, matrices X and S will always represent diagonal matrices with vectors x and s, respectively, on the main diagonal.) The solution to this problem is evident when rewritten in terms of the inner product

$$\langle u, v \rangle_y := \langle u, Hv \rangle$$

[8] Interior-point algorithms for the dual problem always have $s > 0$. They may or may not enforce the dual feasibility condition $s = c - A^{\mathrm{T}}y$ though. If dual feasibility is not enforced, the dual iterate should be thought of as (y, s), if, instead, dual feasibility is enforced, as in the DAS algorithm, one can think of y as the "main" dual iterate and take s to be defined by $s := c - A^{\mathrm{T}}y$. This is the convention used in the discussion of the DAS algorithm. (Alternatively, if A has full rank, s can be viewed as main iterate and y defined by the feasibility equation $A^{\mathrm{T}}y + s = c$.)

corresponding to the positive-definite matrix

$$H := AS^{-2}A^{\mathrm{T}}.$$

We call this inner-product the "H-inner-product", and the norm $\|u\|_y = \langle u, u\rangle_y$ that it induces will be referred to as the "H-norm".[9] Using the H-inner-product, (1.17) becomes

$$\begin{aligned} &\max \langle b_y, \Delta y\rangle_y, \\ &\text{s.t. } \|\Delta y\|_y \leq \beta, \end{aligned} \tag{1.18}$$

where $b_y := H^{-1}b$ is the gradient of the objective in the H-inner-product. The solution is

$$\Delta y = \beta \frac{b_y}{\|b_y\|_y}. \tag{1.19}$$

It can be verified that the update $y^+ = y + \Delta y$ is strictly dual feasible when y is. When $\beta \in (0, 1)$, the constraint in (1.17) or (1.18) can be seen to define an ellipsoid inscribed in the dual feasible region: since y is strictly feasible (i.e., $s = c - A^{\mathrm{T}}y > 0$), it holds that

$$s^+ = c - A^{\mathrm{T}}y^+ = s - A^{\mathrm{T}}\Delta y = S\left(e - \beta \frac{S^{-1}A^{\mathrm{T}}b_y}{\|S^{-1}A^{\mathrm{T}}b_y\|}\right) > 0.$$

This expression makes it clear that the step

$$\Delta y = \beta \frac{b_y}{\|S^{-1}A^{\mathrm{T}}b_y\|_\infty} \tag{1.20}$$

[9] This notation is borrowed from J. Renegar's outstanding book [Ren01]. Note that this H is a function of y (through s), which is why the notation uses a y subscript. The matrix H is the Hessian at y of the logarithmic barrier $f(y) := -\sum_{i=1}^{n} \log(c - a_i^{\mathrm{T}}y)$ for the dual feasible region $\{y \mid A^{\mathrm{T}}y \leq c\}$.

is also feasible. This latter step is called the long-step affine-scaling step (as is the step length that goes almost all the way to the boundary of the feasible set along the same direction) while (1.19) is called the short-step affine-scaling step. In practice, the long-step variant is usually preferable, but, as in [DY91], the proof here is given for the short-step variant. The modification of this analysis for the long-step variant is (supposedly) straightforward.

The DAS algorithm also defines primal iterates or "estimates" (that are generally infeasible with respect to the non-negativity constraints) as the solution to the following least squares problem involving the complementarity condition[10]

$$\begin{aligned} &\min \|Sx\|^2 \\ &\text{s.t. } Ax = b, \end{aligned} \tag{1.21}$$

with solution

$$x = S^{-2}A^{\mathrm{T}}b_y. \tag{1.22}$$

Now that the main pieces are in place, we state a variant of the classic DAS algorithm.

[10] A result that has been independently proven by many authors shows that given any sequence of $s^k > 0$, the corresponding x^k sequence defined by (1.21) remains bounded. (This result can be used to remove the primal nondegeneracy assumption used in the proof of convergence of the DAS outlined below, which comes from [DY91, BT97].) The first proof was by [Dik74], an English version of which can be found in [VL88]; see also [Sai96]. Stewart [Ste89] obtained this result in the form of a bound on the norm of oblique projectors, and provided an independent, geometric proof. O'Leary [O'L90] later proved that Stewart's bound is sharp. It was also proven by Todd in [Tod90]. Finally, in [MTW93], the authors derive it as a consequence of Hoffman's Lemma [Hof52]. We too make use of this result in the analysis of our algorithms developed in Chapters 2 and 3.

Algorithm 3: Dual affine-scaling algorithm for LP

Input: LP data A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$;

Parameters: $\beta \in (0, 1)$, and $\varepsilon > 0$;

Output: $\mathcal{O}(\varepsilon)$-optimal primal-dual solution (x, y, s)

while $x^{\mathrm{T}}s \geq \varepsilon$ *or* $\|[x]_-\|_1 \geq \varepsilon$ **do**

  Compute Δy as in (1.19);

  Set $y^+ := y + \beta\Delta y$ and $s^+ := c - A^{\mathrm{T}}y^+$;

  Set x^+ associated to (y^+, s^+) as in (1.22);

end

In the stopping criterion, $[x]_- := \min\{x, 0\}$, the negative part of x.

Remark 1.8.1. *Here we discuss the stopping criterion used in Algorithm 3, and the meaning of the claimed $\mathcal{O}(\varepsilon)$ solution. When this variant of the DAS method stops, we have $x^{\mathrm{T}}s \leq \varepsilon$, $\|[x]_-\| \leq \varepsilon$, $Ax = b$, and (y, s) feasible. $x^{\mathrm{T}}s = c^{\mathrm{T}}x - b^{\mathrm{T}}y \leq \varepsilon$ implies $b^{\mathrm{T}}y \geq c^{\mathrm{T}}x - \varepsilon$. Now taking $(y, s) = (y^*, s^*)$ optimal in $c^{\mathrm{T}}x = b^{\mathrm{T}}y + x^{\mathrm{T}}s$, we have*

$$c^{\mathrm{T}}x \geq z^* + x^{\mathrm{T}}s^* \geq z^* - \|[x]_-\|_1 \cdot \|s\|_\infty \geq z^* - \epsilon\delta,$$

where z^* is the optimal value of the LP (assumed finite) and $\delta := \min_{\text{optimal } s^*} \|s^*\|_\infty$. So that we have

$$b^{\mathrm{T}}y \geq z^* - \epsilon(1 + \delta).$$

This δ is a kind of condition number for the dual problem [Wri97], but unfortunately may be hard to estimate.

Convergence of the short-step dual affine-scaling algorithm under nondegeneracy assumptions

Under the following nondegeneracy assumptions, a simple proof of convergence of Algorithm 3 can be developed [DY91, BT97, Dik74]:

1. The matrix A has full rank.
2. An interior dual feasible point is given.
3. An interior primal feasible point exists (the dual solution set is nonempty and bounded).
4. Every dual basic feasible solution is non-degenerate, i.e., the inequality constraint $s \geq 0$ has exactly m active components.
5. Every primal basic solution is non-degenerate, i.e., the inequality constraint $x \geq 0$ has exactly $n - m$ active components.

We now sketch a proof of convergence of the dual affine-scaling algorithm. The argument is a combination of arguments used in [DY91] and those used in a convergence analysis of a primal affine-scaling algorithm given in [BT97]. (See also [Dik74].)

Step 0: Algorithm is well defined

The dual iterates remain strictly feasible due to the constraints in (1.17).

Step 1: Ascent and asymptotic complementary slackness

A key fact is that in view of (1.17), Δy is a strict ascent direction for the dual objective $b^{\mathrm{T}}y$. Indeed,

$$b^{\mathrm{T}}\Delta y = \beta \frac{b^{\mathrm{T}}b_y}{\|b_y\|_y} = \beta \|b_y\|_y > 0.$$

Further, by (1.22),

$$\|Sx\| = \|b_y\|_y. \tag{1.23}$$

Therefore, the sequence of dual objective values increases strictly and is bounded, hence it converges to some value z^*. Thus, passing to the limit as $k \to \infty$ in the

relationship

$$z^* \geq b^{\mathrm{T}} y^{k+1} \geq b^{\mathrm{T}} y^k + \beta \|b_y\|_y = b^{\mathrm{T}} y^k + \beta \|S^k x^k\| \geq b^{\mathrm{T}} y^k,$$

gives $\|X^k s^k\| \to 0$, that is, complementary slackness is satisfied asymptotically.

Step 2: Convergence of a subsequence to a dual basic feasible solution and complementary primal basic solution

The assumptions imply that the dual solution set is nonempty and bounded and the nondegeneracy assumptions imply that it is a singleton. This implies that the level set

$$\{y \mid A^{\mathrm{T}} y \leq c,\ b^{\mathrm{T}} y \geq b^{\mathrm{T}} y^0\},$$

which contains the iterates (by monotonicity), is bounded. Thus the sequence of dual iterates remains bounded and therefore has a limit point $\bar{y}$. Let K be an infinite index set on which $y^k \to \bar{y}$. Next it will be shown that this $\bar{y}$ (actually any limit point) is a dual basic feasible solution and that $\{x^k\}$ converges to the associated primal basic solution $\bar{x}$ on K.

Let $\bar{s} = c - A^{\mathrm{T}} \bar{y}$ and (B, N) be the index partition such that $\bar{s}_N > 0$ and $\bar{s}_B = 0$. We claim that A_B is invertible and hence that $\bar{y}$ is the basic feasible solution associated to B and, further, that since $\|X^k s^k\| \to 0$, it holds that $x_N^k \to 0 =: \bar{x}_N$ on K, and $x^k = A_B^{-1}(b - A_N x_N^k) \to A_B^{-1} b := \bar{x}_B$ on K, the primal basic solution associated to B.

To prove the claim, note that, clearly, A_B has at most m independent columns. If it has fewer than m independent columns, then one can find $v \in \mathcal{N}(A_B^{\mathrm{T}})$, such that $v \notin \mathcal{N}(A_N^{\mathrm{T}})$ (since A is full rank). Using this v, and with appropriate t, setting $\tilde{y} := y + tv$, it is possible to get $c_i - a_i^{\mathrm{T}} \tilde{y} = 0$ for some $i \in N$ while maintaining $s_N \geq 0$. Define $\tilde{B} = B \cup i$, and repeat this process until $A_{\tilde{B}}$ has m independent

columns. If $|\tilde{B}| > m$, then selecting m independent columns from $A_{\tilde{B}}$ defines a degenerate dual basic feasible solution, which contradicts our assumption. Thus $|\tilde{B}| = m$. We claim that $B = \tilde{B}$ which would complete the proof. Again, since $\|X^k s^k\| \to 0$, it holds that $x_N^k \to 0$ on K, and hence $A_B x^k = (b - A_N x_N^k) \to b$ on K. If B were strictly contained in $\tilde{B}$, then $A_{\tilde{B}} x^k \to b$ on K as well and so x^k would converge to a degenerate primal basic solution on K, which does not exist under our assumptions.

Step 3: The entire sequence converges.

To show convergence of the sequence, we show that the only limit point of the bounded $\{y^k\}$ sequence is $\bar{y}$. The argument is by contradiction. If there are two limit points (both basic feasible solutions) then there is a subsequence on which the dual iterates are arbitrarily close to one limit point and, on the next iteration, jump arbitrarily close to the second limit point. These limit points are at a positive distance away from each other by the non-degeneracy assumption. This contradicts the ellipsoidal constraint in (1.17) which implies $-1 \leq \frac{s_i^{k+1} - s_i^k}{s_i^k} \leq 1$, i.e., that

$$0 \leq s_i^{k+1} \leq 2 s_i^k.$$

Step 4: $(\bar{y},\ \bar{x})$ **is optimal**

This amounts to showing $\bar{x}_B \geq 0$. The proof is by contradiction. Assume for some $j \in B$ $\bar{x}_j < 0$ and hence $x_j^k < -\delta$ for some $\delta > 0$, and for all k large enough. However (1.22) and (1.23) give

$$\Delta s^k = -A^{\mathrm{T}} \Delta y^k = -\beta \frac{A^{\mathrm{T}} b_y}{\|b_y\|_y} = -\beta \frac{(S^k)^2 x^k}{\|S^k x^k\|},$$

and so $\Delta s_j^k > 0$ for k large enough, which means s_j^k eventually increases at every step, contradicting $\bar{s}_j = 0$.[11]

Step 5: local linear rate

The dual affine-scaling algorithm obeys

$$\frac{b^{\mathrm{T}}\bar{y} - b^{\mathrm{T}}y^{k+1}}{b^{\mathrm{T}}\bar{y} - b^{\mathrm{T}}y^{k}} \leq 1 - \frac{1}{\sqrt{m}} + \varepsilon^k,$$

with $\varepsilon^k \to 0$. The proof is omitted.

We mention now in passing that significant work has been done to eliminate the nondegeneracy assumption and the long step variant has been shown to be globally convergent for all $\beta \leq 2/3$. The analysis is much more involved however [MTW93].

1.8.2 The build-up variant of DAS

In their build-up version of DAS, Dantzig and Ye [DY91] take a subset Q of the columns of A, according to rules described below, and compute

$$\begin{aligned} &\max\ b^{\mathrm{T}}\Delta y \\ &\text{s.t. } \|S_Q^{-1}A_Q^{\mathrm{T}}\Delta y\| \leq \beta < 1 \end{aligned} \tag{1.24}$$

defining $H_Q := A_Q S_Q^{-2} A_Q^{\mathrm{T}}$, and $b_{y,Q} := H_Q^{-1} b$, this can be rewritten as

$$\begin{aligned} &\max\ \langle b_{y,Q}, \Delta y\rangle_{y,Q} \\ &\text{s.t. } \|\Delta y\|_{y,Q} \leq \beta < 1 \end{aligned} \tag{1.25}$$

[11] Perhaps interestingly, our analysis of a primal-dual affine-scaling algorithm comes, along a quite different route, to a very similar argument (inspired there from much earlier work [PTH88]), see section 2.3, Theorem 2.3.8.

The solution to the elementary ellipsoid-constrained problem is given by

$$\Delta y(Q) = \beta \frac{b_{y,Q}}{\|b_{y,Q}\|_{y,Q}} \tag{1.26}$$

The primal variable is defined as

$$x_Q^+ = S_Q^{-2} A_Q^{\mathrm{T}} b_{y,Q}, \tag{1.27}$$

$$x_{\mathbf{n}\setminus Q}^+ = 0. \tag{1.28}$$

Note, x_Q^+ is the solution to

$$\begin{aligned} &\min \|S_Q x_Q\|^2, \\ &\text{s.t. } A_Q x_Q = b. \end{aligned} \tag{1.29}$$

with $s_Q := c_Q - A_Q^{\mathrm{T}} y$.

The constraining ellipsoid now is only inscribed in the region $\{y | A_Q^{\mathrm{T}} y \leq c_Q\}$, which can be seen as an outer approximation to the feasible region, but one may hope that the update $y^+ := y + \Delta y$ is still feasible for the original problem. If indeed this step is feasible with respect to the full constraint set, then it is taken. Otherwise, the algorithm enters a minor-cycle phase where a "blocking constraint" along the direction defined by (1.24) is added to Q and (1.24) is re-solved. If the resulting direction is still infeasible, the process repeats. Clearly this process is guaranteed to terminate once all constraints are added after at most $n - |Q_0|$ minor-cycles, where Q_0 is the initial constraint set. Further, the addition of each constraint in the minor-cycle can be done efficiently by using rank-one updates to recompute $\Delta y(Q)$, etc. Convergence of this method is easily established given the convergence result for the original DAS algorithm, as given above. The necessary adjustments to those arguments are discussed next.

Step 0: Algorithm is well defined

Same.

Step 1: Ascent and asymptotic complementary slackness

It should be clear that $\Delta y(Q)$ is still a strict ascent direction for $b^{\mathrm{T}}y$. The same argument as before gives $\|X^k_{Q^k} s^k_{Q^k}\| \to 0$, and since $x^k_{\mathbf{n}\setminus Q^k} := 0$, it still holds that $\|X^k s^k\| \to 0$

Step 2: Convergence of a subsequence to a dual basic feasible solution and complementary primal basic solution

Same.

Step 3: The entire sequence converges.

The only potential problem is that the relation $0 \leq s_i^{k+1} \leq 2s_i^k$, which held for $i \in B$, and was used to get the contradiction, now holds only if $i \in Q^k$. However this is not a problem really because for $i \in B$, $i \notin Q^k$ can happen only finitely many times since $x_i^k \to \bar{x}_i > 0$, while $x_i^k := 0$ if $i \notin Q^k$.

Step 4: $(\bar{y},\ \bar{x})$ is optimal

The argument is the same since $j \in B$ implies $j \in Q^k$ for all k large enough, as just argued.

Step 5: local linear rate

Same.

Remark 1.8.2. *Actually Dantzig and Ye use stronger assumptions in the proof they give. Here are the exact assumptions from [DY91]:*

1. *$b \neq 0$ and $c \neq 0$.*
2. *Every subset of m columns of A is independent .*
3. *The primal problem is feasible.*
4. *An interior dual feasible point is given.*
5. *Every dual basic feasible solution is non-degenerate.*

6. *Every primal basic solution is non-degenerate.*

(Note Assumptions 5 and 6 imply Assumption 1.) These allow a very similar proof in the unreduced case to the one given, the only significant difference being in step 2 where instead of using boundedness of the dual iterates (which comes from the assumption that the dual solution set is bounded) to obtain a subsequence convergent to a basic solution, they use their Assumptions 2 and 6.

It is interesting that the rule by which the constraints are selected is of no consequence as far as the asymptotic convergence analysis goes. In [DY91], the initial constraint set is chosen according to one of three rules: sort the constraints in descending order according to v_i, where v_i is one of

1. $v_i = 1/s_i^k$,
2. $v_i = x_i^k/s_i^k$,
3. $v_i = x_i^k$,

and keep the first m as Q_0.

The authors add an additional operation to the algorithm: before beginning each minor-cycle they perform a basis inversion on $B = Q_0$ and terminate if it is the optimal basis. They show that under each of the above three rules, the algorithm terminates finitely. This is clear from the asymptotic convergence result under the nondegeneracy assumption (which implies strict complementary slackness of the optimal (x, s)), since eventually, under the above rules, the optimal basis will be identified and selected as Q_0.

Finally we state a version of the Dantzig and Ye's algorithm formally:

Algorithm 4: Dantzig-Ye Build-up affine-scaling algorithm

Input: LP data A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$;

Parameters: $\beta \in (0, 1)$, and $\varepsilon > 0$;

Output: exact or $\mathcal{O}(\varepsilon)$-optimal primal-dual solution (x, y, s)

while $x^{\mathrm{T}}s \geq \varepsilon$ *or* $\|[x]_-\|_1 \geq \varepsilon$ **do**
    Choose initial Q s.t. $\mathrm{rank}(A_Q) = m$, $|Q| = m$;
    if Q *is optimal* **then** Stop;
    while $|Q| < n$ **do**
        Compute $\Delta y(Q)$ as in (1.26) (or update);
        Set $y^+(Q) = y + \Delta y$;
        if $y^+(Q)$ *is infeasible* **then**
            Augment Q with additional constraints;
        else
            Set $y := y^+(Q)$ and $s := c - A^{\mathrm{T}}y(Q)$;
            Set x using (1.28);
            break;
        end
    end
end

1.9 Tone's "active-set" dual potential-reduction method

In 1993 K. Tone [Ton93] used a similar idea as in [DY91], but based his algorithm on the dual potential-reduction (DPR) algorithm of Ye [Ye91]. A benefit of the DPR algorithm over the DAS algorithm is that DPR achieves the best complexity bound known for linear programming. As in the previous section, we will first describe in some detail the parent algorithm.

1.9.1 Ye's (dual) potential reduction algorithm

The name DPR is somewhat of a misnomer, because the algorithm really maintains a sequence (x^k, y^k) of feasible *primal-dual* iterates and, in fact, achieves a constant reduction of the *primal-dual* potential function

$$\varphi_\rho(x, s) = \rho \log(x^{\mathrm{T}} s) - \sum_{i=1}^{n} \log(x_i s_i) \tag{1.30}$$

at each iteration. However, the dual iterates play the lead role, as will be seen. In fact, Ye calls it the "dual-form of the potential reduction algorithm", suggesting, as is indeed the case, that there is a corresponding primal-form.

Any algorithm that can achieve, in every iteration, a constant reduction of size $\delta > 0$ say, in $\varphi_\rho(x, s)$ (with $\rho > n$), achieves a nice complexity bound. To see this, the inequality

$$\varphi_n(x, s) = n \log n + n \log \frac{x^{\mathrm{T}} s}{n} - n \log \left(\prod_{i=1}^{n} x_i s_i \right)^{\frac{1}{n}} \geq n \log n. \tag{1.31}$$

is needed, which follows from taking logarithms of both sides in the arithmetic-geometric mean inequality

$$\frac{x^{\mathrm{T}} s}{n} \geq \left(\prod_{i=1}^{n} x_i s_i \right)^{\frac{1}{n}}.$$

In view of (1.31), it can be seen that after the kth step of the iteration of such algorithm,

$$\begin{aligned} \varphi_\rho(x^0, s^0) - k\delta \geq \varphi_\rho(x^k, s^k) &= \varphi_\rho(x^k, s^k) - \varphi_n(x^k, s^k) + \varphi_n(x^k, s^k) \\ &= (\rho - n) \log((x^k)^{\mathrm{T}} s^k) + \varphi_n(x^k, s^k) \\ &\geq (\rho - n) \log((x^k)^{\mathrm{T}} s^k) + n \log n \end{aligned}$$

will hold, or equivalently,

$$(x^k)^{\mathrm{T}} s^k \leq e^{\frac{\varphi_\rho(x^0,s^0)-k\delta-n\log n}{\rho-n}}$$

will hold. This last expression shows that $(x^k)^{\mathrm{T}} s^k \leq \varepsilon$ is guaranteed after the right hand side is less than ε, which is true for any k greater than or equal to

$$k^* = \left\lceil \frac{\varphi_\rho(x^0, s^0) + (\rho - n)\log\frac{1}{\varepsilon} - n\log n}{\delta} \right\rceil. \tag{1.32}$$

If $\varphi_\rho(x^0, s^0) = \mathcal{O}(\sqrt{n})$ and $\rho = n + \sqrt{n}$, then DPR achieves the best known complexity for LP, namely $\mathcal{O}(\sqrt{n}\log\frac{1}{\varepsilon})$ iterations.[12]

Now we describe how the Ye's DPR algorithm achieves the required constant reduction of φ_ρ. For this, Ye used the dual potential function, defined for $\bar{z} > b^{\mathrm{T}} y$ by

$$\varphi_\rho^{\mathrm{d}}(y, \bar{z}) := \rho\log(\bar{z} - b^{\mathrm{T}} y) - \sum_{i=1}^{n} \log(s_i). \tag{1.33}$$

There is a close relation between the dual and primal-dual potential functions: if (x, s) are primal and dual feasible and $\bar{z} = c^{\mathrm{T}} x$, then $x^{\mathrm{T}} s = \bar{z} - b^{\mathrm{T}} y$, so that

$$\varphi_\rho^{\mathrm{d}}(y, \bar{z}) := \varphi_\rho(x, s) + \sum_{i=1}^{m} \log(x_i),$$

and in particular, for a fixed x^0, $s^0 = c - A^{\mathrm{T}} y^0$, and $s^1 = c - A^{\mathrm{T}} y^1$, it holds that

$$\varphi_\rho^{\mathrm{d}}(y^0, \bar{z}) - \varphi_\rho^{\mathrm{d}}(y^1, \bar{z}) = \varphi_\rho(x^0, s^1) - \varphi_\rho(x^0, s^0).$$

Thus, for fixed x^0, a given decrease in the dual potential function corresponds to the same decrease in the primal-dual potential function.

[12] One might wonder why not take $\rho < n + \sqrt{n}$. It turns out the algorithm cannot always achieve the δ decrease in the potential function for any smaller ρ. See (1.55) below.

Ye's DPR algorithm (as well as Tone's variant) maintains a sequence of points $\{(x^k, y^k, s^k, \bar{z}^k)\}$, which satisfy the relations

$$s^k = c - A^{\mathrm{T}} c^k > 0, \tag{1.34}$$

$$Ax^k = b,\ x^k > 0, \tag{1.35}$$

$$\bar{z}^k = c^{\mathrm{T}} x^k. \tag{1.36}$$

In order to achieve the required reduction of the potential function, two types of steps are needed. The first, which is referred to as the *dual step*, attempts to decrease the dual potential function (and hence the primal-dual potential function) in a direct way by minimizing the linearized potential function over a feasible ellipsoid. The dual step is defined as the solution to

$$\begin{aligned} &\min\ \nabla\varphi_D(y, \bar{z})^{\mathrm{T}} \Delta y, \\ &\text{s.t. } \|S^{-1} A^{\mathrm{T}} \Delta y\| \leq \beta. \end{aligned} \tag{1.37}$$

Note this is the steepest descent direction for $\varphi_D(y, \bar{z})$ with respect to the inner-product $\langle x, y \rangle_y := x^{\mathrm{T}} H y$ on $\mathbb{R}^m$, defined by the Hessian of the log-barrier for the feasible region $H := AS^{-2}A^{\mathrm{T}}$. (Note also, this is the same constraint used in the DAS algorithm in (1.17).) Writing (1.37) in terms of this inner product gives

$$\begin{aligned} &\min\ \langle H^{-1} \nabla\varphi_D(y, \bar{z}), \Delta y \rangle_y, \\ &\text{s.t. } \|\Delta y\|_y \leq \beta. \end{aligned} \tag{1.38}$$

As before, the solution to this problem is now clear,

$$\Delta y = -\beta \frac{d}{\|d\|_y}, \tag{1.39}$$

where $d := H^{-1}\nabla\varphi_D(y,\bar{z})$, and the optimal value of (1.38) is

$$-\beta\|d\|_y.$$

The quantity $\|d\|_y$ gives the magnitude of the directional derivative of $\varphi_\rho^{\mathrm{d}}(y,\bar{z})$ along Δy, as well as the size of the step measured in the H-norm. Defining the *prescaled* step

$$p := -S^{-1}A^{\mathrm{T}}H^{-1}\nabla\varphi_D(y,\bar{z}) \tag{1.40}$$

so that the relation $\|p\| = \|d\|_y$, holds, allows things to be phrased in terms of the Euclidean inner product (as is done in the references). When $\|p\| = \|d\|_y$ is "large", a significant decrease in $\varphi_\rho^{\mathrm{d}}$ (and hence in φ_ρ) along the direction d, is expected. This is indeed the case, and this is what DPR does. Specifically, for an a priori fixed $\gamma \in (0,1)$, if $\|p\| \geq \gamma$, the algorithm takes a dual step, which consist of setting

$$y^+ := y + \Delta y, \tag{1.41}$$

$$s^+ := s + \Delta s, \tag{1.42}$$

$$x^+ := x, \tag{1.43}$$

$$z^+ := z. \tag{1.44}$$

Now we show, following the analysis of [Ye91], that a fixed reduction in φ_ρ can be achieved after a dual step. For this, we use the fact that, by construction, $\|S^{-1}\Delta s\| \leq \beta < 1$, and that the standard bound (at least standard in interior-point methods, see, e.g., [Ye91])

$$e^{\mathrm{T}}d \geq \sum_{i=1}^{n}\log(1+d_i) \geq e^{\mathrm{T}}d - \frac{\|d\|^2}{2(1-\|d\|_\infty)}. \tag{1.45}$$

is valid for any $d \in \mathbb{R}^n$ with $\|d\|_\infty < 1$. The second inequality is the interesting

part; it can be derived from the Taylor expansion

$$\log(1+x) = \sum_{i=1}^{\infty} (-1)^{k-1} \frac{x^k}{k},$$

by bounding terms larger than second order. Using (1.45), gives

$$\begin{aligned}
\varphi_\rho(x, s^+) - \varphi_\rho(x, s) &= \rho \log(1 + \frac{x^{\mathrm{T}} \Delta s}{x^{\mathrm{T}} s}) - \sum_{i=1}^{n} \log(1 + \frac{\Delta s_i}{s_i}) \\
&\leq \frac{\rho}{x^{\mathrm{T}} s} x^{\mathrm{T}} \Delta s - \sum_{i=1}^{n} \log(1 + \frac{\Delta s_i}{s_i}) \\
&\leq -\frac{\rho}{\bar{z} - b^{\mathrm{T}} y} b' \Delta y - e^{\mathrm{T}} S^{-1} \Delta s + \frac{\|S^{-1} \Delta s\|^2}{2(1 - \|S^{-1} \Delta s\|)} \qquad (1.46) \\
&= \nabla \varphi_D(y, \bar{z})^{\mathrm{T}} \Delta y + \frac{\beta^2}{2(1-\beta)} \\
&= -\beta \|d\|_y + \frac{\beta^2}{2(1-\beta)} \\
&\leq -\beta\gamma + \frac{\beta^2}{2(1-\beta)}. \qquad (1.47)
\end{aligned}$$

The first two inequalities use (1.45) (upper then lower) and the last inequality holds if $\|d\|_y = \|p\| \geq \gamma$, as assumed. Whenever this is the case, a constant decrease can be achieved: for example, with $\gamma = \frac{1}{2}$ and $\beta = \frac{3}{5}$, the decrease is at least 0.05.

On the other hand if $\|p\| = \|d\|_y$ is small, then DPR instead takes a primal step, which is described next. Although the potential reduction algorithm is different in spirit than path following methods, which must maintain proximity to the central path, the central path does play a role. In particular, the point on the path with duality measure[13]

$$r := \frac{x^{\mathrm{T}} s}{\rho} = \frac{\bar{z} - b^{\mathrm{T}} y}{\rho}$$

serves as a reference point for the DPR method in the following sense. Considering the central path conditions ($S\hat{x} = re$, along with primal and dual feasibility), DPR

[13]Note that, since $\rho > n$, this is a point with a smaller duality gap than the current iterate.

defines the primal estimates defined by the following least-squares problem.

$$\begin{aligned} &\min \ \|\frac{1}{r}S\hat{x} - e\| \\ &\text{s.t. } A\hat{x} = b. \end{aligned} \tag{1.48}$$

If the optimal value is zero and $\hat{x}$ is nonnegative, then s is on the central path with duality measure r.

In any event, when (1.48) is rewritten as

$$\begin{aligned} &\min \ \|S(\hat{x} - x) + Sx - re\| \\ &\text{s.t. } AS^{-1}S(\hat{x} - x) = 0, \end{aligned}$$

the solution

$$\hat{x} = x - S^{-1}\mathcal{P}_{\mathcal{N}(AS^{-1})}(Sx - re) \tag{1.49}$$

is evident, and it holds that

$$\begin{aligned} \frac{1}{r}S\hat{x} - e &= \left(\frac{1}{r}Sx - e\right) - \mathcal{P}_{\mathcal{N}(AS^{-1})}\left(\frac{1}{r}Sx - e\right) \\ &= \mathcal{P}_{\mathcal{R}(S^{-1}A^{\mathrm{T}})}\left(\frac{1}{r}Sx - e\right). \end{aligned} \tag{1.50}$$

Then, since

$$\nabla\varphi_D(y, \bar{z}) = -\frac{1}{r}b + AS^{-1}e = -AS^{-1}\left(\frac{1}{r}Sx - e\right),$$

so that, from (1.40),

$$p = \mathcal{P}_{\mathcal{R}(S^{-1}A^{\mathrm{T}})}\left(\frac{1}{r}Sx - e\right),$$

which gives the important relations

$$p = \frac{1}{r} S\hat{x} - e, \tag{1.51}$$

$$\hat{x} = rS^{-1}(e + p), \tag{1.52}$$

showing that when $\|p\| < 1$, $\hat{x} > 0$ also holds, so that hence $\hat{x}$ is a primal interior feasible point, giving a candidate primal update. DPR uses this update: whenever $\|p\| < \gamma < 1$, the algorithm takes a primal step, which consists of setting

$$\begin{aligned} y^+ &:= y, \\ s^+ &:= s, \\ x^+ &:= \hat{x}, \\ z^+ &:= c^{\mathrm{T}} x^+. \end{aligned}$$

Ye [Ye91] showed that this *primal update* can also be used to get a constant decrease in φ_ρ. Using (1.50) and (1.52), first note that

$$(x^+) = rS^{-1}(p + e) = \frac{x^{\mathrm{T}} s}{\rho} S^{-1}(p + e) \tag{1.53}$$

$$(x^+)^{\mathrm{T}} s = e^{\mathrm{T}} S x^+ = re^{\mathrm{T}}(p + e) = \frac{x^{\mathrm{T}} s}{\rho}(e^{\mathrm{T}} p + n) \tag{1.54}$$

so that

$$
\begin{aligned}
\varphi_\rho(x^+, s) - \varphi_\rho(x, s) = & \\
&= \rho \log(\frac{(x^+)^{\mathrm{T}} s}{x^{\mathrm{T}} s}) - \sum_{i=1}^{n} \log x_i^+ + \sum_{i=1}^{n} \log x_i \\
&= \rho \log(\frac{e^{\mathrm{T}} p + n}{\rho}) - \sum_{i=1}^{n} \log(r \frac{p_i + 1}{s_i}) + \sum_{i=1}^{n} \log x_i \\
&\leq \rho(\frac{e^{\mathrm{T}} p + n}{\rho} - 1) - \sum_{i=1}^{n} \log(p_i + 1) + n \log \rho - (n \log x^{\mathrm{T}} s - \sum_{i=1}^{n} \log x_i s_i) \\
&\leq e^{\mathrm{T}} p + n - \rho - \sum_{i=1}^{n} \log(p_i + 1) + n \log \frac{\rho}{n} \\
&\leq e^{\mathrm{T}} p + n - \rho - e^{\mathrm{T}} p + \frac{\|p_i\|^2}{2(1 - \|p\|)} + n \log \frac{\rho}{n} \\
&\leq -(\rho - n) + n \log \frac{\rho}{n} + \frac{\gamma^2}{2(1 - \gamma)}, \qquad (1.55)
\end{aligned}
$$

where the lower bound (1.31) for $\varphi_n(x, s)$ has been used in the fifth line. If $\rho = n + \sqrt{n}$, the first two terms in the final bound decrease strictly in n and converge to the value of -0.5, and for $n = 1$ they evaluate to $\log(2) - 1 = -0.3069$. Thus, for example, taking $\gamma = \frac{1}{2}$, the potential function decreases by at least 0.05.

Thus, with this choice of ρ, and many choices of (γ, β), DPR is seen to achieve a constant decrease in φ_ρ whether a dual step or a primal step is used. Therefore, the complexity result (1.32) holds for DPR.

Algorithm 5: Ye's DPR algorithm [Ye91]

Input: LP data A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$, primal feasible x, $\bar{z} = c^{\mathrm{T}}x$;[14]

Parameters: $\rho = n + \sqrt{n},\ \gamma \in (0,1), \beta \in (0,1), \varepsilon > 0$;

Output: ε-optimal primal-dual solution (x, y, s)

while $x^{\mathrm{T}}s \geq \varepsilon$ **do**

  Compute p using (1.40);

  if $\|p\| \geq \gamma$ **then**

    dual step

    Compute Δy using (1.39);

    Set $y := y + \Delta y$ and $s := c - A^{\mathrm{T}}y$;

  else

    primal step

    Compute $\hat{x}$ by (1.49);

    Set $x := \hat{x}$, $\bar{z} := c^{\mathrm{T}}x$;

  end

end

[15] Various, more practical, versions of the above algorithm, which allow for line-searches in the dual step, are possible. As long as the guaranteed potential reduction is achieved, the same convergence and complexity result holds.

1.9.2 Tone's "active-set" variant

Tone's variant of the DPR incorporates constraint-reduction in a similar way to the Dantzig and Ye variant of the dual affine-scaling algorithm. In particular, Tone's

[14] More generally $\bar{z}$ can be any upper bound on the dual optimal value, in which case x should be taken positive such that $x^{\mathrm{T}}s > \varepsilon$.

dual step is defined by the solution to

$$\begin{aligned} &\max \nabla\varphi_\rho^{\mathrm{d}}(y,\bar{z})^{\mathrm{T}}\Delta y, \\ &\text{s.t. } \|S_Q^{-1}A_Q^{\mathrm{T}}\Delta y\| \le \beta(Q). \end{aligned} \tag{1.56}$$

The ellipsoid constraint now is inscribed in a relaxed polyhedron (containing the feasible region) so that, again, the resulting step is not necessarily feasible for any $\beta < 1$. However, rather than fixing β a priori, as in Danzig and Ye, and "minor-cycling" to add new constraints until feasibility is achieved, instead, a special $\beta = \beta(Q)$ is chosen as a function of Q, for which the step is assured to be feasible for the original constraints. (This $\beta(Q)$ need not be computed in practice; it is only needed for the analysis.) A minor-cycle is still used, but its purpose is to achieve sufficient decrease of the potential function rather than feasibility.

We first make a general observation regarding the feasibility question. Given M and N, (symmetric) positive definite matrices, consider the two ellipsoids

$$E_M = \{x \in \mathbb{R}^m \mid x^{\mathrm{T}}Mx \le 1\}, \tag{1.57}$$

$$E_N = \{x \in \mathbb{R}^m \mid x^{\mathrm{T}}Nx \le 1\}. \tag{1.58}$$

It is not hard to see that $E_M \subset E_N$ if and only if

$$x^{\mathrm{T}}Nx \le x^{\mathrm{T}}Mx, \quad \text{for all } x \in \mathbb{R}^m,$$

which is often written as $N \preceq M$. A sufficient condition for this is that $\lambda_{\max}(N) \le \lambda_{\min}(M)$.[16]

[16]This is not necessary however, e.g., $N = \mathrm{diag}(3,1)$, $M = \mathrm{diag}(4,2)$.

Next, define $E_Q(\beta(Q))$ as the ellipsoid indicated in (1.56), or more precisely

$$E_Q(\beta(Q)) := \{y' \mid \|S^{-1}A_Q^{\mathrm{T}}(y' - y)\| \leq \beta(Q)\}.$$

We would like to know for what Q and $\beta(Q)$ the ellipsoid $E_Q(\beta(Q))$ is contained in the feasible region. We know that $E_{\mathbf{n}}(1) = \{y' \mid \|S^{-1}A^{\mathrm{T}}(y' - y)\| \leq 1\}$ is feasible, and, based on the discussion of the previous paragraph, $E_Q(\beta(Q)) \subseteq E_{\mathbf{n}}(1)$ if and only if

$$H \preceq \frac{1}{\beta^2} H_Q, \tag{1.59}$$

(where $H = AS^{-2}A^{\mathrm{T}}$ and $H_Q = A_Q S_Q^{-2} A_Q^{\mathrm{T}}$), or equivalently,

$$H_{\mathbf{n}\setminus Q} \preceq \left(\frac{1}{\beta^2} - 1\right) H_Q.$$

A sufficient condition for the latter relation is

$$\lambda_{\max}(H_{\mathbf{n}\setminus Q}) \leq \left(\frac{1}{\beta^2} - 1\right) \lambda_{\min}(H_Q), \tag{1.60}$$

and the largest β for which (1.60) holds is

$$\beta^*(Q) := \sqrt{\frac{\lambda_{\min}(H_Q)}{\lambda_{\max}(H_{\mathbf{n}\setminus Q}) + \lambda_{\min}(H_Q)}}.$$

In light of these results, since (1.59) holds for $\beta = \beta^*(Q)$, the following lemma found in [Ton93] follows easily.

Lemma 1.9.1. *(i)*

$$\|S^{-1}A^{\mathrm{T}}\Delta y\|^2 \leq \frac{1}{\beta^*(Q)^2} \|S_Q^{-1}A_Q^{\mathrm{T}}\Delta y\|^2$$

(ii) For all $\beta(Q) < \beta^*(Q)$, the solution to (1.56) *is strictly dual feasible.*

(iii) $\beta^*(Q)$ increases if Q is augmented with additional constraints; and $\beta^*(\mathbf{n}) = 1$.

The solution to (1.56), as for (1.37), is given by

$$\Delta y = -\beta(Q)\frac{d(Q)}{\|d(Q)\|_{y,Q}},$$

where $d(Q) := H_Q^{-1}\nabla\varphi_D(y, \bar{z})$, and $\|\cdot\|_{y,Q}$ is the norm defined in terms of H_Q. The optimal value of (1.56) is

$$-\beta(Q)\|d(Q)\|_{y,Q}.$$

As in the DPR algorithm, when $\|d(Q)\|_{y,Q}$ is large (greater than some fixed $\gamma \in (0, 1)$), then the potential function has a large directional derivative along this direction, and a significant decrease of the dual potential function is expected along $d(Q)$. A possible problem now, that did not exist for DPR, is that $\beta^*(Q)$ may be very small, preventing progress. However, if the working set Q is chosen as $Q = \mathbf{n}$ and $\|d(Q)\|_{y,Q} \geq \gamma$, then the analysis for DPR applies ($\beta^*(\mathbf{n}) = 1$), and a constant decrease, say δ^{**}, is guaranteed. The DPR algorithm sets a threshold $\delta^* < \delta^{**}$ and enters a minor-cycle that, starting from an initial Q, repeatedly solves (1.56), and if $\|d(Q)\|_{y,Q} \geq \gamma$, checks if

$$\delta(Q) := \varphi_\rho(x, s) - \varphi_\rho(x, s + \Delta s) \geq \delta^*. \tag{1.61}$$

In fact, after a dual step, using Lemma 1.9.1 (i), and comparing (1.47)-(1.46), it can

be seen that

$$\begin{aligned}
&\varphi_\rho(x, s^+) - \varphi_\rho(x, s) = \\
&\leq -\frac{\rho}{\bar{z} - b^{\mathrm{T}} y} b' \Delta y(Q) + e^{\mathrm{T}} S^{-1} \Delta s(Q) + \frac{\|S^{-1} \Delta s(Q)\|^2}{2(1 - \|S^{-1} \Delta s(Q)\|)} \\
&\leq \nabla \varphi_D(y, \bar{z})^{\mathrm{T}} \Delta y(Q) + \frac{\frac{1}{\beta^*(Q)^2} \|S_Q^{-1} \Delta s_Q(Q)\|^2}{2(1 - \frac{1}{\beta^*(Q)} \|S_Q^{-1} \Delta s_Q(Q)\|)} \\
&\leq -\beta \|d(Q)\|_y + \frac{\frac{\beta^2}{\beta^*(Q)^2}}{2(1 - \frac{\beta}{\beta^*(Q)})} \\
&\leq -\beta\gamma + \frac{\frac{\beta^2}{\beta^*(Q)^2}}{2(1 - \frac{\beta}{\beta^*(Q)})}.
\end{aligned} \tag{1.62}$$

If $\beta(Q)$ is chosen as $\beta(Q) := \tau \beta^*(Q)^2$, with $\tau \leq 0.5$, then $\frac{\beta(Q)}{\beta^*(Q)} \leq 0.5$ (since $\beta^*(Q) \leq 1$) and

$$\varphi_\rho(x, s^+) - \varphi_\rho(x, s) \leq -\beta\gamma + \frac{\frac{\beta^2}{\beta^*(Q)^2}}{2(1 - \frac{\beta}{\beta^*(Q)})} \leq -(\tau\gamma - \tau^2)\beta^*(Q)^2 \tag{1.63}$$

So if $0 < \tau < \gamma < 1$ then a constant decrease is achieved in the dual step. For example, taking $\tau = \frac{1}{3}$ and $\gamma = \frac{1}{2}$, allows $\delta^{**} = \frac{1}{18} \simeq 0.0556$ and say $\delta^* = 0.05$. In any event, if (1.61) holds, then the dual step is taken and the next iteration begins, otherwise more constraints are added to Q and (1.56) is resolved. This process ends once (1.61) holds for some Q or $\|d(Q)\|_{y,Q} < \gamma$. In the latter case, instead a primal step can be taken, which is described next.

The prescaled step $p(Q)$ is defined, analogously to (1.40), as

$$p := \begin{pmatrix} p_Q(Q) \\ p_{\bar{Q}}(Q) \end{pmatrix} := \begin{pmatrix} -S_Q^{-1} A^{\mathrm{T}} H^{-1} \nabla \varphi(y, \bar{z}) \\ 0 \end{pmatrix}. \tag{1.64}$$

Note that $\|p_Q(Q)\| = \|p(Q)\| = \|d(Q)\|_{y,Q}$. The relationships

$$\hat{x}(Q) = rS^{-1}(p(Q) + e), \tag{1.65}$$

$$p(Q) = \frac{1}{r}S\hat{x}(Q) - e, \tag{1.66}$$

are the analogs of (1.51)-(1.52) in the DPR algorithm, also hold, where here, x is defined as the solution to (1.48). In Tone's variant there is no analog for (1.48), but instead $\hat{x}(Q)$ is *defined* by (1.65). It is clear that $\hat{x}(Q)$ so defined is a strictly feasible primal point whenever $\|p(Q)\| < 1$, and the analysis of the primal step for DPR relied only on feasibility and the relationship (1.65). Thus, if ever $\|p(Q)\| = \|d(Q)\|_{y,Q} < \gamma \leq 1$ then the primal step gives the required decrease. Note that in view of (1.66), $p_{\mathbf{n}\setminus Q} = 0$ means that after a primal update, the components $(\hat{x}_{\mathbf{n}\setminus Q}, s_{\mathbf{n}\setminus Q})$ are perfectly centered.[17]

Each iteration terminates either with a dual step satisfying (1.61) or a primal step for which the constant reduction is achieved. It is possible (however unlikely) that, in every iteration, $n - m$ minor-cycle iterations are needed (if constraints are added one at a time to Q), but ultimately, the constant reduction can always be achieved for φ_ρ and so the same (major) iteration complexity results. The hope, of course, is that very few minor-cycles are needed, and each iteration uses only a small subset Q of the total constraint set, thus making the solution of (1.56) cheaper.

As a final remark, one may have become concerned about the choice $\beta(Q) = \tau\beta^*(Q)^2$, since $\beta^*(Q)$ is very expensive to evaluate. This is no real problem however, because in practice, the fixed step of length $\beta(Q)$ is replaced with a line-search. If the constant decrease of δ^* is guaranteed in a dual step by a fixed step, it is certainly achieved by a line search (at least in an exact line search).

We now state Tone's algorithm formally.

[17] We borrow this idea in our algorithm rMPC*, which we propose and analyze in Chapter 2.

Algorithm 6: Tone's Active-Set DPR algorithm [Ton93]

Input: LP data A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$, primal feasible x, $\bar{z} = c^{\mathrm{T}}x$;

Parameters: $\rho = n + \sqrt{n},\ \gamma \in (0,1), \delta^* > 0, \varepsilon > 0$;

Output: ε-optimal primal-dual solution (x, y, s)

while $x^{\mathrm{T}}s \geq \varepsilon$ **do**
  Choose initial Q s.t. $\mathrm{rank}(A_Q) = m$;
  while $\delta < \delta^*$ **do**
    Compute $p(Q)$ using (1.64);
    if $\|p(Q)\| \geq \gamma$ **then** *dual step*
      Compute $\Delta y(Q) = -H_Q^{-1}\varphi_\rho^{\mathrm{d}}(y, \bar{z})$ and set $\Delta s(Q) = -A^{\mathrm{T}}\Delta y(Q)$;
      Set $\tau = \arg\min\{\varphi_\rho(x, s + t\Delta s(Q)) \mid t \geq 0\}$;
      Set $\delta = \varphi_\rho(x, s) - \varphi_\rho(x, s + \tau\Delta s(Q))$;
      if $\delta < \delta^*$ **then**
        Augment Q with additional constraints;
      else
        set $y := y + \tau\Delta y(Q)$ and $s := s + \tau\Delta s(Q)$;
      end
    else *primal step*
      Set $x := \hat{x}(Q)$ and $\bar{z} := c^{\mathrm{T}}x$;
      break; *(sufficient decrease will be satisfied)*
    end
  end
end

In practice, the minor-cycling can greatly slow the algorithm. Tone recommends a rank-one updating procedure for adding new constraints during the minor-cycles. This can help somewhat, but it would still be much better to find a way to eliminate the minor-cycles altogether. In an impressive piece of work, Kaliski and Ye found that this could indeed be done (at the cost of a weakened complexity estimate) for

the highly structured class of connected network flow problems.

1.9.3 Kaliski and Ye's variant of Tone's method for connected network flow problems

Kaliski and Ye [KY93] proposed a variant of Tone's algorithm, deemed the "shortcut potential reduction" algorithm, which was tailored to solving large scale transportation problems and, more generally, connected network flow problems. For such problems, several simplifications can be made to Tone's method, and most importantly the minor-cycling can be avoided.

The transportation problem is a special case of the network flow linear programming problem. It is one of the most important and well studied classes of LPs [BT97], [Lue84]. The problem is as follows: given n_s sources and n_d destinations, with all sources connected to all destinations, determine the minimal cost "flow" in the network that moves the full supply b_i^s available at source i, and satisfies the demand b_j^d at destination j. If the flow from source i to destination j is given by matrix X_{ij}, and x is the column stacked vector of X, and $b := \begin{pmatrix} b^s \\ b^d \end{pmatrix}$, then this results in an LP in standard primal form with constraint matrix A (the node-arc incidence matrix) of size $m \times n$ with $m = n_s + n_d$ and $n = n_s \cdot n_d$, and of the form

$$A = \begin{pmatrix} e^{\mathrm{T}} & 0 & \dots & 0 \\ 0 & e^{\mathrm{T}} & \dots & 0 \\ \dots & & & \\ 0 & 0 & \dots & e^{\mathrm{T}} \\ I & I & \dots & I \end{pmatrix}$$

Notice A is very sparse with all entries either 0 or 1. (The primal simplex algorithm is extremely effective at taking advantage of this problem's structure, the resulting

algorithm is often called the network simplex method.) An important property of the transportation problem is that every basic feasible solution corresponds to a minimal spanning tree (MST) in the network graph, and vice-versa [BT97]. Thus, in the context of constraint-reduction, keeping an MST in the constraint set will guarantee that the normal matrix is invertible.

In their algorithm, the constraints are sorted according to one of three rules, exactly as in Dantzig-Ye. Constraints are added to the set Q, one-by-one according to one of these orderings, until an MST is contained in Q. An efficient sorting routine is developed for this purpose—the hybrid quicksort. Since they consider very large sparse problems, the authors decided to use an iterative method for computing the search direction. The normal equations are solved using preconditioned conjugate gradient (PCG) on a partial normal matrix formed from only those columns of A contained in Q. The partial normal matrix formed from the columns in the MST, which can be efficiently computed, is used as the preconditioner. They claim that this is an extremely efficient preconditioner. In particular, near the solution, they observe that often, merely $m/100$ PCG iterations are required to obtain a 10^{-6} residual. The authors also propose a finite termination method where an advanced iterate is projected onto the optimal primal/dual face. They show that this is effective in the numerical experiments, often saving more than half of the iterations on large problems.

All of Kaliski and Ye's improvements to and specializations of Tone's method are significant. However, we feel that the key achievement of their work is that they show, for the class of connected network flow problems, the minor-cycling in Tone's algorithm can be eliminated. Using only the initial constraint set (all constraints kept in sorted order until the MST is obtained) a fixed reduction in the potential function can be achieved. This reduction is inversely related to the column dimension m which means that complexity estimate of Tone's method is

somewhat weakened. However, this is really somewhat of a theoretical breakthrough for constraint-reduction; it says that, at least for this restricted class of problems, we really do not need the entire constraint set to get a polynomial complexity result. However, both Dantzig-Ye and Tone leave the possibility that in every iteration $Q = \mathbf{n}$ ultimately, so that these algorithms could potentially be much less efficient than the base algorithm due to a huge amount of minor-cycling, which is of course opposite of the desired effect. The following result [KY93, Lemma 4] provides the key:

Lemma 1.9.2. *[KY93] Choose Q so that it contains an MST for the graph, and choose τ_1 and τ_2 so that*

$$s_i^{-2} \geq \tau_1 \text{ for } i \in Q \quad \text{and} \quad s_i^{-2} \leq \tau_2 \text{ for } i \in \bar{Q}$$

Then

$$\lambda_{\min}(H_Q) \geq \tau_1(m-1)^{-2} \quad \text{and} \quad \lambda_{\max}(H_{\mathbf{n}\setminus Q}) \leq 2\tau_2(m-1)$$

and hence

$$\beta^*(Q) \geq \left(1 + \frac{2\tau_2(m-1)}{\tau_1(m-1)^{-2}}\right)^{-1/2} \geq \left(1 + \frac{2\tau_2}{\tau_1}(m-1)^3\right)^{-1/2} = \mathcal{O}(m^{-3/2})$$

In our opinion this is a brilliant result. Their proof relies heavily on the fact that A is a node-arc incidence matrix. Thus, by (1.63), in (1.32), $\delta = \mathcal{O}(\beta^*(Q)^2) = \mathcal{O}(m^{-3})$ so that the iteration complexity becomes $\mathcal{O}(\sqrt{n}m^3 \log \frac{1}{\varepsilon})$.[18]

There is one caveat here: the number of constraints needed to obtain the MST is not bounded a-priori except by n. However, the authors claim that in the numerical experiments, no more than $2m$ were ever needed.

[18]Kaliski and Ye claim a complexity of $\mathcal{O}(\sqrt{n}m^{3/2} \log \frac{1}{\varepsilon})$.

1.10 Den Hertog et al.'s build-up-and-down path following method

In [HRT94], Den Hertog, building on work of Ye [Ye92], proposed a "build-up and down" path-following algorithm based on a dual logarithmic barrier method.

1.10.1 Dual logarithmic barrier method

Den Hertog's base algorithm is the dual logarithmic-barrier path-following method, which generates a sequence of approximate minimizers of the logarithmic barrier function

$$f(y, \mu) = -b^{\mathrm{T}} y - \mu \sum_{i=1}^{n} \log s_i, \tag{1.67}$$

where $s_i = c_i - a_i^{\mathrm{T}} y$, as usual. The exact minimizers of f define the dual central path. The optimality conditions for this problem are

$$\nabla f(y, \mu) = -b + \mu A S^{-1} e = 0, \tag{1.68}$$

$$s = c - A^{\mathrm{T}} y > 0. \tag{1.69}$$

Defining $x := \mu S^{-1} e$, these can be written as

$$Ax = b, \tag{1.70}$$

$$A^{\mathrm{T}} y + s = c, \tag{1.71}$$

$$Xs = \mu e, \tag{1.72}$$

$$(x, s) > 0. \tag{1.73}$$

The Newton step is

$$p = -\nabla^2 f(y,\mu)^{-1}\nabla f(y,\mu) \tag{1.74}$$

$$= -(AS^{-2}A^{\mathrm{T}})^{-1}(-b + \mu A^{\mathrm{T}}S^{-1}e) \tag{1.75}$$

$$= (AS^{-2}A^{\mathrm{T}})^{-1}(b - Ax). \tag{1.76}$$

The log-barrier method works by starting with y^0, a near minimizer of $f(\cdot,\mu_0)$, for a given μ_0, where nearness is measured by the Newton decrement, i.e., the f-Hessian norm of the Newton step for f. This quantity can also be expressed as

$$\delta(y,\mu) = \min\left\{\left\|\frac{Xs}{\mu} - e\right\| \mid Ax = b\right\}, \tag{1.77}$$

which defines a primal estimate x similar to that of the DAS and DPR algorithms in (1.51)-(1.52) and (1.65)-(1.66). A "β-approximate" minimizer of $f(\cdot,\mu)$, or a β-approximate μ-center, is a point y with $\delta(y,\mu) < \beta$. A point y is called simply an "approximate μ-center" if it is a β-approximate μ-center for $\beta = 1$; any such point is in the quadratic convergence region of Newton's method for $f(\cdot,\mu)$.

Once the iteration reaches a β-approximate μ-center, for appropriate β, the barrier parameter μ is then decreased by a fixed fraction, i.e.,

$$\mu^+ := (1-\theta)\mu,$$

and Newton steps are taken on $f(\cdot,\mu^+)$ until an approximate minimizer is again reached. As $\mu \to 0$ the iterates approach the solution to the LP. For more detail see [Her92, Ren01]. For very small θ, namely $\theta = \mathcal{O}(\frac{1}{\sqrt{n}})$, we have the "short-step" path following method, and it can be shown that only one Newton step, which is guaranteed to remain feasible, is needed to re-obtain an approximate minimizer after an update of μ. These methods require only $\mathcal{O}(\sqrt{n}\log\frac{1}{\varepsilon})$ iterations to achieve

an approximate minimizer with an objective value within ε of optimality (this is the best known bound for LP, it is also achieved by DPR and many others). For larger $\theta = \mathcal{O}(1)$ we have the long-step algorithms, and more Newton steps may be needed. In this case the iteration complexity bound increases to $\mathcal{O}(n \log \frac{1}{\varepsilon})$, but paradoxically, practical performance is much better.

1.10.2 A constraint-reduced variant

Den Hertog [HRT94] applied constraint reduction methodology to the logarithmic barrier method using the following approach. Initially, a small set of working constraints is selected, defining a (relaxed) cutting-plane model of the feasible region of the original or "master" problem. Consider the "relaxed" optimization problem of minimizing the objective over the cutting-plane model of the feasible region. The algorithm starts from a point that is simultaneously feasible for the master problem, and is an approximate μ-center for the relaxed problem. (Den Hertog also shows a way to obtain such a point from a "weighted" approximate μ-center for the original problem.)

The algorithm maintains a sequence of iterates feasible for the master problem by following the central path of the relaxed problem, just as is done in the log-barrier method described above. Specifically, at each major iteration, μ is decreased to $\bar{\mu} = (1 - \theta)\mu$ and Newton steps are taken on $f(\cdot, \bar{\mu})$ to bring the iterates back to the vicinity of the central path of the relaxed problem. Since these iterates may stray outside the feasible region of the master problem, this process must be carefully monitored. In particular, the central path of the relaxed problem is not likely to lie entirely in the feasible region for the master problem, so the target point corresponding to $\bar{\mu}$ on the relaxed central path may be infeasible for the master problem.

Thus, at some point (unless we are very lucky), the iterates will approach

or violate a constraint of the master problem not included in the relaxed problem. When this happens, the algorithm reacts by stepping back to the previous iterate (safely in the feasible region of the master problem) and adding the constraints that caused the disruption to the working set. Newton steps for the updated relaxed problem are then taken to bring the iterates near the central path of the new relaxed problem, and then the path-following process resumes. Constraints in the relaxed problem that have large slack (and thus may be expected to be inactive at the solution and/or have little effect on the current path following process) can be pruned at the beginning of each major iteration. Of course, once this is done, additional Newton steps may be needed to return the iterate to the vicinity of the central path of the updated relaxed problem.

Using analysis extending that of Ye [Ye92], the effect of shifting, adding, and deleting constraints on the proximity measure and on the log-barrier are studied, and this analysis is used to show that the algorithm achieves the same polynomial complexity bounds as the standard log-barrier method except with n is replaced by q^*, the maximum number of constraints in relaxed problem. Notably, as in the work of [Ye92], this suggests that both the computational cost per iteration and the *iteration complexity* may be reduced since it is likely that $q^* < n$. However, in general, q^* is not known a-priori, and the only sure upper bound for it n. We now state den Hertog's algorithm formally.

Algorithm 7: den Hertog's build-up and down path-following method

Input: LP data A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$, an approximate μ-center for the relaxed problem defined by the initial constraint set Q; Parameters: $\theta \in (0,1), \varepsilon > 0$;

Output: ε-optimal primal-dual solution (x, y, s)

while $\mu \geq \varepsilon$ **do**

Delete constraints (details omitted);

$\mu := (1-\theta)\mu$;

while $\delta_Q > 0.25$ **do** (Center and add constraints)

Compute the Newton step p for $f_Q(\cdot, \mu)$;

Set $\tilde{y} = \min\{f_Q(y + tp, \mu) \mid c_i - a_i^{\mathrm{T}}(y + tp) > 0\}$;

if $c_i - a_i^{\mathrm{T}}\tilde{y} < \tau$ *for any* $i \in \mathbf{n} \setminus Q$ **then**

Add all such i to Q;

else

$y = \tilde{y}$

end

end

It is not critical that the feasible region of the problem be polyhedral for the basic mechanism of this algorithm to apply, and so, as other cutting-plane algorithms for LP naturally extend to more general convex optimization problems (see sections 1.4.2 and 1.5), so too does this method.

1.10.3 Den Hertog et al.'s logarithmic barrier cutting-plane method

In [HKRT95], den Hertog et al. extended the method developed in [HRT94] to more general convex optimization problems and carried out numerical experiments on a variety of problems, including some coming from semi-infinite programming. The principle is identical to the method for linear programming described above: the

central path of a relaxed cutting-plane model of the feasible region is followed as long as it remains feasible for the master problem. When the path strays outside the master feasible region, the cutting-plane model is updated by adding appropriate constraints, and the path-following process resumes, now following the central path of the updated model.

1.11 Primal-dual (symmetric) interior-point methods

While the algorithms reviewed in the previous sections have nice theoretical convergence properties—Tone's achieves the best known complexity, and den Hertog's perhaps even somewhat improves it—they are all based on dual algorithms.[19] On the other hand, it is the "symmetric", as Ye calls them [Ye97], primal-dual interior-point methods (PDIPMs) that have been incorporated into practical algorithms for large-scale LP and its extensions. Thus, it is only natural to try to apply constraint-reduction methodology to PDIPMs.

The PDIPMs apply Newton's method, or variations thereof, to the equality portion of the perturbed Karush-Kuhn-Tucker (KKT) optimality conditions for the primal-dual pair (1.3), namely,

$$\begin{aligned} A^{\mathrm{T}}y + s - c &= 0, \\ Ax - b &= 0, \\ Xs - \tau e &= 0, \\ (x, s) &\geq 0, \end{aligned} \tag{1.78}$$

[19] See discussion in section 1.9.1 for some qualification of this statement.

with $X = \mathrm{diag}(x)$, $S = \mathrm{diag}(s)$, e the vector of all ones, and τ a positive parameter. As τ ranges over $(0, \infty)$, the unique (if it exists)[20] solution (x, y, s) to this system traces out the primal-dual "central path". Newton-type steps for system (1.78), which are well defined, in particular, when X and S are positive definite and A has full rank, are obtained by solving

$$\begin{pmatrix} 0 & A^{\mathrm{T}} & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}} y - s \\ b - Ax \\ \sigma\mu e - Xs \end{pmatrix}, \tag{1.79}$$

where we have set $\tau = \sigma\mu$, with $\mu = x^{\mathrm{T}} s/n$ the current "duality measure" and $\sigma \in [0, 1]$. This step aims to eliminate the primal and dual infeasibilities $b - Ax$ and $c - A^{\mathrm{T}} y - s$, while setting the pairwise "complementarity" $x_i s_i$ equal to $\sigma\mu$ for all $i \in \{1, 2, \ldots n\}$, i.e., it aims for the point on the central path with duality measure reduced to $\sigma\mu$. At the two extremes for choice of σ, one gets first, with $\sigma = 0$, the "primal-dual affine-scaling" (PDAS) direction, which aims directly for the solution, and second, with $\sigma = 1$, the "centering direction", which aims for the point on the central path with the current value of the duality measure μ. Thus, the step (1.79) can be thought of as a combination of the two steps, with σ defining the blend. Intuitively, $\sigma > 0$ helps to preserve the inequalities that are also part of the KKT conditions (1.78) which are unlikely to hold at the end of a full affine scaling step.

System (1.79) is often solved by first eliminating Δs, giving the symmetric-

[20] System (1.78) has a unique solution for each $\tau > 0$ (equivalently, for *some* τ) if there exists (x, y, s) with $Ax = b$, $A^{\mathrm{T}} y + s = c$ and $(x, s) > 0$ [Wri97, Thm. 2.8, p39]. This is the so called "Slater" or "interior-point" condition.

indefinite "KKT" or "augmented" system

$$\begin{pmatrix} -X^{-1}S & A^{\mathrm{T}} \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}}y - \sigma\mu X^{-1}e \\ b - Ax \end{pmatrix}, \tag{1.80}$$
$$\Delta s = -A^{\mathrm{T}}\Delta y + (c - A^{\mathrm{T}}y - s),$$

or by further eliminating Δx, giving the "normal system"

$$\begin{aligned} AS^{-1}XA^{\mathrm{T}}\Delta y &= b - Ax + AS^{-1}X(c - A^{\mathrm{T}}y - \sigma\mu X^{-1}e), \\ \Delta s &= -A^{\mathrm{T}}\Delta y + (c - A^{\mathrm{T}}y - s), \\ \Delta x &= -S^{-1}X(-A^{\mathrm{T}}\Delta y + c - A^{\mathrm{T}}y - \sigma\mu X^{-1}e). \end{aligned} \tag{1.81}$$

In the dual-feasible ($s = c - A^{\mathrm{T}}y > 0$) affine-scaling ($\sigma = 0$) variant (relevant to [TAW06] and the algorithms developed in Chapters 2 and 3), the normal equations simplify to

$$\begin{aligned} AS^{-1}XA^{\mathrm{T}}\Delta y &= b, \\ \Delta s &= -A^{\mathrm{T}}\Delta y, \\ \Delta x &= -x - S^{-1}X\Delta s. \end{aligned} \tag{1.82}$$

There are many different variations of the PDIPM method for each of the standard classes of IPMs mentioned at the beginning of section 1.7: affine-scaling, potential-reduction, and path-following. There is also a large number of elegant theoretical results regarding their convergence properties. Many of the PDIPMs achieve the best known iteration complexity bound $\mathcal{O}(\sqrt{n}\log\frac{1}{\varepsilon})$ for LP. The books by Ye [Ye97] and Wright [Wri97], which is specifically dedicated to PDIPMs, provide a very nice coverage of these results.

1.12 Tits et al.'s primal-dual affine-scaling algorithm

In [TAW06], the authors proposed a simple constraint-reduction methodology and applied it to two PDIPMs: a primal-dual affine-scaling (PDAS) algorithm and Mehrotra's predictor-corrector (MPC) algorithm. A common component of the prior work [DY91, Ton93, HRT94], discussed in the previous sections, is the "minor-cycle" that adds constraints and tries again when the step generated using the working constraint set fails to pass certain acceptability tests. (As discussed earlier, Kaliski and Ye [KY93] showed that the minor-cycle could be eliminated when Tone's algorithm is applied to connected network flow problems.) In contrast, no minor-cycle is used in the rPDAS algorithm of [TAW06]. As in [DY91, Ton93, HRT94], at each iteration, rPDAS uses a small working set of constraints to generate a step, but this step is not subjected to acceptability tests; it is simply taken. This has the advantage that the cost per iteration can be guaranteed to be cheaper than when the full constraint set is used; however it may preclude polynomial complexity results, as were obtained in [Ton93, HRT94]—it seems likely that some guarantee on the quality of each step, or at least of each group of steps, is needed for such results. Global and local quadratic convergence of rPDAS was proved in [TAW06] (under nondegeneracy assumptions) using a nonlinear programming inspired line of argument [Her82, PTH88].

1.12.1 Constraint-reduced primal-dual affine-scaling algorithm

The algorithm rPDAS has as its parent a PDAS algorithm that simply takes steps along the PDAS direction obtained by solving (1.79) with $\sigma = 0$, and iterates. There are prior convergence analyses available for PDAS [MAR90, Sai94], however, the analysis of [TAW06] is not derived from these approaches, so we omit their

description. The analysis of [TAW06] applies equally well in the unreduced case, as the full set of constraints is always an admissible working set, so the (unreduced) parent and (reduced) child analysis can be viewed as one and the same in this case.

Next, we discuss the method of constraint-reduction used in [TAW06]. Since rPDAS is dual-feasible and uses the Newton direction for the unperturbed KKT equations, i.e., $\tau = 0$ in (1.78), the step equations are of the form (1.82), except rPDAS replaces the normal matrix $AS^{-1}XA^{\mathrm{T}}$ with the partial sum

$$A_Q X_Q S_Q^{-1} A_Q^{\mathrm{T}} = \sum_{i \in Q}^{n} \frac{x_i}{s_i} a_i a_i^{\mathrm{T}}, \tag{1.83}$$

which only includes a subset $Q \subseteq \{1, 2, \ldots n\}$ of "important" constraints (or columns of A). The only restriction imposed in [TAW06] on the choice of Q, is that it must include M, for some $M \geq m$, of the most-active constraints, i.e., those constraints $s_i^{\mathrm{T}} y - c_i = s_i \geq 0$, with smallest slack value s_i. The specific value of M is tied to an assumption that every $m \times M$ submatrix of A be of rank m.[21] The authors denote the admissible set of constraint sets at a dual strictly feasible point y by $\mathcal{Q}_M(y)$.

Algorithm rPDAS then solves (1.81) with no other changes. The resulting direction $(\Delta y, \Delta s)$ is used in a line-search that takes a step nearly to the boundary of the dual feasible region. Specifically the dual variables are updated by

$$(y^+, s^+) := (y + \hat{t}\Delta y, s + \hat{t}\Delta s), \tag{1.84}$$

where

$$\hat{t} := \min\{1, \max\{\beta \bar{t}, t - \|\Delta y\|\}\}, \tag{1.85}$$

$$\bar{t} := \arg\max\{t \in [0, 1] \mid x + t\Delta s \geq 0\}, \tag{1.86}$$

[21] We will have much more to say about such assumptions in Chapter 3.

and where the specific form of $\hat{t}$ preserves strict feasibility ($s > 0$) and allows quadratic convergence, by letting $\hat{t} \to 1$ near the solution (when $\Delta y \to 0$).

The primal variable x is updated to x^+ by clipping the full step $\tilde{x} := x + \Delta x$ from below and above to values $\underline{\chi}$ and $\overline{\chi}$, respectively, using the complicated-looking formula:

$$x^+ := \min\{\max\{\tilde{x}, \varphi\}, \overline{\chi}\}, \tag{1.87}$$

$$\varphi \equiv \varphi(\tilde{x}, \Delta y) := \max\{\underline{\chi}, \|\Delta y\|^2 + \|[\tilde{x}]_-\|^2\} \tag{1.88}$$

where $[\tilde{x}]_- := \max\{\tilde{x}, 0\}$, and the min and max are applied componentwise. The full (Newton) step $\tilde{x}$ is clipped from below by $\varphi(\tilde{x}, \Delta y)$ which keeps the primal iterates away from zero, away from the optimal set, while allowing local quadratic convergence; the update is also explicitly clipped from above to guarantee boundedness of the primal iterates. The global convergence proof has a similar flavor to that of the DAS algorithm described in section 1.8, while the quadratic local rate essentially follows from the fact that its steps are based on the Newton step. Here we formally state algorithm rPDAS of [TAW06] (without stopping criteria).

Algorithm 8: rPDAS algorithm of [TAW06]

Input: LP data: A, b, c; Initial iterate: $y, s = c - A^{\mathrm{T}}y > 0$, working constraint set $Q^0 \subseteq \{1, 2, \ldots n\}$, with $|Q| \geq M$; Parameters: $M \in \{1, 2, \ldots n\}$, $\beta \in (0, 1)$, $\underline{\chi} > 0$, $\overline{\chi} > 0$;

while *forever* **do**

Compute $(\Delta y, \Delta s, \Delta x)$ by solving (1.81), with partial normal matrix (1.83), and set $\tilde{x} = x + \Delta x$;

Compute largest dual feasible step from (1.86), and $\hat{t}$ by (1.85);

Update dual variables to (y^+, s^+) using (1.84);

Update primal variables to x^+ using (1.88);

Choose $Q \in \mathcal{Q}_M(y)$.

end

As discussed above, to the author's knowledge, aside from the analysis of rPDAS in [TAW06], no attempts have been made to date at analyzing constraint-reduced versions of PDIPMs, the leading class of interior-points methods over the past decade. This observation applies in particular to the current "champion" among the PDIPMs, Mehrotra's predictor-corrector algorithm (MPC, [Meh92]), which combines an adaptive choice of the centering parameter σ in (1.79) and a second order correction to the affine-scaling direction, which together have proven to be extremely effective in practice.

1.12.2 A simple constraint-reduced Mehrotra predictor-corrector

At the end of their paper, in [TAW06], the authors also proposed and presented some numerical results for a constraint reduced variant of MPC, called rMPC, although without any attempt at analysis. Their variant was based on the simple and highly

practical version of MPC presented (also without analysis) in [Wri97, Ch. 10].[22] To achieve constraint-reduction, rMPC of [TAW06] used the same method used for their rPDAS algorithm, namely they replace the full normal matrix, with the partial sum (1.83). Next we discuss MPC as presented in [Wri97, Ch. 10], and the minor modification to make it into rMPC, as presented in [TAW06].

From a primal-dual interior-point (x, y, s), MPC/rMPC computes the affine scaling direction $(\Delta x^{\mathrm{a}}, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}})$ by solving (1.81), where rMPC replaces the normal matrix with the reduced normal matrix (1.83), for some admissible set of working constraints. The unreduced MPC can equally well solve the augmented or normal equations as in (1.80) and (1.81), while rMPC (and rPDAS) is rather tied to the normal equations. It then computes the maximum primal and dual feasible step lengths along the affine scaling direction via

$$t_{\mathrm{p}}^{\mathrm{a}} := \arg\max\{t \in [0,1] \mid x + t\Delta x^{\mathrm{a}} \geq 0\}, \tag{1.89}$$

$$t_{\mathrm{d}}^{\mathrm{a}} := \arg\max\{t \in [0,1] \mid s + t\Delta s^{\mathrm{a}} \geq 0\}. \tag{1.90}$$

If this maximum step were actually taken, the duality measure would be reduced from $\mu = x^{\mathrm{T}}s/n$ to

$$\mu^{\mathrm{a}} = \frac{1}{n}(x + t_{\mathrm{p}}^{\mathrm{a}}\Delta x^{\mathrm{a}})^{\mathrm{T}}(s + t_{\mathrm{d}}^{\mathrm{a}}\Delta s^{\mathrm{a}}), \tag{1.91}$$

and this might be thought of as a good step if μ^{a} is much smaller than μ, meaning that not much centering is needed. One of the defining features of MPC is that it chooses its "centering parameter" σ adaptively according to the heuristic formula[23]

$$\sigma := (\mu^{\mathrm{a}}/\mu)^3. \tag{1.92}$$

[22] Investigations of the convergence properties of variants of unreduced MPC can be found in [Meh92, ZZ95, ZZ96, SPT07, Car09].

[23] We show in Chapter 2, that an extension of this formula used in our constraint-reduced variant of MPC, allows quadratic convergence whenever the exponent is ≥ 2.

Next, MPC/rMPC computes its combined centering/corrector direction by solving the linear system

$$\begin{pmatrix} 0 & A^{\mathrm{T}} & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x^{\mathrm{c}} \\ \Delta y^{\mathrm{c}} \\ \Delta s^{\mathrm{c}} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \sigma\mu e - \Delta X^{\mathrm{a}} \Delta s^{\mathrm{a}} \end{pmatrix}, \tag{1.93}$$

which uses the same matrix as in (1.79), with modified right-hand-side for $(\Delta x^{\mathrm{c}}, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}})$. The term $\Delta X^{\mathrm{a}} \Delta s^{\mathrm{a}}$ can be shown to introduce a second-order correction to the affine-scaling search direction that helps it to make rapid progress toward the solution (see [Wri97, chap. 10] for more detail). Note that since the matrix in (1.93) is the same as in (1.79), and its normal-matrix required by rMPC is also the same, the centering-corrector direction can be cheaply computed by reusing any factorization that had been computed in the affine-scaling direction calculation. Algorithm MPC/rMPC then combines the affine-scaling and centering-corrector direction to form the total search direction

$$(\Delta x^{\mathrm{m}}, \Delta y^{\mathrm{m}}, \Delta s^{\mathrm{m}}) := (\Delta x^{\mathrm{a}}, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}}) + (\Delta x^{\mathrm{c}}, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}}), \tag{1.94}$$

and computes the maximum step along this direction as

$$\bar{t}_{\mathrm{p}}^{\mathrm{m}} := \arg\max\{t \in [0,1] \mid x + t\Delta x^{\mathrm{m}} \geq 0\}, \tag{1.95}$$

$$\bar{t}_{\mathrm{d}}^{\mathrm{m}} := \arg\max\{t \in [0,1] \mid s + t\Delta s^{\mathrm{m}} \geq 0\}. \tag{1.96}$$

Finally, MPC/rMPC updates the primal and dual variables by taking a step almost all the way to the boundary by computing

$$t_{\mathrm{p}}^{\mathrm{m}} := \beta \bar{t}_{\mathrm{p}}^{\mathrm{m}}, \; t_{\mathrm{d}}^{\mathrm{m}} := \beta \bar{t}_{\mathrm{d}}^{\mathrm{m}}, \tag{1.97}$$

with $\beta \in (0,1)$ an algorithm parameter, and setting

$$(x^+, y^+, s^+) := (x, y, s) + (t_{\mathrm{p}}^{\mathrm{m}} \Delta x^{\mathrm{m}},\ t_{\mathrm{d}}^{\mathrm{m}} \Delta y^{\mathrm{m}},\ t_{\mathrm{d}}^{\mathrm{m}} \Delta s^{\mathrm{m}}). \tag{1.98}$$

Now we formally present Algorithm MPC/rMPC of [Wri97]/[TAW06].

Algorithm 9: Iteration rMPC [Meh92, Wri97], constraint-reduced as in [TAW06]

Input: LP Data: A,b,c; Initial iterate: $y, s > 0$, $x > 0$, working set of constraints $Q \subseteq \mathbf{n}$;

Parameters: $\beta \in (0,1)$;

while *forever* **do**

Compute affine-scaling direction via (1.81) with $\sigma = 0$ and replacing the normal matrix with (1.83);

Determine the maximum feasible affine-scaling step-length using (1.89)-(1.90) and compute σ using (1.92);

Solve (1.93) using the normal equations for the centering-corrector direction;

Form the total search direction using (1.94) and find the maximum feasible step-length using (1.95)-(1.96);

Update the primal and dual variables to (x^+, y^+, s^+) using (1.97) and (1.98);

Choose $Q^+ \in \mathcal{Q}_M(y^+)$;

end

As stated, Algorithm MPC/rMPC has no known convergence guarantees. Previous approaches to providing such guarantees involve introducing certain safeguards or modifications [Meh92, ZZ95, ZZ96, SPT07, Car09]. In the next chapter, as one of the main contributions of this dissertation, we introduce a set of modifications to Algorithm rMPC that allows us to prove global and local quadratic convergence.

1.13 Nicholls' work on infeasible constraint-reduced predictor corrector algorithms for LP

Further numerical investigation of a constraint-reduced (in the same sense as [TAW06]), variant of MPC was conducted in [Nic09]. This work focused in particular on allowing infeasible iterates. In [Nic09], the author presents some ideas for a convergence analysis for such an algorithm. (A full convergence analysis of a dual-feasible constraint-reduced variant of MPC is developed in [WNTO10], which is also the topic of Chapter 2 of the present dissertation.) Nicholls also investigated constraint-reduction for a different infeasible predictor-corrector algorithm for LP from [Pot96] which is similar to the feasible variant presented in [Wri97, Ch. 5, p.90-96].

1.14 Jung et al.'s work on reduced convex QP for SVMs

In [JOT10, JOT08], an extension of the work of Tits et al. in [TAW06] was carried out for convex quadratic programming (QP or CQP), and the authors applied their algorithms to the QP arising in support vector machine (SVM) training, a popular modern technique for designing classifiers for labeled data (see e.g., [SS01]). Jung proves global and local quadratic convergence of his algorithms by extending the analysis of [TAW06]. In Jung et al.'s papers and dissertation, different practical issues were considered, including development of adaptive constraint selection rules for a specific application (namely, SVM training) and allowing for a (dual) infeasible starting point.

Chapter 2

A convergent constraint-reduced Mehrotra predictor-corrector algorithm

In this chapter, we propose and analyze a convergent constraint-reduced variant of MPC that we term rMPC* to distinguish it from rMPC of [TAW06], described in section 1.12.2. Our algorithm uses a minimally restrictive class of constraint selection rules that are somewhat different than those used in Algorithm 8/rPDAS. These constraint selection rules, as in [TAW06], do not require the minor cycles used in many of the prior constraint-reduced IPMs discussed in the previous chapter. We borrow from the line of analysis of [Her82, PTH88] and especially [TAW06] in our analysis, but we use a somewhat different, and perhaps more natural, perspective on the notion of constraint reduction than was put forth in [TAW06] (see Remark 2.2.1 below). We also prove global convergence under assumptions that are significantly milder than those invoked in the analysis of rPDAS in [TAW06]. We then prove q-quadratic local convergence under appropriate nondegeneracy assumptions. The proposed iteration and stronger convergence results apply, as a limiting case, to a variation of rPDAS, thus, essentially improving on the results of [TAW06]. As a further special case, our results apply to standard unreduced primal-dual affine

scaling. In that context, our conclusions (Theorem 2.3.8 and Remark 2.3.1) are weaker than those obtained in the work of Monteiro et al. [MAR90] or, for a different type of affine scaling (closer to the spirit of Dikin's work [Dik67]), in that of Jansen et al. [JRT90]. In particular, we do not prove polynomial complexity. On the other hand, the specific algorithm we analyze has the advantage of allowing for much larger steps, of the order of one compared to steps no larger than $1/n$ (in [MAR90]) or equal to $1/(15\sqrt{n})$ (in [JRT90]), and convergence results on the dual sequence are obtained without an assumption of primal feasibility. Much of this discussion is taken from a paper by Winternitz et al. [WNTO10].

2.1 Notation and a lemma

As the analysis will become formal in this chapter (and in the next), in this section, we fix our notation which is, for the most part standard, and some of which we have already used in Chapter 1. We use $\|\cdot\|$ to denote the 2-norm or its induced operator norm. Given a vector $x \in \mathbb{R}^n$, we let the corresponding capital letter X denote the diagonal $n \times n$ matrix with x on its main diagonal. We define $\mathbf{n} := \{1, 2, \ldots n\}$ and given any index set $Q \subseteq \mathbf{n}$, we use A_Q to denote the $m \times |Q|$ (where $|Q|$ is the cardinality of Q) matrix obtained from A by deleting all columns a_i with $i \notin Q$. Similarly, we use x_Q and s_Q to denote the vectors of size $|Q|$ obtained from x and s by deleting all entries x_i and s_i with $i \notin Q$. $\mathbf{n} \setminus Q := \mathbf{n} \setminus Q$. We define e to be the column vector of ones, with length determined by context. For a vector v, $[v]_-$ is defined by $([v]_-)_i := \min\{v_i, 0\}$. Lowercase k always indicates an iteration count, and limits of the form $y^k \to y^*$ are meant as $k \to \infty$. Uppercase K generally refers to an infinite index set and the qualification "on K" is synonymous with "for $k \in K$". In particular, "$y^k \to y^*$ on K" means $y^k \to y^*$ as $k \to \infty$, $k \in K$. Further, we define

the dual feasible, dual strictly feasible, and dual solution sets, respectively, as

$$\begin{aligned} F &:= \{y \in \mathbb{R}^m \mid A^{\mathrm{T}} y \leq c\}, \\ F^o &:= \{y \in \mathbb{R}^m \mid A^{\mathrm{T}} y < c\}, \\ F^* &:= \{y \in F \mid b^{\mathrm{T}} y \geq b^{\mathrm{T}} w \text{ for all } w \in F\}. \end{aligned}$$

We term a vector $y \in \mathbb{R}^m$ *stationary* if $y \in F^{\mathrm{s}}$, where

$$F^{\mathrm{s}} := \{y \in F \mid \exists x \in \mathbb{R}^n, \text{ s.t. } Ax = b,\ X(c - A^{\mathrm{T}} y) = 0\}. \tag{2.1}$$

Given $y \in F^{\mathrm{s}}$, every x satisfying the conditions of (2.1) is called *a multiplier associated to the stationary point* y. A stationary vector y belongs to F^* if and only if $x \geq 0$ for some multiplier x. The active set at $y \in F$ is

$$I(y) := \{i \in \mathbf{n} \mid a_i^{\mathrm{T}} y = c_i\}.$$

Next, we define

$$J(G, u, v) := \begin{pmatrix} 0 & G^{\mathrm{T}} & I \\ G & 0 & 0 \\ \mathrm{diag}(v) & 0 & \mathrm{diag}(u) \end{pmatrix} \tag{2.2}$$

and

$$J_{\mathrm{a}}(G, u, v) := \begin{pmatrix} G & 0 \\ \mathrm{diag}(v) & -\mathrm{diag}(u) G^{\mathrm{T}} \end{pmatrix} \tag{2.3}$$

for any matrix G and vectors u and v of compatible dimensions (*cf.* systems (1.79) and (1.80)). Finally, we write down our first lemma which is taken nearly verbatim from [TAW06, Lemma 1].

Lemma 2.1.1. *$J_a(A, x, s)$ is nonsingular if and only if $J(A, x, s)$ is. Further suppose $x \geq 0$ and $s \geq 0$. Then $J(A, x, s)$ is nonsingular if and only if (i) $x_i + s_i > 0$ for all*

i, (ii) $\{a_i : s_i = 0\}$ *is linearly independent, and (iii)* $\{a_i : x_i \neq 0\}$ *spans* $\mathbb{R}^m$.

2.2 Development of a provably convergent variant of rMPC

As mentioned at the end of section 1.12, we will introduce several modification to Algorithm 9 that will allow us to prove some convergence results. Specifically, aside from the constraint-reduction mechanism (to be discussed in section 2.2.1), Algorithm 10/rMPC* proposed below has four significant differences from Algorithm 9/rMPC, all motivated by the structure of the convergence analysis adapted from [Her82, PTH88, TZ94, TAW06]. These differences, which occur in the adaptive selection of the centering parameter in (1.92), the formation of the total direction in (1.94), and in the update of the primal and dual variables in (1.98), are discussed next. Numerical experience suggests that they do not negatively affect the performance of the algorithm.

The first difference is the formula for the centering parameter σ. Instead of using (1.92), we set

$$\sigma := (1 - t^{\mathrm{a}})^{\lambda},$$

where $t^{\mathrm{a}} := \min\{t_{\mathrm{p}}^{\mathrm{a}},\ t_{\mathrm{d}}^{\mathrm{a}}\}$, $t_{\mathrm{p}}^{\mathrm{a}},\ t_{\mathrm{d}}^{\mathrm{a}}$ are defined in (1.89)-(1.90), and $\lambda \geq 2$ is a scalar algorithm parameter. This formula agrees identically with (1.92) when $\lambda = 3$, (x, y, s) is primal and dual feasible, and $t^{\mathrm{a}} = t_{\mathrm{p}}^{\mathrm{a}} = t_{\mathrm{d}}^{\mathrm{a}}$. In general, both formulas result in similar empirical performance, while the new formula simplifies our analysis and allows us to prove quadratic local convergence in section 2.4 below.

The second difference is in formation of the total direction where we introduce a *mixing parameter* $\gamma \in (0, 1]$ and replace (1.94) with

$$(\Delta x^{\mathrm{m}}, \Delta y^{\mathrm{m}}, \Delta s^{\mathrm{m}}) := (\Delta x^{\mathrm{a}}, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}}) + \gamma(\Delta x^{\mathrm{c}}, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}}). \tag{2.4}$$

Nominally we want $\gamma = 1$, but we reduce γ as needed to enforce three properties of our algorithm that are needed in the analysis. The first such property is the monotonic increase of $b^{\mathrm{T}}y$ mentioned previously. While, given dual feasibility, it is readily verified that Δy^{a} is an ascent direction for $b^{\mathrm{T}}y$ (i.e., $b^{\mathrm{T}}\Delta y^{\mathrm{a}} > 0$), this may not be the case for Δy^{m}, defined in (1.94). To enforce monotonicity we choose $\gamma \leq \gamma_1$ where γ_1 is the largest number in $[0, 1]$ such that

$$b^{\mathrm{T}}(\Delta y^{\mathrm{a}} + \gamma_1 \Delta y^{\mathrm{c}}) \geq \theta b^{\mathrm{T}} \Delta y^{\mathrm{a}}, \tag{2.5}$$

with $\theta \in (0, 1)$ an algorithm parameter. It is easily verified that γ_1 is given by

$$\gamma_1 = \begin{cases} 1 & \text{if} \quad b^{\mathrm{T}}\Delta y^{\mathrm{c}} \geq 0, \\ \min\left\{1,\ (1-\theta)\frac{b^{\mathrm{T}}\Delta y^{\mathrm{a}}}{|b^{\mathrm{T}}\Delta y^{\mathrm{c}}|}\right\} & \text{else}\,. \end{cases} \tag{2.6}$$

The second essential property addressed via the mixing parameter is that the centering-corrector component cannot be too large relative to the affine-scaling component. Specifically, we require

$$\|\gamma\Delta y^{\mathrm{c}}\| \leq \psi\|\Delta y^{\mathrm{a}}\|,\ \|\gamma\Delta x^{\mathrm{c}}\| \leq \psi\|\tilde{x}^{\mathrm{a}}\| \text{ and } \gamma\sigma\mu \leq \psi\|\Delta y^{\mathrm{a}}\|, \tag{2.7}$$

where $\psi \geq 0$ is another algorithm parameter.[1] This property is enforced by requiring $\gamma \leq \gamma_0$, where

$$\gamma_0 := \min\left\{\gamma_1, \psi\frac{\|\Delta y^{\mathrm{a}}\|}{\|\Delta y^{\mathrm{c}}\|}, \psi\frac{\|\tilde{x}^{\mathrm{a}}\|}{\|\Delta x^{\mathrm{c}}\|}, \psi\frac{\|\Delta y^{\mathrm{a}}\|}{\sigma\mu}\right\}. \tag{2.8}$$

The final property enforced by γ is that

$$\bar{t}_{\mathrm{d}}^{\mathrm{m}} \geq \zeta t_{\mathrm{d}}^{\mathrm{a}}, \tag{2.9}$$

[1] If $\psi = 0$, then rMPC* reduces to a constraint-reduced affine-scaling algorithm extremely similar to Algorithm 8/rPDAS from [TAW06] reviewed in section 1.12.

where $\zeta \in (0,1)$ is a third algorithm parameter and $\bar{t}_{\mathrm{d}}^{\mathrm{m}}$ depends on γ via (1.96) and (2.4). We could choose γ to be the largest number in $[0, \gamma_0]$ such that (2.9) holds, but this would seem to require a potentially expensive iterative procedure. Instead, rMPC* sets

$$\gamma := \begin{cases} \gamma_0 & \text{if} \quad \bar{t}_{\mathrm{d},0}^{\mathrm{m}} \geq \zeta t_{\mathrm{d}}^{\mathrm{a}}, \\ \gamma_0 \frac{(1-\zeta)\bar{t}_{\mathrm{d},0}^{\mathrm{m}}}{(1-\zeta)\bar{t}_{\mathrm{d},0}^{\mathrm{m}} + (\zeta t_{\mathrm{d}}^{\mathrm{a}} - \bar{t}_{\mathrm{d},0}^{\mathrm{m}})} & \text{else}, \end{cases} \tag{2.10}$$

where

$$\bar{t}_{\mathrm{d},0}^{\mathrm{m}} := \arg\max\{t \in [0,1] \mid s + t(\Delta s^{\mathrm{a}} + \gamma_0 \Delta s^{\mathrm{c}}) \geq 0\}. \tag{2.11}$$

Geometrically, if $\bar{t}_{\mathrm{d},0}^{\mathrm{m}} \geq \zeta t_{\mathrm{d}}^{\mathrm{a}}$ then $\gamma = \gamma_0$, but otherwise $\gamma \in [0, \gamma_0)$ is selected in such a way that the search direction $\Delta s^{\mathrm{m}} = \Delta s^{\mathrm{a}} + \gamma \Delta s^{\mathrm{c}}$ goes through the intersection of the line segment connecting $s + \zeta t_{\mathrm{d}}^{\mathrm{a}} \Delta s^{\mathrm{a}}$ and $s + \zeta t_{\mathrm{d}}^{\mathrm{a}}(\Delta s^{\mathrm{a}} + \gamma_0 \Delta s^{\mathrm{c}})$ with the feasible line segment connecting $s + t_{\mathrm{d}}^{\mathrm{a}} \Delta s^{\mathrm{a}}$ and $s + \bar{t}_{\mathrm{d},0}^{\mathrm{m}}(\Delta s^{\mathrm{a}} + \gamma_0 \Delta s^{\mathrm{c}})$. See Figure 2.1. Since the intersection point $s + \zeta t_{\mathrm{d}}^{\mathrm{a}}(\Delta s^{\mathrm{a}} + \gamma \Delta s^{\mathrm{c}})$ is feasible, (2.9) will hold. Overall we have

$$\gamma \in [0, \gamma_0] \subseteq [0, \gamma_1] \subseteq [0, 1]. \tag{2.12}$$

In spite of these three requirements on γ, it is typical that $\gamma = 1$ in practice, with appropriate choice of algorithm parameters, as in chapter 4, except when aggressive constraint reduction is used— i.e., very few constraints are retained at each iteration.

The remaining two differences between rMPC* and MPC, aside from constraint reduction, are in the update of the primal and dual variables in 1.98. They are both taken from [TAW06]. First, (1.97) is replaced by

$$t_{\mathrm{p}}^{\mathrm{m}} := \max\{\beta \bar{t}_{\mathrm{p}}^{\mathrm{m}},\ \bar{t}_{\mathrm{p}}^{\mathrm{m}} - \|\Delta y^{\mathrm{a}}\|\} \tag{2.13}$$

and similarly for $t_{\mathrm{d}}^{\mathrm{m}}$, to allow for local quadratic convergence. Second, the primal update is replaced by a componentwise clipped-from-below version of the primal

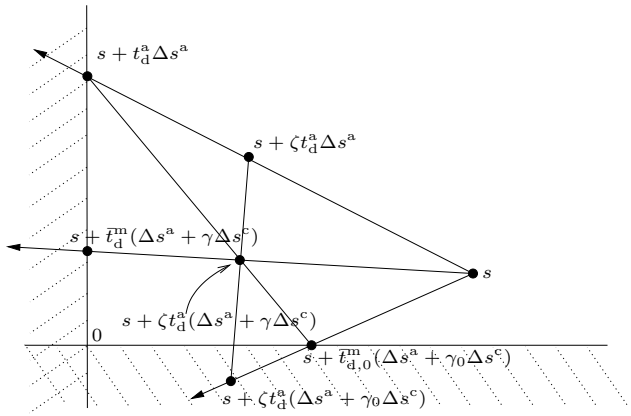

Figure 2.1: Enforcing $\bar{t}_{\rm d}^{\rm m} \geq \zeta t_{\rm d}^{\rm a}$ with γ. The positive orthant here represents the feasible set $s \geq 0$ in two-dimensional slack space (i.e., s-space). The top arrow shows the step taken from some $s > 0$ along the affine scaling direction $\Delta s^{\rm a}$. The bottom arrow is the step along the MPC direction with mixing parameter γ_0. In this picture, the damping factor $\bar{t}_{\rm d,0}^{\rm m}$ is less than $\zeta t_{\rm d}^{\rm a}$, so we do not choose $\gamma = \gamma_0$. Rather, we take a step along the direction from s that passes through the intersection of two lines: the line consisting of points of the form $s + \zeta t_{\rm d}^{\rm a}(\Delta s^{\rm a} + \gamma\Delta s^{\rm c})$ with $\gamma \in [0, \gamma_0]$ and the feasible line connecting $s + t_{\rm d}^{\rm a}\Delta s^{\rm a}$ and $s + \bar{t}_{\rm d,0}^{\rm m}(\Delta s^{\rm a} + \gamma_0\Delta s^{\rm c})$. The maximum feasible step along this direction has length $\bar{t}_{\rm d}^{\rm m} \geq \zeta t_{\rm d}^{\rm a}$.

update in (1.98). Namely, defining $\hat{x} := x + t_{\rm p}^{\rm m}\Delta x^{\rm m}$ and $\tilde{x}^{\rm a} := x + \Delta x^{\rm a}$, for all $i \in \mathbf{n}$, we update x_i to

$$x_i^+ := \max\{\hat{x}_i,\ \min\{\underline{\xi}^{\max}, \varphi\}\}, \tag{2.14}$$

$$\varphi := \|\Delta y^{\rm a}\|^\nu + \|[\tilde{x}^{\rm a}]_-\|^\nu, \tag{2.15}$$

where $\nu \geq 2$ and $\underline{\xi}^{\max} > 0$ (small) are algorithm parameters.[2] The lower bound, $\min\{\underline{\xi}^{\max}, \varphi\}$, ensures that, away from KKT points, the components of x remain bounded away from zero, which is crucial to the global convergence analysis, while allowing for local quadratic convergence. Parameter $\underline{\xi}^{\max}$, the maximum value of the lower bound, is not needed in the convergence analysis, but is important in practice; if $\underline{\xi}^{\max}$ is set sufficiently small, then normally $x^+ = \hat{x}$ and the resulting iteration

[2] In Algorithm 8/rPDAS of [TAW06], the primal update is also clipped from above by a large, user selected value, to insure boundedness of the primal sequence. We show in Lemma 2.3.4 below that such clipping is unnecessary.

emulates the behavior of Algorithm 9/MPC.

2.2.1 A constraint reduction mechanism

Given a working set of constraints Q and a dual-feasible point (x, y, s),[3] we compute an MPC-type direction for the "reduced" primal-dual pair

$$\begin{array}{ccc} \min c_Q^{\mathrm{T}} x_Q & & \max b^{\mathrm{T}} y \\ \text{s.t. } A_Q x_Q = b, & \text{and} & \text{s.t. } A_Q^{\mathrm{T}} y + s_Q = c_Q, \\ x_Q \geq 0, & & s_Q \geq 0. \end{array} \tag{2.16}$$

To that effect, we first compute the "reduced" affine-scaling direction by solving

$$\begin{pmatrix} 0 & A_Q^{\mathrm{T}} & I_Q \\ A_Q & 0 & 0 \\ S_Q & 0 & X_Q \end{pmatrix} \begin{pmatrix} \Delta x_Q^{\mathrm{a}} \\ \Delta y^{\mathrm{a}} \\ \Delta s_Q^{\mathrm{a}} \end{pmatrix} = \begin{pmatrix} 0 \\ b - A_Q x_Q \\ -X_Q s_Q \end{pmatrix} \tag{2.18}$$

[3] If a dual strictly feasible point is not immediately available, we can first use the constraint-reduced method to solve the "phase one" problem

$$\min_{t,y} \{ t \mid A^{\mathrm{T}} y - c \leq te \}, \tag{2.17}$$

for which $(t, y) = (\max\{-c\} + 1, 0)$ is strictly feasible. If an iterate (t, y) of the phase one problem has $t < 0$, then the iteration can be terminated, and y used to start the algorithm on the original problem. If, on the other hand, the phase one problem is solved to optimality, and its optimal value is nonnegative, then the original problem has no strictly feasible point. Recent work, related to that of this dissertation, investigates convergence properties and practical behavior of constraint-reduced PDIPMs that do not require the availability of an initial dual feasible point by adding penalty terms to the objective [HT10], see also [Nic09].

and then the "reduced" centering-corrector direction by solving

$$\begin{pmatrix} 0 & A_Q^{\mathrm{T}} & I_Q \\ A_Q & 0 & 0 \\ S_Q & 0 & X_Q \end{pmatrix} \begin{pmatrix} \Delta x_Q^{\mathrm{c}} \\ \Delta y^{\mathrm{c}} \\ \Delta s_Q^{\mathrm{c}} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \sigma\mu_Q e - \Delta X_Q^{\mathrm{a}} \Delta s_Q^{\mathrm{a}} \end{pmatrix}, \tag{2.19}$$

where $\mu_Q := (x_Q)^{\mathrm{T}}(s_Q)/|Q|$. As discussed above, we combine these components using the mixing parameter γ to get our primal and dual search directions:

$$(\Delta x_Q^{\mathrm{m}}, \Delta y^{\mathrm{m}}, \Delta s_Q^{\mathrm{m}}) := (\Delta x_Q^{\mathrm{a}}, \Delta y^{\mathrm{a}}, \Delta s_Q^{\mathrm{a}}) + \gamma(\Delta x_Q^{\mathrm{c}}, \Delta y^{\mathrm{c}}, \Delta s_Q^{\mathrm{c}}). \tag{2.20}$$

This leaves unspecified the search direction in the $\mathbf{n} \setminus Q$ components of Δx^{m} and Δs^{m}. However, in conjunction with an update of the form (1.98), maintaining dual feasibility from iteration to iteration requires that we set

$$\Delta s_{\mathbf{n}\setminus Q}^{\mathrm{a}} := -A_{\mathbf{n}\setminus Q}^{\mathrm{T}} \Delta y^{\mathrm{a}} \text{ and } \Delta s_{\mathbf{n}\setminus Q}^{\mathrm{c}} := -A_{\mathbf{n}\setminus Q}^{\mathrm{T}} \Delta y^{\mathrm{c}}.$$

Thus, we augment (2.20) accordingly, yielding the search direction for x_Q, y, and s,

$$(\Delta x_Q^{\mathrm{m}}, \Delta y^{\mathrm{m}}, \Delta s^{\mathrm{m}}) = (\Delta x_Q^{\mathrm{a}}, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}}) + \gamma(\Delta x_Q^{\mathrm{c}}, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}}). \tag{2.21}$$

As for $x_{\mathbf{n}\setminus Q}$, we do not update it by taking a step along a computed direction. Rather, inspired by an idea used in Algorithm 6 from [Ton93], we consider the update

$$x_i^+ := \frac{\mu_Q^+}{s_i^+} \quad i \in \mathbf{n} \setminus Q,$$

where $\mu_Q^+ := (x_Q^+)^{\mathrm{T}}(s_Q^+)/|Q|$. This would make $(x_{\mathbf{n}\setminus Q}^+, s_{\mathbf{n}\setminus Q}^+)$ perfectly "centered".

Indeed,

$$\begin{aligned}\mu^+ &= \frac{(x^+)^{\mathrm{T}}(s^+)}{n} = \frac{(x_Q^+)^{\mathrm{T}}(s_Q^+) + (x_{\mathbf{n}\setminus Q}^+)^{\mathrm{T}}(s_{\mathbf{n}\setminus Q}^+)}{n} \\ &= \frac{|Q|}{n}\mu_Q^+ + \sum_{i\in\mathbf{n}\setminus Q}\frac{x_i^+ s_i^+}{n} \\ &= \frac{|Q|}{n}\mu_Q^+ + \frac{n-|Q|}{n}\mu_Q^+ = \mu_Q^+,\end{aligned}$$

and hence $x_i^+ s_i^+ = \mu^+$ for all $i \in \mathbf{n} \setminus Q$. However, in order to ensure boundedness of the primal iterates, we use instead, for $i \in \mathbf{n} \setminus Q$,

$$\hat{x}_i := \frac{\mu_Q^+}{s_i^+}, \quad x_i^+ := \min\{\hat{x}_i, \chi\}, \tag{2.22}$$

where $\chi > 0$ is a large parameter. This clipping is benign because, as proved in the ensuing analysis, under our stated assumptions, all the $n \setminus Q$ components of the vector x constructed by Algorithm 10/rMPC* will be small eventually, regardless of how Q may change from iteration to iteration. In practice, this upper bound will never be active if χ is chosen reasonably large.

Remark 2.2.1. *A somewhat different approach to constraint-reduction, where the motivating idea of ignoring irrelevant constraints is less prominent, is used in Algorithm 8 of [TAW06]. There, as discussed in section 1.12, instead of the reduced systems,* (2.18)-(2.19)*, full systems of equations of the form* (1.79) *are solved via the corresponding normal systems* (1.81)*, only with the normal matrix $AS^{-1}XA^{\mathrm{T}}$ replaced by the reduced normal matrix $A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}}$. Possible benefits of the approach taken here in rMPC* are: (i) the [TAW06] approach is essentially tied to the normal equations, whereas our approach is not, (ii) if we do solve the normal equations* (2.49) *(below) there is a (mild) computational savings over algorithm rMPC of [TAW06], and (iii) computational experiments suggest that rMPC* is at least as efficient as Algorithm 9/rMPC in practice.*

Before formally stating Iteration rMPC*, we describe a general constraint selection rule under which our convergence analysis can be carried out. We use a rule related to the one used in [TAW06] and other past constraint-reduced IPMS, in that we require Q to contain some number of nearly active constraints at the current iterate y.[4] However, the rule here aims to allow the convergence analysis to be carried out under weaker assumptions on the problem data than those used in [TAW06]. In particular, we explicitly require that the selection of Q ensures $\text{rank}(A_Q) = m$, whereas, in [TAW06], this rank condition is enforced indirectly through a rather strong assumption on A. Also, the choice made here makes it possible to (largely) eliminate a strong linear independence assumption, namely, Assumption 3 of [TAW06], equivalent to "nondegeneracy" of all "dual basic feasible solutions".

Before stating the rule, we define two terms used throughout the chapter. For a natural number $M \geq 0$ and a real number $\epsilon > 0$, a set of "M most-active" and the set of "ϵ-active" constraints refer, respectively, to a set of constraints with the M smallest slack values (ties broken arbitrarily) and the set of all constraints with slack value no larger than ϵ.

Rule 2.2.1. *At a dual feasible point y, select Q arbitrarily from the set $\mathfrak{Q}_{\epsilon,M}(y)$ defined below.*

Definition 2.2.1. *Let $\epsilon \in (0, \infty]$, and let $M \in \mathbf{n}$ be an upper bound on the number of constraints active at any dual feasible point. Then a set $Q \subseteq \mathbf{n}$ belongs to $\mathfrak{Q}_{\epsilon,M}(y)$ if and only if the following two conditions hold.*

C_1 : *Q contains all ϵ-active constraints at y among some set of M most-active constraints.*

[4] Of course, nearness to activity can be measured in different ways. Here, the "activity" of a dual constraint refers to the magnitude of the slack value s_i associated to it. When the columns of A are normalized to unit 2-norm, the slack in a constraint is just the Euclidean distance to the constraint boundary. Also see Remark 2.2.4 below on invariance under scaling.

C_2 : *A_Q has full row rank.*

To help clarify Rule 2.2.1, we now describe two extreme variants. First, if the problem is known to be nondegenerate in the sense that the set of vectors a_i associated to dual active constraints at any feasible point y is a linearly independent set, we may set $M = m$ and $\epsilon = \infty$. Then, a minimal Q will consist of m most-active constraints, achieving "maximum" constraint reduction. On the other hand, if we have no prior knowledge of the problem, $M = n$ is the only sure choice, and in this case we may set ϵ equal to a small positive value to enact the constraint reduction.

Rule 2.2.1 leaves quite a bit of freedom in choosing the constraint set. In practice, we have had most success with specific rules that keep a small number, typically $2m$ or $3m$, most-active constraints and then add additional constraints based on heuristics suggested by prior knowledge of the problem structure.

The following two lemmas are immediate consequences of Rule 2.2.1.

Lemma 2.2.2. *Let $x > 0$, $s > 0$, and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ for some $y \in F$. Then $A_Q X_Q S_Q^{-1} A_Q^{\mathrm{T}}$ is positive definite.*

Lemma 2.2.3. *Given $y' \in F$, there exists $\rho > 0$ such that for every $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ with $y \in B(y', \rho) \cap F$ we have $I(y') \subseteq Q$.*

Before specifying Algorithm 10/rMPC*, we state two basic assumptions that guarantee it is well defined.

Assumption 1. *A has full row rank.*

Assumption 2. *The dual strictly feasible set is nonempty.*

All that is needed for the iteration to be well-defined is the existence of a dual strictly feasible point y, that $\mathfrak{Q}_{\epsilon,M}(y)$ be nonempty, and that the linear systems (2.18) and (2.19), of Steps 1 and 3, be solvable. Under Assumption 1, $\mathfrak{Q}_{\epsilon,M}(y)$ is

always nonempty since it then contains $\mathbf{n}$. The solvability of the linear systems then follows from Lemma 2.1.1 using the rank condition C_2 of Definition 2.2.1.

Remark 2.2.2. *The convergence analysis that follows is inspired largely by that of [TAW06], but we make use of significantly weaker assumptions. Both analyses use (at least implicitly) the above two assumptions, but the analysis of [TAW06] also assumes that: 1) every $m \times M$ submatrix of A is full rank for some $M \geq m$ [TAW06, Assumption 1], 2) the dual solution set is nonempty and bounded [TAW06, Assumption 2], and 3) at every dual feasible point, the gradients of all active constraints are linearly independent [TAW06, Assumption 3]. The rPDAS algorithm of [TAW06] also explicitly clips the primal iterates from above. In our analysis of the more general rMPC* below, we show such clipping is unnecessary, we replace 1) with the milder requirement on Q that rank$(A_Q) = m$, and we remove 2) and show that, if the dual solution set is empty, then the algorithm generates a sequence of objective values that improve without bound. Finally, we postpone the application of 3) until the last step of analysis and, alternatively, offer an approach to force convergence to the dual optimal set without 3), see Remark 2.3.1.*

The algorithm definition follows on the next page. Since we refer back to the various steps of the algorithm throughout the analysis, we include rather more detail in its statement than in the previous algorithms.

Algorithm 10: Algorithm rMPC*

Input: LP data: A, b, c; Initial iterate: y, $s = c - A^{\mathrm{T}}y > 0$, $x > 0$;
Parameters: $\beta \in (0,1)$, $\theta \in (0,1)$, $\psi \geq 0$, $\chi > 0$, $\zeta \in (0,1)$, $\lambda \geq 2$,
$\nu \geq 2$, $\underline{\xi}^{\max} \in (0,\infty]$,[5] $\epsilon \in (0,\infty]$ and $M \in \mathbf{n}$;

while *forever* **do**

Step 1 Compute the reduced affine-scaling direction, i.e., choose $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, solve (2.18) for $(\Delta x^{\mathrm{a}}_Q, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}}_Q)$, set $\Delta s^{\mathrm{a}}_{\mathbf{n}\setminus Q} := -A^{\mathrm{T}}_{\mathbf{n}\setminus Q}\Delta y^{\mathrm{a}}$ and compute

$$t^{\mathrm{a}}_{\mathrm{p}} := \arg\max\{t \in [0,1] \mid x_Q + t\Delta x^{\mathrm{a}}_Q \geq 0\}, \tag{2.23}$$

$$t^{\mathrm{a}}_{\mathrm{d}} := \arg\max\{t \in [0,1] \mid s + t\Delta s^{\mathrm{a}} \geq 0\}, \tag{2.24}$$

$$t^{\mathrm{a}} := \min\{t^{\mathrm{a}}_{\mathrm{p}}, t^{\mathrm{a}}_{\mathrm{d}}\}. \tag{2.25}$$

Step 2 Compute the centering parameter

$$\sigma := (1 - t^{\mathrm{a}})^{\lambda}. \tag{2.26}$$

Step 3 Compute the centering-corrector direction, i.e., set $\mu_Q := \frac{(x_Q)^{\mathrm{T}}(s_Q)}{|Q|}$, solve (2.19) for $(\Delta x^{\mathrm{c}}_Q, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}}_Q)$ and set

$$\Delta s^{\mathrm{c}}_{\mathbf{n}\setminus Q} := -A^{\mathrm{T}}_{\mathbf{n}\setminus Q}\Delta y^{\mathrm{c}}.$$

Step 4 Form the total search direction

$$(\Delta x^{\mathrm{m}}_Q, \Delta y^{\mathrm{m}}, \Delta s^{\mathrm{m}}) := (\Delta x^{\mathrm{a}}_Q, \Delta y^{\mathrm{a}}, \Delta s^{\mathrm{a}}) + \gamma(\Delta x^{\mathrm{c}}_Q, \Delta y^{\mathrm{c}}, \Delta s^{\mathrm{c}}), \tag{2.27}$$

where γ is as in (2.10), with μ (in (2.8)) replaced by μ_Q. Set

$$\bar{t}^{\mathrm{m}}_{\mathrm{p}} := \arg\max\{t \in [0,1] \mid x_Q + t\Delta x^{\mathrm{m}}_Q \geq 0\}, \tag{2.28}$$

$$\bar{t}^{\mathrm{m}}_{\mathrm{d}} := \arg\max\{t \in [0,1] \mid s + t\Delta s^{\mathrm{m}} \geq 0\}. \tag{2.29}$$

Step 5 Update the variables: set

$$t^{\mathrm{m}}_{\mathrm{p}} := \max\{\beta\bar{t}^{\mathrm{m}}_{\mathrm{p}},\ \bar{t}^{\mathrm{m}}_{\mathrm{p}} - \|\Delta y^{\mathrm{a}}\|\}, \tag{2.30}$$

$$t^{\mathrm{m}}_{\mathrm{d}} := \max\{\beta\bar{t}^{\mathrm{m}}_{\mathrm{d}},\ \bar{t}^{\mathrm{m}}_{\mathrm{d}} - \|\Delta y^{\mathrm{a}}\|\}, \tag{2.31}$$

and set

$$(\hat{x}_Q, y^+, s^+) := (x_Q, y, s) + (t^{\mathrm{m}}_{\mathrm{p}}\Delta x^{\mathrm{m}}_Q,\ t^{\mathrm{m}}_{\mathrm{d}}\Delta y^{\mathrm{m}},\ t^{\mathrm{m}}_{\mathrm{d}}\Delta s^{\mathrm{m}}). \tag{2.32}$$

Set

$$\tilde{x}^{\mathrm{a}}_i := \begin{cases} x_i + \Delta x^{\mathrm{a}}_i & i \in Q, \\ 0 & i \in \mathbf{n}\setminus Q, \end{cases} \tag{2.33}$$

$$\varphi := \|\Delta y^{\mathrm{a}}\|^{\nu} + \|[\tilde{x}^{\mathrm{a}}]_-\|^{\nu}, \tag{2.34}$$

and for each $i \in Q$, set

$$x^+_i := \max\{\hat{x}_i,\ \min\{\underline{\xi}^{\max}, \varphi\}\}. \tag{2.35}$$

Finally, set

$$\mu^+_Q := \frac{(x^+_Q)^{\mathrm{T}}(s^+_Q)}{|Q|} \tag{2.36}$$

and, for each $i \in \mathbf{n}\setminus Q$, set

$$\hat{x}_i := \frac{\mu^+_Q}{s^+_i}, \tag{2.37}$$

$$x^+_i := \min\{\hat{x}_i, \chi\}. \tag{2.38}$$

end

In the convergence analysis, we will also make use of the quantities $\tilde{x}^{\mathrm{m}}$, $\tilde{s}^{\mathrm{a}}$, and $\tilde{s}^{\mathrm{m}}$ defined similarly to $\tilde{x}^{\mathrm{a}}, \tilde{x}^{\mathrm{m}}$ (the undamped steps)

$$\tilde{x}_i^{\mathrm{m}} := \begin{cases} x_i + \Delta x_i^{\mathrm{m}} & i \in Q, \\ 0 & i \in \mathbf{n} \setminus Q, \end{cases} \tag{2.39}$$

$$\tilde{s}^{\mathrm{a}} := s + \Delta s^{\mathrm{a}}, \tag{2.40}$$

$$\tilde{s}^{\mathrm{m}} := s + \Delta s^{\mathrm{m}}. \tag{2.41}$$

Remark 2.2.3. *Just like MPC/rMPC, rMPC* uses separate step sizes for the primal and dual variables. Often in convergence analyses of MPC-type algorithms, a common step size is assumed, but we found that using separate step sizes works well in practice, and furthermore, was needed in the proof of a critical result (Proposition 2.4.5).*

Remark 2.2.4. *While rMPC* as stated fails to retain the remarkable scaling invariance properties of MPC, invariance under diagonal scaling in the primal space and under Euclidean transformations and uniform diagonal scaling in the dual space can be readily recovered (without affecting the theoretical properties of the algorithm) by modifying iteration rMPC* along lines similar to those discussed in section 5 of [TAW06].*

In closing this section, we note a few immediate results to be used in the sequel. First, the following identities are valid for $j \in \{\mathrm{a}, \mathrm{m}\}$:

$$t_{\mathrm{p}}^j = \min\left\{1, \min\left\{\frac{x_i}{-\Delta x_i^j} \mid i \in Q,\ \Delta x_i^j < 0\right\}\right\}, \tag{2.42}$$

$$t_{\mathrm{d}}^j = \min\left\{1, \min\left\{\frac{s_i}{-\Delta s_i^j} \mid \Delta s_i^j < 0\right\}\right\}. \tag{2.43}$$

[5]The convergence analysis allows for $\underline{\xi}^{\max} = \infty$, i.e., for the simplified version of (2.35): $x_i^+ := \max\{\hat{x}_i,\ \varphi\}$. However a finite, small value of $\underline{\xi}^{\max}$ seems to be beneficial in practice.

Next, the following are direct consequences of equations (2.18)-(2.19) and Steps 1 and 3 of Iteration rMPC*:

$$\Delta s^j = -A^{\mathrm{T}} \Delta y^j \quad \text{for } j \in \{\mathrm{a}, \mathrm{c}, \mathrm{m}\}, \tag{2.44}$$

and, for $i \in Q$,

$$s_i \Delta x_i^{\mathrm{a}} + x_i \Delta s_i^{\mathrm{a}} = -x_i s_i, \tag{2.45}$$

$$\frac{s_i}{-\Delta s_i^{\mathrm{a}}} = \frac{x_i}{\tilde{x}_i^{\mathrm{a}}} \text{ when } \Delta s_i^{\mathrm{a}} \neq 0 \quad \text{and} \quad \frac{x_i}{-\Delta x_i^{\mathrm{a}}} = \frac{s_i}{\tilde{s}_i^{\mathrm{a}}} \text{ when } \Delta x_i^{\mathrm{a}} \neq 0, \tag{2.46}$$

$$s_i \Delta x_i^{\mathrm{m}} + x_i \Delta s_i^{\mathrm{m}} = -x_i s_i + \gamma(\sigma \mu_Q - \Delta x_i^{\mathrm{a}} \Delta s_i^{\mathrm{a}}). \tag{2.47}$$

Further, system (2.18) can alternatively be solved in augmented system form[6]

$$\begin{pmatrix} A_Q & 0 \\ S_Q & -X_Q A_Q^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \Delta x_Q^{\mathrm{a}} \\ \Delta y^{\mathrm{a}} \end{pmatrix} = \begin{pmatrix} b - A_Q x_Q \\ -X_Q s_Q \end{pmatrix}, \tag{2.48}$$
$$\Delta s_Q^{\mathrm{a}} = -A_Q^{\mathrm{T}} \Delta y^{\mathrm{a}},$$

or in normal equations form

$$A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}} \Delta y^{\mathrm{a}} = b, \tag{2.49a}$$

$$\Delta s_Q^{\mathrm{a}} = -A_Q^{\mathrm{T}} \Delta y^{\mathrm{a}}, \tag{2.49b}$$

$$\Delta x_Q^{\mathrm{a}} = -x_Q - S_Q^{-1} X_Q \Delta s_Q^{\mathrm{a}}. \tag{2.49c}$$

[6]This form of the augmented system is equivalent to (1.80) after scaling the second block row by X_Q and then swapping the block rows.

Similarly, (2.19) can be solved in augmented system form

$$\begin{pmatrix} A_Q & 0 \\ S_Q & -X_Q A_Q^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \Delta x_Q^{\mathrm{c}} \\ \Delta y^{\mathrm{c}} \end{pmatrix} = \begin{pmatrix} 0 \\ \sigma\mu_Q e - \Delta X_Q^{\mathrm{a}} \Delta s_Q^{\mathrm{a}} \end{pmatrix}, \tag{2.50}$$
$$\Delta s_Q^{\mathrm{c}} = -A_Q^{\mathrm{T}} \Delta y^{\mathrm{c}},$$

or in normal equations form

$$A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}} \Delta y^{\mathrm{c}} = -A_Q S_Q^{-1} (\sigma\mu_Q - \Delta X_Q^{\mathrm{a}} \Delta s_Q^{\mathrm{a}}), \tag{2.51a}$$
$$\Delta s_Q^{\mathrm{c}} = -A_Q^{\mathrm{T}} \Delta y^{\mathrm{c}}, \tag{2.51b}$$
$$\Delta x_Q^{\mathrm{c}} = -S_Q^{-1} X_Q \Delta s_Q^{\mathrm{c}} + S_Q^{-1} (\sigma\mu_Q - \Delta X_Q^{\mathrm{a}} \Delta s_Q^{\mathrm{a}}). \tag{2.51c}$$

Finally, as an immediate consequence of the definition (2.27) of the rMPC* search direction in Step 4 of Iteration rMPC* and of the expressions (2.8) and (2.10) (in particular (2.8)), we have

$$\|\gamma \Delta y^{\mathrm{c}}\| \leq \psi \|\Delta y^{\mathrm{a}}\|, \quad \gamma\sigma\mu_Q \leq \psi \|\Delta y^{\mathrm{a}}\|. \tag{2.52}$$

2.3 Global convergence analysis

The analysis given here is inspired from the line of argument used in [TAW06] for the rPDAS algorithm, but, as mentioned in the introduction to this chapter, we use less restrictive assumptions.

The following proposition, which builds on [TAW06, Prop. 3], shows that Algorithm rMPC* can be repeated indefinitely and that the dual objective strictly increases.

Proposition 2.3.1. *Let $x > 0$, $s > 0$, and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ for some $y \in \mathbb{R}^m$. Then*

the following hold: (i) $b^{\mathrm{T}}\Delta y^{\mathrm{a}} > 0$*, (ii)* $b^{\mathrm{T}}\Delta y^{\mathrm{m}} \geq \theta b^{\mathrm{T}}\Delta y^{\mathrm{a}}$*, and (iii)* $t_{\mathrm{p}}^{\mathrm{m}} > 0,\ t_{\mathrm{d}}^{\mathrm{m}} > 0,\ y^{+} \in F^{\mathrm{o}},\ s^{+} = c - A^{\mathrm{T}}y^{+} > 0$*, and* $x^{+} > 0$*.*

Proof. Claim (i) follows directly from Lemma 2.2.2, (2.49a) and $b \neq 0$, which imply

$$b^{\mathrm{T}}\Delta y^{\mathrm{a}} = b^{\mathrm{T}}(A_{Q}S_{Q}^{-1}X_{Q}A_{Q}^{\mathrm{T}})^{-1}b > 0.$$

For claim (ii), if $b^{\mathrm{T}}\Delta y^{\mathrm{c}} \geq 0$, then, by claim (i),

$$b^{\mathrm{T}}\Delta y^{\mathrm{m}} = b^{\mathrm{T}}\Delta y^{\mathrm{a}} + \gamma b^{\mathrm{T}}\Delta y^{\mathrm{c}} \geq b^{\mathrm{T}}\Delta y^{\mathrm{a}} \geq \theta b^{\mathrm{T}}\Delta y^{\mathrm{a}},$$

and from Step 4 of Algorithm 10/rMPC*, if $b^{\mathrm{T}}\Delta y^{\mathrm{c}} < 0$ then, using (2.8) and (2.10) ($\gamma \leq \gamma_1$), (2.6), and claim (i), we get

$$\begin{aligned} b^{\mathrm{T}}\Delta y^{\mathrm{m}} &\geq b^{\mathrm{T}}\Delta y^{\mathrm{a}} + \gamma_1 b^{\mathrm{T}}\Delta y^{\mathrm{c}} \geq b^{\mathrm{T}}\Delta y^{\mathrm{a}} + (1-\theta)\frac{b^{\mathrm{T}}\Delta y^{\mathrm{a}}}{|b^{\mathrm{T}}\Delta y^{\mathrm{c}}|}b^{\mathrm{T}}\Delta y^{\mathrm{c}} \\ &= b^{\mathrm{T}}\Delta y^{\mathrm{a}} - (1-\theta)b^{\mathrm{T}}\Delta y^{\mathrm{a}} = \theta b^{\mathrm{T}}\Delta y^{\mathrm{a}}. \end{aligned}$$

Finally, claim (iii) follows from Steps 4 - 5 of Iteration rMPC*. □

It follows from Proposition 2.3.1 that, under Assumption 1, Iteration rMPC* generates an infinite sequence of iterates with monotonically increasing dual objective value. From here on we attach an iteration index k to the iterates.

As a first step, we show that if the sequence $\{y^k\}$ remains bounded (which cannot be guaranteed under our limited assumptions), then it must converge. For this, we make use of the following lemma, a direct consequence of results in [Sai96] (see also [Sai94]).

Lemma 2.3.2. *Let* $A \in \mathbb{R}^{m\times n}$*, full row rank, and* $b \in \mathbb{R}^m$ *be given. Then, (i) there exists* $\rho > 0$ *(depending only on* A *and* b*) such that if, for some positive definite*

diagonal matrix D, Δy solves

$$ADA^{\mathrm{T}}\Delta y = b, \tag{2.53}$$

then

$$\|\Delta y\| \leq \rho b^{\mathrm{T}}\Delta y;$$

and (ii), if a sequence $\{y^k\}$ is such that $\{b^{\mathrm{T}}y^k\}$ is bounded and, for some $\omega > 0$, satisfies

$$\|y^{k+1} - y^k\| \leq \omega b^{\mathrm{T}}(y^{k+1} - y^k) \quad \forall k, \tag{2.54}$$

then $\{y^k\}$ converges.

Proof. The first claim immediately follows from Theorem 5 in [Sai96], noting (as in [Sai94], section 4) that, for some $\alpha > 0$, $\alpha\Delta y$ solves

$$\max\{\ b^{\mathrm{T}}u \mid \|D^{1/2}A^{\mathrm{T}}u\| \leq 1\}.$$

(See also Theorem 7 in [Sai94].)[7] The second claim is proved using the central argument of the proof of Theorem 9 in [Sai96]:

$$\sum_{k=0}^{N-1} \|y^{k+1} - y^k\| \leq \omega \sum_{k=0}^{N-1} b^{\mathrm{T}}(y^{k+1} - y^k) \leq 2\omega v \quad \forall N > 0,$$

where v is an upper bond to $\{b^{\mathrm{T}}y^k\}$, implying that $\{y^k\}$ is Cauchy, and thus converges. □

Lemma 2.3.3. *Suppose Assumptions 1 and 2 hold. If $\{y^k\}$ is bounded then $y^k \to y^*$ for some $y^* \in F$, and if it is not, then $b^{\mathrm{T}}y^k \to \infty$.*

Proof. To prove the lemma it suffices to show that the sequence $\{y^k\}$ generated by rMPC* satisfies inequality (2.54) for some $\omega > 0$. To see why, suppose (2.54) holds.

[7] Note this is of the same form as the ball constrained LP that defines the Algorithm 3/DAS search direction.

If $\{y^k\}$ is bounded then so is $\{b^{\mathrm{T}}y^k\}$ and, in view of Lemma 2.3.2 (ii) and the fact that $\{y^k\}$ is feasible, we have $y^k \to y^*$, for some $y^* \in F$. On the other hand, if $\{y^k\}$ is unbounded, then $\{b^{\mathrm{T}}y^k\}$ is also unbounded (since, in view of Lemma 2.3.2 (ii), having $\{b^{\mathrm{T}}y^k\}$ bounded together with (2.54) would lead to the contradiction that the unbounded sequence $\{y^k\}$ converges). To establish (2.54), in view of (2.32), it suffices to show that, for some $\omega > 0$,

$$\|\Delta y^{\mathrm{m},k}\| \leq \omega b^{\mathrm{T}} \Delta y^{\mathrm{m},k} \quad \forall k.$$

Now, since $\Delta y^{\mathrm{a},k}$ solves the normal equations (2.49a), the hypothesis of Lemma 2.3.2 (i) is validated for $\Delta y^{\mathrm{a},k}$, and thus, for some $\rho > 0$,

$$\|\Delta y^{\mathrm{a},k}\| \leq \rho b^{\mathrm{T}} \Delta y^{\mathrm{a},k} \quad \forall k.$$

With this in hand, we obtain, for all k, using (2.27), (2.12), (2.8), and (2.5),

$$\|\Delta y^{\mathrm{m},k}\| \leq \|\Delta y^{\mathrm{a},k}\| + \gamma^k \|\Delta y^{\mathrm{c},k}\| \leq (1+\psi)\|\Delta y^{\mathrm{a},k}\| \leq (1+\psi)\rho b^{\mathrm{T}} \Delta y^{\mathrm{a},k} \leq (1+\psi)\frac{\rho}{\theta} b^{\mathrm{T}} \Delta y^{\mathrm{m},k},$$

so the sought inequality holds with $\omega := (1+\psi)\frac{\rho}{\theta}$. □

We also have that the primal iterates remain bounded.

Lemma 2.3.4. *Suppose Assumption 1 holds. Then $\{x^k\}$, $\{\tilde{x}^{\mathrm{a},k}\}$, and $\{\tilde{x}^{\mathrm{m},k}\}$ are all bounded.*

Proof. We first show that $\{\tilde{x}^{\mathrm{a},k}\}$ is bounded. Defining $D^k_{Q^k} := X^k_{Q^k}(S^k_{Q^k})^{-1}$ and using (2.49a)-(2.49b) we have $\Delta s^{\mathrm{a},k} = -A^{\mathrm{T}}_{Q^k}(A_{Q^k} D^k_{Q^k} A^{\mathrm{T}}_{Q^k})^{-1} b$, which, using definition (2.33) of $\tilde{x}^{\mathrm{a},k}_{Q^k}$, and (2.49c) gives

$$\tilde{x}^{\mathrm{a},k}_{Q^k} = D^k_{Q^k} A^{\mathrm{T}}_{Q^k}(A_{Q^k} D^k_{Q^k} A^{\mathrm{T}}_{Q^k})^{-1} b. \tag{2.55}$$

Sequences of the form $D^k A^{\mathrm{T}}(AD^k A^{\mathrm{T}})^{-1}$, with A full rank and D^k diagonal and positive definite for all k, are known to be bounded; a proof can be found in [Dik74].[8] Hence $\|\tilde{x}^{\mathrm{a},k}\| = \|\tilde{x}^{\mathrm{a},k}_{Q^k}\| \leq R$ with R independent of k (there are only finitely many choices of Q^k). Finally, boundedness of $\{\tilde{x}^{\mathrm{m},k}\}$ and $\{x^k\}$ is proved as follows. Let R' be such that $\max\{\|x^k\|_\infty, (1+\psi)R, \chi, \underline{\xi}^{\max}\} < R'$, for some k. From (2.39), (2.27), (2.10) ($\gamma \leq \gamma_0$), (2.8), and (2.33), we have

$$\|\tilde{x}^{\mathrm{m},k}\| = \|\tilde{x}^{\mathrm{m},k}_{Q^k}\| = \|x^k_Q + \Delta x^{\mathrm{a},k}_{Q^k} + \gamma \Delta x^{\mathrm{c},k}_{Q^k}\| \leq \|\tilde{x}^{\mathrm{a},k}_{Q^k}\| + \psi \|\tilde{x}^{\mathrm{a},k}_{Q^k}\| \leq (1+\psi)R \leq R', \tag{2.56}$$

and since, as per (2.28), (2.30) and (2.32), $\hat{x}^k_{Q^k}$ is on the line segment between $x^k_{Q^k}$ and the full step $\tilde{x}^{\mathrm{m},k}_{Q^k}$, both of which are bounded in norm by R', we have $\|x^{k+1}_{Q^k}\|_\infty \leq \max\{\|\hat{x}^k_{Q^k}\|_\infty, \underline{\xi}^{\max}\} \leq R'$. On the other hand, the update (2.38) for the $\mathbf{n} \setminus Q^k$ components of x^{k+1}, ensures that

$$\|x^{k+1}_{\mathbf{n}\setminus Q^k}\|_\infty \leq \chi \leq R',$$

and the result follows by induction. □

The global convergence analysis essentially considers two possibilities: either $\Delta y^{\mathrm{a},k} \to 0$ or $\Delta y^{\mathrm{a},k} \not\to 0$. In the former case $y^k \to y^* \in F^{\mathrm{s}}$, which follows from the next lemma. In the latter case, Lemma 2.3.6 and Lemma 2.3.7 show that $y^k \to y^* \in F^*$.

Lemma 2.3.5. *For all k, $A\tilde{x}^{\mathrm{a},k} = b$ and $A\tilde{x}^{\mathrm{m},k} = b$. Further, if Assumption 1 holds and $\Delta y^{\mathrm{a},k} \to 0$ on an infinite index set K, then for all j, $\tilde{x}^{\mathrm{a},k}_j s^k_j \to 0$ and $\tilde{x}^{\mathrm{m},k}_j s^k_j \to 0$, both on K. If, in addition, $\{y^k\}$ is bounded, then $y^k \to y^* \in F^{\mathrm{s}}$ and all limit points of the bounded sequences $\{\tilde{x}^{\mathrm{a},k}\}_{k\in K}$ and $\{\tilde{x}^{\mathrm{m},k}\}_{k\in K}$ are multipliers associated to the*

[8] An English version of the proof of [Dik74] can be found in [VL88]; see also [Sai96]. Stewart [Ste89] obtained this result in the form of a bound on the norm of oblique projectors, and provided an independent, geometric proof. O'Leary [O'L90] later proved that Stewart's bound is sharp.

stationary point y^*.[9]

Proof. The first claim is a direct consequence of the second block equations of (2.18) and (2.19), (2.27), and definitions (2.33), and (2.39). Next, we prove asymptotic complementarity of $\{(\tilde{x}^{\mathrm{a},k}, s^k)\}_{k\in K}$, i.e., that $\tilde{x}_i^{\mathrm{a},k} s_i^k \to 0$ on K for all $i \in \mathbf{n}$. Using the third block equation in (2.18) and, again, using (2.33) we have, for all k,

$$\tilde{x}_j^{\mathrm{a},k} s_j^k = -x_j^k \Delta s_j^{\mathrm{a},k},\ j \in Q^k, \tag{2.57}$$

$$\tilde{x}_j^{\mathrm{a},k} s_j^k = 0,\ j \in \mathbf{n} \setminus Q^k. \tag{2.58}$$

Since x^k is bounded (Lemma 2.3.4), and $\Delta s^{\mathrm{a},k} = -A^{\mathrm{T}} \Delta y^{\mathrm{a},k} \to 0$ on K, this implies $\tilde{x}_j^{\mathrm{a},k} s_j^k \to 0$ on K for all j. We also can prove asymptotic complementarity of $\{(\tilde{x}^{\mathrm{m},k}, s^k)\}_{k\in K}$. Equation (2.47) and (2.39) yield, for all k,

$$s_j^k \tilde{x}_j^{\mathrm{m},k} = -x_j^k \Delta s_j^{\mathrm{m},k} + \gamma^k (\sigma^k \mu_{Q^k}^k - \Delta x_j^{\mathrm{a},k} \Delta s_j^{\mathrm{a},k}),\ j \in Q^k, \tag{2.59}$$

$$s_j^k \tilde{x}_j^{\mathrm{m},k} = 0,\ j \in \mathbf{n} \setminus Q^k. \tag{2.60}$$

Boundedness of $\{\tilde{x}^{\mathrm{a},k}\}_{k\in K}$ and $\{x^k\}$ (Lemma 2.3.4) implies boundedness of $\{\Delta x_{Q^k}^{\mathrm{a},k}\}_{k\in K}$ since $\Delta x_{Q^k}^{\mathrm{a},k} = \tilde{x}_{Q^k}^{\mathrm{a},k} - x_{Q^k}^k$. In addition, $\Delta y^{\mathrm{a},k} \to 0$ on K and (2.52) imply that $\gamma^k \Delta y^{\mathrm{c},k} \to 0$ on K and $\gamma^k \sigma^k \mu_{Q^k}^k \to 0$ on K. The former implies in turn that $\gamma^k \Delta s^{\mathrm{c},k} = -\gamma^k A^{\mathrm{T}} \Delta y^{\mathrm{c},k} \to 0$ on K by (2.44). Thus, in view of (2.27), $\{\Delta s^{\mathrm{m},k}\}_{k\in K}$ and the entire right-hand side of (2.59) converge to zero on K. Asymptotic complementarity then follows from boundedness of $\{\tilde{x}^{\mathrm{m},k}\}$ (Lemma 2.3.4). Finally, the last claim follows directly from the above and from Lemma 2.3.3.

□

Recall the definition $\varphi^k := \|\Delta y^{\mathrm{a},k}\|^\nu + \|[\tilde{x}^{\mathrm{a},k}]_-\|^\nu$ from (2.34). The next two lemmas outline some properties of this quantity.

[9]Such "multipliers" are defined below equation (2.1).

Lemma 2.3.6. *Suppose Assumption 1 holds. If* $\{y^k\}$ *is bounded and* $\liminf_{k\to\infty} \varphi^k = 0$*, then* $y^k \to y^* \in F^*$*.*

Proof. By definition (2.34) of φ^k, convergence of φ^k to zero on some infinite index set K implies that $\Delta y^{\mathrm{a},k} \to 0$ and $[\tilde{x}^{\mathrm{a},k}]_- \to 0$ on K. Lemma 2.3.5 and $[\tilde{x}^{\mathrm{a},k}]_- \to 0$ on K thus imply that $\{y^k\}$ converges, and its limit y^* is optimal. □

Lemma 2.3.7. *Suppose Assumptions 1 and 2 hold and* $\{y^k\}$ *is bounded. If* $\Delta y^{\mathrm{a},k} \not\to 0$*, then* $\liminf_{k\to\infty} \varphi^k = 0$*. Specifically, for any infinite index set* K *on which* $\inf_{k\in K} \|\Delta y^{\mathrm{a},k}\| > 0$*, we have* $\varphi^{k-1} \to 0$ *for* $k \in K$ *as* $k \to \infty$*.*

Proof. We proceed by contradiction. Thus, suppose there exists an infinite set $K' \subseteq K$ on which $\|\Delta y^{\mathrm{a},k}\|$ and φ^{k-1} are both bounded away from zero. Let us also suppose, without loss of generality, that Q^k is constant on K', say equal to some fixed Q, and (by the boundedness assumption) $y^k \to y'$ on K', for some $y' \in F$. Lemma 2.2.3 then guarantees that $I(y') \subseteq Q$. We also note that, since the rule for selecting Q ensures that A_Q has full rank and, as per (2.18), $\Delta s^{\mathrm{a},k} = -A_Q^{\mathrm{T}} \Delta y^{\mathrm{a},k}$, we have that $\|\Delta s^{\mathrm{a},k}\|$ is also bounded away from zero on K'. Define $\delta_1 := \inf_{k\in K'} \|\Delta s^{\mathrm{a},k}\|^2 > 0$, and next note that, in view of (2.35), the fact that $\varphi^{k-1} \geq \varepsilon$, for some $\varepsilon > 0$, implies that $\delta_2 := \inf\{x_i^k \mid i \in Q^k,\ k \in K'\} > 0$. We now note that, by Step 5 of rMPC* and Proposition 2.3.1 (ii), for all $k \in K'$,

$$b^{\mathrm{T}} y^{k+1} = b^{\mathrm{T}}(y^k + t_{\mathrm{d}}^{\mathrm{m},k} \Delta y^{\mathrm{m},k}) \geq b^{\mathrm{T}} y^k + t_{\mathrm{d}}^{\mathrm{m},k} \theta b^{\mathrm{T}} \Delta y^{\mathrm{a},k}. \tag{2.61}$$

Also, from (2.49a) and (2.44), we have for all $k \in K'$,

$$\begin{aligned} b^{\mathrm{T}} \Delta y^{\mathrm{a},k} &= (\Delta y^{\mathrm{a},k})^{\mathrm{T}} A_Q (S_Q^k)^{-1} X_Q^k A_Q^{\mathrm{T}} \Delta y^{\mathrm{a},k} \\ &= (\Delta s_Q^{\mathrm{a},k})^{\mathrm{T}} (S_Q^k)^{-1} X_Q^k \Delta s_Q^{\mathrm{a},k} \geq \frac{\delta_2}{R} \delta_1 > 0, \end{aligned} \tag{2.62}$$

where R is an upper bound on $\{\|s^k\|_\infty\}_{k\in K'}$ (notice $\{s^k\}$ converges on K' since

$\{y^k\}$ does). In view of (2.61)-(2.62), establishing a positive lower bound on $t_{\rm d}^{{\rm m},k}$ for $k \in K'$ will contradict boundedness of $\{y^k\}$, thereby completing the proof.

By (2.31) and since Step 4 of Iteration rMPC* ensures (2.9), we have $t_{\rm d}^{{\rm m},k} \geq \beta \bar{t}_{\rm d}^{{\rm m},k} \geq \beta\zeta t_{\rm d}^{{\rm a},k} \geq 0$. Therefore, it suffices to bound $t_{\rm d}^{{\rm a},k}$ away from zero. From (2.24), either $t_{\rm d}^{{\rm a},k} = 1$ or, for some i_0 such that $\Delta s_{i_0}^{{\rm a},k} < 0$, (without loss of generality we assume such i_0 is independent of $k \in K'$) we have

$$t_{\rm d}^{{\rm a},k} = \frac{s_{i_0}^k}{-\Delta s_{i_0}^{{\rm a},k}}. \tag{2.63}$$

If $i_0 \in \mathbf{n} \setminus Q$, then $\{s_{i_0}^k\}_{k\in K'}$ is bounded away from zero (since $I(y') \subseteq Q$). Then, in this case, the desired positive lower bound for $t_{\rm d}^{{\rm a},k}$ follows if we can show that $\Delta s^{{\rm a},k}$ is bounded on K'. To see that the latter holds, we manipulate $S_Q^k \tilde{x}_Q^{{\rm a},k} = -X_Q^k \Delta s_Q^{{\rm a},k}$ (from (2.49c) and (2.33)) and $\Delta s_Q^{{\rm a},k} = -A_Q^{\rm T} \Delta y^{{\rm a},k}$ to write

$$\Delta s^{{\rm a},k} = A^{\rm T}(A_Q A_Q^{\rm T})^{-1} A_Q (X_Q^k)^{-1} S_Q^k \tilde{x}^{{\rm a},k},$$

which is bounded on K' since $\delta_2 > 0$, s^k is bounded, and $\tilde{x}^{{\rm a},k}$ is bounded (by Lemma 2.3.4). On the other hand, if $i_0 \in Q$, using (2.63) and (2.46) we obtain $t_{\rm d}^{{\rm a},k} = x_{i_0}^k / \tilde{x}_{i_0}^{{\rm a},k}$, which is bounded away from zero on K' since x_Q^k is bounded away from zero on K' and $\tilde{x}_Q^{{\rm a},k}$ is bounded by Lemma 2.3.4. This completes the proof. □

Theorem 2.3.8. *Suppose Assumptions 1 and 2 hold. Then, if $\{y^k\}$ is unbounded, $b^{\rm T} y^k \to \infty$. On the other hand, if $\{y^k\}$ is bounded, then $y^k \to y^* \in F^{\rm s}$. Under the further assumption that, at every dual feasible point, the gradients of all active constraints are linearly independent,[10] it holds that if F^* is not empty, $\{y^k\}$ converges to some $y^* \in F^*$, while if F^* is empty, $b^{\rm T} y^k \to \infty$, so that, in both cases, $\{b^{\rm T} y^k\}$ converges to the optimal dual value.*

Proof. The first claim follows from Lemma 2.3.3. Concerning the second claim,

under Assumptions 1 and 2, the hypothesis of either Lemma 2.3.5 or Lemma 2.3.7 must hold: $\{\Delta y^{\mathrm{a},k}\}$ either converges to zero or it does not. In the latter case, the second claim follows from Lemmas 2.3.7 and 2.3.6 since $F^* \subseteq F^{\mathrm{s}}$. In the former case, it follows from Lemma 2.3.5.

To prove the last claim, it is sufficient to show that, under the stated linear independence assumption, it cannot be the case that $\{y^k\}$ converges to some $y^* \in F^{\mathrm{s}} \setminus F^*$. Indeed, the first two claims will then imply that either $y^k \to F^*$, which cannot occur when F^* is empty, or $b^{\mathrm{T}} y^k \to \infty$, which can only occur if F^* is empty, proving the claim. Now, proceeding by contradiction, suppose that $y^k \to y^* \in F^{\mathrm{s}} \setminus F^*$. It then follows from Lemma 2.3.7 that $\Delta y^{\mathrm{a},k} \to 0$, since, with $y^* \notin F^*$, Lemma 2.3.6 implies that $\liminf_{k\to\infty} \varphi^k > 0$. Lemma 2.3.5 then implies that $S^k \tilde{x}^{\mathrm{a},k} \to 0$, and $A\tilde{x}^{\mathrm{a},k} = b$. Define $J := \{j \in \mathbf{n} \mid \tilde{x}_j^{\mathrm{a},k} \not\to 0\}$. Since $S^k \tilde{x}^{\mathrm{a},k} \to 0$, and since $s^k = c - A^{\mathrm{T}} y^k \to c - A^{\mathrm{T}} y^*$, we have that $s_J^k \to 0$, i.e., $J \subseteq I(y^*)$. Thus, by Lemma 2.2.3, $J \subseteq I(y^*) \subseteq Q^k$ holds for all k sufficiently large. Then, using the second block equation of (2.18) and (2.33), we can write

$$b = A_{Q^k} \tilde{x}_{Q^k}^{\mathrm{a},k} = A\tilde{x}^{\mathrm{a},k} = A_J \tilde{x}_J^{\mathrm{a},k} + A_{\mathbf{n}\setminus J} \tilde{x}_{\mathbf{n}\setminus J}^{\mathrm{a},k}, \tag{2.64}$$

where, by definition of J, the second term in the right hand side converges to zero. Under the linear independence assumption, since $J \subseteq I(y^*)$, A_J must have linearly independent columns and a left inverse given by $(A_J^{\mathrm{T}} A_J)^{-1} A_J^{\mathrm{T}}$. Thus, using (2.64), we have $\tilde{x}_J^{\mathrm{a},k} \to (A_J^{\mathrm{T}} A_J)^{-1} A_J^{\mathrm{T}} b$. Define $\tilde{x}^*$ by $\tilde{x}_J^* := (A_J^{\mathrm{T}} A_J)^{-1} A_J^{\mathrm{T}} b$ and $\tilde{x}_{\mathbf{n}\setminus J}^* := 0$, so that $\tilde{x}^{\mathrm{a},k} \to \tilde{x}^*$. Since $y^* \notin F^*$, $\tilde{x}_{j_0}^* < 0$ for some $j_0 \in J$, and $\tilde{x}_{j_0}^{\mathrm{a},k} < 0$ holds for all k sufficiently large, which implies that $s_{j_0}^k \to 0$. However, from (2.46), for all k large enough,

$$\Delta s_{j_0}^{\mathrm{a},k} = -\frac{s_{j_0}}{x_{j_0}} \tilde{x}_{j_0}^{\mathrm{a},k} > 0,$$

so that, by (2.32), $s_{j_0}^{k+1} > s_{j_0}^k > 0$ holds for all k large enough, which contradicts

$s^k_{j_0} \to 0$. □

Whether $y^k \to y^* \in F^*$ is guaranteed (when F^* is nonempty) without the linear independence assumption is an open question.

Remark 2.3.1. *While a fairly standard assumption, the linear independence condition used in Theorem 2.3.8 to prove convergence to a dual optimal point, admittedly, is rather strong, and may be difficult to verify a priori. We remark here that, in view of the monotonic increase of* $b^{\mathrm{T}}y^k$ *and of the finiteness of the set* $\{b^{\mathrm{T}}y : y \in F^{\mathrm{s}} \setminus F^*\}$*, convergence to a dual optimal point should occur without such assumption if the iterates are subject to perturbations, (say, due to roundoff) assumed to be uniformly distributed over a small ball. Indeed, suppose that* y^k *converges to* $y^* \in F^{\mathrm{s}} \setminus F^*$*, say, with limit dual value equal to* v*. There exists* $\alpha > 0$ *such that, for every* k *large enough, the computed* y^k *will satisfy* $b^T y^k > v$ *with probability at least* α*, so that this will happen for* some k *with probability one. Of course, again due to perturbations,* $b^T y^k$ *could drop below* v *again at some later iteration. This however can be addressed by the following simple modification of the algorithm. Whenever the computed* y^{k+1} *satisfies* $b^T y^{k+1} < b^T y^k$*, discard such* y^{k+1} *and compute* $\Delta y^{\mathrm{p}}(y^k, Q)$ *('p' standing for "pacer" step) by solving an appropriate auxiliary problem, such as the small dimension LP*

$$\max\{b^{\mathrm{T}}\Delta y^{\mathrm{p}} \mid A^{\mathrm{T}}_{Q^k}\Delta y^{\mathrm{p}} \leq s_{Q^k} := c_{Q^k} - A^{\mathrm{T}}_{Q^k}y^k,\ \|\Delta y^{\mathrm{p}}\|_\infty \leq 1\}, \tag{2.65}$$

where $Q \in \mathfrak{Q}_{\epsilon,M}(y^k)$*, and redefine* y^{k+1} *to be the point produced by a long step (close to the largest feasible step) taken from* y^k *in direction* $\Delta y^{\mathrm{p}}(y^k, Q)$*. It is readily shown that the solution* $\Delta y^{\mathrm{p}}(y^k, Q)$ *provides a feasible step that gives uniform ascent near any* $y^k \in F^{\mathrm{s}} \setminus F^*$*. Note that, in "normal" operation, typical stopping criteria will be*

[10]This additional assumption is equivalent to the assumption that "all dual basic feasible solutions are nondegenerate" commonly used in convergence analyses of affine scaling and simplex algorithms, see, e.g., [BT97], section 1.8, and Chapter 3 below.

satisfied before any decrease in $\{b^T y^k\}$ *due to roundoff is observed, and the suggested pacer step will never be used.*

Finally, the following convergence properties of the primal sequence can be inferred whenever $\{y^k\}$ converges to $y^* \in F^*$, without further assumptions.

Proposition 2.3.9. *Suppose that Assumptions 1 and 2 hold and that* $y^k \to y^* \in F^*$. *Then, there exists an infinite index set* K *on which* $\Delta y^{\mathrm{a},k} \to 0$ *and* $\{\tilde{x}^{\mathrm{a},k}\}_{k\in K}$ *and* $\{\tilde{x}^{\mathrm{m},k}\}_{k\in K}$ *converge to the primal optimal set.*

Proof. As in the proof of Theorem 2.3.8 we hinge on (overall) convergence of $\{\Delta y^{\mathrm{a},k}\}$ to zero or not. In the former case, Lemma 2.3.5 implies that for any subsequence K' on which $\{\|\Delta y^{\mathrm{a},k}\|\}_{k\in K'}$ is bounded away from zero, the conclusion of this proposition holds on the "previous" subsequence $K = K' - 1$. In the latter case, Lemma 2.3.7 implies that there exists K on which $\varphi^k \to 0$, which implies that $\Delta y^{\mathrm{a},k} \to 0$ on K and that, by Lemma 2.3.5, $\{\tilde{x}^{\mathrm{a},k}\}$ and $\{\tilde{x}^{\mathrm{m},k}\}$ converge to the primal optimal set on K. □

2.4 Local convergence

If $\{y^k\}$ converges to the optimal set (see Remark 2.3.1), under the additional Assumption 3 (stated below), the iteration sequence $z^k := (x^k, y^k)$ converges q-quadratically to the unique primal-dual solution $z^* := (x^*, y^*)$. (Uniqueness of x^* follows from Assumption 3.) The (lengthy) details of the analysis are deferred to the next subsection.

Assumption 3. *The dual solution set is a singleton, i.e.,* $F^* = \{y^*\}$, *and* $\{a_i : i \in I(y^*)\}$ *is a linearly independent set.*

This assumption supersedes Assumption 2. The final assumption is justified by Remark 2.3.1 at the end of the previous section.

Assumption 4. $y^k \to y^*$.

Theorem 2.4.1. *Suppose Assumptions 1, 3 and 4 hold. Then the iteration sequence* $\{z^k\}$ *converges locally q-quadratically, i.e.,* $z^k \to z^*$ *and there exists* $c^* > 0$ *such that, for all* k *large enough, we have*

$$\|z^{k+1} - z^*\| \leq c^* \|z^k - z^*\|^2.$$

Furthermore, for k *large enough, the rank condition* C_2 *in the definition of* $\mathfrak{Q}_{\epsilon,M}(y^k)$ *is automatically satisfied.*

Remark 2.4.1. *The next subsection, containing the proof of Theorem 2.4.1, is rather long and technical, and may be skimmed lightly without loss of continuity.*

2.4.1 Proof of Theorem 2.4.1

The proof of quadratic convergence of $\{z^k\}$ is in two steps: we first show that, under Assumptions 1, 3, and 4, the iteration sequence converges to the unique primal-dual solution, namely $z^k \to z^*$ (Proposition 2.4.5), and then we show that the convergence occurs with a q-quadratic rate eventually.

The first result is a slight extension of [TAW06, Lemma 13].

Lemma 2.4.2. *Under Assumptions 1, 3, and 4, the unique primal-dual solution* (x^*, s^*) *satisfies strict complementary slackness, i.e.,* $x^* + s^* > 0$. *Further, for any* Q *such that* $I(y^*) \subseteq Q$, $J(A_Q, x^*_Q, s^*_Q)$ *and* $J_a(A_Q, x^*_Q, s^*_Q)$ *are nonsingular.*

Proof. Assumption 3 and the Goldman-Tucker theorem, (e.g. see [Wri97, p.28]) imply strict complementary slackness for the pair (x^*, s^*). Assumption 3 also implies that $\{a_i \mid i \in I(y^*)\} = \{a_i \mid x^*_i \neq 0\}$ consists of exactly m linearly independent vectors. Hence, the three conditions for Lemma 2.1.1 are satisfied, and the non-singularity claim follows. □

The following technical lemma, that relates quantities generated by rMPC*, is called upon in Lemmas 2.4.4 and 2.4.11 to show that the damping coefficients $t_{\rm p}^{\rm m}$ and $t_{\rm d}^{\rm m}$ converge to one and, moreover, that they do so fast enough for quadratic convergence of $\{z^k\}$ to take place.

Lemma 2.4.3. *Suppose Assumptions 1, 3, and 4 hold. Let (x, y, s) satisfy $A^{\rm T}y + s = c$, $s > 0$, and $x > 0$ and let $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, and let $\mathcal{A}$ be some index set satisfying $\mathcal{A} \subseteq Q$.*[11] *Let $\Delta x_Q^{\rm a}$, $\Delta s^{\rm a}$, $\tilde{x}^{\rm a}$, $\tilde{s}^{\rm a}$, $\tilde{x}^{\rm m}$, $\tilde{s}^{\rm m}$, $\bar{t}_{\rm p}^{\rm m}$, and $\bar{t}_{\rm d}^{\rm m}$ be generated by Algorithm 10/rMPC*. If $\tilde{x}_i^{\rm m} > 0$ for all $i \in \mathcal{A}$ and $\tilde{s}_i^{\rm m} > 0$ for all $i \in \mathbf{n} \setminus \mathcal{A}$, then*

$$\bar{t}_{\rm p}^{\rm m} \geq \min\left\{1, \min_{i\in Q\setminus\mathcal{A}}\left\{\frac{s_i}{|\tilde{s}_i^{\rm a}|}\right\}, \min_{i\in Q\setminus\mathcal{A}}\left\{\frac{s_i}{\tilde{s}_i^{\rm m}} - \frac{|\Delta s_i^{\rm a}|}{|\tilde{s}_i^{\rm m}|}\right\}\right\}, \tag{2.66}$$

$$\bar{t}_{\rm d}^{\rm m} \geq \min\left\{1, \min_{i\in\mathcal{A}}\left\{\frac{x_i}{|\tilde{x}_i^{\rm a}|}\right\}, \min_{i\in\mathcal{A}}\left\{\frac{x_i}{\tilde{x}_i^{\rm m}} - \frac{|\Delta x_i^{\rm a}|}{|\tilde{x}_i^{\rm m}|}\right\}\right\}. \tag{2.67}$$

Proof. First consider (2.66). With reference to (2.42), we see that either $\bar{t}_{\rm p}^{\rm m} = 1$, in which case (2.66) is verified, or for some $i_0 \in Q$ with $\Delta x_{i_0}^{\rm m} < 0$, we have

$$\bar{t}_{\rm p}^{\rm m} = \frac{x_{i_0}}{-\Delta x_{i_0}^{\rm m}} < 1. \tag{2.68}$$

Suppose $i_0 \in \mathcal{A}(\subseteq Q)$. Since $\Delta x_{i_0}^{\rm m} < 0$ and $x_{i_0} > 0$, in view of the definition (2.39) of $\tilde{x}^{\rm m}$, the inequality $\tilde{x}_{i_0}^{\rm m} = x_{i_0} + \Delta x_{i_0}^{\rm m} > 0$, which holds by assumption, implies $x_{i_0}/(-\Delta x_{i_0}^{\rm m}) > 1$, contradicting (2.68). Thus we must have $i_0 \in Q \setminus \mathcal{A}$. To complete the proof of (2.66), we consider two possibilities. If

$$|\Delta x_{i_0}^{\rm a}| \geq |\Delta x_{i_0}^{\rm m}|,$$

[11] In applications of this lemma we have in mind $\mathcal{A} = I(y^*)$.

then, using (2.46) we have

$$\bar{t}_{\mathrm{p}}^{\mathrm{m}} = \frac{x_{i_0}}{-\Delta x_{i_0}^{\mathrm{m}}} \geq \frac{x_{i_0}}{|\Delta x_{i_0}^{\mathrm{a}}|} = \frac{s_{i_0}}{|\tilde{s}_{i_0}^{\mathrm{a}}|}, \tag{2.69}$$

and (2.66) is again verified. Alternately, if $|\Delta x_{i_0}^{\mathrm{a}}| < |\Delta x_{i_0}^{\mathrm{m}}|$, then using (2.47) and rearranging terms, we get (see below for explanation of the inequalities)

$$\bar{t}_{\mathrm{p}}^{\mathrm{m}} = \frac{x_{i_0}}{-\Delta x_{i_0}^{\mathrm{m}}} = \frac{s_{i_0}}{\tilde{s}_{i_0}^{\mathrm{m}}} + \frac{\gamma\sigma\mu_Q}{-\Delta x_{i_0}^{\mathrm{m}}\tilde{s}_{i_0}^{\mathrm{m}}} + \gamma\frac{\Delta x_{i_0}^{\mathrm{a}}}{\Delta x_{i_0}^{\mathrm{m}}}\frac{\Delta s_{i_0}^{\mathrm{a}}}{\tilde{s}_{i_0}^{\mathrm{m}}} \geq \frac{s_{i_0}}{\tilde{s}_{i_0}^{\mathrm{m}}} - \gamma\frac{|\Delta x_{i_0}^{\mathrm{a}}|}{|\Delta x_{i_0}^{\mathrm{m}}|}\frac{|\Delta s_{i_0}^{\mathrm{a}}|}{|\tilde{s}_{i_0}^{\mathrm{m}}|} \geq \frac{s_{i_0}}{\tilde{s}_{i_0}^{\mathrm{m}}} - \frac{|\Delta s_{i_0}^{\mathrm{a}}|}{|\tilde{s}_{i_0}^{\mathrm{m}}|},$$

where γ, σ, and μ_Q are as generated by rMPC*. The first inequality follows because the second term is nonnegative: the numerator is nonnegative, $-\Delta x_{i_0}^{\mathrm{m}} > 0$ by assumption, and $\tilde{s}_{i_0}^{\mathrm{m}} > 0$ also by assumption. The second inequality follows since $|\Delta x_{i_0}^{\mathrm{a}}| < |\Delta x_{i_0}^{\mathrm{m}}|$ and $\gamma \leq 1$. So, once again, (2.66) is verified. Finally, inequality (2.67) is proved by a very similar argument that flips the roles of x and s. □

Lemma 2.4.4. *Suppose Assumptions 1, 3, and 4 hold. Given any infinite index set K such that $\Delta y^{\mathrm{a},k} \to 0$ on K, it holds that $\hat{x}^k \to x^*$ on K, and $x^{k+1} \to x^*$ on K.*

Proof. Since, by Assumption 4, $y^k \to y^*$, in view of Lemma 2.2.3, we may assume without loss of generality that $I(y^*) \subseteq Q^k$ for all $k \in K$. Now, since $\Delta y^{\mathrm{a},k} \to 0$ on K and $y^k \to y^*$, (2.44) implies that $\Delta s^{\mathrm{a},k} \to 0$ on K, and Lemma 2.3.5 implies that $\tilde{x}^{\mathrm{a},k} \to x^*$ on K and $\tilde{x}^{\mathrm{m},k} \to x^*$ on K, in particular, that $[\tilde{x}^{\mathrm{a},k}]_- \to 0$ on K. Further, by (2.52), and (2.27), $\Delta y^{\mathrm{m},k} \to 0$ on K which implies, again by (2.44), that $\Delta s^{\mathrm{m},k} \to 0$ on K.

With this in hand, we first show that $\|\hat{x}_{Q^k}^k - x_{Q^k}^*\| \to 0$ on K.[12] We have for all k, using the triangle inequality, (2.39), and (2.32),

[12]Note that the dimension of $\hat{x}_{Q^k}^k - x_{Q^k}^*$, i.e., $|Q^k|$, may vary with k.

$$\|\hat{x}^k_{Q^k} - x^*_{Q^k}\| \leq \|\hat{x}^k_{Q^k} - \tilde{x}^{\mathrm{m},k}_{Q^k}\| + \|\tilde{x}^{\mathrm{m},k}_{Q^k} - x^*_{Q^k}\| \leq |1 - t^{\mathrm{m},k}_{\mathrm{p}}| \, \|\Delta x^{\mathrm{m},k}_{Q^k}\| + \|\tilde{x}^{\mathrm{m},k} - x^*\|. \quad (2.70)$$

Since $\tilde{x}^{\mathrm{m},k} \to x^*$ on K and $\{\Delta x^{\mathrm{m},k}_{Q^k}\}_{k\in K}$ is bounded (since, as per Lemma 2.3.4, $\{x^k\}$ and $\{\tilde{x}^{\mathrm{m},k}\}_{k\in K}$ are both bounded), we need only show $t^{\mathrm{m},k}_{\mathrm{p}} \to 1$ on K. Now, $y^k \to y^*$ implies $s^k \to s^*$ and, since $\Delta y^{\mathrm{a},k} \to 0$ on K, it follows from (2.44), (2.27), (2.52), and (2.41) that $\tilde{s}^{\mathrm{m},k} \to s^*$ on K. Next, since $I(y^*) \subseteq Q^k$, and since $\tilde{x}^{\mathrm{m},k} \to x^*$ on K and $\tilde{s}^{\mathrm{m},k} \to s^*$ on K, strict complementarity of (x^*, s^*) (Lemma 2.4.2) implies that, for all $k \in K$ large enough, $\tilde{x}^{\mathrm{m},k}_i > 0$ for $i \in I(y^*)$ and $\tilde{s}^{\mathrm{m},k}_i > 0$ for $i \in \mathbf{n} \setminus I(y^*)$. Without loss of generality, we assume it holds for all $k \in K$. Therefore, the hypothesis of Lemma 2.4.3 is verified, with $\mathcal{A} = I(y^*)$, for all $k \in K$, and in view of (2.66), since $\Delta s^{\mathrm{a},k} \to 0$ on K and $\{s^k\}$, $\{\tilde{s}^{\mathrm{a},k}\}$ and $\{\tilde{s}^{\mathrm{m},k}\}$ all converge to s^* on K, we have $\bar{t}^{\mathrm{m},k}_{\mathrm{p}} \to 1$ on K (since $s^*_i > 0$ for all $i \in \mathbf{n} \setminus I(y^*)$). Further, by (2.30) and since $\Delta y^{\mathrm{a},k} \to 0$ on K, we also have $t^{\mathrm{m},k}_{\mathrm{p}} \to 1$ on K. So indeed, $\|\hat{x}^k_{Q^k} - x^*_{Q^k}\| \to 0$ on K.

Next, we show that $\|x^{k+1}_{Q^k} - x^*_{Q^k}\| \to 0$ on K. First, let $i \in I(y^*)$ ($\subseteq Q^k$ for all $k \in K$). We have already established that $\varphi^k = \|\Delta y^{\mathrm{a},k}\|^\nu + \|[\tilde{x}^{\mathrm{a},k}]_-\|^\nu \to 0$ on K and $\hat{x}^k_i \to x^*_i > 0$ on K (positivity follows from strict complementary slackness). This implies, by (2.35), that for sufficiently large $k \in K$ we have $x^{k+1}_i = \hat{x}^k_i$, so that $x^{k+1}_i \to x^*_i$ on K. Now consider $i \in \mathbf{n} \setminus I(y^*)$, where $x^*_i = 0$, and consider the set $K_i \subseteq K$ defined by $K_i := \{k \in K \mid i \in Q^k\}$. If K_i is finite, then this i is irrelevant to the limit of $\|x^{k+1}_{Q^k} - x^*_{Q^k}\|$. If K_i is infinite however, then since $\varphi^k \to 0$ on K_i and $\hat{x}^k_i \to x^*_i = 0$ on K_i, we have from (2.35) that $x^{k+1}_i \to 0 = x^*_i$ on K_i. Thus we have shown that $\|x^{k+1}_{Q^k} - x^*_{Q^k}\| \to 0$ on K.

Now, let K' be the subset of K on which $\mathbf{n} \setminus Q^k$ is nonempty. If K' is finite, then the proof of the lemma is already complete. Otherwise, to complete the proof, we show that $\|x^{k+1}_{\mathbf{n}\setminus Q^k} - x^*_{\mathbf{n}\setminus Q^k}\| = \|x^{k+1}_{\mathbf{n}\setminus Q^k}\| \to 0$ on K'. For this, we consider $i \in$

$\mathbf{n} \setminus I(y^*)$ and the set $\overline{K}_i \subseteq K'$ defined by $\overline{K}_i := \{k \in K' \mid i \in \mathbf{n} \setminus Q^k\}$. As before, if $\overline{K}_i$ is finite then this index i is irrelevant to the limits we are interested in. If it is infinite, then by (2.38) we get $x_i^{k+1} \leq \mu_{Q^k}^{k+1}/s_i^{k+1}$ on $\overline{K}_i$. Now, since $\|x_{Q^k}^{k+1} - x_{Q^k}^*\| \to 0$ on K, it follows from complementarity of (x^*, s^*), that $\mu_{Q^k}^{k+1} \to 0$ on K. Since $\{s_i^{k+1}\}$ is bounded away from zero (since $i \in \mathbf{n} \setminus I(y^*)$) and $\mu_{Q^k}^{k+1} \to 0$ on K, we have $x_i^{k+1} \to x_i^* = 0$ on $\overline{K}_i$. Thus, the proof is complete. □

Proposition 2.4.5. *Suppose Assumptions 1, 3 and 4 hold. Then we have (i)* $\Delta y^{\mathrm{a},k} \to 0$ *and* $\Delta y^{\mathrm{m},k} \to 0$, *(ii)* $\tilde{x}^{\mathrm{a},k} \to x^*$ *and* $\tilde{x}^{\mathrm{m},k} \to x^*$, *(iii)* $\hat{x}^k \to x^*$ *and* $x^k \to x^*$, *and (iv)* $\|\Delta x_{Q^k}^{\mathrm{a},k}\| \to 0$ *and* $\|\Delta x_{Q^k}^{\mathrm{m},k}\| \to 0$.

Proof. First we show that $\Delta y^{\mathrm{a},k} \to 0$. Supposing it is not so, take an infinite index set K with $\inf_{k\in K} \|\Delta y^{\mathrm{a},k}\| > 0$. Lemma 2.3.7 then implies that there exists an infinite index set $K' \subseteq K$ on which $\{\Delta y^{\mathrm{a},k-1}\}_{k\in K'}$ and $\{[\tilde{x}^{\mathrm{a},k-1}]_-\}_{k\in K'}$ converge to zero (since $\varphi^{k-1} \to 0$ as $k \to \infty$ with $k \in K'$). We assume without loss of generality that $Q^k = Q$, a constant set, for all $k \in K'$ (since Q^k is selected from a finite set). Lemma 2.4.4 implies $x^k \to x^*$ on K' and since $s^k \to s^*$ (on K' in particular), we have $J(A_Q, x_Q^k, s_Q^k) \to J(A_Q, x_Q^*, s_Q^*)$ on K'. Further, by strict complementarity of (x^*, s^*) and Assumption 3, and since $I(y^*) \subseteq Q$, Lemma 2.4.2 implies that $J(A_Q, x_Q^*, s_Q^*)$ is nonsingular. Using these facts and noting that (2.18) and the inclusion $I(y^*) \subseteq Q$ imply

$$J(A_Q, x_Q^k, s_Q^k) \begin{pmatrix} \tilde{x}_Q^{\mathrm{a},k} \\ \Delta y^{\mathrm{a},k} \\ \Delta s_Q^{\mathrm{a},k} \end{pmatrix} = \begin{pmatrix} 0 \\ b \\ 0 \end{pmatrix} \text{ on } K' \quad \text{and} \quad J(A_Q, x_Q^*, s_Q^*) \begin{pmatrix} x_Q^* \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ b \\ 0 \end{pmatrix}, \tag{2.71}$$

we see that $\Delta y^{\mathrm{a},k} \to 0$ on K'. This gives the desired contradiction and proves that the entire sequence $\{\Delta y^{\mathrm{a},k}\}$ converges to zero. In view of (2.52) and definition (2.27)

of Δy^{m}, the proof of claim (i) is complete.

In view of Lemma 2.3.5 and Lemma 2.4.4, claims (ii) and (iii) are immediate consequences of claim (i). Claim (iv) follows directly from claims (ii) and (iii).

□

From here forward, we focus on the $\{z^k\} = \{(x^k, y^k)\}$ sequence. To prove quadratic convergence of $\{z^k\}$ to $z^* = (x^*, y^*)$, we show that there exist constants $c \geq 0$, and $\rho > 0$ (independent of $z = (x, y)$ and Q) such that for all $z \in B(z^*, \rho) \cap G^o$ and all $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,

$$\|z^+(z, Q) - z^*\| \leq c\|z - z^*\|^2. \tag{2.72}$$

Here

$$B(z^*, \rho) := \{z \in \mathbb{R}^{m+n} \mid \|z - z^*\| \leq \rho\},$$

$$G^o := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^m \mid x > 0,\ y \in F^{\mathrm{o}}\},$$

and $z^+(z, Q)$ is the update to z with the dependence of z^+ on z and Q made explicit. We will use this explicit notation for all quantities that depend on (z, Q) from now on, e.g. $\Delta z^{\mathrm{a}}(z, Q)$, $\tilde{x}^{\mathrm{m}}(z, Q)$, etc. Notice that the set of (z, Q) such that $z \in G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ is precisely the domain of definition of the mappings $z^+(\cdot, \cdot)$, $\Delta z^{\mathrm{a}}(\cdot, \cdot)$, etc., defined by Algorithm 10/rMPC*. We also will use the somewhat abusive notation

$$z_Q := (x_Q, y), \quad \Delta z_Q := (\Delta x_Q, \Delta y).$$

The following lemma gives a neighborhood $B(z^*, \rho^*)$ of z^*, for a certain $\rho^* > 0$, on which we will prove that the quadratic rate inequality (2.72) holds for a certain c. In particular, several useful bounds that simplify the remaining analysis are proven on this neighborhood. We first define a quantity that is guaranteed to be positive

under strict complementarity, which holds under Assumption 3:

$$\varepsilon^* := \min\{1,\ \min_{i\in\mathbf{n}}(s_i^* + x_i^*)\} > 0. \tag{2.73}$$

Lemma 2.4.6. *Suppose Assumptions 1, 3 and 4 hold and let $\beta > 0$. Then there exists $\rho^* > 0$ and $R > 0$ such that, for all $z \in B(z^*, \rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, the following hold:*

(i) $$I(y^*) \subseteq Q \text{ and } \|J_{\mathrm{a}}(A_Q, x_Q, s_Q)^{-1}\| \le R, \tag{2.74}$$

(ii) $$\max\{\|\Delta z_Q^{\mathrm{a}}(z,Q)\|,\ \|\Delta z_Q^{\mathrm{m}}(z,Q)\|, \|\Delta s_Q^{\mathrm{a}}(z,Q)\|,\ \|\Delta s_Q^{\mathrm{m}}(z,Q)\|\} < \varepsilon^*/2, \tag{2.75}$$

(iii) $$\min\{x_i, \tilde{x}_i^{\mathrm{a}}(z,Q), \tilde{x}_i^{\mathrm{m}}(z,Q)\} > \varepsilon^*/2,\ \forall\, i \in I(y^*), \tag{2.76}$$

$$\max\{s_i, \tilde{s}_i^{\mathrm{a}}(z,Q), \tilde{s}_i^{\mathrm{m}}(z,Q)\} < \varepsilon^*/2,\ \forall\, i \in I(y^*), \tag{2.77}$$

$$\max\{x_i, \tilde{x}_i^{\mathrm{a}}(z,Q), \tilde{x}_i^{\mathrm{m}}(z,Q)\} < \varepsilon^*/2,\ \forall\, i \in \mathbf{n} \setminus I(y^*), \tag{2.78}$$

$$\min\{s_i, \tilde{s}_i^{\mathrm{a}}(z,Q), \tilde{s}_i^{\mathrm{m}}(z,Q)\} > \varepsilon^*/2,\ \forall\, i \in \mathbf{n} \setminus I(y^*), \tag{2.79}$$

(iv) $$\beta\bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) < \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) - \|\Delta y^{\mathrm{a}}(z,Q)\|, \tag{2.80}$$

$$\beta\bar{t}_{\mathrm{d}}^{\mathrm{m}}(z,Q) < \bar{t}_{\mathrm{d}}^{\mathrm{m}}(z,Q) - \|\Delta y^{\mathrm{a}}(z,Q)\|. \tag{2.81}$$

Proof. Let $s := c - A^{\mathrm{T}}y$. (Note that, through y, s varies with z.) Consider the (finite) set

$$\mathfrak{Q}^* := \{Q \subseteq \mathbf{n} \mid I(y^*) \subseteq Q\}.$$

By Lemma 2.2.3, $\mathfrak{Q}_{\epsilon,M}(y) \subseteq \mathfrak{Q}^*$ for all y sufficiently close to y^*. To prove the lemma, it suffices to show that we can find $\rho_Q > 0$ and $R_Q > 0$ to establish claims (i)-(iv) for any fixed $Q \in \mathfrak{Q}^*$ and all $z \in B(z^*, \rho_Q)$. Indeed, in view of the finiteness of $\mathfrak{Q}^*$,

the claims will then hold for all $Q \in \mathfrak{Q}^*$ and $z \in B(z^*, \rho^*)$ with $\rho^* := \min_{Q \in \mathfrak{Q}^*} \rho_Q$ and $R := \max_{Q \in \mathfrak{Q}^*} R_Q$. Thus, we now fix $Q \in \mathfrak{Q}^*$ and seek appropriate ρ_Q and R_Q, under which the claims can all be validated.

Claim (i) follows from Lemma 2.4.2, since $I(y^*) \subseteq Q$, and continuity of $J_a(A_Q, \cdot, \cdot)$. Claim (ii) follows from claim (i) and nonsingularity of the limit of the matrix in (2.18) and (2.19). Claim (iii) follows from (2.73), complementarity of (x^*, s^*), and (2.75). Finally, in view of claim (iii) and Lemma 2.4.3 (with $\mathcal{A} = I(y^*)$), claim (iv) follows based on the same argument as used in the proof of claim (ii), after reducing ρ_Q if need be.

□

It is well known that, under nondegeneracy assumptions, Newton's method for solving a system of equations enjoys a quadratic local convergence rate. It should not be too surprising then that an algorithm that is "close enough" to being a Newton method also has a quadratic rate. The following result, borrowed from [TZ94, Proposition 3.10] and called upon in [TAW06], gives a convenient sufficient condition for this "close enough" criterion to be met.

Lemma 2.4.7. *Let $\Phi : \mathbb{R}^n \to \mathbb{R}^n$ be twice continuously differentiable and let $z^* \in \mathbb{R}^n$ be such that $\Phi(z^*) = 0$ and $\frac{\partial \Phi}{\partial z}(z^*)$ is nonsingular. Let $\rho > 0$ be such that $\frac{\partial \Phi}{\partial z}(z)$ is nonsingular whenever $z \in B(z^*, \rho)$. Let $d^{\mathrm{N}} : B(z^*, \rho) \to \mathbb{R}^n$ be the Newton increment $d^{\mathrm{N}}(z) := -\left(\frac{\partial \Phi}{\partial z}(z)\right)^{-1} \Phi(z)$. Given any $\alpha_1 > 0$, there exists $\alpha_2 > 0$ such that the following statement holds: For all $z \in B(z^*, \rho)$ and $z^+ \in \mathbb{R}^n$ such that for each $i \in \mathbf{n}$,*

$$\min\{|z_i^+ - z_i^*|,\ |z_i^+ - (z_i + d_i^{\mathrm{N}}(z))|\} \leq \alpha_1 \|d^{\mathrm{N}}(z)\|^2, \tag{2.82}$$

it holds that

$$\|z^+ - z^*\| \leq \alpha_2 \|z - z^*\|^2.$$

This leads to the following simple corollary, whose idea comes from [TAW06, Theorem 17].

Corollary 2.4.8. *Let Φ, $d^{\mathrm{N}}(z)$, z^* and ρ be as in Lemma 2.4.7. Then given any $\alpha_1 > 0$, there exists $\alpha_3 > 0$ such that the following statement holds: For all $z \in B(z^*, \rho)$ and $z^+ \in \mathbb{R}^n$ such that for each $i \in \mathbf{n}$,*

$$\min\{|z_i^+ - z_i^*|,\ |z_i^+ - (z_i + d_i^{\mathrm{N}}(z))|\} \leq \alpha_1 \max\{\|d^{\mathrm{N}}(z)\|^2,\ \|z - z^*\|^2\}, \tag{2.83}$$

it holds that

$$\|z^+ - z^*\| \leq \alpha_3 \|z - z^*\|^2.$$

Proof. Given $\alpha_1 > 0$, suppose $z \in B(z^*, \rho)$ and $z^+ \in \mathbb{R}^n$ are such that (2.83) holds for all $i \in \mathbf{n}$. If $\|z - z^*\| \leq \|d^{\mathrm{N}}(z)\|$, then (2.83) is identical to (2.82) and Lemma 2.4.7 provides an $\alpha_2 > 0$ such that

$$\|z^+ - z^*\| \leq \alpha_2 \|z - z^*\|^2.$$

If instead $\|d^{\mathrm{N}}(z)\| < \|z - z^*\|$ then, from (2.83), for each $i \in \mathbf{n}$ we have either

$$|z_i^+ - z_i^*| \leq \alpha_1 \|z - z^*\|^2 \quad \text{or} \quad |z_i^+ - (z_i + d_i^{\mathrm{N}}(z))| \leq \alpha_1 \|z - z^*\|^2.$$

In the latter case,

$$|z_i^+ - z_i^*| \leq |z_i^+ - (z_i + d_i^{\mathrm{N}}(z))| + |(z_i + d_i^{\mathrm{N}}(z)) - z_i^*| \leq \alpha_1 \|z - z^*\|^2 + \alpha_0 \|z - z^*\|^2,$$

where α_0 is a constant for the quadratic rate of the Newton step on $B(z^*, \rho)$ (e.g., a Lipschitz constant for $\frac{\partial \Phi}{\partial z}$ times an upper bound for $\frac{\partial \Phi}{\partial z}^{-1}$ on $B(z^*, \rho)$). Overall, we

have thus shown that, for all $i \in \mathbf{n}$,

$$|z_i^+ - z_i^*| \leq d\|z - z^*\|^2,$$

with $d = \max\{\alpha_1 + \alpha_0, \alpha_2\}$. Hence the claims hold with $\alpha_3 = \sqrt{n}d$. □

Following [TAW06], we will apply Corollary 2.4.8 to the equality portion of the KKT conditions for (1.3) with the slack variable s eliminated, namely,

$$\Phi(x, y) := \begin{pmatrix} Ax - b \\ X(c - A^{\mathrm{T}}y) \end{pmatrix} = 0. \tag{2.84}$$

This function is twice continuously differentiable, it vanishes at (x^*, y^*), its Jacobian at $z = (x, y)$ is equal to $J_a(A, x, c - A^{\mathrm{T}}y)$ and is nonsingular at z^* by Lemma 2.4.2 and hence near z^* by continuity, and the corresponding Newton step is $\Delta z^{\mathrm{a}}(z, \mathbf{n})$, the *unreduced* affine-scaling step.

Our first task (Lemmas 2.4.9-2.4.12) is to compare the step taken along the rMPC* direction

$$\hat{z}_Q^+(z, Q) := (\hat{x}_Q(z, Q), y^+(z, Q)) \tag{2.85}$$

to the Q components of the Newton/affine-scaling step, $z_Q + \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})$. Verifying condition (2.83) amounts to verifying one of four alternative inequalities for each component of $z_i^+(z, Q)$. We will use all four alternatives. First, for $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, define

$$T^{\mathrm{m}}(z, Q) := \begin{pmatrix} t_{\mathrm{p}}^{\mathrm{m}}(z, Q) I_{|Q|} & 0 \\ 0 & t_{\mathrm{d}}^{\mathrm{m}}(z, Q) I_m \end{pmatrix}.$$

We can then write

$$\hat{z}_Q^+(z, Q) = z_Q + T^{\mathrm{m}}(z, Q)\Delta z_Q^{\mathrm{m}}(z, Q), \tag{2.86}$$

and we note that

$$\|I - T^{\mathrm{m}}(z, Q)\| = |1 - t^{\mathrm{m}}(z, Q)|, \tag{2.87}$$

where

$$t^{\mathrm{m}}(z, Q) := \min\{t_{\mathrm{p}}^{\mathrm{m}}(z, Q),\ t_{\mathrm{d}}^{\mathrm{m}}(z, Q)\}. \tag{2.88}$$

Now we break the comparison of the rMPC* step to the Newton/affine-scaling step into three pieces using the triangle inequality, equations (2.86) and (2.87), and the fact that $\gamma(z, Q) \leq 1$ by definition. We obtain

$$\begin{aligned}
&\|\hat{z}_Q^+(z, Q) - (z_Q + \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\| \\
&\leq \|\hat{z}_Q^+(z, Q) - (z_Q + \Delta z_Q^{\mathrm{m}}(z, Q))\| + \|\Delta z_Q^{\mathrm{m}}(z, Q) - \Delta z_Q^{\mathrm{a}}(z, Q)\| + \|\Delta z_Q^{\mathrm{a}}(z, Q) - \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\| \\
&= \|(I - T^{\mathrm{m}}(z, Q))\Delta z_Q^{\mathrm{m}}(z, Q)\| + \gamma(z, Q)\|\Delta z_Q^{\mathrm{c}}(z, Q)\| + \|\Delta z_Q^{\mathrm{a}}(z, Q) - \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\| \\
&\leq |1 - t^{\mathrm{m}}(z, Q)|\|\Delta z_Q^{\mathrm{m}}(z, Q)\| + \|\Delta z_Q^{\mathrm{c}}(z, Q)\| + \|\Delta z_Q^{\mathrm{a}}(z, Q) - \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\|.
\end{aligned} \tag{2.89}$$

The next three lemmas bound each component of (2.89) in terms of the norm of the affine-scaling direction. The first is a slightly simplified version of [TAW06, Lemma 16] that provides a bound for the last term in (2.89). An almost identical proof as in [TAW06] applies using ρ^* from Lemma 2.4.6, and we refer the reader there for details.

Lemma 2.4.9. *Suppose Assumptions 1, 3 and 4 hold. Then there exists $c_1 > 0$ such that, for all $z \in B(z^*, \rho^*) \cap G^o$, $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,*

$$\|\Delta z_Q^{\mathrm{a}}(z, Q) - \Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\| \leq c_1\|z - z^*\| \cdot \|\Delta z_Q^{\mathrm{a}}(z, \mathbf{n})\|.$$

An immediate consequence of Lemma 2.4.9 is the following inequality, which is used in the proofs of Lemma 2.4.12 and Theorem 2.4.1 below: for all $z \in B(z^*, \rho^*) \cap$

G^o and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ we have

$$\begin{aligned}\|\Delta z_Q^{\mathrm{a}}(z,Q)\| &\leq \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\| + \|\Delta z_Q^{\mathrm{a}}(z,Q) - \Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\| \\ &\leq (1 + c_1\|z - z^*\|)\|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\| \\ &\leq (1 + c_1\rho^*)\|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\|. \end{aligned} \tag{2.90}$$

The next lemma provides a bound for the second term in (2.89).

Lemma 2.4.10. *Suppose Assumptions 1, 3 and 4 hold. Then there exists $c_2 > 0$ such that for all $z \in B(z^*,\rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,*

$$\|\Delta z_Q^{\mathrm{c}}(z,Q)\| \leq c_2\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2. \tag{2.91}$$

Proof. Let $z \in B(z^*,\rho^*)\cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$. Using (2.50) and the uniform bound R on $\|J_{\mathrm{a}}(A_Q,x_Q,s_Q)^{-1}\|$ from (2.74), we have

$$\begin{aligned}\|\Delta z_Q^{\mathrm{c}}(z,Q)\| &\leq \|J_{\mathrm{a}}(A_Q,x_Q,s_Q)^{-1}\| \left\| \begin{pmatrix} 0 \\ \sigma(z,Q)\mu_Q(z,Q)e - \Delta X_Q^{\mathrm{a}}(z,Q)\Delta s_Q^{\mathrm{a}}(z,Q) \end{pmatrix} \right\| \\ &\leq R\left(\sqrt{n}|\sigma(z,Q)\mu_Q(z,Q)| + \|\Delta X_Q^{\mathrm{a}}(z,Q)\Delta s_Q^{\mathrm{a}}(z,Q)\|\right). \end{aligned} \tag{2.92}$$

Further, in view of (2.44), we have

$$\begin{aligned}\|\Delta X_Q^{\mathrm{a}}(z,Q)\Delta s_Q^{\mathrm{a}}(z,Q)\| &\leq \|\Delta X_Q^{\mathrm{a}}(z,Q)\|\|\Delta s_Q^{\mathrm{a}}(z,Q)\| \\ &\leq \|\Delta x_Q^{\mathrm{a}}(z,Q)\|\|\Delta s_Q^{\mathrm{a}}(z,Q)\| \\ &\leq \|A\|\ \|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2. \end{aligned}$$

Next, we note that $\mu_Q(z,Q) = (x_Q)^{\mathrm{T}}(s_Q)/|Q|$ is bounded on $B(z^*,\rho)\cap G^o$ (by Cauchy-Schwartz and since (2.74) gives $|Q| \geq m$). Thus, to handle the first term in

(2.92) and hence to prove the lemma, it suffices to show that

$$|\sigma(z,Q)| \le d\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2, \tag{2.93}$$

for some d independent of z and Q. Step 2 of rMPC* sets $\sigma = (1-t^{\mathrm{a}}(z,Q))^\lambda$, with $t^{\mathrm{a}}(z,Q)$ as defined in Step 1 of Iteration rMPC*. As usual, when bounding the damping coefficients t_{p}^a and t_{d}^a, there are two very similar arguments to be made for the primal and dual steps. We first bound $t_{\mathrm{d}}^{\mathrm{a}}(z,Q)$ as expressed in (2.43). By Lemma 2.4.6 ((2.75) and (2.79)), we have

$$\frac{s_i}{|\Delta s_i^{\mathrm{a}}(z,Q)|} > \frac{\varepsilon^*/2}{\varepsilon^*/2} = 1 \text{ for all } i \in \mathbf{n} \setminus I(y^*), \tag{2.94}$$

so that, in view of (2.43), either $t_{\mathrm{d}}^{\mathrm{a}}(z,Q) = 1$, in which case $|1 - t_{\mathrm{d}}^{\mathrm{a}}(z,Q)| = 0$, or using equation (2.46), since $I(y^*) \subseteq Q$ (by (2.74)),

$$t_{\mathrm{d}}^{\mathrm{a}}(z,Q) = \frac{s_i}{-\Delta s_i^{\mathrm{a}}(z,Q)} = \frac{x_i}{\tilde{x}_i^{\mathrm{a}}(z,Q)} \text{ for some } i \in I(y^*) \tag{2.95}$$

(here i depends on (z,Q)). In the latter case, using (2.33) and assertion (2.76) of Lemma 2.4.6, we have

$$|1 - t_{\mathrm{d}}^{\mathrm{a}}(z,Q)| = \left|1 - \frac{x_i}{\tilde{x}_i^{\mathrm{a}}(z,Q)}\right| = \left|\frac{\Delta x_i^{\mathrm{a}}(z,Q)}{\tilde{x}_i^{\mathrm{a}}(z,Q)}\right| \le \frac{2}{\varepsilon^*}\|\Delta z_Q^{\mathrm{a}}(z,Q)\|. \tag{2.96}$$

A very similar argument gives

$$|1 - t_{\mathrm{p}}^{\mathrm{a}}(z,Q)| \le \frac{2}{\varepsilon^*}|\Delta s_i^{\mathrm{a}}(z,Q)| \le \frac{2}{\varepsilon^*}\|A\|\|\Delta z_Q^{\mathrm{a}}(z,Q)\|.$$

Since $t^{\mathrm{a}}(z,Q) = \min\{t_{\mathrm{p}}^{\mathrm{a}}(z,Q),\ t_{\mathrm{d}}^{\mathrm{a}}(z,Q)\}$, we get

$$|1 - t^{\mathrm{a}}(z,Q)| \le \frac{2}{\varepsilon^*}\max\{\|A\|, 1\}\|\Delta z_Q^{\mathrm{a}}(z,Q)\|,$$

and, since $\lambda \geq 2$, (2.93) holds with $d = (2/\varepsilon^*)^\lambda \max\{\|A\|^\lambda, 1\}$. This completes the proof.

□

A direct implication of Lemma 2.4.10 (and Lemma 2.4.6 (ii)), which will be used in the proof of Lemmas 2.4.11 and 2.4.12, is that, for all $z \in B(z^*, \rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,

$$\begin{aligned}\|\Delta z_Q^{\mathrm{m}}(z,Q)\| &\leq \|\Delta z_Q^{\mathrm{a}}(z,Q)\| + \|\Delta z_Q^{\mathrm{c}}(z,Q)\| \leq \|\Delta z_Q^{\mathrm{a}}(z,Q)\| + c_2\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2 \\ &\leq \left(1 + c_2\frac{\varepsilon^*}{2}\right)\|\Delta z_Q^{\mathrm{a}}(z,Q)\|. \end{aligned} \tag{2.97}$$

The following lemma provides a bound for the first term in (2.89).

Lemma 2.4.11. *Suppose Assumptions 1, 3 and 4 hold. Then there exists $c_3 > 0$ such that, for all $z \in B(z^*, \rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,*

$$|1 - t^{\mathrm{m}}(z,Q)| \leq c_3\|\Delta z_Q^{\mathrm{a}}(z,Q)\|. \tag{2.98}$$

Proof. Let $z \in B(z^*, \rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$. We first show that

$$|1 - t_{\mathrm{p}}^{\mathrm{m}}(z,Q)| \leq d_1\|\Delta z_Q^{\mathrm{a}}(z,Q)\|,$$

for some d_1 independent of z and Q. Assertions (2.74), (2.76) and (2.79) in Lemma 2.4.6 imply respectively that $I(y^*) \subseteq Q$, $\tilde{x}_i^{\mathrm{m}}(z,Q) > 0$ for $i \in I(y^*)$, and $\tilde{s}_i^{\mathrm{m}}(z,Q) > 0$ for $i \in \mathbf{n} \setminus I(y^*)$. Thus we may apply assertion (2.66) of Lemma 2.4.3 (with $\mathcal{A} = I(y^*)$) to get

$$\begin{aligned}1 - \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) &\leq \max\left\{0, \max_{i\in Q\setminus I(y^*)}\left\{1 - \frac{s_i}{\tilde{s}_i^{\mathrm{a}}(z,Q)}\right\}, \max_{i\in Q\setminus I(y^*)}\left\{1 - \frac{s_i}{\tilde{s}_i^{\mathrm{m}}(z,Q)} + \frac{|\Delta s_i^{\mathrm{a}}(z,Q)|}{\tilde{s}_i^{\mathrm{m}}(z,Q)}\right\}\right\} \\ &\leq \max\left\{\max_{i\in Q\setminus I(y^*)}\left\{\frac{|\Delta s_i^{\mathrm{a}}(z,Q)|}{\tilde{s}_i^{\mathrm{a}}(z,Q)}\right\}, \max_{i\in Q\setminus I(y^*)}\left\{\frac{|\Delta s_i^{\mathrm{m}}(z,Q)|}{\tilde{s}_i^{\mathrm{m}}(z,Q)} + \frac{|\Delta s_i^{\mathrm{a}}(z,Q)|}{\tilde{s}_i^{\mathrm{m}}(z,Q)}\right\}\right\}. \end{aligned}$$

Further, (2.79), (2.44), and (2.97) yield

$$\begin{aligned} 1 - \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) &\leq \frac{2}{\varepsilon^*} \max_{i \in Q \setminus I(y^*)} \{|\Delta s_i^{\mathrm{m}}(z,Q)| + |\Delta s_i^{\mathrm{a}}(z,Q)|\} \\ &\leq \frac{2}{\varepsilon^*}\|A\| \left(\|\Delta y^{\mathrm{m}}(z,Q)\| + \|\Delta y^{\mathrm{a}}(z,Q)\|\right) \\ &\leq \frac{2}{\varepsilon^*}\|A\| \left(2 + c_2\frac{\varepsilon^*}{2}\right) \|\Delta z_Q^{\mathrm{a}}(z,Q)\|. \end{aligned}$$

Finally, by assertion (2.80) of Lemma 2.4.6, we have $\beta \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) < \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) - \|\Delta y^{\mathrm{a}}(z,Q)\|$, so that by (2.30) $t_{\mathrm{p}}^{\mathrm{m}}(z,Q) = \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) - \|\Delta y^{\mathrm{a}}(z,Q)\|$, and

$$1 - t_{\mathrm{p}}^{\mathrm{m}}(z,Q) = 1 - \bar{t}_{\mathrm{p}}^{\mathrm{m}}(z,Q) + \|\Delta y^{\mathrm{a}}(z,Q)\| \leq d_1 \|\Delta z_Q^{\mathrm{a}}(z,Q)\|,$$

with $d_1 := \frac{2}{\varepsilon^*}\|A\|(2 + c_2\varepsilon^*/2) + 1$. By a similar argument that essentially flips the roles of $\mathbf{n} \setminus I(y^*)$ and $I(y^*)$ and the roles of x and s, we get

$$|1 - t_{\mathrm{d}}^{\mathrm{m}}(z,Q)| \leq d_2 \|\Delta z_Q^{\mathrm{a}}(z,Q)\|,$$

with $d_2 := \frac{2}{\varepsilon^*}(2 + c_2\varepsilon^*/2) + 1$. Since $t^{\mathrm{m}}(z,Q) = \min\{t_{\mathrm{p}}^{\mathrm{m}}(z,Q),\ t_{\mathrm{d}}^{\mathrm{m}}(z,Q)\}$, the claim follows with $c_3 := \max\{d_1, d_2\}$. □

The final lemma applies the previous three lemmas to inequality (2.89) to bound the difference between the Q component of our step (2.85) and the Newton step.

Lemma 2.4.12. *Suppose Assumptions 1, 3 and 4 hold. Then there exists $c_4 > 0$ such that, for all $z \in B(z^*, \rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,*

$$\|\hat{z}_Q^+(z,Q) - (z_Q + \Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\| \leq c_4 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\|^2\}. \tag{2.99}$$

Proof. Applying Lemmas 2.4.9, 2.4.10, and 2.4.11 to (2.89) and using definition

(2.85) of $\hat{z}_Q^+(z,Q)$, we get, for all $z \in B(z^*,\rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$,

$$\begin{aligned}
&\|\hat{z}_Q^+(z,Q) - (z_Q + \Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\| \\
&\leq c_3\|\Delta z_Q^{\mathrm{a}}(z,Q)\|\|\Delta z_Q^{\mathrm{m}}(z,Q)\| + c_2\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2 + c_1\|z-z^*\|\|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\| \\
&\leq d\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2 + c_2\|\Delta z_Q^{\mathrm{a}}(z,Q)\|^2 + c_1\|z-z^*\|\|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\|,
\end{aligned}$$

with $d := c_3(1 + c_2\varepsilon^*/2)$ using (2.97). In view of (2.90), the result follows. □

Proof of Theorem 2.4.1. We show that there exists $c^* > 0$ such that, for all $z \in B(z^*,\rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, we have

$$\|z^+(z,Q) - z^*\| \leq c^*\|z-z^*\|^2,$$

where ρ^* is as in Lemma 2.4.6. The claim of quadratic convergence of $\{z^k\}$ will then follow, since, by Assumption 3 and Proposition 2.4.5 (iii), we know that $z^k \to z^*$. First, let us fix an arbitrary $z \in B(z^*,\rho^*) \cap G^o$ and $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, and show that, for each $i \in \mathbf{n}$,

$$\min\{|z_i^+(z,Q) - z_i^*|,\ |z_i^+(z,Q) - (z_i + \Delta z_i^{\mathrm{a}}(z,\mathbf{n}))|\} \leq \alpha_1 \max\{\|\Delta z^{\mathrm{a}}(z,\mathbf{n})\|^2,\ \|z-z^*\|^2\}. \tag{2.100}$$

In view of Corollary 2.4.8, the claim will then follow. Now, in view of Lemma 2.4.12 and definition (2.85) of $\hat{z}^+(z,Q)$, condition (2.100) holds for the $y^+(z,Q)$ components of $z^+(z,Q)$. It remains to verify condition (2.100) for the $x^+(z,Q)$ components of $z^+(z,Q)$.[13]

[13]Note that the bound provided by Lemma 2.4.12 involves the components of $\hat{x}_Q(z,Q)$, while we seek to bound the components of $x_Q^+(z,Q)$.

First let $i \in I(y^*)$ ($\subseteq Q$, by Lemma 2.4.6 (i)). Note that (explanation follows)

$$\|[\tilde{x}^{\mathrm{a}}(z,Q)]_-\|^\nu + \|\Delta y^{\mathrm{a}}(z,Q)\|^\nu < \|\Delta x^{\mathrm{a}}_Q(z,Q)\|^\nu + \|\Delta y^{\mathrm{a}}(z,Q)\|^\nu < 2\left(\frac{\varepsilon^*}{2}\right)^\nu \leq \frac{\varepsilon^*}{2}.$$

The first inequality uses the fact that (since $x > 0$)

$$\|[\tilde{x}^{\mathrm{a}}(z,Q)]_-\| = \|[x_Q + \Delta x^{\mathrm{a}}_Q(z,Q)]_-\| < \|\Delta x^{\mathrm{a}}_Q(z,Q)\|, \tag{2.101}$$

the second inequality uses Lemma 2.4.6 (ii), and the third uses the bounds $\nu \geq 2$ and $\varepsilon^* \leq 1$ (see definition (2.73) of ε^*). On the other hand, by assertion (2.76) of Lemma 2.4.6, the definitions (2.32) of $\hat{x}_i(z,Q)$ and (2.39) of $\tilde{x}^{\mathrm{m}}_i(z,Q)$, we have

$$\frac{\varepsilon^*}{2} < \min\{x_i, \tilde{x}^{\mathrm{m}}_i(z,Q)\} \leq \hat{x}_i(z,Q).$$

Putting these together, we obtain

$$\|[\tilde{x}^{\mathrm{a}}(z,Q)]_-\|^\nu + \|\Delta y^{\mathrm{a}}(z,Q)\|^\nu < \hat{x}_i(z,Q),$$

so that, by (2.35), $x^+_i(z,Q) = \hat{x}_i(z,Q)$ for $i \in I(y^*)(\subseteq Q)$ and hence, in view of Lemma 2.4.12, condition (2.100) also holds for the corresponding components of $z^+(z,Q)$.

Next, consider the components $x^+_i(z,Q)$ with $i \in Q \setminus I(y^*)$. We proceed to establish the inequality

$$\|x^+_{Q\setminus I(y^*)}(z,Q)\| \leq d_2 \max\{\|\Delta z^{\mathrm{a}}_Q(z,\mathbf{n}))\|^2, \|z - z^*\|^2\} \tag{2.102}$$

which, besides establishing (2.100) for the $x^+_{Q\setminus I(y^*)}(z,Q)$ component of $z^+(z,Q)$ (since $x^*_i = 0$ for $i \in Q \setminus I(y^*)$), also serves to help establish (2.100) for the $x^+_{\mathbf{n}\setminus Q}(z,Q)$ component of $z^+(z,Q)$. Thus, let $i \in Q \setminus I(y^*)$. Either we again have $x^+_i(z,Q) =$

$\hat{x}_i(z,Q)$, or we have $x_i^+(z,Q) = \min\{\underline{\xi}^{\max},\ \|[\tilde{x}^{\rm a}(z,Q)]_-\|^\nu + \|\Delta y^{\rm a}(z,Q)\|^\nu\}$. In the former case, we have (explanation follows), for some d_1 independent of z and Q,

$$\begin{aligned}
|x_i^+(z,Q)| = |\hat{x}_i(z,Q)| &= |\hat{x}_i(z,Q) - x_i^*| \\
&\le |\hat{x}_i^+(z,Q) - (x_i + \Delta x_i^{\rm a}(z,\mathbf{n}))| + |(x_i + \Delta x_i^{\rm a}(z,\mathbf{n})) - x_i^*| \\
&\le \|\hat{z}_Q^+(z,Q) - (z_Q + \Delta z_Q^{\rm a}(z,\mathbf{n}))\| + \|(z + \Delta z^{\rm a}(z,\mathbf{n})) - z^*\| \\
&\le c_4 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\rm a}(z,\mathbf{n}))\|^2\} + d_1\|z - z^*\|^2.
\end{aligned}$$

The first inequality is just the triangle inequality, the second is clear, and the third uses Lemma 2.4.12 and the quadratic rate of the Newton step on $B(z^*,\rho^*)$, with ρ^* as in Lemma 2.4.6. In the latter case,

$$\begin{aligned}
|x_i^+(z,Q)| &\le \|[\tilde{x}^{\rm a}(z,Q)]_-\|^\nu + \|\Delta y^{\rm a}(z,Q)\|^\nu \\
&\le \|\Delta x_Q^{\rm a}(z,Q)\|^2 + \|\Delta y^{\rm a}(z,Q)\|^2 = \|\Delta z_Q^{\rm a}(z,Q)\|^2 \\
&\le (1 + c_1\rho^*)^2\|\Delta z_Q^{\rm a}(z,\mathbf{n})\|^2.
\end{aligned}$$

The second inequality uses (2.101), $\|\Delta z_Q^{\rm a}(z,Q)\| \le 1$ (by Lemma 2.4.6 (ii) and the definition (2.73) of ε^*), and $\nu \ge 2$, and the third uses (2.90). So we have established (2.102), and (2.100) follows for the $x^+_{Q\setminus I(y^*)}(z,Q)$ component of $z^+(z,Q)$.

Finally, let $i \in \mathbf{n} \setminus Q$ ($\subseteq \mathbf{n} \setminus I(y^*)$, by Lemma 2.4.6 (i)). We again have $x_i^* = 0$ and using (2.38), we get

$$|x_i^+(z,Q) - x^*| = |x_i^+(z,Q)| \le \mu_Q^+(z,Q)(s_i^+(z,Q))^{-1}. \tag{2.103}$$

By the definitions (2.32) and (2.41) of $s_i^+(z,Q)$ and $\tilde{s}_i^{\rm m}(z,Q)$, and assertion (2.79) of Lemma 2.4.6, we have $s_i^+(z,Q) = s_i + t_{\rm d}^{\rm m}\Delta s_i^{\rm m}(z,Q) \ge \min\{s_i, \tilde{s}_i^{\rm m}(z,Q)\} > \varepsilon^*/2$. Using this fact together with the definition (2.36) of $\mu_Q^+(z,Q)$ and the fact that

$|Q| \geq m$, we may further bound (2.103) as

$$|x_i^+(z,Q) - x^*| \leq \frac{2}{m\varepsilon^*} \left(\sum_{i \in I(y^*)} x_i^+(z,Q) s_i^+(z,Q) + \sum_{i \in Q \setminus I(y^*)} x_i^+(z,Q) s_i^+(z,Q) \right). \tag{2.104}$$

Using boundedness of $B(z^*, \rho^*)$, and Lemma 2.4.6 (ii), to bound $|x_i^+(z,Q)|$ for $i \in I(y^*)$ and $|s_i^+(z,Q)|$ for $i \in \mathbf{n} \setminus I(y^*)$, and then using norm equivalence, we get, for some d_3 and d_4 independent of z and Q,

$$|x_i^+(z,Q) - x^*| \leq d_3 \|s_{I(y^*)}^+(z,Q)\| + d_4 \|x_{Q \setminus I(y^*)}^+(z,Q)\|.$$

Continuing, we have (explanation follows), for some $d_5 - d_8$ all independent of z and Q,

$$\begin{aligned}
|x_i^+(z,Q) - x^*| \leq & d_3 \|s_{I(y^*)}^+(z,Q) - s_{I(y^*)}^*\| + d_5 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\|^2\} \\
\leq & d_6 \|\hat{z}_Q^+(z,Q) - z_Q^*\| + d_5 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\|^2\} \\
\leq & d_6 \|\hat{z}_Q^+(z,Q) - (z_Q + \Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\| + d_6 \|(z_Q + \Delta z_Q^{\mathrm{a}}(z,\mathbf{n})) - z_Q^*\| \\
& + d_5 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\|^2\} \\
\leq & d_7 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n})\|^2\} + d_8 \|z - z^*\|^2 \\
& + d_5 \max\{\|z - z^*\|^2, \|\Delta z_Q^{\mathrm{a}}(z,\mathbf{n}))\|^2\}.
\end{aligned}$$

The first inequality uses the bound (2.102) and the fact that $s_{I(y^*)}^* = 0$. The second uses (2.44), and the third uses the triangle inequality. The final inequality uses Lemma 2.4.12 to bound the first term and the quadratic rate of the Newton step on $B(z^*, \rho^*)$ to bound the second term.

Thus condition (2.100) is verified for all components of $z^+(z,Q)$ and the first claim of Theorem 2.4.1 follows, i.e., we have thus shown that the sequence $\{z^k\} =$

$\{(x^k, y^k)\}$ constructed by Algorithm 10/rMPC* converges q-quadratically to (x^*, y^*). The second claim of Theorem 2.4.1, that for all k large enough, the rank condition C_2 in the definition of $\mathfrak{Q}_{\epsilon,M}(y^k)$ is automatically satisfied, follows from Lemma 2.4.2 and Lemma 2.4.6 (i).

Chapter 3

Rank-degeneracy in constraint-reduced IPMs for LP

The notion of "degeneracy" in linear programming usually refers to the existence of degenerate *basic feasible solutions* (see Section 1.3.1), that is, basic feasible solutions with too many active inequalities. The existence of degenerate basic feasible solutions can cause simplex methods to stall or fail (by "cycling") and can also be problematic for some interior-point methods: nondegeneracy assumptions are used in the analysis of the dual affine-scaling algorithm outlined in section 1.8, that of algorithm rPDAS of [TAW06], and in the proof of the second part of Theorem 2.3.8 of this dissertation.

The present chapter focuses on a different type of linear programming degeneracy, namely, when the constraint matrix A is rank deficient. In such case, we say the constraint matrix A (and the LP) is *rank-degenerate*. (We will expand the meaning of this term in what follows.) When A is rank-degenerate, the feasible region of the standard-form dual problem has no extreme points and, if the problem is feasible, optimality can occur only on an entire face (and the corresponding primal basic feasible solution is degenerate in the sense of the first paragraph). Furthermore, if the cost vector has a nonzero component along $\mathcal{N}(A^{\mathrm{T}})$, the dual problem, if feasible,

will be unbounded, and the primal will be infeasible. Even when the problem has a solution, this situation is problematic for many linear programming algorithms. For example, the standard search directions used in interior-point methods become undefined. Generally, rank-degeneracy is dealt with by preprocessing the problem to remove dependent rows of A or by using some type of regularization.

In the context of constraint-reduction, a type of "artificial" rank-degeneracy can arise even when A itself has full rank, since a simple choice of working constraint set Q may result in an A_Q matrix that is not full rank, i.e., the gradients of the working constraint do not span $\mathbb{R}^m$. (Henceforth, in what might be considered an abuse of terminology we drop the prefix "artificial" when talking about this type of rank-degeneracy.) When this happens, the search directions are no longer well defined. We could make the assumption that precludes the situation, e.g., we can assume that every $m \times m$ sub-matrix of A has full rank. This assumption is used by Dantzig and Ye [DY91] in the build-up DAS algorithm, and by Tits et al. in the rPDAS algorithm of [TAW06], but such assumption is unlikely to hold, in general, in real world problems, and furthermore, may be impossible to verify a priori. In other prior work, including that of Chapter 2 of this dissertation, the assumption is imposed on Q rather than A. That is, the constraint selection rule requires that Q be chosen so $\text{rank}(A_Q) = m$. It is possible to explicitly enforce this, for example, with a Gram-Schmidt orthogonalization procedure, or by updating a rank-revealing pivoted Cholesky factorization, or by schemes that simply add more constraints to Q when $\text{rank}(A_Q) < m$. However, these methods may require more effort than we would like to invest, and may result in large $|Q|$, which is what we aimed to avoid in the first place.

In this chapter, we investigate various ways of efficiently dealing with $\text{rank}(A_Q) < m$. We provide a discussion of a few simple approaches that essentially sidestep the issue, and then present and analyze two algorithms that address it directly.

3.1 Introduction

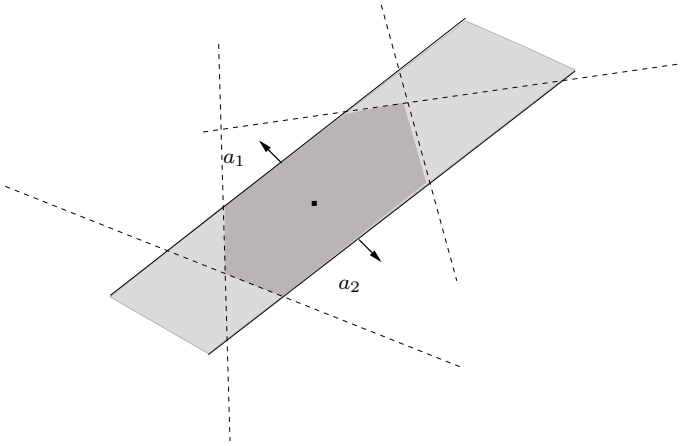

Figure 3.1: Example of artificial degeneracy in two dimensions. From the current iterate (the dot), the two nearest constraints are selected as the working constraints. They do not span $\mathbb{R}^m$, and hence $\text{rank}(A_Q) = 1 < m = 2$.

As in previous chapters, given a working set of constraints Q and a dual-feasible point (x, y, s), we consider computing a PDIP direction for the "reduced" primal-dual pair using (2.16). This leaves the search direction in the $\mathbf{n} \setminus Q$ components of Δx and Δs undefined. We put aside this issue for the time being. Instead, we focus on the possibility that, for arbitrary choice of Q, the coefficient matrix in (2.18) may be singular. This happens when the set of vectors $\{a_i\}_{i \in Q}$ defining the constraints in the dual problem do not span $\mathbb{R}^m$; see Figure 3.1.

3.1.1 Avoiding the issue through assumptions

In [TAW06](and [DY91]), the authors avoid this problem by making the assumption that the constraint matrix A has a special property we will call *M-rank-nondegeneracy*. A constraint matrix is *M-rank-nondegenerate* if every $m \times M$ submatrix has rank m, where $M \geq m$. Likewise, we say A is *M-rank-degenerate* if it is not M-rank-nondegenerate, and if unspecified, we assume $M = m$. Unfortunately,

rank-nondegeneracy is a property that fails to hold for most constraint matrices appearing in "real world" LPs. In particular, suppose $M = m$, then an (m-)rank-nondegenerate constraint matrix can have at most $m - 1$ zeros out of n entries in any row and thus, at most $(m - 1)m$ zeros total. This means, at least if $n \gg m$, that bona-fide sparse problems *are* rank-degenerate. Of course, many non-sparse problems will also be rank-degenerate.

In Chapter 2, we instead require that the choice of the constraint set Q satisfies $\text{rank}(A_Q) = m$. There are some classes of problems where it is relatively simple to satisfy this rank condition. For example, in the work of [KY93], applying Tone's method to transportation problems, including a minimal-spanning-tree of the underlying graph in Q ensures $\text{rank}(A_Q) = m$. However, in general, enforcing the rank condition complicates the selection of Q and greatly restricts the set of admissible Q.

3.1.2 Regularization of the linear systems

Even without constraint-reduction, i.e., when $Q = \mathbf{n}$, there is still the question of whether the Newton-KKT system is nonsingular and numerically well-conditioned. This will not be the case if the rows of A are dependent or nearly so, or if the diagonal matrix $X^{-1}S$ is poorly-conditioned.

A common way to control the condition number of the Newton-KKT systems, without assumption on A, is through regularizing the "KKT" or "augmented" system

$$\begin{pmatrix} -X^{-1}S & A^{\mathrm{T}} \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}}y - \sigma\mu X^{-1}e \\ b - Ax \end{pmatrix}, \tag{3.1}$$

by addition of diagonal factors δI and ρI, namely,

$$\begin{pmatrix} -(X^{-1}S + \rho I) & A^{\mathrm{T}} \\ A & \delta I \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}} y - \sigma\mu X^{-1} e \\ b - Ax \end{pmatrix}, \tag{3.2}$$

see, e.g. [ST96]. Note that, here, the right-hand-side is unmodified; sometimes different right-hand-sides are used, as we will see below. The corresponding normal equations have normal matrix

$$W := (A(X^{-1}S + \rho I)^{-1}A^{\mathrm{T}} + \delta I).$$

If $(x, s) > 0$ then, $\delta > 0$ ensures the normal matrix is nonsingular without assumptions on the rank of A. Furthermore, as shown in, e.g., [ST96], if in addition $\rho > 0$, then the normal matrix has bounded condition number. Indeed,

$$\|W\| \leq \|A\|^2 \|(X^{-1}S + \rho I)^{-1}\| + \delta \leq \frac{\|A\|^2}{\rho} + \delta$$

and

$$\|W^{-1}\| = \lambda_{\max}(W^{-1}) = \frac{1}{\lambda_{\min}(W)} \leq \frac{1}{\delta}$$

so that

$$\kappa(W) = \|W\|\|W^{-1}\| \leq \frac{\|A\|}{\rho\delta} + 1.$$

In the context of constraint-reduction, whether or not $\mathrm{rank}(A) = m$, it may be advantageous to allow choices of Q that result in $\mathrm{rank}(A_Q) < m$. In such cases, regularizations similar to the above may be used to ensure solvability of the reduced PDIP systems. One way to motivate the regularization (3.2) is to think of it as the Newton-KKT step for a perturbed version of the standard linear programming problem (as discussed below). From this point-of-view, when the regularization parameters are set to fixed positive values, we may end up solving the perturbed

problem, and questions regarding the relevance of this solution to original problem arise. For small enough values of the regularization parameters, the solution of this perturbed problem will also be a solution for the original problem [MR79]. In this case the regularization is sometimes called "exact"[FO07]. If assumptions are made that ensure eventual solvability of the PDIP systems, e.g., $\text{rank}(A_Q) = m$ holds near the solution, then it may be reasonable to use such regularization in early iterations, but allow δ and ρ to go to zero in the limit. This is of particular interest in the context of constraint-reduction where the problem may be well-posed, but we introduce the "degeneracy" by ignoring constraints during the iteration. In the rest of this chapter we focus on the δ-regularization only, as this is all that is immediately needed to give us solvable linear systems. Investigation of the use of the ρ-regularization in our algorithms is left as future work.

3.1.3 Sources of the regularization

One simple way to regularize the problem is to add bound constraints to the dual problem in (1.3) of the form

$$-Re \leq y \leq Re \quad \text{or equivalently,} \quad \|y\|_\infty \leq R.$$

We could then base a constraint-reduced algorithm on taking PDIP-type steps for

$$\begin{aligned} &\max\ b^{\mathrm{T}}y \\ &\text{s.t. } A_Q^{\mathrm{T}}y \leq c_Q, \\ &\|y\|_\infty \leq R. \end{aligned} \tag{3.3}$$

If we define $A_0 = [I, -I] \in \mathbb{R}^{m\times 2m}$, $c_0 = Re \in \mathbb{R}^{2m}$, and redefine $\tilde{A} := [A_0, A]$, $\tilde{c} := [c_0; c]$, then we get (3.3) in dual standard form with data $(\tilde{A}, b, \tilde{c})$. If we always

keep the bound constraints in Q, i.e., $\{1, 2, \ldots 2m\} \subset Q$, then $\tilde{A}_Q$ will always be full rank[1] and the convergence analysis put forth in [TAW06] and Chapter 2 can be applied. For such an algorithm to find solutions to the original problem, clearly R must be set large enough so that the "box" $\|y\|_\infty \leq R$ contains a dual solution. If we do not know an a priori bound on a dual solution, we can guess something reasonably large and solve the problem; if some of the bound constraints are active at the solution, we can increase R and try again. Alternatively, we could adjust R "on-the-fly" during the iteration, increasing it if we seem to be terminating with one of the box constraints active.

This method turns out to introduce something similar to the δ-type regularization (discussed in section 3.1.2) for the components $(\Delta x, \Delta y)$. Consider the unreduced case, and partition $\tilde{A} = [A_0, A]$, with $A_0 = [I_m, -I_m]$. The KKT system is then (with $\sigma = 0$ for simplicity)

$$\begin{pmatrix} -X_0^{-1}S_0 & 0 & A_0^{\mathrm{T}} \\ 0 & -X^{-1}S & A^{\mathrm{T}} \\ A_0 & A & 0 \end{pmatrix} \begin{pmatrix} \Delta x_0 \\ \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c_0 - A_0^{\mathrm{T}} y \\ c - A^{\mathrm{T}} y \\ b - Ax - A_0 x_0 \end{pmatrix}.$$

Upon eliminating Δx_0, the component corresponding to the $2m$ bound constraints, this becomes

$$\begin{pmatrix} -X^{-1}S & A^{\mathrm{T}} \\ A & D \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}} y \\ b - Ax \end{pmatrix} \tag{3.4}$$

where $D := A_0 S_0^{-1} X_0 A_0^{\mathrm{T}} = S_0^{-1} X_0 \in \mathbb{R}^{2m \times 2m}$ (since $A_0 = [I_m, -I_m]$), which is similar to the regularization in equation (3.2), except $\rho = 0$ and δI is replaced by

[1] Actually, only one-sided bound constraints are needed to ensure A_Q is full rank, but it seems reasonable to use two-sided constraints since they guarantee existence of a dual solution when the dual is feasible. In either case, keeping such constraints in Q entails very little cost, e.g., in forming (1.83), since their gradients are extremely sparse.

$D = S_0^{-1} X_0$. If the bounds are inactive at the solution, then, for $i \in \{1, 2, \ldots 2m\}$, x_i will converge to zero and s_i will converge to a finite positive value so that the regularization effectively vanishes near the solution.

Alternatively, we could consider centering the box constraints at the current iterate, rather than at the origin. This idea leads to a "trust-region" type algorithm that, at iteration k, takes Newton-KKT steps on the problem

$$\begin{aligned} &\max \ b^{\mathrm{T}} y \\ &\text{s.t. } A^{\mathrm{T}} y \leq c, \\ &\|y - y^k\|_\infty \leq R. \end{aligned}$$

In this case, it may make sense to consider smaller R, which could be adjusted from iteration to iteration, as is typical in many trust-region methods. The "regularized" KKT matrix (i.e., that remaining after eliminating the component of Δx corresponding to the trust region constraint) for this problem is identical to (3.4), although the linear system has a slightly different right-hand side. Alternatively, we could consider a Euclidean-norm trust-region

$$\begin{aligned} &\max \ b^{\mathrm{T}} y \\ &\text{s.t. } A^{\mathrm{T}} y \leq c, \\ &\|y - y^k\|_2 \leq R. \end{aligned} \tag{3.5}$$

The optimality conditions for (3.5) are

$$
\begin{aligned}
Ax + \delta(y - y^k) &= b, \\
A^{\mathrm{T}}y + s &= c, \\
\frac{1}{2}|y - y^k|^2 + \gamma &= \frac{1}{2}R^2, \\
Xs = 0,\ \delta\gamma &= 0, \\
(x, s) \geq 0,\ (\delta, \gamma) &\geq 0
\end{aligned}
\tag{3.6}
$$

where s and γ are slack variables, x is vector of Lagrange multipliers for the constraints $A^{\mathrm{T}}y \leq c$, and δ is the multiplier for the trust-region constraint. After eliminating all but the (x, y) components, the Newton-KKT step for (3.5) is

$$
\begin{pmatrix} -X^{-1}S & A^{\mathrm{T}} \\ A & \delta I \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}}y \\ b - Ax \end{pmatrix},
$$

which is of the form (3.2) with ρ=0. The same linear system can also be arrived at by considering the quadratic program

$$
\begin{aligned}
&\max b^{\mathrm{T}}y - \tfrac{\delta}{2}\|y - y^k\|^2 \\
&\text{s.t. } A^{\mathrm{T}}y \leq c,
\end{aligned}
\tag{3.7}
$$

whose optimality conditions and Newton-KKT step are

$$
\begin{aligned}
A^{\mathrm{T}}y + s &= c, \\
Ax + \delta y &= b, \\
Xs &= 0, \\
(x, s) &\geq 0,
\end{aligned}
$$

and

$$\begin{pmatrix} -X^{-1}S & A^{\mathrm{T}} \\ A & \delta I \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} c - A^{\mathrm{T}} y \\ b - Ax \end{pmatrix}. \tag{3.8}$$

It is this last approach that we will use to motivate our first algorithm, as it lets the δ regularization in (3.2) enter in a simple and clean way.

3.2 Two algorithms

In this section, we describe and analyze two dual-feasible, constraint-reduced, PDIP algorithms for LP designed to deal with the possibility that $\mathrm{rank}(A_Q) < m$. In its kth iteration, the first algorithm selects a value for the regularization parameter δ^k and computes the Newton-KKT direction for the perturbed LP

$$\begin{aligned} &\max b^{\mathrm{T}} y - \tfrac{\delta^k}{2}\|y - y^k\|^2 \\ &\text{s.t. } A^{\mathrm{T}} y \leq c \end{aligned} \tag{3.9}$$

The second algorithm is a variant of one originally developed by Tits et al. in [TAO06]. This algorithm essentially takes the rPDAS step from [TAW06] whenever $b \in R(A_Q)$, which of course is always the case if $\mathrm{rank}(A_Q) = m$, but uses a different update whenever b has a component in the nullspace of A_Q^{T} which we refer to as a "kernel step". As discussed below in section 3.2.2, it turns out that the second algorithm can be naturally motivated (and efficiently implemented) as a limiting case of the first algorithm for vanishing δ.

The following assumption will be in force throughout the analysis of these algorithms.

Assumption 1. *A has full rank.*

Assumption 2. *The primal and dual interior regions are nonempty.*

We also assume for simplicity that $b \neq 0$, although such assumption can be easily removed.

In selecting the constraint set Q, both algorithms use relaxed versions of rule 2.2.1.

Rule 3.2.1. *At a dual feasible point y, select Q arbitrarily from the set $\mathfrak{Q}_{\epsilon,M}(y)$ defined next.*[2]

Definition 3.2.1. *Let $\epsilon \in (0, \infty]$, and let $M \in \{1, 2, \ldots n\}$ be a strict upper bound on the number of constraints active at any dual feasible point. Then a set $Q \subseteq \mathbf{n}$ belongs to $\mathfrak{Q}_{\epsilon,M}(y)$ if and only if Q contains all ϵ-active constraints at y among some set of M most-active constraints.*

This is just Definition 2.2.1 minus the rank condition C_2, and now with M required to be a *strict* upper bound on the number of active constraints at any dual feasible point. See the discussion given after statement of Rule 2.2.1 and Definition 2.2.1, which applies here as well (except that, now, $M = n$ will not work for LPs with n constraints active at some dual feasible point).

3.2.1 Regularized rPDAS

Now we describe and analyze our first algorithm we call the regularized rPDAS, which is based on the regularization (3.7)-(3.8). That is, we use constraint-reduced primal-dual affine-scaling (rPDAS) type steps, i.e., with $\sigma = 0$. (We discuss the adaptive use of positive σ parameters below in section 3.3.)

From a dual feasible interior iterate (x, y, s), the basic search direction is defined by

$$\begin{pmatrix} -X_Q^{-1} S_Q & A_Q^{\mathrm{T}} \\ A_Q & \delta I \end{pmatrix} \begin{pmatrix} \Delta x_Q \\ \Delta y \end{pmatrix} = \begin{pmatrix} c_Q - A_Q^{\mathrm{T}} y \\ b - A_Q x_Q \end{pmatrix}, \tag{3.10}$$

[2]Note this is a redefinition of the symbol $\mathfrak{Q}_{\epsilon,M}(y)$ first defined in Chapter 2.

and

$$\Delta s := -A^{\mathrm{T}}\Delta y, \tag{3.11}$$

which is needed to preserve dual feasibility. The normal equations are

$$(A_Q X_Q S_Q^{-1} A_Q^{\mathrm{T}} + \delta I)\Delta y = b, \tag{3.12}$$

$$\Delta x_Q = -x_Q + X_Q S_Q^{-1} A_Q^{\mathrm{T}} \Delta y. \tag{3.13}$$

When $(x_Q, s_Q) > 0$ and $\delta > 0$, these equations have unique solutions. We also define

$$\tilde{x}_i := \begin{cases} x_i + \Delta x_i & i \in Q, \\ 0 & i \in \mathbf{n} \setminus Q. \end{cases} \tag{3.14}$$

The dual variables (y, s) are updated to (y^+, s^+), via

$$\begin{aligned} y^+ &:= y + t_{\mathrm{d}}\Delta y, \\ s^+ &:= s + t_{\mathrm{d}}\Delta s = (c - A^{\mathrm{T}} y^+), \end{aligned} \tag{3.15}$$

with

$$t_{\mathrm{d}} = \max\{\beta \bar{t}_{\mathrm{d}}, \bar{t}_{\mathrm{d}} - \|\Delta y\|\}, \tag{3.16}$$

$$\bar{t}_{\mathrm{d}} = \min\left\{1, \ \min\left\{\frac{s_i}{-\Delta s_i} \mid \Delta s_i < 0\right\}\right\}. \tag{3.17}$$

The primal update x^+, adapted from Chapter 2, is given by

$$x_i^+ = \begin{cases} \max\{\min\{\varphi, \underline{\xi}^{\max}\}, \ x_i + t_{\mathrm{p}}\Delta x_i\} & i \in Q \\ \min\left\{\frac{\mu_Q^+}{s_i}, \ \overline{\chi}\right\} & i \in \mathbf{n} \setminus Q \end{cases} \tag{3.18}$$

with $\underline{\xi}^{\max} > 0$ and $\overline{\chi} > 0$ (resp. small and large) algorithm parameters and

$$t_{\rm p} = \max\{\beta \bar{t}_{\rm p}, \bar{t}_{\rm p} - \|\Delta y\|\}, \tag{3.19}$$

$$\bar{t}_{\rm p} = \min\left\{1,\ \min\left\{\frac{x_i}{-\Delta x_i} \mid \Delta x_i < 0, i \in Q\right\}\right\}, \tag{3.20}$$

$$\mu_Q^+ := \frac{(x_Q^+)^{\rm T} s_Q^+}{|Q|}, \tag{3.21}$$

$$\varphi := \{\|[\tilde{x}]_-\|^2 + \|\Delta y\|^2\}, \tag{3.22}$$

and for a vector u, $[u\,]_- := \min\{0, u\}$, the minimum taken componentwise. Finally we update the regularization parameter as

$$\delta^+ = \min\{\varphi, \bar{\delta}\}, \tag{3.23}$$

where $\bar{\delta}$ is a small constant (e.g., 10^{-3}).[3]

The above sequence of steps are well defined and can be repeated indefinitely as will be shown in Lemma 3.2.3 below. This justifies attaching an iteration superscript k to the variables to get the following algorithm.

[3]The relationship $\delta^+ \le \varphi \le \min x_i^+,\ \forall\, i \in Q$ (when $\varphi < \underline{\xi}^{\max}$), from (3.18) and (3.23), turns out to be useful in local analysis.

Algorithm 11: Regularized rPDAS

Input: LP data: A, b, c; Initial iterate: $y^0, s^0 = c - A^{\mathrm{T}}y^0 > 0$, $x^0 > 0$;

Parameters: $M \in \mathbf{n}$, $\epsilon > 0$, $\bar{\delta} > 0$, $\underline{\xi}^{\max} > 0$, $\overline{\chi} > 0$, $\beta \in (0, 1)$;

Set $k = 0$, $\delta^0 = \bar{\delta}$, choose Q^0 obeying Rule 3.2.1;

while *forever* **do**

Solve (3.10) for $(\Delta x^k_{Q^k}, \Delta y^k)$.

Update dual variables using (3.15)-(3.17).

Compute $\tilde{x}^k$ and φ^k from (3.14) and (3.22).

Update Q component of primal variables using (3.18)-(3.20).

Compute $\mu^+_{Q^k}$ by (3.21), and update $x^{k+1}_{\mathbf{n}\setminus Q}$ using (3.18)-(3.20).

Set $\delta^{k+1} = \min\{\bar{\delta}, \varphi^k\}$ and choose Q^{k+1} according to Rule 3.2.1.

Set $k = k + 1$.

end

This algorithm is similar to the rPDAS algorithm of [TAW06], and especially to the rMPC* algorithm of Chapter 2: it is a regularized version of rMPC* without the centering-corrector step, i.e., with $\psi = 0$ (see footnote 1 on page 76). Its primary benefit over rPDAS of [TAW06] and rMPC*, is that with $\delta > 0$, we do not need to impose the additional condition on the selection of Q that $\mathrm{rank}(A_Q) = m$, or impose the restrictive nondegeneracy assumption of [TAW06], discussed in section 3.1.1.

Global convergence

Lemma 3.2.2. *Let Δy be as constructed by Algorithm 11. Then $\Delta y \neq 0$ and $b^T \Delta y > 0$.*

Proof. Both claims follow from our assumption that $b \neq 0$ and from the fact that the regularized normal matrix in (3.12) is positive definite when $\delta > 0$. □

The next lemma shows that, under our assumptions, Algorithm 11 is well defined,

and can be repeated *ad infinitum*.

Lemma 3.2.3. *Suppose Assumptions 1 and 2 hold. Then Algorithm 11 is well defined, and constructs quantities with the properties that* $t_{\mathrm{d}} \in (0, \infty)$*,* $y^+ \in F^{\mathrm{o}}$*,* $s^+ = c - A^{\mathrm{T}} y^+ > 0$*, and* $x^+ > 0$*.*

Proof. First, since $s > 0$, it always holds that $\bar{t}_{\mathrm{d}} > 0$. Since $b^{\mathrm{T}} \Delta y > 0$ (Lemma 3.2.2), it follows from non-emptiness of the dual solution set (which is implied by Assumptions 1 and 2), that the largest feasible dual step size $\bar{t}_d$ must be finite. Thus from (3.17), we have $\bar{t}_{\mathrm{d}} \in (0, \infty)$ and thus $t_{\mathrm{d}} \in (0, \infty)$. In view of (3.15)-(3.17), it is always the case that $s^+ = c - A^{\mathrm{T}} y^+ > 0$ and hence $y^+ \in F^o$. Finally, (3.18), (3.22), and the fact that $\Delta y \neq 0$ (Lemma 3.2.2) imply that $x_i^+ > 0$ for $i \in Q$, which, together with the facts that $(x, s) > 0$, $s^+ > 0$, and $\overline{\chi} > 0$, give $x_i^+ > 0$ for $i \in \mathbf{n} \setminus Q$. □

Lemma 3.2.4. *Suppose Assumptions 1 and 2 hold. Then* $\{b^{\mathrm{T}} y^k\}$ *is a strictly increasing sequence and* $\{y^k\}$ *is bounded.*

Proof. That $\{b^{\mathrm{T}} y^k\}$ increases follows from Lemma 3.2.2 and the fact that $t_{\mathrm{d}}^k > 0$ for all k (Lemma 3.2.3). This monotonicity implies that $\{y^k\}$ is confined to the set $\mathcal{S} := \{y \in F | b^{\mathrm{T}} y \geq b^{\mathrm{T}} y^0\}$. When maximizing a concave function over a convex set, boundedness of one super-level set implies boundedness of all others, and our assumptions imply that the dual solution set is nonempty and bounded [Wri97, Thm. 2.3]. Thus, $\mathcal{S}$ is bounded. □

A consequence of monotonicity is that every limit point of $\{y^k\}$ (at least one exists since the sequence is bounded) has the same objective value. Hence, either all limit points are optimal, and by boundedness, $\{y^k\}$ converges the dual optimal set, or the entire sequence stays bounded away from the dual optimal set.

The sequence of primal variables generated by Algorithm 11 is also bounded.

Lemma 3.2.5. *Suppose Assumptions 1 and 2 hold. Then $\{x^k\}$ and $\{\tilde{x}^k\}$ are bounded.*

Proof. Defining $D_Q^k := X_Q^k (S_Q^k)^{-1}$ (we suppress the superscript k on the subscript Q to reduce clutter), from (3.12), (3.13) and (3.14) we have

$$\tilde{x}_Q^k = -D_Q^k A_Q^{\mathrm{T}} (A_Q D_Q A_Q^{\mathrm{T}} + \delta^k I)^{-1} b.$$

To prove that $\{\tilde{x}_Q^k\}$ is bounded, we show that

$$\| D_Q^k A_Q^{\mathrm{T}} (A_Q D_Q^k A_Q^{\mathrm{T}} + \delta^k I)^{-1} \|$$

is bounded by a constant independent of $\delta^k > 0$, $Q^k \in \mathfrak{Q}_{\epsilon,M}(y)$, and diagonal $D_Q^k > 0$. Define

$$\hat{A}_Q := \begin{pmatrix} A_Q & I \end{pmatrix} \quad \text{and} \quad \hat{D}_Q^k := \begin{pmatrix} D_Q^k & 0 \\ 0 & \delta^k I \end{pmatrix},$$

then $\hat{A}_Q$ is full rank and $\hat{D}_Q^k > 0$ for any $\delta^k > 0$, thus we have

$$\left\| \begin{pmatrix} D_Q^k A_Q^{\mathrm{T}} (A_Q D_Q^k A_Q^{\mathrm{T}} + \delta^k I)^{-1} \\ \delta^k (A_Q D_Q^k A_Q^{\mathrm{T}} + \delta^k I)^{-1} \end{pmatrix} \right\| = \left\| \hat{D}_Q^k \hat{A}_Q^{\mathrm{T}} (\hat{A}_Q \hat{D}_Q^k \hat{A}_Q^{\mathrm{T}})^{-1} \right\|.$$

Matrices of the form $DA^{\mathrm{T}}(ADA^{\mathrm{T}})^{-1}$, with A full rank and $D > 0$ diagonal are what Stewart calls scaled pseudo-inverses in the paper [Ste89], where he proves that such matrices have bounded norm independent of D. Calling upon this result we have the bound

$$\left\| \hat{D}_Q^k \hat{A}_Q^{\mathrm{T}} (\hat{A}_Q \hat{D}_Q^k \hat{A}_Q^{\mathrm{T}})^{-1} \right\| \leq R,$$

with R independent of Q^k, D_Q^k and δ^k (note there are only finitely many choices of

Q^k). In particular, we have

$$\|\tilde{x}_Q^k\| = \|D_Q^k A_Q^{\mathrm{T}}(A_Q D_Q^k A_Q^{\mathrm{T}} + \delta^k I)^{-1} b\| \leq R\|b\| =: R'.$$

This establishes boundedness for the $\{\tilde{x}^k\}$ sequence since $\tilde{x}^k_{\mathbf{n}\setminus Q} = 0$ by (3.13). In particular, we have $\|\tilde{x}^k\|_\infty \leq R'$. Suppose, without loss of generality, that $R' > \max\{\|x^k_{Q^k}\|_\infty,\ \underline{\xi}^{\max}, \overline{\chi}\}$ for some k. Then, since $x^k_{Q^k} + t^k_{\mathrm{d}} \Delta x^k_{Q^k}$ is in the convex hull of $\{x^k_{Q^k}, \tilde{x}^k_{Q^k}\}$, we have by (3.18) that $\|x^{k+1}_{Q^k}\|_\infty \leq R'$. Finally, for $i \in \mathbf{n} \setminus Q^k$, x_i^{k+1} is explicitly bounded above by $\overline{\chi}$. The result then follows by induction. □

The global convergence analysis follows that of Chapter 2. Here we hinge on two possibilities: either φ^k stays bounded away from zero, or for some infinite index set K, $\varphi^k \xrightarrow{K} 0$.[4] The next lemma is analogous to Lemma 2.3.5.

Lemma 3.2.6. *Suppose Assumptions 1 and 2 hold and $\Delta y^k \to 0$ on an infinite index set K, then $\tilde{X}^k s^k \xrightarrow{K} 0$ and $A\tilde{x}^k \xrightarrow{K} b$. In particular, all limit points (y', x') of the bounded sequence $\{(y^k, \tilde{x}^k)\}_{k\in K}$, have $y' \in F^{\mathrm{s}}$ with x' an associated multiplier.*

Proof. Since $\{\delta^k\}$ is bounded, the claim that $A\tilde{x}^k \xrightarrow{K} b$ follows from the second block equations of (3.10) and definition (3.14), since they imply

$$A_Q \tilde{x}^k + \delta^k \Delta y^k = b.$$

Next, we prove asymptotic complementarity of $\{(\tilde{x}^k, s^k)\}_{k\in K}$. Using the first block equation in (3.10) and, again, using (3.14) we have, for all k,

$$\tilde{x}^k_j s^k_j = -x^k_j \Delta s^k_j,\ j \in Q^k, \tag{3.24}$$

$$\tilde{x}^k_j s^k_j = 0,\ j \in \mathbf{n} \setminus Q^k. \tag{3.25}$$

[4] In Chapter 2 we instead used the alternatives that $\Delta y \to 0$ or not. The reader can check that this amounts to essentially the same argument.

Since x^k is bounded (Lemma 3.2.5), and $\Delta s^k = -A^{\mathrm{T}}\Delta y^k \to 0$ on K, this implies $\tilde{x}_j^k s_j^k \to 0$ on K for all j. $\square$

The quantity $\varphi^k := \{\|[\tilde{x}^k]_-\|^2 + \|\Delta y^k\|^2\}$, appearing in the primal update (3.18), can be viewed as an indicator of convergence to the optimal set. If $\varphi^k \xrightarrow{K} 0$, then any limit point (y', x') of $\{(\tilde{x}^k, y^k)\}_{k\in K}$ is primal-dual optimal since Lemma 3.2.6 shows it is stationary, while $\varphi^k \xrightarrow{K} 0$ implies the multiplier x' is nonnegative. Thus, in such case, in view of monotonicity of the dual objective, the entire sequence $\{y^k\}$ must converge to the dual optimal set. We have established the following lemma.

Lemma 3.2.7. *Suppose Assumptions 1 and 2 hold. If $\varphi^k \xrightarrow{K} 0$ for some infinite index set K, then $y^k \to F^*$.*

Thus if $y^k \not\to F^*$ then, since $\liminf_{k\to\infty} \varphi^k > 0$ and $\varphi^k \neq 0$, we must have, for some $\varepsilon > 0$, $\varphi^k \geq \varepsilon$ for all k. This, as is shown next, forces $\Delta y^k \to 0$ so that $y^k \to F^s$ by Lemma 3.2.6.

Lemma 3.2.8. *Suppose Assumptions 1 and 2 hold. Then $\varphi^k \geq \varepsilon > 0$ implies $\Delta y^k \to 0$.*

Proof. Suppose $\varphi^k \geq \varepsilon > 0$ and $\Delta y^k \not\to 0$. From the update equation (3.18), $\varphi^k \geq \varepsilon$ for all k implies $x_i^{k+1} \geq \varepsilon$ for all i and k (i.e., $x_i^k \geq \varepsilon$ for all i and $k \geq 1$), and from (3.23) that $\delta^k \geq \min\{\bar{\delta}, \varepsilon\} =: \varepsilon' > 0$. Now, since $\Delta y^k \not\to 0$ and $\{y^k\}$ is bounded, there exists a $\beta > 0$ and an infinite index set K on which $y^k \xrightarrow{K} y'$, for some y', and $\forall\ k \in K$, $\|\Delta y^k\| \geq \beta$ and $Q^k = Q$ constant (since $Q^k \in 2^{\mathbf{n}}$, a finite set). Note also that we must have $I(y') \subseteq Q$ by Rule 3.2.1 and Definition 3.2.1. We then have that for each $k \in K$,

$$M(Q, \delta^k) := (A_Q (S_Q^k)^{-1} X_Q^k A_Q^{\mathrm{T}} + \delta^k I) \succeq \varepsilon' I \succ 0,$$

and therefore

$$b^{\mathrm{T}}\Delta y^k = (\Delta y^k)^{\mathrm{T}} M(Q,\delta^k)\Delta y^k \geq \varepsilon' \|\Delta y^k\|^2 \geq \varepsilon'\beta^2.$$

Next we seek a lower bound for t_d. In view of (3.16), it suffices to lower bound $\bar{t}_{\mathrm{d}}$. Using (3.17), we have $\bar{t}_{\mathrm{d}}^k = 1$ or $\bar{t}_{\mathrm{d}}^k = \frac{s_{i(k)}}{-\Delta s_{i(k)}^k}$ for some $\Delta s_{i(k)}^k < 0$ and index $i(k)$ depending on k. Suppose there is an infinite index set $K' \subseteq K$ such that $i(k) \in I(y')$ for all $k \in K'$. Then, since $I(y') \subseteq Q$, we have, using the first block equation of (3.10), for all $k \in K$ and some $\gamma > 0$,

$$\bar{t}_{\mathrm{d}}^k := \frac{s_{i(k)}^k}{-\Delta s_{i(k)}^k} = \frac{x_{i(k)}^k}{\tilde{x}_{i(k)}^k} \geq \gamma > 0,$$

where we used $x_{i(k)}^k \geq \varepsilon$ and the fact that $\{\tilde{x}^k\}_{k\in K}$ is bounded (Lemma 3.2.5). On the other hand, if there is no such K', then there must be an infinite index set $K'' \subseteq K$, on which $i(k) \in \mathbf{n} \setminus I(y')$, in which case, since $s_{i(k)}^k$ is bounded away from zero on K'', and Δs^k is bounded, we have, for some $\gamma' > 0$, that $\bar{t}_{\mathrm{d}}^k \geq \gamma' > 0$ on K''. (To see that Δs^k is bounded, we use

$$\Delta s^k = -A^{\mathrm{T}}\Delta y^k = -A^{\mathrm{T}} M(Q,\delta^k)^{-1} b,$$

and note that the right-hand-side is bounded, since $0 \prec M(Q,\delta^k)^{-1} \preceq (\varepsilon')^{-1} I$. In either case, the dual objective, which is monotonically increasing overall, increases by a constant amount on an infinite index set, which is impossible, since $\{y^k\}$ is bounded by Lemma 3.2.4. Thus, we must have $\Delta y^k \to 0$. □

Taken together, the previous two lemmas imply that either $y^k \to F^*$, or $\varphi^k \geq \varepsilon > 0$ and thus $\Delta y^k \to 0$, so that every limit point of y^k is stationary.

Lemma 3.2.9. *Suppose Assumptions 1 and 2 hold. Then $\{y^k\}$ converges to the set of stationary points F^{s}.*

As was the case for Theorem 2.3.8, if we invoke a linear independence assumption, convergence to the optimal set can be guaranteed. Specifically, we can use the argument of [TAW06, Lm. 11], followed by a similar argument used in Theorem 2.3.8 (see also [TAW06, Th. 12]). Whether or not this assumption is truly necessary to get convergence to F^* is still an open question; this uncertainty is part of the reason we have held out invoking the assumption until this last step. Furthermore, a similar statement to that made in Remark 2.3.1 also applies here.

Theorem 3.2.10. *Suppose Assumptions 1 and 2 hold. Under the further assumption that, at every dual feasible point, the gradients of all active constraints are linearly independent, it holds that* $y^k \to F^*$.

Quadratic local convergence

A similar local convergence analysis analysis to that in [TAW06] and Chapter 2 applies here. To carry out this analysis we need an additional, rather strong, assumption, albeit an assumption that is typically needed to prove quadratic convergence of Newton methods.

Assumption 3. *The dual solution set is a singleton, i.e.,* $F^* = \{y^*\}$, $y^k \to y^*$, *and the set* $\{a_i \mid c_i = a_i^{\mathrm{T}} y^*\}$ *is linearly independent.*[5]

An obvious consequence of this assumption is that $s^k \to s^* := c - A^{\mathrm{T}} y^*$.

Assumption 3 gives us everything we need to prove quadratic convergence. It implies that the optimal multiplier x^* associated to y^* is unique, that strict complementarity holds,[6] i.e., that

$$x_i^* > 0 \quad \forall i \in I(y^*), \tag{3.26}$$

[5]Note this Assumption just combines Assumptions 3 and 4 used in the section 2.4 local convergence analysis of Algorithm rMPC* .

[6]This follows from Assumption 3 because solvable linear programs always have at least one strictly complementary solution [Wri97, p.28].

and further that

$$\text{span}\{a_i : i \in I(y^*)\} = \mathbb{R}^m. \tag{3.27}$$

Since $y^k \to y^*$, Rule 3.2.1 implies that $I(y^*) \subseteq Q^k$ for all k large enough. Assumption 3 also implies that $A^k_{Q^k}$ has full rank for all k large enough, so that we may allow the regularization parameter δ^k to go to zero.[7] The next lemma gives a few further implications of Assumption 3, including the fact that, indeed, $\delta^k \to 0$. The proof of this lemma uses very similar (actually simpler, since we have no centering-corrector component in this algorithm) arguments to those used in Lemmas 2.4.3-2.4.6, so we omit it.

Lemma 3.2.11. *Suppose Assumptions 1, 2, and 3 hold. Then* $\Delta y^k \to 0$, $\tilde{x}^k \to x^*$, $x^k \to x^*$, $\Delta x^k \to 0$, $\varphi^k \to 0$, $\delta^{k+1} \to 0$, $t^k_{\text{d}} \to 1$, *and* $t^k_{\text{p}} \to 1$.

Now that we know $\delta^k \to 0$, since rMPC* with $\psi = 0$ (see footnote 1 on page 76) converges quadratically under the same assumptions we invoke here, we should expect that quadratic convergence should take place for the regularized rPDAS, as long as δ^k goes to zero "fast enough". This is indeed the case, as the analysis below will show.

As in Chapter 2, the local convergence analysis here uses a simple lemma from [TZ94, Proposition 3.10] (restated in Lemma 2.4.7 of this dissertation) that shows, to prove the quadratic rate inequality,

$$\|z^+ - z\| \leq c\|z - z^*\|^2,$$

it is sufficient to consider one step of Algorithm 11 from $z := (x, y)$ to $z^+ = (x^+, y^+)$

[7] Lemma 3.2.7 says that if $y^k \not\to F^*$ then φ^k is bounded away from zero, so that (3.23) implies $\delta^k \to 0$ is possible *only if* $y^k \to F^*$.

and verify that one of the two conditions

$$|z_i^+ - z_i| \leq c\|z - z^*\|^2, \tag{3.28}$$

$$|z_i^+ - z_i^N| \leq c\|\Delta z^N\|^2, \tag{3.29}$$

holds for each component of the update for every z close enough to z^*, all admissible Q, and all admissible δ (as z^+ will depend only on z, Q, and δ, although we suppress indication of such dependence in our notation for readability).[8] To establish these inequalities, we need a slight modification of Lemma 2.4.9 of the previous chapter (a minor variant of [TAW06, Lemma 16]), which bounds the difference between the reduced, regularized step Δz_Q and the Q-component Δz_Q^N of the unreduced, unregularized (Newton) step. The unreduced, unregularized step Δz^N is, in fact, the Newton step for

$$\Phi(x, y) = \begin{pmatrix} X(c - A^{\mathrm{T}}y) \\ Ax - b \end{pmatrix} = 0,$$

and is given as the solution to the linear system

$$\begin{pmatrix} S & -XA^{\mathrm{T}} \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x^N \\ \Delta y^N \end{pmatrix} = \begin{pmatrix} -Xs \\ b - Ax \end{pmatrix}, \tag{3.30}$$

so that, by eliminating $\Delta x_{\mathbf{n}\setminus Q}^N$ from (3.30), we get

$$\begin{pmatrix} S_Q & -X_Q A_Q^{\mathrm{T}} \\ A_Q & -(A_{\mathbf{n}\setminus Q} X_{\mathbf{n}\setminus Q} S_{\mathbf{n}\setminus Q}^{-1} A_{\mathbf{n}\setminus Q}^{\mathrm{T}}) \end{pmatrix} \begin{pmatrix} \Delta x_Q^N \\ \Delta y^N \end{pmatrix} = \begin{pmatrix} -X_Q s_Q \\ b - A_Q x_Q \end{pmatrix}. \tag{3.31}$$

On the other hand, if we multiply the first block equation of (3.10) through by $-X_Q$,

[8] By admissible we mean quantities that could be generated by our Algorithm 11. For Q this just means $Q \in \mathfrak{Q}_{\epsilon,M}(y)$, while for δ it is a little more subtle, since δ is defined in terms of the "previous" iterate. Nonetheless, there is a relationship (see (3.36) below) between the "current" primal iterate x and δ that is enforced by this algorithm, and that is all we will need.

we get

$$\begin{pmatrix} S_Q & -X_Q A_Q^{\mathrm{T}} \\ A_Q & \delta I \end{pmatrix} \begin{pmatrix} \Delta x_Q \\ \Delta y \end{pmatrix} = \begin{pmatrix} -X_Q s_Q \\ b - A_Q x_Q \end{pmatrix}. \tag{3.32}$$

Noting the equality of the right-hand-sides of equations (3.31) and (3.32), we equate their left-hand-sides and after a bit of manipulation, end up with

$$\begin{pmatrix} S_Q & -X_Q A_Q^{\mathrm{T}} \\ A_Q & \delta I \end{pmatrix} (\Delta z_Q - \Delta z_Q^N) = \begin{pmatrix} 0 & 0 \\ 0 & -(A_{\mathbf{n}\setminus Q} X_{\mathbf{n}\setminus Q} S_{\mathbf{n}\setminus Q}^{-1} A_{\mathbf{n}\setminus Q}^{\mathrm{T}} + \delta I) \end{pmatrix} \Delta z_Q^N.$$

In a neighborhood of z^*, under Assumption 3, Lemma 2.1.1 (or [TAW06, Lemma 1]) can be used to show that the coefficient matrix on the left side of this equation has uniformly bounded inverse near z^* when Q, and δ are admissible. Thus, for all z close enough to z^*, we have the bound

$$\|\Delta z_Q - \Delta z_Q^N\| = c_1 (A_{\mathbf{n}\setminus Q} X_{\mathbf{n}\setminus Q} S_{\mathbf{n}\setminus Q}^{-1} A_{\mathbf{n}\setminus Q}^{\mathrm{T}} + \delta) \|\Delta z_Q^N\|, \tag{3.33}$$

for some c_1 independent of z and Q. Finally since $s^*_{\mathbf{n}\setminus Q}$ is strictly positive, $s_{\mathbf{n}\setminus Q}$ is bounded away from zero close enough to the solution and $x^*_{\mathbf{n}\setminus Q} = 0$. Therefore we have

$$\begin{aligned} \|A_{\mathbf{n}\setminus Q} X_{\mathbf{n}\setminus Q} S_{\mathbf{n}\setminus Q}^{-1} A_{\mathbf{n}\setminus Q}\| &\leq \|A_{\mathbf{n}\setminus Q}\|^2 \|S_{\mathbf{n}\setminus Q}^{-1}\| \|X_{\mathbf{n}\setminus Q}\| \\ &\leq c_2 \|x_{\mathbf{n}\setminus Q}\| = c_2 \|x_{\mathbf{n}\setminus Q} - x^*_{\mathbf{n}\setminus Q}\| \leq c_2 \|z - z^*\|, \end{aligned} \tag{3.34}$$

and from (3.18) and (3.23), if z is close enough to z^*, every admissible δ satisfies

$$\delta = \min\{\phi^-, \bar{\delta}\}, \tag{3.35}$$

where φ^- indicates the value of φ from the previous step. On the other hand, (3.18)

implies that, for the same φ^-,

$$\min\{\varphi^-, \underline{\xi}^{\max}\} \leq \min_{i \in Q} x_i \leq \min_{\{i \in Q \mid x_i^* = 0\}} |x_i - 0| \leq \|x - x^*\| \leq \|z - z^*\|, \tag{3.36}$$

and we note that $\{i \in Q \mid x_i^* = 0\}$ is nonempty, since Rule 3.2.1 ensures that Q contains at least one constraint that is inactive at the solution, i.e., there is some index $i' \in Q$, with $s_{i'}^* > 0$, so that $x_{i'}^* = 0$ by complementarity. Finally, (3.35) and (3.36) then imply that for all z close enough to z^*,

$$\delta \leq \|z - z^*\|,$$

and putting this together with (3.33) and (3.34), we get

$$\|\Delta z_Q - \Delta z_Q^N\| = c\|z - z^*\| \cdot \|\Delta z_Q^N\|,$$

which is the conclusion of [TAW06, Lemma 16] and Lemma 2.4.9. The rest of the local convergence analysis follows arguments made in [TAW06] and/or in Chapter 2, to which we refer the reader.

3.2.2 Regularization in the limit of small δ: motivating the second algorithm

Consider the Δy component of the search direction of Algorithm 11, which satisfies the regularized normal equation (for simplicity of notation, we take $Q = \mathbf{n}$ here)

$$(AS^{-1}XA^{\mathrm{T}} + \delta I)\Delta y = b. \tag{3.37}$$

We investigate what happens in the limit of small regularization parameter δ. For this, it helps to consider a spectral decomposition of the normal matrix

$$M = AS^{-1}XA^{\mathrm{T}} = V\Sigma V^{\mathrm{T}} = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^{\mathrm{T}} \\ V_2^{\mathrm{T}} \end{pmatrix}$$

with the columns of V_1 spanning $\mathcal{R}(M)$, the columns of V_2 spanning $\mathcal{N}(M)$, and $\Sigma_1 > 0$. Using this, the dual step can be expressed as

$$\begin{aligned} \Delta y &= (V\Sigma V^{\mathrm{T}} + \delta V V^{\mathrm{T}})^{-1} b \\ &= V(\Sigma + \delta I)^{-1} V^{\mathrm{T}} b \\ &= \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 + \delta I & 0 \\ 0 & \delta I \end{pmatrix}^{-1} \begin{pmatrix} V_1^{\mathrm{T}} \\ V_2^{\mathrm{T}} \end{pmatrix} b \\ &= V_1(\Sigma_1 + \delta I)^{-1} V_1^{\mathrm{T}} b + \delta^{-1} V_2 V_2^{\mathrm{T}} b. \end{aligned} \tag{3.38}$$

As $\delta \to 0$, the first term of (3.38) converges to the least norm solution to $M\Delta y = b$. The second term of the last expression is δ^{-1} times the projection of b onto $\mathcal{N}(A^{\mathrm{T}}) = \mathcal{N}(M)$. So for vanishing δ, the regularized dual search direction will be along the projection of b onto the nullspace of A^{T}, unless b has no component in this nullspace, i.e., unless Δy is in the range of A, in which case, the search direction will be the least norm solution to the normal equations. Next we analyze an algorithm based on this limit direction.

Remark 3.2.1. *Interestingly, a variant of the kernel step algorithm presented below was first proposed by Tits, Absil and O'Leary [TAO06], before the idea for the regularized rPDAS algorithm came about. Only later was this connection made between the two algorithms.*

3.2.3 Kernel step rPDAS

This algorithm is similar to Algorithm 11, and we are able to prove similar results. The analysis is a blend of arguments developed in an unpublished technical report of Tits, Absil and O'Leary [TAO06], those developed in in Chapter 2 (both of which borrow from [TAW06]), and some new ideas. As before, the algorithm starts from a dual strictly feasible point y^0, and primal (possibly infeasible) interior point, i.e., $x^0 > 0$ but not necessarily with $Ax^0 = b$. We again select Q according to Rule 3.2.1. Next, we determine whether $b \in \mathcal{R}(A_Q)$. If so, we take a "regular step" defined to be the least norm solution to

$$(A_Q S_Q^{-1} X_Q A_Q)\Delta y = b. \tag{3.39}$$

If, instead, $b \notin \mathcal{R}(A_Q)$, then we take a "kernel step" along Δy, which is defined to be the projection of b onto $\mathcal{N}(A_Q^{\mathrm{T}})$. In either case, the primal step is defined in the same way as before using equations (3.13) and (3.18)-(3.20). Notice however, that $\Delta s_Q = 0$ for kernel steps. This implies $\tilde{x}_Q = -S_Q^{-1} X_Q \Delta s_Q = 0$, so that by (3.14), $\tilde{x} = 0$ for kernel steps. In the case of a regular step, the dual step length t_d is chosen according to (3.16)-(3.17), repeated here:

$$t_{\mathrm{d}} = \max\{\beta \bar{t}_{\mathrm{d}}, \bar{t}_{\mathrm{d}} - \|\Delta y\|\}, \tag{3.40}$$

$$\bar{t}_{\mathrm{d}} = \min\left\{1, \ \min\left\{\frac{s_i}{-\Delta s_i} \mid \Delta s_i < 0\right\}\right\}. \tag{3.41}$$

In the case of a kernel step, we define t_{d} differently. First of all, the kernel step length is not limited to 1; we go along this direction until we hit a blocking constraint not in working set Q, and take a step almost all of the way to the boundary. Specifically, defining

$$\hat{\imath} := \arg\min\left\{\frac{s_i}{-\Delta s_i} \mid \Delta s_i < 0\right\}, \tag{3.42}$$

(Proposition 3.2.14 below shows that $\hat{\imath}$ is always well defined) and

$$\overline{s} := \min\{\overline{s}_M, \epsilon\}, \tag{3.43}$$

where $\epsilon > 0$ and M are the defining parameters for $\mathfrak{Q}_{\epsilon,M}(y)$, $\overline{s}_M$ is an Mth smallest entry of s, and t_{d} is chosen as

$$t_{\mathrm{d}} := \frac{s_{\hat{\imath}} - \theta\overline{s}}{|\Delta s_{\hat{\imath}}|}, \tag{3.44}$$

where $\theta \in (0, 1)$ is an algorithm parameter. This choice of t_{d} ensures that the blocking constraint (indexed by $\hat{\imath}$) will be contained in the updated working set Q^+, and enforces a condition that prevents the algorithm from executing only kernel steps (Lemma 3.2.19).

We now state the kernel step rPDAS algorithm. The iteration index will be justified below in Proposition 3.2.14.

Algorithm 12: Kernel-step rPDAS

Input: LP data: A, b, c; Initial iterate: $y^0, s^0 = c - A^{\mathrm{T}}y^0 > 0$, $x^0 > 0$;

Parameters: $M \in \mathbf{n}$, $\epsilon > 0$, $\underline{\xi}^{\max} > 0$, $\overline{\chi} > 0$, $\beta \in (0, 1)$;

Set $k = 0$, choose Q^0 obeying Rule 3.2.1;

while *forever* **do**

  if $b \in R(A_{Q^k})$ **then** (regular step)

    Compute Δy^k as the least norm solution of (3.12);

    Update the dual variables using (3.15)-(3.17);

  else (kernel step)

    Set Δy^k equal to the orthogonal projection of b onto $\mathcal{N}(A^{\mathrm{T}}_{Q^k})$;

    Update the dual variables using (3.42)-(3.44);

  end

  Compute $\Delta x^k_{Q^k}$, $\tilde{x}^k$, and φ^k from (3.13), (3.14), and (3.22);

  Update the Q^k component of the primal variables using (3.18)-(3.20);

  Compute $\mu^+_{Q^k}$ from (3.21), and compute $x^+_{\mathbf{n}\setminus Q}$ using (3.18)-(3.20);

  Choose Q^{k+1} according to Rule 3.2.1;

  Set $k = k + 1$;

end

Global convergence

First we note that if Δy comes from a kernel step, then it cannot be arbitrarily small.

Lemma 3.2.12. *There exists $\gamma > 0$, depending only on A and b, such that $\|\Delta y\| \geq \gamma$ whenever Δy corresponds to a kernel step.*

Proof. A kernel step is taken only when b is not in the range of A_Q, thus not orthogonal to the kernel of A_Q^T. Since kernel steps Δy are obtained by projection of b on the kernel of A_Q^T, they are nonzero. Since there are only finitely many different

such Δy (because there are only finitely many submatrices A_Q), there must exist $\gamma > 0$ such that $\|\Delta y\| \geq \gamma$. □

As in the analysis of the regularized rPDAS, monotonicity of the objective plays a key role. The first lemma shows that Δy generated by the kernel step algorithm is always an ascent direction.

Lemma 3.2.13. *Let Δy be as constructed by Algorithm 12. Then $\Delta y \neq 0$ and $b^T \Delta y > 0$.*

Proof. If $b \in \mathcal{R}(A_Q)$, under the assumption that $b \neq 0$, $\Delta y = 0$ is not a solution to equation (3.39), so the least norm solution Δy is nonzero; and if $b \notin \mathcal{R}(A_Q)$, then b is not orthogonal to $\mathcal{N}(A_Q^{\mathrm{T}})$ and hence its orthogonal projection Δy on $\mathcal{N}(A_Q^{\mathrm{T}})$ again is nonzero. If $b \notin \mathcal{R}(A_Q)$, the second claim is immediate and if $b \in \mathcal{R}(A_Q)$, it follows from positive semidefiniteness of $A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}}$ that

$$(\Delta y)^{\mathrm{T}} b = (\Delta y)^{\mathrm{T}} (A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}}) \Delta y = \|(A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}})^{1/2} \Delta y\|^2 > 0.$$

Indeed, $(A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}})^{1/2} \Delta y$ cannot vanish, since $b = (A_Q S_Q^{-1} X_Q A_Q^{\mathrm{T}}) \Delta y \neq 0$. □

Under a mild assumption, Algorithm 12 is well defined, and can be repeated *ad infinitum*.[9]

Proposition 3.2.14. *Suppose Assumptions 1 and 2 hold. Then Algorithm 12 is well defined, and constructs quantities with the properties that $t_{\mathrm{d}} \in (0, \infty)$, $y^+ \in F^{\mathrm{o}}$, $s^+ = c - A^{\mathrm{T}} y^+ > 0$, and $x^+ > 0$.*

Proof. First, since $s > 0$, it always holds that $\overline{t}_{\mathrm{d}} > 0$. Since $b^{\mathrm{T}} \Delta y > 0$ (Lemma 3.2.13), it follows from nonemptiness of the solution set, which is implied by Assumptions 1

[9] When $b = 0$, which is of course a trivial situation since we require dual feasibility, we have $x^+ = 0$. Still, when $b = 0$ and $x = 0$ and all other requisite conditions are satisfied, the kernel step rPDAS algorithm is well defined, produces again $x = 0$ (which indeed is the optimal x), and can be repeated *ad infinitum*.

and 2, that the largest feasible dual step size must be finite. This implies, in view of (3.40) – (3.42), that $\Delta s_i < 0$ for some i, so $t_{\rm d}$ is always finite and $\hat{\imath}$ is well defined. It is also easy to see that, under a kernel step, $\Delta s_Q = 0$, so we must have $\hat{\imath} \in \mathbf{n} \setminus Q$. In view of (3.15), it is always the case that $s^+ = c - A^{\rm T} y^+$. It remains to show that $t_{\rm d} > 0$ and that $s^+ > 0$ (i.e., $y^+ \in F^o$) and $x^+ > 0$. When $b \in \mathcal{R}(A_Q)$, $t_{\rm d} > 0$ follows from (3.40) and the fact that $\overline{t}_d > 0$. When $b \notin \mathcal{R}(A_Q)$, $t_{\rm d} > 0$ follows from (3.44) and the facts that $|\Delta s_{\hat{\imath}}| > 0$, $s_{\hat{\imath}} > 0$, $\theta \in (0,1)$, $\overline{s} \leq s_{\hat{\imath}}$ (since $\hat{\imath} \in \mathbf{n} \setminus Q$). Next, the inequality $s^+ > 0$ follows from (3.41) and (3.15) when $b \in \mathcal{R}(A_Q)$, while, when $b \notin \mathcal{R}(A_Q)$, we have for all i,

$$s_i^+ = s_i + t_{\rm d} \Delta s_i = s_i \left(1 + t_{\rm d} \frac{\Delta s_i}{s_i} \right) \geq s_i \left(1 + t_{\rm d} \frac{\Delta s_{\hat{\imath}}}{s_{\hat{\imath}}} \right) = s_i \left(1 - \frac{s_{\hat{\imath}} - \theta \overline{s}}{s_{\hat{\imath}}} \right) > 0, \tag{3.45}$$

where we have used definition (3.42) of $\hat{\imath}$ and the fact that $\theta \overline{s} > 0$. Finally, $\Delta y \neq 0$ (Lemma 3.2.13), (3.18) and (3.22) imply that for $i \in Q$, $x_i^+ > 0$, while for $i \in \mathbf{n} \setminus Q$, (3.18), (3.21), $x_Q^+ > 0$, and $s^+ > 0$, again give $x_i^+ > 0$. □

Our analysis focuses on the dual sequence $\{y^k\}$. First, we have the analog of Lemma 3.2.4.

Lemma 3.2.15. *The sequence $\{b^{\rm T} y^k\}$ is strictly monotonically increasing. Further, if Assumptions 1 and 2 hold, then $\{y^k\}$ is bounded.*

Proof. Strict monotonicity of $\{b^{\rm T} y^k\}$ follows from Lemma 3.2.13 (and that $b \neq 0$), Proposition 3.2.14 ($t_{\rm d}^k > 0$), and (3.15). Assumptions 1 and 2 imply that the dual solution set is nonempty and bounded, which is equivalent to the superlevel sets $\{y \in F \mid b^{\rm T} y \geq \alpha\}$ being bounded for all α. Boundedness of $\{y^k\}$ then follows from its feasibility and monotonicity of $\{b^{\rm T} y^k\}$ (Lemma 3.2.13). □

Lemma 3.2.16. *Suppose Assumption 2 holds. Let K be the set of indexes k such that a kernel step is taken from y^k. Then $\sum_{k \in K} \|y^{k+1} - y^k\|$ converges.*

Proof. Since $\{b^T y^k\}$ is nondecreasing and bounded (Lemma 3.2.15), we know that

$$\sum_{k=0}^{\infty} |b^T(y^{k+1} - y^k)| = \sum_{k=0}^{\infty} b^T(y^{k+1} - y^k) \ < \ \infty, \tag{3.46}$$

which implies that

$$\sum_{k \in K} |b^T(y^{k+1} - y^k)| \ < \ \infty. \tag{3.47}$$

To complete the proof, we show that there is a constant C such that

$$\|y^{k+1} - y^k\| < C|b^T(y^{k+1} - y^k)| \quad \forall k \in K. \tag{3.48}$$

Since $y^{k+1} - y^k = t_{\mathrm{d}}^k \Delta y^k$ and $b^T(y^{k+1} - y^k) = t_{\mathrm{d}}^k b^T \Delta y^k$, (3.48) is equivalent to

$$\|\Delta y^k\| < C|b^T \Delta y^k|. \tag{3.49}$$

However, since a kernel step is taken whenever $k \in K$, (3.49) follows from the fact that (due to finiteness of the set of possible Q^k) the angle between b and Δy^k is bounded away from 90 degrees over K. □

As before, the sequence of primal variables generated by Algorithm 12 also remains bounded.

Lemma 3.2.17. *Suppose Assumptions 1 and 2 hold. Then $\{x^k\}$ and $\{\tilde{x}^k\}$ are bounded.*

Proof. We first show that $\tilde{x}^k_{Q^k}$ is bounded. First note that if k is an iteration index corresponding to a kernel step, then $\Delta s^k_{Q^k} = 0$, and therefore, in view of (3.13), (3.14) and (3.11), we have

$$\tilde{x}^k_{Q^k} = -(S^k_{Q^k})^{-1} X^k_{Q^k} \Delta s^k_{Q^k} = 0,$$

while $\tilde{x}_{\mathbf{n}\setminus Q^k} := 0$. If instead, k corresponds to a regular step, then defining $D^k_{Q^k} := X^k_{Q^k}(S^k_{Q^k})^{-1}$, from the definition of the regular step and (3.13)-(3.14), we have

$$\tilde{x}^k_{Q^k} = D^k_{Q^k} A^{\mathrm{T}}_{Q^k} (A_{Q^k} D^k_{Q^k} A^{\mathrm{T}}_{Q^k})^{\dagger} b, \tag{3.50}$$

where $\dagger$ denotes the pseudoinverse. In general for $M \succeq 0$, when $b \in \mathcal{R}(M)$, it holds that $(M + \delta I)^{-1} b \to M^{\dagger} b$ as $\delta \to 0$ (c.f. section 3.2.2). Together with the fact, derived in Lemma 3.2.5, that for some R independent of Q^k, $D^k_{Q^k}$, and any $\delta > 0$,

$$\| D^k_{Q^k} A^{\mathrm{T}}_{Q^k} (A_{Q^k} D^k_{Q^k} A^{\mathrm{T}}_{Q^k} + \delta I)^{-1} b \| \leq R,$$

(3.50) establishes boundedness of the $\{\tilde{x}^k\}$ sequence (since again $\tilde{x}^k_{\mathbf{n}\setminus Q} := 0$). Let $R' \geq R$ be such that $\|\tilde{x}^k\|_\infty \leq R'$ where, without loss of generality, $R' \geq \|x^0\|_\infty$. Thus, since $x^{k+1}_{Q^k}$ is in the convex hull of $\{x^k_{Q^k}, \tilde{x}^k_{Q^k}\}$, we have that $\|x^{k+1}_{Q^k}\|_\infty \leq R'$, while for $i \in \mathbf{n} \setminus Q^k$, x^{k+1}_i is explicitly bounded above by $\overline{\chi}$, and we may assume $R' \geq \overline{\chi}$ without loss of generality. The result then follows by induction.

□

The next lemma is analogous to Lemma 3.2.6.

Lemma 3.2.18. *Suppose Assumption 1 holds and $\Delta y^k \to 0$ on an infinite index set K, then $\tilde{X}^k s^k \xrightarrow{K} 0$ and $A\tilde{x}^k \xrightarrow{K} b$ on K. In particular, all limit points (y', x') of the bounded sequence $\{(y^k, \tilde{x}^k)\}_{k \in K}$ have $y' \in F^{\mathrm{s}}$ with x' a corresponding multiplier.*

Proof. In view of Lemma 3.2.12, there is no loss of generality in assuming that, for all $k \in K$, Δy^k is constructed by a regular step. The claim that $A\tilde{x}^k \xrightarrow{K} b$ is a straightforward consequence of (3.13) and definition (3.14). (In fact, $A\tilde{x}^k = b$ when k corresponds to a regular step.) Next, we prove asymptotic complementarity of

$\{(\tilde{x}^k, s^k)\}_{k\in K}$. Using again (3.13) and (3.14) we have, for all k,

$$\tilde{x}_j^k s_j^k = -x_j^k \Delta s_j^k,\ j \in Q^k, \tag{3.51}$$

$$\tilde{x}_j^k s_j^k = 0,\ j \in \mathbf{n} \setminus Q^k. \tag{3.52}$$

Since x^k is bounded (Lemma 3.2.5), and $\Delta s^k = -A^{\mathrm{T}}\Delta y^k \to 0$ on K, this implies $\tilde{x}_j^k s_j^k \to 0$ on K for all j. □

Lemma 3.2.19. *Suppose that Assumptions 1 and 2 hold. Then the sequence of regular steps is infinite.*

Proof. Proceeding by contradiction, suppose that eventually, for $k \geq k_0$ say, all steps are kernel steps, and consider $k \geq k_0$. Let Π^k be the set of indices of the components of s^k that are smaller than ϵ. If $|\Pi^k| \leq M$, then according to Rule 3.2.1, $\Pi^k \subseteq Q^k$ and so $s_i^{k+1} = s_i^k < \epsilon$ for all $i \in \Pi^k$, while, in view of the step-size rule (3.44), for $\hat{\imath} \in \mathbf{n} \setminus Q^k \subseteq \mathbf{n} \setminus \Pi^k$, $s_{\hat{\imath}}^{k+1} = \theta\overline{s}^k < \epsilon$, so that $|\Pi^{k+1}| \geq |\Pi^k| + 1$. On the other hand, if $|\Pi^k| > M$, again by Rule 3.2.1, there is a subset of Π^k of size M that is contained in Q^k, and again since $s_i^{k+1} = s_i^k < \epsilon$, for all $i \in Q^k$ and $s_{\hat{\imath}}^{k+1} < \epsilon$, we have $|\Pi^{k+1}| \geq M+1$. Thus we can assume without loss of generality that $|\Pi^k| > M$, and thus, for all k, $\overline{s}^k = \overline{s}_M^k$ (from (3.43)).

Now let σ_M^k denote the sum of M smallest entries of s^k, we will show that under the contradiction hypothesis, $\sigma_M^k \to 0$. Yet again, noting that for all $i \in Q^k$, $s_i^{k+1} = s_i^k$, and with $\hat{\imath} \in \mathbf{n} \setminus Q^k$, $s_{\hat{\imath}}^{k+1} = \theta\overline{s}^k = \theta\overline{s}_M^k$, and since $\sigma_M^k \leq M\overline{s}_M^k$, we have

$$\begin{aligned}
\sigma_M^{k+1} &\leq \sigma_M^k + s_{\hat{\imath}}^{k+1} - \overline{s}_M^k \\
&= \sigma_M^k - (1-\theta)\overline{s}_M^k \\
&\leq \left(1 - \frac{1-\theta}{M}\right)\sigma_M^k.
\end{aligned}$$

Thus indeed, $\sigma_M^k \to 0$. Now, due to the finite number of columns of A, there must

exist some $\hat{Q}$ and some indexes $i_1, i_2, \ldots, i_M \in \hat{Q}$ such that, for some infinite index set K, $Q^k = \hat{Q}$ and $\sigma_M^k = \sum_{\ell=1}^M s_{i_\ell}^k$ for all $k \in K$, k large enough. Hence, for $\ell = 1, 2, \ldots, M$, $s_{i_\ell}^k$ goes to zero as k goes to infinity, $k \in K$. If we assume, without loss of generality (since, as per Lemma 3.2.15, y^k is bounded), that y^k converges to some dual-feasible y' on K, all constraints i_ℓ, $\ell = 1, 2, \ldots, M$, are active at y'. Since $i_1, i_2, \ldots, i_M \in \hat{Q}$ and since, by Rule 3.2.1, M is a *strict* upper bound on the number of constraints active at any dual-feasible point, we have our contradiction.[10] □

The next two lemmas are the analogs of Lemmas 3.2.7 and 3.2.8 from the analysis of Algorithm 11. They apply almost identically, except only on infinite index sets K corresponding to regular steps.

Lemma 3.2.20. *Suppose Assumptions 1 and 2 hold. Then, if* $\varphi^k \xrightarrow{K} 0$ *for any infinite index set* K *consisting of regular steps, then* $y^k \to F^*$.

Thus if $y^k \not\to F^*$ then we must have, for some $\varepsilon > 0$, $\varphi^k \geq \varepsilon$ (since $\liminf_{k\to\infty} \varphi^k > 0$ and $\varphi^k \neq 0$) and this will be shown to lead to a contradiction.

Lemma 3.2.21. *Suppose Assumptions 1 and 2 hold and let* K *be the set of regular steps. If* $\inf_{k\in K} \varphi^k \geq \varepsilon > 0$, *then* $\Delta y^k \xrightarrow{K} 0$.

Proof. Suppose $\varphi^k \geq \varepsilon > 0$ and for some infinite $K' \subseteq K$, $\|\Delta y^k\| \geq \varepsilon'$ on K'. From the update equation (3.18), $\varphi^k \geq \varepsilon$ implies $x_i^{k+1} \geq \varepsilon$ for all i and k (i.e., $x_i^k \geq \varepsilon$ for all i and k). Assume, without loss of generality, that $y^k \to y'$ on K' and $Q^k = Q$ is constant for $k \in K'$. Note that we then must have $I(y') \subseteq Q$ by Rule 3.2.1. Also, since y^k is bounded, $s^k = c - A^{\mathrm{T}} y^k$ is also bounded, and we let R be an upper bound for $\|s^k\|$.

Since K' consists of regular steps, Δy^k is the least-norm solution to the normal equations. This implies that $\Delta y^k \perp \mathcal{N}(A_Q^{\mathrm{T}})$ and thus that $\|\Delta s_Q^k\| = \|A_Q^{\mathrm{T}} \Delta y^k\| >$

[10] One of the reasons that Rule 3.2.1 requires M to be a *strict* upper bound on the number of active constraints at any dual feasible point is so that this argument can be made.

$\beta' > 0$ for all $k \in K'$. Therefore, we have, for all $k \in K'$,

$$b^{\mathrm{T}}\Delta y^k = (\Delta y^k)^{\mathrm{T}}(A_Q(S_Q^{-1})^k X_Q^k A_Q^{\mathrm{T}})\Delta y^k = (\Delta s_Q^k)^{\mathrm{T}}(S_Q^{-1})^k X_Q^k \Delta s_Q^k \geq \frac{\varepsilon}{R}\beta' > 0$$

Next we seek a positive lower bound for t_{d}. In view of (3.17), it suffices to lower bound $\bar{t}_{\mathrm{d}}$. From (3.17), either $t_{\mathrm{d}}^k = 1$, or $t_{\mathrm{d}}^k = \frac{s_{i(k)}}{-\Delta s_{i(k)}^k}$ for some $\Delta s_{i(k)}^k < 0$ and index $i(k)$ depending on k. Suppose there is an infinite index set $K'' \subseteq K'$ such that $i(k) \in I(y')$. Then, since $I(y') \subseteq Q$, we have using the first block equation of (3.10), for all $k \in K''$,

$$t_{\mathrm{d}}^k := \frac{s_{i(k)}}{-\Delta s_{i(k)}^k} = \frac{x_{i(k)}}{\tilde{x}_{i(k)}^k} \geq \gamma > 0,$$

where we used $x_{i(k)} \geq \varepsilon$ and $\{\tilde{x}^k\}_{k\in K}$ bounded (Lemma 3.2.5). On the other hand, if there is no such K'' then there must be an infinite index set $K''' \subseteq K'$, on which $i(k) \in \mathbf{n} \setminus I(y')$, in which case, since $s_{i(k)}$ is bounded away from zero on K''', and Δs^k is bounded (as we show next), we have $t_{\mathrm{d}}^k \geq \gamma > 0$ on K'''. To see that Δs^k is bounded, since $\Delta y^k \perp \mathcal{N}(A_Q^{\mathrm{T}})$, we can solve for Δy^k in (3.13) (using (3.14)), and then compute Δs^k from (3.11) to get

$$\Delta s^k = -A^{\mathrm{T}}(A_Q^{\mathrm{T}})^{\dagger} X_Q^{-1} S_Q \tilde{x}_Q^k,$$

and note that the right hand side is bounded on K''', since $\tilde{x}^k$ is bounded and $x^k \geq \varepsilon$ on K'''. In either case, we have that the dual objective, which is monotonically increasing overall, increases by a constant amount on an infinite index set, which is impossible, since we have assumed the dual solution set to be bounded. Thus, we must have $\Delta y^k \to 0$. □

Lemma 3.2.22. *Suppose Assumptions 1 and 2 hold. Then $\{y^k\}$ converges to the set of stationary points, F^{s}.*

Proof. By boundedness, $\{y^k\}$ converges to its set of accumulation points, which we

will show are stationary. Lemma 3.2.16 implies that accumulation points of the entire sequence $\{y^k\}$ must also be accumulation points of $\{y^k\}_{k\in K}$, where K is the set of indices where regular steps are taken, so it suffices to show that accumulation points of $\{y^k\}_{k\in K}$ are all stationary. Clearly, either $\{\varphi^k\}_{k\in K}$ converges to zero or it is bounded away from zero. In the latter case $y^k \xrightarrow{K} F^s$ while in the former case $y^k \xrightarrow{K} F^* \subseteq F^s$. This concludes the proof. □

Once again, invoking the linear independence assumption used in Theorem 2.3.8, we can get the same global convergence to the optimal set. The arguments of [TAW06, Lm. 11,Th. 12] and Theorem 2.3.8 do the job here as well.

Theorem 3.2.23. *Suppose Assumptions 1 and 2 hold. Under the further assumption that, at every dual feasible point, the gradients of all active constraints are linearly independent, it holds that $y^k \to F^*$.*

Quadratic local convergence

We can again prove q-quadratic convergence of $\{z^k\} = \{(x^k, y^k)\}$ generated by the kernel step rPDAS algorithm under Assumption 3. As before, let y^* denote the unique solution to (1.4), i.e., $F^* = \{y^*\}$, let $s^* := c - A^{\mathrm{T}}y^*$, and let x^* be the corresponding multiplier vector, unique in view of Assumption 3. Assumption 3 again implies that strict complementarity holds, i.e.,

$$x_i^* > 0 \quad \forall i \in I(y^*). \tag{3.53}$$

and that

$$\operatorname{span}\{a_i : i \in I(y^*)\} = \mathbb{R}^m. \tag{3.54}$$

From Rule 3.2.1 it follows that for all k large enough, $I(y^*) \subseteq Q^k$, and the assumption ensures that A_{Q^k} has full row rank for all k large enough, so that, eventually,

no kernel steps are taken.

Proposition 3.2.24. *Suppose that Assumption 3 holds. Then, the sequence of kernel steps is finite.*

It follows that, for $k \geq k_0$, for some k_0, the sequences $\{x^k\}$ and $\{y^k\}$ are identical with those produced by setting parameter $\psi = 0$ in Algorithm 10/rMPC* of Chapter 2 (see footnote 1 on page 76), with x^{k_0} and y^{k_0} as initial primal and dual iterates. The analysis of section 2.4 then applies, so that the sequence $\{(x^k, y^k)\}$ converges q-quadratically. (See Theorem 2.4.1, and also Theorem 17 of [TAW06]).

Theorem 3.2.25. *Suppose Assumptions 3 holds. Then $\{(x^k, y^k)\}$ converges to (x^*, y^*) q-quadratically.*

3.3 Adding a barrier term

With some amount of work, a barrier term and a Mehrotra-type corrector component can be added to either of the algorithms discussed in this chapter.[11] This can be done in a similar way as was done for the unregularized case in Chapter 2, where the analysis of [TAW06] was modified to allow such an augmentation to the algorithm, preserving the convergence result. Such addition is expected to improve practical performance significantly. We leave the details for future work.

[11] In fact the method of adding the barrier term *after* computing the affine scaling component was critical in establishing convergence of the algorithm in Chapter 2, and presumably would be here as well. There the centering-corrector term is added with an adaptive weight that allows enforcement of dual ascent and other critical properties of the algorithm.

Chapter 4

Numerical experiments

In this chapter we develop some numerical experience with our algorithms. We investigate some problems that can be formulated as LPs with many inequality constraints, develop some specific constraint selection rules that fit within the general class used in Chapters 2-3, and run some numerical experiments, including a comparison against prior constraint-reduced IPMs. We attempt to identify whether our specific constraint-selection rules work well on each problem, and if not, understand why. We also try to identify good heuristics for constraint selection based on an understanding of the particular structure of the problem class. In section 4.1 below, we consider problems that are not particularly rank-degenerate and focus on our first algorithm rMPC*. Later, in section 4.2, we investigate the performance of the regularized and kernel-step rPDAS algorithms of Chapter 3 on some especially degenerate test problems, and observe some interesting qualitative behavior.

4.1 Numerical experiments with rMPC*

We first focus on Algorithm 10/rMPC*. We discuss in detail our implementation and then present some specific rules for constraint selection and investigate the

performance of rMPC* on two classes of problems. Finally, we make a comparison with some of the other constraint reduced IPMs discussed in Chapter 1. Sections 4.1.1-4.1.4 of this chapter are derived from [WNTO10].

4.1.1 Implementation

Algorithm rMPC* was implemented in Matlab(R) and run on an Intel(R) Pentium(R) Centrino Duo 1.73GHz Laptop machine with 2 GB RAM, Linux kernel 2.6.17 and Matlab 7 (R14). To compute the search directions (2.21) we solved the normal equations (2.49) and (2.51), using Matlab's Cholesky factorization routine `chol`. Parameters for rMPC* were chosen as $\beta := 0.95$, $\theta = 0.1$, $\psi = 10^9$, $\zeta = 0.3$, $\lambda = 3$, $\nu = 3$, $\chi = 10^9$, and $\underline{\xi}^{\max} := 10^{-11}$, and for each problem discussed below, we assume that a small upper bound M on the number of active constraints is available, and so we always take $\epsilon = \infty$. The code was supplied with strictly feasible initial dual points (i.e., $y^0 \in F^{\circ}$),and we set $x^0 := e$, the vector of ones.

We used a stopping criterion adapted from [Meh92, p. 592], based on normalized primal and dual infeasibilities and duality gap. Specifically, taking into account dual feasibility of all iterates, convergence was declared when

$$\texttt{termcrit} := \max\left\{\frac{\|b - Ax\|}{1 + \|x\|}, \frac{|c^{\mathrm{T}}x - b^{\mathrm{T}}y|}{1 + |b^{\mathrm{T}}y|}\right\} < \texttt{tol}, \tag{4.1}$$

where `tol` was set to 10^{-8}.

For algorithm rMPC*, the main focus of this chapter, our analysis assumes Q is selected according to the *general* Rule 2.2.1, i.e., that $Q \in \mathfrak{Q}_{\epsilon,M}(y)$ at each iteration. To complete the description of a *specific* rule for constraint selection, we simply need to specify any additional constraints that are to be included in Q, particularly so that rank condition C_2 of Definition 2.2.1 holds. (We discuss the regularized and kernel-step algorithms that don't have this issue, in section 4.2 below.) A simple

way to deal with C_2 is the following. At each iteration, set Q to be a set of M most active constraints, form the normal matrix and attempt to factor it. If C_2 fails, then the standard Cholesky factorization will fail. At this point simply add the next M most active constraints to Q, and repeat the factorization attempt with $|Q| = 2M$. If it still fails, increase $|Q|$ to $4M$ by adding the next most active constraints, etc. (On the next iteration we revert to using only M constraints.) We refer to this technique as the "doubling" method.

One alternative to the doubling method, discussed in section 3.1.3, is to augment the dual problem with bound constraints on the y variables, i.e., $-\pi e \leq y \leq \pi e$ for some scalar $\pi > 0$, and always include these constraints in Q in addition to the M most active ones. This ensures that C_2 holds, while adding negligible additional work (since the associated constraint vectors are sparse). Furthermore, in practice, π can be chosen large enough so that these constraints are never active at the solution. Both the doubling method and this "bounding" method were used in our tests of rMPC*, as indicated below.

Without resorting to the regularized algorithms (which we believe is the best way to resolve the rank-degeneracy issue), a third possibility, also mentioned in the introduction to Chapter 3, would be to use instead a pivoted Cholesky algorithm that will compute the factor of a nearby matrix [Hig90], regardless of Q. If C_2 fails, the factor can be (efficiently) updated by including additional constraints [GL83, Sec.12.5], chosen according to slack value or otherwise, until the estimated condition number [GL83, p.129] is acceptably small.

We refer to this rule that uses only the M most active constraints, doubling or bounding if needed, as the "Most Active Rule". While simple, the Most Active Rule does not always provide great performance on its own, and it may be desirable to keep additional constraints in Q to boost performance. In the sequel we describe some possible methods for selecting additional constraints, and in section 4.1.3 be-

low, we give a detailed example of how a heuristic may be developed and tailored to a particular class of problems.

4.1.2 Randomly generated problems

As a first test, following [TAW06], we randomly generated a standard form linear program of size $m = 200$, $n = 40000$, by taking $A,\ b,\ y_0 \sim \mathcal{N}(0,1)$ and then normalizing the columns of A. We set $s_0 \sim \mathcal{U}(0,1)$ (uniformly distributed in $(0,1)$) and $c := A^{\mathrm{T}}y_0 + s_0$ which guarantees that the initial iterate is strictly dual feasible. The iteration was initialized with this (s_0, y_0) and $x_0 := e$, the vector of ones. This problem is called the "fully random" problem in [TAW06], where a different x_0 is used.

On this problem class, every $m \times m$ submatrix of A has full rank, so that any $M \geq m$ is valid, and the doubling (or bounding) technique is never needed. Here, it turns out, constraint reduction works extremely well with the simple Most Active Rule as long as M is chosen slightly larger than m. Figure 4.1 shows the results for the Most Active Rule. The points on the plots correspond to different runs on the same problem. The runs differ in the number of constraints M that are retained in Q, which is indicated on the horizontal axis as a fraction of the full constraint set (i.e., M/n is plotted). Thus, the rightmost point corresponds to the experiment without constraint reduction, while the points on the extreme left correspond to the most drastic constraint reduction. To resolve the performance near $M = m$ (the lower bound for M), we have used a logarithmic scale. In the left plot, the vertical axis indicates, for each value of the abscissa, total CPU time to successful termination, as returned by the MATLAB function `cputime`, while the right plot shows the total number of iterations to successful termination. On the timing plot, a horizontal dotted line is used to show the time to solution for rMPC* in the unreduced case. This also essentially gives the performance of the original Iteration

MPC, as stated in section 1.12.2. Indeed, it was observed that the safeguards of rMPC* do not hurt the empirical performance of MPC in the unreduced case.

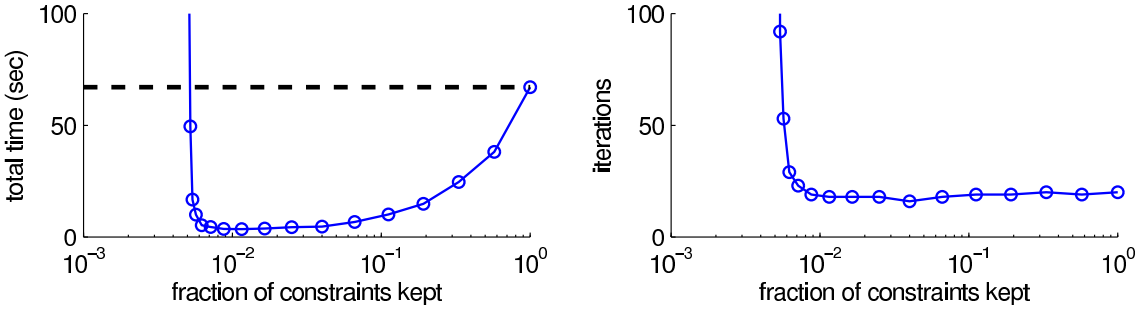

Figure 4.1: Most Active Rule on the size $m = 200$, $n = 40000$ random problem, horizontal axis is on a logarithmic scale. The horizontal dotted black line in the left plot marks the performance of the unreduced MPC algorithm. The vertical asymptote corresponds to keeping constraint sets of size approaching m. In particular, the leftmost data point corresponds to keeping only 205 constraints.

While the random problem has a large amount of redundancy in the constraint set, this may not always be the case, and in general we may not know a priori how many constraints should be kept. We also expect, intuitively, that fewer constraints will be needed as the algorithm nears the solution and the partition into active/inactive constraints (at the solution) becomes better resolved. Thus we would like to find rules that let the algorithm *adaptively* choose how many constraints it keeps at each iteration, i.e., that allow the cardinality of the working set to change from iteration to iteration. As an initial stride towards this end, we consider associating a scalar value v_i to each constraint for $i \in \mathbf{n}$. A large value of v_i indicates that we believe keeping constraint i in Q will improve the search direction and a small value means we believe it will not help or possibly will do harm. In addition to the M constraints selected according to the Most Active Rule, we add up to M' constraints that have $v_i \geq 1$, selecting them in order of largest value v_i first. We refer to this rule as the Adaptive Rule. We propose two specific variants of this rule.

In the first variant, we set

$$v_i = \eta \min_{j \in \mathbf{n}} \{s_j\}/s_i,$$

that is, we add additional constraints that have a slack value smaller than a fixed multiple $\eta > 1$ of the minimum slack. In the second variant, we set

$$v_i = \eta \sqrt{\frac{x_i/s_i}{\max\{x_j/s_j\}}},$$

that is, we add the ith constraint if $\sqrt{x_i/s_i}$ is within a fixed multiple $1/\eta$ of the maximum value of $\sqrt{x_j/s_j}$.[1] This rule combines information from both the primal and dual variables with regard to "activity" of this constraint. Note also that this v_i is the (scaled, square root of the) "coefficient" of the ith constraint in the normal matrix sum (1.16); thus we could interpret this rule as trying to keep the error between the reduced and unreduced normal matrix small. In view of (2.49a), we may expect that constraints with small values of $\sqrt{x_i/s_i}$ do not play much of a role in the construction of Δy^{m}.

Figure 4.2 shows the results of using the second variant of the Adaptive Rule on our random LP. We set $M = 2m$ and plot $(M + M')/n$ on the horizontal axis. The plot shows that, when $\eta = 10$, the average (over an optimization run) time per iteration increases very slowly as the upper bound $M + M'$ on $|Q|$ increases, starting from the lower bound $M = 2m$. (Indeed, the right plot shows that the total number of iterations remains roughly constant.) This means that the average size of $|Q|$ (over a run) itself increases very slowly, i.e., that $|Q|$ departs little from its minimum value $2m$ in the course of a run. If η is increased to 1000, the average value of $|Q|$ increases, which means more variation of $|Q|$ in the course of a run (since $|Q|$ is close to M_0 at the end of the runs: see below); this is the intended behavior. The general behavior of these rules is that in early iterations the v_i are spread out and, with large η, many

[1] The square root allows the use of similar magnitude η for both variants.

will be larger than the threshold value of one. Thus, the iteration usually starts out using M constraints, the upper bound. As the solution is approached, all v_i's tend to zero except those corresponding to active constraints which go to infinity (the second variant needs strict complementarity for this), thus in later iterations only the M_0 most active constraints (the lower bound) will be included in Q. We have observed that this transition from $M + M'$ to M constraints occurs rather abruptly usually over the course of just a few iterations; the choice of η serves to advance or delay this transition. In summary, the Adaptive Rule, like the Most Active Rule, keeps the number of iterations approximately constant over a wide range of choices of $M+M'$, but unlike the Most Active Rule, the time is also approximately constant, remaining much less than that for MPC.

We could think of many variations of the Adaptive Rule. Here we have only considered rules that choose v as a function of the current iterate, whereas we expect that by allowing v to depend on the entire history of iterates and incorporating more prior knowledge concerning the problem structure, etc., better constraint reduction heuristics could be developed. We believe that designing good adaptive rules will be a key to successful and robust application of constraint reduction; we largely leave this for future work.

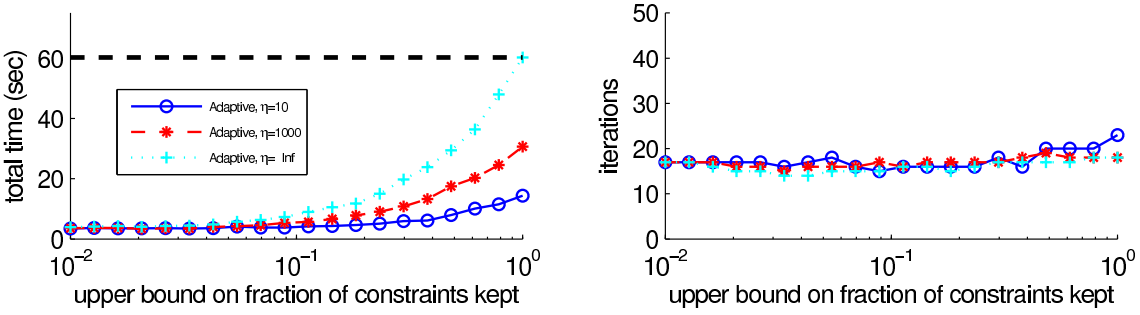

Figure 4.2: Adaptive Rule, second variant with $M = 2m$ and $\eta = 10^1, 10^3, \infty$ (setting $\eta = \infty$ corresponds to the Most Active Rule) on the size $m = 200$, $n = 40000$ random problem, horizontal axis on log scale. Here the horizontal axis represents $M + M'$, the upper bound on the size of the constraint set. Again, the horizontal dotted black line in the left plot marks the performance of the unreduced MPC algorithm.

4.1.3 Discrete Chebyshev approximation problems

Here we investigate a "real-world" application, fitting a linear model to a target vector by minimizing the infinity norm of the residual, namely,

$$\min_u \|Hu - g\|_\infty,$$

where g is the target vector, H is the model matrix, and u is the vector of model parameters. This can be formulated as a linear program in standard dual form

$$\max\{\ -t \mid Hu - g \leq te,\ -Hu + g \leq te\ \}. \tag{4.2}$$

If H has dimension $p \times q$, then the "A matrix" of this LP has dimension $m \times n$ with $m = q + 1$ and $n = 2p$ so that, if $p \gg q$ (as is typical), then $n \gg m$. Dual strictly feasible points are readily available for this problem; we used the dual-feasible point $u_0 = 0$ and $t_0 = \|g\|_\infty + 1$ to initialize the algorithm.

As a specific test, we took $p = 20000$ equally spaced samples of the smooth function

$$g_0(t) = \sin(10t)\cos(25t^2), \quad t \in [0, 1] \tag{4.3}$$

and stacked them in the p-dimensional vector g. For the columns of H, we took the $q = 199$ lowest frequency elements of the discrete Fourier transform (DFT) basis. When converted to (4.2), this resulted in a $m \times n$ linear program with $m = 200$ and $n = 40000$. For this problem, we circumvented the rank condition C_2 by adding the bound constraints $-10^3 \leq y \leq 10^3$ (for a total of 40400 constraints) and always including them in Q.

The initial results were poor: using the basic Most Active Rule with $M = 20m$ and an additional $3m$ randomly selected constraints, rMPC* required over 500 iterations to solve the problem to 10^{-8} accuracy. Numerical evidence suggests that

there are two distinct issues here; the first causes slow convergence in the initial phase of the iteration, reducing `termcrit` (see (4.1)) to around 10^{-2}, and the second causes slow convergence in the later phase of the iteration, further reducing `termcrit` to 10^{-8}.

The first issue is that since, for fixed y, the slack "function" $c - A^{\mathrm{T}}y$ is "smooth"[2]with respect to its index, the most nearly active constraints are all clustered into a few groups of contiguous indices corresponding to the minimal modes of the slack function. Intuitively, this does not give a good description of the feasible set, and furthermore, since the columns of A are also smooth in the index, A_Q is likely to be rank deficient, or nearly so, when only the most active constraints are included in Q, i.e., for the Most Active Rule. This clustering appears to cause slow convergence in the initial phase. This problem can in large part be avoided by adding a small random sample of constraints to Q: vastly improved performance is gained, especially in the initial phase.

The second issue, which persists even after adding random constraints, is that Q is missing certain constraints that appear to be critical in the later phase, namely the local minimizers of the slack function. The omission of these constraints results in very slow convergence in the later phase of the iteration. For example, we ran rMPC* using $M = 3m$ and adding $10m$ random constraints and observed that `termcrit` was reduced below 10^{-2} in 90 iterations, but that another 247 iterations were needed to achieve `termcrit`$< 10^{-8}$. Strikingly, in 88% of these later iterations, the blocking constraint, i.e., the one which limited the line search, was a local minimizer of the slack function not included in Q. If we instead used $M = m$ and again $10m$ random constraints, this happened in nearly 100% of the later iterations.

In light of these observations, we devised a simple heuristic for this class of smooth Chebyshev problems: use a small M, a small number of random constraints,

[2]This is because with the chosen discretization, the slack function is effectively oversampled by a very large factor.

and add the local minimizers of the slack function in Q (it is enough to keep those local minimizers with slack value less than, say, half of the maximum slack value). Note that in this case the size of the constraint set is not fixed a priori nor upper bounded—however since the target vector g and the basis elements have relatively low frequency content, adding the local minimizers generally added only a few (always fewer than m) extra constraints at each iteration.

Additional observations led to further refinement of this heuristic. First, we noted that the random constraints only seem to help in the early iterations and actually seem to slow convergence in the later iterations, so we considered gradually phasing them out as the iteration approached optimality. Second, we noted that in place of a random sample of constraints we could instead include all constraints from a regular grid of the form $\{i, i+j, i+2j, \ldots, i+(k-1)j\} \subseteq \mathbf{n}$ for some integers i, j, k with $i \in \{1, 2, \ldots j\}$, and $jk = n$.

Table 4.1 displays the performance of these various rules on the discrete Chebyshev approximation problem discussed above. The left side of the table describes the rule used: columns MA, RND and GRD give the number of most active, random, and gridded constraints respectively, and columns LM and COOL indicate whether the local minimizers of the slack function are included and whether the random constraints are phased out or "cooled" as the iteration nears optimality. The right side gives the performance of the corresponding rule: the first column lists the CPU time needed to reduce `termcrit` below 10^{-8}, the next two columns give the number of iterations needed to reduce `termcrit` below 10^{-2} and 10^{-8} respectively, and the last column gives the average size of the constraint set during the iteration. The first row of the table describes the unreduced MPC and gives a baseline performance level. The second row again illustrates the failure of the Most Active Rule, while the third and fourth show that adding randomly selected constraints and the local minimizers of the slack function effectively deals with the issues described above.

Rule Description					Performance			
MA	RND	GRD	LM	COOL	cputime	it:10^{-2}	it:10^{-8}	avg. $\|Q^k\|$
n	0	0	no	-	105.8 s	13	31	40400.0
13m	0	0	no	-	382.9 s	758	947	2849.8
3m	10m	0	no	no	180.2 s	82	492	2570.0
1m	10m	0	yes	no	14.5 s	17	41	2307.8
1m	10m	0	yes	yes	9.0 s	21	36	1027.4
1m	0	2m	yes	-	9.5 s	26	41	745.7

Table 4.1: Results of various heuristics on the Chebyshev approximation problem.

The fifth and sixth rows show enhancements of the specialized rule that achieve a 10-fold speed up over unreduced MPC.

4.1.4 Comparison with other algorithms

In this section we make a brief comparison of rMPC* vs. other constraint-reduced interior-point algorithms. We implemented the rPDAS algorithm of [TAW06] (discussed in 1.12 and labeled as `rpdas` on the tables below), rDPR of [Ton93] (discussed in section 1.9 and labeled `rdpr` on the tables), and the build-up DAS of [DY91] (discussed in section 1.8 and labeled on the tables as tt budas-ss and `budas-ls`, for the short and long-step variants respectively) in MATLAB, all using stopping criterion (4.1) with `tol` $= 10^{-8}$.[3] For `rpdas` we used the same implementation and parameters as rMPC*, but with $\psi = 0$. In our implementation of `budas` we used parameters $\beta = 0.95$, and we replaced the finite termination scheme used in [DY91] with our termination criterion (4.1). For `rdpr` we used $\alpha = 0.5$, $\delta^* = 0.05$, and $\rho = 5n$ (the latter attempts to get good practical performance although is not as good theoretically as $\rho = n + \nu\sqrt{n}$ for a constant $\nu > 1$). We also removed the finite termination scheme for `rdpr` (see [Ton93]), and since `rdpr` requires an upper bound on the dual optimal value, we used the optimal value obtained by rMPC*

[3]Stopping criterion (4.1) is appropriate because all tested algorithms produce dual-feasible iterates.

and added 10. Finally, again in `rdpr`, we used an Armijo line search in place of the exact minimizing line search.

For test problems we chose an instance of the 200×40000 random problem described in section 4.1.2 (`rand`), the Chebyshev approximation problem described in section 4.1.3 (`cheb`), and three problems from the netlib collection [net] with $n \gg m$ (`scsd1, scsd6`, and `scsd8`). (We note that the `scsd` problems are of less interest for applying constraint reduction because, as compared to our other test problems, they are of small dimension, less unbalanced, and very sparse which means the cost of forming the normal matrix is much less than $\mathcal{O}(nm^2)$.) We choose initial iterates for the (`cheb`) and (`rand`) as described in sections 4.1.2 and 4.1.3 and, for the `scsd` problems, we used a vector of zeros (dual strictly feasible) as the initial dual iterate and a vector of ones as the initial primal iterate. For each of these problems, we ran each algorithm, first using no reduction as a benchmark, then using a common constraint reduction rule. (Note that all tested algorithms allow for a heuristic constraint selection rule.) The constraint selection rules used in the test were as follows. First, as before, we always set $\epsilon = \infty$. Then, for the random problem we used $M = 2m$ most active constraints and no additional constraints, for the Chebyshev problem we used the rule corresponding to the last row of Table 4.1, and for the `scsd` problems we used $M = 2m$ most active and $2m$ randomly selected constraints. Finally, the remaining constraints were sorted by increasing slack value, and in the case of numerical issues solving the linear systems (in particular if rank condition C_2 of Definition 2.2.1 failed), or if the step was not acceptable, i.e., infeasible in the case of `budas` or did not achieve required decrease in the potential function for `rdpr`, the constraint set was augmented with $2|Q|$ additional constraints, where $|Q|$ refers to the original size of the constraint set, and the step was recomputed.

The results for the unreduced and reduced cases are shown in Tables 4.2 and 4.3,

respectively. The columns of each table are, in order: the problem name (`prob`), the algorithm (`alg`), the final status (`status`), the total running time (`time`), the number of iterations (`iter`), and finally the maximum (`Mmax`) and average (`Mavg`) number of constraints used at each iteration. If any algorithm took more than 600 iterations, we declared the `status` to be a `fail`, and set the time and iteration counts to `Inf`.

In general, the best performance is obtained using rMPC*. To some extent this is to be expected since the base algorithm of rMPC* is a highly efficient primal-dual algorithm, while the others are based on dual only algorithms. The long-step variant of the Dantzig-Ye algorithm (`budas-ls`) also performed well.[4]

4.2 Numerical experiments with the regularized algorithms

In this section, we turn to consider the two regularized algorithms developed in Chapter 3. We introduce a couple of highly degenerate classes of problems, and investigate the performance and qualitative behavior of our algorithms on them. First, we introduce a problem where the level of *rank-degeneracy* can be precisely controlled.

4.2.1 The tube-in-a-cube problem

The procedure for creating a problem in this class is as follows. First, generate $A,\ b,\ y_0 \sim \mathcal{N}(0,1),\ s_0 \sim \mathcal{U}(0,1)$ and normalize the columns of A. Next, project the columns of A onto random $r \leq m$ dimensional subspace, so that A now has

[4]For the random problem, `budas-ls` had to use minor-cycles to increase the constraint set size to 3200, 800, and 800, respectively, in its first three iterations and used $2m = 400$ in the remaining iterations. On the Chebyshev approximation problem in 17 of the 57 iterations, minor-cycle iterations were used that each effectively tripled the constraint size.

prob	alg	status	time	iter	Mmax	Mavg
cheb	rmpc	succ	99.11	29	40400	40400.0
cheb	rdpr	succ	364.89	112	40400	40400.0
cheb	rpdas	fail	Inf	Inf	40400	40400.0
cheb	budas-ss	fail	Inf	Inf	40400	40400.0
cheb	budas-ls	fail	Inf	Inf	40400	40400.0
rand	rmpc	succ	60.50	18	40000	40000.0
rand	rdpr	succ	269.87	82	40000	40000.0
rand	rpdas	succ	67.47	22	40000	40000.0
rand	budas-ss	succ	1117.98	336	40000	40000.0
rand	budas-ls	succ	109.74	33	40000	40000.0
scsd1	rmpc	succ	0.12	10	760	760.0
scsd1	rdpr	succ	0.51	65	760	760.0
scsd1	rpdas	succ	0.08	9	760	760.0
scsd1	budas-ss	succ	0.49	110	760	760.0
scsd1	budas-ls	succ	0.07	17	760	760.0
scsd6	rmpc	succ	0.10	12	1350	1350.0
scsd6	rdpr	succ	0.59	59	1350	1350.0
scsd6	rpdas	succ	0.11	14	1350	1350.0
scsd6	budas-ss	succ	1.92	247	1350	1350.0
scsd6	budas-ls	succ	0.17	20	1350	1350.0
scsd8	rmpc	succ	0.25	10	2750	2750.0
scsd8	rdpr	succ	2.30	61	2750	2750.0
scsd8	rpdas	succ	0.35	14	2750	2750.0
scsd8	budas-ss	succ	9.32	441	2750	2750.0
scsd8	budas-ls	succ	0.57	21	2750	2750.0

Table 4.2: Comparison of algorithms with no constraint reduction.

prob	alg	status	time	iter	Mmax	Mavg
cheb	rmpc	succ	13.35	50	1184	1128.5
cheb	rdpr	fail	Inf	Inf	31427	1616.2
cheb	rpdas	fail	Inf	Inf	15636	2755.8
cheb	budas-ss	succ	52.28	235	1187	1117.0
cheb	budas-ls	succ	17.54	57	3278	1667.1
rand	rmpc	succ	3.31	17	400	400.0
rand	rdpr	succ	30.85	70	400	400.0
rand	rpdas	succ	8.42	51	400	400.0
rand	budas-ss	succ	42.97	271	400	400.0
rand	budas-ls	succ	3.78	19	3200	589.5
scsd1	rmpc	succ	0.09	10	347	340.5
scsd1	rdpr	succ	0.47	63	702	355.9
scsd1	rpdas	succ	0.04	12	345	338.8
scsd1	budas-ss	succ	0.54	105	353	338.3
scsd1	budas-ls	succ	0.10	16	347	338.4
scsd6	rmpc	succ	0.10	12	651	640.3
scsd6	rdpr	succ	0.37	57	652	638.7
scsd6	rpdas	succ	0.09	15	653	639.3
scsd6	budas-ss	succ	1.32	241	656	638.6
scsd6	budas-ls	succ	0.06	19	649	638.6
scsd8	rmpc	succ	0.16	10	1660	1641.7
scsd8	rdpr	succ	1.67	61	2750	1751.1
scsd8	rpdas	succ	0.21	15	1671	1641.7
scsd8	budas-ss	succ	5.34	437	1673	1640.7
scsd8	budas-ls	succ	0.28	19	1675	1639.9

Table 4.3: Comparison of algorithms with constraint reduction.

rank r. Append box constraints $\|y\|_\infty \leq R$ to the dual problem, and add them to A. This makes sure that the problem has a solution, and recovers the condition that $\text{rank}(A) = m$. Finally, set $c = A^{\mathrm{T}} y_0 + s_0$ to ensure (y_0, s_0) is dual strictly feasible, and take $x_0 = e$, the vector of ones. This problem class was named the "tube-in-a-cube" in [TAO06], because of the geometry of the resulting dual feasible region which is illustrated in Figure 4.3 for a 2-dimensional problem (i.e., $m = 2$).

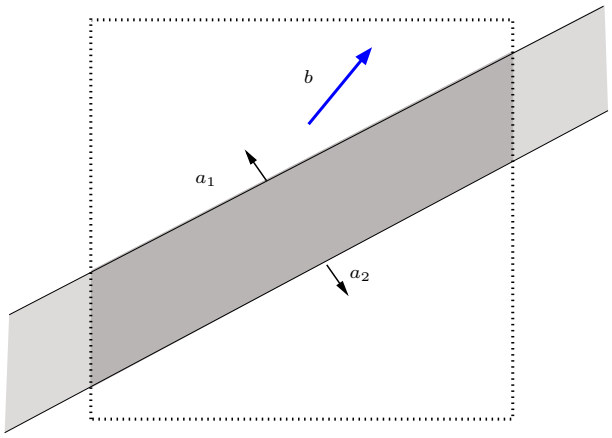

Figure 4.3: Tube-in-a-cube problem example in 2-dimensions. For this problem class, the rank of the constraint matrix A is reduced to a prescribed level r which creates a $k = m - r$ dimensional "tube" in the feasible region ($m = 2$, $r = k = 1$ in the figure). Then a set of bound constraints (the "cube") are added to recover the condition that A have full rank, making the problem solvable.

On this problem we can guess how the kernel step algorithm will behave. Looking at Figure 4.3, it seems intuitively clear that, if we start somewhere in the middle of the feasible region, the first step will be a kernel step that moves the iterate to the top-right part of the feasible set, and the next few iterates will be regular steps that make rapid progress. We can imagine that if the tube is k-dimensional, then we will have to take k kernel steps to get to the proper corner of the cube where the optimal set is. In view of the identification of the kernel step, in section 3.2.2, as a limiting case of a regularized step with vanishing regularization parameter δ, we may expect the regularized algorithm for small $\bar{\delta}$ (from (3.23)) to behave similarly.

In fact, in view of (3.38), we call very long regularized steps with

$$\|\Delta y\| > 0.05\|b\|/\delta, \tag{4.4}$$

"kernel-like" steps, and below we refer to both kernel steps in the kernel step algorithm and kernel-like steps in the regularized algorithm as "kernel-like" steps. The particular form for the threshold comes from (3.38), which suggests it should be proportional to $\|b\|/\delta$, while the multiplier 0.05, has been determined by trial and error to appropriately count the long steps.[5]

The intuitive guess presented above turns out to be a very accurate prediction of the actual behavior of the two algorithms on this problem. Figure 4.4 plots the average number of kernel steps (over 10 runs) for varying degrees of degeneracy (tube-dimension), with "cube" constraint $\|y\|_\infty \leq R$, $R = 10$. The observed behavior is that both algorithms take about k consecutive kernel-like steps, where k is the tube-dimension, during which the objective is increased very rapidly while `termcrit` remains relatively constant, and then switch over to mainly regular-steps which rapidly decrease `termcrit` to tolerance.

Figure 4.5 shows the time to solve and iteration count vs. the fraction of constraint set kept in Q (c.f. Figure 4.1) at each iteration for the regularized rPDAS algorithm, the kernel step rPDAS algorithm, and a regularized variant of rMPC*. Each algorithm used the Most Active rule to select Q. The parameters were set equal for the three algorithms, except, for the rPDAS algorithms, we set $\underline{\xi}^{\max} = 10^{-4}$, while for rMPC* we set it to $\underline{\xi}^{\max} = 10^{-11}$.[6] As expected, rMPC* gives the best

[5] An alternative way to define and count "kernel-like" steps for the regularized rPDAS algorithm is in terms of the size of the projection of b onto $\mathcal{N}(A_Q^{\mathrm{T}})$, namely, whenever this projection is not too small relative to the least norm solution to the normal equations, we count a kernel-like step. This gives very similar results to counting long regularized steps, but is more computationally expensive.

[6] The rPDAS algorithms have numerical difficulties if $\underline{\xi}^{\max}$ is set to a very small value, while we have found that rMPC* works best with it set small, although it is not very sensitive to this parameter on this class of problems.

running time performance, followed by regularized rPDAS and finally kernel step rPDAS, although kernel step rPDAS had fewer iterations than regularized rPDAS. The kernel step method is slow mainly because computing the kernel step, or even determining whether it needs to be computed, is more costly than just solving normal equations. We computed it using Matlab's `null` routine (based on a singular value decomposition) applied to A_Q^{T}, which computes an orthogonal basis for $\mathcal{N}(A_Q^{\mathrm{T}})$. We found that the kernel step algorithm had numerical difficulties if we tried to use `null` on the normal matrix ($\mathcal{N}(M) = \mathcal{N}(A_Q^{\mathrm{T}})$), which is cheaper, but less numerically well behaved. There certainly may be better ways to compute this step, but we will leave such issues for future work.

As in section 4.1.2 (and Figure 4.1), all three algorithms show the benefit of constraint-reduction on this class of problems. (Regularized algorithms rMPC* and rPDAS show the benefit much more on larger tube-in-cube problems, but the solution times for the kernel step rPDAS becomes to long to show on the same graph.)

4.2.2 Random sparse problems

As discussed in section 3.1.1, the sparsity of a constraint matrix A matrix can be correlated with the degree of *rank-degeneracy* that the corresponding LP will exhibit. To quantify this, we introduce a measure of degeneracy $\texttt{degen}(A, N, \sigma_{\mathrm{rank}}) \in [0, 1]$, defined as the average value of $\frac{1}{m}\left(m - \mathrm{rank}(A_{Q_j})\right)$, $j = 1, 2 \ldots N$, where Q_j is a random sample of m columns of A, and the rank is computed as the number of singular values larger than the threshold σ_{rank}. We generated a sequence of random LPs as in section 4.2.1, except that, in the generation of A, we specify the sparsity (or fraction of nonzeros), rather than the explicit tube-dimension. We append bound constraints of the form $\|y\|_\infty \leq R$, again with $R = 10$, to ensure $\mathrm{rank}(A) = m$. Figure 4.6 plots our measure of degeneracy versus the percentage of nonzeros in A,

averaged over 10 problems of size 100×10000, at each sparsity level. Also shown is the average number of kernel-like steps taken by the two algorithms on these problems. We see that there is indeed a strong correlation between degeneracy and sparsity and, as was the case in the tube-in-cube problem, the number of kernel steps that will be taken can be predicted fairly accurately by the estimated "tube-dimension" of the LP computed as $m \times \texttt{degen}(A, N, \sigma_{\text{rank}})$.

4.2.3 Discussion

Both regularized algorithms effectively deal with the rank-degeneracy that constraint-reduced algorithms are subject to. In real-world problems, the rank degeneracy is an issue that must be dealt with, and so it makes sense to incorporate some means of doing so into any practical constraint-reduced algorithm. The kernel step rPDAS algorithm is interesting theoretically, and even outperforms the regularized algorithm on some specially constructed problems (such as the tube-in-cube when the cube is very big compared to the tube-dimension), but its main disadvantage is that the kernel step is more expensive to evaluate than the regularized step. The regularization, on the other hand, adds no additional overhead to rPDAS, and can be easily incorporated into other algorithms, such as rMPC*. Experience shows that using a small regularization in rMPC* does not adversely affect performance on nondegenerate problems, and definitely helps with degenerate ones. While regularization is not a silver bullet solution that will allow for careless application of aggressive constraint-reduction (it is important, in practice, to develop good constraint selection rules for different problem classes), we do definitely recommend its use. In the next chapter, we investigate some real-world problems from the area of digital filter design that result in unbalanced LPs. To solve them we combine the regularized algorithm with the efficient rMPC*, to get a regularized version of rMPC* that, using the constraint selection rule we developed in this chapter for Chebyshev approximation, is shown

to be quite effective on the filter design problems.

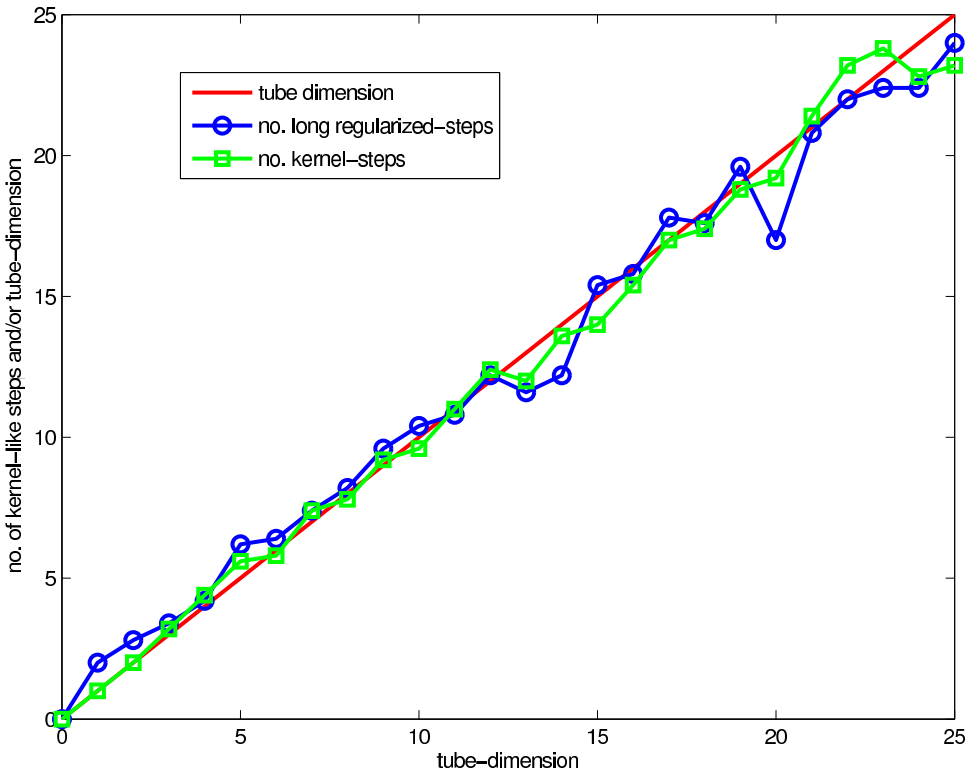

Figure 4.4: Tube-in-cube problem of size 50×2600, "cube" constraint of the form $\|y\|_\infty \leq R$, with $R = 10$, and with varying degrees of degeneracy, namely, with tube-dimension ranging between 0 (nondegenerate) and 25 (highly degenerate). The plot shows the number of kernel steps vs. the tube-dimension for the kernel step algorithm, as well as the number kernel-like directions for the regularized algorithm. For both algorithms we used the Most Active rule with $M = 3m$. The regularized algorithm was run with $\bar{\delta} = 10^{-6}$, and we counted a kernel-like step whenever Δy was "large enough" as defined by (4.4). Examining the results, we see that the tube-dimension is a very good predictor of the number of kernel-like steps that will be taken on the problem. The observed behavior is that both algorithms take about k consecutive kernel-like steps, where k is the tube-dimension, during which the objective is increased rapidly while `termcrit` remains relatively constant, and then switch to mainly over to regular-steps which rapidly decrease `termcrit` to tolerance.

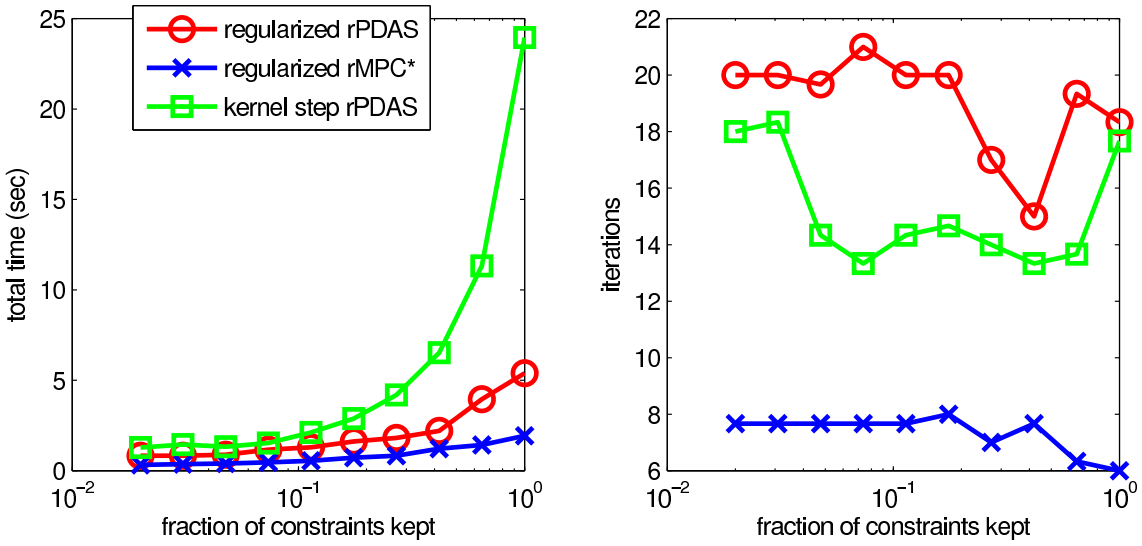

Figure 4.5: Time (in seconds) to solve and iteration count vs. the fraction of the total constraints used at each iteration for an instance of the tube-in-cube problem with $m = 100$, $n = 10000$, tube-dimension $k = 10$, and "cube" constraint of the form $\|y\|_\infty \leq R$, with $R = 10$. Results are shown for the two regularized algorithms and a regularized variant of rMPC*; all use the Most Active rule to select Q. As expected, rMPC* is the best performer. Between the two rPDAS algorithms, the regularized algorithm ran considerably faster, but the kernel step algorithm took fewer iterations. The explanation for why the kernel step method is slow, is that computing the kernel step (or even determining whether it needs to be computed) is costly. We computed it using Matlab's `null` routine (based on a singular value decomposition) applied to A_Q^{T} which, computes an orthogonal basis for $\mathcal{N}(A_Q^{\mathrm{T}})$. We found that the kernel step method had numerical difficulties if it used `null` on the normal matrix, which is cheaper, but less numerically well behaved.

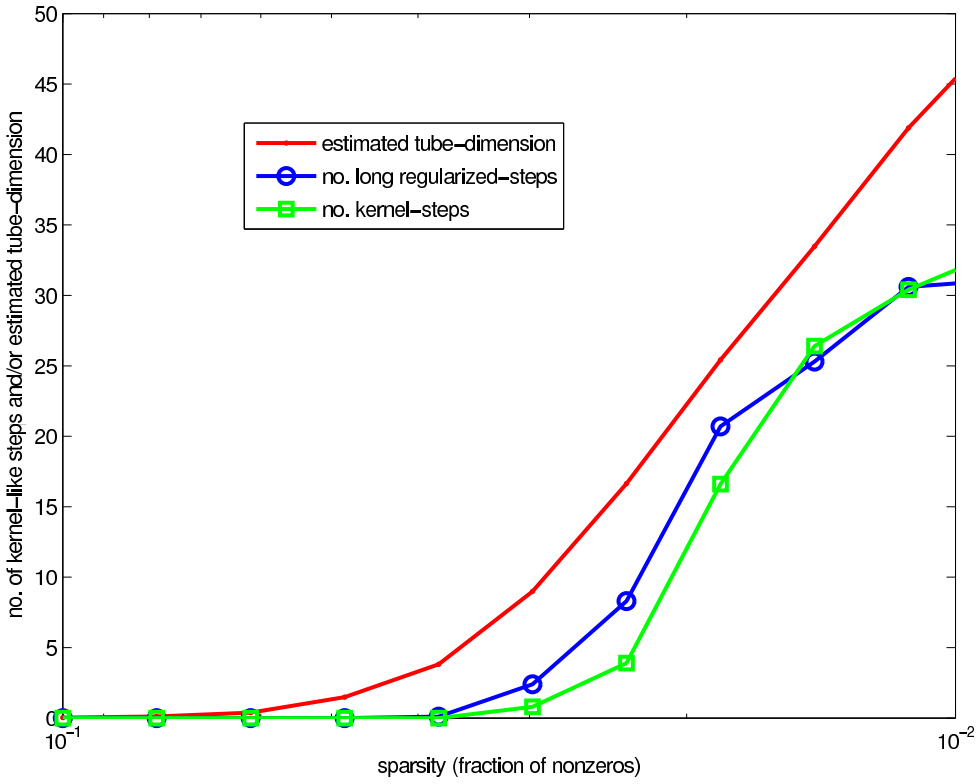

Figure 4.6: Estimated tube-dimension versus percentage of nonzeros in A. The tube-dimension is estimated as $m \times \texttt{degen}(A, N, \sigma_{\text{rank}})$, with $N = 10$ and $\sigma_{\text{rank}} = n \times \texttt{eps}(\|A\|)$, the Matlab default tolerance for rank estimation (`eps` is a built-in Matlab function). Also shown is the number of kernel-like steps taken by the kernel step and regularized rPDAS algorithms vs. the fraction of nonzeros in the sparse constraint matrix A, of a randomly generated sparse LP. (For both algorithms we used the Most Active rule with $M = 3m$.) Down to about 1% nonzeros (note the fraction of nonzeros decreases, so sparsity increases toward the right), the increasing sparsity is accompanied by increasing degeneracy, and the number of kernel-like steps is, again, roughly given by the "tube-dimension"; below 1% density (not shown), the number of kernel-like steps starts to flatten out, and the tube-dimension serves as an upper bound.

Chapter 5

Applications in filter design

Semi-infinite programs (SIP) are optimization problems that have a finite number of variables but an (often uncountably) infinite number of constraints.[1] SIP can be a powerful tool for modeling engineering design problems, with applications including the design of digital filters, antenna array weights, and control systems, e.g., [SN82, HK93, BP95, KM95, WBV98, Pot98, NZ99]. In this chapter, we illustrate the usage of semi-infinite *linear* programming (SILP)—SIPs with linear objective and constraints—in four real-world filter design applications that come from the author's work on guidance, navigation, and control systems at NASA Goddard Space Flight Center. We show that, in each case, these problems can be effectively and efficiently solved by discretizing the constraint set, and then applying the constraint-reduced interior-point methods developed in previous chapters.

5.1 Discretized semi-infinite linear programming

An effective way to solve SILPs is by finely discretizing the constraint set and then applying a linear programming algorithm that can exploit the unbalanced nature of the resulting problem. One such algorithm is the revised primal simplex method

(RPS), described in section 1.3.1; in this chapter, we show that the constraint-reduced interior-point algorithms developed in this dissertation can also be very effective.

First, we briefly discuss a matter of terminology. The problems we treat in this chapter are modeled using the finite variable, infinite constraint form of SILP, which we refer to as the "dual SILP". We prefer this naming convention because, after discretization, our dual SILP is an LP in dual standard form. Many other authors, however, use the opposite convention.

Each of the four problems we consider below can be put into the following framework. We wish to fit the linear model

$$f(\omega) = \sum_{k=0}^{N-1} \phi_k(\omega) y_k + \varepsilon(\omega), \tag{5.1}$$

consisting of a linear combination of the basis functions $\phi_k : \mathbb{R}^M \to \mathbb{R}$, to the target function $f : \mathbb{R}^M \to \mathbb{R}$, by selection of the finite dimensional vector $y \in \mathbb{R}^N$ of variables. The basis and target functions are assumed to depend smoothly on the real vector ω taking values in a compact set $\Omega \subset \mathbb{R}^M$. The difference $\varepsilon(\omega) := f(\omega) - \sum_{k=0}^{N-1} \phi_k(\omega) y_k$, can be taken as a definition of the residual function $\varepsilon : \mathbb{R}^M \to \mathbb{R}$.

The fit is to be done in a Chebyshev or minimax sense, by solving the optimization problem

$$\min_{y \in \mathbb{R}^N} \max_{\omega \in \Omega} W(\omega) |F(\omega) y - f(\omega)|,$$

where we use the matrix-vector notation $F(\omega) y := \sum_{k=0}^{N-1} \phi_k(\omega) y_k$, and have added a real, nonnegative weighting function $W : \mathbb{R}^M \to \mathbb{R}$ to emphasize the errors in different regions of Ω. By introducing the upper bound variable $t \in \mathbb{R}$, we can

[1] Or vice-versa: problems with an infinite number of variables and finite number of constraints are also called SIPs. The two types often arise as the dual optimization problems of one another.

recast this problem as

$$\begin{aligned} &\min_{y,t} \; t \\ &\text{s.t. } W(\omega)|F(\omega)y - f(\omega)| \leq t, \quad \forall\, \omega \in \Omega, \end{aligned} \tag{5.2}$$

with a linear objective and an infinite number of linear constraints.

We will solve these problems by sampling the "constraint" set Ω to obtain a finite set $\Omega_{\mathrm{d}} \subset \Omega$, and then solve the finite linear program that results from replacing Ω with Ω_{d} in (5.2). The solution $(y^*_{\mathrm{d}}, t^*_{\mathrm{d}})$ to the discretized problem will not, in general, satisfy the constraint in (5.2) for all $\omega \in \Omega$. (If $(y^*_{\mathrm{d}}, t^*_{\mathrm{d}})$ is feasible for (5.2), then it is optimal, since the feasible set of the discretized problem contains that of (5.2).) The question then arises as to how finely the constraint set should be sampled. Much of the existing theory on discretization of SIPs focuses on designing a sequence of discretized problems whose solutions converge to the solution to the full problem [HK93, Ree91]. Here, instead, we prefer to solve a single discretized problem, whose solution is, in some sense, nearly optimal for the original problem. We give a simple analysis next that will guide us in the selection of the discretization mesh for our example problems below.

In the following discussion, we assume $\omega \in \Omega \subset \mathbb{R}$, where Ω is an interval of length ν. We assume further that our discretization set Ω_{d} is a regular mesh of L points $\{\omega_j\}_{l=0}^{L-1} \in \Omega$ that satisfies $\min_{\omega_j \in \Omega_{\mathrm{d}}} |\omega - \omega_j| \leq \frac{\nu}{2L}$ for all $\omega \in \Omega$. The question then becomes: How large should L be? To address this question, let us consider the more general SIP

$$\begin{aligned} &\min \; t \\ &\text{s.t. } |g(y, \omega)| \leq t, \; \forall\, \omega \in \Omega, \end{aligned} \tag{5.3}$$

with $g : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}$, and $t \in \mathbb{R}$. Our approach to determining an appropriate L is to set a maximum level ϵ to which we will tolerate violation of the constraints for $\omega \in \Omega \setminus \Omega_{\rm d}$. That is, while at the solution $(y_{\rm d}^*, t_{\rm d}^*)$ to the discretized problem, $|g(y_{\rm d}^*, \omega_j)| \le t_{\rm d}^*$ holds for all $\omega_j \in \Omega_{\rm d}$, we only require

$$|g(y_{\rm d}^*, \omega)| \le t_{\rm d}^* + \epsilon, \quad \forall\, \omega \in \Omega. \tag{5.4}$$

If this holds, then $(y_{\rm d}^*, t_{\rm d}^* + \epsilon)$ is an "ϵ-suboptimal" solution for (5.3), in that, it is feasible (as are all y for large enough t) with value within ϵ of the optimal value t^* of (5.3) (since $t_{\rm d}^* \le t^*$). Suppose R_1 is a Lipschitz constant for $g(\cdot, y_{\rm d}^*)$ on Ω, and let $\omega_i^* = \arg\min_{\omega_j \in \Omega_{\rm d}} |\omega - \omega_j|$, then we have (suppressing dependence on $y = y_{\rm d}^*$ in the notation)

$$|g(\omega)| - t_{\rm d}^* \le |g(\omega)| - |g(\omega_i^*)| \le |g(\omega) - g(\omega_i^*)| \le R_1 |\omega - \omega_i^*| \le R_1 \frac{\nu}{2L}, \tag{5.5}$$

so that choosing

$$L \ge L_1 := \left\lceil \frac{R_1 \nu}{2\epsilon} \right\rceil, \tag{5.6}$$

will imply (5.4). We can improve this bound if we assume further smoothness of g. In considering (5.4), we can restrict attention to ω' that locally maximize $|g(\cdot, y_{\rm d}^*)|$. Let us assume that $t_{\rm d}^* > 0$, $\omega' \in \Omega^\circ$ (the interior of Ω), and that g is continuously differentiable with respect to ω in Ω°, so that we have $\dot g(\omega', y_{\rm d}^*) := \frac{\partial g}{\partial \omega}(\omega', y_{\rm d}^*) = 0$. Take R_2 to be a Lipschitz constant for $\dot g(\cdot, y_{\rm d}^*)$ on Ω° and let $\omega_i^* = \arg\min_{\omega_j \in \Omega_{\rm d}} |\omega' - \omega_j|$. Then we have, for some ω_0 between ω' and ω_i^*,

$$\begin{aligned} |g(\omega')| - t_{\rm d}^* &\le |g(\omega')| - |g(\omega_i^*)| \le |\dot g(\omega_0)||\omega' - \omega_i^*| \\ &\le |\dot g(\omega') - \dot g(\omega_0)||\omega' - \omega_i^*| \le R_2 |\omega' - \omega_i^*|^2 \\ &\le R_2 \frac{\nu^2}{(2L)^2}, \end{aligned} \tag{5.7}$$

so that choosing

$$L \geq L_2 := \left\lceil \frac{\nu}{2}\sqrt{\frac{R_2}{\epsilon}} \right\rceil, \tag{5.8}$$

will imply (5.4) for any $\omega \in \Omega^\circ$, since it holds for any local maximizer ω' of $|g(\cdot, y_{\rm d}^*)|$ on Ω°.[2]

If the smoothness assumption holds, we can use the generally smaller constant L_2 to determine the size of the regular grid. To account for issues at the boundary points of Ω, we can add extra grid points at a distance $\delta = \epsilon/R_1$ from any boundary points. In view of (5.5), this will guarantee that (5.4) holds for all $\omega \in \Omega$.

Before moving on to our applications, we discuss two extensions of the above analysis. First, in some of the problems below, we add side constraints of the form

$$\alpha(\omega) \leq g(y, \omega) \leq \beta(\omega) \quad \forall\, \omega \in \Omega_{\rm side} \tag{5.9}$$

to (5.3), where $\Omega_{\rm side} \subseteq \Omega$. The analysis and bounds derived above still apply (just replace $t_{\rm d}^*$ with $\beta(\omega)$ everywhere), but with the following caveat. If our side constraint is "ϵ-violated", for example if, for some $\omega \in \Omega$, $\beta(\omega) < g(y_{\rm d}^*, \omega) \leq \beta(\omega) + \epsilon$, then we cannot claim that $y_{\rm d}^*$ is ϵ-suboptimal for the original problem in the sense we used above, since now $y_{\rm d}^*$ is infeasible for (5.9), and feasibility is not recovered by simply slackening the upper bound variable $t_{\rm d}^*$ to $t_{\rm d}^* + \epsilon$. However, $y_{\rm d}^*$ could still be thought of as being ϵ-suboptimal if we relax the definition of ϵ-suboptimality so that it only requires ϵ-feasibility.

Second, it also arises below that our semi-infinite variable $\omega \in \Omega \subset \mathbb{R}^M$, with $M > 1$. The above analysis still basically applies if we discretize each dimension of ω independently. This means, however, that the overall size of the discretized constraint set will grow exponentially with M (we will need L^M points if $\Omega = [0, \nu]^M$,

[2]The following related fact is the basis for *local reduction* methods in SIP [HK93]: if a constraint $|g(\omega, y)| \leq t$ is active at a solution (y', t') to the *full SILP* with $t' > 0$, then ω' must be a local maximizer of $|g(\cdot, y')|$, so that under a smoothness assumption, if $\omega' \in \Omega^\circ$, then $\dot{g}(\omega', y') = 0$.

for example). Thus, solving SIPs by fine discretization is subject to the "curse of dimensionality", and so, practically, only low-dimensional semi-infinite variables can be handled (at least based on the above analysis).

5.2 Linear phase, finite-impulse response (FIR) filters

As our first application problem, we consider the design of a complex-coefficient FIR filter with frequency response

$$H(e^{j\omega}) = \sum_{k=0}^{N-1} h_k e^{-j\omega k}, \omega \in [-\pi, \pi]. \tag{5.10}$$

We use a minimax or Chebyshev criterion to approximate a desired response $H_{\mathrm{d}}(e^{j\omega})$, by solving the optimization problem

$$\min \max_{\omega \in [-\pi,\pi]} W(\omega) \left| H(e^{j\omega}) - H_{\mathrm{d}}(e^{j\omega}) \right|, \tag{5.11}$$

where the minimization is with respect to the filter coefficients $\{h_k\}_{k=0}^{N-1}$. Here, $W(\omega)$ is a real, nonnegative weighting function that can be used to indicate the relative importance of the errors in different frequency bands. This problem can be formulated as a minimization problem with linear objective and semi-infinite quadratic or second-order cone constraint[3]

$$\begin{aligned} &\min \ t \\ &\text{s.t. } W(\omega) \left| H(e^{j\omega}) - H_{\mathrm{d}}(e^{j\omega}) \right| \leq t, \quad \forall\, \omega \in [-\pi, \pi]. \end{aligned} \tag{5.12}$$

While, in general, $H(e^{j\omega})$ is a complex function and the constraint in (5.12) is

[3]Constraint reduction can also be beneficial for solving problems with semi-infinite quadratic or second-order-cone constraints, see [JOT10] and sections 7.4 and 7.5 below.

Type I	N odd	$h_k = h^*_{N-1-k}$ for $k = 0, 1 \ldots (N-1)/2$
Type II	N even	$h_k = h^*_{N-1-k}$ for $k = 0, 1 \ldots N/2 - 1$
Type III	N odd	$h_k = -h^*_{N-1-k}$ for $k = 0, 1 \ldots (N-1)/2$
Type IV	N even	$h_k = -h^*_{N-1-k}$ for $k = 0, 1 \ldots N/2 - 1$

Table 5.1: Linear phase FIR symmetry/filter types.

quadratic (after squaring), often we are interested in designing the magnitude of the filter while constraining the phase to be a linear (or affine) function of frequency. The standard method for imposing such constraints reduces (5.12) to an SILP. Linear phase constraints are typically imposed by requiring one of four types of symmetry, summarized in Table 5.1, in the coefficients. The corresponding filters are known as *Type I-IV linear phase FIR filters* [OS99].[4] Under each of these symmetries, the frequency response takes the form

$$H(e^{j\omega}) = A(e^{j\omega})e^{-j\tau\omega},$$

with τ determined by the filter length N and symmetry type, and where $A(e^{j\omega})$ is a real, but not necessarily non-negative, linear function of the real and imaginary components of $\{h_k\}$.

We illustrate this in the case of the Type II linear phase constraint. Define

[4]Such symmetry constraints are sufficient, but not necessary, for $H(e^{j\omega})$ to have linear phase [OS99].

$\tau = (N-1)/2$, so that $N - 1 - \tau = \tau$. Then the frequency response satisfies

$$\begin{aligned} H(e^{j\omega}) &= \sum_{k=0}^{N-1} h_k e^{-j\omega k} = e^{-j\omega\tau} \sum_{k=0}^{N-1} h_k e^{-j\omega(k-\tau)} \\ &= e^{-j\omega\tau} \sum_{k=0}^{\frac{N}{2}-1} h_k e^{-j\omega(k-\tau)} + h_{N-1-k} e^{-j\omega(N-1-k-\tau)} \\ &= e^{-j\omega\tau} \sum_{k=0}^{\frac{N}{2}-1} h_k e^{-j\omega(k-\tau)} + h_k^* e^{j\omega(k-\tau)} \\ &= e^{-j\omega\tau} \sum_{k=0}^{\frac{N}{2}-1} 2\Re\{h_k e^{-j\omega(k-\tau)}\} \\ &= e^{-j\omega\tau} \sum_{k=0}^{\frac{N}{2}-1} 2\alpha_k \cos\omega(k-\tau) + 2\beta_k \sin\omega(k-\tau), \end{aligned}$$

where $h_k = \alpha_k + j\beta_k$. Thus the frequency response

$$H(e^{j\omega}) = e^{-j\omega\tau} A(e^{j\omega})$$

is a pure delay times a real, "magnitude" component

$$A(e^{j\omega}) := \sum_{k=0}^{\frac{N}{2}-1} \alpha_k 2\cos\omega(k-\tau) + \beta_k 2\sin\omega(k-\tau), \tag{5.13}$$

which is linear in (α_k, β_k).

Similar properties hold for the other standard linear phase Type I-IV FIR filters. Because of this form of the frequency response, problem (5.12) becomes a semi-infinite *linear programming* problem in the variables t and (α_k, β_k), $k = 0, 1, \ldots \frac{N}{2} - 1$,

$$\begin{aligned} &\min\ t \\ &\text{s.t. } W(\omega)\left|A(e^{j\omega}) - A_{\mathrm{d}}(e^{j\omega})\right| \le t, \quad \forall\, \omega \in [-\pi, \pi], \end{aligned} \tag{5.14}$$

where $A_{\mathrm{d}}(e^{j\omega})$ is the desired real response (with the phase delay $e^{-j\omega\tau}$ assumed), and $A(e^{j\omega})$ is as in (5.13). The constraint here is indeed linear because, unlike $H(e^{j\omega})$ in (5.12), $A(e^{j\omega})$ is real. Linear side constraints of the form $\alpha(\omega) \leq A(e^{j\omega}) \leq \beta(\omega), \forall\omega \in \Omega_1$, for some $\Omega_1 \subset [-\pi, \pi]$, can also be easily accommodated in (5.14).

Front-end filter for a Global Positioning System (GPS) receiver

In this section we describe the design of a decimation filter used in the signal-processing hardware of a space-based GPS receiver developed by NASA Goddard Space Flight Center (GSFC), called the GSFC-Navigator.

The GPS L1 frequency, Coarse/Acquisition code signal [Off04, ME01] is received and processed by the analog radio frequency (RF) front-end electronics of the receiver, resulting in an intermediate frequency (IF) signal of 2.046MHz bandwidth, centered at 2.556MHz, that is then sampled using a 1.5 bit analog-to-digital converter at a rate of 32.768MHz. The GPS receiver's baseband signal processing blocks require complex (in-phase and quadrature) samples at a rate of 2.048MHz. To accommodate this, a complex coefficient FIR digital filter is applied to suppress the negative frequency image and remove broadband quantization noise. The output of the filter is then decimated by 16 (which causes the signal center-frequency to alias to 0.508MHz) and sent to the baseband signal processing blocks. Figure 5.1 gives a block diagram of the receiver signal path showing the location of the desired decimation filter.

Performance specifications for lowpass, highpass, bandpass, and bandstop filters are usually given by setting a maximum allowable passband ripple and minimum required stopband attenuation level for the magnitude response. For our filter, we would like to achieve at least 30dB of rejection of the stopbands while having passband ripple of no more than $\pm$0.5dB. There is also an image frequency band

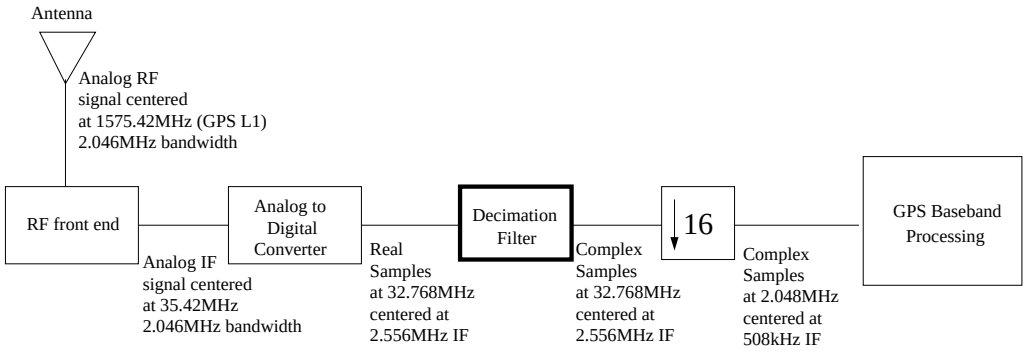

Figure 5.1: GSFC-Navigator GPS receiver signal path with decimation filter highlighted.

number of coefficients	$N = 64$
sampling freq	f_s=65.536MHz
center freq	$\omega_c = 2.556\text{MHz} \times \pi/(f_s/2) = 0.2451\pi$
one sided bandwidth	$B = 1.023\text{MHz} \times \pi/(f_s/2) = 0.0312\pi$
passband low edge	$\omega_\text{l} = \omega_c - B = 0.0468\pi$
passband high edge	$\omega_\text{h} = \omega_c + B = 0.1092\pi$
passband	$[\omega_\text{l}, \omega_\text{h}]$
passband ripple	$\leq 0.5\text{dB}$
transition bandwidth (passband)	$\Delta\omega_\text{p} = (3/N) \times \pi = 0.0469\pi$
transition bandwidth (image)	$\Delta\omega_\text{i} = 0$
stopband	$[-1, \omega_\text{l} - \Delta\omega_\text{p}] \cup [\omega_\text{h} + \Delta\omega_\text{p}, 1]$
stopband magnitude	$\leq -30\text{dB}$
image band	$[-\omega_\text{h}, -\omega_\text{l}]$
image band magnitude	$\leq -60\text{dB}$

Table 5.2: Specification for GSFC-Navigator GPS receiver decimation filter. All frequencies, with the exception of the sampling frequency f_s, are given in units of radians/sample.

$[-\omega_\text{h}, -\omega_\text{l}]$ that we want to reject to a level of -60dB. Table 5.2 summarizes the filter specification, and Figure 5.2 shows it graphically along with an example frequency response that meets all requirements. We will see that the SILP approach to filter design handles this type of specification very naturally, but first we examine an alternative approach.

A traditional approach to filter design, and that which is used in most of the MATLAB filter design tools, is to specify a desired piecewise-linear frequency response as a sequence of K band edges $\{\omega_k\}_{k=1}^K$ and associated amplitudes $\{A(e^{j\omega_k})\}_{k=1}^K$, with transition bands separating any discontinuities in the sequence of amplitudes.

For lowpass, highpass, bandpass, and bandstop filters, the passband and stopband amplitudes are usually chosen to be either 0 or 1, and relative weights are specified for the passbands and stopbands indicating their relative importance. For the GPS filter problem, the frequency bands of interest are

$$[-1, -\omega_{\mathrm{h}} - \Delta\omega_{\mathrm{i}}], \underbrace{[-\omega_{\mathrm{h}}, -\omega_{\mathrm{l}}]}_{\text{image}}, [-\omega_{\mathrm{l}} + \Delta\omega_{\mathrm{i}}, \omega_{\mathrm{l}} - \Delta\omega_{\mathrm{p}}], \underbrace{[\omega_{\mathrm{l}}, \omega_{\mathrm{h}}]}_{\text{passband}}, [\omega_{\mathrm{h}} + \Delta\omega_{\mathrm{p}}, 1],$$

where ω_{l} and ω_{h} are the low and high edges of the passband respectively, and $\Delta\omega_{\mathrm{i}}$ and $\Delta\omega_{\mathrm{p}}$ are the transition bandwidths for the image band and passband. Thus, our sequence of band edges and magnitudes would be given as $\{-1, -\omega_{\mathrm{h}} - \Delta\omega_{\mathrm{i}}, -\omega_{\mathrm{h}}, -\omega_{\mathrm{l}}, -\omega_{\mathrm{l}} + \Delta\omega_{\mathrm{i}}, \omega_{\mathrm{l}} - \Delta\omega_{\mathrm{p}}, \omega_{\mathrm{l}}, \omega_{\mathrm{h}}, \omega_{\mathrm{h}} + \Delta\omega_{\mathrm{p}}, 1\}$ and $\{0, 0, 0, 0, 0, 0, 1, 1, 0, 0\}$, respectively. Then we need to choose a set of weights $\{W_1, W_2, W_3, W_4, W_5\}$ to try to meet the specified ripple and attenuation requirements. Figure 5.3 shows a MATLAB script that uses the Signal Processing Toolbox function `cfirpm` to design complex FIR filter coefficients using a modified Parks-McClellan/Remez exchange algorithm [KM95] to solve problem (5.11). (This routine is, in fact, closely related to the simplex method discussed in section 1.3.1 [BP95, NZ99].) Since the optimal error in (5.11) is not known a priori, this approach can require significant amount of solving, adjusting the weights, and resolving, to meet hard passband-ripple/stopband-attenuation specifications, especially if the specifications are tight.[5]

Contrast this with the SILP approach, which essentially amounts to just writ-

[5]The appropriate weights can be roughly estimated a priori, since the ratio of errors in each band will be the ratios of the respective weights in the solution to (5.11).

ing down the specification

$$\begin{aligned} &\min\ t \\ \text{s.t. } &\left|A(e^{j\omega})\right| \leq t,\ \forall\, \omega \in \Omega_{\text{stop}} \\ &\left|A(e^{j\omega})\right| \leq -60\text{dB},\ \forall\, \omega \in \Omega_{\text{image}}, \\ -0.5\text{dB} \leq &A(e^{j\omega}) \leq 0.5\text{dB},\ \forall\, \omega \in \Omega_{\text{pass}}, \end{aligned} \tag{5.15}$$

with $\Omega_{\text{stop}} := [-1, \omega_{\text{l}} - \Delta\omega_{\text{p}}] \cup [\omega_{\text{h}} + \Delta\omega_{\text{p}}, 1]$, $\Omega_{\text{image}} := [-\omega_{\text{h}}, -\omega_{\text{l}}]$, and $\Omega_{\text{pass}} := [\omega_{\text{l}}, \omega_{\text{h}}]$. Here we have chosen to impose the passband ripple and image band attenuation requirements as constraints, and optimize the stopband attenuation, but we could have optimized the image rejection with fixed stopband, etc. A nice feature of this approach is that linear programming algorithms will usually detect infeasibility, and so in FIR filter design problems, we will know when we need to increase the length N of the filter,[6] whereas solving only (5.11), without the side constraints (the Parks-McClellan approach) this may not be at all obvious.

We will approximately solve this SILP by discretizing the frequency interval $[-\pi, \pi]$. Here, referring back to the discussion of section 5.1, we estimate the Lipschitz constants R_1 and R_2 for this problem and determine how finely the infinite constraint set in (5.14) should be discretized. For this problem, we have

$$g(\omega, y) = A(e^{j\omega}) - A_{\text{d}}(e^{j\omega}),$$

where y is the vector of filter coefficients $\{h_k\}$. Since $A_{\text{d}}(e^{j\omega})$ is piecewise constant, and

$$A(e^{j\omega}) := \sum_{k=0}^{\frac{N}{2}-1} \alpha_k 2\cos\omega(k-\tau) + \beta_k 2\sin\omega(k-\tau),$$

[6] For rMPC*, the phase-one problem (2.17), will terminate with $t > 0$ if the original problem is infeasible.

we have, with $\dot{A}(e^{j\omega})$ and $\ddot{A}(e^{j\omega})$ the first and second derivatives of A with respect to ω,

$$|\dot{g}(\omega, y)| = |\dot{A}(e^{j\omega})| \leq \sum_{k=0}^{\frac{N}{2}-1} 2(k-\tau)(|\alpha_k| + |\beta_k|) \leq (N-2) \max_k |k-\tau|(|\alpha_k| + |\beta_k|)$$

and

$$\begin{aligned} |\ddot{g}(\omega, y)| = |\ddot{A}(e^{j\omega})| &\leq \sum_{k=0}^{\frac{N}{2}-1} 2(k-\tau)^2(|\alpha_k| + |\beta_k|) \\ &\leq 2 \max_k |k-\tau|(|\alpha_k| + |\beta_k|) \sum_{k=0}^{\frac{N}{2}-1} |k-\tau| \\ &= \frac{N^2}{4} \max |k-\tau|(|\alpha_k| + |\beta_k|). \end{aligned} \tag{5.16}$$

For this problem, the quantity $\max_k |k-\tau|(|\alpha_k| + |\beta_k|)$ is, in practice, always bounded by 1.0 (this was discovered by numerical experimentation, and could be strictly enforced by means of additional linear constraints), so we can take

$$R_1 = N, \quad \text{and} \quad R_2 = \frac{1}{2}\frac{N^2}{4} = \frac{N^2}{8}.$$

To determine an appropriate L we next need to choose a value for ϵ. Here we will take $\epsilon = 0.1t_{\mathrm{d}}^* = 10^{-3}$, where we have used the rough prior estimate $t_{\mathrm{d}}^* \approx 10^{-2}$ of the optimal value of (5.15). In this way, the solution to our discretized problem will provide a solution to the full problem that is, at worst, about 10% suboptimal.[7] Plugging these values of R_1, R_2, and ϵ, along with $N = 64$, into the formulas (5.6) and (5.8), we get $L_1 \approx 200000$ and $L_2 \approx 2250$.

Since this problem is relatively small (small being defined as: easily solved on the author's laptop), we lean toward the conservative side, and discretize the

[7] We recognize that this is not totally satisfactory a priori, but once the discretized problem is solved and t_{d}^* is determined, we will know that we have, at worst, a $100 \times \epsilon/t_{\mathrm{d}}^*$ % suboptimal solution for the full SILP (if R_1, R_2 are valid).

interval $[-\pi, \pi]$ using an L=20000 point uniform discretization grid. This results in an LP that has $N + 1 = 65$ variables and 39372 constraints. The problem is easily solved using (a regularized version of) algorithm rMPC* and the results are shown along with the specification in Figure 5.2. The numerical performance of rMPC*, with and without constraint reduction, and that of the revised primal simplex (RPS) method is given along with their performance on our other application problems in Table 5.4 at the end of the chapter.

The flexibility and naturalness of the linear programming approach comes at the cost of an increased computational burden. (For example, the routine `cfirpm` solves (5.11) in under $0.1s$ while rMPC* requires a few seconds. While it is likely that rMPC* could be significantly sped up through code optimization, it is unlikely that it will ever run as fast on (5.11) as `cfirpm`.) For design problems, which may only need to be run a handful of times, the benefits may outweigh the loss in speed. For the interior-point approach to solving the linear programs above, constraint-reduction techniques can significantly mitigate this drawback and offer an alternative to the RPS method. See section 5.6 below for further discussion of numerical performance.

5.3 FIR magnitude response design and spectral factorization

We saw above that the linear phase conditions make the "magnitude" response into a real, linear function $A(e^{j\omega})$ of the design variables (the filter coefficients), and allow (5.11) to be solved as an LP. While linear phase can be a desirable feature, especially when the signals being processed carry information in the phase, for other problems, the magnitude response of the filter is really all that is of interest, and we will be able to achieve design goals on a magnitude response with a lower order

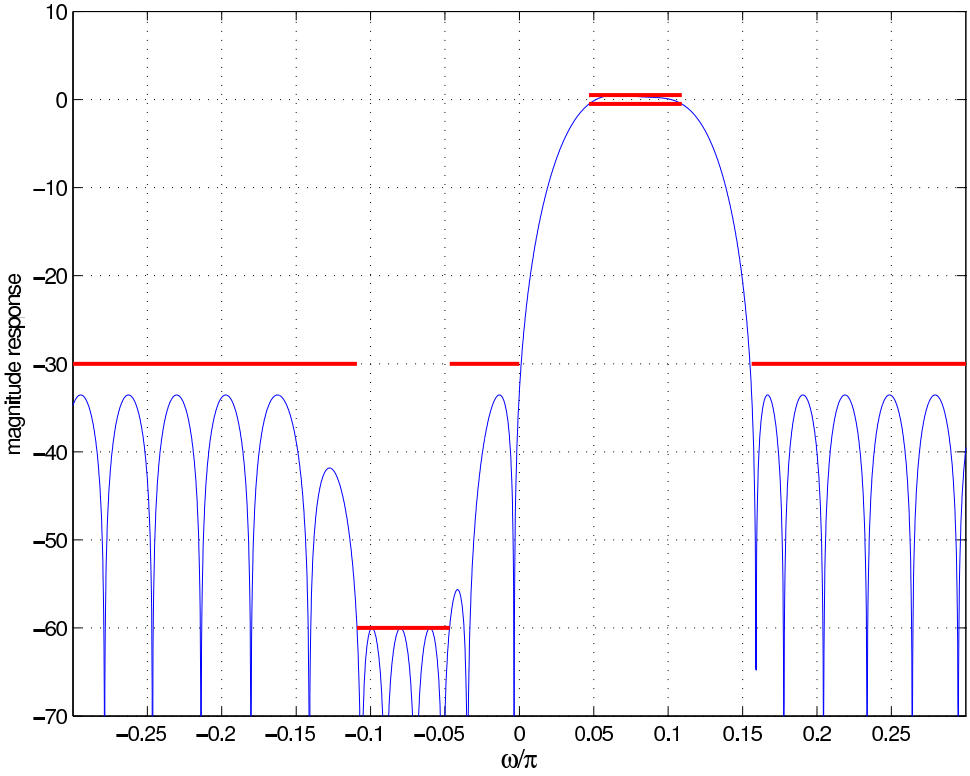

Figure 5.2: Front-end filter specification (in dB) given by thick red horizontal bars. Also shown is the magnitude response of the optimal solution to (5.15), computed by rMPC*, which meets the complete specification. Note that only frequencies from $[-0.3\pi, 0.3\pi]$ are shown; the spec for $|\omega| > 0.3\pi$ is just a continuation of the 30dB attenuation requirement.

filter if we do not impose a linear phase constraint. (For still other problems, linear phase is simply incompatible with the design goals, see section 5.4.) Here, we want to treat the problem

$$\begin{aligned} &\min\ t \\ &\text{s.t. } W(\omega)|H(e^{j\omega})| \leq t, \\ &\alpha(\omega) \leq |H(e^{j\omega})| \leq \beta(\omega), \\ &\forall\ \omega \in [-\pi, \pi]. \end{aligned} \tag{5.17}$$

As noted previously, in general, constraints on the magnitude response of a complex transfer function are not linear, but rather quadratic or second-order-

```
N=64;                    %filter length
fs=65.536;               %sampling freq (in MHz)
wc=2.556*pi/(fs/2);      %center freq (normalized)
bw=1.023*pi/(fs/2);      %signal bandwidth one-sided
wh=wc+bw;                %high side signal band edge
wl=wc-bw;                %low side signal band edge
dwp=pi*3/Ntaps;          %trans. bandwidth adjacent to passband
dwi=eps;                 %trans. bandwidth adjacent to image band

f=[-1,-wh-dwi,-wh,-wl,-wl+dwi, wl-dwp, wl,wh,wh+dwp,1];
a=[0,0,0,0,0,0,1,1,0,0];
w=[1,32,1,1,1];

%use (complex) Parks-McClellan alg. to find minimax optimal
%N-tap FIR filter
h=cfirpm(N-1,f/pi,a,w).';
```

Figure 5.3: MATLAB code for Parks-McClellan based filter design. Since the stopband attenuation specification is -30dB and the image band is -60dB, the error ratio is $10^{-30/20}/10^{-60/20} \approx 32$, so we weigh the image band 32 times more heavily than the nominal stopband.

cone constraints. Rather than imposing linear phase conditions, a different way of making (5.17) into a linear program was put forth by Wu, Boyd and Vandenberghe in [WBV98]. Their idea was to design the magnitude response $H(e^{j\omega})$ of an FIR filter through its *autocorrelation coefficients* $\{r_k\}$ defined by

$$r_k := \sum_{l=-\infty}^{\infty} h_{l+k} h_l^* \tag{5.18}$$

Note that $\{r_k\}$ is the usual autocorrelation sequence of the output $\{y_k\}$ of the filter $H(e^{j\omega})$ when the input $\{x_k\}$ is unit-variance white noise, i.e., $r_k = E(y_{n+k}\ y_n^*)$, where $E(x_{n+k}\ x_n^*) = \delta_k$, E is expectation on the associated probability space, and $\{\delta_k\}$ is the Kronecker delta sequence. Note, also, that $\{r_k\}$ is conjugate symmetric, i.e., $r_k = r_{-k}^*$, which implies, in a similar way as the Type I and II linear phase symmetries do, that the discrete-time Fourier transform (DTFT) $S(e^{j\omega})$ of $\{r_k\}$ is

real and expressible as

$$S(e^{j\omega}) = \sum_{k=-\infty}^{\infty} r_k e^{-j\omega k} = r_0 + \sum_{k=1}^{N-1} \alpha_k 2\cos(\omega k) + \beta_k 2\sin(\omega k). \tag{5.19}$$

Here, $r_k = \alpha_k + j\beta_k$ for $k = \{1, 2, \ldots N-1\}$, and r_0 is real. In the Fourier domain, expression (5.18) translates to

$$S(e^{j\omega}) = |H(e^{j\omega})|^2, \tag{5.20}$$

which shows that $S(e^{j\omega}) \geq 0$ for any autocorrelation sequence coming from an FIR filter $\{h_k\}$. Conversely, if $S(e^{j\omega}) > 0$, and has the representation (5.19) for finite N, then there exists $\{h_k\}$ such that (5.18) hold. That is, there is a FIR filter of length N that has $\{r_k\}$ as its autocorrelation sequence, and if zero-mean white noise is fed as input to this filter, the output will have the specified autocorrelation and power spectrum.[8] The process of recovering a suitable (usually meaning causal and minimum phase) $H(e^{j\omega})$ from $S(e^{j\omega})$ is called *spectral factorization*.

Following [WBV98], our approach to solving (5.17) will be to first solve the linear program

$$\begin{aligned} &\min\ t \\ &\text{s.t. } W^2(\omega) S(e^{j\omega}) \leq t, \quad \forall\, \omega \in \Omega_2, \\ \alpha(\omega)^2 \leq &S(e^{j\omega}) \leq \beta(\omega)^2, \quad \forall\, \omega \in \Omega_1, \\ &S(e^{j\omega}) \geq \delta, \quad \forall\, \omega \in [-\pi, \pi], \end{aligned} \tag{5.21}$$

[8] This is a special case, for finite N, of two basic facts from the theory of second-order stationary random processes: 1) that autocorrelation sequences (positive semi-definite sequences) correspond to nonnegative Fourier transforms (power spectra) which is known as the Wiener-Khintchine theorem or Herglotz Lemma [PP02, Bre00], and 2) that under weak conditions, various *spectral factorization* theorems guarantee that a nonnegative power spectrum can be factored as in (5.20), and viewed as arising as the output spectrum of white noise passing through a minimum-phase, linear time-invariant filter. See, for example, [PP02, Sch91].

with $\delta > 0$, for the real and imaginary components of the autocorrelation coefficients, and then recover filter coefficients by inverting definition (5.18) or equivalently (5.20). The constraint $S(e^{j\omega}) \geq \delta, \forall\ \omega \in [-\pi, \pi]$, that has been added in (5.21), guarantees that a spectral factor $H(e^{j\omega})$ exists, and setting $\delta > 0$ allows for simplified techniques of spectral factorization. The appendix of [WBV98] gives a brief review of spectral factorization techniques, including an efficient technique based on the Fast Fourier Transform, that we have used in our numerical experiments.

A useful application of this approach is in the design of FIR filters to synthesize random noise with a given power spectrum. Such noise is generated by feeding zero mean, unit variance white noise into the filter $H(e^{j\omega})$ obtained from spectral factorization of the optimized $\{r_k\}$ or $S(e^{j\omega})$ from problem (5.21). A particular problem of considerable interest to the author's NASA work is in the synthesis of realistic phase noise based on oscillator vendor phase noise specifications. This noise can then, for example, be used in communication receiver system simulations to investigate the effect on system performance of the receiver and transmitter oscillators. Table 5.3 shows a typical specification of phase noise for a quality temperature compensated crystal oscillator (TCXO) provided by an oscillator vendor. We now consider the design of an FIR filter that synthesizes noise with a power spectrum within $\pm$1dB of this specification.

By interpolating the data in Table 5.3 we arrive at a desired response $S_{\rm d}(e^{j\omega})$, which we would like to approximate within an error that is most naturally specified in decibels. Ideally we would like to minimize an upper bound t on

$$|10 \log_{10} S(e^{j\omega}) - 10 \log_{10} S_{\rm d}(e^{j\omega})| = \left| 10 \log_{10} \frac{S(e^{j\omega})}{S_{\rm d}(e^{j\omega})} \right|,$$

freq offset	$\mathcal{L}(f)$ (dBc/Hz) at 10MHz	$S_x(f)$ (dB-s^2/Hz)	$S_d(e^{j2\pi f})$ (dB-s^2/(rad/sample))
1Hz	-90	-249	-227
10Hz	-100	-259	-237
100Hz	-125	-284	-262
1kHz	-140	-299	-277
10kHz	-145	-304	-282
100kHz	-145	-304	-282
1MHz	-145	-304	-282
>1MHz	-145	-304	-282

Table 5.3: Typical phase noise specification for a 10MHz temperature compensated crystal oscillator (TCXO). The phase noise specification provided in the first column is the "single-side-band" phase noise power density relative to the carrier power. The units are dBc/Hz: the power density in 1Hz bands at different frequency offsets from the carrier, normalized by the total carrier power. This quantity is typically given the symbol $\mathcal{L}(f)$, and is the standard phase noise metric provided by oscillator vendors. The second column gives the power spectrum $S_x(f)$ of the phase noise random process $x(t)$ itself, which has units of seconds. The relationship between these two is $S_x(f) = \mathcal{L}(f)/2(2\pi f_0)^2$, where here the carrier frequency $f_0 = 10$MHz. See, for example, [Vig99]. Thus, column two is offset from column one by $-10\log_{10}(2(2\pi 10^6)^2) = -159$dB. Our simulation will generate the $x(t)$ process at a $T_s = 0.001s$ sampling time and run for 10 seconds, so it is appropriate to try to approximate the phase noise power spectrum between 1Hz and $f_s/2 = 500$Hz. Column three gives $S_{\rm d}(e^{j\omega})$, our desired approximation target, which has units of s^2/(rad/sample), and is obtained from column two by dividing by $2\pi T_s$. Thus, column three is offset from column two by $-10\log 10(2\pi T_s) = 22$dB.

that is, we will minimize t subject to

$$-t \leq 10 \log_{10} \frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} \leq t,$$

or equivalently, minimize v subject to

$$\frac{1}{v} \leq \frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} \leq v. \tag{5.22}$$

Unfortunately, the left hand inequality in (5.22) is not linear (although it is convex and can be converted to a second-order-cone constraint [LB97]). Since we are restricting ourselves to the use of linear constraints, two options are: 1) replace the variable v with a fixed value, e.g. 1dB, and simply require feasibility in the constraint, or 2) define $v := 1+u$, and use the approximation $1/v = 1/(1+u) \approx (1-u)$ to approximately solve (5.22) by instead minimizing u subject to

$$1 - u \leq \frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} \leq 1 + u, \tag{5.23}$$

or equivalently, subject to

$$10 \log_{10}(1-u) \leq 10 \log_{10} \frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} \leq 10 \log_{10}(1+u).$$

If our fitting error is about 1dB or less, the error in the approximation ($10 \log_{10}(1-u)$ in place of $-10 \log_{10} u$) is small, and the difference between (5.22) and (5.23) is quite tolerable.

Back to our example, we determine $S_{\mathrm{d}}(e^{j\omega})$ by linearly interpolating the values in Table 5.3 on a log-log scale, and then, using a combination of the two options

discussed above, we solve

$$\begin{aligned} &\min\ u, \quad \text{s.t.} \\ -u \leq &\frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} - 1 \leq u, \quad \forall\, \omega \in [0,\pi], \\ -1\mathrm{dB} \leq &\frac{S(e^{j\omega})}{S_{\mathrm{d}}(e^{j\omega})} \leq 1\mathrm{dB}, \quad \forall\, \omega \in [0,\pi], \\ &S(e^{j\omega}) \geq \delta, \quad \forall\, \omega \in [0,\pi], \end{aligned} \tag{5.24}$$

where $S(e^{j\omega})$ is as in (5.19). (The final set of constraints $S(e^{j\omega}) \geq \delta$ with $\delta > 0$ are redundant for small δ given the lower bound in the second set of constraints, so in our numerical formulation we actually omit it.) Here, we look for a size $N = 251$ filter with real coefficients, so we discretize $[0, \pi]$ (rather than $[-\pi, \pi]$) using $2^{15} = 32768$ regularly spaced points.[9] The resulting problem is of size 252×163840. Numerical performance is discussed in section 5.6 and summarized in Table 5.4. The optimized filter is shown in Figures 5.4 and 5.5.

5.4 Arbitrary complex response FIR Chebyshev design

In this section, we show that even though the general complex Chebyshev FIR filter approximation problem (where we are interested in approximating a target

[9]We estimate the constants $R_1 \approx 100$, $R_2 \approx 10^4$ (see section 5.1) using some rough approximations and numerical tests. Then, choosing $\epsilon = 10^{-3}$ (based on an estimate of $0.1t^*$), we get from (5.6) and (5.8), $L_1 \approx 3 \times 10^5$, $L_2 \approx 10^4$. So we expect $L = 2^{15}$ should give a sufficiently fine discretization mesh, and the power-of-two value is convenient for the subsequent FFT based spectral factorization.

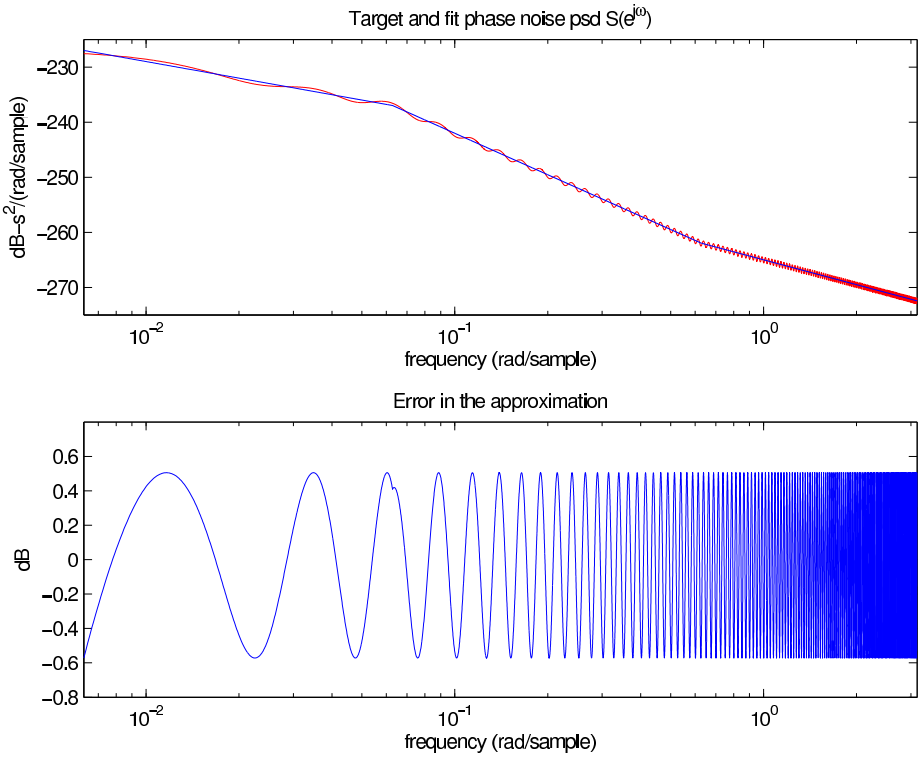

Figure 5.4: FIR magnitude response design problem. Phase noise specification and fit, along with the approximation error which is to about ± 0.6dB.

magnitude *and phase* response)

$$\begin{aligned} &\min \ t \\ &\text{s.t.} \ \left|H(e^{j\omega}) - H_{\rm d}(e^{j\omega})\right| \leq t, \quad \forall\, \omega \in [-\pi, \pi], \end{aligned} \tag{5.25}$$

with $H(e^{j\omega}) = \sum_{k=0}^{N-1} h_k e^{-j\omega k}$, has a semi-infinite quadratic or second-order-cone constraint, it can be formulated *equivalently* as an SILP.[10] A simple property of complex numbers will allow us to do this: if z is complex and r is a positive real constant, then

[10]It is a basic fact in convex analysis that all convex optimization problems can be formulated as SILPs, e.g., using the concept of the *support function* of a convex set [HUL00]. Unfortunately, the general formulation is probably only of theoretical interest, since the resulting SILP has a high-dimensional semi-infinite variable.

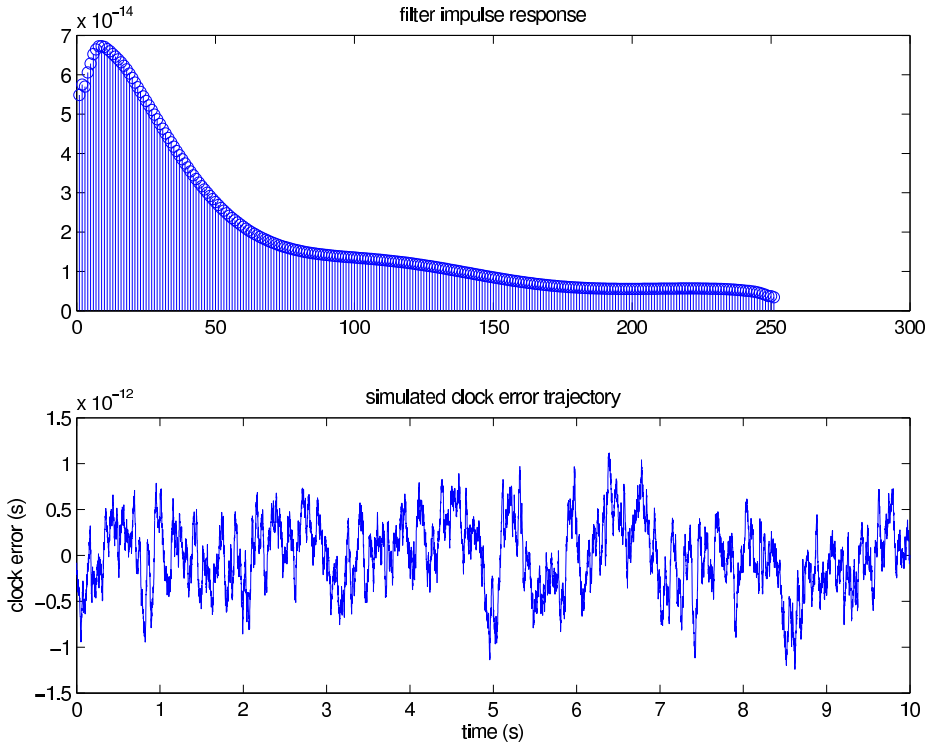

Figure 5.5: FIR magnitude response design problem. The top plot shows the impulse response of the phase noise synthesis filter, which was computed by spectral factorization of the fitted power spectrum. The bottom plot shows a simulated phase noise trajectory.

$$|z| \leq r \quad \Longleftrightarrow \quad \Re\{ze^{j\theta}\} \leq r \quad \forall\ \theta \in [0, 2\pi],$$

where $\Re$ indicates the real part of a complex number. In [BP95], the authors refer to this fact as the *real rotation theorem*. Thus, (5.25) has the equivalent formulation

$$\begin{aligned} &\min\ t \\ &\text{s.t. } \Re\left\{\left(H(e^{j\omega}) - H_{\mathrm{d}}(e^{j\omega})\right) e^{j\theta}\right\} \leq t, \\ &\forall\ \omega \in [-\pi, \pi],\ \theta \in [0, 2\pi], \end{aligned} \tag{5.26}$$

in which the constraints are indeed linear, albeit with index set now of dimension two. This technique was attributed to [SN82] in [BP95], where it was used in filter design problems (see also [NZ99]).

Minimax design of a linear prediction filter

As an application of the above, consider the minimax design of a linear prediction filter in the frequency domain. Our filter will have real coefficients, so the frequency band of interest is $[0, \pi]$, rather than $[-\pi, \pi]$. We define the target response to be the ideal, non-causal one sample advance

$$H_{\mathrm{d}}(e^{j\omega}) := e^{j\omega}.$$

Of course, we cannot accurately approximate a non-causal filter with a causal FIR filter over $\omega \in [0, \pi]$, but we can over a limited band of frequencies, say $[0, 0.35\pi]$. One way to achieve this is by solving the linear program

$$\begin{aligned} &\min\ t \\ &\text{s.t. } W(\omega)\Re\left\{\left(H(e^{j\omega}) - e^{j\omega}\right)e^{j\theta}\right\} \le t, \\ &\forall\, \omega \in [0, \pi],\ \theta \in [0, 2\pi], \end{aligned} \tag{5.27}$$

using the real weighting function $W(\omega) := M$ on $[0, 0.35\pi]$, and $W(e^{j\omega}) := 1$ on $(0.35\pi, \pi]$, where M is a large number. We can also effectively take $M = \infty$, by setting $W(\omega) = 1$ and reducing the range of ω in (5.27) to $[0, 0.35\pi]$. However, in this case, we will need to add some regularizing constraints such as $|h_k| \le R$, for some $R > 0$, in order for the problem to have a solution. Such constraints can actually have practical benefit: they can be used to limit the gain of the filter away from the band of interest, which can serve to make it perform more robustly when the input signal is not strictly bandlimited to $[0, 0.35\pi]$.

We discretize $[0, \pi]$ with 3000 points and $\theta \in [0, 2\pi]$ with 100 points and look for a filter with 25 taps.[11] The resulting LP is of size 26×105050. Numerical performance is discussed in section 5.6 and summarized in Table 5.4 at the end of

this chapter. The solution to this problem gives a linear prediction filter that works quite well for any signal bandlimited to $[0, 0.35\pi]$. Figures 5.6 and 5.8 shows the filter design errors and impulse response, and Figure 5.7 shows the performance on white noise bandlimited to 0.35π.

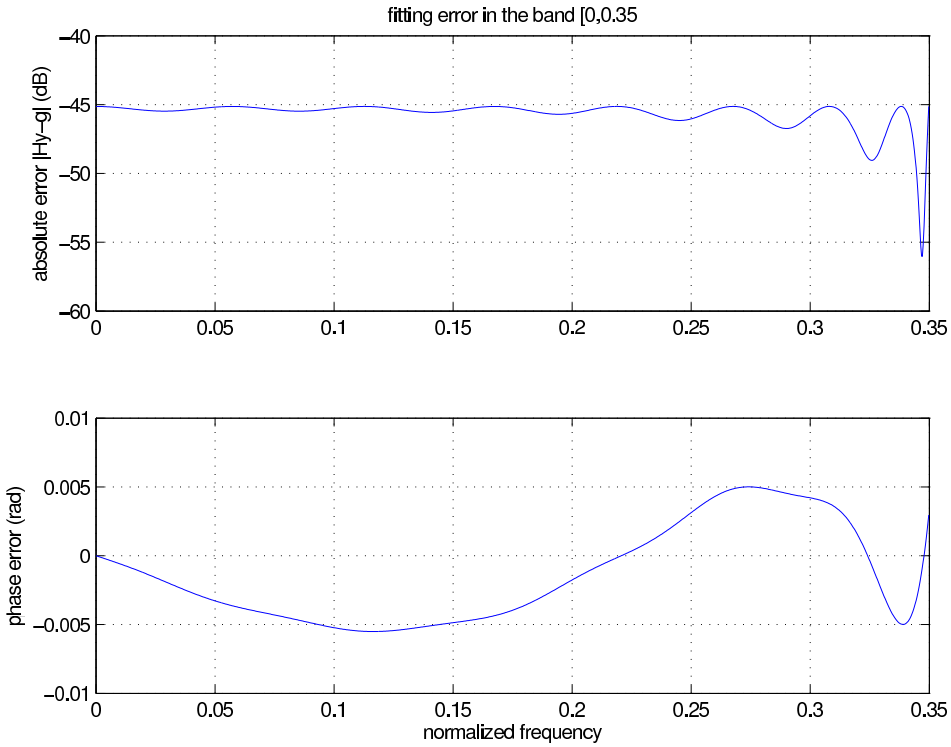

Figure 5.6: Linear prediction filter designed by minimax approximation of the non-causal one-sample advance $H(e^{j\omega}) = e^{j\omega}$ on the interval $[0, 0.35\pi]$. The plots show the absolute error magnitude (top) and phase error (bottom). Above $\omega = 0.35\pi$, of course, the causal filter deviates significantly from the non-causal target. We have, however, prevented wild behavior above $\omega = 0.35\pi$ through the regularizing constraints $|h_k| \leq R$.

5.5 Spatial filters for antenna array beam-steering

Finally, we look at the problem of filter coefficient design for the array signal processor of Figure 5.9. Here we have N sensors located at coordinates $p^l \in R^3$,

[11] Again using some rough approximations and numerical estimates, using (5.6) and (5.8), we determine $L_2 \approx 3000$, $L_1 \approx 3 \times 10^6$. The "$g$" function cannot vary nearly as rapidly in θ as it does in ω, so we only use 100 points to discretize the range of θ.

$l = 0, 1, \ldots N-1$, each independently filtered by a complex FIR filter with response $H_l(e^{j\omega})$ and summed. The problem is to optimize the array filter coefficients to achieve a desired array beam pattern $B(k, \omega)$, defined as the complex response of the array to a plane wave

$$e^{j(k^{\mathrm{T}}x - \omega t)}$$

with frequency ω and wavenumber vector $k \in \mathbb{R}^3$. The beam pattern is the spatial extension of the frequency response which characterizes a linear time invariant system by its response to the complex exponentials $e^{j\omega t}$.

There is a vast literature on weight design and a host of traditional methods based on FFTs and windowing techniques [Tre02], but more recently, optimization based approaches (other than least squares) have been put forth seriously and advocated for their generality [LB97] [WBV98].

When the signals of interest are "narrowband",[12] and we are primarily interested in filtering the input signals with respect to their spatial dimension k, the array processor can restrict $H_l(\omega) = w_l$, a single complex number. In this way the beam pattern of interest is

$$B(k) = \sum_{k=0}^{N-1} w_l e^{k^{\mathrm{T}} p^l},$$

again a complex linear function of the real and imaginary components of w. Then the problem of interest is to optimize the weights so that the beam pattern has desirable properties, i.e., $B(k)$ matches a desired profile. Using a minimax criterion, we have the familiar looking formulation

$$\begin{aligned} &\min\ t \\ &\text{s.t. } |B(k) - B_d(k)| \leq t, \\ &\quad \forall\ k \in K, \end{aligned} \tag{5.28}$$

where B_d is some desired beam pattern and K is the set of wavenumber vectors on which we want $B(k)$ to approximate $B_d(k)$.

This problem is quite similar to the FIR design problem, except now the semi-infinite variable is two-dimensional. (Although $k \in \mathbb{R}^3$, the wave equation constrains its magnitude $|k| = \frac{\omega}{c} = \frac{2\pi}{\lambda}$, so that it is characterized by its direction.) When the locations of the array elements are on a line and regularly spaced—the so called uniform linear array (ULA)—then an exact analogy holds with the design of FIR filters. For example, symmetry conditions analogous to the linear phase conditions of section 5.2 can be imposed on the weight vector (filter coefficients) to make $B(k) = e^{jk^{\mathrm{T}}r}A(k)$ with $A(k)$ real and $r \in \mathbb{R}^3$ constant. In the case of a ULA, the problem formulation of section 5.2 can be used. Here, we instead assume there are $N = 49$ elements, arranged in a rectangular grid of nominal spacing $\frac{\lambda}{2}$ (we set $\lambda = 19$cm, the wavelength of the GPS L1 carrier) about the origin of the $x-y$ plane. The exact positions $\{p^i\}_{i=0}^{N-1} = \{(p_x^i, p_y^i, 0)\}_{i=0}^{N-1}$, are perturbed in their $x-y$ coordinates by random additive Gaussian noise with covariance matrix $\frac{\lambda}{32}I_2$. However, the positions are assumed to be known.

Using $k = -\frac{2\pi}{\lambda}u$, with $u \in \mathbb{R}^3$ a unit vector, and reparameterizing, the beam pattern becomes

$$B(u) = \sum_{l=0}^{N-1} w_l e^{j\frac{2\pi}{\lambda}(p_x^l u_x + p_y^l u_y)}.$$

The parameter u is the vector of direction cosines for the direction-of-arrival of the plane wave, and is related to the elevation ϕ measured down from the z-axis, and azimuth θ measured clockwise from the positive x-axis, by $u_x = \sin\phi\cos\theta$, $u_y = \sin\phi\sin\theta$. A target beam pattern is shown in Figure 5.10 as a function of the x and y components of the unit vector u. As is traditional in the antenna array literature, we show the beam pattern for (u_x, u_y) in the square $[-1, 1] \times [-1, 1]$, but

[12] A signal is considered narrowband if its information bandwidth is very small compared to the inverse propagation delay across the array.

the values of $B(u)$ for $\|u\| > 1$ are irrelevant, as they do not correspond to physical directions of arrival (in the literature this is referred to as the "invisible region"). In this example problem, we are asking for peak gain in a circular region of radius 0.2 around $(u_x, u_y) = (-0.5, 0)$, and strong null of the same shape and size at $(0, -0.5)$, and zero gain elsewhere.[13] The square $[-1, 1] \times [-1, 1]$ in u space is discretized with a grid of 100×100 points.[14]

As in the FIR design, we add transition (or "don't-care") regions around discontinuities in the target pattern and remove the invisible region $\|u\| > 1$ from consideration by deleting these points from the discretized grid. The resulting problem is

$$\begin{aligned} &\min \ t \\ &\text{s.t. } |B(u) - B_d(u)| \leq t, \quad \forall u \in U, \end{aligned} \tag{5.29}$$

where the problem variables are the complex array weights w_l, $l = 1, 2, \ldots N$, and where $U = \{\|u\| \leq 1\} \setminus \{\text{transition regions}\}$. We use the technique of section 5.4 to make the complex magnitude constraint expressible as a set of linear inequalities, thus leading to a third dimension ($\theta \in [0, 2\pi]$) for our semi-infinite parameter. We discretize θ space using 50 points. An alternative approach would be to try to use the complex magnitude design method of section 5.3, but this would require an extension of the formulation and spectral factorization techniques used there, and does not allow for control of the phase of B. Alternatively, we could just treat it

[13] For directions-of-arrival, u with $B(u) = 1$, the array signal-to-noise ratio improvement versus a single element, assuming spatial white noise, is given by $G = \|w\|_2^{-2} := \left(\sum_{i=0}^{N-1} |w_i|^2\right)^{-1}$ [Tre02]. This is an important point, because a good fit to the target beam pattern does not mean that the array will have good noise performance. Therefore, it can be useful to add a term to the objective or a constraint to keep $\|w\|_2$ relatively small. With linear constraints only, we can bound the ∞-norm or 1-norm of w.

[14] As in the previous two problems, we estimated the Lipschitz constants $R_1 \approx 50$ and $R_2 \approx 200$ for this problem using some rough approximations and numerical tests. Since the interval length is 2 for each dimension, and using $\epsilon = 0.05$ we have, using the formulas (5.6) and (5.8), $L_1 \approx 1000$ and $L_2 \approx 65$.

as a semi-infinite quadratic or second-order-cone constraint. The latter approach has been shown to be effective in [WBV98, LB97], but in order to formulate the problem as a linear program, we use the method of section 5.4.

We end up with an LP of size 99×272250. The results of the optimization are shown in Figure 5.11 and numerical performance is discussed in section 5.6 and summarized in Table 5.4.

5.6 Numerical performance

The numerical results of our (regularized) algorithm rMPC* on the problems described in this chapter are summarized in Table 5.4. We have run each problem both using a regularized version of rMPC* (with $\bar{\delta} = 10^{-8}$) with the entire constraint set (i.e., without constraint-reduction) and a significantly reduced set of constraints. Specifically, for each problem, in reduced mode, we kept only $M = \max\{15m, \lfloor 0.1n/m \rfloor\}$ most-active and uniformly gridded constraints at each iteration, as well as the local minimizers of the "slack function". This type of rule was described and justified in Chapter 4.

We also took this opportunity to make a comparison with the revised primal simplex algorithm (RPS) with partial pricing (RPS-PP), which was discussed in section 1.3.1. This is essentially the approach recommended for minimax filter design in [BP95] and [NZ99]. The RPS algorithm has the advantage that the only $O(n)$ operations in each iteration are "pricing" the non-basis columns, i.e., computing $c - A^{\mathrm{T}}y$ to decide which column to bring into the basis (the pivoting strategy). By partial pricing, this operation can often be reduced to no more than $O(m)$, at least for the early iterations. In our implementation we simply selected $3m$ columns at random to price, and added (randomly) $3m$ more if none had negative cost. This seemed to work quite well, much better than two alternative pricing/pivoting rules

(performance not shown in the table) that are popular in simplex implementations [BT97]: 1) use the first negative costed column when checking columns in order, or 2) compute the cost of all columns and select the one with minimal cost. For these problems, the simplex iterations can be made very fast, since the work is only $O(m^3)$ and m is small;[15] the downside of the simplex approach is that it takes a large number of iterations to solve the problems, many more than the interior-point methods. This is clearly visible in the results presented in Table 5.4.

The time and iteration entries in the table include the total time and iteration counts from the phase-1 and phase-2 problems. A phase-1 problem is always needed for the RPS algorithm to find an initial basic feasible solution, and MPC/rMPC* required use of the phase-1 problem (2.17) to find an initial dual strictly feasible point for the Type II linear phase filter design problem and the phase noise synthesis filter problem (for the other problems $y^0 = 0$ was strictly feasible for appropriate t^0). The columns `Mmax` and `Mavg` are the maximum and average size of the working constraint set (or number of columns of A used) in each iteration. For RPS-PP, `Mmax` and `Mavg` always equal m, and for the unreduced MPC, both always equal n. On all problems rMPC* is significantly faster than the unreduced MPC method. Consistent with common wisdom, the interior-point algorithms require far fewer iterations than the RPS method. RPS-PP slightly outperforms rMPC* on the first and third problems, but fails on the phase noise filter problem (the phase-1 procedure fails), and is considerably slower than rMPC* on the antenna array problem.

Finally, we would like to note another important advantage our our approach: *it generates dual feasible iterates with monotonically improving objective.* If our algorithm prematurely terminates, for example, due to numerical problems, or impatience, the final iterate may very well provide a good solution to the problem of interest—the discretized dual SILP. In contrast, since the RPS works on the primal problem, a feasible solution for the dual SILP cannot easily be generated until opti-

prob	alg	status	time	iter	max $\|Q_k\|$	mean $\|Q_k\|$
Linear phase FIR	rps-pp	succ	2.61	629	65	65.0
	mpc	succ	19.91	25	39372	39372.0
	rmpc	succ	5.48	40	1985	1947.3
Phase Noise Filter	rps-pp	fail	Inf	Inf	252	252.0
	mpc	succ	696.47	33	163840	163840.0
	rmpc	succ	112.57	63	7907	7772.4
Linear Predictor	rps-pp	succ	10.14	2672	26	26.0
	mpc	succ	35.72	31	105050	105050.0
	rmpc	succ	12.50	49	1963	1704.5
Antenna Array	rps-pp	succ	130.93	8042	99	99.0
	mpc	succ	299.61	32	272250	272250.0
	rmpc	succ	42.68	35	9769	8598.3

Table 5.4: Numerical performance on the applications problems. Here we compare the performance of the reduced and unreduced MPC algorithms, along with the RPS-PP algorithm. The time and iteration entries on the table include the total time and iteration counts from the phase-1 and phase-2 problems. A phase-1 problem is always needed for the RPS algorithm to find an initial basic feasible solution, and MPC/rMPC* required use of the phase-1 problem (2.17) to find an initial dual strictly feasible point for the Type II linear phase filter design problem and the phase noise synthesis filter problem. On all problems, rMPC* is significantly faster than the unreduced MPC method. Consistent with common wisdom, the interior-point algorithms require far fewer iterations than the RPS method. RPS-PP outperforms rMPC* on the first and third problems, but fails on the phase noise filter problem (the phase-1 procedure fails), and is slower than rMPC* on the antenna array problem.

mality is achieved. Thus, if numerical problems cause the RPS iteration to fail, then all may be lost (at least if feasibility is critical). This is also true for the traditional Parks-McClellan/Remez Exchange methods which are closely related to the RPS method. For further discussion of the benefit of feasible, monotonic algorithms in engineering design problems, see, e.g., [PT93].

[15]Our implementation of RPS is a basic one coded in Matlab with the same level-of-effort and care given to our coding of the interior-point algorithms. For simplicity, we did not use updating procedures discussed in section 1.3.1 to improve the per-iteration work to $O(m^2)$, but we also did not use updating schemes in our implementation of rMPC*. It would be worthwhile to try to make a comparison against a professional simplex software package, but we would want to be able to use different partial pricing rules which may require access to the source. We did try to get iteration counts using Matlab's `linprog` LP solver configured to use the simplex method without partial pricing, but we were unable to solve any of our four test problems using it. While such an effort is still worthwhile, we leave it to future work.

5.7 Discussion

In this chapter we examined four real world filter design problems that were naturally formulated as SILPs. A direct and general approach to solving such problems is to discretize the constraint set and solve the resulting finite LP with a linear programming algorithm. Since the discretized problem is typically very unbalanced, i.e., $n \gg m$, it is important to use an LP algorithm that can take advantage of this structure. The standard approach for solving such LPs is to work on the primal SILP and use an algorithm related to the revised primal simplex (RPS) method with partial pricing/column generation. We demonstrated in this chapter (and in this dissertation) that interior-point algorithms using constraint-reduction methods can also be a viable option for such problems, and that they may offer benefits over RPS-like methods. One benefit is that the constraint-reduced IPMs may actually be more efficient on some problem classes. For our algorithm rMPC* at least, another benefit over the RPS-type methods is that rMPC* is a feasible ascent algorithm for the problem-of-interest (the discretized dual SILP), which means that the iteration can be stopped as soon as the objective is "good-enough", or if numerical difficulties cause the iteration to stop short of optimality, a feasible solution that may be good enough is still generated.

More traditional approaches to solving such design problems may offer improvements in speed over the LP approach, but this usually comes at the cost of reduced flexibility and generality. There is a general trend in engineering to get away from specialized algorithms, and rely more and more on methods of (especially convex) mathematical programming to model and solve design problems [BV04, Nes03]. The author believes this is a good direction to move in, and hopes the work of this dissertation can help in a small way to encourage the trend.

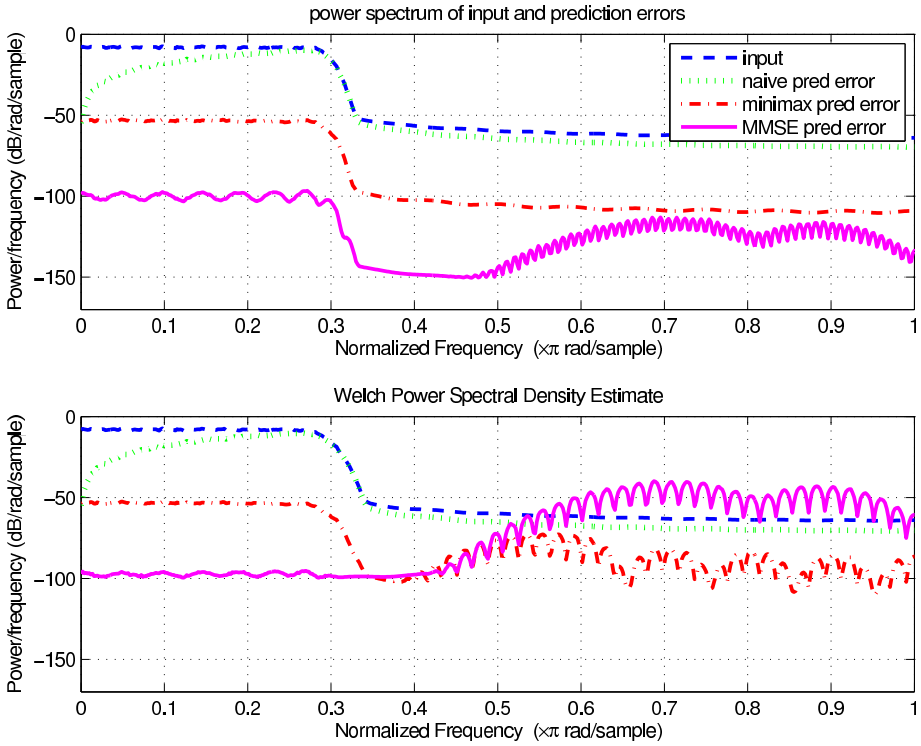

Figure 5.7: Linear prediction filter in action. Both plots show the power spectrum of an input noise signal that has been bandlimited to $[0, 0.35\pi]$, as well as the spectrum of the corresponding prediction error signals for the naive predictor that sets $\hat{x}_{k+1} = x_k$, the minimax predictor, and the minimum mean squared error (MMSE) predictor for ideally bandlimited white noise in $[0, 0.35\pi]$ (designed by solving the Yule-Walker equations [PP02]). The top plot shows the result when the bandlimited input is generated using a 200 tap FIR lowpass filter with passband $[0, 0.25\pi]$, and stopband $[0.35\pi, \pi]$, and the bottom uses a 100 tap FIR with the same passband/stopband. While the difference in the input signal spectrum in each case is hardly noticeable, the performance of the predictors is very different. The power ratio of the input to prediction error signals goes from 45.48dB to 45.46dB using the minimax predictor, while for the MMSE predictor it goes from 91dB to 37dB! The reason for this disparity in robustness can be attributed to the bound constraints that we have put on the filter coefficients for the minimax predictor, which keeps them at a much smaller level than that of the MMSE filter (see Figure 5.8). This controls the behavior of the filter outside of the $[0, 0.35\pi]$ band of interest. (It should be noted that a similar regularization can be added to the MMSE filter by, for example, adding a white noise component to the ideal lowpass correlation sequence, i.e., increasing r_0 just a little.)

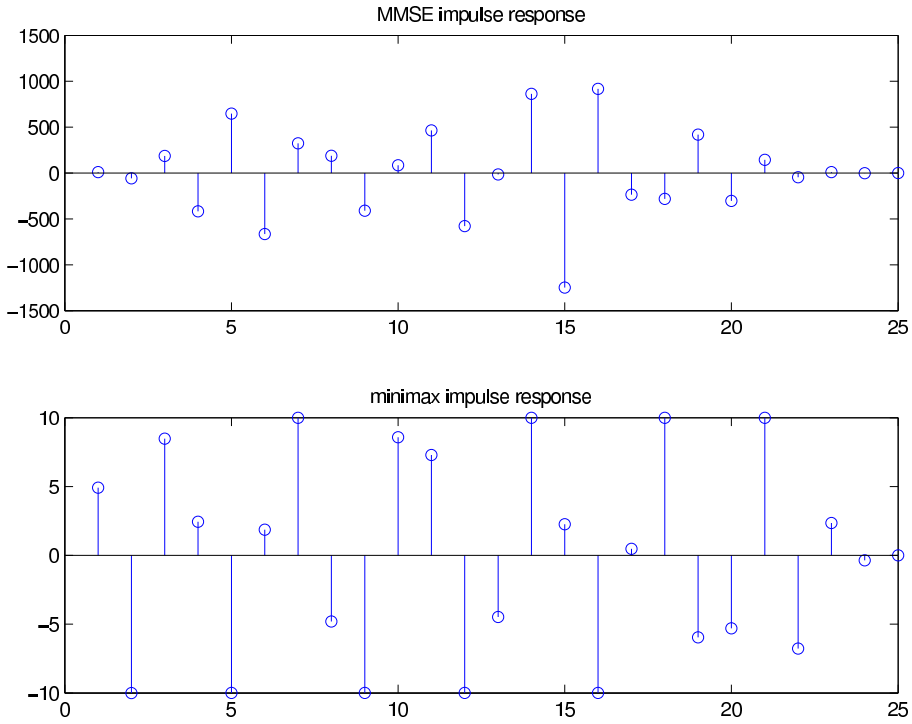

Figure 5.8: Impulse response for the minimax (top) and MMSE (bottom) linear prediction filters. Note the difference in scale. The bound constraints control the minimax filter coefficient magnitudes and have a "regularizing" effect on its performance.

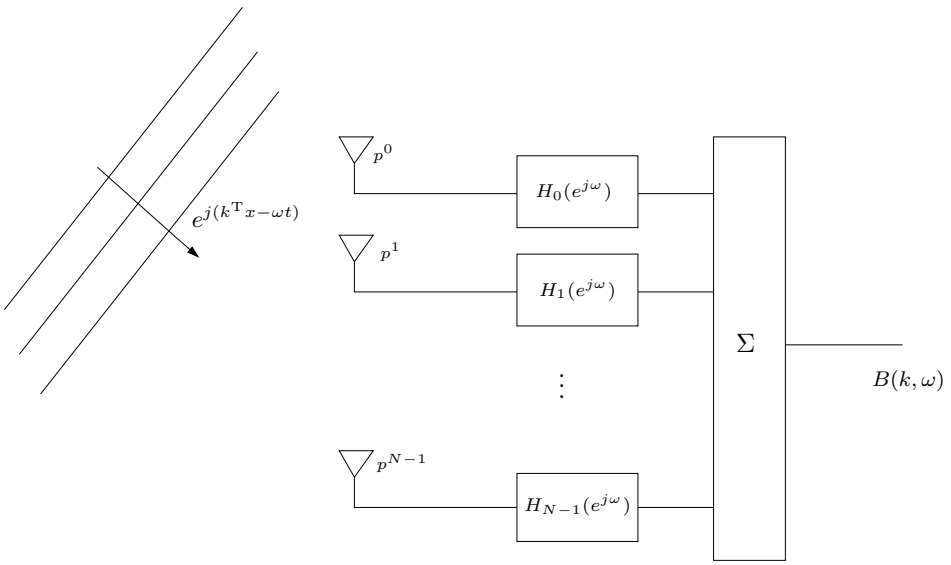

Figure 5.9: A general array signal processor, with array elements at locations p^l, $l = 0, 1, \ldots N-1$. The problem is to design the filters $H_l(e^{j\omega})$, for $l = 0, 1, \ldots N-1$, so as to achieve desirable properties in the beam pattern $B(k, \omega)$, the response of the processor to a plane wave $e^{j(k^{\mathrm{T}}x-\omega t)}$ with frequency ω and wavenumber vector k. When the signals of interest are narrowband so the relative time delays in the time of arrival of the signals to the various elements can be approximated by a phase shift, often the filter is taken to be $H_l(e^{j\omega}) = w_l \in \mathbb{C}$. When the array processor takes this form it may be called a "phased-array" processor.

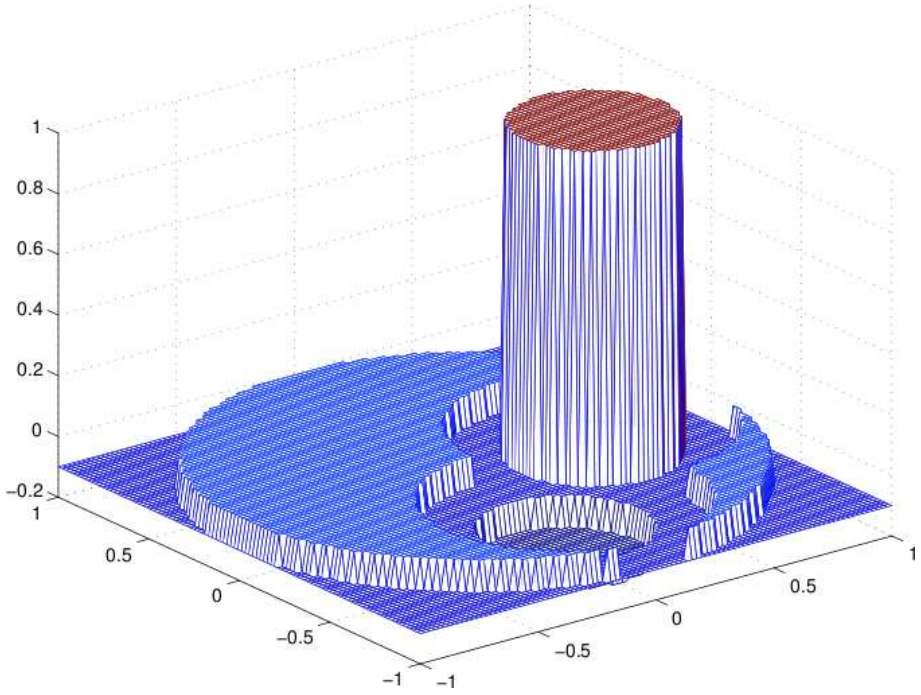

Figure 5.10: Target beam pattern for the antenna array problem. We are asking for 0dB gain, shown at level 1.0, at $(u_x, u_y) = (-0.5, 0)$ and a wide null of the same shape at $(u_x, u_y) = (0, -0.5)$, shown at level -0.2. The "don't care" transition regions are shown at level -0.1 and include the "invisible region", $u_x^2 + u_y^2 > 1$.

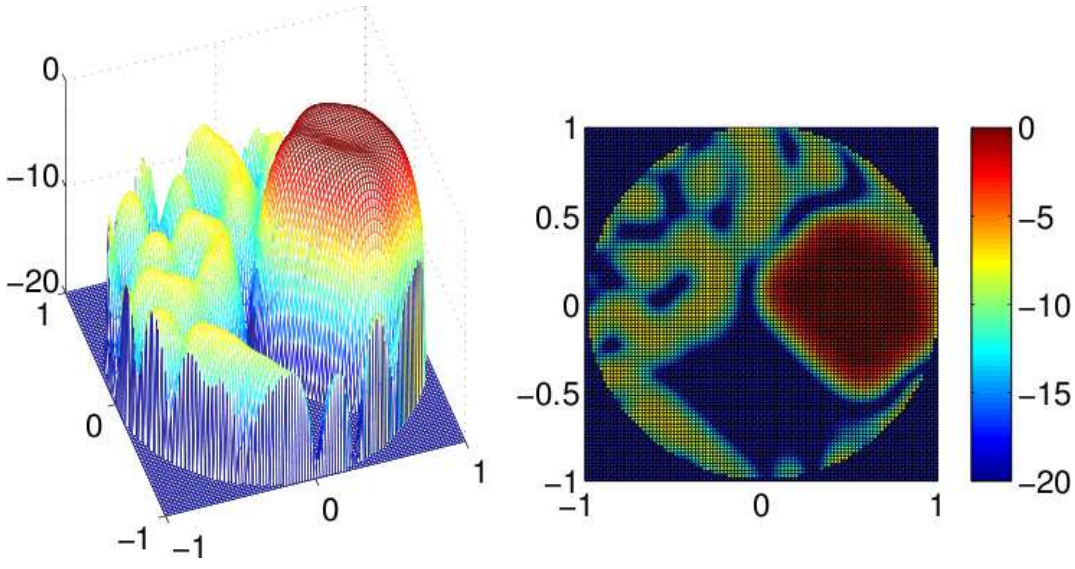

Figure 5.11: Optimized beam pattern for the antenna array problem.

Chapter 6

Summary

In this chapter, we review the main content and contributions of this dissertation.

In Chapter 1 we defined our problem class of interest, the "unbalanced" linear program with many more inequality constraints than variables. For problems in dual standard form, these are LPs with $n \gg m$. Next, we reviewed traditional approaches for solving LPs with many inequalities (and, more generally, convex optimization problems with many inequalities), including a detailed discussion of the revised primal simplex method with partial pricing. After this, we provided a comprehensive review of prior "constraint-reduced" interior-point algorithms, noting that for each of the major classes of interior-point algorithm, namely the affine-scaling, potential-reduction, and path-following methods, a constraint-reduced variant had been developed. In most past work, the analysis of the constraint-reduced variants followed closely that of the parent algorithm through the use of a "minor-cycle" that, at each iteration, built up the working constraint set until an adequate step could be generated. Prior to [TAW06], research had focused on dual algorithms, whereas in [TAW06], the authors considered a constraint-reduced *primal-dual* interior-point (PDIP) algorithm rPDAS, which furthermore, did away with the minor-cycle. In [TAW06], the authors also proposed a simple constraint-reduced version of Mehro-

tra's predictor-corrector (MPC), the PDIP algorithm implemented in virtually all interior-point software for LP and more generally convex-conic programming, as a straightforward extension of their rPDAS algorithm. Numerical results were strong for this algorithm, called rMPC, but no analysis was attempted. Further numerical study of this type of algorithm with some ideas developed toward a convergence analysis was conducted in [Nic09].

In Chapter 2, we proposed and analyzed a convergent constraint-reduced variant of Mehrotra's predictor-corrector algorithm, which we called rMPC* to distinguish it from Algorithm rMPC proposed in [TAW06]. Specifically, rMPC* uses MPC-like search directions computed for "constraint-reduced" versions of the problem; see (2.16). As for other constraint-reduced IPMs, the cost of an iteration of rMPC* can be much less than that of an iteration of MPC; specifically, the high order work, when solving the normal equations by direct methods with dense A, is reduced from $\mathcal{O}(nm^2)$ to $\mathcal{O}(|Q|m^2)$, where the theory allows $|Q|$ to be $\mathcal{O}(m)$ in nondegenerate cases. The primary contribution of this chapter is the global and local quadratic convergence analysis of the algorithm under a very general class of constraint selection rules, and minimal assumptions. The analysis has similarities to that in [TAW06] for the rPDAS algorithm, but the constraint selection rule used in our analysis is more general, the nondegeneracy assumptions are considerably less restrictive, and we employ a somewhat different notion of constraint-reduction that we feel is more natural. The analysis of rMPC* extends to constrained-reduced, as well as unreduced, primal-dual affine scaling as a limit case, thus improving on the results of [TAW06].

In Chapter 3 we discussed the artificial rank-degeneracy that constraint-reduced algorithms are subject to. Prior constraint-reduced IPMs required that the working set of constraints satisfy $\text{rank}(A_Q) = m$. In practice, this condition can be difficult to achieve with simple constraint selection rules, and schemes for enforcing it can

be computationally intensive, or can end up adding back much, or most, of the original constraint set. We briefly discussed some simple methods that essentially sidestep the issue, and then proposed and analyzed two algorithms, the regularized and kernel step rPDAS, that dealt with it directly. The latter algorithm is based on the algorithm proposed and analyzed in [TAO06], but we provide an analysis under somewhat relaxed assumptions. The analysis of each of these algorithms led to similar conclusions as the Chapter 2 analysis of rMPC*, namely global and local quadratic convergence to the optimal set under certain assumptions. Notably, we also uncovered a connection between these two algorithms, showing that the kernel step rPDAS algorithm, which was actually proposed first in [TAO06], could be motivated as a limiting case of the regularized rPDAS algorithm using a very small regularization parameter. Furthermore, since computing kernel steps is expensive (compared to solving normal equations), the kernel step search directions might be approximated as the regularized rPDAS direction with very small parameter.

In Chapter 4 we investigated the numerical performance of the three algorithms discussed in Chapters 2 and 3. First we focused on rMPC*. We developed several constraint selection heuristics and demonstrated the effectiveness of rMPC* on a class of random problems where its performance was remarkably good. On these problems it appeared that we could use constraint reduction to great advantage: the iteration counts were the same whether we used the entire constraint set or only the 1% most nearly active constraints, while computation times were dramatically reduced. We also observed remarkable numerical behavior of rMPC* on a class of discrete Chebyshev approximation problems after the development of a rule for constraint selection tailored to this class of problems. We conducted a limited numerical comparison of rMPC* against the other constraint-reduced IPMs described in Chapter 1, and rMPC* performed favorably. We then investigated the performance of the regularized algorithms on two classes of problems that have high degree of degen-

eracy (in the sense of Chapter 3), the tube-in-cube problem of [TAO06] and a class of sparse random problems. We introduced a (random) measure of the degree of rank-degeneracy based on computing the average rank of randomly sampled $m \times m$ sub-matrices of A. This measure is consistent on the tube-in-cube problem, evaluating to the dimension of the kernel of A^{T} (before adding the bound constraints), while it also verifies the expected correlation between sparsity and degeneracy. We predicted and observed an interesting qualitative behavior of the kernel step rPDAS and regularized rPDAS (with small parameter) on this problem, namely that they took about k kernel-like steps, where k is the degree of rank-degeneracy of the problem, and then switched to regular steps until termination.

In Chapter 5 we investigated some real-world applications from the area of digital filter design. We presented a variety of approaches to filter design problems, each of which results in a semi-infinite linear program that can be discretized, leading to a finite but unbalanced linear program, and then efficiently solved using algorithms designed for unbalanced LPs. The approach proposed in much of the literature on SILP design of filters is to use the revised primal simplex algorithm (RPS). We presented a comparison of the numerical performance of a regularized variant of rMPC* (an obvious extension of our regularized rPDAS) versus the unreduced MPC algorithm, and versus a basic implementation of the traditional RPS method. The constraint-reduced rMPC* greatly outperformed the unreduced MPC, as expected, but overall also outperformed the RPS algorithm, whose performance was uneven on the four filter design problems.

The techniques used in the analysis of rMPC* and the regularized and kernel step rPDAS algorithms (like that of rPDAS in [TAW06]) allow for the elimination of the minor-cycle that had been present in most previous constraint-reduced interior-point algorithms. While this has the benefit of guaranteeing a reduced work per-iteration, it may make it very difficult to attain (polynomial) complexity results,

i.e., global rates of convergence, whereas such results are one of the major strengths of interior-point algorithms. We believe it may require a different approach to the analysis to correct the situation. On the other hand, we believe that our line of analysis should be extensible to general nonlinear programming problems, where complexity results are elusive. These ideas suggest several lines of possible future research. We elaborate on these ideas, and propose some more possible future lines of research in the next and final chapter.

Chapter 7

Future lines of research

7.1 Complexity results

The major benefit of extending constraint-reduction to primal-dual interior-point methods (PDIPMs) is that they generally have better practical performance than primal-only or dual-only IPMs. This is particularly true of MPC, which is the current champion of the IPMs for LP. However, one shortcoming of our work is that we have no complexity results, whereas LP is a polynomial-time class of problems. Even in the context of constraint-reduction, such complexity results have been established by Tone, Kaliski and Ye and den Hertog, et al. It would certainly be nice to develop complexity results for a variant of rMPC* or other constraint-reduced PDIPM.

One source of difficulty in pursuing this goal is that, although rPDAS (of [TAW06]) and rMPC* are primal-dual algorithms, they do not treat the primal and dual variables symmetrically. They are motivated by the geometry of the dual problem, enforcing dual feasibility, while updating the primal variables using a specialized rule (2.14) that does not preserve primal feasibility. Primal feasibility tends to come only near the solution. Standard lines of analysis for PDIPMs that lead to

polynomial complexity results generally require that either the iterates are primal and dual feasible, or become so at a controlled rate. In particular, the "infeasible variants" of path-following methods usually enforce a condition like

$$\|b - Ax^k\| \leq \beta\mu^k,$$

where $\mu^k = (x^k)^{\mathrm{T}} s^k/n$ is the current duality measure [Wri97]. In practice, with such algorithms, the infeasibilities typically converge to zero much faster than the sequence of duality measures. We could attempt to enforce a similar condition by introducing a minor-cycle, as in the work of [DY91, Ton93, HRT94], that adds constraints until the desired condition is satisfied. The infeasible primal-dual potential-reduction algorithm of Mizuno, Kojima and Todd [MKT95], may offer an good parent algorithm on which to base the constraint-reduction.

An alternative approach is to take the "inexact" point-of-view. We view the constraint-reduced step merely as an approximation to the unreduced step. For example, in the work of Tits et al. [TAW06], although motivated by the idea that most constraints could be ignored, the search direction is defined by simply replacing the normal matrix by the reduced normal matrix. We could think of this as an approximation to the full normal matrix, and thus the resulting step as an approximation to the full step.[1] From this point-of-view, it may make sense to try to bound the error in the reduced step, and use or develop algorithms that allow some error or "inexactness" in the step.

Some prior work along the inexact lines is the work of Schurr [Sch06, SOT09], who developed a short-step path following algorithm for conic optimization that allows inexact evaluation of the barrier function gradient and Hessian, and still achieves the best known complexity (see section 7.4 below). In another paper

[1] If very aggressive constraint reduction is used, i.e., if only a small fraction of the total constraint set is used in forming the reduced normal matrix, then this approximation point of view may not be appropriate, at least in early iterations.

[Kor00], a PDIPM for LP is developed that allows some error in the right-hand side of the step equations and still achieves polynomial complexity. The motivation here was to use an iterative solver and compute inexact search directions by stopping the iteration early, but those ideas may be useful for inexactness arising from ignoring constraints.

7.2 Efficient and robust implementation

Efficient updating schemes

Many of the linear algebra tasks arising in linear programming algorithms benefit from introduction of efficient updating schemes. For example in the simplex method, at each iteration, the basis matrix changes only by removal and insertion of a column. Practical simplex implementations perform rank-two updates to a factorization of the basis which requires only $\mathcal{O}(m^2)$ work rather than the $\mathcal{O}(m^3)$ work needed for a from-scratch re-inversion. Similar ideas apply in interior-point algorithms, where the dominant work per iteration, if we solve the normal equations, involves forming and solving systems of equations of the form

$$ADA^{\mathrm{T}}u = b.$$

Since only the diagonal matrix D changes from iteration to iteration, we expect to be able to reuse some information from past iterations. A well known method for doing this was proposed in Karmarkar's seminal paper [Kar84].

In the context of constraint-reduction, still more possibilities arise since often the normal matrix is built-up from a small initial set of columns of A. Several of the papers discussed in Chapter 1 suggest factorization updating schemes. In particular,

in the minor-cycles of the [DY91, Ton93] algorithms, rank-one updates to the normal matrix factorization are recommended. Updating relevant matrix factorizations is something that should be done in any efficient implementation.

Use of iterative solvers

In our numerical experiments, we have almost exclusively solved the normal equations or augmented (KKT) system by direct methods. However, there may be situations (e.g., when $m \gg 1$) where iterative methods such as preconditioned conjugate gradients (PCG) [Saa03] can do the job more efficiently. When this is the case, we will not get exact solutions to the Newton system (2.18), but rather we will compute an approximation $\tilde{\Delta y}$ to the exact Δy. We can control the accuracy (in terms of the residual at least) of $\tilde{\Delta y}$ by deciding when to terminate the iteration. Alternatively, rather than solving the normal equations we could solve the (reduced) augmented system (2.48); there are some well-known advantages to doing this, namely, both the conditioning and the sparsity pattern of the augmented matrix is generally better than that of the normal matrix. The obvious disadvantage is that it is of size $(n+m) \times (n+m)$ rather than $m \times m$, and if the A matrix is dense, particularly when $n \gg m$, there is a great advantage to the normal equations approach; when the A matrix is sparse, as is often the case in large-scale real world problems, then the advantage is considerably less. There has been quite a large amount of work on using iterative methods to solve linear systems arising in primal-dual interior-point methods; see, for example, [WO00, LMO06, Kor00]. Even in the context of constraint-reduction, there has been some investigation: Kaliski and Ye [KY93] used iterative methods to solve the normal equations, in particular, they reported great success with the PCG algorithm using the minimal spanning tree (MST) preconditioner (see section 1.9.3).

It seems that there are a variety of interesting future research avenues. Ques-

tions that arise include the following:

1. In the context of constraint-reduction, when is using an iterative solver advantageous?
2. Which iterative methods are most effective?
3. Is it preferable to solve the normal equations, or the augmented system?
4. What are good preconditioners to use for each?
5. What kind of gains can be expected?
6. How does one preserve or obtain convergence results in the face of inexactness coming from the inexact solutions to the linear systems?

7.3 Specialization to target applications

It seems that general applicability of constraint-reduction with simple constraint selection rules may be somewhat limited. We saw the initial failure of the simplest rules for choosing the working set on the Chebyshev approximation problems. This could be taken as a counterexample to the proposition that for problems with many constraints, a few simple general constraint reduction rules suffice to achieve significant savings in computation.

On the other hand, the promising result of that study was that after some closer examination of the problem structure we were able to come up with a very effective rule for constraint selection for smooth Chebyshev approximation and the result, in our specific example, was a better than 10-fold speed up over the unreduced case. The efficiency of this rule was further confirmed in the observed numerical performance on the application problems of Chapter 5. This can be viewed as a simple instance of the ubiquitous general principle that exploiting problem structure in optimization is critical for the design of efficient methods. There is no free lunch;

we should not expect the simplest general rules to suffice.

Another example of the success of specialization is the work of Kaliski and Ye (see section 1.9.3) who made some significant practical improvements and a theoretical breakthrough in specializing Tone's method (see section 1.9) for connected network flow problems and the transportation problem in particular. Yet another example is the specialization of the PDAS and rMPC algorithms [TAW06] (see section 1.12) to the training of support vector machines [JOT10, JOT08] (see section 1.14).

The bottom line is that one should attempt to specialize constraint selection rules to specific classes of problems to take advantage of the problem structure in using constraint-reduction. This can possibly be done in the context of the general adaptive rule described in section 4.1.2, which assigns a value to v_i to each constraint and selects them in order of largest value. Allowing this vector of values v to depend on the full history of iterates, in effect building-up a "value-function" for the constraints, may be beneficial.

Some potential problem classes that may be worth studying (further) are the Chebyshev approximation problem, Markov decision processes (MDPs), MDPs with special structure, queuing models, network flow problems, and transportation problems. Next we provide some foundation for application to MDPs.

Markov decision processes

Another class of problems that can be formulated as inequality constrained LPs in dual standard form, with many more constraints than variables, is the discrete time finite state (DTFS) Markovian decision process. MDPs form a powerful modeling paradigm for framing stochastic control problems. We use [BT05] as our main reference for this section.

The basic setup goes as follows. Let $\mathcal{X} = \{1, 2, \ldots n\}$ be the state space of a

DTFS controlled Markov chain. At each state x there is a set $\mathcal{U}(x)$ of admissible controls. Under control $u \in \mathcal{U}(x)$, the Markov chain transitions to the next state, with the probability of moving to state $y \in \mathcal{X}$ given by $p(x, y, u)$. A stationary (control) policy μ is an element of the Cartesian product $\Pi_{x\in\mathcal{X}} U(x)$, i.e., a map $\mu : \mathcal{X} \to \mathcal{U} := \cup_{x\in\mathcal{X}} \mathcal{U}(x)$ with $\mu(x) \in U(x)$ for all $x \in \mathcal{X}$. Under a fixed policy μ, the transition probabilities no longer depend on u and can be arranged into the transition matrix P_μ, defined by $[P_\mu]_{xy} = p(x, y, \mu(x))$. Given a distribution on $\mathcal{X}$ arranged into a vector π, after a transition of the Markov chain, the new distribution is $P_\mu^{\mathrm{T}}\pi$.

We are interested in identifying an optimal policy, so we must introduce a cost structure. At each state-control pair (x, u), we assume that a (supposed nonrandom) cost of $g(x, u)$ is incurred. Under each fixed policy this cost can be arranged into a vector $g_\mu \in \mathbb{R}^n$ defined by $[g_\mu]_x = g(x, \mu(x))$.

The expected "discounted" cost J_μ incurred using policy μ starting from state x satisfies the linear equation (linear in $J_\mu \in \mathbb{R}^n$)

$$J_\mu(x) = g(x, \mu(x)) + \alpha \sum_{y\in\mathcal{X}} p(x, y, \mu(x)) J_\mu(y),$$

where $g(x, \mu(x))$ is the cost of the current state under policy μ and

$$\sum_{y\in\mathcal{X}} p(x, y, \mu(x)) J_\mu(y)$$

is the expected future cost; $\alpha \in [0, 1)$ is a discount factor that says we are somewhat less concerned about future costs (and makes the theory work out nicely). Combining these equations for all $x \in \mathcal{X}$ gives the linear system

$$J_\mu = g_\mu + \alpha P_\mu J_\mu, \tag{7.1}$$

(where we make use of, hopefully clear, vector notation) which has solution

$$J_\mu = (I - \alpha P_\mu)^{-1} g_\mu, \tag{7.2}$$

since $\alpha \in [0, 1)$ makes $(I - \alpha P_\mu)$ invertible. Define the operator T_μ by $T_\mu J := g_\mu + \alpha P_\mu J$, so we can write (7.1) as $J_\mu = T_\mu J_\mu$.

We are interested in identifying a policy that starting from state x achieves the minimal cost, among all possible policies. It turns out that under appropriate conditions, there exists a (not necessarily unique) policy that is simultaneously optimal for all initial states. Given an optimal policy the optimal cost is computed by (7.2). On the other hand, invoking Bellman's principle of optimality (see e.g., [BT05]), we have that given the optimal cost vector J^*, an optimal policy is given as

$$\mu^*(x) = \arg \min_{u \in \mathcal{U}(x)} \{g(x, u) + \alpha \sum_{y \in \mathcal{X}} p(x, y, u) J^*(y)\}.$$

Then the optimal cost must satisfy Bellman's equation

$$J^*(x) = \min_{u \in \mathcal{U}(x)} \{g(x, u) + \alpha \sum_{y \in \mathcal{X}} p(x, y, u) J^*(y)\}.$$

The Bellman operator T is defined by

$$(TJ)(x) := \min_{u \in \mathcal{U}(x)} \{g(x, u) + \alpha \sum_{y \in \mathcal{X}} p(x, y, u) J(y)\},$$

so that we can write Bellman's equation succinctly as

$$J^* = TJ^*.$$

Two remarkable properties of T are, first, that it is monotonic, i.e., if $J \leq J'$ (in the componentwise sense), then $TJ \leq TJ'$, and second, that it is a contraction

in the infinity norm, i.e.,

$$\|TJ - TJ'\|_\infty \leq \beta \|J - J'\|_\infty,$$

with $\beta \in [0, 1)$ [BT05].

There are at least three standard approaches for solving MDPs. *Value iteration* is based on the fixed point iteration

$$J_{k+1} = TJ_k,$$

which converges to J^* since T is a contraction. *Policy iteration* generates a sequence of improving policies $\{\mu_k\}$ by alternating policy evaluation, which consists of solving the linear equation

$$J_k = T_{\mu_k} J_k,$$

and policy improvement, which consists of finding μ_{k+1} from the equation

$$T_{\mu_{k+1}} J_k = TJ_k,$$

that is

$$\mu_{k+1}(x) = \arg \min_{u \in \mathcal{U}(x)} \{g(x, u) + \alpha \sum_{y \in \mathcal{X}} p(x, y, u) J_k(y)\}.$$

It can be shown that $J_{k+1} \leq J_k$ with the inequality strict in at least one component. Hence, since there are only finitely many policies (and no policy can repeat since the cost is strictly monotonic), this process converges finitely.

The approach we are interested in, however, is the *linear programming* approach, which is based on the following observation. For any vector $J \in \mathbb{R}^n$ that satisfies $J \leq TJ$ we have, using the monotonicity property of the Bellman operator,

that

$$J \leq TJ \leq T^2 J \leq T^k J,$$

for $k \geq 2$, and letting $k \to \infty$, we get

$$J \leq J^*.$$

Now, since $J^* = TJ^*$, we have that J^* solves the following mathematical program:

$$\begin{aligned} &\max \ c^{\mathrm{T}} J \\ &\text{s.t. } J \leq TJ \end{aligned} \tag{7.3}$$

for any $c \in \mathbb{R}^n$, $c > 0$. This is not an LP, but is equivalent to the following LP in dual standard form with n dual variables and nk constraints, where we assume $|U(x)| = k$ for all x (otherwise there are $n \sum_{x \in \mathcal{X}} |\mathcal{U}(x)|$ constraints).

$$\begin{aligned} &\max \ c^{\mathrm{T}} J \\ &\text{s.t. } J(x) \leq g(x,u) + \alpha \sum_{y \in \mathcal{X}} p(x,y,u) J(y) \\ &\qquad \text{for all } x \in \mathcal{X}, \ u \in \mathcal{U}(x) \end{aligned} \tag{7.4}$$

This LP is unbalanced, and thus a potential candidate for a constraint-reduced IPM, if $k \gg 1$; this can happen, for example, if the control is a finely discretized version of a continuous control input. A perhaps more common situation is that n is a huge number (by Bellman's curse of dimensionality [BT05]) and k is rather modest in comparison. In this case, practitioners often give up on finding J^* and the corresponding optimal policy, but rather settle for an approximate solution derived from an approximation to the cost vector. Often the approximation takes the form of a linear model $J = Hr$, where $H \in \mathbb{R}^{n \times m}$ is fixed, and $r \in \mathbb{R}^m$ with $m \ll n$.

Plugging this model for J in (7.3) leads to

$$\begin{aligned} &\max \ (H^{\mathrm{T}}c)^{\mathrm{T}}r \\ &\text{s.t. } (Hr)(x) \leq g(x,u) + \alpha \sum_{y \in \mathcal{X}} p(x,y,u)(Hr)(y), \\ &\text{for all } x \in \mathcal{X}, \ u \in \mathcal{U}(x), \end{aligned} \tag{7.5}$$

which is an LP with only m variables but still nk constraints. This is sometimes called the *approximate linear programming* (ALP) (which I think is an abbreviation for something like *approximate dynamic programming via linear programming*) approach to solving MDPs [TS93, dFR04, dFR03]. Such problems may be good candidates for applying constraint-reduction techniques. Unfortunately, many times, in real applications, n is an astronomical number. Since the constraint-reduced interior-point algorithms we have discussed so far, including rMPC*, still do at least $\mathcal{O}(mn)$ work at each iteration, these methods still may not be viable. Possible ways forward in this case are by using simplex methods with column generation, or cutting-plane methods if efficient column/constraint generators are available, or possibly constraint sampling methods [dFR04, CC06, CC05, CG08] which sample a subset of the constraints and ignore the rest completely. The reduced ALP (RALP) is then solved for r^*, and the hope is that not too many of the ignored constraints will be violated at the resulting solution, and even if some are, the policy that results using this $J = Hr^*$ is still useful.

Initial tests with rMPC* on an ALP formulation of a simple MDP, have been promising; we believe further investigation could definitely be worthwhile.

7.4 Extension to convex-conic form problems

Any convex optimization problem can be expressed in "conic standard form"

$$\begin{array}{ccc} \min \langle c, x \rangle & & \max \langle b, y \rangle \\ \text{s.t. } Ax = b, & \text{and} & \text{s.t. } A^* y + s = c, \\ x \in K, & & s \in K^*, \end{array} \tag{7.6}$$

where K is a proper cone, K^* is its dual (i.e., $K^* = \{y \mid \langle x, y \rangle \geq 0\}$), and A^* is the adjoint of linear map A, see e.g., [Ren01]. This is a formal generalization of LP, which fits the above form with K the nonnegative orthant. There is a very nice duality theory for conic programming (CP), which parallels that of LP (with a few complications—in particular, strong duality need not hold even when both primal and dual problems are feasible). One of the most remarkable advancements in optimization theory in recent history was the extension of interior-point-methods developed for LP to general convex problems. In a series of research papers culminating in the monograph [NN93], Nesterov and Nemirovskii showed that whenever an (efficiently computable) *self-concordant*[2] barrier for the convex feasible region D is available, then the general convex optimization problem

$$\begin{array}{c} \min \langle c, x \rangle \\ \text{s.t. } x \in D \end{array} \tag{7.7}$$

(any convex optimization problem can be posed in this form as well) can be solved efficiently (in polynomial-time) by interior-point-methods. The basic algorithm, the

[2]These are convex functions that have Lipshitz continuous Hessians in the (affine-invariant) norm induced by their own Hessians. These are functions that are in some sense ideally matched to Newton's method. See [Nes03, Ren01, NN93].

path-following method, approximately solves a sequence of barrier subproblems

$$\min \langle c, x \rangle + \mu F(x). \tag{7.8}$$

The solutions to these problems for $\mu \in (0, \infty)$ trace out the "central-path" which converges to the solution to (7.7) as $\mu \to 0$.

The interior-point theory combines nicely with the duality theory for CP. Given a self-concordant barrier F for the cone K, it can be shown that its conjugate function F_* (see e.g., [Nes03] or [Ren01]) is a self-concordant barrier for the dual cone K^*. Considering the associated barrier problems for the primal and dual instances

$$\begin{array}{cc} \min \langle c, x \rangle + \mu F(x) & \max \langle b, y \rangle - \mu F_*(x) \\ \text{s.t. } Ax = b, & \text{s.t. } A^*y + s = c, \end{array} \tag{7.9}$$

and, using the fact that $-F'(x) \in K^*$ for $x \in K$ and $-F'_*(s) \in K$ for $s \in K^*$, their associated optimality conditions can be written in a nearly symmetric fashion

$$\begin{array}{cc} Ax = b, & Ax = b, \\ A^*y + s = c, & A^*y + s = c, \\ s + \mu F'(x) = 0, & x + \mu F'_*(s) = 0, \\ x \in \text{int}K,\ s \in \text{int}K^*, & x \in \text{int}K,\ s \in \text{int}K^*. \end{array} \tag{7.10}$$

Given a sequence of solutions $\{(x^k, y^k, s^k)\}$ to the primal optimality systems for decreasing μ^k (it is known that the x component of such a sequence converges to the solution of the primal CP instance), it can be shown that the associated dual-feasible sequence $\{(y^k, s^k)\}$ tends to optimality for the dual. In the special situation that the barrier F is *logarithmically homogeneous*[3] which implies, among many other interesting things, that $F'(\mu x) = \frac{1}{\mu} F'(x)$, then the equations $s + \mu F'(x) = 0$ and

[3]Function F is called *logarithmically homogeneous* when it satisfies $F(tu) = F(u) - \nu \log t$, for some $\nu > 0$, e.g., (in the scalar case) $F(u) = -\log u$.

$x + \mu F'_*(s) = 0$ are equivalent, so that the primal and dual central paths coincide. To see this, we make use of the remarkable fact that, in general, $-F'$ and $-F'_*$ are inverses of each other, so that, if $s = -\mu F'(x)$, then

$$-F_*(s) = -F_*(-\mu F'(x)) = -F_*\left(-F'\left(\frac{x}{\mu}\right)\right) = \frac{x}{\mu}, \tag{7.11}$$

and the reverse implication can be shown by a similar argument. Primal-dual interior-point methods for CP use Newton or related directions for one of the above systems, or for equivalent reformulations of one of them. In the special case that K is a *symmetric* (equivalently *self-scaled*) cone,[4] in particular, if K is the non-negative orthant, the second-order cone, or the positive semidefinite cone, then the class of primal-dual interior-point-methods for LP have very nice generalizations to CP; most theoretical results and algorithms for LP have very satisfactory analogs for these classes of problems, see e.g., [NN93, Ren01, Nes03].

Given the prior work on constraint-reduction for LP, it would be natural to consider constraint-reduction in the context of CP, (with K symmetric or not). In the context of LP, we visualize the constraint-reduction procedure as relaxing the original problem and attempting to use or modify a step, or search direction, for the relaxed problem for use in the original problem. Thus, a question is how to form the relaxation in this more general context. Suppose the cone K takes the form

$$K = \cap_{i=1}^{p} K_i,$$

and the barrier for the cone takes the form

$$F(x) = \sum_{i=1}^{p} f_i(x),$$

[4]We refer the reader to [Ren01], and its references, for a definition and detailed exposition on the remarkable properties of symmetric cones and their role in interior point methods.

where f_i is a barrier for the cone K_i. Then the gradient and Hessian for the barrier are of the form

$$F'(x) = \sum_{i=1}^{p} f_i'(x),$$

$$F''(x) = \sum_{i=1}^{p} f_i''(x).$$

The barrier subproblem for the primal problem is

$$\min \langle c, x \rangle + \mu F(x), \tag{7.12}$$

and the associated Newton system is

$$\left(\sum_{i=1}^{p} f_i''(x) \right) d(x) = - \left(c + \mu \sum_{i=1}^{p} f_i'(x) \right).$$

For some $Q \subseteq \{1, 2, \ldots p\}$, we can replace this with

$$\left(\sum_{i \in Q} f_i''(x) \right) d(x) = - \left(c + \mu \sum_{i \in Q} f_i'(x) \right),$$

which could be much less expensive to solve if evaluating the barrier Hessian and gradient is expensive. We could try to develop an algorithm similar to that of rPDAS or rMPC* based on this reduction.

Another way to think of such an approach is as using inexact evaluations of the barrier gradient and Hessian. Schurr considered precisely that idea in [Sch06, SOT09]. Specifically, he considered using inexact evaluations of the barrier gradient and Hessian within a primal-dual short-step path-following algorithm for convex conic optimization problems. Schurr's algorithm uses the Newton step for the first

system in (7.10), namely,

$$\begin{pmatrix} 0 & A^* & I \\ A & 0 & 0 \\ \mu F''(x) & 0 & I \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -s - \mu F'(x) \end{pmatrix}, \tag{7.13}$$

except with the gradient $F'(x)$ and Hessian $F''(x)$ replaced by the approximations $F_1(x)$ and $F_2(x)$, arriving at

$$\begin{pmatrix} 0 & A^* & I \\ A & 0 & 0 \\ \mu F_2(x) & 0 & I \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -s - \mu F_1(x) \end{pmatrix}. \tag{7.14}$$

As is usual in IPMs, the "local inner-product" defined by the barrier Hessian is the natural geometry in which to measure things. In the local norm, the gradient is $F''(x)^{-1}F'(x)$, and the Hessian is just the identity. Thus the approximate gradient is $F''(x)^{-1}F_1(x)$ and the approximate Hessian is $F''(x)^{-1}F_2(x)$, so the error $e_1(x)$ in the local gradient, measured in the local norm, is

$$\begin{aligned} e_1(x) &= \|F''(x)^{-1}(F'(x) - F_1(x))\|_x \\ &= \|F''(x)^{1/2}F''(x)^{-1}(F'(x) - F_1(x))\|, \\ &= \|F''(x)^{-1/2}E_1(x)\|, \end{aligned}$$

where $E_1(x) := F'(x) - F_1(x)$. Defining $E_2(x) := F''(x) - F_2(x)$, we find that the error $e_2(x)$ in the local Hessian, measured in the local-norm induced operator-norm,

is

$$
\begin{aligned}
e_2(x) &= \|I - F''(x)^{-1}F_2(x)\|_x = \|F''(x)^{-1}E_2(x)\|_x \\
&= \sup_z \frac{\|F''(x)^{-1}E_2(x)z\|_x}{\|z\|_x} \\
&= \sup_z \frac{\|F''(x)^{1/2}F''(x)^{-1}E_2(x)F''(x)^{-1/2}F''(x)^{1/2}z\|}{\|F''(x)^{1/2}z\|} \\
&= \sup_z \frac{\|F''(x)^{-1/2}E_2(x)F''(x)^{-1/2}F''(x)^{1/2}z\|}{\|F''(x)^{1/2}z\|} \\
&= \sup_u \frac{\|F''(x)^{-1/2}E_2(x)F''(x)^{-1/2}u\|}{\|u\|} \\
&= \|F''(x)^{-1/2}E_2(x)F''(x)^{-1/2}\| = \|F''(x)^{-1}E_2(x)\|. \qquad (7.15)
\end{aligned}
$$

Schurr proposed a primal-dual short-step IPM for CP, assuming the barrier F to be logarithmically homogeneous, but not requiring K to be symmetric. He proved polynomial complexity of his algorithm under the assumption that the error measures $e_1(x)$ and $e_2(x)$ are kept bounded by certain a priori defined constants ε_1 and ε_2.

Let us consider how we might use this in the LP case. Since we are motivated by ignoring constraints in an LP with many inequality constraints, we will reformulate the standard dual LP (without slack variables) into primal conic standard form. We want the barrier to be a sum of terms with each term corresponding to a constraint in the system, and we want the evaluation of this Hessian to constitute the dominant cost of each iteration. Dual standard form with slack variables (which is in conic form) is inadequate for this purpose since the barrier for the cone (the non-negative orthant) is $-\sum_{i=1}^{n} \log s_i$ which has Hessian S^{-2}; this is cheap to evaluate (the expensive part of the iteration comes in forming the Normal matrix) so the inexact method of Schurr will not be very useful along this tack.

It is simple enough to make this better. Consider the dual standard form LP,

where we use nonstandard names for the data, with $G \in \mathbb{R}^{p\times k}$ and $p \gg k$

$$\begin{aligned} &\max f^{\mathrm{T}}u, \\ &\text{s.t. } Gu \leq g. \end{aligned}$$

Introducing a new variable t we have the equivalent formulation

$$\begin{aligned} &\max f^{\mathrm{T}}u \\ &\text{s.t. } Gu \leq gt, \\ &\quad t = 1, \end{aligned}$$

with variable $x = (u^{\mathrm{T}}, t)^{\mathrm{T}}$ in $\mathbb{R}^{p+1}$. Let $m := 1$, $n := p + 1$, $x := (u^{\mathrm{T}}, t)^{\mathrm{T}}$, $A := e_n^{\mathrm{T}}$ is a row vector of ones, $b := 1$, $c := -f$, and

$$H := (G, -g), \tag{7.16}$$

then we have

$$\begin{aligned} &\min c^{\mathrm{T}}x \\ &\text{s.t. } Ax = b, \\ &\quad x \in K, \end{aligned}$$

where the cone $K := \{x \mid Hx \leq 0\}$. A logarithmically homogeneous self-concordant barrier (see e.g., [NN93, Ren01, Nes03]) for K is

$$F(x) = -\sum_{i=1}^{n} \log(-h_i^{\mathrm{T}}x),$$

where h_i is the ith row of H. F has gradient

$$F'(x) = HDe \tag{7.17}$$

with D a diagonal matrix with $D_{ii} = \frac{1}{-h_i^{\mathrm{T}} x}$, and Hessian

$$F''(x) = HD^2H^{\mathrm{T}}. \tag{7.18}$$

Let us focus on the error in the Hessian, as it is in computing the Hessian approximation where the most significant computational savings can be gained. In the context of constraint-reduction, we will approximate the barrier Hessian, which, in this case, has the form HD^2H^{T}, by the partial sum $H_Q D_Q^2 H_Q^{\mathrm{T}}$. We would like to have a way of ensuring that the appropriate norm of the error matrix $H_{\mathbf{n}\setminus Q} D^2_{\mathbf{n}\setminus Q} H^{\mathrm{T}}_{\mathbf{n}\setminus Q}$ stays within the allowable range, i.e., $e_2(x) < \varepsilon_2$.

Unfortunately, to compute the error $e_2(x)$, we need the full Hessian, which we are trying to avoid computing. So we look for a way to efficiently bound the error. Let us attempt to obtain such bounds for the LP case. To simplify notation define

$$\begin{aligned} M &:= F''(x) = HD^2H^{\mathrm{T}}, \\ M_Q &:= F_2(x) = H_Q D_Q^2 H_Q^{\mathrm{T}}, \\ M_{\mathbf{n}\setminus Q} &:= E_2(x) = F''(x) - F_2(x) = H_{\mathbf{n}\setminus Q} D^2_{\mathbf{n}\setminus Q} H^{\mathrm{T}}_{\mathbf{n}\setminus Q}. \end{aligned}$$

The goal is to obtain a bound on the error $e_2(x)$, without having to compute M or $M_{\mathbf{n}\setminus Q}$. We hope to find a bound in terms of the d_i, and perhaps in terms of properties of the H matrix. We assume, without loss of generality, that the columns

of H are normalized to have unit 2-norm. Then, we have

$$\begin{aligned}
e_2(Q) &= \|M^{-1/2} M_{\mathbf{n}\setminus Q} M^{-1/2}\| \\
&= \|M^{-1} M_{\mathbf{n}\setminus Q}\| \\
&= \|M_{\mathbf{n}\setminus Q} M^{-1}\| \\
&= \|M_{\mathbf{n}\setminus Q}(M_Q^{-1} + \sum_{k=1}^{\infty}(-M_{\mathbf{n}\setminus Q} M_Q^{-1})^k)\| \\
&\leq \|M_{\mathbf{n}\setminus Q} M_Q^{-1}\| + \|M_{\mathbf{n}\setminus Q} M_Q^{-1}\|^2 + c\|M_{\mathbf{n}\setminus Q}\|^3 \\
&\lesssim \|M_{\mathbf{n}\setminus Q} M_Q^{-1}\| =: e_2^{\mathrm{a}}(Q) \\
&\leq \|M_{\mathbf{n}\setminus Q}\| \|M_Q^{-1}\| =: b_2^{a}(Q)
\end{aligned}$$

The approximate bound $e_2^{\mathrm{a}}(x)$ was found to be a very good estimate for $e_2(x)$ in our preliminary experiments, and its elementary bound $b_2^{\mathrm{a}}(Q)$ was quite good as well. Unfortunately, we must avoid explicit formation of $M_{\mathbf{n}\setminus Q}$, and so we seek to further bound $\|M_{\mathbf{n}\setminus Q}\|$ "cheaply". One possibility is the following:

$$\begin{aligned}
\|M_{\mathbf{n}\setminus Q}\| &= \lambda_{\max}(M_{\mathbf{n}\setminus Q}) \\
&\leq \sum_{i=1}^{m} \lambda_i(M_{\mathbf{n}\setminus Q}) = \operatorname{trace}(M_{\mathbf{n}\setminus Q}) \\
&= \operatorname{trace}\left(\sum_{i\in\mathbf{n}\setminus Q} D_{ii}^2 h_i h_i^{\mathrm{T}}\right) \\
&= \sum_{i\in\mathbf{n}\setminus Q} D_{ii}^2 \operatorname{trace}\left(h_i h_i^{\mathrm{T}}\right) \\
&= \sum_{i\in\mathbf{n}\setminus Q} D_{ii}^2 h_i^{\mathrm{T}} h_i = \sum_{i\in\mathbf{n}\setminus Q} D_{ii}^2.
\end{aligned} \tag{7.19}$$

This bound is cheap, but unfortunately, in practice, it is too conservative. The following heuristic (for which we have no good explanation) seems to work well and

is inexpensive to evaluate:

$$\|M_{\mathbf{n}\setminus Q}\| \lesssim \frac{1}{\sqrt{|\mathbf{n}\setminus Q|}} \sum_{i\in\mathbf{n}\setminus Q} D_{ii}^2. \tag{7.20}$$

To test the plausibility of this approach, we generated a random 100×10000 standard form LP and solved it (with rMPC*), we stopped the algorithm at iterations 8, 10, 12, and 14, and computed the errors in the approximation of the normal matrix by M_Q. (Actually we used here $M = AXS^{-1}A^{\mathrm{T}}$ in place of the $M = HD^2H^{\mathrm{T}}$, as it is the quantity available using rMPC*, and similarly for M_Q, but these are closely related, and this method is just meant to get reasonable values for D_{ii}^2 for the test.) Figure 7.1 shows the true error in the Hessian approximation that uses a Q corresponding to the largest values of $D_{ii}^2 = (S_{ii}^{-1}X_{ii})$. Two of the approximate error bounds are also shown. The size of the constraint set $|Q|$ is plotted on the horizontal axis. The horizontal line corresponds to $\varepsilon_2 = 0.08$ (which is a reasonable value according to [Sch06]) and when the error, or an upper bound on the error, falls below this line we may ignore the remaining constraints corresponding to D_{ii}^2 smaller than the crossing point. The approximate bound $\|M_{\mathbf{n}\setminus Q}\|\|M_Q^{-1}\|$ is quite good ($\|M_{\mathbf{n}\setminus Q}M_Q^{-1}\|$, not shown, is even better), while the heuristic from (7.20) is excellent (although it underestimates the error a little in the last plot). The bound $\mathrm{trace}(M_{\mathbf{n}\setminus Q})\|M_Q^{-1}\|$ (not shown) is far too conservative, and unfortunately, is essentially useless for this example. In this example, if we use the heuristic bound, at iteration 10, we can drop about half of the constraints and still keep the error below the threshold, while by iteration 12 to 14, we can drop most of them. In practice, determining an appropriate Q could be done in a minor-cycle, where we choose an initial constraint set Q, evaluate the bound for $e_2(Q)$, call it $b(Q)$, and if $b(Q) \le \varepsilon_2$ (implying $e_2(Q) \le \varepsilon_2$), then we can use the corresponding approximation in computing the step, whereas, if $b(Q) > \varepsilon_2$, then we need to add additional

constraints. If we could tolerate somewhat more error, then the allowable reduction may be significantly greater, so it would be worth investigating if the analysis of Schurr could be refined to allow a larger ε_2. This is a very preliminary test and it just gives an idea of what might be possible. There is a large body of work on matrix theory that we expect can be brought to bear on this problem to help obtain reasonably tight, and cheaply computable bounds on $e_2(Q)$.

If we indeed are able to obtain such bounds then we can use the results of Schurr to show that, with appropriate choice of parameters θ,[5] τ, ε_1, and ε_2, the following algorithm has an iteration complexity bound of $\mathcal{O}(\sqrt{n}\log\frac{1}{\varepsilon})$ (note we may require a "minor-cycle" in the first step), and hopefully considerably less work per iteration.

Algorithm 13: Constraint-reduced algorithm for conic (at least linear) programming

Input: CP Data: A, b, c, K, Initial iterate: $(x_0, y_0, s_0) \in \mathcal{N}(\theta)$;

Parameters: $\theta \in (0,1), \tau > 0, \varepsilon > 0, \varepsilon_1 > 0, \varepsilon_2 > 0$;

Output: ε-optimal primal-dual solution (x, y, s)

while $\mu \geq \varepsilon$ **do**

Select Q so that $e_1(Q) \leq \varepsilon_1$ and $e_2(Q) \leq \varepsilon_2$;

Compute $F_1(x) = \sum_{i \in Q} f_i'(x)$;

Compute $F_2(x) = \sum_{i \in Q} f_i''(x)$;

Solve system (7.13) for $(\Delta x, \Delta y, \Delta s)$;

Set $(x^+, y^+, s^+) := (x, y, s) + (\Delta x, \Delta y, \Delta s)$;

Set $\mu^+ = \tau\mu$;

end

[5]This parameter defines the $\mathcal{N}(\theta)$ "neighborhood" of the central path. See [Sch06, SOT09] for details.

7.5 Extension to general nonlinear, possibly non-convex, problems

At first glance there is no reason why these constraint-reduction methods should not extend to interior-point methods for general inequality constrained nonlinear programming. In fact, the extension to convex QP has already been done [JOT10]. This should be particularly straightforward if the constraints are linear. In contrast to the relatively large amount of work on constraint-reduction in IPMs for LP, excluding the work of Jung on convex QP [JOT10] and den Hertog's work in [HKRT95], there appears to be little prior work for more general cases.

We consider the general inequality constrained problem

$$\begin{aligned} &\min \ f_0(y) \\ &\text{s.t. } f_i(y) \leq 0 \quad i = 1, 2, \ldots n, \end{aligned} \tag{7.21}$$

where $y \in \mathbb{R}^m$, and we assume $n \gg m$. The barrier method for solving (7.21) takes Newton steps on the logarithmic barrier subproblem

$$\min \ f_0(y) + \tau \sum_{i}^{n} -\log(-f_i(y)), \tag{7.22}$$

for decreasing values of $\tau > 0$. The log-barrier function for the region $\{y, \mid f_i(y) \leq 0, \ i = 1, 2, \ldots, n\}$ is

$$\varphi(y) = -\sum_{i=1}^{n} \log(-f_i(y)),$$

which has gradient

$$\nabla\varphi(y) = \sum_{i=1}^{n} \frac{\nabla f_i(y)}{f_i(y)},$$

and Hessian

$$\nabla^2\varphi(y) = \sum_{i=1}^{n} \frac{\nabla^2 f_i(y)}{f_i(y)} - \frac{\nabla f_i(y)\nabla f_i(y)^{\mathrm{T}}}{f_i(y)^2}.$$

If, selecting a small set Q of working constraints, we instead compute the Newton step for the problem

$$\begin{aligned} &\min \ f_0(y) \\ &\text{s.t. } f_i(y) \le 0 \quad i \in Q, \end{aligned} \tag{7.23}$$

with simplified barrier

$$\varphi_Q(y) := f_0(y) + \tau \sum_{i \in Q} -\log(-f_i(y)), \tag{7.24}$$

then the expressions for the gradient and Hessian are correspondingly simplified, with sums only over the index set Q. Fewer evaluations of the gradient and Hessian of the constraint functions will be needed, and furthermore the log-barrier Hessian may be considerably more sparse, making the Newton system easier to solve. If we carefully choose the set Q, then we hope that the resulting step will be good for the original problem.

We can try similar things for the primal-dual interior-point method which takes Newton steps on the equality portion of the perturbed KKT conditions for (7.21):

$$\begin{aligned} f(y) + s &= 0, \\ \nabla_y L(y,x) = \nabla_y f_0(y) + A(y)x &= 0, \\ Xs - \tau e &= 0, \\ (x,s) &\ge 0, \end{aligned} \tag{7.25}$$

where

$$L(y, x) := f_0(y) + x^{\mathrm{T}} f(y)$$

is the Lagrangian for the problem, $\tau \geq 0$ is the perturbation parameter, x the vector of Lagrange multipliers for the inequality constraints,

$$A(y)^{\mathrm{T}} := \begin{pmatrix} \nabla f_1(y)^{\mathrm{T}} \\ \nabla f_2(y)^{\mathrm{T}} \\ \vdots \\ \nabla f_n(y)^{\mathrm{T}} \end{pmatrix}, \quad f(y) := \begin{pmatrix} f_1(y) \\ f_2(y) \\ \vdots \\ f_n(y) \end{pmatrix}, \quad X := \operatorname{diag}(x), \quad S := \operatorname{diag}(s),$$

and $s \geq 0$ is a vector of slack variables for the inequalities. The KKT conditions (1.78) for the LP problem (1.3) are a special case of (7.25). The Newton system for the equality portion of (7.25) is (cf. (1.79))

$$\begin{pmatrix} 0 & A(y)^{\mathrm{T}} & I \\ A(y) & \nabla_y^2 L(y, x) & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} -f(y) - s \\ -\nabla_y f(y) - A(y)x \\ \sigma\mu e - Xs \end{pmatrix}, \tag{7.26}$$

where we have set $\tau = \sigma\mu$, with $\mu = x^{\mathrm{T}} s/n$ the current "duality measure" and $\sigma \in [0, 1]$. Similar statements that were made after the introduction of (1.79) apply.

As in the LP case, system (7.26) can be solved by first eliminating Δs to get the symmetric-indefinite augmented system (cf. (1.80))

$$\begin{pmatrix} -X^{-1}S & A(y)^{\mathrm{T}} \\ A(y) & \nabla_y^2 L(y, x) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} -f(y) - \sigma\mu X^{-1} e \\ -\nabla_y f_0(y) - A(y)x \end{pmatrix}, \tag{7.27}$$

$$\Delta s = -A(y)^{\mathrm{T}} \Delta y - f(y) - s,$$

or by further eliminating Δx to get normal equations (cf. (1.81))

$$\begin{aligned} M(x,y)\Delta y &= -\nabla_y f_0(y) - A(y)x + A(y)S^{-1}X(-f(y) - \sigma\mu X^{-1}e), \\ \Delta s &= -A(y)^{\mathrm{T}}\Delta y - f(y) - s, \\ \Delta x &= -S^{-1}X\left(-A(y)^{\mathrm{T}}\Delta y - f(y) - \sigma\mu X^{-1}e\right). \end{aligned} \tag{7.28}$$

where the "normal matrix" is now

$$\begin{aligned} M(y,x) &:= \nabla_y^2 L(y,x) + A(y)S^{-1}XA(y)^{\mathrm{T}} \\ &= \nabla_y^2 L(y,x) + \sum_{i=1}^{n} \frac{x_i}{s_i}\nabla f_i(y)\nabla f_i(y)^{\mathrm{T}} \\ &= \nabla^2 f_0(y) + \sum_{i=1}^{n}\left(x_i\nabla^2 f_0(y) + \frac{x_i}{s_i}\nabla f_i(y)\nabla f_i(y)^{\mathrm{T}}\right). \end{aligned} \tag{7.29}$$

If the problem (7.21) is convex, then, as in the LP case, the normal matrix is positive semi-definite. A simple form of constraint-reduction, analogous to what is done in [TAW06] for the rPDAS algorithm, is to replace the normal matrix by the reduced version

$$M_Q(y,x) := \nabla_y^2 L(y,x) + \sum_{i\in Q} \frac{x_i}{s_i}\nabla f_i(y)\nabla f_i(y)^{\mathrm{T}}, \tag{7.30}$$

or alternatively with

$$\tilde{M}_Q(y,x) := \nabla^2 f_0(y) + \sum_{i\in Q}\left(x_i\nabla^2 f_i(y) + \frac{x_i}{s_i}\nabla f_i(y)\nabla f_i(y)^{\mathrm{T}}\right), \tag{7.31}$$

and use the back-substitutions in (7.28) to define the complete search direction $(\Delta x, \Delta y, \Delta s)$. Another possibility, analogous to what we have done in the LP algorithms developed in this dissertation, would be to replace the size $n+m$ augmented

system (7.27) with the size $n + |Q|$ system

$$\begin{pmatrix} -X_Q^{-1}S_Q & A_Q(y)^{\mathrm{T}} \\ A_Q(y) & \nabla_y^2 L_Q(y,x) \end{pmatrix} \begin{pmatrix} \Delta x_Q \\ \Delta y \end{pmatrix} = \begin{pmatrix} -f_Q(y) - \sigma\mu X_Q^{-1} e \\ -\nabla_y f_0(y) - A_Q(y)x \end{pmatrix}, \tag{7.32}$$

$$\Delta s_Q = -A_Q(y)^{\mathrm{T}}\Delta y - f_Q(y) - s_Q,$$

which defines a it partial primal-dual search direction $(\Delta x_Q, \Delta y)$. This method does not specify the $\mathbf{n} \setminus Q$ component of Δx, and we would need a method for updating $x_{\mathbf{n}\setminus Q}$, as was the case for our constraint-reduced LP algorithms developed in Chapters 2 and 3.

The lines of analysis used in Chapters 2 and 3 are inspired by that of [TAW06], which was influenced by lines of analysis for nonlinear programming developed in [Her82, PTH88], so we believe it is likely that our constraint-reduced algorithms and analysis could be extended to these more general problems, although it would almost certainly be easier to consider convex problems first.

Example: randomly generated geometric programs

As an example showing that this idea has promise, we generated a random geometric program with many constraints. Geometric programs (GPs, see e.g., [BV04]) (when properly formulated) are convex optimization problems of the form (7.21), where

$$f_j(y) = \mathrm{lse}(C_j y + d_j)$$

and

$$\mathrm{lse}(z) := \log\left(\sum e^{z_j}\right),$$

where subscript j is an index, C_j is a $k(j) \times m$ matrix, and $d_j \in \mathbb{R}^{k(j)}$. (Note $\mathrm{lse}(\cdot)$ stands for log-sum-exp.) A large number of problems in engineering design can be

formulated as GPs, see e.g., [BV04, SB07, Chi05].

Suppose that $k(j) = k$, a constant, and we have n constraints. Then the cost of evaluating the constraints is at least $\mathcal{O}(nkm)$. Since

$$\nabla_z \mathrm{lse}(z) = \frac{e^z}{\sum e^{z_j}}$$

(exponentiation applied componentwise) and

$$\nabla_z^2 \mathrm{lse}(z) = \mathrm{diag}\left(\frac{e^z}{\sum e^{z_j}}\right) - \frac{(e^z)(e^z)^{\mathrm{T}}}{(\sum e^{z_j})^2} = \mathrm{diag}\left(\nabla_z \mathrm{lse}(z)\right) - \nabla_z \mathrm{lse}(z) \nabla_z \mathrm{lse}(z)^{\mathrm{T}},$$

the cost of evaluating all of the constraint gradients

$$\nabla_y f_j(y) = C_j^{\mathrm{T}} \nabla_z \mathrm{lse}(C_j y + d_j)$$

is also $\mathcal{O}(nkm)$ and the cost of the evaluating the constraint Hessians

$$\nabla_y^2 f_j(y) = C_j^{\mathrm{T}} \nabla_z^2 \mathrm{lse}(C_j y + d_j) C_j$$

is $\mathcal{O}(n(km^2 + k^2 m))$. Finally, putting together the normal matrix (7.29) costs $\mathcal{O}(nm^2)$. It appears that simply evaluating the constraint Hessians dominates the computation. We summarize this in Table 7.1 along with the corresponding cost when n is reduced to $|Q| < n$. (Some items are marked "same" on the table because, for example, we still have to fully evaluate the constraints in conducting the line search, etc.) We see that, similar to the linear case, it is reasonable to take $|Q| = \mathcal{O}(n/m)$, thus reducing the dominant cost by a factor of $\mathcal{O}(m)$ assuming the number of line-search iterations (see discussion of algorithm implementation below) does not increase significantly. It may be worth determining the order constants in the table, since we may want to know our possible savings when n is only moderately

operation	full cost	reduced cost
evaluate $f_i(y)$, $\nabla_y f_i(y)$	$\mathcal{O}(nkm)$	same
evaluate $\nabla_y^2 f_i(y)$	$\mathcal{O}(n(km^2 + k^2m))$	$\mathcal{O}(\|Q\|(km^2 + k^2m))$
form normal matrix	$\mathcal{O}(mn^2)$	$\mathcal{O}(\|Q\|m^2)$
solve Newton system	$\mathcal{O}(m^3)$	same
line search (LS)	# of LS iter. $\times$ $\mathcal{O}(nkm)$	same

Table 7.1: Cost of iteration of primal-dual method on GP example.

larger than m.

The random problem has $n = 10000$, $m = 100$ and $k(j) = 5$ for $j = 1, 2, \ldots n$. The components of C_j are sampled independently from the standard normal distribution, and d_j is chosen so that the initial point $y^0 = e$ is strictly feasible, with the constraints evaluating to $s^0 > 0$, for some specified set of initial "slack" values.

We used the primal-dual method with the constraints sorted according to nearness to activity, and we kept the M constraints with smallest values of s_i. (Of course, there are other constraint selection rules possible, e.g., choosing them according to size of x_i or x_i/s_i are the most obvious alternatives.) Table 7.2 shows the time and iterations needed for various fixed M to reduce the duality gap to tolerance. These results are pleasantly surprising. The benefit of constraint-reduction appears

M	iterations	time
10000	84	718s
5000	66	363s
2500	55	214s
1000	49	137s
500	41	96s
250	39	83s
100	45	99s

Table 7.2: Performance of constraint-reduction in the primal-dual method with $\sigma = 0.1$ on random GP of size $k = 5$, $m = 100$, $n = 10000$.

to be major on this problem. In particular, recall that in our experiments for LP, the iteration counts generally increased, or stayed roughly constant, as the number of constraints in the working set were decreased. Here, in contrast, a clear decreasing

trend is present: fewer and fewer iterations are needed as we reduce the constraint set! Of course the iterations are cheaper as well when we keep fewer constraints, so this appears to be a double benefit.

Figure 7.2 shows a handpicked example in two dimensions where the constraint-reduced PDIPM requires far fewer iterations to solve the random GP than the corresponding unreduced method. In particular, the figure shows the trajectories of iterates of the primal-dual method with $\sigma = 0.02$ on a random GP with $m = 2$, $k = 5$, and $n = 100$, using constraint-reduction with $|Q| = 6$ (aggressive reduction) vs. $|Q| = 100$ (no reduction). The unreduced trajectory takes a large number of very short steps, 190 iterations in all, and remains near the boundary of the feasible set throughout. We intentionally started the iterations from a bad initial iterate, with the first constraint nearly active; this apparently causes serious problems for the unreduced algorithm on this class of problems. The constraint-reduced method, on the other hand, appears to be much less sensitive to the poor initial point: there is no such stalling, and only 21 iterations are needed. From such poor starting points, this type of behavior was regularly observed on this class of random GPs. It would be rather interesting if it occurs also on large real world problems. The results of Table 7.2 suggest this may be the case on large instances of randomly generated GPs at least.

The algorithm we used is very simple: we computed the Newton step for the KKT system and then use a backtracking line search [BV04] to maintain feasibility with respect to the inequality constraints; feasibility for the multipliers, i.e., non-negativity, can be enforced more simply. We did not require descent or reduction of "dual" infeasibility (i.e., the norm of the Lagrangian gradient) in the line-search; we actually we had some difficulty meeting such a descent condition with the constraint-reduced variant so we took it out. This is related to the discussion in section 7.1, and is something that will most likely need to be considered in developing provably

convergent variants. Potentially interesting and useful future work would be to develop specific constraint-reduced variants of the barrier and primal-dual methods and investigate their theoretical and practical behavior in detail.

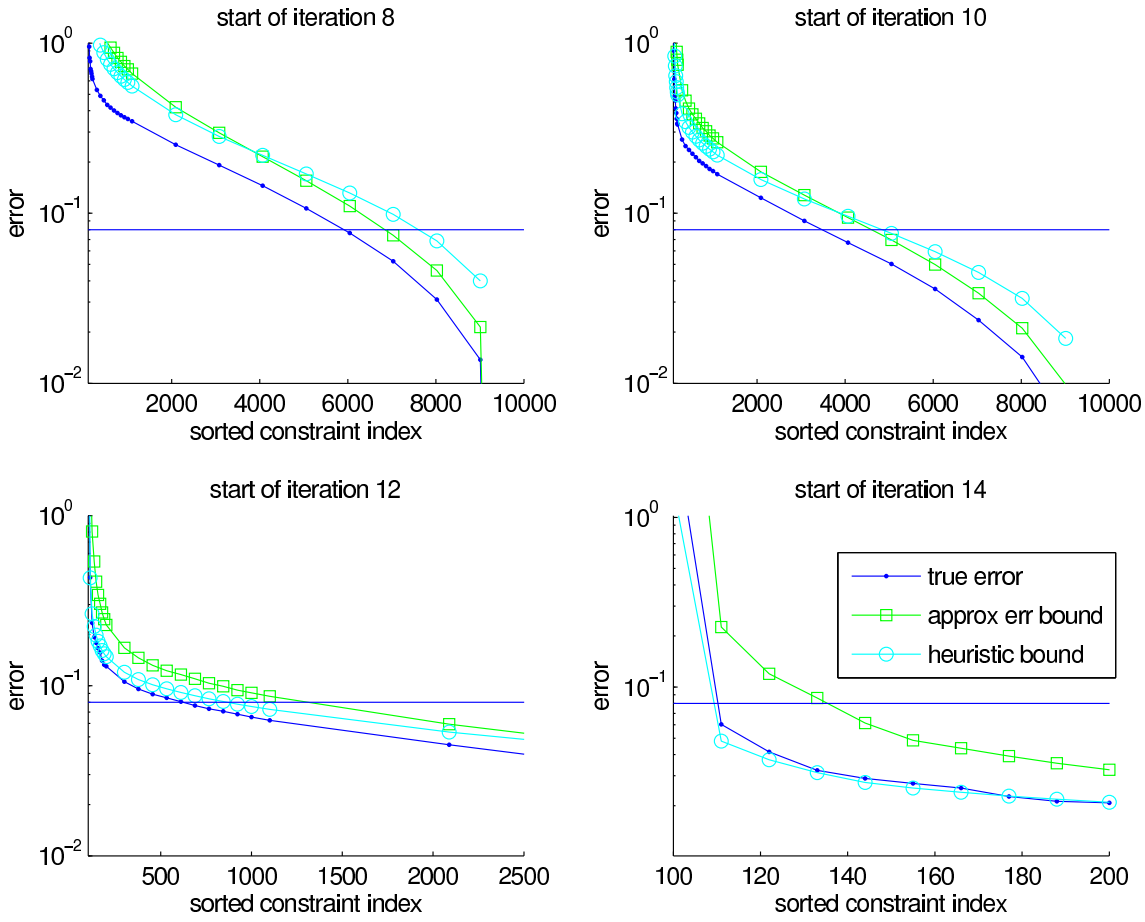

Figure 7.1: Bounding the error in the barrier Hessian approximation. The plots show error in the Hessian partial-sum approximation vs. number of terms in the sum; the terms are taken according to largest value of D_{ii}^2 first. We solved a random 100×10000 LP, using unreduced rMPC* to get reasonable values for D_{ii}^2 (this is not intended to be a totally precise test here) and computed the errors in an approximation at iterations 8, 10, 12, and 14. (The problem was solved to tolerance in 16 iterations total.) Note the difference in scaling of the x-axis on each plot. The horizontal line on each plot specifies the threshold $\varepsilon_2 = 0.08$; all constraints for which the error bound is below this threshold may be omitted, so that by iteration 10, we can drop about half, and by iteration 12 to 14, we can drop almost all of them. The plot shows the exact error, the approximate bound $\|M_{\mathbf{n}\setminus Q}\|\|M_Q^{-1}\|$ (both expensive to evaluate), and the inexpensive heuristic (7.20); all three are in fairly close agreement.

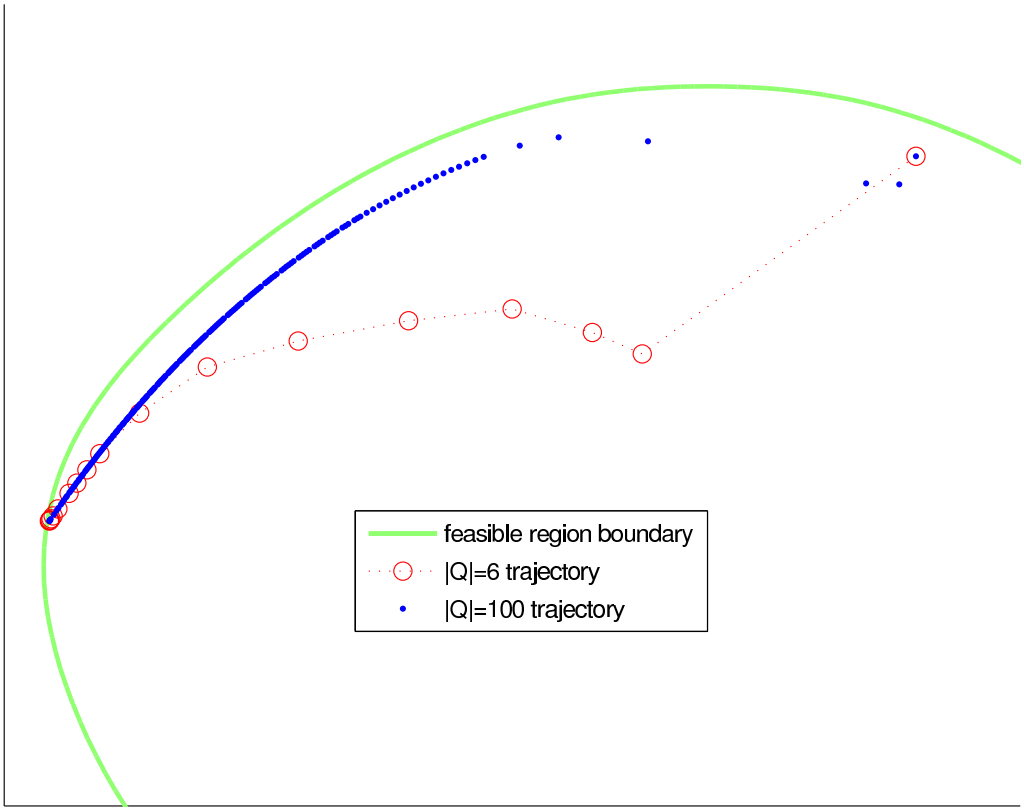

Figure 7.2: Trajectories of iterates of the primal-dual method with $\sigma = 0.02$, on the GP example described above in $n = 2$-dimensional space with $k = 5$ and $m = 100$. We have handpicked this example to show the problems that can occur with many constraints (with small σ) and the potential benefit of constraint-reduction. An intentionally bad initial iterate x^0 is chosen so that $f_i(y^0) = -1$ for $i = 2, 3, \ldots m$, and $f_1(y^0) = 10^{-3}$. The dot-marks show the iterates of the algorithm without constraint-reduction, i.e., using $|Q| = 100$. In this case, the algorithm severely stalls near the boundary: 190 iterations were needed to solve the problem. The circle-marked trajectory shows the iterates of the algorithm using aggressive reduction with $|Q| = 6$. The situation in that case is much better, and only 21 iterations were needed. The thick solid line is the boundary of the feasible region.

Bibliography

[BP95] D. Burnside and T. W. Parks. Optimal design of FIR filters with the complex Chebyshev error criteria. *IEEE Transactions on Signal Processing*, 43(3):605 –616, 1995.

[Bre00] L. Breiman. *Probability*. SIAM Classics in Applied Mathematics, 2000.

[BT97] D. Bertsimas and J. Tsitsiklis. *Introduction to Linear Optimization*. Athena, 1997.

[BT05] D. Bertsekas and J. Tsitsiklis. *Dynamic Programming and Optimal Control, Vols.1&2*. Athena scientific, 2005.

[BV04] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[BV07] S. Boyd and L. Vandenberghe. Localization and cutting plane methods, 2007. Course Notes, Convex Optimization II, Stanford University.

[BVS] S. Boyd, L. Vandenberghe, and J. Skaf. analytic center cutting plane methods. Course Notes, Convex Optimization II, Stanford University. 2007.

[Car09] C. Cartis. Some disadvantages of a mehrotra-type primal-dual corrector interior point algorithm for linear programming. *Applied Numerical Mathematics*, 59:1110–1119, 2009.

[CC05] G. Calafiore and M. C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[CC06] G. Calafiore and M.C. Campi. The scenario approach to robust control design. *IEEE Trans. on Automatic Control*, 51(5):742–753, 2006.

[CG08] M. C. Campi and S. Garatti. The exact feasibility of randomized solutions of robust convex programs. *SIAM Journal on Optimization*, 19(3):1211–1230, 2008.

[Chi05] M. Chiang. Geometric programming for communication systems. *Commun. Inf. Theory*, 2(1/2):1–154, 2005.

[dFR03] D. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[dFR04] D. de Farias and B. Van Roy. On constraint sampling for the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478, 2004.

[Dik67] I. I. Dikin. Iterative solution of problems of linear and quadratic programming. *Doklady Akademiia Nauk SSSR*, 174:747–748, 1967. English Translation: *Soviet Mathematics Doklady*, 1967, Volume 8, pp. 674–675.

[Dik74] I. I. Dikin. On convergence of an iterative process. *Upravlyaemye Systemy*, 12:54–60, 1974. In Russian.

[DNPT06] A. Deza, E. Nematollahi, R. Peyghami, and T. Terlaky. The central path visits all the vertices of the Klee-Minty cube. *Optimization Methods and Software*, 21(5):851–865, 2006.

[DW60] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programming. *Operations Research*, 8(1):101–111, 1960.

[DY91] G. Dantzig and Y. Ye. A build-up interior-point method for linear programming: Affine scaling form. Working paper, Department of Management Science, University of Iowa, 1991.

[EM75] J. Elzinga and T. J. Moore. A central cutting plane algorithm for the convex programming problem. *Mathematical Programming*, 8(1):134–145, 1975.

[FO07] M. P. Friedlander and D. Orban. Exact primal-dual regularization of linear programs, 2007. Presentation given at ICCOPT: Hamilton, Ontario.

[GL83] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1983.

[Her82] J. N. Herskovits. *Développement d'une Méthode Numérique pour l'Optimization Non-Linéaire*. PhD thesis, Université Paris IX - Dauphine, Paris, France, 1982.

[Her92] D. den Hertog. *Interior Point Approach to Linear, Quadratic and Convex Programming, Algorithms and Complexity*. PhD thesis, Faculty of Mathematics and Informatics, TU Delft, NL–2628 BL Delft, The Netherlands, September 1992. Subsequently published by Kluwer Publishers, Dordrecht, The Netherlands, 1994.

[Hig90] N. J. Higham. Analysis of the Cholesky decomposition of a semi-definite matrix. In M. G. Cox and S. J. Hammarling, editors, *Reliable Numerical Computation*, pages 161–185. Oxford University Press, 1990.

[HK93] R. Hettich and K. O. Kortanek. Semi-infinite programming: theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[HKRT95] D. den Hertog, J. Kaliski, C. Roos, and T. Terlaky. A logarithmic barrier cutting plane method for convex programming problems. *Annals of Operations Research*, 58:69–98, 1995.

[Hof52] A. J. Hoffman. On approximate solutions of systems of linear inequalities. *Journal of Research of the National Bureau of Standards*, 49:263–265, 1952.

[HRT94] D. den Hertog, C. Roos, and T. Terlaky. Adding and deleting constraints in the path-following method for LP. In D.Z. Du and J. Sun, editors, *Advances in Optimization and Approximation*, pages 166–185. Kluwer Academic Publishers, 1994.

[HT10] M. He and A. L. Tits. An infeasible constraint-reduced interior-point method for linear programming, 2010. Presented at the 6th Northeast Control Workshop, Johns Hopkins University in Baltimore, Maryland.

[HUL00] J. Hiriart-Urruty and C. Lemarachal. *Fundamentals of Convex Analysis*. Springer, 2000.

[JOT08] J. H. Jung, D. P. O'Leary, and A. L. Tits. Adaptive constraint reduction for training support vector machines. *Electronic Transactions on Numerical Analysis*, (31):156–177, 2008.

[JOT10] J. H. Jung, D. P. O'Leary, and A. L. Tits. Adaptive constraint reduction for convex quadratic programming. *Computational Optimization and Applications*, 2010. To appear.

[JRT90] B. Jansen, C. Roos, and T. Terlaky. A polynomial primal-dual Dikin-type algorithm for linear programming. *Mathematics of Operations Research*, 21(2):341–353, 1990.

[Kar84] N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[Kel60] J. E. Kelley. The cutting plane method for solving convex programs. *Journal of the SIAM*, 8(4):703–712, 1960.

[Kha79] L. G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademiia Nauk SSSR*, 224:1093–1096, 1979. English Translation: *Soviet Mathematics Doklady*, Volume 20, pp. 191–194.

[KM72] V. Klee and G. J. Minty. How good is the simplex algorithm? In O. Shisha, editor, *Inequalities, III*, pages 159–175. Academic Press, 1972.

[KM95] L. J. Karam and J. H. McClellan. Complex Chebyshev approximation for FIR filter design. *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, 42(3):207–216, 1995.

[Kor00] J. Korzak. Convergence analysis of inexact infeasible-interior-point algorithms for solving linear programming problems. *SIAM Journal on Optimization*, 11(1):133–148, 2000.

[KT93] L. G. Khachiyan and M. J. Todd. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61(2):137–159, 1993.

[KY93] J. A. Kaliski and Y. Ye. A short-cut potential reduction algorithm for linear programming. *Management Science*, 39:757–776, 1993.

[LB97] H. Lebret and S. Boyd. Antenna array pattern synthesis via convex optimization. *IEEE Transactions on Signal Processing*, 45(3):526–532, 1997.

[LMO06] Z. Lu, R. D. C. Monteiro, and J. W. O'Neal. An iterative solver-based infeasible primal-dual path-following algorithm for convex quadratic programming. *SIAM J. on Optimization*, 17(1):287–310, 2006.

[Lue84] D. G. Luenberger. *Linear and Nonlinear Programming*. 2nd edition, Addison-Wesley, Menlo Park, 1984.

[MAR90] R. D. C. Monteiro, I. Adler, and M. G. C. Resende. A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension. *Math. of Operations Research*, 15:191–214, 1990.

[ME01] P. Misra and P. Enge. *"Global Positioning System: Signals Measurements, and Performance"*. Ganja-Jamuna Press, 2001.

[Meh92] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[MKT95] S. Mizuno, M. Kojima, and M. J. Todd. Infeasible-interior-point primal-dual potential-reduction algorithms for linear programming. *SIAM Journal on Optimization*, 5(1):52–67, 1995.

[MR79] O. L. Mangasarian and R.R.Myer. Nonlinear perturbation of linear programs. *SIAM Journal on Optimization*, 17(6):745–752, 1979.

[MTW93] R. D. C. Monteiro, T. Tsuchiya, and Y. Wang. A simplified global convergence proof of the affine scaling algorithm. *Annals of Operations Research*, 46-47(2):443–482, 1993.

[Nes95] Y. E. Nesterov. Cutting plane algorithms from analytic centers: efficiency estimates. *Mathematical Programming*, 69(1):149–176, 1995.

[Nes03] Y. E. Nesterov. *Introductory lectures on convex optimization: a basic course*, volume 87 of *Applied Optimization*. Kluwer Academic Publishers, Boston, 2003.

[net] Netlib linear programming test problems. http://www-fp.mcs.anl.gov/OTC/Guide/TestProblems/LPtest/.

[Nic09] S. Nicholls. *Column Generation in Infeasible Predictor-Corrector Methods for Solving Linear Programs*. PhD thesis, University of Maryland, College Park, MD, 2009.

[NN93] Y. E. Nesterov and A. S. Nemirovsky. *Interior Point Polynomial Methods in Convex Programming: Theory and Algorithms*. SIAM Publications. SIAM, Philadelphia, USA, 1993.

[NZ99] S. Nordebo and Z. Zang. Semi-infinite linear programming: A unified approach to digital filter design with time and frequency-domain specifications. *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, 46(6):765–775, 1999.

[Off04] Navstar GPS Joint Program Office. *Navstar GPS Space Segment/Navigation User Interfaces (IS-GPS-200D)*, 2004.

[O'L90] D. P. O'Leary. On bounds for scaled projections and pseudo-inverses. *Linear Algebra and its Applications*, 132:115–117, 1990.

[OS99] A. V. Oppenheim and R. W. Shafer. *Discrete Time Signal Processing*. Prentice Hall Signal Processing Series, 1999.

[Pot96] F. Potra. An infeasible-interior-point predictor-corrector algorithm for linear programming. *SIAM Journal on Optimization*, 6(1):19–32, 1996.

[Pot98] A. W. Potchinkov. *Semi-infinite programming*, chapter 5: The design of nonrecursive digital filters via convex optimization. Kluwer, Boston-London-Dordrecht, 1998.

[PP02] A. Papoulis and S. U. Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, NY, 2002.

[PS82] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[PT93] E. R. Panier and A. L. Tits. On combining feasibility, descent and superlinear convergence in inequality constrained optimization. *Mathematical Programming*, 59(1-3):261–276, 1993.

[PTH88] E. R. Panier, A. L. Tits, and J. N. Herskovits. A QP-free, globally convergent, locally superlinearly convergent algorithm for inequality constrained optimization. *SIAM J. Contr. and Optim.*, 26(4):788–811, 1988.

[Ree91] R. Reemtsen. Discretization methods for the solution of semi-infinite programming problems, 1991.

[Ren01] J. Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization*. SIAM, 2001.

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems, Second Edition*. SIAM, 2003.

[Sai94] R. Saigal. On the primal-dual affine scaling method. Technical report, Dept. of Industrial and Operational Engineering, The University of Michigan, 1994.

[Sai96] R. Saigal. A simple proof of a primal affine scaling method. *Annals of Operations Research*, 62(1):303–324, 1996.

[SB07] S. J. Kim A. Hassibi S. Boyd, L. Vandenberghe. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

[Sch91] L. L. Scharf. *Statistical Signal Processing*. Addison Wesley, 1991.

[Sch06] S. Schurr. *Inexact Primal-Dual Path Following Algorithm for convex conic optimization*. PhD thesis, University of Maryland, College Park, MD, 2006.

[SN82] R. L. Streit and A. H. Nuttall. A general Chebyshev complex function approximation procedure and an application to beamforming. *J. Acoust. Soc. Am*, 71(1):181–190, 1982.

[SOT09] S. P. Schurr, D.P. O'Leary, and A.L. Tits. A polynomial-time interior-point method for conic optimization, with inexact barrier evaluations. *SIAM Journal on Optimization*, 20(1):548–571, 2009.

[SPT07] M. Salahi, J. Peng, and T. Terlaky. On Mehrotra-type predictor-corrector algorithms. *SIAM Journal on Optimization*, 13(4):1377–1397, 2007.

[SS01] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[ST96] M. A. Saunders and J. A. Tomlin. Solving regularized linear programs using barrier methods and KKT systems. Technical report, Stanford University, Department of EES, 1996.

[Ste89] G. W. Stewart. On scaled projections and pseudo-inverses. *Linear Algebra and its Applications*, 112:189–194, 1989.

[TAO06] A. L. Tits, P. A. Absil, and D. P. O'Leary. Constraint reduction for certain degenerate linear programs. July 30-August 4 2006. presented at 19th ISMP.

[TAW06] A.L. Tits, P.A. Absil, and W. Woessner. Constraint reduction for linear programs with many constraints. *SIAM Journal on Optimization*, 17(1):119–146, 2006.

[Tod90] M. J. Todd. A Dantzig-Wolfe like variant of Karmarkar's interior-point linear programming algorithm. *Operations Research*, 38(6):1006–1018, 1990.

[Ton93] K. Tone. An active-set strategy in an interior point method for linear programming. *Mathematical Programming*, 59(1-3):345–360, 1993.

[Tre02] H. L. Van Trees. *Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory*. John Wiley and Sons, Inc., New York, 2002.

[TS93] M. Trick and E. Stanley. A linear programming approach to solving stochastic dynamic programs, 1993. Working Paper, Carnegie Mellon University.

[TZ94] A.L. Tits and J.L. Zhou. A simple, quadratically convergent algorithm for linear and convex quadratic programming. In W.W. Hager, D.W. Hearn, and P.M. Pardalos, editors, *Large Scale Optimization: State of the Art*, pages 411–427. Kluwer Academic Publishers, 1994.

[Van96] R. J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, Boston, 1996. Second Edition: 2001.

[Vig99] J. R. Vig. IEEE standard definitions of physical quantities for fundamental frequency and time metrology—random instabilities. *IEEE standard - 1139-1988*, 1999.

[VL88] R. J. Vanderbei and J. C. Lagarias. I. I. Dikin's convergence result for the affine-scaling algorithm. In J. C. Lagarias and M. J. Todd, editors, *Mathematical Developments Arising from Linear Programming: Proceedings of a Joint Summer Research Conference*, pages 109–119, Bowdoin College, Brunswick, Maine, USA, 1988. American Mathematical Society, Providence, RI, USA, 1990.

[WBV98] S.P. Wu, S. Boyd, and L. Vandenberghe. *Applied Computational Control, Signal and Circuits*, chapter 5: FIR Filter design by Spectral Factorization and Convex Optimization, pages 215–245. Birkhauser, 1998.

[WNTO10] L. B. Winternitz, S. O. Nicholls, A. L. Tits, and D. P. O'Leary. A constraint reduced variant of Mehrotra's predictor-corrector algorithm, 2010. Submitted for publication. Available online: http://www.optimization-online.org/DB_HTML/2007/07/1734.html.

[WO00] W. Wang and D.P. O'Leary. Adaptive use of iterative methods in predictor-corrector interior point methods for linear programming. *Numerical Algorithms*, 25(1–4):387–406, 2000.

[Wri97] S. J. Wright. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, 1997.

[Ye91] Y. Ye. An $O(n^3L)$ potential reduction algorithm for linear programming. *Mathematical Programming*, 50(2):239–258, 1991.

[Ye92] Y. Ye. A potential reduction algorithm allowing column generation. *SIAM J. on Optimization*, 2(1):7–20, 1992.

[Ye97] Y. Ye. *Interior Point Algorithms: Theory and Analysis*. John Wiley, New York, 1997.

[ZZ95] Y. Zhang and D. Zhang. On polynomiality of the Mehrotra-type predictor-corrector interior point algorithms. *Mathematical Programming*, 68(3):303–31, 1995.

[ZZ96] D. Zhang and Y. Zhang. A Mehrotra-type predictor-corrector algorithm with polynomiality and Q-subquadratic convergence. *Annals of Operations Research*, 62(1):131–150, 1996.