ABSTRACT

Title of dissertation: COLUMN GENERATION IN INFEASIBLE PREDICTOR-CORRECTOR METHODS FOR SOLVING LINEAR PROGRAMS

Stacey O. Nicholls, Doctor of Philosophy, 2009

Dissertation directed by: Professor Dianne P. O'Leary
Department of Computer Science
Institute for Advanced Computer Studies

Primal-dual interior-point methods (IPMs) are distinguished for their exceptional theoretical properties and computational behavior in solving linear programming (LP) problems. Consider solving the primal-dual LP pair using an IPM such as a primal-dual Affine-Scaling method, Mehrotra's Predictor-Corrector method (the most commonly used IPM to date), or Potra's Predictor-Corrector method. The bulk of the computation in the process stems from the formation of the normal equation matrix, AD^2A^T, where $A \in \Re^{m \times n}$ and $D^2 = S^{-1}X$ is a diagonal matrix. In cases when $n \gg m$, we propose to reduce this cost by incorporating a column generation scheme into existing infeasible IPMs for solving LPs. In particular, we solve an LP problem based on an iterative approach where we select a "small" subset of the constraints at each iteration with the aim of achieving both feasibility and optimality. Rather than n constraints, we work with $k = |Q| \in [m, n]$ constraints at each iteration, where Q is an index set consisting of the k most nearly active constraints at the current iterate. The cost of the formation of the matrix, $A_Q D_Q^2 A_Q^T$,

reduces from $\Theta(m^2n)$ to $\Theta(m^2k)$ operations, where k is relatively small compared to n. Although numerical results show an occasional increase in the number of iterations, the total operation count and time to solve the LP using our algorithms is, in most cases, small compared to other "reduced" LP algorithms.

COLUMN GENERATION IN INFEASIBLE PREDICTOR-CORRECTOR METHODS FOR SOLVING LINEAR PROGRAMS

by

Stacey O. Nicholls

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
2009

DEDICATION

This dissertation is dedicated to my sons:

Tyler Adedapo Akanji Jagun

and

Sean Adekunle Ayinde Jagun

ACKNOWLEDGMENTS

My journey to completing this dissertation came with many obstacles and challenges. However, there were a great number of people who supported my efforts to help make this thesis possible. I am truly blessed to have had the opportunity to learn from, work with, and be mentored by those who made my graduate experience one that I will always cherish.

First and foremost, I would like to thank my advisor, Dr. Dianne P. O'Leary. Words cannot express how grateful I am to have worked with such an extraordinarily brilliant woman. She gave me very challenging, yet interesting projects to work on over these past few years, and her advice was invaluable. She was always available for me when I needed help and/or advice on issues related to and beyond my research. She has given me enthusiasm for the area of Numerical Optimization, an eye for detail in working mathematical proofs and describing computational results, and an appreciation for all of the time and energy she devotes to her students. She is an exceptional professor and mentor, and I aspire to be just as influential to my students and their careers as she has been to me.

I would also like to thank Dr. André L. Tits. This thesis would not have been possible without his remarkable theoretical ideas and expertise. He has always given me helpful suggestions and advice for my research. He was a valuable asset to me in this process, and I thank him for all he has offered.

Thank you to Dr. André L. Tits, Dr. Eric V. Slud, Dr. Konstantina Trivisa, and Dr. Michael Laskowski for agreeing to serve on my thesis committee. I greatly appreciate the time and energy you dedicated to reviewing the thesis and providing suggestions for revisions.

During my years at UMCP, I was fortunate to receive funding from a number of sources. UMCP offered not only an opportunity for me to gain valuable information from professors, but also an opportunity to teach undergraduate mathematics courses. I taught lecture and recitation classes as a teaching assistant for four years. I must thank Dr. William Schildknecht for giving me the opportunity to teach a variety of mathematics courses at UMCP. These opportunities opened doors for teaching jobs at other universities, and I am grateful for this teaching support. I would also like to thank Dr. Johnetta Davis for her assistance throughout my graduate years. She was pivotal in helping me obtain a GAANN fellowship for which I am truly thankful. Dr. Davis also served as one of the PIs for the Promise Program, a program in which I gained valuable insight about the process to the doctorate. I would like to thank Dr. Michael Boyle for the opportunity to work under the VIGRE grant. This support provided me with an opportunity to work with Dr. Jeffrey Cooper and Dr. Tobias von Petersdorff in designing MATLAB projects for undergraduate students in computational mathematics using concepts in numerical analysis, differential equations, and optimization. I was also funded under the Department of Energy grant DEFG0204ER25655: "Interior Point Algorithms for Optimization Problems", and I would like to thank Dr. Dianne P. O'Leary and Dr. André L. Tits for this support. Lastly, I would like to thank Dr. Daniel Symancyk and the rest of the hiring committee from the Mathematics Department at Anne Arundel Community College for believing in my ability to work as a full-time mathematics professor while finishing my graduate studies.

The staff at UMCP has been extremely helpful throughout my graduate years. I would like to especially thank Ms. Alverda McCoy for all of her help, support, and advice with the AMSC program.

Many of my most memorable graduate school moments are due to a group of friends that: (1) helped make my transition from undergraduate to graduate school a smooth one, (2) offered exceptional advice to help me reach certain milestones in my program, and (3) provided a friendship that will last a lifetime. I particularly would like to give a big thanks to my friend and former roommate, Ms. Joycelyn Wilson, for her immeasurable support throughout the years. I would also like to thank my graduate mentor Dr. Tasha Inniss, for her guidance in my early graduate school years and Dr. Monica Jackson, for her help as a fellow teaching assistant. I greatly appreciate the help and support of all of my friends. Thank you!

My decision to pursue graduate study was largely due to the exceptional faculty in the Mathematics Department at Spelman College. I would like to thank Dr. Sylvia Bozeman for encouraging me to apply to graduate school. I would also like to thank Dr. Nagambal Shah and Dr. Yewande Olubummo for their constant support during my years of study as a math major at Spelman.

I could not have completed my graduate program without the invaluable support of my family. To my fianceé, Adebola "Taheel" Jagun, thank you for your continual encouragement over the years. Our sons, Tyler and Sean Jagun, are a blessing from God, and I am so thankful to have them. They made my dissertation progress even more of a challenge (many sleepless nights) and yet they were my motivation for finishing the degree. I am also truly grateful for the endless love and support of my parents, Herbert and Jacqueline Nicholls. Thank you for always believing in me! To my sister, Cynthia Nicholls, you are a special person in my life and I want to thank you for just being you. Thank you to all of my family members for their encouraging words and support.

It is truly impossible to remember all, and I apologize to those I've inadvertently left out.

Lastly, I faced obstacles and challenges at just about every stage of my graduate program. God made a way for me to overcome these obstacles with the support of the individuals mentioned here. Thank you God for ALL you have provided me!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

For nearly 40 years, the simplex method was deemed the most efficient algorithm for solving linear programming (LP) problems, a class of problems involving the minimization or maximization of a linear function subject to linear constraints. Although the simplex algorithm can be shown to exhibit an exponential number of iterations in the worst case, on practical problems the number of iterations is usually linear in the size of the problem. Furthermore, the operation count per iteration is also rather low; an $m \times m$ linear system is solved at each iteration, where m is the number of constraints in the primal LP. [1]

The work of Narendra Karmarkar [15], however, spawned a new class of methods, interior-point methods (IPMs), capable of outperforming the simplex method on large-scale LPs and (in general) producing polynomial complexity results. Background information on simplex and interior-point methods can be found in [5], [20] and [29]. Unlike the simplex method, IPMs solve a $(2n+m) \times (2n+m)$ linear system at each iteration where most of the computation stems from solving an $m \times m$ system with normal equation matrix AD^2A^T (A is an $m \times n$ constraint matrix and D is a positive diagonal matrix). If $n \gg m$, the operation count per iteration can be quite large. Nonetheless, their low iteration count and speed make

[1] The primal LP is composed of a system of m equality constraints with n variables. The dual LP is the “companion” to the primal LP in which the roles of constraints and variables are reversed. Therefore, the dual LP is composed of a system of n constraints with m variables.

IPMs the method of choice on large-scale LPs. Today, much attention is focused on primal-dual IPMs [29]; they stand out for their exceptional theoretical properties and computational behavior. Most primal-dual IPM codes today are based on Mehrotra's Predictor-Corrector Method (MPC) [17]. MPC combines the essence of the primal-dual framework with ingenious heuristics.

In this thesis, we devise several "reduced" variants of existing infeasible interior-point algorithms [2] for solving primal-dual LP pairs. Specifically, we examine the incorporation of a column generation scheme into algorithms which, through a sequence of iterations, aim to achieve both feasibility and optimality. In our context, a column generation approach refers to considering only a subset of columns of A, or equivalently, a subset of the dual constraints, at each iteration. This reduces the operation count per iteration and, if the subsets are chosen well, we generally observe no increase in the total number of iterations. Our algorithms can outperform other column generation algorithms in total operation count and CPU time.

A vast number of papers have been written on the convergence of infeasible interior-point algorithms. Kojima et al. [16], Potra [22], and Achache et al. [1] devise infeasible algorithms which improve feasibility and find optimal solutions for the primal-dual pair. The algorithms also demonstrate both global convergence and polynomial complexity.

Column generation algorithms within interior-point methods have been analyzed as well. Algorithms for "building up" the number of constraints at each

[2] An "infeasible algorithm" refers to an algorithm that accepts an initial solution that does not satisfy all of the constraints of an LP.

iteration have been developed by Dantzig et al. [6], Ye [31], Hertog et. al. [11], and Goffin et. al. [9]. Starting with a small subset of dual (working) constraints, an iterative search for the optimal solution is conducted until there is a constraint blocking the progression to this solution. Depending on the algorithm, the violated constraint or constraint generated by the algorithm is added to the set, and the process repeats until the optimal solution has been found. Ye [32] considers adding more than one constraint to the working set at each iteration. In all of these algorithms, constraints are never removed from the working set in the current iteration. Ye [30], on the other hand, developed a "build-down" scheme in which a constraint is discarded from the full constraint set if it is detected to be nonbinding in the optimal set. Convergence properties for the aforementioned algorithms are provided along with polynomial complexity analyses for some. Both the "build-up" and "build-down" approaches were combined and analyzed in Hertog et al. [12] where the algorithm was shown to terminate in polynomial time. More recently, a constraint reduction approach was developed in a primal-dual framework [3] by Tits-Absil-Woessner [24]. The authors consider replacing the normal equation matrix for the primal-dual LP with a "reduced" matrix $A_Q D_Q^2 A_Q^T$ by solving a subset of $m \leq |Q| \leq n$ dual constraints at each iteration where Q is a fixed index set associated with the most "promising" dual constraints at the current iteration. Global convergence and local quadratic convergence are proved for their "reduced" affine-scaling algorithm.

The two algorithms we develop in this thesis are *redPDAS* and *redMPC*. The

[3] Here we are working with the primal and dual LPs simultaneously as opposed to working with them individually. The primal-dual framework involves solving the optimality conditions to satisfy both the primal and dual LPs.

redPDAS algorithm is an extension of the Tits-Absil-Woessner algorithm [24] to handle dual infeasible iterates, and the *redMPC* algorithm is a "reduced" version of Mehotra's Predictor-Corrector Method. We allow both primal and dual infeasibility as opposed to Tits-Absil-Woessner who allow only primal infeasibility in the PDAS case.

The remainder of the thesis focuses on the two algorithms, their convergence, and their performance. The column generation scheme used within our algorithms is presented in Chapter 2. In the beginning of Chapter 3, a general discussion of affine-scaling methods is given. The remainder of the chapter includes a description of the Reduced Primal-Dual Affine-Scaling *redPDAS* algorithm along with some convergence results. We also show that with certain modifications, the algorithm is globally and locally convergent. A general discussion of Mehrotra's Predictor-Corrector method is given at the beginning of Chapter 4. The Reduced Mehrotra's Predictor-Corrector *redMPC* algorithm is presented afterward, along with some convergence results. With modifications, the *redMPC* algorithm is also shown to be globally and locally convergent. Numerical experiments are presented for the $redPDAS$ and $redMPC$ algorithms in Chapter 5 as well as a discussion of their results. Chapter 6 provides the conclusion to the dissertation and directions for future work.

Chapter 2

Background

Interior-point methods (IPMs) for a linear program (LP) generate points which lie in the interior of the region defined by the constraints of the problem. A special class of IPMs, primal-dual methods, exhibit exceptional theoretical properties and computational performance. This dissertation examines the effects of incorporating a column generation scheme into primal-dual IPM algorithms. To understand the algorithm framework in the following chapters, we present some background information in this chapter.

The chapter is divided into two sections: (1) the optimality conditions for a primal-dual LP and (2) column generation. In section 2.1, we begin with the standard notation used to express the primal and dual LPs and their optimality conditions. Next, we show how a primal-dual IPM algorithm finds a solution to these optimality conditions. In section 2.2, column generation is discussed in the context of our algorithm framework. We introduce the notation associated with our algorithms and explain how the index set Q associated with the reduced (dual) constraint set is formulated. Finally, we explain how to preserve a strictly interior point (or solution) within our algorithms.

2.1 Optimality Conditions for the Primal-Dual LP

Consider the primal LP in standard form,

$$\min_{x} \{c^T x : Ax = b,\ x \geq 0\}, \tag{2.1}$$

where $A \in \Re^{m \times n}$, $b \in \Re^m$, and $c \in \Re^n$ are the known coefficient matrix and vectors, respectively, and $x \in \Re^n$ is a vector of unknown variables. All vectors in this thesis are column vectors. It is assumed that A has full row rank, $b, c \neq 0$, and $n \gg m$. Solving the primal LP is equivalent to solving the dual LP,

$$\max_{y} \{b^T y : A^T y \leq c\}.$$

An alternative expression of the dual problem results by incorporating a nonnegative slack vector, s, into the dual constraints:

$$\max_{y,s} \{b^T y : A^T y + s = c,\ s \geq 0\}. \tag{2.2}$$

This last expression is often preferred due to its ease of implementation into IPMs. Together, problems (2.1) and (2.2) are referred to as the primal-dual pair.

The Karush-Kuhn-Tucker (KKT) [3] or optimality conditions for the primal-dual pair, (2.1) and (2.2), can be viewed as a mapping F from $\Re^{2n+m}$ to $\Re^{2n+m}$ such that

$$\begin{aligned} F(x,y,s) = \begin{bmatrix} Ax - b \\ A^T y + s - c \\ XSe \end{bmatrix} &= 0, \\ (x,s) &\geq 0, \end{aligned} \tag{2.3}$$

where $X,\ S \in \Re^{n\times n}$ are diagonal matrices constructed from the vectors x and s, respectively, and $e \in \Re^n$ is the vector of all ones [29]. Any vector x that satisfies the conditions $Ax - b = 0$ and $x \geq 0$ is a feasible point for the primal LP. A dual feasible point is any (y, s) satisfying the conditions $A^T y + s - c = 0$ and $s \geq 0$. Thus the KKT conditions say that any point that is both primal feasible and dual feasible and satisfies $XSe = 0$ is optimal for both problems. According to duality theory (the theory that explains the relationship between the primal and dual LPs, discussed, for example, in [20], [4]), if x and (y, s) are feasible for their respective problems, then

$$b^T y \ \leq \ c^T x.$$

That is, the dual objective function value is a lower bound for the primal objective function value. The difference between the primal and dual objective functions, $\left|c^T x - b^T y\right|$, is known as the duality gap. At optimality, x^* solves (2.1), (y^*, s^*) solves (2.2), and the duality gap is zero (i.e. $b^T y^* = c^T x^*$). Furthermore, the condition $X^* S^* e = 0$ implies that at least one of the pair x_j^* or s_j^* must be zero for all $j = 1, 2, \ldots, n$. Since the nonzero components of x and s occur in complementary positions, the condition $XSe = 0$ in (2.3) is referred to as the complementary slackness condition.

In this dissertation, we focus on a class of primal-dual interior-point methods which find solutions to (2.1) and (2.2) by applying variants of Newton's method to (2.3). Let $z = (x, y, s)$ represent the current approximation to the solution and assume that it satisfies the nonnegativity conditions $x > 0$ and $s > 0$. If z is not optimal, Newton's method forms a linear model tangent to F at z and finds a search direction, $\Delta z = (\Delta x, \Delta y, \Delta s) \in \Re^{2n+m}$, by solving:

$$J(A, x, s)\ \Delta z = -F(x, y, s),$$

where J is the Jacobian of F. This gives the linear system

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = - \begin{bmatrix} r_p \\ r_d \\ XSe \end{bmatrix}, \tag{2.4}$$

where $r_p = Ax-b$ and $r_d = A^Ty+s-c$ are the primal and dual residuals, respectively. By eliminating Δs from the linear system, we obtain the following equivalent system (also known as the "augmented system"):

$$\begin{bmatrix} A & 0 \\ S & -XA^T \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = - \begin{bmatrix} r_p \\ -Xr_d + XSe \end{bmatrix},$$

with $\Delta s = -r_d - A^T\Delta y$. By further eliminating Δx, we have the set of equations that form a major component of primal-dual IPM algorithms:

$$\begin{aligned} AD^2A^T\Delta y &= -r_p - AD^2r_d + Ax, \\ \Delta s &= -r_d - A^T\Delta y, \\ \Delta x &= -x - D^2\Delta s, \end{aligned} \tag{2.5}$$

where $D^2 = S^{-1}X$. The equations $AD^2A^T\Delta y = -r_p - AD^2r_d + Ax$ can be equivalently written as $AD^2A^T\Delta y = b - AD^2r_d$ since $r_p = Ax-b$. This system of equations is known as the "normal equations" because they are the normal equations for a linear least squares problem with coefficient matrix DA^T. Since it is assumed that A

has full row rank and D is a positive diagonal matrix, the coefficient matrix AD^2A^T is symmetric and positive definite. As a result, the coefficient matrix can be factored using the Cholesky factorization as $AD^2A^T = LL^T$ where L is lower triangular [20], and this makes computing solutions of the normal equations easy. Once the search direction Δz is computed, the updated solution z^+ is given by

$$z^+ = z + \alpha \Delta z,$$

where $\alpha \in (0, 1]$ is a parameter designed to prevent z^+ from violating the condition $(x^+, s^+) > 0$. The Newton iteration continues until the duality gap falls below a small defined tolerance. This is the basis for primal-dual IPMs, but there are many variants, two of which we discuss in Chapters 3 and 4.

Another way to interpret finding a solution of (2.4) is that we calculate Δz to satisfy the primal constraints,

$$\begin{aligned} A(x + \Delta x) &= b, \\ A\Delta x &= b - Ax, \end{aligned} \tag{2.6}$$

the dual constraints,

$$\begin{aligned} A^T(y + \Delta y) + (s + \Delta s) &= c, \\ A^T\Delta y + \Delta s &= c - A^Ty - s, \end{aligned} \tag{2.7}$$

and the complementary slackness condition,

$$\begin{aligned} (x_j + \Delta x_j)(s_j + \Delta s_j) &= 0, \ \forall j \\ x_j s_j + x_j \Delta s_j + s_j \Delta x_j + \Delta x_j \Delta s_j &= 0, \ \forall j \\ X\Delta s + S\Delta x &= -XSe - \Delta X \Delta S e. \end{aligned} \tag{2.8}$$

Equations (2.6) and (2.7) are precisely the first two blocks of equations in the linear system (2.4). The equations in (2.8) differ from the last block of equations in (2.4) by a nonlinear term, $\Delta X \Delta S e$. To obtain an approximate solution for Δz, we ignore the nonlinear term. This action is permitted since ΔX and ΔS are small in the limit and their product is even smaller. The result is applying Newton's Method to (2.3) and solving the linear system (2.4).

2.2 The Column Generation Heuristic

2.2.1 Overview

The bulk of the computation in solving for the search direction, Δz, stems from the $m \times m$ matrix, AD^2A^T. For dense matrices, it takes $\Theta(m^2 n)$ operations to form the matrix and another $\Theta(\frac{m^3}{3})$ to factor it. On large-scale LPs where $n \gg m$, the incorporation of a column generation scheme into the primal-dual framework can reduce this computational effort.

Originally devised by Gilmore et al. [8] in the early 1960's, column generation is a technique used to solve an LP by generating the columns of A as needed. This is a beneficial tool because most of the columns of A have no effect on the optimal solution. From duality theory, assuming nondegeneracy, there are exactly m zero components of s corresponding to exactly m positive components of x at the optimal solution (see [4], [5], or [20], for example). If x^* solves the primal LP and (y^*, s^*) solves the dual LP, we define Q^* to be the set of indices associated with those m positive components of x^* (or m zero components of s^*). Since the remaining $n - m$

components of x^* are zero, the “reduced” primal LP formed by just the columns associated with the index set Q^* has the same solution as the standard primal LP. Furthermore, since the columns of A are simply the rows of A^T, we can incorporate a column generation scheme into our primal-dual framework by working with just a subset of the dual constraints at each iteration. We try to choose this subset Q so that $Q^* \subset Q$. Our strategy is as follows: given any point $(x, s) > 0$, and any integer k with $m \leq k \leq n$, we define the set $Q \subset N = \{1, 2, \dots n\}$ to be the indices, j, of the dual constraints, $a_j^T y \leq c_j$, associated with the k largest x_j/s_j ratios. We define $A_Q \in \Re^{m \times k}$ to be the submatrix of A comprised of the columns a_j with $j \in Q$. The vectors $(s_Q, x_Q) \in \Re^k$ are comprised of the components s_j and x_j, respectively, with $j \in Q$. If k is relatively small compared to n, it takes only $\Theta(m^2 k)$ operations to form the matrix $A_Q S_Q^{-1} X_Q A_Q^T$. If we can use this matrix in place of $AS^{-1}XA^T$, our algorithm will significantly reduce the amount of work per iteration and possibly take only a fraction of the time to solve a particular LP problem.

We let

$$\Delta\tilde{z} = (\Delta\tilde{x}, \Delta y, \Delta\tilde{s}) \in \Re^{2n+m}$$

be the approximate Newton direction at z^i computed by using $A_Q S_Q^{-1} X_Q A_Q^T$ in place of $AS^{-1}XA^T$ in (2.5). The components of the direction associated with the neglected constraints are set to zero. The directions

$$\Delta z_Q = (\Delta x_Q, \Delta y, \Delta s_Q) \text{ and } \Delta\tilde{z}_Q = (\Delta\tilde{x}_Q, \Delta\tilde{y}, \Delta\tilde{s}_Q)$$

have dimension $2k + m$. These vectors are obtained by deleting the components of Δx_j, Δs_j, $\Delta\tilde{x}_j$, and $\Delta\tilde{s}_j$ with $j \notin Q$.

Our goal is to prove that applying a column generation heuristic to primal-dual IPM algorithms for solving LPs will provide excellent theoretical properties and computational results.

In the following three subsections, we discuss the main ingredients in the reduced algorithms: the initial point, the choice of Q and the update strategy.

2.2.2 Initial Point

There are a number of techniques for generating initial points (see [11], [14], or [17], for example). Our column generation approach requires $(x, s) > 0$ at each iteration. To achieve this, we use the strategy of Mehrotra [17] and describe his technique here.

Assume the columns of A are linearly independent and $b, c \neq 0$ and set

$$\hat{x} = A^T(AA^T)^{-1}b,\ \hat{y} = (AA^T)^{-1}Ac,\ \text{and}\ \hat{s} = c - A^T y.$$

The point $(\hat{x}, \hat{y}, \hat{s})$ satisfies the primal and dual equality constraints. However, in order to satisfy the condition $(x, s) > 0$, Mehrotra defines $\delta_x = \max\left(-1.5 \min\left\{\hat{x}_j\right\}, 0\right)$ and $\delta_s = \max\left(-1.5 \min\left\{\hat{s}_j\right\}, 0\right)$ and then defines

$$\begin{aligned} \hat{\delta}_x &= \delta_x + .5\left[\frac{(\hat{x} + \delta_x e)^T(\hat{s} + \delta_s e)}{\sum_{j=1}^n \hat{s}_j + \delta_s}\right], \text{ and} \\ \hat{\delta}_s &= \delta_s + .5\left[\frac{(\hat{x} + \delta_x e)^T(\hat{s} + \delta_s e)}{\sum_{j=1}^n \hat{x}_j + \delta_x}\right], \end{aligned}$$

where e is the vector of ones. Then, if $x^0 = \hat{x} + \hat{\delta}_x e$, $y^0 = \hat{y}$, and $s^0 = \hat{s} + \hat{\delta}_s e$, we have an initial point

$$z^0 = (x^0, y^0, s^0) \tag{2.9}$$

such that $(x^0, s^0) > 0$. The equality constraints are not necessarily satisfied, so z^0 might be infeasible.

2.2.3 Selection of Q

The choice of Q is based on the $k \in [m, n]$ largest x_j/s_j ratios at each iteration. Assuming nondegeneracy, there will be exactly m components of s converging to zero as the optimal solution is approached. Since the corresponding m components of x are not converging to zero, there will be exactly m x_j/s_j ratios tending to infinity as we tend to the optimal solution. Meanwhile, the remaining $n - m$ components of x converge to zero while the corresponding $n - m$ components of s do not. As a result, the remaining $n - m$ x_j/s_j ratios tend to zero as the optimal solution is approached. Since Q is the index set associated with the $k \in [m, n]$ largest x_j/s_j ratios at each iteration, when we are close enough to the solution, we are guaranteed to find an optimal solution to the "reduced" dual LP based solely on the rows of A^T associated with Q. This implies that we can solve the normal equations with coefficient matrix $A_Q D_Q^2 A_Q^T$. We describe our selection of Q below.

Let C satisfy $0 < C \leq 1$. We specify a lower bound on the number of indices in Q, given by $l_{bnd} = 3m$. We also specify an upper bound u_{bnd} on the number of indices in Q. The upper bound is a predetermined integer between $3m$ and n. Let $N = \{1, 2, \ldots, n\}$. The algorithm for determining Q is described below:

Select the most promising dual constraints:

1. *Sort the values* x_j/s_j *so that*

$$x_{j_1}/s_{j_1} \geq x_{j_2}/s_{j_2} \geq \ldots \geq x_{j_n}/s_{j_n}.$$

2. *Let* Q *consist of the indices associated with the ratios greater than* $C \cdot (x_{j_1}/s_{j_1})$:

```
$Q = \{j_\eta : x_{j_\eta}/s_{j_\eta} > C \cdot (x_{j_1}/s_{j_1}),\ \eta = 1, 2, \ldots, n\};$
$k = |Q|;$
```

3. *Modify* Q *if necessary so that* $l_{bnd} \leq k \leq u_{bnd}$:

```
**if** $k \geq u_{bnd}$
    $Q = \{j_\eta :\ \eta = 1, 2, \ldots, u_{bnd}\};$
**elseif** $k \leq l_{bnd}$
    $Q = \{j_\eta :\ \eta = 1, 2, \ldots l_{bnd}\};$
**end(if)**
$k = |Q|;$
```

In steps 1 and 2 of our algorithm, the ratios x_j/s_j are ordered from largest to smallest and selected if they exceed the product of C and the value of the largest ratio at the current iterate. The indices associated with the selected ratios make up the set Q. The quantity C is an experimental constant chosen based on algorithm performance (see Sections 5.2 and 5.3 for details). Step 3 guarantees that Q contains at least $3m$ indices at each iteration. This is a critical component of the convergence analysis in the following chapters. Computationally, by ensuring Q is of an "appropriate" size relative to m and n, we can reduce the risk of instabilities in the performance of the algorithm. Since $n \gg m$, a value of k close to m is likely to cause the algorithm to cycle through an extremely large number of constraint choices before reaching the optimal solution. A value of k close to n in early iterations is acceptable since steps 2 and 3 of our subroutine will significantly reduce k as the optimal solution is approached. This poses the question, "For which range of ratios k/n, where $k \in (m, n)$, are the "reduced" algorithms most efficient for solving LPs?" This and other topics will be discussed in Chapter 5.

2.2.4 Update Strategy

Throughout the next two chapters, we use the superscript + to represent an update to a quantity within an algorithm. Here we examine the update strategy used within our "reduced" algorithms for the primal and dual variables as well as the dual residuals.

In a general primal-dual IPM algorithm, the primal and dual variables are updated by the equations:

$$\begin{aligned} x^+ &= x + \alpha^p \Delta x, \\ y^+ &= y + \alpha^d \Delta y, \\ s^+ &= s + \alpha^d \Delta s \end{aligned} \tag{2.10}$$

where α^p and α^d are the step lengths between 0 and 1 along the primal and dual search directions, respectively. The dual residuals are updated as follows:

$$\begin{aligned} r_d^+ &= A^T y^+ + s^+ - c \\ &= A^T\left(y + \alpha^d \Delta y\right) + \left(s + \alpha^d \Delta s\right) - c \\ &= \left(A^T y + s - c\right) + \alpha^d\left(A^T \Delta y + \Delta s\right) \\ &= r_d + \alpha^d\left(A^T \Delta y - r_d - A^T \Delta y\right) \\ &= (1-\alpha^d) r_d. \end{aligned}$$

We used the relations $r_d = A^T y + s - c$ and $\Delta s = -r_d - A^T \Delta y$ in the fourth line. In the "reduced" algorithms, we set $\Delta s_j = 0,\ \forall j \notin Q$. By setting $r_d^+ = A^T y^+ + s^+ - c$, we have

$$(r_d)_j^+ = \begin{cases} (1-\alpha^d)(r_d)_j & \text{if } j \in Q \\ (r_d)_j + \alpha^d a_j^T \Delta y & \text{if } j \notin Q \end{cases},$$

where a_j is the j^{th} column of A. Since $(x^0, s^0) > 0$, it can be shown that the components of the initial dual residual r_d^0 are strictly positive and have the same value. Therefore,

$$0 < (1-\alpha^d)(r_d)_j^0 < (r_d)_j^0,\ \forall j \in Q,$$

since $\alpha^d \in (0,1)$. If an index j remains in Q for every iteration, then we have

$$0 < (1-\alpha^d)(r_d)_j^i < (r_d)_j^i, \text{ for } j \in Q,\ \forall i. \tag{2.11}$$

In this case, the components of the dual residual in Q at every iteration remain strictly positive and decrease to zero as the solution is approached. If at any iteration, the j^{th} component of residual is not associated with Q, it will be updated as $(r_d)_j^+ = (r_d)_j + \alpha^d a_j^T \Delta y$. Here the j^{th} component of the dual residual does not monotonically decrease to zero. This could potentially cause the "reduced" algorithms to fail to satisfy dual feasibility. We resolve this issue by incorporating a new update strategy into our algorithms. In the case where $|Q| = n$, we follow the update strategy for a general primal-dual IPM algorithm. Otherwise, the update strategy is described below:

We define $x^+ = x + \alpha^p \Delta x$ and $(y^+ = y + \alpha^d \Delta y, s^+ = s + \alpha^d \Delta s)$ to be the updates on the primal and dual variables, respectively. Since one of our goals is to satisfy dual feasibility $(A^T y^+ - c \leq 0)$, we define $\upsilon = A^T y^+ - c$ and examine its maximum component, denoted υ_ℓ, to determine if we are making progress to this goal. In addition, we define a set $\hat{Q}$ to represent the index set of dual constraints that are added to Q when it is determined we are not making progress towards satisfying dual feasibility. The set $\hat{Q}$ is initially empty with $\hat{Q} \cap Q = \emptyset$. After each iteration, we have the following possible outcomes:

Case 1: Suppose $\ell \in Q$.

The quantity $\upsilon_\ell + s_\ell^+$ is exactly the value of the ℓ^{th} component of the updated dual residual, $(r_d)_\ell^+$. Since this component of the residual is given by $(r_d)_\ell^+ = (1-\alpha^d)(r_d)_\ell$, it remains strictly positive and is smaller in value than $(r_d)_\ell$. Therefore, we can make progress toward satisfying dual feasibility by setting each component of the dual residual to this value [i.e. $r_d^+ = (\upsilon_\ell + s_\ell^+)e$ where $e \in \Re^n$ is the vector of all ones]. To ensure every component of the slack vector remains strictly positive (as a result of this change), we define a new update on the slack vector as $s^{++} = r_d^+ - \upsilon$. Although the primal vector x does not affect the progress toward achieving dual feasibility, it was observed through numerical testing that s^{++} converged to its optimal value much faster than x^{+}. Since the stopping criterion to our algorithms is based on the duality gap falling below a small tolerance, our algorithms failed to converge in reasonable time. Thus, we define a new update on the primal variables

$$x_j^{++} = \begin{cases} x_j^{+} & j \in Q \\ \min\left(s^{++}, x^{+}\right) & j \notin Q \end{cases}$$

to resolve this issue. Thus, the point (x^{++}, y^{+}, s^{++}) is the new approximate solution to the LP and $\hat{Q}$ is set to $\emptyset$ (nullset).

Case 2: Suppose $\ell \notin Q$.

Since the "reduced' algorithms only consider the columns of A associated with Q, the term $a_\ell^T \Delta y$ is never calculated in this case. Since it is unclear what the value of $a_\ell^T \Delta y$ is from iteration to iteration, there is no guarantee that we will progress toward satisfying dual feasibility. As a result, we return to the previous solution [i.e. set $(x^{++}, y^{++}, s^{++}) = (x, y, s)$] and recalculate the solution with $\hat{Q} = \hat{Q} \cup \{\ell\}$.

The algorithm for the update strategy used in "reduced" algorithms when $|Q| < n$ is summarized below.

```
*Update* $(x^{+}, s^{+})$ *and* $r_d$:

    $\upsilon = A^T y^{+} - c$;

    Determine $\upsilon_\ell = \max_j\{\upsilon_j\}$;

    **if** $\ell \in Q$

        $r_d^{+} = \left(\upsilon_\ell + s_\ell^{+}\right) e$;
        $s^{++} = r_d^{+} - \upsilon;$
        $x_j^{++} = \begin{cases} x_j^{+} & j \in Q \\ \min(s^{++}, x^{+}) & j \notin Q \end{cases};$
        $\hat{Q} = \emptyset;$
    else
        Set $x^{++} = x;\ y^{++} = y;\ s^{++} = s;$
        $r_d^{+} = r_d;$
        $\hat{Q} = \hat{Q} \cup \{\ell\};$
    end(if)
```

The *Update* (x^+, s^+) *and* r_d subroutine is designed to update the components of the dual residual and solution even when dual feasibility is satisfied. An alternative approach to updating these terms would be to consider invoking the update scheme only after determining that we have not satisfied dual feasibility (i.e. $\max_j\{a_j^T y^+ - c_j\} > 0$). Once a dual-feasible solution (y^+, s^+) has been determined, the solution can be updated using the general update strategy in (2.10) until the optimal solution has been found.

It should be noted that once the index ℓ is added to Q (in the case $\ell \notin Q$), the algorithm recomputes the search direction from scratch using normal matrix $A_{Q^+} D_{Q^+}^2 A_{Q^+}^T$ where $Q^+ = Q \cup \{\ell\}$. A faster way to compute this search direction would be to compute a rank-1 update. This can be accomplished since the normal matrix based on Q^+ is a known invertible square matrix plus a perturbation matrix:

$$A_{Q^+}D_{Q^+}^2A_{Q^+}^T = A_QD_Q^2A_Q^T + \frac{x_\ell}{s_\ell}a_\ell a_\ell^T$$

The Sherman-Morrison formula can be applied here to provide a numerically inexpensive way to compute the inverse of $A_{Q^+}D_{Q^+}^2A_{Q^+}^T$ based on the rank-1 matrix $\frac{x_\ell}{s_\ell}a_\ell a_\ell^T$.

2.3 Summary

In this chapter, we stated the optimality conditions for a primal-dual LP, explained how to solve for these conditions and presented our column generation heuristic for our "reduced" algorithms. In the next two chapters, we will focus on two "reduced" algorithms, $redPDAS$ and $redMPC$, and state the properties that guarantee their convergence.

Chapter 3

A Reduced Primal-Dual Affine-Scaling (*redPDAS*) Algorithm

3.1 Affine-Scaling Methods

3.1.1 Overview

Primal-dual affine-scaling methods transform a linear program to an equivalent problem via an affine scaling transformation. The transformation repositions the current point to one well inside the boundary of the feasible region determined by the constraints $x_j s_j > 0$ to prevent the progression to the optimal solution from becoming arbitrarily slow. Once the current point is transformed to one close to the "center" of the feasible region, significant progress can be made towards the optimal solution by moving along the search direction (see [19], [20], and [26]).

The primal, dual, and primal-dual affine-scaling methods are explained in this section. In all three cases, the problem is scaled so that the current point (or approximation to the solution) becomes the point e (the vector of all ones) since it is equidistant from the bounds of the region defined by the nonnegativity constraints. A projected steepest descent [1] (ascent) direction is computed to simultaneously decrease (increase) the objective function value and satisfy feasibility. The updated approximation is formed by taking a step from e in the search direction. This point

[1]See Appendix A.

is then transformed back into its original coordinates.

In this chapter, we examine primal-dual affine-scaling algorithms. First we discuss primal and dual variants. Then a general primal-dual affine scaling (PDAS) algorithm is given in section 3.2 and a "reduced" version called *redPDAS* is presented in section 3.3. Preliminary results for the convergence of the *redPDAS* algorithm are provided in section 3.4. Global and local convergence results follow with specific modifications to the algorithm. Finally, the chapter is summarized in section 3.5.

3.1.2 The Primal and Dual Affine-Scaling Methods

Consider the linear scaling transformation $\Phi_D : \Re^n \to \Re^n$, where $\Phi_D(x) = D^{-1}x$ with positive diagonal scaling matrix D. Here the primal variables x are transformed to $\xi = D^{-1}x$. Thus x can be expressed as $D\xi$. The transformation leads to a new primal problem:

$$\begin{array}{lcl} \text{Min}_x \; c^T x & & \text{Min}_\xi \; (Dc)^T \xi \\ \text{s.t.} & & \text{s.t.} \\ \qquad Ax = b & \mapsto & \qquad AD\xi = b \\ \qquad x \geq 0 & & \qquad \xi \geq 0 \end{array}$$

Assuming primal feasibility, $(Ax = b, x \geq 0)$, we compute a search direction $\Delta\xi$ in the transformed space, a projected steepest descent direction that maintains feasibility of the primal equality constraints. The steepest descent direction is given by the negative gradient of the objective function, $-Dc$. Suppose $\xi^+ = \xi + \Delta\xi$, where the superscript $+$ denotes the next iteration. To maintain feasibility, we must satisfy

$$AD(\xi + \Delta\xi) = b,$$

or equivalently

$$AD\Delta\xi = 0.$$

That is, $\Delta\xi$ must lie in the null space of AD. We define

$$P_{AD} = I - DA^T(AD^2A^T)^{-1}AD \tag{3.1}$$

to be an orthogonal projection matrix [2] for AD. Then we want $\Delta\xi$ to be the projection of the steepest descent direction onto the null space of AD. The expression for $\Delta\xi$ is given by

$$\Delta\xi = -P_{AD}Dc. \tag{3.2}$$

Transforming ξ^+ back into the original coordinate system, we have

$$\begin{aligned} x^+ &= D\xi^+ \\ &= D(\xi + \Delta\xi) \\ &= x + D\Delta\xi. \end{aligned}$$

The difference between the new iterate and the current iterate defines the search direction in the original coordinate system. We will denote this direction Δx. We

[2] See Appendix A.

can write $\Delta x = \Phi_D^{-1}(\Delta\xi)$ as

$$\begin{aligned}
\Delta x &= D\Delta\xi \\
&= -DP_{AD}Dc \\
&= [-D^2 + D^2A^T(AD^2A^T)^{-1}AD^2]\; c. \qquad (3.3)
\end{aligned}$$

The second and third statements follow from (3.2) and (3.1), respectively. To make $\xi = e$ we set $D \equiv X$. This gives the direction Δx generated by the primal affine-scaling algorithm.

There is a similiar affine-scaling algorithm for the dual problem. Suppose we have a dual feasible point $(y, s) : A^T y + s = c, s \geq 0$. Consider the linear scaling transformation that maps the dual variables s to $\rho = Ds$ where D is positive and diagonal. The variables y are not transformed since they are unrestricted in value. Then s can be written as $D^{-1}\rho$. Assuming A has full row rank, we can write the transformed problem exclusively in terms of the new variables ρ by solving the dual equality constraints, $A^T y + s = c$ for y and substituting this expression into the problem. The expression for y is obtained by first premultiplying the dual equality constraints by A:

$$\begin{aligned}
A^T y + s &= c, \\
AA^T y + As &= Ac, \\
y &= \left(AA^T\right)^{-1}(Ac - As).
\end{aligned}$$

The last step follows since A has full row rank. Substituting the expression for y

into the objective function of the dual problem gives

$$\begin{aligned}
b^T y &= b^T \left[\left(AA^T\right)^{-1}(Ac - As)\right] \\
&= b^T \left[\left(AA^T\right)^{-1} A\left(c - D^{-1}\rho\right)\right] \\
&= b^T \left(AA^T\right)^{-1} Ac - b^T \left(AA^T\right)^{-1} AD^{-1}\rho.
\end{aligned}$$

The second line uses the relation $s = D^{-1}\rho$. Since we are interested in maximizing the objective function, $b^T y$, of the dual problem over y, this is the same as minimizing $b^T \left(AA^T\right)^{-1} AD^{-1}\rho$ with respect to ρ. The term $b^T \left(AA^T\right)^{-1} Ac$ is constant, so it can be ignored. The dual equality constraints can be expressed as

$$\begin{aligned}
c &= A^T y + s \\
&= A^T \left(AA^T\right)^{-1}(Ac - As) + s \\
&= A^T \left(AA^T\right)^{-1}\left(Ac - AD^{-1}\rho\right) + D^{-1}\rho.
\end{aligned}$$

Collecting terms, we have

$$\left(-A^T\left(AA^T\right)^{-1}A + I\right)D^{-1}\rho \;=\; \left(I - A^T\left(AA^T\right)^{-1}A\right)c,$$

or

$$P_A D^{-1}\rho \;=\; P_A c,$$

where

$$P_A = I - A^T\left(AA^T\right)^{-1}A \tag{3.4}$$

is the orthogonal projection matrix for A. The transformed dual problem becomes:

$$\begin{array}{lcl} \text{Max}_y \; b^T y & & \text{Min}_\rho \; (D^{-1}A^T(AA^T)^{-1}b)^T\rho \\ \text{s.t.} & & \text{s.t.} \\ \qquad A^T y + s = c & \mapsto & \qquad (P_A D^{-1})\rho = P_A c \\ \qquad\qquad s \geq 0 & & \qquad\qquad \rho \geq 0 \end{array}$$

The steepest descent direction in the transformed space is given by

$$-D^{-1}A^T(AA^T)^{-1}b. \tag{3.5}$$

However, to maintain feasibility we must satisfy

$$P_A D^{-1}(\rho + \Delta\rho) \;=\; P_A c,$$

or equivalently

$$P_A D^{-1}\Delta\rho = 0.$$

By premultiplying the steepest descent direction in (3.5) by the orthogonal projection matrix $I - P_{AD} = DA^T(AD^2A^T)^{-1}AD$, where P_{AD} is defined in (3.1) , we have

$$\begin{aligned} \Delta\rho &= -(I - P_{AD})D^{-1}A^T(AA^T)^{-1}b \\ &= -DA^T(AD^2A^T)^{-1}ADD^{-1}A^T(AA^T)^{-1}b \\ &= -DA^T(AD^2A^T)^{-1}b. \end{aligned}$$

It follows that

$$\begin{aligned} P_A D^{-1}\Delta\rho &= (I - A^T(AA^T)^{-1}A)D^{-1}\left(-DA^T(AD^2A^T)^{-1}b\right) \\ &= -A^T(AD^2A^T)^{-1}b + A^T(AA^T)^{-1}AA^T(AD^2A^T)^{-1}b \\ &= -A^T(AD^2A^T)^{-1}b + A^T(AD^2A^T)^{-1}b \\ &= 0. \end{aligned}$$

If the next iterate is $\rho^+ = \rho + \Delta\rho$, we have

$$\begin{aligned} s^+ &= D^{-1}\rho^+ \\ &= s + D^{-1}\Delta\rho \end{aligned}$$

and

$$\Delta s = -A^T(AD^2A^T)^{-1}b. \tag{3.6}$$

We determine Δy to maintain dual feasibility. That is, we must satisfy

$$\begin{aligned} A^Ty^+ + s^+ = c, \\ A^T(y+\Delta y) + (s+\Delta s) = c, \\ A^T\Delta y + \Delta s = 0. \end{aligned} \tag{3.7}$$

This last equation follows from the fact that $A^Ty + s = c$. Premultiplying (3.7) by A and solving for Δy, we find

$$\begin{aligned} \Delta y &= -(AA^T)^{-1}A\Delta s \\ &= (AD^2A^T)^{-1}b. \end{aligned} \tag{3.8}$$

When $D \equiv S^{-1}$, $(\Delta y, \Delta s)$ is the direction generated by the dual affine-scaling algorithm.

3.1.3 The Primal-Dual Affine-Scaling Method

In a primal-dual setting, the scaling matrix is $D \equiv (S^{-1}X)^{1/2}$. The point XSe is repositioned with respect to both nonnegativity constraints, $x > 0$ and $s > 0$.

Assuming primal and dual feasibility, Δx from (3.3) and $(\Delta y, \Delta s)$ from (3.6) and (3.8) with $D \equiv (S^{-1}X)^{1/2}$ are given by

$$\begin{aligned}
\Delta x &= [-S^{-1}X + S^{-1}XA^T(AS^{-1}XA^T)^{-1}AS^{-1}X]\ c \\
&= [-S^{-1}X + S^{-1}XA^T(AS^{-1}XA^T)^{-1}AS^{-1}X]\ (A^Ty + s) \\
&= -S^{-1}XA^Ty - S^{-1}Xs + S^{-1}XA^T(AS^{-1}XA^T)^{-1}AS^{-1}XA^Ty \\
&\quad +S^{-1}XA^T(AS^{-1}XA^T)^{-1}AS^{-1}Xs \\
&= -S^{-1}XA^Ty - x + S^{-1}XA^Ty + S^{-1}XA^T(AS^{-1}XA^T)^{-1}Ax \\
&= -x + S^{-1}XA^T(AS^{-1}XA^T)^{-1}Ax \\
&= -x + S^{-1}XA^T(AS^{-1}XA^T)^{-1}b,
\end{aligned}$$

and

$$\begin{aligned}
\Delta y &= (AS^{-1}XA^T)^{-1}b, \\
\Delta s &= -A^T(AS^{-1}XA^T)^{-1}b.
\end{aligned}$$

Simplifying the expressions for the directions, we have

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

The primal-dual affine-scaling direction is exactly the standard Newton step for solving the optimality conditions for the primal or dual problem:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ XSe \end{bmatrix}. \tag{3.9}$$

3.1.4 Algorithms

The algorithms in this chapter are written completely in terms of the original variables. We let Δx^a and $(\Delta y^a, \Delta s^a)$ be the components of the affine-scaling direction associated with the primal and dual problems, respectively. We start with an initial point (x, y, s) satisfying $x > 0$ and $s > 0$. Primal and dual feasiblity are not required in our algorithms. Let $r_p = Ax - b$ and $r_d = A^T y + s - c$ be the primal and dual residuals, respectively. The search direction we seek is the solution to the system:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^a \\ \Delta y^a \\ \Delta s^a \end{bmatrix} = - \begin{bmatrix} r_p \\ r_d \\ XSe \end{bmatrix}. \tag{3.10}$$

The expressions for the components of this direction are given by

$$\begin{aligned} \Delta y^a &= (AS^{-1}XA^T)^{-1}(b - AS^{-1}Xr_d), \\ \Delta s^a &= -r_d - A^T \Delta y^a, \\ \Delta x^a &= -x - S^{-1}X\Delta s^a. \end{aligned}$$

To avoid moving too far along the path of the search direction, step lengths are incorporated in the algorithms. Step lengths $\hat{\alpha}^p$ and $\hat{\alpha}^d$ are chosen as the largest value (computed by a ratio test) to satisfy $x + \hat{\alpha}^p \Delta x^a \geq 0$ and $s + \hat{\alpha}^d \Delta s^a \geq 0$. To ensure that the step length does not exceed 1, we set $\hat{\alpha}^p = \min\{\hat{\alpha}^p, 1\}$ and $\hat{\alpha}^d = \min\left\{\hat{\alpha}^d, 1\right\}$. To ensure $x^+ > 0$ and $s^+ > 0$, the step lengths are then multiplied by a fixed positive number τ where $0 < \tau < 1$. Therefore, x and s are updated by computing $x^+ = x + \alpha^p \Delta x^a > 0$ and $s^+ = s + \alpha^d \Delta s^a > 0$ where $\left(\alpha^p, \alpha^d\right) = \tau\left(\min\{\hat{\alpha}^p, 1\}, \min\left\{\hat{\alpha}^d, 1\right\}\right)$. The dual variables y are unrestricted in value and updated by $y^+ = y + \alpha^d \Delta y^a$.

Since the expressions for the affine-scaling direction depend on the dual residuals, the dual residuals must be updated within the algorithms as well. The update for the dual residuals can be written as

$$\begin{aligned}
r_d^+ &= A^T y^+ + s^+ - c \\
&= A^T(y + \alpha^d \Delta y^a) + (s + \alpha^d \Delta s^a) - c \\
&= (A^T y + s - c) + \alpha^d (A^T \Delta y^a + \Delta s^a) \\
&= (1 - \alpha^d) r_d.
\end{aligned}$$

Observe that the expressions for the affine-scaling direction do not depend on the primal residuals. As a result, the primal residuals are not incorporated into the algorithms.

The algorithms terminate once x and y are feasible for their respective problems and the duality gap $|c^T x - b^T y|$ falls below a small tolerance, λ.

3.2 A Primal-Dual Affine-Scaling (PDAS) Algorithm

In this section, we summarize a general PDAS algorithm (see Monteiro et al. [19])

A General PDAS Algorithm

Input: (x, y, s) with $x > 0$ and $s > 0$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $r_d = A^T y + s - c$.

Main Algorithm:

while $|c^T x - b^T y| / \max |c^T x, 1| > \lambda$

Compute the affine-scaling direction:

$$\begin{aligned} \Delta y^a &= \left(AS^{-1}XA^T\right)^{-1}\left[b - AS^{-1}Xr_d\right], \\ \Delta s^a &= -r_d - A^T \Delta y^a, \\ \Delta x^a &= -x - S^{-1}X\Delta s^a. \end{aligned}$$

Compute the affine step:

$$\hat{\alpha}^p = \begin{cases} 1 & \text{if } \Delta x_j^a \geq 0, \ \forall j \\ \min_{\Delta x_j^a < 0}\left[-x_j/\Delta x_j^a\right] & \text{otherwise} \end{cases},$$

$$\hat{\alpha}^d = \begin{cases} 1 & \text{if } \Delta s_j^a \geq 0, \ \forall j \\ \min_{\Delta s_j^a < 0}\left[-s_j/\Delta s_j^a\right] & \text{otherwise} \end{cases},$$

$$\alpha^p = \tau \ \min\left(\hat{\alpha}^p, 1\right); \quad \alpha^d = \tau \ \min\left(\hat{\alpha}^d, 1\right).$$

Update the solution:

$$\begin{aligned} x^+ &= x + \alpha^p \Delta x^a, \\ y^+ &= y + \alpha^d \Delta y^a, \\ s^+ &= s + \alpha^d \Delta s^a. \end{aligned}$$

Update the residuals:

$$r_d^+ = \left(1 - \alpha^d\right) r_d.$$

end(while)

3.3 The *redPDAS* Algorithm

The general PDAS algorithm in section 3.2 considers the entire LP data set (A, b, c) at every iteration. In this section, we present a reduced PDAS algorithm, *redPDAS*. The vectors $\Delta\tilde{x}^a \in \Re^{n\times 1}$ and $(\Delta\tilde{y}^a, \Delta\tilde{s}^a) \in \Re^{(m+n)\times 1}$ define the affine-scaling direction in the *redPDAS* algorithm. We refer to $\Delta\tilde{z}^a = (\Delta\tilde{x}^a, \Delta\tilde{y}^a, \Delta\tilde{s}^a)$ as the approximate affine-scaling direction. The vectors $\Delta\tilde{x}^a_Q \in \Re^{k\times 1}$ and $\Delta\tilde{s}^a_Q \in \Re^{k\times 1}$ are composed of the components $\Delta\tilde{x}^a_j$ and $\Delta\tilde{s}^a_j$, respectively, with $j \in Q$. Let $\hat{Q}$ represent the index set of dual constraints that are added to Q when it is determined the algorithm is not making progress towards satisfying dual feasibility. To differentiate the scalar quantities α^p and α^d from the general PDAS algorithm, we use the subscript Q to emphasize their computation within this algorithm.

The *redPDAS* Algorithm

Input: (x, y, s) with $x > 0$ and $s > 0$, $u_{bnd} \geq 3m$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $r_d = A^T y + s - c$, $l_{bnd} = 3m$, $Q = \{1, 2, \ldots, n\}$, $\hat{Q} = \emptyset$ (null set).

Main Algorithm:

while $|c^T x - b^T y| / \max |c^T x, 1| > \lambda$

Select the most promising dual constraints [3]

$$\begin{aligned} Q &= Q \cup \hat{Q}, \\ k &= |Q|. \end{aligned}$$

Compute the affine-scaling direction:

$$\begin{aligned} \Delta \tilde{y}^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} \left[b - A_Q S_Q^{-1} X_Q (r_d)_Q\right], && (3.11) \\ \Delta \tilde{s}_Q^a &= -(r_d)_Q - A_Q^T \Delta \tilde{y}^a, && (3.12) \\ \Delta \tilde{x}_Q^a &= -x_Q - S_Q^{-1} X_Q \Delta \tilde{s}_Q^a. \end{aligned}$$

Compute the affine step:

$$\hat{\alpha}_Q^p = \begin{cases} 1 & \text{if } (\Delta \tilde{x}_Q^a)_j \geq 0, \ \forall j \\ \min_{(\Delta \tilde{x}_Q^a)_j < 0} \left[-(x_Q)_j / (\Delta \tilde{x}_Q^a)_j\right] & \text{otherwise} \end{cases} \quad (3.13)$$

$$\hat{\alpha}_Q^d = \begin{cases} 1 & \text{if } (\Delta \tilde{s}_Q^a)_j \geq 0, \ \forall j \\ \min_{(\Delta \tilde{s}_Q^a)_j < 0} \left[-(s_Q)_j / (\Delta \tilde{s}_Q^a)_j\right] & \text{otherwise} \end{cases} \quad (3.14)$$

$$\alpha_Q^p = \tau \ \min\left(\hat{\alpha}_Q^p, 1\right), \quad \alpha_Q^d = \tau \ \min\left(\hat{\alpha}_Q^d, 1\right). \quad (3.15)$$

[3]The set Q is determined by the algorithm in Section 2.2.3: Selection of Q

```
    Create full length vectors, $\Delta\tilde{x}^a$ and $\Delta\tilde{s}^a$:

        $\Delta\tilde{x}^a_j = \begin{cases} (\Delta\tilde{x}^a_Q)_{j_\eta} & \text{if } \eta \in Q \\ 0 & \text{otherwise} \end{cases},$   (3.16)

        $\Delta\tilde{s}^a_j = \begin{cases} (\Delta\tilde{s}^a_Q)_{j_\eta} & \text{if } \eta \in Q \\ 0 & \text{otherwise} \end{cases}.$   (3.17)

    Update the solution and dual residuals:

        $x^+ = x + \alpha^p_Q \Delta\tilde{x}^a,$   (3.18)
        $y^+ = y + \alpha^d_Q \Delta\tilde{y}^a,$
        $s^+ = s + \alpha^d_Q \Delta\tilde{s}^a.$

    if $k < n$

        Update $(x^+, s^+)$ and $r_d$. [4]

    else

        $r^+_d = \left(1 - \alpha^d_Q\right) r_d.$

    end(if)

    Prepare for the next iteration:

        $x = x^{++},\ s = s^{++},\ y = y^+,\ r_d = r^+_d.$

end(while)
```

[4]The updated solution and dual residuals are determined by the algorithm in Section 2.2.4: Update Strategy

3.4 Convergence Analysis of *redPDAS*

In this section, we provide preliminary results for the convergence of the *redPDAS* algorithm. Specifically, we show that our algorithm is well-defined and that the direction we compute is an approximation to the standard Newton direction used in *PDAS*. With these results and certain modifications to the algorithm, we show how global and local quadratic convergence can be proved. A complete analysis of global and local quadratic convergence follow from Tits et al. [24] which is strongly modeled from the analysis in [23] inspired by [21] and [25].

We define $N = \{1, 2, \ldots, n\}$ and $\bar{Q} = \{j \in N : j \notin Q\}$. We have normalized the problem so that $\|a_j\|_2 = 1$ where a_j denotes the j^{th} column of A. The following assumptions will be needed for the analysis:

Assumptions:

(A1) Every $m \times k$ submatrix of A has full row rank and $b \neq 0$.

(A2) Nondegeneracy conditions hold: If (x^*, y^*, s^*) solves the primal-dual pair then there exist exactly m indices in $\beta = \{j \in N : x_j^* > 0, s_j^* = 0\}$ and $n - m$ indices in $\bar{\beta} = \{j \in N : x_j^* = 0, s_j^* > 0\}$.

The first two lemmas show that the components of the dual residual as well as vectors x and s remain strictly positive.

Lemma 3.1 *: Suppose the initial point z^0 is defined as in Section 2.2.2, with $(x^0, s^0) > 0$. Then if z^0 is not the optimal solution, $r_d^0 > 0$ and $0 < r_d^+ < r_d$.*

Proof: The initial dual residual vector is given by

$$r_d^0 = A^T y^0 + s^0 - c,$$

where $y^0 = \hat{y} = \left(AA^T\right)^{-1} Ac$ and $s^0 = \hat{s} + \hat{\delta}_s e = c - A^T y^0 + \hat{\delta}_s e$. Simplifying this expression gives

$$r_d^0 = \hat{\delta}_s e,$$

where e is the vector of all ones. The expression for $\hat{\delta}_s$ is given by

$$\hat{\delta}_s = \delta_s + .5 \left[\frac{(\hat{x} + \delta_x e)^T (\hat{s} + \delta_s e)}{\sum_{j=1}^n \hat{x}_j + \delta_x} \right],$$

where $\delta_x = \max\left(-1.5 \min\left\{\hat{x}_j\right\}, 0\right)$ and $\delta_s = \max\left(-1.5 \min\left\{\hat{s}_j\right\}, 0\right)$ with $\hat{x} = A^T \left(AA^T\right)^{-1} b$. By Assumption (A1), $\hat{x} \neq 0$ and $\hat{s} \neq 0$. To show $r_d^0 > 0$, we must prove $\hat{\delta}_s > 0$. This can be done by showing (1) $\hat{x} + \delta_x e \geq 0$, (2) $\hat{s} + \delta_s e \geq 0$, and (3) $\sum_{j=1}^n \hat{x}_j + \delta_x e > 0$.

(1) If $\min\left\{\hat{x}_j\right\} < 0$, then $\delta_x = \max\left(-1.5 \min\left\{\hat{x}_j\right\}, 0\right) = -1.5 \min\left\{\hat{x}_j\right\} > 0$. Therefore,

$$\begin{aligned} \hat{x}_j + \delta_x &= \hat{x}_j - 1.5 \min\left\{\hat{x}_j\right\} \\ &\geq \min\left\{\hat{x}_j\right\} - 1.5 \min\left\{\hat{x}_j\right\} \\ &= -.5 \min\left\{\hat{x}_j\right\} \\ &> 0. \end{aligned}$$

If $\min\left\{\hat{x}_j\right\} \geq 0$, then $\delta_x = \max\left(-1.5 \min\left\{\hat{x}_j\right\}, 0\right) = 0$. Therefore,

$$\begin{aligned} \hat{x}_j + \delta_x &= \hat{x}_j \\ &\geq \min\left\{\hat{x}_j\right\} \\ &\geq 0. \end{aligned}$$

This proves $\hat{x} + \delta_x e \geq 0$.

(2) A similar proof shows $\hat{s} + \delta_s e \geq 0$.

(3) Since $\hat{x} \neq 0$, there exists at least one component of $\hat{x}$ such that $\hat{x}_j \neq 0$. Therefore, $\sum_{j=1}^{n} \hat{x}_j + \delta_x e > 0$.

The previous three results imply that

$$\frac{(\hat{x} + \delta_x e)^T (\hat{s} + \delta_s e)}{\sum_{j=1}^{n} \hat{x}_j + \delta_x} \geq 0.$$

If $\min\{\hat{s}_j\} < 0$, then $\delta_s > 0$ and

$$(r_d)_j^0 = \hat{\delta}_s = \delta_s + .5 \left[\frac{(\hat{x} + \delta_x e)^T (\hat{s} + \delta_s e)}{\sum_{j=1}^{n} \hat{x}_j + \delta_x} \right] > 0, \ \forall j.$$

If $\min\{\hat{s}_j\} \geq 0$, then $\delta_s = 0$ and

$$(r_d)_j^0 = \hat{\delta}_s = .5 \left[\frac{(\hat{x} + \delta_x e)^T \hat{s}}{\sum_{j=1}^{n} \hat{x}_j + \delta_x} \right] > 0, \ \forall j,$$

since $(\hat{x}, \hat{s}) \neq 0$. This proves

$$r_d^0 = \hat{\delta}_s e > 0.$$

For the second part of this proof, let $\upsilon = A^T y^+ - c$. It should be noted that the dual residuals are only updated in the case where the index ℓ associated with $\upsilon_\ell = \max_j(\upsilon_j)$ is in Q. The updated dual residuals are given by

$$\begin{aligned} r_d^+ &= \left(\upsilon_\ell + s_\ell^+\right) e \\ &= \left[(1 - \alpha_Q^d)(r_d)_\ell\right] e. \end{aligned}$$

We know $(r_d)_j^0 = \hat{\delta}_s > 0$ for all j. Since the step length $\alpha_Q^d \in (0,1)$, we have

$$0 < (1 - \alpha_Q^d)(r_d)_j < (r_d)_j, \ \forall j.$$

$\square$

Lemma 3.2 *: Suppose* $(x, s) > 0$ *and* $r_d > 0$*. Then (i)* $(x^+, s^+) > 0$; *(ii)* $(x^{++}, s^{++}) > 0$.

Proof: Suppose $x, s > 0$. Then for (i) we have,

$$x^+ \;=\; x + \alpha_Q^p \Delta\tilde{x}^a.$$

Since $\Delta\tilde{x}^a_{\bar{Q}} = 0$, $x_j^+ = x_j > 0$ for all $j \in \bar{Q}$. We now examine x_j^+ with $j \in Q$. By definition,

$$\alpha_Q^p = \tau \begin{cases} 1 & \text{if } (\Delta\tilde{x}_Q^a)_j \geq 0,\ \forall j \\ \min\left(\min_j \left\{ \frac{-(x_Q)_j}{(\Delta\tilde{x}_Q^a)_j} : (\Delta\tilde{x}_Q^a)_j < 0 \right\},\ 1 \right) & \text{otherwise} \end{cases},$$

where $0 < \tau < 1$. If $(\Delta\tilde{x}_Q^a)_j \geq 0$ for all j, then

$$(x_Q^+)_j = (x_Q)_j + \tau(\Delta\tilde{x}_Q^a)_j > 0,\ \forall j.$$

Otherwise, let

$$t_k = \frac{-(x_Q)_k}{(\Delta\tilde{x}_Q^a)_k} = \min_j \left\{ \frac{-(x_Q)_j}{(\Delta\tilde{x}_Q^a)_j} : (\Delta\tilde{x}_Q^a)_j < 0 \right\},$$

where k is the index of the minimum ratio. Suppose $(x_Q^+)_j = (x_Q)_j + \alpha_Q^p(\Delta\tilde{x}_Q^a)_j \leq 0$ for some j. If $\alpha_Q^p = \tau t_k$, then

$$(x_Q^+)_j = (x_Q)_j + \tau t_k(\Delta\tilde{x}_Q^a)_j \;\leq\; 0.$$

Solving for t_k, we have

$$\begin{aligned} \tau t_k(\Delta\tilde{x}_Q^a)_j &\leq -(x_Q)_j, \\ t_k &\geq \frac{-(x_Q)_j}{\tau \cdot (\Delta\tilde{x}_Q^a)_j} > \frac{-(x_Q)_j}{(\Delta\tilde{x}_Q^a)_j}. \end{aligned}$$

This is a contradiction since t_k is the minimum ratio so in this case $x_Q^+ > 0$. If $\alpha_Q^p = \tau$, then $\Delta\tilde{x}_Q^a \geq 0$ by (3.13) and $x_Q > 0$ by assumption. Therefore,

$$x_Q^+ = x_Q + \alpha_Q^p \Delta\tilde{x}_Q^a > 0.$$

A similar analysis holds for s^+. This proves (i).

For (ii), let $\upsilon_\ell = \max_j (\upsilon_j)$ where $\upsilon = A^T y^+ - c$. If $\ell \in Q$, the updated slack vector s^{++} is given by

$$s^{++} \;=\; r_d^+ - \upsilon,$$

where $r_d^+ = \upsilon_\ell + s_\ell^+$. Continuing we have

$$\begin{aligned} s^{++} &= r_d^+ - \upsilon \\ &= \left(\upsilon_\ell + s_\ell^+\right) e - \upsilon. \end{aligned}$$

So, for all j,

$$\begin{aligned} s_j &\geq \upsilon_j + s_l^+ - \upsilon_j \\ &> 0. \end{aligned}$$

Since

$$x_j^{++} = \begin{cases} x_j^+ & j \in Q \\ \min\left(s^{++}, x^+\right) & j \notin Q \end{cases},$$

it follows that $x^{++} > 0$. If $\ell \in \bar{Q}$, we return to the previous iterate and recompute the solution. In this case $(x^{++}, s^{++}) = (x, s) > 0$. []

The next two lemmas show the reduced form of the normal equation matrix and augmented Jacobian of F are nonsingular.

Lemma 3.3 : *Suppose (A2) holds and* $X_Q, S_Q > 0$. *Then* $A_Q S_Q^{-1} X_Q A_Q^T$ *is positive definite.*

Proof: Suppose v is any non-zero vector in $\Re^m$. Then

$$v^T A_Q S_Q^{-1} X_Q A_Q^T v = \|D_Q A_Q^T v\|^2,$$

where $D_Q = S_Q^{-1} X_Q$. Our assumption implies D_Q is positive definite. Since A_Q has full row rank and $v \neq 0$, $A_Q^T v \neq 0$. Therefore $\|D_Q A_Q^T v\|^2 > 0$ and $A_Q S_Q^{-1} X_Q A_Q^T$ is positive definite. []

In Chapter 2, we defined the vector

$$F(x, y, s) = -\begin{bmatrix} r_p \\ r_d \\ XSe \end{bmatrix},$$

where $r_p = Ax - b$ and $r_d = A^T y + s - c$. The Jacobian of F, denoted $J(A, x, s)$, is given by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix}.$$

By eliminating Δs in the linear system,

$$J(A, x, s)\ \Delta z = -F(x, y, s),$$

we have

$$\begin{bmatrix} A & 0 \\ S & -XA^T \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = -\begin{bmatrix} r_p \\ X\,(s - r_d) \end{bmatrix}.$$

The matrix

$$\begin{bmatrix} A & 0 \\ S & -XA^T \end{bmatrix}$$

is referred to as the augmented Jacobian of F, denoted $J_a(A, x, s)$. The next lemma shows $J_a(A_Q, x_Q, s_Q)$ is nonsingular.

Lemma 3.4 *: Suppose (A2) holds and $x_Q, s_Q > 0$. Then the augmented Jacobian of $F(x_Q, y, s_Q)$,*

$$\begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix},$$

is nonsingular.

Proof:

Assume the system given by

$$\begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

has a nonzero solution. This gives the equations

$$A_Q \lambda_2 = 0, \tag{3.19}$$

$$X_Q^T A_Q^T \lambda_1 - S_Q \lambda_2 = 0, \tag{3.20}$$

for some vectors λ_1 and λ_2. Solving equation (3.20) for λ_2 and substituting this expression into (3.19) gives

$$A_Q S_Q^{-1} X_Q A_Q^T \lambda_2 = 0.$$

By Lemma (3.3), $A_Q S_Q^{-1} X_Q A_Q^T$ is positive definite and therefore nonsingular. Therefore, $\lambda_2 = 0$. Equation (3.20) reduces to

$$A_Q^T \lambda_1 = 0,$$

since $x_Q > 0$. This implies $\lambda_1 = 0$ since A_Q has full row rank. This contradicts our assumption that $[\lambda_1, \lambda_2]^T$ is nonzero and the result follows. []

We now show the norm of the difference between the standard direction computed in *PDAS* and *redPDAS* is bounded. The dual residual vector associated with the *redPDAS* algorithm is denoted by $\widetilde{r_d}$. All other notation is standard.

Theorem 3.1 *Suppose assumptions (A1) - (A3) hold. Given $\delta > 0$, let*

$$\Upsilon_\delta = \{z : \|z - z^*\| < \delta,\ x > 0,\ s > 0\}.$$

Given $z \in \Upsilon_\delta$, let Δz^a be the Newton direction at z. Then there exists $\kappa > 0$ such that

$$\left\|\Delta \tilde{z}_Q^a - \Delta z_Q^a\right\| \leq \kappa \|z - z^*\| \|\Delta z^a\|,$$

for all $z \in \Upsilon_\delta$.

Proof:

We use a strategy similar to the one in Tits et al. [24]. Let $z \in \Upsilon_a$. The direction $\Delta \tilde{z}^a$ is given by

$$\begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta \tilde{y}^a \\ \Delta \tilde{x}_Q^a \end{bmatrix} = \begin{bmatrix} b - A_Q x_Q \\ X_Q \left[(\widetilde{r_d})_Q - s_Q\right] \end{bmatrix}. \qquad (3.21)$$

The *PDAS* Newton direction is given by

$$\begin{bmatrix} 0 & A \\ -XA^T & S \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x^a \end{bmatrix} = \begin{bmatrix} b - Ax \\ X\left[r_d - s\right] \end{bmatrix}.$$

Partitioning the previous expression in terms of Q and $\bar{Q}$ gives

$$\begin{bmatrix} 0 & A_Q & A_{\bar{Q}} \\ -X_Q A_Q^T & S_Q & 0 \\ -X_{\bar{Q}} A_{\bar{Q}}^T & 0 & S_{\bar{Q}} \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \\ \Delta x_{\bar{Q}}^a \end{bmatrix} = \begin{bmatrix} b - A_Q x_Q - A_{\bar{Q}} x_{\bar{Q}} \\ X_Q\left[(r_d)_Q - s_Q\right] \\ X_{\bar{Q}}\left[(r_d)_{\bar{Q}} - s_{\bar{Q}}\right] \end{bmatrix}.$$

Premultiplying the third row by $-A_{\bar{Q}} S_{\bar{Q}}^{-1}$ and adding to the first row, we can eliminate $\Delta x_{\bar{Q}}^a$ to obtain the following system of equations,

$$\begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} =$$

$$\begin{bmatrix} b - A_Q x_Q \\ X_Q\left[\left(\widetilde{r_d}\right)_Q - s_Q\right] \end{bmatrix} - \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} (r_d)_{\bar{Q}} \\ X_Q\left[\left(\widetilde{r_d}\right)_Q - (r_d)_Q\right] \end{bmatrix}. \tag{3.22}$$

The first term in the right-hand side of (3.22) is exactly the right-hand side of (3.21). The last k components of the second vector are zero since $r_d = \widetilde{r_d}$. Combining (3.21) and (3.22), we have

$$\begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta \tilde{y}^a \\ \Delta \tilde{x}_Q^a \end{bmatrix} =$$

$$\begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} + \psi,$$

where

$$\psi = \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} (r_d)_{\bar{Q}} \\ 0 \end{bmatrix}.$$

Since

$$\begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} = \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix},$$

and since $J_a(A_Q, x_Q, s_Q)$ is nonsingular by Lemma 3.4, we can express the difference between the two directions by

$$\begin{bmatrix} \Delta \tilde{y}^a \\ \Delta \tilde{x}_Q^a \end{bmatrix} - \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} =$$

$$J_a(A_Q, x_Q, s_Q)^{-1} \left(\begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} + \psi \right). \tag{3.23}$$

The norm of the left-hand side of (3.23) is bounded by the sum of the norms of the individual terms on the right-hand side. Since $J_a(A_Q, x_Q, s_Q)$ is nonsingular for all Q and there are only a finite number of choices of Q,

$$\left\| J_a(A_Q, x_Q, s_Q)^{-1} \right\| \le \kappa_0,$$

for some $\kappa_0 > 0$. The norm

$$\begin{aligned}
\left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \right\| &= \left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T \right\| \\
&= \max_{\|v\|=1} \left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T v \right\|, \text{ where } v \in \Re^{n-k} \\
&= \max_{\|v\|=1} \left\| \sum_{j \in \bar{Q}} \frac{x_j}{s_j} a_j a_j^T v \right\| \\
&\leq \max_{\|v\|=1} \sum_{j \in \bar{Q}} \left| \frac{x_j}{s_j} \right| \left| a_j^T v \right| \|a_j\|, \\
&\quad \text{by the Triangle Inequality} \\
&\leq \sum_{j \in \bar{Q}} \left| \frac{x_j}{s_j} \right|, \text{ by Assumption (A4)} \\
&= \left\| S_{\bar{Q}}^{-1} X_{\bar{Q}} \right\|_1 \\
&\leq (n-k) \left\| S_{\bar{Q}}^{-1} \right\| \left\| X_{\bar{Q}} \right\|, \\
&\quad \text{since } \|h\|_1 \leq n \|h\|_2, \forall h \in \Re^n \\
&= (n-k) \left\| S_{\bar{Q}}^{-1} \right\| \left\| X_{\bar{Q}} - X_{\bar{Q}}^* \right\|, \text{ since } X_{\bar{Q}}^* = 0 \\
&\leq (n-k) \left\| S_{\bar{Q}}^{-1} \right\| \|z - z^*\|.
\end{aligned}$$

The relation in the third to last step can be found, for example, in [10] or [27]. Let

$$\left\| S_{\bar{Q}}^{-1} \right\| = \max_{j \in \bar{Q}} (1/s_j) = \kappa_1. \tag{3.24}$$

Then,

$$\begin{aligned}
&\left\| J_a(A_Q, x_Q, s_Q)^{-1} \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} \right\| \\
&\qquad \leq \left\| J_a(A_Q, x_Q, s_Q)^{-1} \right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \right\| \left\| \begin{bmatrix} \Delta y^a \\ \Delta x_Q^a \end{bmatrix} \right\| \\
&\qquad \leq \kappa_0 \cdot (n-k) \kappa_1 \| z - z^* \| \cdot \| \Delta z^a \| \\
&\qquad = \kappa_2 \| z - z^* \| \| \Delta z^a \|, \qquad (3.25)
\end{aligned}$$

where $\kappa_2 = \kappa_0 \cdot \kappa_1 \cdot (n-k)$. This gives a bound for the norm of the first term in (3.23). Examining the norm of the second term in (3.23), we find

$$\begin{aligned}
\|\psi\| &= \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} (r_d)_{\bar{Q}} \\ 0 \end{bmatrix} \right\| \\
&= \left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} (r_d)_{\bar{Q}} \right\| \\
&= \left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} \left(-A_{\bar{Q}}^T \Delta y^a - \Delta s_{\bar{Q}}^a \right) \right\| \\
&\leq \left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T \right\| \| \Delta y^a \| + \| A_{\bar{Q}} \| \left\| S_{\bar{Q}}^{-1} \right\| \| X_{\bar{Q}} \| \left\| \Delta s_{\bar{Q}}^a \right\| \\
&\leq (n-k) \kappa_1 \| z - z^* \| \| \Delta z^a \| + (n-k) \kappa_1 \| z - z^* \| \| \Delta z^a \|,
\end{aligned}$$

since $\left\| A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T \right\| \leq (n-k) \kappa_1 \| z - z^* \|$, $\left\| S_{\bar{Q}}^{-1} \right\| = \kappa_1$, and $\| X_{\bar{Q}} \| = \left\| X_{\bar{Q}} - X_{\bar{Q}}^* \right\| \leq \| z - z^* \|$. The third line follows since $r_d = -A^T \Delta y^a - \Delta s^a$ in the $PDAS$ algorithm.

The norm of $A_{\bar{Q}}$ is bounded by

$$\begin{aligned}
\left\|A_{\bar{Q}}\right\| &= \max_{v\neq 0} \frac{\left\|A_{\bar{Q}}v\right\|}{\|v\|}, \text{ where } v \in \Re^{n-k} \\
&= \max_{v\neq 0} \frac{\left\|\sum_{j\in\bar{Q}} a_j v_j\right\|}{\|v\|} \\
&\leq \max_{v\neq 0} \frac{\sum_{j\in\bar{Q}} \|a_j\| \, |v_j|}{\|v\|} \\
&= \max_{v\neq 0} \frac{\|v\|_1}{\|v\|} \\
&\leq \max_{v\neq 0} \frac{(n-k)\,\|v\|}{\|v\|} \\
&= n-k.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\left\|J_a(A_Q, x_Q, s_Q)^{-1}\psi\right\| &\leq \left\|J_a(A_Q, x_Q, s_Q)^{-1}\right\| \|\psi\| \\
&\leq \kappa_0 \cdot 2(n-k)\kappa_1 \|z - z^*\| \|\Delta z^a\| \\
&= 2\kappa_2 \|z - z^*\| \|\Delta z^a\|. \qquad (3.26)
\end{aligned}$$

Combining (3.23), (3.25), and (3.26), we have

$$\left\|\Delta\tilde{z}^a_Q - \Delta z^a_Q\right\| \leq 3\kappa_2 \|z - z^*\| \|\Delta z^a\|.$$

Setting $\kappa = 3\kappa_2$ gives the desired result. []

Global and local quadratic convergence proofs can be shown provided specific modifications are made to the *redPDAS* algorithm. These modifications follow from Tits et. al. [24] where a complete convergence analysis can be found. Before we explain these changes, we summarize the differences between the *redPDAS* algorithm and its modified version in Table 3.1:

	redPDAS **algorithm**	**Modified** *redPDAS* **algorithm**
Feasibility:	Accepts a dual-infeasible initial point; strives to achieve dual-feasibility	Requires a dual-feasible point at each iteration
Terms with Q:	$A_Q D_Q^2 A_Q^T$, $\Delta\tilde{x}_Q^a$, $\Delta\tilde{s}_Q^a$, x_Q, s_Q, $(r_d)_Q$	$A_Q D_Q^2 A_Q^T$, $\Delta\tilde{x}_Q^a$
Dual affine step:	$\alpha_Q^d = \tau \ \min\left(\hat{\alpha}_Q^d, 1\right)$ where $0 < \tau < 1$	$\alpha_Q^d = \min\left(\max\left\{\tau\hat{\alpha}_Q^d, \hat{\alpha}_Q^d - \|\Delta y^a\|\right\}, 1\right)$ where $0 < \tau < 1$
Update scheme:	Uses a general updating strategy to compute x^+, y^+, and s^+; provides a unique updating scheme to compute r_d^+ and further update x^+ and s^+	Uses a general updating strategy to compute y^+ and s^+; provides a unique updating scheme to compute x^+

Table 3.1: A summary of the differences between the *redPDAS* algorithm and a modified version of the algorithm adapted from Tits et. al. [24] that proves to be locally and quadratically convergent.

The following preliminary information comes from Tits et. al. [24]:

Let

$$F = \{y : A^T y \leq c\} \tag{3.27}$$

be the feasible solution set for the dual LP problem and

$$I(y) = \{j \in N : a_j^T y = c_j\} \tag{3.28}$$

denote the index set of active constraints at y. It is assumed

(A4) The dual optimal solution set $F^* = \{y^* \in F : b^T y^* \geq b^T y \quad \forall y \in F\}$ is nonempty and bounded,

(A5) For all $y \in F$, $\{a_j : j \in I(y)\}$ is a linearly independent set of vectors, and

(A6) The dual optimal solution set from assumption (A5) is such that $F^* = \{y^*\}$ (i.e. y^* is unique).

Assumptions $(A4)$ and $(A5)$ are required to show global convergence. Assumption $(A6)$, which supersedes $(A4)$, is used to show local quadratic convergence. We now address the modifications required for the convergence analysis.

Two significant changes to the $redPDAS$ algorithm allow a complete convergence analysis. The first modification is maintaining dual-feasibility at each iteration. This implies $s^i = c - A^T y^i > 0$ and $r_d^i = 0$ for all i. Consequently, the algorithm for updating (x^+, s^+) and r_d has no effect on this dual-feasible algorithm and can be discarded. Furthermore, equations (3.12) and (3.14) must be computed with the full data to satisfy the above requirement on the slack variables. As a result, equation (3.17) must be eliminated from the algorithm. The second critical change is the update scheme for the primal variables. The $\Delta\tilde{x}^a_Q$ component of the normal equations remains the same; however the full length vector $\Delta\tilde{x}^a$ in (3.16) is determined before the "Compute the affine step" section, and the primal affine step (3.13) and the updated primal solution (3.18) can be replaced with

$$\begin{aligned} \tilde{x} &= x + \Delta\tilde{x}^a, \\ [\tilde{x}_-]_j &= \min\{\tilde{x}_j, 0\}, \end{aligned}$$

and

$$x_j^+ = \min\left\{\max\left(\min\left\{\|\Delta y^a\|^2 + \|\tilde{x}_-\|^2, \underline{x}\right\}, (\tilde{x}_-)_j\right), x_{\max}\right\} \ \forall j \in N, \quad (3.29)$$

respectively, with $\underline{x} > 0$ and $x_j \leq x_{\max}\ \forall j \in N$. The key to the global convergence analysis is the availability of a dual-feasible point at every iteration along with the condition, $\|\Delta y^a\|^2 + \|\tilde{x}_-\|^2$, on the primal updates. A local quadratic convergence analysis follows provided the above conditions hold and the bound $\hat{\alpha}^d_Q - \|\Delta y^a\|$ is imposed on the dual affine step in equation (3.15). The modified *redPDAS* algorithm is presented below:

The Dual-Feasible *redPDAS* Algorithm

Input: (x, y, s) with $x > 0, x_{\max} > 0$ such that $x_j \leq x_{\max}\ \forall j \in N$ and $s = c - A^T y$, $\underline{x} > 0$, $u_{bnd} \geq 3m$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $l_{bnd} = 3m$, $Q = \{1, 2, \ldots, n\}$, $\hat{Q} = \emptyset$.

Main Algorithm:

while $|c^T x - b^T y| / \max |c^T x, 1| > \lambda$

Select most promising dual constraints [5]

$$Q = Q \cup \hat{Q}.$$

Compute the affine-scaling direction:

$$\begin{aligned}
\Delta \tilde{y}^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T \right)^{-1} b, \\
\Delta \tilde{s}^a &= -A^T \Delta \tilde{y}^a, \\
\Delta \tilde{x}_Q^a &= -x_Q - S_Q^{-1} X_Q \Delta \tilde{s}_Q^a.
\end{aligned}$$

Create the full vector $\Delta \tilde{x}^a$:

$$\Delta \tilde{x}_j^a = \begin{cases} (\Delta \tilde{x}_Q^a)_{j_\eta} & \text{if } \eta \in Q \\ 0 & \text{otherwise} \end{cases}.$$

Set $\tilde{x} = x + \Delta \tilde{x}^a$ and for $j \in N$, define

$$[\tilde{x}_-]_j = \min \{\tilde{x}_j, 0\}.$$

Compute the affine step:

$$\hat{\alpha}_Q^d = \begin{cases} 1 & \text{if } \Delta \tilde{s}_j^a \geq 0, \ \forall j \\ \min_{\Delta \tilde{s}_j^a < 0} \left[-s_j / \Delta \tilde{s}_j^a \right] & \text{otherwise} \end{cases},$$

$$\alpha_Q^d = \min \left(\max \left\{ \tau \hat{\alpha}_Q^d, \hat{\alpha}_Q^d - \|\Delta y^a\| \right\}, 1 \right).$$

Update the solution:

$$x_j^+ = \min \left\{ \max \left(\min \left\{ \|\Delta y^a\|^2 + \|\tilde{x}_-\|^2, \underline{x} \right\}, (\tilde{x}_-)_j \right), x_{\max} \right\} \ \forall j \in N,$$

[5]The set Q is determined by the algorithm in Section 2.2.3: Selection of Q

$$y^+ = y + \alpha_Q^d \Delta \tilde{y}^a,$$

$$s^+ = s + \alpha_Q^d \Delta \tilde{s}^a.$$

end(while)

Under Assumptions $(A1), (A4)$, and $(A5)$ and the aforementioned modifications to the *redPDAS* algorithm, we can produce a proof of global convergence that follows from Tits et al. [24]. We give a brief outline here.

Lemmas (3.2) and (3.3) can be extended to the modified *redPDAS* algorithm to show the algorithm is well-defined. As a result, the Δy^a-component of the normal equations can justifiably be expressed as

$$\Delta \tilde{y}^a = \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} b.$$

Since it is assumed $b \neq 0$, the sequence of dual objective function values $\left\{b^T y\right\}$ is nondecreasing. Let K be an infinite index set. If we assume $\{y\}$ converges to y^* and $\{\Delta y^a\}$ converges to 0 on K, then it can be shown that y^* is stationary (i.e. $Ax^* = b, X^* s^* = 0$) and $\{\tilde{x}\}$ converges to x^* where x^* is the unique Lagrange multiplier associated with y^*. Proving $\{y\}$ converges to F^* (global convergence) requires two main steps. The first step is to prove $\{y\}$ converges to the set of stationary points y^* of the dual LP problem. This is achieved by a contradiction argument: if $\{y\}$ converges to a nonstationary point on K, then $\{\Delta y^a\}$ must converge to 0 on K. The proof is largely dependent on the bound, $\|\Delta y^a\|^2 + \|\tilde{x}_-\|^2$, in the primal updates. The second step is to prove that the mulitplier [6] vectors associated with all limit

[6]The set of $x \in \Re^n$ such that $Ax = b$ and $X(c - A^T y) = 0$.

points of $\{y\}$ are the same. Using the proofs from steps 1 and 2 along with the fact that $\left\{b^T y\right\}$ is nondecreasing, it can be shown $\{y\}$ converges to F^*.

With the additional assumption ($A6$), (local) q-quadratic convergence of the pair (x, y) [7] can be shown. Assumption ($A6$) implies that strict complementary holds, i.e.

$$x_j^* > 0,\ s_j^* = c_j - a_j^T y^* \quad = \quad 0 \qquad \forall j \in I(y^*).$$

Furthermore,

$$\text{span}\left(\{a_j : j \in I(y^*)\}\right) \quad = \quad \Re^m.$$

We can extend the above results to Lemma 3.4 to show $J(A_Q, x_Q^*, s_Q^*)$ is nonsingular. This, in addition to Lemmas 1, 14 and Proposition 15, all in Tits et al. [24], along with Lemma (3.1) with $r_d = 0$ and the bound on the dual affine step, $\hat{\alpha}_Q^d - \|\Delta y^a\|$, provides the necessary tools to complete the q-quadratic convergence analysis.

3.5 Summary

In this chapter, we introduced the *redPDAS* algorithm and explained how specific modifications to the algorithm provide global and local q-quadratic convergence results. With further research, we hope to prove similar results for the *redPDAS* algorithm. Numerical results from *redPDAS* are given in Chapter 5. Before presenting the results we present the *redMPC* algorithm.

[7] The sequence $\left\{(x^i, y^i)\right\}$ converges q-quadratically to (x^*, y^*) if it converges to (x^*, y^*) and there exists a constant κ such that $\left\|(x^{i+1}, y^{i+1}) - (x^*, y^*)\right\| \leq \kappa \left\|(x^i, y^i)\right\|^2$.

Chapter 4

A Reduced Mehrotra Predictor-Corrector (*redMPC*) Algorithm

4.1 MPC Method

4.1.1 Overview

Mehrotra's Predictor-Corrector algorithm generates a sequence of approximate solutions, $z(\mu) = (x(\mu), y(\mu), s(\mu))$, to the perturbed KKT conditions:

$$\begin{aligned} Ax(\mu) - b &= 0, & (4.1) \\ A^T y(\mu) + s(\mu) - c &= 0, & (4.2) \\ X(\mu)S(\mu)e &= \sigma\mu e, & (4.3) \\ (x(\mu), s(\mu)) &> 0, & (4.4) \end{aligned}$$

where $0 \leq \sigma \leq 1$ and $\mu = x^T s/n > 0$. The conditions here differ from the KKT condtions in (2.3) in that the solution $z(\mu)$ is uniquely defined for each $\mu > 0$ and the pairwise products $x_j(\mu)s_j(\mu) = \sigma\mu$ for each j. We define

$$C = \{z(\mu) \mid \mu > 0\}$$

as the central path. The central path defines a trajectory of feasible solutions that steer clear of the boundary of the primal-dual feasible region. As μ decreases to zero, C converges to a primal-dual solution of the linear program (or of the KKT conditions in (2.3)).

Mehrotra's method differs from other interior-point methods in that two linear systems are solved at each iteration, one for the affine-scaling or predictor direction, Δz^a, and one for the centering-corrector direction. The predictor direction is obtained by solving (3.10). The centering-corrector direction, $\Delta z^{cc} = (\Delta x^{cc}, \Delta y^{cc}, \Delta s^{cc})$, is calculated based on the amount of progress the predictor direction has made in reducing μ and the error (or nonlinear term) in (2.8). This direction is found by solving a slightly different system:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^{cc} \\ \Delta y^{cc} \\ \Delta s^{cc} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sigma\mu e - \Delta X^a \Delta S^a e \end{bmatrix}, \tag{4.5}$$

where we assume for convenience that (x, y, s) is primal and dual feasible. By premultiplying the second block of equations by $-X$ and adding this to the third block of equations, we can eliminate Δs^{cc} from the linear system. The result is the following equivalent system:

$$\begin{bmatrix} A & 0 \\ S & -XA^T \end{bmatrix} \begin{bmatrix} \Delta x^{cc} \\ \Delta y^{cc} \end{bmatrix} = \begin{bmatrix} 0 \\ \sigma\mu e - \Delta X^a \Delta S^a e \end{bmatrix},$$

with $\Delta s^{cc} = -A^T \Delta y^{cc}$. By further eliminating Δx^{cc} by premultiplying the second block of equations by $-AS^{-1}$ and adding this to the first block of equations, we have

$$\begin{aligned} \Delta y^{cc} &= -(AS^{-1}XA^T)^{-1}AS^{-1}\left(\sigma\mu e - \Delta X^a \Delta S^a e\right), \\ \Delta s^{cc} &= -A^T \Delta y^{cc}, \\ \Delta x^{cc} &= S^{-1}\left(\sigma\mu e - \Delta X^a \Delta S^a e\right) - S^{-1}X\Delta s^{cc}. \end{aligned}$$

The advantage of computing the centering-corrector direction is that we are able to improve our linear, first-order model of F in (2.3) to a quadratic, second-order model. In addition, Mehrotra's method uses the same coefficient matrix to solve for the two directions, and thus the extra cost of forming this second direction is that of performing a single back-substitution. This additional cost is small. The sum of the predictor and centering-corrector directions is the search direction used to derive an improved solution to the perturbed KKT conditions.

For the remainder of the chapter, we focus on variants of Mehrotra's Predictor-Corrector (MPC) algorithm. A general MPC algorithm is given in section 4.2 and a "reduced" version called $redMPC$ is presented in section 4.3. Preliminary results for the convergence of the $redMPC$ algorithm are provided in section 4.4. Global and local convergence results follow with certain modifications to the algorithm. Finally, the chapter is summarized in section 4.5.

4.1.2 Centering-Corrector Direction

The components of the centering-corrector direction are based on a centering parameter, σ, and a corrector direction derived from the nonlinear term $\Delta X^a \Delta S^a e$. We explain the significance of these terms here.

The affine-scaling direction aims to satisfy the KKT conditions by moving toward the boundary of the feasible region defined by the nonnegative pairwise products, $x_j s_j > 0$. If this direction makes significant progress in reducing the duality measure, μ, while satisfying the conditions $(x, s) > 0$, we will be in good proximity of the central path and closer to satisfying the KKT conditions (since $\mu \to 0$). Therefore, our solution will require little centering (or movement towards the central path). If the affine-scaling direction makes little progress in reducing μ, then we will need a significant amount of centering so that the algorithm can make better progress in reducing μ on the next iteration. To measure the efficiency of the affine-scaling direction, we compare the hypothetical value of μ based on this direction to the previous value of μ. We define

$$\mu^a = \frac{\left(x + \alpha_a^p \Delta x^a\right)^T \left(s + \alpha_a^d \Delta s^a\right)}{n}$$

to be the hypothetical value of μ based on the affine-scaling direction and the centering parameter

$$\sigma = \left[\frac{\mu^a}{\mu}\right]^3 = \left[\frac{\left(x + \alpha_a^p \Delta x^a\right)^T \left(s + \alpha_a^d \Delta s^a\right)}{n\mu}\right]^3$$

to be the cube of the ratio of μ^a and μ. [1] If $\mu^a \ll \mu$, the affine-scaling direction makes good progress in reducing μ and little centering is needed. Therefore, σ is insignificant and is set close to 0. If μ^a is only a little smaller than μ, σ is set close to 1 so that the algorithm can be in a better position to reduce μ on the next iteration.

The centering-corrector direction also uses the affine-scaling direction to compensate for the error in the linear system, (2.4). We can interpret finding a solution $z^+ = z + \Delta z$ to the KKT conditions as calculating Δz to satisfy the primal constraints, the dual constraints, and the complementary slackness condition [see Chapter 2, equations (2.6) - (2.8)] given by

$$(x_j + \Delta x_j)(s_j + \Delta s_j) = 0, \ \forall j,$$

[1]Although it is not a requirement to cube the ratio, Mehrotra [17] found this heuristic for determining σ most efficient through exhaustive computational testing.

or equivalently

$$\begin{aligned} x_j s_j + x_j \Delta s_j + s_j \Delta x_j + \Delta x_j \Delta s_j &= 0, \ \forall j, \\ X \Delta s + S \Delta x &= -XSe - \Delta X \Delta S e. \end{aligned} \tag{4.6}$$

In computing the affine-scaling direction, the last term in (4.6) is ignored. That is,

$$X \Delta s^a + S \Delta x^a = -XSe.$$

Since applying a full step along the affine-scaling direction to (4.6) gives

$$X \Delta s^a + S \Delta x^a + XSe + \Delta X^a \Delta S^a e = \Delta X^a \Delta S^a e,$$

we can define a corrector direction, $\Delta z^{cor} = (\Delta x^{cor}, \Delta y^{cor}, \Delta s^{cor})$, to compensate for this error term:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^{cor} \\ \Delta y^{cor} \\ \Delta s^{cor} \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ \Delta X^a \Delta S^a e \end{bmatrix}.$$

The centering parameter and corrector direction must be computed after the affine-scaling direction due to the fact that they are both dependent on it. In addition, the centering and corrector components can be combined into one step since they are obtained by solving linear systems with the same coefficient matrix. The result is the linear system in (4.5) for calculating the centering-correction direction.

4.2 A MPC Algorithm

In this section, we present a general MPC algorithm similar to the algorithm in [29].

A General MPC Algorithm

Input: (x, y, s) with $x > 0$ and $s > 0$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $\mu = x^T s/n$, $r_d = A^T y + s - c$.

Main Algorithm:

while $|c^T x - b^T y| / \max |c^T x, 1| > \lambda$

Compute the affine-scaling direction:

$$\begin{aligned}
\Delta y^a &= \left(AS^{-1}XA^T\right)^{-1}\left[b - AS^{-1}Xr_d\right], \\
\Delta s^a &= -r_d - A^T \Delta y^a, \\
\Delta x^a &= -x - S^{-1}X\Delta s^a.
\end{aligned}$$

Compute the affine step:

$$\begin{aligned}
\hat{\alpha}_a^p &= \begin{cases} 1 & \text{if } \Delta x_j^a \geq 0,\ \forall j \\ \min_{\Delta x_j^a < 0}\left[-x_j/\Delta x_j^a\right] & \text{otherwise} \end{cases}, \\
\hat{\alpha}_a^d &= \begin{cases} 1 & \text{if } \Delta s_j^a \geq 0,\ \forall j \\ \min_{\Delta s_j^a < 0}\left[-s_j/\Delta s_j^a\right] & \text{otherwise} \end{cases},
\end{aligned}$$

$$\alpha_a^p = \min\left(\hat{\alpha}_a^p, 1\right), \quad \alpha_a^d = \min\left(\hat{\alpha}_a^d, 1\right).$$

Compute the centering parameter:

$$\sigma = \left[\frac{\left(x + \alpha_a^p \Delta x^a\right)^T\left(s + \alpha_a^d \Delta s^a\right)}{n\mu}\right]^3.$$

Compute the centering-corrector direction:

$$\begin{aligned}
\xi &= \sigma\mu S^{-1}e - S^{-1}\Delta X^a \Delta S^a e, \\
\Delta y^{cc} &= -(AS^{-1}XA^T)^{-1}A\xi, \\
\Delta s^{cc} &= -A^T\Delta y^{cc}, \\
\Delta x^{cc} &= \xi - S^{-1}X\Delta s^{cc}.
\end{aligned}$$

Compute the predictor-corrector direction:

$$\begin{aligned}
\Delta x &= \Delta x^a + \Delta x^{cc}, \\
\Delta y &= \Delta y^a + \Delta y^{cc}, \\
\Delta s &= \Delta s^a + \Delta s^{cc}.
\end{aligned}$$

Compute the predictor-corrector step:

$$\begin{aligned}
\hat{\alpha}^p &= \begin{cases} 1 & \text{if } \Delta x_j > 0,\ \forall j \\ \min_{\Delta x_j < 0}\left[x_j/\Delta x_j\right] & \text{otherwise} \end{cases}, \\
\hat{\alpha}^d &= \begin{cases} 1 & \text{if } \Delta s_j > 0,\ \forall j \\ \min_{\Delta s_j > 0}\left[s_j/\Delta s_j\right] & \text{otherwise} \end{cases},
\end{aligned}$$

$$\alpha^p = \tau\ \min\left(\hat{\alpha}^p, 1\right), \quad \alpha^d = \tau\ \min\left(\hat{\alpha}^d, 1\right).$$

Update the variables:

$$\begin{aligned}
x^+ &= x + \alpha^p \Delta x, \\
y^+ &= y + \alpha^d \Delta y, \\
s^+ &= s + \alpha^d \Delta s.
\end{aligned}$$

Update quantities:

$$\begin{aligned} r_d^+ &= \left(1-\alpha^d\right) r_d, \\ \mu^+ &= (x^+)^T(s^+)/n. \end{aligned}$$

end(while) ———————————————-

4.3 The *redMPC* Algorithm

In this section, we present a reduced Mehrotra Predictor-Corrector algorithm, *redMPC*. The vectors $\Delta\tilde{x}^a \in \Re^{n\times 1}$ and $(\Delta\tilde{y}^a, \Delta\tilde{s}^a) \in \Re^{(m+n)\times 1}$ define the affine-scaling direction. Similarily, the vectors $\Delta\tilde{x}^{cc} \in \Re^{n\times 1}$ and $(\Delta\tilde{y}^{cc}, \Delta\tilde{s}^{cc}) \in \Re^{(m+n)\times 1}$ define the centering-corrector direction. We refer to $\Delta\tilde{z} = (\Delta\tilde{x}, \Delta\tilde{y}, \Delta\tilde{s}) = (\Delta\tilde{x}^a + \Delta\tilde{x}^{cc}, \Delta\tilde{y}^a + \Delta\tilde{y}^{cc}, \Delta\tilde{s}^a + \Delta\tilde{s}^{cc})$ as the approximate MPC direction. In accordance with the notation, the vectors $\Delta\tilde{x}_Q \in \Re^{k\times 1}$ and $\Delta\tilde{s}_Q \in \Re^{k\times 1}$ are composed of the components $\Delta\tilde{x}_j$ and $\Delta\tilde{s}_j$, respectively with $j \in Q$. Let $k = |Q|$ and $\hat{Q}$ represent the index set of dual constraints that are added to Q when it is determined the algorithm is not making progress towards satisfying dual feasibility. To differentiate the scalar quantities α_a^p, α_a^d, α^p, α^d, and μ from the general MPC algorithm, we use the subscript Q to emphasize their computation within this algorithm.

———————————————-

The *redMPC* Algorithm

Input: (x, y, s) with $x > 0$ and $s > 0$, $u_{bnd} \geq 3m$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $\mu_Q = x^T s/n$, $r_d = A^T y + s - c$, $l_{bnd} = 3m$, $\hat{Q} = \emptyset$.

Main Algorithm:

while $|c^T x - b^T y| / \max |c^T x, 1| > \lambda$

Select the most promising dual constraints [2]

$$\begin{aligned} Q &= Q \cup \hat{Q}, \\ k &= |Q|. \end{aligned}$$

Compute the affine-scaling direction:

$$\begin{aligned} \Delta\tilde{y}^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} \left[b - A_Q S_Q^{-1} X_Q (r_d)_Q\right], \qquad (4.7) \\ \Delta\tilde{s}_Q^a &= -(r_d)_Q - A_Q^T \Delta\tilde{y}^a, \qquad (4.8) \\ \Delta\tilde{x}_Q^a &= -x_Q - S_Q^{-1} X_Q \Delta\tilde{s}_Q^a. \end{aligned}$$

Compute the affine step:

$$\begin{aligned} \hat{\alpha}_{Q,a}^p &= \begin{cases} 1 & \text{if } \left(\Delta\tilde{x}_Q^a\right)_j \geq 0,\ \forall j \\ \min_{\left(\Delta\tilde{x}_Q^a\right)_j < 0} \left[-(x_Q)_j / \left(\Delta\tilde{x}_Q^a\right)_j\right] & \text{otherwise} \end{cases}, \\ \hat{\alpha}_{Q,a}^d &= \begin{cases} 1 & \text{if } \left(\Delta\tilde{s}_Q^a\right)_j \geq 0,\ \forall j \\ \min_{\left(\Delta\tilde{s}_Q^a\right)_j < 0} \left[-(s_Q)_j / \left(\Delta\tilde{s}_Q^a\right)_j\right] & \text{otherwise} \end{cases} \qquad (4.9) \end{aligned}$$

$$\alpha_{Q,a}^p = \min\left(\hat{\alpha}_{Q,a}^p, 1\right), \qquad \alpha_{Q,a}^d = \min\left(\hat{\alpha}_{Q,a}^d, 1\right). \qquad (4.10)$$

Compute the centering parameter:

$$\sigma_Q = \left[\frac{\left(x_Q + \alpha_{Q,a}^p \Delta\tilde{x}_Q^a\right)^T \left(s_Q + \alpha_{Q,a}^d \Delta\tilde{s}_Q^a\right)}{n\mu_Q}\right]^3. \qquad (4.11)$$

[2] The set Q is determined by the algorithm in Section 2.2.3: Selection of Q

Compute the centering-corrector direction:

$$
\begin{aligned}
\xi_Q &= \sigma_Q \mu_Q S_Q^{-1} e_Q - S_Q^{-1} \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a e_Q, \\
\Delta \tilde{y}^{cc} &= -(A_Q S_Q^{-1} X_Q A_Q^T)^{-1} A_Q \xi_Q, \\
\Delta \tilde{s}_Q^{cc} &= -A_Q^T \Delta \tilde{y}^{cc}, \qquad (4.12) \\
\Delta \tilde{x}_Q^{cc} &= \xi_Q - S_Q^{-1} X_Q \Delta \tilde{s}_Q^{cc}.
\end{aligned}
$$

Compute the predictor-corrector direction:

$$\Delta \tilde{x}_Q = \Delta \tilde{x}_Q^a + \Delta \tilde{x}_Q^{cc}, \qquad (4.13)$$

$$\Delta \tilde{y} = \Delta \tilde{y}^a + \Delta \tilde{y}^{cc}, \qquad (4.14)$$

$$\Delta \tilde{s}_Q = \Delta \tilde{s}_Q^a + \Delta \tilde{s}_Q^{cc}. \qquad (4.15)$$

Compute the predictor-corrector step:

$$
\begin{aligned}
\hat{\alpha}_Q^p &= \begin{cases} 1 & \text{if } (\Delta \tilde{x}_Q)_j > 0, \ \forall j \\ \min_{(\Delta \tilde{x}_Q)_j < 0} \left[-(x_Q)_j / (\Delta \tilde{x}_Q)_j \right] & \text{otherwise} \end{cases}, \\
\hat{\alpha}_Q^d &= \begin{cases} 1 & \text{if } (\Delta \tilde{s}_Q)_j > 0, \ \forall j \\ \min_{(\Delta \tilde{s}_Q)_j < 0} \left[-(s_Q)_j / (\Delta \tilde{s}_Q)_j \right] & \text{otherwise} \end{cases}, \qquad (4.16)
\end{aligned}
$$

$$\alpha_Q^p = \tau \ \min \left(\hat{\alpha}_Q^p, 1 \right), \qquad \alpha_Q^d = \tau \ \min \left(\hat{\alpha}_Q^d, 1 \right). \qquad (4.17)$$

Create full length vectors, $\Delta \tilde{x}$ and $\Delta \tilde{s}$:

$$\Delta \tilde{x}_j = \begin{cases} (\Delta \tilde{x}_Q)_{j_\eta} & \text{if } \eta \in Q \\ 0 & \text{otherwise} \end{cases}, \qquad (4.18)$$

$$\Delta \tilde{s}_j = \begin{cases} (\Delta \tilde{s}_Q)_{j_\eta} & \text{if } \eta \in Q \\ 0 & \text{otherwise} \end{cases}. \qquad (4.19)$$

```
        *Update the solution, the duality measure, and the dual residuals:*

            $x^+ = x + \alpha_Q^p \Delta\tilde{x},$   (4.20)

            $y^+ = y + \alpha_Q^d \Delta\tilde{y},$   (4.21)

            $s^+ = s + \alpha_Q^d \Delta\tilde{s}.$   (4.22)

        **if** $k < n$

            *Update* $(x^+, s^+)$ *and* $r_d$. [3]

        **else**

            $r_d^+ = \left(1 - \alpha_Q^d\right) r_d.$

        **end(if)**

            $\mu_Q^+ = (x^{++})^T (s^{++})/n.$

        *Prepare for the next iteration*

            $x = x^{++},\ s = s^{++},\ y = y^+,\ r_d = r_d^+,\ \mu_Q = \mu_Q^+.$

    **end(while)**
```

4.4 Convergence Analysis of *redMPC*

In this section, we provide some preliminary results for the convergence of the *redMPC* algorithm. We will show that our algorithm is well-defined and that the direction we compute is an approximation to the standard Newton direction used

[3]The updated solution and dual residuals are determined by the algorithm in Section 2.2.4: Update Strategy

in MPC. Using these results and certain modifications to the algorithm, we show how global and local quadratic convergence can be proved. A complete analysis of global and local quadratic convergence follows from Winternitz et al. [28].

In the MPC algorithm, two directions (affine-scaling and centering-corrector) are calculated and added together to produce the MPC direction. The following result, from Theorem 3.1 in Chapter 3, is used within the proof to show superlinear convergence of the affine-scaling algorithm in Tits et. al. [24]:

$$\left\|\Delta\tilde{z}^a_Q - \Delta z^a_Q\right\| \leq \kappa \left\|z - z^*\right\| \left\|\Delta z^a\right\|. \tag{4.23}$$

In our analysis, we will use (4.23) to derive a bound on the centering-corrector component of the MPC direction. The bound we seek is

$$\left\|\Delta\tilde{z}^{cc}_Q - \Delta z^{cc}_Q\right\| \leq \hat{\kappa} \cdot \Phi(z), \tag{4.24}$$

where $\Delta\tilde{z}^{cc}$ is the centering-corrector component of the approximate direction, Δz^{cc} is the centering-corrector component of the Newton direction, $\hat{\kappa} > 0$ and $\Phi(z)$ is a term that tends to zero as we approach the optimal solution. Combining (4.23) and (4.24), we have the norm of the difference between the standard direction computed in MPC and that computed in $redMPC$,

$$\begin{aligned}
\left\|\Delta\tilde{z}_Q - \Delta z_Q\right\| &= \left\|\left(\Delta\tilde{z}^a_Q + \Delta\tilde{z}^{cc}_Q\right) - \left(\Delta z^a_Q + \Delta z^{cc}_Q\right)\right\| \\
&\leq \left\|\Delta\tilde{z}^a_Q - \Delta z^a_Q\right\| + \left\|\Delta\tilde{z}^{cc}_Q - \Delta z^{cc}_Q\right\| \\
&\leq \kappa \left\|z - z^*\right\| \left\|\Delta z^a\right\| + \hat{\kappa} \cdot \Phi(z).
\end{aligned}$$

Let (x^i, y^i, s^i) be the solution at iteration i of MPC and $redMPC$. Assumptions (A1)-(A6) from Chapter 3 will be needed throughout the analysis along with the following assumptions:

(A7) $\|x_Q\| \leq C_1$ and $\|s_Q\| \leq C_2$ where $0 < C_1, C_2 < \infty$.

(A8) $|\sigma_Q \mu| \leq \kappa\mu \|\Delta z^a\|$ where $\kappa > 0$.

The *redMPC* algorithm is well-defined since the iterates for the solution and the dual residual remain strictly positive and the matrices $A_Q S_Q^{-1} X_Q A_Q^T$ and $J_a(A_Q, x_Q, s_Q)$ are positive definite. The proofs follow from Lemmas (3.1) - (3.4) in Chapter 3. The Lemmas presented next are required to show (4.24). The first two lemmas give bounds for terms involving the centering parameter, σ.

Lemma 4.1 *: Suppose assumption (A7) and (4.23) and suppose $\mu = \mu_Q$. Then,*

$$|(\sigma - \sigma_Q)\,\mu| \leq 25\mu + V_3 \|z - z^*\| \|\Delta z^a\| + \zeta \cdot \Theta\left(\|\Delta z^a\|^2\right),$$

where

$$\begin{aligned} \zeta &= \max\{\|z - z^*\|, \mu\} \quad \textit{and} \\ V_3 &= \frac{1}{n}\left[25\left(V_1 + c_o V_2 \|z - z^*\|\right) + 10\left(V_1 + 2V_2\right)\right] \end{aligned}$$

with

$$\begin{aligned} V_1 &= (C_1 + C_2)(c_o + \kappa) + 3 + \kappa_1 \|\Delta z^a\| \quad \textit{and} \\ V_2 &= (C_1 + C_2)\,\kappa \end{aligned}$$

for some $c_o, \kappa, \kappa_1 > 0$.

Proof:

The quantity

$$\begin{aligned}(\sigma - \sigma_Q)\,\mu &= \left[(\mu^a/\mu)^3 - \left(\mu_Q^a/\mu_Q\right)^3\right]\,\mu \\ &= \left(1/\mu^2\right)\left[(\mu^a)^3 - \left(\mu_Q^a\right)^3\right] \\ &= \left(1/\mu^2\right)\left(\mu^a - \mu_Q^a\right)\left[(\mu^a)^2 + \mu^a \cdot \mu_Q^a + \left(\mu_Q^a\right)^2\right], \qquad (4.25)\end{aligned}$$

since $\mu = \mu_Q$. We begin by bounding the term $\mu^a - \mu_Q^a$. We can express this term by:

$$\begin{aligned}\mu^a - \mu_Q^a &= \left[(x + \alpha_a^p \Delta x^a)^T \left(s + \alpha_a^d \Delta s^a\right) - \left(x + \alpha_{Q,a}^p \Delta \tilde{x}^a\right)^T \left(s + \alpha_{Q,a}^d \Delta \tilde{s}^a\right)\right]/n \\ &= \Big[\alpha_a^d x^T \Delta s^a + \alpha_a^p s^T \Delta x^a + \alpha_a^p \alpha_a^d \left(\Delta x^a\right)^T \Delta s^a \\ &\quad - \alpha_{Q,a}^d x^T \Delta \tilde{s}^a - \alpha_{Q,a}^p s^T \Delta \tilde{x}^a - \alpha_{Q,a}^p \alpha_{Q,a}^d \left(\Delta \tilde{x}^a\right)^T \Delta \tilde{s}^a\Big]/n.\end{aligned}$$

By partitioning the right-hand side of the last expression into Q and $\bar{Q}$, we can eliminate the terms involving $\Delta \tilde{x}_{\bar{Q}}^a$ and $\Delta \tilde{s}_{\bar{Q}}^a$ since $\Delta \tilde{x}_{\bar{Q}}^a = \Delta \tilde{s}_{\bar{Q}}^a = 0$. This gives

$$\begin{aligned}\mu^a - \mu_Q^a &= \Big[\alpha_a^d x_Q^T \Delta s_Q^a + \alpha_a^p s_Q^T \Delta x_Q^a + \alpha_a^p \alpha_a^d \left(\Delta x_Q^a\right)^T \Delta s_Q^a \\ &\quad + \alpha_a^d x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a + \alpha_a^p s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a + \alpha_a^p \alpha_a^d \left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a \\ &\quad - \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a - \alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a - \alpha_{Q,a}^p \alpha_{Q,a}^d \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\Big]/n.\end{aligned}$$

By adding in, subtracting out, and grouping certain terms, we have

$$\begin{aligned}
\mu^a - \mu^a_Q &= \big[\alpha^d_a x_Q^T \Delta s^a_Q - \boldsymbol{\alpha^d_a x_Q^T \Delta \tilde{s}^a_Q} + \boldsymbol{\alpha^d_a x_Q^T \Delta \tilde{s}^a_Q} - \alpha^d_{Q,a} x_Q^T \Delta \tilde{s}^a_Q \\
&\quad + \alpha^p_a s_Q^T \Delta x^a_Q - \boldsymbol{\alpha^p_a s_Q^T \Delta \tilde{x}^a_Q} + \boldsymbol{\alpha^p_a s_Q^T \Delta \tilde{x}^a_Q} - \alpha^p_{Q,a} s_Q^T \Delta \tilde{x}^a_Q \\
&\quad + \alpha^p_a \alpha^d_a \left(\Delta x^a_Q\right)^T \Delta s^a_Q - \boldsymbol{\alpha^p_{Q,a} \alpha^d_{Q,a} \left(\Delta x^a_Q\right)^T \Delta s^a_Q} \\
&\quad + \boldsymbol{\alpha^p_{Q,a} \alpha^d_{Q,a} \left(\Delta x^a_Q\right)^T \Delta s^a_Q} - \alpha^p_{Q,a} \alpha^d_{Q,a} \left(\Delta \tilde{x}^a_Q\right)^T \Delta \tilde{s}^a_Q \\
&\quad + \alpha^d_a x_{\bar{Q}}^T \Delta s^a_{\bar{Q}} + \alpha^p_a s_{\bar{Q}}^T \Delta x^a_{\bar{Q}} + \alpha^p_a \alpha^d_a \left(\Delta x^a_{\bar{Q}}\right)^T \Delta s^a_{\bar{Q}}\Big] / n, \\
&= \big[\alpha^d_a x_Q^T \left(\Delta s^a_Q - \Delta \tilde{s}^a_Q\right) + \left(\alpha^d_a - \alpha^d_{Q,a}\right) x_Q^T \Delta \tilde{s}^a_Q \\
&\quad + \alpha^p_a s_Q^T \left(\Delta x^a_Q - \Delta \tilde{x}^a_Q\right) + \left(\alpha^p_a - \alpha^p_{Q,a}\right) s_Q^T \Delta \tilde{x}^a_Q \\
&\quad + \left(\alpha^p_a \alpha^d_a - \alpha^p_{Q,a} \alpha^d_{Q,a}\right) \left(\Delta x^a_Q\right)^T \Delta s^a_Q \\
&\quad + \alpha^p_{Q,a} \alpha^d_{Q,a} \Big[\left(\Delta x^a_Q\right)^T \Delta s^a_Q - \left(\Delta \tilde{x}^a_Q\right)^T \Delta \tilde{s}^a_Q\Big] \\
&\quad + \alpha^d_a x_{\bar{Q}}^T \Delta s^a_{\bar{Q}} + \alpha^p_a s_{\bar{Q}}^T \Delta x^a_{\bar{Q}} + \alpha^p_a \alpha^d_a \left(\Delta x^a_{\bar{Q}}\right)^T \Delta s^a_{\bar{Q}}\Big] / n.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\left|\mu^a - \mu^a_Q\right| &\le \frac{1}{n}\big[\left|\alpha^d_a\right| \left\|x_Q\right\| \left\|\Delta s^a_Q - \Delta \tilde{s}^a_Q\right\| + \left|\alpha^d_a - \alpha^d_{Q,a}\right| \left\|x_Q\right\| \left\|\Delta \tilde{s}^a_Q\right\| \\
&\quad + \left|\alpha^p_a\right| \left\|s_Q\right\| \left\|\Delta x^a_Q - \Delta \tilde{x}^a_Q\right\| + \left|\alpha^p_a - \alpha^p_{Q,a}\right| \left\|s_Q\right\| \left\|\Delta \tilde{x}^a_Q\right\| \\
&\quad + \left|\alpha^p_a \alpha^d_a - \alpha^p_{Q,a} \alpha^d_{Q,a}\right| \left\|\Delta x^a_Q\right\| \left\|\Delta s^a_Q\right\| \\
&\quad + \left|\alpha^p_{Q,a} \alpha^d_{Q,a} \Big[\left(\Delta x^a_Q\right)^T \Delta s^a_Q - \left(\Delta \tilde{x}^a_Q\right)^T \Delta \tilde{s}^a_Q\Big]\right| \\
&\quad + \left|\alpha^d_a\right| \left\|x_{\bar{Q}}\right\| \left\|\Delta s^a_{\bar{Q}}\right\| + \left|\alpha^p_a\right| \left|s_{\bar{Q}}^T \Delta x^a_{\bar{Q}}\right| + \left|\alpha^p_a \alpha^d_a\right| \left|\left(\Delta x^a_{\bar{Q}}\right)^T \Delta s^a_{\bar{Q}}\right|\Big].
\end{aligned}$$

We can bound $\left|\mu^a - \mu^a_Q\right|$ by applying the result in (4.23) and proving $\left|\alpha^p_a - \alpha^p_{Q,a}\right|$ and $\left|\alpha^d_a - \alpha^d_{Q,a}\right|$ are small close to the optimal solution. We will use the following results (see [13]): Assume $v, w \in \Re^n$. Then,

$$\begin{aligned}\left|\max_k \left(\frac{v}{w}\right)_k\right| &\leq \frac{\|v\|_1}{\|w\|_\infty}, && (4.26)\\ \|v\|_1 &\leq \sqrt{n}\,\|v\|, && (4.27)\\ \frac{1}{\|w\|_\infty} &\leq \sqrt{n}\frac{1}{\|w\|}. && (4.28)\end{aligned}$$

Let

$$\tilde{\kappa} = \max\left\{\frac{\left\|\Delta z_Q^a\right\|}{\left\|\Delta x_Q^a\right\|}, \frac{\left\|\Delta z_Q^a\right\|}{\left\|\Delta s_Q^a\right\|}\right\}. \qquad (4.29)$$

We examine the components of the current solution for which $\left\{\left(\Delta\tilde{x}_Q^a\right)_j, \left(\Delta\tilde{s}_Q^a\right)_r\right\} < 0$, $\{\alpha_{Q,a}^p, \alpha_{Q,a}^d\} \leq 1$, and

$$\begin{aligned}(x_Q)_j + \alpha_a^p\left(\Delta x_Q^a\right)_j &= 0, \qquad (s_Q)_r + \alpha_a^d\left(\Delta s_Q^a\right)_r = 0,\\ (x_Q)_j + \alpha_{Q,a}^p\left(\Delta\tilde{x}_Q^a\right)_j &= 0, \qquad (s_Q)_r + \alpha_{Q,a}^d\left(\Delta\tilde{s}_Q^a\right)_r = 0.\end{aligned}$$

Here, $\alpha_a^p = -(x_Q)_j/\left(\Delta x_Q^a\right)_j$, $\alpha_{Q,a}^p = -(x_Q)_j/\left(\Delta\tilde{x}_Q^a\right)_j$, $\alpha_a^d = -(s_Q)_r/\left(\Delta s_Q^a\right)_r$, and $\alpha_{Q,a}^d = -(s_Q)_r/\left(\Delta\tilde{s}_Q^a\right)_r$. The expression

$$\begin{aligned}\left|\alpha_a^p - \alpha_{Q,a}^p\right| &= \left|\frac{-(x_Q)_j}{\left(\Delta x_Q^a\right)_j} + \frac{(x_Q)_j}{\left(\Delta\tilde{x}_Q^a\right)_j}\right|\\ &= \left|\frac{(x_Q)_j\left[\left(\Delta\tilde{x}_Q^a\right)_j - \left(\Delta x_Q^a\right)_j\right]}{\left(\Delta x_Q^a\right)_j\cdot\left(\Delta\tilde{x}_Q^a\right)_j}\right|\\ &\leq \left|\frac{(x_Q)_j}{\left(\Delta\tilde{x}_Q^a\right)_j}\right|\cdot\left|\frac{\left(\Delta\tilde{x}_Q^a\right)_j - \left(\Delta x_Q^a\right)_j}{\left(\Delta x_Q^a\right)_j}\right|, \ \forall j\\ &\leq \left|\alpha_{Q,a}^p\right|\cdot\left|\frac{\left(\Delta\tilde{x}_Q^a - \Delta x_Q^a\right)_l}{\left(\Delta x_Q^a\right)_l}\right|,\end{aligned}$$

where l is the index of the maximum ratio. Continuing, we have

$$\begin{aligned}
\left|\alpha_a^p - \alpha_{Q,a}^p\right| &\le \left|\alpha_{Q,a}^p\right| \cdot \frac{\left\|\Delta\tilde{x}_Q^a - \Delta x_Q^a\right\|_1}{\left\|\Delta x_Q^a\right\|_\infty}, \text{ by (4.26)} \\
&\le \sqrt{k}\cdot\sqrt{k}\ \left(\frac{\left\|\Delta\tilde{x}_Q^a - \Delta x_Q^a\right\|}{\left\|\Delta x_Q^a\right\|}\right), \text{ by (4.27), (4.28)} \\
&\qquad\qquad\qquad\qquad\qquad \text{and } \left|\alpha_{Q,a}^p\right| \le 1 \\
&\le k \cdot \frac{\kappa\left\|z - z^*\right\| \left\|\Delta z_Q^a\right\|}{\left\|\Delta x_Q^a\right\|}, \text{ by (4.23)} \\
&\le k\kappa\left\|z - z^*\right\| \cdot \tilde{\kappa}, \text{ by (4.29)} \\
&= c_o\left\|z - z^*\right\|. \qquad (4.30)
\end{aligned}$$

where $c_o = k \cdot \kappa \cdot \tilde{\kappa}$. Similarily,

$$\begin{aligned}
\left|\alpha_a^d - \alpha_{Q,a}^d\right| &= \left|\frac{-(s_Q)_r}{\left(\Delta s_Q^a\right)_r} + \frac{(s_Q)_r}{\left(\Delta\tilde{s}_Q^a\right)_r}\right| \\
&= \left|\frac{(s_Q)_r\left[\left(\Delta\tilde{s}_Q^a\right)_r - \left(\Delta s_Q^a\right)_r\right]}{\left(\Delta s_Q^a\right)_r \cdot \left(\Delta\tilde{s}_Q^a\right)_r}\right| \\
&\le \left|\frac{(s_Q)_r}{\left(\Delta\tilde{s}_Q^a\right)_r}\right| \cdot \left|\frac{\left(\Delta\tilde{s}_Q^a\right)_r - \left(\Delta s_Q^a\right)_r}{\left(\Delta s_Q^a\right)_r}\right|, \ \forall r \\
&\le \left|\alpha_{Q,a}^d\right| \cdot \left|\frac{\left(\Delta\tilde{s}_Q^a - \Delta s_Q^a\right)_p}{\left(\Delta s_Q^a\right)_p}\right|,
\end{aligned}$$

where p is the index of the maximum ratio. Therefore,

$$\begin{aligned}
\left|\alpha_a^d - \alpha_{Q,a}^d\right| &\le \left|\alpha_{Q,a}^d\right| \cdot \frac{\left\|\Delta\tilde{s}_Q^a - \Delta s_Q^a\right\|_1}{\left\|\Delta s_Q^a\right\|_\infty}, \text{ by (4.26)} \\
&\le \sqrt{k}\cdot\sqrt{k}\ \left(\frac{\left\|\Delta\tilde{s}_Q^a - \Delta s_Q^a\right\|}{\left\|\Delta s_Q^a\right\|}\right), \text{ by (4.27), (4.28)} \\
&\le k \cdot \frac{\kappa\left\|z - z^*\right\| \left\|\Delta z_Q^a\right\|}{\left\|\Delta s_Q^a\right\|}, \text{ by (4.23)} \\
&\le k \cdot \kappa\left\|z - z^*\right\| \cdot \tilde{\kappa}, \text{ by (4.29)} \\
&= c_o\left\|z - z^*\right\|. \qquad (4.31)
\end{aligned}$$

Recall, the absolute value of the difference between μ^a and μ_Q^a is bounded by

$$\begin{aligned}
\left|\mu^a - \mu_Q^a\right| \leq \; & \frac{1}{n}\Big[\left|\alpha_a^d\right| \|x_Q\| \left\|\Delta s_Q^a - \Delta \tilde{s}_Q^a\right\| + \left|\alpha_a^d - \alpha_{Q,a}^d\right| \|x_Q\| \left\|\Delta \tilde{s}_Q^a\right\| \\
& + \left|\alpha_a^p\right| \|s_Q\| \left\|\Delta x_Q^a - \Delta \tilde{x}_Q^a\right\| + \left|\alpha_a^p - \alpha_{Q,a}^p\right| \|s_Q\| \left\|\Delta \tilde{x}_Q^a\right\| \\
& + \left|\alpha_a^p \alpha_a^d - \alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| \\
& + \left|\alpha_{Q,a}^p \alpha_{Q,a}^d \left[\left(\Delta x_Q^a\right)^T \Delta s_Q^a - \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right]\right| \\
& + \left|\alpha_a^d\right| \|x_{\bar{Q}}\| \left\|\Delta s_{\bar{Q}}^a\right\| + \left|\alpha_a^p\right| \left|s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a\right| + \left|\alpha_a^p \alpha_a^d\right| \left|\left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right|\Big].
\end{aligned}$$

Using the results from (4.23), (4.30), and (4.31), we can bound the 9 terms on the right-hand side of the above inequality:

1. $\left|\alpha_a^d\right| \|x_Q\| \left\|\Delta s_Q^a - \Delta \tilde{s}_Q^a\right\| \leq C_1 \cdot \kappa \|z - z^*\| \|\Delta z^a\|$ by assumption (A7), (4.23) and the fact that $\left|\alpha_a^d\right| \leq 1$.

2. To bound the second term, $\left|\alpha_a^d - \alpha_{Q,a}^d\right| \|x_Q\| \left\|\Delta \tilde{s}_Q^a\right\|$, we must first bound the quantity $\left\|\Delta \tilde{s}_Q^a\right\|$.

$$\begin{aligned}
\left\|\Delta \tilde{s}_Q^a\right\| &= \left\|\Delta \tilde{s}_Q^a - \mathbf{\Delta s_Q^a} + \mathbf{\Delta s_Q^a}\right\| \\
&\leq \left\|\Delta \tilde{s}_Q^a - \Delta s_Q^a\right\| + \left\|\Delta s_Q^a\right\| \\
&\leq \kappa \|z - z^*\| \|\Delta z^a\| + \|\Delta z^a\| \\
&= (1 + \kappa \|z - z^*\|) \|\Delta z^a\|. \qquad (4.32)
\end{aligned}$$

The third line follows from (4.23). Therefore,

$$\begin{aligned}
\left|\alpha_a^d - \alpha_{Q,a}^d\right| \|x_Q\| \left\|\Delta \tilde{s}_Q^a\right\| &\leq c_o \|z - z^*\| C_1 (1 + \kappa \|z - z^*\|) \|\Delta z^a\| \\
&= c_o C_1 (1 + \kappa \|z - z^*\|) \|z - z^*\| \|\Delta z^a\|.
\end{aligned}$$

This result follows from assumption (A7), (4.31), and (4.32).

3. $|\alpha_a^p| \|s_Q\| \left\|\Delta x_Q^a - \Delta \tilde{x}_Q^a\right\| \leq C_2 \cdot \kappa \|z - z^*\| \|\Delta z^a\|$ (similar to the bound on term 1).

4. $\left|\alpha_a^p - \alpha_{Q,a}^p\right| \|s_Q\| \left\|\Delta \tilde{x}_Q^a\right\| \leq c_o C_2 (1 + \kappa \|z - z^*\|) \|z - z^*\| \|\Delta z^a\|$ (similar to the bound on term 2).

5. $\left|\alpha_a^p \alpha_a^d - \alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\|$

$$\begin{aligned}
&= \left|\alpha_a^p \alpha_a^d - \boldsymbol{\alpha_a^p \alpha_{Q,a}^d} + \boldsymbol{\alpha_a^p \alpha_{Q,a}^d} - \alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| \\
&= \left|\alpha_a^p \left(\alpha_a^d - \alpha_{Q,a}^d\right) + \alpha_{Q,a}^d \left(\alpha_a^p - \alpha_{Q,a}^p\right)\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| \\
&\leq \left(|\alpha_a^p| \left|\alpha_a^d - \alpha_{Q,a}^d\right| + \left|\alpha_{Q,a}^d\right| \left|\alpha_a^p - \alpha_{Q,a}^p\right|\right) \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| \\
&\leq (c_o \|z - z^*\| + c_o \|z - z^*\|) \|\Delta z^a\| \|\Delta z^a\| \\
&= 2c_o \|z - z^*\| \|\Delta z^a\|^2 . \qquad (4.33)
\end{aligned}$$

The fourth line follows from (4.30), (4.31) and the fact that $|\alpha_a^p| \leq 1$ and $\left|\alpha_{Q,a}^d\right| \leq 1$.

6. $\left|\alpha_{Q,a}^p \alpha_{Q,a}^d \left[\left(\Delta x_Q^a\right)^T \Delta s_Q^a - \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right]\right|$

$$\begin{aligned}
&= \Big|\alpha_{Q,a}^p \alpha_{Q,a}^d \left(\Delta x_Q^a\right)^T \Delta s_Q^a - \boldsymbol{\alpha_a^p \alpha_{Q,a}^d \left(\Delta x_Q^a\right)^T \Delta \tilde{s}_Q^a} \\
&\quad + \boldsymbol{\alpha_a^p \alpha_{Q,a}^d \left(\Delta x_Q^a\right)^T \Delta \tilde{s}_Q^a} - \alpha_{Q,a}^p \alpha_{Q,a}^d \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\Big| \\
&= \Big|\alpha_{Q,a}^d \left(\Delta x_Q^a\right)^T \left(\alpha_{Q,a}^p \Delta s_Q^a - \alpha_a^p \Delta \tilde{s}_Q^a\right) \\
&\quad + \alpha_{Q,a}^d \left(\Delta \tilde{s}_Q^a\right)^T \left(\alpha_a^p \Delta x_Q^a - \alpha_{Q,a}^p \Delta \tilde{x}_Q^a\right)\Big| \\
&\leq \left|\alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\alpha_{Q,a}^p \Delta s_Q^a - \alpha_a^p \Delta \tilde{s}_Q^a\right\| \\
&\quad + \left|\alpha_{Q,a}^d\right| \left\|\Delta \tilde{s}_Q^a\right\| \left\|\alpha_a^p \Delta x_Q^a - \alpha_{Q,a}^p \Delta \tilde{x}_Q^a\right\| .
\end{aligned}$$

It suffices to bound $\left\|\alpha^p_{Q,a}\Delta s^a_Q - \alpha^p_a \Delta\tilde{s}^a_Q\right\|$ and $\left\|\alpha^p_a \Delta x^a_Q - \alpha^p_{Q,a}\Delta\tilde{x}^a_Q\right\|$. The bound for $\left\|\alpha^p_{Q,a}\Delta s^a_Q - \alpha^p_a \Delta\tilde{s}^a_Q\right\|$ is given by:

$$\begin{aligned}
&\left\|\alpha^p_{Q,a}\Delta s^a_Q - \alpha^p_a \Delta\tilde{s}^a_Q\right\| \\
&\qquad = \left\|\alpha^p_{Q,a}\Delta s^a_Q - \boldsymbol{\alpha^p_a \Delta s^a_Q} + \boldsymbol{\alpha^p_a \Delta s^a_Q} - \alpha^p_a \Delta\tilde{s}^a_Q\right\| \\
&\qquad \leq \left\|\left(\alpha^p_{Q,a} - \alpha^p_a\right)\Delta s^a_Q\right\| + \left\|\alpha^p_a\left(\Delta s^a_Q - \Delta\tilde{s}^a_Q\right)\right\| \\
&\qquad \leq \left|\alpha^p_{Q,a} - \alpha^p_a\right|\left\|\Delta s^a_Q\right\| + \left|\alpha^p_a\right|\left\|\Delta s^a_Q - \Delta\tilde{s}^a_Q\right\| \\
&\qquad \leq c_o\left\|z - z^*\right\|\left\|\Delta z^a\right\| + \kappa\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\qquad = \left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|. \qquad (4.34)
\end{aligned}$$

The fourth line follows from (4.30), (4.23) and the fact that $|\alpha^p_a| \leq 1$. A similar analysis shows that

$$\left\|\alpha^p_a \Delta x^a_Q - \alpha^p_{Q,a}\Delta\tilde{x}^a_Q\right\| \leq \left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|. \qquad (4.35)$$

Therefore,

$$\begin{aligned}
&\left|\alpha^p_{Q,a}\alpha^d_{Q,a}\left[\left(\Delta x^a_Q\right)^T\Delta s^a_Q - \left(\Delta\tilde{x}^a_Q\right)^T\Delta\tilde{s}^a_Q\right]\right| \\
&\qquad \leq \left|\alpha^d_{Q,a}\right|\left\|\Delta x^a_Q\right\|\left\|\alpha^p_{Q,a}\Delta s^a_Q - \alpha^p_a\Delta\tilde{s}^a_Q\right\| \\
&\qquad\quad + \left|\alpha^d_{Q,a}\right|\left\|\Delta\tilde{s}^a_Q\right\|\left\|\alpha^p_a\Delta x^a_Q - \alpha^p_{Q,a}\Delta\tilde{x}^a_Q\right\| \\
&\qquad \leq \left\|\Delta z^a\right\|\cdot\left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\qquad\quad + \left(1 + \kappa\left\|z - z^*\right\|\right)\left\|\Delta z^a\right\|\cdot\left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\qquad = \left(2 + \kappa\left\|z - z^*\right\|\right)\left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|^2.
\end{aligned}$$

7. $\left|\alpha_a^d\right| \left\|x_{\bar{Q}}\right\| \left\|\Delta s_{\bar{Q}}^a\right\|$

$$\begin{aligned}
&= \left|\alpha_a^d\right| \left\|x_{\bar{Q}} - x_{\bar{Q}}^*\right\| \left\|\Delta s_{\bar{Q}}^a\right\| \\
&\leq \|z - z^*\| \|\Delta z^a\|, \qquad (4.36)
\end{aligned}$$

since $\left|\alpha_a^d\right| \leq 1$.

8. Since $\Delta x_{\bar{Q}}^a = -x_{\bar{Q}} - S_{\bar{Q}}^{-1} X_{\bar{Q}} \Delta s_{\bar{Q}}^a$, we have

$$\begin{aligned}
\left|\alpha_a^p\right| \left|s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a\right| &= \left|\alpha_a^p\right| \left|s_{\bar{Q}}^T \left(-x_{\bar{Q}} - S_{\bar{Q}}^{-1} X_{\bar{Q}} \Delta s_{\bar{Q}}^a\right)\right| \\
&\leq \left|-x_{\bar{Q}}^T s_{\bar{Q}} - x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a\right| \\
&\leq \left|x_{\bar{Q}}^T s_{\bar{Q}}\right| + \left|x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a\right| \\
&\leq n\mu + \left\|x_{\bar{Q}}\right\| \left\|\Delta s_{\bar{Q}}^a\right\| \\
&\leq n\mu + \|z - z^*\| \|\Delta z^a\|, \qquad (4.37)
\end{aligned}$$

with $\left|\alpha_a^p\right| \leq 1$.

9. $\left|\alpha_a^p \alpha_a^d\right| \left|\left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right|$

$$\begin{aligned}
&= \left|\alpha_a^p \alpha_a^d\right| \left|\left(-x_{\bar{Q}} - S_{\bar{Q}}^{-1} X_{\bar{Q}} \Delta s_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right| \\
&= \left|\alpha_a^p \alpha_a^d\right| \left|-x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a - \left(\Delta s_{\bar{Q}}^a\right)^T X_{\bar{Q}} S_{\bar{Q}}^{-1} \Delta s_{\bar{Q}}^a\right| \\
&\leq \left|x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a\right| + \left|\left(\Delta s_{\bar{Q}}^a\right)^T X_{\bar{Q}} S_{\bar{Q}}^{-1} \Delta s_{\bar{Q}}^a\right| \\
&\leq \left\|x_{\bar{Q}}\right\| \left\|\Delta s_{\bar{Q}}^a\right\| + \left\|\Delta s_{\bar{Q}}^a\right\| \left\|X_{\bar{Q}}\right\| \left\|S_{\bar{Q}}^{-1}\right\| \left\|\Delta s_{\bar{Q}}^a\right\| \\
&\leq \|z - z^*\| \|\Delta z^a\| + \|\Delta z^a\| \|z - z^*\| \kappa_1 \|\Delta z^a\| \\
&= (1 + \kappa_1 \|\Delta z^a\|) \|z - z^*\| \|\Delta z^a\|, \qquad (4.38)
\end{aligned}$$

where $\left\|S_{\bar{Q}}^{-1}\right\| \leq \kappa_1$ follows from (3.24).

Combining the bounds in 1-9, we have

$$\begin{aligned}
\left|\mu^a - \mu_Q^a\right| &\leq \frac{1}{n}\left[\left|\alpha_a^d\right|\left\|x_Q\right\|\left\|\Delta s_Q^a - \Delta\tilde{s}_Q^a\right\| + \left|\alpha_a^d - \alpha_{Q,a}^d\right|\left\|x_Q\right\|\left\|\Delta\tilde{s}_Q^a\right\|\right. \\
&\quad + \left|\alpha_a^p\right|\left\|s_Q\right\|\left\|\Delta x_Q^a - \Delta\tilde{x}_Q^a\right\| + \left|\alpha_a^p - \alpha_{Q,a}^p\right|\left\|s_Q\right\|\left\|\Delta\tilde{x}_Q^a\right\| \\
&\quad + \left|\alpha_a^p\alpha_a^d - \alpha_{Q,a}^p\alpha_{Q,a}^d\right|\left\|\Delta x_Q^a\right\|\left\|\Delta s_Q^a\right\| \\
&\quad + \left|\alpha_{Q,a}^p\alpha_{Q,a}^d\left[\left(\Delta x_Q^a\right)^T\Delta s_Q^a - \left(\Delta\tilde{x}_Q^a\right)^T\Delta\tilde{s}_Q^a\right]\right| \\
&\quad \left. + \left|\alpha_a^d\right|\left\|x_{\bar{Q}}\right\|\left\|\Delta s_{\bar{Q}}^a\right\| + \left|\alpha_a^p\right|\left|s_{\bar{Q}}^T\Delta x_{\bar{Q}}^a\right| + \left|\alpha_a^p\alpha_a^d\right|\left|\left(\Delta x_{\bar{Q}}^a\right)^T\Delta s_{\bar{Q}}^a\right|\right] \\
&\leq \frac{1}{n}\left[C_1\kappa\left\|z - z^*\right\|\left\|\Delta z^a\right\| + c_o C_1\left(1 + \kappa\left\|z - z^*\right\|\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|\right. \\
&\quad C_2\kappa\left\|z - z^*\right\|\left\|\Delta z^a\right\| + c_o C_2\left(1 + \kappa\left\|z - z^*\right\|\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad + 2c_o\left\|z - z^*\right\|\left\|\Delta z^a\right\|^2 + \left(2 + \kappa\left\|z - z^*\right\|\right)\left(c_o + \kappa\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|^2 \\
&\quad + \left\|z - z^*\right\|\left\|\Delta z^a\right\| + n\mu + \left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad \left. + \left(1 + \kappa_1\left\|\Delta z^a\right\|\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\|\right].
\end{aligned}$$

We can simplify the last expression by factoring out the quantity $\left\|z - z^*\right\|\left\|\Delta z^a\right\|$

from some terms and grouping other terms. This gives

$$\begin{aligned}
\left|\mu^a - \mu_Q^a\right| &\leq \mu + \frac{1}{n}\left[C_1\kappa + c_o C_1\left(1 + \kappa\left\|z - z^*\right\|\right) + C_2\kappa + c_o C_2\left(1 + \kappa\left\|z - z^*\right\|\right)\right. \\
&\quad \left. + 2 + \left(1 + \kappa_1\left\|\Delta z^a\right\|\right)\right]\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad + \frac{1}{n}\left[2c_o + \left(2 + \kappa\left\|z - z^*\right\|\right)\left(c_o + \kappa\right)\right]\left\|z - z^*\right\|\left\|\Delta z^a\right\|^2 \\
&= \mu + \frac{1}{n}\left[\left(C_1 + C_2\right)\kappa + \left(C_1 + C_2\right)c_o\left(1 + \kappa\left\|z - z^*\right\|\right)\right. \\
&\quad \left. + 3 + \kappa_1\left\|\Delta z^a\right\|\right]\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad + \frac{1}{n}\left[2c_o + \left(2 + \kappa\left\|z - z^*\right\|\right)\left(c_o + \kappa\right)\right]\left\|z - z^*\right\|\left\|\Delta z^a\right\|^2 \\
&= \mu + \frac{1}{n}\left[\left(C_1 + C_2\right)\left(c_o + \kappa\right) + 3 + \kappa_1\left\|\Delta z^a\right\|\right. \\
&\quad \left. + \left(C_1 + C_2\right)c_o\kappa\left\|z - z^*\right\|\right]\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad + \left\|z - z^*\right\| \cdot \Theta\left(\left\|\Delta z^a\right\|^2\right).
\end{aligned}$$

Let

$$V_1 = \left(C_1 + C_2\right)\left(c_o + \kappa\right) + 3 + \kappa_1\left\|\Delta z^a\right\| \text{ and} \tag{4.39}$$

$$V_2 = \left(C_1 + C_2\right)\kappa. \tag{4.40}$$

Then

$$\begin{aligned}
\left|\mu^a - \mu_Q^a\right| &\leq \mu + \left(\frac{V_1 + c_o V_2\left\|z - z^*\right\|}{n}\right)\left\|z - z^*\right\|\left\|\Delta z^a\right\| \\
&\quad + \left\|z - z^*\right\| \cdot \Theta\left(\left\|\Delta z^a\right\|^2\right).
\end{aligned} \tag{4.41}$$

The absolute value of the remaining pieces in (4.25) can be bounded by

$$\left|\left(1/\mu^2\right)\left[\left(\mu^a\right)^2 + \mu^a \cdot \mu_Q^a + \left(\mu_Q^a\right)^2\right]\right| \leq \left|\frac{\mu^a + \mu_Q^a}{\mu}\right|^2. \tag{4.42}$$

In the analysis that follows, we concentrate on bounding the square root of the expression on the right of (4.42). We have

$$
\begin{aligned}
&\left|\left(\mu^a+\mu_Q^a\right)/\mu\right| \\
&\quad = (1/n\mu)\left[\left(x+\alpha_a^p\Delta x^a\right)^T\left(s+\alpha_a^d\Delta s^a\right)+\left(x+\alpha_{Q,a}^p\Delta\tilde{x}^a\right)^T\left(s+\alpha_{Q,a}^d\Delta\tilde{s}^a\right)\right] \\
&\quad = (1/n\mu)\Big[2\ x^Ts+\alpha_a^d x^T\Delta s^a+\alpha_a^p s^T\Delta x^a+\alpha_a^p\alpha_a^d\left(\Delta x^a\right)^T\Delta s^a \\
&\qquad + \alpha_{Q,a}^d x^T\Delta\tilde{s}^a+\alpha_{Q,a}^p s^T\Delta\tilde{x}^a+\alpha_{Q,a}^p\alpha_{Q,a}^d\left(\Delta\tilde{x}^a\right)^T\Delta\tilde{s}^a\Big].
\end{aligned}
$$

By partitioning the right-hand side of the last expression into Q and $\bar{Q}$, we can eliminate the terms involving $\Delta\tilde{x}_{\bar{Q}}^a$ and $\Delta\tilde{s}_{\bar{Q}}^a$ since $\Delta\tilde{x}_{\bar{Q}}^a=\Delta\tilde{s}_{\bar{Q}}^a=0$. This gives

$$
\begin{aligned}
\left|\left(\mu^a+\mu_Q^a\right)/\mu\right| &= 2+(1/n\mu)\Big[\alpha_a^d x_Q^T\Delta s_Q^a+\alpha_a^p s_Q^T\Delta x_Q^a+\alpha_a^p\alpha_a^d\left(\Delta x_Q^a\right)^T\Delta s_Q^a \\
&\quad +\alpha_a^d x_{\bar{Q}}^T\Delta s_{\bar{Q}}^a+\alpha_a^p s_{\bar{Q}}^T\Delta x_{\bar{Q}}^a+\alpha_a^p\alpha_a^d\left(\Delta x_{\bar{Q}}^a\right)^T\Delta s_{\bar{Q}}^a \\
&\quad + \alpha_{Q,a}^d x_Q^T\Delta\tilde{s}_Q^a+\alpha_{Q,a}^p s_Q^T\Delta\tilde{x}_Q^a+\alpha_{Q,a}^p\alpha_{Q,a}^d\left(\Delta\tilde{x}_Q^a\right)^T\Delta\tilde{s}_Q^a\Big].
\end{aligned}
$$

The first term on the right-hand side of the last expression follows since $\mu=\left(x^Ts\right)/n$. By adding in, subtracting out, and grouping certain terms, we have

$\left|\left(\mu^a + \mu_Q^a\right)/\mu\right|$

$$
\begin{aligned}
&= 2 + (1/n\mu)\left[\alpha_a^d x_Q^T \Delta s_Q^a - \boldsymbol{\alpha_{Q,a}^{d} x_Q^T \Delta \tilde{s}_Q^a} + \boldsymbol{\alpha_{Q,a}^{d} x_Q^T \Delta \tilde{s}_Q^a} + \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a\right.\\
&\quad + \alpha_a^p s_Q^T \Delta x_Q^a - \boldsymbol{\alpha_{Q,a}^{p} s_Q^T \Delta \tilde{x}_Q^a} + \boldsymbol{\alpha_{Q,a}^{p} s_Q^T \Delta \tilde{x}_Q^a} + \alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a\\
&\quad + \alpha_a^p \alpha_a^d \left(\Delta x_Q^a\right)^T \Delta s_Q^a - \boldsymbol{\alpha_{Q,a}^{p} \alpha_{Q,a}^{d} \left(\Delta x_Q^a\right)^T \Delta s_Q^a}\\
&\quad + \boldsymbol{\alpha_{Q,a}^{p} \alpha_{Q,a}^{d} \left(\Delta x_Q^a\right)^T \Delta s_Q^a} + \alpha_{Q,a}^p \alpha_{Q,a}^d \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\\
&\quad \left. + \alpha_a^d x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a + \alpha_a^p s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a + \alpha_a^p \alpha_a^d \left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right]\\
&= 2 + (1/n\mu)\left[x_Q^T\left(\alpha_a^d \Delta s_Q^a - \alpha_{Q,a}^d \Delta \tilde{s}_Q^a\right) + s_Q^T\left(\alpha_a^p \Delta x_Q^a - \alpha_{Q,a}^p \Delta \tilde{x}_Q^a\right)\right.\\
&\quad + 2\left(\alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a + \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a\right) + \left(\alpha_a^p \alpha_a^d - \alpha_{Q,a}^p \alpha_{Q,a}^d\right)\left(\Delta x_Q^a\right)^T \Delta s_Q^a\\
&\quad + \alpha_{Q,a}^p \alpha_{Q,a}^d \left[\left(\Delta x_Q^a\right)^T \Delta s_Q^a + \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right]\\
&\quad \left. + \alpha_a^d x_{\bar{Q}}^T \Delta s_{\bar{Q}}^a + \alpha_a^p s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a + \alpha_a^p \alpha_a^d \left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right].
\end{aligned}
$$

Therefore,

$\left|\left(\mu^a + \mu_Q^a\right)/\mu\right|$

$$
\begin{aligned}
&\le 2 + \left|\frac{1}{n\mu}\right| \left[\|x_Q\| \left\|\alpha_a^d \Delta s_Q^a - \alpha_{Q,a}^d \Delta \tilde{s}_Q^a\right\| + \|s_Q\| \left\|\alpha_a^p \Delta x_Q^a - \alpha_{Q,a}^p \Delta \tilde{x}_Q^a\right\|\right.\\
&\quad + 2\left|\alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a + \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a\right| + \left|\alpha_a^p \alpha_a^d - \alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\|\\
&\quad + \left|\alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left|\left(\Delta x_Q^a\right)^T \Delta s_Q^a + \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right|\\
&\quad \left. + \left|\alpha_a^d\right| \left\|x_{\bar{Q}}\right\| \left\|\Delta s_{\bar{Q}}^a\right\| + \left|\alpha_a^p\right| \left|s_{\bar{Q}}^T \Delta x_{\bar{Q}}^a\right| + \left|\alpha_a^p \alpha_a^d\right| \left|\left(\Delta x_{\bar{Q}}^a\right)^T \Delta s_{\bar{Q}}^a\right|\right].
\end{aligned}
$$

We can bound the 8 terms on the right-hand side of the above inequality by using some of the previous results:

I. $\|x_Q\| \left\|\alpha_a^d \Delta s_Q^a - \alpha_{Q,a}^d \Delta \tilde{s}_Q^a\right\| \le C_1 (c_o + \kappa) \|z - z^*\| \|\Delta z^a\|$, by assumption (A7) and (4.34) with α_a^p and $\alpha_{Q,a}^p$ replaced by α_a^d and $\alpha_{Q,a}^d$, respectively.

II. $\|s_Q\| \left\|\alpha_a^p \Delta x_Q^a - \alpha_{Q,a}^p \Delta \tilde{x}_Q^a\right\| \leq C_2 \left(c_o + \kappa\right) \|z - z^*\| \|\Delta z^a\|$, by assumption (A7) and (4.35).

III. $2\left|\alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a + \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a\right|$

$$\begin{aligned}
&= 2\left|\alpha_{Q,a}^p s_Q^T \Delta \tilde{x}_Q^a - \alpha_{\mathbf{Q,a}}^{\mathbf{p}} \mathbf{s_Q^T \Delta x_Q^a} + \alpha_{\mathbf{Q,a}}^{\mathbf{p}} \mathbf{s_Q^T \Delta x_Q^a}\right. \\
&\quad \left. + \alpha_{Q,a}^d x_Q^T \Delta \tilde{s}_Q^a - \alpha_{\mathbf{Q,a}}^{\mathbf{d}} \mathbf{x_Q^T \Delta s_Q^a} + \alpha_{\mathbf{Q,a}}^{\mathbf{d}} \mathbf{x_Q^T \Delta s_Q^a}\right| \\
&= 2\left|\alpha_{Q,a}^p s_Q^T \left(\Delta \tilde{x}_Q^a - \Delta x_Q^a\right) + \alpha_{Q,a}^d x_Q^T \left(\Delta \tilde{s}_Q^a - \Delta s_Q^a\right)\right. \\
&\quad \left. + \alpha_{Q,a}^p s_Q^T \Delta x_Q^a + \alpha_{Q,a}^d x_Q^T \Delta s_Q^a\right| \\
&\leq 2\left[\left|\alpha_{Q,a}^p\right| \|s_Q\| \left\|\Delta \tilde{x}_Q^a - \Delta x_Q^a\right\| + \left|\alpha_{Q,a}^d\right| \|x_Q\| \left\|\Delta \tilde{s}_Q^a - \Delta s_Q^a\right\|\right. \\
&\quad \left. + \left|\alpha_{Q,a}^p s_Q^T \Delta x_Q^a + \alpha_{Q,a}^d x_Q^T \Delta s_Q^a\right|\right] \\
&\leq 2\left[C_2 \kappa \|z - z^*\| \|\Delta z^a\| + C_1 \kappa \|z - z^*\| \|\Delta z^a\|\right. \\
&\quad \left. + \left|\max\left\{\alpha_{Q,a}^p, \alpha_{Q,a}^d\right\}\right| \left|s_Q^T \Delta x_Q^a + x_Q^T \Delta s_Q^a\right|\right] \\
&\leq 2\left[\left(C_1 + C_2\right) \kappa \|z - z^*\| \|\Delta z^a\| + n\mu\right].
\end{aligned}$$

The fourth line follows from (4.23). The fifth line follows since $\left|\max\left\{\alpha_{Q,a}^p, \alpha_{Q,a}^d\right\}\right| \leq 1$ and $\left|s_Q^T \Delta x_Q^a + x_Q^T \Delta s_Q^a\right| = \left|-x_Q^T s_Q\right| \leq n\mu$ by (2.4).

IV. $\left|\alpha_a^p \alpha_a^d - \alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| \leq 2c_o \|z - z^*\| \|\Delta z^a\|^2$, by (4.33).

V. $\left|\alpha_{Q,a}^p \alpha_{Q,a}^d\right| \left|\left(\Delta x_Q^a\right)^T \Delta s_Q^a + \left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right|$

$$\begin{aligned}
&\leq \left|\alpha_{Q,a}^p\right| \left|\alpha_{Q,a}^d\right| \left(\left|\left(\Delta x_Q^a\right)^T \Delta s_Q^a\right| + \left|\left(\Delta \tilde{x}_Q^a\right)^T \Delta \tilde{s}_Q^a\right|\right) \\
&\leq \left\|\Delta x_Q^a\right\| \left\|\Delta s_Q^a\right\| + \left\|\Delta \tilde{x}_Q^a\right\| \left\|\Delta \tilde{s}_Q^a\right\| \\
&\leq \|\Delta z^a\|^2 + \left(1 + \kappa \|z - z^*\|\right)^2 \|\Delta z^a\|^2 \\
&= \left[1 + \left(1 + \kappa \|z - z^*\|\right)^2\right] \|\Delta z^a\|^2.
\end{aligned}$$

The second line follows since $\left|\alpha_{Q,a}^{p}\right| \leq 1$ and $\left|\alpha_{Q,a}^{d}\right| \leq 1$. The third line follows from (4.32) since the bound on $\left\|\Delta\tilde{x}_{Q}^{a}\right\|$ is the same as the bound on $\left\|\Delta\tilde{s}_{Q}^{a}\right\|$.

The bounds for the remaining terms (VI. - VIII.) follow from (4.36) - (4.38). Combining the bounds in I-VIII., we have

$$
\begin{aligned}
&\left|\left(\mu^{a}+\mu_{Q}^{a}\right)/\mu\right| \\
&\leq\ 2+\left|\frac{1}{n\mu}\right|\left[\left\|x_{Q}\right\|\left\|\alpha_{a}^{d}\Delta s_{Q}^{a}-\alpha_{Q,a}^{d}\Delta\tilde{s}_{Q}^{a}\right\|+\left\|s_{Q}\right\|\left\|\alpha_{a}^{p}\Delta x_{Q}^{a}-\alpha_{Q,a}^{p}\Delta\tilde{x}_{Q}^{a}\right\|\right. \\
&\quad +2\left|\alpha_{Q,a}^{p}s_{Q}^{T}\Delta\tilde{x}_{Q}^{a}+\alpha_{Q,a}^{d}x_{Q}^{T}\Delta\tilde{s}_{Q}^{a}\right|+\left|\alpha_{a}^{p}\alpha_{a}^{d}-\alpha_{Q,a}^{p}\alpha_{Q,a}^{d}\right|\left\|\Delta x_{Q}^{a}\right\|\left\|\Delta s_{Q}^{a}\right\| \\
&\quad +\left|\alpha_{Q,a}^{p}\alpha_{Q,a}^{d}\right|\left|\left(\Delta x_{Q}^{a}\right)^{T}\Delta s_{Q}^{a}+\left(\Delta\tilde{x}_{Q}^{a}\right)^{T}\Delta\tilde{s}_{Q}^{a}\right| \\
&\quad \left.+\left|\alpha_{a}^{d}\right|\left\|x_{\bar{Q}}\right\|\left\|\Delta s_{\bar{Q}}^{a}\right\|+\left|\alpha_{a}^{p}\right|\left|s_{\bar{Q}}^{T}\Delta x_{\bar{Q}}^{a}\right|+\left|\alpha_{a}^{p}\alpha_{a}^{d}\right|\left|\left(\Delta x_{\bar{Q}}^{a}\right)^{T}\Delta s_{\bar{Q}}^{a}\right|\right] \\
&\leq\ 2+\left|\frac{1}{n\mu}\right|\left[C_{1}\left(c_{o}+\kappa\right)\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+C_{2}\left(c_{o}+\kappa\right)\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|\right. \\
&\quad +2\left[\left(C_{1}+C_{2}\right)\kappa\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+n\mu\right]+2c_{o}\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|^{2} \\
&\quad +\left[1+\left(1+\kappa\left\|z-z^{*}\right\|\right)^{2}\right]\left\|\Delta z^{a}\right\|^{2}+\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\| \\
&\quad \left.+n\mu+\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+\left(1+\kappa_{1}\left\|\Delta z^{a}\right\|\right)\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|\right].
\end{aligned}
$$

By factoring out $\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|$ from select terms, grouping other terms, and applying (4.39) and (4.40), we have

$$
\begin{aligned}
&\left|\left(\mu^{a}+\mu_{Q}^{a}\right)/\mu\right| \\
&\qquad \leq\ 2+\left|\frac{1}{n\mu}\right|\left(\left[2+\left(C_{1}+C_{2}\right)\left(c_{o}+\kappa\right)+\left(1+\kappa_{1}\left\|\Delta z^{a}\right\|\right)\right.\right. \\
&\qquad\quad \left.+2\left(C_{1}+C_{2}\right)\kappa\right]\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+3n\mu \\
&\qquad\quad \left.+\left(2c_{o}\left\|z-z^{*}\right\|+\left[1+\left(1+\kappa\left\|z-z^{*}\right\|\right)^{2}\right]\right)\left\|\Delta z^{a}\right\|^{2}\right) \\
&\qquad =\ 2+\left|\frac{1}{n\mu}\right|\left(V_{1}+2V_{2}\right)\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+3+\Theta\left(\left\|\Delta z^{a}\right\|^{2}\right) \\
&\qquad =\ 5+\left|\frac{1}{n\mu}\right|\left(V_{1}+2V_{2}\right)\left\|z-z^{*}\right\|\left\|\Delta z^{a}\right\|+\Theta\left(\left\|\Delta z^{a}\right\|^{2}\right).
\end{aligned}
$$

Squaring the last result gives

$$\left|\left(\mu^a + \mu_Q^a\right)/\mu\right|^2$$

$$\leq \quad 25 + \left|\frac{10}{n\mu}\right| \left(V_1 + 2V_2\right) \|z - z^*\| \|\Delta z^a\| + \Theta\left(\|\Delta z^a\|^2\right). \tag{4.43}$$

Combining the bounds in (4.41) and (4.43), we have the final result:

$$\begin{aligned}
\left|\left(\sigma - \sigma_Q\right)\mu\right| &= \left|\mu^a - \mu_Q^a\right| \left|\frac{\left(\mu^a\right)^2 + \mu^a \cdot \mu_Q^a + \left(\mu_Q^a\right)^2}{\mu^2}\right| \\
&\leq \left|\mu^a - \mu_Q^a\right| \left|\frac{\mu^a + \mu_Q^a}{\mu}\right|^2 \\
&\leq \left[\mu + \left(\frac{V_1 + c_o V_2 \|z - z^*\|}{n}\right) \|z - z^*\| \|\Delta z^a\| \right. \\
&\quad \left. + \|z - z^*\| \cdot \Theta\left(\|\Delta z^a\|^2\right)\right] \\
&\quad \cdot \left[25 + \left|\frac{10}{n\mu}\right| \left(V_1 + 2V_2\right) \|z - z^*\| \|\Delta z^a\| + \Theta\left(\|\Delta z^a\|^2\right)\right] \\
&= 25\mu + \frac{25}{n}\left(V_1 + c_o V_2 \|z - z^*\|\right) \|z - z^*\| \|\Delta z^a\| \\
&\quad + \frac{10}{n}\left(V_1 + 2V_2\right) \|z - z^*\| \|\Delta z^a\| \\
&\quad + \|z - z^*\| \cdot \Theta\left(\|\Delta z^a\|^2\right) + \mu \cdot \Theta\left(\|\Delta z^a\|^2\right) \\
&\leq 25\mu + V_3 \|z - z^*\| \|\Delta z^a\| \\
&\quad + \max\left\{\|z - z^*\|, \mu\right\} \cdot \Theta\left(\|\Delta z^a\|^2\right) \\
&\leq 25\mu + V_3 \|z - z^*\| \|\Delta z^a\| + \zeta \cdot \Theta\left(\|\Delta z^a\|^2\right),
\end{aligned} \tag{4.44}$$

where

$$\begin{aligned}
\zeta &= \max\left\{\|z - z^*\|, \mu\right\} \text{ and} \\
V_3 &= \frac{1}{n}\left[25\left(V_1 + c_o V_2 \|z - z^*\|\right) + 10\left(V_1 + 2V_2\right)\right].
\end{aligned}$$

[]

Lemma 4.2 *: Suppose assumption (A8) and the assumptions of Lemma 4.1 hold. Then,*

$$|\sigma\mu| \leq 25\mu + (V_3 + \kappa)\,\zeta\,\|\Delta z^a\| + \zeta \cdot \Theta\left(\|\Delta z^a\|^2\right).$$

Proof:

$$\begin{aligned}
|\sigma\mu| &= |\sigma\mu - \sigma_Q\mu + \sigma_Q\mu| \\
&\leq |(\sigma - \sigma_Q)\,\mu| + |\sigma_Q\mu| \\
&\leq 25\mu + V_3\,\|z - z^*\|\,\|\Delta z^a\| + \zeta \cdot \Theta\left(\|\Delta z^a\|^2\right) \\
&\quad +\kappa\mu\,\|\Delta z^a\| \\
&\leq 25\mu + (V_3 + \kappa)\max\left\{\|z - z^*\|\,, \mu\right\}\|\Delta z^a\| \\
&\quad +\zeta \cdot \Theta\left(\|\Delta z^a\|^2\right) \\
&= 25\mu + (V_3 + \kappa)\,\zeta\,\|\Delta z^a\| + \zeta \cdot \Theta\left(\|\Delta z^a\|^2\right).
\end{aligned}$$

[]

The next two lemmas give bounds on the error components, $\Delta X^a_{\bar{Q}} \Delta S^a_{\bar{Q}} e_{\bar{Q}}$ and $\left(\Delta X^a_Q \Delta S^a_Q - \Delta\tilde{X}^a_Q \Delta\tilde{S}^a_Q\right) e_Q$.

Lemma 4.3 *: Suppose (3.24) holds (i.e. $\left\|S^{-1}_{\bar{Q}}\right\| \leq \kappa_1$ where $\kappa_1 > 0$). Then*

$$\left\|\Delta X^a_{\bar{Q}} \Delta S^a_{\bar{Q}} e_{\bar{Q}}\right\| \leq (1 + \kappa_1\,\|\Delta z^a\|)\,\|z - z^*\|\,\|\Delta z^a\|.$$

Proof:

The $\Delta X^a_{\bar{Q}}$ component of the standard affine-scaling direction is given by $\Delta X^a_{\bar{Q}} =$

$-X_{\bar{Q}} - S_{\bar{Q}}^{-1} X_{\bar{Q}} \Delta S_{\bar{Q}}^a$. Therefore, we have

$$\begin{aligned}
\left\|\Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}}\right\| &= \left\|\left(-X_{\bar{Q}} - S_{\bar{Q}}^{-1} X_{\bar{Q}} \Delta S_{\bar{Q}}^a\right) \Delta S_{\bar{Q}}^a e_{\bar{Q}}\right\| \\
&\leq \left\|X_{\bar{Q}}\right\| \left\|\Delta s_{\bar{Q}}^a\right\| + \left\|S_{\bar{Q}}^{-1}\right\| \left\|X_{\bar{Q}}\right\| \left\|\Delta S_{\bar{Q}}^a\right\| \left\|\Delta s_{\bar{Q}}^a\right\| \\
&\leq \|z - z^*\| \|\Delta z^a\| + \kappa_1 \|z - z^*\| \|\Delta z^a\|^2 \\
&= (1 + \kappa_1 \|\Delta z^a\|) \|z - z^*\| \|\Delta z^a\| .
\end{aligned}$$

The third line follows since $\left\|X_{\bar{Q}}\right\| = \left\|X_{\bar{Q}} - X_{\bar{Q}}^*\right\| \leq \|z - z^*\|$ and $\left\|S_{\bar{Q}}^{-1}\right\| \leq \kappa_1$.

[]

Lemma 4.4 *: Suppose (4.23) and (4.32) holds. Then*

$$\left\|\left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a\right) e_Q\right\| \leq (2 + \kappa \|z - z^*\|) \kappa \|z - z^*\| \|\Delta z^a\|^2 .$$

Proof:

The analysis to bound $\left\|\Delta \tilde{x}_Q^a\right\|$ is similar to the analysis on the bound of $\left\|\Delta \tilde{s}_Q^a\right\|$ which is shown in (4.32). Therefore,

$$\begin{aligned}
&\left\|\left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a\right) e_Q\right\| \\
&= \left\|\left(\Delta X_Q^a \Delta S_Q^a - \mathbf{\Delta \tilde{X}_Q^a \Delta S_Q^a} + \mathbf{\Delta \tilde{X}_Q^a \Delta S_Q^a} - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a\right) e_Q\right\| \\
&= \left\|\left(\Delta X_Q^a - \Delta \tilde{X}_Q^a\right) \Delta s_Q^a + \Delta \tilde{X}_Q^a \left(\Delta s_Q^a - \Delta \tilde{s}_Q^a\right)\right\| \\
&\leq \left\|\Delta X_Q^a - \Delta \tilde{X}_Q^a\right\| \left\|\Delta s_Q^a\right\| + \left\|\Delta \tilde{X}_Q^a\right\| \left\|\Delta s_Q^a - \Delta \tilde{s}_Q^a\right\| \\
&\leq \kappa \|z - z^*\| \|\Delta z^a\| \|\Delta z^a\| + (1 + \kappa \|z - z^*\|) \|\Delta z^a\| \kappa \|z - z^*\| \|\Delta z^a\| \\
&= (2 + \kappa \|z - z^*\|) \kappa \|z - z^*\| \|\Delta z^a\|^2 ,
\end{aligned}$$

where the fourth line follows from (4.23) and (4.32). []

Before we state the theorem to show (4.24), we present a lemma to show the standand centering-corrector direction is bounded by the standard affine-scaling direction.

Lemma 4.5 *: Suppose (3.24), Assumptions (A7) - (A8) and the assumptions of Lemma (4.2) hold. Let*

$\left\|(A_Q S_Q^{-1} X_Q A_Q^T)^{-1}\right\| \leq C_3$ *where* $0 < C_3 < \infty$*. Then,*

$$\left\|\Delta z_Q^{cc}\right\| \leq \tau_3 \left\|\Delta z^a\right\|,$$

where

$$\begin{aligned}
\tau_3 &= \tau_2 + \left\|A_Q^T\right\| \tau_2 + \left(\tau_1 + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2\right), \\
\tau_2 &= C_2 \left\|A_Q\right\| \tau_1, \\
\tau_1 &= \kappa_1 \left(k\left[25 + (V_3 + \kappa)\zeta + \zeta \cdot \Theta\left(\left\|\Delta z^a\right\|\right)\right] + \left\|\Delta z^a\right\|\right).
\end{aligned}$$

Proof:

Let $\xi_Q = \sigma\mu S_Q^{-1} e_Q - S_Q^{-1} \Delta X_Q^a \Delta S_Q^a e_Q$. Then,

$$\begin{aligned}
\left\|\xi_Q\right\| &= \left\|\sigma\mu S_Q^{-1} e_Q - S_Q^{-1} \Delta X_Q^a \Delta S_Q^a e_Q\right\| \\
&\leq |\sigma\mu| \left\|S_Q^{-1}\right\| \left\|e_Q\right\| + \left\|S_Q^{-1}\right\| \left\|\Delta X_Q^a \Delta S_Q^a e_Q\right\| \\
&\leq \left[25\mu + (V_3 + \kappa)\zeta \left\|\Delta z^a\right\| + \zeta \cdot \Theta\left(\left\|\Delta z^a\right\|^2\right)\right] \cdot \kappa_1 \cdot k \\
&\quad + \kappa_1 \cdot \left\|\Delta z^a\right\|^2 \\
&\leq \kappa_1 \left(k\left[25 + (V_3 + \kappa)\zeta + \zeta \cdot \Theta\left(\left\|\Delta z^a\right\|\right)\right] + \left\|\Delta z^a\right\|\right) \cdot \max\left\{\left\|\Delta z^a\right\|, \mu\right\} \\
&= \tau_1 \cdot \max\left\{\left\|\Delta z^a\right\|, \mu\right\},
\end{aligned}$$

where the third line follows from (3.24) and Lemma (4.2) and

$$\tau_1 = \kappa_1 \left(k \left[25 + (V_3 + \kappa)\, \zeta + \zeta \cdot \Theta \left(\|\Delta z^a\| \right) \right] + \|\Delta z^a\| \right).$$

Using the previous result, we have

$$\begin{aligned} \|\Delta y^{cc}\| &= \left\| -(A_Q S_Q^{-1} X_Q A_Q^T)^{-1} A_Q \xi_Q \right\| \\ &\leq \left\| (A_Q S_Q^{-1} X_Q A_Q^T)^{-1} \right\| \|A_Q\| \|\xi_Q\| \\ &\leq C_3 \|A_Q\| \tau_1 \cdot \max\{\|\Delta z^a\|, \mu\} \\ &= \tau_2 \cdot \max\{\|\Delta z^a\|, \mu\}, \end{aligned}$$

where $\tau_2 = C_2 \|A_Q\| \tau_1$. The bounds on $\left\|\Delta s_Q^{cc}\right\|$ and $\left\|\Delta z_Q^{cc}\right\|$ are given by:

$$\begin{aligned} \left\|\Delta s_Q^{cc}\right\| &= \left\| -A_Q^T \Delta y^{cc} \right\| \\ &\leq \left\|A_Q^T\right\| \|\Delta y^{cc}\| \\ &\leq \left\|A_Q^T\right\| \tau_2 \cdot \max\{\|\Delta z^a\|, \mu\}, \\ \left\|\Delta x_Q^{cc}\right\| &= \left\| \xi_Q - S_Q^{-1} X_Q \Delta s_Q^{cc} \right\| \\ &\leq \|\xi_Q\| + \left\|S_Q^{-1}\right\| \|X_Q\| \left\|\Delta s_Q^{cc}\right\| \\ &\leq \tau_1 \cdot \max\{\|\Delta z^a\|, \mu\} + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2 \cdot \max\{\|\Delta z^a\|, \mu\} \\ &= \left(\tau_1 + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2\right) \cdot \max\{\|\Delta z^a\|, \mu\}, \end{aligned}$$

where the second to the last line follows from (3.24). Using the above bounds for the components of the centering-corrector direction, we have

$$\begin{aligned}
\left\|\Delta z_Q^{cc}\right\| &\le \|\Delta y^{cc}\| + \left\|\Delta s_Q^{cc}\right\| + \left\|\Delta x_Q^{cc}\right\| \\
&\le \tau_2 \cdot \max\{\|\Delta z^a\|, \mu\} + \left\|A_Q^T\right\| \tau_2 \cdot \max\{\|\Delta z^a\|, \mu\} \\
&\quad + \left(\tau_1 + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2\right) \cdot \max\{\|\Delta z^a\|, \mu\} \\
&= \left[\tau_2 + \left\|A_Q^T\right\| \tau_2 + \left(\tau_1 + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2\right)\right] \cdot \max\{\|\Delta z^a\|, \mu\} \\
&= \tau_3 \cdot \max\{\|\Delta z^a\|, \mu\},
\end{aligned}$$

where $\tau_3 = \tau_2 + \left\|A_Q^T\right\| \tau_2 + \left(\tau_1 + \kappa_1 C_1 \left\|A_Q^T\right\| \tau_2\right)$. $\square$

We now present the analysis to show (4.24).

Theorem 4.1 *Suppose Assumptions (A1) - (A8), (3.24), (4.23), $\left\|(A_Q S_Q^{-1} X_Q A_Q^T)^{-1}\right\| \le C_3$ where $0 < C_3 < \infty$, and $\mu = \mu_Q$ hold. Let*

$$\Upsilon = \{z : \|z - z^*\| < \delta,\ x > 0,\ s > 0\}$$

where $\delta > 0$. Suppose $J_a(A_Q, x_Q, s_Q)$ is nonsingular for all Q. Let Δz^a be the Newton direction at z. Then, there exists $\hat{\kappa} > 0$ such that

$$\left\|\Delta \tilde{z}_Q^{cc} - \Delta z_Q^{cc}\right\| \le \hat{\kappa} \cdot \Phi(z)$$

for all $z \in \Upsilon$ with $\Phi(z) \to 0$ as the optimal solution is approached.

Proof. Let $z = (x, y, s)$ where $x > 0$ and $s > 0$. Then, $\Delta \tilde{y}^{cc}$ and $\Delta \tilde{x}_Q^{cc}$ are given by

$$\begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta \tilde{y}^{cc} \\ \Delta \tilde{x}_Q^{cc} \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{\omega}_Q \end{bmatrix}, \tag{4.45}$$

where $\tilde{\omega}_Q = \sigma_Q \mu e_Q - \Delta \tilde{X}^a_Q \Delta \tilde{S}^a_Q e_Q$. The components of the centering-corrector direction, Δy^{cc} and Δx^{cc} satisfy

$$\begin{bmatrix} 0 & A \\ -XA^T & S \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega \end{bmatrix},$$

where $\omega = \sigma \mu e - \Delta X^a \Delta S^a e$. Partitioning the previous expression in terms of Q and $\bar{Q}$ gives

$$\begin{bmatrix} 0 & A_Q & A_{\bar{Q}} \\ -X_Q A_Q^T & S_Q & 0 \\ -X_{\bar{Q}} A_{\bar{Q}}^T & 0 & S_{\bar{Q}} \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \\ \Delta x^{cc}_{\bar{Q}} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega_Q \\ \omega_{\bar{Q}} \end{bmatrix}.$$

Premultiplying the third row by $-A_{\bar{Q}} S_{\bar{Q}}^{-1}$ and adding to the first row, we can eliminate $\Delta x^{cc}_{\bar{Q}}$ to obtain the following system of equations,

$$\begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \end{bmatrix} = \begin{bmatrix} -A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \omega_Q \end{bmatrix}.$$

Adding $\tilde{\omega}_Q$ to both sides of the last expression, we have

$$\begin{aligned} \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \end{bmatrix} &+ \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \tilde{\omega}_Q \end{bmatrix}. \end{aligned} \tag{4.46}$$

The right-hand side of (4.46) is precisely the right-hand side of (4.45). Let

$$J_a(A_Q, x_Q, s_Q) = \begin{bmatrix} 0 & A_Q \\ -X_Q A_Q^T & S_Q \end{bmatrix}.$$

Equating the left-hand sides of (4.45) and (4.46) gives

$$\begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}X_{\bar{Q}}A_{\bar{Q}}^T & A_Q \\ -X_QA_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} + \begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}\omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} = J_a(A_Q, x_Q, s_Q) \begin{bmatrix} \Delta \tilde{y}^{cc} \\ \Delta \tilde{x}_Q^{cc} \end{bmatrix}.$$

Notice that $\begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}X_{\bar{Q}}A_{\bar{Q}}^T & A_Q \\ -X_QA_Q^T & S_Q \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} =$

$$J_a(A_Q, x_Q, s_Q) \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} + \begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}X_{\bar{Q}}A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix}.$$

Now solving for $\Delta \tilde{z}_Q^{cc} - \Delta z_Q^{cc}$, we have

$$\begin{aligned} \begin{bmatrix} \Delta \tilde{y}^{cc} \\ \Delta \tilde{x}_Q^{cc} \end{bmatrix} - \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} &= J_a(A_Q, x_Q, s_Q)^{-1} \begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}X_{\bar{Q}}A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} \\ &+ J_a(A_Q, x_Q, s_Q)^{-1} \begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}\omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix}. \end{aligned}$$

Taking norms of both sides, we have

$$\left\| \begin{bmatrix} \Delta\tilde{y}^{cc} \\ \Delta\tilde{x}^{cc}_Q \end{bmatrix} - \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \end{bmatrix} \right\|$$

$$\leq \left\| J_a(A_Q, x_Q, s_Q)^{-1} \right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \right\| \left\| \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \end{bmatrix} \right\| \tag{4.47}$$

$$+ \left\| J_a(A_Q, x_Q, s_Q)^{-1} \right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} \right\|. \tag{4.48}$$

The bound on (4.47) is given by

$$\left\| J_a(A_Q, x_Q, s_Q)^{-1} \right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \right\| \left\| \begin{bmatrix} \Delta y^{cc} \\ \Delta x^{cc}_Q \end{bmatrix} \right\|$$

$$\leq \kappa_2 \left\| z - z^* \right\| \left\| \Delta z^{cc}_Q \right\|$$

$$\leq \kappa_2 \left\| z - z^* \right\| \tau_3 \max \left\{ \left\| \Delta z^a \right\|, \mu \right\}. \tag{4.49}$$

The first inequality follows from (3.25) and the last inequality follows from Lemma 4.5. We will now show a bound for

$$\left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} \right\|.$$

Let $M = A_{\bar{Q}} S_{\bar{Q}}^{-1}$; then

$$\left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} \right\|^2$$

$$= \left\| \begin{bmatrix} M \left(\sigma \mu e_{\bar{Q}} - \Delta X^a_{\bar{Q}} \Delta S^a_{\bar{Q}} e_{\bar{Q}} \right) \\ \left[(\sigma_Q - \sigma) \mu e_Q - \left(\Delta\tilde{X}^a_Q \Delta\tilde{S}^a_Q - \Delta X^a_Q \Delta S^a_Q \right) \right] e_Q \end{bmatrix} \right\|^2.$$

Expanding the right-hand side of this norm, we have

$$\begin{aligned}
&\left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix} \right\|^2 \\
\leq\ & \left\| M \Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}} \right\|^2 + (\sigma\mu)^2 \left\| M e_{\bar{Q}} \right\|^2 + \left\| \left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a \right) e_Q \right\|^2 \\
& + \left[(\sigma - \sigma_Q) \mu \right]^2 \left\| e_Q \right\|^2 + 2 \left| \sigma\mu \right| \left\| \Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}} \right\| \left\| M e_{\bar{Q}} \right\| \\
& + 2 \left| (\sigma - \sigma_Q) \mu \right| \left\| e_Q \right\| \left\| \left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a \right) e_Q \right\| \\
\leq\ & \left\| M \right\|^2 \left\| \Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}} \right\|^2 + (n-k) \left\| M \right\|^2 (\sigma\mu)^2 \\
& + \left\| \left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a \right) e_Q \right\|^2 + k \left[(\sigma - \sigma_Q) \mu \right]^2 \\
& + 2\sqrt{n-k} \left\| M \right\| \left| \sigma\mu \right| \left\| \Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}} \right\| \\
& + 2\sqrt{k} \left| (\sigma - \sigma_Q) \mu \right| \left\| \left(\Delta X_Q^a \Delta S_Q^a - \Delta \tilde{X}_Q^a \Delta \tilde{S}_Q^a \right) e_Q \right\|.
\end{aligned}$$

The six terms on the right-hand side of the last inequality can be bounded by previous results within this section. Let $\zeta = \max\left\{ \left\| z - z^* \right\|, \mu \right\}$. The bounds are as follows:

i. $\left\| M \right\|^2 \left\| \Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}} \right\|^2$

$$\begin{aligned}
&\leq \left\| M \right\|^2 \left(1 + \kappa_1 \left\| \Delta z \right\| \right)^2 \left\| z - z^* \right\|^2 \left\| \Delta z^a \right\|^2 \\
&\leq \zeta^2 \cdot \Theta\left(\left\| \Delta z^a \right\|^2 \right).
\end{aligned}$$

The second line follows from Lemma 4.3.

ii. $(n-k)\left\|M\right\|^2(\sigma\mu)^2$

$$\begin{aligned}
&\leq (n-k)\left\|M\right\|^2\left[25\mu+(V_3+\kappa)\zeta\left\|\Delta z^a\right\|+\zeta\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)\right]^2\\
&= (n-k)\left\|M\right\|^2\left[625\mu^2+(V_3+\kappa)^2\zeta^2\left\|\Delta z^a\right\|^2+50\mu(V_3+\kappa)\zeta\left\|\Delta z^a\right\|\right.\\
&\quad\left.+50\mu\zeta\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)\right]\\
&\leq (n-k)\left\|M\right\|^2\left[625\zeta^2+\zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|\right)+\zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)\right]\\
&\leq \zeta^2\left[625(n-k)\left\|M\right\|^2+\Theta\left(\left\|\Delta z^a\right\|\right)\right].
\end{aligned}$$

The bound on $\sigma\mu$ follows from Lemma 4.2.

iii. $\left\|\left(\Delta X_Q^a\Delta S_Q^a-\Delta\tilde{X}_Q^a\Delta\tilde{S}_Q^a\right)e_Q\right\|^2$

$$\begin{aligned}
&\leq (2+\kappa\left\|z-z^*\right\|)^2\kappa^2\left\|z-z^2\right\|^2\left\|\Delta z^a\right\|^4\\
&\leq \zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|^4\right).
\end{aligned}$$

The second line follows from Lemma 4.4.

iv. $k\left[(\sigma-\sigma_Q)\mu\right]^2$

$$\begin{aligned}
&\leq k\left[25\mu+V_3\left\|z-z^*\right\|\left\|\Delta z^a\right\|+\zeta\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)\right]^2\\
&\leq k\left[625\mu^2+V_3^2\zeta^2\left\|\Delta z^a\right\|^2+\zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|^4\right)\right.\\
&\quad\left.+50\mu V_3\zeta\left\|\Delta z^a\right\|+50\mu\zeta\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)+2V_3\zeta\cdot\Theta\left(\left\|\Delta z^a\right\|^3\right)\right]\\
&\leq k\left[625\zeta^2+\zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|\right)-\zeta^2\cdot\Theta\left(\left\|\Delta z^a\right\|^2\right)\right]\\
&\leq \zeta^2\left[625k+\Theta\left(\left\|\Delta z^a\right\|\right)\right].
\end{aligned}$$

The second line follows from Lemma 4.1.

v. $2\sqrt{n-k}\,\|M\|\,|\sigma\mu|\,\left\|\Delta X^a_{\bar{Q}}\Delta S^a_{\bar{Q}}e_{\bar{Q}}\right\|$

$$\begin{aligned}
&\leq 2\sqrt{n-k}\,\|M\|\left[25\mu+(V_3+\kappa)\,\zeta\,\|\Delta z^a\|+\zeta\cdot\Theta\left(\|\Delta z^a\|^2\right)\right]\cdot\\
&\quad\left[(1+\kappa_1\|\Delta z^a\|)\,\|z-z^*\|\,\|\Delta z^a\|\right]\\
&\leq \zeta^2\cdot\Theta\left(\|\Delta z^a\|\right).
\end{aligned}$$

The second line follows from Lemmas 4.2 and 4.3.

vi. $2\sqrt{k}\,|(\sigma-\sigma_Q)\,\mu|\,\left\|\left(\Delta X^a_Q\Delta S^a_Q-\Delta\tilde{X}^a_Q\Delta\tilde{S}^a_Q\right)e_Q\right\|$

$$\begin{aligned}
&\leq 2\sqrt{k}\left[25\mu+V_3\,\|z-z^*\|\,\|\Delta z^a\|+\zeta\cdot\Theta\left(\|\Delta z^a\|^2\right)\right]\cdot\\
&\quad\left[(2+\kappa\,\|z-z^*\|)\,\kappa\,\|z-z^*\|\,\|\Delta z^a\|^2\right]\\
&\leq \zeta^2\cdot\Theta\left(\|\Delta z^a\|^2\right)+\zeta^2\cdot\Theta\left(\|\Delta z^a\|^3\right)+\zeta^2\cdot\Theta\left(\|\Delta z^a\|^4\right)\\
&\leq \zeta^2\cdot\Theta\left(\|\Delta z^a\|^2\right).
\end{aligned}$$

The second line follows from Lemmas 4.1 and 4.4.

Combining i.-vi., we have

$$\left\|\begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}\omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix}\right\|^2$$

$$\begin{aligned}
&\leq \|M\|^2 \left\|\Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}}\right\|^2 + (n-k)\|M\|^2(\sigma\mu)^2 \\
&\quad + \left\|\left(\Delta X_Q^a \Delta S_Q^a - \Delta\tilde{X}_Q^a \Delta\tilde{S}_Q^a\right) e_Q\right\|^2 + k\left[(\sigma-\sigma_Q)\mu\right]^2 \\
&\quad + 2\sqrt{n-k}\,\|M\|\,|\sigma\mu| \left\|\Delta X_{\bar{Q}}^a \Delta S_{\bar{Q}}^a e_{\bar{Q}}\right\| \\
&\quad + 2\sqrt{k}\,|(\sigma-\sigma_Q)\mu| \left\|\left(\Delta X_Q^a \Delta S_Q^a - \Delta\tilde{X}_Q^a \Delta\tilde{S}_Q^a\right) e_Q\right\| \\
&\leq \zeta^2 \cdot \left(\|\Delta z^a\|^2\right) + \zeta^2\left[625(n-k)\|M\|^2 + \Theta\left(\|\Delta z^a\|\right)\right] \\
&\quad + \zeta^2 \cdot \Theta\left(\|\Delta z^a\|^4\right) + \zeta^2\left[625k + \Theta\left(\|\Delta z^a\|\right)\right] + \zeta^2 \cdot \Theta\left(\|\Delta z^a\|\right) \\
&\quad + \zeta^2 \cdot \Theta\left(\|\Delta z^a\|^2\right).
\end{aligned}$$

Factoring out ζ^2 and combining like terms gives

$$\left\|\begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}\omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix}\right\|^2$$

$$\begin{aligned}
&\leq \zeta^2\left[625\left((n-k)\|M\|^2 + k\right) + \Theta\left(\|\Delta z^a\|\right) + \Theta\left(\|\Delta z^a\|^2\right)\right. \\
&\quad \left. + \Theta\left(\|\Delta z^a\|^4\right)\right] \\
&\leq \zeta^2\left[625\left((n-k)\|M\|^2 + k\right) + \Theta\left(\|\Delta z^a\|\right)\right].
\end{aligned}$$

Taking the square root of both sides, we have

$$\left\|\begin{bmatrix} A_{\bar{Q}}S_{\bar{Q}}^{-1}\omega_{\bar{Q}} \\ \tilde{\omega}_Q - \omega_Q \end{bmatrix}\right\| \leq \zeta\sqrt{\kappa_3 + \Theta\left(\|\Delta z^a\|\right)}, \tag{4.50}$$

where

$$\kappa_3 = 625\,(n-k)\,\|M\|^2.$$

Combining (4.47), (4.48), (4.49), and (4.50), we have

$$\begin{aligned}
\left\|\Delta\tilde{z}_Q^{cc} - \Delta z_Q^{cc}\right\| &\leq \left\|J_a(A_Q, x_Q, s_Q)^{-1}\right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} X_{\bar{Q}} A_{\bar{Q}}^T & 0 \\ 0 & 0 \end{bmatrix} \right\| \left\| \begin{bmatrix} \Delta y^{cc} \\ \Delta x_Q^{cc} \end{bmatrix} \right\| \\
&\quad + \left\|J_a(A_Q, x_Q, s_Q)^{-1}\right\| \left\| \begin{bmatrix} A_{\bar{Q}} S_{\bar{Q}}^{-1} \omega_{\bar{Q}} \\ \omega_Q - \tilde{\omega}_Q \end{bmatrix} \right\| \\
&\leq \kappa_2 \left\|z - z^*\right\| \tau_3 \cdot \max\left\{\left\|\Delta z^a\right\|, \mu\right\} + \zeta\kappa_0\sqrt{\kappa_3 + \Theta\left(\left\|\Delta z^a\right\|\right)} \\
&\leq \zeta \cdot \Theta\left(\max\left\{\left\|\Delta z^a\right\|, \mu\right\}\right) + \zeta\kappa_0\sqrt{\kappa_3 + \Theta\left(\left\|\Delta z^a\right\|\right)} \\
&\leq \hat{\kappa} \cdot \zeta
\end{aligned}$$

where $\hat{\kappa} = \kappa_0 \max\left\{\Theta\left(\max\left\{\left\|\Delta z^a\right\|, \mu\right\}\right), \sqrt{\kappa_3 + \Theta\left(\left\|\Delta z^a\right\|\right)}\right\} = \sqrt{\kappa_3 + \Theta\left(\left\|\Delta z^a\right\|\right)} > 0$ since $\Theta\left(\max\left\{\left\|\Delta z^a\right\|, \mu\right\}\right) \to 0$ and $\kappa_3 > 0$. Furthermore, let $\Phi(z) = \zeta$. Then, $\Phi(z) = \max\left\{\left\|z - z^*\right\|, \mu\right\} \to 0$ as the optimal solution is approached. []

Global and local quadratic convergence follow from Winternitz et al. [28] with five specific changes to the *redMPC* algorithm. The differences between the *redMPC* algorithm and its modified version are summarized in Table 4.1. The first modification is maintaining dual-feasibility at each iteration. This implies $s^i = c - A^T y^i > 0$ and $r_d^i = 0$ for all i. Consequently, the algorithm for updating (x^+, s^+) and r_d has no effect on this dual-feasible algorithm and can be discarded. Furthermore, equations (4.8), (4.12), (4.9), and (4.16) must be computed with the full data to satisfy the above requirement on the slack variables. As a result, equation (4.19) must be eliminated from the algorithm. The second difference is the formula

	redMPC **algorithm**	**Modified** *redMPC* **algorithm**
Feasibility:	Accepts dual-infeasible initial point; strives to achieve dual-feasibility	Requires dual-feasible point at each iteration
Terms with Q:	$A_Q D_Q^2 A_Q^T$, $\Delta\tilde{x}_Q^a$, $\Delta\tilde{s}_Q^a$, $\Delta\tilde{x}_Q^{cc}$, $\Delta\tilde{s}_Q^{cc}$, x_Q, s_Q, $(r_d)_Q$	$A_Q D_Q^2 A_Q^T$, $\Delta\tilde{x}_Q^a$, x_Q
Centering parameter:	$\sigma_Q = \left(\mu_Q^a/\mu_Q\right)^3$ where $\mu_Q^a = \frac{\left(x_Q+\alpha_{Q,a}^p\Delta\tilde{x}_Q^a\right)^T\left(s_Q+\alpha_{Q,a}^d\Delta\tilde{s}_Q^a\right)}{n}$	$\sigma_Q = (1-\alpha_{Q,a})^\rho$ where $\rho \geq 2$
MPC direction:	$\Delta\tilde{x}_Q = \Delta\tilde{x}_Q^a + \Delta\tilde{x}_Q^{cc}$ $\Delta\tilde{y} = \Delta\tilde{y}^a + \Delta\tilde{y}^{cc}$ $\Delta\tilde{s}_Q = \Delta\tilde{s}_Q^a + \Delta\tilde{s}_Q^{cc}$	$\Delta\tilde{x}_Q = \Delta\tilde{x}_Q^a + \gamma\Delta\tilde{x}_Q^{cc}$ $\Delta\tilde{y} = \Delta\tilde{y}^a + \gamma\Delta\tilde{y}^{cc}$ $\Delta\tilde{s} = \Delta\tilde{s}^a + \gamma\Delta\tilde{s}^{cc}$ where $0 < \gamma < 1$
Predictor-Corrector step:	$\alpha_Q^p = \tau \ \min\left(\hat{\alpha}_Q^p, 1\right)$ $\alpha_Q^d = \tau \ \min\left(\hat{\alpha}_Q^d, 1\right)$ where $0 < \tau < 1$	$\alpha_Q^p = \max\left\{\tau\hat{\alpha}_a^p, \hat{\alpha}_Q^p - \|\Delta y^a\|\right\}$ $\alpha_Q^d = \max\left\{\tau\hat{\alpha}_Q^d, \hat{\alpha}_Q^d - \|\Delta y^a\|\right\}$ where $0 < \tau < 1$
Update scheme:	General update strategy for x^+, y^+, and s^+; unique update scheme for r_d^+ with further updates for x^+ and s^+	General update strategy for y^+ and s^+; unique update scheme for x^+

Table 4.1: A summary of the differences between the *redMPC* algorithm and a modified version of the algorithm adapted from Winternitz et. al. [28] that proves to be locally and quadratically convergent.

for the centering parameter, σ_Q. Replacing (4.11) with

$$\sigma_Q = (1 - \alpha_{Q,a})^{\rho} ,$$

where $\alpha_{Q,a} = \min\left\{\alpha^p_{Q,a}, \alpha^d_{Q,a}\right\}$ and $\rho \geq 2$, simplifies the analysis. When $\rho = 3$, (x, y, s) are primal and dual feasible, and $\alpha_{Q,a} = \alpha^p_{Q,a} = \alpha^d_{Q,a}$, the formulas agree. The third change involves replacing (4.13) - (4.15) with

$$(\Delta x_Q, \Delta y, \Delta s) = \left(\Delta x^a_Q, \Delta y^a, \Delta s^a\right) + \gamma\left(\Delta x^{cc}_Q, \Delta y^{cc}, \Delta s^{cc}\right),$$

where $\gamma \in (0, 1]$ is a mixing parameter. This parameter is essential to the analysis; it is defined to ensure certain properties hold for convergence. In particular, γ ensures

(1) Δy is an ascent direction for $b^T y$ (i.e. $b^T \Delta y > 0$ implies $b^T y^+ > b^T y$),

(2) if Δy^a is "small" then Δy and $\gamma \sigma_Q \mu_Q$ are also "small", and

(3) $\hat{\alpha}^d_Q \geq \zeta \alpha^d_{Q,a}$ where $\zeta \in (0, 1)$.

See Winternitz et al. [28] for details. The fourth modification involves incorporating lower bounds into the predictor-corrector step. The equations in (4.17) can be replaced with

$$\alpha^p_Q = \max\left\{\tau \hat{\alpha}^p_a, \hat{\alpha}^p_Q - \|\Delta y^a\|\right\} \tag{4.51}$$

and

$$\alpha^d_Q = \max\left\{\tau \hat{\alpha}^d_Q, \hat{\alpha}^d_Q - \|\Delta y^a\|\right\} \tag{4.52}$$

respectively, where the primal $(\hat{\alpha}^p_Q - \|\Delta y^a\|)$ and dual $(\hat{\alpha}^p_Q - \|\Delta y^a\|)$ lower bounds allow local quadratic convergence. The fifth change to the *redMPC* algorithm is in the update scheme for the primal variables. The $\Delta\tilde{x}_Q^a$ and $\Delta\tilde{x}_Q^{cc}$ components of the normal equations remain the same while the update in (4.20) - (4.22) is replaced by

$$\left(\hat{x}_Q, y^+, s^+\right) \quad = \quad \left(x_Q + \alpha_Q^p \Delta\tilde{x}, y + \alpha_Q^d \Delta\tilde{y}, s + \alpha_Q^d \Delta\tilde{s}\right),$$

where α_Q^p and α_Q^d are given in (4.51) and (4.52), respectively. Let $\underline{x} > 0$, $x_j \leq x_{\max}$ for all $j \in N$, and $k = |Q|$. The primal update, x^+, is defined as follows:

For all $j \in Q$,

$$x_j^+ \quad = \quad \min\left\{\max\left(\min\left\{\|\Delta y^a\|^\nu + \left\|\tilde{x}_-^a\right\|^\nu, \underline{x}\right\}, \hat{x}_j\right), x_{\max}\right\}, \qquad (4.53)$$

where

$$\begin{aligned} \tilde{x}_j^a &= \begin{cases} x_j + \Delta\tilde{x}_j^a & j \in Q \\ 0 & j \in \bar{Q} \end{cases}, \\ \left(\tilde{x}_-^a\right)_j &= \min\left\{\tilde{x}_j^a, 0\right\}. \end{aligned}$$

Using (4.53), we can express μ_Q^+ as

$$\mu_Q^+ = \frac{\left(x_Q^+\right)^T s_Q^+}{k}.$$

Thus, for all $j \in \bar{Q}$, we have

$$\begin{aligned} \hat{x}_j &= \frac{\mu_Q^+}{s_j^+}, \\ x_j^+ &= \min\left\{\hat{x}_j, x_{\max}\right\}. \end{aligned}$$

The dual-feasible *redMPC* algorithm is presented below:

The Dual-Feasible *redMPC* Algorithm

Input: (x, y, s) with $x > 0, x_{\max} > 0$ such that $x_j \leq x_{\max}$ $\forall j \in N$ and $s = c - A^T y >$

0, $\underline{x} > 0$, $\rho \geq 2$, $\nu \geq 2$, $u_{bnd} \geq 3m$, $0 < \tau < 1$, convergence tolerance λ.

Initialize: $l_{bnd} = 3m$, $Q = \{1, 2, \ldots, n\}$, $\hat{Q} = \emptyset$.

Main Algorithm:

while $|c^T x - b^T y| / \max|c^T x, 1| > \lambda$

Select the most promising dual constraints [4]

$$\begin{aligned} Q &= Q \cup \hat{Q}, \\ k &= |Q|, \\ \mu_Q &= \frac{x_Q^T s_Q}{k}. \end{aligned}$$

Compute the affine-scaling direction:

$$\begin{aligned} \Delta\tilde{y}^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} b, \\ \Delta\tilde{s}^a &= -A^T \Delta\tilde{y}^a, \\ \Delta\tilde{x}_Q^a &= -x_Q - S_Q^{-1} X_Q \Delta\tilde{s}_Q^a. \end{aligned}$$

Compute the affine step:

$$\hat{\alpha}_{Q,a}^p = \begin{cases} 1 & \text{if } \left(\Delta\tilde{x}_Q^a\right)_j \geq 0, \ \forall j \\ \min_{\left(\Delta\tilde{x}_Q^a\right)_j < 0} \left[-(x_Q)_j / \left(\Delta\tilde{x}_Q^a\right)_j\right] & \text{otherwise} \end{cases},$$

$$\hat{\alpha}_{Q,a}^d = \begin{cases} 1 & \text{if } \ \Delta\tilde{s}_j^a \geq 0, \ \forall j \\ \min_{\Delta\tilde{s}_j^a < 0} \left[-(s_Q)_j / \Delta\tilde{s}_j^a\right] & \text{otherwise} \end{cases},$$

$$\alpha_{Q,a} = \min\left\{\hat{\alpha}_{Q,a}^p, \hat{\alpha}_{Q,a}^d\right\}.$$

Compute the centering parameter:

$$\sigma_Q = (1 - \alpha_{Q,a})^\rho.$$

[4]The set Q is determined by the algorithm in Section 2.2.2: Selection of Q

Compute the centering-corrector direction:

$$\begin{aligned}
\xi_Q &= -S_Q^{-1}\Delta\tilde{X}_Q^a\Delta\tilde{S}_Q^a e_Q - \sigma_Q\mu_Q S_Q^{-1}e_Q,\\
\Delta\tilde{y}^{cc} &= -(A_Q S_Q^{-1}X_Q A_Q^T)^{-1}A_Q\xi_Q,\\
\Delta\tilde{s}^{cc} &= -A^T\Delta\tilde{y}^{cc},\\
\Delta\tilde{x}_Q^{cc} &= \xi_Q - S_Q^{-1}X_Q\Delta\tilde{s}_Q^{cc}.
\end{aligned}$$

Compute the predictor-corrector direction:

$$\begin{aligned}
\Delta\tilde{x}_Q &= \Delta\tilde{x}_Q^a + \gamma\Delta\tilde{x}_Q^{cc},\\
\Delta\tilde{y} &= \Delta\tilde{y}^a + \gamma\Delta\tilde{y}^{cc},\\
\Delta\tilde{s} &= \Delta\tilde{s}^a + \gamma\Delta\tilde{s}^{cc},
\end{aligned}$$

where $\gamma \in (0,1]$ ensures $(1)-(3)$ hold.

Compute the predictor-corrector step:

$$\begin{aligned}
\hat{\alpha}_Q^p &= \begin{cases} 1 & \text{if } (\Delta\tilde{x}_Q)_j > 0,\ \forall j \\ \min_{(\Delta\tilde{x}_Q)_j<0}\left[-(x_Q)_j/(\Delta\tilde{x}_Q)_j\right] & \text{otherwise} \end{cases},\\
\hat{\alpha}_Q^d &= \begin{cases} 1 & \text{if } \Delta\tilde{s}_j > 0,\ \forall j \\ \min_{\Delta\tilde{s}_j<0}\left[-(s_Q)_j/\Delta\tilde{s}_j\right] & \text{otherwise} \end{cases},
\end{aligned}$$

$$\begin{aligned}
\alpha_Q^p &= \max\left\{\tau\hat{\alpha}_a^p, \hat{\alpha}_Q^p - \|\Delta y^a\|\right\},\\
\alpha_Q^p &= \max\left\{\tau\hat{\alpha}_Q^p, \hat{\alpha}_Q^p - \|\Delta y^a\|\right\}.
\end{aligned}$$

Update the variables:

$$\begin{aligned}
\hat{x}_Q &= x_Q + \alpha_Q^p \Delta\tilde{x}, \\
y^+ &= y + \alpha_Q^d \Delta\tilde{y}, \\
s^+ &= s + \alpha_Q^d \Delta\tilde{s}.
\end{aligned}$$

For all $j \in Q$,

$$x_j^+ = \min\left\{\max\left(\min\left\{\|\Delta y^a\|^\nu + \left\|\tilde{x}_-^a\right\|^\nu, \underline{x}\right\}, \hat{x}_j\right), x_{\max}\right\},$$

where

$$\begin{aligned}
\tilde{x}_j^a &= \begin{cases} x_j + \Delta\tilde{x}_j^a & j \in Q \\ 0 & j \in \bar{Q} \end{cases}, \\
\left(\tilde{x}_-^a\right)_j &= \min\left\{\tilde{x}_j^a, 0\right\}.
\end{aligned}$$

Set

$$\mu_Q^+ = \frac{\left(x_Q^+\right)^T s_Q^+}{k}.$$

For all $j \in \bar{Q}$,

$$\begin{aligned}
\hat{x}_j &= \frac{\mu_Q^+}{s_j^+}, \\
x_j^+ &= \min\left\{\hat{x}_j, x_{\max}\right\}.
\end{aligned}$$

The key to the global convergence analysis is the availability of a dual-feasible point at every iteration, the definition of the mixing parameter γ, and the lower bound condition on the primal updates. A local quadratic convergence analysis follows provided the above conditions hold and the bounds $\hat{\alpha}_Q^p - \|\Delta y^a\|$ and $\hat{\alpha}_Q^d - \|\Delta y^a\|$

are imposed on the predictor-corrector step in equation (4.17). We provided a brief outline of the convergence proof for the modified $redPDAS$ algorithm in Chapter 3. The proof for the modified $redMPC$ algorithm follows a similar format. We refer the reader to Winternitz et al. [28] for specific details.

4.5 Summary

In this chapter, we introduced the $redMPC$ algorithm and stated how specific changes to the algorithm provide global and local q-quadratic convergence results. With further research, we hope to prove similar results for the $redMPC$ algorithm. In the next chapter, we test the performance of the $redPDAS$ and $redMPC$ algorithms against their counterparts without constraint reduction.

Chapter 5

Numerical Experiments

5.1 Overview

Algorithms *redPDAS* and *redMPC* were implemented in MATLAB (v.6.5, R13) and run on an Intel(R) Pentium(R) M Processor CPU 1.60GHz Laptop machine with 512 MB of RAM. Each algorithm was tested against the general version of the algorithm (see Sections 3.2 and 4.2) with reduced normal equations,

$$\begin{aligned}\Delta y^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1}\left[b - AS^{-1}Xr_d\right],\\ \Delta s^a &= -r_d - A^T\Delta y^a,\\ \Delta x^a &= -x + S^{-1}X\Delta s^a.\end{aligned}$$

For simplicity, the general version of the algorithms with this slight modification will be referred to as $PDAS$ and MPC. Algorithm *redMPC* was also tested against a modified version of the reduced MPC algorithm (*ipas*35) in Tits et. al. [24]. In our numerical experiments, we simply refer to this algorithm by *ipas*35. The $redMPC$ and *ipas*35 algorithms require the same initial point condition, $(x, s) > 0$, which can easily be accommodated using the algorithm of Section 2.2.2. Unfortunately, the *redPDAS* algorithm could not be tested with the same starting point as the reduced PDAS ($rPDAS$) from [24] since *redPDAS* requires $s > 0$ at each iteration and $rPDAS$ requires $s = c - A^Ty$ but not $s > 0$ at each iteration. Although the combination $s = c - A^T y > 0$ is required for optimality, an initial point which satisfies both conditions simultaneously is difficult to achieve. Therefore, the redPDAS algorithm was tested just with the PDAS algorithm by using the initial point algorithm of Section 2.2.2.

The parameters for *redPDAS* and *redMPC* were chosen as $\tau = .99$ and $u_{bnd} = \hat{k}$ where $\hat{k}$ was selected from 25 linearly spaced vector values (rounded to the nearest integer) ranging from $3m$ to n. The value of $\hat{k}$ is fixed, however k varies from iteration to iteration such that $3m \leq k \leq \hat{k} \leq n$. The stopping criterion was adapted from [17] and is based on the error in the duality gap,

$$\left|c^T x - b^T y\right| / \max\left(c^T x, 1\right) < \lambda,$$

where $\lambda = 10^{-8}$.

5.1.1 Test Problems

The first set of test problems (TAW1 - TAW5) were selected from the test problems used in the code developed by Tits et. al. [24]. The LP data sets (A, b, c) were generated using the MATLAB commands `rand, randn`, and `ones`. The commands `rand(m,n)` and `randn(m,n)` generate an $m \times n$ matrix with uniformly and normally distributed random numbers, respectively. The entries in `rand(m,n)` are chosen from a uniform distribution on the interval (0.0,1.0). The entries in `randn(m,n)` are chosen from a normal distribution with mean zero and variance 1 (i.e $N(0, 1)$). The command `ones(m,n)` creates an $m \times n$ matrix of all ones. Table 5.1 displays the test problem name, a description of the problem, and the MATLAB commands used to define the data (A, b, c). The initial point (x^0, y^0, s^0) was generated as discussed in the previous section, unless otherwise stated in the table. For more information about these test problems, see [24].

Problems	**Description**	**Data**
TAW1	Constraints tangent to unit sphere	A^*, $b =$ `randn(m,1)`, $c =$ `ones(n,1)`, $y^0 = \hat{y}/\text{norm}(\hat{y})$ where $\hat{y} =$ `randn(m,1)`.
TAW2	Random (normal) constraints	A^*, $b =$ `randn(m,1)`, $c = A^T\hat{y} + \hat{s}$, where $\hat{y} =$ `rand(m,1)`, $\hat{s} =$ `rand(n,1)`.
TAW3	Random as in WT-18Jul2003 (previous version of TAW)	$A = 2$ `rand(m,n) - ones(m,n)`, $b = 2$ `rand(m,1) - ones(m,1)`, $c = A^T\hat{y} + \hat{s}$, where $\hat{y} = b, \hat{s} =$ `rand(n,1)`.
TAW4	RandomLP	$A =$ `randn(m,n)`, $b =$ `randn(m,1)`, $c = A^T\hat{y} +$ `rand(n,1)`, where $\hat{y} =$ `randn(m,1)`.
TAW5	SIPND	A^*, $b =$ `randn(m,1)`, $c =$ `ones(n,1)`, $y^0 =$ `zeros(m,1)`.

Table 5.1: Random Problems (TAW1 - TAW5) from Tits et. al. [24]. Problems with constraint matrix A^* generate each column of A with `randn(m,1)` and then normalize to make each column have norm one.

In the second set of test problems (RAND1 - RAND5), the matrix A and the vectors b and c were randomly generated, for some specified m and n, so that both the primal and dual problems produced nondegenerate solutions at each iteration and terminated with an optimal solution in a finite number of iterations. The entries in A were randomly generated from a normal distribution with mean 0 and variance 1. Each data value in A was then multiplied by 2 and rounded to the next integer. The entries in the vector b were randomly generated from a uniform distribution on the interval (0,1). Each entry in b was then multiplied by 100 and rounded to the next integer. The vector c is the vector of all ones with its first entry multiplied by -1. Table 5.2 shows the test problem name, the dimension of each problem, and its optimal solution. The optimal solution was determined by MATLAB's linear programming solver, LINPROG.

Problems	**Dimensions**	**Optimal Solution**
RAND1	$m = 5, n = 100$	18.28000000
RAND2	$m = 12, n = 500$	29.15542400
RAND3	$m = 24, n = 780$	34.05845050
RAND4	$m = 58, n = 1004$	58.61768707
RAND5	$m = 75, n = 2016$	61.42186535

Table 5.2: RandomTest Problems (RAND1 - RAND5) with specified m and n and known optimal solution

The third set of test problems comes from the Netlib test problems for LPs [2]. A search for problems that satisfied the criterion of $n \gg m$ led to approximately

12 problems. However, ensuring A_Q had full rank at each iteration reduced the set of 12 problems to 3. These three problems come from the SCSD series. Table 5.3 shows the name of the test problem, the dimension of each problem, and its optimal solution.

Problems	**Dimensions**	**Optimal Solution**
SCSD1	$m = 77, n = 760$	8.66666666
SCSD6	$m = 147, n = 1350$	50.50000000
SCSD8	$m = 397, n = 2750$	904.99999999

Table 5.3: Netlib Problems from [2] with specified m and n and known optimal solution

5.2 *redPDAS* Experiments

The selection of the most promising dual constraints in the working set Q at each iteration is based on a subroutine that selects the ratios x_j/s_j greater than $C \cdot \max(x_j/s_j)$ where C is an experimental constant. Figure 5.1 shows the average time (in seconds) to solve 50 randomly generated problems over varying values of C. This experiment was conducted using the test problem TAW4 with $m = 50$, $n = 20000$, and $|Q| \in [3m, u_{bnd}]$ where u_{bnd} is fixed to n.

Figure 5.1 shows that the average time to solve 50 randomly generated problems for $PDAS$ is more than 2 times that of *redPDAS* for C values between 10^{-16}

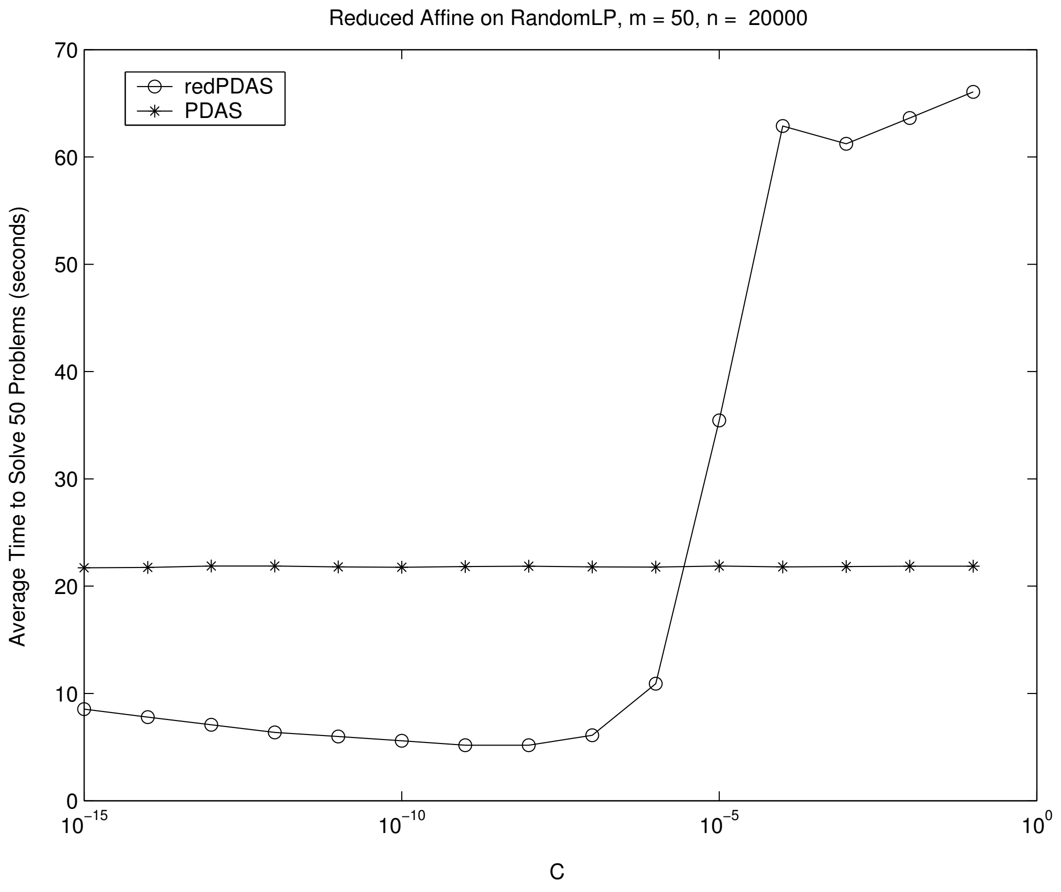

Figure 5.1: Performance of the *redPDAS* algorithm against the *PDAS* on test problem TAW4 with $m = 50, n = 20000$, and $|Q| \in [3m, u_{bnd}]$ where u_{bnd} is fixed to n. The average time (in seconds) to solve 50 randomly generated problems from TAW4 is shown over varying values of C ranging from 10^{-16} to 10^{-1}.

and 10^{-6}. When $C > 10^{-5}$, the time for the *redPDAS* algorithm to solve an LP nearly triples relative to the *PDAS* algorithm. This is most likely caused by an insufficient number of dual constraints in the working set at later iterations. The subroutine for determining Q guarantees the set contains at least m indices at each iteration and that its size is between $l_{bnd} = \min(u_{bnd}, 3m)$ and $u_{bnd} = \hat{k}$ where $\hat{k}$ is an element from 25 linearly spaced vector values (rounded to the nearest integer) ranging from $3m$ to n. However, as the algorithm approaches the optimal solution, m x_j/s_j ratios tend to infinity since $s_j \to 0$ for all j in the optimal set. If C is large enough, the selection of ratios x_j/s_j greater than $C \cdot \max(x_j/s_j)$ (i.e. $j \in Q$) may be too small at early iterations. If too few (or zero) ratios are chosen, the *redPDAS* algorithm defines Q to contain the indices j from $n - l_{bnd} + 1$ to n of the largest ratios where $l_{bnd} = \min(3m, u_{bnd})$. Since $u_{bnd} = n$, $m = 50$, and $n = 20000$, the number of dual constraints in the working set may be as small as 149 (out of 20000) in early iterations. This can cause the algorithm to take a large number of iterations before reaching the optimal solution. According to Figure 5.1, the *redPDAS* algorithm performs best against the *PDAS* algorithm when C is set 10^{-8}.

In the experiments that follow, the *redPDAS* and *PDAS* algorithms solve 25 randomly generated problems from each test set (TAW, RAND, and SCSD) over various values of $3m < \hat{k} \leq n$ with C fixed to 10^{-8}. A comparison of the total CPU time, the CPU time to form and solve the normal matrix, and the mean number of iterations is shown. Each one is plotted against various $\hat{k}/n$ ratios. In many cases, the graphs are missing data values for small $\hat{k}/n$ values. These missing values are caused by one of two scenarios: (i) the algorithm fails to terminate in a "reasonable" amount of time (maximum time alloted to solve an LP is set to 10^6 seconds) or (ii) MATLAB generates NaN (not a number) values for the solution as a result of the normal matrix becoming numerically singular.

We begin by examining the experiments based on the first set of test problems (TAW1 - TAW5).

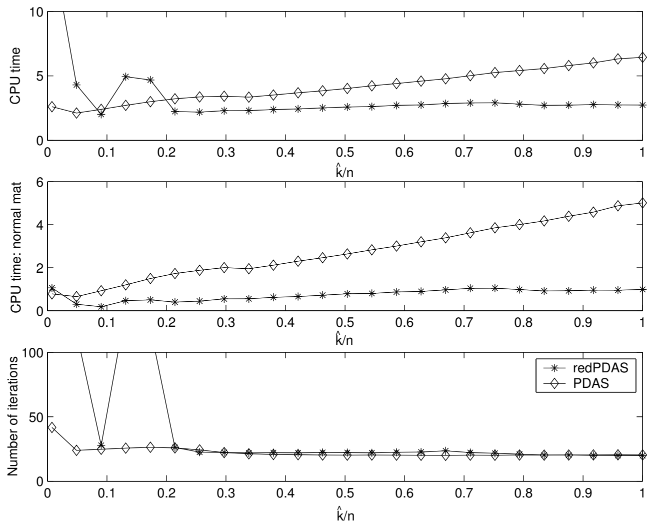

Figure 5.2: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from TAW1 (constraints tangent to the unit sphere) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

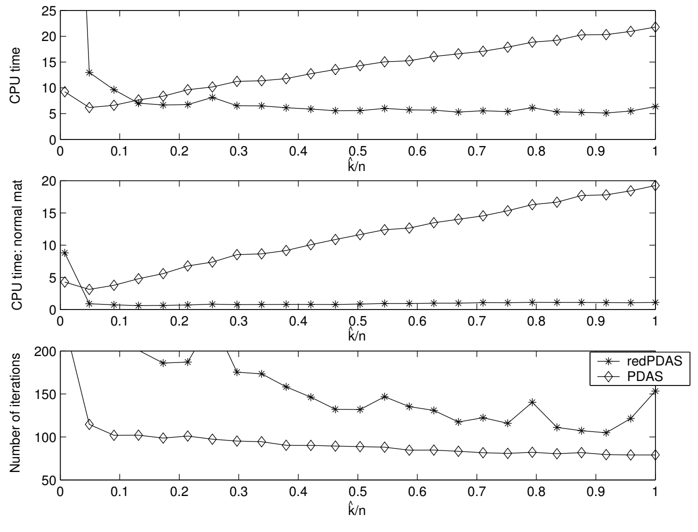

Figure 5.3: Comparison of the *redPDAS* algorithm versus *PDAS* algorithm using 25 randomly generated test problems from TAW2 (random [normal] constraints) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

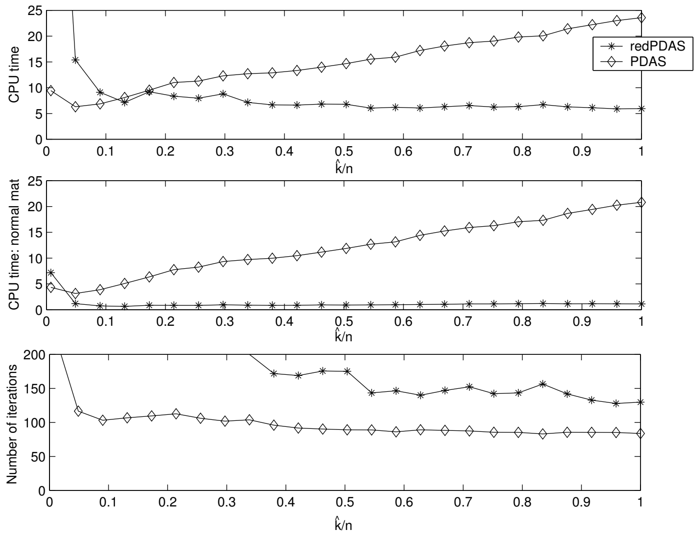

Figure 5.4: Comparison of the *redPDAS* algorithm versus *PDAS* algorithm using 25 randomly generated test problems from TAW3 (Random as in WT-18July2003) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

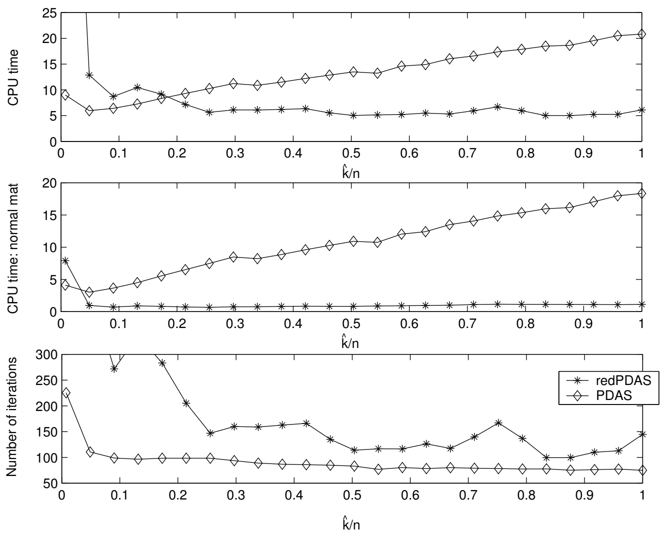

Figure 5.5: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from TAW4 (RandomLP) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

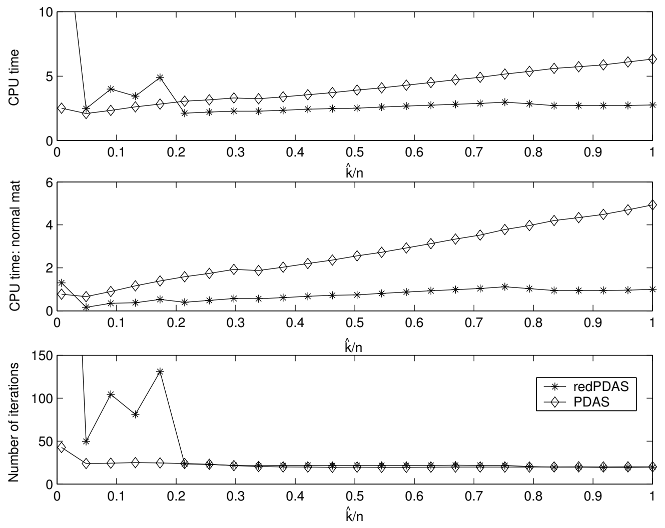

Figure 5.6: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from TAW5 (SIPND) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

For the first test set (TAW), the $PDAS$ algorithm displayed (in general) an increase in CPU time and a constant number of iterations to solve an LP over increasing values of the ratio $\hat{k}/n$. On the other hand, the $redPDAS$ algorithm remained (in general) constant in CPU time while the number of iterations fluctuated and often times exceeded the general algorithm. The time difference between the algorithms became increasing large as $\hat{k}/n$ increased, with the $redPDAS$ algorithm outperforming $PDAS$ for every test problem with $\hat{k}/n > .2$.

We now examine the experiments based on the second set of test problems (RAND1 - RAND5).

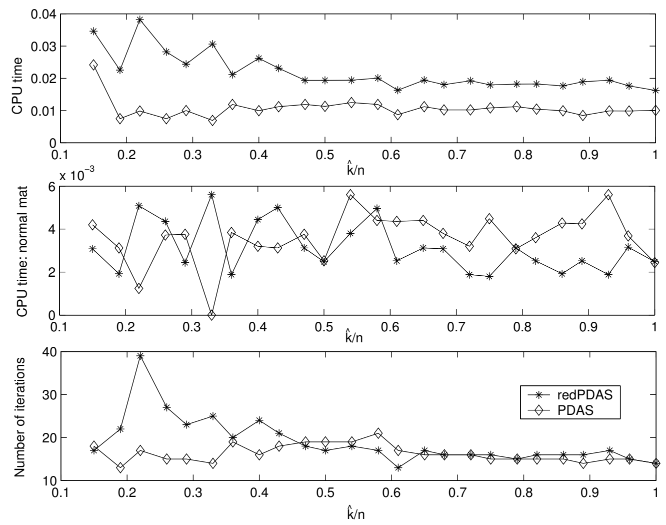

Figure 5.7: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from RAND1 with $m = 5$, $n = 100$, and $u_{bnd} = \hat{k}$ where $15 \leq \hat{k} \leq 100$.

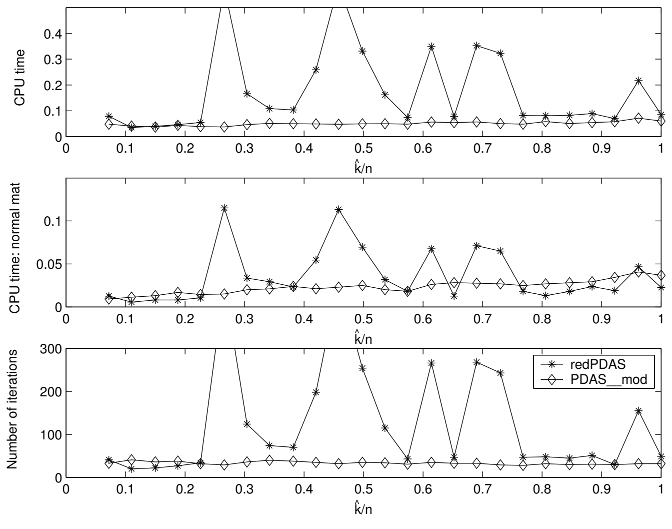

Figure 5.8: Comparison of the *redPDAS* algorithm versus *PDAS* algorithm using 25 randomly generated test problems from RAND2 with $m = 12$, $n = 500$, and $u_{bnd} = \hat{k}$ where $36 \leq \hat{k} \leq 500$.

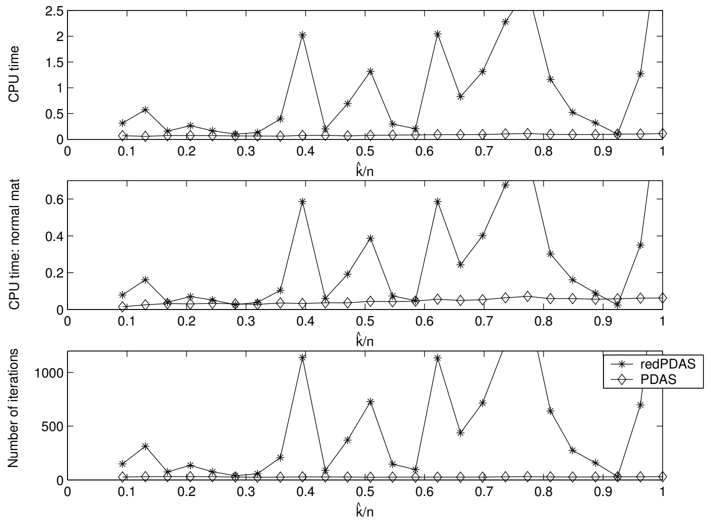

Figure 5.9: Comparison of the *redPDAS* algorithm versus *PDAS* algorithm using 25 randomly generated test problems from RAND3 with $m = 24$, $n = 780$, and $u_{bnd} = \hat{k}$ where $72 \leq \hat{k} \leq 780$.

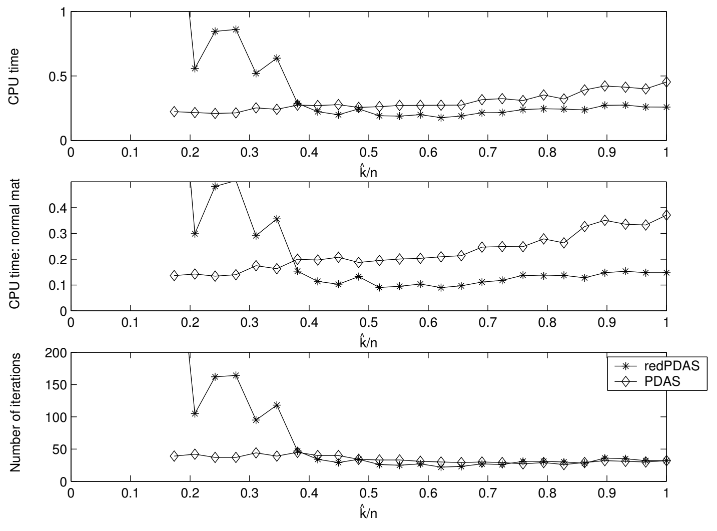

Figure 5.10: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from RAND4 with $m = 58$, $n = 1004$, and $u_{bnd} = \hat{k}$ where $174 \leq \hat{k} \leq 1004$.

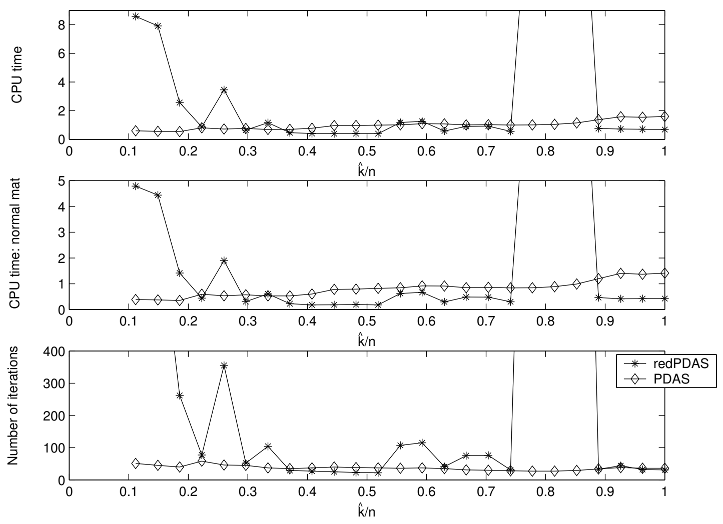

Figure 5.11: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using 25 randomly generated test problems from RAND5 with $m = 75$, $n = 2016$, and $u_{bnd} = \hat{k}$ where $225 \leq \hat{k} \leq 2016$.

The performance of the algorithms on the second test set was quite different. The graphs show missing data values for $\hat{k}/n < .15$. For the remaining $\hat{k}/n$ values, the $PDAS$ algorithm was generally constant in CPU time and number of iterations. The $redPDAS$ displayed a very erratic behavior on almost all of the test sets in this category. Experiments using data sets RAND4 and RAND5 show more favorably for the $redPDAS$ algorithm than those using data sets RAND1 - RAND3, possibly due to $m \ll n$ in the first case.

Finally, we examine the experiments based on the Netlib test problems (SCSD1, SCSD6, and SCSD8).

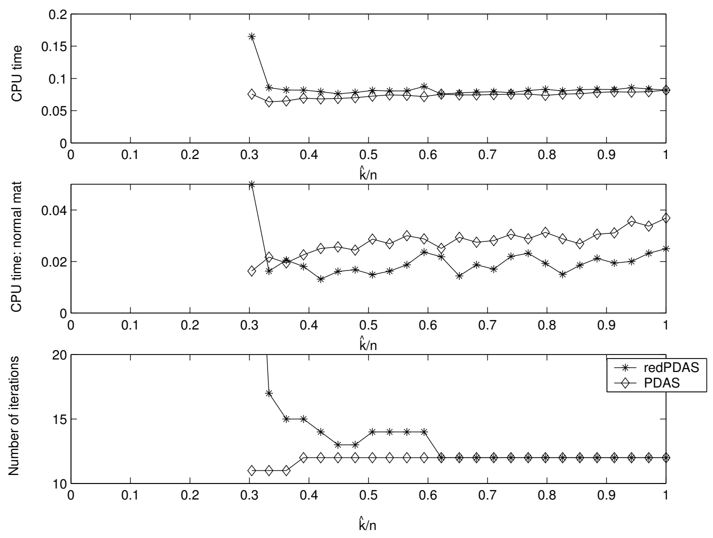

Figure 5.12: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using the Netlib test problem SCSD1 with $m = 77$, $n = 760$, and $u_{bnd} = \hat{k}$ where $231 \leq \hat{k} \leq 760$.

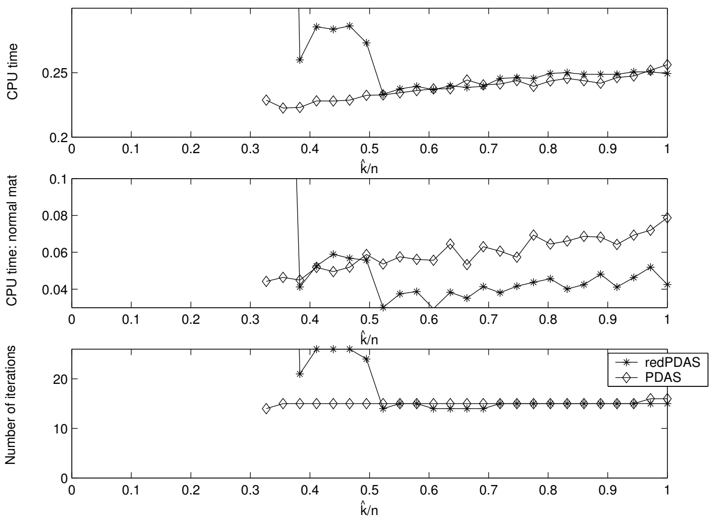

Figure 5.13: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using the Netlib test problem SCSD6 with $m = 147$, $n = 1350$, and $u_{bnd} = \hat{k}$ where $441 \leq \hat{k} \leq 1350$.

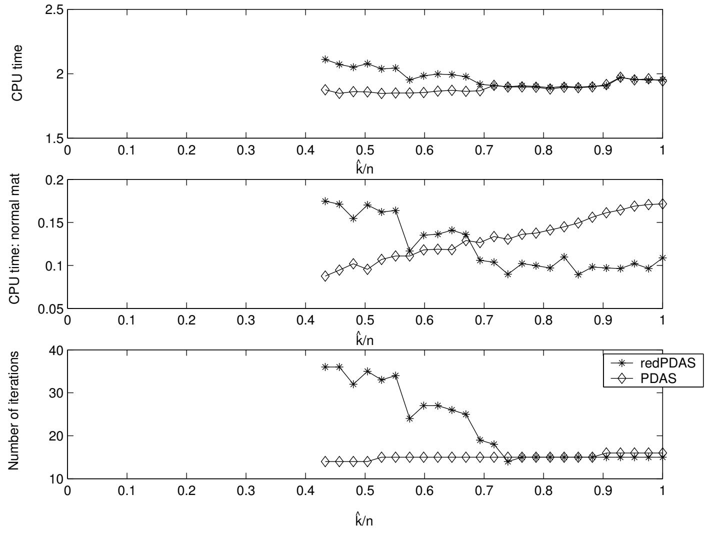

Figure 5.14: Comparison of the $redPDAS$ algorithm versus $PDAS$ algorithm using the Netlib test problem SCSD8 with $m = 397$, $n = 2750$, and $u_{bnd} = \hat{k}$ where $1191 \leq \hat{k} \leq 2750$.

For the third test set (Netlib problems), a large portion of the data is missing for the early $\hat{k}/n$ values. The $PDAS$ algorithm remains consistent in its performance on CPU time and number of iterations on the SCSD problems. In addition, it outperforms the $redPDAS$ algorithm for some $\hat{k}/n$ values between 0.3 and 0.7. However, the $redPDAS$ and $PDAS$ algorithms performance is almost identical in CPU time and number of iterations as $\hat{k}/n \to 1$.

In the next section, we examine experiments based on Mehrotra's Predictor-Corrector Method.

5.3 *redMPC* Experiments

The $redMPC$ experiments test the performance of the $redMPC$ algorithm against the reduced MPC algorithm in Tits et. al [24] ($ipas35$) and a general MPC algorithm with reduced normal equations (MPC). Figure (5.15) shows the average time (in seconds) to solve 50 randomly generated problems over varying values of C using the test problem TAW4 with $m = 50$, $n = 20000$, and $|Q| \in [3m, u_{bnd}]$ where u_{bnd} is fixed to n.

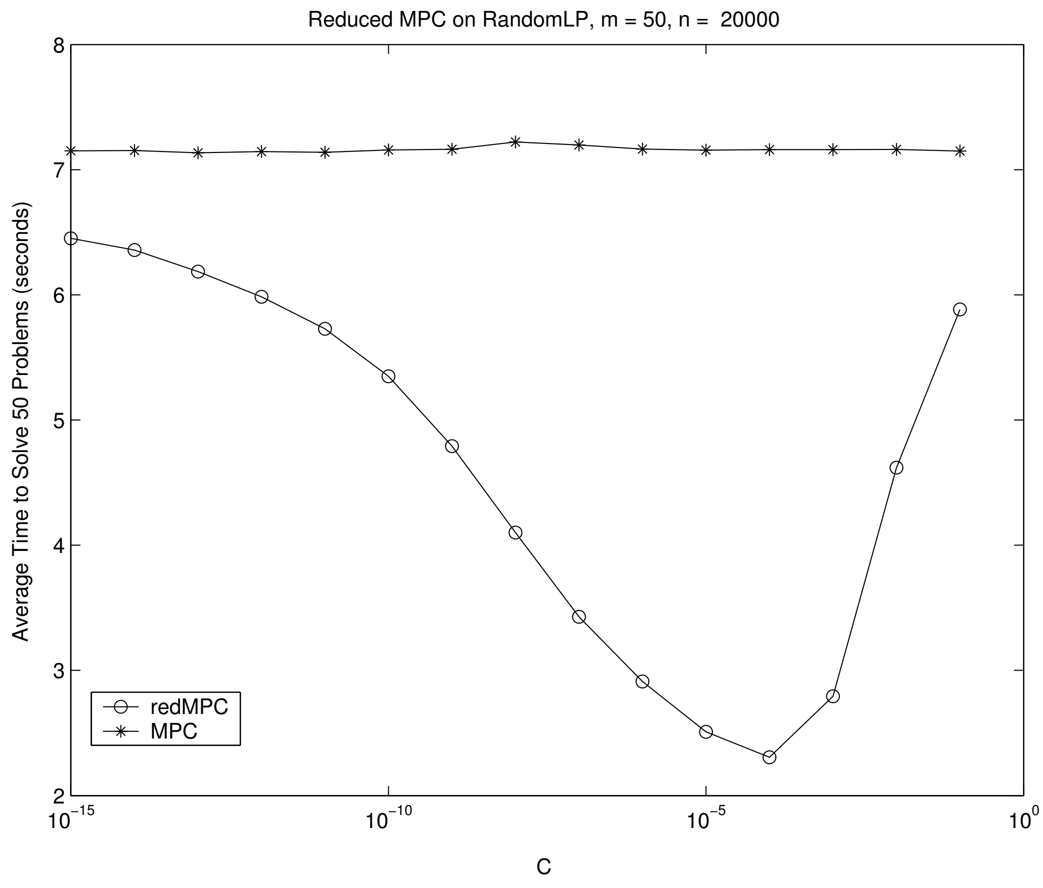

Figure 5.15: Performance of the *redMPC* algorithm against the *MPC* using test problem TAW4 with $m = 50, n = 20000$, $|Q| \in [3m, u_{bnd}]$ where u_{bnd} is fixed to n. The average time (in seconds) to solve 50 randomly generated problems from TAW4 is shown over varying values of C ranging from 10^{-16} to 10^{-1}.

The *redMPC* outperformed the *MPC* algorithm for all values of C between 10^{-16} and 10^{-1}. In the experiments that follow, the *redMPC*, *MPC*, and *ipas*35 algorithms solve 25 randomly generated problems from each test set (TAW, RAND, and SCSD) over various values of $3m < \hat{k} \leq n$ with C fixed to 10^{-4}. A comparison of the total CPU time, the CPU time to form and solve the normal matrix, and the mean number of iterations is shown. Each one is plotted against various $\hat{k}/n$ ratios.

We begin by examining the experiments based on the first set of test problems (TAW1 - TAW5).

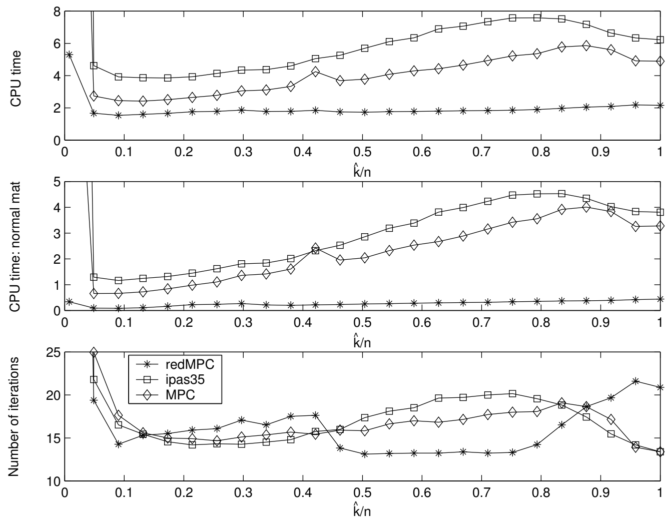

Figure 5.16: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from TAW1 (constraints tangent to the unit sphere) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

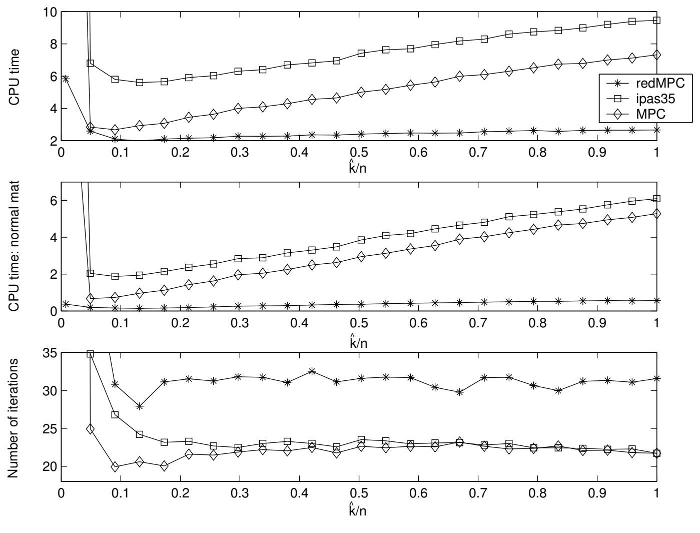

Figure 5.17: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from TAW2 (random [normal] constraints) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

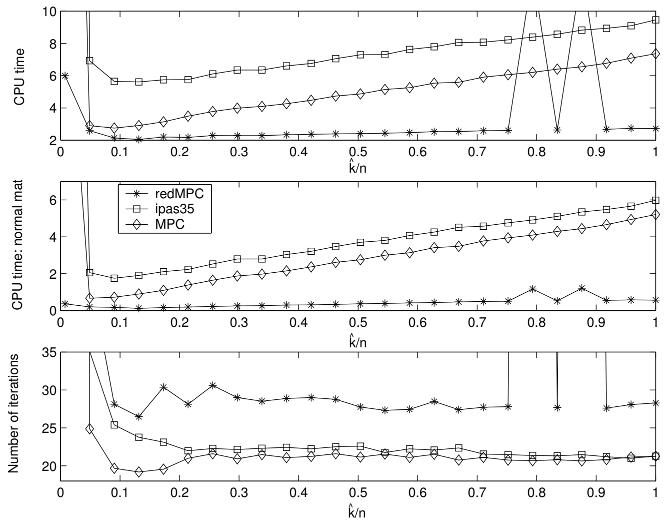

Figure 5.18: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from TAW3 (Random as in WT-18July2003) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

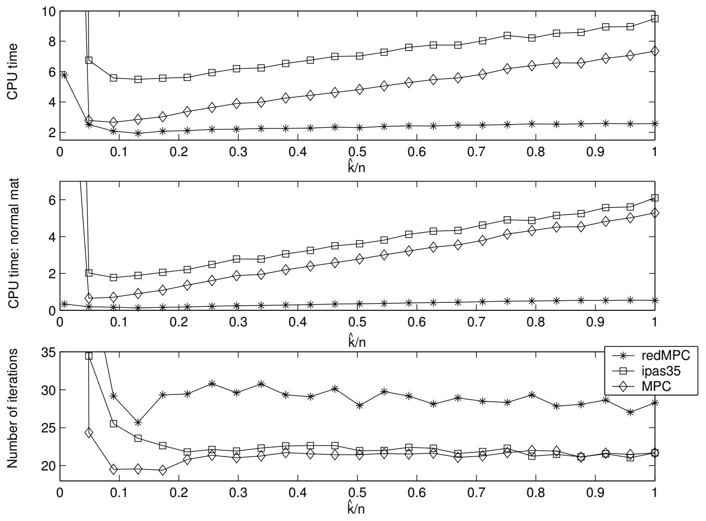

Figure 5.19: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from TAW4 (RandomLP) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

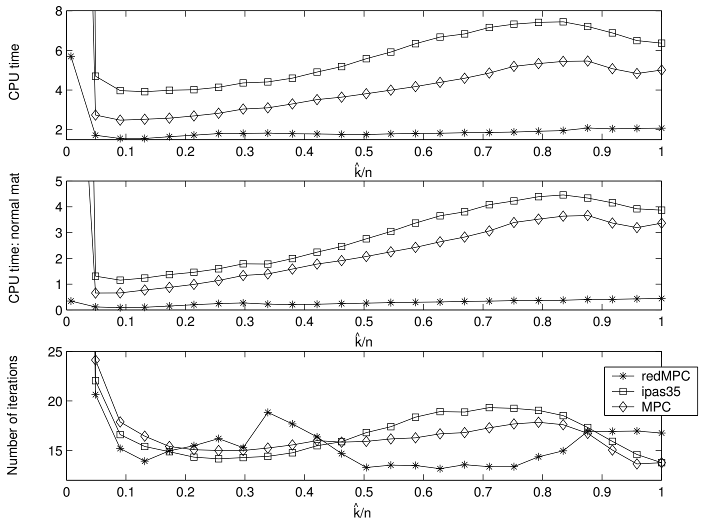

Figure 5.20: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from TAW5 (SIPND) with $m = 50$, $n = 20000$, and $u_{bnd} = \hat{k}$ where $150 \leq \hat{k} \leq 20000$.

For problem sets TAW2-TAW4, the MPC and $ipas35$ algorithms displayed (in general) an increase in CPU time and a constant number of iterations to solve an LP over increasing values of the ratio $\hat{k}/n$. The $redMPC$ algorithm, however, remained (in general) constant in CPU time while the number of iterations fluctuated and exceeded the competition. In problem sets TAW1 and TAW5, the MPC and $ipas35$ algorithms show an increase in CPU time as $\hat{k}/n$ increases from .1 to .8, then a small decrease in time for $\hat{k}/n$ values between .8 to 1. The number of iterations to solve an LP fluctuates for all of the algorithms. In all of the problem sets (TAW1 - TAW5), the time difference between the MPC and $ipas35$ algorithms and the $redMPC$ algorithm became increasing large as $\hat{k}/n$ increased, with the $redMPC$ algorithm outperforming the other algorithms in every test set for all $\hat{k}/n > .2$.

We now examine the MPC experiments based on the second set of test problems (RAND1 - RAND5).

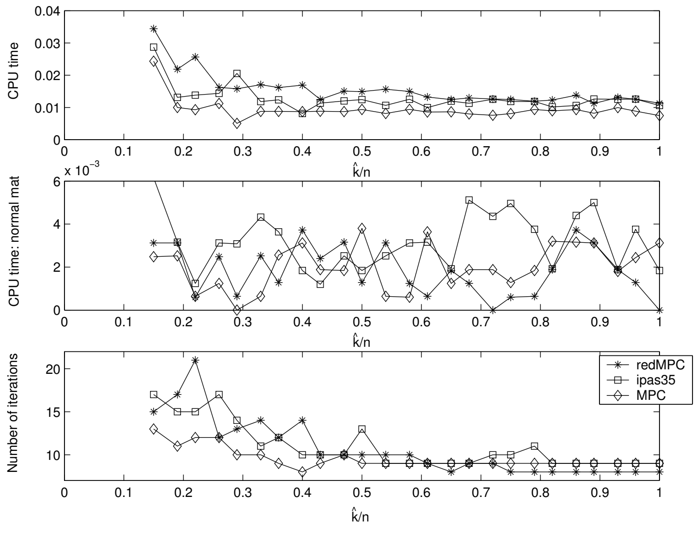

Figure 5.21: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from RAND1 with $m = 5$, $n = 100$, and $u_{bnd} = \hat{k}$ where $15 \leq \hat{k} \leq 100$.

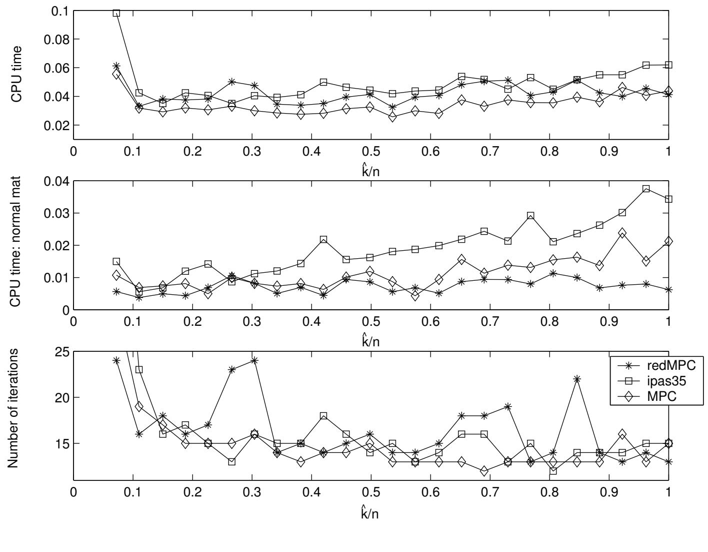

Figure 5.22: Comparison of the *redMPC* algorithm versus the *MPC* and *ipas*35 algorithms using 25 randomly generated test problems from RAND2 with $m = 12$, $n = 500$, and $u_{bnd} = \hat{k}$ where $36 \leq \hat{k} \leq 500$.

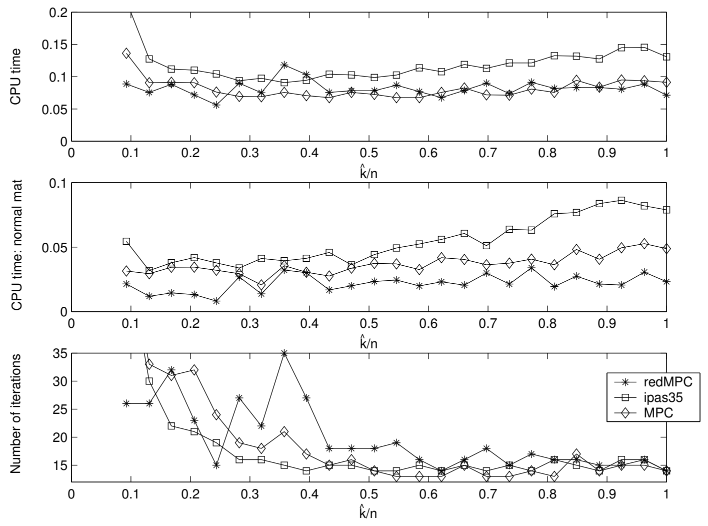

Figure 5.23: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using 25 randomly generated test problems from RAND3 with $m = 24$, $n = 780$, and $u_{bnd} = \hat{k}$ where $72 \leq \hat{k} \leq 780$.

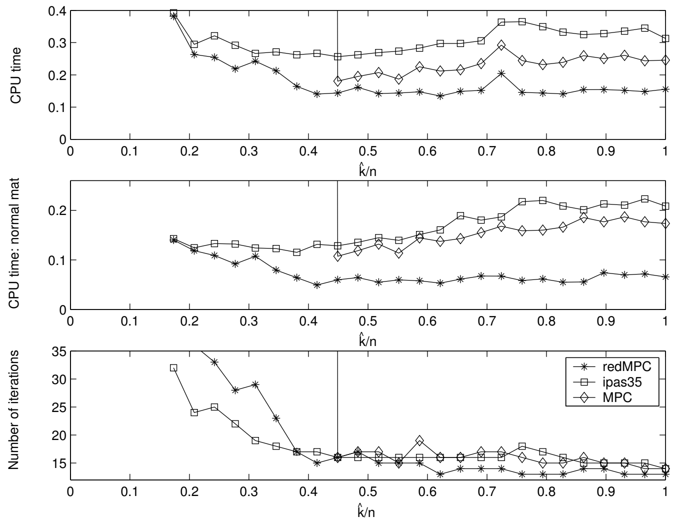

Figure 5.24: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using 25 randomly generated test problems from RAND4 with $m = 58$, $n = 1004$, and $u_{bnd} = \hat{k}$ where $174 \leq \hat{k} \leq 1004$.

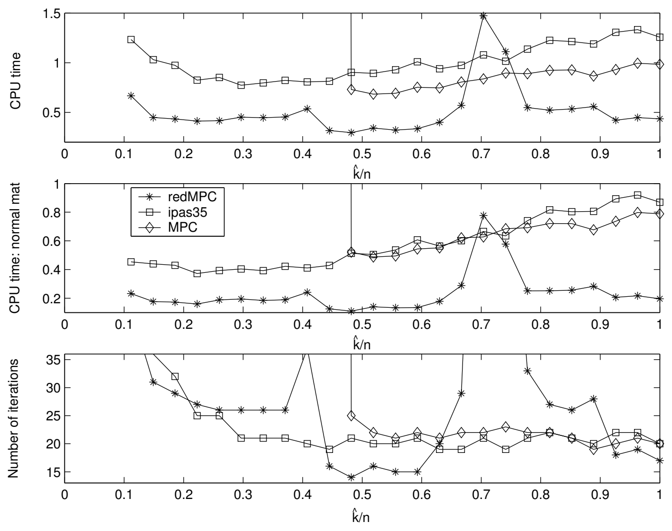

Figure 5.25: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using 25 randomly generated test problems from RAND5 with $m = 75$, $n = 2016$, and $u_{bnd} = \hat{k}$ where $225 \leq \hat{k} \leq 2016$.

The performance of the algorithms on the second test set was quite different from the performance on the TAW test set. It can be observed that the graphs show missing data values for $\hat{k}/n < .15$. For the remaining $\hat{k}/n$ values, the performance of the algorithms varied. Unlike the $redPDAS$ algorithm, the $redMPC$ algorithm generally solved an LP in less time than the MPC and $ipas35$ algorithms. In fact, as the problem size increased, the total CPU time between the MPC and $ipas35$ algorithms and the $redMPC$ algorithm increased. With the exception of the RAND1 problem set, the same trend occurred when determining the CPU time to form and solve the normal matrix. In all of the problem sets, the algorithms generally solved an LP in the same number of iterations for $\hat{k}/n$ close to 1. For all other $\hat{k}/n$ values, the number of iterations to solve an LP varied for each algorithm.

Lastly, we examine the experiments based on the Netlib test problems (SCSD1, SCSD6, and SCSD8).

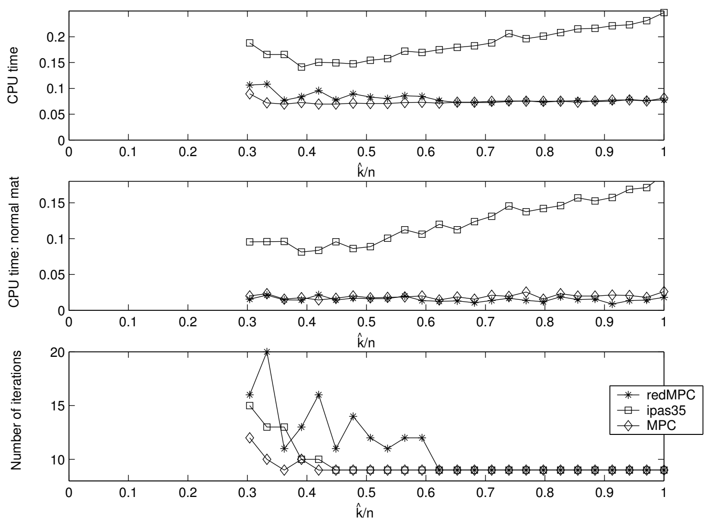

Figure 5.26: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using the Netlib test problem SCSD1 with $m = 77$, $n = 760$, and $u_{bnd} = \hat{k}$ where $231 \leq \hat{k} \leq 760$.

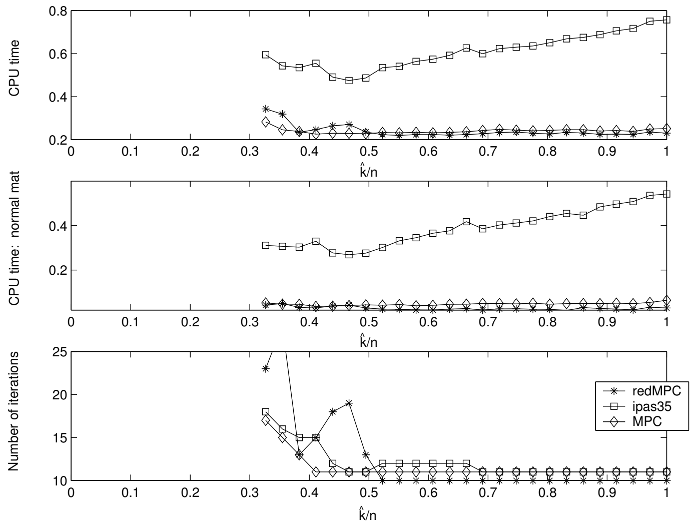

Figure 5.27: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using the Netlib test problem SCSD6 with $m = 147$, $n = 1350$, and $u_{bnd} = \hat{k}$ where $441 \leq \hat{k} \leq 1350$.

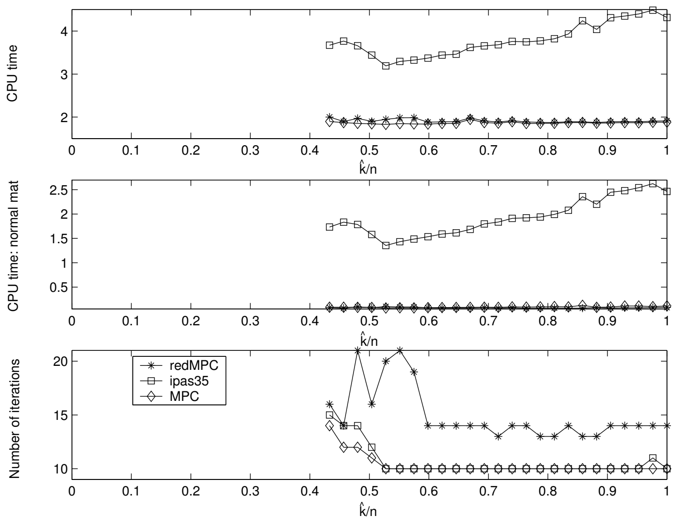

Figure 5.28: Comparison of the $redMPC$ algorithm versus the MPC and $ipas35$ algorithms using the Netlib test problem SCSD8 with $m = 397$, $n = 2750$, and $u_{bnd} = \hat{k}$ where $1197 \leq \hat{k} \leq 2750$.

For the third test set (Netlib problems), a large portion of the data is missing for the early $\hat{k}/n$ values. The *redMPC* and *MPC* outperform *ipas*35 in CPU time. As $\hat{k}/n$ increases, the time to solve an LP with *ipas*35 increases whereas the time to solve an LP with the other algorithms is small and fixed. The number of iterations to solve test problem SCSD8 for the *redMPC* algorithm is somewhat larger than for the other algorithms. For test problems SCSD1 and SCSD6, the number of iterations to solve is almost identical as $\hat{k}/n \to 1$.

5.4 Discussion

The experiments in the previous sections demonstrate that the *redPDAS* and *redMPC*, in general, outperform their general counterparts in terms of CPU time. The *redMPC* and *MPC* algorithms also perform well against *ipas*35. Furthermore, with the exception of a few cases (i.e. the small values of $\hat{k}/n$ where the algorithm(s) did not display information), the algorithms in this chapter outperformed MATLAB's LP solver, LINPROG, in total CPU time. In this section, we will give reason or speculation as to a few of the outcomes.

Figures (5.1) and (5.15) show the average time (in seconds) to solve 50 randomly generated problems over varying values of C. These experiments were conducted using the test problem TAW4 with $m = 50$, $n = 20000$, and u_{bnd} fixed to n. It was shown that the "reduced" algorithms performed exceptionally well against their general counterparts. We also performed these experiments on other test problems with $m \ll n$. Although there was some fluctuation in the value of C, it remained within 1/100 of the value we found in each case. Therefore, we felt confident the "reduced" algorithms would perform well on other test problems with $m \ll n$ based on our specific choice of C determined by one of the problem sets used in this thesis, TAW4.

The *redPDAS* and *redMPC* algorithms (in general) displayed small CPU times and a large number of iterations over the various $\hat{k}/n$ values. The small CPU times can be attributed to the initial point calculation, the formation of "reduced" matrices and vectors, and temporary matrices/vectors defined for A_Q, D_Q, and the dual residual, $(r_d)_Q$. On the other hand, the large number of iterations are the result of the update strategy used for the dual residual and solution. Since a small number of constraints ($\ll n$) are chosen at each iteration, the algorithms have the potential of cycling through a large number of solutions before reaching the optimal solution.

MATLAB's LP Interior-Point Solver, LIPSOL, is capable of solving a wide range of problems including large-scale LPs, see [33], [34]. On our test problems (i.e. problems with $m \ll n$), the "reduced" algorithms outperformed LIPSOL in CPU time. In experimentation, we discovered by first converting a sparse matrix A to a full matrix and then using a QR-factorization of A to solve for the initial point, the time to formulate the initial point is significantly reduced. Prior to this conversion, approximately 80% of the CPU time was devoted to calculating the initial point, thus causing the time to solve the "reduced" algorithms to exceed the time for LIPSOL to solve the same LP. All of the algorithms (including LIPSOL), calculate the normal matrix using MATLAB's Cholesky-Infinity factorization (cholinc function). This is necessary to preserve full rank and calculate efficient search directions.

5.5 Summary

In this chapter, we have demonstrated that algorithms $redPDAS$ and $redMPC$ are more effective than their general counterparts in solving LPs with $m \ll n$. This is clearly shown by the results displayed using the TAW problem sets (with $m = 50$ and $n = 20000$). In the second and third test sets, the difference between the m and n values was not as large. As a result, the general algorithms performed just as well and in some cases even better than the "reduced" algorithms. The $ipas35$ algorithm was least effective in solving LPs compared to the $redMPC$ and MPC algorithms.

Chapter 6

Conclusions and Further Study

We have presented the $redPDAS$ algorithm and the $redMPC$ algorithm, new column generation algorithms for solving linear programming problems (LPs). Experiments conducted on these algorithms demonstrate that they are, in general, more effective for solving LPs with $m \ll n$ than their general counterparts ($PDAS$ and MPC). Although there is no known convergence proof at this time for the $redPDAS$ and $redMPC$ algorithms, we believe our column generation strategy will prove to be beneficial in solving other optimization problems as well.

In this dissertation, we demonstrated that by reducing the $m \times m$ matrix AD^2A^T, an essential component for solving for the search direction on large-scale LPs, we can reduce the amount of computations per iteration and significantly reduce the time to solve a LP problem. This was illustrated by the results of the experiments conducted using the random problem sets from TAW. The time difference between the "reduced" and general algorithms became increasing large as k/n increased, with the "reduced" algorithms outperforming their general counterparts on every test problem from this set with $k/n > .2$. Although the total number of iterations to solve the "reduced" LP often exceeded that of the general problem, the computational cost per iteration of the "reduced" problem was significantly lower due to its size. The "reduced" algorithms were not as effective in reducing solution time for the other problem sets. In these problem sets, the difference between m and n was not significantly large. As a result, the general algorithms performed just as well and in some cases even better than the "reduced" algorithms.

Based on the results of the experiments conducted using the random problem sets with $m \ll n$, we believe our column generation scheme can be applied to other algorithms with similar success. One algorithm we have begun to investigate is Potra's algorithm [22]. Potra's algorithm is of particular interest because he allows for a primal-dual infeasible starting point and proves his algorithm has polynomial complexity under certain conditions. Potra uses a three-step method to improve feasibility, optimality, and centrality of the iterates. He combines the two steps of Mizuno-Todd-Ye [18] with a third to achieve feasibility and optimality at the same rate. We developed a "reduced" version of Potra's algorithm called *redPC*, presented in Appendix B. The *redPC* algorithm also uses this three-step method, however feasibility and optimality may not necessarily be achieved at the same rate. Despite this, we aim to show global convergence and investigate the complexity of our algorithm.

In addition to further investigating the *redPC* algorithm, there are computational issues that can be researched in all of the "reduced" algorithms. One topic of research is the method in which the set Q is chosen. We chose Q based on the $k \in [m, n]$ largest x_j/s_j ratios at each iteration. Tits et al. [24], however, selects a fixed index set Q associated with the most "promising" dual constraints based on the smallest slack values, s_j, at each iteration. Dantzig and Ye [6] start with m constraints and consider building up the constraint set based on the index set associated with all $s_j < 2^{-\epsilon}$ at each iteration. There are a number of ways to select the index set Q and determining if one technique saves time and computational effort over others is worth investigating. Another topic of investigation is the update strategy used within our "reduced" algorithms. Our update strategy is based on the amount of progress we are making towards satisfying dual feasibility. However, Potra's algorithm uses an update strategy based on improving feasibility, optimality, and centrality of the iterates at the same rate. Experimentation on the *redPC* algorithm may show a reduction in the total number of iterations using this update concept. One other focus for research is determining a "good" initial point. The initial point we used in the *redPDAS* and *redMPC* algorithms is the same initial point used in Mehrotra's algorithm [17]. However, there are many papers in the field of interior point methods that concentrate on starting-point strategies (see [7] or [14]) Our algorithms perform quite well overall against standard algorithms and we may be able to improve our results by using a better starting point.

Finally, the use of interior-point methods in column generation settings has been extended to applications in semi-definite programming problems, network design problems, and second order cone programming problems, to name a few. We believe our strategy will be very beneficial for solving other LPs and eventually for solving more general problems.

Appendix A

Topics in Numerical Optimization

A.1 Steepest-Descent Method

Consider the n-dimensional problem

$$\min_{x \in R^n} \{f(x) : x \in S\} \tag{A.1}$$

where S is the set of feasible points defined by a set of constraints (or $\Re^n$). If (A.1) is optimal at the current solution x, then x solves the problem. Otherwise, assume x is feasible and let $x^+ = x + \alpha\Delta x$ be the updated solution to (A.1) with search direction Δx and step length $\alpha > 0$. Using only the first two terms of the Taylor's series, we can approximate $f(x^+)$ by

$$\begin{aligned} f(x^+) &= f(x + \alpha\Delta x), \\ &\approx f(x) + \alpha\Delta x^T \nabla f(x) \end{aligned}$$

where $\nabla f(x)$ is the gradient of f at x. Therefore, for small α, $f(x + \alpha\Delta x) < f(x)$ provided $\Delta x^T \nabla f(x) < 0$. The direction Δx that provides the greatest descent per unit step is given by $\Delta x = -\nabla f(x)$, hence the name *steepest-descent*. A step length $\alpha > 0$ is then computed to determine x^+ so that $x^+ \in S$.

In the case of a linear program

$$\min_{x \in \Re^n} \left\{c^T x : Ax = b,\ x \geq 0\right\}, \tag{A.2}$$

where A is an $m \times n$ matrix of full row rank, the steepest descent method computes $\Delta x = -c$ as the search direction if the current solution is not optimal. Assuming the current solution x is feasible, the updated solution x^+ is determined to maintain feasibility (i.e. $x^+ \in S = \{Ax = b,\ x \geq 0\}$). Therefore we require

$$Ax^+ = Ax + \alpha A \Delta x = b \tag{A.3}$$

and

$$x^+ = x + \alpha \Delta x > 0. \tag{A.4}$$

The equations in (A.3) imply that we need $A\Delta x = 0$ since $x \in S$ and $\alpha > 0$. That is, Δx must lie in the null space of A. By premultiplying $\Delta x = -c$ by the $n \times n$ matrix

$$P_A = I - A^T \left(AA^T\right)^{-1} A$$

we have a *projected steepest descent* direction (i.e. $\Delta x = -P_A c$) which simultaneously decreases the objective function value and satisfies $A\Delta x = 0$. The matrix P_A has the effect of projecting any nonzero vector in $\Re^n$ onto the null space of A. Further discussion of this topic is given in the next section.

The inequalities in (A.4) are satisfied by computing an appropriate step length $\alpha > 0$ along the search direction $\Delta x = -P_A c$.

A.2 Orthogonal Projection Matrix

The $n \times n$ matrix

$$P_A = I - A^T \left(AA^T\right)^{-1} A,$$

which was presented in the previous section, is also termed an *orthogonal projection matrix*. Its name comes from the fact that premultiplying any n-dimensional vector x by P_A produces an orthogonal projection of x onto the null space of A. See Figure A.1.

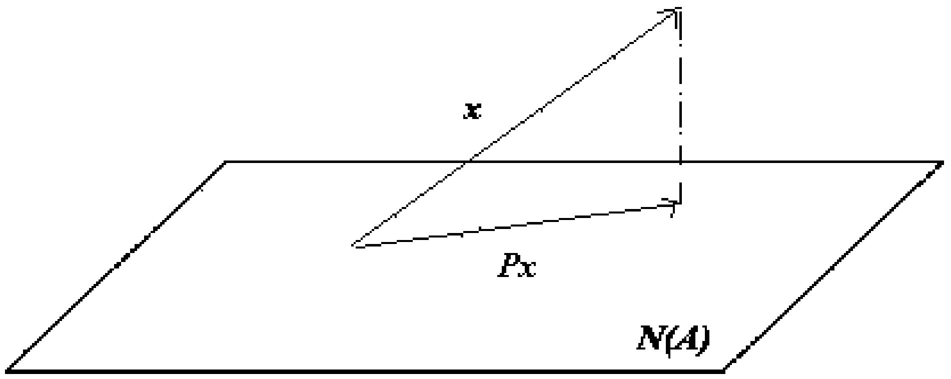

Figure A.1: A n-dimensional vector x is projected onto the null space of A [denoted $N(A)$] by the orthogonal projection matrix $P = I - A^T(AA^T)^{-1}A$.

The orthogonal projection matrix is symmetric (i.e. $P_A^T = P_A$) and idempotent (i.e. $P_A^2 = P_A$). In addition, it can be formed from any $m \times n$ matrix of full row rank. For example, the matrix

$$P_{AD} = I - DA^T \left(AD^2A^T\right)^{-1} AD,$$

is an orthogonal projection matrix into the null space of AD where D is a diagonal matrix.

Appendix B

The *redPC* Algorithm

Potra's three-step method involves solving three linear systems to improve feasibility, optimality, and centrality of the iterates. Our method follows Potra's method by solving three "reduced" linear systems in an attempt to reach the optimal solution for the primal-dual pair. In solving the first linear system, we aim to improve optimality while keeping feasibility the same. In the second linear system, we strive to improve feasibility while keeping optimality the same. Finally, in the third linear system, we aim to "centralize" the iterates for improved movement within the next iteration. Our "reduced" version is similar to the *redPDAS* and *redMPC* algorithms in that we consider only those columns (components) of a matrix (vector) associated with Q.

B.1 Background

Let $r_p = Ax - b$ and $r_d = A^T y + s - c$ be the primal and dual residuals, respectively. The search direction computed to improve optimality or decrease μ is given by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^a \\ \Delta y^a \\ \Delta s^a \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ XSe \end{bmatrix}. \tag{B.1}$$

The expressions for the components of $\Delta z^a = (\Delta x^a, \Delta y^a, \Delta s^a)$ are given by

$$\begin{aligned}
\Delta y^a &= (AS^{-1}XA^T)^{-1}Ax, \\
\Delta s^a &= -A^T\Delta y^a, \\
\Delta x^a &= -x - S^{-1}X\Delta s^a.
\end{aligned}$$

This is precisely the affine-scaling direction in (3.9) with $r_p = r_d = 0$. On the other hand, the search direction computed to improve feasibility or decrease the residuals is given by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \overline{\Delta x}^a \\ \overline{\Delta y}^a \\ \overline{\Delta s}^a \end{bmatrix} = - \begin{bmatrix} r_p \\ r_d \\ 0 \end{bmatrix}. \tag{B.2}$$

This is the affine-scaling direction in (3.9) with $XSe = 0$. Solving for $\overline{\Delta z}^a = \left(\overline{\Delta x}^a, \overline{\Delta y}^a, \overline{\Delta s}^a\right)$, we have

$$\begin{aligned}
\overline{\Delta y}^a &= -(AS^{-1}XA^T)^{-1}(r_p + AS^{-1}Xr_d), \\
\overline{\Delta s}^a &= -r_d - A^T\overline{\Delta y}^a, \\
\overline{\Delta x}^a &= -S^{-1}X\overline{\Delta s}^a.
\end{aligned}$$

The sum of Δz^a and $\overline{\Delta z}^a$ is the affine-scaling direction.

Let $z = (x, y, s)$. Step lengths $\hat{\theta}$ and $\hat{\rho}$ are chosen so that the point $\tilde{z} = z + \hat{\theta}\Delta z^a + \hat{\rho}\overline{\Delta z}^a \in N_\beta$ where

$$N_\beta = \left\{(\widetilde{x}, \widetilde{s}) : \widetilde{x} > 0, \widetilde{s} > 0, \left\|\widetilde{X}\widetilde{s} - \widetilde{\mu}e\right\| \leq \beta\widetilde{\mu}\right\}$$

with

$$0 < \alpha < \beta \leq 2\alpha \leq \frac{\sqrt{2}}{1+\sqrt{2}} < 1.$$

The third search direction, $\Delta z^{cc} = (\Delta x^{cc}, \Delta y^{cc}, \Delta s^{cc})$ is computed to improve the centrality of the iterates. This linear system is given by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \widetilde{S} & 0 & \widetilde{X} \end{bmatrix} \begin{bmatrix} \Delta x^{cc} \\ \Delta y^{cc} \\ \Delta s^{cc} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \widetilde{\mu} e - \widetilde{X}\widetilde{s} \end{bmatrix}. \tag{B.3}$$

The expressions for the centering-corrector direction, $\Delta z^{cc} = (\Delta x^{cc}, \Delta y^{cc}, \Delta s^{cc})$, are given by

$$\begin{aligned} \Delta y^{cc} &= (A\widetilde{S}^{-1}\widetilde{X}A^T)^{-1}A\left(\widetilde{x} - \widetilde{\mu}\widetilde{S}^{-1}e\right), \\ \Delta s^{cc} &= -A^T\Delta y^{cc}, \\ \Delta x^{cc} &= -\widetilde{x} + \widetilde{S}^{-1}\left(\widetilde{\mu}e - \widetilde{X}\Delta s^{cc}\right). \end{aligned}$$

The updated solution, z^+, is obtained by setting

$$z^+ = \left(x^+, y^+, s^+\right) = \tilde{z} + \Delta z^{cc},$$

where z^+ satisfies $(x^+, s^+) \in N_\alpha = \{(x^+, s^+), \|X^+ s^+ - \mu^+ e\| \leq \alpha\mu\}$.

B.2 Potra's Predictor-Corrector Algorithm

It is assumed that A is an $m \times n$ matrix with full row rank. Therefore, we denote its Moore-Penrose pseudoinverse by $A^\dagger = A^T(AA^T)^{-1}$. The initial solution (or starting point) has the form

$$x^0 = \zeta e, \; s^0 = \sigma e, \; y^0 = 0,$$

where

$$\zeta \geq \left\|A^{\dagger}b\right\|_{\infty}, \ \sigma \geq \|c\|_{\infty}.$$

We define step lengths $\hat{\rho} \in (0,1]$ and $\hat{\theta}$ such that

$$(\mu^{+}, r_p^{+}, r_d^{+}) = (1-\hat{\rho})(\mu, r_p, r_d)$$

and

$$\hat{\theta} = \chi(\hat{\rho}).$$

The details surrounding the compution of the step lengths in the affine-scaling direction can be found in Potra [22]. Potra's predictor-corrector algorithm is stated below:

Input: (x, y, s) with $x = \zeta e \geq \|A^{\dagger}b\|_{\infty}$ e, $s = \sigma e \geq \|c\|_{\infty}$ e, and $y = 0$.

Initialize: $\epsilon \geq 0$; $r_p = Ax - b$; $r_d = A^T y + s - c$.

Main Algorithm:

```
while $x^T s > \epsilon$, $\|r_p\| > \epsilon$, or $\|r_d\| > \epsilon$

    Compute affine-scaling directions:

        I. Compute $\Delta z^a = (\Delta x^a, \Delta s^a, \Delta y^a)$:

            $\Delta y^a = \left(AS^{-1}XA^T\right)^{-1} Ax,$

            $\Delta s^a = -A^T \Delta y^a,$

            $\Delta x^a = -x - S^{-1}X\Delta s^a.$
```

II. Compute $\overline{\Delta z}^a = \left(\overline{\Delta x}^a, \overline{\Delta s}^a, \overline{\Delta y}^a\right)$:

$$\begin{aligned}
\overline{\Delta y}^a &= -(AS^{-1}XA^T)^{-1}(r_p + AS^{-1}Xr_d), \\
\overline{\Delta s}^a &= -r_d - A^T\overline{\Delta y}^a, \\
\overline{\Delta x}^a &= -S^{-1}X\overline{\Delta s}^a.
\end{aligned}$$

Compute affine steps:

Set

$$\begin{aligned}
\delta &= \frac{(\Delta x^a)^T\,\overline{\Delta s}^a + (\Delta s^a)^T\,\overline{\Delta x}^a}{n\mu}, \\
\eta &= \frac{\left(\overline{\Delta x}^a\right)^T\overline{\Delta s}^a}{n\mu},
\end{aligned}$$

and compute

$$\begin{aligned}
\hat{\rho} &= \min\left(\left\{\rho \in (0,1) : \rho \cdot R(\rho) \leq \left(\beta^2\mu^2 - \|f\|^2\right)\right\} \cup \{1\}\right), \\
\hat{\theta} &= \left(\frac{1+\delta\hat{\rho}}{1-\eta\hat{\rho}}\right)\hat{\rho}.
\end{aligned}$$

Compute $\hat{z} \in N_\beta$*:*

Compute $\hat{z} = z + \hat{\theta}\Delta z^a + \hat{\rho}\overline{\Delta z}^a$. If $\hat{\rho} = 1$, then $\hat{z} \in F^*$ (Optimal solution set). Terminate.

Compute centering-corrector direction:

$$\begin{aligned}
\Delta y^{cc} &= (A\widetilde{S}^{-1}\widetilde{X}A^T)^{-1}A\left(\widetilde{x} - \widetilde{\mu}\widetilde{S}^{-1}e\right), \\
\Delta s^{cc} &= -A^T\Delta y^{cc}, \\
\Delta x^{cc} &= -\widetilde{x} + \widetilde{S}^{-1}\left(\widetilde{\mu}e - \widetilde{X}\Delta s^{cc}\right).
\end{aligned}$$

Update solution

Compute $z^+ = \tilde{z} + \Delta z^{cc}$.

end(while)

B.3 The *redPC* Algorithm

The *redPC* algorithm is simply a "reduced" version of Potra's algorithm. This algorithm is similar to the *redPDAS* and *redMPC* algorithms in that we consider only those columns (components) of a matrix (vector) associated with Q. We let $\hat{r}_p = A_Q x_Q - b$ be the primal residual based on Q and define $\hat{\rho}^*$ to be a parameter such that

$$0 \leq \hat{\rho}^* < \hat{\rho} \leq 1.$$

The *redPC* is presented below:

```
Input: $(x, y, s)$ with $x = \zeta e \geq \|A^{\dagger}b\|_{\infty}$ e, $s = \sigma e \geq \|c\|_{\infty}$ e, and $y = 0$; $m < u_{bnd} \leq n$.
Initialize: $\epsilon = 10^{-8}$; $l_{bnd} = \min\{u_{bnd}, 3m\}$; $r_p = Ax - b$; $r_d = A^T y + s - c$.
Main Algorithm:
while $x^T s > \epsilon$, $\|r_p\| > \epsilon$, or $\|r_d\| > \epsilon$
    Select the most promising dual constraints.
    Compute affine-scaling directions:
```

I. Compute $\Delta z_Q^a = \left(\Delta x_Q^a, \Delta s_Q^a, \Delta y^a\right)$

$$\begin{aligned}
\Delta y^a &= \left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} A_Q x_Q, \\
\Delta s_Q^a &= -A_Q^T \Delta y^a, \\
\Delta x_Q^a &= -x_Q - S_Q^{-1} X_Q \Delta s_Q^a.
\end{aligned}$$

II. Compute $\overline{\Delta z}_Q^a = \left(\overline{\Delta x}_Q^a, \overline{\Delta s}_Q^a, \overline{\Delta y}^a\right)$

$$\begin{aligned}
\overline{\Delta y}^a &= -\left(A_Q S_Q^{-1} X_Q A_Q^T\right)^{-1} \left[\hat{r}_p + A_Q S_Q^{-1} X_Q (r_d)_Q\right], \\
\overline{\Delta s}_Q^a &= -(r_d)_Q - A_Q^T \overline{\Delta y}^a, \\
\overline{\Delta x}_Q^a &= -S_Q^{-1} X_Q \overline{\Delta s}_Q^a.
\end{aligned}$$

Compute affine steps:

Set

$$\begin{aligned}
\delta &= \frac{\left(\Delta x_Q^a\right)^T \overline{\Delta s}_Q^a + \left(\Delta s_Q^a\right)^T \overline{\Delta x}_Q^a}{n\mu}, \\
\eta &= \frac{\left(\overline{\Delta x}_Q^a\right)^T \overline{\Delta s}_Q^a}{n\mu}, \\
\xi &= \frac{\left(\overline{\Delta x}_Q^a\right)^T \overline{\Delta s}_Q^a}{n\mu}, \\
\gamma &= \frac{\left(1 - \hat{\rho}^*\right) x_{\bar{Q}}^T \overline{\Delta s}_{\bar{Q}}^a}{n\mu}, \\
\tau &= \frac{x_Q^T s_Q}{n\mu},
\end{aligned}$$

and compute

$$\begin{aligned}
\hat{\rho} &= \min\left(\left\{\rho \in (0,1) : \rho \cdot \bar{S}(\rho) \le \eta^4 \left(\beta^2\mu^2 - \|f\|^2\right)\right\} \cup \{1\}\right), \\
\hat{\rho}^* &= \beta\hat{\rho}, \\
\hat{\theta} &= \left(\frac{1 + \gamma + \left[\xi - (1-\tau)\beta\right]\hat{\rho}}{\eta - \delta\hat{\rho}}\right)\hat{\rho}.
\end{aligned}$$

Compute $\hat{z}$:

Set $\Delta x^a_{\bar{Q}} = 0,\ \overline{\Delta x}^a_{\bar{Q}} = -\frac{\hat{\rho}^*}{\hat{\rho}} x_{\bar{Q}},\ \Delta s^a_{\bar{Q}} = -A^T_{\bar{Q}} \Delta y^a,\ \overline{\Delta s}^a_{\bar{Q}} = (r_d)_{\bar{Q}} - A^T_{\bar{Q}} \overline{\Delta y}^a$ and compute $\hat{z} = z + \hat{\theta}\Delta z^a + \hat{\rho}\overline{\Delta z}^a$. If $\hat{\rho} = 1$, then $\hat{z} \in F^*$ (Optimal solution set). Terminate.

Compute centering-corrector direction:

$$\begin{aligned}
\Delta y^{cc} &= (A_Q \widetilde{S}_Q^{-1} \widetilde{X}_Q A_Q^T)^{-1} A_Q \left(\widetilde{x}_Q - \widetilde{\mu} \widetilde{S}_Q^{-1} e_Q \right), \\
\Delta s^{cc}_Q &= -A^T_Q \Delta y^{cc}, \\
\Delta x^{cc}_Q &= -\widetilde{x}_Q + \widetilde{S}_Q^{-1} \left(\widetilde{\mu} e_Q - \widetilde{X}_Q \Delta s^{cc}_Q \right).
\end{aligned}$$

Update solution

Set $\Delta x^{cc}_{\bar{Q}} = 0,\ \Delta s^{cc}_{\bar{Q}} = 0$ and compute $z^+ = \hat{z} + \Delta z^{cc}$.

end(while)

In Potra's algorithm, the affine-scaling directions are computed in two separate steps to improve feasibility and optimality. Using the separate components of the affine-scaling direction, Potra derives and incorporates the affine steps, $\hat{\rho}$ and $\hat{\theta}$, into an intermediate solution ($\hat{z} = z + \hat{\theta}\Delta z^a + \hat{\rho}\overline{\Delta z}^a$) to determine optimality. Although the details surrounding the affine steps have been eliminated in this thesis, they can be easily followed in Potra [22]. If the solution is not optimal, the centering-corrector direction is computed to produce a new solution.

There are signficant differences between Potra's algorithm and the *redPC* algorithm. The main difference is the "reduced" size of the matrices and vectors based on the set Q. Reducing the size of the data, however, drastically increased the amount of mathematical computation and detail in deriving the affine steps for the intermediate solution. Although the *redPC* algorithm can be shown to exhibit global convergence, we have been unsuccessful in proving polynomial complexity. Since the components of the directions in $\bar{Q}$ can be defined in different ways based on the choice of $\hat{\rho}^*$, variants of the *redPC* algorithm will be considered in future research.

BIBLIOGRAPHY

[1] M. Achache, H. Roumili, and A. Keraghel. A numerical study of an infeasible primal-dual path-following algorithm for linear programming. *Applied Mathematics and Computation*, 186:1472–1479, 2007.

[2] Various Authors. Netlib lp test problems. `http://www-fp.mcs.anl.gov/OTC/Guide/TestProblems/LPtest/`, 1985. Retrieved July 2004.

[3] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms 2nd Ed.* John Wiley and Sons, Hobeken, NJ, 1993.

[4] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization.* Athena Scientific, Belmont, MA, 1997.

[5] George B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ, 1963.

[6] George B. Dantzig and Yinyu Ye. A build-up interior-point method for linear programming: Affine scaling form. *Working Paper, Department of Management Science, University of Iowa*, 1991.

[7] Michael Gertz, Jorge Nocedal, and Annick Sartenaer. A starting-point strategy for nonlinear interior methods. *Applied Math Letters*, 17:945–952, 2004.

[8] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research*, 9:849–859, 1961.

[9] J.-Louis Goffin, Z.-Quan Luo, and Y. Ye. *On the Complexity of a Column Generation Algorithm for Convex or Quasiconvex Feasibility Problems.* Large Scale Optimization: State of the Art, W. W. Hager, D. W. Hearn and P.M. Pardalos, Editors, Kluwer Academic Publishers B.V, 1994.

[10] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, Baltimore, MD, 1989.

[11] D.den Hertog, Cornelis Roos, and Támas Terlaky. A build-up variant of the logarithmic barrier method for lp. *Operations Research Letters*, 12:181–186, 1992.

[12] D.den Hertog, Cornelis Roos, and Tamas Terlaky. Adding and deleting constraints in the logarithmic barrier method for lp. *Advances in Optimization and Approximation*, pages 166–185, 1994.

[13] Wikipedia Foundation Inc. Matrix norm. `http://en.wikipedia.org/wiki/Matrix\_norm`, 2002. Retrieved May 2007.

[14] Elizabeth John and E. Alper Yildirim. Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. `http://www.optimization-online.org/DB_FILE/2006/05/1389.pdf`, 2006. Retrieved April 2008.

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

[16] Masakazu Kojima, Nimrod Megiddo, and Shinji Mizuno. Polynomiality of infeasible-interior-point algorithms for linear programming. *Mathematical Programming*, 67:109–119, 1994.

[17] Sanjay Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[18] S. Mizuno, Michael J. Todd, and Yinyu Ye. On adaptive-step primal-dual interior-point algorithms for linear programming. *Mathematical Operations Research*, 18:964–981, 1993.

[19] Renato D.C. Monteiro, Ilan Adler, and Mauricio G.C. Resende. A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension. *SIAM Journal on Optimization*, 2(1):7–20, 1990.

[20] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming.* McGraw-Hill, New York, NY, 1996.

[21] E.R. Panier, André L. Tits, and J.N. Herskovits. A qp-free globally convergent, globally superlinearly convergent algorithm for inequality constrained optimization. *SIAM Journal on Control and Optimization*, 26(4):788–811, 1988.

[22] Florian A. Potra. An infeasible-interior-point predictor-corrector algorithm for linear programming. *SIAM Journal on Optimization*, 6:19–32, 1996.

[23] André L. Tits. An interior point method for linear programming, with an active set flavor. technical report TR-99-47, Institute for Systems Research, University of Maryland, College Park, MD, 1999.

[24] André L. Tits, P.-A. Absil, and William P. Woessner. Constraint reduction for linear programs with many inequality constraints. *SIAM Journal on Optimization*, 17:119–146, 2006.

[25] André 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.

[26] Robert J. Vanderbei. *Linear Programming: Foundations and Extensions.* Kluwer Academic Publishers, Boston, MA, 1996.

[27] David S. Watkins. *Fundamentals of Matrix Computations*. Wiley, New York, NY, 2002.

[28] Luke B. Winternitz, Stacey O. Nicholls, André L. Tits, and Dianne P. O'Leary. A constraint-reduced variant of the Mehrotra predictor-corrector algorithm. `http://www.optimization-online.org/ARCHIVE\_DIGEST/2007-07.html`, 2007. Retrieved October 2007.

[29] Steven J. Wright. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, PA, 1997.

[30] Yinyu Ye. A build-down scheme for linear programming. *Mathematical Programming*, 46:61–72, 1990.

[31] Yinyu Ye. A potential reduction algorithm allowing column generation. *SIAM Journal on Optimization*, 2(1):7–20, 1992.

[32] Yinyu Ye. Complexity analysis of the analytic center cutting plane method that uses mulitple cuts. *Mathematical Programming*, 78:85–104, 1997.

[33] Yin Zhang. Solving large-scale linear programs by interior-point methods under the MATLAB environment. technical report TR96-01, Department of Mathematics and Statistics, University of Maryland Baltimore County, Baltimore, MD, 1996.

[34] Yin Zhang. User's guide to LIPSOL linear-programming interior point solvers v0.4. *Optimization Methods and Software*, 11 & 12:385–396, 1999.