Nonsmooth Newton Methods for Solving the Best Approximation Problem; with Applications to Linear Programming

by

Tyler Weames

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics & Optimization

Waterloo, Ontario, Canada, 2023

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this thesis, we study the effects of applying a modified Levenberg-Marquardt regularization to a nonsmooth Newton method. We expand this application to exact and inexact nonsmooth Newton methods and apply it to the best approximation constrained to a polyhedral set problem. We also demonstrate that linear programs can be represented as a best approximation problem, extending the application of nonsmooth Newton methods to linear programming. This application provides us with insight into an external path following algorithm that, like the simplex method, takes a finite number of steps on the boundary of the polyhedral set. However, unlike the simplex method, these steps do not use basic feasible solutions.

Acknowledgements

I would first like to thank my supervisor Henry Wolkowicz. He is a very kind, compassionate, and patient teacher with a lot of wisdom to share. His incredible passion for Mathematics and Optimization continues to push me to grow as a student. Henry is also great at leading by example, he never asks you to do something that he could not do himself. I also appreciate Henry's genuine passion for life, and more specifically life in the Ontario area. He is constantly giving me ideas for how to integrate mine and my partner's life with the activities and people of Ontario.

I would also like to thank Professor Walaa Moursi and Professor Yair Censor for working with Henry and I on this problem, as well as for their valuable company. Meeting with Walaa and Yair has provided me with wonderful insight into the world outside of Canada. I am fortunate to count them as colleagues and friends.

I would like to thank Professor Heinz Bauschke, Professor Warren Hare, and Professor Yves Lucet of UBCO. I want to thank Heinz Bauschke for introducing me to convex optimization and sparking my passion for optimization. I want to thank Warren Hare, and Yves Lucet for their supervision during my directed studies at UBCO. They helped to foster my interests in numerical and derivative-free optimization. From them I have learned many valuable skills that I still carry with me at my studies at the University of Waterloo.

I am very grateful to my readers Professor Walaa Moursi and Professor Steve Vavasis for their meaningful feedback on my thesis.

I especially want to thank my partner Victoria Lange. She has been nothing short of extraordinarily supportive of me on my academic journey. Together we moved across Canada from our home in British Columbia to settle in Ontario. She has helped make Ontario feel more like home instead of like a home away from home. Words cannot express how grateful I am to have her in my life.

I want to express special thanks to my friends and family that have helped to support me during my studies. I am very grateful to my grandparents who moved out to Ontario around the same time I did. I am very fortunate to have you both in my life, and I am always grateful for my grandma's wonderful cooking. I would also like to thank the many friends I have made while living in Ontario. You have helped to make Kitchener-Waterloo feel like home. I would also like to give a final thanks to my friends in British Columbia. Thank you for keeping in touch and reminding me that even if we are 4000 kilometers apart that we can still laugh and revel in each others company.

I would like to give a final acknowledgment to everyone that has supported me on my journey so far. While I acknowledge that this thesis is my work, I could not have done it without the compassion and wisdom that everyone around me has shared.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

In this thesis we apply a modified Levenberg-Marquardt regularization to a nonsmooth Newton method to solve the *best approximation constrained to a polyhedral set* problem. We demonstrate numerically that a nonsmooth Newton method applied to a Moreau decomposition of the optimality conditions of the best approximation problem is highly competitive. We further show our method to be more efficient than many other methods chosen from both academic and professional areas. We also develop the novel approach of solving a linear program that exploits the sensitivity analysis of a solution to the best approximation problem.

The *best approximation problem,* ***BAP***, aims to find the nearest point x^* to a given point v, where x^* belongs to a given feasible region. In general, we denote the feasible region as C and represent the **BAP** as follows

$$x^*(v) = \underset{x \in C}{\operatorname{argmin}} \|x - v\| . \tag{1.1}$$

The **BAP** has a wide range of applications and arises naturally in many areas of industry, optimization, and approximation theory. We now describe some of the fundamental problems in this field with greater detail to motivate the discussion on how to solve the **BAP** more efficiently.

In intensity-modulated radiation therapy there are powerful energy beams that are shaped to match the shape of cancer cells moving through an arc while they deliver the radiation. The intensity of each beam is allowed to vary, making the precision to deliver the optimal amount of radiation to treat each cancer cell paramount. By discretizing the irradiated body into voxels and the external radiation field into beamlets, we can represent this problem as a feasibility-seeking problem with polyhedral constraints as done in [53]. Since delivering an acceptable treatment plan with minimal radiation intensities is preferable, this feasibility seeking problem can be replaced by the **BAP**. In other words, it can be represented as (1.1) with C being a set of linear inequalities intersected with the nonnegative orthant, and v being the zero-vector. This was examined in [52] with a simultaneous version of Hildreth's sequential algorithm for norm minimization

In fixed point theory, finding fixed points of nonexpansive mappings has been a growing field of importance. In the case where the nonexpansive mappings are projections onto some closed convex sets, then the fixed point problem becomes the convex feasibility problem [3]. The Halpern-Lions-Wittmann-Bauschke method, (**HLWB**), which we will expand on in Section 4.1, is an algorithm that solves the **BAP** using the common fixed points set of a family of firmly nonexpansive operators

as presented in [2]. As discussed above, intensity-modulated radiation therapy is one application of this problem. Other applications of this nature can be found in [1]. In particular, a detailed set of examples verifying the feasibility of a design drawn from Computer-Assisted-Design (CAD) software.

Lastly, we discuss an application of **BAP** used to solve an **LP**. It was proven in [38, 48] that linear programs can be equivalently expressed as a **BAP** constrained to a polyhedral set given a sufficiently large parameter R. In [39] Mangasarian proposes a Newton-type method with a barrier term on the objective. More discussion on how we approach solving the **LP** using the theory developed for the **BAP** using a regularized nonsmooth Newton method as well as remarks on Mangasarian's approach are in Chapter 5.

1.1 Notation

We work in a *Euclidean space,* X, with *inner product,* $\langle \cdot, \cdot \rangle$ and induced *norm,* $\|x\| = \sqrt{\langle x, x \rangle}$. Let $C \subseteq X$. Then we denote the *interior of* C, $\text{int}(C)$, and the *relative interior of* C, $\text{relint}(C)$. The set $\text{B}(x, \varepsilon)$ represents the *open ball* centered at x of radius ε.

Depending on context, we denote the vector of zeroes with length n as 0_n, or an n by n matrix of zeroes as 0_n. Furthermore, we denote the *identity matrix* as I and I_n denotes the identity matrix of size $\mathbb{R}^{n \times n}$. We denote the vector of all ones as e, where e_i is the (unit) vector of all zeroes except the ith entry, which has a one. Let $x \in \mathbb{R}^n$. Then we denote the linear map $\text{Diag}(x) : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ where the elements of x are placed on the diagonal of the $n \times n$ matrix, with the nondiagonal entries equal to 0. Furthermore, let $A \in \mathbb{R}^{n \times n}$. Then the linear map $\text{diag}(A) : \mathbb{R}^{n \times n} \to \mathbb{R}^n$ takes a square matrix A to its diagonal in $\mathbb{R}^n$.

For index sets $\mathcal{I}$ and $\mathcal{J}$, we denote the submatrix of A using the columns indexed by $\mathcal{I}$ as $A_{\mathcal{I}}$, and the submatrix of A using the columns indexed by $\mathcal{I}$ and the rows indexed by $\mathcal{J}$ as $A_{(\mathcal{J},\mathcal{I})}$. Alternatively, when the index set $\mathcal{I}$ (or $\mathcal{J}$) is not a subset of the possible indices, but the entire set, we use the MATLAB notation $A_{(\mathcal{J},:)}$ (or $A_{(:,\mathcal{I})}$) to define the submatrix comprised of the rows of A corresponding to $\mathcal{J}$ (or the columns of A corresponding to $\mathcal{I}$). In the case where a subscript for A exists, for example A_1, we denote the submatrix of A_1 using the columns indexed by $\mathcal{I}$ as $A_1(\mathcal{I})$, and the submatrix of A_1 using the columns indexed by $\mathcal{I}$ and the rows indexed by $\mathcal{J}$ as $A_1(\mathcal{J}, \mathcal{I})$.

Let $x \in \mathbb{R}^n$. Then we denote the projection of the vector x onto the nonnegative orthant as $x_+ = (\max\{0, x_i\})_{i=1}^n$. Furthermore, we denote the projection of the vector x onto the nonpositive orthant as $x_- = (\min\{(0, x_i\})_{i=1}^n$. Let $A \in \mathbb{R}^{m \times n}$. Then we denote the Moore-Penrose pseudoinverse of A as $A^\dagger$.

1.2 Contributions and Organization

This thesis is separated into six chapters. In Chapter 2 we present the necessary background in convex and nonsmooth analysis, linear optimization, and time complexity.

The main results of the thesis appear throughout Chapters 3 to 5. In Chapters 3 and 4 of the thesis we discuss the **BAP**. We introduce a nonsmooth Newton method used to solve the **BAP** as well as numerical results that we compare to **HLWB**. We have the following contributions in Chapter 3 and Chapter 4:

1. We present basic theory for the **BAP** as well as its Lagrangian dual as shown in Theorem 3.1.1. This includes an application of the Moreau decomposition that yields a single equation capturing all three KKT optimality conditions (primal feasibility, dual feasibility, and complementary slackness). This single equation, $F(y)$, is in the dimension of the dual variable y, and we examine applications where the dimension of the dual variable is far smaller than the dimension of the primal variable x.

2. We present the *regularized nonsmooth Newton method,* ***RNNM***, where no line search is used for guaranteeing convergence. This is handled by introducing a robust regularization of the constraint set to decrease the distance of the inital point to the optimal solution (See Lemma 5.2.1).

3. We show that the regularization from a modified *Levenberg-Marquardt,* ***LM***, method yields a descent direction (See Lemma 3.2.1.)

4. We use MATLAB to present numerical results of **RNNM** that exactly and inexactly solves for the search direction. These numerical results are for medium-sized problems, and compare the performance of **RNNM** to **HLWB** [2], *quadratic programming proximal augmented Lagrangian method*, (**QPPAL**) [35], and MATLAB's commerical solver *lsqlin*. We conclude that the performance of **RNNM** is surprisingly impressive for a simple code that does not take advantage of .mex files. The code outperforms many other solvers on every problem.

In Chapter 5 we introduce some background on linear programming. The background presented includes a brief history of linear programming, and examines how interior point methods are related to Newton methods. We also have the following contributions in Chapter 5:

1. We consider solving the nearest point problem as a **BAP** subproblem of the **LP** to find a sufficiently large R such that the **BAP** subproblem provides an optimal solution for the **LP**. We do this by using sensitivity analysis to increase the parameter R in Lemma 5.2.1 until Rc lies within the normal cone of the optimal face (See Theorem 5.2.3). We call this method the *stepping stones external path following method,* ***SSEPF***. When the optimal face is a vertex, then we have found the unique optimal solution. Otherwise, we have found the minimum norm solution for the **LP**. We prove this approach terminates in a finite number of iterations.

2. We present numerical results of **SSEPF** for solving large-scale linear programs. In our application, we also use **RNNM** to solve the **BAP** subproblem, providing numerical results for **RNNM** applied to the large-scale **BAP**. We compare the performance of **SSEPF** using **RNNM** to solve the **BAP** subproblem to MATLAB's *linprog* code, using both the *dual simplex* and the *interior-point method* algorithms. We also compare with Mosek's *dual simplex* and *interior-point method*, and with the *semismooth Newton inexact proximal augmented Lagrangian method,* ***SNIPAL*** [34].

Chapter 2

Background

2.1 Convex Analysis Background

We now present some background knowledge on convex analysis and convex optimization. We start with introducing convex sets in Section 2.1.1, convex functions in Section 2.1.2, cones in Section 2.1.3, projections in Section 2.1.4, and convex optimization theory in Sections 2.1.5 and 2.1.6. Readers that are familiar with the basics of convex analysis can skip this section. For missing results, see [9, 43, 50] and the references therein.

2.1.1 Convex Sets

Definition 2.1.1 (Convex combination)**.** *Let $x_i \in X$, $\lambda_i \in [0,1]$, and $\sum_{i=1}^{k} \lambda_i = 1$ for $i = 1, \ldots, k$. Then the vector sum $\lambda_1 x_1 + \ldots + \lambda_k x_k$ is called a convex combination of the vectors.*

Definition 2.1.2 (Convex set)**.** *A subset $C \subseteq X$ is convex if*

$$\lambda \in [0,1],\ x, y \in C \implies \lambda x + (1-\lambda) y \in C.$$

Definition 2.1.3 (Affine set)**.** *A subset $C \subseteq X$ is affine if*

$$\lambda \in \mathbb{R},\ x, y \in C \implies \lambda x + (1-\lambda) y \in C.$$

Example 2.1.4 (Half-spaces and hyperplanes)**.** *Let $a \in X$ and $\alpha \in \mathbb{R}$, then the following sets are denoted as half-spaces and hyperplanes respectively*

$$H_{\leq,=} := H_{\leq,=}(a, \alpha) = \{x \in X \mid \langle a, x \rangle \, (\leq, =) \, \alpha\}.$$

Both halfspaces and hyperplanes are examples of convex sets.

Theorem 2.1.5. *The intersection of an arbitrary collection of convex sets in X is a convex set.*

Remark 2.1.6. *The intersection of half-spaces is a convex set.*

Proof. See Example 2.1.4 and Theorem 2.1.5 □

Definition 2.1.7 (Convex hull)**.** *Let $S \subseteq X$. The convex hull of S, denoted by* $\operatorname{conv}(S)$*, is the intersection of all convex sets containing S, i.e., the smallest convex set of X containing S.*

Theorem 2.1.8. *Let $S \subseteq X$. Then* $\operatorname{conv}(S)$ *consists of all the convex combinations of the elements of S.*

2.1.2 Convex Functions

Definition 2.1.9 (Epigraph of a function)**.** *Let $f : X \to [-\infty, +\infty]$. The epigraph of f is*

$$\operatorname{epi}(f) = \{(x, \alpha) \,|\, f(x) \leq \alpha\} \subseteq X \times \mathbb{R}.$$

Definition 2.1.10 (Proper function)**.** *Let $f : X \to [-\infty, +\infty]$. Then* $\operatorname{dom} f = \{x \in X \,|\, f(x) < +\infty\}$ *and f is proper if* $\operatorname{dom} f \neq \emptyset$ *and $f(x) > -\infty, \forall x \in X$.*

Theorem 2.1.11 (Jensen's inequality)**.** *Let $f : X \to [-\infty, +\infty]$. Then f is convex if and only if*

$$x, y \in \operatorname{dom}(f),\ \lambda \in (0,1) \implies f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y). \tag{2.1}$$

Remark 2.1.12. *If the inequality in (2.1) is strict, then the function f is called strictly convex.*

Definition 2.1.13 (Lower semicontinuous)**.** *Let $f : X \to [-\infty, +\infty]$, and let $x \in X$. Then f is lower semicontinuous, lsc , at x if*

$$\liminf_{x_n \to x} f(x_n) \geq f(x), \qquad \text{for every sequence } x_n \to x.$$

Remark 2.1.14. *The following are well known properties of lower semicontinuity and lower semicontinuous functions:*

(1) f is lsc if f is lsc at every point in x.

(2) If f is continuous, then it is lsc .

(3) f is lsc if and only if $\operatorname{epi}(f)$ *is closed.*

Theorem 2.1.15. *Let $f : X \to]-\infty, +\infty]$ be lsc and proper, and let $\emptyset \neq C \subseteq X$ be compact such that $C \cap \operatorname{dom} f \neq \emptyset$. Then*

(1) f is bounded below over C;

(2) f attains its minimum value over C.

Definition 2.1.16 (Subgradients)**.** *Let $f : X \to]-\infty, +\infty]$ be proper, and let $x \in \operatorname{dom} f$, $u \in X$. Then u is a subgradient of f at x if*

$$f(y) \geq f(x) + \langle u, y - x \rangle, \text{ for all } y \in X.$$

The subdifferential of f at x is the set of all subgradients of f at x, i.e.,

$$\partial f : X \rightrightarrows X : x \mapsto \{u \in X \,|\, f(y) \geq f(x) + \langle u, y - x \rangle\}.$$

Definition 2.1.17 (Coercivity)**.** *Let $f : X \to]-\infty, +\infty]$. Then f is coercive if*

$$\lim_{\|x\| \to +\infty} f(x) = +\infty.$$

2.1.3 Cones and Faces

We now define cones and faces with the intention of discussing the Moreau Decomposition in Section 2.1.4, and the use of facial reduction in Section 2.3.4.

Definition 2.1.18 (Cone)**.** *Let $K \subseteq X$. Then K is a cone if for every $k \in K$ and $\lambda \geq 0$, we have $\lambda k \in K$.*

In other words, a cone is a set that is nonnegative homogeneous. A cone $K \subseteq X$ is a *convex cone* if it satisfies the properties of a cone and convexity. We assume in the following definitions that $K \subseteq X$ is a cone.

Definition 2.1.19 (Pointed cone)**.** *A cone K is a pointed cone if it contains no line, i.e., if*

$$x, -x \in K \implies x = 0.$$

Definition 2.1.20 (Proper cone)**.** *A cone K is a proper cone if K is convex, closed, pointed, and has nonempty interior.*

Proper cones can be used to describe *generalized inequalities* with a partial ordering in X. In other words, let K be a proper cone, then the *partial ordering on K*, $\succeq_K$, has the following properties:

(1) $\succeq_K$ is reflexive: $x \succeq_K x$.

(2) $x \succeq_K y \implies x - y \in K$ and $x \succ_K y \implies x - y \in \operatorname{int} K$.

(3) $\succeq_K$ is antisymmetric: $x \succeq_K y \implies x - y \in K$ and $y \succeq_K x \implies x = y$.

(4) $\succeq_K$ is preserved under addition: if $x \succeq_K y$ and $u \succeq_K v$, then $x + u \succeq_K y + v$.

(5) $\succeq_K$ is transitive: $x \succeq_K y$ and $y \succeq_K z$ implies $x \succeq_K z$.

(6) $\succeq_K$ is preserved under nonnegative scaling: if $x \succeq_K y$ and $\alpha \geq 0$, then $\alpha x \succeq_K \alpha y$.

We note that the above properties in items 4 and 5 hold for the *strict partial ordering on K*, $\succ_K$, and $\succ_K$ is preserved under positive scaling. We note that in the case where $K = \mathbb{R}_+$, then the partial ordering $\succeq_{\mathbb{R}_+}$ is equivalent to the ordering $\geq$ in $\mathbb{R}$.

Definition 2.1.21 (Dual cone)**.** *The dual cone of a cone K is the set*

$$K^* := \{y \in X \mid \langle x, y \rangle \geq 0, \forall\, x \in K\}.$$

If a cone is equal to its dual cone, $K = K^*$, then it is called a *self-dual cone*. Important examples of proper self-dual cones are $\mathbb{R}^n_+$ and $\mathbb{S}^n_+$.

Definition 2.1.22 (Polar cone)**.** *The polar cone of a cone K is the set*

$$K^\circ := \{y \in X \mid \langle x, y\rangle \leq 0, \forall\, x \in K\}.$$

Definition 2.1.23 (Normal cone)**.** *Let C be a nonempty convex subset of X and let $x \in X$. Then the normal cone of C at x is*

$$N_C(x) = \begin{cases} u \in X \mid \sup_{c\in C}\langle c - x, u\rangle \leq 0 & x \in C, \\ \emptyset & x \notin C. \end{cases}$$

We now discuss the decomposition of cones into faces for our discussion on facial reduction in Section 2.3.4.

Definition 2.1.24 (Face)**.** *Let K be a convex cone. A convex cone $F \subseteq K$ is called a face of K, denoted $F \trianglelefteq K$, if the following implication holds*

$$x \in K, y \in K, x + y \in F \implies x, y \in F.$$

A face $F \trianglelefteq K$ is a *proper face* if it is neither empty or all of K.

Definition 2.1.25 (Exposed face)**.** *Any set of the form $F = v^\perp \cap K$ for some vector $v \in K^*$, is called an exposed face of K. The vector v is called the exposing vector of F.*

A convex cone K is called a *facially exposed cone* if all of its faces are exposed faces.

Definition 2.1.26 (Conjugate face)**.** *With any face F of a convex cone K, we denote the conjugate face as $F^\lhd := K^* \cap F^\perp$.*

We note that all x in the relative interior of F expose the conjugate face $F^\lhd$. Therefore, the conjugate face is always an exposed face.

Definition 2.1.27 (Minimal face)**.** *The minimal face of a convex cone K containing a set $S \subseteq K$ is the intersection of all faces of K containing S. This is denoted as* face(S, K).

2.1.4 Projections

Definition 2.1.28 (Distance function)**.** *Given a nonempty set $C \subseteq X$, the distance function to C, $d_C(x)$, is defined by*

$$d_C(x) = \inf_{y\in C} \|x - y\|.$$

Definition 2.1.29 (Projection)**.** *Let $\emptyset \neq C \subseteq X$, let $x \in X$ and let $p \in C$. Then p is a projection of x onto C if*

$$d_C(x) = \|x - p\|.$$

If every point in X has exactly one projection onto C, the projection operator onto C, denoted by P_C, is the operator that maps each point $x \in C$ onto $P_C x$.

Remark 2.1.30. *Let C be nonempty closed convex subset of X. Then for every $x \in X$ the projection $P_C x$ exists and is unique.*

Theorem 2.1.31 (Moreau Decomposition)**.** *Let $K \subseteq X$ be a convex cone, and K° be the polar cone of K. Then the following statements are equivalent:*

(1) $z = x + y$, $x \in K, y \in K^\circ$ *and* $\langle x, y\rangle = 0$.

(2) $x = P_K z$ *and* $y = P_{K^\circ} z$.

An interesting generalization of the Moreau Decomposition of polar cones to conjugate cones is described as follows and can be found in [22].

Definition 2.1.32 (Conjugate Decomposition)**.** *Let K be a closed convex cone in $\mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$. A point $z \in \mathbb{R}^n$ is said to have a conjugate decomposition with respect to K and A if there exists x and y such that the following conditions hold:*

(1) $z = x + y$, $x \in K, y \in K^A := \{s \,|\, x^T(A + A^T)s \leq 0, \forall\, x \in K\}$;

(2) $x^T(A + A^T)y = 0$.

The closed convex cone K^A is called the conjugate cone.

For general $A \in \mathbb{R}^{n\times n}$ and closed convex cone K, a conjugate decomposition of z does not necessarily exist. We note that when A is the identity matrix, then the conjugate decomposition is the Moreau Decomposition.

2.1.5 Unconstrained Optimization

For $f : X \to]-\infty, +\infty]$ and f is convex, the *unconstrained convex optimization problem* is

$$\min_x f(x).$$

We note that the *minimizers of f* are denoted as the set $\operatorname{argmin} f$. Under the assumption that f is a convex function the set of minimizers have the following property.

Theorem 2.1.33 (Fermat)**.** *Let $f : X \to]-\infty, +\infty]$ be proper. Then the minimizers of f are*

$$\operatorname{argmin} f = \{x \in X \,|\, 0 \in \partial f(x)\}.$$

Furthermore, since f is convex, the local minimizers are also global minimizers.

Proposition 2.1.34. *If f is strictly convex, coercive, then there is a unique minimizer of f.*

Definition 2.1.35 (Descent Direction)**.** *Consider the problem $\min_x f(x)$ and f is convex. Then a vector $d \neq 0$ is called a descent direction at x if for some $\varepsilon > 0$ sufficiently small we have*

$$f(x + td) < f(x), \quad \textit{for all } 0 < t < \varepsilon.$$

The notion of a descent direction is important for algorithmic processes minimizing an objective function. The outline of these processes can often be generalized to choosing a descent direction, of which there are many, and an appropriate step length. There is an incredible amount of theory for picking the descent direction and an appropriate step length. For missing results a reader may refer to [9] and the references therein.

2.1.6 Constrained Optimization

We now study the following *constrained optimization problem* where we restrict the *objective function* $f(x)$ to be a proper convex function, the *inequality constraints* $g_i(x)$ to be proper convex functions, and the *equality constraints* $h_j(x)$ to be affine functions

$$\begin{array}{ll} \min_x & f(x) \\ \text{subject to} & g_i(x) \leq 0,\ i = 1, \dots, m \\ & h_j(x) = 0,\ j = 1, \dots, p. \end{array} \tag{2.2}$$

Since the domain of (2.2) is the intersection of convex sets, the feasible domain is convex by Theorem 2.1.5. Therefore, we call the problem defined by (2.2) a *convex optimization* problem. Furthermore, we assume the *feasible domain* of (2.2) is nonempty and denote it as $C := \operatorname{dom} f \cap \bigcap_{i=1}^m \operatorname{dom} g_i \cap \bigcap_{j=1}^p \operatorname{dom} h_j$. Next we introduce the notion of a feasible descent direction. Recall the definition of a descent direction,

$$f(x + td) < f(x),\ \forall\, 0 < t < \varepsilon.$$

Then we similarly define a feasible descent direction for (2.2).

Definition 2.1.36 (Feasible Descent Direction)**.** *Consider the convex problem with $f : \Omega \to \mathbb{R}$, $\emptyset \neq \Omega \subset \mathbb{R}^n$ a convex set, and*

$$\begin{array}{ll} \min & f(x) \\ \textit{subject to} & x \in C. \end{array}$$

A vector $d \neq 0$ is called a feasible descent direction at $x \in C$ if there exists $\varepsilon > 0$ such that $f(x + \varepsilon d) < f(x)$ and $x + td \in C$ for all $t \in [0, \varepsilon]$.

We introduce the notion of Lagrange multipliers and duality with assumptions of convexity on the objective function and the feasible domain. For the subsequent duality theory for convex optimization we refer to [9]. We assume that the feasible domain of (2.2) is nonempty and denote it as $C := \operatorname{dom} f \cap \bigcap_{i=1}^m \operatorname{dom} g_i \cap \bigcap_{j=1}^p \operatorname{dom} h_j$. We define $(\lambda_1, \dots, \lambda_m) \times (\mu_1, \dots, \mu_p) \in \mathbb{R}^m \times \mathbb{R}^p$ to be a vector of *Lagrange multipliers* or the *dual variables* if $\lambda_i \geq 0$, $\forall i \in \{1, \dots, m\}$. Then the *Lagrangian* of (2.2) is the function $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ defined as follows

$$L(x, \lambda, \mu) := f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \mu_j h_j(x). \tag{2.3}$$

We define the *Lagrange dual function* or *dual functional* $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as the minimum value of the Lagrangian over x for $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p$

$$g(\lambda,\mu) := \inf_{x\in C} L(x,\lambda,\mu) = \inf_{x\in C}\left(f(x) + \sum_{i=1}^{m}\lambda_i g_i(x) + \sum_{j=1}^{p}\mu_j h_j(x)\right). \tag{2.4}$$

The dual functional provides a lower bound on the *optimal value* p^* of the problem (2.2) for every feasible *primal-dual pair* $(\tilde{x},\tilde{\lambda},\tilde{\mu})$, i.e.,

$$g(\tilde{\lambda},\tilde{\mu}) = \inf_{x\in C} L(x,\tilde{\lambda},\tilde{\mu}) \le L(\tilde{x},\tilde{\lambda},\tilde{\mu}) \le f(\tilde{x}). \tag{2.5}$$

The above inequality holds for every feasible primal-dual pair $(\tilde{x},\tilde{\lambda},\tilde{\mu})$, but loses meaning when $g(\tilde{\lambda},\tilde{\mu}) = -\infty$. Therefore, we consider the dual variables feasible when $\tilde{\lambda} \ge 0$ and $(\tilde{\lambda},\tilde{\mu}) \in \operatorname{dom} g$. As mentioned previously, the inequality in (2.5) implies that for all feasible dual variables we have $g(\tilde{\lambda},\tilde{\mu}) \le f(x^*) = p^*$. This notion is referred to as *weak duality*.

Since the dual functional provides a lower bound for p^*, it is natural to formulate an optimization problem that finds the best lower bound, i.e.,

$$\begin{array}{ll} \max_{\lambda,\mu} & g(\lambda,\mu) \\ \text{subject to} & \lambda \ge 0. \end{array} \tag{2.6}$$

The problem defined in (2.6) is called the *Lagrangian dual problem* of (2.2), and (2.2) is called the *primal problem*. We refer to the optimal solution of (2.6), (λ^*,μ^*), as the *dual optimal pair* or *optimal Lagrange multipliers*. The optimal value attained for (2.6) is denoted as d^* and we note that $d^* \le p^*$. If $d^* = p^*$, then we have a *zero optimality gap*. Furthermore, if the dual optimal value is attainable, then we have *dual attainment*. If we have zero optimality gap and dual attainment, then we say that *strong duality* holds.

In cases where $d^* \ne p^*$, we say we have an *optimality gap* of $p^* - d^*$. Before discussing the necessary conditions for strong duality to hold for convex optimization, we define the Slater point.

Definition 2.1.37 (Slater point)**.** *A point $\bar{x} \in C$ of (2.2) is called a Slater point if*

$$g_i(\bar{x}) < 0,\ \forall\, i = \{1,\dots,m\}.$$

The Slater point is also referred to as a strictly feasible point.

The conditions for strong duality to hold are dependent on the properties of (2.2). In the case of linear programming, if the primal problem is feasible and has a finite optimal value, then strong duality always holds. However, in convex optimization we require stronger conditions known as *constraint qualifications*. An example of a constraint qualification for convex programming is the existence of a Slater point, which is known as *Slater's condition*.

Now that we have defined the Lagrangian, we define the KKT conditions.

Definition 2.1.38 (KKT Conditions)**.** *Let (x^*,λ^*,μ^*) be the optimal primal-dual pair for problem (2.2). Assume that $f, g_1,\dots,g_m, h_1,\dots,h_p$ are differentiable functions. Then the Karush-Kuhn-Tucker, KKT, conditions are*

$$\begin{aligned}
\nabla f(x^*) + \textstyle\sum_{i=1}^m \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^p \mu_j^* \nabla h_j(x^*) = 0, & \quad \textit{(stationarity)} \\
\left.\begin{aligned} g_i(x^*) \leq 0, \\ h_j(x^*) = 0, \end{aligned}\right\} & \quad \textit{(primal feasibility)} \\
\lambda_i^* \geq 0, & \quad \textit{(dual feasibility)} \\
\lambda_i^* g_i(x^*) = 0. & \quad \textit{(complementary slackness)}
\end{aligned} \tag{2.7}$$

Therefore, suppose that (2.2) is a convex optimization problem and there exists a Slater point $\bar{x}$ as described in Definition 2.1.37. Then the KKT conditions as described in Definition 2.1.38 are both necessary and sufficient for guaranteeing the primal-dual pair (x^*, λ^*, μ^*) is optimal. In other words, (x^*, λ^*, μ^*) is an optimal primal-dual solution for problems (2.2) and (2.6) with zero duality gap and dual attainment if and only if (x^*, λ^*, μ^*) satisfies (2.7).

2.2 Nonsmooth Analysis Background

We now present some background knowledge on nonsmooth analysis. We introduce the generalized gradient and generalized Jacobian in Section 2.2.1. These results are utilized in Chapter 3

2.2.1 Generalized Gradient

This section contains contents on nonsmooth analysis and generalized gradients from [14, Section 1, Section 2.1].

Definition 2.2.1 (Local Lipschitz continuity)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$, and let $x \in \mathbb{R}^n$. The function f is said to be L-Lipschitz near x if there exists $L > 0$ and $\varepsilon > 0$ such that we have*

$$|f(y) - f(x)| \leq L \|y - x\|, \forall\, x, y \in \mathrm{B}(x, \varepsilon).$$

Definition 2.2.2 (Generalized directional derivative)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz continuous near $x \in \mathbb{R}^n$and let d be a vector in $\mathbb{R}^n$. Then for $y \in \mathbb{R}^n$ and $t > 0$, we define the generalized directional derivative of f at x in the direction d, denoted $f^\circ(x; d)$, as follows*

$$f^\circ(x; d) = \limsup_{y \to x,\, t \downarrow 0} \frac{f(y + td) - f(y)}{t}.$$

In general, a function can be Lipschitz continuous on its domain, but not differentiable. An example of this is the function $f(x) = |x|$. We note that all future discussion of the generalized directional derivative refers to the *Clarke subdifferential* or *Clarke generalized derivative*.

Corollary 2.2.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz continuous near $x \in \mathbb{R}^n$, and let $d \in \mathbb{R}^n$. Then*

$$\frac{f(y + td) - f(y)}{t} \leq L \|d\|.$$

Proof.

$$\begin{aligned}\frac{f(y+td)-f(y)}{t} &\leq \left\| \frac{f(y+td)-f(y)}{t} \right\| \\ &= \frac{\|f(y+td)-f(y)\|}{t} \\ &\leq \frac{L\|y+td-y\|}{t} \\ &= \frac{tL\|d\|}{t} \\ &= L\|d\|.\end{aligned}$$

□

Definition 2.2.4 (Generalized gradient)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz continuous near $x \in \mathbb{R}^n$. Then the generalized gradient of f at x is*

$$\partial^\circ f(x) := \{z \in \mathbb{R}^n \mid f^\circ(x;d) \geq \langle d, z\rangle,\ \forall d \in \mathbb{R}^n\}.$$

The generalized gradient is a generalization of the gradient in the absence of smoothness; and is a generalization of the subgradient in the absence of convexity.

Theorem 2.2.5 ([42, Rademacher's therorem])**.** *A locally Lipschitz function is differentiable almost everywhere.*

The following is another characterization of the generalized gradient defined in Definition 2.2.4 that takes advantage of locally Lipschitz functions that are differentiable almost everywhere.

Theorem 2.2.6. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz continuous near $x \in \mathbb{R}^n$ and Ω_f be the set of points where f fails to be locally Lipschitz, and suppose $S \in \mathbb{R}^n$ is any set of Lebesgue measure 0. Then*

$$\partial^\circ f(x) = \operatorname{conv}\{\lim \nabla f(x_i) \mid x_i \to x,\ x_i \notin S,\ x_i \notin \Omega_f\}.$$

In other words, $\partial^\circ f(x)$ is the convex hull of all limit points of the form form $\lim \nabla f(x_i)$, where $\{x_i\}$ is any sequence which converges to x while avoiding $S \cup \Omega_f$.

We now consider the vector valued function $F : \mathbb{R}^n \to \mathbb{R}^m$. In other words, we express F as the vector of component-wise functions, $F := [f_1(x), f_2(x), \ldots, f_m(x)]$. We assume that f_i is Lipschitz near x for all $x \in \mathbb{R}^n$ and for all $i \in \{1, \ldots, m\}$, and therefore F is by definition locally Lipschitz. Thus, Rademacher's theorem stipulates that all f_i (and therefore F) are differentiable almost everywhere on any neighborhood of x in which F is Lipschitz. We will denote the set of points of which F fails to be differentiable as Ω_F. We shall denote $J(x) \in \mathbb{R}^{m\times n}$ as the Jacobian of F whenever x is a point at which the necessary partial derivatives exist.

Definition 2.2.7 (Generalized Jacobian)**.** *The generalized Jacobian of F at x, denoted $\partial^\circ F(x)$, is the convex hull of all $m \times n$ matrices V obtained as the limit of a sequence of the form $J(x_i)$, where $x_i \to x$ and $x_i \notin \Omega_F$. In other words, we have*

$$\partial^\circ F(x) = \operatorname{conv}(\lim JF(x_i) \mid x_i \to x,\ x_i \notin \Omega_F).$$

Furthermore, we say that $\partial^\circ F(x)$ is nonsingular if every $V \in \partial^\circ F(x)$ is nonsingular.

2.3 Linear Programming

The purpose of this section is to present the necessary background knowledge of linear programming that will be utilized in subsequent chapters of this thesis. The definitions and theorems referenced in this section can be found in [7]. Readers that are familiar with the basics of linear programming can skip this section. The contents of this section will not be be needed until Chapter 5. The reader may want to revisit this section.

2.3.1 Extreme points and Basic Feasible Solutions

We start by formally defining a polyhedral set as an intersection of finitely many half-spaces and hyperplanes.

Definition 2.3.1 (Polyhedron)**.** *A polyhedron is the set*

$$P := \{x \in \mathbb{R}^n \mid Ax \,(\leq, =, \geq)\, b\},\ A \in \mathbb{R}^{m\times n},\ b \in \mathbb{R}^m.$$

The expression of the polyhedron above emphasizes that both inequality and equality constraints are appropriate representations of polyhedra. When discussing the polyhedron P, we will always assume $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $x \in \mathbb{R}^n$. Now that we have defined a polyhedral set, we provide geometric definitions of a vertex and an extreme point.

Definition 2.3.2 (Vertex)**.** *Let P be a polyhedron. A point $x \in P$ is a vertex of P if there exists a vector $c \in \mathbb{R}^n$ such that $c^T x < c^T y$ for all y satisfying $y \in P$ and $y \neq x$.*

In other words, x is a vertex of P if and only if there exists a hyperplane, $H_= := \{y \mid c^T y = c^T x\}$, such that the intersection of P and $H_=$ is x.

Definition 2.3.3 (Extreme Point)**.** *Let P be a polyhedron. A vector $x \in P$ is an extreme point of P if we cannot find two distinct vectors $y, z \in P$, both distinct from x, with scalar $\lambda \in (0, 1)$ such that*

$$x = \lambda y + (1 - \lambda)z.$$

In other words, we cannot form a line segment from y to z that contains x.

The geometric definitions for vertices and extreme points are difficult to utilize algorithmically. Intuitively, we want to provide an algebraic definition of an extreme point of a polyhedral set that also lies in the nonnegative orthant. To do so, we need terminology about active sets.

Definition 2.3.4 (Active Constraint)**.** *If a vector x satisfies $A_{(i,:)}x = b_i$ for some $i \in \{1, \ldots, m\}$, then we say the corresponding constraint is an active constraint.*

If there are n constraints that are active at a vector $x \in \mathbb{R}^n$, then x satisfies a system of n linear equations in n variables. Therefore, this system of active constraints has a unique solution if and only if the active constraints are linearly independent.

Theorem 2.3.5. *Let $x \in \mathbb{R}^n$ and $I = \{i \mid A_{(i,:)}x = b_i\}$ be the set of indices of the constraints that are active at x. Then the following are equivalent:*

1. *There exists n vectors in in the set $\{A_{(i,:)} \mid i \in I\}$, which are linearly independent.*
2. *The vectors $A_{(i,:)}$, $i \in I$, span $\mathbb{R}^n$. In other words, every $\bar{x} \in \mathbb{R}^n$ can be expressed as a linear combination of the vectors $A_{(i,:)}$.*
3. *The system of equations $A_{(i,:)}x = b_i$ has a unique solution for all $i \in I$.*

Therefore, as a consequence of Theorem 2.3.5, if there are at least n linearly independent active constraints at a point x, then we can explicitly express x as the unique solution of $A^T_{(i,:)}x = b_i$. We call this characterization of x a basic solution.

Definition 2.3.6 (Basic and basic feasible solution)**.** *Let P be a polyhedron, and let $x \in \mathbb{R}^n$.*

1. *The vector x is a basic solution if at least n of the active constraints at x are linearly independent.*
2. *If x is a basic solution that satisfies all of the constraints of P, we say it is a basic feasible solution.*

Theorem 2.3.7. *Let P be a nonempty polyhedron and let $x \in P$. Then the following are equivalent:*

1. *x is a vertex of P.*
2. *x is an extreme point of P.*
3. *x is a basic feasible solution of P.*

Definition 2.3.6 describes basic feasible solutions in the general polyhedral setting.

Now, we specialize to polyhedra in the *standard equality form*, $P := \{x \in \mathbb{R}^n \mid Ax = b \in \mathbb{R}^m\}$, and assume that the rows of A are linearly independent. Since the rows of A are in $\mathbb{R}^n$, we will assume that $0 < m \leq n$. We now define a *linear program* in standard equality form,

$$\begin{aligned} \max \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ & x \geq 0. \end{aligned} \tag{2.8}$$

Recall from our earlier discussion that there must be at least n active constraints. Since P is in standard equality form, there are m equality constraints. Therefore, we have m active constraints from $Ax = b$, and we must have $n - m$ active nonnegativity constraints to obtain a basic solution. In other words, there are at least $n - m$ variables x_i such that $x_i = 0$, $i \in \{1, \ldots, n\}$.

Theorem 2.3.8. *Consider the constraints $Ax = b$ and $x \geq 0$ and assume that $A \in \mathbb{R}^{m \times n}$ has linearly independent rows. A vector $x \in \mathbb{R}^n$ is a basic solution if and only if $Ax = b$, and there exists indices $B := \{B_1, \ldots, B_m\}$ such that for all $i \in B$ and $j \notin B$ we have:*

(1) The columns A_i are linearly independent.

(2) If $j \notin B$, then $x_j = 0$.

We conclude the following from Definition 2.3.6 and Theorem 2.3.8. For $i \in B$ and $j \notin B$ if the columns A_i are linearly independent, $x_j = 0$, and $x_i \geq 0$, then x is a *basic feasible solution.*

2.3.2 Degeneracy

Now that we have established what basic feasible solutions are, we can begin our discussion on degeneracy, i.e., what happens when more than n constraints are active.

By Definition 2.3.6, we require n of the active constraints to be linearly independent. Therefore, when degeneracy is present in the basic solution, the active set is not full rank. In this case, we say we have a degenerate basic solution. When dealing with a linear program in standard equality form as in (2.8), then we get the special case of a degenerate basic solution.

Definition 2.3.9 (Degenerate basic solution)**.** *Consider the standard form polyhedron $P = \{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$ and let x be a basic solution. The vector x is a degenerate basic solution if more than $n - m$ components of x are zero.*

Furthermore, we say that x is a *degenerate basic feasible solution* if $x \in \mathbb{R}^n_+$ and more than $n - m$ components of x are zero.

A notion of degeneracy that has been recently established in [26, 27] is that Definition 2.3.9 can be separated into two cases of degenerate basic feasible solutions depending on whether strict feasibility fails or not. The first kind of degeneracy is observed when the linear program does not have a Slater point. When strict feasibility fails in linear programming, then every basic feasible solution is degenerate. We discuss this in more detail in Section 2.3.4.

2.3.3 Duality

We now describe duality in the linear programming setting with ideas from [7, Chapter 4] and Section 2.1.6. We refer to (2.8) as the *primal* ***LP***. The dual of (2.8) is derived from the dual functional as discussed in (2.4). Therefore, we have the following *dual* ***LP***

$$\begin{aligned} \min \quad & b^T y \\ \text{subject to} \quad & A^T y \geq c. \end{aligned} \tag{2.9}$$

As discussed in Section 2.1.6, any feasible solution of the dual **LP**, (2.9), is a lower bound for any feasible solution of the primal **LP**. Finally, we can guarantee strong duality holds without the existence of a Slater point in the primal **LP**.

Theorem 2.3.10 (Strong duality in linear programming)**.** *Suppose that the optimal value p^* for the primal* ***LP*** *is finite. Then the optimal value for the dual is d^*, and we have $d^* = p^*$. Furthermore, both values for the primal* ***LP*** *and dual* ***LP*** *are attained.*

2.3.4 Theorem of the Alternative

In this section, we examine the theorem of the alternative for the existence of a strictly feasible point for the primal and dual **LP**. An example of a way to quantitatively measure the extent that strict feasibility holds is by measuring the inconsistency of certain auxiliary systems with respect to the primal and dual **LP**. This measurement provides an estimate of the problem's stability. For

example, if a strictly feasible point does not exist in the primal **LP**, then for any tolerance ε one can find a perturbation of $\|\Delta b\| < \varepsilon$ such that the problem

$$\begin{array}{rl} \max & c^T x \\ \text{subject to} & Ax = b + \Delta b \\ & x \geq 0, \end{array}$$

is infeasible.

The following results for linear programming can be found in [7, 19, 50].

Theorem 2.3.11 (Separating hyperplane theorem)**.** *Let P and Q be nonempty convex sets such that $P \cap Q = \emptyset$, and either P or Q is compact. Then there exists a vector $c \in \mathbb{R}^n$ such that $\sup\{c^T x \,|\, x \in P\} < \inf\{c^T x \,|\, x \in Q\}$.*

We note that we are interested in a case of Theorem 2.3.11 where one set is a polyhedron, and the other a cone. Therefore, when we consider one of the sets to be a closed convex cone, the hyperplane separation theorem becomes the following theorem.

Theorem 2.3.12 (Homogeneous separation)**.** *Consider a nonempty closed convex set Q and a closed convex cone K with nonempty interior. Then exactly one of the following holds*

(1) The set Q intersects the interior of K.

(2) There exists a vector $0 \neq v \in K^*$ satisfying $\langle v, x\rangle \leq 0$ for all $x \in Q$.

Moreover, for any exposing vector v satisfying Item 2, the region $Q \cap K$ is contained in the proper face $v^\perp \cap K$.

We now discuss the main result of interest from Theorem 2.3.11 and Theorem 2.3.12, where the cone we consider is $K = \mathbb{R}^n_+$.

Theorem 2.3.13 (Theorem of the alternative for the primal)**.** *Consider the feasible polyhedron, $P = \{x \in \mathbb{R}^n \,|\, Ax = b, x \geq 0\}$, for the primal **LP**. Then exactly one of the following holds*

1. *The primal **LP** is strictly feasible, i.e., P has a Slater point.*
2. *The auxiliary system is consistent:*

$$0 \neq A^T y \geq 0 \qquad \text{and} \qquad b^T y \leq 0. \tag{2.10}$$

*Assuming that the primal **LP** is feasible, then the auxiliary system in (2.10) is equivalent to*

$$0 \neq A^T y \geq 0 \qquad \text{and} \qquad b^T y = 0. \tag{2.11}$$

Therefore, any exposing vector v satisfying either of the systems, (2.10) and (2.11), yields a proper face $(A^T y)^\perp \cap \mathbb{R}^n_+$ containing P.

In Algorithm 2.1 we describe a pseudocode for generating a linear program that fails strict feasibility in the primal **LP** problem. We note that we define the polyhedral constraints for the primal **LP** problem as $P := \{x \in \mathbb{R}^n \mid Ax = b,\, x \geq 0\}$, and for the dual **LP** problem $Q := \{(y, z) \in \mathbb{R}^{m\times n} \mid A^T y + z = c,\, z \geq 0\}$. We then make the following assumptions: $A \in \mathbb{R}^{m\times n}$, A is full-row rank, $n-r$ is the dimension of the relative interior of P, and $r < m < n$. We take advantage of the orthonormal columns of the QR decomposition to construct A_1 such that $A_1^T \hat{y} = 0$, and construct A_2 such that $A_2^T \hat{y} > 0$. We note that the **LP** constructed by Algorithm 2.1 does not fail dual strict feasibility by construction.

Algorithm 2.1 Generation of an **LP** where strict feasibility fails in the primal problem

Require: $m, n, r \in \mathbb{Z}$, $\hat{y}$, $y \in \mathbb{R}^m$, $\hat{v} \in \mathbb{R}^{n-r}_{++}$, $z \in \mathbb{R}^n_+$, $\hat{A}_1 \in \mathbb{R}^{(m-1)\times(n-r)}$, $A_2 \in \mathbb{R}^{m\times r}$.

1: **Output.** $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$
2: $[Q, \sim] = \text{qr}(\hat{y})$
3: $\tilde{A}_1 = Q_{(2:m)}$
4: $A_1 = \tilde{A}_1 \hat{A}_1$
5: inds = find$(A_2^T \hat{y} = 0)$
6: **while** length(inds) > 0 **do**
7: $\quad A_2$(:,inds) = randn(m,length(inds))
8: $\quad$ inds = find$(A_2^T \hat{y} = 0)$
9: **end while**
10: neginds = find$(A_2^T \hat{y} < 0)$
11: A_2(:,neginds) $= -A_2$(:,neginds)
12: $A = [A_1 A_2]$
13: $\hat{x} = \begin{pmatrix} \hat{v} \\ 0_r \end{pmatrix}$
14: $b = A\hat{x}$
15: $c = A^T y + z$

There are two alterations that can be made to Algorithm 2.1. If it is desirable to know the optimal solution for implementing warm-start strategies and guaranteeing dual strict feasibility, then we need to find a basic feasible solution to the primal linear system P using Phase 1 simplex; we will call this solution $\hat{x}$. We then construct $\hat{z}$ such that it satisfies strict complementarity with $\hat{x}$. It is then left to build c as we do in line 15 of Algorithm 2.1. Since we have constructed $\hat{z}$ such that strict complementarity is satisfied, we know the dual solution $(y, \hat{z})$ has exactly n active constraints in Q. Therefore, by Theorem 3.1 and Corollary 3.6 of [27], we have guaranteed the existence of a Slater point. It is important to note that if we want $\hat{z}$ to be a degenerate optimal dual solution, we simply include the desired degree of dual degeneracy and relax the condition that $\hat{z}$ needs to satisfy strict complementarity with $\hat{x}$, but there is no guarantee of dual strict feasibility by construction.

If we want to construct an **LP** that fails strict primal and strict dual feasibility simultaneously, we can build $\hat{z}$ such that it satisfies strict complementarity with $\hat{x}$ as defined in line 13. We have the following exposing vector d for the dual **LP**

$$0 \neq d \in \text{null}\left(\begin{bmatrix} A \\ \hat{z}^T \end{bmatrix}\right). \tag{2.12}$$

We will prove (2.12) in Remark 2.3.15 at the end of this section.

We now define the theorem of the alternative for the dual **LP**.

Theorem 2.3.14 (Theorem of the alternative for the dual)**.** *Consider the feasible polyhedron, $P_D = \{y \in \mathbb{R}^m \mid A^T y \leq c\}$, for the dual **LP**. Then exactly one of the following holds*

1. *The dual **LP** is strictly feasible.*
2. *The auxiliary system is consistent:*

$$0 \neq x \geq 0, \qquad Ax = 0, \qquad \text{and} \qquad c^T x \leq 0. \tag{2.13}$$

*Assuming that the dual **LP** is feasible, then the auxiliary system in (2.13) is equivalent to*

$$0 \neq x \geq 0, \qquad Ax = 0, \qquad \text{and} \qquad c^T x = 0. \tag{2.14}$$

Therefore, any exposing vector x satisfying either of the systems, (2.13) and (2.14), yields a proper face $x^\perp \cap \mathbb{R}^n_+$ containing P_D.

In Algorithm 2.2 we describe a pseudocode for generating a linear program that fails strict feasibility in the dual problem, and recall the definitions of P and Q. Then we make the following assumptions: $A^T \in \mathbb{R}^{n\times m}$, A^T is full row rank, $m - r$ is the dimension of the relative interior of Q, and $r < m < n$. We take advantage of the orthonormal columns of the QR decomposition to construct A such that $A\hat{x} = 0$, and construct c such that $c^T\hat{x} = 0$. We note that the **LP** constructed by Algorithm 2.2 does not fail primal strict feasibility by construction.

Algorithm 2.2 Generation of an **LP** where strict feasibility fails in the dual problem

Require: $m, n, r \in \mathbb{Z}$, $x \in \mathbb{R}^n_+$, $y \in \mathbb{R}^m$, $\hat{u} \in \mathbb{R}^r_{++}$, $\hat{v} \in \mathbb{R}^{n-r}_{++}$, $\hat{A} \in \mathbb{R}^{m\times n-1}$.

1: **Output.** $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

2: $\hat{x} = \begin{pmatrix} 0_{n-r} \\ \hat{u} \end{pmatrix}$

3: $[Q, \sim] = \mathrm{qr}(\hat{x})$

4: $\tilde{A} = Q_{(2:n)}$

5: $A = \hat{A}\tilde{A}^T$

6: $z = \begin{pmatrix} \hat{u} \\ 0_r \end{pmatrix}$

7: $c = A^T y + z$

8: $b = Ax$

If it is desirable to know the optimal solution for implementing warm-start strategies and guaranteeing primal strict feasibility, then we need to find a basic feasible solution to the dual linear system Q using Phase 1 simplex; we will call this solution $\hat{z}$. We then construct $\hat{x}$ such that it satisfies strict complementarity with $\hat{z}$. It is then left to build b as we do in line 8 of Algorithm 2.2. Since we have constructed $\hat{x}$ such that strict complementarity is satisfied, we know the primal solution $\hat{x}$ has exactly n active constraints in P. Then similar to our discussion above, by Theorem 3.1 and Corollary 3.6 of [27], we have guaranteed the existence of a Slater point. It

is important to note that if we want $\hat{x}$ to be a degenerate optimal primal solution, we simply include the desired degree of primal degeneracy and relax the condition that $\hat{x}$ needs to satisfy strict complementarity with $\hat{z}$, but there is no guarantee of primal strict feasibility by construction.

Remark 2.3.15. *Consider an **LP** constructed by Algorithm 2.1 such that $\hat{x}$ is defined in line 13, $\hat{z}$ satisfies strict complementarity with $\hat{x}$, and $c = A^T y + \hat{z}$. Then the exposing vector for the dual **LP** is*

$$0 \neq d \in \text{null}\left(\begin{bmatrix} A \\ \hat{z}^T \end{bmatrix}\right).$$

Proof. Consider $0 \neq d \in \text{null}\left(\begin{bmatrix} A \\ \hat{z}^T \end{bmatrix}\right)$ and recall that d is an exposing vector of the dual **LP** if it satisfies the auxiliary system (2.14) in Theorem 2.3.14. Then since $d \in \text{null}\left(\begin{bmatrix} A \\ \hat{z}^T \end{bmatrix}\right)$, it is also in the nullspace of A. Therefore, we have $Ad = 0$. It is left to show that $c^T d = 0$. Recall that $c = A^T y + \hat{z}$. Then

$$\begin{aligned} c^T d &= (A^T y + \hat{z})^T d \\ &= y^T A d + \hat{z}^T d \\ &= 0, \text{ by definition of } d. \end{aligned}$$

Thus, d satisfies the auxiliary system (2.14) in Theorem 2.3.14, and is therefore an exposing vector of the dual **LP**. □

2.4 Time Complexity

We now briefly define and describe the terminology on *time complexity* needed to discuss algorithmic efficiency. The definitions referenced in this section can be found in [9, C.1] and [47]. The cost of performing numerical linear algebra is usually expressed as the total number of *flops* or *floating-point operations* required to perform the computation. In this thesis, we define a flop as one addition, subtraction, multiplication, or division of two floating-point numbers. Therefore, to evaluate the complexity of an algorithm, we count the total number of flops an algorithm performs and express it as a function of the dimension of the vectors and matrices involved in the numerical linear algebra computations. The most common expressions of flop counts as a function are polynomial and exponential functions.

When discussing the efficiency of an algorithm from the perspective of flops and time complexity, we will be using big-O notation.

Definition 2.4.1 (big-O notation)**.** *Let f and g be functions such that $f, g : \mathcal{N} \to \mathbb{R}^p$. Then we say that $f(n) = O(g(n))$ if there exists positive integers c and n_0 such that for all $n \geq n_0$ we have*

$$f(n) \leq cg(n).$$

When $f(n) = O(g(n))$, we say that $g(n)$ is an asymptotic upper bound for $f(n)$.

In other words, if $f(n) = O(g(n))$, then f is less than or equal to g by a constant factor. To better illustrate Definition 2.4.1 we provide an example of the relationship between total flops and big-O notation from numerical linear algebra.

Example 2.4.2 (Matrix-vector multiplication)**.** *Let $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m\times n}$. Then the matrix-vector multiplication of Ax takes on the form*

$$Ax = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n \\ A_{21}x_1 + A_{22}x_2 + \cdots + A_{2n}x_n \\ \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n \end{bmatrix}.$$

We notice that $A_{(1,:)}x$ is an inner product of two vectors in $\mathbb{R}^n$, and comprises of n multiplications and $n-1$ additions for a total of $2n-1$ flops. Therefore, the matrix-vector multiplication of Ax takes $m(2n-1) = 2mn - m$ flops which is equivalent to $O(mn)$ in big-O notation. This is because $2mn - m \leq 2mn \leq cmn$ for $c \geq 2$.

When the total flop count of an algorithm can be expressed or bounded above by a polynomial, we consider that algorithm to be efficient. In other words, if $f : \mathcal{N} \to \mathbb{R}^p$ is a function of the number of flops of an algorithm, and $f(n) \leq cg(n)$ where g is a polynomial, then we consider the algorithm to be efficient. It is also common to call an algorithm that satisfies the conditions described as a *polynomial time algorithm* or *efficient algorithm*.

Chapter 3

Solving the BAP with a Regularized Nonsmooth Newton Method, RNNM

We now discuss the **BAP** and solving it in more detail. We start by explicitly defining the constraint space and using the theory developed in Section 2.1 and Section 2.2 to take advantage of our problem. Recall the **BAP** as defined in (1.1), then we define the **BAP** constrained to the polyhedral set in standard form. Suppose we are given $A \in \mathbb{R}^{m\times n}$, A is full-row and no columns of A are 0, $b \in \mathbb{R}^m$, and $v \in \mathbb{R}^n$. Then we define the projection onto the polyhedral set intersected with the nonnegative cone as

$$x^*(v) := \begin{array}{ll} \operatorname{argmin}_x & \frac{1}{2}\|x-v\|^2 \\ \text{subject to} & Ax = b \\ & x \in \mathbb{R}^n_+. \end{array} \tag{3.1}$$

In other words, $x^*(v)$ is the unique optimal solution of (3.1), and $p^*(v) := \frac{1}{2}\|x^*(v)-v\|^2$ is the optimal value. In Section 3.1 we discuss the theory necessary to take advantage of the polyhedral structure of (3.1). These results are from [12] and motivate the use of nonsmooth newton methods to solve the **BAP** with polyhedral constraints.

In Section 3.1 we develop the theory of the **BAP** that can be taken advantage of using a Moreau decomposition. In Section 3.2 we examine the properties of the Gauss-Newton descent direction, and how we can use them to derive the descent properties of certain modified Levenberg-Marquardt regularizations of the Jacobian. In Section 3.3 we derive the Jacobian of our nonsmooth Newton direction and use an optimal diagonal scaling that provides the generalized Jacobian matrix, $V \in \partial F(y)$ with the smallest ω condition number. In Section 3.4 we provide the theory needed to produce a known optimal solution of the **BAP** for our numerical testing. In Section 3.5 we derive the Moreau decomposition for the case when (3.1) includes free variables in the set of decision variables.

3.1 Basic theory of the BAP

We now discuss some theory of (3.1) which can be used to motivate the use of generalized Newton methods. First, we assume that the problem in (3.1) has a nonempty feasible set with a Slater point, i.e.,

$$P := \{x \in \mathbb{R}^n_+ \mid Ax = b\} \neq \emptyset \text{ and } \exists x \in \mathbb{R}^n_{++} \cap P.$$

By assuming the above, we can conclude that the objective function of (3.1) is strictly convex, coercive, and has a nonempty domain. Therefore, by Proposition 2.1.34 the optimal solution is uniquely attained, and the optimal value finite. Furthermore, the constraint set is nonempty, closed, and convex. Therefore, strong duality holds. Throughout the rest of this chapter, we will refer to the functions $F(y)$ and $f(y)$ as follows

$$F(y) := A(v + A^T y)_+ - b, \qquad f(y) := \frac{1}{2} \|F(y)\|^2 . \tag{3.2}$$

Theorem 3.1.1. *Consider the generalized simplex best approximation problem (3.1) with optimal solution and primal optimal value $x^*(v)$ and $p^*(v)$, respectively. Then the following hold:*

(1) The optimal solution $x^*(v)$ exists and is unique. Moreover, strong duality holds and the dual problem of (3.1) is the maximization of the dual functional, $\phi(y, z)$:

$$p^*(v) = d^*(v) := \max_{\substack{z \in \mathbb{R}^n_+ \\ y \in \mathbb{R}^m}} \phi(y, z) := -\frac{1}{2} \left\|z + A^T y\right\|^2 + y^T(Av - b) - z^T v.$$

(2) Let $y \in \mathbb{R}^m$. Then

$$F(y) = 0 \iff y \in \operatorname*{argmin}_u f(u) \text{ and } x^*(v) = (v + A^T y)_+. \tag{3.3}$$

Proof. Recall the definition of the Lagrangian in (2.3), then the Lagrangian for (3.1) is

$$L(x, y, z) := \frac{1}{2} \|x - v\|^2 + y^T(b - Ax) - z^T x.$$

(1) Since the solution of (3.1) is a projection onto a nonempty closed convex set, then the optimal solution exists and is unique by Remark 2.1.30. Furthermore, strong duality holds as the duality gap is zero and the dual is attained.

We then find the stationary point of the Lagrangian to be

$$\nabla_x L(x, y, z) = 0 \iff x - v + A^T y - z = 0 \iff x = v + A^T y - z.$$

Therefore, at its stationary point the Lagrangian is

$$\begin{aligned} L(x, y, z) &= \tfrac{1}{2} \left\|v + A^T y + z - v\right\|^2 + y^T(b - A(v + A^T y + z)) - z^T(v + A^T y + z) \\ &= \tfrac{1}{2} \left\|A^T y + z\right\|^2 + y^T b - y^T A v - (A^T y)^T (A^T y + z) - z^T v - z^T (A^T y + z) \\ &= \tfrac{1}{2} \left\|A^T y + z\right\|^2 + y^T b - y^T A v - (A^T y + z)^T (A^T y + z) - z^T v \\ &= -\tfrac{1}{2} \left\|z + A^T y\right\|^2 + y^T(b - Av) - z^T v. \end{aligned}$$

Therefore, the best lower bound of the dual functional, $g(y, z)$, is

$$\begin{array}{rcll} d^* & = & \max_{y\in\mathbb{R}^m, z\in\mathbb{R}^n_+} \min_{x\in\mathbb{R}^n_+} & L(x,y,z) \quad (= \frac{1}{2}\|x-v\|^2 + y^T(b-Ax) - z^Tx) \\ & = & \max_{x\in\mathbb{R}^n_+, y\in\mathbb{R}^m, z\in\mathbb{R}^n_+} & \{L(x,y,z) \,|\, \nabla_x L(x,y,z) = 0\} \\ & = & \max_{x\in\mathbb{R}^n_+, y\in\mathbb{R}^m, z\in\mathbb{R}^n_+} & \{L(x,y,z) \,|\, x = v + A^Ty + z\} \\ & = & \max_{y\in\mathbb{R}^m, z\in\mathbb{R}^n_+} & -\frac{1}{2}\left\|z + A^Ty\right\|^2 + y^T(b - Av) - z^Tv. \end{array}$$

Moreover, $p^* := p^*(v) = d^* := d^*(v)$, and the dual value is attained.

(2) For sufficiency, we examine the KKT optimality conditions for the primal-dual variables (x, y, z)

$$\begin{array}{ll} \nabla_x L(x,y,z) = x - v - A^Ty - z = 0,\ z \in \mathbb{R}^n_+ & \text{(dual feasibility)} \\ \nabla_y L(x,y,z) = Ax - b = 0,\ x \in \mathbb{R}^n_+,, & \text{(primal feasibility)} \\ \nabla_z L(x,y,z) \equiv x \in (\mathbb{R}^n_+ - z)^\circ. & \text{(complementary slackness)} \end{array}$$

The above KKT conditions can be rewritten as the following:

$$\begin{pmatrix} x - v - A^Ty - z \\ Ax - b \\ z^Tx \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \qquad x, z \in \mathbb{R}^n_+,\ y \in \mathbb{R}^m. \tag{3.4}$$

It follows from the dual feasibility condition that $x - v - A^Ty - z = 0 \iff v + A^Ty = x + (-z)$. Consider this equality with the complementary slackness condition, then we have

$$x^Tz = 0,\ x, z \in \mathbb{R}^n_+,\ -z \in \mathbb{R}^n_- = (\mathbb{R}^n_+)^\circ,$$

and we learn that $x - z$ forms the Moreau decomposition of $v + A^Ty$. That is,

$$x = (v + A^Ty)_+ \text{ and } -z = (v + A^Ty)_- \iff z = -(v + A^Ty)_-.$$

Substituting $x = (v + A^Ty)_+$ into (3.4) we obtain the following simplification of the optimality conditions

$$x = (v+A^Ty)_+ \text{ and } A(v+A^Ty)_+ = b \implies z = -(v+A^Ty)_-,\ x^Tz = 0,\ x, z \in \mathbb{R}^n_+,\ x - v - A^Ty - z = 0.$$

Equivalently, $F(y) = 0$ for some $y \in \mathbb{R}^m$.

For necessity, let $y \in \mathbb{R}^m$ be given and suppose that $F(y) = 0$. Let $\bar{x} = (v + A^Ty)_+$, i.e., $\bar{x}$ is primal feasible. Let $\bar{z} = -(v+A^Ty)_-$. Then we get nonnegative feasibility for the dual variable $\bar{z}$ and complementary slackness. In other words, we have $\bar{z} \geq 0$ and $\bar{x}^T\bar{z} = 0$. Then from the Moreau decomposition we have the following equation

$$(v + A^Ty) = \bar{x} - \bar{z} \iff \bar{x} - v - A^Ty - \bar{z} = 0.$$

In other words, dual feasibility holds. Thus the KKT conditions are satisfied and we have proven necessity. The KKT conditions now imply that $\bar{x}(v)$ is optimal. Moreover, $F(y) = 0$ implies that $y \in \operatorname{argmin}_u f(u)$. Therefore, y solves the nonlinear least squares problem.

□

3.2 Nonlinear Least Squares

The **BAP** as described in (3.1) is equivalent to the minimization of $f(y)$ as defined in (3.2). In other words, it is equivalent to solving a nonlinear least squares problem where the projection is the source of nonlinearity.

This system can be recharacterized by introducing the potentially nonsmooth projection of a vector x onto the nonnegative, respectively nonpositive, orthant, i.e., x_+ and x_- respectively. Note that in the differentiable case, the gradient of the squared residual $f(y)$ in (3.2) is

$$\nabla f(y) = J(y)^T F(y),$$

where $J(y)$ is the Jacobian matrix. We note that we have differentiability of the function $P_+(w) := w_+$ if and only if $\{i \,|\, w_i = 0\} = \emptyset$. This is equivalent to saying $w - w_+$ is in the relative interior of the normal cone of $\mathbb{R}^n_+$ at w_+, see [44, Page 7], [20].

In the differentiable case, the Gauss-Newton direction is the solution of the (consistent) Gauss-Newton equation and can be found in [40]

$$J(y)^T J(y)\Delta y = -J(y)^T F(y). \tag{3.5}$$

Recall that $A^\dagger$ denotes the Moore-Penrose inverse of A, then solving for the best least squares solution Δy in (3.5) yields

$$\Delta y = -J(y)^\dagger F(y).$$

Therefore, the directional derivative of f in the direction Δy satisfies the following inequality

$$\begin{aligned} \Delta y^T \nabla f(y) &= (J(y)^\dagger F(y))^T(-(J(y)^T F(y))) \\ &= -\left\| P_{\text{range}(J(y)^T)} F(y) \right\|^2, \text{ where } P \text{ is the orthogonal projection} \\ &< 0, \qquad \text{if } F(y) \notin \text{null}(J(y)^T). \end{aligned} \tag{3.6}$$

We conclude in the differentiable case that the Gauss-Newton direction Δy is a descent direction when $F(y) \neq 0$. The *Levenberg-Marquardt,* ***LM*** , method is a popular and cheap way to handle the singularity of $J(y)$. To perform the **LM** regularization we do the following substitution for $\lambda > 0$

$$J(y)^T J(y) \leftarrow J(y)^T J(y) + \lambda I.$$

We can see that we maintain a descent direction with a similar approach if the assumption in (3.7) holds. Furthermore, this approach avoids the product $J(y)^T J(y)$. By avoiding the product $J(y)^T J(y)$, we do not unnecessarily introduce ill-conditioning nor do we lose sparsity.

Lemma 3.2.1. *Consider the nonlinear least squares problem in* (3.2)*. Let* $y \in \mathbb{R}^m$*, with* F *differentiable at* y*. Let* $\lambda > 0$ *and let* Δy *be the unique solution of*

$$(J(y) + \lambda I)\Delta y = -F(y).$$

*Then $J(y)$ is positive semidefinite, and Δy is the simplified **LM** direction and is a descent direction if and only if*

$$F(y) \neq 0. \tag{3.7}$$

Proof. By the feasibility and full-row rank assumptions for (3.1), we conclude that $0 = \min_y f(y)$ and that the assumption satisfies

$$F(y) \neq 0 \iff JF(y) \neq 0. \tag{3.8}$$

We observe that for our application, J is symmetric positive semidefinite from (3.11) and (3.12). Let $J = UDU^T$ denote the orthogonal spectral decomposition. The simplified regularization of **LM** uses $(J + \lambda I)\Delta y = -F$. Therefore, we get

$$\Delta y = -(J + \lambda I)^{-1}F = -U(D + \lambda I)^{-1}U^T F.$$

Therefore, the directional derivative of f at y in the direction of Δy is

$$\begin{aligned}
\Delta y^T \nabla f(y) &= -\left(U(D + \lambda I)^{-1}U^T F\right)^T (UDU^T F) \\
&= -(U^T F)^T (D + \lambda I)^{-1} D(U^T f) \\
&= -(U^T F)^T D^{\frac{1}{2}} (D + \lambda I)^{-1} D^{\frac{1}{2}} (U^T F) \\
&= -(D^{\frac{1}{2}}(U^T F))^T (D + \lambda I)^{-1} D^{\frac{1}{2}} (U^T F) \\
&< 0 \iff (D^{\frac{1}{2}} U^T) F \neq 0.
\end{aligned}$$

By (3.8), the latter is not zero if and only if (3.7) holds. This completes the proof. □

3.3 Well Conditioned Generalized Jacobian

Recall that earlier we denoted the orthogonal projection operator onto the nonnegative orthant as $P_+ w = w_+$. Further recall the optimal conditions derived in (3.4). Then we can make the following observation on where the projection is differentiable

$$Aw_+ = A(P_+ w) = (AP_+)w_+ = (AP_+)(P_+ w) = \sum_{w_i > 0} w_i A_i.$$

Therefore, the columns of A correspond to the positive variables of w at points where the projection is differentiable. We note that

$$v + A^T y > 0 \implies J(\Delta y) = AIA^T \Delta y = AA^T \Delta y. \tag{3.9}$$

We now define the following 3 index sets, $\mathcal{I}_+$, $\mathcal{I}_0$, $\mathcal{I}_-$, respectively

$$\mathcal{I}_{+,0,-} := \mathcal{I}_{+,0,-}(y) = \{i \,|\, (v + A^T y)(>, =, <)\, 0\}.$$

Then for $\Delta y > 0$ we have

$$\begin{aligned} F(y+\Delta y)-F(y) &= A(v+A^T(y+\Delta y))_+ - A(v+A^Ty)_+ \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y+\Delta y)}(v+A^T(y+\Delta y))_iA_i - \sum_{i\in\mathcal{I}_+(y)}(v+A^Ty)_iA_i \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y)}(v+A^T(y+\Delta y))_iA_i + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)}(v+A^T(y+\Delta y))_iA_i \\ &\quad + \textstyle\sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_-(y)}(v+A^T(y+\Delta y))_iA_i - \sum_{i\in\mathcal{I}_+(y)}(v+A^Ty)_iA_i. \end{aligned} \tag{3.10}$$

In other words, from (3.10) we have three cases when separating the summation, all involving whether $i \in \mathcal{I}_+(y)$, $i \in \mathcal{I}_0(y)$, or $i \in \mathcal{I}_-(y)$. Therefore, we choose sufficiently small $\Delta y > 0$ such that for $i \in \mathcal{I}_-(y)$, then $i \in \mathcal{I}_0(y+\Delta y) \cap \mathcal{I}_-(y)$. With sufficiently small Δy, (3.10) simplifies as $\sum_{i\in\mathcal{I}_0(y+\Delta y)\cap\mathcal{I}_-(y)}(v+A^T(y+\Delta y))_iA_i = 0$. Therefore, using the results from (3.9) and the definition of $\mathcal{I}_{+,0,-}$ we have

$$\begin{aligned} F(y+\Delta y)-F(y) &= \textstyle\sum_{i\in\mathcal{I}_+(y)}(v+A^T(y+\Delta y))_iA_i + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)}(v+A^T(y+\Delta y))_iA_i \\ &\quad - \textstyle\sum_{i\in\mathcal{I}_+(y)}(v+A^Ty)_iA_i \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y)}(A^T(\Delta y))_iA_i + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)}(v+A^T(y+\Delta y))_iA_i \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y)} A_iA_i^T\Delta y + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)}(v+A^T(y+\Delta y))_iA_i \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y)} A_iA_i^T\Delta y + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)}(A^T\Delta y)_iA_i \\ &= \textstyle\sum_{i\in\mathcal{I}_+(y)} A_iA_i^T\Delta y + \sum_{i\in\mathcal{I}_+(y+\Delta y)\cap\mathcal{I}_0(y)} A_iA_i^T\Delta y. \end{aligned}$$

Then, the first summation is over the fixed index set $\mathcal{I}_+(y)$, while the second summation is dependent on $(A^T\Delta y) > 0$. Suppose that $A^T_{\mathcal{I}_0}\Delta y = e_i$ is consistent for each $i \in \mathcal{I}_0$. Then we can choose whether or not to add the corresponding column i to the generalized Jacobian. In other words, we only need a maximal linearly independent subset of the columns of $A_{\mathcal{I}_0}$. Let $\bar{\mathcal{I}}_0 \subseteq \mathcal{I}_0$ be a maximum linearly independent subset.

We consider the subgradient that follows [25] with the change being we use a maximum linearly independent subset of $\mathcal{I}_0$ which we compute using a QR decomposition as done in the MATLAB code *licols* [1]

$$\mathcal{U}(y) := \left\{ u \in \mathbb{R}^n \,|\, u_i \in \begin{cases} \{1\}, & \text{if } i \in \mathcal{I}_+, \\ [0,1], & \text{if } i \in \bar{\mathcal{I}}_0, \\ \{0\}, & \text{if } i \in \mathcal{I}_- \cup (\mathcal{I}_0 \backslash \bar{\mathcal{I}}_0). \end{cases} \right\}. \tag{3.11}$$

Then the generalized Jacobian of the nonlinear system at $y \in \mathbb{R}^m$ is given by the set

$$\partial F(y) = \{A \text{ Diag}(u)\, A^T \,|\, u \in \mathcal{U}(y)\}. \tag{3.12}$$

Let $y_0 \in \mathbb{R}^m$, then the nonsmooth Newton method for solving $F(y) = 0$ consists of the following iterative process

$$y^{k+1} = y^k - V_k^{-1}F(y^k),\ V_k \in \partial F(y^k), \tag{3.13}$$

[1] The code for licols can be found at https://www.mathworks.com/matlabcentral/fileexchange/77437-extract-linearly-independent-subset-of-matrix-columns and was submitted by the user Matt J

where V_k is a generalized Jacobian matrix taken from the generalized Jacobian $\partial F(y^k)$. We note that defining $M = Diag(u)$ with $u \in \mathcal{U}(y)$ gives us

$$AMA^T = \sum_{i \in \mathcal{I}_+ \cup \bar{\mathcal{I}}_0} u_i A_i A_i^T, \quad u_i = 1 \text{ for } i \in \mathcal{I}_+,\ u_i \in [0,1] \text{ for } i \in \bar{\mathcal{I}}_0. \tag{3.14}$$

We note that for positive diagonal M, and a rectangular matrix B, the ranks of B, BM, $(BM)(BM)^T$ are all the same.

Remark 3.3.1. *Since we have freedom in choosing the values $u_i \in [0,1]$ for $i \in \bar{\mathcal{I}}_0$, we follow the optimal diagonal scaling in [16, Prop. 2.1(v)], [28, Thm. 5.2] to minimize the ω condition number, as defined in [15]. We then choose the generalized Jacobian by setting*

$$u_i = \min\{1, 1/\|A_i\|^2\},\ \forall i \in \bar{\mathcal{I}}_0.$$

Therefore, the generalized Jacobian matrix we choose is nonsingular if and only if $A_{\mathcal{I}_+ \cup \mathcal{I}_0}$ is full rank m. Moreover, for large problems we expect $\|A_i\| > 1$, and therefore $u_i < 1$. This goes against the intuitive choice of making u_i as large as possible, or in this case equal to one. Note that all elements of $\partial F(y)$ are invertible if and only if $A_{\mathcal{I}_+}$ is invertible. Furthermore, there exists an invertible element if and only if $A_{\mathcal{I}_+ \cup \mathcal{I}_0}$ is full rank m.

We finish this section with the addition of the pseudocode for the regularized nonsmooth Newton method, **RNNM**, in Algorithm 3.1. We also include the pseudocode for **RNNM** when the linear system is solved inexactly in Algorithm A.1. We note that **RNNM** is a part of a class of semismooth Newton algorithms. However, since we do not use any semismooth theory when developing **RNNM** we have opted to omit semismooth theory from our discussion. For the missing results on semismooth Newton methods see [21, 41] and the references therein.

Algorithm 3.1 **BAP** of v for constraints $Ax = b, x \geq 0$; exact Newton direction

Require: $v \in \mathbb{R}^n, y_0 \in \mathbb{R}^m$, $(A \in \mathbb{R}^{m \times n}, \text{rank}(A) = m)$, $b \in \mathbb{R}^m$, $\varepsilon > 0$, maxiter $\in \mathbb{N}$.

Output. Primal-dual opt.: $x_{k+1}, (y_{k+1}, z_{k+1})$

Initialization. $k \leftarrow 0$, $x_0 \leftarrow (v + A^T y_0)_+$, $z_0 \leftarrow (x_0 - (v + A^T y_0))_+$,
$F_0 = Ax_0 - b$, stopcrit $\leftarrow \|F_0\| / (1 + \|b\|)$

while $((\text{stopcrit} > \varepsilon)\,\&\,(k \leq \text{maxiter}))$ **do**
  $V_k = \sum_{i \in \mathcal{I}_+} A_i A_i^T + \sum_{i \in \bar{\mathcal{I}}_0} \frac{1}{\|A_i\|^2} A_i A_i^T$
  $\lambda = \min(1e^{-3},\ \text{stopcrit})$
  $\bar{V} = (V_k + \lambda I_m)$
  solve pos. def. system $\bar{V} d = -F_k$ for Newton direction d
  updates
    $y_{k+1} \leftarrow y_k + d$
    $x_{k+1} \leftarrow (v + A^T y_{k+1})_+$
    $z_{k+1} \leftarrow (x_{k+1} - (v + A^T y_k))_+$
    $F_{k+1} \leftarrow Ax_{k+1} - b$ (residual)
    stopcrit $\leftarrow \|F_{k+1}\| / (1 + \|b\|)$
    $k \leftarrow k + 1$
end while

3.4 Vertices and Dual Cones

In the numerical tests in Chapter 4 we can decide on the characteristics of the optimal solution using the properties of vertices of a polyhedron P. Recall the definition of a vertex from Definition 2.3.2, and of the dual cone Definition 2.1.21. Then we have the following characterizations with respect to the index sets $\mathcal{I}_+$, $\mathcal{I}_0$, $\mathcal{I}_-$.

Lemma 3.4.1 (Vertex and dual cone)**.** *Suppose that $x(y) = (v + A^T y)_+ \in P$, where $y \in \mathbb{R}^m$. Then the following are equivalent:*

(1) $x(y)$ is a vertex of P.

(2) $A_{\mathcal{I}_+(y)}$ is full column rank.

(3) $\begin{bmatrix} A_{\mathcal{I}_+} & A_{\mathcal{I}_0 \cup \mathcal{I}_-} \\ 0 & I_{\mathcal{I}_0 \cup \mathcal{I}_-} \end{bmatrix}$ is full column rank n.

Moreover:

(a) the corresponding generalized Jacobian in (3.14) and Remark 3.3.1, is nonsingular if $x(y)$ is a nondegenerate vertex.

(b) the dual cone of the feasible set P at $x = x(y)$ is

$$(P - x)^\circ = \{w \,|\, w = A^T u + z,\ u \in \mathbb{R}^m,\ z \in \mathbb{R}^n_+,\ x^T z = 0\}. \tag{3.15}$$

Proof. Without loss of generality we can permute the columns of A and corresponding components of x and have $A = \begin{bmatrix} A_{\mathcal{I}_+} & A_{\mathcal{I}_0} & A_{\mathcal{I}_-} \end{bmatrix}$. We know from Theorem 2.3.7 that a vertex is equivalently an extreme point and a basic feasible solution. Therefore, $x(y)$ is a vertex if and only if $A_{\mathcal{I}_+}$ is a valid basis matrix if and only if the active set being full rank n. Then the active set of constraints is

$$\begin{bmatrix} A_{\mathcal{I}_+} & A_{\mathcal{I}_0 \cup \mathcal{I}_-} \\ 0 & I_{\mathcal{I}_0 \cup \mathcal{I}_-} \end{bmatrix} x = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{3.16}$$

This has the unique solution $x(y)$ if and only if $A_{\mathcal{I}_+}$ is full column rank. This shows the three equivalences items 1 to 3, as well as the nonsingularity of the generalized Jacobian that we choose as claimed in item a.

From the optimality conditions we have that the gradient of the objective satisfies

$$x - v = A^T y + \sum_{j \in \mathcal{I}_0 \cup \mathcal{I}_-} z_j e_j,$$

where e_j is the j-th unit vector. Furthermore, we know that $x - v$ is in the dual cone at x if and only if x is optimal. Therefore, this yields the description of the polar cone at x as claimed in item b. □

Remark 3.4.2 (Degeneracy of optimal solutions)**.** *Let x be a boundary point of P. Then the polar cone of P at x is given in (3.15). Moreover, x is the optimal solution of (3.1) if, and only if, $x - v \in (P - x)^\circ$. In other words, we can choose v such that*

$$v = x - A^T u - z,\ z \geq 0,\ z^T x = 0.$$

Furthermore, we can choose z such that $x + z > 0$ and have no degeneracy, or we can choose $z_i = 0$, $i \in \{1, \ldots, n\}$ and increase the degeneracy. For these choices we still get x optimal. As mentioned above, it is shown in [20] that

$$x^*(v) \text{ is differentiable at } \bar{v} \iff (x^*(\bar{v}) - \bar{v}) \in \operatorname{relint}(P - x^*(\bar{v}))^\circ,$$

This justifies our use of the Levenberg-Marquardt regularization.

3.5 Projection and Free variables

We now consider the application where some of the variables in the **BAP** are free.

3.5.1 Projection with Free Variables

Consider the following problem where some of the variables are free

$$\begin{array}{rrlll} & x(v) := & \operatorname{argmin}_{x_1,x_2} & \frac{1}{2}\|x - v\|^2, \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \\ \text{(P)} & & \text{subject to} & Ax = b \in \mathbb{R}^m \\ & & & x_1 \in \mathbb{R}^{n_1}_+,\ x_2 \in \mathbb{R}^{n_2}, \\ \text{optimal value: } p^*_f(v) & = & & \frac{1}{2}\|x(v) - v\|^2, \end{array} \tag{3.17}$$

Theorem 3.5.1. *Consider the **BAP** with free variables (3.17) and assume the feasible set is nonempty. Then the optimum $x(v)$ exists and is unique. Moreover, let*

$$F_f(y) := A \begin{pmatrix} \big((v + A^T y)_1\big)_+ \\ (v + A^T y)_2 \end{pmatrix} - b, \quad f_f(y) = \frac{1}{2}\|F_f(y)\|^2. \tag{3.18}$$

Then $F_f(y) = 0 \iff y \in \operatorname{argmin} f_f(y)$, and

$$x(v) = \begin{pmatrix} \big((v + A^T y)_1\big)_+ \\ (v + A^T y)_2 \end{pmatrix}, \text{ for any root } F_f(y) = 0. \tag{3.19}$$

Let the primal optimal value be denoted by $p^*_f(v) = \frac{1}{2}\|x(v) - v\|^2$. Then strong duality holds and the dual problem is the following maximization problem:

$$p_f^*(v) = d_f^*(v) := \max_{z_1 \in \mathbb{R}^{n1}_+, y \in \mathbb{R}^m} \phi_f(y, z_1) := -\frac{1}{2} \left\| \begin{pmatrix} z_1 \\ 0 \end{pmatrix} - A^T y \right\|^2 + y^T(Av - b) - z_1^T v_1.$$

Proof. To prove Theorem 3.5.1, we modify the proof of Theorem 3.1.1. Therefore, we consider the *Lagrangian,* $L_f(x, y, z)$, for (3.17)

$$L_f(x, y, z) = \frac{1}{2} \|x - v\|^2 + y^T(b - Ax) - z_1^T x_1, \quad \nabla_x L_f(x, y, z) = x - v - A^T y - \begin{pmatrix} z_1 \\ 0 \end{pmatrix}. \tag{3.20}$$

We see that solving for a stationary point gives us the following

$$\nabla_x L_f(x, y, z) = 0 \implies x - v - A^T y - z = 0, \quad z = \begin{pmatrix} z_1 \\ 0 \end{pmatrix}.$$

Therefore, we have $x = v + A^T y + z$ at its stationary point and the definition of z given above. Evaluating the Lagrangian at its stationary point gives us the following

$$\begin{array}{rcl} L_f(x, y, z) & = & \frac{1}{2} \left\|v + A^T y + z - v\right\|^2 + y^T(b - A(v + A^T y + z)) - z^T(v + A^T y + z) \\ & = & \frac{1}{2} \left\|A^T y + z\right\|^2 + y^T b - y^T A v - (A^T y)^T (A^T y + z) - z^T v - z^T (A^T y + z) \\ & = & \frac{1}{2} \left\|A^T y + z\right\|^2 + y^T b - y^T A v - (A^T y + z)^T (A^T y + z) - z^T v \\ & = & -\frac{1}{2} \left\|z + A^T y\right\|^2 + y^T(b - Av) - z^T v. \end{array}$$

Similar to Theorem 3.1.1, since the solution of (3.17) is a projection onto a nonempty closed convex set, the optimum exists and is unique by Remark 2.1.30. Therefore, strong duality holds as the duality gap is zero and the dual is attained. Thus, the Lagrangian dual is

$$\begin{array}{rcll} d^* & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y} \min_{x_1 \in \mathbb{R}^{n_1}_+, x_2} & L_f(x, y, z) = \frac{1}{2} \|x - v\|^2 + y^T(b - Ax) - z_1^T x_1 \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y, x_1 \in \mathbb{R}^{n_1}_+, x_2} & \{L_f(x, y, z_1) \,|\, \nabla_x L_f(x, y, z_1) = 0\} \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y, x_1 \in \mathbb{R}^{n_1}_+, x_2} & \{L_f(x, y, z) \,|\, x = v + A^T y + z\} \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y} & -\frac{1}{2} \left\|z + A^T y\right\|^2 + y^T(b - Av) - z^T v. \end{array}$$

Therefore, we derive the *KKT optimality conditions* for the primal dual variables (x, y, z) with $z = \begin{pmatrix} z_1 \\ 0 \end{pmatrix}, x_1 \geq 0, z_1 \geq 0$, as follows

$$\begin{array}{ll} \nabla_x L_f(x, y, z) = x - v - A^T y - z = 0, & \text{(dual feasibility)} \\ \nabla_y L_f(x, y, z) = Ax - b = 0, & \text{(primal feasibility)} \\ \nabla_z L_f(x, y, z) \ni x \in (\mathbb{R}^n_+ - z)^\circ. & \text{(complementary slackness } z_1^T x_1 = 0) \end{array}$$

The standard KKT optimality conditions for primal-dual variables (x, y, z) can be rewritten as:

$$\begin{pmatrix} x - v - A^T y - z \\ Ax - b \\ z^T x \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad x_1, z_1 \in \mathbb{R}^{n_1}_+, y \in \mathbb{R}^m, \; z = \begin{pmatrix} z_1 \\ 0 \end{pmatrix}.$$

Note $v + A^T y = x - z = x + (-z)$. Therefore, since $z_1 \geq 0$ and $z = \begin{pmatrix} z_1 \\ 0 \end{pmatrix}$ implies $z \in \mathbb{R}^n_+$, this is a Moreau decomposition of $v + A^T y$, with $x^T z = 0, x, z \in \mathbb{R}^n_+$, $x = (v + A^T y)_+$. Therefore, we get $A(v + A^T y)_{+_f} = b$, where the definition of $+_f$ is a modification of $+$ such that we project only the first part corresponding to x_1 onto the nonnegative orthant $\mathbb{R}^{n_1}_+$, it follows that $z_1 = -\left((v + A^T y)_1\right)_-$.

Then the optimality conditions

$$A \begin{pmatrix} \left((v + A^T y)_1\right)_{+_f} \\ (v + A^T y)_2 \end{pmatrix} = b, \; x_1 = \left((v + A^T y)_1\right)_+, x_2 = (v + A^T y)_2$$

imply that

$$z = -(v + A^T y)_-, \; z^T x = 0, x, \; z \in \mathbb{R}^n_+, x - v - A^T y - z = 0.$$

In other words, $F_f(y) = 0$ for some $y \in \mathbb{R}^m$. □

Recall that for a basic feasible solution we need n active constraints. The equality constraints $Ax = b$ account for m active constraints, leaving $n - m$ constraints needing to be active. These $n - m$ constraints will be from the indices $1, 2, \ldots, n_1$, i.e., referring to the constrained variables in x_1. We now determine how many basic variables there will be in (3.17)

$$m_1 = n_1 - (n - m) = m - (n - n_1) = m - n_2 \implies m_1 = m - n_2, \text{ basic variables.}$$

Chapter 4

Numerics for Best Approximation Problem

We now compare the *Regularized Nonsmooth Newton Method, (**RNNM**)*, with the *Halpern-Lions-Wittmann-Bauschke*, (**HLWB**) [2], as described in Section 4.1, MATLAB's *lsqlin* interior point solver, and the *quadratic programming proximal augmented Lagrangian method*, (**QPPAL**) [35], as described in Section 4.2. When making these comparisons, we will be using exact and inexact methods for solving a system of linear equations.

We show in our experiments that **RNNM** (exact) significantly outperforms the other methods. These experiments are performed with an i7-4930k @ 3.2GHz, and 16 GBs of RAM. The software used is MATLAB 2022b and Mosek 9.1.9.

4.1 The Halpern-Lions-Wittmann-Bauschke method, HLWB

We explicitly define the projection operator as in Definition 2.1.29 with respect to the polyhedral set $Q := \{x \in \mathbb{R}^n \mid Ax \leq b\}$ with $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. We note that Q can also be equivalently described as an intersection of half-spaces. We denote $P_Q(v)$ to be the orthogonal projection of v onto the polyhedral set Q. As this does not have an explicit representation for an intersection of half-spaces larger than 2 (see [5]), we examine the orthogonal projection onto each individual half-space.

$$P_i(v) = v + \min\left\{0, \frac{b_i - A_{(i,:)}x}{\left\|A_{(i,:)}\right\|^2}\right\} A_{(i,:)}. \tag{4.1}$$

Since there are convex sets C that do not have explicitly defined projection operators, such as Q, the invention of methods that use projection operators of a subset of C to project a point v onto C is a growing field of interest. The methods that employ projection operators such as (4.1) are called *projection methods*. The **HLWB** method for linear inequalities is a *projection method* that uses projection operators for individual half-spaces of Q. More specifically, it iteratively takes a convex combination of the projection operator $P_i(v)$ and the point v (often referred to as the *anchor point*) using a sequence of parameters. We now provide the properties that this sequence of parameters requires.

Definition 4.1.1 (*steering sequence*). *A sequence* $(\sigma_k)_{k=0}^{\infty}$ *is called a steering sequence if it has the following properties:*

$$\begin{aligned} &\sigma_k \in [0,1] \text{ for all } k \geq 0, \text{ and } \lim_{k\to\infty} \sigma_k = 0, \\ &\textstyle\sum_{k=0}^{\infty} \sigma_k = \infty, \qquad\qquad (\text{or equivalently, } \textstyle\prod_{k=0}^{\infty}(1-\sigma_k) = 0), \\ &\textstyle\sum_{k=0}^{\infty} |\sigma_{k+1} - \sigma_k| < \infty. \end{aligned} \tag{4.2}$$

The third property in (4.2) was introduced by Wittmann, see [36].

For solving problem (3.1), we provide the pseudocode in Algorithm A.2 that uses the projection operator as defined in (4.1). The following is a generalized version of the **HLWB** algorithm with respect to linear inequalities.

Algorithm 4.1 Cyclic **HLWB** algorithm for linear inequalities

Initialization: Choose an arbitrary initialization point $x_0 \in R^n$

Iterative Step: Given the current iterate x_k, calculate the next iterate x_{k+1} by

$$x_{k+1} = \sigma_k v + (1-\sigma_k) P_{i_k}(x_k), \tag{4.3}$$

where v is the given anchor point, $i_k = k \bmod m$, P_{ik} is the projection operator for the ik-th hyperplane as defined in (4.1), and $(\sigma_k)_{k=0}^{\infty}$ is a steering sequence.

We note that the **HLWB** algorithm has a much broader formulation that applies to the **BAP** with respect to the common fixed points set of a family of firmly nonexpansive (FNE) operators presented in Bauschke [2] and [4, Chap. 30]. For more details on the **BAP** see [17] and the references therein.

The family of iterative projection methods for the **BAP** includes but is not limited to the **HLWB** method, Dykstra's algorithm (see [10] and [4, Theorem 30.7]), Haugazeau's algorithm (see [23] and [4, Corollary 30.15]), and Hildreth's algorithm [24, 32]. There are also simultaneous versions of some of these algorithms available, see, e.g., [11]. A string-averaging **HLWB** algorithm, which encompasses the sequential, the simultaneous and other variants of the **HLWB** algorithm, recently appeared in [13].

4.1.1 Time Complexity of RNNM and HLWB

We now discuss theoretical time complexity differences of **RNNM** and **HLWB** . From the **RNNM** algorithm, Algorithm 3.1, we can see that the worst-case time complexity per iteration is $O(m^3 + m^2 n)$ flops [1], of which every step but solving the linear system is efficiently parallelizable. We note that in line 7 of Algorithm 3.1, the linear system we are solving is positive definite and sparse. Therefore, the linear system can be solved efficiently using the Cholesky decomposition. From the **HLWB** algorithm, Algorithm A.2, we can see that worst-case time complexity per iteration is $O(mn)$ and per sweep is $O(m^2 n)$, of which every step is efficiently parallelizable. [2]

[1] See Algorithm 3.1 lines 4-12, the total time complexity respectively is: $m^2n + m^2 + m^3 + n + 2n + mn + 2n + mn + n + m + 1 = m^2n + m^3 + m^2 + 2mn + 5n + m + 1 = O(m^3 + m^2 n)$.

[2] See Algorithm A.2 lines 5-12; the total time complexity respectively per iteration that projects onto a half space is $(2n+2)+1+(n+2)+(mn+m+1) = mn+3n+m+6 = O(mn)$ flops. Similarly, the total time complexity respectively per iteration that projects onto the nonnegative orthant is: $n+1+(n+2)+(mn+m+1) = mn+2n+m+4 = O(mn)$

From the perspective of theoretical time complexity it would be easy to assume that as a first-order method, **HLWB** is the preferable algorithm. This is because each of its iterations are composed of operations that are completely parallelizable and each first-order sweep has an overall lower worst-case time-complexity. However, without performing numerical tests with varying parameters m and n, we cannot yet conclude how a first-order method compares to a second-order method in terms of desired performance, especially as m and n get extremely large as observed in practice.

4.2 Quadratic Programming Proximal Augmented Lagrangian method, QPPAL

We now discuss the **QPPAL** algorithm. **QPPAL** is a sophisticated algorithm that employs the use of first and second-order methods. Designed for solving large-scale convex quadratic programming problems, its use of semismooth Newton methods for solving the augmented Lagrangian subproblem makes it an appropriate candidate for comparison in our numerical testing.

To provide more context on how **QPPAL** is solving the **BAP**, we introduce the general quadratic programming problem in Euclidean space:

$$\begin{array}{ll} \min_x & \frac{1}{2}x^TQx + c^Tx \\ \text{subject to:} & \\ & Ax = b, \\ & x \in C, \end{array} \tag{4.4}$$

where $C := \{x \in \mathbb{R}^n \,|\, l \leq x \leq u\}$. The vectors l and u are variable bounds that are given and satisfy $-\infty \leq l \leq u \leq +\infty$. Further assumptions placed on the use of **QPPAL** is that a Slater point exists. Since the constraints are linear and the objective in (4.4) is a convex quadratic function, we know strong duality holds and we can therefore discuss the dual of (4.4). We use the restricted-Wolfe dual from [33] of (4.4) as the following maximization problem

$$\begin{array}{ll} \max_{w,y,z} & -\delta_C^*(-z) - \frac{1}{2}w^TQw + b^Ty \\ \text{subject to:} & \\ & z - Qw + A^Ty = c, \\ & w \in \text{range}(Q), \end{array} \tag{4.5}$$

where $\delta_C(\cdot)$ denotes the indicator function of C and $\delta_C^*(\cdot)$ is its convex conjugate function. For our applications, Q is positive definite (nonsingular), so the last constraint is redundant and is ignored.

Now that we have defined (4.4) and its dual (4.5), we elaborate on the augmented Lagrangian method and how **QPPAL** approaches this problem. The philosophy of **QPPAL** is to implement the augmented Lagrangian method on the dual problem, (4.5). Doing so and reformulating (4.5) as a minimization problem yields the following augmented Lagrangian with a barrier term $\rho > 0$

$$L_\rho(w, y, z; x) = \delta_C^*(-z) + \frac{1}{2}w^TQw - b^Ty + \frac{\rho}{2}\left\|z - Qw + A^Ty - c + \frac{1}{\rho}x\right\|^2 - \frac{1}{2\rho}\|x\|^2. \tag{4.6}$$

flops of which all flops are efficiently parallelizable. Therefore, in terms of sweeps the **HLWB** method computes $m(mn + 3n + m + 6) + mn + 2n + m + 4 = m^2n + 4mn + m^2 + 2n + 7m + 4 = O(m^2n)$ flops.

The augmented Lagrangian method performs the following iterative steps on (4.6)

$$\begin{array}{rcl}(w^{k+1}, y^{k+1}, z^{k+1}) & = & \operatorname{argmin}_{(w,y,z)} L_\rho(w, y, z; x^k), \\ x^{k+1} & = & x^k + \tau\rho(A^T y^{k+1} - c),\end{array} \tag{4.7}$$

where the first step of the iterative process in (4.7) is the augmented Lagrangian subproblem and $\tau \in (0, 2)$ guarantees convergence. The main issue with the augmented Lagrangian method is solving the augmented Lagrangian subproblem. Computing $(w^{k+1}, y^{k+1}, z^{k+1})$ with high accuracy is both difficult and expensive, especially in high dimensions. To overcome this difficulty, **QPPAL** requires a warm-starting point that is close enough to the optimal solution of (4.4) to make augmented Lagrangian subproblem less expensive to solve.

To provide a good initial point for the augmented Lagrangian method, **QPPAL** performs a warm-start using a Gauss-Seidel-based inexact semi-proximal augmented Lagrangian algorithm. This is referred to as Phase 1 of **QPPAL** by the authors. For the pseudocode and more details on how this is done see [35, Section 3.1] and the references therein.

Once a warm-start point by Phase 1 of **QPPAL** is provided, a semismooth Newton method for solving the augmented Lagrangian subproblem can be formulated using an orthogonal projection onto C. This is referred to as Phase 2 of **QPPAL** by the authors. For the pseudocode and more details on how this is done see [35, Section 3.3] and the references therein.

4.3 Comparison of Algorithms for solving the BAP

We now discuss how we perform our numerical experiments and our methodologies for comparison. For these methodologies we refer to a discussion on techniques for comparisons of algorithms given in [6]. In particular. we include performance profiles as defined in [18], and tables that include details on the performances of **RNNM** (exact and inexact), **HLWB** , *lsqlin*, and for **QPPAL**.

We compare **RNNM** to **HLWB**, MATLAB's *lsqlin* method, and **QPPAL** by generating a test problem of the form specified in (3.1). We generate the problem such that the anchor v lies in the relative interior of the normal cone of the feasible polyhedron at a vertex $\hat{x}$. Since v lies in the normal cone of of the feasible polyhedron at $\hat{x}$, $\hat{x}$ is the closest point to v, and thus the optimal solution. Furthermore, we set $\|A\| = 1$ and $\|v\| = 1$ to ensure meaningful comparisons as no convergence results for **RNNM** solving (3.1) have been proven, as far as we know.

For the detailed comments regarding initialization and stopping criterion we refer to the pseudocodes of **RNNM** and **HLWB** in Appendix A. The **RNNM** algorithm starts with initializing $x_0 \leftarrow (v + A^T y_0)_+$, where $y_0 = 0_m$. Then $x_0 \leftarrow (v + A^T y_0)_+$ reduces to $x_0 \leftarrow \max(v, 0)$ in the initialization stage of **RNNM**. This occurs before the first Newton step, therefore, to ensure all algorithms start at the same point, we initialize $x_0 \leftarrow \max(v, 0)$ for **HLWB**, and provide $x_0 \leftarrow \max(v, 0)$ as a warm start for MATLAB's *lsqlin* solver. However, since **QPPAL** performs an **ADMM** warm-start as described in Section 4.2, there is no way to provide a warm start point for it.

Since **RNNM** solves a reduced KKT condition for a convex problem with a Slater point, the term $\frac{\|F(y_k)\|}{1+\|b\|}$ is a sufficient relative residual to serve as a stopping condition for **RNNM**. We note that the stopping criterion for **HLWB** is measured at the end of a sweep rather than at the end of an iteration. Since **HLWB** converges in the limit, but does not have proper stopping criterion,

we use the relative primal feasibility residual, $\frac{\|A\hat{x}_k - b\|}{1+\|b\|}$, as the stopping criterion. Furthermore, for **HLWB** we use $\hat{x}_k$ instead of x_k in the stopping criterion as $\hat{x}_k$ is nonnegative at the end of every sweep. The *lsqlin* solver uses first-order optimality conditions, i.e., the KKT system as its stopping criterion. As in *lsqlin*, **QPPAL** uses first-order optimality conditions, and we report the relative optimality gap,

$$\frac{|p^* - d^*|}{\left(1 + \frac{|p^*|+|d^*|}{2}\right)},$$

for the relative residual of **QPPAL** .

When implementing **QPPAL**,, we use **QPPAL** 's Cholesky decomposition direct solver as opposed to its inexact solver. Furthermore, we increase the maximum number of iterations for the two phases of **QPPAL** to match the maximum number of sweeps the other methods utilize. Lastly, we inform **QPPAL** that the quadratic used is the identity.

In Section 4.3.1, we generate problems such that v lies in the relative interior of the normal cone of a nondegenerate vertex. We add additional experimental results and present them in Appendix B.1, observing a similar set of comparisons between the algorithms presented in this section. Furthermore, we provide detailed experimental results such that v lies in the relative interior of the normal cone of a degenerate vertex in Appendix B.2. These tests, and the performance of the **RNNM** algorithm help to motivate the theory and potential practice of using **RNNM** for the **LP** application, as seen in Section 5.3.

For the performance profiles in Section 4.3.1, we use the following notation from [6]. Let P denote our set of problems with varying m, n, and density. Similarily, let S represent our set of solvers, **RNNM** (exact and inexact), **HLWB**, *lsqlin*, and **QPPAL**. We define the performance measure $t_{p,s} > 0$ for each pair $(p, s) \in P \times S$ as the computational time of solver s to solve problem p. For each problem $p \in P$ and solver $s \in S$, we define the performance ratio as

$$r_{p,s} = \begin{cases} \frac{t_{p,s}}{\min\{t_{p,s} \,|\, s \in S\}}, & \text{if convergence test passed,} \\ \infty, & \text{if convergence test failed.} \end{cases}$$

The solver s that performs the best on problem p will have a performance ratio of 1. Solvers that perform worse than s on problem p will satisfy $t_{p,s} > 1$. In other words, the larger the performance ratio, the worse the solver performed on problem p.

The performance profile of a solver s is defined as

$$\rho_s(\tau) = \frac{1}{|P|} \text{ size}\{p \in P \,|\, r_{p,s} \leq \tau\}.$$

Therefore, $\rho_s(\tau)$ represents the relative portion of time in which the performance ratio $r_{p,s}$ for solver s is within a factor $\tau \in \mathbb{R}$ of the best possible performance ratio.

4.3.1 Numerical Comparisons

We tested the algorithms with optimal solutions at: nondegenerate vertices, degenerate vertices and non-vertices. They all exhibited similar results. Therefore, in this section we present the results

for nondegenerate vertex solutions. We begin with choosing v for (3.1) such that the optimum is uniquely a nondegenerate vertex of P. In the tables below we vary m, n, and the problem density to illustrate the changes in each solver's performance. A data point in each table is the arithmetic mean of 5 randomly generated problems of the specified parameters that also satisfy $\|A\| = 1$, $\|v\| = 0.1$. For example, the first row of Table 4.1 represents a problem with parameters $m = 500$, $n = 3000$, and a density of 0.0081, and each solver will solve 5 randomly generated problems of the form discussed in (3.1), and the average time and relative residual from solving all 5 problems is displayed in the table. The desired stopping tolerance for the tables and performance profiles is $\varepsilon = 10^{-14}$ and maximum iterations (sweeps for **HLWB**) is 2000 for all solvers.

We remark that the regularization parameter of **RNNM** is chosen in an adaptive way. It takes into account the relative residual as defined in line 13 of Algorithm 3.1, the norm of the Newton direction, and the norm of v. The purpose of this is to decrease the magnitude of λ, or the amount of regularization, as we approach the optimal solution while accounting for the norms of the Newton direction and v. This regularization parameter is explicitly defined as

$$\lambda_{k+1} = \text{mean}\left(\left(10^{-2}F_k\right)\max(1, \log_{10}(\|d_k\|)), \left(10^{-3}F_k\right)\max(1, \log_{10}(\|v\|)), 10^{-3}F_k\right), \tag{4.8}$$

where F_k is the relative residual at iteration k, and d_k is the Newton direction.

The empirical evidence from Tables 4.1 to 4.3 demonstrates the superiority of the **RNNM** (exact) approach over the other solvers. Since the **RNNM**'s reduced KKT system is $m \times m$ and solved efficiently using the Cholesky Decomposition, its performance should be most affected as m varies or density increases. This theoretical observation can be seen in Tables 4.1 to 4.3, as the **RNNM** (exact and inexact) algorithm is slower to converge for increasing m and density, but is not affected by an increase in n.

From Figure 4.1 the empirical evidence shows similar results to the tables, but better demonstrates the differences in performance between **RNNM** (exact) and the other solvers as it more concretely shows **RNNM** outperforming the other solvers. The problems in Figure 4.1a are similar to those of Table 4.1 except m varies by 100 from 100 to 2000. Similarly, the problems in Figure 4.1b have n varying by 100 from 3000 to 5000, and Figure 4.1c has density varying by 1% from 1% to 100%. In every performance profile, the **RNNM** (exact) algorithm clearly outperforms the other solvers in our experiments, with **RNNM** (inexact) performing well for an inexact method on mid-sized problems. Conversely, **HLWB** is relatively slow on these problems. This can be attributed to its linear convergence rate. Due to its linear convergence, it will perform a large number of sweeps, which can amount to millions of iterations on certain problems with large m. Performance profiles can be found in Appendix B.1 with the stopping tolerances $\varepsilon = 10^{-2}, 10^{-4}$, to illustrate that **RNNM** (exact) outperforms HLWB and *lsqlin* at different tolerances, but **QPPAL** remains competitive. In Appendix B.2 we include tables and performance profiles for the case that the solution x^* is degenerate.

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
500	3000	8.1e-01	4.23e-02	1.51e-01	1.54e+02	3.77e+00	1.14e+00	1.96e-16	8.26e-16	2.25e-04	7.26e-17	1.72e-17
1000	3000	8.1e-01	4.40e-01	9.97e-01	3.71e+02	5.37e+00	2.15e+00	2.70e-16	1.95e-15	2.14e-04	3.87e-17	2.70e-17
1500	3000	8.1e-01	1.17e+00	3.23e+00	6.09e+02	7.02e+00	4.69e+00	3.41e-17	6.73e-16	2.27e-04	3.95e-17	1.16e-17
2000	3000	8.1e-01	2.49e+00	7.51e+00	8.67e+02	1.02e+01	7.81e+00	6.11e-17	3.11e-17	2.24e-04	3.14e-17	-2.74e-17

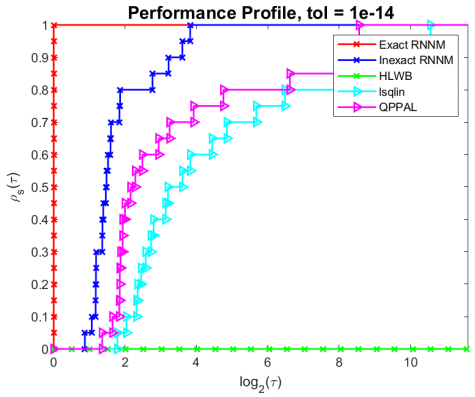
Table 4.1: Varying problem sizes m; comparing computation time and relative residuals.

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
200	3000	8.1e-01	3.12e-03	3.69e-02	4.45e+01	3.50e+00	8.66e-01	8.64e-18	7.39e-17	2.56e-04	6.52e-16	5.89e-17
200	3500	8.1e-01	3.08e-03	4.05e-02	5.17e+01	4.93e+00	1.00e+00	9.07e-18	1.26e-17	2.78e-04	1.23e-15	2.15e-17
200	4000	8.1e-01	3.24e-03	3.70e-02	5.82e+01	7.31e+00	1.09e+00	1.46e-16	8.91e-16	2.80e-04	3.21e-16	-9.18e-18
200	4500	8.1e-01	3.99e-03	4.17e-02	6.58e+01	1.01e+01	1.18e+00	1.80e-15	2.05e-16	3.13e-04	4.61e-17	1.71e-16

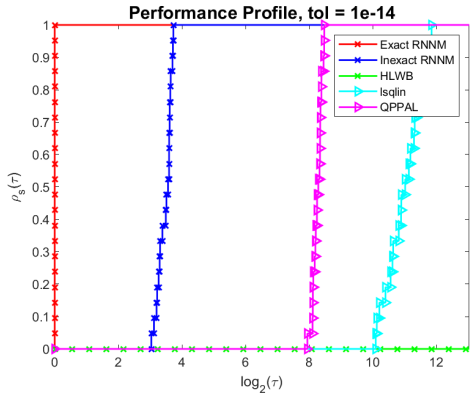
Table 4.2: Varying problem sizes n; comparing computation time and relative residuals.

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
300	1000	25	5.69e-02	2.66e-01	4.55e+01	3.30e-01	1.20e+00	2.83e-17	1.14e-17	1.50e-04	8.61e-17	5.99e-17
300	1000	50	5.43e-02	2.28e-01	5.39e+01	3.08e-01	1.82e+00	1.23e-16	1.97e-17	1.44e-04	8.08e-16	1.42e-17
300	1000	75	7.75e-02	2.86e-01	5.36e+01	3.16e-01	1.49e+01	4.83e-16	1.72e-17	1.62e-04	3.49e-16	-3.43e-16
300	1000	100	7.27e-02	2.47e-01	4.65e+01	3.00e-01	2.54e+02	5.66e-16	2.15e-17	1.63e-04	1.91e-15	1.04e-14

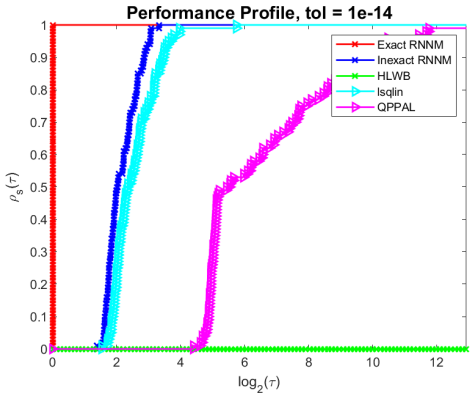
Table 4.3: Varying problem density; comparing computation time and relative residuals.



(a) Varying problem sizes m.



(b) Varying problem sizes n.



(c) Varying problem density.

Figure 4.1: Performance profiles for problems with varying m, n, and densities for nondegenerate vertex solutions.

Chapter 5

Theoretical Background for Linear Programming

Linear programming has a very rich history of applications and algorithms. In 1947 we saw Dantzig model the mechanization of military planning during World War 2 as a linear program and develop the simplex method to solve it. Around the same time we saw Kantorovich and Koopmans independently weave classical economic and transport problems such as military expenditures and optimal routing into linear programs, for which they shared the 1975 Nobel prize in economics. Despite its great success, the simplex method is not considered an *efficient* algorithm, as its worst-case solve time is $O(2^n)$.[1] Nevertheless, the boundless number of problems that could be expressed as a linear program motivated the development of polynomial time solvers. In 1979 Khachiyan modified the Shor, Nemirovski, and Yudin ellipsoid method [46, 54] for linear programming, and is the first polynomial-time algorithm devised for linear programming [30]. This marked a major breakthrough for linear programming. Despite the ellipsoid methods inferior practical performance compared to the simplex method, the ellipsoid methods polynomial-time complexity sparked increased interest in developing efficient polynomial-time linear programming algorithms. In 1984 Karmarker developed the first interior point method for linear programming [29], afterwards a revolution of research in linear programming began to unfold that would later be regarded as the interior point revolution.

5.1 Optimality Conditions

One of the most widely used interior point methods are primal-dual interior point methods. Primal-dual interior point methods examine various Newton equations of the logarithmic barrier function, i.e., recall the primal **LP** (2.8) and dual **LP** (2.9), then the vector $x^* \in \mathbb{R}^n$ is a solution of (2.8) if and only if there exists vectors $(y^*, z^*) \in \mathbb{R}^m \times \mathbb{R}^n$ for which the following equations hold for $(x, y, z) = (x^*, y^*, z^*)$

[1] Examples of the simplex method having a worst-case solve time of $O(2^n)$ can be found using the Klee-Minty cube [31]. Despite the worst-case solve-time being $O(2^n)$, the average case in *practice* has been observed to be polynomial-time under various probability distributions for random matrices $A \in \mathbb{R}^{m \times n}$ [8, 45] .

$$\begin{aligned} Ax &= b, \\ A^T y + z &= c, \\ x_i z_i &= 0,\ i = \{1, \ldots, n\}, \\ (x, z) &\geq 0. \end{aligned} \tag{5.1}$$

We remark that since $x, z \geq 0$, $x_i z_i = 0 \equiv x^T z = 0$.

Then the optimality conditions of (5.1) can be restated as the function $F : \mathbb{R}^{2m+n} \to \mathbb{R}^{2m+n}$:

$$\begin{aligned} F(x, y, z) &= \begin{bmatrix} Ax - b \\ A^T y + z - c \\ XZe \end{bmatrix} = 0, \\ & (x, z) \geq 0,\ X = \operatorname{diag}(x_1, \ldots, x_n),\ Z = \operatorname{diag}(z_1, \ldots, z_n). \end{aligned} \tag{5.2}$$

Therefore, the goal of primal-dual methods is to provide a procedure for how to follow a perturbation of the complementarity condition $x^T z = 0$, i.e., we use the equation $XZe = \mu$ in (5.2). This new formulation of (5.2) is called the *central path*, and stems from the log-barrier function of the nonnegativity constraint.

Then given F as defined in (5.2), interior point methods use Newton's method to form a linear model for $F(x, y, z)$ around the current point (x_k, y_k, z_k). We then obtain the search directions $(\Delta x, \Delta y, \Delta z)$ by solving the following system of linear equations at (x_k, y_k, z_k)

$$\nabla F(x_k, y_k, z_k) \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = -F(x_k, y_k, z_k). \tag{5.3}$$

We can usually only take very small steps in the pure Newton method as described in (5.3). Primal-dual methods aim to iteratively update the current point given $(\Delta x, \Delta y, \Delta z)$ based on a primal and dual step length α_p and α_d respectively, i.e.,

$$\begin{aligned} x_{k+1} &\leftarrow x_k + \alpha_p \Delta x, \\ y_{k+1} &\leftarrow y_k + \alpha_d \Delta y, \\ z_{k+1} &\leftarrow z_k + \alpha_d \Delta z. \end{aligned}$$

Here we have only discussed primal-dual interior point methods, but in general methods for computing μ in $XZe = \mu$ and the step lengths that interior point methods can take are rich in variety and can be studied in more detail in [51].

While interior point methods are highly regarded and have applications in general nonlinear programming, they are not the only Newton-type approach to linear programming. For example, in [39] Mangasarian proposes a Newton-type method that utilizes the equivalence between the **BAP** and the **LP**, as stated in [38, 48]. Mangasarian does this by solving a parametric exterior penalty formulation of the **LP**

$$\min_x c^T x \text{ subject to } Ax \leq b,$$

for any sufficiently small but finite value of the penalty parameter ρ. Then, for any sufficiently small penalty parameter an exact minimum norm solution with respect to the 2-norm is computed.

The parametric exterior penalty formulation of the **LP** is the following problem:

$$\min_x \rho c^T x + \frac{1}{2}\|(Ax-b)_+\|^2.$$

Therefore, the objective in this approach is to determine an appropriate barrier parameter such that the solution y of the quadratic program is the minimum norm solution of the dual **LP**. In other words, we denote ρ as the barrier parameter and have the following quadratic problem

$$\begin{array}{ll} \max_y & (b^T y + \frac{\rho}{2}\|y\|^2) \\ \text{subject to:} & \\ & A^T y + c = 0, \\ & y \geq 0. \end{array}$$

If $\bar{\rho}$ is the largest possible barrier parameter such that y is a minimum norm solution of the dual **LP**, then we have for any barrier parameter ρ such that $0 < \rho \leq \bar{\rho}$, y remains a minimum norm solution of the dual **LP**. The weakness in this approach, as acknowledged by Mangasarian in [39, Remark 3.1], is that determining an appropriate barrier parameter such that the solution y of the quadratic program is the minimum norm solution of the dual **LP** is too expensive. To overcome this, Mangasarian claims that a small enough barrier term, $\rho = 10^{-3}$, is more than sufficient to attain the optimal solution of the **LP**. Mangasarian supports this claim with numerics for medium-sized **LP**'s.

In our contributions, we use the theory from [38, 48], but differ from Mangasarian's approach by taking advantage of the highly accurate solution provided by **RNNM** to perform the necessary sensitivity analysis needed to change the current basis. Compared to Mangasarian's Newton method, this is incredibly inexpensive, as it involves the simple computation of a simplex-like ratio test. The paper that we published outlining our proposed external path following algorithm in Section 5.2 and the respective numerical results in Section 5.3 can be found in [12].

5.2 An External Path Following Algorithm for Solving Linear Programs

We consider the maximization primal **LP** in standard equality form

$$\text{(PLP)} \qquad \begin{array}{lll} p^*_{LP} := & \max & c^T x \\ & \text{subject to} & Ax = b \in \mathbb{R}^m \\ & & x \in \mathbb{R}^n_+. \end{array} \tag{5.4}$$

The dual LP is

$$\text{(DLP)} \qquad \begin{array}{lll} d^*_{LP} := & \min & b^T y \\ & \text{subject to} & A^T y - z = c \in \mathbb{R}^n \\ & & z \in \mathbb{R}^n_+. \end{array} \tag{5.5}$$

We assume that A is full row rank and that the optimal value is finite. By the fundamental theorem of linear programming, we can now guarantee that strong duality holds for the primal and dual problems, i.e., equality $p^*_{LP} = d^*_{LP}$ holds and both optimal values are *attained*.

We will see in Lemma 5.2.1 that the solution to (PLP) is the limit of the sequence of projections of the vectors $v_R = Rc \in \mathbb{R}^n$ onto the feasible set as $R \uparrow \infty$.

Lemma 5.2.1 ([37–39, 49]). *Suppose that A, b, and c are given LP data such that A is full-row rank, and the **LP** has a finite optimal value p^*_{LP}. For each $R > 0$ we define*

$$x^*(R) := \begin{array}{ll} \operatorname{argmin}_x & \frac{1}{2}\|x - Rc\|^2 \\ \textit{subject to} & Ax = b \in \mathbb{R}^m \\ & x \in \mathbb{R}^n_+. \end{array} \tag{5.6}$$

Then x^* is the minimum norm solution of (PLP) if and only if there exists $\bar{R} > 0$ such that

$$R \geq \bar{R} \implies x^* = x^*(R) = \operatorname{argmin}\left\{\frac{1}{2}\|x - Rc\|^2 \mid Ax = b,\, x \in \mathbb{R}^n_+\right\}. \tag{5.7}$$

We would like an R that is not too large, but large enough such that $R\|c\| > \|x^*\|$. We use the following estimate to start our algorithm:

$$R = \min\left\{50, \frac{\sqrt{mn}\,\|b\|}{1 + \|c\|}\right\}. \tag{5.8}$$

To avoid numerical complications from large numbers, we consider the following equivalent problem that uses the scaling $\frac{1}{R}b$ rather than Rc.

Corollary 5.2.2. *Let $A, b, c, R, x^*(R)$ be defined as in Lemma 5.2.1. Then*

$$\frac{1}{R}x^*(R) = w^*(R) := \begin{array}{ll} \operatorname{argmin}_w & \frac{1}{2}\|w - c\|^2 \\ \textit{subject to:} & Aw = \frac{1}{R}b \in \mathbb{R}^m \\ & w \in \mathbb{R}^n_+. \end{array} \tag{5.9}$$

Proof. By examining the objective in Lemma 5.2.1, we see the following relationship

$$\|x - Rc\|^2 = R^2\left\|\frac{1}{R}x - c\right\| = R^2\|w - c\|,\ w = \frac{1}{R}x.$$

Substituting for $w = \frac{1}{R}x$ into the constraints of Lemma 5.2.1 gives us $Ax = b \iff A(Rw) = b \iff Aw = \frac{1}{R}b$, and $R > 0 \implies w \in \mathbb{R}^n_+$. It then follows that the optimal solution of Lemma 5.2.1 is independent of R^2, and so discarding the constant R^2 gives us (5.9). □

5.2.1 Exploiting Sensitivity Analysis to Warm Start the BAP

Recall the scaling done in Corollary 5.2.2 and the relationship between the scaling for c and the decision variable x

$$x^*(R) = Rw^*(R).$$

To simplify notation, we ignore the optimality symbol $(\cdot)^*$ and examine the optimality conditions of (3.4) with respect to $w = w(R)$ as in (5.9)

$$\begin{pmatrix} w - c - A^T y - z \\ Aw - \frac{1}{R} b \\ z^T w \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \qquad w, z \in \mathbb{R}^n_+,\ y \in \mathbb{R}^m. \tag{5.10}$$

Then we can conclude that

$$\lim_{R\to\infty} P_{\text{range}(A^T)} w(R) = 0,\ \lim_{R\to\infty} Rw(R) = x^*, \text{ the optimum of the } \textbf{LP}\ (5.6).$$

The optimality conditions become

$$w = c + A^T y + z,\ b = ARw = AR(c + A^T y)_+,\ w^T z,\ w, z \geq 0. \tag{5.11}$$

Therefore, $\|w\| = \left\|c + A^T y + z\right\|$ indicates that the $\|w\|$ is a measure of the error for dual feasibility, and therefore a measure for the accuracy of Rw as the optimum of the original **LP**.

Given the current R and approximate optimal triplet $(w(R), y(R), z(R))$ for (5.6), we would like to find a new $R_n \geq R$ that is a better approximation of the optimal solution for (5.4). Furthermore, we would also like to use the corresponding $y_n = y(R) + \Delta y$ for solving the **BAP** with the new R_n. In this way, the **BAP** serves as a subproblem for the **LP** algorithm proposed in Theorem 5.2.3, and y_n serves as a warm-start for solving the **BAP** subproblem.

Theorem 5.2.3. *Suppose $R > 0$ is given and the triplet $(w, y, z) = (w(R), y(R), z(R))$ is primal-dual optimal for (5.9), i.e., (5.10) is satisfied. Let*

$$\begin{gathered} \mathcal{N} = \mathcal{N}(z) := \{i \,|\, z_i > 0\},\ \mathcal{B} = \mathcal{B}(w) := \{i \,|\, w_i > 0\},\ \mathcal{Z} = \mathcal{Z}(w, z) := \{i \,|\, w_i = z_i = 0\}; \\ e = \begin{pmatrix} b_{\mathcal{B}} - Rw_{\mathcal{B}} \\ -(b_{\mathcal{N}} + Rz_{\mathcal{N}}) \end{pmatrix}, \quad f = \begin{pmatrix} Rb_{\mathcal{B}} \\ -Rb_{\mathcal{N}} \end{pmatrix}, \end{gathered} \tag{5.12}$$

where $b_{\mathcal{B}}, b_{\mathcal{N}}$ are defined in (5.16) and (5.21) respectively. Then the maximum value for increasing R and maintaining both optimality and the indices in the bases sets $\mathcal{B}, \mathcal{N}, \mathcal{Z}$ is

$$R_n = \min\{f_i/e_i \,|\, e_i > 0, f_i > 0,\ \forall\, i\}. \tag{5.13}$$

Furthermore, the corresponding changes $\Delta w, \Delta y, \Delta z$ that result in $w + \Delta w, y + \Delta y, z + \Delta z$ being optimal for R_n are given in the proof in (5.16), (5.15), and (5.21) respectively.

*Moreover, if $R_n = \infty$, then the optimal solution of the **LP** has been found.*

Proof. We first want to find the maximum increase in R that keeps the current basis $\mathcal{B}$ optimal for (5.6). In other words, we maintain the following

$$z_i \geq 0,\ \forall\, i \in \mathcal{N}, w_i \geq 0,\ \forall\, i \in \mathcal{B}, w_i = z_i = 0,\ \forall\, i \in \mathcal{Z}.$$

To maintain the feasibility from the three basis sets in (5.12), we examine the effect of sensitivity analysis on the optimality conditions as defined in (5.10). We will denote the new R attained through sensitivity analysis as R_n. Then we have the following equations:

(1) From primal feasibility we have

$$A_{\mathcal{B}} w_{\mathcal{B}} = \frac{1}{R} b \text{ and } A_{\mathcal{B}}(w_{\mathcal{B}} + \Delta w_{\mathcal{B}}) = \frac{1}{R_n} b \implies A_{\mathcal{B}} \Delta w_{\mathcal{B}} = \left(\frac{1}{R_n} - \frac{1}{R} \right) b.$$

(2) Since $z_{\mathcal{B}} = 0$, then from dual feasibility we have that $w_{\mathcal{B}} - c_{\mathcal{B}} - A_{\mathcal{B}}^T y = 0$ and $w_{\mathcal{B}} + \Delta w_{\mathcal{B}} - c_{\mathcal{B}} - A_{\mathcal{B}}^T (y + \Delta y) = 0 \implies \Delta w_{\mathcal{B}} = A_{\mathcal{B}}^T (\Delta y)$, and therefore

$$A_{\mathcal{B}} \Delta w_{\mathcal{B}} = A_{\mathcal{B}} A_{\mathcal{B}}^T (\Delta y) = \left(\frac{R - R_n}{R R_n} \right) b. \tag{5.14}$$

Now that we have solved for $\Delta w_{\mathcal{B}}$, it is left to solve for Δy and $\Delta z_{\mathcal{N}}$. Since $w_{\mathcal{Z}} = z_{\mathcal{Z}} = 0$, then from similar logic to how we obtain (5.14) we have

$$-c_{\mathcal{Z}} - A_{\mathcal{Z}}^T (y + \Delta y) = 0 \implies A_{\mathcal{Z}}^T (\Delta y) = 0.$$

Lastly, since $w_{\mathcal{N}} = 0$, we have

$$-c_{\mathcal{N}} - A_{\mathcal{N}}^T (y + \Delta y) - (z_{\mathcal{N}} + \Delta z_{\mathcal{N}}) = 0 \implies \Delta z_{\mathcal{N}} = -A_{\mathcal{N}}^T (\Delta y).$$

Therefore, we have two equations to solve Δy. When strict complementarity fails, we choose a full column rank matrix $V_{\mathcal{Z}}$ that satisfies $\text{range}(V_{\mathcal{Z}}) = \text{null}(A_{\mathcal{Z}}^T)$. Otherwise, we define $V_{\mathcal{Z}}$ to be the identity. Then we solve to get

$$\Delta y_p := V_{\mathcal{Z}} (A_{\mathcal{B}} A_{\mathcal{B}}^T V_{\mathcal{Z}})^{\dagger} b, \; \Delta y := \left(\frac{R - R_n}{R R_n} \right) \Delta y_p. \tag{5.15}$$

By assumption $b \in \text{range}(B)$, therefore a solution always exists for (5.15). To maintain the primal feasibility and nonnegativity of $w_{\mathcal{B}} + \Delta w_{\mathcal{B}}$, we have the following inequality

$$-w_{\mathcal{B}} \leq \Delta w_{\mathcal{B}} = A_{\mathcal{B}}^T (\Delta y) = A_{\mathcal{B}}^T \left(\frac{R - R_n}{R R_n} \right) \Delta y_p = -\left(\frac{R_n - R}{R R_n} \right) A_{\mathcal{B}}^T \Delta y_p, \; b_{\mathcal{B}} := A_{\mathcal{B}}^T \Delta y_p. \tag{5.16}$$

We note that the result in (5.16) simplifies in the nondegenerate case as we have $V_{\mathcal{Z}} = I$

$$\begin{aligned} A_{\mathcal{B}}^T \left(\tfrac{R_n - R}{R R_n} \right) \Delta y_p &= \left(\tfrac{R_n - R}{R R_n} \right) A_{\mathcal{B}}^T V_{\mathcal{Z}} (A_{\mathcal{B}} A_{\mathcal{B}}^T V_{\mathcal{Z}})^{\dagger} b \\ &= -\left(\tfrac{R_n - R}{R R_n} \right) A_{\mathcal{B}}^{\dagger} b_{\mathcal{B}} \\ &= -\left(\tfrac{R_n - R}{R R_n} \right) b_{\mathcal{B}}, \; b_{\mathcal{B}} := A_{\mathcal{B}}^{\dagger} b. \end{aligned}$$

Therefore, to get an upperbound for R_n we rearrange the inequalities in (5.16) to the following

$$(R_n - R) b_{\mathcal{B}} \leq (R R_n) w_{\mathcal{B}} \iff R_n (b_{\mathcal{B}} - R w_{\mathcal{B}}) \leq R b_{\mathcal{B}}. \tag{5.17}$$

We note that the above inequality holds for the trivial case $R = R_n$.

To find the maximum possible R_n given the current (w, y, z) and check that $R_n \neq +\infty$, we use an **LP** type ratio test. We set the two vectors to be

$$e_{\mathcal{B}} = (b_{\mathcal{B}} - Rw_{\mathcal{B}}),\ f_{\mathcal{B}} = Rb_{\mathcal{B}}.$$

For simplicity in our notation, we ignore the subscript $\mathcal{B}$ and refer to $e_{\mathcal{B}}, f_{\mathcal{B}}$ as e, f. Furthermore, we note that we cannot have both $e_i > 0, f_i \leq 0$, as (5.17) would not hold. Therefore, we choose R_n to be the maximum that satisfies the following ratio test

$$R_n := \{f_i/e_i \,|\, f_i > 0,\ e_i > 0,\ i \in \mathcal{B}\}. \tag{5.18}$$

We note that the minimum over the empty set is $+\infty$ and $\max_i\{f_i/e_i \,|\, f_i < 0,\ e_i < 0,\ i \in \mathcal{B}\} \leq R_n$ always holds since $R_n = R > 0$ satisfies the inequality. If $A_{\mathcal{B}}$ is full column rank or $b_{\mathcal{B}} = w_{\mathcal{B}}$, then we have the best least-squares solution and get $R_n = \infty$.

Now that we have done the ratio test for $w_{\mathcal{B}}$, it is left to do the ratio test for $z_{\mathcal{N}}$. Note that we set $\Delta z_i = \Delta w_i = 0,\ \forall\, i \in \mathcal{Z}$. Therefore, to maintain the dual feasibility and nonnegativity of $z_{\mathcal{N}} + \Delta z_{\mathcal{N}}$, we have the following inequality

$$-z_{\mathcal{N}} \leq \Delta z_{\mathcal{N}} = -A_{\mathcal{N}}^T(\Delta y) = -A_{\mathcal{N}}^T\left(\frac{R - R_n}{RR_n}\right)\Delta y_p = \left(\frac{R_n - R}{RR_n}\right)A_{\mathcal{N}}^T\Delta y_p =: \left(\frac{R_n - R}{RR_n}\right)b_{\mathcal{N}}. \tag{5.19}$$

Therefore, to get an upperbound for R_n we rearrange the inequalities in (5.19) to the following

$$(R_n - R)b_{\mathcal{N}} \geq (RR_n)z_{\mathcal{N}} \iff R_n(-b_{\mathcal{N}} - Rz_{\mathcal{N}}) \leq -Rb_{\mathcal{N}}. \tag{5.20}$$

We note that the above inequality holds for the trivial case $R = R_n$.

We again find the maximum R_n and check that $R_n \neq \infty$ using an **LP** type ratio test. We set the two vectors to be $e_{\mathcal{N}} = -(b_{\mathcal{N}} + Rz_{\mathcal{N}}),\ f_{\mathcal{N}} = -Rb_{\mathcal{N}}$. For simplicity in our notation we ignore the subscript $\mathcal{N}$ and refer to $e_{\mathcal{N}}, f_{\mathcal{N}}$ as e, f. Furthermore, we note that we cannot have $e_i > 0$ and $f_i \leq 0$, as (5.20) would not hold. Therefore, we choose R_n to be the maximum that satisfies the following ratio test

$$R_n := \{f_i/e_i \,|\, f_i > 0,\ e_i > 0,\ i \in \mathcal{N}\}. \tag{5.21}$$

Again, we note that the minimum over the empty set is $+\infty$ and $\max_i\{f_i/e_i \,|\, f_i < 0,\ e_i < 0,\ i \in \mathcal{N}\} \leq R_n$ always holds since $R_n = R > 0$ satisfies the inequality.

Finally, we choose R_n as the minimum of the two values found in (5.18) and (5.21) to satisfy the necessary primal and dual feasibility conditions.

If $R_n = \infty$, then increasing R no longer changes the bases, and therefore we have found an optimal basis and equivalently the sufficiently large R such that the optimal solution to the **LP** has been found. □

Theorem 5.2.3 illustrates the external path following algorithm that we are using. Its use requires highly accurate solutions to the **BAP** subproblem. The theorem uses the triplet $(w, y, z) = (w(R), y(R), z(R))$ from the **BAP** subproblem to find specific values of R, *stepping stones on the path*, such that the current basis changes. Once we find the next *stepping stone* is equal to infinity, we know that we have found the optimal basis for the **LP**. Thus, we have an external path following

algorithm with parameter R, but we only choose specific points on this path to step on. The algorithm is efficient for nondegenerate problems as $\mathcal{Z} = \emptyset$ gives us accurate sensitivity analysis. For highly degenerate problems, restricting $\Delta w_i = \Delta z_i = 0$, $\forall i \in \mathcal{Z}$ can severely restrict how much we can increase R by. See Section 5.3 for more details.

5.2.2 Upper and Lower Bounds for the LP Problem

The optimal solution from the projection problems (5.6) and (5.9) provides a feasible x, and we get the corresponding ***LP*** *lower bound* $c^T x^*(R)$. It is now left to find the upper bound.

Recall from Section 5.2.1 that primal feasibility and complementary slackness hold for $x(R) = Rw$ and z, and this is identical for the **LP** problem. Therefore, we need to find the dual variable y_{LP} to satisfy the **LP** dual feasibility condition

$$z_{\mathsf{LP}} = A^T y_{\mathsf{LP}} - c \geq 0.$$

However, the optimality conditions as described in (5.10) give us

$$A^T(-y) = z + c - w,\ 0 \leq z = A^T(-y) - c + w,\ w \geq 0.$$

Therefore, as discussed previously, we see that the norm of w becomes small as R tends to infinity and we get dual feasibility, $y(R) \to y_{\mathsf{LP}}$. But, at each iteration we have

$$z - w = A^T(-y) - c,\ w, z \geq 0,\ w^T z = 0, y \equiv y_R. \tag{5.22}$$

We can write the required dual feasibility equations using the indices for $i \in \mathcal{B}$ as defined in Theorem 5.2.3.

$$A_i^T y - c_i \in \begin{cases} \{0\}, & \text{if } w_i > 0, \\ \mathbb{R}_+, & \text{if } w_i = 0. \end{cases}$$

Therefore, recall the definitions of $\mathcal{B}$ and $\mathcal{N}$ as defined in Theorem 5.2.3. Then for a given y_R that satisfies (5.22), we consider the nearest dual **LP** feasible system with unknowns $z \geq 0$ and y_{LP}.

Lemma 5.2.4. *Let (w, y, z) be the approximate optimal triplet from (5.11) and $\mathcal{B}$ the support as defined in Theorem 5.2.3. Consider the* ***BAP*** *for the given dual variables.*

$$\begin{aligned} \begin{pmatrix} y^*_{\mathsf{LP}} \\ z^*_{\mathsf{LP}} \end{pmatrix} \in \quad & \operatorname{argmin} \quad \tfrac{1}{2}\|(-y) - y_{\mathsf{LP}}\|^2 + \tfrac{1}{2}\|0 - (z_{\mathsf{LP}})_{\mathcal{B}}\|^2 + \tfrac{1}{2}\|z_{\mathcal{N}} - (z_{\mathsf{LP}})_{\mathcal{N}}\|^2 \\ & \textit{subject to} \quad \begin{bmatrix} A_{\mathcal{B}}^T & -I & 0 \\ A_{\mathcal{N}}^T & 0 & -I \end{bmatrix} \begin{pmatrix} y_{\mathsf{LP}} \\ (z_{\mathsf{LP}})_{\mathcal{B}} \\ (z_{\mathsf{LP}})_{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} c_{\mathcal{B}} \\ c_{\mathcal{N}} \end{pmatrix} \\ & \qquad\qquad y_{\mathsf{LP}} \textit{ free},\ z_{\mathsf{LP}} = \begin{pmatrix} (z_{\mathsf{LP}})_{\mathcal{B}} \\ (z_{\mathsf{LP}})_{\mathcal{N}} \end{pmatrix} \geq 0. \end{aligned} \tag{5.23}$$

Then the optimal value of the ***LP*** *(5.4) satisfies the upper bound*

$$p^*_{\mathsf{LP}} \leq b^T y^*_{\mathsf{LP}}.$$

*Moreover, suppose that $z_{\mathcal{B}} = 0$. Then equality holds and the **LP** is solved with primal-dual optimum pair (w, y_{LP}).*

Proof. Recall that the optimal value p^*_{LP} is finite. The proof of the bound follows from weak duality in linear programming. Equality follows from the optimality conditions since primal feasibility and complementary slackness hold with w. □

5.3 Solving Large Sparse Linear Programs

We now apply (5.6) and Theorem 5.2.3 to solve large-scale randomly generated **LP**s, and problems from the NETLIB dataset. We call this method the *stepping stones external path following algorithm, (**SSEPF**)*, and note that we use the estimate for a starting R given in (5.8). The stepping stones are found using R_n in (5.13). We add a small positive scalar to R_n to ensure that we change the basis of x at each iteration. For simplicity, we restrict ourselves to nondegenerate **LP**s for the randomly generated problems.

We compare **SSEPF** to the MATLAB *linprog* code, Mosek, and *semismooth Newton inexact proximal augmented Lagrangian method*, (**SNIPAL**) [34]. For the MATLAB *linprog* and Mosek software, we examine both the dual simplex and interior point method solvers. We scale the randomly generated problems such that $\|A\| = 1$ and the optimal solution x^* satisfies $\|x^*\| = 1$. A data point in Table 5.1 is the arithmetic mean of 5 randomly generated problems of the specified parameters, m, n, and density. We exclude instances from Table 5.1 where a method fails to provide a solution, but these instances are plotted in Figure 5.1 as a failure to converge. Since the smallest stopping tolerance allowed by *linprog* is $\varepsilon = 10^{-10}$, a linear program is considered successfully solved in the performance profile of Figure 5.1 if the optimality gap is less than or equal to $\varepsilon = 10^{-8}$. The maximum number of iterations for *linprog* and Mosek is the default number, and for **SNIPAL** it is 2000. The relative residual shown in Table 5.1 is the sum of the relative primal feasibility, dual feasibility, and complementary slackness. In other words, let (x^*, y^*, z^*) be the optimal solution an algorithm returns, then the relative residual as shown in Table 5.1 is

$$\frac{\|Ax^* - b\|}{1 + \|b\|} + \frac{\left\|z^* - A^T y^* + c\right\|}{1 + \|c\|} + \frac{(x^*)^T z^*}{1 + \max(\|x^*\|, \|z^*\|)}.$$

We note that we are using the exact **RNNM** direction to solve the **BAP** subproblem when discussing the performance of **SSEPF**, and using (4.8) to compute the regularization parameter. We label **SSEPF**'s performance in Table 5.1 and Figure 5.1 as **SSEPF-RNNM**. Furthermore, we use the abbreviations Linprog DS and Linprog IPM to refer to *linprog's* dual simplex and interior point method, respectively. Likewise, we use similar abbreviations for Mosek.

From Table 5.1, the empirical evidence demonstrates that the stepping stone approach performs better than MATLAB's dual simplex and interior point method on most problems, and has proven to be quite competitive with Mosek's dual simplex and interior point method. This becomes more evident as the sizes of the problems grow and the problems become sparser. In other words, we see that our code fully exploits sparsity in **LP**. This can be seen when observing the performance

of **SSEPF-RNNM** with respect to time on the rows of Table 5.1 where the problem density decreases. Despite the increase in problem dimension, the decrease in density leads to an increase in performance in comparison to the previous row. Another thing to notice is that in rows 5-9 of Table 5.1, *linprog's* interior point method and Mosek's dual simplex method failed to converge to a solution after having reached the default maximum number of iterations.

In Section 4.3.1, the performance profiles were constructed by looking at smaller intervals of varying m, n and density. For example Table 4.1 shows results where m varies by increments of 500, but in Figure 4.1a m varies by increments of 100. Since *linprog's* interior point method and Mosek's dual simplex method struggled with obtaining the desired primal feasibility, as seen in Table 5.1, Figure 5.1 shows the performance of each solver with respect to all 50 problems instead of examining the average performance.

It is important to note that the performance profile exhibits more failed solutions from the dual simplex and interior point methods of MATLAB. We have tried taking the maximum of the primal feasibility, dual feasibility, and complementary slackness returned by MATLAB's *linprog* function instead of the sum, and both revealed equivalent results. In other words, we are not sure why there are more problems failing at this tolerance than reported by MATLAB, but it further distinguishes our stepping stone approach from MATLAB's *linprog* algorithms. Mosek, and more specifically Mosek's interior point method is very competitive, as Figure 5.1 shows. Unfortunately, **SNIPAL** failed to converge on every problem in this dataset. We have seen it converge successfully on some random linear programming problems, but none of the ones that we generated in our Numerical Experiments section. It is worth noting that the table which shows the average performance of 5 randomly generated problems with respect to a set of parameters indicates that **SSEPF-RNNM** performs better than Mosek's interior point method in 7 out of 10 rows in the table.

Specifications			Time (s)						Rel. Resids.					
m	n	% density	SSEPF-RNNM	Linprog DS	Linprog IPM	MOSEK DS	MOSEK IPM	SNIPAL	SSEPF-RNNM	Linprog DS	Linprog IPM	MOSEK DS	MOSEK IPM	SNIPAL
2e+03	5e+03	1.0e-01	8.94e-02	3.09e-02	4.50e-02	1.46e-01	1.64e-01	6.90e+00	3.38e-17	2.63e-16	4.88e-09	1.31e-16	1.53e-16	2.14e-04
2e+03	1e+04	1.0e-01	9.64e-02	4.84e-02	7.53e-02	1.49e-01	1.93e-01	8.31e+00	2.82e-17	6.00e-16	1.60e-04	1.31e-16	2.89e-16	1.72e-04
2e+03	1e+05	1.0e-01	1.68e-01	3.91e-01	7.45e-01	5.41e-01	6.56e-01	1.94e+01	1.48e-17	7.45e-17	1.72e-05	8.84e-17	8.57e-17	1.55e-04
5e+03	1e+04	1.0e-01	9.97e+01	2.08e-01	1.39e+01	4.26e-01	2.65e+00	5.54e+01	5.55e-17	4.16e-16	5.02e-07	1.67e-14	3.20e-16	2.29e-04
5e+03	1e+05	1.0e-01	7.64e+01	7.24e-01	1.42e+02	1.12e+00	8.51e+00	7.85e+01	2.36e-17	9.31e-11	6.38e-05	3.13e-16	1.79e-16	1.58e-04
5e+03	5e+05	1.0e-01	2.30e+02	6.97e+00	6.54e+02	7.02e+00	1.52e+01	1.70e+02	1.52e-17	1.87e-10	3.73e-05	3.92e-16	1.68e-16	1.48e-04
2e+04	1e+05	1.0e-02	6.32e-01	9.46e-01	5.68e+00	1.05e+00	2.49e+00	4.28e+01	1.36e-17	3.55e-06	4.33e-07	1.99e-06	1.28e-16	1.42e-04
2e+04	5e+05	1.0e-02	6.66e-01	4.46e+00	3.78e+01	5.63e+00	9.28e+00	1.23e+02	8.48e-18	3.37e-06	8.83e-07	1.36e-06	2.89e-16	1.10e-04
2e+04	1e+06	1.0e-02	1.85e+00	9.30e+00	6.50e+01	1.17e+01	1.59e+01	2.06e+02	7.08e-18	4.34e-06	6.27e-06	1.76e-06	9.65e-17	1.12e-04
1e+05	1e+07	1.0e-03	7.38e+00	1.06e+01	6.14e+00	9.35e+01	9.60e+01	1.56e+03	1.39e-18	1.39e-18	1.39e-18	1.76e-17	1.76e-17	5.90e-05

Table 5.1: **LP** application results averaged on 5 randomly generated problems per row.

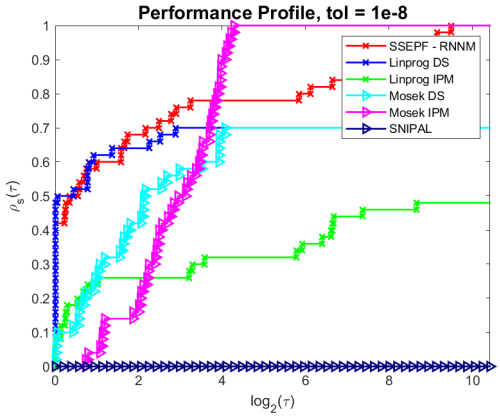


Figure 5.1: Performance Profiles for **LP** application with respect to all problems.

We also consider the first five problems in alphabetical order from the subset of the NETLIB dataset where primal strict feasibility (PSF) holds [27, Sect. 4.2.2]. We then check dual strict feasibility (DSF) and include the value of the constant we obtain from solving the theorem of the alternative. In other words, a large, respectively small, constant indicates an algebraically *fat*, respectively *thin*, feasible set. Failure, or near failure, of strict feasibility correlates with the difficulty **RNNM** experiences in our observed numerics. We successfully solve two of the five problems. We think that the difficulties from the NETLIB dataset is due to the dual feasible set being very thin for some problems. For example, in Table 5.2, the problems 25fv47 and lotfi have a very thin feasible set in the dual problem. It is important to note that the performance of **SSEPF-RNNM** on the blend problem is significantly worse than the other solvers. A common issue with SSEPF-RNNM when solving the blend problem as well as rows 4-6 of Table 5.1 is that at certain tolerances, **RNNM** uses the maximum number of iterations (2000) to solve the **BAP** subproblem. In other words, even though we are performing a warm-start with the solution from the previous **BAP** subproblem, **RNNM** can fail to converge to the desired relative tolerance. However, even though **RNNM** failed to converge, it still provides a solution that is very close to the optimal solution, i.e., instead of solving the **BAP** subproblem to within a relative tolerance of 10^{-14}, it returns a solution that is within a relative tolerance of 10^{-12} or 10^{-13}. There are at least two solutions to this issue. First, we can decrease the length of the Newton step when the iteration count is large. Using this heuristic shows significant improvement in performance when solving the blend problem. Secondly, if **RNNM** fails to converge to within the specified relative tolerance of 10^{-14}, we can try a larger relative tolerance, such as 10^{-13}. This strategy has shown to be crucial when trying to solve problems like 25fv47, where we are not able to solve the **BAP** subproblem with high accuracy due to its thin dual feasible set.

Problem:	Primal Strict Feas.	Dual Strict Feas.
25fv47	2.00e-01	2.01e-17
afiro	9.00e+00	1.19e-01
blend	7.30e-02	3.49e-03
israel	3.71e+00	1.38e-03
lotfi	1.00e+00	1.89e-10

Table 5.2: Primal and Dual strict feasibility of NETLIB problems.

	Time (s)						Rel. Resids.					
Problem:	SSEPF-RNNM	Linprog DS	Linprog IPM	MOSEK DS	MOSEK IPM	SNIPAL	SSEPF-RNNM	Linprog DS	Linprog IPM	MOSEK DS	MOSEK IPM	SNIPAL
25fv47	Inf	2.01e-01	1.01e-01	3.76e-01	1.54e-01	1.20e+01	Inf	2.30e-15	2.25e-15	5.51e-16	1.09e-14	7.36e-05
afiro	2.62e-02	7.71e-03	2.91e-03	9.16e-02	9.01e-02	9.81e-02	1.97e-16	3.67e-16	8.62e-14	7.49e-17	1.43e-13	9.39e-11
blend	1.42e+02	8.48e-03	3.81e-03	9.12e-02	9.03e-02	1.58e+00	5.37e-15	4.78e-14	1.31e-13	1.33e-15	1.63e-15	1.30e-03
israel	Inf	1.07e-02	2.79e-02	9.33e-02	9.82e-02	3.27e+00	Inf	7.15e-16	8.44e-14	6.57e-16	8.93e-12	5.21e-05
lotfi	Inf	9.63e-03	7.86e-03	9.41e-02	9.43e-02	2.00e+00	Inf	4.61e-14	3.38e-14	1.17e-16	9.05e-13	4.35e-05

Table 5.3: LP application results on the NETLIB problems.

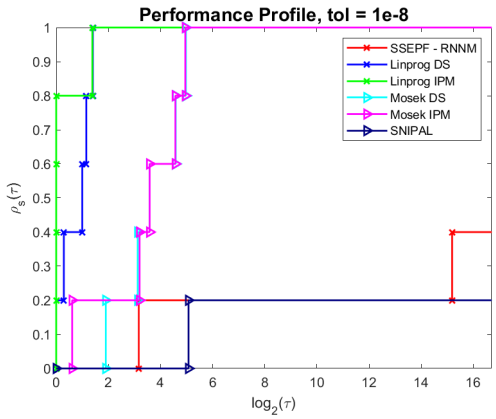


Figure 5.2: Performance Profiles for LP application with respect to the Netlib problems.

Our algorithm has difficulties with highly degenerate problems where the optimal solution is not unique. Moreover, the optimal minimum norm solution that our algorithm finds can fail strict complementarity. In other words, we find that there are many $i \in \{1, \ldots, n\}$ such that $x_i + z_i = 0$. The loss of strict complementarity results in a generalized Jacobian with low rank as few columns of A are chosen in (3.14). Additionally, the sensitivity analysis of Theorem 5.2.3 has difficulty increasing R. Finally, the failure of strict complementarity indicates that the gradient at optimality is not in the relative interior of the normal cone, Lemma 3.4.1, Item b, indicating failure of differentiability of the projection.

Chapter 6

Conclusion

In this thesis we showed that the **BAP** when the constraint set is the intersection of a linear manifold with the nonnegative cone can be solved efficiently and with high accuracy for medium and large-scale sparse problems when using a nonsmooth Newton method. We study the application of the nonsmooth Newton method on a Moreau Decomposition of the KKT conditions. The optimality conditions that arise from the Moreau Decomposition are independent of the number of variables. However, the nonsmoothness of the KKT conditions raises the question of how to choose a subgradient when utilizing the nonsmooth Newton method. To choose the appropriate subgradients, we use the problem examined in [16, 28] to minimize the condition number of the Jacobian.

Despite the conditioning consideration, we often still encounter a singular Jacobian. We handle this using a modified Levenberg-Marquardt regularization with an adaptive regularization parameter. We have proven this modified Levenberg-Marquardt regularization is a descent direction under certain assumptions that are maintained by the properties of our Jacobian and problem structure. As this is a descent direction, there are implementations that can be made that guarantee global convergence such as line-search method that satisfy Wolfe-type conditions [40]. Even without a line search, **RNNM** solves the **BAP** very competitively, outperforming professional and academic codes such as MATLAB's *lsqlin* and **QPPAL** [35]. Furthermore, we provide a preprocessing scaling technique for when the argument of the projection v is too far from the solution x^* for **RNNM** to converge, as seen in Corollary 5.2.2.

Another result in this thesis includes the application of the **BAP** to solving a **LP**. We do this by using a solution from the **BAP** to perform the necessary sensitivity analysis to guarantee we have a large enough parameter such that Lemma 5.2.1 is true. We have also proven that we can achieve finite termination using our sensitivity analysis given an optimal solution of the best approximation problem. In other words, we use the sensitivity analysis on the solution from the **BAP**, we estimate an appropriate new parameter R, and resolve the **BAP**. This terminates in a finite number of iterations. We have shown that **SSEPF-RNNM** is very competitive with professional and academic codes such as Mosek and **SNIPAL** when solving nondegenerate **LP**'s where strict feasibility holds.

References

[1] H. Bauschke and V. Koch. Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. 1 2013. 2

[2] H.H. Bauschke. The approximation of fixed points of compositions of nonexpansive mappings in hilbert space. *Journal of Mathematical Analysis and Applications*, 202:150–159, 1996. 2, 3, 32, 33

[3] H.H. Bauschke and J.M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, 38(3):367–426, 1996. 1

[4] H.H. Bauschke and P.L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, Cham, second edition, 2017. With a foreword by Hédy Attouch. 33

[5] H.H. Bauschke, D. Mao, and W.M. Moursi. How to project onto the intersection of a closed affine subspace and a hyperplane. 2022. 32

[6] V. Beiranvand, W. Hare, and Y. Lucet. Best practices for comparing optimization algorithms. *Optimization and Engineering*, 18(4):815–848, 2017. 35, 36

[7] D. Bertsimas and J.N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific books. Athena Scientific, 1997. 13, 15, 16

[8] K.H. Borgwardt. *The simplex method: a probabilistic analysis*, volume 1. Springer Science & Business Media, 2012. 39

[9] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004. 4, 9, 19

[10] J.P. Boyle and R.L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in order restricted statistical inference (Iowa City, Iowa, 1985)*, volume 37 of *Lect. Notes Stat.*, pages 28–47. Springer, Berlin, 1986. 33

[11] Y. Censor. Computational acceleration of projection algorithms for the linear best approximation problem. *Linear Algebra Appl.*, 416(1):111–123, 2006. 33

[12] Y. Censor, W.M. Moursi, T. Weames, and H. Wolkowicz. Regularized nonsmooth newton algorithms for best approximation, 2023. 21, 41

[13] Y. Censor and A. Nisenbaum. String-averaging methods for best approximation to common fixed point sets of operators: the finite and infinite cases. *Fixed Point Theory Algorithms Sci. Eng.*, pages Paper No. 9, 21, 2021. 33

[14] F.H. Clarke. *Optimization and nonsmooth analysis*, volume 5 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 1990. 11

[15] J. E. Dennis, Jr. and H. Wolkowicz. Sizing and least-change secant methods. *SIAM Journal on Numerical Analysis*, 30(5):1291–1314, 1993. 27

[16] J.E. Dennis Jr. and H. Wolkowicz. Sizing and least-change secant methods. *SIAM J. Numer. Anal.*, 30(5):1291–1314, 1993. 27, 51

[17] F. Deutsch. *Best approximation in inner product spaces*, volume 7 of *CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC*. Springer-Verlag, New York, 2001. 33

[18] E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91(2, Ser. A):201–213, 2002. 35

[19] D. Drusvyatskiy and H. Wolkowicz. *The many faces of degeneracy in conic optimization*. 2017. 16

[20] F. Facchinei and J.-S. Pang. *Finite-dimensional variational inequalities and complementarity problems*, volume 1. Springer, 2003. 24, 29

[21] Francisco Facchinei and Jong-Shi Pang. *Finite-dimensional variational inequalities and complementarity problems*, volume 2. Springer, 2003. 27

[22] S.-P. Han and O.L. Mangasarian. Conjugate cone characterization of positive definite and semidefinite matrices. *Linear Algebra Appl.*, 56:89–103, 1984. 8

[23] Y. Haugazeau. *Sur les Inéquations Variationnelles et la Minimisation de Fonctionnelles Convexes*. PhD thesis, University de Paris, 1968. 33

[24] C. Hildreth. A quadratic programming procedure. *Naval Res. Logist. Quart.*, 4:79–85, 1957. 33

[25] H. Hu, H. Im, X. Li, and H. Wolkowicz. A semismooth Newton-type method for the nearest doubly stochastic matrix problem. *Math. Oper. Res.*, May, 2023. arxiv.org/abs/2107.09631, 35 pages. 26

[26] H. Im. *Implicit Loss of Surjectivity and Facial Reduction: Theory and Applications*. PhD thesis, University of Waterloo, 2023. 15

[27] H. Im and H. Wolkowicz. Revisiting degeneracy, strict feasibility, stability, in linear programming. *European J. Oper. Res.*, 310(2):495–510, 2023. 35 pages, 10.48550/ARXIV.2203.02795. 15, 17, 18, 49

[28] W.L. Jung, D. Torregrosa-Belen, and H. Wolkowicz. Preconditioning, numerical computations and the Omega-condition number. Technical report, University of Waterloo, Waterloo, Canada, 2023 in progress. 31 pages. 27, 51

[29] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984. 39

[30] L.G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980. 39

[31] V. Klee and G.J. Minty. How good is the simplex algorithm. *Inequalities*, 3(3):159–175, 1972. 39

[32] A. Lent and Y. Censor. Extensions of Hildreth's row-action method for quadratic programming. *SIAM J. Control Optim.*, 18(4):444–454, 1980. 33

[33] X. Li, D. Sun, and K.-C. Toh. Qsdpnal: a two-phase augmented lagrangian method for convex quadratic semidefinite programming. *Mathematical Programming Computation*, 10(4):703–743, 2018. 34

[34] X. Li, D. Sun, and K.-C. Toh. An asymptotically superlinearly convergent semismooth Newton augmented Lagrangian method for linear programming. *SIAM J. Optim.*, 30(3):2410–2440, 2020. 3, 47

[35] L. Liang, X. Li, D. Sun, and K.-C. Toh. QPPAL: a two-phase proximal augmented Lagrangian method for high-dimensional convex quadratic programming problems. *ACM Trans. Math. Software*, 48(3):Art. 33, 27, 2022. 3, 32, 35, 51

[36] G. López, V. Martín-Márquez, and H.-K. Xu. Halpern's iteration for nonexpansive mappings. In *Nonlinear analysis and optimization I. Nonlinear analysis*, volume 513 of *Contemp. Math.*, pages 211–231. Amer. Math. Soc., Providence, RI, 2010. 33

[37] O.L. Mangasarian. Iterative solution of linear programs. *SIAM J. Numer. Anal.*, 18(4):606–614, 1981. 42

[38] O.L. Mangasarian. Normal solutions of linear programs. Number 22, pages 206–216. 1984. Mathematical programming at Oberwolfach, II (Oberwolfach, 1983). 2, 40, 41, 42

[39] O.L. Mangasarian. A Newton method for linear programming. *J. Optim. Theory Appl.*, 121(1):1–18, 2004. 2, 40, 41, 42

[40] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006. 24, 51

[41] L. Qi and J. Sun. A nonsmooth version of Newton's method. *Mathematical programming*, 58(1-3):353–367, 1993. 27

[42] H. Rademacher. Uber partielle und totale differenzierbarkeit i. *Math. Ann.*, 89:340–359, 1919. 12

[43] R.T. Rockafellar. *Convex analysis*. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 1997. Reprint of the 1970 original, Princeton Paperbacks. 4

[44] E. Sarabi. A characterization of continuous differentiability of proximal mappings of composite functions. URL: https://www.math.uwaterloo.ca/~hwolkowi/F22MOMworkshop.d/FslidesSarabi.pdf, 10 2022. 24th Midwest Optimization Meeting, MOM24. 24

[45] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998. 39

[46] N.Z. Shor. Cut-off method with space extension in convex programming problems. *Cybernetics*, 13(1):94–96, 1977. 39

[47] M. Sipser. *Introduction to the Theory of Computation*. Course Technology, Boston, MA, third edition, 2013. 19

[48] P.W. Smith and H. Wolkowicz. A nonlinear equation for linear programming. *Mathematical programming*, 34(2):235–238, 1986. 2, 40, 41

[49] P.W. Smith and H. Wolkowicz. A nonlinear equation for linear programming. *Math. Programming*, 34(2):235–238, 1986. 42

[50] H. Wolkowicz, R. Saigal, and L. Vandenberghe. *Handbook of semidefinite programming: theory, algorithms, and applications*, volume 27. Springer Science & Business Media, 2012. 4, 16

[51] S.J. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, 1997. 40

[52] Y. Xiao, Y. Censor, D. Michalski, and J.M. Galvin. The least-intensity feasible solution for aperture-based inverse planning in radiation therapy. *Annals of Operations Research*, 119:183–203, 2003. 1

[53] Y. Xiao, J. Galvin, M. Hossain, and R. Valicenti. An optimized forward-planning technique for intensity modulated radiation therapy. *Medical Physics*, 27(9):2093–2099, 2000. 1

[54] D.B. Yudin and A.S. Nemirovskii. Informational complexity and efficient methods for the solution of convex extremal problems. *Matekon*, 13(2):22–45, 1976. 39

Appendix A

Pseudocodes for Generalized Simplex

The pseudocodes described in Algorithms 3.1, A.1 and A.2 solve (3.1) using the exact and inexact nonsmooth Newton methods **RNNM**, and **HLWB** respectively.

Algorithm A.1 **BAP** of v for constraints $Ax = b, x \geq 0$, inexact Newton direction

Require: $v \in \mathbb{R}^n, y_0 \in \mathbb{R}^m$, $(A \in \mathbb{R}^{m\times n}, \text{rank}(A) = m)$, $b \in \mathbb{R}^m$, $\varepsilon > 0$, maxiter $\in \mathbb{N}$.

Output. Primal-dual: $x_{k+1}, (y_{k+1}, z_{k+1})$

Initialization. $k \leftarrow 0$, $x_0 \leftarrow (v + A^T y_0)_+$, $z_0 \leftarrow (x_0 - (v + A^T y_0))_+$,
$\delta \in (0, 1]$, $\nu \in [1 + \frac{\delta}{2}, 2]$, and a sequence θ such that $\theta_k \geq 0$ and $\sup_{k\in\mathbb{N}} \theta_k < 1$
$F_0 = Ax_0 - b$, stopcrit $\leftarrow \|F_0\| / (1 + \|b\|)$

while $((\text{stopcrit} > \varepsilon) \,\&\, (k \leq \text{maxiter}))$ **do**

  $V_k = \sum_{i\in\mathcal{I}_+} A_i A_i^T + \sum_{i\in\bar{\mathcal{I}}_0} \frac{1}{\|A_i\|^2} A_i A_i^T$

  $\lambda = (\text{stopcrit})^\delta$

  $\bar{V} = (V_k + \lambda I_m)$

  solve $\bar{V} d = -F_k$ for Newton direction d such that residual $\|r_k\| \leq \theta_k \|F_k\|^\nu$

  updates

    $y_{k+1} \leftarrow y_k + d$

    $x_{k+1} \leftarrow (v + A^T y_{k+1})_+$

    $z_{k+1} \leftarrow (x_{k+1} - (v + A^T y_k))_+$

    $F_{k+1} \leftarrow Ax_{k+1} - b$ (residual)

    stopcrit $\leftarrow \|F_{k+1}\| / (1 + \|b\|)$

    $k \leftarrow k + 1$

end while

Algorithm A.2 Extended **HLWB** algorithm

Require: $v \in \mathbb{R}^n, (A \in \mathbb{R}^{m\times n}, \text{rank}(A) = m)$, $b \in \mathbb{R}^m$, $\varepsilon > 0$, maxiter $\in \mathbb{N}$.

Output. x_{k+1}

Initialization. $k \leftarrow 0$, $msweeps \leftarrow 0$ $x_0 \leftarrow max(v, 0)$, $\hat{x}_0 \leftarrow x_0$, $i_0 = 1$

stopcrit $\leftarrow \|A\hat{x}_0 - b\| / (1 + \|b\|)$ $(= \|F_0\| / (1 + \|b\|))$

while ((stopcrit $> \varepsilon$) & ($k \leq$ maxiter)) **do**

  if $1 \leq i_k \leq m$ **then**

    $\hat{x}_k = x_k + \frac{b_{i_k} - a_{i_k}^T x_k}{\|a_{i_k}\|^2} a_{i_k}$

  else

    $\hat{x}_k = \max(0, x_k)$

  end if

  updates

    $\sigma_k = \frac{1}{k+1}$

    $x_{k+1} \leftarrow \sigma_k v + (1 - \sigma_k)\hat{x}_k$

    stopcrit $\leftarrow \|A\hat{x}_k - b\| / (1 + \|b\|)$

    if $k (\text{mod } m + 1) = 0$ **then**

      $msweeps = msweeps + 1$

    end if

    $i_k = k (\text{mod } m) + 1$

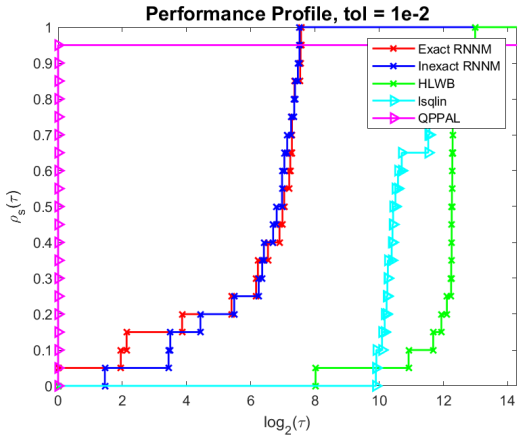
    $k \leftarrow k + 1$

end while

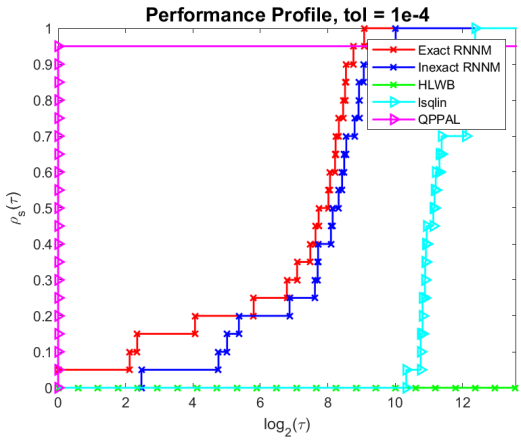
Appendix B

Additional Tables and Performance Profiles

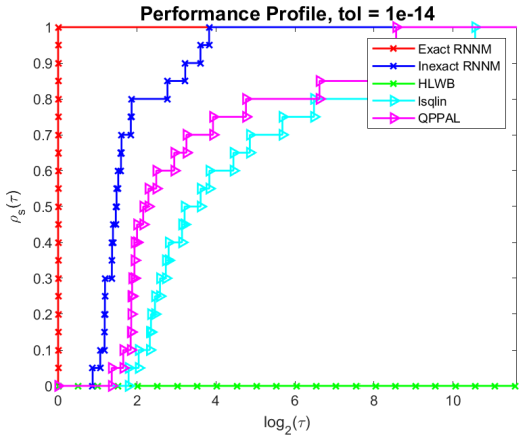
B.1 Nondegenerate Solutions



(a) tol = 10^{-2}

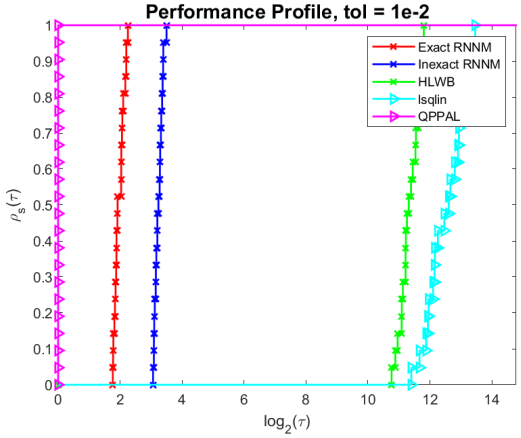


(b) tol = 10^{-4}

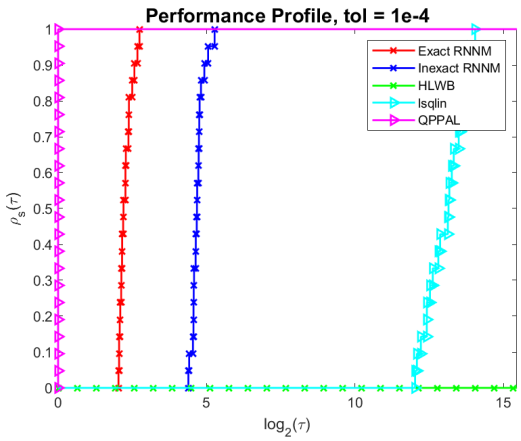


(a) tol = 10^{-14}

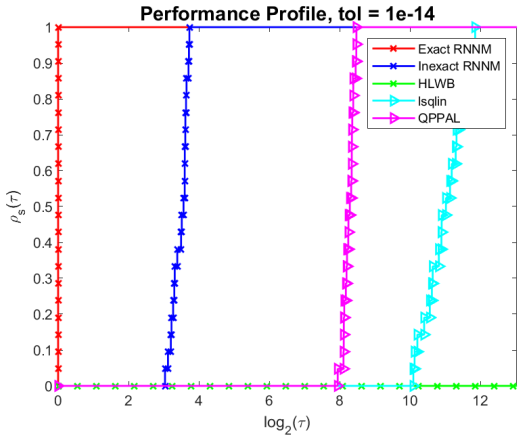
Figure B.2: Performance Profiles for varying m for nondegenerate vertex solutions.



(a) tol = 10^{-2}

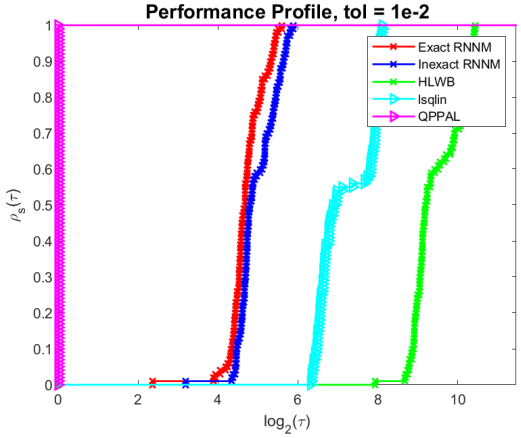


(b) tol = 10^{-4}

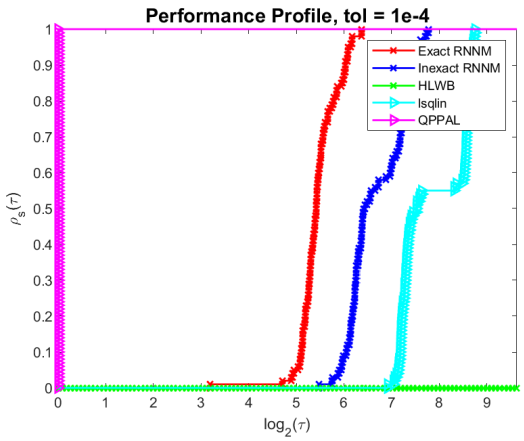


(a) tol = 10^{-14}

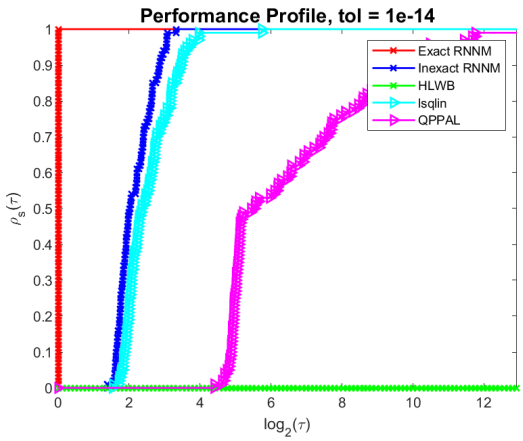
Figure B.4: Performance Profiles for varying n for nondegenerate vertex solutions.



(a) tol = 10^{-2}



(b) tol = 10^{-4}



(a) tol = 10^{-14}

Figure B.6: Performance Profiles for varying density for nondegenerate vertex solutions.

B.2 Degenerate Solutions

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
500	3000	8.1e-01	4.99e-02	1.83e-01	1.54e+02	3.80e+00	2.66e+01	2.88e-15	5.37e-17	2.25e-04	8.20e-17	3.07e-16
1000	3000	8.1e-01	4.64e-01	1.12e+00	3.71e+02	5.98e+00	3.63e+00	5.23e-18	4.35e-15	2.04e-04	6.42e-17	-2.20e-16
1500	3000	8.1e-01	3.45e+00	4.53e+00	6.14e+02	7.07e+00	8.61e+00	3.89e-18	2.28e-18	1.99e-04	2.02e-16	1.14e-16
2000	3000	8.1e-01	8.22e+00	8.02e+00	8.71e+02	1.94e+03	2.88e+01	1.27e-16	9.78e-17	2.16e-04	3.94e-18	-1.66e-16

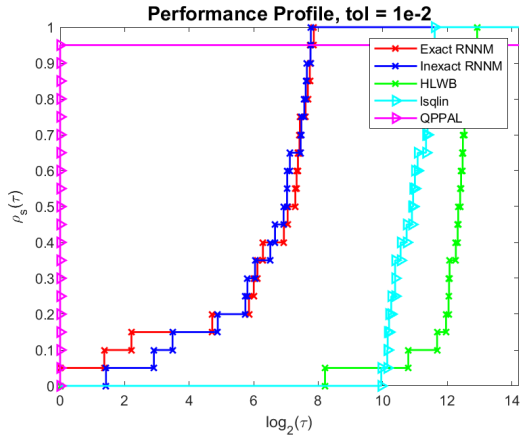
Table B.1: Varying problem sizes m and comparing computation time with relative residual for degenerate vertex solutions.

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
200	3000	8.1e-01	2.52e-03	3.94e-02	4.40e+01	3.34e+00	9.36e+00	7.23e-18	2.71e-18	2.43e-04	3.53e-17	-8.12e-16
200	3500	8.1e-01	2.41e-03	3.81e-02	5.09e+01	5.16e+00	1.80e+01	2.69e-16	2.10e-18	2.70e-04	6.63e-16	-1.10e-15
200	4000	8.1e-01	3.17e-03	3.62e-02	5.87e+01	7.07e+00	1.13e+01	5.29e-18	2.77e-18	2.69e-04	4.69e-17	2.30e-15
200	4500	8.1e-01	3.49e-03	4.17e-02	6.57e+01	9.39e+00	2.43e+01	4.94e-18	1.90e-18	3.11e-04	5.94e-16	2.94e-15

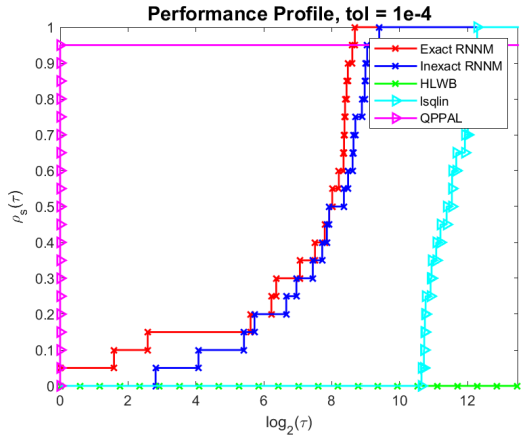
Table B.2: Varying problem sizes n and comparing computation time with relative residual for degenerate vertex solutions.

Specifications			Time (s)					Rel. Resids.				
m	n	% density	Exact	Inexact	HLWB	lsqlin	QPPAL	Exact	Inexact	HLWB	lsqlin	QPPAL
300	1000	25	3.90e-02	4.37e-01	4.58e+01	3.25e-01	2.25e+00	5.12e-17	1.23e-17	1.34e-04	9.04e-16	-5.76e-16
300	1000	50	6.52e-02	3.55e-01	5.42e+01	3.24e-01	1.38e+01	2.54e-17	9.84e-16	1.49e-04	6.69e-18	-1.90e-15
300	1000	75	9.85e-02	2.94e-01	5.32e+01	3.33e-01	5.41e+01	3.76e-17	3.06e-16	1.54e-04	3.17e-17	4.48e-15
300	1000	100	1.50e-01	3.03e-01	4.79e+01	2.88e-01	2.96e+02	4.43e-17	2.01e-17	1.47e-04	1.42e-16	-9.59e-14

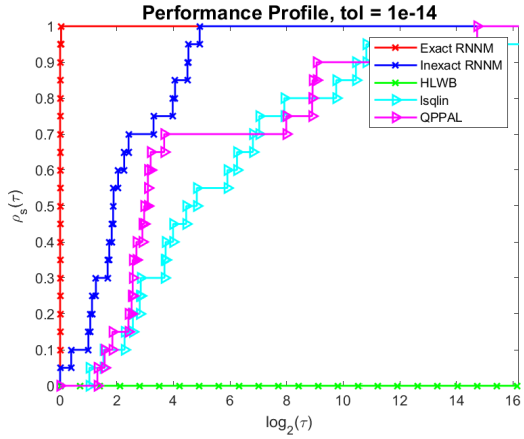
Table B.3: Varying problem density and comparing computation time with relative residual for degenerate vertex solutions.



(a) tol $= 10^{-2}$

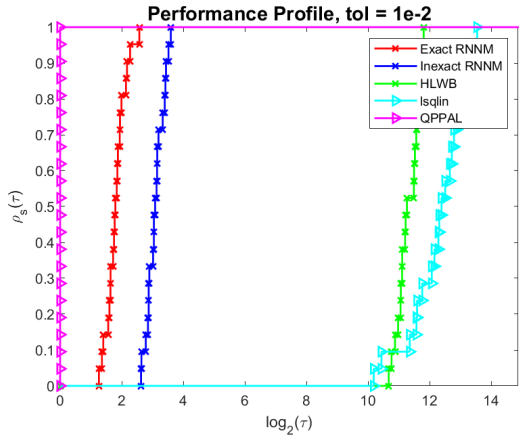


(b) tol $= 10^{-4}$

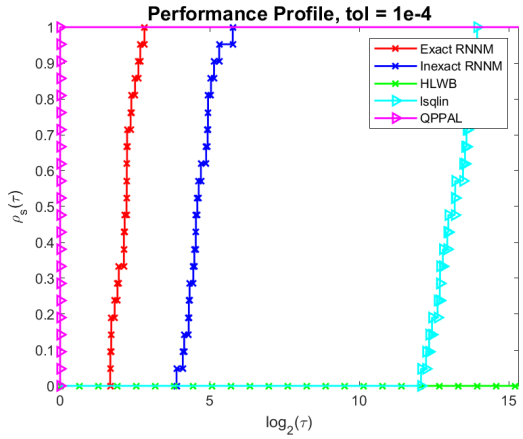


(c) tol $= 10^{-14}$

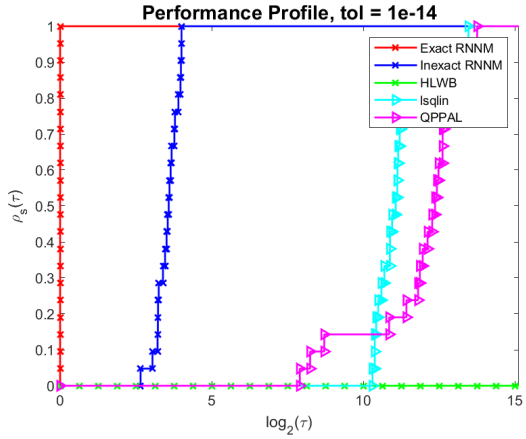
Figure B.7: Performance Profiles for varying m for degenerate vertex solutions.



(a) tol $= 10^{-2}$

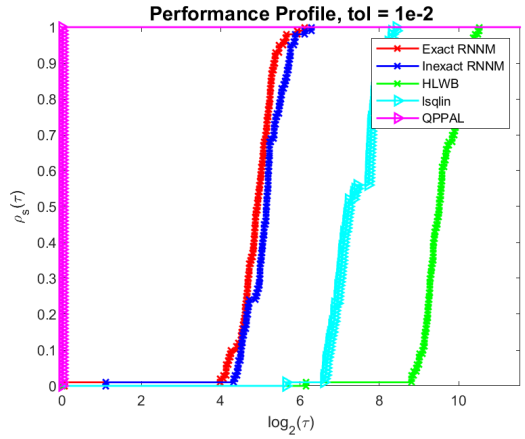


(b) tol $= 10^{-4}$

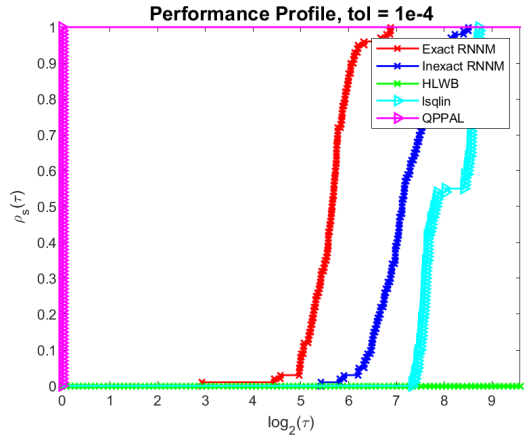


(c) tol $= 10^{-14}$

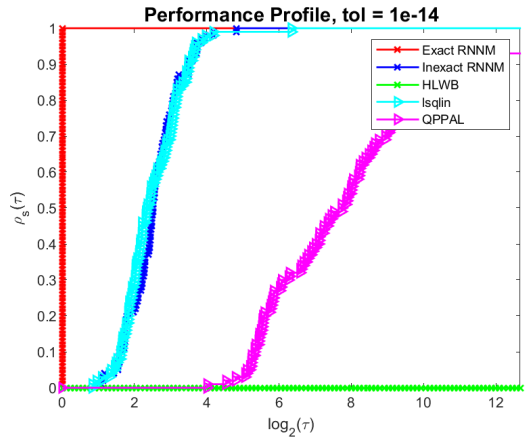
Figure B.8: Performance Profiles for varying n for degenerate vertex solutions.



(a) tol = 10^{-2}



(b) tol = 10^{-4}



(c) tol = 10^{-14}

Figure B.9: Performance Profiles for varying density for degenerate vertex solutions.

Index