Regularized Nonsmooth Newton Algorithms for Best Approximation with Applications *

Yair Censor† Walaa M. Moursi‡ Tyler Weames‡ Henry Wolkowicz‡

submitted Monday Dec. 19, 2022

Abstract

We consider the problem of finding the best approximation point from a polyhedral set, and its applications, in particular to solving large-scale linear programs. The classical projection problem has many various and many applications. We study a regularized nonsmooth Newton type solution method where the Jacobian is singular; and we compare the computational performance to that of the classical projection method of Halperin-Lions-Wittmann-Bauschke (HLWB).

We observe empirically that the regularized nonsmooth method significantly outperforms the HLWB method. However, the HLWB has a convergence guarantee while the nonsmooth method is not monotonic and does not guarantee convergence due in part to singularity of the generalized Jacobian.

Our application to solving large-scale linear programs uses a parametrized projection problem. This leads to a *stepping stone external path following* algorithm. Other applications are finding triangles from branch and bound methods, and generalized constrained linear least squares. We include scaling methods that improve the efficiency and robustness.

Keywords: best approximation, projection methods, Halperin-Lions-Wittmann-Bauschke algorithm, nonsmooth and semismooth methods, sparse large-scale linear programming, constrained linear least squares.

AMS subject classifications: 46N10, 49J52, 65K10, 90C05, 90C46, 90C59, 65F10

Contents

*PLEASE NOTE: We are including a table of contents, lists of tables, index, to help the referees. We fully intend to delete these before any final version of the paper.

†Departement of Mathematics, University of Haifa, Mt. Carmel, Haifa 3498838, Israel. Research supported by the ISF-NSFC joint research plan Grant Number 2874/19.

‡Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1; Research supported by The Natural Sciences and Engineering Research Council of Canada

List of Tables

List of Algorithms

List of Figures

1 Introduction

The *best approximation problem, BAP* arises in many areas of optimization and approximation theory. In particular, we study finding the best approximation x^* to a given point v from a *polyhedral set,* $P \subset \mathbb{R}^n$, n-dimensional Euclidean space; namely, find $x^*(v) \in \mathbb{R}^n$

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

There is an abundance of theory, algorithms, and applications for this problem. We follow a Newton type approach of an *elegant* compact optimality condition, even though the corresponding Jacobian resulting from the optimality conditions is possibly nonsmooth and/or singular. We include a regularization, as well as an inexact approach for large-scale problems. Empirical evidence illustrates the surprising success of this approach.

We include several applications. In particular, we solve large-scale linear programming, (LP), problems using a parametrized projection problem. This introduces an efficient (*stepping stone*) external path following algorithm. In addition, we consider large-scale systems of triangle inequalities. In our applications we do not assume differentiability and/or nonsingularity of the generalized Jacobian. We introduce a Newton type approach for our applications that overcomes the nonsmooth difficulties by applying regularization and scaling. We then provide extensive testing and comparisons to illustrate the surprising high efficiency, accuracy, speed, and robustness of our propsed method.

The main contributions of the paper are as follows. (i) First, we present the basics for the main projection problem, see Theorem 2.1 below. This includes an application of the Moreau decomposition that yields a *single elegant equation* that captures all three, primal and dual feasibility and complementarity optimality conditions of the problem. (ii) Second, we present the nonsmooth, regularized Newton method. No line search is used. (See Section 2.1.1 below.) (iii) We show that the regularization from a modified Levenberg-Marquardt method yields a descent direction. (See Lemma 2.4 below.) (iv) We present our empirical test results that include an external path following approach to solving large-scale linear programs that fully exploits sparsity. (See Section 5 below.) (v) We compare computationally our algorithm with the HLWB algorithm that belongs to a class of projection methods usually developed and investigated in the field of fixed point theory.

1.1 Related Work

Our approach uses a special decomposition from the optimality conditions that allows for a Newton method with a cone projection applied to a system whose size is of the order of the number of linear equality constraints forming the polyhedron P. This approach first appeared in infinite dimensional Hilbert space applications, e.g., [11, 17, 18, 37], where the projection mapping is differentiable, and typically P is the intersection of a cone and a linear manifold. This approach was applied to a parametrized quadratic problem to solve finite-dimensional linear programs in [44]. (See our application Section 4.1, below. In this finite-dimensional case differentiability was lost.) The approach in infinite-dimensional Hilbert spaces was followed up and extended in the theory of *partially finite programs* in [9, 10] and the many references therein. Further references are given in [3, 32, 43].

As mentioned above, differentiability is lost in the finite-dimensional cases, e.g., in [44]. This led to the application of semismoothness [38]. In particular, semismoothness for a nondifferentiable Newton type method is introduced and applied in [39, 40]. Further applications for nearest doubly stochastic and nearest Euclidean distance matrices are presented in [2, 30]. A regularized semismooth approach for general composite convex programs is given in [45].

The optimum point $x^*(v)$ is often called the *projection of* v *onto the polyhedral set* and is known to be unique. Differentiability properties are nontrivial as discussed in, e.g., [29]. A characterization of differentiability in terms of normal cones is given in [23]. Further results and connections to semismoothness is in, e.g., [25, 29]. A survey presentation on differentiability properties is at [42].

2 Projection onto a Polyhedral Set

We begin with the projection onto the polyhedral set given in *standard form*, since every polyhedron can be transformed into this form. Suppose we are given $v \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m\times n}$, rank $A = m$. We define the following *projection onto a polyhedral set*, i.e., the *best approximation problem, BAP* to the *generalized simplex*,

$$
\text{(P)} \qquad
\begin{array}{rcl}
x^*(v) := & \operatorname{argmin}_x & \frac{1}{2}\left\|x-v\right\|^2 \\
 & \text{s.t.} & Ax = b \\
 & & x \in \mathbb{R}^n_+,
\end{array}
\tag{2.1}
$$

$$
\text{optimal value: } p^*(v) \quad = \quad \frac{1}{2}\left\|x^*(v)-v\right\|^2,
$$

i.e., the optimum and optimal value are, respectively, $x^*(v), p^*(v)$; and $\mathbb{R}^n_+$ is the nonnegative orthant. We now proceed to derive the regularized nonsmooth Newton method, (***RNNM***) to solve (2.1).

2.1 Basic Theory and Algorithm

In this section we briefly describe the properties of problem (2.1) as well as some background and motivation behind using a generalized Newton method. We assume that

$$P := \{x \in \mathbb{R}^n_+ : Ax = b\} \neq \varnothing. \tag{2.2}$$

Problem (2.1) has a strongly convex smooth objective function and nonempty closed convex constraint set. Therefore, the optimal value is finite, uniquely attained, and strong duality holds. In the following, we precisely formulate this conclusion.

Throughout the rest of the paper we set[1]

$$F(y) := A(v + A^T y)_+ - b, \quad f(y) := \frac{1}{2}\|F(y)\|^2. \tag{2.3}$$

Theorem 2.1. *Consider the generalized simplex best approximation problem (2.1) with primal optimal value and optimum $p^*(v)$ and $x^*(v)$, respectively. Then the following hold:*

(i) The optimum $x^*(v)$ exists and is unique. Moreover, strong duality holds and the dual problem of (2.1) is the maximization of the dual functional, $\phi(y,z)$:

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

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

$$F(y) = 0 \iff y \in \operatorname{argmin} f(y) \text{ and } x^*(v) = (v + A^T y)_+. \tag{2.4}$$

Proof. Recall that the *Lagrangian* $L(x,y,z)$ for (2.1), and its gradient, are respectively

$$L(x,y,z) = \frac{1}{2}\|x - v\|^2 + y^T(b - Ax) - z^T x, \quad \nabla_x L(x,y,z) = x - v - A^T y - z. \tag{2.5}$$

(i): The solution of the problem (2.1) is a projection onto a nonempty polyhedral set, which is a closed and convex set, see (2.2). Therefore, the optimum exists and is unique and strong duality holds, i.e., there is a zero duality gap and the dual is attained.

Let x be a stationary point of the Lagrangian i.e., $\nabla_x L(x,y,z) = 0$. Then we have the following equivalent representation

$$x = v + A^T y + z.$$

[1] Let $x \in \mathbb{R}^n$. Here and elsewhere we use x_+ (respectively x_-) to denote projection of the vector x onto the nonnnegative orthant defined by $x_+ = (\max\{0, x_i\})_{i=1}^n$ (respectively onto the nonpositive orthant defined by $x_- = (\min\{0, x_i\})_{i=1}^n$) .

It then follows that at a stationary point x we have

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

The Lagrangian dual is

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

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

(ii): Now the *KKT optimality conditions* for the primal-dual variables (x, y, z) are[2]:

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

142 The above KKT conditions can be rewritten as :

$$\begin{bmatrix} x - v - A^Ty - z\\ Ax - b\\ z^Tx \end{bmatrix} = \begin{pmatrix}0\\0\\0\end{pmatrix}, \quad x, z\in\mathbb{R}^n_+, y\in\mathbb{R}^m. \tag{2.6}$$

It follows from the dual feasibility that $v + A^Ty = x - z = x + (-z)$. Together with the complementary slackness we have

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

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

$$x = (v + A^Ty)_+ \text{ and } -z = (v+A^Ty)_-;\text{ equivalently, } z = -(v+A^Ty)_-. \tag{2.7}$$

Substituting for $x = (v + A^Ty)_+$ we obtain a simplification of the optimality conditions in (2.6) as follows

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

143 equivalently; $F(y) = 0$, for some $y \in \mathbb{R}^m$. The inverse implication is clear. □

[2] Let $S \subset \mathbb{R}^n$. Here and elsewhere we use S^+ to denote the *polar cone* of the set S.

2.1.1 Nonlinear Least Squares; Jacobians

The BAP as described in (2.1) is equivalent to the minimization of $f(y)$ in (2.3), i.e, to a nonlinear least squares problem where the nonlinearity arises from the projection.

This system can be recharacterized by introducing the, possibly nonsmooth, projection of a vector p onto the nonnegative, respectively nonpositive, orthant denoted $p_+ = \operatorname{argmin}_x\{\|x-p\| : x \geq 0\}$, respectively $p_- = \operatorname{argmin}_x\{\|x-p\| : x \leq 0\}$. In general, we can define the *Moreau decomposition* of p with respect to $\mathbb{R}^n_+$ as $p = p_+ + p_-$, $p_+^T p_- = 0$.

Note that in the differentiable case the gradient of the squared residual $f(y)$ is

$$\nabla f(y) = (F'(y))^* F(y),$$

where $(\cdot)^*$ denotes the adjoint (here adjoint is transpose and F' denotes the Jacobian matrix). We note that we have differentiability of the function $h(w) := w_+$ if, and only if, $\{i : w_i = 0\} = \varnothing$ if, and only if, $w - w_+$ is in the relative interior of the normal cone of $\mathbb{R}^n_+$ at w_+ (negative of the polar cone at w_+).

We now discuss the framework of nonsmooth terminology needed to discuss generalized gradients.

Definition 2.2 ((local) Lipschitz continuity)**.** *Let $\Omega \subseteq \mathbb{R}^n$. A function $H : \Omega \to \mathbb{R}^n$ is Lipschitz continuous on Ω if there exists $K > 0$ such that*

$$\|H(y) - H(z)\| \leq K\|y - z\|, \forall y, z \in \Omega.$$

H is locally Lipschitz continuous on Ω if for each $x \in \Omega$ there exists a neighbourhood U of x such that H is Lipschitz continuous on U.

Let $\Omega \subseteq \mathbb{R}^n$. It follows from Rademacher's Theorem [24,41] that if $H : \Omega \to \mathbb{R}^n$ locally Lipschitz on Ω then H is Frechét differentiable almost everywhere on Ω. Following Clarke [19, Def. 2.6.1], we recall the following definition of the *generalized Jacobian*[3]

Definition 2.3 (generalized Jacobian)**.** *Suppose that $H : \mathbb{R}^m \to \mathbb{R}^m$ is locally Lipschitz. Let D_H be the set of points such that F is differentiable. Let $H'(y)$ be the usual Jacobian matrix at $y \in D_H$. The generalized Jacobian of G at y, $\partial H(y)$, is the convex hull*[4] *of all matrices obtained as the limit of usual Jacobians, defined as follows*

$$\partial H(y) = \operatorname{conv}\left\{ \lim_{\substack{y_i \to y \\ y_i \in D_H}} H'(y_i) \right\}.$$

In addition, $\partial H(y)$ is called nonsingular if every $V \in \partial H(y)$ is nonsingular.

Let $H : \mathbb{R}^m \to \mathbb{R}^m$ be locally Lipschitz. In the differentiable case, if $H'(y)$ is invertible, the Newton direction is the solution of the Newton equation

$$(H'(y))^*(H'(y))\Delta y = -(H'(y))^* H(y) \text{ ; equivalently, } \quad H'(y)\Delta y = -H(y).$$

[3] For our application we restrict ourselves to square Jacobians.

[4] Let $S \subset \mathbb{R}^n$. The convex hull of S, denoted $\operatorname{conv}(S)$ is the smallest convex set containing S.

Solving for Δy yields

$$\Delta y = -\left((H'(y))^*(H'(y))\right)^{-1}(H'(y))^*H(y) = -H'(y)^{-1}H(y). \tag{2.8}$$

Therefore, the directional derivative of f in the direction of Δy therefore satisfies

$$\begin{array}{rcl}\Delta y^T \nabla f(y) & = & -\left((H'(y))^*H(y)\right)^T \left((H'(y))^*(H'(y))\right)^{-1}(H'(y))^*H(y) \\ & < & 0,\end{array}$$

Hence Δy is a descent direction in this case.

The *Levenberg-Marquardt* method is a popular method for handling singularity in $(H'(y))^*(H'(y))$ by using the substitution/regularization $(H'(y))^*H'(y) \leftarrow ((H'(y))^*H'(y)) + \lambda I, \lambda > 0$. We now see that we maintain a descent direction with a similar simplified approach.

Lemma 2.4. *Let $y \in \mathbb{R}^m$. Suppose that $F(y) = 0$. Let $\lambda > 0$ and let Δy be the solution of*

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

Then Δy is the (simplified) Levenberg-Marquardt direction and is always a descent direction.

Proof. For simplicity, set $J = J(y) = F'(y)$, and observe that J is positive semidefinite. The regularization of Levenberg-Marquardt type uses

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

The positive semidefiniteness of J implies that $J + \lambda I$ is investible, hence

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

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

$$\begin{array}{rcl}\Delta y^T \nabla f(y) & = & -\left((J+\lambda I)^{-1}(J^TF)\right)^T J^TF \\ & = & -(J^TF)^T\left((J+\lambda I)^{-1}\right)J^TF \\ & < & 0.\end{array}$$

This completes the proof. □

2.1.2 Maximum Rank Generalized Jacobian

Recall the optimality conditions derived following (2.6). If we denote the orthogonal projection operator onto the nonnegative orthant by $\mathcal{P}_+(w) = w_+$, then

$$Aw_+ = A(\mathcal{P}_+w) = (A\mathcal{P}_+)w_+ = (A\mathcal{P}_+)(\mathcal{P}_+w) = (A\mathcal{P}_+)w_+ = \sum_{w_i \geq 0} A_i w_i.$$

Here A_i is the i-th column of A. Thus we see that at points where the projection is differentiable, the columns of A that are chosen correspond to the nonnegative (basic) variables of w. We note that

$$v + A^Ty \geq 0 \implies F'(\Delta y) = AIA^T\Delta y = AA^T\Delta y.$$

Following [30], we define the following set

$$\mathcal{U}(y) := \left\{ u \in \mathbb{R}^n \ , \ u_i \in \ \left\{ \begin{array}{cl} 1, & \text{if } (v + A^T y)_i > 0, \\ [0,1]\,, & \text{if } (v + A^T y)_i = 0, \\ 0, & \text{if } (v + A^T y)_i < 0. \end{array} \right. \right\}. \tag{2.9}$$

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

$$\partial F(y) = \{A \ \mathrm{Diag}(u)\, A^T \mid u \in \mathcal{U}(y)\}. \tag{2.10}$$

Let $y_0 \in \mathbb{R}^m$. The nonsmooth Newton method for solving $F(y) = 0$ generates the following iterates

$$y^{k+1} = y^k - V_k^{-1} F(y^k),\ V_k \in \partial F(y^k). \tag{2.11}$$

Let

$$\mathcal{I}_+ := \mathcal{I}_+(y) = \{i : \mathrm{sign}_+(v + A^T y) = 1\}, \quad \mathcal{I}_0 := \mathcal{I}_0(y) = \{i : \mathrm{sign}_+(v + A^T y) = 0\}.$$

We note that, defining $M = \mathrm{Diag}(u)$,

$$AMA^T := A\,\mathrm{Diag}(u)A^T = \sum_{i \in \mathcal{I}_+} A_i A_i^T + \sum_{i \in \mathcal{I}_0} \alpha_i A_i A_i^T, \quad \alpha_i \in [0,1], \forall i \in \mathcal{I}_0.$$

Then the maximum (resp. minimum) rank for AMA^T is obtained by choosing $\alpha_i = 1, \forall i \in \mathcal{I}_0$ ($\alpha_i = 0, \forall i \in \mathcal{I}_0$, resp.). We use the modified sign function

$$\mathrm{sign}_+(w) = \left\{ \begin{array}{ll} 1, & \text{if } w \geq 0, \\ 0, & \text{if } w < 0. \end{array} \right.$$

Then the *maximum rank generalized Jacobian* is obtained from

$$AMA^T = \sum_{i \in \mathcal{I}_+} A_i A_i^T.$$

2.1.3 Vertices and Polar Cones

In our tests we can decide on the characteristics of the optimal solution using the properties of (degenerate) vertices.

Lemma 2.5 (*vertex* and *polar 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 nonsingular,*
3. *the corresponding generalized Jacobian, (2.10), is nonsingular.*

Moreover, the polar cone of the feasible set P at $x = x(y)$ is

$$(P - x)^+ = \{w : w = A^T u + z, u \in \mathbb{R}^m, z \in \mathbb{R}^n_+, x^T z = 0\}. \tag{2.12}$$

Proof. Without loss of generality we can permute the columns of A using the index sets $\mathcal{I}_+, \mathcal{I}_0$, and have $A = [A_{\mathcal{I}_+} \ A_{\mathcal{I}_0}]$. Therefore, the active set of equality constraints is

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

This has the unique solution $x(y)$ if, and only if, $A_{\mathcal{I}_+}$ is nonsingular.

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

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

where e_j is the j-th unit vector. And we know that $x - v$ is in the polar cone at x if, and only if, x is optimal. Therefore at a vertex, this yields the description of the polar cone at x. □

Remark 2.6 (degeneracy of optimal solutions)**.** *Let x be a boundary point of P. Then the polar cone of P at x is given in (2.12). Moreover, x is the optimal solution of (2.1) if, and only if, $x - v \in (P - x)^+$, i.e., we can choose v with*

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

In fact, we can choose z so that $x + z > 0$ and have no degeneracy or choose $z = 0$ and have high degeneracy. For these choices we still get x optimal. As mentioned above, it is shown in [23] that

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

This justifies our use of the Levenberg-Marquardt regularization.

The pseudocodes for solving (2.1) using the exact and inexact nonsmooth Newton methods are presented below in Appendix A in Algorithms A.1 and A.2, respectively.

3 Cyclic HLWB Projection for Best Approximation

A notable aspect of this work is the computational comparison of our semismooth algorithm with the method of Halpern-Lions-Wittmann-Bauschke, (HLWB). The convergence analysis of the method has its roots in the field of fixed point theory. For the readers' convenience we provide a brief description and some relevant references.

Problem 3.1 (*best approximation problem for linear inequalities*)**.** *Given an $m \times n$ matrix A and a vector $b \in R^m$ such that*

$$Q := \{x \in R^n : Ax \leq b\} \neq \varnothing, \tag{3.1}$$

and a point $v \in R^n$, $v \notin Q$, called the anchor point, find the orthogonal projection of v onto Q, denoted by $P_Q(v)$.

The set Q is the intersection of m half-spaces. Denote the i-th half-space of (3.1) by

$$H_i := \{x \in R^n \ : \ x^T a^i \leq b_i\}. \tag{3.2}$$

The orthogonal projection of a point $v \in R^n$ onto H_i, denoted by $P_i(v)$, is

$$P_i(v) = v + \min\left\{0, \frac{b_i - y^T a^i}{\|a^i\|^2}\right\} a^i. \tag{3.3}$$

The HLWB algorithm for this problem is a *projection method* that employs projections onto the individual half-spaces of (3.2) and makes use of a sequence of, so called, steering parameters.

Definition 3.2 (*steering sequence*)**.** *A real sequence* $(\sigma_k)_{k=0}^{\infty}$ *is called a steering sequence if it has the following properties:*

$$\begin{array}{ll} \sigma_k \in [0,1] \ \textit{for all} \ k \geq 0, \ \textit{and} \ \lim_{k\to\infty} \sigma_k = 0, & \\ \sum_{k=0}^{\infty} \sigma_k = \infty & \left(\textit{or, equivalently,} \ \prod_{k=0}^{\infty}(1-\sigma_k) = 0\right), \\ \sum_{k=0}^{\infty} |\sigma_{k+1} - \sigma_k| < \infty. & \end{array} \tag{3.4}$$

Observe that although $\sigma_k \in [0,1]$, the definition rules out the option of choosing all σ_k equal to zero or all equal to one because of contradictions with the other properties. The third property in (3.4) was introduced by Wittmann, see, e.g., the review paper of López, Martin-Márquez and Xu [33].

Algorithm 3.1 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

$$x^{k+1} = \sigma_k v + (1 - \sigma_k) P_{i(k)}(x^k), \tag{3.5}$$

where v is the given anchor point, $i(k) = k \bmod m{+}1$ and $(\sigma_k)_{k=0}^{\infty}$ is a steering sequence.

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 [4], also Bauschke and Combettes [6, Chap. 30]. For more on the BAP, see, e.g., Deutsch's book [21]. The family of iterative projection methods for the BAP includes, in addition to the HLWB method, also Dykstra's algorithm [12], [6, Theorem 30.7], Haugazeau's algorithm [26], [6, Corollary 30.15], and Hildreth's algorithm [28, 31]. There are also simultaneous versions of some of these algorithms available, see, e.g., [13]. A string-averaging HLWB algorithm, which encompasses the sequential, the simultaneous and other variants of the HLWB algorithm, recently appeared in [14].

More on applications of BAP and the HLWB algorithm are given in Appendix C.

4 Applications

We consider several applications of the best approximation problem, (2.1). Perhaps the most interesting is the following approach to solving a linear program, LP.

4.1 Solving Linear Programs

We consider a maximization primal LP in standard equality form

$$\text{(PLP)} \qquad \begin{array}{rll} p^*_{LP} := & \max & c^T x \\ & \text{s.t.} & Ax = b \in \mathbb{R}^m \\ & & x \in \mathbb{R}^n_+. \end{array} \tag{4.1}$$

The dual LP is

$$\text{(DLP)} \qquad \begin{array}{rll} d^*_{LP} := & \min & b^T y \\ & \text{s.t.} & A^T y - z = c \in \mathbb{R}^n \\ & & z \in \mathbb{R}^n_+. \end{array} \tag{4.2}$$

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

We now see in Lemma 4.1 that the solution to (PLP) is the limit of the projection of the vector $v_R = Rc \in \mathbb{R}^n$ onto the feasible set as $R \uparrow \infty$.[5]

Lemma 4.1 ([34–36,44]). *Let the given LP data be A, b, c with finite optimal value p^*_{LP}. For each $R > 0$ define*

$$\begin{array}{rll} x^*(R) := & \operatorname{argmin}_x & \frac{1}{2}\|x - Rc\|^2 \\ & s.t. & Ax = b \in \mathbb{R}^m \\ & & x \in \mathbb{R}^n_+. \end{array} \tag{4.3}$$

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

$$R \geq \bar{R} \implies x^* = x^*(R) = \operatorname{argmin}\left\{\frac{1}{2}\|x - Rc\|^2 \,:\, Ax = b,\, x \in \mathbb{R}^n_+\right\}. \tag{4.4}$$

In our application, as we would like an R that is not too large but large enough so that $Rc > \|x^*\|$. We use the estimate

$$R = \min\left\{50, \frac{\sqrt{mn}\,\|b\|}{1 + \|c\|}\right\}. \tag{4.5}$$

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 4.2. *Let $A, b, c, R, x^*(R)$ be defined as in Lemma 4.1. Then*

$$\begin{array}{rll} \frac{1}{R}x^*(R) = w^*(R) := & \operatorname{argmin}_w & \frac{1}{2}\|w - c\|^2 \\ & s.t. & Aw = \frac{1}{R}b \in \mathbb{R}^m \\ & & w \in \mathbb{R}^n_+. \end{array} \tag{4.6}$$

Proof. From

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

we substitute for x and obtain: $A(Rw) = b \iff Aw = \frac{1}{R}b$. The result follows from the observation that argmin does not change after discarding the constant R^2. □

[5] Note that our algorithm identifies infeasibility but we do not consider that aspect in this paper.

4.1.1 Warm Start; Stepping Stone External Path Following

We consider the scaling in Corollary 4.2 and recall the relation between the scaling for c with variable x:

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

(To simplify notation, we ignore the optimality symbol $(\cdot)^*$.) The optimality conditions from Theorem 4.5 for $w = w(R)$ in Corollary 4.2 are:

$$\begin{bmatrix} w - c - A^T y - z \\ Aw - \frac{1}{R} b \\ z^T w \end{bmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad w, z \in \mathbb{R}^n_+, y \in \mathbb{R}^m. \tag{4.7}$$

We conclude that

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

The optimality conditions are now

$$w = c + A^T y + z, \, b = ARw = AR(c + A^T y)_+, \quad w^T z = 0, x, z \geq 0. \tag{4.8}$$

This means that $\|w\|$ is an estimate for the error in dual feasibility, i.e., an estimate for the accuracy of Rw as the optimum of the original LP.

Given the current R and the approximate optimal triple (w, y, z), we would like to find a good new R_n and a corresponding y_n to send to the projection algorithm for a warm start process. We use sensitivity analysis for the projection problem. In the sequel $A^\dagger$ denotes the generalized (Moore-Penrose) inverse of a matrix A.

Theorem 4.3. *Suppose that the triple (w, y, z) is optimal for (4.6); i.e., satisfies (4.7). Let*

$$\begin{array}{c} \mathcal{N} = \mathcal{N}(z) = \{i \, : \, z_i > 0\}, \quad \mathcal{B} = \mathcal{B}(w) = \{1 : n\} \backslash \mathcal{N}; \\ b_{\mathcal{B}} = A_{\mathcal{B}}^T \left(A_{\mathcal{B}} A_{\mathcal{B}}^T\right)^\dagger b, \quad b_{\mathcal{N}} = A_{\mathcal{N}}^T \left(A_{\mathcal{B}} A_{\mathcal{B}}^T\right)^\dagger b; \\ e = \begin{pmatrix} (b_{\mathcal{B}} - Rw_{\mathcal{B}}) \\ -(b_{\mathcal{N}} + Rz_{\mathcal{N}}) \end{pmatrix}, \quad f = \begin{pmatrix} Rb_{\mathcal{B}} \\ -Rb_{\mathcal{N}} \end{pmatrix}. \end{array} \tag{4.9}$$

Then the maximum value for increasing R without changing the basis is

$$R_n = \min\{e_i / f_i : e_i > 0\}. \tag{4.10}$$

The corresponding changes $\Delta w, \Delta y, \Delta z$ that result in $w + \Delta w, y + \Delta y, z + \Delta z$ optimal for R_n are given in the proof that follows.

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

Proof. We want to find the maximum increase in R that keeps the current basis $\mathcal{B}$ optimal for (4.6). We have

$$\begin{array}{c} A_{\mathcal{B}}(w_{\mathcal{B}} + \Delta w) = \frac{1}{R_n} b \implies A_{\mathcal{B}} \Delta w = \left(\frac{1}{R_n} - \frac{1}{R}\right) b \\ w_B + \Delta w - c_{\mathcal{B}} - A_{\mathcal{B}}^T (y + \Delta y) = 0 \implies \Delta w = A_{\mathcal{B}}^T (\Delta y) \implies A_{\mathcal{B}} A_{\mathcal{B}}^T (\Delta y) = \left(\frac{R - R_n}{R R_n}\right) b \\ -c_{\mathcal{N}} - A_{\mathcal{N}}^T (y + \Delta y) - (z_{\mathcal{N}} + \Delta z) = 0 \implies \Delta z = -A_{\mathcal{N}}^T (\Delta y) \end{array}$$

We now set

$$\Delta y_p = \left(A_{\mathcal{B}} A_{\mathcal{B}}^T\right)^{\dagger} b, \quad \Delta y = \left(\frac{R - R_n}{RR_n}\right) \Delta y_p.$$

We have

$$-w_{\mathcal{B}} \leq \Delta w = A_{\mathcal{B}}^T \left(\frac{R - R_n}{RR_n}\right) \Delta y_p = -\left(\frac{R_n - R}{RR_n}\right) A_{\mathcal{B}}^T \left(A_{\mathcal{B}} A_{\mathcal{B}}^T\right)^{\dagger} b =: -\left(\frac{R_n - R}{RR_n}\right) b_{\mathcal{B}}.$$

We get that

$$(R_n - R) b_{\mathcal{B}} \leq (RR_n) w_{\mathcal{B}} \implies R_n(b_{\mathcal{B}} - R w_{\mathcal{B}}) \leq R b_{\mathcal{B}}.$$

To find the maximum R_n and check that it is not $R_n = \infty$, we use an LP type ratio test. We set the two vectors to be $e = (b_{\mathcal{B}} - R w_{\mathcal{B}})$, $f = R b_{\mathcal{B}}$. Note that the inequality holds trivially for $R_n = R$. Therefore, we cannot have $e_i > 0, f_i \leq 0$. We choose R_n to be the maximum that satisfies:

$$\max_i \{f_i/e_i, \text{ if } f_i < 0, e_i < 0\} \leq R_n = \min_i \{f_i/e_i, \text{ if } f_i > 0, e_i > 0\},$$

where the minimum over the empty set is taken to be $+\infty$.

We now need to similarly do a ratio test for z. We have

$$-z_{\mathcal{N}} \leq \Delta z = -A_{\mathcal{N}}^T \left(\frac{R - R_n}{RR_n}\right) \Delta y_p = \left(\frac{R_n - R}{RR_n}\right) A_{\mathcal{N}}^T \left(A_{\mathcal{B}} A_{\mathcal{B}}^T\right)^{\dagger} b =: \left(\frac{R_n - R}{RR_n}\right) b_{\mathcal{N}}.$$

We get that

$$(R_n - R) b_{\mathcal{N}} \geq -(RR_n) z_{\mathcal{N}} \implies R_n(-b_{\mathcal{N}} - R z_{\mathcal{N}}) \leq -R b_{\mathcal{N}}.$$

We again find the maximum R_n and check that we do not have $R_n = \infty$ using an LP type ratio test. We set the two vectors to be $e = -(b_{\mathcal{N}} + R z_{\mathcal{N}})$, $f = -R b_{\mathcal{N}}$. Recall that the inequality holds trivially for $R_n = R$. Therefore, we cannot have $e_i > 0, f_i \leq 0$. We choose R_n to be the maximum that satisfies:

$$\max_i \{f_i/e_i, \text{ if } f_i < 0, e_i < 0\} \leq R_n = \min_i \{f_i/e_i, \text{ if } f_i > 0, e_i > 0\}.$$

We choose R_n as the minimum of the above two values found.

Finally, if $R_m = \infty$, then the basis does not change as R increases to infinity, i.e., the optimal basis has been found. □

The above Theorem 4.3 illustrates the external path following algorithm that we are using. The theorem finds specific values of R, *stepping stones on the path*, where the current choice of columns of A changes. Once we find that the next *stepping stone* is at infinity, we know that we have found the optimal choice of columns of A. Thus we have an external path following algorithm with parameter R but we only choose specific points on this path to *step* on.

4.1.2 Upper and Lower Bounds for the LP Problem

The optimal solution from the projection problems (4.3) and (4.6) provides a feasible x, and we get the corresponding LP *lower bound* $c^T x^*(R)$. The upper bound is not as easy and more important in stopping the algorithm.

Note that in Section 4.1.1 primal feasibility and complementary slackness hold for $x(R) = Rw, z$ and this is identical for the LP problem. We therefore need to find y_{LP} to satisfy the LP dual feasibility

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

But, from the projection problem optimality conditions we have

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

As seen above, this means that in the limit, w is small and we do get dual feasibility $y(R) \to y_{\mathsf{LP}}$. But at each iteration we actually have

$$z - w = A^T(-y) - c,\ z, w \geq 0, z^T w = 0, \quad y \cong y_R. \tag{4.11}$$

We can write the required dual feasibility equations using the indices for $w_i > 0$.

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

Recall the definitions of $\mathcal{N}, \mathcal{B}$ in (4.9). Then for a given y_R from the optimality conditions from the projection problem (4.11), we consider the nearest dual LP feasible system with unknowns $z \geq 0, y_{\mathsf{LP}}$. Note that we are using the projection with free variables, Section 4.2.

Lemma 4.4. *Let $w, y = y_R, z$ be approximate optimal solutions from (4.8) and $\mathcal{B}$ the support defined in (4.9). Consider the nearest dual feasibility program*

$$\begin{array}{rll} \begin{pmatrix} y_{\mathsf{LP}}^* \\ z_{\mathsf{LP}}^* \end{pmatrix} \in & \operatorname{argmin} & \frac{1}{2}\|(-y_R) - y_{\mathsf{LP}}\|^2 + \frac{1}{2}\|0 - z_{\mathcal{B}}\|^2 + \frac{1}{2}\|(z_R)_{\mathcal{N}} - z_{\mathcal{N}}\|^2 \quad \left(= \frac{1}{2}\|v - x\|^2\right) \\ & s.t. & \begin{bmatrix} A_{:\mathcal{B}}^T & -I & 0 \\ A_{:\mathcal{N}}^T & 0 & -I \end{bmatrix} \begin{pmatrix} y_{\mathsf{LP}} \\ z_{\mathcal{B}} \\ z_{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} c_{\mathcal{B}} \\ c_{\mathcal{N}} \end{pmatrix} \\ & & y_{\mathsf{LP}} \text{ free},\ z_{\mathsf{LP}} = \begin{pmatrix} z_{\mathcal{B}} \\ z_{\mathcal{N}} \end{pmatrix} \geq 0. \end{array} \tag{4.12}$$

Then the optimal value of the LP (4.1) *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. □

4.2 Projection and Free Variables

For many applications, some of the variables are free and not all the variables are in the objective function. We consider these two cases. Note this can arise when the objective is a general least squares problem e.g., $\min \|Bx-c\|^2$ and we add the constraint $Bx - w = 0$ and substitute the free variable w into the objective function.

4.2.1 Projection with Free Variables

We first consider the problem with some of the variables free:

$$
\text{(P)} \qquad \begin{array}{rcl}
x(v) := & \operatorname{argmin}_{x_1,x_2} & \frac{1}{2}\|x-v\|^2, \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \, v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \\
 & \text{s.t.} & Ax = b \in \mathbb{R}^m \\
 & & x_1 \in \mathbb{R}^{n_1}_+, \, x_2 \in \mathbb{R}^{n_2}, \\
 & & \\
\text{optimal value: } p^*_f(v) & = & \frac{1}{2}\|x(v)-v\|^2,
\end{array} \tag{4.13}
$$

Theorem 4.5. *Consider the generalized simplex best approximation problem with free variables (4.13). Assume that the feasible set is nonempty. Then the optimum $x(v)$ exists and is unique. Moreover, let*

$$
F_f(y) := A\begin{pmatrix} ((v+A^Ty)_1)_+ \\ (v+A^Ty)_2 \end{pmatrix} - b, \quad f_f(y) = \frac{1}{2}\|F_f(y)\|^2. \tag{4.14}
$$

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

$$
x(v) = \begin{pmatrix} ((v+A^Ty)_1)_+ \\ (v+A^Ty)_2 \end{pmatrix}, \text{ for any root } F_f(y) = 0. \tag{4.15}
$$

Let $p^*_f(v) = \frac{1}{2}\|x(v)-v\|^2$ denote the primal optimal value. Then strong duality holds and the dual problem of (4.13) is the maximization of the dual functional, $\phi_f(y,z_1)$:

$$
p^*_f(v) = d^*_f(v) := \max_{z_1 \in \mathbb{R}^{n1}_+, y \in \mathbb{R}^m} \phi(y,z_1) := -\frac{1}{2}\left\|\begin{pmatrix} z_1 \\ 0 \end{pmatrix} - A^Ty\right\|^2 + y^T(Av-b) - z_1^Tv_1.
$$

Proof. We modify the proof of Theorem 2.1. The *Lagrangian*, $L_f(x,y,z)$ for (4.13) is

$$
L_f(x,y,z) = \frac{1}{2}\|x-v\|^2 + y^T(b-Ax) - z_1^Tx_1, \quad \nabla_x L_f(x,y,z) = x - v - A^Ty - \begin{pmatrix} z_1 \\ 0 \end{pmatrix}. \tag{4.16}
$$

Solving for a stationary point means

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

Therefore, with this definition of z, we still have at a stationary point that

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

As in Theorem 2.1, the problem (4.13) is a projection onto a nonempty polyhedral set, a closed and convex set. The optimum exists and is unique and strong duality holds, i.e., there is a zero duality gap $p_f^* = d_f^*$, and the dual value is attained. The Lagrangian dual is

$$\begin{array}{rcll} d^* & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y} \min_x & L_f(x,y,z) = \frac{1}{2}\|x - v\|^2 + y^T(b - Ax) - z_1^Tx_1 \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y, x} & \{L_f(x,y,z_1) : \nabla_x L_f(x,y,z_1) = 0\} \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y, x} & \{L_f(x,y,z) : x = v + A^Ty + z\} \\ & = & \max_{z_1 \in \mathbb{R}^{n_1}_+, y} & -\frac{1}{2}\left\|z + A^Ty\right\|^2 + y^T(b - Av) - z^Tv. \end{array}$$

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

$$\begin{array}{ll} \nabla_x L_f(x,y,z) = x - v - A^Ty - z = 0, & \text{(dual feasibility)} \\ \nabla_y L_f(x,y,z) = Ax - b = 0, & \text{(primal feasibility)} \\ \nabla_z L_f(x,y,z) \cong x \in (\mathbb{R}^n_+ - z)^+. & \text{(complementary slackness } z_1^Tx_1 = 0) \end{array}$$

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

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

Note $v + A^Ty = x - z = x + (-z)$. Therefore this is a Moreau decomposition of $v + A^Ty$, with $x^Tz = 0, x, z \in \mathbb{R}^n_+$, $x = (v + A^Ty)_+$. Therefore, we get $A(v + A^Ty)_+ = b$, where we modify the definition of $_+$ so that we project only the first part corresponding to x_1 onto the nonnegative orthant $\mathbb{R}^{n_1}_+$ and then this means $z_1 = -\left((v + A^Ty)_1\right)_-$.

We get the optimality conditions

$$\begin{array}{rl} A\begin{pmatrix} \left((v + A^Ty)_1\right)_+ \\ (v + A^Ty)_2 \end{pmatrix} & = b, \; x_1 = \left((v + A^Ty)_1\right)_+, x_2 = (v + A^Ty)_2 \\ \Longrightarrow & z = -(v + A^Ty)_-, z^Tx = 0, x, z \in \mathbb{R}^n_+, x - v - A^Ty - z = 0, \end{array}$$

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

For a vertex, a BFS, we need n active constraints. The equality constraints $Ax = b$ account for m, leaving $n - m$ to choose among $1, \ldots, n_1$, the constrained variables in x_1. This leaves

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

4.3 Triangle Inequalities

We can obtain an efficient projection onto a large set of triangle inequalities that arise as cuts in graph problems. We let $G = (V, E)$ denote a graph and

$$\mathcal{T} = \{(u, v, w) : u < v < w \in V\}$$

and define the triangle inequalities

$$(I) \quad \left\{ \begin{array}{rcl} x_{vw} - x_{uv} - x_{uw} & \leq & 0 \\ x_{uw} - x_{uv} - x_{vw} & \leq & 0 \\ x_{uv} - x_{vw} - x_{uw} & \leq & 0 \\ \forall (u, v, w) \in \mathcal{T} & & \\ 0 \leq x_{uv} \leq 1, \, \forall (u, v) \in E & & \end{array} \right\} \tag{4.17}$$

We could rewrite this as a standard feasibility seeking problem or best approximation problem, i.e. given a $\bar{x}$ we want to find the nearest point to $\bar{x}$ that satisfies a subset of triangle inequalities denoted with T:

$$\min \frac{1}{2}\|x - \bar{x}\|^2 \text{ s.t. } Tx + sI = 0, x + tI = e, \; x, s, t \geq 0.$$

By abuse of notation, we let $x = \begin{pmatrix} x \\ s \\ t \end{pmatrix}$.

$$A = \begin{bmatrix} T & I & 0 \\ I & 0 & I \end{bmatrix}, \quad b = \begin{pmatrix} 0 \\ e \end{pmatrix}.$$

Example 4.6 (Max-Cut Graph Problem)**.** *This means the graph has weights W_{ij} on the edges x_{ij}. We want to maximize $\frac{1}{4}\sum_{ij} W_{ij}(1 - z_i z_j)$, where z_i is ± 1 depending which set the i-th node is in. The constraint here is $z_i^2 = 1, \forall i$. The Laplacian $L = L(W)$ can be used to get the following equivalent problem*

$$\begin{array}{rll} \max_z & \operatorname{trace} ZW & = (z^T L z) \\ s.t. & \operatorname{diag}(Z) = e & (Z = zz^T) \\ & Z \succeq 0 & \end{array}$$

The relaxation ignores the rank one constraint on Z. If the optimal Z is rank one we can recover the optimal solution for the original NP-hard MC problem using the factorization $Z = zz^T$. Otherwise you can use the first column of the eigenvector for the largest eigenvalue as an approximate and do a rounding. (Goemans-Williamson Theorem guarantees 87.14 approx percent of optimal value)

The SDP relaxation is

$$\begin{array}{rl} \max_z & \operatorname{trace} ZW \\ s.t. & \operatorname{diag}(Z) = e \\ & Z \succeq 0 \end{array}$$

This relaxation is an excellent relaxation but if it fails we can add violated triangle inequalities to improve the solution. In a splitting approach we need an efficient projection onto a set of triangle inequalities.

Example 4.7 (Binary)**.** *For a binary* $0,1$ *problem with* $x \in \mathbb{R}^n$ *we add the constraint* $x_i^2 - x_i = 0, \forall i$ *and then lift to matrix space*

$$Y_x = \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^T.$$

We now relax the rank constraint and solve an SDP wiht $Y_x \succeq 0$*. For example with the added constraint that* $Ax = b$*. We choose* V *so that* $\operatorname{range}(V) = Null\left(\begin{bmatrix} -b^T \\ A^T \end{bmatrix}\right)$ *and use the facial reduction*

$$Y_x = VRV^T,\ R \succeq 0.$$

The original problem is x *binary and* $Ax = b$*. We replace this by equivalent problem*

$$\min 0 \ s.t.\ \|Ax - b\|^2 = 0, x \circ x - x = 0.$$

We now look at the Lagrangian dual, homogenized with α*. We let* $y = \begin{pmatrix} \alpha \\ x \end{pmatrix}$*. The Lagrangian is*

$$\begin{array}{rcl} L(x,\lambda,w) & = & 0 + \lambda\|Ax - \alpha b\|^2 + \sum_i w_i(x_i^2 - \alpha x_i) + t(1-\alpha^2) \\ & = & \lambda\left(x^T A^T A x - 2\alpha b^T A x + \alpha^2 \|b\|^2\right) \\ & & \quad + \sum_i w_i(x_i^2 - \alpha x_i) + t - t\alpha^2 \\ & = & y^T \begin{bmatrix} \lambda\|b\|^2 - t & -\lambda b^T A - w^T/2 \\ -\lambda A^T b - w/2 & \lambda A^T A + \operatorname{Diag}(w)) \end{bmatrix} y + t. \end{array}$$

The Lagrangian dual is:

$$\begin{array}{rcl} d^* & := & \max_{\lambda,w,t} \min_{x,\alpha} L(x,\lambda,w) \\ & = & \max_{\lambda,w,t} \left\{ t : \begin{bmatrix} \lambda\|b\|^2 - t & -\lambda b^T A - w^T/2 \\ -\lambda A^T b - w/2 & \lambda A^T A + \operatorname{Diag}(w)) \end{bmatrix} \succeq 0 \right\} \\ & = & \begin{array}{cl} \max_{\lambda,w,t} & t \\ s.t. & \lambda \begin{bmatrix} \|b\|^2 & -b^T A \\ -A^T b & A^T A) \end{bmatrix} + \begin{bmatrix} 0 & -w^T/2 \\ -w/2 & \operatorname{Diag}(w)) \end{bmatrix} - tE_{00} \succeq 0 \end{array} \end{array}$$

5 Numerics

In this section we compare the Regularized Nonsmooth Newton Method, (***RNNM***), (exact and inexact) with the HLWB method [4] described in Section 3, as well as with Matlab's *lsqlin* interior point solver . Recall our BAP, (2.1), and the pseudocode for HLWB in Algorithm A.3 in Appendix A. We show that in our experiments ***RNNM*** (exact) significantly outperforms the other methods. These experiments are done with an i7-4930k @ 3.2GHz, 16 GBs of RAM, and Matlab 2022b software.

Before we see the differences in performance of the algorithms, we elaborate on how we implement the HLWB method, see also Section 3. HLWB projects onto individual convex sets, and then computes the next iterate, x^{k+1}, by taking a specific convex combination dictated by a sequence of steering parameters, see Definition 3.2, and the initial point v, commonly called the anchor Problem 3.1. Traditionally, each projection is called an *iteration*, and the collection of these iterations is defined as a *sweep*, e.g., [6]. In the context of our problem (2.1), HLWB is iterating onto one of the hyperplanes (sets) defined by A, denoted a_{i_k}, as well as the nonnegative orthant. We have completed a sweep once the projection onto all the hyperplanes and onto the nonnegative orthant have been completed. (See steps 14-16 of Algorithm A.3.) Thus we relate one sweep of HLWB with one iteration of ***RNNM*** .

5.1 Time Complexity

Since ***RNNM*** is a second-order method while HLWB is a first-order method, we now discuss theoretical time complexity differences. From the ***RNNM*** Algorithm, Algorithm A.1, we can see that worst-case time complexity is $O(m^3 + m^2 n)$ [6] flops, of which every step but solving the linear system is efficiently parallelizable. It is worth mentioning that in Step 6, the linear system we are solving is positive definite and sparse. Therefore, it can be solved efficiently using the Cholesky decomposition. From the HLWB Algorithm, Algorithm A.3, 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. [7]

From the perspective of theoretical time complexity it would be easy to assume that HLWB is the preferable algorithm as each of it's iterations are composed of operations that are completely parallelizable and each first-order sweep has an overall lower 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.

5.2 Comparison of Algorithms

When performing our numerical experiments, we refer to the discussion on techniques for comparisons of algorithms given in [8]. In particular, we include performance profiles [22] and tables of the performances for ***RNNM*** (exact and enexact), HLWB, and *lsqlin*.

We compare the HLWB algorithm to ***RNNM*** by generating the problem (2.1) such that v lies in the relative interior of the normal cone (negative of the polar cone) of a vertex of the feasible polyhedron, and therefore the vertex is the closest point to v. More specifically, since no convergence results for ***RNNM*** solving (2.1) as far as we know have been proven, for these experiments we ensure that $\|A\| = 1$, $\|v\| = 0.1$.

The ***RNNM*** Algorithm starts with initializing $x_0 \leftarrow (v + A^T y_0)_+$ where either $y_0 = 0_m$ or we are given a y_0 for a warm start as discussed in our LP application, then $x_0 \leftarrow (v + A^T y_0)_+$ reduces to $x_0 \leftarrow \max(v, 0)$ in the initialization stage of ***RNNM*** . 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.

Since ***RNNM*** solves a reduced KKT condition for a convex problem, then $\frac{\|F(y_k)\|}{1+\|b\|}$ is a sufficient relative residual and stopping condition for ***RNNM*** . Since HLWB is a first order method, it's stopping criterion will be measured at the end of a sweep as opposed to an iteration. Furthermore, HLWB does not have any proper stopping criterion but converges in the limit, so we will use primal

---

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

[7] See Algorithm A.3 lines 5-11, 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 respectvely per iteration that projects onto the nonnegative orthant is: $n+1+(n+2)+(mn+m+1) = mn+2n+m+4 = O(mn)$ flops of which all flops are efficiently parallelizable. Therefore, in terms of sweeps the HLWB method computes $m(mn + 3n + m + 6) + mn + 2n + m + 4 = m^2 n + 4mn + m^2 + 2n + 7m + 4 = O(m^2 n)$ flops.

feasibility as the stopping criterion, i.e., $\frac{\|Ay_k - b\|}{1+\|b\|}$. Note that we use y_k instead of x_k in the stopping criterion as y_k is nonnegative at the end of every sweep. Lastly, the *lsqlin* solver will be using it's first-order optimality conditions, which we will make relative by dividing by $1 + \|b\|$.

In Section 5.2.1, we generate problems such that v lies in the relative interior of the normal cone of a nondegenerate vertex. We also tested for degenerate vertices but observed very similar results. These tests, and the performance of the ***RNNM*** Algorithm motivates the theory and potential practice of using ***RNNM*** for LP applications, as seen in Section 5.3.

For the performance profiles in Section 5.2.1 we use the following notation from [8]. Let P be our set of problems, i.e., problems with changing m, n, and density, and let S be our set of solvers, i.e., ***RNNM*** (exact and inexact), HLWB, and *lsqlin*. Then, we define the performance measure, $t_{p,s} > 0$ obtained for each pair $(p, s) \in P \times S$ with respect to the computational time it took for solver S to solve problem P. Then, for each problem $p \in P$ and solver $s \in S$, we define the performance ratio as

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

Clearly, the solver s that performs the best on problem p will have a performance ratio of 1, and any solvers that perform worse than s will satisfy $t_{p,s} > 1$, i.e., the larger the performance ratio, the worse the solver performed on problem p.

The performance profile of a solver s is then defined as

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

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

5.2.1 Numerical Comparisons

Note that we tested with optimal solutions at nondegenerate, degenerate vertices and non vertices. They exhibited similar results. Therefore, we present results restricted to nondegenerate vertices. We begin with choosing v for (2.1) such that the optimum is uniquely a nondegenerate vertex of P. In the tables below we vary size of 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 5.1 represents a problem with parameters $m = 500$, $n = 2000$, and a density of 0.0081, and each solver will solve 5 randomly generated problems of the form discussed in (2.1), 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) is 2000 for all solvers.

From Tables 5.1 to 5.3, the empirical evidence demonstrates the superiority of the ***RNNM*** (exact) approach to the other solvers. Since the ***RNNM***'s reduced KKT system is $m \times m$ and solved using the Cholesky Decomposition, it's performance should be affected most noticeabley as m varies or density increases. This theoretical observation can be see in Tables 5.1 to 5.3, 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 5.1 the empirical evidence shows similar results to the tables, but better demonstrates the differences in performance between ***RNNM*** (exact) and the other solvers. The problems in Figure 5.1a are similar to that of Table 5.1 except m varies by 100 from 100 to 2000. Similarly, the problems in Figure 5.1b has n varying by 100 from 3000 to 5000, and Figure 5.1c has density varying by 1% from 1% to 100%. In every performance profile, the RMMN (exact) Algorithm clearly outperforms the other solvers, with RMMN (inexact) performing well for an inexact method on mid-sized problems. As should be expected, HLWB is relatively slow on these problems, this can be attributed to it's linear convergence rate, as 2000 sweeps 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 the other solvers at different tolerances.

Table 5.1: Varying problem sizes m; comparing computation time and relative residuals

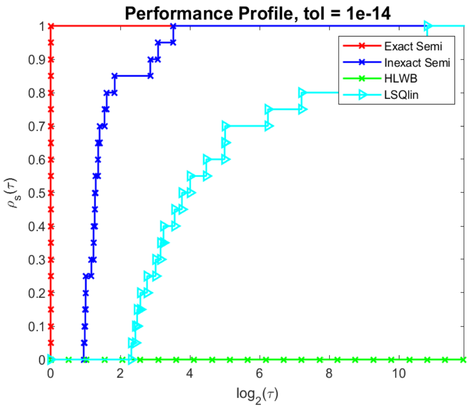
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
100	3000	8.1e-01	1.13e-02	2.71e-02	2.07e+01	4.89e+00	1.11e-16	1.30e-15	2.47e-04	1.07e-15
600	3000	8.1e-01	8.49e-02	2.48e-01	2.28e+02	6.42e+00	2.46e-17	2.90e-16	2.26e-04	1.25e-15
1100	3000	8.1e-01	6.89e-01	1.36e+00	4.83e+02	9.40e+00	8.44e-16	1.12e-15	2.11e-04	7.95e-16
1600	3000	8.1e-01	1.80e+00	4.65e+00	7.79e+02	1.23e+01	7.53e-18	3.66e-16	2.29e-04	5.59e-16

Table 5.2: Varying problem sizes n; comparing computation time and relative residuals

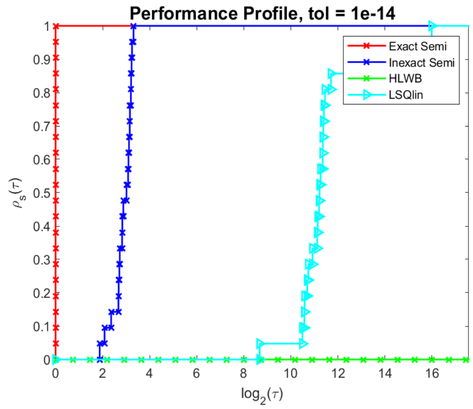
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
200	3000	8.1e-01	1.02e-02	6.36e-02	5.25e+01	5.35e+00	5.08e-16	2.32e-18	2.59e-04	1.81e-15
200	3500	8.1e-01	4.18e-03	3.74e-02	6.10e+01	7.39e+00	9.30e-16	6.08e-17	2.69e-04	2.25e-15
200	4000	8.1e-01	3.68e-03	3.53e-02	6.97e+01	1.07e+01	1.64e-16	2.64e-16	2.85e-04	1.21e-15
200	4500	8.1e-01	6.08e-03	3.92e-02	7.84e+01	1.47e+01	7.17e-16	1.19e-17	3.22e-04	1.83e-15
200	5000	8.1e-01	5.11e-03	3.67e-02	8.66e+01	1.89e+01	5.87e-18	1.43e-16	3.03e-04	2.60e-15

Table 5.3: Varying problem density; comparing computation time and relative residual

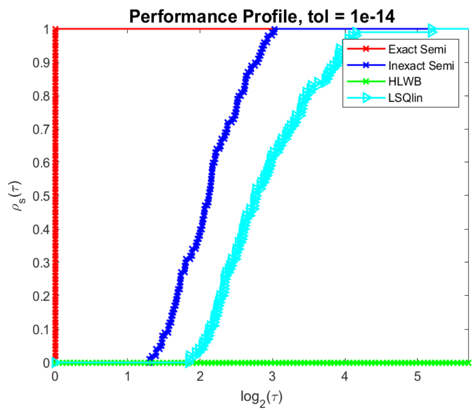
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
300	1000	1.0e+00	1.43e-02	6.69e-02	1.83e+01	5.21e-01	2.45e-15	9.21e-16	1.51e-04	1.25e-15
300	1000	2.6e+01	4.51e-02	2.57e-01	5.18e+01	4.69e-01	6.26e-16	1.45e-17	1.55e-04	3.98e-16
300	1000	5.1e+01	6.77e-02	3.00e-01	6.19e+01	4.51e-01	1.65e-16	1.56e-17	1.58e-04	1.70e-16
300	1000	7.6e+01	9.55e-02	3.15e-01	6.26e+01	5.06e-01	4.03e-17	3.27e-16	1.66e-04	8.81e-17
300	1000	9.6e+01	1.08e-01	3.33e-01	5.64e+01	4.63e-01	1.35e-16	1.48e-15	1.56e-04	1.14e-17



(a) Varying m



(b) Varying n



(c) Varying density

Figure 5.1: Performance Profiles for problems with varying m, n, and densities for nondegenerate vertex solutions

5.3 Solving Large Sparse Linear Programs

We now apply (4.3) along with Theorem 4.3 to solve large-scale LPs. We note that we use the estimate for a starting R given in (4.5). The stepping stones are found using R_n in (4.10). We add a decreasing small scalar to R_n to ensure that we do not stay at the same set of columns of A. For simplicity for these early experiments, we restrict ourselves to nondegenerate LPs.

We compare with the MATLAB *linprog* code, using both the dual simplex and the interior-point algorithm. We use randomly generated problems scaled so that $\|A\| = 1, x_0 > 0, \|x_0\| = 1, b = Ax$. A data point in Table 5.4 is the arithmetic mean of 5 randomly generated problems of the specified parameters. We exclude lines[8] where a failure occurred. The smallest stopping tolerance *linprog* will allow is $\varepsilon = 10^{-10}$, so the performance profile in Figure 5.2 has been adjusted accordingly. The maximum number of iterations for *linprog* is the default number. The relative residual shown Table 5.4 is the sum of relative primal feasibility, dual feasibility, and complementary slackness. In other words, let (x^*, y^*, z^*) be the optimal solution that the stepping stone algorithm or *linprog* return, then the relative residual as shown in the table is

[8]This only happened for the interior point code.

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

From Table 5.4, the empirical evidence demonstrates the stepping stone approach performs better than MATLAB's dual simplex and interior point method on most problems. This becomes more evident as the size of the problems grow and the problems become sparser, i.e., we see that our code fully exploits sparsity in LP. For example, notice that in rows 5-9, the interior point method failed to converge to a solution in the default maximum number of iterations.

In Section 5.2.1, the performance profiles were constructed by looking at smaller intervals of varying m, n and density. For example Table 5.1 shows results where m varies by 500, but in Figure 5.1a m varies by 100. Since the interior point method struggled with obtaining reasonable primal feasibility Table 5.4, Figure 5.2 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 from 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.

Specifications			Time (s)			Rel. Resids.		
m	n	% density	Semismooth	Dual Simplex	Int. Point	Semismooth	Dual Simplex	Int. Point
2e+03	5e+03	1.0e-01	8.84e-02	6.76e-02	4.97e-02	3.38e-17	2.63e-16	4.88e-09
2e+03	1e+04	1.0e-01	9.54e-02	4.92e-02	7.58e-02	2.82e-17	6.00e-16	1.60e-04
2e+03	1e+05	1.0e-01	1.65e-01	3.92e-01	7.45e-01	1.48e-17	7.45e-17	1.72e-05
5e+03	1e+04	1.0e-01	9.68e+01	2.07e-01	1.38e+01	5.55e-17	4.16e-16	5.02e-07
5e+03	1e+05	1.0e-01	7.69e+01	7.27e-01	1.41e+02	2.36e-17	9.31e-11	6.38e-05
5e+03	5e+05	1.0e-01	2.31e+02	7.05e+00	-	1.52e-17	1.87e-10	-
2e+04	1e+05	1.0e-02	5.90e-01	9.51e-01	-	1.36e-17	3.55e-06	-
2e+04	5e+05	1.0e-02	6.58e-01	4.48e+00	-	8.48e-18	3.37e-06	-
2e+04	1e+06	1.0e-02	1.51e+00	9.39e+00	-	7.08e-18	4.34e-06	-
1e+05	1e+07	1.0e-03	5.55e+00	1.06e+01	6.10e+00	1.39e-18	1.39e-18	1.39e-18

Table 5.4: LP application results averaged on 5 randomly generated problems per row

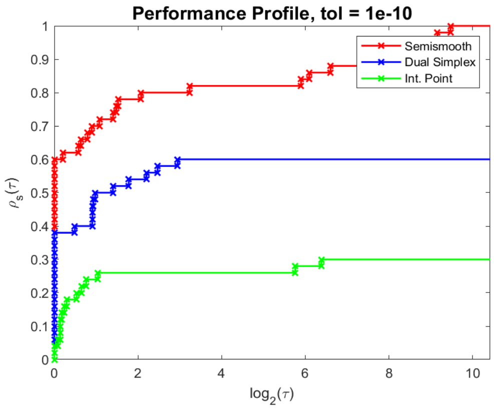


Figure 5.2: Performance Profiles for LP application wrt all problems

6 Conclusion

In this paper we consider the theory and applications of the projection onto a polyhedral set. We studied an elegant optimality condition, derived using the Moreau decomposition, that allowed for a, possibly both nonsmooth and singular, Newton type method. However, this needed a perturbation of a max-rank choice of a generalized Jacobian, i.e., application of nonsmooth analysis and regularization. The regularization guaranteed a decent direction but the method was not necessarily monotonic decreasing. We presented extensive comparisons with the HLWB approach, e.g., [4] and found that we far outperformed HLWB in both speed and accuracy.

We presented several applications including solving large, sparse, linear programs. These early tests were very efficient and outperformed the MATLAB *linprog* code we used for comparison again in both speed and accuracy. The approach can be considered as a *stepping stone external path following* as we follow an external path with parameter R in the objective function; but we only consider a discrete number of points on the path that are found using sensitivity analysis. In general, very few stepping stones are needed, often just <u>one</u>.

A Pseudocodes for Generalized Simplex

The pseudocodes described in Algorithms A.1 to A.3 solves (2.1) using the exact and inexact nonsmooth Newton methods, respectively.

Algorithm A.1 Best Approx. of v for constraints $Ax = b, x \geq 0$; exact Newton direction

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

```
Output. Primal-dual opt.: x_{k+1}, (y_{k+1}, z_{k+1})
Initialization. k ← 0, x_0 ← (v + A^T y_0)_+, z_0 ← (x_0 − (v + A^T y_0))_+,
                F_0 = A x_0 − b, stopcrit ← ‖F_0‖ / (1 + ‖b‖)
while ((stopcrit > ε) & (k ≤ maxiter)) do
    V_k = Σ_{i∈I_+} A_{:i} A_{:i}^T
    λ = min(1e^{−3}, stopcrit)
    V̄ = (V_k + λ I_m)
    solve pos. def. system V̄ d = −F_k for Newton direction d
    updates
    y_{k+1} ← y_k + d
    x_{k+1} ← (v + A^T y_{k+1})_+
    z_{k+1} ← (x_{k+1} − (v + A^T y_k))_+
    F_{k+1} ← A x_{k+1} − b  (residual)
    stopcrit ← ‖F_{k+1}‖ / (1 + ‖b‖)
    k ← k + 1
end while
```

Algorithm A.2 Best Approx. of v for constraints $Ax = b, x \geq 0$, Inexact Newton Direction

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

```
Output. Primal-dual: x_{k+1}, (y_{k+1}, z_{k+1})
Initialization. k ← 0, x_0 ← (v + A^T y_0)_+, z_0 ← (x_0 − (v + A^T y_0))_+,
                δ ∈ (0, 1], ν ∈ [1 + δ/2, 2], and a sequence θ such that θ_k ≥ 0 and sup_{k∈N} θ_k < 1
                F_0 = A x_0 − b, stopcrit ← ‖F_0‖ / (1 + ‖b‖)
while ((stopcrit > ε) & (k ≤ maxiter)) do
    V_k = Σ_{i∈I_+} A_{:i} A_{:i}^T
    λ = (stopcrit)^δ
    V̄ = (V_k + λ I_m)
    solve V̄ d = −F_k for Newton direction d such that residual ‖r_k‖ ≤ θ_k ‖F_k‖^ν
    updates
    y_{k+1} ← y_k + d
    x_{k+1} ← (v + A^T y_{k+1})_+
    z_{k+1} ← (x_{k+1} − (v + A^T y_k))_+
    F_{k+1} ← A x_{k+1} − b  (residual)
    stopcrit ← ‖F_{k+1}‖ / (1 + ‖b‖)
    k ← k + 1
end while
```

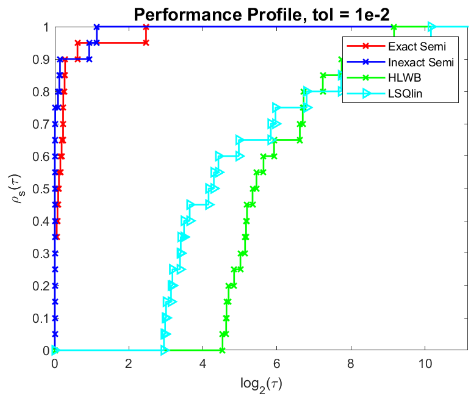
Algorithm A.3 Extended HLWB algorithm

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

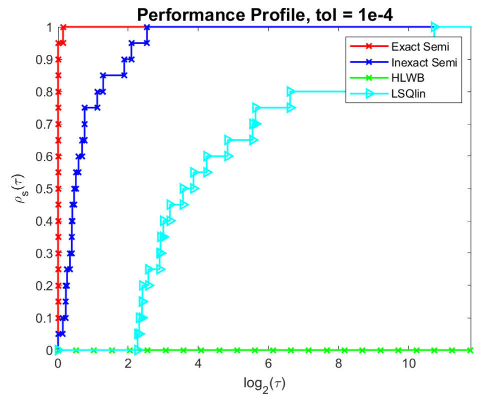
Output. x_{k+1}
Initialization. $k \leftarrow 0$, $msweeps \leftarrow 0$ $x_0 \leftarrow max(v, 0)$, $y_0 \leftarrow x_0$, $i_0 = 1$
stopcrit $\leftarrow \|Ay_0 - b\| / (1 + \|b\|)$ $(= \|F_0\| / (1 + \|b\|))$
while ((stopcrit $> \varepsilon$) & ($k \leq$ maxiter)) **do**
  if $1 \leq i(k) \leq m$ **then**
    $y_k = x_k + \frac{b_{i_k} - \langle a_{i_k}, x^k\rangle}{\|a_{i_k}\|^2} a_{i_k}$
  else
    $y_k = \max(0, x_k)$
  end if
  updates
  $\sigma_k = \frac{1}{k+1}$ (change to $\sigma_k = \frac{1}{msweeps+1}$??)
  $x^{k+1} \leftarrow \sigma_k v + (1 - \sigma_k) y^k$
  stopcrit $\leftarrow \|Ay_k - b\| / (1 + \|b\|)$
  $k \leftarrow k + 1$
  if $k \mod (m + 1) == 0$ **then**
    $msweeps = msweeps + 1$
  end if
  $i_k = k (\text{mod } m) + 1$
end while

B Additional Performance Profiles

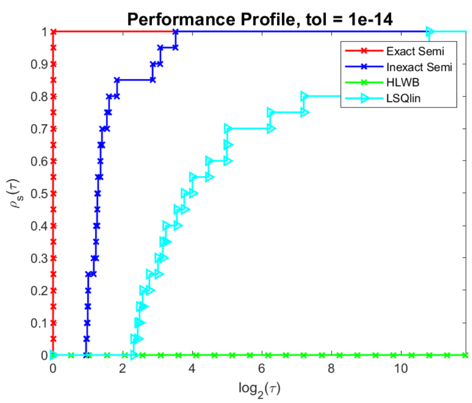
B.1 Nondegenerate



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

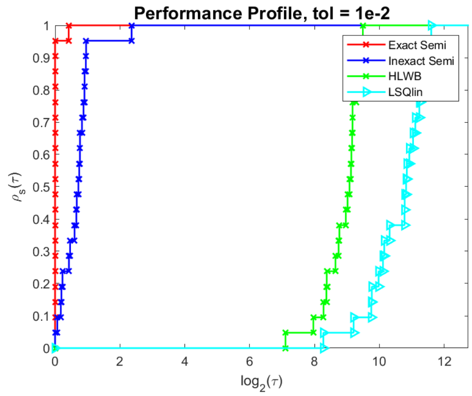


(b) tol $= 10^{-4}$

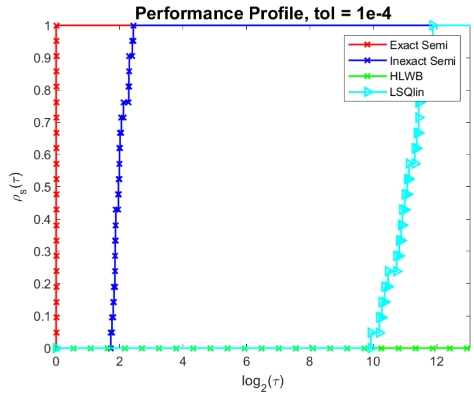


(c) tol $= 10^{-14}$

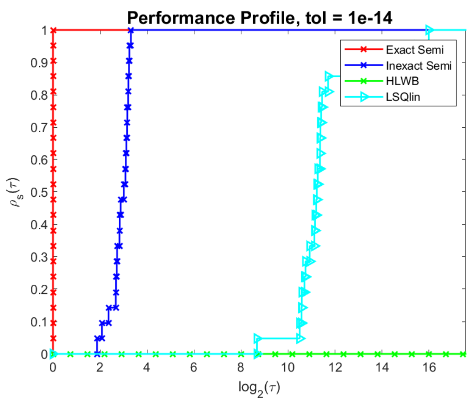
Figure B.1: Performance Profiles for varying m for nondegenerate vertex solutions



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

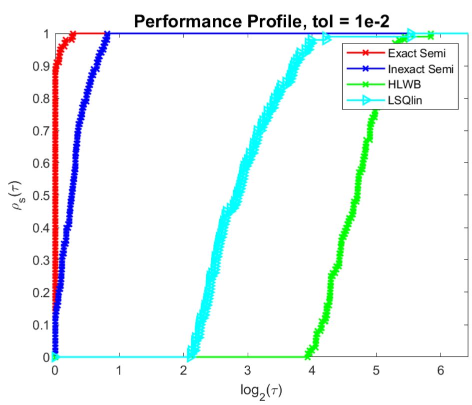


(b) tol $= 10^{-4}$

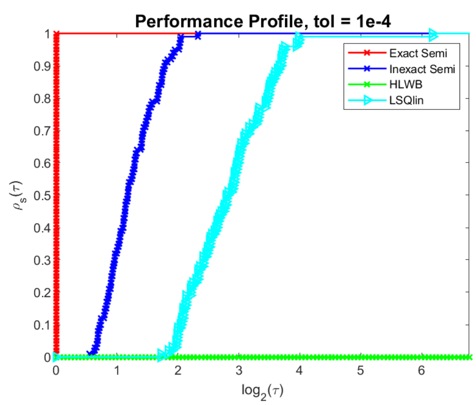


(c) tol $= 10^{-14}$

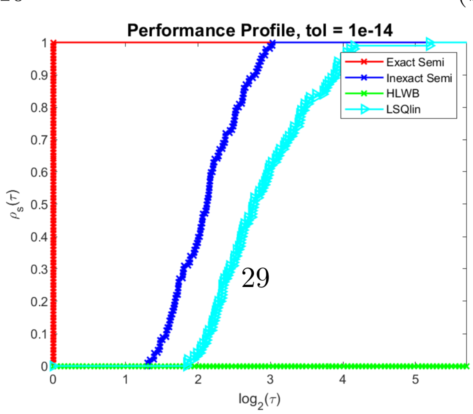
Figure B.2: Performance Profiles for varying n for nondegenerate vertex solutions



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



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



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

B.2 Degenerate

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

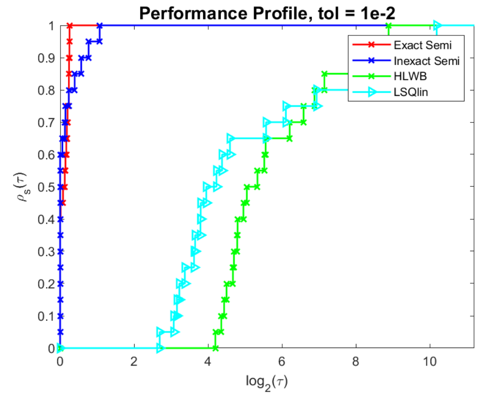
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
100	3000	8.1e-01	1.85e-02	2.83e-02	2.10e+01	5.33e+00	8.29e-16	1.14e-17	2.53e-04	8.96e-16
600	3000	8.1e-01	8.61e-02	3.40e-01	2.30e+02	6.19e+00	1.79e-15	4.96e-17	2.17e-04	1.17e-15
1100	3000	8.1e-01	1.10e+00	2.28e+00	4.87e+02	1.05e+01	1.99e-15	2.45e-15	2.09e-04	3.35e-16
1600	3000	8.1e-01	3.62e+00	1.47e+01	7.75e+02	1.31e+01	3.17e-17	2.61e-15	2.23e-04	2.23e-16

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

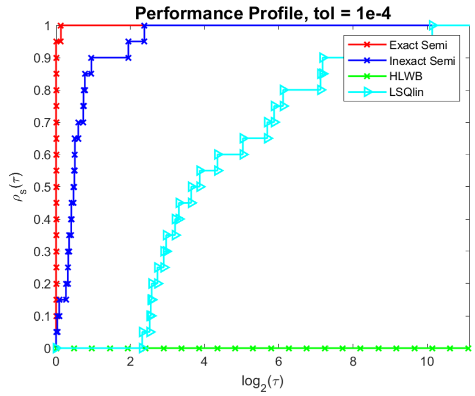
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
200	3000	8.1e-01	1.26e-02	4.62e-02	5.04e+01	5.16e+00	1.94e-17	4.80e-16	2.48e-04	2.35e-15
200	3500	8.1e-01	3.73e-03	3.55e-02	6.04e+01	1.74e+02	4.18e-16	1.94e-16	2.80e-04	5.85e-17
200	4000	8.1e-01	4.57e-03	4.06e-02	6.77e+01	1.22e+01	1.13e-15	7.38e-16	2.89e-04	1.21e-15
200	4500	8.1e-01	7.94e-03	5.06e-02	7.42e+01	1.77e+01	6.39e-17	1.48e-15	3.17e-04	1.44e-16
200	5000	8.1e-01	6.54e-03	4.33e-02	7.91e+01	5.52e+01	5.75e-17	1.45e-15	3.23e-04	2.20e-15

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

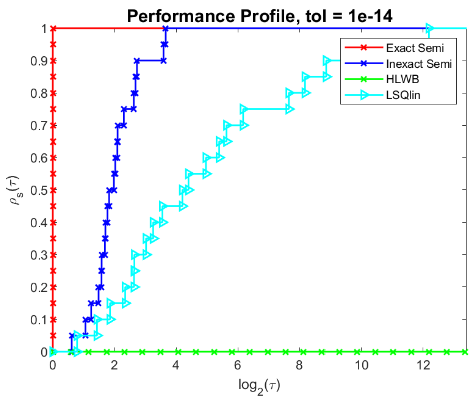
Specifications			Time (s)				Rel. Resids.			
m	n	% density	Exact	Inexact	HLWB	LSQlin	Exact	Inexact	HLWB	LSQlin
300	1000	1.0e+00	1.42e-02	8.28e-02	1.72e+01	5.91e-01	1.89e-16	6.67e-18	1.47e-04	1.35e-16
300	1000	2.6e+01	5.68e-02	4.93e-01	5.17e+01	4.50e-01	2.31e-16	4.05e-17	1.51e-04	6.81e-16
300	1000	5.1e+01	8.82e-02	4.39e-01	6.18e+01	4.71e-01	1.81e-15	1.13e-15	1.45e-04	3.88e-16
300	1000	7.6e+01	1.24e-01	3.96e-01	6.00e+01	5.40e-01	2.13e-15	1.49e-15	1.51e-04	1.47e-16
300	1000	9.6e+01	1.46e-01	4.14e-01	5.49e+01	5.51e-01	4.43e-17	1.32e-15	1.58e-04	3.55e-17



(a) tol = 10^{-2}

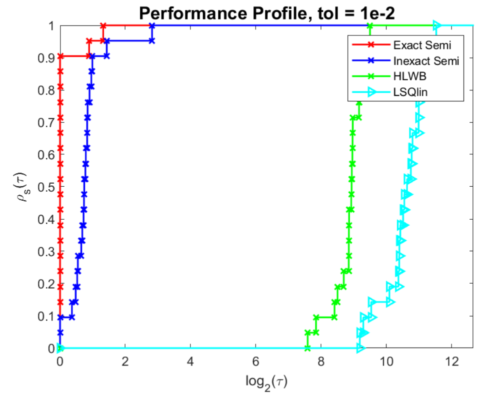


(b) tol = 10^{-4}

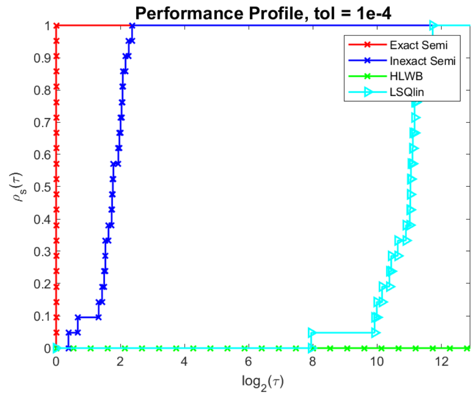


(c) tol = 10^{-14}

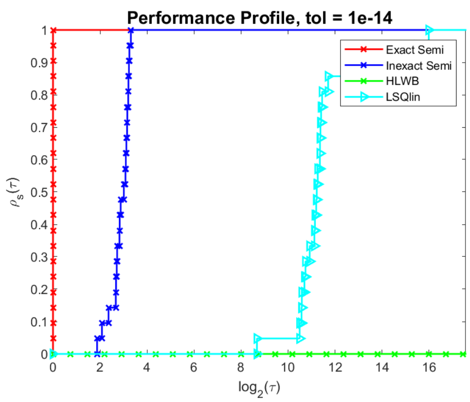
Figure B.4: Performance Profiles for varying m for degenerate vertex solutions



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

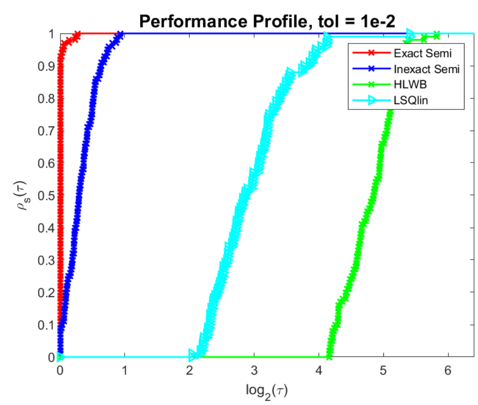


(b) tol $= 10^{-4}$

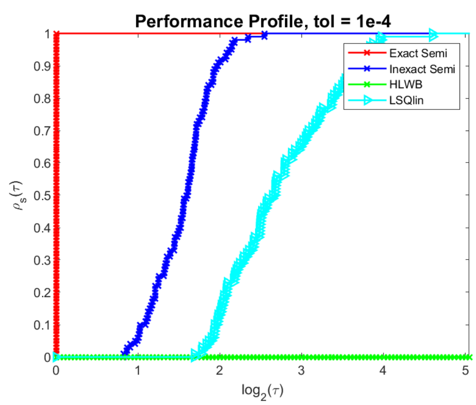


(c) tol $= 10^{-14}$

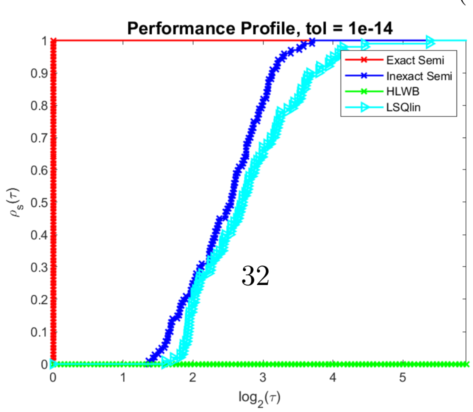
Figure B.5: Performance Profiles for varying n for degenerate vertex solutions



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



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



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

C Applications of the BAP and the HLWB algorithm

The BAP and the HLWB algorithm play important roles in mathematical and technological problems. We give two examples.

Example C.1 (Finding best approximation pairs for two intersections of closed convex sets)**.** *The problem of finding a best approximation pair of two sets, which in turn generalizes the well-known convex feasibility problem [5], has a long history that dates back to work by Cheney and Goldstein in 1959 [16]. This problem was recently revisited in [1] where an alternating HLWB (A-HLWB) algorithm was proposed and studied that can be used when the two sets are finite intersections of half-spaces. Motivated by that [7] presented alternative algorithms that utilize projection and proximity operators. Their modeling framework is able to accommodate even convex sets and their numerical experiments indicate that these methods are competitive and in some cases superior to the A-HLWB algorithm. The practical importance of the problem of finding a best approximation pair of two sets stems from its relevance to real-world situations wherein the feasibility-seeking modeling is used and there are two disjoint constraints sets. One set represents "hard" constraints, i.e., constraints the must be met, while the other set represents "soft" constraints which should be observed as much as possible, see, e.g., [20]. Under such circumstances, the desire to find a point in the hard constraints set that will be closest to the set of soft constraints leads to the problem of finding a best approximation pair of the two sets.*

Least intensity modulated treatment plan in radiotherapy. *The intensity-modulated radiation therapy (IMRT) treatment planning problem in its fully-discretized modeling is represented by a system of linear inequalities as in* (3.2) *with nonnegativity constraints. The unknown vector x represents radiation intensities and if it is a solution of the linear feasibility problem then it fulfills all the planning prescriptions dictated by the oncologist. In such a feasibility-seeking approach several solutions are acceptable but a solution that is closest to the origin will use the least possible intensities that still fulfill the constraints. delivering an acceptable treatment plan with less radiation intensities is preferable and so one replaces the feasibility-seeking problem by a BAP of approximating the origin by a point from the feasible sets, i.e., by seeking the projection of the origin onto the feasible set. Such an approach was used, e.g., in [46] where a simultaneous version of Hildreth's sequential algorithm for norm minimization over linear inequalities, [28, 31], [15, Algorithm 6.5.2] was combined with a norm-minimizing image reconstruction algorithm of Herman and Lent [27], called ART4 (Algebraic Reconstruction Technique 4), which handles in a special effective manner interval inequalities.*

Index

Data Availability and Conflict of Interest Statement

The codes for generating both the data and the output is available at the paper link at URL www.math.uwaterloo.ca/~hwolkowi/henry/reports/ABSTRACTS.html or by request from one of the authors.

The authors declare no competing interests.

References

[1] R. Aharoni, Y. Censor, and Z. Jiang. Finding a best approximation pair of points for two polyhedra. *Comput. Optim. Appl.*, 71(2):509–523, 2018. 33

[2] S. Al-Homidan and H. Wolkowicz. Approximate and exact completion problems for Euclidean distance matrices using semidefinite programming. *Linear Algebra Appl.*, 406:109–141, 2005. 4

[3] L.E. Andersson and T. Elfving. Best constrained approximation in Hilbert space and interpolation by cubic splines subject to obstacles. *SIAM J. Sci. Comput.*, 16(5):1209–1232, 1995. 4

[4] 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. 11, 19, 25

[5] H.H. Bauschke and J.M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Rev.*, 38(3):367–426, 1996. 33

[6] 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. 11, 19

[7] H.H. Bauschke, S. Singh, and X. Wang. Finding best approximation pairs for two intersections of closed convex sets. *Comput. Optim. Appl.*, 81(1):289–308, 2022. 33

[8] V. Beiranvand, W. Hare, and Y. Lucet. Best practices for comparing optimization algorithms. *Optim. Eng.*, 18(4):815–848, 2017. 20, 21

[9] J.M. Borwein and A.S. Lewis. Partially finite convex programming, part I, duality theory. *Math. Program.*, 57:15–48, 1992. 4

[10] J.M. Borwein and A.S. Lewis. Partially finite convex programming, part II, explicit lattice models. *Math. Program.*, 57:49–84, 1992. 4

[11] J.M. Borwein and H. Wolkowicz. A simple constraint qualification in infinite-dimensional programming. *Math. Programming*, 35(1):83–96, 1986. 4

[12] 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. 11

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

[14] 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. 11

[15] Y. Censor and S.A. Zenios. *Parallel optimization*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 1997. Theory, algorithms, and applications, With a foreword by George B. Dantzig. 33

[16] W. Cheney and A.A. Goldstein. Proximity maps for convex sets. *Proc. Amer. Math. Soc.*, 10:448–450, 1959. 33

[17] C.K. Chui, F. Deutsch, and J.D. Ward. Constrained best approximation in Hilbert space. *Constr. Approx.*, 6(1):35–64, 1990. 4

[18] C.K. Chui, F. Deutsch, and J.D. Ward. Constrained best approximation in Hilbert space. II. *J. Approx. Theory*, 71(2):213–238, 1992. 4

[19] F.H. Clarke. *Optimization and Nonsmooth Analysis*. Canadian Math. Soc. Series of Monographs and Advanced Texts. John Wiley & Sons, 1983. 7

[20] P.L. Combettes and P. Bondon. Hard-constrained inconsistent signal feasibility problems. *IEEE Transactions on Signal Processing*, 47:2460–2468, 1999. 33

[21] 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. 11

[22] E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91(2, Ser. A):201–213, 2002. 20

[23] F. Facchinei and J.-S. Pang. *Finite-dimensional variational inequalities and complementarity problems*, volume 1. Springer, 2003. 4, 10

[24] H. Federer. *Geometric measure theory*. Die Grundlehren der mathematischen Wissenschaften, Band 153. Springer-Verlag New York Inc., New York, 1969. 7

[25] M. Goh and F. Meng. On the semismoothness of projection mappings and maximum eigenvalue functions. *J. Global Optim.*, 35(4):653–673, 2006. 4

[26] Y. Haugazeau. *Sur les Inéquations Variationnelles et la Minimisation de Fonctionnelles Convexes*. PhD thesis, University de Paris, 1968. 11

[27] G.T. Herman and A. Lent. A family of iterative quadratic optimization algorithms for pairs of inequalties, with application in diagnostic radiology. *Math. Programming Stud.*, (9):15–29, 1978. Mathematical programming in use. 33

[28] C. Hildreth. A quadratic programming procedure. *Naval Res. Logist. Quart.*, 4:79–85, 1957. 11, 33

[29] J.-B. Hiriart-Urruty. Unsolved Problems: At What Points is the Projection Mapping Differentiable? *Amer. Math. Monthly*, 89(7):456–458, 1982. 4

[30] H. Hu, J. Im, X. Li, and H. Wolkowicz. A semismooth Newton-type method for the nearest doubly stochastic matrix problem. Technical report, 2021. 4, 9

[31] A. Lent and Y. Censor. Extensions of Hildreth's row-action method for quadratic programming. *SIAM J. Control Optim.*, 18(4):444–454, 1980. 11, 33

[32] C. Li and X.Q. Jin. Nonlinearly constrained best approximation in Hilbert spaces: the strong chip and the basic constraint qualification. *SIAM J. Optim.*, 13(1):228–239, 2002. 4

[33] 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. 11

[34] O.L. Mangasarian. Iterative solution of linear programs. *SIAM J. Numer. Anal.*, 18(4):606–614, 1981. 12

[35] O.L. Mangasarian. Normal solutions of linear programs. Number 22, pages 206–216. 1984. Mathematical programming at Oberwolfach, II (Oberwolfach, 1983). 12

[36] O.L. Mangasarian. A Newton method for linear programming. *J. Optim. Theory Appl.*, 121(1):1–18, 2004. 12

[37] C.A. Micchelli, P.W. Smith, J. Swetits, and J.D. Ward. Constrained l_p approximation. *Journal of Constructive Approximation*, 1:93–102, 1985. 4

[38] R. Mifflin. Semismooth and semi-convex functions in constrained optimization. *SIAM J. Cont. Optim.*, 15:959–972, 1977. 4

[39] H. Qi and D. Sun. A quadratically convergent Newton method for computing the nearest correlation matrix. *SIAM J. Matrix Anal. Appl.*, 28(2):360–385, 2006. 4

[40] L. Qi and J. Sun. A nonsmooth version of Newton's method. *Mathematical programming*, 58(1-3):353–367, 1993. 4

[41] H. Rademacher. Uber partielle und totale differenzierbarkeit i. *Math. Ann.*, 89:340–359, 1919. 7

[42] 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. 4

[43] I. Singer. *Best approximation in normed linear spaces by elements of linear subspaces*. Die Grundlehren der mathematischen Wissenschaften, Band 171. Publishing House of the Academy of the Socialist Republic of Romania, Bucharest; Springer-Verlag, New York-Berlin, 1970. Translated from the Romanian by Radu Georgescu. 4

[44] P.W. Smith and H. Wolkowicz. A nonlinear equation for linear programming. *Math. Programming*, 34(2):235–238, 1986. 4, 12

[45] X. Xiao, Y. Li, Z. Wen, and L. Zhang. A regularized semi-smooth Newton method with projection steps for composite convex programs. *J. Sci. Comput.*, 76(1):364–389, 2018. 4

[46] 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. 33