The Duality between the Perceptron Algorithm and the von Neumann Algorithm

Dan Li
Lehigh University

Tamás Terlaky
Lehigh University

Report: 12T-011

The Duality between the Perceptron Algorithm and the von Neumann Algorithm

Dan Li* and Tamás Terlaky†

November 1, 2012

Abstract The perceptron and the von Neumann algorithms were developed to solve Linear Feasibility Problems. In this paper, we investigate and reveal the duality relationship between these two algorithms. The specific forms of Linear Feasibility Problems solved by the perceptron and the von Neumann algorithms are a pair of alternative systems by the Farkas Lemma. A solution of one problem serves as an infeasibility certificate of its alternative system. Further, we adapt an Approximate Farkas Lemma to interpret the meaning of an approximate solution from its alternative perspective. The Approximate Farkas Lemma also enables us to derive bounds for the distance to feasibility or infeasibility from approximate solutions of the alternative systems. Based on these observations, we interpret variants of the perceptron algorithm as variants of the von Neumann algorithm, and vice-versa; as well as transit the complexity results from one family to the other.

Keywords Linear feasibility problem · Perceptron algorithm · von Neumann algorithm · Duality · Approximate Farkas Lemma

Mathematics Subject Classification 49N15 · 90C05 · 90C60 · 68Q25

1 Introduction

Linear Optimization (LO) is the problem of minimizing or maximizing a linear objective function subject to a system of linear inequalities and equations. The Linear Feasibility Problem (LFP) is to find a feasible solution to a linear inequality system. Considering an LO problem with zero as its objective function, then every feasible solution is optimal. From this point of view, the LFP is a special case of LO. On the other hand, it is well known [2,16] that by the LO duality theorem, any LO problem can be transformed to an equivalent LFP. LFPs can be written in various forms. In this paper, we consider two of them. The first one is

$$A^T y > 0, \tag{1}$$

where matrix $A \in \mathbb{R}^{m\times n}$ with its column vectors $a_1, a_2, \ldots, a_n \in \mathbb{R}^m$ and $y \in \mathbb{R}^m$. We assume that $\|a_i\|_2 = 1$ for all $i = 1, 2, \ldots, n$. This assumption is not changing the status of feasibility of problem (1). The second form we consider is

$$Ax = 0, \quad e^T x = 1, \quad x \geq 0, \tag{2}$$

where $x \in \mathbb{R}^n$, $e \in \mathbb{R}^n$ is the vector of all one. Without loss of generality [1], we can assume that matrix A has the same properties as in problem (1). Observe that problem (2) is a standard form of the LFP with a convexity constraint [7]. Let conv(A) represent the convex hull of the points a_i. If the origin $0 \in$ conv(A), then problem (2) is feasible and can be considered as a weighted center problem [7], i.e., the problem of assigning nonnegative weights x_i to the points a_i so that their weighted center of gravity is the origin 0.

Several algorithms are proposed for solving LFPs [2,16,17], such as simplex methods [2,17], ellipsoid [12,13] and interior point [16] methods, and variants of the perceptron [3,4,18] and the von Neumann [5,7] algorithms. All of these algorithms aim to find a feasible solution to an LO, or equivalently to a LFP. They either deliver a feasible solution, or provide an evidence of infeasibility. Ellipsoid and interior

*Dept. of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA. E-mail: dal207@lehigh.edu

†Dept. of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA. E-mail: terlaky@lehigh.edu

Research supported by a Start-up package of Lehigh University, bethlehem, PA.

point methods are polynomial time algorithms while the perceptron and the von Neumann algorithms are not polynomial time [2, 12, 16, 17].

In this paper, we focus on the perceptron algorithm and the von Neumann algorithms. These two algorithms solve LFPs in the forms of (1) and (2), respectively. Therefore, we also call problem (1) the perceptron problem and problem (2) the von Neumann problem. By the Farkas Lemma, problems (1) and (2) are alternative systems to each other; and consequently, the perceptron and the von Neumann algorithms can be interpreted in a close duality relationship. This duality relationship allows us to transit one algorithm as a variant of the other, as well as we can transit their complexity results. For instance, the perceptron algorithm could be applied to an infeasible instance of problem (2), because its Farkas alternative system is a feasible LFP in the form of (1). The discovery of this duality not only provides a profound insight into both of the algorithms, but also results in new variants of the algorithms. In particular, the duality relationship of the two algorithms can be used to detect unexpected infeasibility caused by data error.

There is one major difference between these two algorithms. When problem (1) is feasible, the perceptron algorithm obtains an exact feasible solution – which satisfies all the constraints – with finite number of iterations. However, the von Neumann algorithm is a infinite algorithm. In finite number of iterations, the solution returned is approximately feasible – the solution does not necessary strctly satisfy the equality constraint $Ax = 0$. In order to identify an exact solution to problem (2), Dantzig [6] proposed a "bracketing" procedure: apply the von Neumann algorithm $m + 1$ times to finding approximate solutions to $m + 1$ perturbations of problem (2) and then combine these approximate solutions to get an exact solution to the original unperturbed problem. However in practice, this method cannot detect infeasibility, and it fails if problem (2) is close to the boundary of feasibility and infeasibility [6].

When we obtain an approximate solution without any knowledge of the feasibility status of the problem, we would like to explore the meaning of this approximate solution from the dual side and provide some quantified information about the feasibility of the dual problem. Todd and Ye [21] derived Approximate Farkas Lemmas from the gauge duality results of Freund [10], that are extensions of the Farkas Lemma [2,16,17]. We adapt an Approximate Farkas Lemma to problems (1) and (2), The Lemma serves as a basis of obtaining more meaningful information for the output of the algorithms. An ϵ-solution of problem (2) implies that the distance of the corresponding problem (1) to infeasibility is no more than ϵ.

This paper is structured as follows. In Section 2, we review the perceptron and von Neumann algorithms, and their complexity results. Then we analyze the duality relationship between these two algorithms and discuss the use of Approximate Farkas Lemma in Section 3. In Section 4 and 5, we interpret variants of the perceptron algorithm as variants of the von Neumann algorithm and the other way around. The complexity results are also transferred from one family to the other. Finally, in Section 6 we state our conclusions and suggest avenues for future work.

2 Preliminaries

For convenience, we first introduce the following notations.

The vector $x(y)$ – Let Δ_n be the unit simplex, i.e., $\Delta_n = \{x : x \geq 0, \|x\|_1 = 1\}$. For $y \in \mathbb{R}^m$, define $x(y) = \text{argmin } \{y^T Ax \,|\, x \in \Delta_n\}$. Thus, we have $Ax(y) = a_s$, or equivalently $x(y) = e_s$ if and only if $a_s^T y = \min \{a_i^T y \,|\, i = 1, \cdots, n\}$. Observe that a_s and so $x(y)$ are not necessarily unique.

The quantity ρ – A common feature of the perceptron and the von Neumann algorithms is that their iteration complexity depends on a quantity ρ [3, 9], that quantifies how far the given instance is from the boundary of infeasibility and feasibility. Given an LFP, we define the quantity ρ in general terms as follows.

Definition 1. *Given an LFP, if the problem is feasible, ρ is defined as the radius of the largest inscribed ball contained in the feasible region. Otherwise, ρ measures the distance to feasibility.*

If the problem is feasible (infeasible), the quantity ρ indicates how far the constraints can be shifted or rotated so that the problem becomes infeasible (feasible). The larger ρ is, the harder to turn a feasible (infeasible) problem to infeasible(feasible). Therefore, ρ can be seen as a measurement of the robustness of feasibility or infeasibility. For different algorithms and problem forms, ρ has slightly different definitions. We will discuss them in details in Section 3.2. In order to distinguish ρ in the different algorithms, we use

$\rho_p(A)$ to represent ρ in the perceptron algorithm and $\rho_v(A)$ for the von Neumann algorithm. In Section 2.1 and 2.2, we discuss that the quantity ρ plays an essential role in the theoretical complexity of the perceptron and von Neumann algorithms.

2.1 The Perceptron Algorithm

The Classical Perceptron Algorithm, used in machine learning [18], is designed to solve data classification problems: Given a set of points with each point labeled as either positive or negative. The problem is to find a separating hyperplane, which separates all positive points from the negative ones. By some transformations [3,18] those problems lead to LFPs in the form of (1). The Classical Perceptron Algorithm assumes that problem (1) is strictly feasible, it starts from the origin, and calls the classical perceptron update at each iteration. The classical perceptron update finds a violated constraint and move the current iterate y by one unit perpendicularly towards the violated constraint. The algorithm is described as follows.

Algorithm 1. ***The Classical Perceptron Algorithm***

```
Initialization Let $y^0$ be the all-zero vector. $k = 0$.
While True Do
Let $a_s = Ax(y^k)$.
If $a_s^T y^k > 0$, STOP and return $y^k$.
  Else
      $y^{k+1} = y^k + a_s$,
      $k = k + 1$.
End While
```

The perceptron algorithm has the following complexity result.

Theorem 2.1. *[15] Assume that the LFP (1) is strictly feasible. Then in at most*

$$\left\lceil \frac{1}{\rho_p(A)^2} \right\rceil$$

iterations, the Perceptron Algorithm terminates with a feasible solution.

2.2 The von Neumann Algorithm

The von Neumann algorithm, published by Dantzig [5,7] in 1991, solves LFPs in the form of (2). Before presenting the von Neumann algorithm, we need to define what an ϵ-*approximate solution* (or ϵ-solution for short reference) of problem (2) is.

Definition 2. *An $x \in \Delta_n$ is called an ϵ-solution of problem (2) if $\|Ax\| \leq \epsilon$.*

Algorithm 2. ***The von Neumann Algorithm***

```
Initialization
Choose any $x \in \Delta_n$.
Let $b^0 = Ax^0$ and $k = 0$.
While $\|b^k\| \geq \epsilon$ Do
  1. Find $a_s$ which makes the largest angle with the vector $b^k$, i.e., $a_s = Ax(b^k)$.
     Let $\nu_k = a_s^T b^k$.
  2. If $\nu_k > 0$, STOP, problem (2) is infeasible.
```

3. Let e_s be the unit vector corresponding to index s. Let

$$\begin{aligned}\lambda &= \frac{1-\nu_k}{\|b^k\|^2 - 2\nu_k + 1},\\ x^{k+1} &= \lambda x^k + (1-\lambda)e_s,\\ b^{k+1} &= Ax^{k+1},\\ k &= k+1.\end{aligned}$$

End While

In Algorithm 2, the vector b is also called the residual at the current iterate. Dantzig assumed that problem (2) is feasible and derived an upper bound for the computational complexity [7] as follows.

Theorem 2.2. *[7] Let $\epsilon > 0$ and assume that problem (2) is feasible. Then the von Neumann Algorithm provides an ϵ-solution for problem (2) in at most*

$$\left\lceil \frac{1}{\epsilon^2} \right\rceil$$

iterations.

Epelman and Freund [9] gave a different complexity analysis, and showed that when problem (2) is strictly feasible or strictly infeasible, then the iteration complexity of the von Neumann algorithm is linear in $\log(1/\epsilon)$ and $1/\rho_v(A)^2$.

Theorem 2.3. *[9] Suppose that $\rho_v(A) > 0$ and let $\epsilon > 0$.*

1) If problem (2) is strictly feasible, then in at most

$$\left\lceil \frac{2}{\rho_v(A)^2} \ln \frac{1}{\epsilon} \right\rceil$$

iterations, the von Neumann algorithm obtains an ϵ-solution of problem (2).

2) If problem (2) is strictly infeasible, then in at most

$$\left\lfloor \frac{1}{\rho_v(A)^2} \right\rfloor$$

iterations a certificate of infeasibility is given by the von Neumann algorithm.

When problem (2) is feasible, both the complexity bounds proved by Dantzig (Theorem 2.2) and Epelman and Freund (Theorem 2.3) are for obtaining an ϵ-approximate solution. Neither of them is dominant. When $\rho_v(A)$ is large, the complexity bound proved by Epelman and Freund is better. Otherwise, the one by Dantzig is better.

Theoretically the von Neumann Algorithm does not provide an exact solution, it only converges to a solution. Dantizig [6] proposed a “bracketing” procedure to identify an exact solution in finite number of iterations. By applying the von Neumann Algorithm to $m+1$ perturbed problems, $m+1$ approximate solutions are generated. A weighted sum of these approximate solutions yields an exact solution to the original unperturbed problem. This requires the solution of an $(m+1)\times(m+1)$ system of linear equations. This procedure has the following complexity.

Theorem 2.4. *[6] Suppose that problem (2) is strictly feasible. By applying the Dantzig's “bracketing” procedure, an exact feasible solution is found in*

$$\frac{4(m+1)^3}{\rho_v(A)^2}$$

iterations.

3 Duality

In this section, we first employ the Farkas Lemma to analyze the duality relationship between problem (1) and problem (2). This observation provides the foundation for the duality between the perceptron and the von Neumann algorithms. Then we extend the definition of the quantity ρ to infeasible problems and give ρ meaningful explanations for different problems. At last, we propose to utilize an Approximate Farkas Lemma to interpret an approximate solution from its dual perspective.

3.1 Alternative Systems

Recall that $\text{conv}(A)$ represents the convex hull of the points a_i. Assume that problem (1) is feasible and y is a feasible solution. In this case, the hyperplane with normal vector y separates $\text{conv}(A)$ from the origin, what implies that problem (2) is infeasible. Conversely, if problem (2) is infeasible, then there exists at least one separating hyperplane that can separate $\text{conv}(A)$ from the origin. In other words, there exists a vector y such that $A^T y > 0$, which means problem (1) is feasible. Therefore, problem (1) and problem (2) are a pair of alternative systems. This conclusion can also be verified by the Farkas Lemma [2,16,17]. According to the Farkas Lemma, the alternative system of problem (2) is

$$\begin{array}{rcl} A^T y + e\eta & \geq & 0 \\ \eta & < & 0. \end{array} \tag{3}$$

Problem (3) can equivalently be written as $A^T y > 0$, which is the form of problem (1). Thus, problems (1) and (2) are alternative systems to each other, i.e., exactly one of them is solvable. Since the perceptron and the von Neumann algorithms solve problems (1) and (2) respectively, the duality relationship between these two problems leads to a duality between the two algorithms.

3.2 The Quantity ρ

Section 2 gives a general definition of the quantity ρ. In this section, we discuss its special forms for the different problem forms in the different algorithms. The Classical Perceptron Algorithm shown in Section 2.1 assumes that problem (1) is strictly feasible. Thus, ρ is only defined for feasible problems in [8]. In order to make the discussions about the duality complete, we extend the definition of ρ to infeasible cases.

Definition 3. *Consider the LFP (1).*

1. *If problem (1) is feasible [8], then the quantity $\rho_p(A)$ is the radius of the largest inscribed ball that fits in the feasible region, and the center of the ball is on the unit sphere. It is calculated by*

$$\rho_p(A) = \max_{\|y\|=1} \min_i \{a_i^T y\}. \tag{4}$$

2. *If problem (1) is infeasible, then $\rho_p(A)$ is the distance to feasibility, i.e.,*

$$\rho_p(A) = \min_{\|y\|=1} \max_i \{-a_i^T y\}. \tag{5}$$

Note that when problem (1) is feasible, $\rho_p(A) > 0$ if and only if it is strictly feasible. On the other hand, the specific ρ for problem (2) in the von Neumann algorithm is defined as follows.

Definition 4. *[9] The quantity ρ is the distance from the origin to the boundary $\partial(\text{conv}(A))$ of the feasible set $\text{conv}(A)$, i.e.,*

$$\rho_v(A) = \inf\{\|h\| : h \in \partial(\text{conv}(A))\}. \tag{6}$$

Definition 4 also defines quantity $\rho_v(A)$ with two different meanings depending on the feasibility or infeasibility of problem (2).

1. If problem (2) is feasible, then $\rho_v(A)$ is the radius of the largest inscribed ball in $\text{conv}(A)$ centered at the origin. It can be calculated [14] by

$$\rho_v(A) = \min_{\|y\|=1} \max_i \{-a_i^T y\}. \tag{7}$$

When problem (2) is feasible but not strictly feasible, i.e., the origin is on the boundary of conv(A), then $\rho_v(A) = 0$.

2. If problem (2) is infeasible, then $\rho_v(A)$ is the distance from the origin to conv(A), i.e., $\rho_v(A)$ is the radius of the largest separating ball centered at the origin. It can be calculated as

$$\rho_v(A) = \max_{\|y\|=1} \min_i \{a_i^T y\}. \tag{8}$$

Comparing (4), (5), (7), and (8), it is easy to see that when problem (2) is infeasible (feasible), the quantity ρ is computed in the same way as the one when problem (1) is feasible (infeasible). This observation originates from the alternative systems relationship of these two problems.

3.3 Interpreting Approximate Solutions

Instead of providing an exact feasible solution as the perceptron algorithm does, the von Neumann algorithm returns an ϵ-solution when it terminates in a finite number of iterations. Analogously, the Modified Perceptron Algorithm [4] – a variant of the perceptron algorithm – returns a σ-feasible solution when the perceptron problem (1) is close to the boundary of feasibility and infeasibility. A σ-feasible solution is also an approximate solution which we will discuss later. When ϵ or σ is a fixed number, an ϵ-solution or a σ-feasible solution is not sufficient to draw a firm conclusion about feasibility of the problem. In these two sections, our goal is to give some interpretations of these two approximate solutions from their alternative perspective and answer the following questions.

- How to derive meaningful information from these approximate solutions?
- What conclusion can be drawn about the feasibility status of the problems?

The duality relationship discussed in Section 3.1 is directly derived from the Farkas Lemma. The two problems (1) and (2) are a pair of alternative systems and therefore, exactly one of them is solvable. Recall that both the Classical Perceptron and the von Neumann algorithms are non-polynomial algorithms. When the problems are close to the boundary of feasibility and infeasibility, both the classical perceptron and von Neumann algorithms are inefficient. It takes exponentially many iterations for the algorithms to obtain a clear answer about solvability of the problems. Therefore, we would like that some variants of the algorithms could provide an approximate solution, or some indications of approximate feasibility or infeasibility. Due to the alternative system relationship between problems (1) and (2), a proof of infeasibility for one problem can be given by a solution to the other one. We are interested in exploring approximate solutions to this pair of alternative systems and their interpretations for their alternative systems. We first discuss σ-feasible solutions. They are defined as follows [4].

Definition 5. *A vector y is a σ-feasible solution (or σ-solution for short reference) to problem (1) if $A^T\bar{y} \geq -\sigma e$, where σ is a small positive number.*

According to the Definition 5, a σ-solution allows slight violations to the constraints in problem (1); and thus it is an approximate solution. Recall that we analyze the meaning of the quantity ρ in Section 3.2. From Definition 4, we obtain the following theorem.

Theorem 3.1. *The following two statements are equivalent.*

(a) The perceptron problem (1) has a σ-solution.

(b) There is no ball in conv(A) *centered at the origin with radius larger than σ.*

This theorem can be derived directly from the definition of the largest inscribed ball in conv(A) (7). Theorem 3.1 shows that a σ-feasible solution to the perceptron problem (1) indicates that the corresponding von Neumann problem (2) is either infeasible, or if it is feasible then it is close to infeasibility. As a result, we define a σ-solution as a σ-infeasibility certificate for the von Neumann problem (2).

Definition 6. *A vector y is a σ-infeasibility certificate for the von Neumann problem (2) if $A^T\bar{y} \geq -\sigma e$.*

By combining Theorem 3.1 and Definition 6, the following corollary can be derived.

Corollary 1. *A σ-infeasibility certificate indicates that the von Neumann problem (2) is either infeasible or feasible with $\rho_v(A) \leq \sigma$.*

3.4 Near Infeasibility and the Approximate Farkas Lemma

In the previous section, we interpret a σ-feasible solution to the preceptron problem (1) as a σ-infeasibility certificate of the von Neumann problem (2). In this section, we explore whether we can obtain an analogous result about an ϵ-solution. Our major tool is the Approximate Farkas Lemma [21].

Approximate Farkas lemmas are derived by Todd and Ye [21] from the general gauge duality results of Freund [10]. These lemmas are extensions of the Farkas Lemma [17] and quantify how certain approximate feasible solutions to a system of inequalities indicate infeasibility, or almost infeasibility of the alternative system of inequalities. In order to adapt the Approximate Farkas Lemma, we first transfer a strictly feasible problems (1) to an optimization problem. Consider the following optimization problem

$$\begin{array}{rll} \alpha_{\tilde{y}} = & \min & \|\tilde{y}\| \\ & \text{s.t.} & A^T\tilde{y} \;\; \geq \;\; e, \end{array} \tag{9}$$

where $\tilde{y} \in \mathbb{R}^m$, matrix A has the same definition as in problem (1), and $\alpha_{\tilde{y}}$ denote the optimal value. This optimization problem is to find a feasible solution to inequality system $A^T\tilde{y} \geq e$ with the shortest length. Comparing problem (9) and problem (1), any feasible solution to problem (9) is also a feasible solution to problem (1). On the other hand, if y^* is a strictly feasible solution to problem (1), i.e., all coordinates of $A^Ty^* > 0$, then $\tilde{y}^* = \frac{y^*}{(y^*)^TAx(y^*)}$ is a feasible solution to problem (9), where $(y^*)^TAx(y^*)$ gives the value of the smallest coordinate of A^Ty^*. Thus, feasibility of problem (9) is equivalent to strict feasibility of problem (1). When problem (1) is strictly feasible, we can put a ball into the feasible region of $A^Ty > 0$. If the radius of the ball is fixed to 1, then $\alpha_{\tilde{y}}$ measures the minimal distance from the center of this unit ball to the origin. Recall that $\rho_p(A)$ measures the radius of the largest ball in the feasible region with its center on the unit sphere. The closer problem (1) is to infeasibility, the narrower the feasible region is, the smaller $\rho_p(A)$ is, and the further the unit ball is from the origin, i.e., the larger $\alpha_{\tilde{y}}$ is. Thus, $\alpha_{\tilde{y}}$ can be seen as another measure of the robustness of problem (1). The following result highlights the geometrical relationship between $\alpha_{\tilde{y}}$ and $\rho_p(A)$:

$$\alpha_{\tilde{y}} = \frac{1}{\rho_p(A)}. \tag{10}$$

By adapting the Approximate Farkas Lemma [21] to our problem, we obtain:

Lemma 1. *(Approximate Farkas Lemma) Consider the optimization problems*

$$(\text{GP}): \;\; \alpha_{\tilde{y}} = \min\left\{\|\tilde{y}\| \mid A^T\tilde{y} \geq e\right\}, \qquad \text{and}$$

$$(\text{GD}): \;\; \beta_b = \min\left\{\|b\| \mid Ax = b, e^Tx = 1, x \geq 0\right\}.$$

Then $\alpha_{\tilde{y}}\beta_b = 1$.

The special case $0 \cdot (+\infty)$ is interpreted as 1. It is easy to see that problem (GD) is a perturbed variant of problem (2). When problem (1) is strictly feasible, then β_b gives the minimal distance between the origin and conv(A), which is the radius $\rho_v(A)$ of the largest ball defined by (8) that measures the infeasibility of problem (2). In other words, β_b indicates the minimum correction needed to make problem (2) feasible. By Lemma 1 and (10), we have $\beta_b = \frac{1}{\alpha_{\tilde{y}}} = \rho_p(A)$. Therefore, the Approximate Farkas Lemma verifies that $\rho_p(A)$ for a feasible perceptron problem (1) is the same as $\rho_v(A)$ for the corresponding infeasible von Neumann problem (2). Thus, any feasible solution to problem (GP) is an infeasibility certificate of problem (2), and thus it gives a lower bound for the distance to feasibility.

On the other hand, if problem (2) is feasible, then any of its feasible solutions is an optimal solution to the optimization problem (GD) with $\beta_b = 0$. Then, according to the Approximate Farkas Lemma we have $\alpha_{\tilde{y}} = +\infty$. This implies that problem (GP) is infeasible, and so Lemma 1 reduces to the original Farkas Lemma. In this case, problem (1) is either infeasible, or feasible but not strictly feasible.

We have the following theorem which utilizes the Approximate Farkas Lemma in giving interpretation to approximate solutions.

Theorem 3.2. *The following three statements are equivalent.*

(a) The von Neumann problem (2) has an ϵ-solution.

(b) A unit ball cannot be put closer than $1/\epsilon$ to the origin in the feasible region of problem (1).

(c) There is no ball in the feasible region of problem (1) centered on the unit sphere with radius larger than ϵ.

Proof. Problem (2) has an ϵ-solution x' such that $\beta'_b = \|b'\| = \|Ax'\| \leq \epsilon$, if and only if $\beta_b \leq \beta'_b \leq \epsilon$. By Lemma 1, this holds if and only if $\frac{1}{\epsilon} \leq \frac{1}{\beta_b} = \alpha_{\tilde{y}}$. The inequality $\frac{1}{\epsilon} \leq \alpha_{\tilde{y}}$ holds if and only if

$$\nexists \tilde{y} \text{ such that } \|\tilde{y}\| < \frac{1}{\epsilon} \text{ and } A^T\tilde{y} \geq e. \tag{11}$$

By scaling, (11) is equivalent to

$$\nexists y \text{ such that } \|y\| \leq 1 \text{ and } A^Ty > \epsilon e. \tag{12}$$

Statement (11) is the same as statement (b). By the definition of $\rho_p(A)$, statement (12) indicates that $\rho_p(A) \leq \epsilon$. Thus, statements (a), (b), and (c) are equivalent. $\square$

Theorem 3.2 shows that an ϵ-solution to the von Neumann problem (2) implies that the corresponding perceptron problem (1) is either infeasible, or if it is feasible then it is close to infeasibility. Therefore, we define such an ϵ-solution as an ϵ-infeasibility certificate for problem (1).

Definition 7. *A vector y is an ϵ-infeasibility certificate for the perceptron problem (1) if there exist a vector $x \in \Delta_n$ such that $Ax = y$ and $\|y\| \leq \epsilon$.*

We can derive the following corollary from Theorem 3.2.

Corollary 2. *An ϵ-infeasibility certificate indicates that the perceptron problem (1) is either infeasible, or feasible with $\rho_p(A) \leq \epsilon$.*

Since ϵ is a small positive number, it means that the norm of any feasible solution $\tilde{y}'$ of problem (GP), if it exists, is at least as large as $\frac{1}{\epsilon}$. Thus, if problem (GP) is close to infeasibility, any feasible solutions –if it exists– has to be large. For example, if problem (GP) is feasible and the distance to infeasibility is as small as 10^{-10}, then the magnitude of any feasible solution $\tilde{y}'$ has to be at least as large as 10^{10}.

In this section we utilized the definition of the quantity ρ and the Approximate Farkas Lemma to interpret approximate solutions so that we can draw more definitive conclusions about the solutions or feasibility of the corresponding problems. In addition, when the respective variants of the perceptron and the von Neumann algorithms terminate at a certain point, then the Approximate Farkas Lemma allows a more precise interpretation of the output and provides meaningful information about the solution.

Inspired by the alternative system relationship of problems (1) and (2), we investigate the duality of the perceptron and the von Neumann algorithms. In Section 4, different versions of the perceptron algorithm are interpreted as variants of the von Neumann algorithm as they are applied to problem (2). In Section 5, we interpret variants of the von Neumann algorithm from the perspective of the perceptron algorithm. By exploring this intriguing duality of these algorithms, we not only gain new insight into the intimate relationship of these algorithms, but also derive several novel variants of these algorithms with their corresponding complexity results.

4 From Perceptron to von Neumann

Since problem (1) and (2) are alternative systems to each other, the perceptron algorithm can be operated on problem (2) with proper adjustments. The complexity results for the feasible case of the perceptron algorithm are adaptable for the infeasible case of the von Neumann algorithm. Since the perceptron algorithm has several variants, we discuss them in the following subsections.

In order to make our discussions more transparent, we rename the two spaces. The perceptron algorithm solves problems in form (1) to get a solution y if the problem is feasible. Thus, we call the space $\mathbb{R}^m$ in which the vector y lives the *perceptron space*. Similarly, because the von Neumann algorithm solves problem (2), we call $\mathbb{R}^n$ the *von Neumann space*. Note that the vector $b^k = Ax^k$ in the von Neumann algorithm, presented as Algorithm 2, is in the perceptron space. This reflects the duality of the two problems and also indicates some close relationships between the two algorithms. Matrix A can be seen as a linear operator between the perceptron and the von Neumann spaces.

4.1 The Normalized Perceptron Algorithm

We first revisit, Algorithm 1, the Classical Perceptron Algorithm. It starts from $y^0 = 0$ and at iteration k, y^k makes one unit step in the direction of the most violated constraint. Let x^k be the corresponding vector that satisfies $y^k = Ax^k$. We have $x^0 = 0$ and $x^{k+1} = x^k + e_s$, where $e_s = x(y)$. According to this relationship, x^k is a sequence of vectors in the von Neumann space with $x^k \geq 0$ and $\|x^k\|_1 = k$ for all $k \in \mathbb{N}$. On the other hand, observe that the last two constraints in problem (2), $e^T x = 1, x \geq 0$ restrict vector x to be in the unit simplex Δ_n. The comparison of x^k at iteration k in the Classical Perceptron Algorithm and x in problem (2) leads to a normalized version of the perceptron algorithm [20]. Assume that problem (1) is feasible. The Normalized Perceptron Algorithm is as follows.

Algorithm 3. ***The Normalized Perceptron Algorithm***

```
Initialization Let $y^0 = 0$ and $k = 0$.
While True Do
Let $a_s = Ax(y^k)$.
If $a_s^T y^k > 0$, STOP and return $y^k$.
  Else
    $\theta_k = \frac{1}{k+1}$,
    $y^{k+1} = (1-\theta_k)y^k + \theta_k a_s$,
    $k = k+1$.
If $\|y^k\| \leq \epsilon$, STOP and return $y^k$ as an $\epsilon$-infeasibility certificate.
End While
```

Note that at the end of each iteration, the iterate y^k is inspected. The algorithm terminates if y^k is an ϵ-infeasibility certificate. This stopping criterion is derived from the von Neumann side after we successfully explain an approximate solution. In the Normalized Perceptron Algorithm, the k-th iterate y^k is divided by k to satisfy $y^k = Ax^k$ for some $x^k \in \Delta_n$. Thus, x^k is a vector x in the von Neumann space, as well as y^k can be interpreted as the corresponding b^k vector in the von Neumann algorithm. If the Normalized Perceptron Algorithm starts from $x^0 = 0$ and x^k can be updated to satisfy $x^k \in \Delta_n$ and $y^k = Ax^k$, then we get a variant of the von Neumann algorithm. To ease understanding, the derived Normalized von Neumann Algorithm is described in full details as follows.

Algorithm 4. ***The Normalized von Neumann Algorithm***

```
Initialization Let $x^0 = 0$, $b^0 = Ax^0$, and $k = 0$.
While $\|b^k\| \geq \epsilon$ Do
Let $a_s = Ax(b^k)$.
If $a_s^T b^k > 0$, STOP, return $b^k$ as an infeasibility certificate of problem (2).
  Else
    $\theta_k = \frac{1}{k+1}$,
    $x^{k+1} = (1-\theta_k)x^k + \theta_k x(b^k)$,
    $b^{k+1} = Ax^{k+1}$,
    $k = k+1$.
End While
Return $x^k$ as an $\epsilon$-solution.
```

When applying to problem (2), the Normalized von Neumann Algorithm has the following complexity result.

Theorem 4.1. *Let $\epsilon > 0$.*

1) If problem (2) is feasible, then the Normalized von Neumann Algorithm provides an ϵ-solution in at most

$$\left\lceil \frac{1}{\epsilon^2} \right\rceil$$

iterations.

2) If problem (2) is strictly infeasible, then in at most

$$\left\lceil \frac{1}{\rho_v(A)^2} \right\rceil$$

iterations an infeasibility certificate is given.

Proof. When problem (2) is strictly infeasible, then its alternative problem (1) is strictly feasible. Since y^k, generated by the Normalized Perceptron Algorithm, is exactly the same as y^k in the Classical Perceptron Algorithm divided by k, the complexity result of the Classical Perceptron Algorithm stated in Theorem 2.1 is also valid for Algorithm 3, the Normalized Perceptron Algorithm. Thus, the complexity result for strictly infeasible problems is an immediate corollary of Theorem 2.1.

Now we prove the complexity when problem (2) is feasible. By using induction on k, we prove that $\|b^k\| \leq \frac{1}{\sqrt{k}}$. For $k = 1$, since the algorithm starts with $b^0 = 0$,

$$\|b^1\| = \|(1-\theta_0)b^0 + \theta_0 Ax(b^0)\| = \|Ax(b^0)\| = 1,$$

where the last equality results from $\|a_i\| = 1$ for $i = 1, \dots, n$.

Now, suppose that we have $\|b^{k-1}\| \leq \frac{1}{\sqrt{k-1}}$. At the iteration k, we obtain

$$\begin{aligned}
\|b^k\|^2 &= \|(1-\theta_{k-1})b^{k-1} + \theta_{k-1}a_s\|^2 \\
&= (1-\theta_{k-1})^2\|b^{k-1}\|^2 + \theta_{k-1}^2\|a_s\|^2 + 2\theta_{k-1}(1-\theta_{k-1})(a_s^T b^{k-1}) \\
&\leq \frac{1}{k^2}\left[(k-1)^2\|b^{k-1}\|^2 + 1\right] \leq \frac{1}{k}
\end{aligned}$$

The first inequality must be true because $a_s^T b^{k-1} \leq 0$ when problem (2) is feasible. The second inequality holds due to the inductive hypothesis $\|b^k\| \leq 1/\sqrt{k}$. Thus, to obtain an ϵ-solution, after k iterations, it is sufficient to have

$$\epsilon = \|b^k\| \leq 1/\sqrt{k}.$$

Therefore, the algorithm needs at most $\lceil 1/\epsilon^2 \rceil$ iterations. □

Recall that Theorem 2.1 provides the complexity result for feasible perceptron problems. If problem (1) is strictly feasible, then the Classical Perceptron Algorithm returns a feasible solution in at most $1/\rho_p(A)^2$ iterations. However, there is no published result for infeasible perceptron problems. Now by transiting Theorem 4.1 back to the perceptron problems, we obtain the following new result for the Classical Perceptron Algorithm.

Theorem 4.2. *Let $\epsilon > 0$. If problem (1) is infeasible, then the Classical/Normalized Perceptron Algorithm provides an ϵ-infeasibility certificate in at most*

$$\left\lceil \frac{1}{\epsilon^2} \right\rceil$$

iterations, which indicates that there is no such an ϵ-ball in the feasible region.

The complexity bound in Theorem 4.2 only depends on the value ϵ, the accuracy of the infeasibility certificate, but does not depend on the geometry of the problem.

4.2 The Smooth Perceptron Algorithm

Soheili and Peña [20] proposed a smooth version of the perceptron algorithm and showed that it can be seen as a smooth first-order algorithm. This deterministic variant retains the original simplicity of the perceptron algorithm and its complexity is improved by almost a factor of $1/\rho_p(A)$ compared to the Classical Perceptron Algorithm. The improved complexity result is given in Theorem 4.3. We first introduce the Smooth Perceptron Algorithm.

Given $\mu > 0$, $x(y)$ is smoothed by the entropy prox-function

$$x_\mu(y) = \frac{e^{\frac{-A^T y}{\mu}}}{\left\| e^{\frac{-A^T y}{\mu}} \right\|_1}, \tag{13}$$

where the expression $e^{\frac{-A^T y}{\mu}}$ denotes the n-dimensional vector

$$e^{\frac{-A^T y}{\mu}} = \left[e^{\frac{-a_1^T y}{\mu}}, e^{\frac{-a_2^T y}{\mu}}, \dots, e^{\frac{-a_n^T y}{\mu}} \right]^T.$$

The Smooth Perceptron Algorithm is as follows.

Algorithm 5. ***The Smooth Perceptron Algorithm***

```
Initialization Let $y^0 = \frac{Ae}{n}$, $\mu_0 = 1$, and $x^0 = x_{\mu_0}(y^0)$. $k = 0$.
While True Do
Let $a_s = Ax(y^k)$.
If $a_s^T y^k > 0$, STOP and return $y^k$.
  Else
    $\theta_k = \frac{2}{k+3}$,
    $y^{k+1} = (1-\theta_k)(y^k + \theta_k A x^k) + \theta_k^2 A x_{\mu_k}(y^k)$,
    $\mu_{k+1} = (1-\theta_k)\mu_k$,
    $x^{k+1} = (1-\theta_k)x^k + \theta_k x_{\mu_{k+1}}(y^{k+1})$,
    $k = k+1$.
End While
```

Compared to the complexity of the Classical Perceptron Algorithm stated in Theorem 2.1, Theorem 4.3 shows that the Smooth Perceptron Algorithm has a complexity result with $\frac{1}{\rho_p(A)\sqrt{\log(n)}}$ improvement.

Theorem 4.3. *[20] Assume that problem (1) is strictly feasible. The Smooth Perceptron Algorithm terminates in at most*

$$\frac{2\sqrt{\log(n)}}{\rho_p(A)} - 1$$

iterations with a feasible solution.

Analogous to the Normalized Perceptron Algorithm, the Smooth Perceptron Algorithm can also be applied to problem (2) when it is infeasible. Iterate y^k in the perceptron space plays the role of the vector b^k in the von Neumann algorithm. Since b^k is updated so that $Ax^k = b^k$, we derive the corresponding vector x^k.

Algorithm 6. ***The Smooth von Neumann Algorithm***

```
Initialization Let $x^0 = \frac{e}{n}$, $b^0 = Ax^0 = \frac{Ae}{n}$, $\mu_0 = 1$, and $\tilde{x}^0 = x_{\mu_0}(y^0)$. $k = 0$.
While True Do
```

```
Let $a_s = Ax(b^k)$
If $a_s^T b^k > 0$, STOP and return $b^k$ as an infeasibility certificate.
  Else
```

$$\begin{aligned}
\theta_k &= \frac{2}{k+3}, \\
b^{k+1} &= (1-\theta_k)(b^k + \theta_k A\tilde{x}^k) + \theta_k^2 A x_{\mu_k}(b^k), \\
\mu_{k+1} &= (1-\theta_k)\mu_k, \\
\tilde{x}^{k+1} &= (1-\theta_k)\tilde{x}^k + \theta_k x_{\mu_{k+1}}(b^{k+1}), \\
x^{k+1} &= (1-\theta_k)(x^k + \theta_k \tilde{x}^k) + \theta_k^2 x_{\mu_k}(b^k), \\
k &= k+1.
\end{aligned}$$

End While

Compare Algorithm 5 and 6. Iterate y^k in Algorithm 5 is the same as vector b^k in Algorithm 6, and its corresponding x^k satisfying $y^k = Ax^k$ is the vector x in problem (2). It is easy to see that $x^k \in \Delta_n$ for all iterations. Therefore, the complexity result of Theorem 4.3 applies to Algorithm 6 as well when problem (2) is infeasible. We derive the following corollary from Theorem 4.3.

Corollary 3. *Assume that problem (2) is strictly infeasible. The Smooth von Neumann Algorithm terminates in at most*

$$\frac{2\sqrt{\log(n)}}{\rho_v(A)} - 1$$

iterations with a certificate of infeasibility b^k such that $A^T b^k > 0$.

Recall that the Smooth Perceptron Algorithm has an improved complexity result compared to the Normalized Perceptron Algorithm when problem (1) is feasible. Thus, if problem (2) is infeasible, then after interpreted in the von Neumann space, the Smooth von Neumann Algorithm also enjoys an almost $1/\rho_v(A)$ complexity improvement compared to the one presented in Corollary 4.1.

Independently of our work, Soheili and Peña [19] proposed a version of smooth von Neumann algorithm called Iterated Smooth Perceptron-von Neumann (ISPVN) Algorithm. It is also based on the duality relationship between problems (1) and (2). Instead of using the entropy prox-function as the Smooth Perceptron Algorithm, it employs the Euclidean prox-function to smooth $x(y)$. The merit of the ISPVN Algorithm is that when problem (1) is infeasible with $\rho_p(A) > 0$, the ISPVN Algorithm solves its alternative system (2). Thus, the ISPVN Algorithm could handle both problems (1) and (2) simultaneously. It either finds a feasible solution to problem (1) in $O\left(\frac{\sqrt{n}}{\rho_p(A)} \log\left(\frac{1}{\rho_p(A)}\right)\right)$ elementary iterations, or finds an ϵ-solution to the corresponding problem (2) in $O\left(\frac{\sqrt{n}}{\rho_v(A)} \log\left(\frac{1}{\epsilon}\right)\right)$ elementary iterations. Both of the iteration complexity of the ISPVN Algorithm are better than these of the perceptron and the von Neumann algorithms. However, in the case when problem (2) is infeasible, the Smooth von Neumann Algorithm stated in Algorithm 6 has a better complexity bound.

5 From von Neumann to Perceptron

After interpreting variants of the perceptron algorithm from its dual perspective in Section 4, in this section we show how to interpret the von Neumann algorithm as a variant of the perceptron algorithm and how to apply it to problem (1).

5.1 The Original von Neumann Algorithm

The von Neumann algorithm was reviewed in Section 2.2. Note that in the von Neumann algorithm, iterates x^k are always in the unit simplex. The vector $b^k = Ax^k$ in the von Neumann algorithm can play the role of vector y in the perceptron space.

Algorithm 7. ***The von Neumann Algorithm in the perceptron space***

```
Initialization
Choose any $x^0 \in \Delta_n$. Let $y^0 = Ax^0$ and $k = 0$.
While $\|y^k\| \geq \epsilon$ Do
    1. Let $\mu_k = \|y^k\|$ and $\nu_k = (Ax(y^k))^T y^k$, where $x(y^k) = \underset{x\in\Delta_n}{\operatorname{argmin}} \{(y^k)^T Ax\}$.
    2. If $\nu_k > 0$, STOP and return $y^k$ as a solution.
    3. Update
        $\lambda = \frac{1-\nu_k}{\mu_k^2 - 2\nu_k + 1}$,
        $\mu_{k+1}^2 = \lambda\nu_k + (1-\lambda)$,
        $y^{k+1} = \lambda y^k + (1-\lambda)Ax(y^k)$,
        $k = k+1$.
End While
```

According to Theorem 2.3, there are two possible outcomes of the von Neumann algorithm. If problem (2) is strictly infeasible, the von Neumann algorithm provides an infeasibility certificate. Then the alternative case in the perceptron space is that problem (1) is strictly feasible. Thus, applying the von Neumann algorithm to problem (1) will provide a feasible solution y^k. On the other hand, if problem (2) is strictly feasible, then the von Neumann algorithm will return an ϵ-solution with $\|b^k\| < \epsilon$, and $m+1$ applications of the von Neumann Algorithm allows to get an exact solution [6], which is interpreted as an exact infeasibility certificate for problem (1). However, if problem (2) is neither strictly feasible with an ϵ-ball in the feasible set, nor strictly infeasible with at least ϵ distance to feasibility, then an ϵ-solution interpreted in the perceptron space implies an ϵ-infeasibility certificate of problem (1). An ϵ-solution/ϵ-infeasibility certificate could have two possible meanings.

1. If problem (2) is feasible, then problem (1) is infeasible.
2. If problem (2) is infeasible, then an ϵ-solution implies that the distance to the infeasibility of problem (2) is less than ϵ, i.e., $\rho_v(A) < \epsilon$; and the radius of the largest inscribed ball in the feasible region of problem (1) is $\rho_p(A) < \epsilon$. This means that though problem (1) is feasible, it is almost infeasible. The distance to the infeasibility is less then ϵ.

Thus, problem (1) is either infeasible or ϵ-close to infeasibility. From Theorem 2.3 the following complexity result can be derived for Algorithm 7.

Theorem 5.1. *Let $\epsilon > 0$.*

1) If problem (1) is strictly feasible, then in at most

$$\left\lfloor \frac{1}{\rho_p^2} \right\rfloor$$

iterations the von Neumann Algorithm finds a feasible solution to problem (1).

2) If problem (1) is strictly infeasible, then in at most

$$\left\lceil \frac{2}{\rho_p^2} \ln \frac{1}{\epsilon} \right\rceil$$

iterations the von Neumann Algorithm provides an ϵ-infeasibility certificate.

5.2 The Optimal Pair Adjustment Algorithm

Gonçalves et al. [11] introduced three variants of the von Neumann algorithm named as Weight-Reduction, Optimal Pair Adjustment (OPA), and Projection Algorithms. Among these three variants, the OPA Algorithm has the best performance in computational experiments. The basic idea of the OPA Algorithm is to move the residual b^k in Algorithm 2 closer to the origin 0 as much as possible at each update step. It gives the maximum possible freedom to two weights: the one in column a_{s^+} which has the largest angle with b^k and the one in column a_{s^-} which has the smallest angle with b^k. At each iteration, it finds the optimal value for these two coordinates and adjusts the remaining ones. The OPA Algorithm is as follows.

Algorithm 8. ***The Optimal Pair Adjustment Algorithm***

Initialization

Choose any $x^0 \in \Delta_n$. Let $b^0 = Ax^0$.

Given a small positive number ϵ.

While $\|b^k\| \geq \epsilon$ **Do**

1. **Find** the vectors a_{s^+} and a_{s^-} which make the largest and smallest angles with the current iterate y^k:
$$\begin{aligned} s^+ &= \operatorname*{argmin}_{i=1,\ldots,n} \{a_i^T b^k\}, \\ s^- &= \operatorname*{argmin}_{i=1,\ldots,n} \{a_i^T b^k | x_i > 0\}, \\ \nu_k &= a_{s^+}^T b^k. \end{aligned}$$
2. **If** $\nu_k > 0$, STOP, return b^k as a feasible solution to problem (2).
3. **Solve** the subproblem
$$\begin{aligned} \min \quad & \|\bar{b}\|^2 \\ \text{s.t.} \quad & \lambda_0(1 - x_{s^+}^k - x_{s^-}^k) + \lambda_1 + \lambda_2 = 1, \\ & \lambda_i \geq 0, \text{ for } i = 1, 2, \end{aligned}$$
where $\bar{b} = \lambda_0(b^k - x_{s^+}^k a_{s^+} - x_{s^-}^k a_{s^-}) + \lambda_1 a_{s^+} + \lambda_2 a_{s^-}$.
4. **Update**
$$\begin{aligned} b^{k+1} &= \lambda_0(b^k - x_{s^+}^k a_{s^+} - x_{s^-}^k a_{s^-}) + \lambda_1 a_{s^+} + \lambda_2 a_{s^-}, \\ x_i^{k+1} &= \begin{cases} \lambda_0 x_i^k & i \neq s^+, s^-, \\ \lambda_1, & i = s^+, \\ \lambda_2, & i = s^-. \end{cases} \\ k &= k+1. \end{aligned}$$

End While

The OPA Algorithm has a better performance than the original von Neumann algorithm in practice [11], and Gonçalves proves that in theory it is at least as good as the original von Neumann algorithm.

Theorem 5.2. *[11] The decrease in $\|b^k\|$ obtained by an iteration of the OPA Algorithm is at least as large as that obtained by one iteration of the von Neumann Algorithm.*

Therefore, the OPA and the von Neumann Algorithms share the same theoretical complexity as given in Theorem 2.3.

In the dual space, the residual b^k is the normalized iterate y^k in the perceptron algorithm. The column which has the largest angle with b^k corresponds the "most infeasible" constraint for y^k. Since a feasible solution to the von Neumann problem is an infeasibility certificate for the corresponding perceptron problem, the faster the residual b^k moves closer to 0, the sooner infeasibility of the perceptron problem is detected. Therefore, the OPA algorithm outperforms the von Neumann algorithm when problem (1) is infeasible. In this section, we describe the equivalent OPA Perceptron Algorithm.

Algorithm 9. ***The Optimal Pair Adjustment Perceptron Algorithm***

Initialization

Choose any $x^0 \in \Delta_n$. Let $y^0 = Ax^0$ and $u^0 = \|y^0\|$. $\theta = 1$.

Given a small positive number ϵ.

While $u^k \geq \epsilon$ **Do**

1. **Find** the vectors a_{s^+} and a_{s^-} which make the largest and smallest angles with the current iterate y^k:

$$\begin{aligned} s^+ &= \operatorname*{argmin}_{i=1,\ldots,n} \{a_i^T y^k\}, \\ s^- &= \operatorname*{argmin}_{i=1,\ldots,n} \{a_i^T y^k | x_i > 0\}, \\ \nu_k &= a_{s^+}^T y^k. \end{aligned}$$

2. **If** $\nu_k > 0$, STOP. Return y^k as a feasible solution to problem (1).
3. **Solve** the subproblem

$$\begin{aligned} \min \quad & \|\bar{y}\|^2 \\ \text{s.t.} \quad & \lambda_0(1 - x_{s^+}^k - x_{s^-}^k) + \lambda_1 + \lambda_2 = 1, \\ & \lambda_i \geq 0, \ \text{for } i = 1, 2, \end{aligned}$$

where $\bar{y} = \lambda_0(\frac{y^k}{\theta} - x_{s^+}^k a_{s^+} - x_{s^-}^k a_{s^-}) + \lambda_1 a_{s^+} + \lambda_2 a_{s^-}$.

4. **Update**

If $\lambda_0 = 0$, then

$$\begin{aligned} \theta &= 1, \\ y^{k+1} &= \lambda_1 a_{s^+} + \lambda_2 a_{s^-}. \end{aligned}$$

Else,

$$\begin{aligned} \theta &= \frac{\theta}{\lambda_0} \\ y^{k+1} &= y^k + (\theta\lambda_1 - x_{s^+}^k) a_{s^+} + (\theta\lambda_2 - x_{s^-}^k) a_{s^-}. \end{aligned}$$

End If

$$\begin{aligned} u^{k+1} &= \frac{1}{\theta}\|y^{k+1}\|, \\ x_i^{k+1} &= \begin{cases} \lambda_0 x_i^k & i \neq s^+, s^-, \\ \lambda_1, & i = s^+, \\ \lambda_2, & i = s^-. \end{cases} \\ k &= k + 1. \end{aligned}$$

End While

The subproblem in Step 3 can be solved by enumerating all possibilities that satisfy the KKT conditions [11]. Analogous to Algorithm 7, if problem (1) is feasible and Algorithm 9 terminates with $u^k < \epsilon$, then there is no ϵ-ball contained in the feasible cone centered on the unit sphere. In this case, problem (1) is close to infeasibility and y^k is an ϵ-infeasibility certificate. After interpreted as a variant of the perceptron algorithm, the OPA Perceptron Algorithm has the complexity result as stated in Theorem 5.1.

6 Summary and Future Work

The perceptron and the von Neumann algorithms are used to solve LFPs in different forms. In this paper, we reveal the duality relationship between these algorithms. This observation is based on the fact that the forms of the LFPs these algorithms deal with are a pair of Farkas alternative systems. This relationship enables us to interpret variants of the perceptron algorithm as variants of the von Neumann algorithm, and vice versa. The dual interpretation of the algorithms allows us to transit the complexity results to the new algorithms too. The interpretation of an approximate solution is crucial during the algorithms transit. By utilizing the Approximate Farkas Lemma to make the solution meaningful for the alternative system and the transit complete. A major difference of these two algorithm families is that the perceptron algorithm assumes that the problem is feasible while the von Neumann algorithm solves both feasible and infeasible problems. Therefore, in this paper, we show that the infeasibility of perceptron problems are detected by the interpreted von Neumann algorithm (Algorithm 7) and the OPA perceptron algorithm (Algorithm 9); and the Normalized von Neumann Algorithm (Algorithm 4) – interpreted from the Normalized Perceptron Algorithm is applicable for both strictly infeasible and feasible von Neumann problems. Furthermore, when problem (1) is infeasible, we derive a complexity result for the Classical Perceptron Algorithm from the perspective of the von Neumann space.

There is another variant of the perceptron algorithm – the Modified Perceptron Algorithm [4]. It starts from a random vector y. In order to interpret it from the von Neumann perspective, finding the corresponding vector x is a critical step. In addition, the Modified Perceptron Algorithm returns a σ-feasible solution – which is also an approximate solution – when the perceptron problem is feasible with small quantity $\rho_p(A)$ – smaller than a given threshold. Therefore, we are aiming to design a modified version of von Neumann algorithm which can return an infeasibility certificate when the problem is almost infeasible. Its interpreted variant will benefit detecting infeasibility of perceptron problems when $\rho_p(A)$ is small.

References

[1] I. Bárány and S. Onn. Colourful linear programming and its relatives. *Mathematics of Operations Research*, 22(3), 1997.

[2] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization.* Athena Scientific, 1997.

[3] A. Blum and J. Dunagan. Smoothed analysis of the perceptron algorithm for linear programming. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 905–914, 2002.

[4] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1), 1998.

[5] G. B. Dantzig. Converting a converging algorithm into a polynomially bounded algorithm. Technical Report SOL 91-5, Stanford University, 1991.

[6] G. B. Dantzig. Bracketing to speed convergence illustrated on the von Neumann algorithm for finding a feasible solution to a linear program with a convexity constraint. Technical Report SOL 92-6, Stanford University, 1992.

[7] G. B. Dantzig. An ϵ-precise feasible solution to a linear program with a convexity constraint in $1/\epsilon^2$ iterations independent of problem size. Technical Report SOL 92-5, Stanford University, 1992.

[8] J. Dunagan and S. Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. In *Proceedings of STOC'04*, pages 315–320. ACM Press, 2004.

[9] M. A. Epelman and R. M. Freund. Condition number complexity of an elementary algorithm for resolving a conic linear system. Operations Research Center, Massachusetts Institute of Technology. Working paper 319-97, 1997.

[10] R. M. Freund. Dual gauge programs, with applications to quadratic programming and the minimum-norm problem. *Mathematical Programming*, 38:47–67, 1987.

[11] J. P. M. Gonçalves. *A family of linear programming algorithms based on the von Neumann algorithm.* PhD thesis, Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, 2004.

[12] L. G. Khachian. A polynomial algorithm for linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[13] E. Klafszky and T. Terlaky. On the ellipsoid method. *Radovi Mathematicki*, 8:269–280, 1992.

[14] D. Li. On rescaling algorithms for linear optimization, 2011. Ph.D. Proposal, revised. Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, United States.

[15] M. Minsky and S. A. Papert. *Perceptrons: An Introduction To Computational Geometry.* MIT Press, 1969.

[16] C. Roos, T. Terlaky, and J.-P. Vial. *Interior Point Methods for Linear Optimization.* Springer, 2006.

[17] A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, 1998.

[18] J. Shawe-Taylor and N. Cristianini. *Support Vector Machines and Other Kernel-Based Learning Methods.* Cambridge University Press, 2000.

[19] N. Soheili and J. Peña. A primal-dual smooth perceptron-von Neumann algorithm. Working paper, Carnegie Mellon University, 2012.

[20] N. Soheili and J. Peña. A smooth perceptron algorithm. *SIAM Journal on Optimization*, 22(2):728–737, 2012.

[21] M. J. Todd and Y. Ye. Approximate Farkas Lemmas and stopping rules for iterative infeasible-point algorithms for linear programming. *Mathematical Programming*, 81:1–21, 1998.