A Deterministic Rescaled Perceptron Algorithm

Javier Peña* Negar Soheili †

June 25, 2013

Abstract

The perceptron algorithm is a simple iterative procedure for finding a point in a convex cone F. At each iteration, the algorithm only involves a query to a separation oracle for F and a simple update on a trial solution. The perceptron algorithm is guaranteed to find a point in F after $\mathcal{O}(1/\tau_F^2)$ iterations, where τ_F is the width of the cone F.

We propose a version of the perceptron algorithm that includes a periodic rescaling of the ambient space. In contrast to the classical version, our rescaled version finds a point in F in $\mathcal{O}(m^5 \log(1/\tau_F))$ perceptron updates. This result is inspired by and strengthens the previous work on randomized rescaling of the perceptron algorithm by Dunagan and Vempala [*Math. Program.* 114 (2006), 101–114] and by Belloni, Freund, and Vempala [*Math. Oper. Res.* 34 (2009), 621–641]. In particular, our algorithm and its complexity analysis are simpler and shorter. Furthermore, our algorithm does not require randomization or deep separation oracles.

1 Introduction

The relaxation method, introduced in the classical articles of Agmon [1], and Motzkin and Schoenberg [16], is a conceptual algorithmic scheme for solving the feasibility problem

$$y \in F. \tag{1}$$

Here $F \subseteq \mathbb{R}^m$ is assumed to be an open convex set with an available *separation oracle:* Given a test point $y \in \mathbb{R}^m$, the oracle either certifies that $y \in F$ or else it finds a hyperplane separating y from F, that is, $u \in \mathbb{R}^m, b \in \mathbb{R}$ such that $\langle u, y\rangle \leq b$ and $\langle u, v\rangle > b$ for all $v \in F$. The relaxation method starts with an arbitrary initial trial solution. At each iteration, the algorithm queries the separation oracle for F at the current trial solution y. If $y \in F$ then the algorithm terminates. Otherwise, the algorithm generates a new trial point $y_+ = y + \eta u$ for some step length $\eta > 0$ where $u \in \mathbb{R}^m, b \in \mathbb{R}$ determine a hyperplane separating y from F as above.

*Tepper School of Business, Carnegie Mellon University, USA, jfp@andrew.cmu.edu

†Tepper School of Business, Carnegie Mellon University, USA, nsoheili@andrew.cmu.edu

The perceptron algorithm can be seen as a particular type of relaxation method for the problem (1). It applies to the case when F is the interior of a convex cone. It usually starts at the origin as the initial trial solution and each update is of the form $y_+ = y + \frac{1}{\|u\|}u$. The perceptron algorithm was originally proposed by Rosenblatt [19] for the polyhedral feasibility problem $A^{\mathrm{T}}y > 0$. As it was noted by Belloni, Freund, and Vempala [6], the algorithm readily extends to the more general problem (1) when F is the interior of a convex cone, as described above. Furthermore, Belloni et al. [6, Lemma 3.1] showed that the classical perceptron iteration bound of Block [8] and Novikoff [17] also holds in general: The perceptron algorithm finds a solution to (1) in at most $\mathcal{O}(1/\tau_F^2)$ perceptron updates, where τ_F is the width of the cone F :

$$\tau_F := \sup_{\|y\|=1} \{r \in \mathbb{R}_+ : \mathbb{B}(y, r) \subseteq F\}. \tag{2}$$

Here $\mathbb{B}(y, r)$ denotes the Euclidean ball of radius r centered at y, that is $\mathbb{B}(y, r) = \{u \in \mathbb{R}^m : \|u - y\| \leq r\}$. Similar results also hold for the relaxation method as established by Goffin [14].

Since their emergence in the fifties, both the perceptron algorithm and the relaxation method have played major roles in machine learning and in optimization. The perceptron algorithm has attractive properties concerning noise tolerance [9]. It is also closely related to large-margin classification [12] and to the highly popular and computationally effective Pegasos algorithm [20] for training support-vector machines. There are also numerous papers in the optimization literature related to various versions and variants of the relaxation method [2, 3, 4, 5, 10].

A major drawback of both the perceptron algorithm and the relaxation method is their lack of theoretical efficiency in the standard bit model of computation [15]. In particular, when $F = \{y : A^{\mathrm{T}}y > 0\}$ with $A \in \mathbb{Z}^{m\times n}$, the perceptron algorithm may have exponential worst-case bit-model complexity because τ_F can be exponentially small in the bit-length representation of A. Our main contribution is a variant of the perceptron algorithm that solves (1) in $\mathcal{O}(m^5 \log(1/\tau_F))$ perceptron updates. In particular, when $F = \{y : A^{\mathrm{T}}y > 0\}$ with $A \in \mathbb{Z}^{m\times n}$, our algorithm is polynomial in the bit-length representation of A. Aside from its theoretical merits, given the close connection between the perceptron algorithm and first-order methods [21], our algorithm provides a solid foundation for potential speed ups in the convergence of the widely popular first-order methods for large-scale convex optimization. Some results of similar nature have been recently obtained by Gilpin et al. [13] and by O'Donoghue and Candès [18].

Our algorithm is based on a periodic rescaling of the space $\mathbb{R}^m$ in the same spirit as in previous work by Dunagan and Vempala [11], and by Belloni, Freund, and Vempala [6]. In contrast to the rescaling procedure in [11, 6], which is randomized and relies on a deep separation oracle, our rescaling procedure is deterministic and relies only on a separation oracle. The algorithm performs at most $\mathcal{O}(m \log(1/\tau_F))$ rescaling steps and at most $\mathcal{O}(m^4)$ perceptron updates between rescaling steps. When $F = \{y \in \mathbb{R}^m : A^{\mathrm{T}}y > 0\}$ for $A \in \mathbb{R}^{m\times n}$, a simplified version of the algorithm has iteration bound $\mathcal{O}(m^2n^2 \log(1/\tau_F))$. A *smooth* version of this algorithm, along the lines developed by Soheili and

Peña [21], in turn has the improved iteration bound $\mathcal{O}(mn\sqrt{m\log(n)}\log(1/\tau_F))$.

Our rescaled perceptron algorithm consists of an outer loop with two main phases. The first one is a perceptron phase and the second one is a rescaling phase. The perceptron phase applies a restricted number of perceptron updates. If this phase does not find a feasible solution, then it finds a unitary vector $d \in \mathbb{R}^m$ such that

$$F \subseteq \left\{ y \in \mathbb{R}^m : 0 \leq \langle d, y \rangle \leq \frac{1}{\sqrt{6m}} \|y\| \right\}.$$

This inclusion means that the feasible cone is nearly perpendicular to d. The second phase of the outer loop, namely a rescaling phase, stretches $\mathbb{R}^m$ along d and is guaranteed to enlarge the volume of the set $\{y \in F : \|y\| = 1\}$ by a constant factor. This in turn implies that the algorithm must halt in at most $\mathcal{O}(m\log(1/\tau_F))$ outer iterations.

2 Polyhedral case

For ease of exposition, we first consider the case $F = \{y \in \mathbb{R}^m : A^{\mathrm{T}} y > 0\}$ for $A \in \mathbb{R}^{m \times n}$.

Assumption 1

(i) The space $\mathbb{R}^m$ is endowed with the canonical dot inner product $\langle u, v \rangle := u^{\mathrm{T}} v$.

(ii) $A = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix}$ where $\|a_i\| = 1$ for $i = 1, \ldots, n$.

(iii) The problem $A^{\mathrm{T}} y > 0$ is feasible. In particular, $\tau_F > 0$.

For $j = 1, \ldots, n$ let $e_j \in \mathbb{R}^n$ denote the vector with jth component equal to one and all other components equal to zero.

Rescaled Perceptron Algorithm

1. let $B := I$; $\tilde{A} := A$; $N := 6mn^2$
2. (Perceptron Phase)
   $x_0 := 0 \in \mathbb{R}^n$; $y_0 := 0 \in \mathbb{R}^m$;
   for $k = 0, 1, \ldots, N-1$
   if $\tilde{A}^{\mathrm{T}} y_k > 0$ then HALT output $B y_k$
   else
   let $j \in \{1, \ldots, n\}$ be such that $\tilde{a}_j^{\mathrm{T}} y_k \leq 0$
   $x_{k+1} := x_k + e_j$
   $y_{k+1} := y_k + \tilde{a}_j$
   end if
   end for
3. (Rescaling Phase)
   $j = \underset{i=1,\ldots,n}{\mathsf{argmax}} \langle e_i, x_N \rangle$
   $B := B(I - \frac{1}{2}\tilde{a}_j \tilde{a}_j^{\mathrm{T}})$; $\tilde{A} := (I - \frac{1}{2}\tilde{a}_j \tilde{a}_j^{\mathrm{T}})\tilde{A}$
   normalize the columns of $\tilde{A}$
4. Go back to Step 2.

The rescaled perceptron algorithm changes the initial constraint matrix A to a new matrix $\tilde{A} = B^{\mathrm{T}}A$. Thus when $\tilde{A}^{\mathrm{T}}y > 0$, the non-zero vector By returned by the algorithm solves $A^{\mathrm{T}}y > 0$.

Now we can state a special version of our main theorem.

Theorem 1 *Assume $A \in \mathbb{R}^{m\times n}$ satisfies Assumption 1. Then the rescaled perceptron algorithm terminates with a solution to $A^{\mathrm{T}}y > 0$ after at most*

$$\frac{1}{\log(1.5)} \cdot \left((m-1)\log\left(\frac{1}{\tau_F\sqrt{1-\tau_F^2}}\right) + \frac{1}{2}\log(\pi)\right) = \mathcal{O}\left(m\log\left(\frac{1}{\tau_F}\right)\right)$$

rescaling steps. Since the algorithm performs $\mathcal{O}(mn^2)$ perceptron updates between rescaling steps, the algorithm terminates after at most

$$\mathcal{O}\left(m^2n^2\log\left(\frac{1}{\tau_F}\right)\right)$$

perceptron updates.

The key ingredients in the proof of Theorem 1 are the three lemmas below. The first of these lemmas states that if the perceptron phase does not solve $\tilde{A}^{\mathrm{T}}y > 0$, then the rescaling phase identifies a column $\tilde{a}_j$ of $\tilde{A}$ that is nearly perpendicular to the feasible cone $\{y : \tilde{A}^{\mathrm{T}}y \geq 0\}$. The second lemma in turn implies that the rescaling phase increases the volume of this cone by a constant factor. The third lemma states that the volume of the initial feasible cone $F = \{y : A^{\mathrm{T}}y \geq 0\}$ is bounded below by a factor of τ_F^{m-1}.

Lemma 1 *If the perceptron phase in the rescaled perceptron algorithm does not find a solution to $\tilde{A}^{\mathrm{T}}y > 0$ then the vector $\tilde{a}_j$ in the first step of the rescaling phase satisfies*

$$\{y : \tilde{A}^{\mathrm{T}}y \geq 0\} \subseteq \left\{y : 0 \leq \tilde{a}_j^{\mathrm{T}}y \leq \frac{1}{\sqrt{6m}}\|y\|\right\}. \tag{3}$$

Proof: Observe that at each iteration of the perceptron phase we have

$$\|y_{k+1}\|^2 = \|y_k\|^2 + 2a_j^{\mathrm{T}}y_k + 1 \leq \|y_k\|^2 + 1.$$

Hence $\|y_k\|^2 \leq k$. Also, throughout the perceptron phase $x_k \geq 0,\ y_k = \tilde{A}x_k$ and $\|x_{k+1}\|_1 = \|x_k\|_1 + 1$. Thus if the perceptron phase does not find a solution to $\tilde{A}^{\mathrm{T}}y > 0$ then the last iterates y_N and x_N satisfy $x_N \geq 0, \|x_N\|_1 = N = 6mn^2$ and $\|y_N\| = \|\tilde{A}x_N\| \leq \sqrt{N} = n\sqrt{6m}$. In particular, the index j in the first step of the rescaling phase satisfies $\langle e_j, x_N\rangle \geq \|x_N\|_1/n = 6mn$. Next observe that if $\tilde{A}^{\mathrm{T}}y \geq 0$ then

$$0 \leq 6mn\,\tilde{a}_j^{\mathrm{T}}y \leq \langle e_j, x_N\rangle\,\tilde{a}_j^{\mathrm{T}}y \leq x_N^{\mathrm{T}}\tilde{A}^{\mathrm{T}}y \leq \|\tilde{A}x_N\|\,\|y\| \leq n\sqrt{6m}\|y\|.$$

So (3) follows. ■

The following two lemmas rely on geometric arguments concerning the unit sphere $\mathbb{S}^{m-1} := \{u \in \mathbb{R}^m : \|u\| = 1\}$. Given a measurable set $C \subseteq \mathbb{S}^{m-1}$, let $\mathrm{Vol}(C)$ denote its volume in $\mathbb{S}^{m-1}$.

We rely on the following construction proposed by Betke [7]. Given $a \in \mathbb{S}^{m-1}$ and $\alpha > 1$, let $\Psi_{a,\alpha} : \mathbb{S}^{m-1} \to \mathbb{S}^{m-1}$ denote the transformation

$$u \mapsto \frac{(I + (\alpha - 1)aa^{\mathrm{T}})u}{\|(I + (\alpha - 1)aa^{\mathrm{T}})u\|} = \frac{u + (\alpha - 1)(a^{\mathrm{T}}u)a}{\sqrt{1 + (\alpha^2 - 1)(a^{\mathrm{T}}u)^2}}.$$

This transformation stretches the sphere in the direction a. The magnitude of the stretch is determined by α.

Lemma 2 *Assume $a \in \mathbb{S}^{m-1}$, $0 < \delta < 1$, and $\alpha > 1$. If $C \subseteq \{y \in \mathbb{S}^{m-1} : 0 \leq a^{\mathrm{T}}y \leq \delta\}$ is a measurable set, then*

$$\mathrm{Vol}\left(\Psi_{a,\alpha}(C)\right) \geq \frac{\alpha}{\left(1 + \delta^2\left(\alpha^2 - 1\right)\right)^{m/2}}\mathrm{Vol}(C). \tag{4}$$

In particular, if $\delta = \frac{1}{\sqrt{6m}}$ and $\alpha = 2$ then

$$\mathrm{Vol}\left(\Psi_{a,\alpha}(C)\right) \geq 1.5\mathrm{Vol}(C). \tag{5}$$

Proof: Without loss of generality assume $a = e_m$. Also for ease of notation, we shall write Ψ as shorthand for $\Psi_{a,\alpha}$. Under these assumptions, for $y = (\bar{y}, y_m) \in \mathbb{S}^{m-1}$ we have

$$\Psi(\bar{y}, y_m) = \frac{(\bar{y}, \alpha y_m)}{\sqrt{\alpha^2 + (1 - \alpha^2)\|\bar{y}\|^2}}.$$

To calculate the volume of C and of $\Psi(C)$, consider the differentiable map $\Phi : \mathbb{B}^{m-1} \to \mathbb{R}^m$, defined by $\Phi(v) = \left(v, \sqrt{1 - \|v\|^2}\right)$ that maps the unit ball $\mathbb{B}^{m-1} := \{v \in \mathbb{R}^{m-1} : \|v\| \leq 1\}$ to the surface of the hemisphere $\{(\bar{y}, y_m) \in \mathbb{S}^m : y_m \geq 0\}$ containing the set C. The volume of C is

$$\mathrm{Vol}(C) = \int_{\Phi^{-1}(C)} |||\Phi'|||dv.$$

where $|||\Phi'|||$ denotes the volume of the $(m-1)$-dimensional parallelepiped spanned by the vectors $\partial\Phi/\partial v_1, \ldots, \partial\Phi/\partial v_{m-1}$. Likewise, the volume of $\Psi(C)$ is

$$\mathrm{Vol}(\Psi(C)) = \int_{\Phi^{-1}(C)} |||(\Psi \circ \Phi)'|||dv.$$

Hence to prove (4) it suffices to show that

$$\frac{|||(\Psi \circ \Phi)'(v)|||}{|||\Phi'(v)|||} \geq \frac{\alpha}{\left(1 + \delta^2\left(\alpha^2 - 1\right)\right)^{m/2}} \text{ for all } v \in \Phi^{-1}(C). \tag{6}$$

Some straightforward calculations show that for all $v \in \mathrm{int}(\mathbb{B}^{m-1})$

$$|||(\Psi \circ \Phi)'(v)||| = \frac{\alpha}{\sqrt{1 - \|v\|^2}\left(\alpha^2 + (1 - \alpha^2)\|v\|^2\right)^{m/2}} \quad \text{and} \quad |||\Phi'(v)||| = \frac{1}{\sqrt{1 - \|v\|^2}}.$$

Hence for all $v \in \mathrm{int}(\mathbb{B}^{m-1})$

$$\frac{|||(\Psi \circ \Phi)'(v)|||}{|||\Phi'(v)|||} = \frac{\alpha}{\left(\alpha^2 + (1 - \alpha^2)\|v\|^2\right)^{m/2}}.$$

To obtain (6), observe that if $v \in \Phi^{-1}(C)$ then $0 \leq 1 - \|v\|^2 \leq \delta^2$ and thus

$$\alpha^2 + (1-\alpha^2)\|v\|^2 \leq 1 + \delta^2\left(\alpha^2 - 1\right).$$

If $\delta = \frac{1}{\sqrt{6m}}$ and $\alpha = 2$ then

$$\frac{\alpha}{\left(1+\delta^2\left(\alpha^2-1\right)\right)^{m/2}} = \frac{2}{\left(1+\frac{1}{2m}\right)^{2m/4}} \geq \frac{2}{\exp(0.25)} \geq 1.5.$$

Thus (5) follows from (4). ■

Lemma 3 *Assume $F \subseteq \mathbb{R}^m$ is a closed convex cone. Then*

$$\mathrm{Vol}(F \cap \mathbb{S}^{m-1}) \geq \left(\tau_F\sqrt{1-\tau_F^2}\right)^{m-1} \frac{1}{2\sqrt{\pi}}\mathrm{Vol}(\mathbb{S}^{m-1}). \tag{7}$$

Proof: From the definition of the cone width τ_F it follows that $\mathbb{B}(z, \tau_F) \subseteq F$ for some $z \in F$ with $\|z\| = 1$. Therefore $(1-\tau_F^2)z + v \in F$ for all $v \in \mathbb{R}^m$ such that $\|v\| \leq \tau_F\sqrt{1-\tau_F^2}$ and $\langle z, v\rangle = 0$. This implies that $F \cap \mathbb{S}^{m-1}$ contains a spherical cap of $\mathbb{S}^{m-1}$ with base radius $\tau_F\sqrt{1-\tau_F^2}$. Hence

$$\mathrm{Vol}(F \cap \mathbb{S}^{m-1}) \geq \left(\tau_F\sqrt{1-\tau_F^2}\right)^{m-1} \mathrm{Vol}(\mathbb{B}^{m-1}).$$

The bound (7) now follows from the facts $\mathrm{Vol}(\mathbb{B}^{m-1}) = \frac{\pi^{\frac{m-1}{2}}}{\Gamma(\frac{m-1}{2}+1)}$, $\mathrm{Vol}(\mathbb{S}^{m-1}) = \frac{2\pi^{\frac{m}{2}}}{\Gamma(\frac{m}{2}+1)}$, and $\Gamma(\frac{m}{2}+1) \geq \Gamma(\frac{m-1}{2}+1)$. ■

Proof of Theorem 1: Let $\tilde{F} := \{y \in \mathbb{R}^m : \tilde{A}^{\mathrm{T}}y \geq 0\}$. Observe that the rescaling phase rescales $\tilde{F}$ to $(I + \tilde{a}_j\tilde{a}_j^{\mathrm{T}})\tilde{F}$. Therefore, Lemma 1 and Lemma 2 imply that after each rescaling phase the quantity $\mathrm{Vol}(\tilde{F} \cap \mathbb{S}^{m-1})$ increases by a factor of 1.5 or more. Since the set $\tilde{F} \cap \mathbb{S}^{m-1}$ is always contained in a hemisphere, we conclude that the number of rescaling steps before the algorithm halts cannot be larger than

$$\frac{1}{\log(1.5)} \cdot \log\left(\frac{\mathrm{Vol}(\mathbb{S}^{m-1})/2}{\mathrm{Vol}(F \cap \mathbb{S}^{m-1})}\right)$$

To finish, apply Lemma 3. ■

3 General case

The gist of the algorithm for the general case of a convex cone F is the same as that of the polyhedral case presented above. We just need a bit of extra work to identify a suitable direction for the rescaling phase. To do so, we maintain a collection of $2m$ index sets $S_j,\ j = \pm1, \pm2, \ldots, \pm m$. This collection of sets helps us determine a subset of update steps that align with each other. The sum of these steps in turn defines the appropriate direction for rescaling.

Assumption 2

(i) The space $\mathbb{R}^m$ is endowed with the canonical dot inner product $\langle \cdot, \cdot \rangle$.

(ii) $F \subseteq \mathbb{R}^m$ is the non-empty interior of a convex cone. In particular, $\tau_F > 0$.

(iii) There is an available separating oracle for the cone F: Given $y \in \mathbb{R}^m$ the oracle either determines that $y \in F$ or else it finds a non-zero vector $u \in F^* := \{u : \langle u, v \rangle > 0 \text{ for all } v \in F\}$ such that $\langle u, y \rangle \leq 0$.

For $j = 1, \ldots, m$ let $e_j \in \mathbb{R}^m$ denote the vector with jth component equal to one and all other components equal to zero.

Observe that for a non-singular matrix $B \in \mathbb{R}^{m \times m}$, we have $(B^{-1}F)^* = B^{\mathrm{T}}F^*$. Thus a separation oracle for $\tilde{F} := B^{-1}F$ is readily available provided one for F is: Given $y \in \mathbb{R}^m$, apply the separation oracle for F to the point By. If $By \in F$ then $y \in B^{-1}F = \tilde{F}$. If $By \notin F$, then let $u \in F^*$ be a non-zero vector such that $\langle u, By \rangle \leq 0$. Thus $\langle B^{\mathrm{T}}u, y \rangle = \langle u, By \rangle \leq 0$ with $B^{\mathrm{T}}u \in (B^{-1}F)^* = \tilde{F}^*$. Consequently, throughout the algorithm below we assume that a separation oracle for the rescaled cone $\tilde{F}$ is available.

General Rescaled Perceptron Algorithm

1. let $B := I$; $\tilde{F} := F$; $N := 24m^4$
2. for $j = \pm 1, \pm 2, \ldots, \pm m$
   $S_j := \emptyset$
   end for
3. (Perceptron Phase)
   $u_0 := 0 \in \mathbb{R}^m$; $y_0 := 0 \in \mathbb{R}^m$;
   for $k = 0, 1, \ldots, N-1$
   if $y_k \in \tilde{F}$ then HALT and output By_k
   else
   let $u_k \in \tilde{F}^*$ be such that $\langle u_k, y_k \rangle \leq 0$ and $\|u_k\| = 1$
   $y_{k+1} = y_k + u_k$
   $j := \underset{i=1,\ldots,m}{\mathsf{argmax}} |\langle e_i, u_k \rangle|$
   if $\langle e_j, u_k \rangle > 0$ then $S_j := S_j \cup \{k\}$
   else $S_{-j} := S_{-j} \cup \{k\}$
   end if
   end if
   end for
4. (Rescaling Phase)
   $i = \underset{j=\pm 1,\ldots,\pm m}{\mathsf{argmax}} |S_j|$
   $d := \frac{\sum_{k \in S_i} u_k}{\| \sum_{k \in S_i} u_k \|}$
   $B := B(I - \frac{1}{2} dd^{\mathrm{T}})$; $\tilde{F} := (I + dd^{\mathrm{T}})\tilde{F}$
5. Go back to Step 2.

The general rescaled perceptron algorithm changes the initial cone F to $\tilde{F} = B^{-1}F$. Thus when $y \in \tilde{F}$, we have $By \in F$. Notice that although the above algorithm implicitly performs this transformation, its steps do not involve inverting any matrices or solving any system of equations.

Now we can state the general version of our main theorem.

Theorem 2 *Assume $F \subseteq \mathbb{R}^m$ is such that Assumption 2 holds. Then the general rescaled perceptron algorithm terminates with a solution to $y \in F$ after at most*

$$\frac{1}{\log(1.5)} \cdot \left((m-1) \log \left(\frac{1}{\tau_F \sqrt{1-\tau_F^2}} \right) + \frac{1}{2} \log(\pi) \right) = \mathcal{O}\left(m \log\left(\frac{1}{\tau_F} \right) \right)$$

rescaling steps. Since the algorithm performs $\mathcal{O}(m^4)$ perceptron updates between rescaling steps, the algorithm terminates after at most

$$\mathcal{O}\left(m^5 \log\left(\frac{1}{\tau_F} \right) \right)$$

perceptron updates.

The proof of Theorem 2 is almost identical to the proof of Theorem 1. All we need is the following analog of Lemma 1.

Lemma 4 *If the perceptron phase in the general rescaled perceptron algorithm does not find a solution to $y \in \tilde{F}$ then the vector d in the rescaling phase satisfies*

$$\tilde{F} \subseteq \left\{ y : 0 \leq \langle d, y \rangle \leq \frac{1}{\sqrt{6m}} \|y\| \right\}. \tag{8}$$

Proof: Proceeding as in the proof of Lemma 1, it is easy to see that if the perceptron phase does not find a solution to $y \in \tilde{F}$ then the last iterate $y_N = \sum_{k=0}^{N-1} u_k$ satisfies $\|y_N\|^2 \leq N = 24m^4$. Since $\{e_1, \ldots, e_m\}$ is an orthonormal basis and each u_k satisfies $\|u_k\| = 1$, we have $|\langle e_j, u_k \rangle| \geq 1/\sqrt{m}$ for $j = \operatorname*{argmax}_{i=1,\ldots,m} |\langle e_i, u_k \rangle|$. Furthermore, since $|\bigcup_{j=\pm 1,\ldots,\pm m} S_j| = N = 24m^4$ it follows that the set S_i in the rescaling phase must have at least $12m^3$ elements. Thus

$$\left\| \sum_{k \in S_i} u_k \right\| \geq \left| \left\langle e_{|i|}, \sum_{k \in S_i} u_k \right\rangle \right| = \sum_{k \in S_i} \left| \langle e_{|i|}, u_k \rangle \right| \geq \frac{|S_i|}{\sqrt{m}} \geq 12m^{5/2}. \tag{9}$$

On the other hand, for all $y \in \tilde{F}$ we have

$$0 \leq \sum_{k \in S_i} \langle u_k, y \rangle \leq \sum_{k=0}^{N-1} \langle u_k, y \rangle = \langle y_N, y \rangle \leq \|y_N\| \|y\| \leq \sqrt{24} m^2 \|y\|. \tag{10}$$

Putting (9) and (10) together, it follows that for all $y \in \tilde{F}$

$$0 \leq \langle d, y \rangle \leq \frac{\sqrt{24} m^2 \|y\|}{12 m^{5/2}} = \frac{1}{\sqrt{6m}} \|y\|.$$

Hence (8) holds. ■

4 Smooth version for the polyhedral case

Consider again the case when $F = \{y \in \mathbb{R}^m : A^{\mathrm{T}} y > 0\}$, where $A \in \mathbb{R}^{m \times n}$. We next show that in this case the perceptron phase can be substituted by a *smooth perceptron phase* by relying on the machinery developed by Soheili and Peña [21]. This leads to an algorithm with a substantially improved convergence rate but whose work per main iteration is roughly comparable to that in the rescaled perceptron algorithm.

Suppose A satisfies Assumption 1. For $\mu > 0$ let $x_\mu : \mathbb{R}^m \to \mathbb{R}^n$ be defined by

$$x_\mu(y) = \frac{e^{-A^{\mathrm{T}} y}}{\|e^{-A^{\mathrm{T}} y}\|_1}.$$

In this expression $e^{\frac{-A^{\mathrm{T}} y}{\mu}}$ is shorthand for the n-dimensional vector

$$e^{\frac{-A^{\mathrm{T}} y}{\mu}} := \begin{bmatrix} e^{\frac{-a_1^{\mathrm{T}} y}{\mu}} \\ \vdots \\ e^{\frac{-a_n^{\mathrm{T}} y}{\mu}} \end{bmatrix}.$$

Let $\mathbf{1} \in \mathbb{R}^n$ denote the n-dimensional vector of all ones. Consider the following smooth version of the rescaled perceptron algorithm.

Smooth Rescaled Perceptron Algorithm

1. let $B := I$; $\tilde{A} := A$; $N := \lfloor 7n\sqrt{m \log(n)} \rfloor$
2. (Smooth Perceptron Phase)
   $y_0 := \frac{\tilde{A}\mathbf{1}}{n}$; $\mu_0 := 2$; $x_0 := x_{\mu_0}(y_0)$;
   for $k = 0, 1, 2, \ldots, N-1$
   if $\tilde{A}^{\mathrm{T}} y_k > 0$ then HALT and output $B y_k$
   else
   $\theta_k := \frac{2}{k+3}$;
   $y_{k+1} := (1-\theta_k)(y_k + \theta_k \tilde{A} x_k) + \theta_k^2 \tilde{A} x_{\mu_k}(y_k)$;
   $\mu_{k+1} := (1-\theta_k)\mu_k$;
   $x_{k+1} := (1-\theta_k) x_k + \theta_k x_{\mu_{k+1}}(y_{k+1})$;
   end if
   end for
3. (Rescaling Phase)
   $j = \underset{i=1,\ldots,n}{\mathsf{argmax}} \langle e_i, x_N \rangle$
   $B := B(I - \frac{1}{2}\tilde{a}_j \tilde{a}_j^{\mathrm{T}})$; $\tilde{A} := (I - \frac{1}{2}\tilde{a}_j \tilde{a}_j^{\mathrm{T}})\tilde{A}$
   normalize the columns of $\tilde{A}$
4. Go back to Step 2.

Theorem 3 *Assume $A \in \mathbb{R}^{m \times n}$ satisfies Assumption 1. Then the smooth rescaled perceptron algorithm terminates with a solution to $A^{\mathrm{T}} y > 0$ after at*

most

$$\frac{1}{\log(1.5)} \cdot \left((m-1)\log\left(\frac{1}{\tau_F\sqrt{1-\tau_F^2}}\right) + \frac{1}{2}\log(\pi) \right) = \mathcal{O}\left(m \log\left(\frac{1}{\tau_F}\right)\right)$$

rescaling steps. Since the algorithm performs $\mathcal{O}(n\sqrt{m\log(n)})$ perceptron updates between rescaling steps, the algorithm terminates after at most

$$\mathcal{O}\left(mn\sqrt{m\log(n)}\log\left(\frac{1}{\tau_F}\right)\right)$$

perceptron updates.

Proof: This proof is a modification of the proof of Theorem 1. It suffices to show that if the smooth perceptron phase in the rescaled perceptron algorithm does not find a solution to $\tilde{A}^{\mathrm{T}}y > 0$ then the vector $\tilde{a}_j$ in the first step of the rescaling phase satisfies

$$\{y : \tilde{A}^{\mathrm{T}}y \geq 0\} \subseteq \left\{ y : 0 \leq \tilde{a}_j^{\mathrm{T}} y \leq \frac{1}{\sqrt{6m}}\|y\| \right\}. \tag{11}$$

Indeed, from [21, Lemma 4.2] it follows that if the perceptron phase does not find a solution to $\tilde{A}^{\mathrm{T}}y > 0$, then $\|\tilde{A}x_N\|^2 \leq \frac{8\log(n)}{(N+1)^2} \leq \frac{8}{49mn^2} \leq \frac{1}{6mn^2}$. Since $x_N \geq 0$ and $\|x_N\|_1 = 1$, the index j in the rescaling phase satisfies $\langle e_j, x_N\rangle \geq \frac{1}{n}$. Therefore, if $\tilde{A}^{\mathrm{T}}y \geq 0$ then

$$0 \leq \frac{1}{n}\,\tilde{a}_j^{\mathrm{T}}y \leq \langle e_j, x_N\rangle\,\tilde{a}_j^{\mathrm{T}}y \leq x_N^{\mathrm{T}}\tilde{A}^{\mathrm{T}}y \leq \|\tilde{A}x_N\|\,\|y\| \leq \frac{1}{\sqrt{6mn}}\|y\|.$$

So (11) follows. ■

References

[1] S. Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6(3):382–392, 1954.

[2] E. Amaldi, P. Belotti, and R. Hauser. A randomized algorithm for the maxfs problem. In *IPCO*, pages 249–264, 2005.

[3] E. Amaldi and R. Hauser. Boundedness theorems for the relaxation method. *Math. Oper. Res.*, 30(4):939–955, 2005.

[4] H. H. Bauschke and J. M. Borwein. Legendre functions and the method of random Bregman projections. *J. Convex Anal.*, 4:27–67, 1997.

[5] H. H. Bauschke, J. M. Borwein, and A. Lewis. The method of cyclic projections for closed convex sets in Hilbert space. *Contemporary Math*, 204:1–38, 1997.

[6] A. Belloni, R. Freund, and S. Vempala. An efficient rescaled perceptron algorithm for conic systems. *Math. Oper. Res*, 34(3):621–641, 2009.

[7] U. Betke. Relaxation, new combinatorial and polynomial algorithms for the linear feasibility problem. *Discrete & Computational Geometry*, 32:317–338, 2004.

[8] H. D. Block. The perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34:123–135, 1962.

[9] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1-2):35–52, 1998.

[10] S. Chubanov. A strongly polynomial algorithm for linear systems having a binary solution. *Math. Program.*, 134:533–570, 2012.

[11] J. Dunagan and S. Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. *Math. Program.*, 114(1):101–114, 2006.

[12] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296, 1999.

[13] A Gilpin, J. Peña, and T. Sandholm. First-order algorithm with $\mathcal{O}(\ln(1/\epsilon))$ convergence for ϵ-equilibrium in two-person zero-sum games. *Math. Program.*, 133:279–298, 2012.

[14] J. Goffin. The relaxation method for solving systems of linear inequalities. *Math. Oper. Res.*, 5:388–414, 1980.

[15] J. Goffin. On the non-polynomiality of the relaxation method for systems of linear inequalities. *Math. Program.*, 22:93–103, 1982.

[16] T. S. Motzkin and I. J. Schoenberg. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6(3):393–404, 1954.

[17] A. B. J. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume XII, pages 615–622, 1962.

[18] B. O'Donoghue and E. J. Candès. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, To Appear.

[19] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Cornell Aeronautical Laboratory, Psychological Review*, 65(6):386–408, 1958.

[20] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: primal estimated sub-gradient solver for SVM. *Math. Program.*, 127:3–30, 2011.

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