A POLYNOMIAL ALGORITHM FOR LINEAR FEASIBILITY PROBLEMS GIVEN BY SEPARATION ORACLES

SERGEI CHUBANOV

University of Siegen, Germany

ABSTRACT. We consider the problem of finding nonzero vectors in full-dimensional polyhedral cones given by systems of linear inequalities. It is assumed that there is a separation oracle which, given a vector, verifies whether it belongs to the cone and, in the case of infeasibility, returns a violated constraint of the respective system of linear inequalities. The algorithm proposed in this paper runs in polynomial oracle time, i.e., in a number of arithmetic operations and calls to the separation oracle bounded by a polynomial in the number of variables and in the maximum binary size of an entry of the coefficient matrix. This algorithm is much simpler than traditional polynomial algorithms such as the ellipsoid method and the volumetric barrier method. In particular, we do not need the notion of volume to prove the polynomial complexity of our algorithm.

1. INTRODUCTION

In this paper, we propose an algorithm which is able to solve systems of linear inequalities given by a separation oracle in polynomial oracle time, i.e., in a number of arithmetic operations and calls to the separation oracle which is polynomially bounded in the number of variables and in the maximum binary size of an entry of the coefficient matrix. The sizes of the numbers in the course of the algorithm are polynomially bounded as well.

This algorithm is a generalization of the polynomial algorithm for linear programming whose variants are presented in Chubanov [2] and [3]. That algorithm is based on a procedure which finds either a solution or a sufficiently short convex combination of the rows of the coefficient matrix. Whenever a sufficiently short combination is found, that algorithm performs a suitable coordinate rescaling, i.e., it multiplies some columns of the matrix by some positive value.

The algorithm proposed in the present paper performs rescalings (or space dilations, from another point of view) of a more general form. This type of rescaling is in fact used by the ellipsoid method. When applied to a linear system $Ax \geq b$, the first iteration of the ellipsoid method considers a ball containing a feasible solution provided that the system is feasible. Then it finds a constraint violated at the center and constructs an ellipsoid of a smaller volume containing the intersection of the ball with the halfspace defined by the violated constraint. Now let A be multiplied from the right by the inverse of the matrix of a linear transformation which maps the ellipsoid into a ball. The described steps are applied to the obtained system, with respect to the current ball, and so on. The ellipsoid method can be implemented to run in polynomial time, as proved by Khachiyan [8].

E-mail address: sergei.chubanov@uni-siegen.de.

Procedures for finding short vectors of the mentioned form, combined with rescaling of this type, lead to polynomial algorithms for linear programming; see e.g. Dadush, Végh, and Zambelli [5], where a polynomial algorithm is proposed for conic linear systems given explicitly and Peña and Soheili [11], who present a polynomial algorithm based on the perceptron method. The algorithm of [11] runs in a polynomial number of calls to the separation oracle. In both [5] and [11], the main argument used to show that the number of rescalings is polynomially bounded is based on analyzing the relative growth of the volumes of some bodies associated with the problem in question.

To guarantee that a polynomial-time separation oracle leads to a polynomial running time, one should ensure that the input for the separation oracle is of polynomial size whenever the oracle is called. One should note that both [5] and [11] use the real model of computation and do not provide any analysis of the bit complexity in the traditional Turing model.

So the contribution of this paper is the following:

- We show that a simple generalization of the procedure proposed in Chubanov [3] can be combined with the rescaling technique so that both the running time and the sizes of the numbers are polynomially bounded. That is, a polynomial separation oracle leads to a polynomial running time.
- Our complexity analysis is based only on the most fundamental properties of the determinant such as Silvester's determinant theorem and Hadamard's inequality. As a result, we obtain a simple non-ellipsoidal algorithm which solves linear systems given by separation oracles in polynomial oracle time.

2. Algorithm

We will denote a zero vector by $\mathbf{0}$ and an all-one vector by $\mathbf{1}$. The dimensions of these vectors are determined by the context.

Let us consider a linear feasibility problem written in the following conic form:

$$Ax \geq \mathbf{0}, x \neq \mathbf{0}, \tag{2.1}$$

where A is a rational matrix with n columns and the linear system $Ax \geq \mathbf{0}$ is given by a separation oracle, i.e., by an algorithm that decides whether a given point x is feasible and returns a violated constraint if x is infeasible. The problem asks for a nonzero vector $x \in \mathbb{R}^n$ in the cone defined by $Ax \geq \mathbf{0}$.

Let m denote the number of rows of A. Note that m may be exponential in n. Further, whenever we say that a certain parameter such as the running time or the size of a rational number is polynomially bounded, we mean that this parameter is bounded by a polynomial in the number of variables n and in the maximum binary size of a coefficient of A.

We assume w.l.o.g. that the squares of the (Euclidean) norms of the rows of A belong to $[1, 3/2]$. (If necessary, we could use a procedure to normalize the respective coefficient vector, whenever a constraint is returned by the separation oracle.)

Also, we assume that $Ax \geq \mathbf{0}$ either defines a *full-dimensional* cone or has no nonzero solutions.

Let us now consider $y \in \mathbb{R}^m$ such that

$$y \geq \mathbf{0}, \quad \mathbf{1}^T y = 1, \quad \mathbf{1}^T \mathbf{1}_y \leq 2n. \tag{2.2}$$

The $\mathbf{1}$ is the all-one vector of dimension m and $\mathbf{1}_y$ is the characteristic vector of y, i.e., the binary vector having ones at exactly those positions where y has nonzero components. In (2.2), we require that y is a nonnegative vector whose components sum up to 1 and which has at most $2n$ nonzero components.

Note that Karatheodory's lemma implies that any convex combination of the rows of A can be represented in the form $A^T y$ where y has no more than $n+1$ nonzero components, i.e., where y satisfies (2.2).

Let y_k be a maximum component of y. Note that $1 \geq y_k \geq 1/(2n)$. Let A_k be the respective row of A. Consider a vector a. Let $\Delta = A_k^T - a$. For any x with $Ax \geq \mathbf{0}$ we have the following:

$$\begin{aligned} \tfrac{1}{2n}|a^T x| &\leq y_k|a^T x| \\ &= |y_k(A_k^T - \Delta)^T x| \\ &\leq |y_k A_k x| + y_k|\Delta^T x| \\ &\leq |y^T Ax| + |\Delta^T x| \\ &\leq (\|y^T A\| + \|\Delta\|) \cdot \|x\|. \end{aligned} \tag{2.3}$$

The procedure described in the proof of the lemma below will further be called the *basic procedure.*

Lemma 2.1. *Given $\mu \in (0,1]$, in $O(n^2/\mu^2)$ calls to the separation oracle and $O(n^4/\mu^2)$ arithmetic operations we can find a solution of (2.1) or a vector a with $\|a\|^2 \in [1,2]$ such that*

$$|a^T x| \leq \mu\|x\| \tag{2.4}$$

for all x with $Ax \geq \mathbf{0}$, and the components of a are integer multiples of $\frac{\mu^2}{4n^2}$.

Proof. Consider an arbitrary rational nonzero vector in $\mathbb{R}^n$ of polynomial size (say, $\mathbf{1}$). Let $A_l x \geq 0$ be a violated constraint (otherwise, return that vector as a feasible solution). Set $y := \mathbf{e}_l$, where $\mathbf{e}_l$ denotes the vector whose lth component is equal to one and whose other components are zeros. Further, we assume that y is stored as a list of its nonzero components. If $A^T y$ is a feasible solution, then we are done. Otherwise, let $A_i x \geq 0$ be an inequality of the system $Ax \geq \mathbf{0}$ violated at $A^T y$. Let

$$y' := \alpha y + (1-\alpha)\mathbf{e}_i, \quad \alpha = \frac{A_i(A_i^T - A^T y)}{\|A_i^T - A^T y\|^2}.$$

Geometrically, $A^T y'$ is the orthogonal projection of $\mathbf{0}$ on the segment $[A^T y, A_i^T]$. In the special case when $\|A^T y'\| = 0$, we are done (see the stopping condition below). Otherwise, it is not hard to prove that, taking into account $\|A_i\|^2 \in [1, 3/2]$,

$$\frac{1}{\|A^T y'\|^2} \geq \frac{1}{\|A^T y\|^2} + \frac{1}{\|A_i\|^2} \geq \frac{1}{\|A^T y\|^2} + \frac{1}{2}. \tag{2.5}$$

A similar observation lies in the basis of the procedure used in Chubanov [3], which in turn can be viewed as a variation of the procedure proposed by von Neumann (and communicated by Dantzig in [6]) to solve systems of linear inequalities with a given accuracy. Note that the number of nonzero components of y' may exceed that of y by at most one. Set $y := y'$ and repeat the above steps. To ensure that the number of nonzero components of y does not exceed $2n$ in the above process, whenever y' has $2n+1$ nonzero components, we find a vertex solution of

$$A_J^T v = A^T y', \mathbf{1}^T v = 1, v \geq \mathbf{0}, \tag{2.6}$$

where J is the set of nonzero indices of y' and A_J is the submatrix of A corresponding to the nonzero components of y'. Let y'_J be obtained from y' by taking the components with indices in J. Since $v = y'_J$ is a feasible solution of (2.6) and A_J^T has at most $2n+1$ columns, we can find a vertex solution v' of (2.6) in time $O(n^3)$, starting with y'_J; for technical details see, for instance, [4]. Then we set $y_J := v'$, where y_J refers to the components with indices in J, and continue the described procedure. We stop when a solution of (2.1) is found or $\|A^T y\| \le \mu/(4n)$. In the latter case, let y_k be a maximum component of y and a be obtained by rounding the negative components of A_k^T down and the positive components up to the nearest multiple of $\mu^2/(4n^2)$. Then $\|\Delta\| \le \mu/(4n)$, where $\Delta = A_k^T - a$. The required inequality (2.4) takes place according to (2.3). Moreover, since $\|A_k\|^2 \in [1, 3/2]$,

$$1 \le \|A_k\| \le \|a\| \le \|A_k\| + \frac{1}{4} \le 2.$$

Note that (2.5) implies that the number of iterations of the above procedure is bounded by $O(n^2/\mu^2)$. To compute y' at each iteration, we need to call the separation oracle and perform $O(n)$ arithmetic operations. The $O(n^3)$ algorithm for finding vertex solutions of (2.6) is needed only if the current y' has $2n+1$ nonzeros. On the other hand, the number of nonzero components of y increases by at most one with each iteration of the procedure, which means that the $O(n^3)$ algorithm is needed no more frequently than after every n iterations of the procedure. The rounding of A_k to obtain a is needed only once and runs in $O(n \log n)$ time. Thus, we obtain the required complexity estimates. ∎

Given $a \in \mathbb{R}^n$, each $x \in \mathbb{R}^n$ can be uniquely represented as

$$x = w_x + \lambda_x a,$$

where

$$a^T w_x = 0.$$

We understand the pair (w_x, λ_x) as a function of x. Consider the following linear operator:

$$f(x) = w_x + 2\lambda_x a.$$

Its inverse is

$$f^{-1}(x) = w_x + \frac{1}{2}\lambda_x a.$$

The matrices of f and f^{-1} are

$$I + \frac{1}{\|a\|^2} a a^T \text{ and } I - \frac{1}{2\|a\|^2} a a^T,$$

respectively. That is, to compute $f(x)$ and $f^{-1}(x)$, we can multiply x from the left by the respective matrices.

Lemma 2.2. *Let $\|a\|^2 \in [1, 2]$. Let x be such that*

$$|a^T x| \le \frac{\|x\|}{\sqrt{6n}}.$$

Then

$$\|f(x)\| \le \|x\| \cdot \sqrt{1 + \frac{1}{n}}.$$

.

Proof. The x can be represented as

$$x = w_x + \lambda_x a.$$

Note that,

$$|\lambda_x| \cdot \|a\|^2 = |a^T x| \le \frac{\|x\|}{\sqrt{6n}}.$$

Then, since $\|a\|^2 \geq 1$,

$$|\lambda_x| \leq \frac{\|x\|}{\sqrt{6n}}.$$

Taking into account $\|a\|^2 \leq 2$, we can write

$$\|f(x)\|^2 = \|w_x + 2\lambda_x a\|^2 = \|w_x\|^2 + 4\lambda_x^2\|a\|^2 = \|x\|^2 + 3\lambda_x^2\|a\|^2 \leq \|x\|^2 + \frac{\|x\|^2}{n}.$$

That is, we have proved the required inequality. ∎

Assume that (2.1) is feasible and let X^0 be an $n \times n$ matrix, where each column x^0 has the following property:

$$Ax^0 \geq \mathbf{0}, \quad \|x^0\| \leq 1. \tag{2.7}$$

It is clear that the set of such matrices is not empty whenever (2.1) is feasible.

Now we consider the following algorithm:

Algorithm 2.1.

Input: A positive integer t;
Output: Either a solution of (2.1) or a decision that

$$|\det X^0| \leq \left(\frac{e}{4}\right)^{t/2} \tag{2.8}$$

(where e is Euler's number) for all $n \times n$-matrices X^0 with (2.7).

0. Let D be the identity matrix.
1. Let $\bar{A}$ denote the matrix obtained from AD by normalizing the rows so that the squares of their norms get into $[1, 3/2]$. The obtained system defines the same cone as $AD \geq \mathbf{0}$. For normalizing a row of AD whenever necessary, we use any suitable polynomial-time procedure (which is further called the normalization procedure) being able, given a vector v, to find a positive rational value $\gamma(v)$ such that $\gamma(v)^2 \cdot \|v\|^2 \in [1, 3/2]$. That is,
$$\bar{A}_i = \gamma(A_i D) \cdot A_i D, \quad i = 1, \ldots, m.$$
Setting $\mu := 1/\sqrt{6n}$ and using the basic procedure (described in the proof of Lemma 2.1), find $z = \bar{A}^T y$ which is nonzero and feasible for $\bar{A}x \geq \mathbf{0}$ or a vector a with $\|a\|^2 \in [1, 2]$, such that
$$|a^T x| \leq \frac{\|x\|}{\sqrt{6n}}$$
for all x with $\bar{A}x \geq \mathbf{0}$ and the components of a are integer multiples of
$$\mu^2/(4n^2) = 1/(24n^3).$$
2. If the respective z is found, then return $x^* = Dz$. Else,
$$D := D\left(I - \frac{1}{2\|a\|^2} aa^T\right).$$
3. Repeat the above steps. If the number of repetitions exceeds t, then stop.

Remark 2.1. *A separation oracle for the current system $\bar{A}x \geq \mathbf{0}$ can be easily obtained from the original separation oracle; x^0 is feasible for this system if and only if it is feasible for $ADx \geq \mathbf{0}$. That is, x^0 is feasible for the current system if and only if Dx^0 is feasible for the original system $Ax \geq \mathbf{0}$. Let $A_i x \geq \mathbf{0}$ be violated at $x = Dx^0$. Then $\gamma(A_i D) \cdot A_i Dx \geq 0$ is violated at $x = x^0$. Thus, all we need to find a violated constraint of $\bar{A}x \geq \mathbf{0}$, at a given x^0, is to call the original separation oracle for Dx^0 and compute $\gamma(A_i D)$, where $A_i D$ corresponds to the violated constraint* *found, by means of the normalization procedure. Note that, when A_i is found, the row vector A_iD can be computed in $O(n^2)$ arithmetic operations.*

Assume that (2.1) is feasible and let X^0 be an $n \times n$ matrix, where each column x^0 has the property (2.7).

Consider the first iteration of Algorithm 2.1 and assume that Step 2 does not return a feasible solution. Let D be the matrix constructed at Step 2. It is the matrix of f^{-1} defined with respect to a. That is,

$$D = I - \frac{1}{2\|a\|^2} aa^T, \quad D^{-1} = I + \frac{1}{\|a\|^2} aa^T.$$

According to Step 1, since a feasible solution is still not found, for all feasible solutions x of $Ax \geq \mathbf{0}$ we have

$$|a^T x| \leq \frac{\|x\|}{\sqrt{6n}}.$$

This inequality is guaranteed by Lemma 2.1. Then, for each column x^0 of X^0,

$$|a^T x^0| \leq \frac{\|x\|}{\sqrt{6n}}.$$

Therefore, by Lemma 2.2, for each column x^0 of X^0,

$$\|D^{-1}x^0\| = \|f(x^0)\| \leq \|x^0\| \cdot \sqrt{1 + \frac{1}{n}} \leq \sqrt{1 + \frac{1}{n}}. \tag{2.9}$$

Then,

$$2|\det X^0| = |\det D^{-1}X^0| \leq \left(\sqrt{1 + \frac{1}{n}}\right)^n < e^{\frac{1}{2}}. \tag{2.10}$$

To obtain the equation in (2.10), we use the fact that

$$\det\left(I + \frac{1}{\|a\|^2} aa^T\right) = 1 + \frac{\|a\|^2}{\|a\|^2} = 2.$$

by Silvester's determinant identity. To obtain the last but one inequality in (2.10), we use (2.9) (the norms of the columns of X^0 are not greater than 1) and Hadamard's inequality, which implies that the absolute value of the determinant of a square matrix is not greater than the product of the norms of its columns.

Consider the first s iterations of Algorithm 2.1 and the respective matrix D at the end of the sth iteration. Applying the above argument inductively, we obtain:

$$2^s|\det X^0| = |\det D^{-1}X^0| \leq \left(\sqrt{1 + \frac{1}{n}}\right)^{ns} < e^{\frac{s}{2}}. \tag{2.11}$$

The set of all x^0 with (2.7) is a compact convex set having full dimension. Let X^* be a matrix with the maximum determinant among the matrices X^0 whose columns satisfy (2.7). Then

$$|\det X^*| > 0.$$

According to Remark 2.1, each iteration of the basic procedure uses the normalization procedure after a violated constraint of $ADx \geq \mathbf{0}$ is found. Let T be an upper bound on the running time needed for the normalization procedure at each iteration of the basic procedure in the course of Algorithm 2.1.

Theorem 2.1. *Algorithm 2.1 can be implemented to find a solution in*

$$O\left(n^3 \log(|\det X^*|^{-1})\right).$$

calls to the separation oracle and

$$O\left((n^5 + n^3 T)\log(|\det X^*|^{-1})\right)$$

arithmetic operations, provided that the system in question (the system (2.1)) is feasible and $Ax \geq \mathbf{0}$ defines a full-dimensional cone.

Proof. Let t be sufficiently large. The number of iterations of the algorithm is bounded by

$$O\left(\log(|\det X^*|^{-1})\right),$$

which follows from (2.11) (note that $e^{1/2} < 2$). At Step 1, we call the basic procedure with

$$\mu = 1/\sqrt{6n}.$$

Then, by Lemma 2.1, the basic procedure runs inu $O(n^3)$ iterations whenever it is called from Algorithm 2.1. The separation oracle for the current system $\bar{A}x \geq \mathbf{0}$ requires one call to the separation oracle for the original system and $O(n^2)$ arithmetic operations; see Remark 2.1. ∎

Corollary 2.1. *Algorithm 2.1 can be implemented to solve (2.1) in a polynomial number of arithmetic operations and calls to the separation oracle. If the separation oracle is a polynomial algorithm, then Algorithm 2.1 solves (2.1) in polynomial time.*

Proof. Using standard results in linear programming, it is not hard to determine a polynomially bounded value t such that

$$|\det X^*| > \left(\frac{e}{4}\right)^{t/2},$$

provided that (2.1) is feasible (in this case, by our assumption, $Ax \geq \mathbf{0}$ defines a full-dimensional cone). If the algorithm does not return a solution, then (2.1) is infeasible because otherwise we arrive to a contradiction due to the above inequality and (2.8).

The normalization procedure can be implemented so that T is polynomially bounded. To see this, note that the binary size of D is bounded by a polynomial in n and $\log|\det X^*|^{-1}$.

Now we should prove that the basic procedure can be modified so as to guarantee that the sizes of the numbers in the course of the procedure are polynomially bounded, the theoretical running time being preserved. Let the basic procedure be applied to the system $Ax \geq \mathbf{0}$. Consider an iteration of the procedure and let y be the new vector y constructed at this iteration. Let this iteration be not the last one. Then

$$\|A^T y\| > \frac{\mu}{2n} \geq \sqrt{8\delta} \geq 8\delta > \delta, \tag{2.12}$$

where

$$\delta = \frac{\mu^2}{16n^2}.$$

Let y be stored as a list of its nonzero values. (The respective rows of A can also be stored as a list.) By the size of y we understand the sum of binary sizes of the nonzero components of y.

Using the fact that y has at most $2n$ nonzero components according to the basic procedure, in $O(n \log n)$ time (see the procedure given below), one can find a vector $y^\#$ such that

$$y^\# \geq \mathbf{0}, \quad \mathbf{1}^T y^\# = 1, \quad y^\# \leq \mathbf{1}_y,$$

whose size is bounded by a polynomial in n such that

$$\|\Delta\| \leq \frac{\delta^2}{1-\delta}.$$

where

$$\Delta = A^T(y^\# - y).$$

Now we give a procedure for constructing $y^\#$. Below, we show that a vertex solution of the following system gives us a suitable $y^\#$:

$$\mathbf{1}^T z = 1, \quad \kappa \left\lfloor \frac{y}{\kappa} \right\rfloor \leq z \leq \kappa \left\lceil \frac{y}{\kappa} \right\rceil,$$

where

$$\kappa = \frac{1}{4n^2} \cdot \frac{\delta^2}{1-\delta}.$$

The running time of $O(n \log n)$ refers to constructing the lower and upper bounds, in the above system, for those components which correspond to the nonzero components of y (the number of nonzero components of y is bounded by $2n$.) The other lower and upper bounds on the components of z are zero. For each solution z of the above system, taking into account that the norms of the rows of A are bounded by 2, we can write

$$\|A^T(z-y)\| \leq 2n\|z-y\| \leq 2n\|z-y\|_1 \leq 4n^2\kappa = \frac{\delta^2}{1-\delta}.$$

A vertex solution (represented by enumerating the nonzero components) can be found in time $O(n)$ starting with $z = y$. Its size is bounded by a polynomial in n. So we have shown how to find a suitable $y^\#$ of polynomial size.

Note that

$$\|\Delta\| \leq \frac{\delta}{1-\delta} \cdot \|A^T y\|,$$

because of (2.12). It follows that

$$\frac{1}{\|A^T y^\#\|} = \frac{1}{\|A^T y + \Delta\|} \geq \frac{1}{\|A^T y\| + \|\Delta\|} = \frac{1}{\|A^T y\|} \cdot \frac{\|A^T y\|}{\|A^T y\| + \|\Delta\|} \geq \frac{1}{\|A^T y\|} - \frac{\delta}{\|A^T y\|}.$$

Then, taking into account (2.12), we conclude that

$$\frac{1}{\|A^T y^\#\|^2} \geq \frac{1}{\|A^T y\|^2} - \frac{2\delta}{\|A^T y\|^2} \geq \frac{1}{\|A^T y\|^2} - \frac{1}{4} \geq \frac{1}{\|A^T y^0\|^2} + \frac{1}{4},$$

where y^0 is the old y. To obtain the last inequality, we use (2.5).

Set $y := y^\#$ at the end of each iteration of the basic procedure. Let the obtained procedure be called the modified basic procedure. The theoretical running time is preserved because the reciprocal of $\|A^T y\|^2$ in the modified procedure increases by a constant with each iteration, which follows from the above inequalities.

At each iteration of the main algorithm, $\|a\|^2 \in [1, 2]$ and the components of a are integer multiples of $\mu^2/(4n^2)$. Therefore, the size of D is polynomially bounded in n and in the number of iterations, at each iteration of the main algorithm. That is, using the modified basic procedure in place of the original one, we guarantee that the sizes of the numbers in the course of the algorithm are polynomially bounded. Therefore, Algorithm 2.1 can be implemented to solve the problem in question in polynomial oracle time. ∎

References

[1] Agmon, Sh. 1954. The relaxation method for linear inequalities. *Canad. J. Math.* **6**, 382-392.

[2] Chubanov, S. 2012. A strongly polynomial algorithm for linear systems having a binary solution. *Mathematical Programming* **134**, 533-570.

[3] Chubanov, S. 2014. A polynomial projection algorithm for linear feasibility problems. *Mathematical Programming.* DOI 10.1007/s10107-014-0823-8

[4] Chubanov, S. 2016. A Polynomial-Time Descent Method for Separable Convex Optimization Problems with Linear Constraints. *SIAM Journal on Optimization* **26**, 856-889.

[5] Dadush, D., Végh, L., and Zambelli, G. Rescaling Algorithms for Linear Programming Part I: Conic feasibility. arXiv:1611.06427v3

[6] Dantzig, G.: An ε-precise feasible solution to a linear program with a convexity constraint in $1/\varepsilon^2$ iterations independent of problem size. Tech. rep. SOL 92-5 (1992)

[7] Goffin, J.L. 1982. On the non-polynomiality of the relaxation method for systems of linear inequalities. *Mathematical Programming* **22**, 93-103.

[8] Khachiyan, L.G. 1979. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR 244* (English translation: Soviet Math. Dokl. **20**, 191-194).

[9] Korte, B., and Vygen, J. 2002. Combinatorial optimization: theory and algorithms. Springer.

[10] Motzkin, Th., Schoenberg, I. J. 1954. The relaxation method for linear inequalities. *Canad. J. Math.* **6**, 393-404.

[11] Peña, J. and Soheili, N.. A deterministic rescaled perceptron algorithm. Mathematical Programming, 155(1-2):497-510, 2016.