Mathematical Programming 63 (1994) 83–108
North-Holland

A deep cut ellipsoid algorithm for convex programming: Theory and applications

J.B.G. Frenk and J. Gromicho*

Econometric Institute, Erasmus University, Rotterdam, Netherlands

S. Zhang

Department of Econometrics, University of Groningen, Netherlands

Received 21 October 1991
Revised manuscript received 15 February 1993

This paper proposes a deep cut version of the ellipsoid algorithm for solving a general class of continuous convex programming problems. In each step the algorithm does not require more computational effort to construct these deep cuts than its corresponding central cut version. Rules that prevent some of the numerical instabilities and theoretical drawbacks usually associated with the algorithm are also provided. Moreover, for a large class of convex programs a simple proof of its rate of convergence is given and the relation with previously known results is discussed. Finally some computational results of the deep and central cut version of the algorithm applied to a min–max stochastic queue location problem are reported.

Key words: Convex programming, deep cut ellipsoid algorithm, rate of convergence, location theory, min–max programming.

1. Introduction

This paper is divided into two parts.

The first part, contained in Section 2, discusses the ellipsoid algorithm. In this part a so-called deep cut version of this algorithm for solving a class of convex programming problems is presented. Also, rate of convergence results are given. We emphasize that the convergence proof of the computationally attractive deep cut version is simple and elementary contrary to the proof of a similar result for a corresponding central cut version as reported for the unconstrained case in [16] and [34] and for the constrained case in [24]. Moreover, the proof unifies results for deep and central cut versions and shows the influence of deep cuts on the convergence rate. Finally, it can be extended to a large class of quasiconvex programs (cf. [12]).

The second part, contained in Section 3, presents a min–max model in location theory in

Correspondence to: Prof. J.B.G. Frenk, Econometric Institute, Erasmus University, P.O. Box 1783, 3000 DR Rotterdam, Netherlands.

*Author on leave from D.E.I.O. (Universidade de Lisboa, Portugal). This research was supported by J.N.I.C.T. (Portugal) under contract number BD/631/90-RM.

0025-5610 © 1994—The Mathematical Programming Society, Inc. All rights reserved
SSDI 0025-5610(93)E0033-B

which the objective function incorporates the waiting time for service of customers. The objective uses also so-called (convex) disutility functions and for the linear case the objective function is worked out in detail. To this special convex programming problem we apply the two versions of the algorithm and report the computational results.

2. The ellipsoid algorithm

Before proposing a deep cut version of the ellipsoid algorithm we present a general overview.

2.1. Overview

Early papers by Shor (cf. [32] and [31]) are considered to be the start of the ellipsoid algorithm. Later, Yudin and Nemirovsky (cf. [42] and [43]) observed its implications in convex programming. The explicit statement of this algorithm is due to Shor (cf. [33]). The algorithm became very well-known by a publication of Khachiyan in 1979 stating that the ellipsoid algorithm can be used to prove the polynomial time solvability of linear programming problems (cf. [21]). Later, the ellipsoid algorithm has been used to prove the polynomial time solvability of a large class of combinatorial optimization problems (cf. [18] or [19]). For a very well written survey of the early applications of the ellipsoid algorithm to linear programming we refer to [3]. Recently the connections between the ellipsoid algorithm and the quasi-Newton algorithm for nonlinear programming and Karmarkar's algorithm for linear programming have also been studied (cf. [17] and [41]). Contrary to its behavior in linear programming it also seems (cf. [8], [7] and [9]) that a central cut version of the ellipsoid algorithm is robust for general nonlinear programming problems and, relative to efficiency, competitive with other general purpose algorithms.

For a mathematical description of the ellipsoid algorithm we need to introduce an ellipsoid. A set $E \subseteq \mathbb{R}^n$ is called an ellipsoid if there exists a vector $\boldsymbol{a} \in \mathbb{R}^n$ and a positive definite $n \times n$-matrix $\boldsymbol{A}$ such that

$$E = E(\boldsymbol{A}; \boldsymbol{a}) := \{\boldsymbol{x} \in \mathbb{R}^n \colon (\boldsymbol{x} - \boldsymbol{a})^{\mathrm{t}} \boldsymbol{A}^{-1} (\boldsymbol{x} - \boldsymbol{a}) \leqslant 1\}.$$

Moreover, in order to determine whether a given hyperplane in $\mathbb{R}^n$ with normal $\boldsymbol{c}$ intersects an ellipsoid $E(\boldsymbol{A}; \boldsymbol{a})$ we observe (cf. [19]) that

$$\min\{\boldsymbol{c}^{\mathrm{t}} \boldsymbol{x} \colon \boldsymbol{x} \in E(\boldsymbol{A}; \boldsymbol{a})\} = \boldsymbol{c}^{\mathrm{t}} \boldsymbol{a} - \sqrt{\boldsymbol{c}^{\mathrm{t}} \boldsymbol{A} \boldsymbol{c}} \tag{1}$$

and

$$\max\{\boldsymbol{c}^{\mathrm{t}} \boldsymbol{x} \colon \boldsymbol{x} \in E(\boldsymbol{A}; \boldsymbol{a})\} = \boldsymbol{c}^{\mathrm{t}} \boldsymbol{a} + \sqrt{\boldsymbol{c}^{\mathrm{t}} \boldsymbol{A} \boldsymbol{c}}. \tag{2}$$

This implies that the hyperplane

$$H(\beta) := \{\boldsymbol{x} \in \mathbb{R}^n \colon \boldsymbol{c}^{\mathrm{t}} \boldsymbol{x} = \beta\}$$

with $-1 \leqslant \alpha \leqslant 1$ and $\alpha := (\boldsymbol{c}^{\mathrm{t}} \boldsymbol{a} - \beta) / \sqrt{\boldsymbol{c}^{\mathrm{t}} \boldsymbol{A} \boldsymbol{c}}$ has a nonempty intersection with $E(\boldsymbol{A}; \boldsymbol{a})$. It is now possible to construct for $-1/n < \alpha \leqslant 1$ a minimum volume ellipsoid $E(\boldsymbol{A}_1; \boldsymbol{a}_1)$

containing the intersection $E(\boldsymbol{A};\boldsymbol{a})\cap H^-(\beta)$ with $H^-(\beta):=\{\boldsymbol{x}\in\mathbb{R}^n\colon \boldsymbol{c}^{\mathrm{t}}\boldsymbol{x}\leqslant\beta\}$ the lower halfspace corresponding to $H(\beta)$ and this ellipsoid has a strictly smaller volume than $E(\boldsymbol{A};\boldsymbol{a})$. Moreover, its formula is given by (cf. [3] or [19])

$$\boldsymbol{a}_1=\boldsymbol{a}-\frac{1+n\alpha}{n+1}\boldsymbol{b}, \tag{3}$$

$$\boldsymbol{A}_1=\frac{n^2}{n^2-1}(1-\alpha^2)\left(\boldsymbol{A}-\frac{2(1+n\alpha)}{(n+1)(1+\alpha)}\boldsymbol{b}\boldsymbol{b}^{\mathrm{t}}\right), \tag{4}$$

with

$$\alpha:=(\boldsymbol{c}^{\mathrm{t}}\boldsymbol{a}-\beta)/\sqrt{\boldsymbol{c}^{\mathrm{t}}\boldsymbol{A}\boldsymbol{c}} \quad\text{and}\quad \boldsymbol{b}:=\boldsymbol{A}\boldsymbol{c}/\sqrt{\boldsymbol{c}^{\mathrm{t}}\boldsymbol{A}\boldsymbol{c}}.$$

Taking the same matrix $\boldsymbol{Q}$ as described on page 151 of [28] and copying with some obvious modifications the proof in Proposition 2.7 and 2.8 of [28] one can show that $\boldsymbol{A}_1$ is positive definite given that $\alpha^2<1$ and $\boldsymbol{A}$ is positive definite.

This concludes our brief description of the ellipsoid algorithm. Observe that the main problem in applying this algorithm is to construct in each step a hyperplane in such a way that the optimal solution of our optimization problem belongs to the intersection of the current ellipsoid and the constructed lower halfspace. We note here that for $\alpha=0$ (resp. $0<\alpha\leqslant 1$) the hyperplane is called a valid central cut (resp. valid deep cut).

2.2. *Analysis and description of the algorithm*

Consider the problem

$$\text{(P)}\qquad \inf_{\boldsymbol{x}\in S} f(\boldsymbol{x})$$

where $f\colon\mathbb{R}^n\to\mathbb{R}$ denotes a finite convex function on $\mathbb{R}^n$ and $S\subseteq\mathbb{R}^n$ some closed convex set. In this paper we assume that this so-called feasible region S is given by

$$S:=\{\boldsymbol{x}\in\mathbb{R}^n\colon g_j(\boldsymbol{x})\leqslant 0, j=1,\ldots,m\}$$

with $g_j:\mathbb{R}^n\to\mathbb{R}, j=1,\ldots,m$, a set of finite convex functions on $\mathbb{R}^n$. It should be noted here since each function g_j, $1\leqslant j\leqslant m$, is actually continuous on $\mathbb{R}^n$ (cf. [1]) that S is indeed a closed convex set. Moreover, since the maximum of a finite number of finite convex functions is finite and convex we may take in the definition of S, without loss of generality, the number of different functions equal to one, i.e. $m=1$. For simplicity we will call it g instead of g_1. A similar argument also applies to the objective function f and so optimization problem (P) also covers min–max problems.

In order to introduce a deep cut version of the ellipsoid algorithm we need to make the following assumption.

Assumption 2.2.1. An optimal solution $\boldsymbol{x}^*$ of (P) exists for which an upper bound r on the Euclidean norm of $\boldsymbol{x}^*$ is known, i.e. $\|\boldsymbol{x}^*\|_2<r$.

As observed by one of the referees the technique of generating deep cuts that we are going to present for both the objective function and for the constraint was first introduced in [35] and later submitted to extensive computational study in [7].

Let us introduce now the set $B(\mathbf{0},r) := \{\boldsymbol{x} \in \mathbb{R}^n: \|\boldsymbol{x}\|_2 < r\}$. Clearly, following Assumption 2.2.1,

$$f(\boldsymbol{x}^*) = \min\{f(\boldsymbol{x}): \boldsymbol{x} \in S\} = \min\{f(\boldsymbol{x}): \boldsymbol{x} \in S \cap B(\mathbf{0},r)\}$$

and so we can start the ellipsoid algorithm by taking $\overline{B(\mathbf{0},r)} := \{\boldsymbol{x} \in \mathbb{R}^n: \|\boldsymbol{x}\|_2 \leqslant r\}$ as the initial ellipsoid $E(\boldsymbol{A}_0; \mathbf{0})$ with $\boldsymbol{A}_0 = \rho \boldsymbol{I}$ and $\rho := r^2$.

Suppose we are at the mth step of the procedure and the current ellipsoid $E(\boldsymbol{A}_m; \boldsymbol{a}_m)$ contains $\boldsymbol{x}^*$. Let us define the lowest recorded feasible value until iteration m as $l_m := \min\{f(\boldsymbol{a}_k): k \leqslant m, \boldsymbol{a}_k \in S\}$. After observing that clearly $l_m \geqslant f(\boldsymbol{x}^*)$ for every $m \in \mathbb{N}$ we may distinguish three different cases.

Case 1. $\boldsymbol{a}_m \in S \cap B(\mathbf{0},r)$. Since f is finite and convex on $\mathbb{R}^n$ it follows that for every $\boldsymbol{x} \in S$ the subgradient set $\partial f(\boldsymbol{x})$ is nonempty (cf. [1]) and hence for every $\boldsymbol{d}_m \in \partial f(\boldsymbol{a}_m)$ the so-called subgradient inequality holds

$$f(\boldsymbol{x}^*) \geqslant f(\boldsymbol{a}_m) + \boldsymbol{d}_m^{\mathrm{t}}(\boldsymbol{x}^* - \boldsymbol{a}_m). \tag{5}$$

Observe, if $\boldsymbol{d}_m = \mathbf{0}$ then $\boldsymbol{a}_m$ is optimal and therefore there is no need for a cut. For a derivation of a deep or central valid cut with respect to f observe the following. If $\boldsymbol{d}_m^{\mathrm{t}}(\boldsymbol{x}^* - \boldsymbol{a}_m) > l_m - f(\boldsymbol{a}_m)$ it follows by (5) that

$$f(\boldsymbol{x}^*) > f(\boldsymbol{a}_m) + l_m - f(\boldsymbol{a}_m) = l_m$$

and this is not possible by the definition of $\boldsymbol{x}^*$. Hence $\boldsymbol{x}^*$ must belong to the lower halfspace $H^-(\beta_m) := \{\boldsymbol{x} \in \mathbb{R}^n: \boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{x} \leqslant \beta_m\}$ with $\beta_m := \boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{a}_m + l_m - f(\boldsymbol{a}_m)$. We will now verify whether the hyperplane $H(\beta_m)$ corresponds to a valid cut. Observe by the subgradient inequality for f, $\boldsymbol{x}^* \in E(\boldsymbol{A}_m; \boldsymbol{a}_m)$ and (1) that

$$\begin{aligned} 0 \leqslant f(\boldsymbol{a}_m) - l_m \leqslant f(\boldsymbol{a}_m) - f(\boldsymbol{x}^*) &\leqslant \boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{a}_m - \boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{x}^* \\ &\leqslant \boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{a}_m - \min\{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{x}: \boldsymbol{x} \in E(\boldsymbol{A}_m; \boldsymbol{a}_m)\} \\ &= \sqrt{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{A}_m\boldsymbol{d}_m} \end{aligned} \tag{6}$$

and hence

$$0 \leqslant \alpha_m := \frac{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{a}_m - \beta_m}{\sqrt{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{A}_m\boldsymbol{d}_m}} = \frac{f(\boldsymbol{a}_m) - l_m}{\sqrt{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{A}_m\boldsymbol{d}_m}} \leqslant 1 \tag{7}$$

implying that $H(\beta_m)$ is a valid cut. Clearly this is a valid deep cut whenever $l_m < f(\boldsymbol{a}_m)$ and it can be derived using only one additional computation. Substituting $\alpha := \alpha_m$, $\beta := \beta_m$ and $\boldsymbol{c} := \boldsymbol{d}_m$ it follows by (3) and (4) that in this case a smaller volume ellipsoid $E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$ can be constructed satisfying $\boldsymbol{x}^* \in E(\boldsymbol{A}_m; \boldsymbol{a}_m) \cap H^-(\beta_m) \subseteq E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$ and so we are finished discussing the construction of a valid cut for f. In the remainder we will refer to

such a cut as an objective cut. Finally, in order to derive a stopping rule, notice by the definition of α_m and (6) that

$$0 \leqslant l_m - f(\boldsymbol{x}^*) = l_m - f(\boldsymbol{a}_m) + f(\boldsymbol{a}_m) - f(\boldsymbol{x}^*) \leqslant (1-\alpha_m)\sqrt{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{A}_m \boldsymbol{d}_m} \tag{8}$$

holds for every objective cut $m \geqslant 0$ and hence

$$l_m - f(\boldsymbol{x}^*) \leqslant \min\{(1-\alpha_k)\sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k}\colon k \leqslant m,\ k \text{ is an objective cut}\}. \tag{9}$$

Case 2. $\boldsymbol{a}_m \in B^{\mathrm{c}}(\boldsymbol{0}, r)$.[1] If this subcase holds we construct a valid cut with respect to the function $h(\boldsymbol{x}) = \|\boldsymbol{x}\|_2$. We shall refer to such cuts as norm cuts. Observe that h is convex and its gradient $\nabla h(\boldsymbol{x})$ exists for every $\boldsymbol{x} \neq \boldsymbol{0}$ and equals $\boldsymbol{x}/\|\boldsymbol{x}\|_2$. Clearly by the subgradient inequality, Assumption 2.2.1 and (1) it follows that

$$\begin{aligned} r > h(\boldsymbol{x}^*) &\geqslant h(\boldsymbol{a}_m) + \nabla h(\boldsymbol{a}_m)^{\mathrm{t}}(\boldsymbol{x}^* - \boldsymbol{a}_m) \\ &= \nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{x}^* \geqslant \min\{\nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{x}\colon \boldsymbol{x} \in E(\boldsymbol{A}_m; \boldsymbol{a}_m)\} \\ &= \|\boldsymbol{a}_m\|_2 - \sqrt{\nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{A}_m \nabla h(\boldsymbol{a}_m)} \end{aligned} \tag{10}$$

and so we conclude by the second inequality in (10) that $\boldsymbol{x}^*$ must belong to the lower halfspace $H^-(r) := \{\boldsymbol{x} \in \mathbb{R}^n\colon \nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{x} \leqslant r\}$. Moreover, applying (10) again we obtain

$$0 \leqslant \alpha_m := \frac{\nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{a}_m - r}{\sqrt{\nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{A}_m \nabla h(\boldsymbol{a}_m)}} = \frac{\|\boldsymbol{a}_m\|_2 - r}{\sqrt{\nabla h(\boldsymbol{a}_m)^{\mathrm{t}} \boldsymbol{A}_m \nabla h(\boldsymbol{a}_m)}} < 1 \tag{11}$$

and this yields that the hyperplane $H(r)$ is a valid cut. Clearly this is a valid deep cut whenever $\|\boldsymbol{a}_m\|_2 > r$. Substituting $\alpha := \alpha_m$, $\beta := r$ and $\boldsymbol{c} := \nabla h(\boldsymbol{a}_m)$ it follows by (3) and (4) that also in this case a smaller volume ellipsoid $E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$ can be constructed satisfying $\boldsymbol{x}^* \in E(\boldsymbol{A}_m; \boldsymbol{a}_m) \cap H^-(\beta_m) \subseteq E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$.

Finally we consider the last subcase.

Case 3. $\boldsymbol{a}_m \in S^{\mathrm{c}} \cap B(\boldsymbol{0}, r)$. If this holds we construct a valid deep cut with respect to the function g. We shall refer to such cuts as constraint cuts. As in the first subcase we obtain

$$0 \geqslant g(\boldsymbol{x}^*) \geqslant g(\boldsymbol{a}_m) + \boldsymbol{d}_m^{\mathrm{t}}(\boldsymbol{x}^* - \boldsymbol{a}_m) \geqslant g(\boldsymbol{a}_m) - \sqrt{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{A}_m \boldsymbol{d}_m} \tag{12}$$

with some nonzero $\boldsymbol{d}_m \in \partial g(\boldsymbol{a}_m)$ and hence by the second inequality in (12) $\boldsymbol{x}^*$ belongs to the lower halfspace defined by $H^-(\beta_m) := \{\boldsymbol{x} \in \mathbb{R}^n\colon \boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{x} \leqslant \beta_m\}$ with $\beta_m := \boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{a}_m - g(\boldsymbol{a}_m)$. Moreover, applying (12) again

$$0 \leqslant \alpha_m := \frac{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{a}_m - \beta_m}{\sqrt{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{A}_m \boldsymbol{d}_m}} = \frac{g(\boldsymbol{a}_m)}{\sqrt{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{A}_m \boldsymbol{d}_m}} \leqslant 1 \tag{13}$$

and this yields that the hyperplane $H(\beta_m)$ is a valid cut. Clearly it is a valid deep cut

[1] $A^{\mathrm{c}} \subseteq \mathbb{R}^n$ denotes the complement of the set A in $\mathbb{R}^n$.

whenever $g(\boldsymbol{a}_m) > 0$. Substituting $\alpha := \alpha_m$, $\beta := \beta_m$ and $\boldsymbol{c} := \boldsymbol{d}_m$ it follows by (3) and (4) that again in this case a smaller volume ellipsoid $E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$ can be constructed satisfying $\boldsymbol{x}^* \in E(\boldsymbol{A}_m; \boldsymbol{a}_m) \cap H^-(\beta_m) \subseteq E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$.

This concludes the description of the three disjoint subcases and leads to the determination of the smaller volume ellipsoid to be used in the $(m+1)$th step.

Before giving a complete description of the algorithm we recall (cf. [3, 19]) that the ellipsoid $E(\boldsymbol{A}_{m+1}; \boldsymbol{a}_{m+1})$ given by

$$\boldsymbol{A}_{m+1} := \delta_m(\boldsymbol{A}_m - \sigma_m \boldsymbol{b}_m \boldsymbol{b}_m^{\mathrm{t}}), \qquad \boldsymbol{a}_{m+1} := \boldsymbol{a}_m - \tau_m \boldsymbol{b}_m,$$

with the updating values

$$\delta_m := \frac{n^2(1-\alpha_m^2)}{n^2-1}, \quad \sigma_m := \frac{2(1+n\alpha_m)}{(n+1)(1+\alpha_m)}, \quad \tau_m := \frac{1+n\alpha_m}{n+1}$$

and

$$\boldsymbol{b}_m := \boldsymbol{A}_m \boldsymbol{d}_m / \sqrt{\boldsymbol{d}_m^{\mathrm{t}} \boldsymbol{A}_m \boldsymbol{d}_m}$$

is the smallest volume ellipsoid containing $E(\boldsymbol{A}_m; \boldsymbol{a}_m) \cap H^-(\beta_m)$.

The algorithm consists now of the following steps.

Step 0. let $m := 0$, $\boldsymbol{A}_0 := \rho \boldsymbol{I}$ and $\boldsymbol{a}_0 := \boldsymbol{0}$;
Step 1. **if** $\boldsymbol{a}_m$ is feasible and optimal **then** goto Step 4
else goto Step 2;
Step 2. **if** $\boldsymbol{a}_m \notin B(\boldsymbol{0}, r)$ **then** apply a norm cut
else if $\boldsymbol{a}_m \notin S$ **then** apply a constraint cut
else apply an objective cut;
Step 3. update the ellipsoid, let $m := m+1$ and return to Step 1;
Step 4. stop.

This algorithm includes both the central and the deep cut versions. For the central cut just take $\alpha_m := 0$, for the deep cut evaluate α_m according to the subcases discussed in this subsection.

Except for the first condition in Step 2 this algorithm is similar to the variant V1V3 of the ellipsoid algorithm studied in [7]. Our contribution to Step 2 is expressed by the first rule to be evaluated which aims to improve the numerical stability of the algorithm by trying to keep the centers of the generated ellipsoids inside a bounded region of the space.

Finally we observe for the general case that the above algorithm might be difficult to implement due to the non-availability of a computationally easy optimality check. Although in some cases a fast algorithm is available to check for optimality (cf. [11]) this might in general not be true especially for the nondifferentiable case. This difficulty is caused by the fact that it is sometimes not possible to derive an easy description of the subgradient sets of the functions f and g. Therefore we need to introduce a computationally easy stopping rule to apply in Step 1 of the algorithm. If we are interested in an absolute error of less than a

given $\varepsilon > 0$ we observe by (8) that this will be achieved if at step m an objective cut is performed and the inequality

$$(1-\alpha_m)\sqrt{\boldsymbol{d}_m^{\mathrm{t}}\boldsymbol{A}_m\boldsymbol{d}_m} < \varepsilon$$

holds. However, if we know additionally that $f(\boldsymbol{x}^*) > 0$ it is sometimes more reasonable to consider the relative error

$$(l_m - f(\boldsymbol{x}^*))/f(\boldsymbol{x}^*).$$

The algorithm will now be stopped at the mth step if an objective cut is performed at this step and the inequality

$$\begin{aligned}&\min\{(1-\alpha_k)\sqrt{\boldsymbol{d}_k^{\mathrm{t}}\boldsymbol{A}_k\boldsymbol{d}_k}\colon k \leqslant m,\ k \text{ is an objective cut}\}\\ &\quad < \varepsilon \max\{f(\boldsymbol{a}_k) - \sqrt{\boldsymbol{d}_k^{\mathrm{t}}\boldsymbol{A}_k\boldsymbol{d}_k}\colon k \leqslant m,\ k \text{ is an objective cut}\}\end{aligned}$$

holds. Observe by (6) that $f(\boldsymbol{x}^*) \geqslant f(\boldsymbol{a}_k) - \sqrt{\boldsymbol{d}_k^{\mathrm{t}}\boldsymbol{A}_k\boldsymbol{d}_k}$ for every objective cut $k \leqslant m$. If the stopping rule is satisfied this yields that

$$f(\boldsymbol{x}^*) \geqslant \max\{f(\boldsymbol{a}_k) - \sqrt{\boldsymbol{d}_k^{\mathrm{t}}\boldsymbol{A}_k\boldsymbol{d}_k}\colon k \leqslant m,\ k \text{ is an objective cut}\} > 0$$

and hence we finally obtain by (9) and the stopping rule that

$$(l_m - f(\boldsymbol{x}^*))/f(\boldsymbol{x}^*) < \varepsilon$$

and so we have found a feasible solution within a $1+\varepsilon$ relative error of the optimal solution. This stopping rule was used in our computational experiments discussed in the last section. On the other hand, if it turns out that $\alpha_k = 1$ for some k (this is possible only if k is not a norm cut) then by (1) the intersection of $E(\boldsymbol{A}_k;\ \boldsymbol{a}_k)$ and $H^-(\beta_k)$ consists of one point which is necessarily the optimal solution since $\boldsymbol{x}^* \in E(\boldsymbol{A}_k;\ \boldsymbol{a}_k) \cap H^-(\beta_k)$ holds for every $k \geqslant 0$.

In the next subsection we will provide a simple and elementary convergence proof which covers both versions of the algorithm.

2.3. Convergence proof

In this subsection we assume that the described algorithm has already performed m steps, $m = 1, 2, \ldots$, with centers $\boldsymbol{a}_k$, $k \leqslant m$, and no optimality check or stopping rule was applied. By the last remark in the previous subsection we may assume without loss of generality that $0 \leqslant \alpha_k < 1$ and $\boldsymbol{d}_k \neq \boldsymbol{0}$ for every $k \leqslant m$.

For the proof of Lemma 2.3.2 we now need some well-known results from linear algebra.

Lemma 2.3.1. *For every matrix* $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ *and vectors* $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ *such that* $\det(\boldsymbol{A}) \neq 0$ *and* $\det(\boldsymbol{A} + \boldsymbol{a}\boldsymbol{b}^{\mathrm{t}}) \neq 0$ *we have*

$$(\boldsymbol{A} + \boldsymbol{A}\boldsymbol{b}^{\mathrm{t}})^{-1} = \boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}\boldsymbol{a}\boldsymbol{b}^{\mathrm{t}}\boldsymbol{A}^{-1}}{1 + \boldsymbol{b}^{\mathrm{t}}\boldsymbol{A}^{-1}\boldsymbol{a}},$$

$$\det(\boldsymbol{A} + \boldsymbol{a}\boldsymbol{b}^{\mathrm{t}}) = (1 + \boldsymbol{b}^{\mathrm{t}}\boldsymbol{A}^{-1}\boldsymbol{a})\det(\boldsymbol{A}).$$

Proof. The first formula is called the Sherman–Morrison formula and can be found in [23]. For the proof of the second formula we observe by well-known elementary properties of determinants (cf. [23]) that

$$\begin{aligned}\det(A+ab^{\mathrm{t}}) &= \det\left(\begin{bmatrix}1 & b^{\mathrm{t}}\\ 0 & A+ab^{\mathrm{t}}\end{bmatrix}\right)\\ &= \det\left(\begin{bmatrix}1 & b^{\mathrm{t}}\\ -a & A\end{bmatrix}\right)\\ &= \det\left(\begin{bmatrix}1+b^{\mathrm{t}}A^{-1}a & 0^{\mathrm{t}}\\ -a & A\end{bmatrix}\right)\\ &= (1+b^{\mathrm{t}}A^{-1}a)\det(A)\end{aligned}$$

which finishes the proof. □

In order to prove the next lemma we introduce for every positive definite matrix A_k and $d_k \neq 0$ the ratio $e_k := \sqrt{d_k^{\mathrm{t}} A_k d_k} / \|d_k\|_2$.

Lemma 2.3.2. *If the described algorithm has performed m steps without applying the stopping rule or an optimality check then*

$$\sum_{k=0}^{m}\left(\prod_{j=0}^{k-1}\delta_j\right)\frac{\sigma_k}{1-\sigma_k}e_k^{-2} \geqslant \frac{n}{\rho}\left(\prod_{k=0}^{m}\frac{1}{\sqrt[n]{1-\sigma_k}}-1\right)$$

with $\prod_{j=0}^{-1}\delta_j = 1$.

Proof. From Lemma 2.3.1 and the remarks at the beginning of this subsection it is easy to verify that

$$A_{m+1}^{-1} = \delta_m^{-1}\left(A_m^{-1} + \frac{\sigma_m}{1-\sigma_m}\frac{d_m d_m^{\mathrm{t}}}{d_m^{\mathrm{t}} A_m d_m}\right), \tag{14}$$

$$\det(A_{m+1}) = \delta_m^n(1-\sigma_m)\det(A_m). \tag{15}$$

Moreover, if $\mathrm{tr}(A)$ denotes the trace of matrix A this yields by (14) that

$$\mathrm{tr}(A_{m+1}^{-1}) = \delta_m^{-1}\,\mathrm{tr}(A_m^{-1}) + \delta_m^{-1}\frac{\sigma_m}{1-\sigma_m}e_m^{-2}.$$

Iterating the previous formula for $m \geqslant 0$ we obtain

$$\mathrm{tr}(A_{m+1}^{-1}) = \mathrm{tr}(A_0^{-1})\prod_{k=0}^{m}\delta_k^{-1} + \sum_{k=0}^{m}\left(\prod_{j=k}^{m}\delta_j^{-1}\right)\frac{\sigma_k}{1-\sigma_k}e_k^{-2}$$

and, since $\mathrm{tr}(A_0^{-1}) = n/\rho$, this simplifies to

$$\operatorname{tr}(A_{m+1}^{-1}) = \frac{n}{\rho}\prod_{k=0}^{m}\delta_k^{-1} + \sum_{k=0}^{m}\left(\prod_{j=k}^{m}\delta_j^{-1}\right)\frac{\sigma_k}{1-\sigma_k}\,e_k^{-2}. \tag{16}$$

If (15) is also iterated for $m \geqslant 0$ it follows that

$$\det(A_{m+1}) = \det(A_0)\prod_{k=0}^{m}(\delta_k^n(1-\sigma_k))$$

and this implies using $\det(A_0) = \rho^n$ and $\det(A^{-1}) = \det(A)^{-1}$ that

$$\det(A_{m+1}^{-1}) = \rho^{-n}\prod_{k=0}^{m}(\delta_k^{-n}(1-\sigma_k)^{-1}). \tag{17}$$

Since $\operatorname{tr}(A_{m+1}^{-1}) = \sum_{i=1}^{n}\lambda_i$ and $\det(A_{m+1}^{-1}) = \prod_{i=1}^{n}\lambda_i$ with λ_i, $i=1, \ldots, n$, the positive eigenvalues of the positive definite matrix A_{m+1}^{-1} we obtain by the geometric-arithmetic mean inequality (cf. [38]) that

$$n\sqrt[n]{\det(A_{m+1}^{-1})} \leqslant \operatorname{tr}(A_{m+1}^{-1}). \tag{18}$$

Substituting (16) and (17) in (18) we finally obtain

$$\sum_{k=0}^{m}\left(\prod_{j=k}^{m}\delta_j^{-1}\right)\frac{\sigma_k}{1-\sigma_k}\,e_k^{-2} \geqslant \frac{n}{\rho}\left(\prod_{k=0}^{m}\delta_k^{-1}\right)\left(\prod_{k=0}^{m}\frac{1}{\sqrt[n]{1-\sigma_k}} - 1\right)$$

and multiplying the last expression by $\prod_{k=0}^{m}\delta_k$ the desired result follows. □

Let us now define the following parameters as functions of the space dimension

$$a := (n^2-1)/n^2, \qquad b := \sqrt[n]{(n+1)/(n-1)},$$

and note that $ab > \mathrm{e}^{1/n^2} > 1$ for every $n \geqslant 2$.

For each iteration we also define $D_m := \min\{(1-\alpha_k)e_k : k \leqslant m\}$.

Theorem 2.3.1. *If the deep cut ellipsoid algorithm, without applying a stopping rule or optimality check, is executed an infinite number of iterations then*

$$\lim_{m\to\infty} D_m = 0.$$

Moreover, the convergence is geometric at a rate of $1/\sqrt{ab}$ *if* $\alpha_m = 0$ *for every* m *(central cut version) and at a possibly higher rate whenever* $\alpha_m > 0$ *for some* m *(deep cut version).*

Proof. D_m is clearly a nonincreasing and nonnegative sequence. Also, without loss of generality, we may assume as observed in the beginning of this subsection that $0 \leqslant \alpha_k < 1$ and $\boldsymbol{d}_k \neq \boldsymbol{0}$ for every $k \geqslant 0$.

Observe that after some rewriting the inequality

$$\frac{2}{n-1}\sum_{k=0}^{m}a^{-k}\prod_{j=0}^{k-1}(1-\alpha_j^2)\,\frac{1+n\alpha_k}{(1-\alpha_k)e_k^2} \geqslant \frac{n}{\rho}\left(b^{m+1}\prod_{k=0}^{m}\sqrt[n]{\frac{1+\alpha_k}{1-\alpha_k}} - 1\right)$$

follows from Lemma 2.3.2. Since $0 \leqslant \alpha_k < 1$ for every k, the inequalities $\prod_{j=0}^{k-1}(1-\alpha_j^2) \leqslant 1$ and $1+n\alpha_k < 1+n$ are satisfied and hence the above expression implies

$$2b^n \sum_{k=0}^{m} a^{-k} \frac{1}{(1-\alpha_k)e_k^2} > \frac{n}{\rho}\left(b^{m+1} \prod_{k=0}^{m} \sqrt[n]{\frac{1+\alpha_k}{1-\alpha_k}} - 1\right). \tag{19}$$

By definition $D_m \leqslant (1-\alpha_k)e_k$ for every $k \leqslant m$ and since $(1-\alpha_k)^2 \leqslant (1-\alpha_k)$ it follows that

$$D_m^2 \leqslant (1-\alpha_k)e_k^2.$$

Hence by (19) we obtain that

$$2b^n D_m^{-2} \sum_{k=0}^{m} a^{-k} > \frac{n}{\rho}\left(b^{m+1} \prod_{k=0}^{m} \sqrt[n]{\frac{1+\alpha_k}{1-\alpha_k}} - 1\right).$$

Now, observing that $\sum_{k=0}^{m} a^{-k} = (1-a^{-(m+1)})/(1-a^{-1})$, the last inequality yields

$$D_m^2 < \mathscr{D}_m^2 := 2\rho \frac{(n+1)^2}{n} \frac{(a^{-1})^{m+1}-1}{b^{m+1}\prod_{k=0}^{m} \sqrt[n]{(1+\alpha_k)/(1-\alpha_k)} - 1}.$$

Note that $\prod_{k=0}^{m} \sqrt[n]{(1+\alpha_k)/(1-\alpha_k)} \geqslant 1$ for every m and so

$$\mathscr{D}_m^2 \leqslant \mathscr{C}_m^2 := 2\rho \frac{(n+1)^2}{n} \frac{(a^{-1})^{m+1}-1}{b^{m+1}-1}.$$

Hence it follows using $1/(ab) < 1$ that $\mathscr{C}_m^2 \to 0$ as $m \to \infty$ and its geometric convergence rate is of the order $1/(ab)$. However, if $\alpha_k > 0$ for some k then $\mathscr{D}_m^2$ might decrease faster, and so this might also hold for D_m^2. Finally if $D_m^2 \to 0$ at a rate of at least $1/(ab)$ then $D_m \to 0$ at a rate of at least $1/\sqrt{ab}$. □

Still using elementary techniques we will relate the behavior of the sequence D_m, $m \geqslant 0$, to the behavior of the nonincreasing sequence $l_m - f(x^*)$, $m \geqslant 0$. In order to do so the following notation is necessary.

$$D_m^h := \min\{(1-\alpha_k)e_k \colon k \leqslant m,\ k \text{ is a norm cut}\},$$

$$D_m^g := \min\{(1-\alpha_k)e_k \colon k \leqslant m,\ k \text{ is a constraint cut}\},$$

$$D_m^f := \min\{(1-\alpha_k)e_k \colon k \leqslant m,\ k \text{ is an objective cut}\}.$$

To avoid ambiguities D_m^h, D_m^g and D_m^f are set equal to infinity if the corresponding sets are empty. Since at each step the algorithm either performs a norm cut, a constraint cut or an objective cut is clear that $D_m = \min\{D_m^h, D_m^g, D_m^f\}$.

It is now possible to prove the next result for D_m^h.

Lemma 2.3.3. *There exists a positive constant $\delta > 0$ such that $D_m^h \geqslant \delta$ for every $m \geqslant 0$.*

Proof. Let $m \geqslant 0$ be given and suppose there exists some $k \leqslant m$ such that the algorithm

performs a norm cut during step k. If such a k does not exist we are finished since in that case $D_m^h = +\infty$. To continue the analysis of the first case we observe by Assumption 2.2.1 that there exists some $\delta > 0$ such that $\|x^*\|_2 \leqslant r - \delta$ and hence by (10) it follows that

$$r - \delta \geqslant \|x^*\|_2 \geqslant \|a_m\|_2 - \sqrt{\nabla h(a_m)^t A_m \nabla h(a_m)}.$$

This implies by (11) and $\|\nabla h(a_m)\|_2 = 1$ that

$$\begin{aligned}(1-\alpha_m)e_m &= \left(1 - \frac{\|a_m\|_2 - r}{\sqrt{\nabla h(a_m)^t A_m \nabla h(a_m)}}\right)\frac{\sqrt{\nabla h(a_m)^t A_m \nabla h(a_m)}}{\|\nabla h(a_m\|_2} \\ &= \sqrt{\nabla h(a_m)^t A_m \nabla h(a_m)} - \|a_m\|_2 + r \\ &\geqslant \delta > 0\end{aligned}$$

and so the result is proved. □

By Theorem 2.3.1 and Lemma 2.3.3 it follows that there exists some m_0 such that $D_m = \min\{D_m^g, D_m^f\}$ for $m \geqslant m_0$. This means that for m big enough we only have to study the behavior of the sequences D_m^g and D_m^f. Another elementary result useful for the proof of the main theorem is given by the next lemma.

Lemma 2.3.4. *If* $L_{m_n} := \{x \in \mathbb{R}^n : f(x) \leqslant f(x^*) + nD_{m_n}\}$ *and* $l_{m_n} - f(x^*) > nD_{m_n}$ for some $m_n \in \mathbb{N}$ *then* $S_{m_n} \subseteq E(A_k; a_k)$ *for every* $k \leqslant m_n$ *with* $S_{m_n} = L_{m_n} \cap S \cap \overline{B(0,r)}$.

Proof. The result will be verified by induction. Clearly $S_{m_n} \subseteq \overline{B(0,r)} = E(A_0; a_0)$. Suppose now $S_{m_n} \subseteq E(A_k; a_k)$ for some $k \leqslant m_n - 1$ and consider a_k. If a_k belongs to $S^c \cap B(0,r)$ the algorithm performs a constraint cut and so $d_k \in \partial g(a_k)$ and $\beta_k = d_k^t a_k - g(a_k)$. By the subgradient inequality applied to g it follows for every $x \in S$ that

$$d_k^t(x - a_k) \leqslant g(x) - g(a_k) \leqslant -g(a_k)$$

and hence $S \subseteq H^-(\beta_k)$. This implies $S_{m_n} \subseteq H^-(\beta_k)$ and by the induction hypothesis we obtain $S_{m_n} \subseteq H^-(\beta_k) \cap E(A_k; a_k) \subseteq E(A_{k+1}; a_{k+1})$. Similarly one can show for $a_k \in B^c(0,r)$ that $\overline{B(0,r)} \subseteq H^-(\beta_k)$ and so $S_{m_n} \subseteq H^-(\beta_k) \cap E(A_k; a_k) \subseteq E(A_{k+1}; a_{k+1})$. Finally consider the case that a_k belongs to $S \cap B(0,r)$. If this holds we obtain $d_k \in \partial f(a_k)$ and $\beta_k = l_k - f(a_k)$ and by the subgradient inequality and $l_{m_n} - f(x^*) > nD_{m_n}$ it follows for every $x \in L_{m_n}$ that

$$d_k^t(x - a_k) \leqslant f(x) - f(a_k) \leqslant f(x^*) + nD_{m_n} - f(a_k) < l_{m_n} - f(a_k) \leqslant l_k - f(a_k).$$

Hence $L_{m_n} \subseteq H^-(\beta_k)$ and as in the previous cases it follows $S_{m_n} \subseteq H^-(\beta_k) \cap E(A_k; a_k) \subseteq E(A_{k+1}; a_{k+1})$. □

In order to prove the main theorem and link the behavior of D_m to D_m^f we need the following regularity condition. This condition is related to Slater's condition which shows up in the proof of strong duality in convex programming (cf. [38]).

Assumption 2.3.1. There exists some $\boldsymbol{x} \in \overline{B(\mathbf{0},r)}$ with $g(\boldsymbol{x}) < 0$.

We may now prove the following convergence theorem.

Theorem 2.3.2. *If the deep cut ellipsoid algorithm without applying a stopping rule or an optimality check is executed an infinite number of iterations and Assumption* 2.2.1 *and* 2.3.1 *hold then*

$$\lim_{m \uparrow \infty} l_m = f(\boldsymbol{x}^*).$$

Moreover, it follows that

$$\limsup_{m \uparrow \infty} \frac{l_m - f(\boldsymbol{x}^*)}{D_m} < \infty.$$

Proof. Clearly by Theorem 2.3.1 the sequence l_m converges to $f(\boldsymbol{x}^*)$ if $\limsup_{m \uparrow \infty} (l_m - f(\boldsymbol{x}^*))/D_m < \infty$. Observe that this also implies that the sequence $l_m - f(\boldsymbol{x}^*)$ has at least the same convergence rate as the sequence D_m. In order to prove this result we first assume that there exists an optimal solution $\boldsymbol{x}^*$ with $g(\boldsymbol{x}^*) < 0$ and $\|\boldsymbol{x}^*\|_2 < r$. If this holds there exists some $\delta > 0$ such that $g(\boldsymbol{x}^*) \leqslant -\delta$. Moreover, if during step k a constraint cut is performed we obtain by (12) that

$$-\delta \geqslant g(\boldsymbol{x}^*) \geqslant g(\boldsymbol{a}_k) - \sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k}$$

and hence by (13)

$$\begin{aligned}(1-\alpha_k)e_k &= (1 - g(\boldsymbol{a}_k)/\sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k})\sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k}/\|\boldsymbol{d}_k\|_2 \\ &= (\sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k} - g(\boldsymbol{a}_k))/\|\boldsymbol{d}_k\|_2 \\ &\geqslant \delta/\|\boldsymbol{d}_k\|_2.\end{aligned} \tag{20}$$

Observe now by the convexity of g on $\overline{B(\mathbf{0},r)}$ that g is Lipschitz continuous on $\overline{B(\mathbf{0},r)}$ with Lipschitz constant L_g (cf. [38]) and so $\partial g(\boldsymbol{a}_k) \subseteq \overline{B(\mathbf{0},L_g)}$ for every $\boldsymbol{x} \in \overline{B(\mathbf{0},r)}$. This implies $\|\boldsymbol{d}_k\|_2 \leqslant L_g$ and by (20) we obtain $(1-\alpha_k)e_k \geqslant \delta/L_g > 0$. Hence $D_m^g \geqslant \delta/L_g > 0$ for every $m \geqslant 0$ and so by Lemma 2.3.3 and Theorem 2.3.1 there exists some m_1 such that $D_m = D_m^f$ for every $m \geqslant m_1$. To conclude the analysis of this case we observe by (9) that

$$l_m - f(\boldsymbol{x}^*) \leqslant \min\{(1-\alpha_k)e_k\|\boldsymbol{d}_k\|_2 : k \leqslant m,\ k \text{ is an objective cut}\}$$

and so since, as for g, the function f is Lipschitz continuous on $S \cap \overline{B(\mathbf{0},r)}$ with Lipschitz constant L_f this implies by the previous inequality using $\|\boldsymbol{d}_k\|_2 \leqslant L_f$ that

$$l_m - f(\boldsymbol{x}^*) \leqslant L_f D_m^f = L_f D_m \tag{21}$$

for every $m \geqslant m_1$ and hence the result is proved for this case. To start the analysis of the other case suppose that all the optimal solutions with $\|\boldsymbol{x}^*\|_2 < r$ satisfy $g(\boldsymbol{x}^*) = 0$ and assume by contradiction that

$$\limsup_{m \uparrow \infty} \frac{l_m - f(\boldsymbol{x}^*)}{D_m} = \infty.$$

This implies that there exists an increasing sequence $\{m_n\}_{n=1}^{\infty}$ with $m_n \uparrow \infty$ and $l_{m_n} - f(\boldsymbol{x}^*) > nD_{m_n}$. By Assumption 2.3.1 it follows that $g(\hat{\boldsymbol{x}}) := \min\{g(\boldsymbol{x}): \boldsymbol{x} \in \overline{B(\mathbf{0},r)}\} = -\delta$ for some $\delta > 0$ and so $\hat{\boldsymbol{x}} \in \overline{B(\mathbf{0},r)}$ can not be optimal. Hence by Theorem 2.3.1 there exists some n_0 such that $\hat{\boldsymbol{x}} \notin S_{m_n}$ for every $n \geqslant n_0$ with the set S_{m_n} defined as in Lemma 2.3.4. Since f is continuous it follows for every $n \geqslant n_0$ that there exists some $\lambda_n \in (0,1)$ such that $\boldsymbol{x}_n := \lambda_n \hat{\boldsymbol{x}} + (1-\lambda_n)\boldsymbol{x}^* \in \overline{B(\mathbf{0},r)}$ with $f(\boldsymbol{x}_n) = f(\boldsymbol{x}^*) + \frac{1}{2} nD_{m_n}$. Moreover, by the convexity of g and the definition of $\hat{\boldsymbol{x}}$ we obtain

$$(g(\boldsymbol{x}^*) - g(\boldsymbol{x}_n))/\|\boldsymbol{x}^* - \boldsymbol{x}_n\|_2 \geqslant (g(\boldsymbol{x}^*) - g(\hat{\boldsymbol{x}}))/\|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2$$

and this yields using $g(\boldsymbol{x}^*) = 0$ and $g(\hat{\boldsymbol{x}}) = -\delta$ that

$$-g(\boldsymbol{x}_n) \geqslant \delta \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_2 / \|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2. \tag{22}$$

By the Lipschitz continuity of f on $S \cap \overline{B(\mathbf{0},r)}$ with Lipschitz constant L_f it follows that $\frac{1}{2} nD_{m_n} = f(\boldsymbol{x}_n) - f(\boldsymbol{x}^*) \leqslant L_f \|\boldsymbol{x}_n - \boldsymbol{x}^*\|_2$ and so by (22),

$$-g(\boldsymbol{x}_n) \geqslant n\delta D_{m_n} / (2\|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2 L_f). \tag{23}$$

Consider now some arbitrary $k \leqslant m_n$ and suppose at step k a constraint cut is performed. Since by (23) and the definition of $\boldsymbol{x}_n$ we obtain that $\boldsymbol{x}_n \in S_{m_n} \subseteq E(\boldsymbol{A}_k; \boldsymbol{a}_k)$, $k \leqslant m_n$, (see Lemma 2.3.4) it follows by (12) that

$$g(\boldsymbol{x}_n) \geqslant g(\boldsymbol{a}_k) + \boldsymbol{d}_k^{\mathrm{t}}(\boldsymbol{x}_n - \boldsymbol{a}_k) \geqslant g(\boldsymbol{a}_k) - \sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k}$$

and so by (13),

$$\begin{aligned}(1-\alpha_k) e_k \|\boldsymbol{d}_k\|_2 &= (1 - g(\boldsymbol{a}_k)/\sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k}) \sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k} \\ &= \sqrt{\boldsymbol{d}_k^{\mathrm{t}} \boldsymbol{A}_k \boldsymbol{d}_k} - g(\boldsymbol{a}_k) \\ &\geqslant -g(\boldsymbol{x}_n).\end{aligned}$$

This implies using $\|\boldsymbol{d}_k\|_2 \leqslant L_g$ with L_g the Lipschitz constant of g on $\overline{B(\mathbf{0},r)}$ that

$$L_g D_{m_n}^g \geqslant -g(\boldsymbol{x}_n)$$

and hence by (23),

$$L_g D_{m_n}^g \geqslant n\delta D_{m_n} / (2\|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2 L_f). \tag{24}$$

On the other hand, if at step m_n an objective cut is performed we obtain by (21) and $l_{m_n} - f(\boldsymbol{x}^*) > nD_{m_n}$ that

$$L_f D^f_{m_n} \geqslant l_{m_n} - f(\boldsymbol{x}^*) > nD_{m_n}. \tag{25}$$

Combining now (25) and (24) it follows for n sufficiently large and satisfying

$$\min\left\{\frac{n\delta}{2\|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2 L_f L_g}, \frac{n}{L_f}\right\} > 1$$

that

$$D_{m_n} = \min\{D^g_{m_n}, D^f_{m_n}\} \geqslant \min\left\{\frac{n\delta}{2\|\boldsymbol{x}^* - \hat{\boldsymbol{x}}\|_2 L_f L_g}, \frac{n}{L_f}\right\} D_{m_n} > D_{m_n}.$$

This yields a contradiction and so it must follow that

$$\limsup_{m \uparrow \infty} \frac{l_m - f(\boldsymbol{x}^*)}{D_m} < \infty$$

completing the proof. □

Note that Theorem 2.5 of [16] gives exactly the same convergence rate as Theorem 2.3.2 (designated by the author as $c_n^{1/n}$, and equal to $1/\sqrt{ab}$) but in [16] only the central cut version applied to unconstrained problems is analyzed, and the convergence proof presented there is much more complicated. This proof was extended to the constraint case by Luthi (cf. [24]) but still covering only central cuts. Besides, contrary to our elementary and more natural approach, a deep result in convex analysis about volumes of so-called concave arrays is needed in [24]. This result can only be applied if f is convex, while our approach with some obvious modifications can also be used if f is quasiconvex (cf. [12]). So, on one hand we prove similar results by easier and elementary techniques, while on the other hand we extend the above mentioned results to a deep cut version.

We also note that our Step 2 provides a simple way to guarantee the existence of suitable L_f and L_g without imposing f or g to be Lipschitz continuous on the whole space of $\mathbb{R}^n$.

A final comment concerns open feasible sets. As we will see in the next section some applications fall into this category. For such problems the condition $g(\boldsymbol{x}^*) < 0$ is naturally satisfied and the convergence of the algorithm is also proved in this case by considering only the first part of the proof of Theorem 2.3.2.

This finishes our theoretical analysis of the ellipsoid algorithm. The application discussed in the next section will provide a test problem for it. Observe that the absence of an efficient algorithm to solve this nondifferentiable location problem was the main motive to derive a deep cut version of the ellipsoid algorithm. However, in the near future we intend to test this algorithm on more general problems.

3. An application

Consider, as an example, the problem of locating an ambulance depot to handle the accidents in a given district. Whenever an accident occurs a call is generated and one of the available response or service units at this depot is assigned to it and required to travel to the scene of the accident. Clearly the assignment of a service unit to a call will result in the non-availability of this unit during some random period of time. Since the occurrence of accidents is a random process, this may lead to the non-availability of all the units at the same time and so an incoming call facing this needs to wait for service. By this example it is clear that the decision where to locate a depot should take these congestion effects into account. After specifying the queueing discipline a reasonable objective to consider in this example would be to minimize the maximum of the average lengths of time between the arrival of a call from one of the possible accident sites and the arrival of a unit at that site. For simplicity we assume that the number of accident sites is finite. This objective is clearly of the min-max type. However, before discussing a simplified and mathematical tractable version of the above example (only one unit and *First Come First Served* (FCFS) queueing discipline) we first review the existing min–max single facility location models in the plane and their solution procedures. Observe that these models do not incorporate the probabilistic nature of the arrival process of customers and thereby the possible non-availability of servers at the facility is ignored by them.

The most studied min–max type location problem in the plane is the classical weighted Euclidean 1-center problem. This problem can be stated as follows: given n demand points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ belonging to the plane, find a point $\boldsymbol{x}^{\mathrm{t}} = (x_1, x_2)$ such that the function

$$\max_{1 \leqslant i \leqslant n} \{w_i d(\boldsymbol{x}, \boldsymbol{x}_i)\}$$

is minimized, where the distance function $d(\boldsymbol{x}, \boldsymbol{x}_i)$ is given by the Euclidean norm, $\|\cdot\|_2$. It is called Rawls problem for general norms (cf. [39]).

Sylvester (cf. [37]) introduced the Euclidean version of this problem in 1857 for equal weights w_i. It is easy to see that its solution is given by the center of the smallest circle containing all the given demand points. Shamos and Hoey (cf. [30]) presented for this problem an algorithm which uses the so-called "farthest point Voronoi diagram" which can be constructed in $O(n \log n)$ time. Other solutions for this so-called unweighted case can be found in Rademacher and Toeplitz (cf. [29]), Courant and Robbins (cf. [5]), Smallwood (cf. [36]), Nair and Chandrasekeran (cf. [27]) and Elzinga and Hearn (cf. [10]). Finally, Megiddo (cf. [26]) introduced an algorithm with $O(n)$ time complexity. This algorithm is based on the analysis of linear programming problems up to 3 dimensions. Megiddo's procedure is theoretically very efficient, but it is not clear how to adapt it for arbitrary $\mathscr{L}_p$-norms. For general $\mathscr{L}_p$-norms with $1 < p < \infty$ the problem is clearly a continuous convex programming problem. It is interesting to note here that locating m centers, using the Euclidean norm, was proved by Masuyama, Ibaraki and Hasegawa (cf. [25]) to be NP-hard.

A major difficulty of the above convex objective function is its nondifferentiability in a

infinite number of points. Therefore it is not possible, at least theoretically, to apply standard techniques from nonlinear programming and so special purpose algorithms had to be developed. Unfortunately, these special purpose algorithms cannot be applied to the model derived in the next subsection. However, this model can be solved by the ellipsoid algorithm. Observe that the same algorithm can also be applied to the classical min–max problem and the min-sum version (cf. [14]) of the model to be discussed in the next subsection.

3.1. A min–max stochastic queue location model

Let $\boldsymbol{x}_i^{\mathrm{t}} = (x_{i_1}, x_{i_2})$, $i = 1, 2, \ldots, n$, denote n demand points in the plane and $\boldsymbol{x}^{\mathrm{t}} = (x_1, x_2)$ the location of a facility containing one server. Assume that each demand point $\boldsymbol{x}_i$ generates demands over time according to a Poisson process $\{\underline{\mathscr{P}}_i(t); t \geqslant 0\}$ with parameter λh_i, where $h_i > 0$, $i = 1, 2, \ldots, n$, $\sum_{i=1}^n h_i = 1$ and $\lambda > 0$. The Poisson processes $\underline{\mathscr{P}}_1(t), \underline{\mathscr{P}}_2(t), \ldots, \underline{\mathscr{P}}_n(t)$ are independent, and hence the overall demand process $\underline{\mathscr{P}}(t) := \sum_{i=1}^n \underline{\mathscr{P}}_i(t)$ is again a Poisson process with rate λ.

Regarding the example, let "server" designate the response unit at the ambulance depot, "customer" designate each accident and "arrival of a customer" designate each call generated by an accident.

The travel speed of the server is assumed to be a constant v, and the distance $d(\boldsymbol{x}, \boldsymbol{x}_i)$ between the facility at $\boldsymbol{x}$ and the demand point $\boldsymbol{x}_i$ is measured by some norm $\|\cdot\|$ so that $d(\boldsymbol{x}, \boldsymbol{x}_i) = \|\boldsymbol{x} - \boldsymbol{x}_i\|$. This implies that the service time of a customer located at demand point $\boldsymbol{x}_i$ equals $(2/v)\|\boldsymbol{x} - \boldsymbol{x}_i\|$ if it is assumed (without loss of generality) that on-scene and off-scene service times (i.e. the time spent by the server at the demand point and at the facility, cf. [2]) are equal to zero. Moreover, each time the server finishes his (or her) service at some demand point, he (or she) returns to the facility and starts serving the next client in the queue. A FCFS queueing discipline is assumed.

The following random variables are needed in order to introduce an objective function for this problem.

- $\underline{d}_l :=$ the demand point generating the lth arriving customer;
- $\underline{w}_l(\boldsymbol{x}) :=$ the time between the arrival of the lth arriving customer and the assignment of the server to this customer, if the facility is located at $\boldsymbol{x}$;
- $\underline{\tau}_l(\boldsymbol{x}) :=$ the service time of the lth arriving customer, if the facility is located at $\boldsymbol{x}$;
- $\underline{s}_l(\boldsymbol{x}) :=$ the actual waiting time of the lth arriving customer before the arrival of the server at demand point $\underline{d}_l$ to serve this customer, if the facility is located at $\boldsymbol{x}$.

It is not difficult to verify, using the independence of the Poisson arrival processes, that the random variables $\underline{d}_l$, $l \geqslant 1$, are independent and identically distributed with $P\{\underline{d}_l = i\} = h_i$, $i = 1, 2, \ldots, n$.

Moreover, conditioning on the event $\{\underline{d}_l = i\}$ it turns out that

$$\underline{\tau}_l(\boldsymbol{x}) = (2/v)\|\boldsymbol{x} - \boldsymbol{x}_i\|, \quad \text{if } \underline{d}_l = i,$$

and

$$\underline{s}_l(\boldsymbol{x}) = \underline{w}_l(\boldsymbol{x}) + \tfrac{1}{2}\underline{\tau}_l(\boldsymbol{x}).$$

In order to introduce a customer-oriented objective one possibility is to assume that an arriving customer, using the framework of utility theory, associates with his (or her) actual waiting time a certain level of dissatisfaction. This gives rise to the following assumption.

Assumption 3.1.1. If the facility is located at $\boldsymbol{x}$, and the lth arriving customer is generated by the demand point $\boldsymbol{x}_i$, then the customer's random dissatisfaction cost equals $f_i(\underline{s}_l(\boldsymbol{x}))$, where $f_i : [0,\infty) \to [0,\infty), f_i(0) = 0$, is some nondecreasing left-continuous disutility function, $1 \leqslant i \leqslant n$.

Note that there is no loss of generality to assume that all customers of a given demand point $\boldsymbol{x}_i$ share the same disutility function f_i. Indeed, if a fraction p of customers generated by the demand point $\boldsymbol{x}_i$ has different disutilities, this demand point may be divided into two separate dummy demand points with arrival rates $\lambda h_i p$ and $\lambda h_i(1-p)$ and the desired property is achieved. However, for notational convenience we assume in the remainder that the set of demand points consists of distinct points.

Classical location theory distinguishes two major objectives. One possibility is to minimize the average disutilities aggregated over all the customers (min-sum), while the other is to minimize the maximum of the average disutilities from customers located at demand point $\boldsymbol{x}_i$, $1 \leqslant i \leqslant n$, (min–max). Only the min-max objective will be discussed here. For a discussion of the min-sum type objective corresponding to the Stochastic Queue Location Problem in the plane the reader is referred to [14] and [44].

In order to introduce this min–max objective, let us define

- $\underline{i}_l :=$ the index of the lth arriving customer coming from demand point $\boldsymbol{x}_i$;
- $\underline{c}_{m,i}(\boldsymbol{x}) :=$ the total random disutility value of the first m customers from demand point $\boldsymbol{x}_i$ if the facility is located at $\boldsymbol{x}$.

Clearly

$$\underline{c}_{m,i}(\boldsymbol{x}) = \sum_{l=1}^{m} f_i(\underline{s}_{\underline{i}_l}(\boldsymbol{x})) = \sum_{l=1}^{m} f_i\left(\underline{w}_{\underline{i}_l}(\boldsymbol{x}) + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|\right),$$

which, taking expectations, yields

$$\mathscr{E}[\underline{c}_{m,i}(\boldsymbol{x})] = \sum_{l=1}^{m} \mathscr{E}\left[f_i\left(\underline{w}_{\underline{i}_l}(\boldsymbol{x}) + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|\right)\right]. \tag{26}$$

Some observations are needed in order to evaluate for every $1 \leqslant i \leqslant n$ the random variable $\underline{w}_{\underline{i}_l}(\boldsymbol{x})$. The underlying queueing model can be seen as a M/G/1 queue with FCFS queueing discipline and n different customer classes (cf. [4]), where a customer belongs to customer class i if located at demand point $\boldsymbol{x}_i$. Clearly, in this framework, $\underline{w}_l(\boldsymbol{x})$ represents the waiting time in the queue of the lth arriving customer and hence the random process $\{\underline{w}_l(\boldsymbol{x}) : l \geqslant 1\}$ is the waiting time process (in the queue) of a M/G/1 system with arrival rate $\lambda = \sum_{i=1}^{n} \lambda h_i$

and service time distribution $B(\tau)$ equal to the weighted average of the service time distributions $B_i(\tau)$ of each customer-class i, i.e.

$$B(\tau) := P\{\underline{\tau}_l(\boldsymbol{x}) \leqslant \tau\} = \sum_{i=1}^{n} h_i B_i(\tau) = \sum_{i=1}^{n} h_i 1_{\{(2/v)\|\boldsymbol{x}-\boldsymbol{x}_i\| \leqslant \tau\}},$$

where

$$1_A := \begin{cases} 1 & \text{if } A \text{ occurs,} \\ 0 & \text{otherwise.} \end{cases}$$

By the above observation and well-known results for the M/G/1 queue (cf. chapter 8 of [22]) it follows that $\underline{w}_l(\boldsymbol{x})$ converges $\uparrow$ almost surely to an almost surely finite random variable $\underline{w}_\infty(\boldsymbol{x})$ if and only if $\boldsymbol{x} \in \Omega$, where

$$\Omega := \{\boldsymbol{x} \in \mathbb{R}^2 \colon (2\lambda/v) m_1(\boldsymbol{x}) < 1\}$$

and

$$m_1(\boldsymbol{x}) := \sum_{i=1}^{n} h_i \|\boldsymbol{x} - \boldsymbol{x}_i\|$$

denotes the Weber function (cf. [40]).

Since by definition $\underline{i}_l \geqslant l$ and the random variable $\underline{w}_{\underline{i}_l}(\boldsymbol{x})$ is completely determined by the independent service times of customers arriving before customer $\underline{i}_l$ and the independent arrival times up to customer $\underline{i}_l$ we obtain as in (8.10) and (8.11) of [22] that

$$\underline{w}_l(\boldsymbol{x}) \leqslant \underline{w}_{\underline{i}_l}(\boldsymbol{x}) \leqslant \underline{w}_{\underline{i}_l+1}(\boldsymbol{x}) \leqslant \underline{w}_\infty(\boldsymbol{x}) \quad \text{a.s.} \tag{27}$$

Hence also $\underline{w}_{\underline{i}_l}(\boldsymbol{x}) \uparrow \underline{w}_\infty(\boldsymbol{x})$ almost surely if and only if $\boldsymbol{x} \in \Omega$. By the monotonicity and left-continuity of the disutility functions and (27) this implies, using the monotone convergence theorem (cf. [15]) that

$$\mathscr{E}\left[f_i\left(\underline{w}_{\underline{i}_l}(\boldsymbol{x}) + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|\right)\right] \uparrow \mathscr{E}\left[f_i\left(\underline{w}_\infty(\boldsymbol{x}) + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|\right)\right].$$

So for every $\boldsymbol{x} \in \Omega$, the average expected cost $c_{(i)}(\boldsymbol{x})$ per customer from demand point $\boldsymbol{x}_i$ exists and by (26) this equals

$$c_{(i)}(\boldsymbol{x}) = \lim_{m\to\infty} \frac{1}{m} \mathscr{E}[\underline{c}_{m,i}(\boldsymbol{x})] \uparrow \mathscr{E}\left[f_i\left(\underline{w}_\infty(\boldsymbol{x}) + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|\right)\right] \leqslant \infty.$$

Clearly, to avoid pathological cases we have to assume for a given set of disutility functions f_i, $1 \leqslant i \leqslant n$, that $\mathscr{E}[f_i(\underline{w}_\infty(\boldsymbol{x}) + (1/v)\|\boldsymbol{x}-\boldsymbol{x}_i\|)]$ is finite for every $\boldsymbol{x} \in \Omega$. Observe, since the service times $\tau_l(\boldsymbol{x})$, $l \geqslant 1$, are uniformly bounded for every $\boldsymbol{x} \in \Omega$, that this assumption holds for any increasing polynomial f_i (cf. [20]).

The above assumption now gives rise to the following proper optimization problem

$$(\mathrm{P}_0) \qquad \min_{\boldsymbol{x} \in \Omega} c_{\max}(\boldsymbol{x})$$

where

$$c_{\max}(\boldsymbol{x}) := \max_{1 \leqslant i \leqslant n} c_{(i)}(\boldsymbol{x}).$$

The next theorem mentions a general property for each function $c_{(i)}$.

Theorem 3.1.1. *The function $c_{(i)} : \Omega \to \mathbb{R}$ is convex on Ω if the corresponding disutility function f_i is nondecreasing and convex. Moreover, if f_i is only nondecreasing then $c_{(i)} : \Omega \to \mathbb{R}$ is quasiconvex on Ω.*

Proof. The proof can be found in [14] or [44], and hence it is omitted. □

Remarks. 1. The above theorem also holds if we assume that the overall demand process is a renewal process and each time a demand occurs this demand is generated by demand point $\boldsymbol{x}_i$ with probability h_i. Moreover, the trials to decide which demand point has generated the arriving demand are independent tossings. In this case the underlying queueing model is a GI/G/1 queue (cf. [22]).

2. By Theorem 3.1.1 it follows immediately that $c_{\max} : \Omega \to \mathbb{R}$ is convex on Ω whenever all disutility functions f_i, $1 \leqslant i \leqslant n$, are nondecreasing and convex. Moreover, if at least one of the disutility functions f_i is only nondecreasing we obtain that $c_{\max} : \Omega \to \mathbb{R}$ is quasiconvex on Ω.

Generally, it is not possible to evaluate $c_{(i)}(\boldsymbol{x})$ explicitly. However for polynomials and in particular the simple case of linear disutility functions, like $f_i(t) = c_i t$ for $1 \leqslant i \leqslant n$, and using a $\mathscr{L}_p$-norm it is possible to derive a closed analytical expression for $c_{(i)}(\boldsymbol{x})$ (cf. [14] or [44]). For the linear case this is given by

$$c_{(i)}(\boldsymbol{x}) := c_i \left(\frac{(2\lambda/v^2)\sum_{j=1}^n h_j \|\boldsymbol{x}-\boldsymbol{x}_j\|_p^2}{1-(2\lambda/v)\sum_{j=1}^n h_j \|\boldsymbol{x}-\boldsymbol{x}_j\|_p} + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|_p \right).$$

Since linear functions are both convex and concave the optimization problem (P_0) given by

$$\min_{\boldsymbol{x} \in \Omega} c_{\max}(\boldsymbol{x})$$

where

$$c_{\max}(\boldsymbol{x}) := \max_{1 \leqslant i \leqslant n} c_i \left(\frac{(2\lambda/v^2)\sum_{j=1}^n h_j \|\boldsymbol{x}-\boldsymbol{x}_j\|_p^2}{1-(2\lambda/v)\sum_{j=1}^n h_j \|\boldsymbol{x}-\boldsymbol{x}_j\|_p} + \frac{1}{v}\|\boldsymbol{x}-\boldsymbol{x}_i\|_p \right)$$

and

$$\Omega := \{\boldsymbol{x} \in \mathbb{R}^2 \colon (2\lambda/v) m_1(\boldsymbol{x}) < 1\}$$

is a very special case of the convex programming problem (P) with an open feasible region (see the discussion at the end of the previous section).

We note that it is possible in this case to establish conditions for a feasible $\boldsymbol{a}_m$ to be optimal. In fact, it is necessary and sufficient for optimality that $0 \in \partial c_{\max}(\boldsymbol{a}_m)$ (cf. [1]). In spite of being in general difficult to determine the subgradient set of a general convex function this can be done for this particular case.

First recall the following result due to Dubovitsky and Milyutin (cf. [6]).

Lemma 3.1.1. *If f_i, $1 \leqslant i \leqslant n$, are finite continuous convex functions on Ω and $f_{\max} := \max_{1 \leqslant i \leqslant n} f_i$ then the subgradient set of $f_{\max}$ at $\boldsymbol{x} \in \Omega$ is given by*

$$\partial f_{\max}(\boldsymbol{x}) = \operatorname{conv}\left(\bigcup_{i \in I(\boldsymbol{x})} \partial f_i(\boldsymbol{x}) \right)$$

with $I(\boldsymbol{x}) = \{i: f_i(\boldsymbol{x}) = f_{\max}(\boldsymbol{x})\}$.

Proof. The proof can be found in [6] and hence it is omitted. □

Considering each function $c_{(i)}(\boldsymbol{x})$ it is differentiable everywhere except at the demand points $\boldsymbol{x}_l$, $1 \leqslant l \leqslant n$.

Suppose that $\boldsymbol{x} = \boldsymbol{x}_l$ for a given l. Define, if $i \neq l$,

$$\bar{c}_{(i)}(\boldsymbol{x}) := c_i \left(\frac{(2\lambda/v^2) \sum_{j \neq l} h_j \|\boldsymbol{x} - \boldsymbol{x}_j\|_p^2}{1 - (2\lambda/v) \sum_{j \neq l} h_j \|\boldsymbol{x} - \boldsymbol{x}_j\|_p} + \frac{1}{v} \|\boldsymbol{x} - \boldsymbol{x}_i\|_p \right),$$

and, for $i = l$,

$$\bar{c}_{(l)}(\boldsymbol{x}) := c_l \frac{(2\lambda/v^2) \sum_{j \neq l} h_j \|\boldsymbol{x} - \boldsymbol{x}_j\|_p^2}{1 - (2\lambda/v) \sum_{j \neq l} h_j \|\boldsymbol{x} - \boldsymbol{x}_j\|_p}.$$

Note that $\bar{c}_{(i)}(\boldsymbol{x})$ and $\bar{c}_{(l)}(\boldsymbol{x})$ are differentiable in $\boldsymbol{x}_l$, and so $\nabla \bar{c}_{(i)}(\boldsymbol{x}_l)$ and $\nabla \bar{c}_{(l)}(\boldsymbol{x}_l)$ exists.

Let also, if $i \neq l$,

$$\Gamma_i := \frac{\bar{c}_{(i)}(\boldsymbol{x}_l) - (c_i/v) \|\boldsymbol{x}_l - \boldsymbol{x}_i\|_p}{1 - (2\lambda/v) \sum_{j \neq l} h_j \|\boldsymbol{x}_l - \boldsymbol{x}_j\|_p} \frac{2\lambda}{v} h_l,$$

and, if $i = l$,

$$\Gamma_l := \frac{\bar{c}_{(l)}(\boldsymbol{x}_l)}{1 - (2\lambda/v) \sum_{j \neq l} h_j \|\boldsymbol{x}_l - \boldsymbol{x}_j\|_p} \frac{2\lambda}{v} h_l + \frac{c_l}{v}.$$

The following lemma fully characterizes the subgradient set of the nondifferentiable points of $c_{(i)}(\boldsymbol{x})$.

Lemma 3.1.2. *Let $\|\cdot\|_p$ denote $\mathscr{L}_p$-norm with $p \geqslant 1$, and $\|\cdot\|_q$ denote the conjugate $\mathscr{L}_q$-norm $(1/p + 1/q = 1)$, then*

$$\partial c_{(i)}(\boldsymbol{x}_l) = \{\boldsymbol{d} \in \mathbb{R}^2 : \|\boldsymbol{d} - \nabla \bar{c}_{(i)}(\boldsymbol{x}_l)\|_q \leqslant \Gamma_i\}$$

for $1 \leqslant i \leqslant n$.

Proof. The proof can be found in [44] or [13] and hence it is omitted. □

In order to test if a point $\boldsymbol{x}$ is optimal for $c_{\max}(\boldsymbol{x})$ one need to decide if $0 \in \partial c_{\max}(\boldsymbol{x})$.

If $\boldsymbol{x}$ is not a demand point, and so $\partial c_{(i)}(\boldsymbol{x}) = \{\nabla c_{(i)}(\boldsymbol{x})\}$ for every i, the problem reduces to the decision problem whether $\mathbf{0}$ belongs to the convex hull of a set of points in the plane and hence it can be solved in linear time (cf. [11] or [26]). If $\boldsymbol{x}$ is a demand point an efficient solution procedure is presented in [11].

A final remark in this subsection concerns the existence of an initial ellipsoid.

Suppose the optimal solution of (P_0) exists, and is denoted by $\boldsymbol{x}^*$. Then $\boldsymbol{x}^*$ is a feasible solution of (P_0). This means that

$$\frac{2\lambda}{\upsilon} \sum_{j=1}^{n} h_j \|\boldsymbol{x}^* - \boldsymbol{x}_j\|_p < 1.$$

By the triangle inequality of a norm we obtain

$$\frac{2\lambda}{\upsilon} \left\| \sum_{j=1}^{n} h_j(\boldsymbol{x}^* - \boldsymbol{x}_j) \right\|_p \leqslant \frac{2\lambda}{\upsilon} \sum_{j=1}^{n} h_j \|\boldsymbol{x}^* - \boldsymbol{x}_j\|_p < 1$$

and hence

$$\left\| \boldsymbol{x}^* - \sum_{j=1}^{n} h_j \boldsymbol{x}_j \right\|_p < \frac{\upsilon}{2\lambda}.$$

Since

$$\|\boldsymbol{x}\|_p \geqslant \|\boldsymbol{x}\|_\infty = \max\{|x_1|, |x_2|\} \geqslant \frac{1}{\sqrt{2}} \sqrt{|x_1|^2 + |x_2|^2} = \frac{1}{\sqrt{2}} \|\boldsymbol{x}\|_2$$

for any $\boldsymbol{x} \in \mathbb{R}^2$ and $p \geqslant 1$ it follows that

$$\left\| \boldsymbol{x}^* - \sum_{j=1}^{n} h_j \boldsymbol{x}_j \right\|_2 \leqslant \sqrt{2} \left\| \boldsymbol{x}^* - \sum_{j=1}^{n} h_j \boldsymbol{x}_j \right\|_p < \frac{\upsilon}{\sqrt{2}\lambda}.$$

From the above inequality we obtain that the optimal point $\boldsymbol{x}^*$ must be contained in a circle with center $\sum_{j=1}^{n} h_j \boldsymbol{x}_j$ and radius $\upsilon/(\sqrt{2}\lambda)$. This circle provides an initial ellipsoid for our algorithm.

3.2. Computational results

In order to test the algorithm it was completely coded by us in Sun Pascal and no commercial routines were used except the standard functions and procedures of the language. The program includes the optimality test discussed in the previous subsection which was applied to each feasible center. The program was compiled and executed on a Sun Sparc Station SLC using the default double precision (64-bit IEEE floating point format) real numbers of the Sun Pascal language. The computational experience was carried over 600 uncorrelated instances of the problem discussed in the last subsection. Those instances were randomly

generated in the following way. We start by describing the selection of the problem parameters.

For the problems being tested, the number n of demand points belongs to $\{10, 25, 50, 100, 250, 500\}$.

The disutility function of each demand point x_i is chosen to be a linear function with coefficient $c_i = 250$, i.e. $f_i(t) = c_i t$ with $c_i = 250$ for every i.

For the $\mathscr{L}_p$-norm being used, we take $p \in \{1.1, 1.5, 2.0, 2.5, 3.0\}$, while the overall Poisson arrival rate is set to $\lambda = 0.001$ and the fraction h_i of arrivals from the demand point x_i is determined as follows. We uniformly draw numbers from the interval $[0,1)$, say $\tilde{h}_i$, $1 \leqslant i \leqslant n$, and set h_i equal to $h_i = \tilde{h}_i / (\sum_{j=1}^{n} \tilde{h}_j)$ for every $1 \leqslant i \leqslant n$.

Now we describe the procedure to generate the demand points. Al the demand points are generated within the square $[0, 250] \times [0, 250]$, for which a clustered structure is created using the following procedure. First we draw two integers m_1 and m_2 ranging from 1 to 20, and then we divide the square $[0, 250] \times [0, 250]$ into $(m_1 + 1)(m_2 + 1)$ subsquares by generating randomly m_1 x-axis coordinates and m_2 y-axis coordinates in $(0, 250)$ (cf. Figure 1). Then we label these subsquares from 1 to $(m_1 + 1)(m_2 + 1)$.

Subsequently we randomly choose according to these labels some given number of subsquares. In each chosen subsquare we uniformly draw a given number of demand points. Finally, the remaining demand points are uniformly drawn from the original square $[0, 250] \times [0, 250]$ and added to the already existing set of demand points, in a total of n points.

In order to procedure "constrained" examples we compute after the generation of each instance the value of the speed v of the server according to the following procedure. First a pair of values for v is produced with the property that for the smallest value the feasible set Ω is empty and for the biggest value the feasible set Ω includes all the demand points. Subsequently binary search is applied to the corresponding interval until a value of v is found for which during the first 10 iterations of the algorithm both constraint and objective cuts are generated.

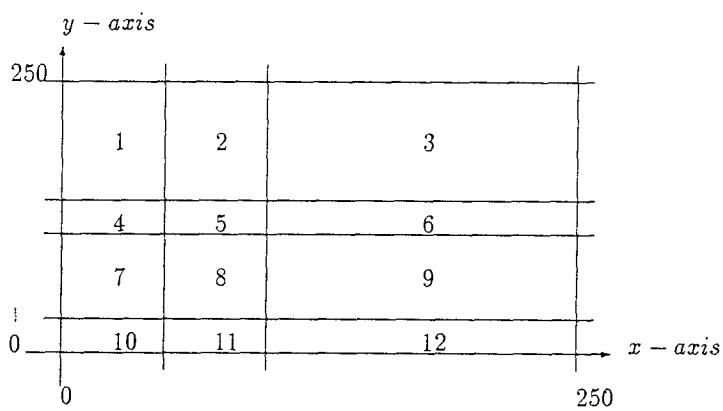

Fig. 1. Clustered problem $m_1 = 2$, $m_2 = 3$.

Finally, the tolerance parameter used in the stopping rule is $\varepsilon := 5 \times 10^{-6}$ and a relative error measurement as described in Subsection 2.2 was used.

In Table 1 we summarize the results of our computational experience.

For each pair (n, p) 20 uncorrelated instances of the problem were generated according to the procedure described above and each of them was solved by both versions of the algorithm. Hence the entries of Table 1 are averages of the corresponding values.

The columns under *deep cut statistics* include the percentage of objective cuts generated

Table 1

Results of the ellipsoid algorithm

Problem		Central cuts		Deep cuts		Deep cut statistics					% reduction	
n	P	it	time	it	time	% o	$\bar{\alpha}_o$	% c	$\bar{\alpha}_c$	$\bar{\alpha}$	it	time
10	1.1	72.2	0.29	58.6	0.24	90	0.054	10	0.089	0.058	18.5	17.7
10	1.5	67.7	0.27	56.9	0.23	87	0.054	13	0.047	0.052	15.2	13.4
10	2.0	61.4	0.11	52.2	0.10	88	0.050	12	0.056	0.050	14.8	13.8
10	2.5	65.8	0.27	53.9	0.22	89	0.056	11	0.060	0.057	16.7	17.2
10	3.0	66.7	0.26	54.5	0.22	85	0.057	15	0.042	0.054	17.2	16.0
25	1.1	75.5	0.70	58.1	0.54	90	0.068	10	0.070	0.068	22.6	22.3
25	1.5	62.2	0.57	52.2	0.49	85	0.053	15	0.056	0.053	16.1	14.0
25	2.0	62.4	0.22	53.6	0.20	85	0.052	15	0.042	0.050	13.2	9.9
25	2.5	61.8	0.56	52.9	0.48	89	0.049	11	0.059	0.049	14.3	14.7
25	3.0	62.5	0.56	52.8	0.48	84	0.051	16	0.050	0.049	15.6	14.9
50	1.1	74.5	1.31	58.8	1.04	88	0.059	12	0.081	0.061	20.5	20.2
50	1.5	61.0	1.09	50.9	0.91	87	0.055	13	0.060	0.054	16.1	16.1
50	2.0	61.7	0.42	51.5	0.35	86	0.055	14	0.060	0.055	16.6	16.9
50	2.5	59.3	1.09	48.8	0.89	89	0.055	11	0.069	0.056	17.7	18.5
50	3.0	60.5	1.10	51.6	0.95	83	0.046	17	0.055	0.047	14.6	13.0
100	1.1	70.5	2.52	56.0	2.01	89	0.064	11	0.063	0.063	20.5	20.0
100	1.5	58.5	2.06	50.9	1.80	88	0.045	12	0.058	0.046	12.7	12.3
100	2.0	61.5	0.80	51.2	0.67	85	0.051	15	0.061	0.052	16.4	15.9
100	2.5	60.1	2.08	51.4	1.78	86	0.049	14	0.065	0.050	14.2	13.9
100	3.0	58.9	2.05	51.0	1.79	87	0.053	13	0.053	0.052	13.0	12.6
250	1.1	70.0	5.99	56.6	4.86	88	0.058	12	0.064	0.057	19.1	18.8
250	1.5	60.1	5.15	50.9	4.38	86	0.055	14	0.056	0.054	15.5	15.0
250	2.0	60.7	1.95	51.7	1.66	85	0.053	15	0.051	0.052	14.9	14.8
250	2.5	59.3	5.16	49.2	4.30	88	0.052	12	0.066	0.054	16.8	16.5
250	3.0	61.2	5.28	51.8	4.48	86	0.052	14	0.072	0.053	15.4	15.0
500	1.1	67.3	11.55	54.3	9.34	86	0.063	14	0.052	0.060	19.3	19.1
500	1.5	59.8	10.27	50.3	8.71	88	0.051	12	0.060	0.052	15.7	15.1
500	2.0	62.0	3.88	51.9	3.27	87	0.051	13	0.069	0.052	16.3	15.9
500	2.5	60.9	10.36	50.8	8.67	86	0.053	14	0.070	0.055	16.7	16.3
500	3.0	59.0	10.16	50.3	8.69	88	0.052	12	0.055	0.052	14.8	14.4

by the deep cut version in column % o and the average depth of the corresponding cut in column $\bar{\alpha}_o$. Similar values concerning constraint cuts are listed in columns % c and $\bar{\alpha}_c$. The column $\bar{\alpha}$ shows the total average depth of a cut. In the generated examples no norm cuts were produced which may be explained by the rather loose determination of the starting ellipsoid and by the stability and good behavior of our test problem.

Each *time* column refers to the execution time in seconds of the Sun Station measured by the available standard *clock* function of the Sun Pascal compiler. This corresponds to the elapsed time from the start to the end of the ellipsoid procedure. During the execution of the ellipsoid procedure no input or output operations are performed. The optimality test (cf. [11]) is included in these times.

We note that the *time* values for $p=2.0$ correspond to a special situation since the computations of the Euclidean distance and the corresponding derivatives can be simplified.

Comparing the two last columns of percentage reductions one can see that the behavior of the algorithm reflects that the deep cut version does not imply any significant extra computational effort. In fact, every reduction in *it* (iterations) is followed by an approximate reduction in *time*.

As a final remark we observe that using deep cuts reduces approximately 16% on both the computational time and the number of iterations.

Previous experiences where the examples were generated in a way that most of the iterations corresponded to objective cuts, i.e. almost every center belongs to $S \cap B(\mathbf{0},r)$, show averages of 25% reduction which is confirmed in [13] where an unconstrained convex problem (the weighted $\mathscr{L}_p$ 1-center or Rawls problem) is solved by the ellipsoid algorithm.

The results obtained in [7] agree in general with our results but show a trend of instability in the deep cut version when applied to some test problems. We believe that our modified Step 2 may contribute to increase the stability of the algorithm but more extensive computational tests need to be performed.

Acknowledgment

The authors like to thank the anonymous referees for their constructive remarks which greatly improved this paper.

References

[1] M.S. Bazaraa and C.M. Shetty, *Nonlinear Programming: Theory and Algorithms* (Wiley, New York, 1979).

[2] O. Berman, R.C. Larsen and S.S. Chiu, "Optimal server location on a network operating as an M/G/1 queue," *Operations Research* 33 (1985) 746–771.

[3] R.G. Bland, D. Goldfarb and M.J. Todd, "The ellipsoid method: A survey," *Operations Research* 29 (1981) 1039–1091.

[4] O.J. Boxma, "Workloads and waiting times in single-server systems with multiple customer classes," *Queueing Systems* 5 (1989) 185–214.

[5] R. Courant and H. Robbins, *What is Mathematics?* (Oxford University Press, Oxford, 1941).

[6] A.Y. Dubovitsky and A.A. Milyutin, "Extremum problems in the presence of restriction," *USSR Computational Mathematics and Mathematical Physics* 3 (1965) 1–80.
[7] S.T. Dziuban, J.G. Ecker and M. Kupferschmid, "Using deep cuts in an ellipsoid algorithm for nonlinear programming," *Mathematical Programming Study* 25(5) (1985) 93–107.
[8] J.G. Ecker and M. Kupferschmid, "An ellipsoid algorithm for nonlinear programming," *Mathematical Programming* 27 (1983) 83–106.
[9] J.G. Ecker and M. Kupferschmid, "A computational comparison of the ellipsoid algorithm with several nonlinear programming algorithms," *SIAM Journal on Control and Optimization* 23(5) (1985) 657–674.
[10] J. Elzinga and D. Hearn, "Geometrical solutions for some minimax location problems," *Transportation Science* 6 (1972) 379–394.
[11] J.B.G. Frenk, J. Gromicho, M. Fridrich and S. Zhang, "An efficient algorithm to check whether **0** belongs to the convex hull of a finite number of $\mathscr{L}_p$-circles." Report 9204/A, Erasmus University Rotterdam (Rotterdam, 1992).
[12] J.B.G. Frenk, J. Gromicho and S. Zhang, "A deep cut ellipsoid method for quasiconvex programming," (1992), in preparation.
[13] J.B.G. Frenk, J. Gromicho and S. Zhang, "General models in min–max continuous location: Theory and solution techniques," (1993), submitted for publication.
[14] J.B.G. Frenk, M. Labbé, R. Visscher and S. Zhang, "The stochastic queue location problems in the plane," Report 8948/A, Erasmus University Rotterdam (Rotterdam, 1989).
[15] C. George. *Exercises in Integration* (Springer, New York, 1984).
[16] J.L. Goffin, "Convergence rates of the ellipsoid method on general convex functions," *Mathematics of Operations Research* 8 (1983) 135–150.
[17] J.L. Goffin, "Variable metric relaxation methods, part II: The ellipsoid method," *Mathematical Programming* 30 (1984) 147–162.
[18] M. Grötschel, L. Lovàsz and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica* 1 (1981) 169–197.
[19] M. Grötschel, L. Lovàsz and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization* (Springer, Berlin, 1988).
[20] D.P. Heyman and M.J. Sobel, *Stochastic Models in Operations Research, Vol. 1* (McGraw-Hill, New York, 1982).
[21] L.G. Khachiyan, "A polynomial algorithm in linear programming," *Doklady Akademii Nauk SSSR* 244 (1979) 1093–1979 [In Russian.]
[22] L. Kleinrock, *Queueing Systems: Theory, Vol. 1* (Wiley, New York, 1975).
[23] P. Lancaster and M. Tismenetsky, *The Theory of Matrices* (Academic Press, New York, 2nd ed., 1985).
[24] H.J. Luthi, "On the solution of variational inequalities by the ellipsoid method," *Mathematics of Operations Research* 10 (1985) 515–522.
[25] S. Masugama, T. Ibaraki and T. Hasegawa, "The computational complexity of the m-center problems on the plane," *The Transactions of the IECE of Japan* 64 (1981) 57–64.
[26] N. Megiddo, "Linear-time algorithms for linear programming in $\mathbb{R}^3$ and related problems," *SIAM Journal on Computing* 12 (1983) 759–776.
[27] K.P.K. Nair and R. Chandresakaran, "Optimal location of a single server center of certain types," *Naval Research Logistics Quarterly* 18 (1971) 503–510.
[28] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization* (Wiley, New York, 1988).
[29] H. Rademacher and O. Toeplitz, *The Enjoyment of Mathematics* (Princeton University Press, Princeton, NJ, 1957).
[30] M.I. Shamos and D. Hoey, "Closest-point problems," in: *Proceedings of the 16th Annual IEEE Symposium on Foundations of Computer Science* (1975) pp. 151–162.
[31] N.Z. Shor, "Convergence rate of the gradient descent method with dilation of the space," *Cybernetics* 6 (1970) 102–108.
[32] N.Z. Shor, "Utilization of the operation of space dilation in the minimization of convex functions," *Cybernetics* 6 (1970) 7–15.
[33] N.Z. Shor, "Cut-off method with space extension in convex programming problems," *Cybernetics* 12 (1977) 94–96.
[34] N.Z. Shor, "New development trends in nondifferentiable optimization," *Cybernetics* 13 (1977) 881–886.
[35] N.Z. Shor and V.I. Gershovich, "Family of algorithms for solving convex programming problems," *Cybernetics* 15 (1980) 502–508.

[36] R.D. Smallwood, "Minimax detection station placement," *Operations Research* 13 (1965) 636–646.
[37] J.J. Sylvester, "A question in the geometry of the situation," *Quarterly Journal of Pure and Applied Mathematics* 1 (1857) 79.
[38] D.E. Varberg and A.W. Roberts, *Convex Functions* (Academic Press, New York, 1973).
[39] J.E. Ward and R.E. Wendell, "Using blocks norms for location modelling," *Operations Research* 33 (1985) 1074–1090.
[40] A. Weber, "Uber den standort der industrien," (1909).
[41] Y. Ye, "Karmarkar's algorithm and the ellipsoid method," *Operations Research Letters* 6 (1987) 177–182.
[42] D.B. Yudin and A.S. Nemirovsky, "Evaluation of the informational complexity of mathematical programming problems," *Ekonomika i Matematicheskie Metody* 12 (1976) 128–142 [In Russian.]
[43] D.B. Yudin and A.S. Nemirovsky," Informational complexity and efficient methods for the solution of convex extremal problems," *Ekonomika i Matematicheskie Metody* 12 (1976) 357–369 [In Russian.]
[44] S. Zhang, "Stochastic queue location problems," PhD thesis, Tinbergen Institute Series 14 (Rotterdam, The Netherlands, 1991).