Simple Iterative Methods for Linear Optimization over Convex Sets

Daniel Dadush[1⋆], Christopher Hojny[2], Sophie Huiberts[1], and Stefan Weltge[3]

[1] Centrum Wiskunde & Informatica, Netherlands
{dadush,s.huiberts}@cwi.nl
[2] Eindhoven University of Technology, Netherlands
c.hojny@tue.nl
[3] Technical University of Munich, Germany
weltge@tum.de

Abstract. We give simple iterative methods for computing approximately optimal primal and dual solutions for the problem of maximizing a linear functional over a convex set K given by a separation oracle. In contrast to prior work, our algorithms directly output primal and dual solutions and avoid a common requirement of binary search on the objective value.

Under the assumption that K contains a ball of radius r and is contained inside the origin centered ball of radius R, using $O(R^4/r^2\varepsilon^2)$ iterations and calls to the oracle, our main method outputs a point $x \in K$ together with a non-negative combination of the inequalities outputted by the oracle and one inequality of the R-ball certifying that x is an additive ε-approximate solution. In the setting where the inner r-ball is centered at the origin, we give a simplified variant which outputs a multiplicative $(1+\varepsilon)$-approximate primal and dual solutions using $O(R^2/r^2\varepsilon^2)$ iterations and calls to the oracle. Similar results hold for packing type problems. Our methods are based on variants of the classical Von Neumann and Frank–Wolfe algorithms.

Our algorithms are also easy to implement, and we provide an experimental evaluation of their performance on a testbed of maximum matching and stable set instances. We further compare variations of our method to the standard cut loop implemented using Gurobi. This comparison reveals that in terms of iteration count, our methods converge on average faster than the standard cut loop on our test set.

Keywords: linear optimization · separation oracle · iterative method

1 Introduction

Let $K \subseteq \mathbb{R}^n$ be a compact convex set that contains a (Euclidean) ball of radius $r > 0$ and is contained inside the origin centered ball of radius $R > 0$. We have

⋆ This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement QIP–805241)

access to K through a separation oracle (SO), which, given a point $x \in \mathbb{R}^d$, either asserts that $x \in K$ or returns a linear constraint valid for K but violated by x. Given $c \in \mathbb{R}^n$, $\|c\| \leq 1$, we consider the problem of approximately solving

$$\max\{\langle c, x\rangle : x \in K\}. \tag{1}$$

Approximate linear optimization in the SO model is one of the fundamental settings in optimization. The model is relevant for a wide variety of *implicit* optimization problems, where an explicit description of the defining inequalities for K is either too large to store or not fully known.

The SO model was first introduced in [26] where it was shown that an additive ε-approximate solution to (1) can be obtained using $O(n \log(R/(\varepsilon r)))$ queries via the center of gravity method and $O(n^2 \log(R/(\varepsilon r)))$ queries via the ellipsoid method. This latter result was used by Khachiyan [23] to give the first polynomial time method for linear programming. The study of oracle-type models was greatly extended in the classic book [19], where many applications to combinatorial optimization were provided. Further progress on the SO model was given by Vaidya [30], who showed that the $O(n \log(R/(\varepsilon r)))$ oracle complexity can be efficiently achieved using the so-called volumetric barrier as a potential function, where the best current running time for such methods (using a different barrier) was given very recently [24,22].

While the above methods are theoretically very efficient, they have seldom been applied in practice. This is perhaps due to the complexity of the algorithms themselves, which makes designing optimized implementations quite difficult. Furthermore, low accuracy solutions are often sufficient, e.g., within 1% of optimal, and hence the overhead required to obtain $\log(1/\varepsilon)$ convergence may not be advantageous. For this purpose, a common approach in optimization is to find low-cost per iteration methods that achieve convergence rates that are polynomial in $1/\varepsilon$. While many such low-cost methods have been developed for related optimization models, comparatively little work in this regard has been done for optimization in the SO model. This paper aims to develop robust, simple to implement, low-cost per iteration methods for optimization in the SO model. Before detailing our contributions, we review the relevant work in related models.

To begin, there has been a tremendous amount of work in the context of first-order methods [3,1], where the goal is to minimize a possibly complicated function, given by a gradient oracle, over a *simple domain* K (e.g., the simplex, cube, ℓ_2 ball). For these methods, often variants of (sub-)gradient descent, it is generally assumed that computing (Euclidean) projections onto K as well as linear optimization over K, i.e., problem (1), is easy. If one only assumes access to a linear optimization (LO) oracle on K, K can become more interesting (e.g., the shortest-path or spanning-tree polytope). In this "projection-free" context, one of the most popular methods is the so-called Frank–Wolfe (FW) algorithm [16] (see [21] for a modern treatment), which iteratively computes a convex combination of vertices of K to obtain an approximate minimizer of a smooth convex function.

In the context of combinatorial optimization, there has been a considerable line of work on solving (implicit) packing and covering problems using the so-called multiplicative weights update (MWU) framework [29,27,18]. In this framework, one must be able to implement an MWU oracle, which in essence computes optimal solutions for the target problem after the "difficult" constraints have been aggregated according to the current weights. This framework has been applied for getting fast $(1 \pm \varepsilon)$-approximate solutions to multicommodity flow [29,18], packing spanning trees [8], the Held–Karp approximation for TSP [7], and more, where the MWU oracle computes shortest paths, minimum cost spanning trees, minimum cuts respectively in a sequence of weighted graphs. The MWU oracle is in general just a special type of LO oracle. This oracle can often in fact be interpreted as a SO oracle which returns a maximally violated constraint. While certainly related to the SO model, it is not entirely clear how to adapt MWU to work with a vanilla SO, in particular in settings unrelated to packing and covering.

A final line of work, which directly inspires our results, has examined simple iterative methods for computing a point in the interior of a cone Σ that directly apply in the SO model. The application of simple iterative methods for solving conic feasibility problems can be traced to Von Neumann in 1948 (see [13]), and a variant of this method, the perceptron algorithm [28] is still very popular today. Von Neumann's algorithm computes a convex combination of the defining inequalities of the cone, scaled to be of unit length, of nearly minimal Euclidean norm. The separation oracle is called to find an inequality violated by the current convex combination, and this inequality is then used to make current convex combination shorter (in an analogous way to FW). This method is guaranteed to find a point in the cone in $O(1/\rho^2)$ iterations, where ρ is the so-called width of Σ (the radius of the largest ball contained in Σ centered at a point of norm 1). Starting in 2004, polynomial time variants of this and related methods (i.e., achieving $\log 1/\rho$ dependence) have been found [4,14,9], which iteratively "rescale" the norm to speed up the convergence. These rescaled variants can also be applied in the oracle setting [2,10,12] with appropriate adaptations. One shortcoming of the conic approaches is that they are currently not well-adapted for solving linear optimization problems. At present, the main way to apply these techniques is to reduce optimization to feasibility via binary search, which is highly undesirable in practice. An additional drawback of binary search is that it makes it difficult to naturally compute both primal and dual solutions, which are often available in the MWU approaches (when they are applicable).

Our Contributions We give simple iterative methods for computing approximately optimal primal and dual solutions for (1). After $O({R^4}/{r^2\varepsilon^2})$ iterations our main method outputs a point $x^* \in K$ together with an explicit non-negative combination of the inequalities outputted by the oracle and a single inequality valid for the R-ball, resulting in an inequality of the form $\langle c, x\rangle \leq \langle c, x^*\rangle + \varepsilon$. Thus, the primal-dual pair yields a proof for

$$\langle c, x^*\rangle \leq \mathrm{OPT} \leq \langle c, x^*\rangle + \varepsilon$$

that can be immediately verified, where $\mathrm{OPT} = \max\{\langle c, x\rangle : x \in K\}$. In the case where the inner r-ball is centered at the origin, which we call the *polar setting*, we give a simplified method which outputs a primal-dual pair certifying

$$\langle c, x^*\rangle \leq \mathrm{OPT} \leq (1+\varepsilon)\langle c, x^*\rangle$$

after $O(R^2/r^2\varepsilon^2)$ iterations. An analogous result is obtained for packing problems where $K \subseteq \mathbb{R}^n_+$ is down-closed ($0 \leq x \leq y, y \in K \Rightarrow x \in K$), $c \in \mathbb{R}^n_+$, where the requirement $r\mathbb{B}_2 \subseteq K$ is replaced by $r\mathbb{B}_2 \cap \mathbb{R}^n_+ \subseteq K$. In their basic variants, all algorithms are very easy to implement. In particular, every iteration corresponds to a low-cost Frank–Wolfe style update and requires one call to the oracle.

One notable benefit of our algorithms is that they can be implemented in a mostly parameter-free way. Firstly, the desired accuracy ε need not be specified in advance, and second, the parameters r, R are only used in very weak ways for initialization purposes, which can often be avoided entirely. This property turns out to be very helpful for our experimental implementation.

Beyond the basic algorithms above, we also provide more sophisticated practical variants and demonstrate their effectiveness compared to a standard LP cut-loop in computational experiments. We now give a very brief idea of our basic techniques, how they relate to prior work, and then mention some highlights from our experimental evaluation.

Techniques Let us give an outline of our algorithms. In every iteration, we will maintain a pair (γ_t, q_t) consisting of a scalar $\gamma_t \in \mathbb{R}$ and a vector q_t encoding an inequality that is valid for K. We aim to prove that $\langle c, x\rangle \leq \gamma_t$ is valid for K as well, by iteratively moving q_t "closer" to $\langle c, x\rangle \leq \gamma_t$ while keeping it valid for K. We refer to $\langle c, x\rangle \leq \gamma_t$ as the *target inequality*. The specifics on what "close to" means will differ depending on the setting we are in.

It is helpful to think of γ_t as the value of the best incumbent solution. If q_t were to certify that $\langle c, x\rangle \leq \gamma_t$ is valid for K, then we would know that the current solution is optimal. The key insight is that if q_t is close to $\langle c, x\rangle \leq \gamma_t$, then q_t together with $K \subseteq R\mathbb{B}_2$ will certify that $\langle c, x\rangle \leq \gamma_t + \varepsilon$, i.e., that the current solution is close to optimal.

In every iteration of our algorithms, we use q_t to obtain a candidate feasible solution x_t. If the oracle asserts that $x_t \in K$, then by construction x_t will have value $\gamma_{t+1} := \langle c, x_t\rangle > \gamma_t$. Otherwise, the oracle will return a constraint $\langle a, x\rangle \leq b$ that is valid for K but violated by x_t. We compute a non-negative linear combination q_{t+1} of q_t and the returned constraint $\langle a, x\rangle \leq b$. These latter updates are heavily inspired by the Frank–Wolfe algorithm.

No matter which case occurs, the distance between q_{t+1} and $\langle c, x\rangle \leq \gamma_{t+1}$ will be significantly smaller than the distance between q_t and $\langle c, x\rangle \leq \gamma_t$. We will initialize q_1 to encode the trivially valid inequality $\langle 0, x\rangle \leq 1$ and initialize γ_1 to any value less than the upper bound we aim to certify for $\max\{\langle c, x\rangle : x \in K\}$.

In terms of prior work, our algorithms are most directly inspired by the previously discussed conic feasibility algorithms. Here we extend them to work naturally in the optimization context, where we now support both primal and dual updates. During primal updates, we update our lower bound on the objective value, and during dual updates, we update our current convex combination to get closer to the objective. We note that we are not the first to try and apply the above ideas in the context of optimization. As far as we are aware, the first such works are due to Boyd [5,6], who was motivated to find an alternative to the standard cut-loop for integer programming. Boyd's algorithm is for the polar setting only and uses binary search to guess the optimal value of the objective. It is also not designed to output approximate primal-dual certificates as we achieve, though his method does approximately certify the optimal value. The basic ideas however are similar, and so one can view our algorithms as both improvements and generalizations of Boyd's method.

Experimental Findings We have implemented different variants of our algorithm in `Python` and compare them with a standard cutting plane procedure. Experiments for matching and stable set problems show that the value of the LP given by the constraints separated up to iteration t drops much faster for our (enhanced) methods than it does for the standard cut loop. If we initialize γ_1 with the optimum value of (1), then even the most basic version of our algorithm outperforms the standard cut loop. Our `Python` implementation is available on GitHub.

Organization We start by fixing some notation and providing auxiliary lemmas in Section 2. In Section 3 we present our simpler method for the case where the inner r-ball is centered at the origin. We also show how the method can be adapted to packing problems. Our main algorithm for the general case is treated in Section 4. Both sections come with an analysis of the respective methods. In Section 5 we discuss variants of our methods that are based on more sophisticated implementations. We demonstrate their performance in computational experiments given in Section 6.

2 Preliminaries

In what follows, we denote by $\|\cdot\|$ the Euclidean norm, by dist the distance with respect to this norm, and by $\mathbb{B}_2$ the Euclidean unit ball (whose dimension will be clear from the context). For two vectors $x, y \in \mathbb{R}^n$ we denote by $[x, y] \coloneqq \operatorname{conv}(\{x, y\})$, the convex hull of x and y. For a finite set S, we let $\Delta^S \coloneqq \{x \in \mathbb{R}^S_+ : \langle \mathbf{1}, x\rangle = 1\}$ be the standard simplex on vectors whose coordinates are indexed by the elements in S.

The following two lemmas will be used in later parts.

Lemma 1. *Let $x, y, z \in \rho\mathbb{B}_2 \subseteq \mathbb{R}^n$ satisfy $\langle x - y, x - z\rangle \geq \varepsilon\|x - z\|^2$, where $\rho \geq 0$, $\varepsilon > 0$. Then we have* $\operatorname{dist}([x, y], z)^2 \leq \left(1 - \frac{\varepsilon^2}{4\rho^2}\|x - z\|^2\right)\|x - z\|^2$.

Proof of Lemma 1. Note that $x = y$ would imply $x = z$, in which case the claim is trivial. Thus, we may assume $x \neq y$ and consider $\lambda \coloneqq \frac{\langle x-y, x-z\rangle}{\|x-y\|^2} \geq 0$. If $\lambda \leq 1$, then $x + \lambda(y - x) \in [x, y]$ and hence

$$\begin{aligned}\operatorname{dist}([x,y],z)^2 \le \|x + \lambda(y-x) - z\|^2 &= \|x-z\|^2 - \frac{\langle x-y, x-z\rangle^2}{\|x-y\|^2} \\ &\le \|x-z\|^2 - \frac{\varepsilon^2\|x-z\|^4}{\|x-y\|^2} \\ &= \left(1 - \tfrac{\varepsilon^2}{\|x-y\|^2}\|x-z\|^2\right)\|x-z\|^2 \\ &\le \left(1 - \tfrac{\varepsilon^2}{4\rho^2}\|x-z\|^2\right)\|x-z\|^2.\end{aligned}$$

In the case $\lambda \ge 1$, we have $\varepsilon \le 1$ since

$$\begin{aligned}\langle x-y, x-z\rangle \le \|x-y\|\|x-z\| &\le \frac{\|x-y\|^2 + \|x-z\|^2}{2} \\ &\le \frac{\langle x-y, x-z\rangle + \|x-z\|^2}{2}.\end{aligned}$$

The inequalities follow respectively by Cauchy–Schwarz, AM–GM and $\lambda \ge 1$. Now $\langle x-y, x-z\rangle \le \|x-z\|^2$ immediately follows. Using this fact, we obtain

$$\begin{aligned}\operatorname{dist}([x,y],z)^2 \le \|y-z\|^2 &= \|(x-y)-(x-z)\|^2 \\ &= \|x-y\|^2 - 2\langle x-y, x-z\rangle + \|x-z\|^2 \\ &\le -\langle x-y, x-z\rangle + \|x-z\|^2 \\ &\le (1-\varepsilon)\|x-z\|^2 \\ &\le \left(1-\varepsilon^2\right)\|x-z\|^2 \\ &\le \left(1 - \tfrac{\varepsilon^2}{4\rho^2}\|x-z\|^2\right)\|x-z\|^2. \qquad \square\end{aligned}$$

Lemma 2. *Let* $d_1, \dots, d_t > 0$ *and* $\eta \ge 0$ *such that* $d_{i+1} \le (1 - \eta d_i)d_i$ *holds for* $i = 1, \dots, t-1$. *Then we have* $\frac{1}{d_t} \ge \frac{1}{d_1} + (t-1)\eta$.

Proof. The claim follows from observing that for each $i \in [t-1]$ we have

$$\frac{1}{d_{i+1}} \ge \frac{1}{(1-\eta d_i)d_i} \ge \frac{1-(\eta d_i)^2}{(1-\eta d_i)d_i} = \frac{1+\eta d_i}{d_i} = \frac{1}{d_i} + \eta. \qquad \square$$

3 Convex sets containing the origin

We first consider the setting where the inner r-ball is centered at the origin, i.e., $r\mathbb{B}_2 \subseteq K \subseteq R\mathbb{B}_2$, since it is conceptually simpler. Note that, up to rescaling, every linear inequality that is valid for K can be written as $\langle a, x\rangle \le 1$ for some $a \in \mathbb{R}^n$. Let $\mathcal{A} \subseteq \mathbb{R}^n$ be the set of all such a (i.e., $\mathcal{A}$ is the polar body of K). Whenever we query the separation oracle with a point $x \in \mathbb{R}^n \setminus K$, we assume that we will receive some $a \in \mathcal{A}$ with $\langle a, x\rangle > 1$. Let us consider Algorithm 1, which receives some $\gamma_1 > 0$ as an input, and which we initialize with $q_1 = \mathbf{0} \in \mathcal{A}$.

Algorithm 1 for convex sets containing the origin in the interior

Input: $c \in \mathbb{R}^n \setminus \mathbf{0}$, $\gamma_1 > 0$

1: $f_1 \leftarrow \frac{1}{\gamma_1} c$, $q_1 \leftarrow \mathbf{0}$, $A_1 \leftarrow \{\mathbf{0}\}$
2: **for** $t = 1, 2, \dots$ **do**
3:     **if** $\langle f_t - q_t, f_t + q_t \rangle \leq 0$ **then**
4:         $\gamma_{t+1} \leftarrow \gamma_t$, $f_{t+1} \leftarrow f_t$, $A_{t+1} \leftarrow A_t$
5:         $q_{t+1} \leftarrow \arg\min\{\|f_{t+1} - q\| : q \in [\mathbf{0}, q_t]\}$
6:     **else**
7:         $x_t \leftarrow \frac{2}{\langle f_t - q_t, f_t + q_t \rangle}(f_t - q_t)$
8:         **if** $x_t \in K$ **then**
9:             $\gamma_{t+1} \leftarrow \langle c, x_t \rangle$, $f_{t+1} \leftarrow \frac{1}{\gamma_{t+1}} c$, $A_{t+1} \leftarrow A_t$
10:             $q_{t+1} \leftarrow \arg\min\{\|f_{t+1} - q\| : q \in [\mathbf{0}, q_t]\}$
11:         **else**
12:             get $a_t \in \mathcal{A}$ with $\langle a_t, x_t \rangle > 1$
13:             $\gamma_{t+1} \leftarrow \gamma_t$, $f_{t+1} \leftarrow f_t$, $A_{t+1} \leftarrow A_t \cup \{a_t\}$
14:             $q_{t+1} \leftarrow \arg\min\{\|f_{t+1} - q\| : q \in [a_t, q_t]\}$
15:         **end if**
16:     **end if**
17:     maintain $q_{t+1} = \sum_{a \in A_{t+1}} \mu_a^{t+1} a$ where $\mu^{t+1} \in \Delta^{A_{t+1}}$
18: **end for**

In every iteration, $A_t \setminus \{\mathbf{0}\}$ will consist of all vectors (constraints) returned by the separation oracle until this point. Each target inequality $\langle c, x \rangle \leq \gamma_t$ is represented by the vector $f_t = {}^1\!/\!_{\gamma_t} c$. Note that, whenever f_{t+1} has been computed, q_{t+1} is the projection of f_{t+1} onto the line segment between q_t and a vector in A_t. This means that q_{t+1} can be easily computed and is well-defined. We also see that $q_t \in \operatorname{conv}(A_t)$, and hence Line 17 has a straightforward implementation. To see that the overall algorithm is well-defined, note that whenever x_t is computed in Line 7, the denominator $\langle f_t - q_t, f_t + q_t \rangle$ is positive due to Line 3. If the algorithm determines that $x_t \in K$ holds, then we have

$$\frac{\gamma_{t+1}}{\gamma_t} = \frac{\langle c, x_t \rangle}{\gamma_t} = \langle f_t, x_t \rangle = \frac{2\langle f_t, f_t - q_t \rangle}{\langle f_t - q_t, f_t + q_t \rangle} = 1 + \frac{\|f_t - q_t\|^2}{\langle f_t - q_t, f_t + q_t \rangle} \geq 1. \quad (2)$$

This implies that $\gamma_1 \leq \gamma_2 \leq \dots$, and hence each f_t is well-defined.

In the remainder of this section, we prove the following result.

Theorem 1. *Suppose $\gamma_1 \geq \frac{r}{2}\|c\|$. Running Algorithm 1 for t iterations, we obtain a point in K with objective value γ_t such that*

$$\gamma_t \leq \mathrm{OPT} \leq \left(1 + \frac{8R}{r\sqrt{t}}\right)\gamma_t.$$

Expressing q_t as a convex combination of vectors in A_t together with a single linear inequality valid for $R\mathbb{B}_2$ yields an explicit proof of the latter inequality.

Let us remark that a point in K with objective value at least $\frac{r}{2}\|c\|$ can be easily found: If r is known to us, we may simply choose ${}^r\!/\!_{\|c\|} \cdot c \in K$. Otherwise, we may test whether the points $x_i = \frac{R}{2^i\|c\|}c$, $i = 1, 2, \dots$ are contained in K. The first point x_i for which this holds will satisfy $\langle c, x_i\rangle \geq \frac{r}{2}\|c\|$. This requires at most $\log_2(R/r)$ calls to the oracle (and does not require knowledge of r).

The proof of Theorem 1 is based on the following observation. Recall that in each iteration we maintain $q_t = \sum_{a\in A_t} \mu_a^t a$ as a convex combination of the constraints in A_t. As $a \in A_t \subseteq \mathcal{A}$ was outputted by the oracle, we know that $\langle a, x\rangle \leq 1$ holds for all $x \in K$. Moreover, we know that the inequality $\langle f_t - q_t, x\rangle \leq \|f_t - q_t\|R$ is also valid for K since $K \subseteq R\mathbb{B}_2$. Thus, we see that the inequality

$$\langle c, x\rangle = \gamma_t\langle f_t, x\rangle = \gamma_t\langle q_t, x\rangle + \gamma_t\langle f_t - q_t, x\rangle \leq \gamma_t(1 + \|f_t - q_t\|R) \tag{3}$$

is valid for K. In what follows, we will provide a bound on $\|f_t - q_t\|$. More specifically, Theorem 1 is implied by the following result.

Lemma 3. *In each iteration we have $\|f_t - q_t\| \leq \frac{4\rho}{\sqrt{t}}$, where $\rho := \max\{1/r, \|c\|/\gamma_1\}$.*

Note that $\gamma_1 \geq \frac{r}{2}\|c\|$ implies $\rho = 2/r$, and hence the bound in Theorem 1 follows from (3). We claim that Lemma 3 is implied by our next key lemma.

Lemma 4. *We always have $\|f_{t+1} - q_{t+1}\|^2 \leq \left(1 - \frac{1}{16\rho^2}\|f_t - q_t\|^2\right)\|f_t - q_t\|^2$.*

To see this, let $d_i := \|f_i - q_i\|^2$ for $i = 1, \dots, t$. By Lemma 4 we see that $d_{i+1} \leq (1 - \eta d_i)d_i$ holds for $i = 1, \dots, t-1$, where $\eta = 1/16\rho^2$. If $d_t = 0$, Lemma 3 follows immediately. Otherwise, by Lemma 2 we obtain $1/d_t \geq 1/d_1 + (t-1)\eta$. Note that $1/d_1 = (\gamma_1/\|c\|)^2 \geq \eta$, and hence we have $\frac{1}{d_t} \geq t\eta$, which yields Lemma 3.

Proof of Lemma 4. First, observe that we have $\|f_t\| \leq \|f_1\| \leq \rho$. Second, recall that q_t is a convex combination of the vectors in A_t. The norm of every vector in A_t is bounded by ρ since $1/\rho\mathbb{B}_2 \subseteq K$. Thus, we see that also $\|q_t\| \leq \rho$ holds. We will use both inequalities throughout the proof.

Suppose first that $\langle f_t - q_t, f_t + q_t\rangle \leq 0$, in which case we have

$$\langle q_t, q_t - f_t\rangle \geq \langle q_t, q_t - f_t\rangle + \tfrac{1}{2}\langle f_t - q_t, f_t + q_t\rangle = \tfrac{1}{2}\|q_t - f_t\|^2.$$

Note that $\|f_{t+1} - q_{t+1}\|^2 = \mathrm{dist}([q_t, \mathbf{0}], f_t)^2$, and hence the claim follows from Lemma 1 with $x = q_t$, $y = \mathbf{0}$, $z = f_t$, and $\varepsilon = 1/2$.

Otherwise, the algorithm checks whether $x_t \in K$. If $x_t \in K$ holds, then by (2) we have $\frac{\gamma_t}{\gamma_{t+1}} \in (0, 1]$. This means that $\frac{\gamma_t}{\gamma_{t+1}}q_t \in [\mathbf{0}, q_t]$ and hence

$$\|f_{t+1} - q_{t+1}\|^2 \leq \|f_{t+1} - \tfrac{\gamma_t}{\gamma_{t+1}}q_t\|^2 = \left(\tfrac{\gamma_t}{\gamma_{t+1}}\right)^2\|f_t - q_t\|^2.$$

By (2) we also have $0 < \frac{\gamma_t}{\gamma_{t+1}} = 1 - \frac{\|f_t - q_t\|^2}{2\langle f_t, f_t - q_t\rangle} \leq 1 - \frac{\|f_t - q_t\|}{2\|f_t\|}$ and hence

$$\left(\tfrac{\gamma_t}{\gamma_{t+1}}\right)^2 \leq \left(1 - \tfrac{\|f_t - q_t\|}{2\|f_t\|}\right)^2 \leq 1 - \tfrac{\|f_t - q_t\|^2}{4\|f_t\|^2} \leq 1 - \tfrac{1}{4\rho^2}\|f_t - q_t\|^2.$$

It remains to consider the case $x_t \notin K$, in which we have $\langle a_t, x_t\rangle > 1$. By the definition of x_t the latter is equivalent to $\langle q_t - a_t, q_t - f_t\rangle > \frac{1}{2}\|f_t - q_t\|$. The claim follows again from Lemma 1 with $x = q_t$, $y = a_t$, $z = f_t = f_{t+1}$, and $\varepsilon = 1/2$. $\square$

Packing problems It turns out that our previous method and analysis can be also applied to convex sets of the form $K = \{x \in \mathbb{R}^n_+ : \langle a, x\rangle \leq 1 \text{ for all } a \in \mathcal{A}\}$ and non-negative objectives $c \in \mathbb{R}^n_+$ with only a minor modification. Clearly, in this case K does not contain the origin-centered r-ball. Note that this assumption was only used in Lemma 4, which is based on the fact that all vectors returned by the oracle have bounded norm. Let us suppose that $r\mathbb{B}_2 \cap \mathbb{R}^n_+ \subseteq K \subseteq R\mathbb{B}_2$. Note that if we query the oracle with non-negative points only, then it will always return vectors from $\mathcal{A}$, whose norm is at most $1/r$. We can assume that each x_t is non-negative by the following additional step: Right after Line 16 we may replace q_{t+1} by the component-wise minimum of f_{t+1} and q_{t+1}, without increasing the norm of q_{t+1} or $f_{t+1} - q_{t+1}$. The resulting vector $f_{t+1} - q_{t+1}$ is non-negative, and so x_{t+1} will be as well. Note that with this modification we may not be able to maintain q_t as a convex combination of vectors in A_t, but we can maintain a point $q'_t \in \operatorname{conv}(A_t)$ such that $q_t \leq q'_t$, which yields a dual certificate in a similar fashion.

4 General convex sets

Let us now turn to the case where the inner r-ball is not centered at the origin, i.e., there exists a (possibly unknown) point $z \in \mathbb{R}^n$ such that $z + r\mathbb{B}_2 \subseteq K \subseteq R\mathbb{B}_2$. We assume that $\mathcal{A}$ consists of pairs $(a, b) \in \mathbb{R}^n \times \mathbb{R}$ such that $\langle a, x\rangle \leq b$ is valid for K. If we query the oracle with a point $y \in \mathbb{R}^n \setminus K$, we will receive some $(a, b) \in \mathcal{A}$ with $\langle a, y\rangle > b$. Without loss of generality, we may assume that $\|a\| = 1$ holds for all $(a, b) \in \mathcal{A}$. We will also require $\|c\| = 1$.

Again, we assume that R is known to us. We will use the norm $\|(a, b)\|_R \coloneqq \sqrt{\|a\|^2 + b^2/R^2}$ and the inner product $\langle (a, b), (a', b')\rangle_R \coloneqq \langle a, a'\rangle + bb'/R$ where $a, a' \in \mathbb{R}^n$ and $b, b' \in \mathbb{R}$. Note that the norm $\|\cdot\|_R$ is not induced by the inner product $\langle\cdot,\cdot\rangle_R$. The applicable Cauchy–Schwarz inequality for this inner product is $\langle f, g\rangle_R \leq \|f\|_R\|g\|$.

Consider Algorithm 2, which receives c and R as an input. In every iteration, the set $A_t \setminus (\mathbf{0}, R)$ consists of those vectors (a, b) returned by the oracle until this point. Let us verify that the algorithm is well-defined. Note that x_t is only used when $\alpha_t > 0$ due to Line 4, so x_t is well-defined whenever it is used. We have $\gamma_{t+1} \geq \gamma_t$ in every iteration due to Line 7, so $\beta_t \geq 1$. In Lines 6, 9, 14, and 18, we define p_{t+1} by finding the minimum-norm point in a line segment. Since the $\|\cdot\|_R$ is strongly convex, p_{t+1} is uniquely defined and can be easily computed. Note that p_{t+1} always remains a convex combination of $-f_{t+1}$ and elements of A_{t+1}, so Line 20 has a clear implementation.

Similar to Theorem 1, we obtain the following result.

Theorem 2. *Running Algorithm 2 for $t \geq 4096R^2/r^2$ iterations, we obtain a point in K with objective value γ_t such that*

$$\gamma_t \leq \mathrm{OPT} \leq \gamma_t + \tfrac{128R^2}{r\sqrt{t}}.$$

Expressing q_t as a convex combination of vectors in A_t together with a single linear inequality valid for $R\mathbb{B}_2$ yields an explicit proof of the latter inequality.

Algorithm 2 for general convex sets

Input: $c \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, $R > 0$

$\gamma_1 \leftarrow -R$, $f_1 \leftarrow (c, \gamma_1)$, $p_1 \leftarrow q_1 \leftarrow (\mathbf{0}, R)$, $A_1 \leftarrow \{(\mathbf{0}, R)\}$
for $t = 1, 2, \ldots$ **do**
  $(-\alpha_t x_t, \alpha_t R) \leftarrow p_t$
  if $\alpha_t \leq 0$ **then**
    $\gamma_{t+1} \leftarrow \gamma_t$, $f_{t+1} \leftarrow f_t$, $A_{t+1} \leftarrow A_t$
    $p_{t+1} \leftarrow \arg\min\{\|p\|_R : p \in [p_t, (\mathbf{0}, R)]\}$
  else if $\langle c, x_t \rangle \leq \gamma_t$ **then**
    $\gamma_{t+1} \leftarrow \gamma_t$, $f_{t+1} \leftarrow f_t$, $A_{t+1} \leftarrow A_t$
    $p_{t+1} \leftarrow \arg\min\{\|p\|_R : p \in [p_t, -f_t]\}$
  else if $x_t \in K$ **then**
    $\gamma_{t+1} \leftarrow \langle c, x_t \rangle$, $f_{t+1} \leftarrow (c, \gamma_{t+1})$, $A_{t+1} \leftarrow A_t$
    $\beta_t \leftarrow 1 + \lambda_t(\gamma_{t+1} - \gamma_t)$
    $p'_{t+1} \leftarrow \frac{1-\lambda_t}{\beta_t} q_t + \frac{\beta_t - 1}{\beta_t}(\mathbf{0}, R) - \frac{\lambda_t}{\beta_t} f_{t+1}$
    $p_{t+1} \leftarrow \arg\min\{\|p\|_R : p \in [p'_{t+1}, -f_{t+1}]\}$
  else
    get $(a_t, b_t) \in \mathcal{A}$ with $\langle a_t, x_t \rangle > b_t$
    $\gamma_{t+1} \leftarrow \gamma_t$, $f_{t+1} \leftarrow f_t$, $A_{t+1} \leftarrow A_t \cup \{(a_t, b_t)\}$
    $p_{t+1} \leftarrow \arg\min\{\|p\|_R : p \in [p_t, (a_t, b_t)]\}$
  end if
  maintain $p_{t+1} = (1 - \lambda_{t+1}) q_{t+1} - \lambda_{t+1} f_{t+1}$ where
    $q_{t+1} = \sum_{a \in A_{t+1}} \mu_a^{t+1} a$, $\mu^{t+1} \in \Delta^{A_{t+1}}$, $\lambda_{t+1} \in [0, 1]$
end for

The proof is again based on the fact that we can bound the gap in terms of the norm of the maintained points.

Lemma 5. *If* $\|p_t\|_R \leq \frac{r}{8R}$ *in Algorithm 2, then* $\mathrm{OPT} \leq \gamma_t + \frac{8R(\mathrm{OPT} - \langle c, z \rangle)}{r} \|p_t\|_R$.

Proof. Let $x \in K$ be arbitrary. Recall that $p_t = (1 - \lambda_t) q_t - \lambda_t f_t$. Assuming that $\lambda_t > 0$, we have

$$\begin{aligned} \langle c, x \rangle - \gamma_t = \langle f_t, (x, -R) \rangle_R &= \langle \tfrac{1-\lambda_t}{\lambda_t} q_t, (x, -R) \rangle_R + \langle f_t - \tfrac{1-\lambda_t}{\lambda_t} q_t, (x, -R) \rangle_R \\ &\leq \langle f_t - \tfrac{1-\lambda_t}{\lambda_t} q_t, (x, -R) \rangle_R \leq \frac{1}{\lambda_t} \|p_t\|_R \|(x, -R)\| \leq \frac{2R}{\lambda_t} \|p_t\|_R. \end{aligned} \tag{4}$$

Here, the inequalities arise from q_t being a non-negative combination of valid constraints, the Cauchy–Schwarz inequality, and $\|(x, -R)\| \leq 2R$, respectively. The claim follows immediately if $\lambda_t \geq 1/2$.

Suppose that $\lambda_t < 1/2$. The current vector q_t is a convex combination of constraint vectors $(a, b) \in \mathcal{A}$ and $(\mathbf{0}, R)$. For both, we have

$$\begin{aligned} \langle (a, b), (-z, R) \rangle_R &= b - \langle a, z \rangle = b - \langle a, z - \tfrac{a}{\|a\|} r \rangle + r\|a\| \geq r \\ \langle (\mathbf{0}, R), (-z, R) \rangle_R &= R \geq r, \end{aligned}$$

and so $\langle q_t, (-z, R)\rangle_R \geq r$. Moreover, we have

$$r/2 \leq (1-\lambda_t)r \leq \langle (1-\lambda_t)q_t, (-z, R)\rangle_R = \langle p_t, (-z, R)\rangle_R + \lambda_t \langle f_t, (-z, R)\rangle_R \\ \leq 2R\|p_t\|_R + \lambda_t \langle f_t, (-z, R)\rangle_R.$$

Using $2R\|p_t\|_R \leq r/4$ we obtain

$$\lambda_t \geq \frac{r}{4\langle f_t, (-z, R)\rangle_R} = \frac{r}{4(\gamma_t - \langle c, z\rangle)} \geq \frac{r}{4(\mathrm{OPT} - \langle c, z\rangle)},$$

and the claim follows again from (4). □

It remains to show that $\|p_t\|_R$ decreases at certain rate. Theorem 2 will follow from the next claim.

Lemma 6. *In each iteration we have* $\|p_{t+1}\|_R^2 \leq (1 - \frac{1}{8}\|p_t\|_R^2)\|\|p_t\|_R^2$.

Proof. Consider the map $\phi : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ given by $\phi((a, b)) = (a, b/R)$. Then clearly $\|(a, b)\|_R = \|\phi(a, b)\|$.

In each of the Lines 6, 9, and 18, we minimize $\|p\|_R = \|\phi(p)\|$ over $p \in [p_t, v]$ for some vector $v \in \mathbb{R}^n$ with $\langle \phi(v) - \phi(p_t), \phi(p_t)\rangle \geq \|\phi(p_t)\|^2$. Moreover, we have $\|\phi(v)\| \leq \sqrt{2}$ in every iteration. Using Lemma 1 with $x = \phi(p_t)$, $y = \phi(v)$, $z = \mathbf{0}$, $\varepsilon = 1$, and $\rho = \sqrt{2}$ the result follows for these cases.

The remaining case to consider is Line 14. Here we can calculate directly that $\langle \phi(p'_{t+1}), \phi(f_{t+1})\rangle = 0$ holds. Applying Lemma 1 for $x = \phi(p_{t+1})$, $y = \phi(p'_{t+1})$, $z = \mathbf{0}$, $\varepsilon = 1$, and $\rho = \sqrt{2}$, we get $\|\phi(p_{t+1})\|^2 \leq (1 - \frac{1}{8}\|\phi(p'_{t+1})\|^2)\|\|\phi(p'_{t+1})\|^2$. Moreover, note that we have $p'_{t+1} = p_t/\beta_t$. As we saw at the start of this section, we have $\beta_t \geq 1$ and hence $\|\phi(p'_{t+1})\| \leq \|\phi(p_t)\| \leq \|\phi(p_1)\| = 1$. This implies

$$(1 - \tfrac{1}{8}\|\phi(p'_{t+1})\|^2)\|\|\phi(p'_{t+1})\|^2 \leq (1 - \tfrac{1}{8}\|\phi(p_t)\|^2)\|\|\phi(p_t)\|^2.$$ □

Proof of Theorem 2. From $\|p_1\|_R = 1$ and Lemmas 2 and 6, we know that in iteration t we have $\|p_t\|_R \leq \frac{8}{\sqrt{t}}$. By the lower bound on t, this implies $\|p_t\|_R \leq \frac{r}{8R}$ and hence Lemma 5 yields $\mathrm{OPT} \leq \gamma_t + \frac{8R(\mathrm{OPT} - \langle c, z\rangle)}{r}\|p_t\|_R$. On the one hand, we obtain $\mathrm{OPT} < \gamma_t + \frac{128R^2}{r\sqrt{t}}$. The proof of Lemma 5 explains how q_t can be used to derive an explicit proof of this bound. On the other hand, we see that $\mathrm{OPT} < \gamma_t + (\mathrm{OPT} + R)$ holds, or, equivalently, $\gamma_t > -R = \gamma_1$. Thus, γ_t is the objective value of a feasible solution encountered during the algorithm. □

5 More sophisticated updates

Both Algorithm 1 and 2 are quite flexible in terms of how q_{t+1} is computed. Regarding Algorithm 1, the guarantee of Theorem 1 is preserved for any choice of $q_{t+1} \in \mathrm{conv}(A_{t+1})$ with $\|f_{t+1} - q_{t+1}\| \leq \min\{\|f_{t+1} - q\| : q \in [q_t, a_t]\}$. We describe a couple of possible alternative ways of computing q_{t+1}. Using these in some iterations or every iteration can significantly reduce the necessary number of iterations in experiments although we have no theoretical analysis proving so.

Fully corrective One possible update is to use a 'fully corrective' choice for q_{t+1}. This is similar to the algorithms of Boyd [5,6], and involves choosing $q_{t+1} = \arg\min\{\|f_{t+1} - q\| : q \in \mathrm{conv}(A_{t+1})\}$ for convex sets containing the origin in their interior, or $q_{t+1} = \arg\min\{\|f_{t+1} - q\| : q \in \mathrm{conv}(A_{t+1}) - \mathbb{R}^n_{\geq 0}\}$ when solving problems with non-negativity constraints. Computing q_{t+1} in this way requires solving a convex quadratic program. Although this is significantly more expensive than projecting f_{t+1} onto the line segment $[q_t, a_t]$, it enables the algorithm to make much better use of its known constraints and greatly improves the number of iterations necessary to attain a small gap. We show this experimentally in Section 6.

Partially corrective In case the full-size quadratic program is too expensive to solve, we can restrict the cost by considering a subset $S \subseteq A_{t+1}$ with $a_t \in S$ and $q_t \in \mathrm{conv}(S)$ and computing $q_{t+1} = \arg\min\{\|f_{t+1} - q\| : q \in \mathrm{conv}(S)\}$.

Sparsifying the convex combination In Algorithm 1 we maintain $q_t = \sum_{a \in A_t} \mu^t_a a$ where $\mu^t \in \Delta^{A_t}$. For numerical reasons, as well as for performing partially corrective updates as mentioned in the previous paragraph, it can be helpful if μ^t has small support.

To accomplish this, we can use Caratheodory's Theorem and solve a small number of linear systems to find a convex combination $q_t = \sum_{a \in A_t} \bar{\mu}^t_a a$ with $\bar{\mu}^t \in \Delta^{A_t}$ having at most $n+1$ non-zero coordinates. Alternatively, we can simultaneously try to greedily reduce the distance to f_t by solving the linear program

$$\max\{\lambda \ : \ q_t + \lambda(f_t - q_t) \in \mathrm{conv}(A_t)\}.$$

Since $\lambda = 0$ is known to be feasible, the optimal value λ^* allows us to compute $(1-\lambda^*)q_t + \lambda^* f_t \in \mathrm{conv}(A_t)$ having $\|((1-\lambda^*)q_t + \lambda^* f_t) - f_t\| \leq \|q_t - f_t\|$. Finding a basic solution to the linear program will result in a convex combination $\bar{\mu}^t \in \Delta^{A_t}$ with at most n non-zero coordinates such that $(1-\lambda^*)q_t + \lambda^* f_t = \sum_{a \in A_t} \bar{\mu}^t_a a$.

Non-negativity constraints Instead of computing $q_{t+1} = \arg\min\{\|f_{t+1} - q\| : q \in [q_t, v]\}$ for some v and then replacing q_{t+1} by the coordinate-wise minimum $\min(f_{t+1}, q_{t+1})$, we can instead obtain q_{t+1} by solving the following small quadratic program

$$\min\left\{\|f_{t+1} - p\| \ : \ p \leq q, \ q \in [q_t, v]\right\} \tag{5}$$

Instead of $\|f_{t+1} - q_{t+1}\|$, this step aims to decrease $\|f_{t+1} - \min(q_{t+1}, f_{t+1})\|$, thereby making use of the non-negativity constraints as well as possible. Note that all $p \in [\min(f_t, q_t), v]$ are feasible for the above program, meaning that the decrease in distance from this update is at least as good as from the procedure described at the end of Section 3.

Solving (5) merely requires computing $\arg\min\{\mathrm{dist}(f_t, q - \mathbb{R}^n_+) : q \in [v, q_t]\}$, which is trivial if $v = \mathbf{0}$ and $q_t \geq \mathbf{0}$, and otherwise requires minimizing a convex function over a line segment. The objective function $\mathrm{dist}(f_t, q - \mathbb{R}^n_+)$ is a piecewise-quadratic function with at most $n+1$ pieces, so it can be easily minimized.

Table 1: Comparison of iteration counts for different frequencies of fully corrective step.

initialization:	standard	optimal
maximum matching:		
frequency 0	169.25	86.75
frequency 1	45.50	30.12
frequency 10	51.88	32.25
LP	92.62	
maximum stable set:		
frequency 0	62.75	55.88
frequency 1	36.31	35.19
frequency 10	39.75	36.69
LP	106.06	

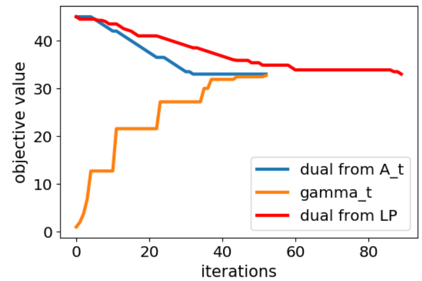

Fig. 1: Typical primal/dual bounds for matching with frequency 10.

6 Computational experiments

This section aims to investigate the performance of variants of Algorithm 1 in comparison with a classical oracle-based cutting plane procedure. Our testset consists of packing problems, for which we implemented the variant of Algorithm 1 described at the end of Section 3 and a cutting plane procedure in `Python`. The reference cutting plane algorithm is initialized with an LP relaxation of the packing problem and in each iteration an oracle is queried to separate an optimal solution of the relaxation. If a cutting plane is found, the relaxation is updated and the process is iterated until no further separating inequality is found or the iteration limit is hit. Besides the basic version of Algorithm 1, we also implemented the possibility to call a fully corrective step (as explained in Section 5) at the end of each k-th iteration. To allow for a more realistic setup, it is also possible to specify certain inequalities that are known to the user, which will be added to the set A_0 and used to initialize the LP relaxation of the cutting plane method. The implementation, a detailed description of how to reproduce our results including generation of instances, instructions on how our code can be adapted to further problem classes, and plots showing the results for all tested instances is available at GitHub[4]; the numerical experiments are based on the version with git hash `b426c53a`.

The oracles queried in our implementation are mixed-integer programming (MIP) models that search for a maximally violated inequality for the underlying problems. To solve the MIP oracles and the LP relaxations of the cutting plane method, our implementation allows calling `SCIP 7.0` [17] via its `Python` interface [25] or `Gurobi` [20]. The numerical results presented below are based on experiments using the MIP solver `Gurobi 9.0` and `Python 3.8.5`. Note that MIP solvers do not necessarily solve the MIP models and relaxations exactly. For this reason, we have introduced a tolerance that we use for comparisons and to decide whether a violated cut exists. Due to the numerical inaccuracies, the

[4] https://github.com/christopherhojny/supplement_simple-iterative-methods-linopt-convex-sets

point q_t computed in our algorithm is thus also just an approximation of a true convex combination.

We tested our implementation on instances of the maximum cardinality matching and maximum stable set problem. A maximum cardinality matching in an undirected graph $G = (V, E)$ can be computed by solving the linear program $\max\{\sum_{e \in E} x_e \;:\; x \in [0,1]^E$ and satisfies 6a and 6b$\}$, where

$$\sum\nolimits_{e \in \delta(v)} x_e \leq 1, \qquad \text{for all } v \in V, \tag{6a}$$

$$\sum\nolimits_{e \in E[U]} x_e \leq \tfrac{|U|-1}{2}, \qquad \text{for all } U \subseteq V \text{ with } |U| \text{ odd}, \tag{6b}$$

with $\delta(v)$ being the set of all edges incident with v and $E[U]$ being the edges in E with both endpoints in U, see Edmonds [15]. Our separation oracle models the separation problem for (6b) as a MIP, and separates (6a) trivially. If a violated inequality exists, the oracle outputs one with the biggest absolute violation.

Unlike for the matching problem, no linear programming model is known to find a maximum stable set in arbitrary graphs $G = (V, E)$. Instead, we used our algorithm to find an optimal solution of the clique relaxation. If $\mathcal{Q}(G)$ denotes the set of all node sets of cliques in G, the *clique relaxation* is given by $\max\{\sum_{v \in V} x_v \;:\; \sum_{v \in Q} x_v \leq 1 \text{ for all } Q \in \mathcal{Q}(G),\; x \in [0,1]^V\}$. We refer to the non-trivial constraints as clique constraints and, if the clique is an edge, as edge constraints. Our oracle for separating clique constraints is again based on a MIP.

Our test set for the matching problem consists of 16 random instances with 500 nodes, generated as follows. For each $r \in \{30, 33, \ldots, 75\}$ we build an instance by sampling r triples of nodes $\{u, v, w\}$ and adding the edges of the induced triangles to the graph. We believe that these instances are interesting because the r triangles give rise to many constraints of type (6b). For the stable set problem, we used the 16 graphs from the Color02 symposium [11] with 100–150 nodes except for DSJC125.9 since the LP cutting loop could not solve this instance within our iteration limit. Moreover, we conducted experiments with different initializations of γ_1. Note that if the feasible regions of our problems are full-dimensional, they contain a standard simplex and thus the packing problem contains a ball of radius $r = 1/\sqrt{d}$ centered at the origin if d is the number of variables. In the *standard initialization* we thus use $\gamma_1 = r\|c\|$; in the *optimal initialization*, we initialize γ_1 with the value of an optimal solution to evaluate the impact of knowing good incumbent solutions in our algorithm.

All methods terminate when the following criterion is met. In each iteration, we compute the optimal value of the LP relaxation defined by the initial and all separated constraints. If this value is at most 1% above the true optimum value, we stop. Moreover, a limit of 1000 iterations is used for all experiments. We stress that the number of iterations in the cutting plane method is not the number of simplex iterations, but the number of separated inequalities.

In our first experiment, we investigate the impact of the frequency k of the fully corrective step and the initialization of γ_1. Table 1 compares the iteration count of the standard cut loop with our algorithm for frequency $k \in \{0, 1, 10\}$, where 0 disables fully corrective steps. Column "standard" ("optimal") reports

Table 2: Comparison of iteration counts for different initial constraints in fully corrective algorithm.

initial constr.	standard	optimal	LP
maximum matching:			
upper bound	73.00	69.75	—
basic	45.50	30.12	92.62
maximum stable set:			
upper bound	60.75	61.62	225.00
basic	36.31	35.19	106.06

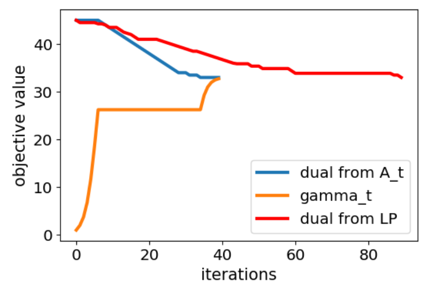

Fig. 2: Typical primal/dual bounds for matching with frequency 1.

on the standard (optimal) initialization as discussed above. In these experiments, we initialized the LP relaxation and the constraints for the fully corrective step with upper bound constraints, (6a) for matching and edge constraints for stable set problems, respectively.

We see that our algorithm in the standard initialization finds a good approximation of the optimal dual value much faster than the cutting plane method, except for matching instances with $k = 0$. With the optimal initialization, our methods perform even better (and we improve over the cutting plane method on both problem types even for $k = 0$). An explanation for the latter effect is that our algorithm does not only perform dual iterations but also primal iterations to improve γ_t. If we initialize γ_1 optimally, no primal steps are necessary, which allows faster dual progress. This hypothesis is supported by Figures 1 and 2, which show the typical changes in primal and dual values for a matching instance for different k. Interestingly, Table 1 also shows that it is not necessary to perform a fully corrective step in every iteration.

Our second experiment investigates the impact of the selected initial constraints used for the LP relaxation and the fully corrective step. We either add upper bound constraints or basic constraints (upper bound constraints plus (6a) or edge constraints). In our algorithm, we used a fully corrective frequency of $k = 1$. Table 2 summarizes our experiments. We do not report on the iteration count for the cutting plane method for matching, because it could not solve any instance within the iteration limit; in the remaining experiments, all instances could be solved. The results show that knowing good initial constraints reduces the number of iterations in both algorithms. However, we also see that the performance of our algorithm is less dependent on the initial constraints than the cutting plane method.

References

1. Beck, A.: First-Order Methods in Optimization. Society for Industrial and Applied Mathematics (Oct 2017). https://doi.org/10.1137/1.9781611974997, https://doi.

org/10.1137/1.9781611974997
2. Belloni, A., Freund, R.M., Vempala, S.: An efficient rescaled perceptron algorithm for conic systems. Mathematics of Operations Research **34**(3), 621–641 (2009)
3. Ben-Tal, A., Nemirovski, A.: Lectures on Modern Convex Optimization. Society for Industrial and Applied Mathematics (Jan 2001). https://doi.org/10.1137/1.9780898718829, https://doi.org/10.1137/1.9780898718829
4. Betke, U.: Relaxation, new combinatorial and polynomial algorithms for the linear feasibility problem. Discrete & Computational Geometry **32**(3) (May 2004). https://doi.org/10.1007/s00454-004-2878-4, https://doi.org/10.1007/s00454-004-2878-4
5. Boyd, E.A.: On the complexity of a cutting plane algorithm for solving combinatorial linear programs. SIAM Journal on Discrete Mathematics **9**(3), 365–376 (Aug 1996). https://doi.org/10.1137/s089548019325900x, https://doi.org/10.1137/s089548019325900x
6. Boyd, E.: A fully polynomial epsilon approximation cutting plane algorithm for solving combinatorial linear programs containing a sufficiently large ball. Operations Research Letters **20**(2), 59–63 (Feb 1997). https://doi.org/10.1016/s0167-6377(96)00048-x, https://doi.org/10.1016/s0167-6377(96)00048-x
7. Chekuri, C., Quanrud, K.: Approximating the held-karp bound for metric TSP in nearly-linear time. In: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS). IEEE (Oct 2017). https://doi.org/10.1109/focs.2017.78, https://doi.org/10.1109/focs.2017.78
8. Chekuri, C., Quanrud, K.: Near-linear time approximation schemes for some implicit fractional packing problems. In: Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics (Jan 2017). https://doi.org/10.1137/1.9781611974782.51, https://doi.org/10.1137/1.9781611974782.51
9. Chubanov, S.: A strongly polynomial algorithm for linear systems having a binary solution. Mathematical Programming **134**(2), 533–570 (Feb 2011). https://doi.org/10.1007/s10107-011-0445-3, https://doi.org/10.1007/s10107-011-0445-3
10. Chubanov, S.: A polynomial algorithm for linear feasibility problems given by separation oracles. Optimization Online, Jan (2017)
11. Color02 - computational symposium: Graph coloring and its generalizations. available at (2002), http://mat.gsia.cmu.edu/COLOR02
12. Dadush, D., Végh, L.A., Zambelli, G.: Rescaling algorithms for linear conic feasibility. Mathematics of Operations Research **45**(2), 732–754 (May 2020). https://doi.org/10.1287/moor.2019.1011, https://doi.org/10.1287/moor.2019.1011
13. Dantzig, G.B.: Converting a converging algorithm into a polynomially bounded algorithm. Tech. rep., Technical report, Stanford University, 1992. 5.6, 6.1, 6.5 (1991)
14. Dunagan, J., Vempala, S.: A simple polynomial-time rescaling algorithm for solving linear programs. Mathematical Programming **114**(1), 101–114 (Feb 2007). https://doi.org/10.1007/s10107-007-0095-7, https://doi.org/10.1007/s10107-007-0095-7
15. Edmonds, J.: Maximum matching and a polyhedron with 0,1-vertices. Journal of Research of the National Bureau of Standards **69B**(1–2), 125–130 (1964)
16. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval research logistics quarterly **3**(1-2), 95–110 (1956)

17. Gamrath, G., Anderson, D., Bestuzheva, K., Chen, W.K., Eifler, L., Gasse, M., Gemander, P., Gleixner, A., Gottwald, L., Halbig, K., Hendel, G., Hojny, C., Koch, T., Le Bodic, P., Maher, S.J., Matter, F., Miltenberger, M., Mühmer, E., Müller, B., Pfetsch, M.E., Schlösser, F., Serrano, F., Shinano, Y., Tawfik, C., Vigerske, S., Wegscheider, F., Weninger, D., Witzig, J.: The SCIP Optimization Suite 7.0. Technical report, Optimization Online (March 2020), http://www.optimization-online.org/DB_HTML/2020/03/7705.html
18. Garg, N., Könemann, J.: Faster and simpler algorithms for multicommodity flow and other fractional packing problems. SIAM Journal on Computing **37**(2), 630–652 (Jan 2007). https://doi.org/10.1137/s0097539704446232, https://doi.org/10.1137/s0097539704446232
19. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization, vol. 2. Springer-Verlag (1988). https://doi.org/10.1007/978-3-642-78240-4
20. Gurobi Optimizer. https://www.gurobi.com/
21. Jaggi, M.: Revisiting Frank-Wolfe: Projection-free sparse convex optimization. Proceedings of Machine Learning Research, vol. 28, pp. 427–435. PMLR, Atlanta, Georgia, USA (17–19 Jun 2013), http://proceedings.mlr.press/v28/jaggi13.html
22. Jiang, H., Lee, Y.T., Song, Z., Wong, S.C.w.: An improved cutting plane method for convex optimization, convex-concave games, and its applications. In: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing. p. 944–953. STOC 2020, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3357713.3384284, https://doi.org/10.1145/3357713.3384284
23. Khachiyan, L.G.: A polynomial algorithm in linear programming (in russian). Doklady Akademiia Nauk SSSR 224 **224**, 1093–1096 (1979), english Translation: Soviet Mathematics Doklady 20, 191-194.
24. Lee, Y.T., Sidford, A., Wong, S.C.: A faster cutting plane method and its implications for combinatorial and convex optimization. In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science. pp. 1049–1065 (2015). https://doi.org/10.1109/FOCS.2015.68
25. Maher, S., Miltenberger, M., Pedroso, J.P., Rehfeldt, D., Schwarz, R., Serrano, F.: PySCIPOpt: Mathematical programming in python with the SCIP optimization suite. In: Mathematical Software – ICMS 2016, pp. 301–307. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-42432-3_37
26. Nemirovsky, A., Yudin, D.: Informational complexity and efficient methods for solution of convex extremal problems. Ékonomika i Mathematicheskie Metody **12** (1983)
27. Plotkin, S.A., Shmoys, D.B., Tardos, É.: Fast approximation algorithms for fractional packing and covering problems. Mathematics of Operations Research **20**(2), 257–301 (May 1995). https://doi.org/10.1287/moor.20.2.257, https://doi.org/10.1287/moor.20.2.257
28. Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review **65**(6), 386–408 (1958). https://doi.org/10.1037/h0042519, https://doi.org/10.1037/h0042519
29. Shahrokhi, F., Matula, D.W.: The maximum concurrent flow problem. J. ACM **37**(2), 318–334 (Apr 1990). https://doi.org/10.1145/77600.77620, http://doi.acm.org/10.1145/77600.77620
30. Vaidya, P.M.: A new algorithm for minimizing convex functions over convex sets. Mathematical Programming **73**(3), 291–341 (Jun 1996). https://doi.org/10.1007/bf02592216, https://doi.org/10.1007/bf02592216