The Random-Facet Simplex Algorithm on Combinatorial Cubes *

Bernd Gärtner, ETH Zürich, Switzerland

January 11, 2002

Abstract

The RANDOM-FACET algorithm is a randomized variant of the simplex method which is known to solve any linear program with n variables and m constraints using an expected number of pivot steps which is *subexponential* in both n and m. This is the theoretically fastest simplex algorithm known to date if $m \approx n$; it provably beats most of the classical deterministic variants which require $\exp(\Omega(n))$ pivot steps in the worst case. RANDOM-FACET has independently been discovered and analyzed ten years ago by Kalai as a variant of the primal simplex method, and by Matoušek, Sharir and Welzl in a dual form.

The essential ideas and results connected to RANDOM-FACET can be presented in a particularly simple and instructive way for the case of linear programs over *combinatorial n-cubes*. I derive an explicit upper bound of

$$\sum_{\ell=1}^{n} \frac{1}{\ell!}\binom{n}{\ell} \leq \exp(2\sqrt{n}) - 1$$

on the expected number of pivot steps in this case, using a new technique of "fingerprinting" pivot steps. This bound also holds for *generalized* linear programs, similar flavors of which have been introduced and studied by several researchers.

I then review an interesting class of generalized linear programs, due to Matoušek, showing that RANDOM-FACET may indeed require an expected number of $\exp(\Omega(\sqrt{n}))$ pivot steps in the worst case.

The main new result of the paper is a proof that all actual linear programs in Matoušek's class are solved by RANDOM-FACET with an expected polynomial number of $O(n^2)$ pivot steps. This proof exploits a combinatorial property of linear programming which has only recently been discovered by Holt and Klee. The result establishes the first scenario in which an algorithm that works for generalized linear programs "recognizes" proper linear programs. Thus, despite Matoušek's worst-case result, the question remains open whether RANDOM-FACET (or any other simplex variant) is a polynomial-time algorithm for linear programming.

Finally, I briefly discuss extensions of the combinatorial cube results to the general case.

*This work was supported by a grant from the Swiss Science Foundation (SNF), Project No. 21-50647.97. Parts of it appeared in the Proceedings of the RANDOM'98 workshop [7].

1 Introduction

This introductory section prepares the ground for the results derived in the following three technical sections. It introduces and motivates the essential terminology and concludes with a formal description of the RANDOM-FACET algorithm for (generalized) linear programs over combinatorial cubes.

Combinatorial Cubes. Consider a linear inequality system $Ax \leq b$ over n variables and $2n$ inequalities (i.e. A is a $(2n \times n)$-matrix, b a $2n$-vector and x an n-vector of variables). The polyhedron

$$\mathcal{P} = \{x \mid Ax \leq b\}$$

is called a *combinatorial n-cube* (or simply combinatorial cube) if it is combinatorially equivalent to the unit n-cube $C^n := [0,1]^n$. In particular, $\mathcal{P}$ is a polytope with 2^n vertices. In general, two polytopes are called combinatorially equivalent if they have isomorphic face lattices [23]. Figure 1 depicts a combinatorial cube in $\mathbb{R}^3$. The generating system of six inequalities is

$$\begin{array}{rcccl} 0 & \leq & x_1 & \leq & 1, \\ x_1/3 & \leq & x_2 & \leq & 1 - x_1/3, \\ x_2/3 & \leq & x_3 & \leq & 1 - x_2/3. \end{array}$$

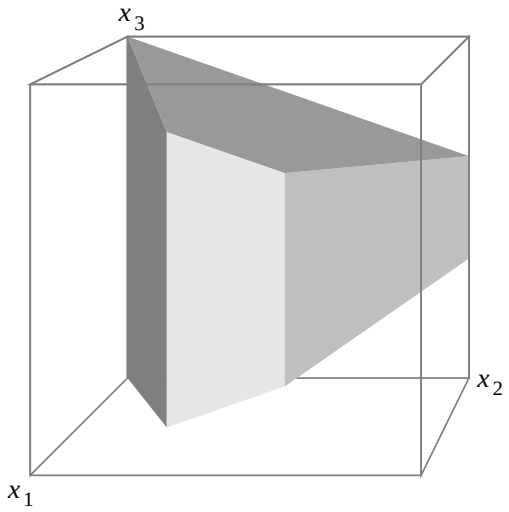

Figure 1: Combinatorial cube

Linear programs and the simplex method. A *linear program* is the problem of minimizing a linear function in n variables, subject to a set of m linear inequality constraints, written as

$$\begin{array}{lll} \text{(LP)} & \text{minimize} & c^T x \\ & \text{subject to} & Ax \leq b, \end{array} \tag{1}$$

where c is an n-vector (the *objective function*), A an $(m \times n)$-matrix, b an m-vector and x an n-vector of variables. A feasible solution to (1) is any vector $\tilde{x}$ such that $A\tilde{x} \leq b$. A feasible solution $\tilde{x}$ is optimal if

$$c^T\tilde{x} = \min\{c^T x \mid Ax \leq b\},$$

provided the minimum exists. If the *feasible region* $\mathcal{P} = \{x \mid Ax \leq b\}$ is bounded and nonempty (i.e. $\mathcal{P}$ is a polytope), then for all c, there is a vertex of $\mathcal{P}$ which is optimal. Let us assume in the sequel that $\mathcal{P}$ is a polytope; this certainly holds if $\mathcal{P}$ is a combinatorial cube, and it can be assumed in the general case without loss of generality.

The *simplex method* is a class of algorithms which compute a sequence of vertices $v_0, v_1, \ldots, \ldots, v_t, t \geq 0$ such that v_0 is some vertex, v_t is an optimal vertex, and v_i is adjacent to v_{i-1} in $\mathcal{P}$. Moreover, $c^T v_i < c^T v_{i-1}$ holds for all $i \in \{1, \ldots, t\}$.

It is a theorem that such a sequence always exists. If $\mathcal{P}$ is *simple* (i.e. all vertices have n incident edges), the sequence can be obtained in a greedy fashion; the number of arithmetic operations required to perform a *pivot step* (computation of the respective next vertex in the sequence) is polynomial in n and m. Combinatorial cubes are simple polytopes, and even in general, simplicity can be achieved by means of symbolic perturbation; v_0 is found by solving an auxiliary linear program whose initial vertex is readily available. For an in-depth treatment of linear programming and the simplex method, I recommend Chvátal's book [3].

In order to specialize the simplex method to a concrete algorithm, one has to provide a *pivot rule* according to which the next vertex is chosen in case there is more than one option during a pivot step. The pivot rule crucially determines the efficiency of the algorithm (measured in terms of the number t of pivot steps required to solve the problem). As an example, consider the linear program

$$\begin{array}{lrcccl} \text{minimize} & x_3 & & & & \\ \text{subject to} & 0 & \leq & x_1 & \leq & 1, \\ & x_1/3 & \leq & x_2 & \leq & 1 - x_1/3, \\ & x_2/3 & \leq & x_3 & \leq & 1 - x_2/3, \end{array} \tag{2}$$

whose feasible region is the combinatorial cube of Figure 1. Starting with the vertex $v^{(0)} = (0, 0, 1)^T$, the pivot rule that locally maximizes the progress in every pivot step (Local-Max) will find the optimal vertex $(0, 0, 0)^T$ in one step (Figure 2 left), while the one that locally minimizes the progress (Local-Min) has to go through all vertices in turn (Figure 2, right).

The example can be generalized to the n-variable linear program

$$\begin{array}{lrcccl} \text{minimize} & x_n & & & & \\ \text{subject to} & 0 & \leq & x_1 & \leq & 1, \\ & x_{i-1}/3 & \leq & x_i & \leq & 1 - x_{i-1}/3, \quad i = 2, \ldots, n. \end{array} \tag{3}$$

The feasible region is a combinatorial n-cube. Starting from $(0, \ldots, 0, 1)^T$, Local-Max results in one pivot step, while Local-Min gives rise to the worst possible number of $2^n - 1$ pivot steps.

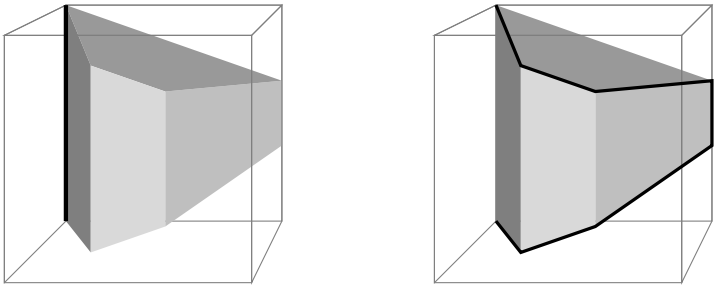

Figure 2: Pivot rules: Local-Max and Local-Min

It might seem that this worst-case behavior is simply a consequence of using a very bad pivot rule, and that it can easily be avoided in general by applying more sensible rules, like Local-Max. However, this intuition is wrong. Specifically for Local-Max, there exists a linear program with n variables and $3n-1$ inequality constraints such that Local-Max leads to an exponential number of at least

$$3 \cdot 2^{\lfloor n/2 \rfloor} - 2$$

pivot steps for a suitable initial vertex $v^{(0)}$ [14, 1]. Klee and Minty [18] were the first to show that a suitable rescaling of the linear program (3) provides a worst-case instance (meaning $2^n - 1$ pivot steps) for Dantzig's Rule, the pivot rule originally proposed by Dantzig, the inventor of the simplex method [4]. Therefore, we will refer to the feasible region of (3) as the n-dimensional *Klee-Minty cube*.

Klee and Minty's construction started a chain of results that established similar instances (many of them combinatorial cubes) for a variety of other pivot rules,[1] see [1] for a recent unified view of all the known constructions. The major open problem is to find a pivot rule (or disprove its existence) whose worst-case behavior is polynomial in n and m, the number of variables and constraints of the linear program.

We are far from this goal now; even in the special case of linear programs over combinatorial n-cubes, no deterministic pivot rule is known to have runtime better than $\exp(\Theta(n))$ in the worst case.

Randomization. The fact that deterministic strategies (in our case pivot rules) are susceptible to attacks by a malicious adversary is a common phenomenon. For example, the variant of *Quicksort* which always chooses the first element as the dividing element takes quadratic time for an already sorted input sequence. This calls for the use of *randomization*. In case of *Quicksort*, choosing the dividing element at random yields an expected $O(n \log n)$ time algorithm for a length-n sequence. The performance may still be quadratic if the random choices turn out to be poor, but the *expected* runtime of $O(n \log n)$ holds for *all* inputs.

[1] a notable exception is Zadeh's rule [22] whose worst-case performance is still unknown.

Below we show that there is a randomized pivot rule, RANDOM-FACET, whose expected number of pivot steps is bounded by

$$\sum_{\ell=1}^{n} \frac{1}{\ell!}\binom{n}{\ell} \leq e^{2\sqrt{n}} - 1$$

for any linear program over a combinatorial n-cube. We say that RANDOM-FACET has *subexponential* expected worst-case performance.

Polytope Orientations. RANDOM-FACET (like other randomized pivot rules to be mentioned later) refrains from using any metric properties of the linear program. Instead, it only looks at the relative order of adjacent vertices induced by the objective function c. If the feasible region of (1) is the polytope $\mathcal{P}$, this information can be recorded in form of a *polytope orientation*, a digraph whose underlying undirected graph is isomorphic to the vertex-edge graph of $\mathcal{P}$. Figure 3 shows the cube orientation determined by the linear program (2). We will apply the term Klee-Minty cube also to this orientation and its n-dimensional counterpart induced by the linear program (3).

This abstraction leads to coordinate-free *combinatorial models* of linear programs. I will describe and analyze RANDOM-FACET exclusively in terms of such models. In order to do this, it is important (and very instructive) to understand how the 'geometric' properties of linear programming and their underlying polytopes manifest themselves in the combinatorial models.

Definition 1.1 *A digraph $G = (V, E)$ is called* linearly inducible *if there exists a polytope $\mathcal{P}$ and a linear objective function c, generic with respect to $\mathcal{P}$,*[2] *such that G is isomorphic to the graph $G' = (V', E')$, where V' is the set of vertices of $\mathcal{P}$ and*

$$E' := \{(v, v') \mid v \text{ adjacent to } v' \text{ in } \mathcal{P}, \text{ and } c^T v > c^T v'\}.$$

In this case we also say that G is linearly induced *by $\mathcal{P}$ (via c).*

Observation 1.2 *Let $G = (V, E)$ be linearly induced by $\mathcal{P}$.*

(i) G is acyclic.

(ii) Let $G' = (V', E')$ be the subgraph of G induced by the vertex set V' corresponding to a nonempty face of $\mathcal{P}$. Then G' has a unique sink (and also a unique source).

Proof. Property (i) is obvious. For (ii), the crucial fact (which also makes the simplex method work) is that any vertex which is not optimal has a neighbor of smaller objective function value. Because G comes from a generic objective function c, the optimal vertex is the only sink. This extends to the subgraphs induced by the nonempty faces, because each face of $\mathcal{P}$ is itself a polytope. Finally, considering the objective function $-c$ yields unique sources. □

While these are rather simple combinatorial properties 'inherited' from linear programming, the following one is less obvious and has only been discovered recently [13].

[2] $c^T v \neq c^T v'$ for any pair of vertices v, v' of $\mathcal{P}$

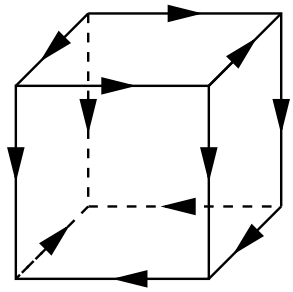

Figure 3: Orientation linearly induced by the Klee-Minty cube of Figure 1, via $c^T = (0, 0, 1)$

Lemma 1.3 *Let $G = (V, E)$ be linearly induced by an n-dimensional polytope $\mathcal{P}$, and let $s \in V$ be the unique source of G and $t \in V$ the unique sink of G. Then G has n directed paths from s to t, any two of which share only s and t.*

This lemma shows that the two cube orientations depicted in Figure 4 are not linearly induced by any combinatorial 3-cube, because only two suitable paths can be found in both cases. In contrast, the Klee-Minty cube of Figure 3 allows three disjoint paths from source to sink. Among the 18 (up to isomorphism) cube orientations which satisfy properties (i) and (ii) of Observation 1.2, the ones of Figure 4 are the only two that are not linearly inducible [9]. More generally, Klee and Mihalisin have shown that *any* orientation of a 3-polytope which has 3 disjoint source-sink paths on top of properties (i) and (ii) is actually linearly inducible [21]. The generalization of this to higher dimensions fails [10].

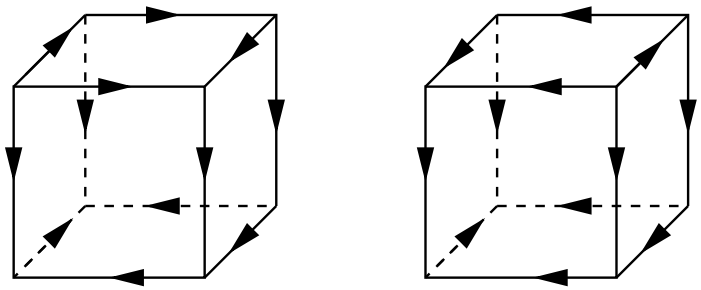

Figure 4: Cube orientations which are not linearly inducible

Abstract Objective Functions. RANDOM-FACET will also work for polytope orientations that are not necessarily linearly inducible. However, acyclicity and unique sinks in all nonempty faces are necessary for the analysis. Orientations with these properties come from *abstract objective functions* [12, 17].

Definition 1.4 *Let V be the vertex set of a polytope $\mathcal{P}$, ϕ an injective function that maps*

V to some totally ordered set. Let $G = (V, E)$ be the polytope orientation defined by

$$E := \{(v, v') \mid v \text{ adjacent to } v' \text{ in } \mathcal{P}, \text{ and } \phi(v) > \phi(v')\}.$$

ϕ is called an abstract objective function (AOF) if every subgraph $G' = (V', E')$ induced by the vertex set V' of a nonempty face of $\mathcal{P}$ has a unique sink.

Obviously, any generic linear objective function c defines an abstract objective function via $\phi(v) = c^T v$, but the converse is not true: topologically sorting any of the digraphs of Figure 4 results in an order of the vertices which determines an abstract objective function; due to Lemma 1.3, this order cannot be realized by any linear function.

Definition 1.5 *A digraph $G = (V, E)$ is called* combinatorially inducible *if there exists a polytope $\mathcal{P}$ and an abstract objective function ϕ on its set of vertices V', such that G is isomorphic to the graph $G' = (V', E')$, where*

$$E' := \{(v, v') \mid v \text{ adjacent to } v' \text{ in } \mathcal{P}, \text{ and } \phi(v) > \phi(v')\}.$$

In this case we also say that G is combinatorially induced *by $\mathcal{P}$ via ϕ.*

In this definition, $\mathcal{P}$ can be replaced by any polytope which is combinatorially equivalent to $\mathcal{P}$. Then, the digraphs of Figure 4 are combinatorially induced by the unit 3-cube C^3.

Combinatorial inducibility does not require the existence of disjoint paths as in Lemma 1.3. By an aforementioned result [10] we still get only an approximation to linear inducibility, even if we enforce this condition, so it seems that by dropping it we give away even more information.

However, the point is that we hardly know how to make use of this extra information. This paper describes a first nontrivial scenario in which properties of linear programming that go beyond combinatorial inducibility (namely, the disjoint paths) are applied in the analysis of randomized pivot rules. Previous results (including the $\exp(2\sqrt{n}) - 1$ bound for RANDOM-FACET on combinatorial n-cubes) only require combinatorial inducibility.

One reason for this is that combinatorially inducible orientations already share key features with the linearly inducible ones. The following is a straightforward generalization of the linear case [23, 17].

Theorem 1.6 *Let $G = (V, E)$ be combinatorially induced by a* simple *polytope $\mathcal{P}$, via an abstract objective function ϕ. Let $h_k(\mathcal{P}, \phi)$ denote the number of vertices with indegree k in G. Then*

$$h_k(\mathcal{P}, \phi) = h_k(\mathcal{P}),$$

*where $h(\mathcal{P}) := (h_0(\mathcal{P}), \ldots, h_n(\mathcal{P}))^T$ is the so-called h-*vector *of $\mathcal{P}$. In particular $h_k(\mathcal{P}, \phi)$ does not depend on ϕ.*

Proof. Let $f_i(\mathcal{P})$ denote the number of faces of $\mathcal{P}$ of dimension i. Because each face has a unique sink in G, the number of sinks of i-dimensional faces is $f_i(\mathcal{P})$. On the other

hand, every vertex of indegree k is the sink of exactly $\binom{k}{i}$ faces of dimension i (here we use the simplicity of $\mathcal{P}$). This yields

$$f_i(\mathcal{P}) = \sum_{k=0}^{n} \binom{k}{i} h_k(\mathcal{P}, \phi), \text{ or } f(\mathcal{P}) = M \cdot h(\phi),$$

where $f(\mathcal{P}) = (f_0(\mathcal{P}), \ldots, f_n(\mathcal{P}))^T$, $h(\mathcal{P}, \phi) = (h_0(\mathcal{P}, \phi), \ldots, h_n(\mathcal{P}, \phi))^T$ and M is an upper-triangular matrix with ones on the diagonal. Therefore, M is invertible, and

$$h(\mathcal{P}, \phi) = M^{-1} f(\mathcal{P}) =: h(\mathcal{P}).$$

□

Corollary 1.7 *Let $G = (V, E)$ be combinatorially induced by a simple polytope $\mathcal{P}$. Then every subgraph induced by the vertex set of a nonempty face of $\mathcal{P}$ has a unique source.*

Proof. Let c be a linear objective function, generic with respect to $\mathcal{P}$ and let F be an i-dimensional face of $\mathcal{P}$, $i \geq 0$. F is a simple polytope again, and because c and $-c$ are abstract objective functions on F, we get the *Dehn-Sommerville relations* [23]

$$h_k(F) = h_k(F, c) = h_{i-k}(F, -c) = h_{i-k}(F).$$

This implies $h_0(F, \phi) = h_i(F, \phi) = 1$ for all abstract objective functions ϕ. □

The h-vector of C^n (and of all combinatorial n-cubes) is given by $h_k(C^n) = \binom{n}{k}, k = 0, \ldots, n$. Consequently, in all orientations depicted in Figures 3 and 4, there are three vertices of indegree one (and three of indegree two, plus a source and a sink).

Below we mostly deal with orientations G induced by C^n via an abstract objective function ϕ. In this case, we will not mention C^n anymore and say that G is induced by ϕ.

The Random-Facet Algorithm. Here is the outline of the algorithm. Assume we are given a linear program whose feasible region is a *simple* n-polytope $\mathcal{P}$, and whose objective function c is generic with respect to $\mathcal{P}$. Moreover, we have an initial vertex v of $\mathcal{P}$. Among all n facets of P that contain v, we choose a facet F at random and recursively find the optimal vertex v' in F. If the unique neighbor v'' of v' which is not in F has higher objective function value than v', we return v', otherwise we repeat the above steps, now starting from v'' (which we reach from v' by a pivot step). The recursion bottoms out if the polytope under consideration consists only of a single vertex (a 0-polytope), in which case this vertex is returned. Because every pivot step strictly decreases the objective function value, vertices cannot repeat and the algorithm eventually terminates with the unique vertex of smallest objective function value.

One has to show that this outline is indeed an incarnation of the simplex method with suitable pivot rule [6, 5]. Because decisions are exclusively based on comparing values $c^T v'$ and $c^T v''$, where v', v'' are adjacent in $\mathcal{P}$, RANDOM-FACET can directly be formulated for linearly inducible polytope orientations. It is still guaranteed to work for

polytope orientations combinatorially induced by $\mathcal{P}$, via an abstract objective function ϕ: by induction, one proves that the vertex eventually returned must be a sink. Because there is a unique sink, it must be the sink defined by the vertex of lowest ϕ-value.

We now specialize to n-cube orientations induced by abstract objective functions. Identifying the vertices of C^n with $GF(2)^n$, the n-dimensional vector space over the 2-element field $GF(2)$, any such orientation can be specified by a digraph $G = (V, E)$, where $V = GF(2)^n$ and

$$E = \{(v, v') \mid \exists i : v - v' = \mathbf{e}_i, \text{ and } \phi(v) > \phi(v')\}.$$

Here, ϕ is some abstract objective function on C^n and $\mathbf{e}_i$ denotes the i-th unit vector.

The (vertex sets defining) nonempty faces of C^n can be described by pairs (v, S), $v \in V$, $S \subseteq [n] := \{1, \ldots, n\}$, where

$$(v, S) \simeq \{v' \in V \mid v_i = v'_i \; \forall i \notin S\}.$$

The dimension of (v, S) is $|S|$, and we have $(v, [n]) \simeq C^n$ and $(v, \emptyset) \simeq \{v\}$. I will write $v' \in (v, S)$ if the vertex v' is in the face defined by (v, S).

We are prepared to give pseudocode for the procedure RANDOM-FACET now, formulated in terms of the abstract objective function ϕ. Given a face (v, S), the recursive procedure finds the unique vertex $\operatorname{sink}_\phi(v, S) \in (v, S)$ such that

$$\phi(\operatorname*{sink}_\phi(v, S)) = \operatorname*{opt}_\phi(v, S) := \min\{\phi(v') \mid v' \in (v, S)\}.$$

Algorithm 1.8

RANDOM-FACET(v, S):
  IF $S = \emptyset$ THEN
    RETURN v
  ELSE
    choose $i \in S$ uniformly at random
    $v' :=$RANDOM-FACET$(v, S \setminus \{i\})$
    IF $\phi(v') < \phi(v' + \mathbf{e}_i)$ THEN
      RETURN v'
    ELSE
      $v'' := v' + \mathbf{e}_i$
      RETURN RANDOM-FACET(v'', S)
    END
  END

The reader will see from Theorem 2.4 below (or even now) that the correctness of the algorithm and the expected number of pivot steps are unaffected if the face (v'', S) is replaced with $(v'', S \setminus \{i\})$ in the second recursive call. This replacement even seems favorable, because it eliminates some redundant recursive calls later. However, the resulting algorithm does not generalize to the above informal description of RANDOM-FACET in a straightforward manner, so I have chosen to stick to the version presented above.

2 The Subexponential Bound

This section is devoted to the analysis of RANDOM-FACET, which is summarized by Theorem 2.3 below. Let the dimension $n > 0$ of the cube and the abstract objective function ϕ be fixed for the rest of this section.

Any concrete run of RANDOM-FACET(v, S) generates a binary *computation tree* with root (v, S). If $S = \emptyset$, there are no children, otherwise the left child is a computation tree with root $(v, S \setminus \{i\})$, where i is the element chosen by the algorithm for the first recursive call. In the case where a second recursive call takes place, we also have a right child with root (v'', S). (v, S) is connected to its children by edges labeled with i. Edges connecting a tree node to its right child are *pivot edges*, and their number equals the number of pivot steps that have been performed. The label of a pivot edge is the index of the coordinate which gets flipped in the corresponding pivot step. Figure 6 depicts a possible computation tree for the Klee-Minty cube of Figure 3. Pivot edges are drawn in bold and numbered by order of appearence during the algorithm. Figure 5 depicts the directed path to the sink corresponding to that specific computation tree (in bold).

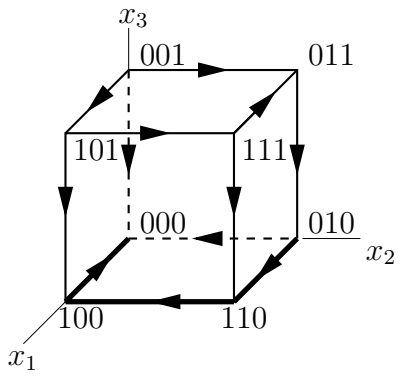

Figure 5: Cube orientation with vertices in $GF(2)^n$

With respect to a fixed computation tree, every pivot edge e can be mapped to a subset $T(e)$ of S, consisting of all labels of pivot edges that are encountered on the path $P(e)$ from e to the root of the tree. We also say that $P(e)$ *collects* the elements of $T(e)$. In Figure 6, we get

$$T(e_1) = \{1\}, \quad T(e_2) = \{2\}, \quad T(e_3) = \{1, 2\}.$$

If there is a pivot edge such that $T = T(e)$, we say that T is *assumed* in (v, S), written as $T \leftarrow_\phi (v, S)$. Obviously, we must have $T \subseteq S$ for that. Keep in mind that these notions are relative to a fixed computation tree. Below, we prove that if $T \leftarrow_\phi (v, S)$, then there is a unique edge e such that $T = T(e)$. For this, we need the notion of *fixed* indices.

Definition 2.1 *Index $i \in S$ is called* fixed *in (v, S) if there exists a vertex u with $\phi(v) < \mathrm{opt}_\phi(u, S \setminus \{i\})$, equivalently if $\phi(v) < \mathrm{opt}_\phi(v + \mathbf{e}_i, S \setminus \{i\})$ (u must differ from v in the i-th coordinate, hence $(u, S \setminus \{i\}) = (v + \mathbf{e}_i, S \setminus \{i\})$).*

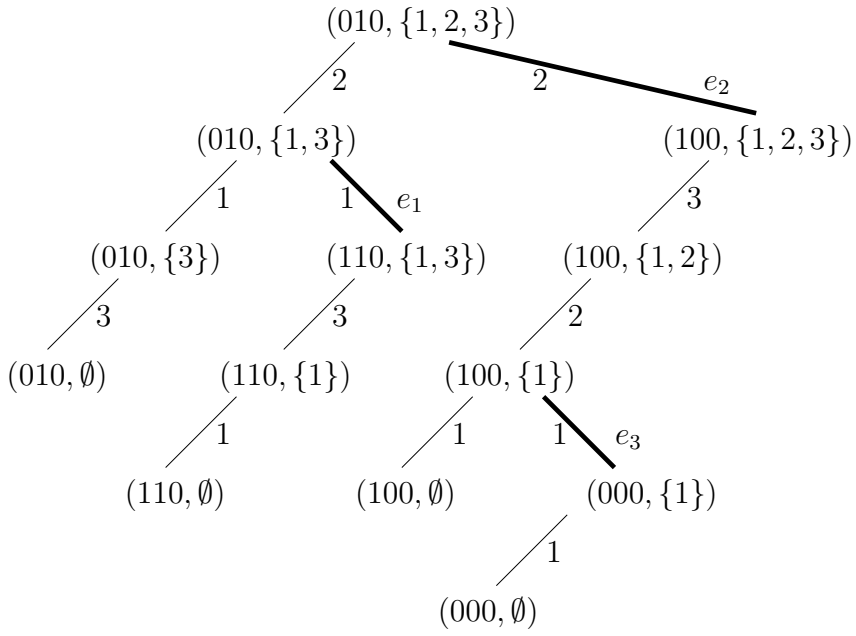

Figure 6: Computation tree generated by RANDOM-FACET

If i is fixed, all vertices encountered in the call to RANDOM-FACET(v, S) must have the same i-th coordinate v_i, because the vertices in (v, S) that differ from v in coordinate i span exactly the face $(u, S \setminus \{i\})$ of Definition 2.1. Thus, they all have higher ϕ-value and cannot be met anymore.

Lemma 2.2 *Let (v, S) with $|S| > 0$ be the root of a computation tree in which $(v, S \setminus \{i\})$ is the left child of (v, S); fix $T \subseteq S, |T| > 0$.*

(i) With $v' := \mathrm{sink}_\phi(v, S \setminus \{i\})$, $v'' := v' + \mathbf{e}_i$,

$$T \underset{\phi}{\leftarrow}(v, S) \Leftrightarrow T \underset{\phi}{\leftarrow}(v, S \setminus \{i\}), \quad \text{if } i \notin T, \tag{4}$$

and

$$T \underset{\phi}{\leftarrow}(v, S) \Leftrightarrow T \setminus \{i\} \underset{\phi}{\leftarrow}(v'', S) \text{ and } \phi(v'') < \phi(v'), \quad \text{if } i \in T. \tag{5}$$

For this we stipulate that $\emptyset$ is assumed in every pair.

(ii) If $T \leftarrow_\phi (v, S)$, there is a unique pivot edge e such that $T = T(e)$.

While (4) is easily seen, (5) follows once we can show that not both T and $T \setminus \{i\}$ can be assumed in (v'', S). This holds, because

$$\phi(v'') < \phi(v') = \underset{\phi}{\mathrm{opt}}(v, S \setminus \{i\}),$$

so i is fixed in (v'', S). This means, i cannot occur as a pivot edge label in the right subtree. (4) and (5) also show that e is determined by T, thus (ii) holds: if $i \notin T$, e must

be in the left subtree, where we can locate it recursively. If $i \in T$, then either e is the edge connecting (v, S) to (v'', S) (if $T = \{i\}$), or e is in the right subtree where $T \setminus \{i\}$ determines it.

Theorem 2.3 *The expected number of pivot steps taken by* RANDOM-FACET *on any pair* $(v, S), |S| = k$ *is bounded by*

$$f(k) := \sum_{\ell=1}^{k} \frac{1}{\ell!} \binom{k}{\ell} \leq e^{2\sqrt{k}} - 1.$$

Proof. We show that

$$\text{prob}(T \underset{\phi}{\leftarrow}(v, S)) \leq \frac{1}{|T|!}, \quad T \subseteq S, \tag{6}$$

where we refer to the probability distribution over the set of computation trees generated by the call RANDOM-FACET(v, S). By Lemma 2.2 (ii), the sum of these probabilities (disregarding the empty set) equals the expected number of pivot steps, which yields the first part of the theorem. For the second part, we use the following estimate.

$$\begin{aligned} f(k) + 1 &= \sum_{\ell=0}^{k} \binom{k}{\ell} \frac{1}{\ell!} \leq \sum_{\ell=0}^{k} \frac{k^\ell}{\ell!} \frac{1}{\ell!} = \sum_{\ell=0}^{k} \frac{k^\ell}{\ell!^2} \\ &= \sum_{\ell=0}^{k} (\frac{\sqrt{k}^\ell}{\ell!})^2 \leq (\sum_{\ell=0}^{k} \frac{\sqrt{k}^\ell}{\ell!})^2 \leq (\sum_{\ell=0}^{\infty} \frac{\sqrt{k}^\ell}{\ell!})^2 = e^{2\sqrt{k}}. \end{aligned}$$

It remains to prove (6). Note that in general, different sets are assumed with different probabilities, so there is no obvious symmetry argument from which the claim follows. We proceed by induction, noting that for $|T| = 0$ or $|S| = 0$, the statement is trivial. If $|T| = \ell > 0, |S| = k > 0$, we assume that (6) holds for all pairs (T', S') such that $|S'| < k$ or $|T'| < \ell$. Define

$$v^{(i)} := \underset{\phi}{\text{sink}}(v, S \setminus \{i\}) + \mathbf{e}_i$$

and set $T^+ := \{i \in T \mid \phi(v^{(i)}) < \text{opt}_\phi(v, S \setminus \{i\})\}$. For $i \in S$, let $\text{prob}(T \leftarrow_\phi (v, S)|i)$ denote the probability that $T \leftarrow_\phi (v, S)$, conditioned on the event that RANDOM-FACET(v, S) choses index i. By Lemma 2.2 (i),

$$\text{prob}(T \underset{\phi}{\leftarrow}(v, S)|i) = \begin{cases} \text{prob}(T \leftarrow_\phi (v, S \setminus \{i\}), & i \notin T, \\ \text{prob}(T \setminus \{i\} \leftarrow_\phi (v^{(i)}, S), & i \in T^+, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $p := \text{prob}(T \leftarrow_\phi (v, S))$ satisfies

$$p = \frac{1}{k} \left(\sum_{i \in S \setminus T} \text{prob}(T \underset{\phi}{\leftarrow}(v, S \setminus \{i\})) + \sum_{i \in T^+} \text{prob}(T \setminus \{i\} \underset{\phi}{\leftarrow}(v^{(i)}, S)) \right). \tag{7}$$

For $T \setminus \{i\}$ to be assumed in $(v^{(i)}, S)$, it is necessary that no element of $T \setminus \{i\}$ is fixed in $(v^{(i)}, S)$. This is only possible if

$$\operatorname*{opt}_{\phi}(v, S \setminus \{i\}) > \operatorname*{opt}_{\phi}(v, S \setminus \{j\}), \quad \forall j \in T \setminus \{i\}, \tag{8}$$

otherwise

$$\phi(v^{(i)}) < \operatorname*{opt}_{\phi}(v, S \setminus \{i\}) \leq \operatorname*{opt}_{\phi}(v, S \setminus \{j\})$$

for some j, meaning that j is fixed in $(v^{(i)}, S)$. Because there is at most one index i which satisfies (8), we can choose an index $t \in T$ such that

$$\begin{aligned} p &\leq \frac{1}{k}\left(\sum_{i \in S \setminus T} \operatorname{prob}(T \underset{\phi}{\leftarrow}(v, S \setminus \{i\})) + \operatorname{prob}(T \setminus \{t\} \underset{\phi}{\leftarrow}(v^{(t)}, S))\right) \\ &\leq \frac{1}{k}\left(\frac{k-\ell}{\ell!} + \frac{1}{(\ell-1)!}\right) = \frac{1}{k}\left(\frac{k-\ell}{\ell!} + \frac{\ell}{\ell!}\right) = \frac{1}{\ell!}. \end{aligned} \tag{9}$$

□

If i is fixed in the root (v, S) of a computation tree, then i is also fixed in all nodes (v', S') such that $i \in S'$. In particular, $\operatorname{opt}_{\phi}(v', S') = \operatorname{opt}_{\phi}(v', S' \setminus \{i\})$. Using these observations and induction, (7) can be used to prove that

$$\operatorname{prob}(T \underset{\phi}{\leftarrow}(v, S)) = \operatorname{prob}(T \underset{\phi}{\leftarrow}(v, S \setminus \{i\})). \tag{10}$$

This implies the following theorem, which also underlies the discussion at the end of the previous section.

Theorem 2.4 *Let i be fixed in (v, S). Then $\operatorname{opt}_{\phi}(v, S) = \operatorname{opt}_{\phi}(v, S \setminus \{i\})$. Moreover,* Random-Facet(v, S) *and* Random-Facet$(v, S \setminus \{i\})$ *perform the same expected number of pivot steps.*

3 Matoušek's Abstract Objective Functions

Let $A \in GF(2)^{n \times n}$ be an invertible, upper-triangular matrix (it follows that A has one-entries along the main diagonal). A defines a bijection $\phi : GF(2)^n \mapsto GF(2)^n$ via

$$\phi(v) = Av.$$

We prove below that ϕ defines an abstract objective function on the n-cube under lexicographic order of the values, meaning that $w < w'$ if $w_i = 0$ at the largest index where w differs from w'. Interpreting w as the n-bit binary number $w_n \cdots w_1$, this is the canonical order among the natural numbers. For simplicity, we identify ϕ with the matrix A.

Here are a few examples and remarks which illustrate the construction. Let

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then the order of the vertices is as follows (according to the interpretation above, we will write the values Av as natural numbers).

Av	7	6	5	4	3	2	1	0
v	$\begin{pmatrix}0\\0\\1\end{pmatrix}$	$\begin{pmatrix}1\\0\\1\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\0\end{pmatrix}$	$\begin{pmatrix}1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\end{pmatrix}$

Thus, A induces a hamiltonian path, and the resulting cube orientation is already familiar: it is the Klee-Minty cube of Figure 3 (see Figure 5 for the assignment of vertices). In general, if A is the $(n \times n)$-matrix with all entries above the diagonal being equal to one, it induces the same n-cube orientation as the linear program (3). The matrices

$$A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad A_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

induce the orientations of Figure 4 which we have identified as not being linearly inducible. Figure 7 also shows the corresponding abstract objective function values.

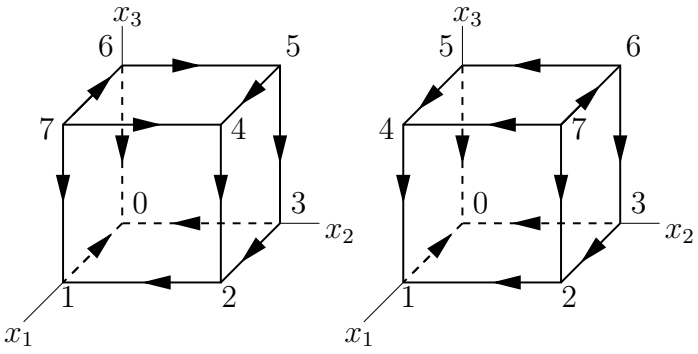

Figure 7: Cube orientations induced by A_1 (left) and A_2 (right)

The orientation induced by A can be obtained directly by using the following inductive construction. Assume we already know the orientation of the face $(\mathbf{0}, [k-1])$, which is the set of vertices v such that $v_k = \cdots = v_n = 0$. To obtain the missing orientations within

$$(\mathbf{0}, [k]) = (\mathbf{0}, [k-1]) \cup (\mathbf{e}_k, [k-1]),$$

we first direct all edges from $(\mathbf{e}_k, [k-1])$ to $(\mathbf{0}, [k-1])$. Within $(\mathbf{e}_k, [k-1])$, all orientations in coordinate direction $x_i, i < k$ are either equal to the ones of the corresponding edges in $(\mathbf{0}, [k-1])$, or exactly reversed. The former happens if $a_{ik} = 0$, the latter is obtained for $a_{ik} = 1$. For example, in Figure 7 (left), the upper and lower facet have the same orientations in direction x_1, but reversed orientations with respect to x_2. This is the case because $a_{13} = 0$ and $a_{23} = 1$ for the matrix $A = A_1$.

The $2^{\binom{n}{2}}$ different invertible, upper-triangular matrices A define the class $\mathcal{A}$ of *Matoušek functions*, and the induced orientations are the *Matoušek orientations*. These are

very special orientations, and of course there are combinatorially inducible orientations which are not isomorphic to any member of this class. In particular, the Matoušek orientations are "combed", meaning that there are opposite facets F, F' such that all edges between them are oriented from F to F'. Figure 8 depicts a combinatorially (actually, linearly) inducible orientation which is not combed.

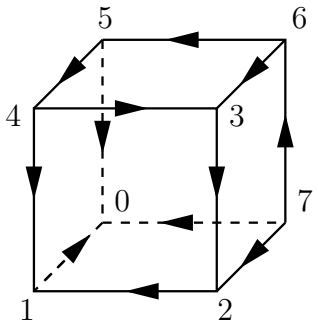

Figure 8: Cube orientation which is not combed

For the formal proof that the Matoušek functions actually define abstract objective functions, we need some notation.

Definition 3.1 *Let $A \in GF(2)^{n\times n}$, $v \in GF(2)^n$, $S \subseteq [n]$ and $i \in [n]$.*

(i) $A(S) \in GF(2)^{|S|\times|S|}$ denotes the principal minor of A consisting of the entries of A in rows and columns with indices in S ($A(S)$ is invertible and upper-triangular again),

(ii) A_i is the i-th column of A, and

(iii) $v_S \in GF(2)^{|S|}$ is the subvector of v consisting of the entries with indices in S.

Theorem 3.2 *Let $A \in \mathcal{A}$ be a Matoušek function.*

(i) v is a sink in (v, S) (with respect to the Matoušek orientation induced by A) if and only if $(Av)_S = \mathbf{0}$.

(ii) A is an abstract objective function.

Proof. (i) v is a sink if and only if for all $i \in S$, $Av < A(v + \mathbf{e}_i)$. This is equivalent to $(Av)_i = 0$, because i is the largest index where Av differs from $A(v + \mathbf{e}_i) = Av + A_i$.

(ii) Assume there are two sinks $v, v' \in (v, S)$ and define $u := v - v'$. Then we have $u_{[n]\setminus S} = \mathbf{0}$ and $(Au)_S = \mathbf{0}$ by (i). This implies $A(S)u_S = \mathbf{0}$, and because $A(S)$ is invertible as well, we get $u_S = \mathbf{0}$. Hence $u = \mathbf{0}$ and $v = v'$. □

Now we are prepared to describe the behavior of the RANDOM-FACET algorithm on a given Matoušek function A. To this end, we will reformulate it as a process dealing with *values* $w = Av$ instead of *vertices* v. Because A is a bijection, this can be done, and the resulting algorithm turns out to be easier to handle in the sequel. Given a pair (S, w), the following routine finds the value $\mathrm{opt}_A(A^{-1}w, S)$.

Algorithm 3.3

RANDOM-FACET(S, w):
  IF $S = \emptyset$ THEN
    RETURN w
  ELSE
    choose $i \in S$ at random
    $w' :=$RANDOM-FACET$(S \setminus \{i\}, w)$
    IF $w'_i = 0$ THEN
      RETURN w'
    ELSE
      $w'' := w' + A_i$
      RETURN RANDOM-FACET(S, w'')
    END
  END

Let $w' = Av', w'' = Av''$. In the proof of Theorem 3.2 we have already seen that

$$\begin{aligned} w'_i = 0 &\Leftrightarrow Av' < A(v' + \mathbf{e}_i) \text{ and} \\ w'' = w' + A_i &\Leftrightarrow v'' = v' + \mathbf{e}_i, \end{aligned}$$

so Algorithm 3.3 is equivalent to Algorithm 1.8 via the coordinate transformation $v \to Av$. In other words, when we replace every node (v', S') in a computation tree for RANDOM-FACET(v, S) with the node (S', Av'), we obtain a computation tree for the call RANDOM-FACET(S, Av), and vice versa.

The property of RANDOM-FACET(S, w) that comes in handy below is that its expected peformance is determined by w_S and $A(S)$, although coordinates with indices not in S are affected by the pivot steps. In contrast, RANDOM-FACET(v, S) manipulates only coordinates in S, but takes decisions based on other coordinates.

The one-to-one correspondence allows us to extend previous terminology to the new setup. For fixed $S, w, v := A^{-1}w$ and $i \in S$, we define

$$\underset{A}{\text{opt}}(S, w) := \underset{A}{\text{opt}}(v, S), \tag{11}$$

$$T \underset{A}{\leftarrow}(S, w) :\Leftrightarrow T \underset{A}{\leftarrow}(v, S), \tag{12}$$

$$i \text{ fixed in } (S, w) :\Leftrightarrow i \text{ fixed in } (v, S). \tag{13}$$

The fact that RANDOM-FACET(S, w) only looks at w_S and $A(S)$ means that for all A, A' and w, w' such that $A(S) = A'(S)$ and $w_S = w'_S$,

$$\text{prob}(T \underset{A}{\leftarrow}(S, w)) = \text{prob}(T \underset{A'}{\leftarrow}(S, w')) \tag{14}$$

holds. The following Lemma establishes a few further facts which we need later.

Lemma 3.4 *Fix $A, S, w, v := A^{-1}w$ and $i \in S$.*

(i) i is fixed in (S, w) if and only if

$$i > \mathrm{hi}(S, w) := \max\{j \in S \mid w_j = 1\}.$$

(ii) $\mathrm{opt}_A(S \setminus \{i\}, w) \neq \mathrm{opt}_A(S \setminus \{i\}, w + \mathbf{e}_i)$.

(iii) The orientation induced by A on the face (v, S) is isomorphic to the orientation induced by $A(S)$ on the $|S|$-cube $(\mathbf{0}, [|S|])$.

Proof. (i) Define $v' := \mathrm{sink}_A(v + \mathbf{e}_i, S \setminus \{i\})$. We inductively show that both statements are equivalent to

$$(Av)_j = (Av')_j, \text{ equivalently } v_j = v'_j, \quad k > i, \tag{15}$$
$$(Av)_i = 0, \tag{16}$$
$$(Av')_i = 1. \tag{17}$$

The equivalence in (15) holds because A is upper-triangular.

First suppose i is fixed in (S, w) or $i > \mathrm{hi}(S, w)$. We assume that (15) already holds for $j > t > i$ (initially, we set $t = n$). If $t \in S$, then $(Av')_t = 0$ by Theorem 3.2 (i), and $(Av)_t = 0$ follows from $Av < Av'$ if i is fixed (Definition 2.1) and immediately if $i > \mathrm{hi}(S, w)$. If $t \notin S$, then $v_t = v'_t$.

This means, i is the largest index such that $v_i \neq v'_i$, and this implies $(Av)_i \neq (Av')_i$. With $Av < Av'$ or $i > \mathrm{hi}(S, w)$, (16) and (17) follow.

On the other hand, given (15), (16) and (17), it immediately follows that $Av < Av'$, so i is fixed in (S, w). Moreover, we have already seen that if $j \in S, j \geq i$, then $(Av)_j = 0$, which shows $i > \mathrm{hi}(S, w)$.

(ii) Let $v' := A^{-1}(w + \mathbf{e}_i)$. i is the largest index where $w = Av$ and $w + \mathbf{e}_i = Av'$ differ, which also yields $v_i \neq v'_i$. Thus, $(v, S \setminus \{i\})$ and $(v', S \setminus \{i\})$ are disjoint faces, meaning that

$$\underset{A}{\mathrm{opt}}(S \setminus \{i\}, w) = \underset{A}{\mathrm{opt}}(v, S \setminus \{i\}) \neq \underset{A}{\mathrm{opt}}(v', S \setminus \{i\}) = \underset{A}{\mathrm{opt}}(S \setminus \{i\}, w + \mathbf{e}_i).$$

(iii) The isomorphism is provided by the mapping

$$v \to (v + \underset{A}{\mathrm{sink}}(v, S))_S.$$

To see this, start with the equation

$$Av = A(v + \underset{A}{\mathrm{sink}}(v, S)) + A \underset{A}{\mathrm{sink}}(v, S).$$

Because $(v + \mathrm{sink}_A(v, S))_{[n]\setminus S} = (A \,\mathrm{sink}_A(v, S))_S = \mathbf{0}$, this further gives

$$(Av)_S = A(S)(v + \underset{A}{\mathrm{sink}}(v, S))_S. \tag{18}$$

For $i \in S$ and $v' := v + \mathbf{e}_i$, (v, v') is an edge in the orientation induced by A on (v, S) if and only if $(Av)_i = 1$. By (18), this is equivalent to having an edge between $(v + \mathrm{sink}_A(v, S))_S$ and $(v' + \mathrm{sink}_A(v', S))_S = (v + \mathrm{sink}_A(v, S) + \mathbf{e}_i)_S$ in the orientation induced by $A(S)$ on $(\mathbf{0}, [|S|])$. ∎

4 Matoušek's Lower Bound

In this section we prove Matoušek's result that RANDOM-FACET may perform an expected number of $\exp(\Theta(\sqrt{n}))$ pivot steps in the worst case [20]. This means, the analysis of the previous section is essentially tight. The lower bound is obtained by running RANDOM-FACET on a random Matoušek function A, with random start value w.

The interpretation of this result is that RANDOM-FACET is not a polynomial-time algorithm for the class of combinatorially inducible cube orientations. However, it is still possible that RANDOM-FACET takes polynomial time on all *linearly* inducible cube orientations. In fact, we prove in the next section that RANDOM-FACET takes an expected number of only $O(n^2)$ pivot steps for all linearly inducible Matoušek orientations, and this bound is tight for the n-dimensional Klee-Minty cube [20, 8].

Moreover, there is no superquadratic lower bound for RANDOM-FACET on *any* linearly inducible cube orientation. In particular, the question whether RANDOM-FACET leads to a polynomial-time simplex algorithm for linear programming remains completely open.

Theorem 4.1 *Let $\mathcal{A}$ be the class of Matoušek funtions on the n-cube, $S \subseteq [n], |S| = k$. The expected number of pivot steps taken by* RANDOM-FACET(S, w)*, averaged over all $A \in \mathcal{A}$ and all $w \in GF(2)^n$, equals*

$$g(k) := \sum_{\ell=1}^{k} \frac{(1/2)^\ell}{\ell!} \binom{k}{\ell}.$$

For $k > 0$,

$$g(k) \geq \frac{1}{16\sqrt{k}} e^{\sqrt{2k}} - 1.$$

Proof. For fixed T and $R \supseteq T$, we show that, averaged over all A and all $w \in (\mathbf{0}, R)$,

$$\operatorname{prob}(T \underset{A}{\leftarrow} (S, w)) = \frac{(1/2)^{|T|}}{|T|!}. \tag{19}$$

Using Lemma 2.2 (ii), the first part follows. For the estimate in the second part, we provide a proof in the appendix.

For $|T| = 0$, (19) is trivial. For $|T| > 0$, define

$$p_R = \operatorname*{prob}_R(T \underset{A}{\leftarrow} (S, w)) := \frac{1}{2^{|R|}} \sum_{w \in (\mathbf{0}, R)} \operatorname{prob}(T \underset{A}{\leftarrow} (S, w))$$

and

$$p_{\mathcal{A},R} = \operatorname*{prob}_{\mathcal{A},R}(T \underset{A}{\leftarrow} (S, w)) := \frac{1}{|\mathcal{A}|} \sum_{A \in A} \operatorname*{prob}_R(T \underset{A}{\leftarrow} (S, w)),$$

the latter being the quantity we are concerned with.

Let us specialize equation (7) to a fixed Matoušek function A now. With $p = \operatorname{prob}(T \leftarrow_A (S, w))$, we get

$$p = \frac{1}{k}\left(\sum_{i \in S \setminus T} \operatorname{prob}(T \underset{A}{\leftarrow}(S \setminus \{i\}), w) + \sum_{i \in T^+} \operatorname{prob}(T \setminus \{i\} \underset{A}{\leftarrow}(S, w^{(i)})\right),$$

where

$$w^{(i)} := Av^{(i)} = A \operatorname*{sink}_A(v, S \setminus \{i\}) + A_i, \quad v = A^{-1}w.$$

As before, $T \setminus \{i\}$ can only be assumed in $(S, w^{(i)})$ if no element of $T \setminus \{i\}$ is fixed in $(S, w^{(i)})$. Here, this implies $i = \max(T)$. To see this, observe that by Theorem 3.2, $A \operatorname{sink}_A(v, S \setminus \{i\})$ agrees with the unit vector $\mathbf{e}_i$ on all coordinates with index in S (here we use $i \in T^+$). Therefore,

$$w_S^{(i)} = (\mathbf{e}_i + A_i)_S, \tag{20}$$

so

$$\operatorname{hi}(S, w^{(i)}) \leq i - 1.$$

If $i \neq t := \max(T)$, then $t > i$ is fixed in $(S, w^{(i)})$ by Lemma 3.4 (i). Together with (20) and (14), this lets us rewrite p as

$$p = \frac{1}{k}\left(\sum_{i \in S \setminus T} \operatorname{prob}(T \underset{A}{\leftarrow}(S \setminus \{i\}), w) + \operatorname{prob}(T \setminus \{t\} \underset{A}{\leftarrow}(S, \mathbf{e}_t + A_t))[t \in T^+]\right), \tag{21}$$

where $[\cdot]$ is the indicator variable for the event in brackets. Summing equation (21) over all $w \in (\mathbf{0}, R)$ yields

$$p_R = \frac{1}{k}\left(\sum_{i \in S \setminus T} \operatorname*{prob}_R(T \underset{A}{\leftarrow}(S \setminus \{i\}), w) + \frac{1}{2}\operatorname{prob}(T \setminus \{t\} \underset{A}{\leftarrow}(S, \mathbf{e}_t + A_t))\right). \tag{22}$$

Here, we have used

$$\frac{1}{2^{|R|}} \sum_{w \in (\mathbf{0}, R)} [t \in T^+] = \frac{1}{2}.$$

In other words, $\operatorname{opt}(S \setminus \{t\}, w) \neq \operatorname{opt}(S, w)$ for exactly half of the w's. This follows from Lemma 3.4 (ii) with $i = t$.

Summing equation (22) over all $A \in \mathcal{A}$ yields

$$p_{\mathcal{A},R} = \frac{1}{k}\left(\sum_{i \in S \setminus T} \operatorname*{prob}_{\mathcal{A},R}(T \underset{A}{\leftarrow}(S \setminus \{i\}), w) + \frac{1}{2|\mathcal{A}|}\sum_{A \in \mathcal{A}} \operatorname{prob}(T \setminus \{t\} \underset{A}{\leftarrow}(S, \mathbf{e}_t + A_t))\right). \tag{23}$$

To proceed further, we invoke two facts. First,

$$\operatorname{prob}(T \setminus \{t\} \underset{A}{\leftarrow}(S, w)) = \operatorname{prob}(T \setminus \{t\} \underset{A}{\leftarrow}(S \setminus \{t\}, w)),$$

for all $w \in (0,[t-1])$, because in this case $t > \mathrm{hi}(S,w)$ is fixed in (S,w), cf. (10) and Lemma 3.4 (i). Second,

$$\mathrm{prob}(T \setminus \{t\} \underset{A}{\leftarrow} (S \setminus \{t\}, w)) = \mathrm{prob}(T \setminus \{t\} \underset{A'}{\leftarrow} (S \setminus \{t\}, w))$$

for any matrix A' which differs from A only in the t-th column, see (14).

This lets us conclude that

$$\begin{aligned}
\frac{1}{2|\mathcal{A}|} \sum_{A \in \mathcal{A}} \mathrm{prob}(T \setminus \{t\} \underset{A}{\leftarrow} (S, \mathbf{e}_t + A_t)) &= \frac{1}{2|\mathcal{A}|} \sum_{w \in (\mathbf{0},[t-1])} \sum_{\substack{A \in \mathcal{A} \\ \mathbf{e}_t + A_t = w}} \mathrm{prob}(T \setminus \{t\} \underset{A}{\leftarrow} (S,w)) \\
&= \frac{1}{2|\mathcal{A}|} \sum_{w \in (\mathbf{0},[t-1])} \frac{1}{2^{t-1}} \sum_{A \in \mathcal{A}} \mathrm{prob}(T \setminus \{t\} \underset{A}{\leftarrow} (S,w)) \\
&= \frac{1}{2} \underset{\mathcal{A},[t-1]}{\mathrm{prob}} (T \setminus \{t\} \underset{A}{\leftarrow} (S,w)).
\end{aligned}$$

Plugging this into (23) gives

$$\underset{\mathcal{A},R}{\mathrm{prob}}(T \underset{A}{\leftarrow} (S,w) = \frac{1}{k} \left(\sum_{i \in S \setminus T} \underset{\mathcal{A},R}{\mathrm{prob}}(T \underset{A}{\leftarrow} (S \setminus \{i\}), w) + \frac{1}{2} \underset{\mathcal{A},[t-1]}{\mathrm{prob}} (T \setminus \{t\} \underset{A}{\leftarrow} (S,w)) \right),$$

from which (19) easily follows inductively, similar to our previous derivation of (6) from (9). □

5 Linearly Inducible Matoušek Orientations

This section deals with the Matoušek orientations of the n-cube that are linearly inducible. I will first show that the generating matrix A must have a special structure in this case. From this, it will follow that Random-Facet takes an expected number of $O(n^2)$ pivot steps on any linearly inducible Matoušek orientation. This result has first been published in [7].

Lemma 5.1

(i) Let $A \in \mathcal{A}$ be a Matoušek function on the n-cube. If the orientation defined by A is linearly inducible, then for all $S \subseteq [n], |S| = 3$,

$$A(S) \neq A_1 := \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \quad \textit{and} \quad A(S) \neq A_2 := \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

(ii) For any matrix $A \in A$ such that $A(S) \neq A_1, A_2$ for all $S \subseteq [n], |S| = 3$, the inverse A^{-1} has at most one off-diagonal one-entry per row.

Part (i) uses the fact that any 3-face of a linearly inducible orientation is linearly inducible. Therefore, no such 3-face orientation can be isomorphic to the Matoušek orientations generated by A_1 and A_2: Figure 7 shows that they fail to satisfy the path condition established by Lemma 1.3. With Lemma 3.4 (iii), the claim follows.

The heart of the Lemma is part (ii). For example, if A is the unit matrix I_n, the corresponding orientation is linearly inducible, namely via the objective function $c^T x = x_1 + \cdots + x_n$ on the standard cube C^n. The statement of the Lemma holds for $A^{-1} = I_n$, because this matrix has no off-diagonal one-entries at all.

A more intersting example is the matrix

$$A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix},$$

which generates the n-dimensional Klee-Minty cube. This orientation is linearly induced by the linear program (3). In this case,

$$A^{-1} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 1 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix},$$

and A^{-1} has one off-diagonal one-entry in all rows except the last one.

Proof. Part (i) has already been dealt with, so consider part (ii). Because all principal minors $A([i]), i \in [n]$ are invertible, each column A_i can be written in the form

$$A_i = \mathbf{e}_i + \sum_{j \in J(i)} A_j, \tag{24}$$

where $J(i) \subseteq [i-1]$ is a unique index set. We now prove two claims. For this let $A = (a_{ij}), 1 \leq i, j \leq n$, i.e. a_{ij} is the element in row i and column j.

Claim 1. For all i, the columns $A_j, j \in J(i)$ are disjoint in the sense that no two of them have a one-entry in the same row.

Proof (of Claim 1): assume on the contrary that $k \leq j_1 < j_2$ exist such that $j_1, j_2 \in J(i)$ and $a_{k,j_1} = a_{k,j_2} = 1$. Suppose that the pair (k, j_1) is lexicographically largest with this property.

Case 1. $k < j_1$. Then we have the situation of Figure 9 (left), and because (k, j_1) was lexicographically largest with $a_{k,j_1} = a_{k,j_2} = 1$, we get $\alpha = a_{j_1,j_2} = 0$. In this case, $A(\{k, j_1, j_2\}) = A_2$, a contradiction to our assumption.

Case 2. $k = j_1$. Then the situation is as in Figure 9 (right). Again, the choice of (k, j_1) ensures that in rows j_1 and j_2, all entries in columns $J(i) \setminus \{j_1, j_2\}$ are zero (in Figure 9 (right), these entries are in the shaded area). The definition of $J(i)$ then implies $\alpha = a_{j_2,i} = 1, \beta = a_{j_1,i} = 0$, which in turn yields $A(\{j_1, j_2, i\}) = A_1$, a contradiction.

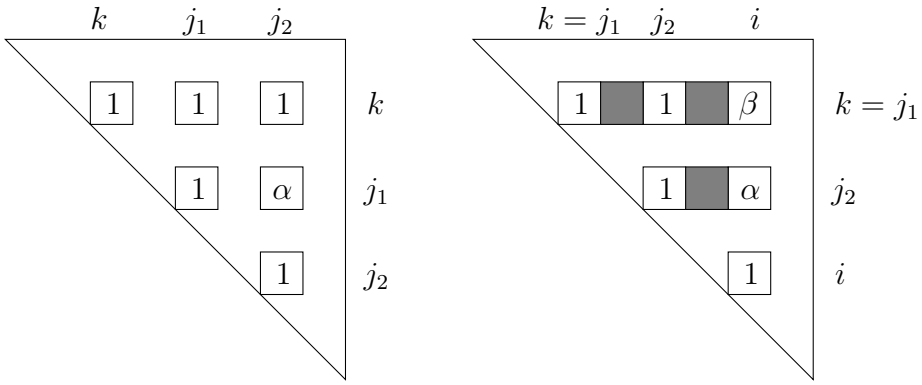

Figure 9: Proof of Claim 1. Case $k < j_1$ (left), $k = j_1$ (right)

Claim 2. The index sets $J(i), i \in [n]$ are pairwise disjoint, i.e. in representing the columns according to (24), every column A_j contributes to at most one other column A_i.

Proof (of Claim 2): Assume a column A_j contributes to two distinct columns $A_{i_1}, A_{i_2}, j < i_1 < i_2$. By Claim 1, A_j is disjoint from all other columns that contribute to A_{i_1} resp. A_{i_2}, so we get $a_{j,i_1} = a_{j,i_2} = 1$, see Figure 10 (left).

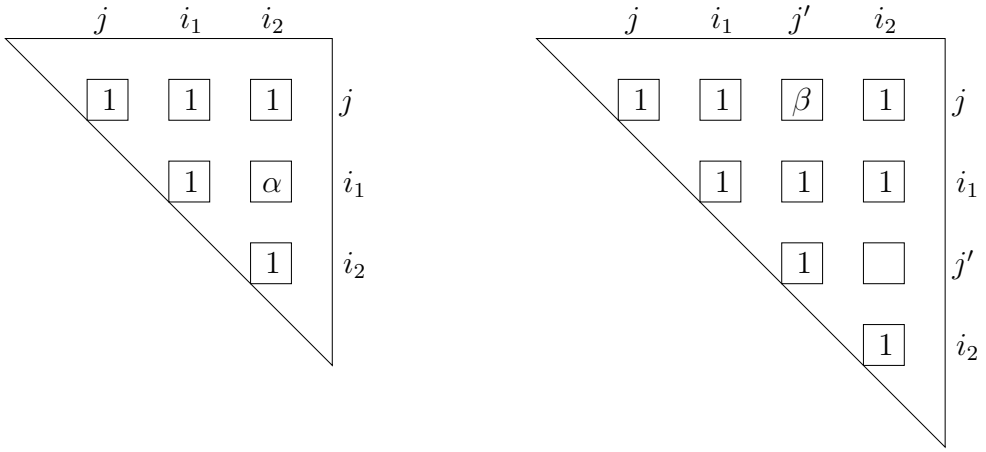

Figure 10: Proof of Claim 2

In order to avoid the forbidden minor A_2, we must have $\alpha = a_{i_1,i_2} = 1$. By Claim 1, there is a unique index $j' \in J(i_2), j' \geq i_1$ which is "responsible" for the one-entry

α, meaning that $a_{i_1,j'} = 1$ (see Figure 10 (right)). Because $j, j' \in J(i_2)$, columns A_j and $A_{j'}$ are disjoint, which implies $j' \neq i_1$ and also $\beta = a_{j,j'} = 0$. Then, however, $A(\{j, i_1, j'\}) = A_1$, a contradiction.

Now we are prepared to prove (ii), which follows if we can show that the columns of $M := I_n + A^{-1}$ are disjoint in the sense of Claim 1. The i-th column of M is given as

$$M_i = \mathbf{e}_i + A_i^{-1} = A^{-1}(A_i + \mathbf{e}_i) = A^{-1} \sum_{j \in J(i)} A_j = \sum_{j \in J(i)} \mathbf{e}_j.$$

The disjointness of the $J(i), i \in [n]$ now immediately implies the disjointness of the columns M_i. □

Lemma 5.1 (ii) implies that A^{-1} is extremely sparse in the realizable case, and this will entail that RANDOM-FACET is fast on A. Before we go into the formal analysis, here is the intuitive argument why this is the case, and how the inverse matrix A^{-1} comes in. Consider starting the algorithm on the pair $(v, [n])$. With probability $1/n$, the first recursive call will compute $v' = \mathrm{opt}(v, [n] \setminus \{i\})$, for any $i \in [n]$. We know from Lemma 3.2 (i) that $Av' = 0$ or $Av' = \mathbf{e}_i$, and only in the latter case, the algorithm performs a second recursive call, starting with the vertex $v'' = v' + \mathbf{e}_i$. It follows that

$$v'' = A^{-1}A(v' + \mathbf{e}_i) = A^{-1}(\mathbf{e}_i + A_i) = A_i^{-1} + \mathbf{e}_i.$$

This means, the possible v'' coming up in the second recursive call are nothing else than the pairwise disjoint columns of the matrix M considered in the proof of Theorem 5.1. Now, if that call fixes some position $j \in [n]$ in its first recursive step, it fixes a zero position with high probability, because $v''_j \neq 0$ for at most one i. This means that already after the first recursive call in RANDOM-FACET$(v'', [n])$, the problem has been optimally solved in $n - 1$ out of n cases, because fixing a zero means that the optimal vertex 0 lies in the facet that is being considered.

In the following formal derivation of the $O(n^2)$ bound, it will be more convenient to argue about pairs (S, w) instead.

Let $P(S, w)$ be the expected number of pivot steps in a call to RANDOM-FACET(S, w), and define

$$w^{(i)} := \mathbf{e}_i + A_i, \quad i \in [n].$$

The following equation is the basis of the analysis.

Lemma 5.2 *Let $|S| = m > 0$. With*

$$w^{(i,j)} := \mathrm{opt}(S \setminus \{j\}, w^{(i)}),$$

$$P(S, w^{(i)}) = \frac{1}{m} \sum_{j \in S} \Big(P(S \setminus \{j\}, w^{(i)}) + (1 + P(S, w^{(j)}))[w_j^{(i,j)} = 1] \Big). \tag{25}$$

The proof follows immediately from the description of RANDOM-FACET in Algorithm 3.3, together with the following observation: the value $w' = w^{(i,j)}$ computed from the first recursive call satisfies $w'_k = 0$ for all $k \in S \setminus \{j\}$ by Lemma 3.2 (i). Exactly if $w'_j = 1$,

we have a second recursive call with value $w'' = w' + A_j$. We do not necessarily have $w'' = w^{(j)}$ as suggested by the equation, but when we restrict both values to S, they agree. By (14), we then have $P(S, w'') = P(S, w^{(j)})$.

Define

$$\overline{P}(S) = \sum_{i \in S} P(S, w^{(i)}).$$

Using (25) and the fact that $P(S \setminus \{j\}, w^{(j)}) = P(S, w^{(j)})$ (j is fixed in $(S, w^{(j)})$, so Theorem 2.4 applies), we obtain

$$\overline{P}(S) = \frac{1}{m-1} \sum_{j \in S} \left(\overline{P}(S \setminus \{j\}) + (1 + P(w^{(j)}, S)) \sum_{i \in S} [w_j^{(i,j)} = 1] \right), \quad m > 1. \tag{26}$$

The claim now is that $w_j^{(i,j)} = 1$ for at most one i. First observe that $w_j^{(j,j)} = 0$ (we have $\mathrm{hi}(S, w^{(j)}) \leq j - 1$, which also holds for the pair $(S, w^{(j,j)})$). For $i \neq j$ we can argue as follows. By definition of $w^{(i,j)}$ we know that

(i) $w_k^{(i,j)} = 0$ for all $k \in S \setminus \{j\}$, and

(ii) $(A^{-1}(w^{(i)} - w^{(i,j)}))_k = 0$, for all $k \notin S \setminus \{j\}$.

The second condition holds because the vectors corresponding to the two values are in the same facet $(v, S \setminus \{j\})$. Condition (ii) implies $(A(S)^{-1}(w_S^{(i)} - w_S^{(i,j)}))_j = 0$. Using the definition of $w^{(i)}$ and property (i), one deduces that $w_j^{(i,j)}$ is equal to entry i in row j of $A(S)^{-1}$. By Theorem 5.1 (ii) (which also applies to $A(S)$), at most one of these entries is nonzero, and this proves the claim. Then, (26) implies

$$\overline{P}(S) \leq \frac{1}{m-2} \sum_{j \in S} \overline{P}(S \setminus \{j\}) + \frac{m}{m-2}, \quad m > 2.$$

If we let $\overline{P}(m) := \max_{|S|=m} \overline{P}(S)$ we get $\overline{P}(m) \leq (m/(m-2))(\overline{P}(m-1) + 1)$, for $m > 2$ and $\overline{P}(2) \leq 1$ (by directly inspecting the possible cases), from which we obtain

$$\overline{P}(m) \leq 3\binom{m}{2} - m, \quad m > 2. \tag{27}$$

To conclude the analysis we observe that

$$P(S, w) \leq \frac{1}{m} \sum_{j \in S} \left(P(S \setminus \{j\}, w) + 1 + P(S, w^{(j)}) \right),$$

for all start values w. With (27) and $P(m) := \max_{w, |S|=m} P(S, w)$ we get $P(0) = 0, P(1) = 1$ and

$$P(m) \leq P(m-1) + 1 + \frac{1}{m} \left(3\binom{m}{2} - m \right), \quad m > 2$$

from which

$$P(m) \leq \frac{3}{2}\binom{m}{2} + 2$$

follows. This gives the main result of this section.

Theorem 5.3 *If $A \in \mathcal{A}$ defines a linearly inducible Matoušek orientation, then the expected number of pivot steps in Algorithm* Random-Facet *is bounded by*

$$\frac{3}{2}\binom{n}{2} + 2 \approx \frac{3}{4}n^2,$$

for any start vertex v (start value w, respectively).

Lemma 5.1 shows that among the $2^{\binom{n}{2}}$ Matoušek functions, at most $n! \approx 2^{n \log n}$ define linearly inducible orientations. Here, $n!$ is the number of matrices $A^{-1} \in \mathcal{A}$ which have no more than one off-diagonal one-entry per row. We do not know whether all of them actually are linearly inducible. This might not seem plausible, because we have only exploited conditions on the 3-faces so far. On the other hand, these "local" conditions already imply a "global" sparsity condition.

6 Discussion

In this paper, I have proved upper and lower bounds on the expected performance of the Random-Facet simplex algorithm. For combinatorially inducible n-cube orientations, a tight bound of $\exp(\Theta(\sqrt{n}))$ holds, while for linearly inducible orientations, no lower bound better than $\Omega(n^2)$ is known. The situation is summarized in Figure 11. The major open problem is to resolve the question mark in the figure, i.e. to find an improved upper bound for linearly inducible cube orientations, or to extend the lower bound construction to the linear case.

The subexponential upper bound generalizes to linear programs whose feasible regions are not combinatorial cubes. Kalai, and independently Matoušek, Sharir and Welzl, have shown that linear programs with n variables and m constraints can be solved with an expected number of

$$\exp(O(\sqrt{n \log m}))$$

pivot steps [16, 19].

In case of $m = O(n)$, the log-factor can be eliminated, so that the bound for combinatorial cubes appears as a special case of a general result. However, currently this general result is only achieved with algorithms that are more complicated than Random-Facet [16, 6].

The algorithm of Matoušek, Sharir and Welzl is exactly the dual simplex variant corresponding to Random-Facet [11]. Like Random-Facet, it also works (and has even been designed) for an abstract problem class which generalizes linear programming. The class of *LP-type problems* subsumes many geometric optimization problems, and the algorithm in [19] is the fastest known algorithm for many concrete problems in the class.

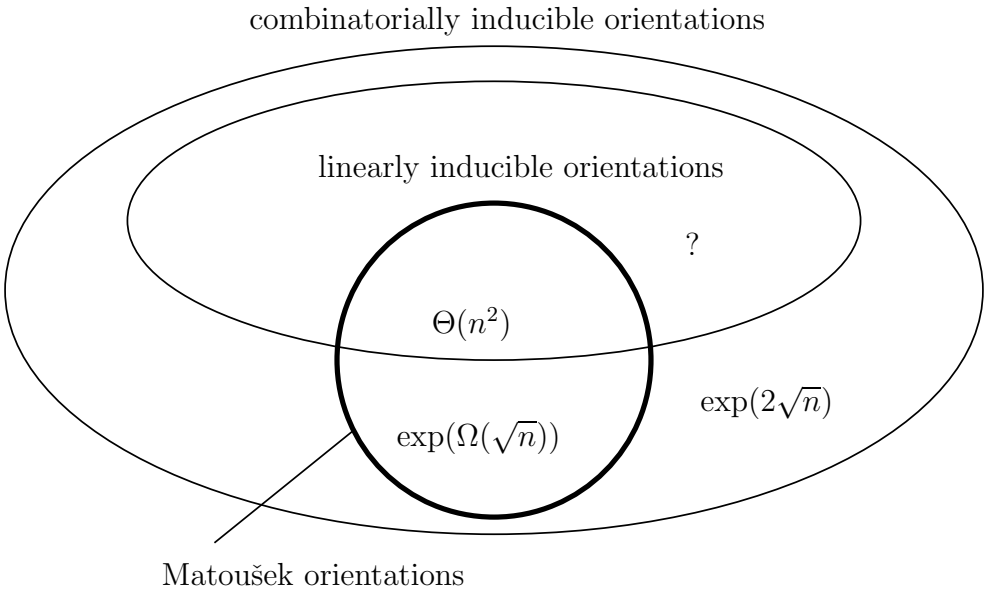

Figure 11: Results for Random-Facet on combinatorial cubes

Although LP-type problems are more general than combinatorially inducible polytope orientations, the idea that underlies the generalization is very similar in both cases.

There are other randomized variants of the simplex method which have been studied. The most natural one is the Random-Edge algorithm. Given a vertex of the feasible region, Random-Edge chooses the next vertex uniformly at random from all incident vertices with smaller objective function value. Despite its simplicity, very little is known for this rule. In particular, no nontrivial upper bound on its expected performance could be established so far. The known lower bounds do not rule out the possibility that Random-Edge is an expected polynomial-time simplex variant [8, 2].

Still other randomized rules have been proposed by Kalai [16] and recently by Joswig and Kaibel [15].

Even for the seemingly simple case of combinatorial cubes, the worst-case complexity of the simplex method is unresolved. In this paper, I have surveyed the known results for one particular randomized pivot rule, and even these results leave a huge gap between the upper and lower performance bounds. Still, the results demonstrate that randomization can help; moreover, further combinatorial properties of linear programming which have not been understood so far, might eventually lead to improved bounds.

Appendix

This section derives an explicit lower bound for the function $g(k)$ defined in Theorem 4.1. Fix some integer $t \in \{1, \dots, k\}$.

$$\begin{aligned} g(k)+1 &= \sum_{\ell=0}^{k} \frac{(1/2)^\ell}{\ell!^2} k(k-1)\cdots(k-\ell+1) \geq \sum_{\ell=0}^{k} \frac{(1/2)^\ell}{\ell!^2}(k-\ell+1)^\ell \\ &\geq \sum_{\ell=0}^{t-1} \frac{(1/2)^\ell}{\ell!^2}(k-t+1)^\ell = \sum_{\ell=0}^{t-1} \left(\frac{\sqrt{(k-t+1)/2}^\ell}{\ell!} \right)^2 \\ &\geq \frac{1}{t} \left(\sum_{\ell=0}^{t-1} \frac{\sqrt{(k-t+1)/2}^\ell}{\ell!} \right)^2, \end{aligned}$$

where the last inequality uses $\|x\|_1 \leq \sqrt{t}\|x\|_2$ for $x \in \mathbb{R}^t$. Via the well-known tail estimate for the exponential series,

$$\sum_{\ell=t}^{\infty} \frac{x^\ell}{\ell!} \leq 2\frac{x^t}{t!}, \quad x \leq \frac{t+1}{2},$$

we can further bound $g(k)$ as follows.

$$\begin{aligned} g(k)+1 &\geq \frac{1}{t} \left(e^{\sqrt{(k-t+1)/2}} - \sum_{\ell=t}^{\infty} \frac{\sqrt{(k-t+1)/2}^\ell}{\ell!} \right) \\ &\geq \frac{1}{t} \left(e^{\sqrt{(k-t+1)/2}} - \frac{\sqrt{(k-t+1)/2}^t}{t!} \right)^2. \end{aligned}$$

This requires $\sqrt{(k-t+1)/2} \leq (t+1)/2$ which holds if $t = \lfloor 2\sqrt{k} \rfloor$. With Stirling's formula, we get

$$\begin{aligned} \frac{\sqrt{(k-t+1)/2}^t}{t!} &\leq \frac{e^t}{\sqrt{2\pi t}} \left(\frac{\sqrt{(k-t+1)/2}}{t} \right)^t \\ &= \frac{e^t}{\sqrt{2\pi t}} \left(1 + \frac{\sqrt{(k-t+1)/2} - t}{t} \right)^t \leq \frac{1}{\sqrt{2\pi t}} e^{\sqrt{(k-t+1)/2}}. \end{aligned}$$

This implies

$$g(k)+1 \geq \frac{1}{t} \left(1 - \frac{1}{\sqrt{2\pi t}} \right)^2 e^{2\sqrt{(k-t+1)/2}} \geq \frac{1}{2\sqrt{k}} \left(1 - \frac{1}{\sqrt{4\pi}} \right)^2 e^{\sqrt{2}(\sqrt{k}-1)} \geq \frac{1}{16\sqrt{k}} e^{\sqrt{2k}}.$$

Compared to the true order of magnitude of $g(k)$, this estimate is off by a factor of $\sqrt[4]{k}$. One can show that

$$g(k) = \Theta\left(\frac{e^{\sqrt{2k}}}{\sqrt[4]{k}} \right),$$

see [20].

References

[1] N. Amenta and G. M. Ziegler. Deformed products and maximal shadows. In J. Chazelle, J.B. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational geometry*, Contemporary Mathematics. Amer. Math. Soc., 1998.

[2] A. Z. Broder, M. E. Dyer, A. M. Frieze, P. Raghavan, and E. Upfal. The worst-case running time of the random simplex algorithm is exponential in the height. *Inform. Proc. Letters*, 56(2):79–81, 1995.

[3] V. Chvátal. *Linear Programming*. W. H. Freeman, New York, NY, 1983.

[4] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton, NJ, 1963.

[5] L. Finschi, K. Fukuda, and H.-J. Lüthi. Towards a unified framework for randomized pivoting algorithms in linear programming. In *Operations Research Proceedings*, pages 113–122. Springer, 1998.

[6] B. Gärtner. *Randomized Optimization by Simplex-Type Methods*. PhD thesis, Freie Universität Berlin, 1995.

[7] B. Gärtner. Combinatorial linear programming: Geometry can help. In *Proc. Second Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM'98)*, volume 1518 of *Lecture Notes in Computer Science*, pages 82–96. Springer-Verlag, 1998.

[8] B. Gärtner, M. Henk, and G. M. Ziegler. Randomized simplex algorithms on Klee-Minty cubes. *Combinatorica*, 18(3):349–372, 1998.

[9] B. Gärtner and V. Kaibel. Abstract objective function graphs on the 3-cube – a characterization by realizability. Technical Report TR 296, Dept. of Computer Science, ETH Zürich, 1998.

[10] B. Gärtner, József Solymosi, Falk Tschirschnitz, Pavel Valtr, and Emo Welzl. One line and n points. In *Proc. 33rd annual ACM Symposium on Theory of Computing (STOC)*, 2001, to appear.

[11] M. Goldwasser. A survey of linear programming in randomized subexponential time. *ACM-SIGACT News*, 26(2):96–104, 1995.

[12] K. Williamson Hoke. Completely unimodal numberings of a simple polytope. *Discr. Applied Math.*, 20:69–81, 1988.

[13] F. Holt and V. Klee. A proof of the strict monotone 4-step conjecture. In J. Chazelle, J.B. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational geometry*, Contemporary Mathematics. Amer. Math. Soc., 1998.

[14] R. G. Jeroslow. The simplex algorithm with the pivot rule of maximizing criterion improvement. *Discr. Math.*, 4:367–377, 1973.

[15] M. Joswig and V. Kaibel. Randomized simplex algorithms and random cubes. Manuscript.

[16] G. Kalai. A subexponential randomized simplex algorithm. In *Proc. 24th annu. ACM Symp. on Theory of Computing.*, pages 475–482, 1992.

[17] G. Kalai. Linear programming, the simplex algorithm and simple polytopes. *Math. Programming*, 79:217–233, 1997.

[18] V. Klee and G. J. Minty. How good is the simplex algorithm? In O. Shisha, editor, *Inequalities III*, pages 159–175. Academic Press, 1972.

[19] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16:498–516, 1996.

[20] J. Matoušek. Lower bounds for a subexponential optimization algorithm. *Random Structures & Algorithms*, 5(4):591–607, 1994.

[21] J. Mihalisin and V. Klee. Convex and linear orientations of polytopal graphs. *Discrete Comput. Geom.*, 24:421–436, 2000.

[22] N. Zadeh. What is the worst case behavior of the simplex algorithm? Technical Report 27, Dept. Operations Research, Stanford University, 1980.

[23] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, 1994.