Halfplane Intersection and Linear Programming

Dan Halperin, Tel Aviv University

Figures and pseudocode taken from:

CGAA: Chapter 4 in Computational Geometry Algorithms and Applications, de Berg et al

Overview

- The casting problem
- Half-plane intersection
- Linear programming (LP) in low dimensions
- Minimum enclosing disc
- The casting problem, take II (later on in the semester)

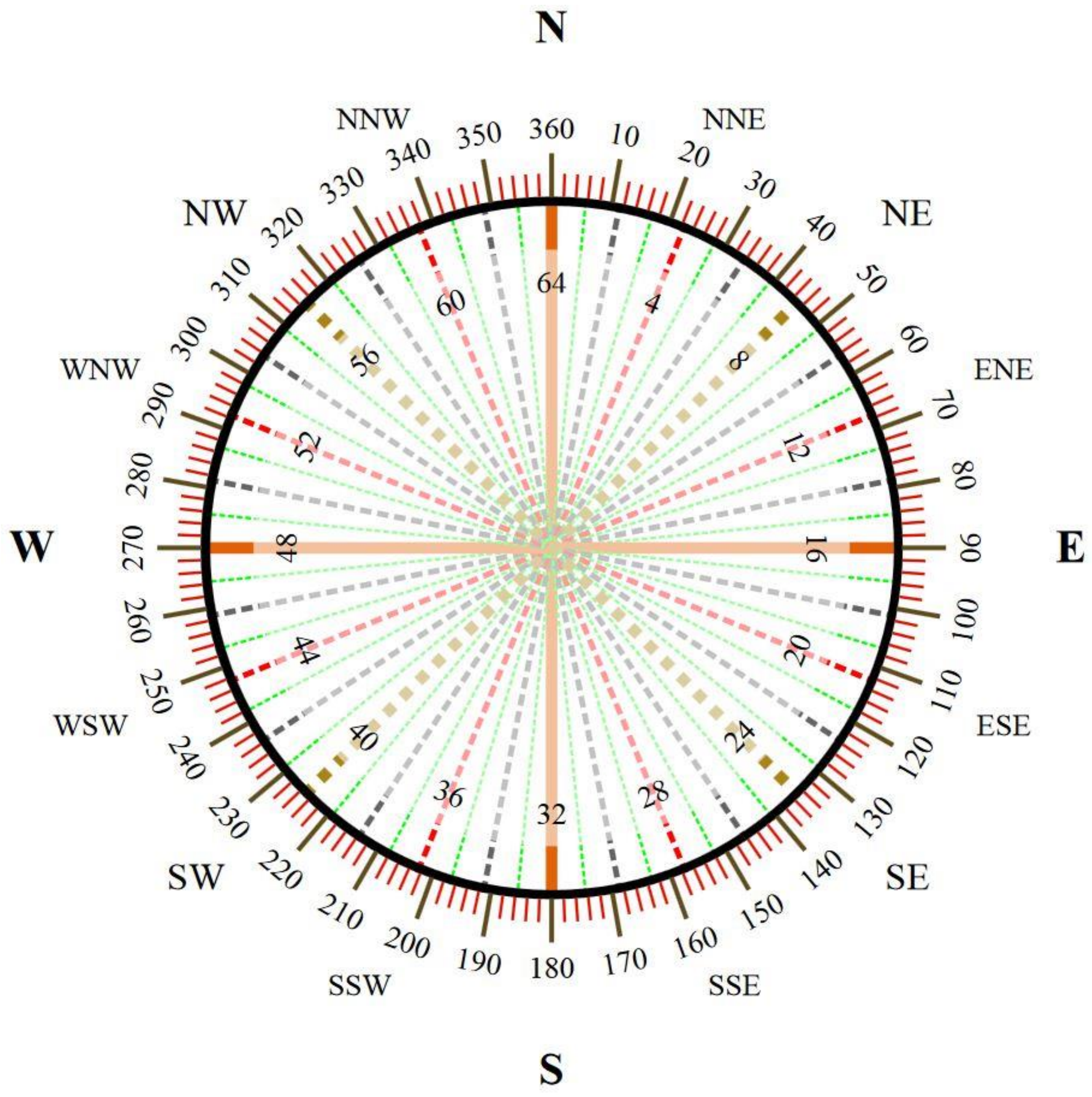
Credits

- CGAA: Chapter 4 in Computational Geometry Algorithms and Applications, Linear Programming, by de Berg et al
- L5vK: Lecture 5 in Computational Geometry, Casting a polyhedron, by Marc van Kreveld

Slides, Part I

- Lecture 5 in Computational Geometry, Casting a polyhedron, by Marc van Kreveld (Slides4a):
  - The casting problem
  - Half-plane intersection
  - Linear programming (LP) in low dimensions
- Supplements, here:
  - The space of directions
  - Probabilistic analysis of LP2D
  - Unbounded LP2D
  - LP3D

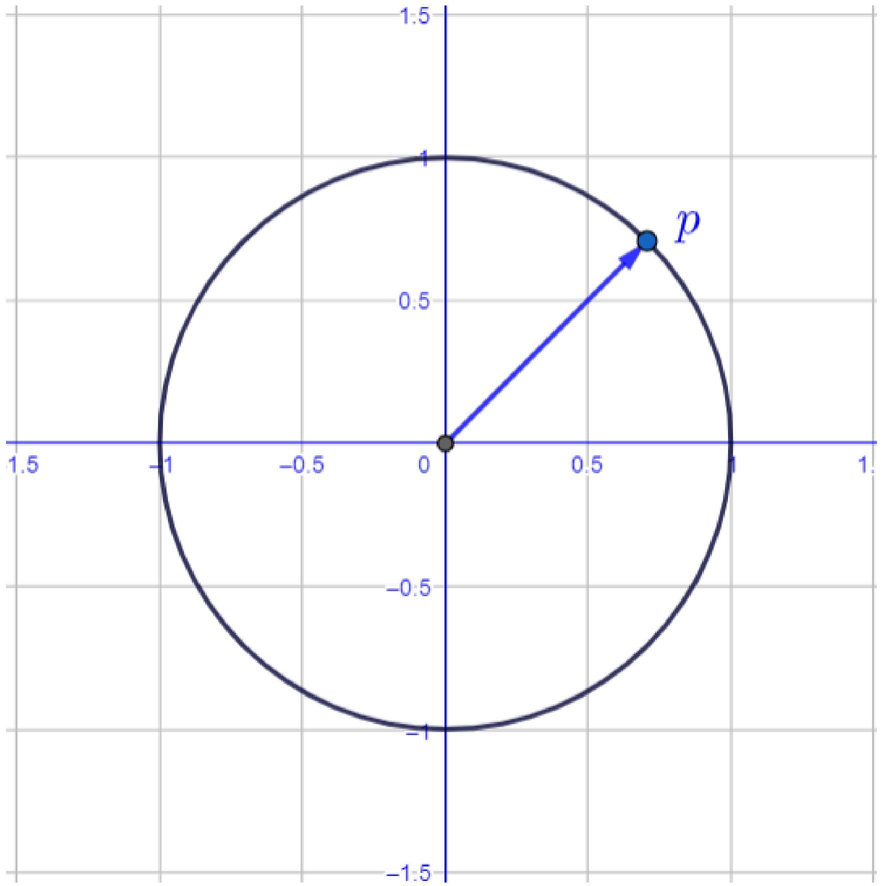
The space of directions



[blog.weatherflow.com]

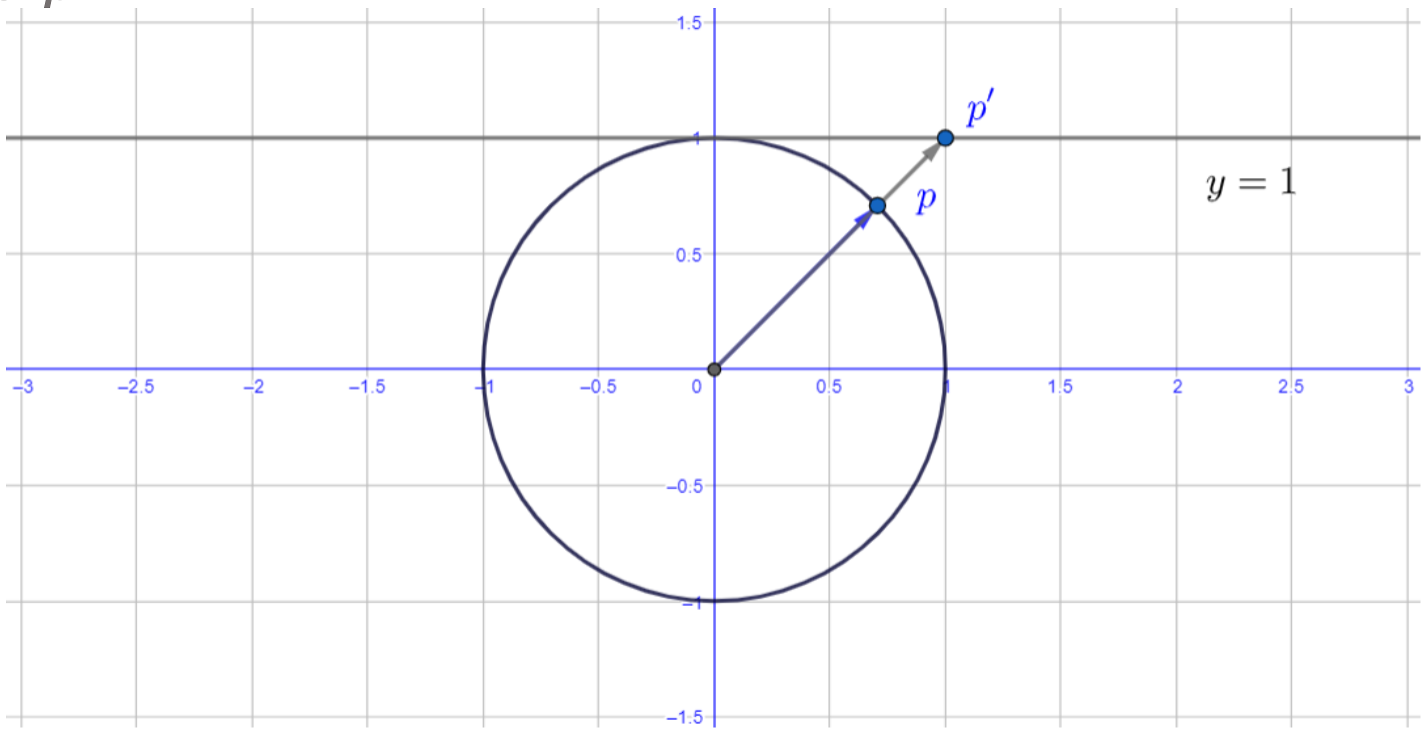
Representing directions in the plane

- Using the unit circle S^1
- The point p represents the vector from the origin to p



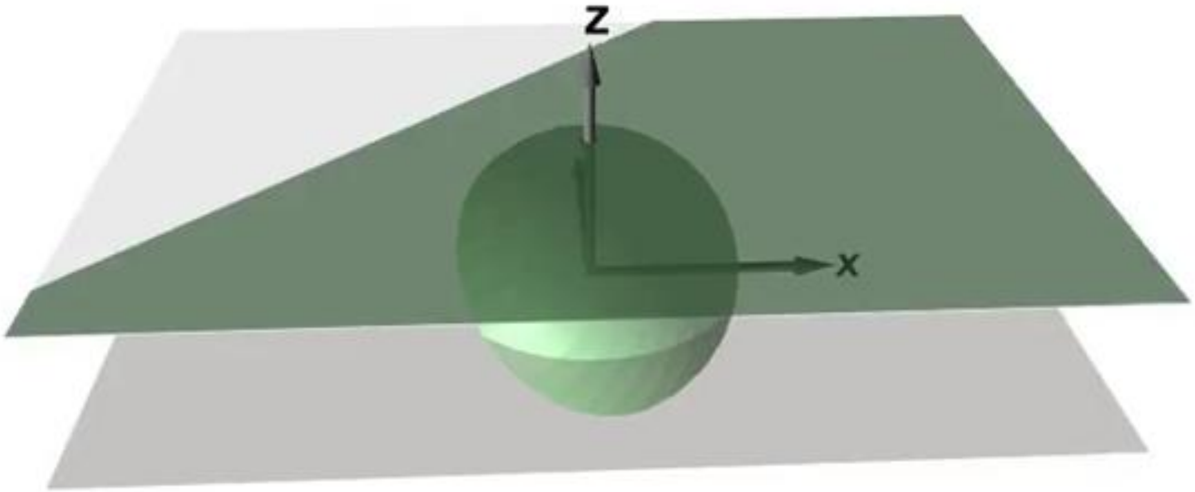
Representing upward directions

- Using the line $y = 1$
- The point p' represents the vector from the origin to p'



Representing directions in (3-)space

- Using the unit sphere S^2
- Upward directions: the plane $z = 1$



LP2D randomized, bounded

Analysis

Algorithm 2DRANDOMIZEDBOUNDEDLP($H, \vec{c}, m_1, m_2$)

Input. A linear program $(H \cup \{m_1, m_2\}, \vec{c})$, where H is a set of n half-planes, $\vec{c} \in \mathbb{R}^2$, and m_1, m_2 bound the solution.

Output. If $(H \cup \{m_1, m_2\}, \vec{c})$ is infeasible, then this fact is reported. Otherwise, the lexicographically smallest point p that maximizes $f_{\vec{c}}(p)$ is reported.

1. Let v_0 be the corner of C_0.
2. Compute a *random* permutation $h_1, \ldots, h_n$ of the half-planes by calling RANDOMPERMUTATION($H[1 \cdots n]$).
3. **for** $i \leftarrow 1$ **to** n
4.   **do if** $v_{i-1} \in h_i$
5.   **then** $v_i \leftarrow v_{i-1}$
6.   **else** $v_i \leftarrow$the point p on ℓ_i that maximizes $f_{\vec{c}}(p)$, subject to the constraints in H_{i-1}.
7.   **if** p does not exist
8.   **then** Report that the linear program is infeasible and quit.
9. **return** v_n

Running time analysis

- Random permutation: $O(n)$
- Line 5, the `easy' case, $v_{i-1} \in H_i$: $O(1)$, which is $O(n)$ over all iterations
- It remains to estimate the running time of all the `hard' cases
- Let X_i be the indicator random variable, which is 1 if Step i is a hard case and 0 otherwise
- The running time of all the hard cases: $\sum_{i=1}^{n} O(i)X_i$
- Randomized analysis—the expected value of the running time $E(\sum_{i=1}^{n} O(i)X_i)$

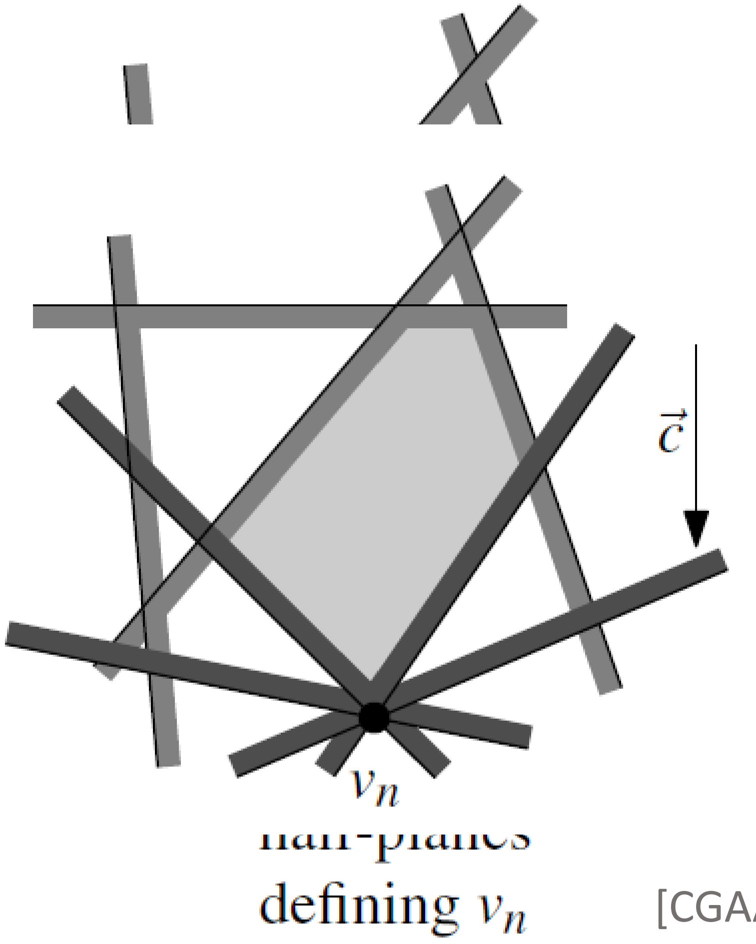
- In what follows, we assume that the first two half-planes h_1, h_2 are fixed, so our randomized analysis focuses on steps $i = 3, \ldots, n$

Running time analysis, cont'd

- Randomized analysis—the expected value of the running time $E(\sum_{i=3}^{n} O(i)X_i)$
- Using linearity of expectation, $E(\sum_{i=3}^{n} O(i)X_i) = \sum_{i=3}^{n} O(i)E(X_i)$
- $E(X_i) = ?$
- The expected value of an indicator random variable is the probability that $X_i = 1$
- Let's start with X_n

The last step

- What is the probability that $X_n = 1$?
- $\Pr[X_n = 1] \leq \frac{2}{n-2}$
- Why at most?
  - more than two lines meet at v_n
  - v_n is defined by h_1 or h_2



[CGAA]

The ith step

- What is the probability that $X_i = 1$?
- Let's first assume that the set of half-planes h_3 ... h_i is fixed
- Then the previous argument holds and the probability is $\leq \frac{2}{i-2}$
- But every subset of $i - 2$ half-planes has the same probability to be those h_3 ... h_i
- Hence, $\Pr[X_i = 1] \leq \frac{2}{i-2}$

Summary

- $E(\sum_{i=3}^{n} O(i)X_i) = \sum_{i=3}^{n} O(i)E(X_i) \leq \sum_{i=3}^{n} O(i)\frac{2}{i-2} = O(n)$
- Together with the other costs, the algorithm runs in expected $O(n)$ time

LP2D unbounded

Possible outcomes of LP2D

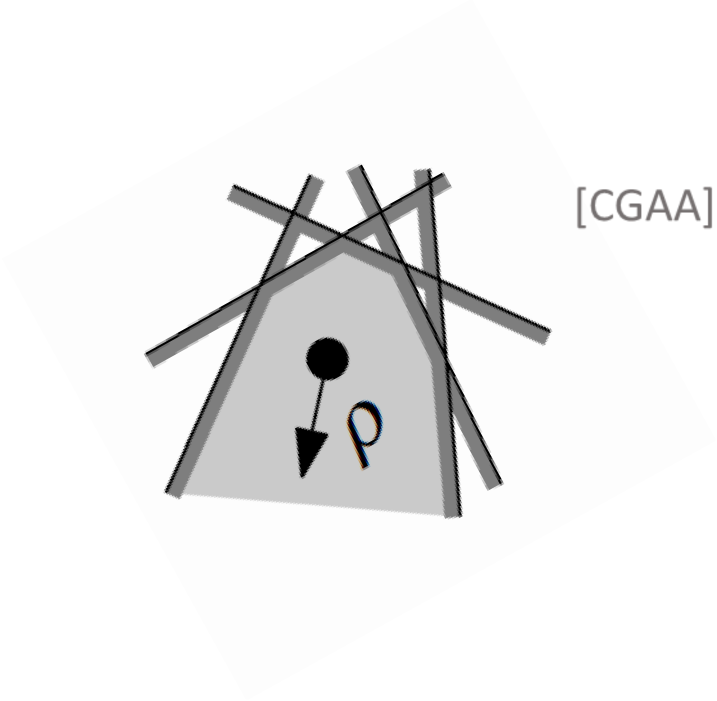
- We now consider also the case where the program may be unbounded
- Possible output:
  - The optimal (maximal) solution, as before
  - The program is infeasible, as before
  - The program is unbounded: a ray along which the solution gets arbitrarily large values

Overall scheme for general LP2D

- We will start by testing for the unbounded case, with possible outcomes:
  - The program is infeasible, stop
  - The program is unbounded, with the desired ray, stop
  - The program is bounded, together with two witness half-planes h_1, h_2 for the boundedness, continue to the previous, bounded, procedure
- Notice that our guarantee that the program is bounded does not preclude the case that it is infeasible, which will be found by the previous (bounded) procedure

Notation

- LP2D$(H, \vec{c})$
- $H = \{h_1, \dots, h_n\}$
- $\vec{c}$ is the objective vector
- The LP is unbounded if there is a ray ρ fully contained in the feasible region and such that the objective function grows arbitrarily as we proceed along ρ away from its terminus p
- We denote the ray's direction by $\vec{d}$
- $\rho = \{p + \lambda\vec{d}\colon \lambda > 0\}$



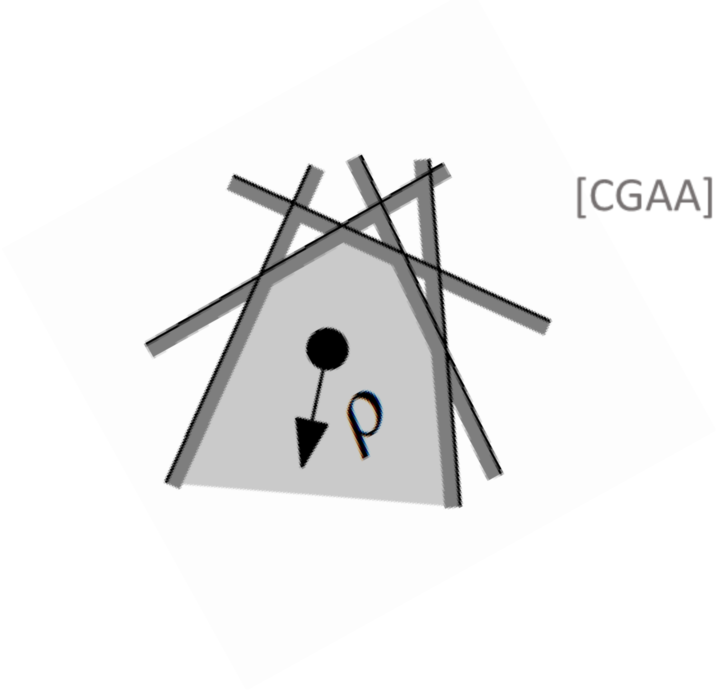
[CGAA]

Notation, cont'd

- For a half-plane $h \in H$, $\vec{\eta}\,(h)$ is the normal to the line defining the half-plane and pointing into the feasible region of h

Necessary conditions for unboundedness

$\rho = \{p + \lambda\vec{d}\colon \lambda > 0\}$



[CGAA]

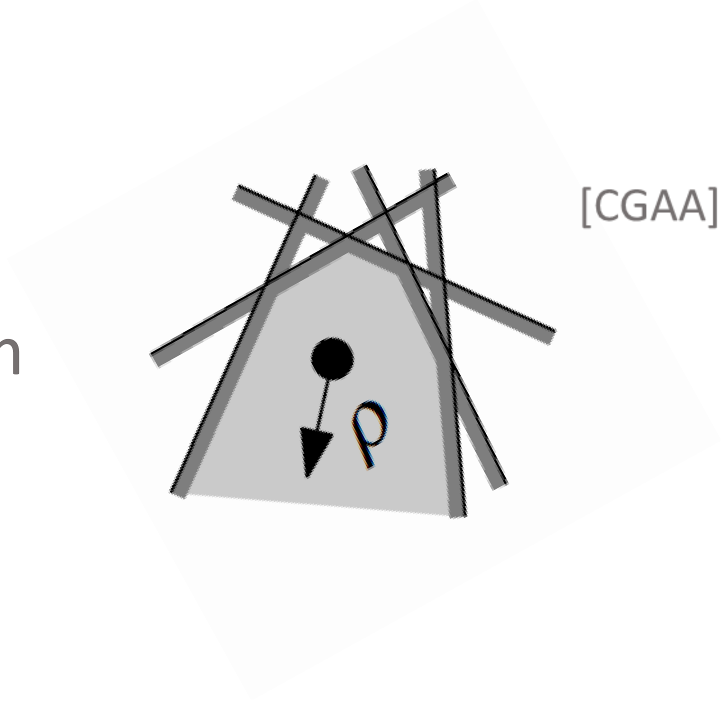
- $\vec{d} \cdot \vec{c} > 0$
- for each half-plane $h \in H$, $\vec{\eta}\,(h) \cdot \vec{d} \geq 0$

Let $H' = \{h \in H\colon \vec{\eta}\,(h) \cdot \vec{d} = 0\}$, the we also require the following boundary condition:

- the linear program $(H', \vec{c})$ is feasible

Conditions for unboundedness

Claim: $(H, \vec{c})$ is unbounded iff there is a direction $\vec{d}$ with $\vec{d} \cdot \vec{c} > 0$ such that for each $h \in H$, $\vec{\eta}\,(h) \cdot \vec{d} \geq 0$, and the linear program $(H', \vec{c})$ is feasible



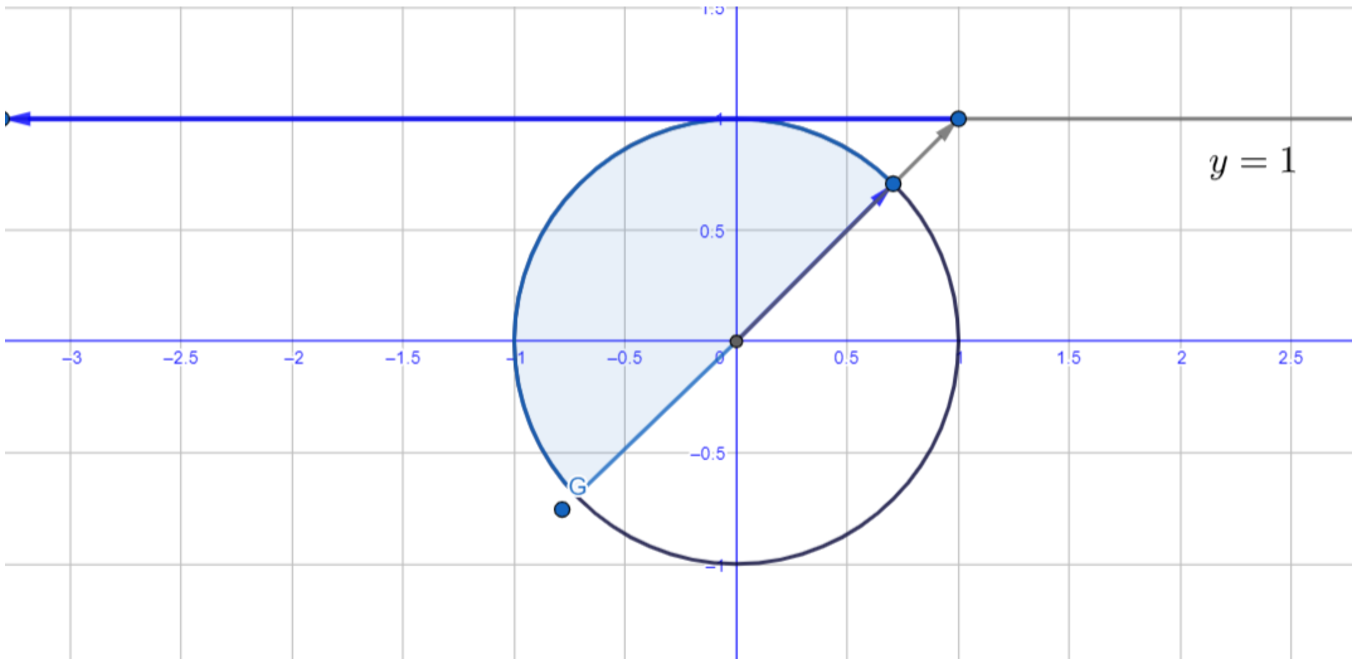
[CGAA]

- We showed that the conditions are necessary. We will show that they are sufficient by constructing the witness ray ρ

Constructing the ray ρ

- Assume $\vec{c} = (0,1)$
- Then the ray must be directed upward, and we can represent the possible directions $\vec{d}$ by the line $y = 1$, $\vec{d}$ =(d_x,1)
- Every constraint of the form $\vec{\eta}\,(h) \cdot \vec{d} \geq 0$ becomes a half-line, ray, on the line $y = 1$
- The valid directions:

The intersection of all these rays

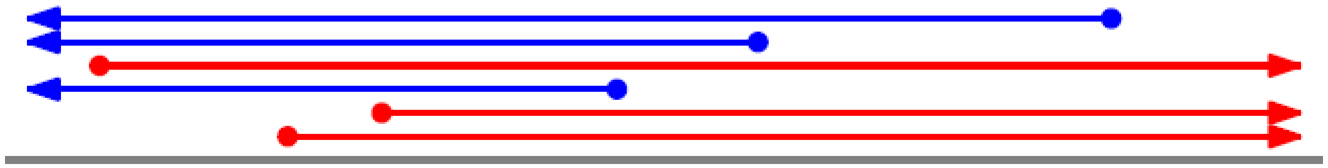


The valid directions

- Recall

The common intersection of a set of half-lines in 1D:

- Determine the endpoint p_l of the rightmost left-bounded half-line
- Determine the endpoint p_r of the leftmost right-bounded half-line
- The common intersection is $[p_l, p_r]$ (can be empty)

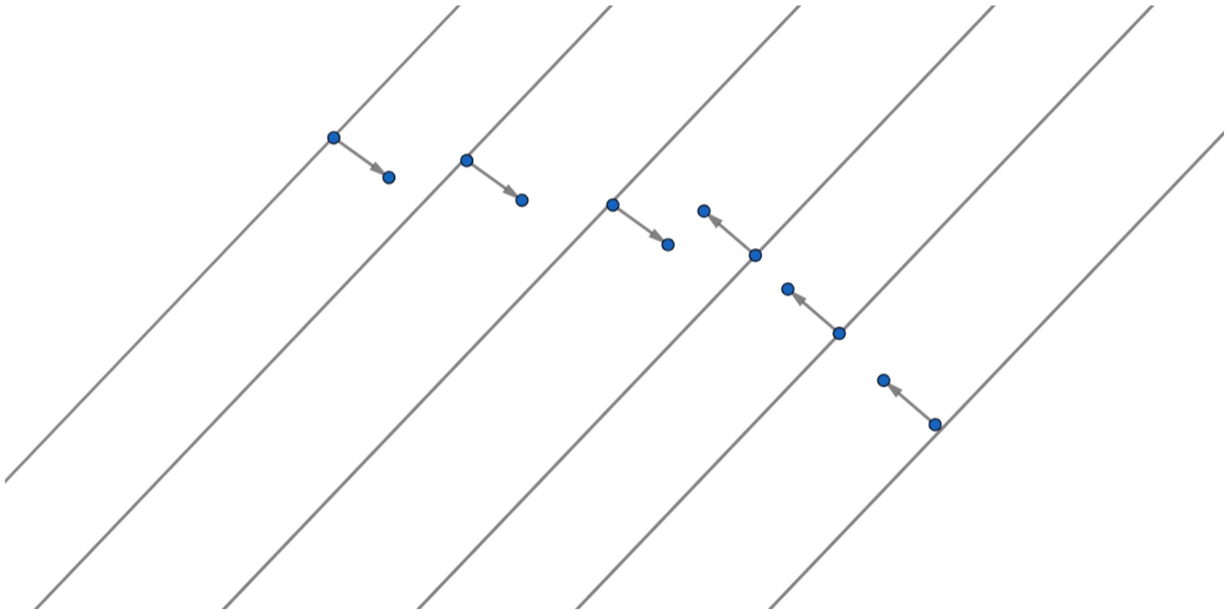


[L5vK]

- Assume first that the intersection is not empty
- We take the direction at the left endpoint of the interval (if it is bounded only on the right, we take the right endpoint)
- Denote it by $\vec{d}$
- We first need to check that $\vec{d}$ is valid (relevant only if the interval is a single point)
- The validity check is in the original plane, where we aim to construct the ray ρ
- The test: is the linear program $(H', \vec{c})$ is feasible

Is the linear program $(H', \vec{c})$ is feasible?

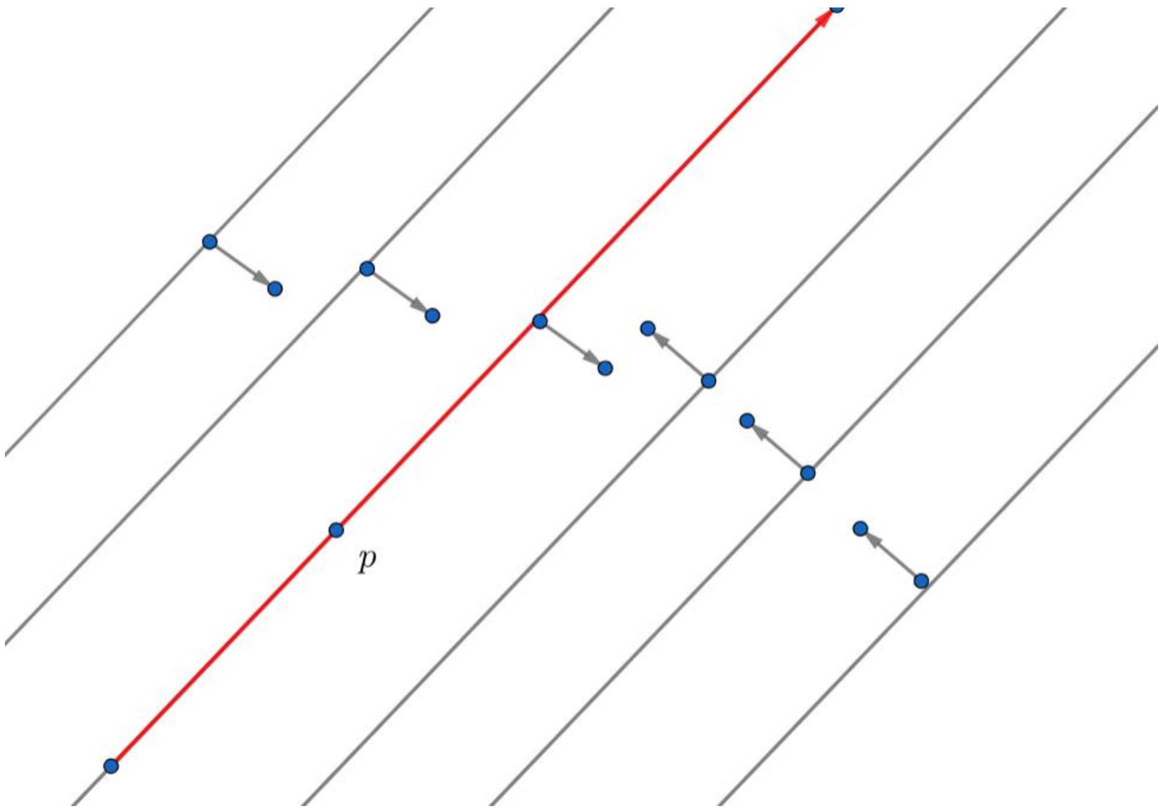
- Recall: $H' = \{h \in H: \vec{\eta}\,(h) \cdot \vec{d} = 0\}$



- Outcome 1: If infeasible, report that the original LP infeasible and stop

If $(H', \vec{c})$ is feasible

- Outcome 2: Stick to the left wall of the feasible region and construct the ray there

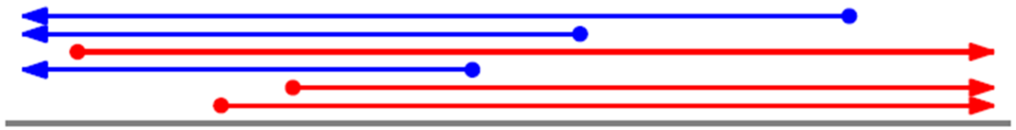


If no valid direction $\vec{d}$ exists

- Namely $p_l > p_r$, then the half-planes inducing p_l and p_r are witnesses to the boundedness of the LP
- Outcome 3: Go to the bounded procedure and start with these two half-planes as h_1 and h_2

The common intersection of a set of half-lines in 1D:

- Determine the endpoint p_l of the rightmost left-bounded half-line
- Determine the endpoint p_r of the leftmost right-bounded half-line
- The common intersection is $[p_l, p_r]$ (can be empty)



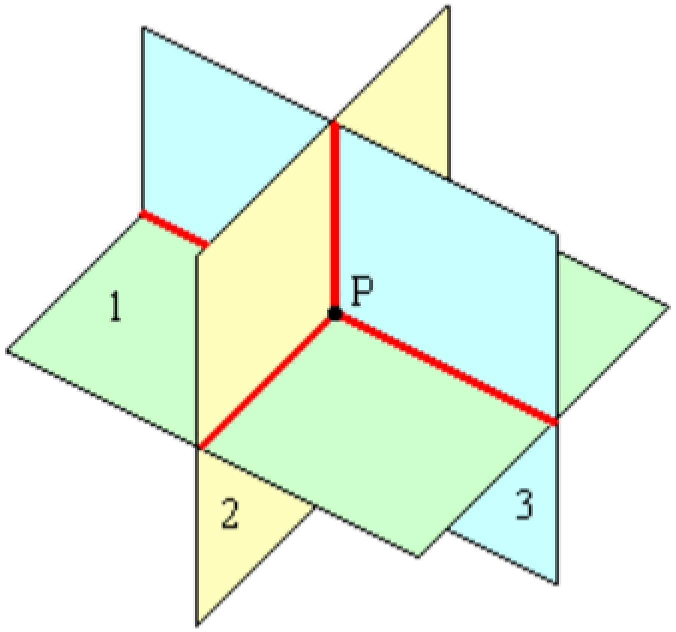
LP3D

The input

- LP3D$(H, \vec{c})$
- $H = \{h_1, \ldots, h_n\}$, half-spaces
  - h_i is bounded by the plane g_i
- $\vec{c}$ is the objective vector
- Assume we have already run LP3DUnbounded, and obtained three witnesses to the boundedness of LP3D$(H, \vec{c})$
- Let's rename these three half-space h_1, h_2, h_3
- They define an optimum v_3

The incremental step

- We now add h_4
- If $v_3 \in h_4$ then $v_4 := v_3$
- Else, v_4 lies on g_4
- How do we find v_4?



[gdbooks/3dcollisions]

THE END