Linear Programming

Goal is to find a "good" point \( \mathbb{R}^d \),
\( (x_1, \ldots, x_d) \)

Subject to linear inequalities called constraints
\[
a_1 x_1 + a_2 x_2 + \ldots + a_d x_d = b
\]

input
Constraints form the feasible polytope.

Want feasible point maximizing a linear objective function

\[ c_1 x_1 + \ldots + c_d x_d \]

\( c = (c_0, \ldots, c_d) \) is a vector in \( \mathbb{R}^d \).

Want to max \( c \cdot x \).
Optimal solution lies at a polytope vertex

An LP problem:

\[
\begin{align*}
\text{max} & \quad c_1 x_1 + \ldots + c_d x_d \\
\text{s.t.} & \quad a_{i,1} x_1 + \ldots + a_{i,d} x_d \leq b_i, \\
& \quad a_{2,1} x_1 + \ldots + a_{2,d} x_d \leq b_2, \\
& \quad \vdots \\
& \quad a_{n,1} x_1 + \ldots + a_{n,d} x_d \leq b_n
\end{align*}
\]
-or-

\[
\begin{align*}
\max_{x} & \quad c^T x \\
\text{s.t.} & \quad A x \leq b \\
\text{s.t.} & \quad c^T x \text{ are d-dim.} \\
& \quad b \text{ is u.dim.} \\
& \quad A \text{ is n} \times l \text{ matrix}
\end{align*}
\]

3 Cases:

- Feasible: an optimal point exists at some vertex
- Infeasible: feasible polytope is empty
- Unbounded: polytope unbounded in direction of \( c \) so no finite opt. sol. exists
Say we're given halfspaces $H = \{h_1, \ldots, h_n \in \mathbb{R}^d \}$ and an objective vector $c$ in $\mathbb{R}^d$. Seidel ['91] : incremental construction

Assume solution is bounded by some $\{h_1, \ldots, h_n \in \mathbb{R}^d \} \subseteq H$

Find in $O(n)$ time (see book)
For \( d \in i \in n, \) let
\[
H_i := \{ h, \ldots, h \} \cup \hat{w}
\]
\( \hat{w} \) is optimal for \( H_i \).

We want to find \( v_n \).

Suppose we know \( v_{n-1} \) and want \( v_n \).

**Case 1:** \( v_n \in h \). We're done!
\[
\begin{align*}
V_n &= V_{n-1} \\
v_n &= v_n
\end{align*}
\]

**Case 2:** \( v_n \notin h \). Need to find new \( v_n \neq v_{n-1} \).
Lemma: In Case 2, $v$ lies in boundary $l_n$ of $h_n$.

Proof: $v_{n-1}v_n$ crosses $l_n$ at some point $p$.

$p$ is feasible for $H_n$.

$c.v_{n-1} \prec c.p \preceq c.v_n$
So, project $c$ onto $l_n$ to get a $(d-1)$-dim objective.

Find intersection of $\{h_1, \ldots, h_{n-1}\}$ with $l_n$ to get $(d-1)$-dim half spaces.

Project it all to $\mathbb{R}^{d-1}$.

Recourse & lift solution back to $\mathbb{R}^d$.

$O(dn)$ time to project.
Using Gaussian elimination.
$O(n)$ time alg for $d=1$. 
Analysis: $O(n^2)$ in $\mathbb{R}^2$.

$W_d(n)$: running time in $\mathbb{R}^d$ with $n$ constraints

$$W_d(n) = \begin{cases} 
1 & \text{if } n = 1 \\
 n & \text{if } d = 1 \\
 W_d(n-1) + d + dn + W_{d-1}(n-1) & \text{otherwise}
\end{cases}$$

$$= O(n^d)$$
Randomized Incremental Linear Programming

**Input:** A set $H = \{h_1, \ldots, h_n\}$ of halfspaces in $\mathbb{R}^d$, such that the first $d$ define an initial feasible vertex $v_d$, and the objective vector $c$.

**Output:** The optimum vertex $v$ or an error status indicating that the LP is infeasible.

1. If $d = 1$, solve the LP by brute force in $O(n)$ time.
2. Find an initial subset of $d$ halfspaces $\{h_1, \ldots, h_d\}$ that provide a bounded solution $v_d$. (If no such set exists, report that the LP is unbounded.)
3. Randomly select a halfspace from the remaining set $\{h_{d+1}, \ldots, h_n\}$. Call this $h_n$. Recursively solve the LP on the remaining $n-1$ halfspaces, letting $v_{n-1}$ denote the result. (If the LP is infeasible, then return this.)
4. If ($v_{n-1} \in h_n$) return $v_{n-1}$ as the final answer.
5. Otherwise, intersect $\{h_1, \ldots, h_{n-1}\}$ with the $(d-1)$-dimensional hyperplane $\ell_n$ that bounds $h_n$ and project onto $\mathbb{R}^{d-1}$. Let $c'$ be the projection of $c$ onto $\ell_n$ and then onto $\mathbb{R}^{d-1}$. Recursively solve the resulting $(d-1)$-dimensional LP with $n-1$ halfspaces. (If the LP is infeasible, then return this.) Project the optimal vertex back onto $\ell_n$, and return this point.
Treat running time as a random variable and bound its expectation.

So what is average time over all (n-d)! permutations?

Say d=2...

Expectation is linear:

Expectation of a weighted sum of r.v.s is the weighted sum of their expectations.
So find expected time to process each $h_i$.

Say Case 1 takes 1 unit of time, & Case 2 takes $\dot{w}$ units.

$X_{\dot{w}}$: time to process $h_{\dot{w}}$

$p_{\dot{w}}$: $\text{Prob}[v_{\dot{w}} \neq v_{\dot{w}-1}]

\begin{align*}
E[X_{\dot{w}}] &= p_{\dot{w}} \cdot \dot{w} + (1-p_{\dot{w}}) \cdot 1 \\
&= p_{\dot{w}} \cdot \dot{w} + 1
\end{align*}$
"Backwards analysis":

There is some subset \( \{ h_3, \ldots, h_B \} \). Fix any arbitrarily bad subset of them.

Any order of them is equally likely.

\( V_i \) lies at intersection of \( \Xi \) half-planes.

\( V_i \not\leq V_i' \) iff we add one of those 2 last
\[ \rho_{\hat{w}} = \frac{2}{\hat{w}} \]

So, \( \mathbb{E}[X_{\hat{w}}] \leq \rho_{\hat{w}} \cdot \hat{w} + 1 \)

Expected \( \leq 3 \)

Total running time:
\[ n \cdot \mathbb{E}[X_{\hat{w}}] \leq 0(n) \]
\[ \hat{w} = 3 \]
$T_d(n)$: expected time for $n$ constraints in $\mathbb{R}^d$

$T_d(1) = 1$ \quad $T_d(n) = n$

$T_d(n) = T_d(n-1) + d + \sum_{i=1}^{n-1} p_i \cdot (d \cdot n + T_d(n-1))$

$p_n \leq d/n$

$T_d(n) \leq O(d! \cdot n)$

Prob. exceed expectation by a factor of $6 \leq O((1/c)^{bd})$ for some constant $c$. 