Linear Programming

A powerful tool to model and solve optimization problems.
- useful both in theory and practice
- efficient in theory and practice.

Like flows, the applications are not always obvious. So we'll see how to model problems as LPs.
(and a bit about how LPs are solved).

Here's an example.

maximize $5x_1 + x_2$

s.t.

$-1 \leq x_1 + x_2 \leq 7$

$-2x_1 + 3x_2 \geq 4$

$x_1 \geq 0, x_2 \geq 0$.

So among all the points in the feasible region (the red region) which one maximizes $(5x_1 + x_2)$?

This is an LP. maximize/minimize a linear function (objective)

subject to linear inequality constraints.

(over the reals)
Reminder: if \( x_1, x_2, \ldots, x_n \) real valued variables, \( X = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \) vector.

- \( l(x) \) is a \underline{linear function} if \( l(x+y) = l(x) + l(y) \) for \( x, y \in \mathbb{R}^n \).

\[ l(x) = a_0 + a_1 x_1 + a_2 x_2 \ldots + a_n x_n \] for \( a_0, a_1, \ldots, a_n \in \mathbb{R} \).

- An LP: \( \min / \max \langle c, x \rangle \) \( \text{s.t.} \) \( \langle a^{(i)}, x \rangle \geq b_i \)

\[ \langle a^{(i)}, x \rangle \geq b_i \]

\[ \langle a^{(m)}, x \rangle \geq b_m \]

Inequalities

\( \text{not \ strict, } > \text{ not allowed} \)

\( \geq \text{ not allowed, only } \geq \) and \( \leq \).

Non Examples: \( \max x_1^2 \) or \( \max |x_1| + 5|x_2| \)

or \( \max \min (x_1, x_2) \)

Also constraints \( x_1 > 0 \) or \( \underline{\text{non-linear function}} \geq b \)

\( \text{eg } x_1^2 \geq 5 \text{ etc.} \)

\( \text{non linear? not linear prog.} \)
Our goals:

1. See how to model problems as LPs.
   - Some not so obvious

2. Then basic geometric / algebraic ideas behind solving LPs.
   - Simplex algo (very popular, used everywhere)
     - (almost) not an exaggeration.
     - Built into Excel, say.
   - Ellipsoid
     - (polynomial but not used much outside theory)
   - Interior point
     - (fast in both theory and practice)
   - Multiplicative weights
     - Leads into next topics - convex optimization and online algorithms

3. Along the way see duality
   - Which underlies the beautiful structure of LPs.
   - How do you prove a solution is optimal?
LPs: very powerful modeling language.
Captures many problems we have seen.
+ Shortest path
+ Min Spanning Tree
+ Max Flow.
+ Min Cost Max Flow.
All can be solved by LPs!

Of course: General LP solvers may give slower algorithms.

Also: in general we may not get the integer solutions
unless we are careful.

LP solver: given an LP, returns a solution that is optimal.

Notahin: solution is feasible if it satisfies all constraints.
Solution is optimal if feasible and has best objective function.

We will see some examples in the next lecture.
- Shortest path.
- Some scheduling / routing problems
- "diet" problem.
Examples:

1. **Max flow in** \( G = (V, E) \) from \( s \) to \( t \).

   (the definition is almost a linear program itself!)

   \[
   \begin{align*}
   \text{Constraints} & : \quad \sum_{v \in E} f_{sv} \\
   \text{max} \quad \sum_{v \in E} f_{sv} \\
   \text{objective} & : \quad \sum_{u \in E} f_{uv} = \sum_{w \in E} f_{uw} \quad \forall v \\
   \text{constraints} & : \quad f_{uv} \geq 0 \quad \forall \text{edges } (u,v) \in E \\
   & \quad f_{uv} \leq \text{cap}_{uv} \quad \forall \text{edges } (u,v) \in E
   \end{align*}
   \]

   Assume as always: no edges into \( s \), out of \( t \).

   \[ \forall \text{vertices } v \neq s, t. \]

   Variables: \( f_{uv} : (u,v) \in E \quad \text{one per edge}. \)

2. Examples in the 451 notes, (do those first)

3. **Shortest path length from** \( s \) to \( t \).

   Min cost max flow:

   - first write an LP as in 1, find the max flow value, \( F^* \)
   - then write another LP.

   \[
   \begin{align*}
   \sum_v f_{sv} &= M \\
   \sum_u f_{uv} &= \sum_w f_{uw} \quad \forall v \\
   0 &\leq f_{uv} \leq \text{cap}_{uv} \\
   \min \sum_cost_{uv} f_{uv}
   \end{align*}
   \]

   Exercise: write as simple LP!
4. **Shortest path from $s \to t$** in an edge weighted graph, \( \text{len}(e) \)

(say negative edge lengths but no negative cycles.)

(a) **First approach**: ask for min cost flow that sends 1 unit of flow from $s \to t$.

\[
\begin{align*}
\text{min } & \sum e \text{. } \text{len}(e) \cdot f_e \\
\text{st } & \sum f_e = 1 \\
& \sum f_e = \sum f_e \quad \forall v \neq s, t \\
& f_e \geq 0.
\end{align*}
\]

Claim: the flow will only be on shortest $s$-$t$ path.

and so, LP value = shortest path length.

**Proof**: exercise!

(b) **Second approach**: imagine the graph nodes as little rocks and edges as strings that connect these rocks. \( \text{len}(e) = \) length of string $e$. (Strings do not stretch.)

Now hold $s$ and try to pull $t$ as far as possible from $s$.

**Ans**: can make $t$ be at distance exactly the shortest path distance from $s$.

Because that edges will become tight.
So here's the LP.

**Vars:** \( d_u \) ("how far can you make \( u \) be from \( s \))

**Constraint:**
\[ d_s = 0. \]
\[ d_u \leq d_u + \text{length}(u,v) \quad \text{V edges } (u,v) \]

**Objective:** \( \max d_t. \)
"pull \( t \) as far as possible from \( s \)."

Thus two LPs for shortest path.

\[
\begin{align*}
\max & \quad d_t \\
\text{st} & \quad d_u \leq d_u + \text{length}(u,v) \quad \forall (u,v) \in E \\
& \quad d_s = 0
\end{align*}
\]

\[
\begin{align*}
\min & \quad \sum \text{length}(u,v) f(u,v) \\
\text{st} & \quad \sum f(s,v) \\
& \quad \sum f_{uv} = \sum f_{uw} + f_{uw} \forall (u,v) \in E \\
& \quad f_{uv} \geq 0
\end{align*}
\]

These LPs have the same value.
Actually are "duals" to each other.

We will see soon in a couple lectures.

This is not an efficient way to solve shortest paths. 😊
But it illustrates the power of LPs.
And also a good example of how two very different LPs can solve same problem.
L1-regression:

Given a set of points \((x_i, y_i), \ldots, (x_n, y_n) \in \mathbb{R}^2\)

where \(x_i \in \mathbb{R}, \ y_i \in \mathbb{R}\)

Want to fit a line such that the vertical distance between the "predicted value" of \(y\) and the actual value is small, summed over all points.

i.e., suppose we predict \(y = ax + b\) as the predicted relationship between \(x\) & \(y\).

Then error for point \((x_i, y_i)\)

\[
= (ax_i + b) - y_i
\]

\((L_2)\) least squares regression

\[
\min_{a,b} \sum_i (ax_i + b - y_i)^2
\]

L1-regression

\[
\min_{a,b} \sum_i |(ax_i + b) - y_i|
\]

So find a predictor (a line \(y = ax + b\)) that minimizes this.
What are variables?

Need to find $a, b$ s.t. line is $y = ax + b$.

(Note: Just because something is called $x$ or $y$ does not mean it's the variable 😐)

Good: Attempt 1: What constraints?

None.
Objective? $\min \sum_i |ax_i + b - y_i|$

Not a linear constraint!! Absolute values are not linear!

OK. Attempt 2:

Define auxiliary variable $z_i$ for each $i \leq 1, 2 \ldots n$.

Claim: $ax_i + b - y_i \leq z_i \quad \iff \quad |ax_i + b - y_i| \leq z_i$

$z_i \geq 0$

So if we minimize $z_i$ subject to these constraints, then $z_i = |ax_i + b - y_i|$

Good: $\min \sum_i z_i$

\begin{align*}
&\text{s.t. } ax_i + b - y_i \leq z_i \quad \forall i = 1 \ldots n \\
&\quad -z_i \leq ax_i + b - y_i \quad \forall i \\
&\quad z_i \geq 0 \quad \forall i \\
&\quad a, b \text{ unconstrained}
\end{align*}

Solves L1-regression.

? General idea for replacing absolute value signs in minimization problems.
(5) Recall **compressive sensing**.

(aka: sensing a sparse signal using linear measurements)

there: picked \( A \in \mathbb{R}^{k \times D} \) with \( k = \Omega(s \log D/s) \) rows.

entries are iid Normal \((0,1)\) r.v.s.

then seek:

\[
\begin{align*}
\text{min} & \quad \|x\|_1 \\
\text{st.} & \quad Ax = b \\
& \quad x \in \mathbb{R}^D
\end{align*}
\]

sensormatrix \( A \) \( \rightarrow \) measurements

Unknown \( s \)-sparse signal

(Theorem from Lec ) why this returns the correct \( x \)

(if \( x \) is \( s \)-sparse)

To solve the \( \ell_1 \)-min, again same idea. define \( z_1, z_2 \ldots z_D \)

and write LP.

\[
\begin{align*}
\text{min} & \quad \sum_i z_i \\
\text{st.} & \quad Ax = b \\
& \quad X_i \leq z_i \\
& \quad -z_i \leq X_i \\
& \quad z_i \geq 0
\end{align*}
\]

equivalent to the non-LP above.

\( \Leftrightarrow \) solves the compressive sensing problem

See code on the webpage

for some examples.
Can keep going with examples, but have made the point.

LPs are a powerful language.

So next time we see how to solve them.

And the main ideas behind why efficient algs exist.

To prepare: — will help to recall

Def: a set \( K \subseteq \mathbb{R}^d \) is **convex** if \( \forall x, y \in K \), the line segment connecting them is in \( K \).

formally: \( \forall x, y \in K, \forall \lambda \in [0,1], \ (\lambda x + (1-\lambda)y) \in K \)

Def: given a vector \( x \), it is **normal** to the set

\[ N_x = \{ y : \langle x, y \rangle = 0 \} \]

this is a hyperplane.
OK: Let's go back to LP example.

\[
\begin{align*}
\text{max } & \ 5x_1 + x_2 \\
\text{s.t. } & \ x_1 + x_2 \leq 7 \\
& \ -2x_1 + 3x_2 \geq -4 \\
& \ x_1 \geq 0, \ x_2 \geq 0.
\end{align*}
\]

We figured out the feasible points, but which point is maximizer?

**Fact 1:** Since maximizing linear function, optimal point is on the boundary (if there is an optimal point)

**Good. so where on the boundary?**

**Fact 2:** Assuming feasible region is bounded (does not go off to infinity) then optimal point is at a "corner".

(Note: in 2-d, corners are easy to understand. Can make it formal in general soon)

Above 4 corners, where? \((0,0), (2,0), (0,7), (5,2)\)

Can check brute force. best is \((5,2)\) with value 29.

But in general (high dimensions) # of corners can be large.

So need a smart way to enumerate over corners.

\(\text{In 2d, # corners is } \leq \# \text{ of linear constraints, can draw & brute force}\)
LP in "general form"

\[
\begin{align*}
\text{min } & \sum_{i=1}^{n} c_i x_i \\
\text{st. } & \sum_{j=1}^{m} a_{ij} x_j \geq b_i \\
\text{+ } & \sum_{j=1}^{m} a_{mj} x_j = b_m \\
\end{align*}
\]

- \(n\) variables \(x_1, x_2, \ldots, x_n\) \(\geq 0\)
- \(m\) constraints \(a_{ij} x_j \geq b_i\)
- 1 objective function \(\sum_{i=1}^{n} c_i x_i\)

\(\Rightarrow\) \(\text{inner product notation}\)

\(\Rightarrow\) \(\text{all components } \geq 0\)

Can convert other linear constraints to this form:

1. **Inequality, the wrong way?**
   
   \[5x_1 - 3x_2 \leq 7\]
   
   \(\Rightarrow\) \(-5x_1 + 3x_2 \geq -7\).

   Flip the signs everywhere.

2. **Equality?**
   
   \[x_1 - x_2 = 9\]
   
   \(\Rightarrow\) \(x_1 - x_2 \geq 9\) and then fix the inequality \(\leq\).

3. **\(x_i\) allowed to be positive or negative.**

   This is a bit sneaky/cute. Replace \(x_i\) by 2 vars \(x_i^+ - x_i^-\).

   \(x_i^+\) is "positive part of \(x_i\). \(x_i^-\) = "negative part". Both \(\geq 0\).
So. \[
\begin{align*}
\min & \quad c_1^T x + c_2^T x_2 \\
\text{s.t.} & \quad 3x_1 + x_2 \geq 9 \\
& \quad x_2 \geq 0
\end{align*}
\] \[\Rightarrow \]
\[
\begin{align*}
\min & \quad G(x_1^+ - x_1^-) + c_2^T x_2 \\
\text{s.t.} & \quad 3(x_1^+ - x_1^-) + x_2 \geq 9 \\
& \quad x_1^+, x_1^-, x_2 \geq 0.
\end{align*}
\]

Exercise: Both LPs are equivalent, have same solution, if you think of \( x_i = (x_i^+ - x_i^-) \).

4) Maximization LP instead of minimization?

Flip the signs of the objective and then minimize.

\[
\begin{align*}
\max & \quad x_1 + x_2 + x_3 \\
\text{s.t.} & \quad A x \geq b \\
& \quad x \geq 0
\end{align*}
\]

\[\Rightarrow \]
\[
\begin{align*}
\min & \quad -(x_1 + x_2 - x_3) \\
\text{s.t.} & \quad A x \geq b \\
& \quad x \geq 0.
\end{align*}
\]

Just notational convenience to have "general form". There can always talk about one crisp object.

\[
\begin{align*}
\min & \quad c^T x \\
\text{s.t.} & \quad A x \geq b \\
& \quad x \geq 0
\end{align*}
\]