Flows in Optimization

Stefano Fochesatto

University of Alaska Fairbanks

December 4, 2023

Overview

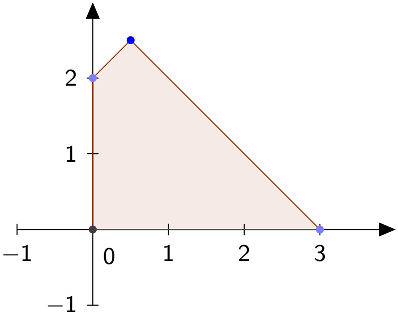
- Brief Introduction to Linear Optimization
- Geometry of Linear Programming
- Minimum Cost Network Flow Problem
- Different Flavors of Flow Problems
  - Max Flow
  - Min Cut
  - Shortest Path
  - Assignment (Weighted Matching)
- Lazy Snapping (Image Segmentation)

Linear Optimization

A *Linear Programming(LP) Problem* is an optimization problem with a linear *objective function* and linear *constraints*. A *feasible set* S, is the set of points satisfying the constraints.

Figure: Example of an *LP* Problem, and feasible set S.

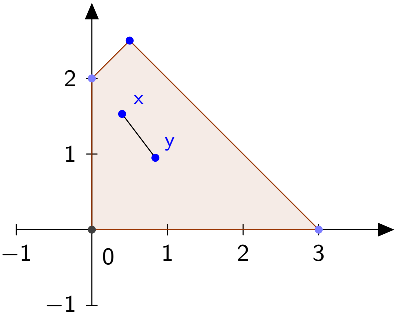
$$\begin{aligned} \underset{x}{\text{minimize:}} \quad & z = -x_1 - 2x_2 \\ \text{subject to:} \quad & -x_1 + x_2 \leq 2 \\ & x_1 + x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned}$$



Linear Optimization

- *LP*-Problems have *convex (set)* feasible sets.

Figure: S is a convex set.



$$\alpha x + (1-\alpha)y \in S \qquad \text{for all } 0 \le \alpha \le 1$$

- *LP*-Problems have *convex (function)* objective functions.

$$f(\alpha x + (1-\alpha)y) \ge \alpha f(x) + (1-\alpha) f(y) \qquad \text{for all } 0 \le \alpha \le 1$$

Linear Optimization

- An LP-problem is in *standard form* if it is written in the following form where $b \geq 0$,

$$\begin{aligned} \underset{x}{\text{minimize:}} \quad & z = c^T x \\ \text{subject to:} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

  Note $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$. We call A the *constraint matrix*.
- All LP-problems can be written in standard form.

Linear Optimization

- For an *LP*-problem in standard form, we define it's *dual LP*-problem as,

Figure: Primal and Dual *LP*-problems

$$\begin{aligned} \underset{x}{\text{minimize:}} \quad & z = c^T x \\ \text{subject to:} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$$\begin{aligned} \underset{y}{\text{maximize:}} \quad & w = b^T y \\ \text{subject to:} \quad & A^T y = c \\ & y \geq 0 \end{aligned}$$

Strong Duality

For a pair of Primal-Dual *LP*-problems if one has an optimal solution then so does the other, and the optimal objective values are equal.

Linear Optimization

- We say a solution is x is a *basic feasible solution* if,
  - The columns of A corresponding to $x_i \neq 0$ are linearly independent.
  - $x \geq 0$ and $Ax = b$.

- For any LP-problem, if S is bounded and non-empty then by convexity there exist a finite optimal solution and such a solution is a basic feasible solution.

Fundamental Theorem of Linear Programming

For an LP-problem in standard form, x is an extreme point of S if and only if x is a basic feasible solution.

Geometry of Linear Programming

To summarize...

- Every *LP*-problem can be written in standard form.
- For a standard form *LP*-problem, the optimal solution is an extreme point of the feasible set S.
- Strong Duality allows us to certify that our solution is the optimal one.

The *Simplex Algorithm*, which is actually used to solve *LP*-problems, is based on these principles.

That's great. . . but what's the point?

Flow Problems $\subset$ *LP*-Problems [1]

[1] At least, the ones we've seen

Preface on Terminology

The field of Optimization is very opinionated about terminology (just like Graph Theory)

- *Network* $==$ *Graph*
- *flow* is also an arbitrary unit assigned to each vertex. (ex: the vertex s has a flow of $f(s, N(s))$)
- Edges are also assigned a *cost*, which is the cost of moving one unit of flow across said edge.

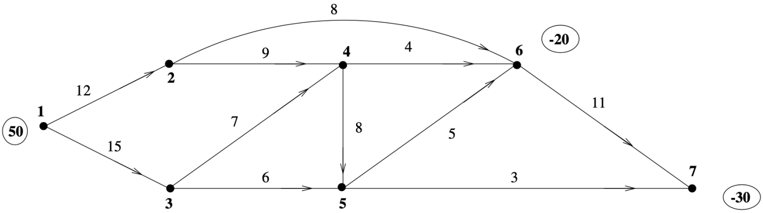
Minimum Cost Network Flow Problem

- Suppose we have a directed graph $G = (V, E)$, let b be a vector of flows assigned to each vertex, and c be a vector of costs for each arc.
- Vertex i with $b_i > 0$ is a *source*, $b_i < 0$ is a *sink*.
- We define *Supply* S, and *Demand* D as,

$$S = \sum_{\{i:b_i>0\}} b_i, \qquad D = \sum_{\{i:b_i<0\}} b_i.$$

- It is necessary that $S - D = 0$, such a network is called *balanced*.

Figure: Graph $G = (V, E)$ with flow b and weights c



Minimum Cost Network Flow Problem

A solution to the such a problem, is a flow f which satisfies the supply and demand ($|f| = S = -D$) but minimizes cost.

- Objective Function (x is a flow):

$$\underset{x}{\text{minimize:}} \;\; z = c^T x$$

  where $x_{(i,j)} \in x$ represents flow through arc (i,j), and $c_{(i,j)} \in c$ is cost through arc (i,j).

- Constraints:

$$\sum_j x_{i,j} - \sum_k x_{k,i} = b_i \text{ for all } i \in V.$$

  This ensures x satisfies supply and demand, for example,

$$x_{1,2} + x_{1,3} = 50, \qquad x_{2,4} + x_{2,6} - x_{1,2} = 0.$$

- Key Observation: x represents edges, b represents vertices....

Minimum Cost Network Flow Problem

- Observe that the following is the constraint matrix A, for the problem above.

$$A = \begin{bmatrix} 1 & 1 & & & & & & & & & \\ -1 & & 1 & 1 & & & & & & & \\ & -1 & & & 1 & 1 & & & & & \\ & & -1 & & -1 & & 1 & 1 & & & \\ & & & & & -1 & -1 & & 1 & 1 & \\ & & & -1 & & & & -1 & -1 & & 1 \\ & & & & & & & & & -1 & -1 \end{bmatrix}$$

- Minimum and maximum (capacity) flow through arc $x_{(i,j)}$, denoted by vectors L and U are enforced by inequality constraints.
- Described as an LP-problem,

$$\begin{aligned} \underset{x}{\text{minimize:}} \quad & z = c^T x \\ \text{subject to:} \quad & Ax = b \\ & L \leq x \leq U \end{aligned}$$

Maximum Flow

- Suppose we have a directed graph $G = (V, E)$, capacities $u_{(i,j)}$, a source vertex 1, and sink vertex m and we wish to find the maximum flow f.
- Represented as an optimization problem,

$$
\begin{aligned}
\underset{x,f}{\text{maximize:}}\quad & z = f \\
\text{subject to:}\quad & \sum_j x_{1,j} - \sum_k x_{k,1} = f, \\
& \sum_j x_{i,j} - \sum_k x_{k,i} = 0, \qquad i = 2, \ldots, m-1 \\
& \sum_j x_{m,j} - \sum_k x_{k,m} = -f, \\
& 0 \le x_{i,j} \le u_{i,j}.
\end{aligned}
$$

- explain...

Maximum Flow

- Add arc $(m, 1)$ with $u_{(m,1)} = \infty$ to uncouple, we get

$$\begin{aligned} \underset{x}{\text{minimize:}} \quad & z = -x_{m,1} \\ \text{subject to:} \quad & \sum_j x_{i,j} - \sum_k x_{k,i} = 0, \quad i = 1, \ldots, m \\ & 0 \le x_{i,j} \le u_{i,j} \end{aligned}$$

- This is a Minimum Cost Network Flow problem in disguise
  - Let $b = 0$, $c_{(m,1)} = -1$ and zero otherwise, $U = u$ and $L = 0$.
  - All of costs were zero . . .

Minimum Cut

- We have shown in class that the Maximum Flow = Minimum Cut,
  - Strong Duality implies that the Dual LP-problem computes the minimum cut.
- Represented as an optimization problem (Dual of original Max-flow),

$$
\begin{aligned}
\underset{y,v}{\text{minimize:}} \quad & w = \sum u_{i,j} v_{i,j} \\
\text{subject to:} \quad & y_m - y_1 = 1 \\
& y_i - y_j + v_{i,j} \geq 0, \quad \text{for all arcs } (i,j) \\
& v_{i,j} \geq 0
\end{aligned}
$$

- y represents indicator for disjoint sets N_1 and N_0, v represents the indicator for the separating set of arcs.

Shortest Path

- Finding the shortest path between two vertices v_0 and v_f, on a directed graph $G = (V, E)$.
  - Minimum Cost Network Flow problem with one source v_0 and one sink v_f and flows are 1 and -1 respectively.
  - $c_{(i,j)}$ is distance.

- More efficient algorithms exists, the more popular ones require $c_{(i,j)} \geq 0$.

Assignment (Weighted Matching)

- Recall the stable matching problem (medical students to residencies).
  - Let $c_{i,j}$ be 'happiness' score for matching student i with program j.
  - We want to find a matching which maximizes happiness.
- Let $G = (A \cup B, E)$ be a directed complete bipartite graph
- A represents student, B represents programs, arcs go from A to B
- Vertices in A are all sources with flow 1, with B defined analogously.
- Cost are defined by $c_{i,j}$ and instead we maximize the objective function.

Lazy Snapping

- Image segmentation, is the process by which a subject and background are identified.

Figure: Example Segmentation

- Lazy Snapping is a technique for image segmentation which uses flows.
  - User selects some source and sink pixels.
  - The image is converted into a min-cut flow problem. (this part is hard)
  - Resulting disjoint sets identify the subject and background.

Lazy Snapping

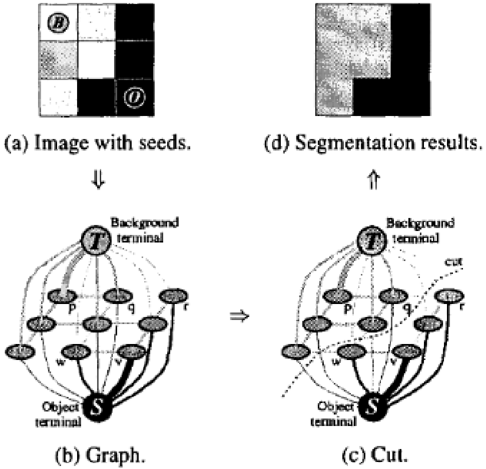
- It's relatively easy to frame this as an optimization problem.
- We define an energy function, $E(X)$ where X is indicator for subject pixels.

$$E(X) = \sum_{i \in V} E_1(x_i) + \lambda \sum_{(i,j) \in E} E_2(x_i, x_j).$$

- E_1 and E_2 are defined in Li et al.(2004)
  - $\sum_{i \in V} E_1(x_i)$ is small when pixels identified by X are the same color.
  - $E_2(x_i, x_j)$ identifies pixels x_i, x_j where $X(x_i) \neq X(x_j)$, and is large when the pair of pixels are a similar color.
- Converting energy functions like E into flow problems is described in Boykov & Jolly (2001).

Lazy Snapping

Figure: Image to Flow



Lazy Snapping

- Demo Time !!!