

CS 467/567: Linear Programming

Stefan D. Bruda

Winter 2020

LINEAR PROGRAMMING



- **Linear function:** $f(x_1, x_2, \dots, x_n) = \sum_{j=1}^n a_j x_j$
- **Linear constraint:** $g(x_1, \dots, x_n) \bullet b$ for some linear function g and either
 - $=$ (linear equality) or $\bullet \in \{\leq, \geq\}$ (linear inequalities)
- A **linear programming problem** is the problem of optimizing (minimizing or maximizing) a linear function subject to a finite set of linear constraints; an instance of this problem is a **linear program**
- Solving 2-variable optimization problems:
 - The linear constraints form a **convex region** in the (x_1, x_2) -Cartesian coordinate system (the **simplex**)
 - The set of points $f(x_1, x_2) = z$ form a **line** whose slope is independent of z
 - The goal becomes finding the optimal (maximal/minimal) z with a non-empty intersection between the simplex and the line, which **always corresponds to a vertex of the simplex** if the simplex is bounded
 - The **simplex algorithm** starts from an arbitrary vertex of the simplex, keeps moving to a neighbor whose value is no smaller/larger than that of the current vertex
 - The algorithm terminates when it reaches a local maximum/minimum (a vertex with all the neighbors having a smaller/larger objective value)
 - This is also the global maximum/minimum **because the simplex is convex**
- Idea can be trivially generalized to an n -dimensional spaces

CS CS 467/567 (S. D. Bruda)

Winter 2020

1 / 12

THE STANDARD FORMS OF LINEAR PROGRAMS



- **Standard form:** Given n real numbers c_j , $1 \leq j \leq n$ and mn real numbers a_{ij} , $1 \leq i \leq m$, $1 \leq j \leq n$, find n real numbers x_j , $1 \leq j \leq n$ that:
 - maximize the **objective function** $\sum_{j=1}^n c_j x_j$
 - subject to the **constraints** $\sum_{j=1}^n a_{ij} x_j \leq b_i$, $1 \leq i \leq m$ and $x_j \geq 0$, $1 \leq j \leq n$
- **Slack form:** all constraints are either equality constraints or $x_j \geq 0$
- Terminology for linear programs:
 - An assignment \bar{x} of the variables x_j , $1 \leq j \leq n$ can be a **feasible solution** (satisfies all constraints) or an **infeasible solution** (violates at least one constraint)
 - A solution \bar{x} has the **objective value** $c^T \bar{x}$
 - A solution \bar{x} whose $c^T \bar{x}$ is the maximum of all the feasible solutions is an **optimal solution** and its $c^T \bar{x}$ is the **optimal objective value**
 - If a linear program does not have any feasible solutions then it is **infeasible**, otherwise it is **feasible**
 - If a linear program does not have a finite optimal objective value then it is **unbounded**

CONVERSION TO STANDARD AND SLACK FORMS



- Conversion to standard form:
 - To convert a minimization instance into a maximization instance: negate all the coefficients in the objective function
 - If some variable x_j does not have the constraint $x_j \geq 0$:
 - Replace every occurrence of x_j with $x_j' - x_j''$
 - Add the constraints $x_j' \geq 0$ and $x_j'' \geq 0$
 - If we have an equality constraint $g(x_1, \dots, x_n) = b$: replace it with two constraints $g(x_1, \dots, x_n) \geq b$ and $g(x_1, \dots, x_n) \leq b$
 - If we have a constraint $g(x_1, \dots, x_n) \geq b$: we multiply both sides with (-1) (and we flip the comparison, and we distribute the (-1) into the sum)
- Conversion of standard form into slack form:
 - Each constraint $\sum_{j=1}^n a_{ij} x_j \leq b_i$ is rewritten as the two constraints

$$s = b_i - \sum_{j=1}^n a_{ij} x_j \quad s \geq 0$$

- s is a new variable (**slack** or **basic variable** as opposed to a **nonbasic variable**)... that just does not happen to appear in the objective function



- **Problem:** Given a graph $G = (V, E)$ with a weight function $w : E \rightarrow \mathbb{R}$, a source vertex $s \in V$ and a target vertex $t \in V$, find d_t , the weight of the shortest path from s to t
- **Linear program:** maximize d_t subject to the following constraints:

$$d_s = 0 \quad d_v \leq d_u + w(u, v) \text{ for each } (u, v) \in E$$

- d_u is the weight of the path from s to u
- Note: $\bar{d}_v = \min_u \text{ s.t. } (u, v) \in E \{d_u + w(u, v)\}$ so \bar{d}_v is the maximal value smaller than all the values in $\{d_u + w(u, v)\}$ so we need to maximize \bar{d}_v (the minimization nature of the problem is given by the constraints)



- **Flow network:** graph $G = (V, E)$ with a capacity function $c : E \rightarrow \mathbb{R}^+$ and two designated vertices $s, t \in V$ such that $(u, v) \in E \Rightarrow (v, u) \notin E$
 - Convenient abuse of notation: $c(u, v) = 0$ whenever $(u, v) \notin E$
- **Flow** in G : function $f : V \times V \rightarrow \mathbb{R}$ satisfying the following constraints
 - **Capacity constraint:** $0 \leq f(u, v) \leq c(u, v)$ for all $u, v \in V$
 - **Flow conservation:** $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$ for all $u \in V \setminus \{s, t\}$
- **Problem:** Given a flow network, find a flow that maximizes $f(s, t)$
- **Linear program:** maximize $\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs}$ subject to the following constraints:

$$\begin{aligned} f_{uv} &\leq c(u, v) && \text{for each } u, v \in V \\ \sum_{v \in V} f_{vu} &= \sum_{v \in V} f_{uv} && \text{for each } u \in V \setminus \{s, t\} \\ f_{uv} &\geq 0 && \text{for each } u, v \in V \end{aligned}$$



- Input: a flow network $G = (V, E)$ with capacity function c
- Additional input: Commodities K_1, K_2, \dots, K_k
 - $K_i = (s_i, t_i, d_i)$ where s_i/t_i are the source/target vertices for K_i , and d_i is the demand (desired flow value) for K_i
 - Aggregate flow: $f_{uv} = \sum_{i=1}^k f_{iuv}$, where f_{iuv} is the flow for K_i from u to v
- **Problem:** Given a flow network and k commodities, determine whether there exists an aggregate flow f such that $f_{uv} \leq c(u, v)$ for all $(u, v) \in E$
- **Linear program:** maximize 0 subject to the following constraints:

$$\begin{aligned} \sum_{i=1}^k f_{iuv} &\leq c(u, v) && \text{for each } u, v \in V \\ \sum_{v \in V} f_{iuv} - \sum_{v \in V} f_{ivu} &= 0 && \text{for each } u \in V \setminus \{s, t\} \\ \sum_{v \in V} f_{is_i v} &= \sum_{v \in V} f_{iv s_i} = d_i && \text{for each } 1 \leq i \leq k \\ f_{iuv} &\geq 0 && \text{for each } u, v \in V \text{ and } 1 \leq i \leq k \end{aligned}$$

- Solving multicommodity flow as a linear programming problem is the only known efficient algorithm for this problem



- A slack form is a tuple (N, B, A, b, c, ν) which denotes the following linear program

$$\begin{aligned} z &= \nu + \sum_{j \in N} c_j x_j \\ x_i &= b_i - \sum_{j \in N} a_{ij} x_j && \text{for } i \in B \end{aligned}$$

with the implicit understanding that $x_i \geq 0$ for all $i \in N \cup B$

- N contains the indices of all nonbasic variables, $|N| = n$
- B contains the indices of all basic variable, $|B| = m$
- $N \cup B = \{1, 2, \dots, n + m\}$



One iteration = **pivot operation**

- 1 Set all the nonbasic variables to 0, solve for the basic variables = **basic solution**
- 2 Select a nonbasic variable x_e with positive coefficient (**entering variable**) in the objective function and increase its value as much as possible without constraint violation
- 3 The above increase makes one basic variable x_l zero (the **leaving variable**)
- 4 Reformulate the constraints such that x_e becomes basic and x_l nonbasic
- 5 If all the coefficients c_j are negative then the current basic solution is the optimal solution, otherwise repeat from Step 1



Algorithm PIVOT($(N, B, A, b, c, \nu), l, e$) returns $(N', B', A', b', c', \nu')$

- Compute the coefficients of the equation for new basic variable x_e :
 - 1 $b'_e \leftarrow b_l/a_{le}$
 - 2 **for each** $j \in N \setminus \{e\}$ **do** $a'_{ej} \leftarrow a_{ij}/a_{le}$
 - 3 $a'_{el} \leftarrow 1/a_{le}$
- Compute the coefficients of the remaining constraints:
 - 1 **for each** $i \in B \setminus \{l\}$ **do**
 - 1 $b'_i \leftarrow b_i - a_{ie}b'_e$
 - 2 **for each** $j \in N \setminus \{e\}$ **do** $a'_{ij} \leftarrow a_{ij} - a_{ie}a'_{ej}$
 - 3 $a'_{il} \leftarrow -a_{ie}a'_{el}$
- Compute the objective function:
 - 1 $\nu' \leftarrow \nu + c_e b'_e$
 - 2 **for each** $j \in N \setminus \{e\}$ **do** $c'_j \leftarrow c_j - c_e a'_{ej}$
 - 3 $c'_l \leftarrow -c_e a'_{el}$
- Compute the new basic and nonbasic variables:
 - 1 $N' \leftarrow N \setminus \{e\} \cup \{l\}$
 - 2 $B' \leftarrow B \setminus \{l\} \cup \{e\}$



Algorithm SIMPLEX(A, b, c) returns $(\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$

- 1 $(N, B, A, b, c, \nu) \leftarrow \text{INITIALIZE-SIMPLEX}(A, b, c)$
 - 2 let Δ be a vector of size n
 - 3 **while** some index $j \in N$ has $c_j > 0$
 - 1 choose an index $e \in N$ such that $c_e > 0$
 - 2 **for each** $i \in B$ **do**
 - if $a_{ie} > 0$ **then** $\Delta_i \leftarrow b_i/a_{ie}$ **else** $\Delta_i \leftarrow \infty$
 - 3 choose $l \in B$ such that Δ_l is minimum over Δ
 - 4 **if** $\Delta_l = \infty$ **then return** "unbounded"
 - 5 **else** $(N, B, A, b, c, \nu) \leftarrow \text{PIVOT}((N, B, A, b, c, \nu), l, e)$
 - 4 **for** $i \leftarrow 1$ **to** n **do**
 - 1 **if** $i \in B$ **then** $\bar{x}_i \leftarrow b_i$ **else** $\bar{x}_i \leftarrow 0$
- Input is a linear program in standard form
 - INITIALIZE-SIMPLEX returns a slack form for which the initial basic solution is feasible (or a suitable message if the linear program is infeasible)



Lemma

Let L be a program in standard form and x_0 a new variable. Let L_{aux} be:

$$\begin{aligned} & \text{Maximize } -x_0 \\ & \text{subject to } \sum_{j=1}^n a_{ij}x_j - x_0 \leq b_i \text{ for } 1 \leq i \leq m \text{ and } x_j \geq 0 \text{ for } 0 \leq j \leq n \end{aligned}$$

Then L is feasible iff the optimal objective value for L_{aux} is 0.

INITIALIZE-SIMPLEX then works as follows:

- Let b_k be the minimum b_i
- If $b_k \geq 0$ then the initial solution is feasible so convert to slack and return
- Form L_{aux} as in the lemma, convert it to the slack form (N, B, A, b, c, ν)
- Let $l = n + k$ and perform PIVOT($(N, B, A, b, c, \nu), l, 0$); the basic solution is now feasible for L_{aux}
- Use the while loop of SIMPLEX to find an optimal solution for L_{aux} ; return "infeasible" if $\bar{x}_0 \neq 0$
- Remove x_0 from the constraints, restore the original objective function for L , but replace basic variables with the right hand side of its constraint
- Return this final slack form



Theorem

If SIMPLEX fails to terminate in at most $\binom{n+m}{m}$ iterations then it cycles.
 So SIMPLEX either reports that the linear program is unbounded or terminates with a feasible solution in at most $\binom{n+m}{m}$ iterations

Theorem (Fundamental theorem of linear programming)

Any linear program L given in standard form either:

- 1 has an optimal solution with a finite objective value,
- 2 is infeasible, or
- 3 is unbounded.

If L is infeasible or unbounded, then SIMPLEX returns “infeasible” or “unbounded”, respectively. Otherwise SIMPLEX returns an optimal solution with a finite value.