# Linear Programming

**Definition.** ***General Linear Programs***

1. ***Linear Function*** *Given* $a_1, \cdots, a_n \in \mathbb{R}$*, and variables* $x_1, \cdots, x_n$*, define a linear function* $f$ *of those variables by*

$$f(x_1, \cdots, x_n) = \sum_{j=1}^{n} a_j x_j$$

2. ***Linear equality & inequalities*** *If* $b \in \mathbb{R}$ *and* $f$ *is a linear function, then*

$$f(x_1, \cdots, x_n) = b$$

*is a linear equality and the inequaliie*

$$f(x_1, \cdots, x_n) \leq b \qquad f(x_1, \cdots, x_n) \geq b$$

*are linear inequalities*

3. ***Linear Constraints*** *are either linear equalities or linear inequalities*

4. ***Linear programming problem*** *Either minimizing or maximizing a linear fucntion subject to a finite set of linear constraints. If want to minimize, then linear program is a* ***minimization linear problem****, otherwise its called a* ***maximization linear problem***

5. ***Feasible solution*** *Any setting of variable* $x_1, \cdots, x_n$ *that satisifies all constraints a feasible solution to the linear program*

6. ***Feasible Region*** *a convex set of feasible solutions for which we which to maximize the objective function*

7. ***Objective value*** *value of the objective function at a particular point in the feasible solution*

8. ***Graphical solution*** *If 2 variables, then we can use the let* $z$ *be the objective. Such curve have the property that the intersection between the curve and the feasible solution is the set of feasible solutions with objective value* $z$*. A optimal solution to linear program occurs at a vertex of a feasible region, since the curve that intersect the feasible region for which maximum* $z$ *is obtained is on the boundary of the feasible region. This holds for higher dimension curves as well*

9. ***Simplex*** *For* $n$ *variables, each constraint defines a half-space in* $n$*-dimensional space, the feasible region formed by the intersection of these half spaces is a simplex. The objective function is a hyperplane, and because of convexity, an optimal solution still occurs at a vertex of the simplex*

10. ***Simplex alogrithm*** *takes as input a linear program and returns an optimal solution. It starts as some vertex of the simplex and performs a sequence of iterations. In each iteration, it moves along an edge of the simplex from a current vertex to a neighboring vertex whose objective value is no smaller than that of the current vertex. The algorithm terminates when it reaches a local minimum, i.e. all neighboring vertices have a smaller objective value.*

    **Lemma.** ***Duality*** *Since a feasible region is convex and objective function is linear, a local optimum from a simplex algorithm is a global optimum*

    (a) *Write linear program in slack form*

    (b) ***Pivot*** *Make one variable basic and another nonbasic*

**Definition.** ***Standard form***

1. ***Specification*** *Given* $n$ *real number* $c_1, \cdots, c_n \in \mathbb{R}$ *and* $m$ *real number* $b_1, \cdots, b_m \in \mathbb{R}$ *and* $mn$ *real number* $a_{ij}$ *for* $i = 1, \cdots, m$ *and* $j = 1, \cdots, n$. *We wish to find* $n$ *real numbers* $x_1, \cdots, x_n$ *such that*

$$\begin{aligned} \textit{Maximize } & \sum_{k=1}^{n} c_j x_j \\ \textit{subject to } & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \textit{ for } i = 1, \cdots, m \\ & x_j \geq 0 \textit{ for } j = 1, \cdots, n \end{aligned}$$

   *Note standard form requires the* $n$ *nonnegative constraints on* $x_1, \cdots, x_n$. *Alternatively, let* $A = (a_{ij})$ *be* $m \times n$ *matrix,* $b = (b_i)$ *a* $m$*-vector,* $c = (c_j)$ *a* $n$*-vector, and* $x = (x_j)$ *an* $n$*-vector. Then*

$$\begin{aligned} \textit{Maximize } & c^T x \\ \textit{subject to } & Ax \leq b \\ & x \geq 0 \end{aligned}$$

   *Therefore a standard form can be expressed with* $(A, b, c)$

2. ***Re-definition***

    (a) ***Feasible & Infeasible solution*** *A setting of variable* $\bar{x}$ *satisfies all constraints a fesasible solution, whereas a setting of* $\bar{x}$ *that fails to satisfy at least one constraint if an infeasible solution .*

(b) ***Objective Value*** *A solution $\bar{x}$ has objective value $c^T\bar{x}$*

(c) ***Optimal solution & Optimal Objective Value*** *a feasible solution $\bar{x}$ whose objective value is maximum over all feasible solutions is an optimal solution, its objective value $c^T\bar{x}$ is the optimal objective vlaue*

(d) ***Feasible & Unfeasible LP*** *If a linear program has no feasible solution, then it is infeasible, otherwise it is feasible*

(e) ***Unbounded LP*** *If a linear program has some feasible solution but does not have a finite optimal objective value, then LP is unbounded*

3. ***Converting linear program (4 types) to standard form***

(a) ***Equivalent LP***

i. *Two maximization linear programs L and L" are equivalent if for each feasible solution $\bar{x}$ to L with objetive value z, there is a corresponding solution $\bar{x}'$ to $L'$ with objective value z, and vice versa*

ii. *A minimization linear program L and a maximization linear program $L'$ are equivalent if for each feasible solution $\bar{x}$ to L with objective value z, there is a corresponding feasible solution $\bar{x}$ to $L'$ with objective value $-z$, and vice versa*

(b) ***Objective function is a minimization rather than a maximization***

*Negate coefficients ($c' = -c$) in the objective function.*

*2 LP's are equivalent since we have the same feasible solution (constraints unchanged) and for each feasible solution, the objective value in L is the negative of the objective value in $L'$ hence 2 linear programs are equivalent*

(c) ***There might be variables without nonnegativity constraints***

*Rerplace each occurrence of a variable variable $x_j$ without nonnegativity constraint by $x_j' - x_j''$, and add the nonnegativity constraint $x_j' > 0$ and $x_j'' > 0$*

(d) ***There might be equality constraints***

*Replace equality constraints with a pair of inequality constraints*

$$f(x_1, \cdots, x_n) \leq b \qquad f(x_1, \cdots, x_n) \geq b$$

(e) ***There might be $\geq$ inequality constraints***

*Multiple the greater than or equal to $\geq$ constraints to less than or equal to $\leq$ constraints by multiplying these constraints by -1*

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

**Definition.** ***Slack form***

1. ***Slack variable*** *Given inequality constraints* $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$, *we have*

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

   *where* $s$ *is a slack variable because it measures the slack, or difference, between left-hand and right-hand sides of equation. We can use this methods to convert from standard form to slack form, where the only inequality constraints are the nonnegativity constraints*

2. ***Conversion from standard to slack form*** *Use* $x_{n+i}$ *instead of* $s$ *to denote the slack variable associated with the* $i$*-th inequality. The* $i$*-th constriant is therefore*

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij}x_j \qquad\qquad x_{n+i} \geq 0$$

3. ***Basic & Nonbasic variables*** *Given a slack form with a set of equality constriants, one of variables on left-hand side of equality and all others on the right-hand side. The variables on the left-hand side of equalities are basic variables, and those on the right-hand side are nonbasic variables. Nonbasic variables are the only varaibles that constitutes the objective function*

4. ***Slack Form*** *Let* $z$ *be the value of the objective function and linear inequalities be converted to a set of slack variables. Omit the nonnegativity constraints since it is assumed that all variables are nonnegative. Let* $N$ *be the set of indices of nonbasic variables, let* $B$ *be set of indices of the basic variables, we always have* $|N| = n$ *and* $|B| = m$, *where* $N \cup B = \{1, \cdots, n+m\}$.

   (a) *equations are indexed by entries of* $B$

   (b) *variables on RHS of equation are index by entries of* $N$

   *Let* $A, b, c$, *be constants and coefficients. Let* $v$ *be the constant term in objective function. Therefore, we define a slack form by a tuple* $(N, B, A, b, c, v)$ *where*

$$z = v + \sum_{j \in N} c_j x_j$$
$$x_i = b_i - \sum_{j \in N} a_{ij}x_j \qquad \textit{for } i \in B$$

   *Note indices into* $A, b, c$ *are no necessarily sets of contiguous integers, they depend on the index sets* $B$ *and* $N$

## Formulating problems as Linear Programs

**Definition.** ***Shortest path*** *Given a weighte, directed graph $G = (V, E)$ with weights $w : E \to \mathbb{R}$ and source $s$ and destination $t$. Wish to ompute the value $d_t$, i.e. the weight of a shortest path from $s$ to $t$. We can formulate it as LP as follows*

$$\begin{aligned} &\textit{Maximize} && d_t \\ &\textit{Subject to} && d_v \le d_u + w(u,v) \qquad \textit{for each } (u,v) \in E \\ & && d_s = 0 \end{aligned}$$

*The bellman-form algorithm sets source vertex distance $d_s = 0$ and never changes it. When the algorithm terminates, it has computed, for each $v$, a value $d_v$ such that for each edge $(u, v) \in E$, we have $d_v \le d_u + w(u, v)$*
*Note we are* ***maximizing*** *$d_t$ for 2 reasons*

1. *setting $\bar{d}_v = 0$ for all $v \in V$ yields optimal solution without solving shortest-path problem*
2. *Maximize because an optimnal solution to shortest path problem sets each $\bar{d}_v$ to be $\underset{u:(u,v)\in E}{Min}\{d_u + w(u,v)\}$ (considers all incident edges to $v$) such that $d_v$ is the maximum value that is less than or equal to values in the set $\{\bar{d}_u + w(u,v)\}$. We maximize $d_v$ for all vertex $v$ on a shortest path from $s$ to $t$ subject to constraints, and maximizing $d_t$ achieves this...*

**Definition.** ***Maximum flow*** *Given directed graph $G = (V, E)$ with nonnegative capacity $c : E \to \mathbb{R}^+$ and two vertices, a source $s$ and a sink $t$. A flow $f : V \times V \to \mathbb{R}$ satisfies capacity constraint and flow conservation. A maximum flow is a flow that satisfies these constraints and maximizes the flow value. Also we assume $c(u, v) = 0$ if $(u, v) \notin E$ and no antiparallel edges*

$$\begin{aligned} &\textit{Maximize} && \sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} \ \textit{(Value of a flow)} \\ &\textit{Subject to} && f_{uv} \le c(u,v) \qquad \textit{for each } u,v \in V \ \textit{(capacity constraint)} \\ & && \sum_{v\in V} f_{vu} = \sum_{v\in V} f(u,v) \quad \textit{for each } u \in V \setminus \{s,t\} \ \textit{(flow conservation)} \\ & && f_{uv} \ge 0 \end{aligned}$$

**Definition.** ***Min-Cost-Flow*** *Consider the that in addition to a capacity of each edge $(u, v)$ in a max flow problem, we are givne a real-valued cost $a(u, v)$. Assume $c(u, v) = 0$ if $(u, v) \notin E$ and that there are no antiparallel edges. If we send $f_{uv}$ units of flow over edge $(u, v)$, we incur a cost of $a(u, v)f_{uv}$. Given a flow demand $d$. We wish to send $d$ units of flow from $s$ to $t$ while minimizing the total cost*

$$\sum_{(u,v)\in E} a(u,v) f_{uv}$$

*incurred by he flow. The linear program is similar to max flow problem with the constraint that value of flow be exactly d cost.*

$$\begin{aligned}
\textbf{\textit{Minimize}} \quad & \sum_{(u,v)\in E} a(u,v) f_{uv} \\
\textbf{\textit{Subject to}} \quad & f_{uv} \le c(u,v) \quad \textit{for each } u, v \in V \\
& \sum_{v\in V} f_{vu} - \sum_{v\in V} f_{uv} = 0 \quad \textit{for each } u \in V \setminus \{s,t\} \\
& |f| = \sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} = d \\
& f_{uv} \ge 0 \quad \textit{for each } u, v \in V
\end{aligned}$$

**Definition.** ***Multicommodity flow*** *Given a directed graph $G = (V, E)$ in which each edge $(u, v) \in E$ has nonnegative capacity $c(u, v) \ge 0$. Assume $c(u, v) = 0$ for $(u, v)] \neq E$ and graph has no antiparallel edges. In addition, given $k$ different commodities, $K_1, \cdots, K_k$ where we specify commodity $i$ by triple $K_i = (s_i, t_i, d_i)$. Here $s_i$ is source of $i$ and $t_i$ is sink of $i$ and $d_i$ is demand for $i$, which is the desired flow value for the commodity from $s_i$ to $t_i$. Define a flow for commodity $i$ by $f_i$ ($f_{iuv}$ is flow of commodity $i$ from $u$ to $v$) be real-valued function satisfying flow-conservation and capacity constraint. Define $f_{uv}$ be* ***aggregate flow****, to be the sum of the various commodity flows, i.e.*

$$f_{uv} = \sum_{i=1}^{k} f_{iuv}$$

*where the aggregate flow $f_{uv}$ must be no more than capacity of edge $(u, v)$. Since we are tyring to determine if such flow exists,we write a linear program with a null objective function ...*

$$\begin{aligned}
\textbf{\textit{Minimize}} \quad & 0 \\
\textbf{\textit{Subject to}} \quad & \sum_{i=1}^{k} f_{iuv} \le c(u,v) \quad \textit{for each } u, v \in V \\
& \sum_{v\in V} f_{iuv} - \sum_{v\in V} f_{ivu} = 0 \quad \textit{for each } i = 1, \cdots, k \textit{ and } u \in V - \{s_i, t_i\} \\
& \sum_{v\in V} f_{is_iv} - \sum_{v\in V} f_{ivs_i} = d_i \quad \textit{for each } i = 1, \cdots, k \\
& f_{iuv} \ge 0 \quad \textit{for each } u, v \in V \textit{ and } i = 1, \cdots k
\end{aligned}$$

## Simplex algorithm

**Definition.** ***Simplex algorithm***

1. ***Steps***

   (a) *In each iteration, find basic solution from slack form of linear program*

      i. *set each nonbasic variable $N$ to 0,*

      ii. *compute the values of basic variables $B$ from the equality constraint*

      iii. *compute objective value using objective function*

   (b) *Convert one slack form into an equivalent slack form*

      i. *Pick a nonbasic* ***entering variable*** *$x_e \in N$ such that if we were to increase that variable's value from 0, then the objective function would increase, too (i.e. positive coefficient in objective function)*

      ii. *Raise the nonbasic variable $x_e$ as much as possible without violating any constraint (i.e. until some basic* ***leaving variable*** *$x_l$ becomes 0, slack becomes tight)*

      iii. *rewrite the slack form, exchanging the roles of $x_e$ and $x_l$, substitute expression of the now-basic $x_e$ to other constraint/objective*

   (c) *stops when there is no applicable entering variable left (i.e. coefficient of nonbasic variable in objective function have all negative coefficient)*

2. *Definition*

   (a) ***Tight*** *An equality constraint is tight for a particular setting of its nonbasic variables if they cause the constraint's basic variable to become 0. A setting of nonbasic variable that makes a basic variable become negative violates that constraint, therefore slack*

      i. *maintains explicitly how far each constraint is from being tight*

      ii. *help to determine how much we can increase values of nonbasic variables without violating any constraints*

   (b) ***Basic feasible solution*** *A basic solution that is also feasible. (As we run the simplex algorithm, the basic solution is almost always a basic feasible solution except for maybe the first several..)*

3. ***Pivoting*** *Exchanging role of $x_l$ with $x_e$*

   (a) *takes a slack form given by tuple $(N, B, A, b, c, v)$ as input, index $l$ of leaving variable $x_l$ and index $e$ of entering variable $x_e$*

   (b) *Returns a tuple $(\hat{N}, \hat{B}, \hat{A}, \hat{b}, \hat{c}, \hat{v})$ describing the new slack form*

**Lemma.** *Consider a call to* $\text{PIVOT}(N, B, A, b, c, v, l, e)$ *in which $a_{le} \neq 0$. Let values returned be $(\hat{N}, \hat{B}, \hat{A}, \hat{b}, \hat{c}, \hat{v})$, and let $\overline{x}$ denote the basic solution after the call, then*

1. *$\overline{x}_j = 0$ for each $j \in \hat{N}$*

*2.* $\overline{x}_e = \dfrac{b_l}{a_{le}}$ *( since all other nonbasic variable set to 0)*

*3.* $\overline{x} = b_i - a_{ie}\hat{b}_e$ *for each* $i \in \hat{B} \setminus \{e\}$

**Definition.** ***Formal simplex algorithm***

1. Initialize-Simplex$(A, b, c)$ *takes input of linear program in standard form,*
   (a) *if problem infeasible, returns a message that program is infeasible*
   (b) *otherwise, return a slack form for which the initial basic solution is feasible*
2. *Pick the next positive-coefficient nonbasic variable e in the objective function such that* $c_e > 0$
3. *Now populate* $\triangle_i = \dfrac{b_i}{a_{ie}}$ *for* $i \in B$, *i.e. the amount by which e can increase by before the i-th basic variable becomes zero (i.e. satisfy nonnegativity constraint)*
   (a) *if* $\triangle_i = \infty$, *then solution unbounded*
   (b) *otherwise pick an index* $l \in B$ *such that* $\triangle_i$ *is minimized. and call* Pivot *on e and l which returns the next slack form*

   *Loop until no variable has positive coefficient in objective function*
4. *Based on the last slack form, compute and return the corresponding basic solution* $\overline{x}$
   (a) *if* $i \in B$, *then* $\overline{x}_i = b_i$
   (b) *otherwise,* $\overline{x}_i = 0$

**Lemma.** *Given a linear program* $(A, b, c)$, *suppose call to* Initialize-Simplex *returns a slack form for which the basic solutin is feasible, then if* Simplex *returns a solution, that solution is a feasible solution to the linear program. If* Simplex *returns unbounded then the linear program is unbounded*

**Lemma.** *Let* $(A, b, c)$ *be a linear program in standard form. Given a set* $B$ *of basic variables, the associated slack form is uniquely determined.*

**Lemma.** *If* Simplex *fials to terminate in at most* $\binom{n+m}{m}$ *iterations, the it cycles.*

*Proof.* By previous lemma, set $B$ uniquely determines. There are $n + m$ variables and $|B| = m$, and therefore, there are at most $\binom{n+m}{m}$ ways to choose $B$. Thus at most $\binom{n+m}{m}$ unique slack forms. Therefore if Simplex runs for more than $\binom{n+m}{m}$ iterations, it must cycle $\square$

**Lemma.** BLAND'S RULE *To prevent cycling, we pick $e$ and $l$ such that we break ties by always choosing the variable with smallest index, then* SIMPLEX *must terminate*

**Lemma.** *Assume* INITIALIZE-SIMPLEX *returns a slack form for which the basic solutin is feasible,* SIMPLEX *either reports a linear program is unbounded or it terminates with a feasible solution in at most $\binom{n+m}{m}$ iterations.*

## Duality

**Definition.**