1 Linear Programming

1.1 Standard Form of LP

Decision variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)^\top$
Objective coefficients $\mathbf{c} = (c_1, c_2, \ldots, c_n)^\top$
Right-hand-side constraints $\mathbf{b} = (b_1, b_2, \ldots, b_m)^\top$
Structural coefficients $A \in \mathbb{R}^{m \times n}$
Maximize $z = \sum_{j=1}^{n} c_j x_j$
subject to $\sum_{j=1}^{n} a_{ij} x_j = b_i, i = 1, ..., m$
. $x_j \ge 0, j = 1, ..., n$
where $b_i \ge 0, n > m$

Transforming an LP to Standard Form

- Nonzero lower bound
  suppose $x_j \ge l_j, l_j \ne 0$, replace $x_j = x'_j + l_j, x'_j \ge 0$
- Non-positive upper bound
  suppose $x_j \ge u_j, u_j \le 0$, replace $x_j = u_j - x'_j$
- Unrestricted (or Free) Variables
  - $x_1 + x_2 = 8$, replace $x_2 = 8 - x_1$ if x_2 is free
  - define $x_j^+, x_j^- \ge 0$, use $x_j^+ - x_j^-$ substitute x_j
- Inequality Constraints
  define slack variable $s_1 \ge 0$, $ax \ge b \leftrightarrow ax - s_1 = b$

1.2 Solving LP

Basic Feasible Solution

- **Definition**. For an LP in the standard form, a basic solution is called a BFS if it satisfies the non-negativity constraints.
- **Theorem**
  - $\exists$ a feasible solution $\leftrightarrow$ $\exists$ a BFS.
  - $\exists$ an optimal FS $\leftrightarrow$ $\exists$ an optimal BFS.
- **Additional remark**. Each BFS corresponds to a corner point in the graphic representation of LP.

Given $\mathbf{x} = (\mathbf{x}_B, \mathbf{x}_N)$ with $\mathbf{x}_N = 0$. Write $\mathbf{A} = (\mathbf{B}, \mathbf{N})$ (columns of $\mathbf{A}$ corresponding to variables in $\mathbf{x_B}, \mathbf{x_N}$)
$\mathbf{Ax} = \mathbf{Bx_B} + \mathbf{Nx_N} = \mathbf{Bx_B} = \mathbf{b}, \mathbf{x_B} = \mathbf{B^{-1}b}$
Proof. Assume $\mathbf{x}$ is a feasible solution, but not BFS. Can find a $\mathbf{y}$ ($\mathbf{Ay} = 0$) making $\mathbf{x} + k\mathbf{y}$ to be BFS.

Simplex method A simplex form $\leftrightarrow$ a BFS:

- Each basic variable corresponds to a row, and value of basic variable is right-hand side of row
- The value of the objective function is equal to the right-hand side of row 0
- Every basic variable appears in one and only one equation, but not row 0
- Each basic variable has the coefficient 1 in the equation it appears
- Each equation has only one basic variable
- Variable z only appears in row 0 with coefficient 1

The BFS is optimal iff row 0 has no negative numbers

Optimality Test Suppose $\mathbf{x} = (\mathbf{x_B}, \mathbf{x_N})$ is a BFS
$\mathbf{x_B} = \mathbf{B^{-1}(b - Nx_N)} = \mathbf{B^{-1}b}$
$\mathbf{z} = \mathbf{c_B x_B} + \mathbf{c_N x_N} = \mathbf{c_B B^{-1} b}$
if non-basic variable becomes non-zero:
$\mathbf{z} = \mathbf{c_B x_B} + \mathbf{c_N x_N} = \mathbf{c_B B^{-1}(b - Nx_N)} + \mathbf{c_N x_N}$
$= \mathbf{c_B B^{-1} b} - (\mathbf{c_B B^{-1} N} - \mathbf{c_N})\mathbf{x_N}$
consider a non-basic variable x_k increased
$\mathbf{z} = \mathbf{c_B B^{-1} 1 b} - (\mathbf{c_B B^{-1} A_k} - \mathbf{c_k})\mathbf{x_k}$
$\bar{c_k} = \mathbf{c_B B^{-1} A_k} - c_k$ referred as reduced cost
The current basic solution is optimal if and only if the reduced cost is nonnegative for all non-basic variables.

Ratio Test non-basic variable x_k increased
$\mathbf{x_B} = \mathbf{B^{-1}(b - Nx_N)} = \mathbf{B^{-1}b} - \mathbf{B^{-1}A_k x_k}$
To keep non-negativity, $(\mathbf{x_B})_\mathbf{i} \ge 0$
ratio test $\underset{i}{\operatorname{argmin}} (\mathbf{B^{-1}b})_\mathbf{i} / (\mathbf{B^{-1}A_k})_\mathbf{i}$

1.3 Sensitivity Analysis

Shadow Price Optimal of dual variable: $\lambda = \mathbf{c_B B}^{-1}$
In the optimal solution, if a constraint is not tight ($<$ or $>$), then its shadow price must be 0, if tight ($=$), may or may not be 0

Constraint Analysis

- Non-basic variable $\mathbf{c}_j + \Delta c_j$
  $\mathbf{c_B B^{-1} A_j} - (c_j + \Delta c_j) \ge 0$
- Basic variable $\mathbf{c}_j + \Delta c_j$
  $(\mathbf{c_B} + \Delta c_j)\mathbf{B^{-1}}(\mathbf{A_N}, \mathbf{I}) - (\mathbf{c_N}, \mathbf{0}, ..) \ge 0$
  optimal solution unchange within range, vice versa
- $\mathbf{b}_r$: $\mathbf{x'_B} = \mathbf{B^{-1}b'} \ge 0$, $\Delta S = \Delta \mathbf{b}_r \cdot \lambda_r$
  optimal solution always change. Basic variable change beyond range, vice versa.

1.4 Duality

Primal: max $z = \sum_{i=1}^{n} c_i x_i$
s.t. $\sum_{i=1}^{n} a_{ji} x_i \le b_j, j = 1, ..., m$
$x_i \ge 0, i = 1, ..., n$
Dual: min $w = \sum_{j=1}^{m} b_j y_j$
s.t. $\sum_{j=1}^{m} a_{ji} y_j \ge c_i, i = 1, ..., n$
$y_j \ge 0, j = 1, ..., m$

Primal model (MAX)	Dual model (MIN)
Constraint j is $\le$	Variable $y_j \ge 0$
Constraint j is $=$	Variable y_j is free
Constraint j is $\ge$	Variable $y_j \le 0$
Variable $x_i \ge 0$	Constraint i is $\ge$
Variable x_i is free	Constraint i is $=$
Variable $x_i \le 0$	Constraint i is $\le$

Weak Duality $Z(x) \le W(y)$
proof: $\mathbf{cx} \le \mathbf{yAx} \le \mathbf{yb}$

Strong Duality If either of Primal or Dual has an optimal feasible bounded solution, then 1. the other problem also has *ofbs*, 2. $z = w$
proof: suppose $\mathbf{x}^* = (\mathbf{x_B}, \mathbf{x_N})$ is optimal solution,
Reduced cost for non-basic variables $\mathbf{c_B B^{-1} N} - \mathbf{c_N} \ge 0$,
Let $\mathbf{y} = \mathbf{c_B B^{-1}}$, we have:

- $\mathbf{yA} = \mathbf{y}(\mathbf{B}\ \mathbf{N}) = (\mathbf{c_B}, \mathbf{c_B B^{-1} N}) \ge (\mathbf{c_B}, \mathbf{c_B}) = \mathbf{c}$
- $\mathbf{y}$ is feasible to the dual
- $\mathbf{yb} = \mathbf{c_B B^{-1} b} = \mathbf{c_B x_B} = \mathbf{cx}^*$
- $\mathbf{y}$ is optimal to the dual

Complementary Slackness Any feasible solution $\mathbf{x}, \mathbf{y}$, they both are optimal iff for any i:
(1) $\mathbf{x_i} > 0 \rightarrow (\pi \mathbf{A})_\mathbf{i} = \mathbf{c_i}$, (2) $\mathbf{x_i} = 0 \leftarrow (\pi \mathbf{A})_\mathbf{i} > \mathbf{c_i}$
proof: If 1,2 true, $(\mathbf{yA} - \mathbf{c})\mathbf{x} = \mathbf{0} \rightarrow \mathbf{yAx} = \mathbf{cx}$ or $\mathbf{yb} = \mathbf{cx}$, from strong duality, $\mathbf{x}, \mathbf{y}$ optimal.
If $\mathbf{x}, \mathbf{y}$ optimal, $\mathbf{yb} = \mathbf{cx}$ and $(\mathbf{yA} - \mathbf{c})\mathbf{x} = \mathbf{0}$, implies 1,2.

1.5 Transportation Problem

source with supplying capacity s_i, destination with demand d_i, cost c_{ij}, transportation plan x_{ij}
min $\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$
$\sum_{j=1}^{n} x_{ij} = s_i, i = 1, ..., m$
$\sum_{i=1}^{m} x_{ij} = d_i, i = 1, ..., n$
$x_{ij} \ge 0, i = 1, ..., m, j = 1, ..., n$.
If total supply = demand, coefficience matrix contains $k \le m + n - 1$ linearly independent row vectors, k basic variables, k non-zero elements in optimal solution.
$\mathbf{p} = \mathbf{c_B B^{-1}}$ is basic variables for dual problem, contains k elements. Thus always a shadow price = 0.

1.6 Cooperative Game

N: set of players
$V(S)$: function gives cost of a coalition S(subset of N)
An allocation is a distribution of Ground Coalition V(N), s.t. $\sum_{i \in N} x_i = V(N)$
develop core allocation rules $\sum_{i \in S} x_i \le V(S), \forall S$

1.7 Max Flow Problem

$G = (N, A)$, flow on arc x_{ij}, capacity of flow in arc U_{ij}, source node s, destination node t, Max flow value $v(x)$.
Original network G, feasible flow x, residual network $G(x)$, residual capacity r_{ij}, argumenting path P.

Min cut The capacity of a cut (S,T) is sum of capacities of all forward arcs, CAP(S,T)=$\sum_{i \in S} \sum_{j \in T} u_{ij}$.
Dual LP (min cut): $\pi_i \in \{0, 1\}$ two sets, w_{ij} cut edge
min $\sum u_{ij} w_{ij}$ *s.t.* $\pi_1 - \pi_i + w_{1i} \ge 0, \pi_i - \pi_n + w_{in} \ge 0, -\pi_1 + \pi_n \ge 1, w_{ij} \ge 0, \pi$ free

Weak Duality Theorem for Max Flow Problem
define **flow across the cut**: flow on forward arcs - backward $F_x(S, T) = \sum_{i \in S} \sum_{j \in T} x_{ij} - \sum_{i \in S} \sum_{j \in T} x_{ji}$
Claim: 1. $F_x(S, T) = v$ =flow into t. 2. $F_x(S, T) \le$ capacity of a cut. **Weak**: $v(x) \le CAP(S, T)$

Strong Duality: Max Flow Min Cut Theorem
The following are equivalent. $1 \Rightarrow 2, 3 \Rightarrow 1, 2 \Rightarrow 3$

1 A flow x is maximum
2 There is no augmenting path in $G(x)$.
3 There is an s-t cut (S, T) whose capacity is the flow value of x.
* **Corollary**. The maximum flow value is the minimum value of a cut

1.8 Min Cost Network Flow

- **Problem input**
  - Network G=(N,A)
  - Flow cost c_{ij} for each arc (i, j) in A
  - Lower and upper bounds l_{ij}, u_{ij} for each arc
  - Supply or demand $b(j)$ for each node j
- **Decisions** Flow x_{ij} for each arc
- **Objective** min total flow cost $\sum_{(i,j)} x_{ij} c_{ij}$
- **Constraints**
  - Lower and upper bounds
  - Flow conservation
  - A necessary condition for the problem to be feasible is that total supply is equal to total demand

LP Formulation min $\mathbf{cx}$ *s.t.* $\mathbf{Ax} = \mathbf{b}, \mathbf{x} \ge 0$
$A_{ij} \in \{-1, 0, 1\}$, each column has exactly one 1 & -1
LP with Consecutive 1' s in Columns (Each row k is multiplied by -1 and added to row k+1) $\rightarrow$ IP

1.9 IP

LP with $x_i \in \{0,1\}$, LP is relaxation of IP.

Constraints

- $y_{A1} + y_{A2} + y_{A3} \leq 3y_{A4}$ Design A can be used at sites 1,2,3 only if it is also selected for site 4
- $\sum y_i \leq nw$ Decision w is implied if any one of the other n decisions is chosen
- $\sum y_i \leq n - 1 + w$ Decision w is implied if all the other n decisions are chosen
- $\sum y_i \leq k-1+(n-(k-1))w$ Decision w is implied if at least k of the other n decisions are chosen

B&B Tree

1.10 Dynamic Program

dp recursion, initial condition, optimal solution

1.11 Complexity Analysis

$O(f(n))$ if at most $cf(n)$ for all $n \geq n_0$

Recognition Version of Optimization .

optimization problem P1: minimizing $f(x)$
recognition problem P2: ?$\exists x_0$ s.t. $f(x_0) \leq k$
P1 $\rightarrow$ P2, P2 $\underset{log}{\rightarrow}$ P1

Problem Reduction P1 polynomially reduces to P2: exists a polynomial-time algorithm that solves P1 by using the algorithm for solving P2 at unit cost

Problem Transformation If for every instance A1 of problem A, we can construct in polynomial-time an instance B1 of B, such that A1 has a Yes answer if and only if B1 has a Yes answer, then we say A polynomially transforms to B

Problem Classification .

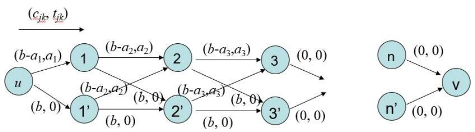
P∈NP, NPC = NP∩NP-hard

- **P** polynomial
- **NP** verify in polynomial
- **NP-complete** (1) P1 $\in$ NP (2) all other problems in class NP polynomially transform to P1
- **NP-hard** all other problems in class NP polynomially reduce to P1

NPC: Hamiltonian Cycle, Partition, Traveling Salesman, Longest Path

Proving a problem is NPC (1) P2 $\in$ NP (2) find P1 $\in$ NPC, polynomially transform P1 to P2

- **TSP** Given an instance of HCP with G=(N,A), we construct an instance of TSP with G=(N,A) and $c_i j = 1$ for all arc (i,j) in A, and the integer k=n. A Hamiltonian cycle in HCP is equivalent to a tour in TSP with total cost being n(=k).
- **LPP** s→t cost ≥ L? network G' inheriting G except that G' has on more node n+1 and all arcs (j,1) in G are replaced by arcs (j,n+1), each arc in G' having a cost being 1, and node s being node 1, node t being node n+1, and L=n
- **Dynamic Lot Sizing with Production Capacity(DLSC)** production capacity u_t, demand d_t, inventory holding cost $h_t(y_t) = h_t y_t$, production cost $f_t(x_t) = K_t + c_t x_t$
  Partition, construct DLSC of T=n periods. Each period t, t= 1,...,n, the production capacity $u_t = a_t$, holding cost $h_t(y_t) = 0$, and production cost $f_t(x_t) = a_t$, if $x_t > 0$, ie., $c_t = 0$. The demand $d_t = 0, t = 1, ..., n-1$ and $d_T = B$.
- **Constrained Shortest Path (CSP)** Partition
  csp: u to v, $\sum c_{jk} \leq C, \sum t_{jk} \leq T$



Weakly & Strongly NPC strongly: NPC even under the similarity assumption
weakly: has pseudo-polynomial time algorithm

1.12 Handling NP-hard

Bounding by Relaxation $Pi : \min\{F_i(x)|x \in X_i\}$
P_j is called a relaxation of P_i if (1) $X_i \subseteq X_j$, (2) $F_i(x) \geq F_j(x)$. Then $F_i(x^*_{fi}) \geq F_j(x^*_{fj})$

Lagrangian Relaxation max problem:
$z_{IP} = \max cx$ s.t. $A_1 x \leq b_1, A_2 x \leq b_2, x \in Z^n_+$
$z_{LR}(\lambda) = \max cx + \lambda(b_1 - A_1 x)$ s.t. $A_2 x \leq b_2, x \in Z^n_+$
$\forall \lambda \geq 0, z_{LR}(\lambda) \geq z_{IP}$
Lagrangian Dual $Z_{LD} = \min\{z_{LR}(\lambda)|\lambda \geq 0\}$

LP and Lagrangian Relaxation .

LP: $Z^* = \min\{cx|Ax = b, Dx = e, x \geq 0\}$

Dual: $Z^* = \max\{ub + ve|uA + vD \leq c\}$
optimal solution satisfy: $(u^*A + v^*D - e)x^* = 0$
LR: $L(\lambda) = \min\{cx + \lambda(b - Ax)|Dx = e, x \geq 0\}$
$L(u^*) = \max\{L(\lambda)\} = Z^*$

Dual .

LP: $\min\{cx|Ax = b, x \geq 0\}$ LPD: $\max\{yb|yA \leq c\}$
$y^*A - c = 0$ by Complementary Slackness
$L(u) = \min\{cx + u(b - Ax)|x \geq 0\}, L(u^*) = \max L(u)$
let $u = y^*, L(y^*) = \min\{(c - y^*A)x + y^*b|x \geq 0\}$ =LPD

2 Non-linear Programming

$\min f(\mathbf{x})$ s.t. $g_i(\mathbf{x}) \leq b_i, i = 1, ..., m$

Convex Function - Concave Function
$f(\alpha x_1 + (1-\alpha)x_2) \leq \alpha f(x_1) + (1-\alpha)f(x_2)$
1-D: $f(\cdot)$ is convex if $f''(\cdot) \geq 0$
Multiple-Dimension: Hessian matrix $\mathbf{H}_{ij} = \partial^2 f/\partial x_i \partial x_j$ is positive semi-definite $\mathbf{x}^T H \mathbf{x} \geq 0, \forall \mathbf{x}$

Convex Set $\forall \mathbf{x}_1, \mathbf{x}_2 \in S, \alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2 \in S$
if $g_i(\cdot)$ is convex function, the feasible region is convex

Convex Programming NLP, convex set, convex function, then local minimum is global minimum

Operations that preserve convexity .

Suppose $f_i(\mathbf{x})$ are convex functions

- $\sum w_i f_i(\mathbf{x})$ is convex if $w_i \geq 0$
- $\max\{f_i(\mathbf{x})\}$ is convex, proof: definition inequality

Consider $f(x) = h(g(x))$, then f is convex(concave) if h is convex(concave) and non-decreasing(increasing), and g is convex or concave.

2.1 Find the Minimum

Newton's Method $\forall x_k$, $q(x) := f(x_k) + f'(x_k)(x - x_k) + 0.5f''(x_k)(x - x_k)^2$
find $q'(x_{k+1}) = f'(x_k) + f''(x_k)(x_{k+1} - x_k) = 0$ i.e., $x_{k+1} = x_k - f'(x_k)/f''(x_k)$

NLP with Equality Constraints .

$\min f(\mathbf{x})$ s.t. $g_i(\mathbf{x}) = 0, i = 1, ..., m$
$L(x, \lambda) = f(x) + \sum \lambda_i g_i(x)$
stationary point: $\partial L/\partial x_i = \partial L/\partial \lambda_i = 0$

NLP with Inequality Constraints $\min f(\mathbf{x})$ s.t. $h_i(\mathbf{x}) = 0, i = 1, ..., p. g_i(\mathbf{x}) \leq 0, i = 1, ..., m$
$L(x, \lambda, \mu) = f(x) + \sum \lambda_i h_i(x) + \sum \mu_i g_i(x), \mu_i \geq 0$

Kuhn-Tucker necessary conditions: $\partial L/\partial x_j = \partial L/\partial \lambda_i = 0, \partial L/\partial \mu_i \leq 0, \mu_i g_i(x) = 0, \mu_i \geq 0$

2.2 Selected OR Models and Methods

Minimum spanning tree

- **Kruskal**' s Algorithm $O(nm)$
  Cut Optimality Condition, min-max route
- **Prim**' s Algorithm $O(m + n \log n)$
- **Sollin**' s Algorithm $O(m \log n)$
  each step find min arc for each sub-tree

Bin packing

- **Next Fit** F(NF)<=2OPT-1
- **First Fit** F(FF)<=1.7OPT
- **Best Fit**

Shortest path Floyd-Warshall $O(n^3)$