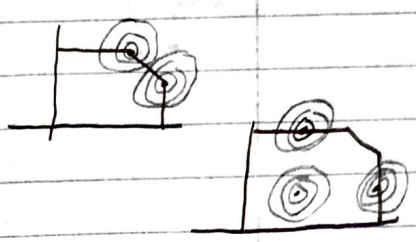
Intro NLP, Defn Concavity, Thm $\text{local max} \Rightarrow \text{global max}$, $f''(x) \geq 0$ or ≤ 0

Non-Linear Programming

- solutions not necessarily at a corner point
- can have finitely many optimal solutions
- optimal solutions in the interior
- local optima are not necessarily global optima

Applications

* Portfolio Allocation
* Profit Maximization
* Volume Maximization



Suppose you have n investments. The return on investment i in the next period is R_i, a random variable. Let x_i be the percentage of money to invest in i.

$$\text{Max } Z = \underbrace{E\left[\sum_{i=1}^{n} x_i R_i\right]}_{\text{return}} - \underbrace{\mu}_{\text{indicates level of risk willingness}} \text{Var}\left[\sum_{i=1}^{n} x_i R_i\right]$$

Constraints: $\sum_{i=1}^{n} x_i = 1$, $x_i \geq 0 \ \forall i$ $\Rightarrow Z = \sum_{i=1}^{n} x_i E(R_i) - \mu \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j \text{cov}(R_i, R_j)$ ← non-linear!

Solve $f'(x) = 0$. Is this sufficient and necessary for x to be a max? NO!

→ not sufficient: min, plateau, local max, local min

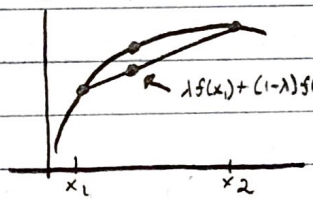
→ not necessary! not differentiable, boundary max or min, increasing function

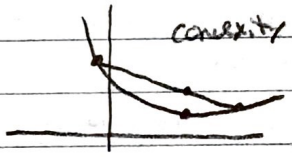
Want f continuous, differentiable, domain of f is unbounded, concave!

$\Rightarrow$ $f'(x_0) = 0$ iff x_0 is a global max.

Defn: A function is concave if for any x_1, x_2 in the domain of f and $\lambda \in [0,1]$ we have

$$f(\lambda x_1 + (1-\lambda)x_2) \geq \lambda f(x_1) + (1-\lambda) f(x_2)$$





Defn: A function is convex if " $\leq$ "

Theorem: If f is concave on $\mathbb{R}$, then any local maximum of f is a global max of f.

Proof: Let $\bar{x}$ be a local max of f. Suppose $\bar{x}$ is not a global max. Then there exists an $\bar{x}' \neq \bar{x}$ s.t. $\bar{x}'$ is a global max of f and $f(\bar{x}') > f(\bar{x})$. Thus, for $\lambda \in [0,1]$

$$f(\lambda\bar{x} + (1-\lambda)\bar{x}') \geq \lambda f(\bar{x}) + (1-\lambda) f(\bar{x}') > \lambda f(\bar{x}) + (1-\lambda) f(\bar{x}) = f(\bar{x})$$

Since we can choose λ arbitrarily close to 1, we can find find points with larger function values which violates that $\bar{x}$ is a local max. ↯ Thus $\bar{x}$ is a global max ■

Thm: concavity $\Rightarrow$ continuity. no proof given.

Thm: If f'' exists everywhere then $f''(x) \geq 0$ everywhere $(\Leftarrow)$ f is convex

$f''(x) \leq 0$ everywhere $(\Rightarrow)$ f is concave

Solving $f'(x)=0$. Use numerical approximations because solving analytically can be a pain.

→ **Bisection Method:** (f is concave) Looking for global max x^* know lower & upper bound $\underline{x}$ and $\bar{x}$.

Let $\hat{x} = \frac{\underline{x}+\bar{x}}{2}$. If $f'(\hat{x})>0$ then $\underline{x} := \hat{x}$, If $f'(\hat{x})<0$ then $\bar{x} := \hat{x}$.

Stop when $\hat{x}$ is within ϵ of the optimal solution (pick ϵ in advance) $\Leftrightarrow$ $\bar{x} - \underline{x} < 2\epsilon$

* See handout for example with $f(x) = e^x \cos(x) + x$, Max on $[0, \pi]$.

Bisection method takes K iterations where $\frac{\Delta \bar{x}}{2^k} \leq 2\epsilon \Rightarrow K = \log_2\left(\frac{\Delta \bar{x}}{\epsilon}\right) - 1$

→ **Newton's Method:** root finding method. Converges very quickly to optimal solutions.

Given x_i, The equation of the tangent line $(x-x_i)m = (y-y_i)$

$\Rightarrow y = m(x-x_i) + y_i$. $y_i = f(x_i)$, $y=0$, $m = f'(x_i)$

$\Rightarrow 0 = f'(x_i)(x-x_i) + f(x_i)$ $\Rightarrow x_{i+1} = x_i - \frac{f(x_i)}{f(x_{i+1})}$

But, we want $f'(x)=0$, so $x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)}$

→ **Screwing with Newtons Method**

loop!

Multivariate NLP

① All partial derivatives $=0$ OR ② on a boundary OR ③ where at least one partial derivative DNE

$\nabla f \equiv$ grad f is the vector of partial derivatives $\nabla f(x,y) = f_x(x,y)\hat{i} + f_y(x,y)\hat{j}$

① $\Leftrightarrow$ $\nabla f(x,y) = \vec{0}$

∇f points in the direction of maximum increase for f.

Directional Derivative: $f_{\vec{u}}(x,y) = \nabla f(x,y) \cdot \vec{u}$ ($\vec{u}$ is a unit vector)

$\vec{u}\cdot\vec{v} = \|\vec{u}\|\,\|\vec{v}\| \cos\theta$ where θ is the angle between $\vec{u}$ and $\vec{v}$.

$f_{\vec{u}}(x,y) = \|\nabla f(x,y)\|\,\|\vec{u}\| \cos\theta = \|\nabla f(x,y)\| \cos\theta$ what value of θ maximizes this? $\theta = 0$!

Hessian matrix $H(f) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix}$ look at $\det(H)$

If $\nabla f(a,b) = \vec{0}$, $\#\$$ $\det(H(a,b)) = 0$ Then second partials test fails

$\$\$$ $\det(H(a,b)) < 0$ you have a saddle point at (a,b)

$\$\$$ $\det(H(a,b)) > 0$ $\#\$$ f_{xx} or $f_{yy} > 0$ $\Rightarrow$ local min at (a,b)

$\$\#$ f_{xx} or $f_{yy} < 0$ $\Rightarrow$ local max at (a,b)

$d(a,b) = f_{xx}(a,b)\, f_{yy}(a,b) - [f_{xy}(a,b)]^2$

If f_{xx} & f_{yy} are both positive why don't we necessarily have a min at (a,b)? Because of twisting

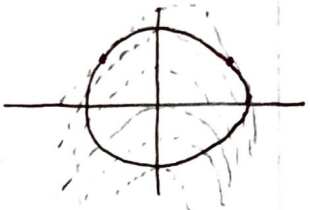
$f(x,y) = x^4 + y^4 - 4xy + 1$: $\nabla f(x,y) = (4x^3 - 4y)\hat{i} + (4y^3 - 4x)\hat{j} \Rightarrow \nabla f(x,y) = 0 \Leftrightarrow \begin{matrix} 4x^3 - 4y = 0 & \Leftrightarrow & x^3 = y \\ 4y^3 - 4x = 0 & \Leftrightarrow & y^3 = x \end{matrix}$

$x = 0, \pm 1$ If $x=0$ $y=0$, if $x=1$, $y=1$, if $x=-1$, $y=-1$ next look at Hessian.

$\det(H)$ $(0,0)$ < 0 $\Rightarrow$ saddle

$(1,1)$ > 0, $f_{xx} > 0$ local min

$(-1,-1)$ > 0 $f_{xx} > 0$ local min



Lagrange Multipliers: Multivariate constrained optimization

Maximize $f(x,y) = x^2+y$ s.t. $x^2+y^2=1$

∇f is perpendicular to circle at optimal point. $g(x,y) = x^2+y^2$ when $g=1$. Think of circle as a level curve

∇g at optimal points perpendicular to the circle $\Rightarrow$ $\boxed{\nabla f(x,y) = \lambda \nabla g(x,y)}$ λ = Lagrange Multiplier

$\nabla f(x,y) = \langle 2x, 1\rangle$ $\nabla g(x,y) = \langle 2x, 2y\rangle$

$2x = \lambda 2x$ $\quad 1 = \lambda 2y$ $\quad x^2+y^2=1$ $\Rightarrow (0,1), (0,-1), \left(\pm\frac{\sqrt{3}}{2}, \frac{1}{2}\right)$

First method to produce max & min on interior. Use Lagrange Multipliers to find max/min on boundaries.

What does λ mean? - Let (x_0, y_0) be the optimal solution to maximize $f(x,y)$ s.t. $g(x,y)=c$.

express as $(x_0(c), y_0(c))$. How does $f(x_0(c), y_0(c))$ change as c changes?

$\frac{df}{dc}\Big|_{(x_0,y_0)} = \frac{df}{dx_0}\frac{dx_0}{dc} + \frac{df}{dy_0}\frac{dy_0}{dc}$ $\quad \left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial y_0}\right) = \lambda\left(\frac{\partial g}{\partial x_0}, \frac{\partial g}{\partial y_0}\right)$

at a point of optimal value. $\Rightarrow \frac{df}{dc}\Big|_{(x_0,y_0)} = \lambda\frac{\partial g}{\partial x_0}\frac{dx_0}{dc} + \lambda\frac{\partial g}{\partial y_0}\frac{dy_0}{dc} = \lambda\frac{\partial g}{\partial c}\Big|_{(x_0,y_0)} = \lambda \rightarrow$ shadow price

λ is the shadow price of f with respect to the constraint $g(x,y)=c$ [note: only instantaneous shadow price]

$f''(x) \le 0 \Leftrightarrow f$ is concave (if f'' exists $\forall x \in \mathbb{R}$)

Characterization of Concavity

Defn: A function $f(\vec{x})$ is concave, if $\forall \vec{x}_1, \vec{x}_2 \in \mathbb{R}^n$

$f(\lambda\vec{x}_1 + (1-\lambda)\vec{x}_2) \ge \lambda f(\vec{x}_1) + (1-\lambda) f(\vec{x}_2)$

Alternate characterization: Suppose f has continuous first partials in $\mathbb{R}^n$, then

f is concave $\Leftrightarrow$ $f(\vec{x}) \le f(\vec{x}_0) + \nabla f(\vec{x}_0)(\vec{x}-\vec{x}_0)$ $\forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$

$\rightarrow$ Single variable $y = f(x_0) + f'(x_0)(x-x_0)$

Max $f(x,y)$
s.t. $g(x,y)=c$
$h(x,y)=d$
$\Big\}$ $\nabla f = \lambda\nabla g + \mu\nabla h$

Taylor's Theorem in 2 variables (up to first order)

If f is twice-differentiable on $\mathbb{R}^2$ then for some (a,b) in the line segment between (x,y) and (x_0,y_0)

$f(x,y) = f(x_0,y_0) + f_x(x_0,y_0)(x-x_0) + f_y(x_0,y_0)(y-y_0) + \frac{1}{2}\left[f_{xx}(a,b)(x-x_0)^2 + 2f_{xy}(a,b)(x-x_0)(y-y_0) + f_{yy}(a,b)(y-y_0)^2\right]$

Using ∇f and matrices this is $f(x,y) = f(x_0,y_0) + \nabla f(x_0,y_0)\cdot\begin{bmatrix} x-x_0, y-y_0\end{bmatrix} + \frac{1}{2}\begin{bmatrix} x-x_0, y-y_0\end{bmatrix}\begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy}\end{bmatrix}\begin{bmatrix} x-x_0 \\ y-y_0\end{bmatrix}$

for $S: \mathbb{R}^n \rightarrow \mathbb{R}$, this is $f(\vec{x}) = f(\vec{x}_0) + \nabla f(\vec{x}_0)[\vec{x}-\vec{x}_0] + \frac{1}{2}[\vec{x}-\vec{x}_0]^T H(\vec{z})[\vec{x}-\vec{x}_0]$ for some $\vec{z}$

Note: $\vec{x}-\vec{x}_0$ is a column vector that is a convex combination of $\vec{x}$ and $\vec{x}_0$

★ **Theorem:** Suppose f has continuous second partial derivatives in $\mathbb{R}^n$. Then f is concave $\Leftrightarrow$ its Hessian Matrix is negative semi-definite.

Defn: A is negative semi-definite $\Leftrightarrow$ $\vec{x}^T A \vec{x} \le 0$ $\forall \vec{x} \in \mathbb{R}^n$ (or $\mathbb{C}^n$).

Proof: H is negative semidefinite $\forall \vec{z} \in \mathbb{R}^n \Leftrightarrow [\vec{x}-\vec{x}_0]^T H(\vec{z})[\vec{x}-\vec{x}_0] \le 0$ $\forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$

$\Leftrightarrow$ $f(\vec{x}) \le f(\vec{x}_0) + \nabla f(\vec{x})[\vec{x}-\vec{x}_0]$ $\forall \vec{x}, \vec{x}_0 \in \mathbb{R}^n$

$\Leftrightarrow$ f is concave on $\mathbb{R}^n$ ■

Ex: $f(x_1,x_2,x_3) = 4x_1^2 + x_2^2 + x_3^2 - 2x_2x_3$ is convex. (Positive semi-definite Hessian)

details
↓

$$[x_1\ x_2\ x_3]\begin{bmatrix} 8 & 0 & 0 \\ 0 & 2 & -2 \\ 0 & -2 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 8x_1^2 + 2x_2^2 + 2x_3^2 - 4x_2x_3$$

$$= 8x_1^2 + 2(x_2^2 - 2x_2x_3 + x_3^2) = 8x_1^2 + 2(x_2 - x_3)^2 \Rightarrow \text{PSD Hessian} \Leftrightarrow \text{convex!}$$

Other characteristics of NSD

① All eigenvalues are non-positive (if)

② See text on Matrix Algebra for characterization terms of principle minors.

Thm: If $f_i(\vec{x})$ is concave, then $\sum_{i=1}^{n} f_i(\vec{x})$ is also

Thm: If $f(\vec{x})$ is concave, and $q > 0$ then

$q f(\vec{x})$ is concave AND $-q f(\vec{x})$ is convex.

Thm: If $f(\vec{x})$ is concave and $g(y)$ is increasing then $g(f(\vec{x}))$ is concave

Ex: $f(x_1, x_2, x_3) = e^{-(x_1^2 + x_2^2 + x_3^2)}$

e^x is increasing, $-x_1^2 - x_2^2 - x_3^2$ is concave

thus f is concave. Also

$$H_f(x,y) = \begin{bmatrix} -2 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -2 \end{bmatrix} \Rightarrow \text{NSD} \Rightarrow \text{concave.}$$

Last time: f concave $\Leftrightarrow$ Hessian of f is negative semidefinite.

3/24

Today: Numerical technique for multivariable optimization.

- want to move perpendicularly to the level curve (in direction of the gradient)
- how far do you move in the direction of the gradient? → move until you stop increasing

→ Specifics: Suppose we're at $\vec{x}$ and moving in the direction of grad $f(\vec{x})$. Our next point is $\vec{x} + t\nabla f(\vec{x})$ for some t. What's the right choice for t?

Maximize $f(\vec{x} + t\nabla f(\vec{x}))$ – single variable optimization, t is a variable

How do we know when to stop? – Book recommends stopping when $|\nabla f(\vec{x}_i^i)| < \epsilon \ \forall j$ (i = the iteration #) "gradient search" aka "steepest ascent"

→ Ex: Maximize $f(x_1, x_2) = x_1x_2 - 3x_2 - x_1^2 - x_2^2$ using gradient search with $\epsilon = 0.01$

Start at $(0,0)$ $\quad \nabla f(x_1, x_2) = (x_2 - 2x_1)\hat{i} + (x_1 - 3 - 2x_2)\hat{j}$

$\nabla f(0,0) = 3\hat{j}$ Now find t. (Iteration 1) Find maximizer t^*

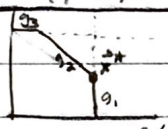
Maximize $f((0,0) + t(0,3)) = f(0,3t) = 9t - 9t^2$ $\Rightarrow$ max at $\frac{-9}{-18} = \frac{1}{2} = t^*$

Next point: $(0, 3/2)$ $\quad \nabla f(0, 3/2) = (3/2, 0) \Rightarrow$ Iteration 2 $f((0,3/2) + t(3/2, 0)) = f(t\,3/2, 3/2)$

$= \left(\frac{3}{2}\right)^2 t + \frac{9}{2} - \left(\frac{3}{2}\right)^2 t^2 - \left(\frac{3}{2}\right)^2 \quad t^* = \frac{-\left(\frac{3}{2}\right)^2}{-2\left(\frac{3}{2}\right)^2} = 1/2 \Rightarrow$ next point is $(3/4, 3/2)$

$\nabla f\left(\frac{3}{4}, 3/2\right) = \frac{3}{4}\hat{j} \Rightarrow$ keep iterating. TOR tutorial gives optimal solution at $(0.996, 1.998)$, $f(x) =$ [illegible]

Linear



in order for $\vec{x}^*$ to be optimal, the level curves of f must be steeper than g_2, but not as steep as g_1.

∇f must be between ∇g_1 and ∇g_2.

thus $\nabla f(\vec{x}^*) = u_1 \nabla g_1(\vec{x}^*) + u_2 \nabla g_2(\vec{x}^*)$ where $u_1, u_2 \geq 0$ $\quad \nabla f$ is in the cone of ∇g_1 and ∇g_2

→ General NLP Maximize $f(\vec{x})$ s.t. $g_i(\vec{x}) \leq b_i$ for $1 \leq i \leq n$, $i \in N$, $\vec{x} \geq \vec{0}$

→ For $\vec{x}^*$ to be optimal we must have:

① $\nabla f(\vec{x}^*)$ is a linear combination of the gradients of the binding constraints.

② The multipliers in the linear combo are non-negative

③ $\vec{x}^*$ satisfies the constraints in the problem.

★★ → KKT conditions: If f and all the g_i's are differentiable, f is concave, the g_i's are convex, and the vectors in $\{\nabla g_i(\vec{x}^*) \mid g_i(\vec{x}^*) = b_i\}$ are linearly independent then: the following are necessary and sufficient conditions for the $\vec{x}^*$ to be optimal for the general NLP problem:

3/25

① • $\nabla f(\vec{x}^*) = \sum_{i=1}^{m} u_i \nabla g_i(\vec{x}^*) - \vec{v}$ ← accounts for non-negativity constraints

② • $g_i(\vec{x}^*) - \vec{b}_i \leq \vec{0}$ for each i

③ • $u_i(g_i(\vec{x}^*) - b_i) = 0$ for each i

④ • $-\vec{x}^* \leq 0$

⑤ • $v_j x_j = 0$ for each j

⑥ • $\vec{u}, \vec{v} \geq \vec{0}$

General NLP Problem

Max $f(\vec{x})$

s.t. $g_i(\vec{x}) \leq b_i \ \forall i \quad i \in \{1, \ldots, m\}$

$\vec{x} \geq \vec{0}$

3, 5 if constraints not binding, then coefficient is zero.

Last time: f concave $\Leftrightarrow$ Hessian of f is negative semidefinite.

3/24

Today: Numerical technique for multivariable optimization.

- want to move perpendicularly to the level curve (in direction of the gradient)
- how far do you move in the direction of the gradient? → move until you stop increasing

→ Specifics: Suppose we're at $\vec{x}$ and moving in the direction of grad $f(\vec{x})$. Our next point is $\vec{x} + t\nabla f(\vec{x})$ for some t. What's the right choice for t?

Maximize $f(\vec{x} + t\nabla f(\vec{x}))$ - single variable optimization, t is a variable

How do we know when to stop? - Book recommends stopping when $|\nabla f(\vec{x}_j^i)| < \epsilon \ \forall j$ (i is the iteration #) "gradient search" aka "steepest ascent"

→ Ex: Maximize $f(x_1, x_2) = x_1x_2 - 3x_2 - x_1^2 - x_2^2$ using gradient search with $\epsilon = 0.01$

Start at $(0,0)$ $\quad \nabla f(x_1, x_2) = (x_2 - 2x_1)\hat{i} + (x_1 - 3 - 2x_2)\hat{j}$

$\nabla f(0,0) = 3\hat{j}$ Now find t. (Iteration 1) Find maximizer t^*

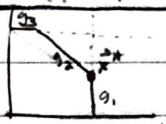
Maximize $f((0,0) + t(0,3)) = f(0,3t) = 9t - 9t^2$ $\Rightarrow$ max at $\frac{-9}{-18} = \frac{1}{2} = t^*$

Next point: $(0, 3/2)$ $\quad \nabla f(0, 3/2) = (3/2, 0)$ $\Rightarrow$ Iteration 2 $f((0,3/2) + t(3/2, 0)) = f(t\,3/2, 3/2)$

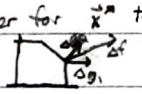
$= \left(\frac{3}{2}\right)^2 t + \frac{9}{2} - \left(\frac{3}{2}\right)^2 t^2 - \left(\frac{3}{2}\right)^2$ $\quad t^* = \frac{-\left(\frac{3}{2}\right)^2}{-2\left(\frac{3}{2}\right)^2} = 1/2$ $\Rightarrow$ Next point is $(3/4, 3/2)$

$\nabla f(3/4, 3/2) = 3/4\,\hat{j}$ $\Rightarrow$ keep iterating. TOR tutorial gives optimal solution at $(0.996, 1.998)$, $\nabla f(\cdot) =$ [illegible]

Linear



in order for $\vec{x}^*$ to be optimal, the level curves of f must be steeper than g_2, but not as steep as g_1.



∇f must be between ∇g_1 and ∇g_2.

thus $\nabla f(\vec{x}^*) = u_1 \nabla g_1(\vec{x}^*) + u_2 g_2(\vec{x}^*)$ where $u_1, u_2 \geq 0$ $\quad \nabla f$ is in the cone of ∇g_1 and ∇g_2

→ General NLP Maximize $f(\vec{x})$ s.t. $g_i(\vec{x}) \leq b_i$ for $1 \leq i \leq m$, $i \in N$, $\vec{x} \geq \vec{0}$

→ For $\vec{x}^*$ to be optimal we must have:

① $\nabla f(\vec{x}^*)$ is a linear combination of the gradients of the binding constraints.

② The multipliers in the linear combo are non-negative

③ $\vec{x}^*$ satisfies the constraints in the problem.

★★ → KKT conditions: If f and all the g_i's are differentiable, f is concave, the g_i's are convex, and the vectors in $\{\nabla g_i(\vec{x}^*) \mid g_i(\vec{x}^*) = b_i\}$ are linearly independent then: the following are necessary and sufficient conditions for the $\vec{x}^*$ to be optimal for the general NLP problem:

3/25

① • $\nabla f(\vec{x}^*) = \sum_{i=1}^{m} u_i \nabla g_i(\vec{x}^*) - \vec{v}$ ← accounts for non-negativity constraints

② • $g_i(\vec{x}^*) - \vec{b}_i \leq \vec{0}$ for each i

③ • $u_i(g_i(\vec{x}^*) - b_i) = 0$ for each i

④ • $-\vec{x}^* \leq 0$

⑤ • $v_j x_j = 0$ for each j

⑥ • $\vec{u}, \vec{v} \geq \vec{0}$

General NLP problem

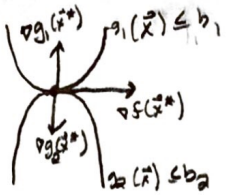
Max $S(\vec{x})$

s.t. $g_i(\vec{x}) \leq b_i \ \forall i \quad i \in \{1, \ldots, m\}$

$\vec{x} \geq \vec{0}$

3, 5 if constraints not binding, then coefficient is zero.

odd example.



Why are they called KKT conditions?

Kuhn-Tucker conditions (early 1950's) come up with it,

Karush - (1939 master's thesis at Chicago)

Ex with KKT: Max $f(x_1,x_2) = x_1 + 2x_2 - x_2^2$ S.t. $x_1 + x_2 \le 1$, $x_1, x_2 \ge 0$

$\nabla f(x_1,x_2) = \hat{\imath} + (2 - 3x_2^2)\hat{\jmath}$ $\quad \nabla g(x_1,x_2) = \hat{\imath} + \hat{\jmath}$

$\Rightarrow$ ① $1 = u_1(1) - v_1$ ② $x_1 + x_2 \le 1$ ③ $u_1(x_1 + x_2 - 1) = 0$ | ⑤ ⑥ $v_1 x_1 = 0$, $v_2 x_2 = 0$, All ≥ 0

② $(2 - 3x_2^2) = u_2(1) - v_2$

Solve: $u_1 \ge 1$ because of ①

$\Rightarrow$ $x_1 + x_2 = 1$ because of ③

Suppose $x_2 = 0$

$\Rightarrow$ $2 = u_1 - v_2$ ②

$\Rightarrow$ $x_1 = 1$ (from above)

$\Rightarrow$ $v_1 = 0$ (from ⑤)

$\Rightarrow$ $u_1 = 1$ (from ①) $\Rightarrow$ $2 = -v_2$ from ② contradiction.

~~Suppose $x_1 = 0$~~ Thus $x_2 > 0$, $\Rightarrow$ $\boxed{v_2 = 0}$

Suppose $x_1 = 0$

$\Rightarrow$ $x_2 = 1$ $\Rightarrow$ $u_1 = -1$ $\Rightarrow$ ④

Thus $x_1 > 0$, $\Rightarrow$ $\boxed{v_1 = 0}$

solve $u_1 = 1$,

$2 - 3x_2^2 = 1$ $\Rightarrow$ $\sqrt{1/3} = x_2 = \frac{\sqrt{3}}{3}$

$\Rightarrow$ $x_1 = \frac{3 - \sqrt{3}}{3}$

$f\left(\frac{3-\sqrt{3}}{3}, \frac{\sqrt{3}}{3}\right) = \frac{3-\sqrt{3}}{3} + 2\sqrt{3}/3 - \frac{1}{3\sqrt{3}} = \frac{9 + 2\sqrt{3}}{9} \approx 1.38$

12.68 Max $f(x_1, x_2) = 20x_1 + 10x_2$

$x_1^2 + x_2^2 \leq 1$

$2x_1 + 2x_2 \leq 1$

$x_1 \geq 0, \quad x_2 \geq 0$

$\nabla f(x,y) = \langle 20, 10 \rangle$

$\nabla f(x,y) = \lambda \nabla g(x,y)$ $\quad \nabla f(x,y) = \mu \nabla h(x,y)$

$20 = \lambda 2x \quad 10 = \lambda 2y$ $\quad \langle 20,10 \rangle = \mu(2,2)$

$\lambda x = 10 \quad \lambda y = 5$

$x^2 + y^2 = 1$

$x^2 = \left(\frac{10}{\lambda}\right)^2 \quad y^2 = \left(\frac{5}{\lambda}\right)^2$

$x^2 + y^2 = 1$

$\Rightarrow 10^2 + 5^2 = \lambda^2$

$\lambda^2 = 125$

$\lambda = \pm 5\sqrt{5}$

$\Rightarrow x = \pm \frac{10}{5\sqrt{5}}, \quad y = \pm \frac{1}{\sqrt{5}}$

$\left(\frac{2}{\sqrt{5}}\right)^2 + \left(\frac{1}{\sqrt{5}}\right)^2$

$= \frac{4}{5} + \frac{1}{5} = 1 \checkmark$

$\left(\pm \frac{2}{\sqrt{5}}, \pm \frac{1}{\sqrt{5}}\right)$

$x_1 = 2/\sqrt{5}$

$x_2 = 1/\sqrt{5}$

$u_1 = -5\sqrt{5}$

$u_2 = 0$

$v_1 = 0$

$v_2 = 0$

$\lambda \mu y = \left(\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}}\right)$

$\frac{4}{\sqrt{5}} + \frac{2}{\sqrt{5}} \leq 1$

$\frac{6}{\sqrt{5}} \leq 2$

3/31/08

$h_i(\vec{x}) = d_i \qquad i = 1, 2, \ldots, \ell$

Max $f(\vec{x})$

s.t. $g_i(\vec{x}) \le b_i \qquad i = 1, 2, \ldots, m$

$\vec{x} \ge \vec{0}$

Now do we deal with the equality constraints

$h_i(\vec{x}) = d_i$

$\Rightarrow \quad h_i(\vec{x}) \le d_i$

& $h_i(\vec{x}) \ge d_i$

$\Rightarrow \quad -h_i(\vec{x}) \le -d_i$

KKT conditions

$$\nabla f(\vec{x}) = \sum_{i=1}^{m} u_i \nabla g(\vec{x}) - \vec{v} + \sum_{i=1}^{\ell} w_i \nabla h_i(\vec{x})$$

$g_i(\vec{x}) \le b_i$

$u_i (g_i(\vec{x}) - b_i) = 0$ $\qquad$ $h_i(\vec{x}) = d_i \quad i = 1, \ldots, \ell$

$x_j \; v_j = 0$

$\vec{x}, \vec{u}, \vec{v} \ge \vec{0}$

↑ not non-negative.

↓ no need for complementarity conditions.

Quadratic Programming

Maximize $f(\vec{x}) = \vec{c}\vec{x} + \frac{1}{2}\vec{x}^T Q \vec{x}$

s.t. $A\vec{x} \leq \vec{b}$

$\vec{x} \geq \vec{0}$

in what sense is this quadratic?

gradient is nice.

Max $x_1 + x_2 - \frac{1}{2}x_1^2 - x_2^2 + x_1 x_2$

s.t. $x_1 + x_2 \leq 3$

$-2x_1 - 3x_2 \leq -6$

$x_1, x_2 \geq 0$

$A = \begin{bmatrix} 1 & 1 \\ -2 & -3 \end{bmatrix}$

what is Q

$$Q = \begin{bmatrix} -1 & 1 \\ 1 & -2 \end{bmatrix}$$

$$\frac{1}{2}\begin{bmatrix} x_1 & x_2 \end{bmatrix}\begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} =$$

$$= \frac{1}{2}q_{11}x_1^2 + \frac{1}{2}q_{12}x_1x_2 + \frac{1}{2}q_{21}x_2x_1 + \frac{1}{2}q_{22}x_2^2$$

KKT conditions

$\nabla f(x_1, x_2) = \langle 1 - x_1 + x_2,\ 1 + x_1 - 2x_2 \rangle$

$\nabla g_1(x_1, x_2) = \langle 1, 1 \rangle$

$\nabla g_2(x_1, x_2) = \langle -2, -3 \rangle$

$1 - x_1 + x_2 = u_1 - 2u_2 - v_1$

$1 + x_1 - 2x_2 = u_1 - 3u_2 - v_2$

$x_1 + x_2 \leq 3$

$-2x_1 - 3x_2 \leq 0$

$u_1(x_1 + x_2 - x_3) = 0$

$u_2(-2x_1 - 3x_2 + 6) = 0$

$v_1 x_1 = 0$

$v_2 x_2 = 0$

$x_1, x_2, u_1, u_2, v_1, v_2 \geq 0$

Can use simplex method to find a feasible solution to KKT conditions which is then guaranteed to be optimal for the quadratic problem.

Quadratic Programming

Maximize $f(\vec{x}) = \vec{c}\vec{x} + \frac{1}{2}\vec{x}^T Q \vec{x}$

s.t. $A\vec{x} \leq \vec{b}$

$\vec{x} \geq \vec{0}$

in what sense is this quadratic?

gradient is nice.

Max $x_1 + x_2 - \frac{1}{2}x_1^2 - x_2^2 + x_1 x_2$

s.t. $x_1 + x_2 \leq 3$

$-2x_1 - 3x_2 \leq -6$

$x_1, x_2 \geq 0$

$A = \begin{bmatrix} 1 & 1 \\ -2 & -3 \end{bmatrix}$

what is Q

$$Q = \begin{bmatrix} -1 & 1 \\ 1 & -2 \end{bmatrix}$$

$$\frac{1}{2}\begin{bmatrix} x_1 & x_2 \end{bmatrix}\begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} =$$

$$= \frac{1}{2}q_{11}x_1^2 + \frac{1}{2}q_{12}x_1x_2 + \frac{1}{2}q_{21}x_2x_1 + \frac{1}{2}q_{22}x_2^2$$

KKT conditions

$\nabla f(x_1, x_2) = \langle 1 - x_1 + x_2,\ 1 + x_1 - 2x_2 \rangle$

$\nabla g_1(x_1, x_2) = \langle 1, 1 \rangle$

$\nabla g_2(x_1, x_2) = \langle -2, -3 \rangle$

$1 - x_1 + x_2 = u_1 - 2u_2 - v_1$

$1 + x_1 - 2x_2 = u_1 - 3u_2 - v_2$

$x_1 + x_2 \leq 3$

$-2x_1 - 3x_2 \leq 0$

$u_1(x_1 + x_2 - x_3) = 0$

$u_2(-2x_1 - 3x_2 + 6) = 0$

$v_1 x_1 = 0$

$v_2 x_2 = 0$

$x_1, x_2, u_1, u_2, v_1, v_2 \geq 0$

Can use simplex method to find a feasible solution to KKT conditions which is then guaranteed to be optimal for the quadratic problem.

KKT conditions for general quadratic program

$$\vec{c}^T + Q\vec{x} = A^T\vec{u} - \vec{v}$$

$$A\vec{x} \leq b$$

$$\vec{u}^T(\vec{b} - A\vec{x}) = 0 \quad \text{since } \vec{b} - A\vec{x} \geq \vec{0}$$

$$\vec{v}^T\vec{x} = 0$$

$$\vec{x}, \vec{u}, \vec{v} \geq \vec{0}$$

Wolfe's Method

- only guaranteed to work when either $\vec{c} = 0$ or $f(\vec{x})$ is strictly concave

$\Rightarrow f(\lambda\vec{x}_1 + (1-\lambda)\vec{x}_2) > \lambda f(\vec{x}_1) + (1-\lambda) f(\vec{x}_2)$

Thm: f is strictly concave iff its Hessian Q is negative definite

Def. ND: $\vec{x}^T Q \vec{x} < 0 \quad \forall \vec{x} \neq \vec{0}$

ND $\Leftrightarrow$ All eigenvalues strictly negative.

4/1/2008 Yesterday "Quadratic Programming

KKT conditions in general

$\vec{c}^T + Q\vec{x} = A^T\vec{u} - \vec{v}$

$A\vec{x} \le \vec{b}$

$\vec{u}^T(\vec{b} - A\vec{x}) = 0$

$\vec{v}^T\vec{x} = 0$

$\vec{x}, \vec{u}, \vec{v} \ge \vec{0}$

Need to introduce slack variables.

Let $\vec{y} = \vec{b} - A\vec{x}$

$-Q\vec{x} + A^T\vec{u} - \vec{v} = \vec{c}^T$

$A\vec{x} + \vec{y} = \vec{b}$

$\vec{u}^T\vec{y} = 0$

$\vec{x}, \vec{u}, \vec{v}, \vec{y} \ge 0$

Let's verify that the conditions implying Wolfe's algorithm gives an optimal solution hold.

- f is strictly concave or $\vec{c} = \vec{0}$

$f(x_1, x_2) = x_1 + x_2 - \frac{1}{2}x_1^2 - x_2^2 + x_1x_2$

$Q = \begin{bmatrix} -1 & 1 \\ 1 & -2 \end{bmatrix}$ want $\vec{x}^TQ\vec{x} < 0 \quad \forall \vec{x} \neq \vec{0}$

$\vec{x}^TQ\vec{x} = -x_1^2 - 2x_1x_2 - 2x_2^2$

$= -(x_1 - x_2)^2 - x_2^2$

≤ 0

thus $\vec{x}^TQ\vec{x} < 0 \quad \forall \vec{x} \neq 0$

$\vec{x}^TQ\vec{x} = 0$

$\Rightarrow -(x_1 - x_2)^2 - x_2^2 = 0$

$\Rightarrow x_2^2 = -(x_1 - x_2)^2$

$\Rightarrow x_2 = 0$ & $x_1 - x_2 = 0$

$\Rightarrow x_1 = 0$

$x_1 - x_2 + u_1 - 2u_2 - v_1 = 1$
$-x_1 + 2x_2 + u_1 - 3u_2 - v_2 = 1$
$x_1 + x_2 + y_1 = 3$
$-2x_1 - 3x_2 + y_2 = -6 \Leftarrow$ negative RHS
$v_1x_1 = v_2x_2 = u_1y_1 = u_2y_2 = 0$
$x_1, x_2, y_1, y_2, u_1, u_2, v_1, v_2 \ge 0$

} no slacks

(introduce artificial variables)

(multiply by -1, but $y_2 = -6$, so then we'll add another artificial variable)

Introduce z_1, z_2, z_3

Objective: Min $z = z_1 + z_2 + z_3$

$\Leftrightarrow$ Max $-z = -z_1 + -z_2 + -z_3$

$\Leftrightarrow$ Max $-z + z_1 + z_2 + z_3 = 0$

Separable Programming

Def: A function $f(\vec{x})$ is separable if $f(\vec{x})$ can be expressed as $f(\vec{x}) = \sum_{j=1}^{n} f_j(x_j)$

(there are no "cross terms")

Ex. $f(\vec{x}) = x_1^2 - x_1 + e^{x_2} + \sin(x_3^2)$

is separable.

Separable Programming Problem

Maximize $f(\vec{x})$

$g_i(\vec{x}) \leq b_i$

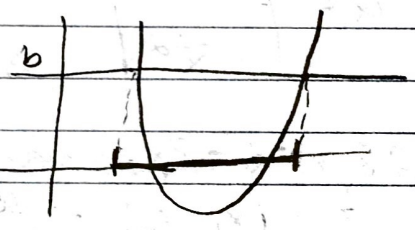
$\vec{x} \geq \vec{0}$

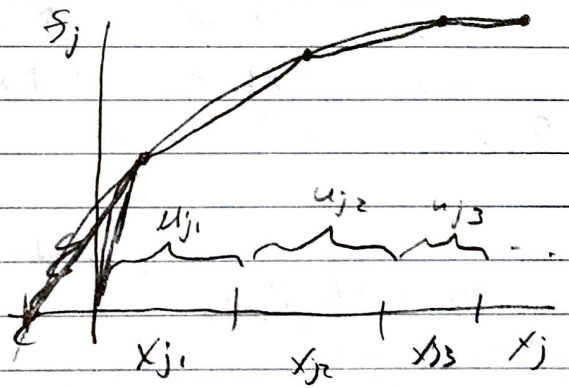
where f is concave & separable

& g_i's are convex & separable

Note: $g(x) \leq 0$

where g_i's convex





s_{j1} is the slope of the 1st segment.

$f_j(x_j) \approx s_{j1} x_{j1} + s_{j2} x_{j2} + \dots + s_{j5} x_{j5}$

where $x_{ji} \leq u_{ji} \quad \forall i$

$x_j = x_{j1} + x_{j2} + x_{j3} + \dots + x_{j5}$

violated LP $\{ x_{j,i+1} = 0$ if $x_{ji} < u_{ji}$, $i = 0, 1, 2, 3, 4.$

Max $30x_1 + 35x_2 - 2x_1^2 - 3x_2^2$

s.t. $x_1^2 + 2x_2^2 \leq 250$ ←

$x_1 + x_2 \leq 20$ ←

$x_1, x_2 \geq 0$

$f_1(x_1) = 30x_1 - 2x_1^2$ $\quad g_{11}(x_1) = x_1^2$ $\quad g_{12}(x_2) = 2x_2^2$

$f_2(x_2) = 35x_2 - 3x_2^2$ $\quad g_{21}(x_1) = x_1$ $\quad g_{22}(x_2) = x_2$

$0 \leq x_1 \leq 20$ $\quad 0 \leq x_2 \leq 20$

Break points at 0, 5, 10, 15, 20

$f_1(0) = 0$ $\quad f_{11} = \frac{100-0}{5-0} = 20$

$f_2(5) = 100$

$f_3(10) = 180$

$f_4(15) = 0$

$f_5(20) = 200$

$f_1(x_1) = 20x_{11} + 0x_{21} - 20x_{31} + 40x_{41}$

Maximize $20x_{11} + 0x_{21} - 20x_{31} + 20x_{12} - 10x_{22} - 40x_{32} - 70x_{42}$

s.t. $5x_{11} + 15x_{21} + 25x_{31} + 35x_{41} + 10x_{12} + 30x_{22} + 50x_{32} + 70x_{42} \leq 250$

$x_{11} + x_{21} + x_{31} + x_{41} + x_{12} + x_{22} + x_{32} + x_{42} \leq 20$

$0 \leq x_{ij} \leq 5 \quad \forall i, j$

Approx optimal solution $z = 200$

with $x_1 = x_{11} = 5$, $\quad x_2 = x_{12} = 5$

Actual solution $z = 214.58$

$x_1 = 7.5$, $\quad x_2 = 5.83$

Combinatorial optimization

- problems can be modeled using graphs / networks
- sometimes problems can be expressed using sets or matrices

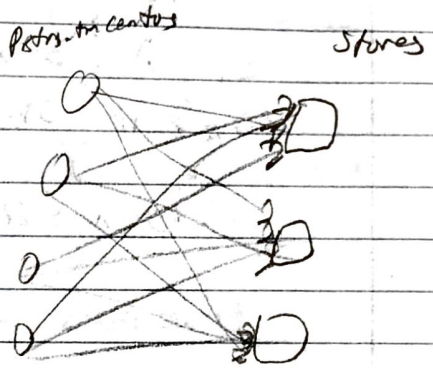
Examples:

① Network problems

Ⓐ transportation problems

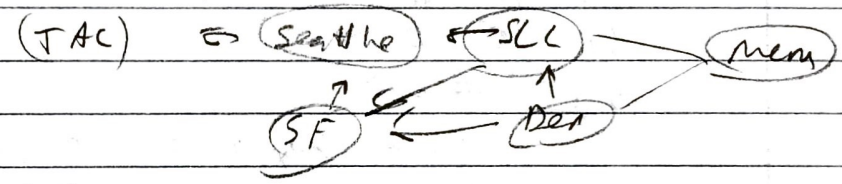
Ⓑ Assignment problems

Kidneys to donors.



$\det(AB) = \det(A)\det(B)$

ⓒ other network problems,



② other graph problems

- Minimum spanning tree
- Maximum Flow
- Traveling salesman,
- Project management

③ [illegible]

other OC problems

① set covering

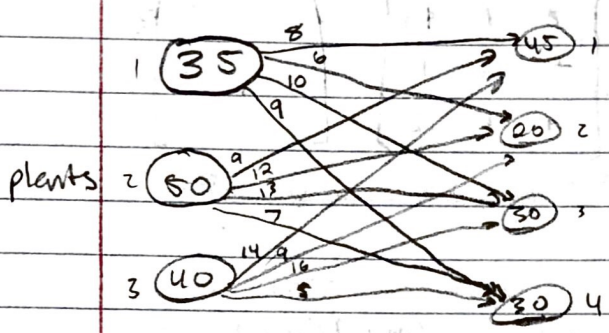
② set packing

③ knapsack problem

Transportation Problem → LP

In this problem total supply = total demand

→ balanced transportation problem



4 → 0 dummy to balance things

Transportation Problem

Assumptions

① total supply equals total demand

② the cost of distributing commodity from source i to destination j is proportional to the amount of x_{ij}

	1	2	3
1	2	1	1
2	1	2	1
3	1	1	2

$1 \bmod 3 = 1$

$2 \bmod 3 = 2$

$1 - 2 = -1 \pmod 3$

	1	2	3	4	5
1	2	1	1		
2	1	2		1	
3	1		2		2
4		1		2	1
5			1		2

Mathematical formulation of the general transportation problem,

$$\text{Maximize} \quad \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{ij} = d_j \quad \forall j \quad \text{(demand constraints)}$$

$$\sum_{j=1}^{n} x_{ij} = s_i \quad \forall i \quad \text{(supply constraints)}$$

What if we have excess supply?

introduce a dummy city with zero cost associated with assigning things to it.

What if we have excess demand,

introduce a dummy source.

what if an assignment is infeasible?

— put a prohibitively large cost on that assignment

$$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \to \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

Constraints?

$m+n$

number of rows $m+n+1$

number of columns $mn + m+n+1 = (m+1)(n+1)$

BV	z	x_{ij}	z_i	z_{m+j}	RHS
z	-1	c_{ij}	M	M	0
z_i	0	1	1		s_i
z_{m+j}	0	1		1	d_j

LU decomposition.

$$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

$$c_b \cdot AB^{-1} - \vec{C}$$

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + 1$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

BV z

$c_{ij} - u_i - v_j$ $\quad u - u_i$ $\quad u - v_j$

$-1 \bmod 3 = 2$

$-2 \bmod 3 = 1$

$2 - 1 = 1$

$$-\sum_{i=1}^{m} s_i u_i - \sum_{j=1}^{n} d_j v_j$$

what does u_i mean?

Shadow prices/dual variable for constraint associated with z_i

	1	2	3
1	2	1	1
2	1	2	1
3	1	1	2

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

	1	2	3	4	
1	2	0	1	0	2
2	0	2	0	1	# 2 or $-2 \pmod 4$
3	1	0	2	0	
4	0	1	0	2	

Source \ Destination	1	2	3	4	Supply	u_i
1	8	6	10	9	35	
2	9	12	13	7	50	
3	14	9	16	5	40	
Demand	45	20	30	30		
v_j						

If x_{ij} is basic, cell ij contains the value of x_{ij}

if x_{ij} is non-basic, cell ij contains $c_{ij} - u_i - v_j$

4/15/2008 Yesterday: Assignment Problem.

Hungarian Method:

- if you add a constant to every entry in a row or column of the cost table, then the optimal solution is unchanged.

Proof (for rows)

Suppose we add cost K to every entry in row ℓ.

new objective function

$$\sum_{i=1,\, i\neq\ell}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{j=1}^{n} (c_{\ell j} + K) x_{\ell j}$$

$$= \sum_{i=1,\, i\neq\ell}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{j=1}^{n} c_{\ell j} x_{\ell j} + K \sum_{j=1}^{n} x_{\ell j}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + K \quad \text{since } \sum_{j=1}^{n} x_{\ell j} = 1$$

$\| 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0 \| = 1$

Since constraints haven't changed, adding a constant to the objective function does not change the optimal solution.

	C	C	D	T	K
BS	37.7	32.9	33.8	37	35.4
BS	43.4	33.1	42.2	34.7	41.8
B	53.4	28.5	38.9	30.4	33.6
F	29.2	26.4	29.6	28.8	31.1
D	0	0	0	0	0

Round

Subtract smallest number from each row.

	C	C	O	T	K
B	4.8	0	0.9	4.1	2.5
B	10.3	0	9.1	1.6	8.7
F	4.8	0	10.4	1.9	5.1
F	2.8	0	3.2	2.1	4.7
Dummy	0	0	0	0	0

~~Dead~~

Then Subtract smallest # in each column.
(but does it get us any where)

Next

strike out 0 column & rows. (& any row / column with a zero in it)
pick out smallest number left.
Subtract 0.9 from every entry in the table.

Add .9 to each entire row & column that struck out.

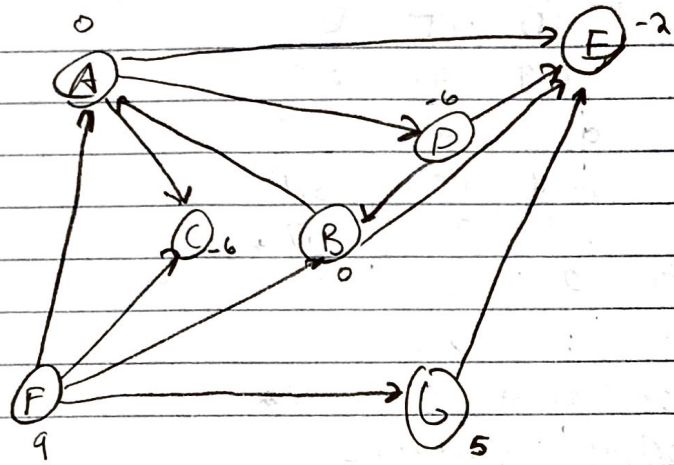
3.9	0	0	3.2	1.6
9.4	0	8.2	0.7	7.8
3.9	0	9.5	1.0	4.2
1.4	0	2.3	1.2	3.8
0	0.9	0	0	0

Now Iterate again.

3.9	.7	0	3.2	1.6
8.7	0	7.5	0	7.1
3.2	0	8.8	.3	3.5
1.2	0	1.6	.5	3.1
0	1.6	0	0	0

Iterate again -1.2

2.7	0.7	0	3.2	0.4
7.5	0	7.5	0	5.9
2.0	0	9.8	6.3	2.3
0	0	1.6	0.5	1.9
0	2.8	1.2	1.2	0



A network consists of a set N of nodes and a set A of arcs.

We assume that total supply = total demand

If b_i is the supply for node i
we have $\sum_{i \in N} b_i = 0$

$A \subseteq \{(i,j) : i,j \in N, i \neq j\}$

→ arcs are directed.
→ no arcs from a node to itself.

How do I send flow from supplies to demands at least cost?

Let x_{ij} be the flow on arc (i,j)

constraints

① $x_{ij} \geq 0 \quad \forall\, i,j \in A$

② flow-balance or flow conservation at each node,
flow in + supply = flow out.

Robert Hildebrand

(2) rewritten.

$$f_{in} - f_{out} = -supply.$$

$$\sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = -b_v$$

$$\forall v \in N$$

Let c_{ij} be the per-unit cost of assigning flow on arc (i,j)

Minimize

$$\sum_{(i,j)\in A} c_{ij} x_{ij}$$

Uncapacitated minimal cost-flow problem.

A tree is a connected graph with no cycles.

Thm: A network with n nodes is a tree if it has $n-1$ arcs and no loops.

Thm: Spanning trees $\Leftrightarrow$ bases in a network.

pf: (mostly)

A basis in a network has $n-1$ basic variables.

Suppose we have a loop consisting of variables x_{12}, x_{23}, x_{31} all of which are in the basis.

$\Rightarrow$ ↓

Original Simplex tableau

	x_{12}	x_{23}	x_{31}	. . .
node 1	-1	0	1	
node 2	1	-1	0	
node 3	0	1	-1	
	0	0	6	
	0			
	0			
	0			

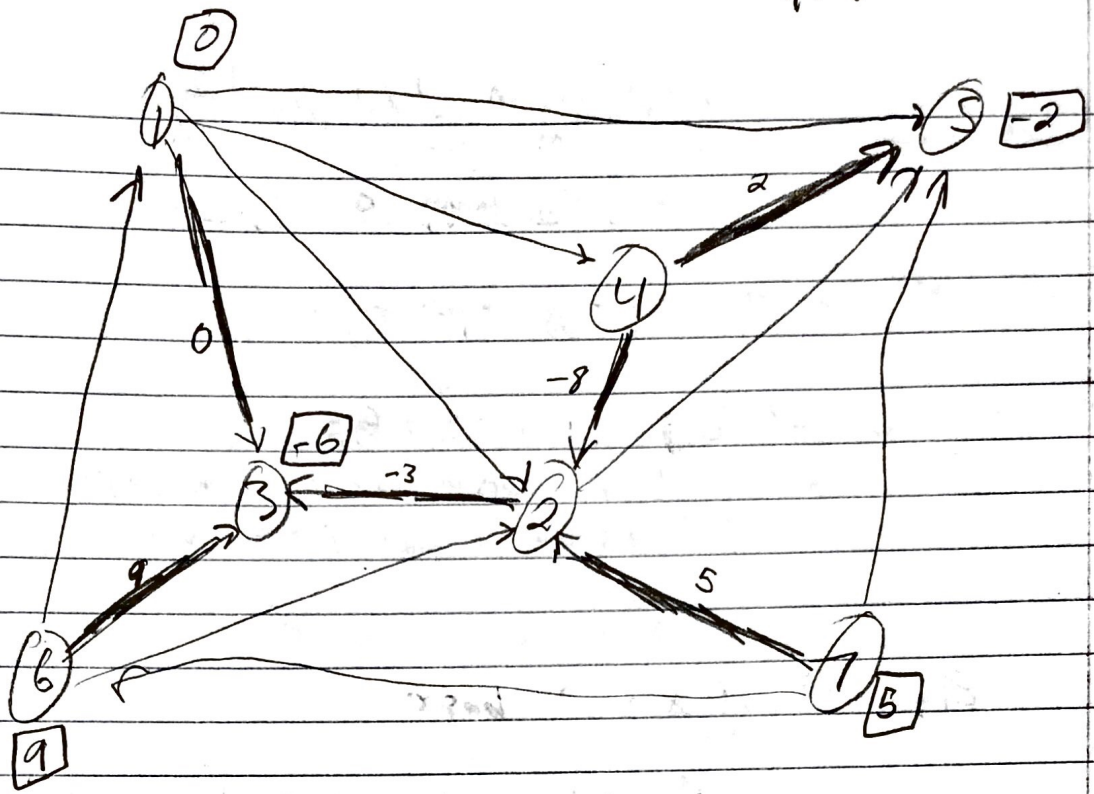
Colums are linearly dependent (have Rank 2)

=) Can't have x_{12}, x_{23}, x_{31} all in the bases at the same time.

=) The simplex method would never choose x_1, x_{23}, x_{31} to be in the basis simultaneously.

=) Thus Bases in a network consist of $n-1$ arcs and no loops, and are thus spanning trees.

4/17/08



Basic feasible Solution

- Naive Approach
- use Max flow Algorithm.
- create a super source connected to all sources
- create a Super sink with connections from all sinks.

How do we test for optimality?
- requires discussion of Duality.

Primal Variables	Dual variables
x_{ij}'s (arcs, original)	z_{ij}'s (arcs, slacks) – non-negative
t_i's (nodes, slack)	y_i's (nodes, original) no-restrictions

constraints in dual problem

$-y_i + y_j \leq c_{ij}$

$-y_i + y_j + z_{ij} = c_{ij}$

$\Rightarrow z_{ij} = c_{ij} + y_i - y_j$

↖ numbers in row z in the simplex tableau for primal vars x_{ij}

$z_{ij} = 0$ if x_{ij} is basic

$c_{ij} + y_i = y_j$ if x_{ij} is basic

(one more variable y_i than constraints)

Integrality

If the original data are all integers, then so will the optimal solution.

What if there are capacity constraints on the arcs?

$x_{ij} \le u_{ij}$ → don't affect flow balance

Slack variable $\Rightarrow$ $x_{ij} + t_{ij} = u_{ij}$

which constraints contain x_{ij} & t_{ij}

	x_{ij}	t_{ij}	RHS
Node i	-1		$-b_i$
Node j	1		$-b_j$
x_{ij} upper bound	1	1	u_{ij}

	x_{ij}	t_{ij}	RHS
Node i	-1	0	$-b_i$
Node j	0	-1	$-b_j - u_{ij}$
(capacity node)	1	1	u_{ij}

Yesterday Networks

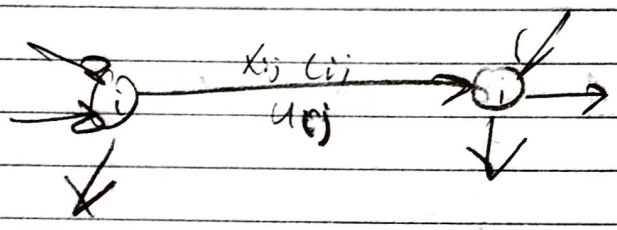
Spanning trees ⇔ bases in a network

~~Spanning Tree ⇔~~

Proof:

$x_{ij} + t_{ij} = u_{ij}$

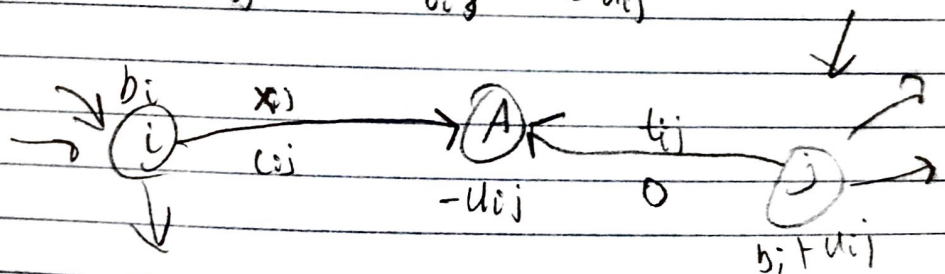
upper bound constraint on arc



	x_{ij}	t_{ij}	RHS
node i	$-x_{ij}$		$= -b_i$
node j	$+x_{ij}$		$= -b_j$
upper bound	x_{ij}	t_{ij}	$= u_{ij}$

⇓

	x_{ij}	t_{ij}	RHS
node i	$-x_{ij}$		$= -b_i$
node j		$-t_{ij}$	$= b_j - u_{ij}$
upper bound	x_{ij}	t_{ij}	$= u_{ij}$



Yesterday Networks

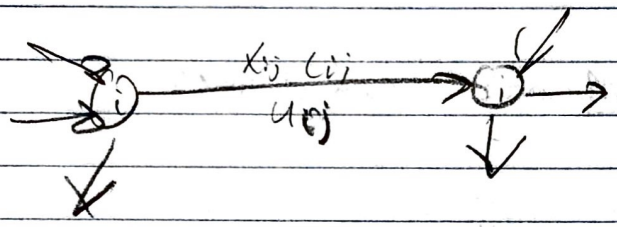
Spanning trees <-> bases in a network

~~Spanning Tree Th~~

Proof:

$x_{ij} + t_{ij} = u_{ij}$

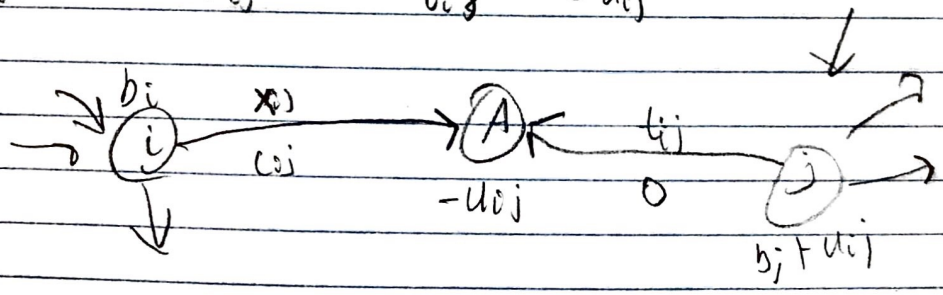
upper bound constraint on arc



	x_{ij}	t_{ij}	RHS
node i	$-x_{ij}$		$= -b_i$
node j	$+x_{ij}$		$= -b_j$
upper bound	x_{ij}	t_{ij}	$= u_{ij}$

$\Downarrow$

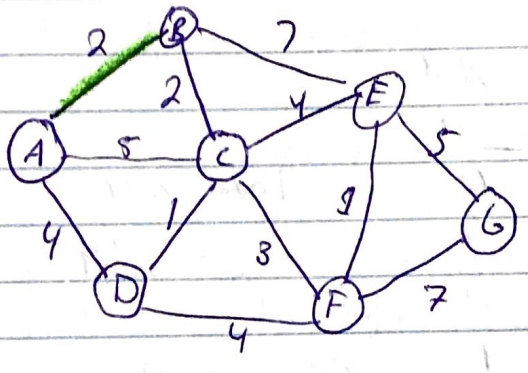
	x_{ij}	t_{ij}	RHS
node i	$-x_{ij}$		$= -b_i$
node j		$-t_{ij}$	$= b_j - u_{ij}$
upper bound	x_{ij}	t_{ij}	$= u_{ij}$



Minimum Spanning tree
Shortest Path
Maximum flow. Through a network.

Yesterday: Network Simplex with upper bounds

A company wants to install a fiber optic cable between its businesses.
It wants to be connected s.t. cost is at a minimum.



Minimum Spanning Tree

↙ visit all nodes, no cycles

Model as LP

directed arc.

Let $x_{ij} = \begin{cases} 1 & \text{if arc } ij \text{ is in the tree} \\ 0 & \text{if not} \end{cases}$

Objective function

Minimize $\sum_{\{i,j\} \in A} c_{ij} x_{ij}$

Constraints

$\sum_{\{i,j\}} x_{ij} = |N| - 1$

$\sum_{\{i,j\} \in A(S)} x_{ij} \leq |S| - 1$ for each $S \subseteq N$

$2^{|N|} - 2$ constraints.

Prim's Algorithm

~~start at a node~~

~~Calculate~~ $V = \{u\}$ with $u \in N$.

① · Let ~~V consist of any node u~~ in the ~~node~~ set N

② · Calculate the shortest distance from nodes in V to all nodes immediately reachable from V.

③ · Let v be a node with the shortest distance from nodes in V.

④ · Let $V = V \cup \{v\}$ and go to ②

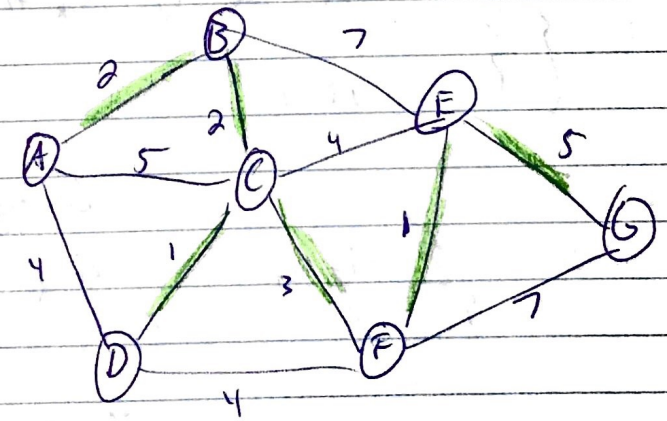
⑤ Stop if there is no such V in ③

---

ex.

V	All cost
A	0
B	2
C (from B)	2
D (from C)	1
F (from C)	3
E (from F)	1
G (From E)	5

Total cost = 14



Proof of Prim's Method (for connected networks)

Let T be the output from Prim's Algorithm (V, S)

Part 1: T is a spanning tree
Part 2: T is minimal in the set of spanning trees.

At each stage of the algorithm a node not in V is added to V. Thus at no point in Prim's algorithm does (V, S) contain a cycle.

Since the network is connected, Prim's Algorithm will add a node to V at each stage of the network until there are no nodes in the network not in V.

Thus T contains all nodes in the network.

We add an arc (and a node) at each iteration of the algorithm except the first, which only adds a node.

T contains $|N|$ nodes and $|N|-1$ arcs.
Thus T is a Spanning Tree.

∎ (Part 1)

Part 2 (Induction)

First, we've shown that a spanning tree exists. Since there at most a finite number of spanning trees, at least one must be minimal.

We'll now prove by induction that T_k, the tree contained in iteration k of Prim's Algorithm, is contained in some minimum spanning tree.

Base case: T_1 is contained in a minimum spanning tree

T_1 is a single node, and a MST exists and must contain all nodes. ✓

Induction Step

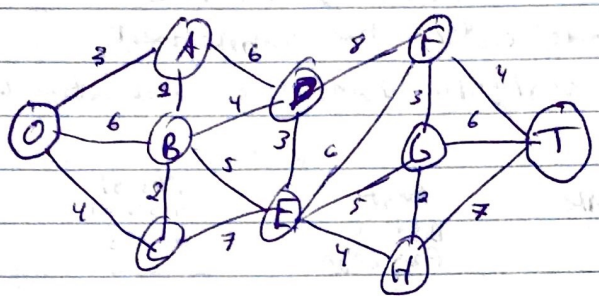
Assume T_k is contained in some minimum spanning tree T^*.

Let $\{u, v\}$ be the arc added in iteration $k+1$, with $u \in V$ and $v \notin V$.

If $\{u, v\} \in T^*$ then T_{k+1} is contained in a minimum spanning tree.

Today: Shortest Path problem

A fire department serves a rural area with several towns. Need to know shortest route from the station to each town.



Assumptions

① undirected arcs.

② non-negative costs on arcs

Let $x_{ij} = \begin{cases} 1 & \text{if arc } \{i,j\} \text{ is in the path} \\ 0 & \text{otherwise} \end{cases}$

Minimize $\sum_{\{i,j\} \in A} c_{ij} x_{ij}$

Need connected start at origin & end at destination.

s.t. $-\sum_{\{l,j\} \in A} x_{lj} + \sum_{\{i,l\}} x_{il} = 0 \quad \forall l \in N - \{O,T\}$

$\sum_{\{O,j\} \in A} x_{Oj} = 1$

$\sum_{\{i,T\}} x_{iT} = 1$

$x_{ij} \in \{0,1\} \quad \forall \{i,j\} \in A$

Dijkstra's Algorithm – for shortest Paths

- Let your current sub-network consist of the origin.
- Calculate the shortest path from the origin to any node not currently in the subnetwork
- Let this node enter the subnetwork

Repeat until destination is in the subnetwork.

Iteration N	Solved nodes	Reachable unsolved nodes	Best distance from origin	Nearest unsolved node	New are
1	O	A B C	3 6 4	A	{O, A}
2	O, A	C D E	4 4 9	B	{A, B}
3	O, A, B	C D E	4 8 9	C	{O, C}
4	O, A, B, C	D E	8 9	D	{B, D}
5	O, A, B, C, D	E F	9 16	E	{B, E}
6	O A B C D E	F G H	15 14 15	H	{E, H}
7	O A B C D E H	F G T	15 14 20	G	{E, G}
8	O A B C D E G H	F T	15 20	F	{E, F}
9	All {T}	T	19	T	{F, T}

Theorem When a source node O, a destination node T, and a network (N, A) with a path from O to T is input to Dijkstra's Algorithm, the output is a shortest path from O to T.
* Assuming there are no no negative cycles. or just no negative cycles.

Proof: (by induction)
The node V that enters the subnetwork in iteration k has had the shortest path from O to v_k calculated correctly.

Base case:
Node O enters the subnetwork initially.
Clearly the shortest path from O to itself has length 0 ✓

Let v_k be the node that enters the subnetwork in iteration k.
Let ~~$L(v_k)$ Suppose~~ be that the length of the path from O to v_k

Suppose there is another path P from O to v_k with a shorter length.

Let w be the node in P directly before v_k

Either w is in the k-1 subnetwork or not.
But w cannot be in the subnetwork because Dijkstra would have chosen path P in iteration k and it did not.

Since w is not in the subnetwork, there must be some node u that is the first node in P not in the subnetwork.
Since $L(v_k) > L(P), \geq L(w) \geq L(u)$
there is a path from O to u shorter than path from O to v_k and thus that would be the next node chosen. ■

Stable Marriage Problem

Assign med students to hospitals

↓ ranked list of hospitals

↓ ranked list of students

What is the best way to make this assignment?

How do you measure an optimal assignment.

① Most matches of ① ↔ ②

② Minimize sum of ranks or average

⇒ Might lead to an unstable pair (i, j) s.t. i & j both prefer each other over their actual matching

A stable matching has no unstable pairs.

② no unstable pairs

1962 Gale and Shapley

American Mathematical Monthly

~~Mon~~

Men	Preference
A	c b d a
B	b a c d
C	b d a c
D	c a d b

Women	P
a	A B D C
b	C A D B
c	C B D A
d	B A C D

a matching: Aa, Bb, Cc, Dd
C and b would rather be with eachother
=) unstable.

M	Current Proposal	Accept?	Current Partners
A, B, C, D	Ac	yes	Ac
B, C, D	Bb	yes	Bb, Ac
C, D	C, b	yes	Cb, Ac
B, D	Ba	yes	Ba, Ac, Cb
D	Dc	yes	Dc, Ba, Cb
A	Ab	no	Dc, Ba, Cb
A	Ad	yes	Dc, Ba, Cb, Ad

Donald Knuth

- Invented field of Algorithm studies
- Invented TeX.

Stable Marriage and its Relation to other Combinatorial Problems

Thm. The Stable-marriage algorithm works.
~~Prove~~ Produces a stable matching

Proof: Suppose Dan is matched with Claire but prefers Agnes to Claire.
Show Agnes does not prefer Dan to whoever she is matched with.

⇒ Agnes turned Dan down.
at some point.

Thus she must have rejected Dan in favor of someone she likes better.

⇒ Agnes must be with someone she likes better than Dan.

Generalizes, this shows it is impossible for the algorithm to produce an unstable match (i.e. we are done)!!!!

Corollary: Each pair of preference matrices has a stable matching.

Let (i,j) be stable partners if there is some stable matching in which i and j are assigned to each other.

Lemma: In the stable matching algorithm, a woman never rejects a stable partner.

Suppose the first time a woman rejects a stable partner in a particular running of the algorithm occurs when Clare rejects Don in favor of Jeff.

Clare must prefer Jeff.

Let S be the stable matching that pairs Clare & Don.

Let Kelly be the person Jeff is assigned to in S.

So Jeff & Kelly are stable partners.

Thus Jeff prefers Kelly to Clare.

Because otherwise we would be unstable.

Thus Kelly must have rejected Jeff at some point in the running of the algorithm in order for the Jeff/Clare engagement to be possible.

Thus this happened before the first time $\Rightarrow \Leftarrow$

Corollary: Each man is assigned to his best stable partner by the algorithm

Proof:

Men ask in order of preference
Since A woman never rejects a stable partner

...

...

Corollary: The order of selection of the men from set M does not affect the stable matching produced by the algorithm.

Integer Programming

→ LP with Integers:
some assumptions except divisibility are satisfied.

Examples: Assignment Students of schools
Assignment problem (*)
Transportation (some)
Minimum Spanning tree (**)
Shortest path. (*)

(*) binary decision variables

Some IP problems using Binary DV

(2) perfect matching.

You have $2n$ individuals that must be assigned to eachother in pairs.

∃ ① pairing roommates

Let:

Let $x_{ij} = \begin{cases} 1 & \text{if person } i \text{ is assigned to } j \\ 0 & \text{otherwise} \end{cases}$

for $i < j$

Let c_{ij} = cost of assigning person i to person j.

$i < j$

How many decision variables? $\frac{2n(n-1)}{2} = n(2n-1)$

Minimize $\sum_{i}^{2n-1} \sum_{i+1}^{2n} c_{ij} x_{ij}$

S.t. $\sum_{i=1}^{\ell-1} x_{i\ell} + \sum_{j=\ell+1}^{2n} x_{\ell j} = 2 \quad \forall \ell \in \{1, \ldots, 2n\}$

Tuesday Potcet Modeling power

Today.

We have a set of potential sites for fire stations $N = \{1, 2, \ldots, n\}$

We have a set of communities that need fire coverage
We have a cost c_j for building fire stations.

The set of communities that can be served by fire station j is M_j.

We want to find a subset of N that covers M at minimum cost.

⇒ Set covered problem.

Let $x_i = \begin{cases} 1 & \text{if a station is built at site } j \\ 0 & \text{if not} \end{cases}$

Minimize $\sum_{j=1}^{n} c_j x_j$

S.t. ≥ 1 } for each community.

Let A be the $m \times n$ incidence matrix
s.t.

$$a_{ij} = \begin{cases} 1, & \text{if comm } i \text{ is served by station } j \\ 0 & \text{otherwise} \end{cases}$$

Tuesday perfect Modeling primer

Today.

We have a set of potential sites for fire stations $N = \{1, 2, \ldots, n\}$

We have a set of communities that need fire coverage
We have a cost c_j for building fire stations.

The set of communities that can be served by fire station j is M_j.

We want to find a subset of N that covers M at minimum cost.

⇒ Set covered problem.

Let $x_j = \begin{cases} 1 & \text{if a station is built at site } j \\ 0 & \text{if not} \end{cases}$

Minimize $\sum_{j=1}^{n} c_j x_j$

S.t. ≥ 1 } for each community.

Let A be the $m \times n$ incidence matrix
s.t.

$$a_{ij} = \begin{cases} 1, & \text{if comm. } i \text{ is served by station } j \\ 0 & \text{otherwise} \end{cases}$$

Site 1 covers 2, 4

2 covers 1, 3, 4

3 covers 1, 2

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

$A\vec{x}$ $\Rightarrow$ constraints

$$A\vec{x} \geq \vec{1}$$

$$\min \sum_{j=1}^{n} c_j x_j$$

s.t. $A\vec{x} \geq \vec{1}$

$x_j \in \{0, 1\}$

$j \in \{1, 2, \ldots, n\}$

Event 1 can occur at most 2 times.

Event 1 can occur iff event 2 occurs

x_1 x_2

$x_1 = x_2 \Rightarrow x_1 - x_2 = 0$

Event 2 can occur only if Event 1 occurs.

$x_2 \leq x_1 \Rightarrow x_2 - x_1 \leq 0$

Let y represent a range of possible values associated with event 2.

$0 \leq y \leq u$

Event 2 occurs only if event 1 occurs.

$0 \leq y \leq u x_1$

$\Rightarrow y - u x_1 \leq 0$

Placing Power Plants in geographic regions

Let c_j be the cost of placing plant at location j.

Let h_{ij} be the per unit cost of satisfying community i's demand from plant j.

Let b_i be the demand of community i.

Let u_j be the capacity of plant j.

Decision variables.

1. where to put the plants. ($0/1$)
2. How much of plant j's energy should be sent to community i.

① $x_j = \begin{cases} 1 & \text{if we put a plant at country } j \\ 0 & \text{otherwise} \end{cases}$

② y_{ij} = amount of energy selected to consume by i from plant j.

Capacity Facility Location Problem

Minimize $\sum_{j=1}^{n} c_j x_j + \sum_{i=1}^{m}\sum_{j=1}^{n} h_{ij} y_{ij}$

(capacity) $\sum_{i=1}^{m} y_{ij} \leq u_j x_j \quad \forall j \in \{1, 2, \ldots, n\}$

(demand) $\sum_{j=1}^{n} y_{ij} \geq b_i \quad \forall i \in \{1, 2, \ldots, m\}$

$y_{ij} \geq 0, \quad x_j \in \{0, 1\}$
$\forall i, j \in m \times n$

Fixed charge Network flow Problem

→ This is the network flow problem with the addition that there is a fixed cost associated with using a network.

Need: supply/demand at each node.
capacity constraints
Node set & arc set.

Let (N, A) be the network.
Let d_i be the demand at node i, $i \in N$.
Let u_{ij} be the capacity constraint on arc (i,j), $(i,j) \in A$

Let c_{ij} be the per unit cost on arc (i,j).
Let h_{ij} be the fixed cost associated with arc (i,j)

Let y_{ij} be the amount of flow on arc (i,j)

Let $x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is used} \\ 0 & \text{if not.} \end{cases}$

$$\text{Minimize} \sum_{(i,j) \in A} x_{ij} h_{ij} + c_{ij} y_{ij}$$

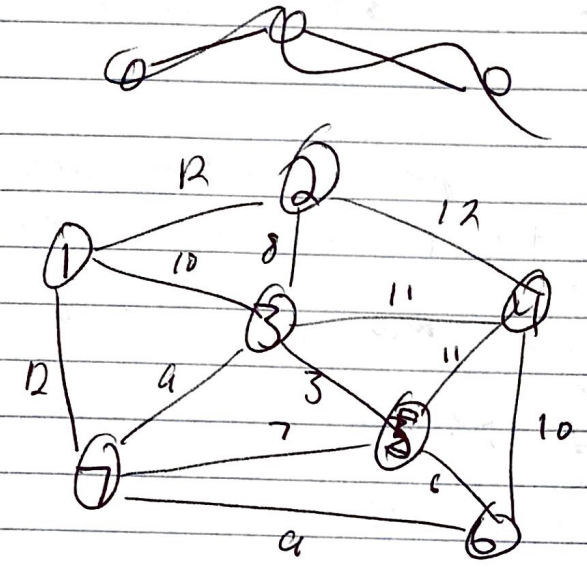
$$y_{ij} \leq u_{ij} x_{ij} \quad \text{for each } (i,j) \in A$$

$$\sum y_{ik} - \sum y_{kj} = d_k \quad k \in N$$

$$y_{ij} \geq 0 \quad \forall i,j$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A$$

Traveling Salesman Problem



Start at a node

Find a path that visits every node and ends up back at the start.

A complete graph K_n

How many paths like this on a complete network?

$$\frac{(n-1)!}{2}$$

since double counting since not a directed graph.

NP-Hard.

Let $x_{ij} = \begin{cases} 1, & \text{if } ij \text{ is connected} \\ 0, & \text{if not.} \end{cases}$

Let c_{ij} be cost of trans ij

minimize $\sum_{(i,j) \in A} x_{ij} c_{ij}$

s.t. $\sum_{(i,j) \in A} x_{ij} = |N|$

$\sum_{(i,l)} x_{il} = 1 \quad \forall l \in N$

$\sum_{(l,i)} x_{li} = 1 \quad \forall l \in N$

$\sum_{i,j \in S} x_{ij} \leq |S| - 1 \quad S \subset N$

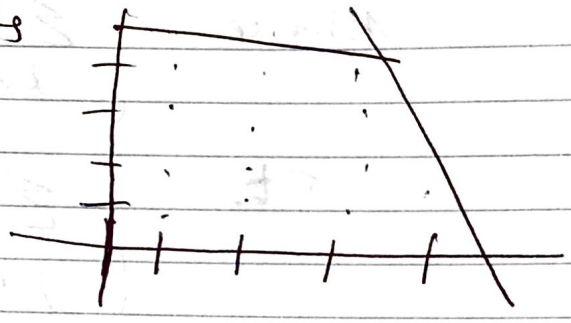
$x_{ij} \in \{0, 1\} \quad (i,j) \in N.$

Today Branch-and-bound algorithm for IP

Solution approaches for solving IP's.

① enumeration.

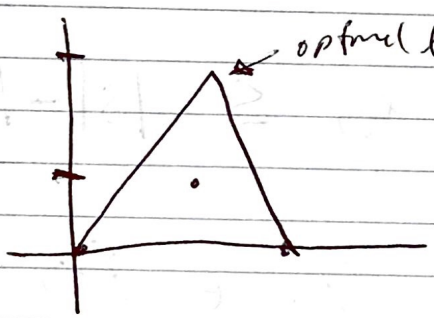
- finitely many points

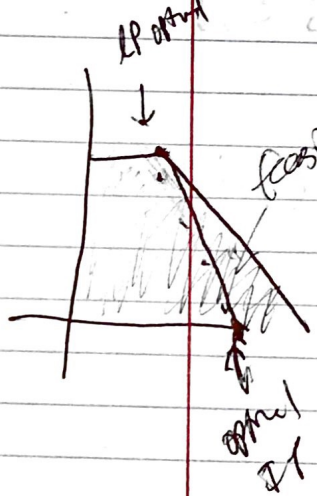


② modify LP algorithm

→ drop integer constraints & solve resulting LP using...

- round solution.





Problems

- Rounding may give an infeasible solution
- May give very off rounding solution

5/5/08

Maximize $7x_1 + 2x_2$

S.t. $-x_1 + 2x_2 \leq 4$

$5x_1 + x_2 \leq 20$

$2x_1 + 2x_2 \geq 7$

$x_1, x_2 \geq 0$, $x_1, x_2 \in \mathbb{Z}$