Math Tripos Part IB: Optimization

Michael Li

May 7, 2016

Lagrangian methods
General formulation of constrained problems; the Lagrangian sufficiency theorem. Interpretation of Lagrange multipliers as shadow prices. Examples. [2]

Linear programming in the nondegenerate case
Convexity of feasible region; sufficiency of extreme points. Standardization of problems, slack variables, equivalence of extreme points and basic solutions. The primal simplex algorithm, artificial variables, the two-phase method. Practical use of the algorithm; the tableau. Examples. The dual linear problem, duality theorem in a standardized case, complementary slackness, dual variables and their interpretation as shadow prices. Relationship of the primal simplex algorithm to dual problem. Two person zero-sum games. [6]

Network problems
The Ford-Fulkerson algorithm and the max-flow min-cut theorems in the rational case. Network flows with costs, the transportation algorithm, relationship of dual variables with nodes. Examples. Conditions for optimality in more general networks; *the simplex-on-a-graph algorithm*. [3]

Practice and applications
Efficiency of algorithms. The formulation of simple practical and combinatorial problems as linear programming or network problems. [1]

Contents

1 Introduction and preliminaries

Note. Almost *everything* is going to be a vector so we do not bold them.

1.1 Constrained optimization

Definition. The general problem is of *constrained optimization* is

$$\text{minimize } f(x) \text{ subject to } h(x) = b,\ x \in X$$

$x \in \mathbb{R}^n$ is the *vector of decision variables*; $f : \mathbb{R}^n \to \mathbb{R}$ the *objective function*; $h : \mathbb{R}^n \to \mathbb{R}^m$, $b \in \mathbb{R}^m$ the *functional constraints*; $X \subseteq \mathbb{R}^n$ the *regional constraint*.

This is the most general form, as maximizing f is same as minimizing $-f$. Constraints in form $h(x) \geq b$ is same as $h(x) - z = b, z \geq 0$, where z is the *slack variable*.

1.2 Linear programming

Linear programming is the case where everything's linear. As we are lazy, we generalize the constraints into the following form:

Definition. The *general form* of a linear program is

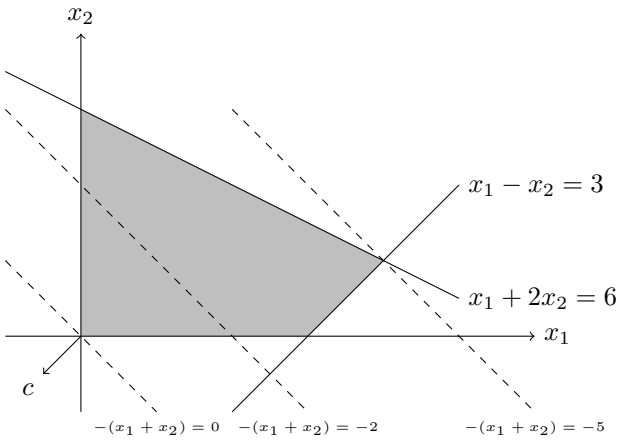
$$\text{minimize } c^T x \text{ subject to } Ax \geq b,\ x \geq 0$$

The *standard form* replaces $Ax \geq b$ with $Ax = b$.

To adapt to actual conditions, such as when x is unconstrained, we let $x = x^+ - x^-$ where x^+, x^- is positive. Others follow similarly.

Example. We want to minimize $-(x_1 + x_2)$, where $x_1, x_2 \geq 0$ subject to

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



The shaded region is the *feasible region*, and c is the *cost vector*. The dotted lines are lines where the objective is constant, so we move it to the rightest point, here at the intersection. But how do we do it in the general case? We have *absolutely* no idea.

But we *do* know how to do *un*constrained optimization.

1.3 Review of unconstrained optimization

Let $f : \mathbb{R}^n \to \mathbb{R}$, $x^* \in \mathbb{R}^n$.To minimize f, the gradient $\nabla f(x^*)$ is necessarily 0, But as we know from Vector Calculus, this is not sufficient, so we have convexity:

Definition. A region $S \subseteq \mathbb{R}^n$ is *convex* iff for $\delta \in [0,1]$, $x, y \in S$, we always have $\delta x + (1-\delta)y \in S$. So if S is convex, the line joining any two points in S lies in S.

Definition. A function $f : S \to \mathbb{R}$ is *convex* if for $x, y \in S$ and $\delta \in [0,1]$, $\delta f(x) + (1-\delta)f(y) \geq f(\delta x + (1-\delta)y)$ is always satisfied.

We have the following lemma:

Lemma. Let f be twice differentiable. Then f is convex on a convex set S if

$$H_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$$

, the Hessian matrix, is positive semidefinite for all $x \in S$. This fancy term means:

Definition. A matrix H is *positive semi-definite* if $v^T H v \geq 0$ for all $v \in \mathbb{R}^n$.

Theorem. Let $X \subseteq \mathbb{R}^n$ be convex, $f : \mathbb{R}^n \to \mathbb{R}$ twice differentiable on X. If $x^* \in X$ satisfy $\nabla f(x^*) = 0$ and H is positive semidefinite on X, then x^* minimizes f on X.

Proof is not given, but note linear functions are convex so this theorem is useful.

2 The method of Lagrange multipliers

We want to minimize $f(x)$ subject to $h(x) = b$, $x \in X$. Call the constraint (P).

Definition (Lagrangian)**.** The *Lagrangian* of a constraint (P) is, for $\lambda \in \mathbb{R}^m$:

$$L(x, \lambda) = f(x) - \lambda^T (h(x) - b).$$

If we minimize L over both x and λ, we will find the minimal solution. Sometimes.

Theorem (Lagrangian sufficiency)**.** Let $x^* \in X$ and $\lambda^* \in \mathbb{R}^m$ such that

$$L(x^*, \lambda^*) = \inf_{x \in X} L(x, \lambda^*) \quad \text{and} \quad h(x^*) = b.$$

Then x^* is optimal for (P).

Proof. We first define the "feasible set": let $X(b) = \{x \in X : h(x) = b\}$. Then:

$$\begin{aligned} \min_{x \in X(b)} f(x) &= \min_{x \in X(b)} (f(x) - \lambda^{*T}(h(x) - b)) \quad \text{[since } h(x) - b = 0\text{]} \\ &\geq \min_{x \in X} (f(x) - \lambda^{*T}(h(x) - b)) = f(x^*) - \lambda^{*T}(h(x^*) - b) = f(x^*). \end{aligned}$$

□

Example. Minimize $x_1 + x_2 - 2x_3$ subject to

$$\begin{aligned} x_1 + x_2 + x_3 &= 5 \\ x_1^2 + x_2^2 &= 4 \end{aligned}$$

The Lagrangian is

$$\begin{aligned} L(x,\lambda) &= x_1 - x_2 - 2x_3 - \lambda_1(x_1 + x_2 + x_3 - 5) - \lambda_2(x_1^2 + x_2^2 - 4) \\ &= ((1-\lambda_1)x_1 - 2\lambda_2 x_1^2) + ((-1-\lambda_1)x_2 - \lambda_2 x_2^2) \\ &+ (-2-\lambda_1)x_3 + 5\lambda_1 + 4\lambda_2 \end{aligned}$$

To use the theorem above, we need a finite minimum for L. Then we see that $\lambda_1 = -2$, since x_3 can take any value. Also, the terms in x_1 and x_2 needs $\lambda_2 < 0$.

With these in mind, we find a minimum by setting all first derivatives to be 0:

$$\frac{\partial L}{\partial x_1} = 1 - \lambda_1 - 2\lambda_2 x_1 = 3 - 2\lambda_2 x_1 \quad \& \quad \frac{\partial L}{\partial x_2} = -1 - \lambda_1 - 2\lambda_2 x_2 = 1 - 2\lambda_2 x_2$$

Since these must be both 0, we must have

$$x_1 = \frac{3}{2\lambda_2}, \quad x_2 = \frac{1}{2\lambda_2}.$$

To show that this is indeed a minimum, we look at the Hessian matrix:

$$HL = \begin{pmatrix} -2\lambda_2 & 0 \\ 0 & -2\lambda_2 \end{pmatrix}$$

It is positive semidefinite when $\lambda_2 < 0$, the condition we need for finite minimum.

Let $Y = \{\lambda : \mathbb{R}^2 : \lambda_1 = -2, \lambda_2 < 0\}$ be our helpful values of λ. We have shown above that for every $\lambda \in Y$, $L(x,\lambda)$ has a unique minimum at $x(\lambda) = (\frac{3}{2\lambda_2}, \frac{1}{2\lambda 2}, x_3)^T$.

Now all we have to do is find λ and x such that $x(\lambda)$ satisfies the functional constraints. Solving algebra gives:

$$x_1 = -3\sqrt{\frac{2}{5}}, \quad x_2 = -\sqrt{\frac{2}{5}}, \quad x_3 = 5 + 4\sqrt{\frac{2}{5}}.$$

And by the theorem, this is optimal.

So far so good. For inequality constraints $h(x) \leq b$, we proceed as follows:

(i) Introduce slack variables to obtain the equivalent problem, to minimize $f(x)$ subject to $h(x) + z = b$, $x \in X$, $z \geq 0$.

(ii) Compute the Lagrangian

$$L(x,z,\lambda) = f(x) - \lambda^T(f(x) + z - b).$$

(iii) Find

$$Y = \left\{ \lambda : \inf_{x \in X, z \geq 0} L(x,z,\lambda) > -\infty \right\}.$$

(iv) For each $\lambda \in Y$, minimize $L(x,z,\lambda)$, ie. find

$$x^*(\lambda) \in X, \quad z^*(\lambda) \geq 0 \quad \text{such that} \quad L(x^*(\lambda), z^*(\lambda), \lambda) = \inf_{x \in X, z \geq 0} L(x,z,\lambda)$$

(v) Find $\lambda^* \in Y$ such that

$$h(x^*(\lambda^*)) + z^*(\lambda^*) = b.$$

Then by Lagrangian sufficiency condition, $x^*(\lambda^*)$ is optimal for the problem.

2.1 Lagrange duality

Consider the problem

$$\text{minimize } f(x) \text{ subject to } h(x) = b, x \in X.$$

Denote this as (P). Define the dual function $g : \mathbb{R}^m \to \mathbb{R}$ as

$$g(\lambda) = \inf_{x \in X} L(x, \lambda).$$

where $L(x, \lambda)$ is the Lagrangian. ie, fix λ, and see how small L can get. As before, let

$$Y = \{\lambda \in \mathbb{R}^n : g(\lambda) > -\infty\}.$$

Then we have

Theorem (Weak duality)**.** If $x \in X(b)$ (ie. x satisfies both the functional and regional constraints) and $\lambda \in Y$, then

$$\sup_{\lambda \in Y} g(\lambda) \leq \inf_{x \in X(b)} f(x).$$

Proof.

$$\begin{aligned} g(\lambda) &= \inf_{x' \in X} L(x', \lambda) \\ &\leq L(x, \lambda) \\ &= f(x) - \lambda^T (h(x) - b) \\ &= f(x). \end{aligned}$$

□

So we can solve a dual problem - instead of minimizing f, we maximize g subject to $\lambda \in Y$, the *dual variables*. Denote this as (D). The original problem (P) is *primal*.

Definition (Strong duality)**.** (P) and (D) are said to satisfy *strong duality* if

$$\sup_{\lambda \in Y} g(\lambda) = \inf_{x \in X(b)} f(x).$$

Problems satisfying strong duality iff the method of Lagrange multipliers work.

Example. Again consider the problem to minimize $x_1 - x_2 - 2x_3$ subject to

$$\begin{aligned} x_1 + x_2 + x_3 &= 5 \\ x_1^2 + x_2^2 &= 4 \end{aligned}$$

We saw that

$$Y = \{\lambda \in \mathbb{R}^2 : \lambda_1 = -2, \lambda_2 < 0\} \quad \& \quad x^*(\lambda) = \left(\frac{3}{2\lambda_2}, \frac{1}{2\lambda_2}, 5 - \frac{4}{2\lambda_2} \right).$$

The dual function is

$$g(\lambda) = \inf_{x \in X} L(x, \lambda) = L(x^*(\lambda), \lambda) = \frac{10}{4\lambda_2} + 4\lambda_2 - 10.$$

The dual is the problem to

$$\text{maximize } \tfrac{10}{4\lambda_2} + 4\lambda_2 - 10 \text{ subject to } \lambda_2 < 0.$$

We can do the calculation and see primal and dual have the same optimal value.

Right now, what we've got isn't helpful, because we won't know if our problem satisfies strong duality!

2.2 Detour: Complementary Slackness + Shadow Prices

Consider the constraint problem. This theorem states (proof tedious, omitted):

Theorem. Assume all functions are continuously differentiable. Suppose for each $b \in \mathbb{R}^n$, $\phi(b)$ is optimal for f and λ^* are the corresponding *dual variables*. Then

$$\frac{\partial \phi}{\partial b_i} = \lambda_i^*.$$

λ_i^*, the *shadow price*, is how much f ($\phi(b)$) increases if we relax the constraint b.

Then we can apply the results to slack variables. Assume we have $h(x^*) = b$. Now we can freely move z around without changing the x^* value, as long as the slack variable is positive. Then to make sure f is optimal under z_i, we must have $\partial\phi\partial b_i = 0$ (necessary condition for a maximum) or $z_i = 0$, because a maximum can only exist at stationary points or boundaries. Then this gives that $z_j\lambda_j = 0$.

This is called *complementary slackness* and it simplifies our calculation considerably:

Example. maximize $x_1 - 3x_2$ subject to ($z_1, z_2 \geq 0$ are slack variables):

$$x_1^2 + x_2^2 + z_1 = 4$$
$$x_1 + x_2 + z_2 = 2$$

The Lagrangian is

$$L(x, z, \lambda) = ((1-\lambda_2)x_1 - \lambda_1 x_1^2) + ((-3-\lambda_2)x_2 - \lambda_1 x_2^2) - \lambda_1 z_1 - \lambda_2 z_2 + 4\lambda_1 + 2\lambda_2.$$

To ensure finite minimum, we need $\lambda_1, \lambda_2 \leq 0$.

By complementary slackness, $\lambda_1 z_1 = \lambda_2 z_2 = 0$. We can then consider the cases $\lambda_1 = 0$ and $z_1 = 0$ separately, and save a lot of algebra.

2.3 Supporting hyperplanes and convexity

We use the fancy term "hyperplane" to denote planes in higher dimensions (in an n-dimensional space, a hyperplane has $n-1$ dimensions).

Definition (Supporting hyperplane)**.** A hyperplane $\alpha : \mathbb{R}^m \to \mathbb{R}$ is *supporting* to ϕ at b if α intersects ϕ at b and $\phi(c) \geq \alpha(c)$ for all c.

Theorem. (P) satisfies strong duality iff there exists a supporting hyperplane at b.

Proof. ($\Leftarrow$) Suppose there is a supporting hyperplane. Then since the plane passes through $\phi(b)$, it must be of the form

$$\alpha(c) = \phi(b) + \lambda^T(c - b).$$

Since this is supporting, $\phi(c) \geq \alpha(c)$. This holds for all $c \in \mathbb{R}^m$, so with above:

$$\begin{aligned}\phi(b) \leq \inf_{c\in\mathbb{R}^m}(\phi(c) - \lambda^T(c-b)) &= \inf_{c\in\mathbb{R}^m}\inf_{x\in X(c)}(f(x) - \lambda^T(h(x)-b)) \\ &= \inf_{x\in X} L(x,\lambda) = g(\lambda)\end{aligned}$$

By weak duality, $g(\lambda) \leq \phi(b)$. So $\phi(b) = g(\lambda)$. So strong duality holds.

($\Rightarrow$). Assume now that we have strong duality. There exists λ that for all $c \in \mathbb{R}^m$,

$$\begin{aligned}\phi(b) = g(\lambda) = \inf_{x \in X} L(x, \lambda) &\leq \inf_{x \in X(c)} L(x, \lambda) \\ &= \inf_{x \in X(c)} (f(x) - \lambda^T (h(x) - b)) = \phi(c) - \lambda^T (c - b)\end{aligned}$$

So $\phi(b) + \lambda^T (c - b) \leq \phi(c)$. This defines a supporting hyperplane. □

Theorem. Suppose $\phi : \mathbb{R}^m \to \mathbb{R}$ is convex and $b \in \mathbb{R}^m$ lies in interior of the set of points where ϕ is finite. Then there is a supporting hyperplane to ϕ at b.

This is the supporting hyperplane theorem. Complementing that is the following:

2.4 A sufficient condition for convexity of ϕ

Theorem. Let

$$\phi(b) = \inf_{x \in X} \{f(x) : h(x) \leq b\}$$

If X, f, h are convex, then so is ϕ (assuming feasibility and boundedness).

Proof. Consider $b_1, b_2 \in \mathbb{R}^m$ such that $\phi(b_1)$ and $\phi(b_2)$ are defined. Let $\delta \in [0, 1]$ and define $b = \delta b_1 + (1 - \delta) b_2$. We want to show that $\phi(b) \leq \delta \phi(b_1) + (1 - \delta)\phi(b_2)$.

Consider $x_1 \in X(b_1)$, $x_2 \in X(b_2)$, and let $x = \delta x_1 + (1 - \delta) x_2$. By convexity of X, $x \in X$. By convexity of h,

$$h(x) = h(\delta x_1 + (1 - \delta) x_2) \leq \delta h(x_1) + (1 - \delta) h(x_2) \leq \delta b_1 + (1 - \delta) b_2 = b$$

So $x \in X$ and $h(x) \leq b$. Since $\phi(b)$ is optimal (inf), then by convexity of f:

$$\phi(b) \leq f(x) = f(\delta x_1 + (1 - \delta) x_2) \leq \delta f(x_1) + (1 - \delta) f(x_2)$$

This holds for any $x_1 \in X(b_1)$ and $x_2 \in X(b_2)$. So by taking infimum:

$$\phi(b) \leq \delta \phi(b_1) + (1 - \delta)\phi(b_2).$$

□

$h(x) = b$ is equivalent to $h(x) \leq b$ and $-h(x) \leq -b$. So the result holds for problems with equality constraints if both h and $-h$ are convex, ie. if $h(x)$ is linear:

Theorem. If a linear program is feasible and bounded, then it satisfies strong duality.

3 Solutions of linear programs

3.1 Basic solutions

Linear objective functions and constraints give "corner" or *extreme point* optimals that is not a convex combination of two distinct points. (See example at start of sheet)

Definition. An *extreme point* x in a convex set S is a point that if $y, z \in S$ and $\delta \in (0, 1)$ satisfy $x = \delta y + (1 - \delta) z$, then $x = y = z$.

Definition. A solution $x \in \mathbb{R}^n$ to the optimization problem is *basic* if it has at most m non-zero entries (out of n), corresponding to length of vector b (which is m). B is called the *basis*, and x_i are the *basic variables* if $i \in B$.

Definition. A basic solution is *non-degenerate* if it has exactly m non-zero entries.It is *feasible* if it satisfies $x \geq 0$ (being a solution only requires $Ax = b$).

Example. Consider the linear program with slack variables and $x_1, x_2, z_1, z_2 \geq 0$:

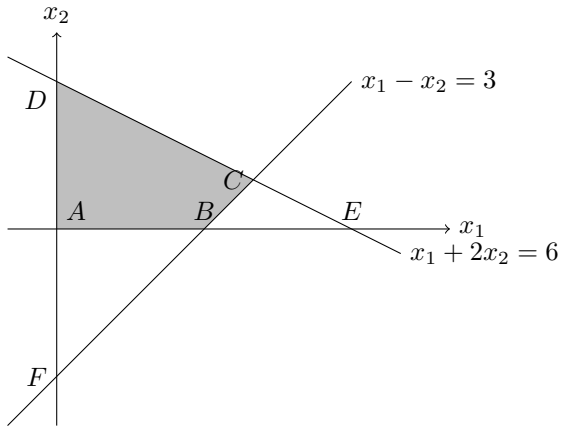
$$\text{maximize } f(x) = x_1 + x_2 \text{ subject to}$$

$$\begin{aligned} x_1 + 2x_2 + z_1 &= 6 \\ x_1 - x_2 + z_2 &= 3 \end{aligned}$$

Since we have 2 constraints, a basic solution would require 2 non-zero entries, and thus 2 zero entries. The possible basic solutions are

	x_1	x_2	z_1	z_2	$f(x)$
A	0	0	6	3	0
B	0	3	0	6	3
C	4	1	0	0	5
D	3	0	3	0	3
E	6	0	0	-4	6
F	0	-3	12	0	-3

Among all 6, E and F are *not* feasible solutions since they have negative entries. So the basic feasible solutions are A, B, C, D.



3.2 Extreme points and optimal solutions

We assume in the following that rows and any set of m columns of A (in $Ax = b$ optimization) are linearly independent. If not, delete the extra rows/columns.

Theorem. A vector x is a basic feasible solution of $Ax = b$ if and only if it is an extreme point of the set $X(b) = \{x' : Ax' = b, x' \geq 0\}$.

We will not prove this. (See 2D) We will prove the following:

Theorem. If (P) is feasible and bounded, then there exists an optimal solution that is a basic feasible solution.

Proof. Let x be optimal of (P). If x has at most m non-zero entries, it is basic feasible, and we are done. Now suppose x has $r > m$ non-zero entries. Since it is not an extreme point, we have $y \neq z \in X(b)$, $\delta \in (0,1)$ such that $x = \delta y + (1-\delta)z$.

We will show there is an optimal solution strictly fewer than r non-zero entries. Then claim follows by induction.When $x_i = 0$, $y_i = z_i = 0$ as $y, z \geq 0$, so y and z at most has r non-zero entries. If they have less than r, we are done. If not, for any $\hat{\delta}$, let

$$x_{\hat{\delta}} = \hat{\delta}y + (1-\hat{\delta})z = z + \hat{\delta}(y-z).$$

Since $c^T x = \delta c^T y + (1-\delta)c^T z$,and $c^T x \geq c^T y, c^T z$ by optimality of x, we have $c^T x = c^T y = c^T z$, so y and z are optimal. So notice $x_{\hat{\delta}}$ is optimal for every $\hat{\delta} \in \mathbb{R}$.

Moreover, $y - z \neq 0$, and all non-zero entries of $y - z$ occur in rows where x is non-zero as well. We can thus choose $\hat{\delta} \in \mathbb{R}$ such that $x_{\hat{\delta}} \geq 0$ and $x_{\hat{\delta}}$ has strictly fewer than r non-zero entries. □

For the maximum of a convex function over a compact convex set X, any point in X is a convex combination of extreme points, so the value of f on any point in the interior is less than the value on the best extreme point. (Quick exercise for reader)

Note. Basically what this section says is that our solutions must be on the vertices of polytopes in n dimensions, where n is the number of explanatory variables we have.

3.3 Linear programming duality

Consider the linear program in general form with slack variables,

$$\text{minimize } c^T x \text{ subject to } Ax - z = b,\ x, z \geq 0$$

We have $X = \{(x, z) : x, z \geq 0\} \subseteq \mathbb{R}^{m+n}$. The Lagrangian is

$$L(x, z, \lambda) = c^T x - \lambda^T (Ax - z - b) = (c^T - \lambda^T A)x + \lambda^T z + \lambda^T b.$$

Since x, z can be arbitrarily positive, this has a finite minimum if and only if

$$c^T - \lambda^T A \geq 0, \lambda^T \geq 0.$$

Call the feasible set Y. Then for fixed $\lambda \in Y$, the minimum of $L(x, z, \lambda)$ is attained when $(c^T - \lambda^T A)x$ and $\lambda^T z = \lambda^T (Ax - b) = 0$ by complementary slackness. So

$$g(\lambda) = \inf_{(x,z)\in X} L(x, z, \lambda) = \lambda^T b.$$

The dual is thus

$$\text{maximize } \lambda^T b \text{ subject to } A^T \lambda \leq c, \lambda \geq 0$$

Theorem. The dual of the dual of a linear program is the primal.

Proof. It suffices to show this for the linear program in general form. We have shown above that the dual problem is

$$\text{minimize } -b^T \lambda \text{ subject to } -A^T \lambda \geq -c, \lambda \geq 0.$$

Taking the dual of this problem (treating this as primal), gives the primal problem. □

Theorem. Let x and λ be feasible for the primal and the dual of the linear program in general form. Then x and λ and optimal if and only if they satisfy complementary slackness, ie. if

$$(c^T - \lambda^T A)x = 0 \text{ and } \lambda^T(Ax - b) = 0.$$

Proof. If x and λ are optimal, and as every linear program satisfies strong duality:

$$c^T x = \lambda^T b = \inf_{x' \in X}(c^T x' - \lambda^T(Ax' - b)) \leq c^T - \lambda^T(Ax - b) \leq c^T x$$

Since $Ax \geq b$ and $\lambda \geq 0$. The first and last term are the same so inequality is an equality. Therefore

$$\lambda^T b = c^T x - \lambda^T(Ax - b) = (c^T - \lambda^T A)x + \lambda^T b. \;\Rightarrow\; (c^T - \lambda^T A)x = 0.$$

Also,

$$c^T x - \lambda^T(Ax - b) = c^T x \;\Rightarrow\; \lambda^T(Ax - b) = 0.$$

On the other hand, suppose we have complementary slackness, ie.

$$(c^T - \lambda^T A)x = 0 \text{ and } \lambda^T(Ax - b) = 0,$$

then

$$c^T x = c^T x - \lambda^T(Ax - b) = (c^T - \lambda^T A)x + \lambda^T b = \lambda^T b.$$

Hence by weak duality, x and λ are optimal. □

3.4 Simplex method

Note. This section looks harder than it actually is. Just do some questions and you will understand how it works. We are basically moving along lines in n dimensional spaces to reach from vertex to vertex.

We adopt the following notation: let $A \subseteq \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Assume A has full rank. Let B be a basis with $|B| = m$, meaning at most m non-zero entries. We rearrange the columns so that all basis columns are on the left. Then our matrices become:

$$\begin{aligned} A_{m\times n} &= \begin{pmatrix}(A_B)_{m\times m} & (A_N)_{m\times(n-m)}\end{pmatrix} \\ x_{n\times 1} &= \begin{pmatrix}(x_B)_{m\times 1} & (x_N)_{(n-m)\times 1}\end{pmatrix}^T \\ c_{1\times n} &= \begin{pmatrix}(c_B)_{m\times 1} & (c_N)_{(n-m)\times 1}\end{pmatrix}. \end{aligned}$$

Then

$$Ax = b \;\Rightarrow\; A_B x_B + A_N x_N = b. \;\Rightarrow\; x_B = A_B^{-1}(b - A_N x_N).$$

In particular, when $x_N = 0$, then

$$x_B = A_B^{-1} b.$$

The general tableau is then

Basis components	Other components	
$A_B^{-1}A_B = I$	$A_B^{-1}A_N$	$A_B^{-1}b$
$c_B^T - c_B^T A_B^{-1} A_B = 0$	$c_N^T - c_B^T A_B^{-1} A_N$	$-c_B^T A_B^{-1} b$

3.4.1 The simplex tableau

$$f(x) = c^T x = c_B^T x_B + c_N^T x_N = c_B^T A_B^{-1}(b - A_N x_N) + c_N^T x_N$$
$$= c_B^T A_B^{-1} b + (c_N^T - c_B^T A_B^{-1} A_N) x_N.$$

We will maximize $c^T x$ by choosing a basis such that $c_N^T - c_B^T A_B^{-1} A_N \leq 0$ and $A_B^{-1} b \geq 0$. Then for any feasible solution $x \in \mathbb{R}^n$, we must have $x_N \geq 0$. So:

$$f(x) \leq c_B^T A_B^{-1} b.$$

So if we choose $x_B = A_B^{-1} b$, $x_N = 0$, we have a basic feasible (optimal) solution.

Hence our objective is to pick a basis making $c_N^T - c_B^T A_B^{-1} A_N \leq 0$ while keeping $A_B^{-1} b \geq 0$. To do this, suppose this is not attained. Say $(c_N^T - c_B^T A_B^{-1} A_N)_i > 0$.

We can increase the objective function value by increasing $(x_N)_i$. As we increase $(x_N)_i$, due to constraints, other variables decrease,so when another variable, $(x_B)_j$, hits 0, we stop. However, if we can do this indefinitely, then the problem is unbounded.

Intuitively, we switched basis variable $(x_B)_j$ with $(x_N)_i$. If $(c_N^T - c_B^T A_B^{-1} A_N)$ is still positive, we repeat. The simplex method is systemically doing the steps above.

3.4.2 Using the Tableau

a_{ij}	a_{i0}
a_{0j}	a_{00}

a_{i0} is b, a_{0j} is the objective function, and a_{00} is initially 0. The method follows below:

(i) Find an initial basic feasible solution. Check whether $a_{0j} \leq 0$ for every j. If so, the current solution is optimal. Stop.

(ii) If not, choose a *pivot column* j such that $a_{0j} > 0$. Choose a *pivot row* $i \in \{i : a_{ij} > 0\}$ that minimizes a_{i0}/a_{ij}, which ensures no basic variables become negative before the chosen one is 0. If multiple rows minimize a_{i0}/a_{ij}, then things *might* go wrong (*degenerate*). If $a_{ij} \leq 0$ for all i, the problem is unbounded.

(iii) Update tableau by multiplying row i by $1/a_{ij}$ (so $a_{ij} = 1$), and add multiples of row i to each row $k \neq i$ so that $a_{kj} = 0$ for all $k \neq i$. Then this is basic feasible, as all right-hand columns apart from a_{00} are positive due to our choice of row. If optimal, we stop. If not, GOTO (ii).

Example. Consider the following problem:

Maximize $x_1 + x_2$, $x_1, x_2, z_1, z_2 \geq 0$ subject to

$$x_1 + 2x_2 + z_1 = 6 \quad \text{and} \quad x_1 - x_2 + z_2 = 3$$

	x_1	x_2	z_1	z_2	
Constraint 1	1	2	1	0	6
Constraint 2	1	-1	0	1	3
Objective	1	1	0	0	0

In the *simplex tableau* above,the identity matric in z_1 and z_2 columns correspond to basic feasible solution: $z_1 = 6, z_2 = 3, x_1 = x_2 = 0$. It's not optimal, so we increase x_2, choose first row, multiply the first row by $\frac{1}{2}$, and subtract/add it to the other rows:

	x_1	x_2	z_1	z_2	
Constraint 1	$\frac{1}{2}$	1	$\frac{1}{2}$	0	3
Constraint 2	$\frac{2}{3}$	0	$\frac{1}{2}$	1	6
Objective	$\frac{1}{2}$	0	$-\frac{1}{2}$	0	-3

Our new basic feasible solution is $x_2 = 3, z_2 = 6, x_1 = z_1 = 0$. We see the number in the bottom-right is $-f(x)$. Then we repeat for final solution.

3.5 The two-phase simplex method

If we don't have an identity matrix (basic feasible) to start with, we use the *two-phase simplex method* to reach one, as illustrated by example below:

Example. Consider the example:

$$\text{maximize } -6x_1 - 3x_2, x_1, x_2, z_1, z_2, z_3 \geq 0 \text{ subject to}$$

$$x_1 + x_2 - z_1 = 1 \quad \text{and} \quad 2x_1 - x_2 - z_2 = 1 \quad \text{and} \quad 3x_2 + z_3 = 2$$

Now we don't have a basic feasible solution, since we would need $z_1 = z_2 = -1, z_3 = 2$, which is not feasible. So we add *more* variables, called the artificial variables.

$$\text{maximize } -6x_1 - 3x_2, x_1, x_2, z_1, z_2, z_3, y_1, y_2 \geq 0 \text{ subject to}$$

$$x_1 + x_2 - z_1 + y_1 = 1 \quad \text{and} \quad 2x_1 - x_2 - z_2 + y_2 = 1 \quad \text{and} \quad 3x_2 + z_3 = 2$$

We try to make y_1 and y_2 both 0 and find a basic feasible solution. Then we throw away y_1 and y_2 for a basic feasible for our original problem. Thus, our problem becomes:

$$\text{minimize } y_1 + y_2, x_1, x_2, z_1, z_2, z_3, y_1, y_2 \geq 0 \text{ subject to}$$

$$x_1 + x_2 - z_1 + y_1 = 1 \quad \text{and} \quad 2x_1 - x_2 - z_2 + y_2 = 1 \quad \text{and} \quad 3x_2 + z_3 = 2$$

By minimizing y_1 and y_2, we will make them zero. Our simplex tableau is

x_1	x_2	z_1	z_2	z_3	y_1	y_2	
1	1	-1	0	0	1	0	1
2	-1	0	-1	0	0	1	1
0	3	0	0	1	0	0	2
-6	-3	0	0	0	0	0	0
0	0	0	0	0	-1	-1	0

Note that we keep both our original and "kill-y_i" objectives. (we only care about killing y for now) We have $y_1 = y_2 = 1, z_3 = 2$ as basic feasible. However, y_1, y_2 being basis columns should not have other non-zero entries. (two -1s at the bottom) So we add the first two rows to the last to obtain the table on the left.

x_1	x_2	z_1	z_2	z_3	y_1	y_2	
1	1	-1	0	0	1	0	1
2	-1	0	-1	0	0	1	1
0	3	0	0	1	0	0	2
-6	-3	0	0	0	0	0	0
3	0	-1	-1	0	0	0	2

$\Rightarrow$

x_1	x_2	z_1	z_2	z_3	y_1	y_2	
0	3	-2	1	0	2	-1	1
1	1	-1	0	0	1	0	1
0	3	0	0	1	0	0	2
0	3	-6	0	0	6	0	6
0	0	0	0	0	-1	-1	0

Then we do our standard simplex step 2 and 3 (x_1 with second row, then z_2 with first row) to obtain the table on the right above. (after adding/subtracting rows)

y_1 and y_2 are no longer in the basis, and take value 0. So drop all ys to get table on left:

x_1	x_2	z_1	z_2	z_3	
0	3	-2	1	0	1
1	1	-1	0	0	1
0	3	0	0	1	2
0	3	-6	0	0	6

$\Rightarrow$

x_1	x_2	z_1	z_2	z_3	
0	1	$-\frac{2}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$
1	0	$-\frac{1}{3}$	$-\frac{1}{3}$	0	$\frac{2}{3}$
0	0	2	-1	1	1
0	0	-4	-1	0	5

We see a basic feasible solution $z_1 = x_1 = 1, z_3 = 2$. We pick x_2 as the pivot column, and the first row as the pivot row. Then we have the table on the right.

Since the last row is all negative, we have complementary slackness. (This is because the derivative with respect to the slack variables is not zero, so must be at the boundary), so this is a optimal. So $x_1 = \frac{2}{3}, x_2 = \frac{1}{3}, z_3 = 1$ is a feasible solution, and optimal value is -5.

Note that the bottom right entry is the negative of the optimal value, not the optimal value itself! (See general tableau at start of section)

4 Non-cooperative games

4.1 Games and Solutions

Definition. A two-player or *bimatrix game* is given by matrices $P, Q \in \mathbb{R}^{m\times n}$. The *row player* chooses a row $i \in \{1, \cdots, m\}$, while the *column player* chooses a column $j \in \{1, \cdots, n\}$ independently. The players then get payoffs P_{ij} and Q_{ij} respectively.

Definition. Players can play randomly. The set of *strategies* the row/column player can have is, respectively: (vector represent probabilities of picking each row/column)

$$X = \{x \in \mathbb{R}^m : x \geq 0, \sum x_i = 1\} \quad \text{and} \quad Y = \{y \in \mathbb{R}^n : y \geq 0, \sum y_i = 1\}$$

A strategy profile $(x, y) \in X \times Y$ induces a lottery, and we write $p(x, y) = x^T Py$ for the row player expected payoff. If $x_i = 1$ for some i, we call x a *pure strategy*.

Example. Alice and bob commit a crime together and are caught. They can be silent (S) or testify (T). If both testify, they get 5 years; if one testify then the testifier is free and other gets 10 years; if both silent then they get 2 years. So have a payoff table:

	S	T
S	$(2, 2)$	$(0, 3)$
T	$(3, 0)$	$(1, 1)$

Payoffs are relative so actual values don't matter. Higher payoff is desired so more jail time means less payoff. Here it is always strictly better to testify, so T is a *dominant strategy*, and $(1,1)$ is *Pareto dominated* by $(2,2)$ as both are better off at $(2,2)$.

Example. The chicken game involves two people driving at each other, where both of them can choose to chicken out (C) or not (D). The payoff is like the following:

	C	D
C	$(2,2)$	$(1,3)$
D	$(3,1)$	$(0,0)$

There is no dominating strategy, so define the *security level* of the row player to be:

$$\max_{x\in X}\min_{y\in Y} p(x,y) = \max_{x\in X}\min_{j\in\{1,\dots,n\}}\sum_{i=1}^{m} x_i p_{ij}.$$

This is the strategy the row player can employ that minimizes the possible loss. This is called the maximin strategy. We can formulate this as a linear program:

$$\text{maximize } v, (x \geq 0) \text{ so that } \sum_{i=1}^{m} x_i p_{ij} \geq v \quad \text{for all } j = 1, \cdots, n \quad \text{and} \quad \sum_{i=1}^{m} x_i = 1$$

Here the maximin strategy is to chicken. However, this is not what we want as if both players employ this strategy, you should not chicken instead.

Definition. A strategy $x \in X$ is a *best response* to $y \in Y$ if for all $x' \in X$

$$p(x,y) \geq p(x',y)$$

A pair (x,y) is an *equilibrium* if x is the best response against y and vice versa.

Example. In the chicken game, the equilibrium is $(3,1)$ and $(1,3)$, and there is a mixed equilibrium in which the players pick the options with equal probability.

Theorem (Nash, 1961)**.** Every bimatrix game has an equilibrium.

4.2 The minimax theorem

Definition. A bimatrix game is a *zero-sum game*, or matrix game, if $q_{ij} = -p_{ij}$ for all i, j, ie. the total payoff is always 0 and only one matrix is needed.

Theorem (von Neumann, 1928)**.** If $P \in \mathbb{R}^{m\times n}$. Then

$$\max_{x\in X}\min_{y\in Y} p(x,y) = \min_{y\in Y}\max_{x\in X} p(x,y). \quad \text{aka} \quad \max_{x\in X}\min_{y\in Y} p(x,y) = -\max_{y\in Y}\min_{x\in X} -p(x,y).$$

The equation states that the maximized minimum payoff it the minimized maximum payoff. So when both players use minimax (minimize maximum payoff of opponent), they reach equilibrium (where one gets payoff V and other $-V$, see equation on right).

Proof. The optimal value of $\max\min p(x,y)$ is a solution to the linear program

$$\text{maximize } v, (x \geq 0) \text{ so that } \sum_{i=1}^{m} x_i p_{ij} \geq v \quad \text{for all } j = 1, \cdots, n \quad \text{and} \quad \sum_{i=1}^{m} x_i = 1$$

Adding slack variable $z \in \mathbb{R}^n$ with $z \geq 0$, we obtain the Lagrangian:

$$L(v, x, z, w, y) = v + \sum_{j=1}^{n} y_j \left(\sum_{i=1}^{m} x_i p_{ij} - z_j - v \right) - w \left(\sum_{i=1}^{m} x_i - 1 \right),$$

where $w \in \mathbb{R}$ and $y \in \mathbb{R}^n$ are Lagrange multipliers. This is equal to

$$\left(1 - \sum_{j=1}^{n} y_j \right) v + \sum_{i=1}^{m} \left(\sum_{j=1}^{n} p_{ij} y_j - w \right) x_i - \sum_{j=1}^{n} y_j z_j + w.$$

This has finite minimum iff $\sum y_i = 1, \sum p_{ij} y_j \leq w \ \forall i$, and $y \geq 0$. So the dual is:

$$\text{minimize } w, (y \geq 0) \text{ so that } \sum_{j=1}^{n} p_{ij} y_j \leq w \quad \text{for all } i \quad \text{and} \quad \sum_{j=1}^{n} y_j = 1$$

Thus, this corresponds to column player using minimax, so optimal of the dual is $\min_{y \in Y} \max_{x \in X} p(x, y)$. Result follows from strong duality. □

Theorem. $(x, y) \in X \times Y$ is an equilibrium of a payoff matrix P zero-sum game iff

$$\min_{y' \in Y} p(x, y') = \max_{x' \in X} \min_{y' \in Y} p(x', y') \quad \text{and} \quad \max_{x' \in X} p(x', y) = \min_{y' \in Y} \max_{x' \in X} p(x', y')$$

5 Network problems

5.1 Definitions

Definition. A *directed graph* or *network* is a pair $G = (V, E)$, where V is the set of vertices and $E \subseteq V \times V$ the set of edges. If $(u, v) \in E$, there is an edge from u to v.

Definition. The degree of a vertex $u \in V$ is # of $v \in V$ so that (u, v) or $(v, u) \in E$.

Definition. A walk from v_1 to v_k is a sequence of vertices $v_1, \cdots v_k$ with $(v_i, v_{i+1}) \in E \ \forall i$, and is a path if $v_1, \cdots v_k$ are pairwise distinct, or a cycle when $v_1, \cdots, v_{k-1}$ are pairwise distinct and $v_1 = v_k$. An *undirected walk* allows $(v_{i+1}, v_i) \in E$.

Definition. A graph is *connected* if for any pair of vertices, there is an undirected path between them. A *tree* is a connected graph without (undirected) cycles. The *spanning tree* of a graph $G = (V, E)$ is a tree (V', E') with $V' = V$ and $E' \subseteq E$.

5.2 Minimum-cost flow problem

Let $G = (V, E)$ be a directed graph with $|V| = n$ vertices, and $b \in \mathbb{R}^n$. For each edge, we assign a cost, an lower bound and a upper bound, denoted $C, \underline{M}, \overline{M} \in \mathbb{R}^{n \times n}$.

b_i is the amount produced ($b_i > 0$)/consumed ($b_i < 0$) at ith node, which is then called a *source*/*sink* respectively. c_{ij} is the unit cost of moving stuff from i to j (0 if no edge), and $\underline{m}_{ij}/\overline{m}_{ij}$ are lower/upper bounds on the flow in $(i, j) \in E$ respectively.

$x \in \mathbb{R}^{n \times n}$ is a minimum-cost flow if it minimizes the cost of transferring stuff, while satisfying the constraints, or it minimizes $\sum_{(i,j) \in E} c_{ij} x_{ij}$ subject to

$$b_i + \sum_{j:(j,i) \in E} x_{ji} = \sum_{j:(i,j) \in E} x_{ij} \ \ \forall i \in V \quad \text{and} \quad \underline{m}_{ij} \leq x_{ij} \leq \overline{m}_{ij} \ \ \forall (i, j) \in E.$$

This problem is linear so we can use simplex. But this is very inefficient and we have a better way. For the system to work, we need:

$$\sum_{i\in V} b_i = 0.$$

To simplify, we convert it into a *circulation problem*, where $b_i = 0$ for all i, by adding a vertex so sources/sinks give/take extra stuff to/from there. An *uncapacitated problem* is a case where $\underline{m}_{ij} = 0$ and $\overline{m}_{ij} = \infty\ \forall (i,j) \in E$. If an uncapacitated problem is bounded, it is equivalent to one with finite capacities. But we can simplify further:

5.3 The transportation problem

The transportation problem is a special case where the graph is a *bipartite graph*, ie. we can split the vertices into two halves A, B, where all edges flow from a vertex in A to a vertex in B. We call A vertices the *suppliers* and B vertices the *consumers*.

In this case, the problem is to minimize $\displaystyle\sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij}$, with $x \geq 0$ subject to

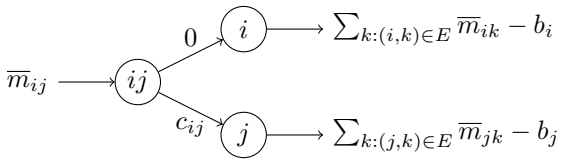
$$\sum_{j=1}^{m} x_{ij} = s_i \text{ for } i = 1, \cdots, n \quad \text{and} \quad \sum_{i=1}^{n} x_{ij} = d_j \text{ for } j = 1, \cdots, m$$

s_i/d_i is the supply/demand of each supplier/consumer. We have $s \in \mathbb{R}^n, d \in \mathbb{R}^m$ satisfying $s, d \geq 0$, $\sum s_i = \sum d_j$,and $c \in \mathbb{R}^{n\times m}$ representing the cost of transferal.

Theorem. Every minimum cost-flow problem with finite capacities or non-negative costs has an equivalent transportation problem.

Proof. For a minimum-cost flow problem on network (V, E). wlog assume that $\underline{m}_{ij} = 0$ for all $(i,j) \in E$. If not, $\underline{m}_{ij}$ to 0, $\overline{m}_{ij}$ to $\overline{m}_{ij} - \underline{m}_{ij}$, b_i to $b_i - \underline{m}_{ij}$, b_j to $b_j + \underline{m}_{ij}$, x_{ij} to $x_{ij} - \underline{m}_{ij}$, shipping the minimum secretly.

Also, assume finite capacities as we have finite amount of stuff so max shipping through each path is finite. We construct the transportation problem as followed:



Now we let every $i \in V$ be a consumer, and consume $\left(\sum_{k:(i,k)\in E} \overline{m}_{ik}\right) - b_i$. For every edge (i,j), add a supplier ij with supply $\overline{m}_{ij}$, an edge to i with cost 0 and an edge to j with cost c_{ij}.

For example if edge (i,j) has capacity 5 to start, and we use 3, then in the new one, we send 3 from ij to j, and 2 to i. But how do we ensure same constraints?

For any flow x originally, the new flow on (ij, j) is x_{ij} and the flow on (ij, i) is $\overline{m}_{ij} - x_{ij}$. So the constraints of the new network is satisfied iff

$$\sum_{k:(i,j)\in E} (\overline{m}_{ik} - x_{ij}) + \sum_{k:(k,i)\in E} x_{ki} = \sum_{k:(i,k)\in E} \overline{m}_{ik} - b_i,$$

where the left hand side is the total flow into i. This statement is equivalent to:

$$b_i + \sum_{k:(k,i)\in E} x_{ki} - \sum_{k:(i,k)\in E} x_{ik} = 0,$$

which is the constraint for the node i in the original minimal-cost flow problem. □

To solve the problem, having two lagrangian multipliers for the suppliers and consumers constraints is useful as the Lagrangian is, after simplifying:

$$L(x,\lambda,\mu) = \sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij} - \lambda_i + \mu_j)x_{ij} + \sum_{i=1}^{n}\lambda_i s_i - \sum_{j=1}^{m}\mu_j d_j.$$

We used $+$ sign and λ_i symbol for supplier constraints and $-$ with μ_i for consumer constraints. As $x \geq 0$, the Lagrangian has finite minimum iff $c_{ij} - \lambda_i + \mu_j \geq 0$ i, j. So by complementary slackness:

$$(c_{ij} - \lambda_i + \mu_j)x_{ij} = 0 \ \ \forall i, j.$$

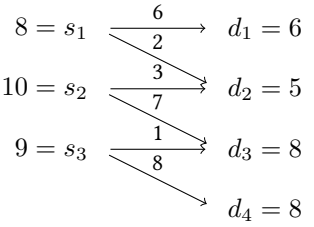
Then we have a tableau, with rows for suppliers and columns for consumers:

	μ_1	μ_2	μ_3	μ_4	
λ_1	$\lambda_1-\mu_1$ | x_{11} | c_{11}	$\lambda_1-\mu_2$ | x_{12} | c_{12}	$\lambda_1-\mu_3$ | x_{13} | c_{13}	$\lambda_1-\mu_4$ | x_{14} | c_{14}	s_1
λ_2	$\lambda_2-\mu_1$ | x_{21} | c_{21}	$\lambda_2-\mu_2$ | x_{22} | c_{22}	$\lambda_2-\mu_3$ | x_{23} | c_{23}	$\lambda_2-\mu_4$ | x_{24} | c_{24}	s_2
λ_3	$\lambda_3-\mu_1$ | x_{31} | c_{31}	$\lambda_3-\mu_2$ | x_{32} | c_{32}	$\lambda_3-\mu_3$ | x_{33} | c_{33}	$\lambda_3-\mu_4$ | x_{34} | c_{34}	s_3
	d_1	d_2	d_3	d_4	

Example. We have suppliers with supplies $8, 10$ and 9, and consumers with demands $4, 5, 8, 8$. We create an initial feasible solution starting from first supplier and consumer and supplying until one runs out space/stuff. We then fill the table:

6 | 5	2 | 3	| 4	| 6	8
| 2	3 | 7	7 | 4	| 1	10
| 5	| 6	1 | 2	8 | 4	9
6	5	8	8		

$\Leftrightarrow$



This shows our solution is a spanning tree. In general, n suppliers and m consumers gives $n + m$ vertices, $n + m - 1$ edges, and $n + m - 1$ dual constraints. So choosing one lagrange multiplier determines the rest. We choose $\lambda_1 = 0$ as:

$$(c_{ij} - \lambda_i + \mu_i)x_{ij} = 0,$$

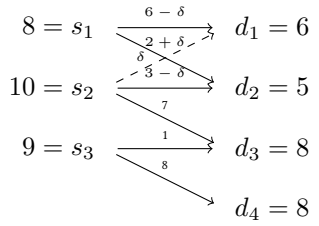
for edges in the spanning tree, $x_{ij} \neq 0$. So $c_{ij} - \lambda_i + \mu_i = 0$. Thus $\mu_1 = -5$. We then fill in other Lagrange multipliers and the values of $\lambda_i - \mu_i$ to obtain:

	-5	-3	0	-2
			0	2
0	6 | 5	2 | 3	| 4	| 6
	9			6
4	| 2	3 | 7	7 | 4	| 1
	7	5		
2	| 5	| 6	1 | 2	8 | 4

The dual feasibility condition is $\lambda_i - \mu_i \leq c_{ij}$. If it is always satisfied, we are done. Otherwise, we add an edge, say from supplier 2 to consumer 1 to make a cycle. We increase the flow on the new edge by δ so that another edge in the cycle goes to zero:

$6-\delta$	5	$2+\delta$	3		4	6
δ	2	$3-\delta$	7	7	4	1
	5		6	1	2	8	4

$\Leftrightarrow$



Biggest possible δ is 3. So we have, after re-computing the Lagrange multipliers:

	-5		-3		-7		-9	
					7		9	
0	3	5	5	3		4		6
			0				9	
-3	3	2		7	7	4		1
	0		-2					
-5		5		6	1	2	8	4

We see a violation at the middle right (matrix element 23). So we do it again with $\delta = 7$:

3	5	5	3		4		6
3	2		7	$7-\delta$	4	δ	1
	5		6	$1+\delta$	2	$8-\delta$	4

$\Rightarrow$

	-5		-3		-2		-4	
					2		4	
0	3	5	5	3		4		6
			0		-1			
-3	3	2		7		4	7	1
	5		3					
0		5		6	8	2	1	4

No more violations. Finally. So this is the optimal solution.

5.4 The maximum flow problem

Suppose a network (V, E) has a single source 1 and a single sink n. Cost is zero but there is finite capacity and we want to move as much stuff from 1 to n as possible.

We can make this a minimum-cost flow problem by adding an edge from n to 1 with -1 cost and infinite capacity. Then minimal cost flow will maximize flow from n to 1 (thus 1 to n). But we don't have to, as we have the *max-flow min-cut theorem.*

5.4.1 The max-flow min-cut theorem

Definition. Suppose $G = (V, E)$ with capacities C_{ij} for $(i, j) \in E$. A *cut* of G is a partition of V into two sets. For $S \subseteq V$, the *capacity* of the cut $(S, V \setminus S)$ is

$$C(S) = \sum_{(i,j)\in(S\times(V\setminus S))\cap E} C_{ij},$$

ie. the total capacity of the edges "cut apart" by the cut.

Let x be a feasible flow vector sending δ units from 1 to n. For $X, Y \subseteq V$, define

$$f_x(X, Y) = \sum_{(i,j)\in(X\times Y)\cap E} x_{ij},$$

ie. the overall amount of flow from X to Y. Then for any cut $S \subseteq V$ with $1 \in S, n \in V \setminus S$, we claim that

$$\delta = \sum_{i \in S} \left(\sum_{j:(i,j)\in E} x_{ij} - \sum_{j:(j,i)\in E} x_{ji} \right).$$

As by flow conversation, for any $i \neq 1$, $\sum_{j:(i,j)\in E} x_{ij} - \sum_{j:(j,i)\in E} x_{ji} = 0$, and for $i = 1$, it is δ. So the sum is δ. Hence

$$\begin{aligned}\delta = f_x(S, V) - f_x(V, S) &= f_x(S, S) + f_x(S, V \setminus S) - f_x(V \setminus S, S) - f_x(S, S)\\ &= f_x(S, V \setminus S) - f_x(V \setminus S, S) \leq f_x(S, V \setminus S) \leq C(S)\end{aligned}$$

This states the obvious:the flow through the cut is less than the capacity through it.

Theorem (Max-flow min-cut theorem)**.** Let δ be an optimal solution. Then

$$\delta = \min\{C(S) : S \subseteq V, 1 \in S, n \in V \setminus S\}$$

Proof. Consider any feasible flow vector x. Call a path $v_0, \cdots, v_k$ an *augmenting path* if the flow along the path can be increased. Formally, it is a path that satisfies

$$x_{v_{i-1}v_i} < C_{v_{i-1}v_i} \text{ or } x_{v_i v_{i-1}} > 0$$

for $i = 1, \cdots, k$. The first condition says that we have a forward edge where we have not hit the capacity, while the second condition says that we have a backwards edge with positive flow. If these conditions are satisfied, we can increase the (forward) flow of each edge, and the total flow increases. Now assume that x is optimal and let

$$S = \{1\} \cup \{i \in V : \text{ there exists an augmenting path from 1 to } i\}.$$

Since there is an augmenting path from 1 to S, we can increase flow from 1 to any vertex in S. So $n \notin S$ by optimality. So $n \in V \setminus S$.

We have previously shown that

$$\delta = f_x(S, V \setminus S) - f_x(V \setminus S, S).$$

We now claim that $f_x(V \setminus S, S) = 0$. If it is not 0, it means that there is a node $v \in V \setminus S$ such that there is flow from v to a vertex $u \in S$. Then we can add that edge to the augmenting path to u to obtain an augmenting path to v.

Also, we must have $f_x(S, V \setminus S) = C(S)$. Or else, we can still send more things to the other side so there is an augmenting path. So we have

$$\delta = C(S).$$

□

5.4.2 Ford-Fulkerson algorithm

The algorithm is simple:

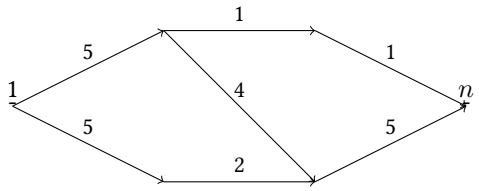
(i) Start from a feasible flow x, eg. $x = \mathbf{0}$.

(ii) If there is no augmenting path for x form 1 to n, then x is optimal.

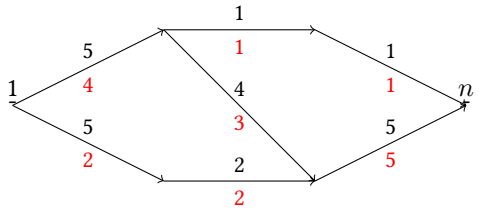
(iii) Find an augmenting path for x from 1 to n, and send a maximum amount of flow along it. Go to 2.

The max-flow min-cut theorem plays a role in step (ii). We can immediately prove that it is maximal by finding an appropriate cut.

Example. Consider the diagram



We can keep adding flow until we reach



(red is flow, black is capacity). We know this is an optimum, since our total flow is 6, and we can draw a cut with capacity 6: