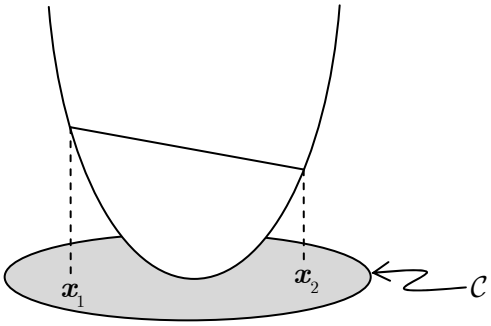
OPTIMIZATION I

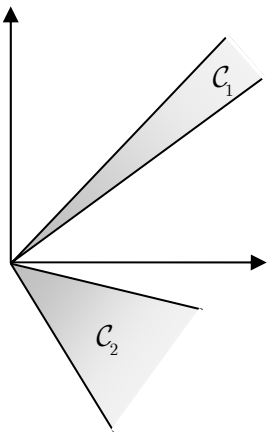
Introduction – Lots of Geometry

- *Some definitions*
  - ***Definition (Convex set)***: The set $\mathcal{C} \subseteq \mathbb{R}^n$ if *convex* if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{C}$ and $\lambda \in [0,1]$, $\boldsymbol{x}(\lambda) = \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \mathcal{C}$. Note that the intersection of a finite number of convex sets is convex.
  - ***Definition (Convex function)***: A function $f(\boldsymbol{x})$ defined on a convex set $\mathcal{C} \subseteq \mathbb{R}^n$ is *convex* if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{C}$, the linear interpolation between those two points lies above the curve:

$$f\left(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2\right) \leq \lambda f\left(\boldsymbol{x}_1\right) + (1-\lambda) f\left(\boldsymbol{x}_2\right) \qquad \lambda \in [0,1]$$

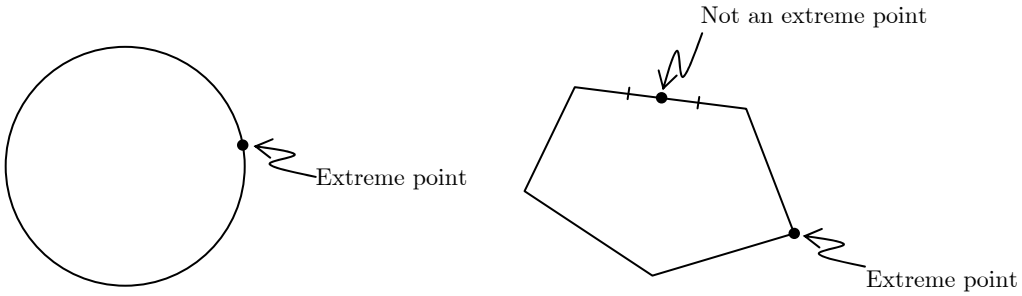


  - ***Definition (Cone)***: A set $\mathcal{C} \in \mathbb{R}^n$ is a *cone* if for all $\boldsymbol{x} \in \mathcal{C}$ and any $\lambda \geq 0$, $\lambda \boldsymbol{x} \in \mathcal{C}$:

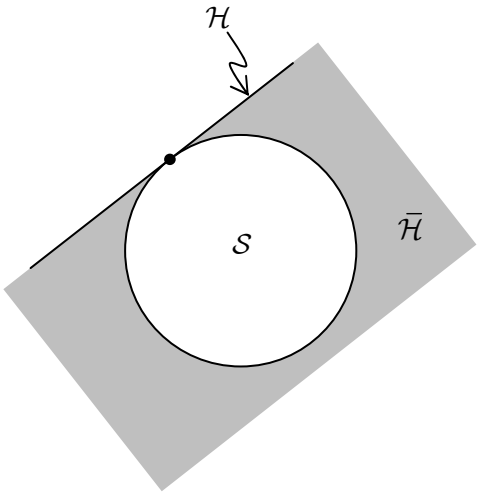


The set $\left\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} = A\boldsymbol{\alpha}, \boldsymbol{\alpha} \geq 0, A \in \mathbb{R}^{n \times m}, \boldsymbol{\alpha} \in \mathbb{R}^m\right\}$ is the *cone generated by the columns of A*.

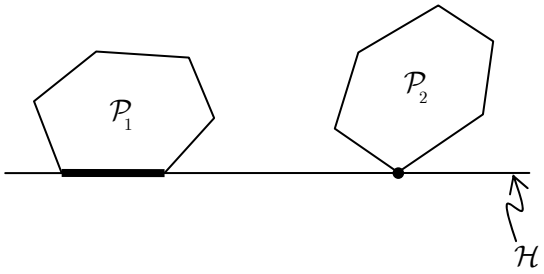
- ***Definition (extreme point)***: An *extreme point* of the convex set $\mathcal{C}$ is a point $\boldsymbol{x} \in \mathcal{C}$ that cannot be written as a convex combination of other points in $\mathcal{C}$.



- ***Definition (Convex Combination)***: A *convex combination* of points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_k$ is a point $\boldsymbol{x} = \sum_{i=1}^{k} \lambda_i \boldsymbol{x}_i$, such that $\boldsymbol{\lambda} \geq 0$ and $\sum_{i=1}^{k} \lambda_i = 1$. The set of convex combinations of a set points is the smallest convex set containing all the points; it is called the *convex hull* of these points.
- ***Definition (Hyperplane)***: The set $\mathcal{H} = \left\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a} \cdot \boldsymbol{x} = \beta, \boldsymbol{a} \in \mathbb{R}^n, \beta \in \mathbb{R}\right\}$ is called a *hyperplane* with *normal* $\boldsymbol{a}$. The set $\bar{\mathcal{H}} = \left\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a} \cdot \boldsymbol{x} \leq \beta\right\}$ is a *closed halfspace*, and $\mathcal{H}$ is its *bounding hyperplane*.
- ***Definition (Afine set)***: A set $\mathcal{C}_a \in \mathbb{R}^n$ is an *affine set* if for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{C}_a$ and $\lambda \in (-\infty, \infty)$, $\boldsymbol{x}(\lambda) = \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \mathcal{C}_a$. A hyperplane is an example of an affine set. Roughly speaking, an affine set is a subspace that need not contain the original.
- ***Definition (Polyhedron)***: A *polyhedron* is a set which is the intersection of a finite number of closed hyperplanes. It is necessarily convex. If the polyhedron is non-empty and bounded (ie: there exists a large ball it lies inside of), it is called a *polytope*.
- ***Definition (Dimension)***: The *dimension* of an affine set $\mathcal{C}_a$ is the maximum number of linearly independent vectors in $\mathcal{C}_a$.
- ***Definition (Supporting hyperplane)***: A *supporting hyperplane* of a closed, convex set $\mathcal{C}$ is a hyperplane $\mathcal{H}$ such that $\mathcal{H} \cap \mathcal{C} \neq \varnothing$ and $\mathcal{C} \subseteq \bar{\mathcal{H}}$:



- ***Definition (Face)***: Let $\mathcal{P}$ be a non-empty polyhedron and let $\mathcal{H}$ be any supporting hyperplane. The intersection $\mathcal{P} \cap \mathcal{H} = \mathcal{F}$ is a *face* of $\mathcal{P}$. The whole polyhedron could be a face, if the $\mathcal{P}$ and $\mathcal{H}$ are both two-dimensional! For example, the thick line and the point in the example below are both faces of their respective polyhedra:



We give special names to faces of particular dimensions:

Face	***Dimension***
Vertex	0
Edge	1
Facet	$d-1$

Another way of looking at the concept of a vertex is as a point $\boldsymbol{x} \in \mathcal{P}$ that is such that there exists some $\boldsymbol{c}$ such that $\boldsymbol{c} \cdot \boldsymbol{x} < \boldsymbol{c} \cdot \boldsymbol{y}$ for all $\boldsymbol{y} \in \mathcal{P}$ and $\boldsymbol{y} \neq \boldsymbol{x}$. In other words, we insist $\mathcal{P} \cap \mathcal{H} = \mathcal{F} = \boldsymbol{x}$. This simply means that our face is 0-dimensional, and the definitions are therefore equivalent.

- ***Polyhedra in standard form***
  - The definition of a polyhedron above (in terms of the intersection of a number of half-spaces) can be written as $\mathcal{P} = \left\{ A\boldsymbol{x} \leq \boldsymbol{b} : A \in \mathbb{R}^{n \times m}, \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{b} \in \mathbb{R}^m \right\}$, where the rows of A contain the normals of the various hyperplanes defining the polyhedron.
  - It is often convenient, however, to write the polyhedron in an equivalent *standard form* $\mathcal{P}' = \left\{ A'\boldsymbol{x} = \boldsymbol{b} : \boldsymbol{x} \geq \boldsymbol{0}, A \in \mathbb{R}^{n' \times m}, \boldsymbol{x} \in \mathbb{R}^{n'}, \boldsymbol{b} \in \mathbb{R}^m \right\}$, where the rows of A are linearly independent. This involves a number of steps:
    - Re-write each inequality constraint $\boldsymbol{A}^{\text{row } i} \cdot \boldsymbol{x} \leq b_i$ as the equality $\boldsymbol{A}^{\text{row } i} \cdot \boldsymbol{x} + s_i = b_i$, where $s_i \geq 0$. s_i becomes a new variable.
    - Eliminate any linearly independent rows in A (this does not alter the problem – see page 57 of B&T for proof). Note that this implies that, in standard form, $m \leq n$ – in other words, the number of constraints is less than or equal to the number of variables.)
    - Replace any unconstrained variables x_i with two new variables x_i^+ and x_i^-, both constrained to be positive, and add the constraint $x_i = x_i^+ - x_i^-$. [The validity of this step is not entirely obvious, but for the simplex method, it works].
- ***Algebraic Characterization of Vertices & Extreme Points***
  - In the previous section, we provided definitions of *vertices* and *extreme points*. It would seem logical that the solution of a linear program should lie at one of these points. In this section, we see that these concepts are equivalent, and we develop an algebraic characterization of such points.
  - We present two characterizations – the first is in terms of polyhedra in non-standard form, which is more useful to gain an intuitive grasp of the concept, and the second in terms of polyhedra in standard form, which we will use hereafter.
  - ***Theorem***: Let $\mathcal{P} = \left\{ \boldsymbol{x} : A\boldsymbol{x} \geq \boldsymbol{b}, A'\boldsymbol{x} = \boldsymbol{b}' \right\}$ be a non-empty polyhedron, and let $\boldsymbol{x} \in \mathcal{P}$. The following three statements are equivalent:
    1. $\boldsymbol{x}$ is a vertex

2. $\boldsymbol{x}$ is an extreme point
3. All equality constraints are active at $\boldsymbol{x}$, some of the inequality constraints are active, and out of all the constraints that are active at $\boldsymbol{x}$, n of them are linearly independent [*Note*: we say the vectors $\boldsymbol{a}_i$ are linearly independent if the system of equations $\boldsymbol{a}_i \cdot \boldsymbol{x} = b_i$ has a unique solution (see p48 of B&T)].

Proof: See p50, B&T.

- ***Theorem***: Let $\mathcal{P} = \left\{\boldsymbol{x} \in \mathbb{R}^n : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\right\}$ be a non-empty polyhedron in standard form, and let $\boldsymbol{x} \in \mathcal{P}$. Then the following three statements are equivalent

1. $\boldsymbol{x}$ is an extreme point of $\mathcal{P}$.
2. The columns of A corresponding to the strictly positive components of $\boldsymbol{x}$ are linearly independent. More precisely, there exists a subset $\mathcal{B}$ of columns of A that are linearly independent such that $x_i = 0$ for all $i \notin \mathcal{B}$
3. $\boldsymbol{x}$ is a vertex of $\mathcal{P}$.

At first sight, statement (2) in this theorem seems somewhat different to statement (3) in the previous theorem, because here, we talk in terms of *columns* of A (variables), whereas in the previous theorem, we talked in terms of *rows* of A (constraints). In fact, the two are equivalent; we discuss two ways of seeing this; the first in terms of the rows of A, and the second in terms of the columns:

- The fact that all variables $\notin \mathcal{B}$ are equal to 0 already creates $n - |\mathcal{B}|$ linearly independent active constraints. We need all remaining active constraints to include at least $|\mathcal{B}|$ also be linearly independent constraints; in other words, we need $|\mathcal{B}|$ of the rows of the matrix B to be linearly independent. Another way of saying this is that we need all the columns of the matrix B (there are $|\mathcal{B}|$ of them) to be linearly independent.
- Another way of stating the constraint $A\boldsymbol{x} = \boldsymbol{b}$ is that we need to synthesize $\boldsymbol{b}$ from a non-negative linear combination of the columns of A; in the example of the diet problem, $\boldsymbol{b}$ is our requirement in nutrient-

space, and the columns of A describe the nutrient-content of each food. We have to combine the foods to get our requirements.

Now, if we find a group of columns that are linearly dependent, it means that they can be describes in terms of each other. In the diet problem, water, sugar, lemon and lemonade might form such a group, because it is possible, from 3 of these, to get all the nutrients available in the fourth item

In that case, we *must* "standardize" the problem by having one of these variables set to 0 – otherwise, our problem is indeterminate. In the example of the diet problem, we must choose to leave out one of {water, sugar, lemon, lemonade}, because there is an infinite combination of these that will give us any combination of nutrients.

Proof: We will prove $1 \Rightarrow 2 \Rightarrow 3 \Rightarrow 1$. We first define some general notation. We will be considering a point $\boldsymbol{x}$ in the polyhedron; we will define $\overline{\boldsymbol{x}}$ to be the strictly positive component of $\boldsymbol{x}$, and $\tilde{\boldsymbol{x}}$ to be those components that are 0. We will divide the columns of A correspondingly:

$$\boldsymbol{x} = \begin{pmatrix} \overline{\boldsymbol{x}} > 0 \\ \tilde{\boldsymbol{x}} = 0 \end{pmatrix} \qquad\qquad A = \left[\overline{A} \,\middle\vdots\, \tilde{A} \right]$$

When we talk of other vectors $\boldsymbol{y}$, $\overline{\boldsymbol{y}}$ will refer to those components of $\boldsymbol{y}$ corresponding to those in $\overline{\boldsymbol{x}}$, and similarly for $\tilde{\boldsymbol{y}}$.

- $\boxed{1 \Rightarrow 2}$ Assume 2 is false, and let $\boldsymbol{x}$ be an extreme point. Our assumption that 2 is false implies that there exists a $\overline{\boldsymbol{w}}$ such that $\overline{A}\overline{\boldsymbol{w}} = \boldsymbol{0}$. Thus, it is true that $\overline{A}\left(\overline{\boldsymbol{x}} + \varepsilon\overline{\boldsymbol{w}}\right) = \boldsymbol{b}$, for all ε. However, since $\overline{\boldsymbol{x}}$ is strictly positive, we can always find ε small enough, say $\overline{\varepsilon}_+$ and $\overline{\varepsilon}_-$, so that $\overline{\boldsymbol{x}} \pm \overline{\varepsilon}_\pm \overline{\boldsymbol{w}} \geq 0$.

Now, let $\boldsymbol{w} = \begin{pmatrix} \overline{\boldsymbol{w}}^\top & 0 \end{pmatrix}^\top$ and set $\boldsymbol{x}' = \boldsymbol{x} + \overline{\varepsilon}_+ \boldsymbol{w}$, and $\boldsymbol{x}'' = \boldsymbol{x} - \overline{\varepsilon}_- \boldsymbol{w}$. We have that $\boldsymbol{x}', \boldsymbol{x}'' \in \mathcal{P}$. However, $\boldsymbol{x} = \frac{1}{2}\boldsymbol{x}' + \frac{1}{2}\boldsymbol{x}''$. This means that $\boldsymbol{x}$ is not an extreme point.

Thus, by the contra-positive, $1 \Rightarrow 2$.

- $\boxed{2 \Rightarrow 3}$ Choose a point $\boldsymbol{x} \in \mathcal{P}$ that satisfies 2. We would like to show that there exists a supporting hyperplane $\mathcal{H}$ to the polygon $\mathcal{P}$ such that $\mathcal{P} \cap \mathcal{H} = \boldsymbol{x}$.

  We postulate that the hyperplane $\mathcal{H} = \left\{ \boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{c} \cdot \boldsymbol{x} = 0 \right\}$ satisfies this requirement, where

  $$\boldsymbol{c} = \begin{pmatrix} \overline{\boldsymbol{c}} \\ \tilde{\boldsymbol{c}} \end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{e} \end{pmatrix} \qquad \boldsymbol{e} = \text{vector of 1's}$$

  Now, let's do the proof step-by-step:

  - $\boxed{\boldsymbol{x} \in \mathcal{H}}$ We have

    $$\boldsymbol{c} \cdot \boldsymbol{x} = \begin{pmatrix} 0 \\ \boldsymbol{e} \end{pmatrix} \cdot \begin{pmatrix} \overline{\boldsymbol{x}} \\ \tilde{\boldsymbol{x}} \end{pmatrix} = \tilde{\boldsymbol{c}} \cdot \tilde{\boldsymbol{x}} = 0$$

  - $\boxed{\mathcal{P} \subseteq \overline{\mathcal{H}}}$ Consider any $\boldsymbol{y} \in \mathcal{P} \setminus \left\{ \boldsymbol{x} \right\}$. We have

    $$\boldsymbol{c} \cdot \boldsymbol{x} = \begin{pmatrix} 0 \\ \boldsymbol{e} \end{pmatrix} \cdot \begin{pmatrix} \overline{\boldsymbol{y}} \\ \tilde{\boldsymbol{y}} \end{pmatrix} = \boldsymbol{e} \cdot \tilde{\boldsymbol{y}}$$

    But since $\boldsymbol{y} \in \mathcal{P}$, we must have $\tilde{\boldsymbol{y}} \geq \boldsymbol{0}$. Thus, $\boldsymbol{c} \cdot \boldsymbol{x} \geq 0$.

  - $\boxed{\mathcal{P} \cap \mathcal{H} = \boldsymbol{x}}$ Imagine $\boldsymbol{y} \in \mathcal{H} \cap \mathcal{P}$. This means that $\boldsymbol{y} \cdot \boldsymbol{c} = 0$, but since all the components of $\boldsymbol{y}$ are positive (since it in the polyhedron), the only way this can happen is if $\tilde{\boldsymbol{y}} = \boldsymbol{0}$. Furthermore, since $\boldsymbol{y}$ is in the polyhedron, $A\boldsymbol{y} = \boldsymbol{b}$. Thus

    $$A\boldsymbol{x} - A\boldsymbol{y} = \boldsymbol{0}$$
    $$\left(\overline{A}\overline{\boldsymbol{x}} - \overline{A}\overline{\boldsymbol{y}} \right) + \left(\tilde{A}\tilde{\boldsymbol{x}} - \tilde{A}\tilde{\boldsymbol{y}} \right) = \boldsymbol{0}$$

    We have already established that $\tilde{\boldsymbol{y}} = \boldsymbol{0}$, and by definition, $\tilde{\boldsymbol{x}} = \boldsymbol{0}$

    $$\overline{A}\left(\overline{\boldsymbol{x}} - \overline{\boldsymbol{y}} \right) = \boldsymbol{0}$$

By assumption (2), however, this can only happen if $\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}} = \boldsymbol{0} \Rightarrow \bar{\boldsymbol{x}} = \bar{\boldsymbol{y}} \Rightarrow \boldsymbol{x} = \boldsymbol{y}$.

- $\boxed{3 \Rightarrow 1}$ Choose a point $\boldsymbol{x} \in \mathcal{P}$ that satisfies (3), and assume that (1) is not true; in other words, for some $\lambda \in [0,1]$ and $\boldsymbol{x}', \boldsymbol{x}'' \in \mathcal{P}$, we can write $\boldsymbol{x} = \lambda \boldsymbol{x}' + (1-\lambda)\boldsymbol{x}''$.

  Our assumption that $\boldsymbol{x}$ satisfies (3) implies that there exists a vector $\boldsymbol{c}$ such that $\boldsymbol{c} \cdot \boldsymbol{x} < \boldsymbol{c} \cdot \boldsymbol{x}'$ and $\boldsymbol{c} \cdot \boldsymbol{x} < \boldsymbol{c} \cdot \boldsymbol{x}''$. Now:

$$\begin{aligned} \boldsymbol{c} \cdot \boldsymbol{x} &= \boldsymbol{c} \cdot \left[\lambda \boldsymbol{x}' + (1-\lambda)\boldsymbol{x}''\right] \\ &= \lambda \boldsymbol{c} \cdot \boldsymbol{x}' + (1-\lambda)\boldsymbol{x}'' \\ &> \lambda \boldsymbol{c} \cdot \boldsymbol{x} + (1-\lambda)\boldsymbol{c} \cdot \boldsymbol{x} \\ &> \boldsymbol{c} \cdot \boldsymbol{x} \end{aligned}$$

  This is a contradiction. $\boldsymbol{x}$ cannot be on the line between $\boldsymbol{x}'$ and $\boldsymbol{x}''$ and also "below" both of them.

- Note that the theorem above says nothing of *how many* variables the set $\mathcal{B}$ must contain. The case $|\mathcal{B}| = \text{rank } A = m$, however, is a natural choice, because the constraint $A\boldsymbol{x} = \boldsymbol{b}$ already includes m constraints, and
  - Choosing $|\mathcal{B}| > m$ is impossible, since A contains only m rows.
  - Choosing $|\mathcal{B}| < m$ would imply choosing more than $n - m$ non-negativity constraints, which, in total, would result in more than n constraints. The resulting system would be over-defined, and might not have a solution.

  We therefore define…

- ***Definition (Basis)***: A linearly independent set of m columns $\left\{\boldsymbol{A}^{\text{col } B_1}, \cdots, \boldsymbol{A}^{\text{col } B_m}\right\}$ of A is a *basis* for the column space of A. [Note: if A contains no linearly independent rows, then rank $A = m$, and our definition boils down to the fact a basis is a *maximally linearly independent* set of m columns].

  $B = \left[\boldsymbol{A}^{\text{col } B_1}, \cdots, \boldsymbol{A}^{\text{col } B_m}\right]$ is called the *basis matrix* and the associated vector of variables $\boldsymbol{x}_B$ that solves $B\boldsymbol{x}_B = \boldsymbol{b}$ is called the vector of *basic variables*. Other variables (and columns of A) are called *non-basic*:

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_B \\ \boldsymbol{x}_N \end{pmatrix} \begin{matrix} m \text{ of them} \\ n-m \text{ of them} \end{matrix} \qquad A = \left[B \,\middle\vdots\, N \right]$$

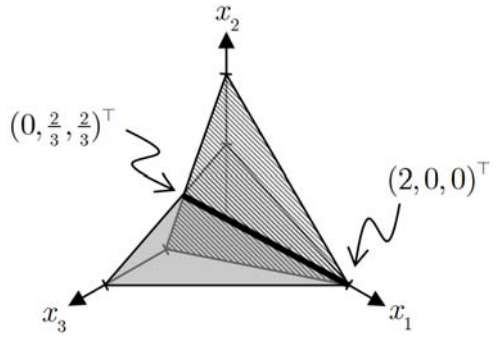
$\boldsymbol{x}$ is called a *basic solution*. There is no guarantee, however, that solving $B\boldsymbol{x}_B = \boldsymbol{b}$ will lead to $\boldsymbol{x} \geq \boldsymbol{0}$. If it does, the solution is also called a *basic feasible solution*.

- ***Definition (adjacent basis)***: Two distinct basic solutions are said to be *adjacent* if we can find $n - 1$ linearly independent constraints that are active at both of them. If two adjacent basic solutions are also feasible, then the line segment that joins them is called an *edge* of the feasible set. In terms of polyhedra in standard form, two bases are said to be adjacent if they share all but one basic columns.

- ***Degeneracy***
  - ***Definition (Degenerate vertex)***: A vertex $\boldsymbol{x}$ is said to be *degenerate* when more than n of the constrains are active at $\boldsymbol{x}$.
  - ***Definition (Degenerate basis)***: A basic feasible solution $\boldsymbol{x}$ is said to be *degenerate* if some component of $\boldsymbol{x}_B$ is 0. Otherwise, it is called non-degenerate. This is equivalent to the previous definition, because it implies that more than $n-m$ of the non-negativity constraints are tight at $\boldsymbol{x}$.
  - For an example of degeneracy in a standard-form problem, consider the following problem:

$$\left.\begin{matrix} x_1 + 2x_2 + x_2 = 2 \\ x_1 + x_2 + 2x_3 = 2 \end{matrix}\right\} \Rightarrow \begin{pmatrix} 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

In this case, the polyhedron is simply the thick line in the figure below. Clearly, it only contains two vertices:



The vertex $\boldsymbol{x} = (0, \frac{2}{3}, \frac{2}{3})^\top$ is at the intersection of 3 planes; since there are three variables, it is non-degenerate. This corresponds to the basis matrix

$$B = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Clearly, each entry of $\boldsymbol{x}$ corresponding to a column in the basis is non-zero.

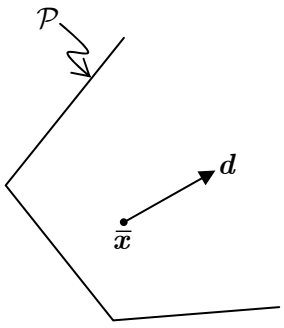
The vertex $\boldsymbol{x} = (2,0,0)^\top$, however, is at the intersection of 4 planes. It is degenerate, and corresponds to two *different* bases

$$B = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \qquad\qquad B = \begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}$$

In each case, one of the variables corresponding to a basic column is 0.

Representation & Optimality

- In this section, we prove what might be called the "fundamental Theorem of Linear Programming" – that the optimal solution of a linear program occurs at a vertex.
- ***The Representation Theorem***
  - Before proving our fundamental theorem, we prove that polyhedra can be represented in a very useful form.
  - ***Definition (recession direction)***: A *recession direction* of the polyhedron $\mathcal{P}$ is a non-zero vector $\boldsymbol{d} \in \mathbb{R}^n$ such that, for any $\bar{\boldsymbol{x}} \in \mathcal{P}$, $\left\{\boldsymbol{x} : \boldsymbol{x} = \bar{\boldsymbol{x}} + \theta \boldsymbol{d}, \theta \in \mathbb{R}_+\right\} \in \mathcal{P}$



For a polyhedron in standard form, $\boldsymbol{d}$ is a recessive direction if and only if $A\boldsymbol{d} = 0$ (so that we remain feasible as we move along that direction), $\boldsymbol{d} \geq \boldsymbol{0}$ (so that we never become negative as we move along that direction) and $\boldsymbol{d} \neq 0$.

- ***Theorem (Representation)***: Any point $\boldsymbol{x} \in \mathcal{P} \equiv \left\{\boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0\right\}$ can be written as

$$\boldsymbol{x} = \sum_{i \in V} \lambda_i \boldsymbol{v}^i + \alpha \boldsymbol{d} \qquad \lambda_+ = 1, \lambda_i \geq 0, \alpha \geq 0$$

where $\left\{\boldsymbol{v}^i : i \in V\right\}$ is the set of vertices of the polyhedron and $\boldsymbol{d}$ is a recession direction.

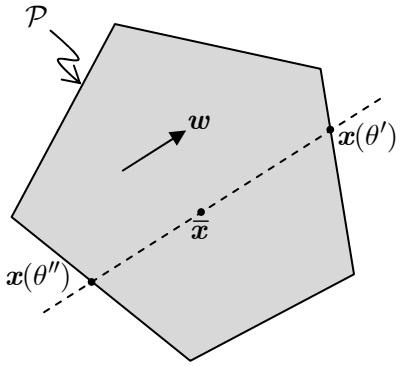
- ***Proof***: We prove this by induction, on the number of strictly positive components in $\boldsymbol{x}$.

  Suppose the theorem holds true if $\boldsymbol{x}$ has $p-1$ strictly positive components, and consider an $\boldsymbol{x}$ with p strictly positive components. If $\boldsymbol{x}$ is a vertex, the theorem is trivially true. If $\boldsymbol{x}$ is not a vertex, the columns of A corresponding to the positive components of $\boldsymbol{x}$ are linearly dependent. This implies that there exists a $\boldsymbol{w} \neq \boldsymbol{0}$ such that $w_i = 0$ if $x_i = 0$ and $A\boldsymbol{w} = \boldsymbol{0}$.

  Now, consider points of the form $\boldsymbol{x}(\theta) = \boldsymbol{x} + \theta \boldsymbol{w}$. Clearly, $A\boldsymbol{x}(\theta) = \boldsymbol{b}$ for all θ. As we move along $\boldsymbol{w}$, we will either hit a non-negativity constraint, or go on forever (if $\boldsymbol{w}$ is a recession direction). We consider these two cases:

  - $\boxed{\boldsymbol{w} \text{ has both +ve and } - \text{ve components}}$ In that case, we'll hit a non-negativity constraint. Let
    - $\theta' =$ smallest positive θ such that $\boldsymbol{x}(\theta')$ has at most $p-1$ strictly positive components.
    - $\theta'' =$ largest negative θ such that $\boldsymbol{x}(\theta'')$ has at most $p-1$ strictly positive components.

Diagrammatically:

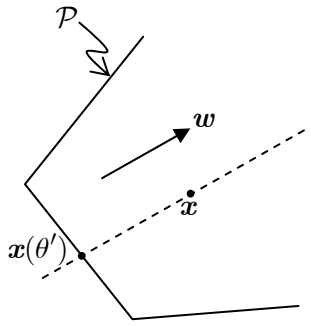


We can then write

$$\boldsymbol{x} = \mu \boldsymbol{x}(\theta') + (1-\mu)\boldsymbol{x}(\theta'') \qquad \mu = \frac{\theta''}{\theta'' - \theta'}$$

But by the inductive hypothesis, the points $\boldsymbol{x}(\theta')$ and $\boldsymbol{x}(\theta'')$ can be written as in the statement of the Theorem. Thus, so can $\boldsymbol{x}$.

- $\boxed{\boldsymbol{w} \geq \boldsymbol{0} \text{ or } \boldsymbol{w} \leq \boldsymbol{0}}$ In that case, we'll hit a non-negativity constraint going in one direction, but we'll go on forever in the other direction. Assume that $\boldsymbol{w} \geq \boldsymbol{0}$ [the other case is similar]. Let θ' be the largest negative θ such that $\boldsymbol{x}(\theta')$ has at most $p - 1$ strictly positive components. Diagrammatically:



We can then write

$$\boldsymbol{x} = \boldsymbol{x}(\theta') + \theta \boldsymbol{w}$$

But $\boldsymbol{w}$ is a recession direction (because $A\boldsymbol{w} = \boldsymbol{0}$ and $\boldsymbol{w} \geq 0$) and by the inductive hypothesis, $\boldsymbol{x}(\theta')$ can be written as required. Thus, so can $\boldsymbol{x}$.

The "starting point" for the induction is somewhat difficult to find. It is tempting to use $\boldsymbol{x} = \boldsymbol{0}$, but this point might not even be in the polyhedron! It

turns out that an appropriate starting point is simply the point in the polyhedron with the smallest number of strictly positive components. This *must* be a vertex, because if it was not, we could carry out the steps outlined above and find a point with fewer strictly positive components – this is a contradiction.

[Note that it is *not* always the case a polyhedron must have vertices – for example, the polyhedron $\left\{\boldsymbol{x} \in \mathbb{R}^2 : x_1 \geq 0, x_1 \leq 1\right\}$ has no vertices. However, the non-negativity constrains of the standard-form polyhedron ensure there is at least one.

- ***The Fundamental Theorem***
  - ***Theorem (Fundamental Theorem of Linear Programming)***: If $\mathcal{P} \neq \varnothing$, then the minimum $\min_{\boldsymbol{x}\in\mathcal{P}} \boldsymbol{c}\cdot\boldsymbol{x}$ is either attained at a vertex of $\mathcal{P}$ or unbounded.

    Proof: We consider two cases:

    - ***Case 1 – $\mathcal{P}$ has a recession direction $\boldsymbol{d}$ such that $\boldsymbol{c}\cdot\boldsymbol{d} < 0$***: in that case, the problem is unbounded, because for any $\bar{\boldsymbol{x}} \in \mathcal{P}$, $\boldsymbol{c}\cdot\boldsymbol{x}(\theta) = \boldsymbol{c}\cdot\left(\bar{\boldsymbol{x}} + \theta\boldsymbol{d}\right) = \boldsymbol{c}\cdot\bar{\boldsymbol{x}} + \theta\boldsymbol{c}\cdot\boldsymbol{d} \downarrow -\infty$ as $\theta \to \infty$.
    - ***Case 2 – $\mathcal{P}$ has no such recession direction***: in that case, consider any point $\bar{\boldsymbol{x}} \in \mathcal{P}$. By our Representation Theorem, we can write $\bar{\boldsymbol{x}} = \sum \lambda_i \boldsymbol{v}^i + \alpha\boldsymbol{d}$, where $\lambda_+ = 1, \lambda_i \geq 0, \alpha \geq 0$. We then have

$$\begin{aligned}\boldsymbol{c}\cdot\bar{\boldsymbol{x}} &= \sum \lambda_i \boldsymbol{c}\cdot\boldsymbol{v}^i + \alpha\left(\boldsymbol{c}\cdot\boldsymbol{d}\right)\\ &\leq \sum \lambda_i \boldsymbol{c}\cdot\boldsymbol{v}^i\\ &\leq \sum \lambda_i \min_{i\in v}\left(\boldsymbol{c}\cdot\boldsymbol{v}^i\right)\\ &= \min_{i\in v}\left(\boldsymbol{c}\cdot\boldsymbol{v}^i\right)\end{aligned}$$

    Thus, the minimum is indeed attained at a vertex.

Simplex

- We have thus far established that the optimum of a linear program occurs at one of the vertices of the feasible region. We now consider the *simplex algorithm*, an efficient method of jumping for vertex to vertex while constantly improving the objective function.

- ***Representation in terms of emanating directions***
  - Consider a polyhedron $A\boldsymbol{x} = \boldsymbol{b}$, where $A = \left[B, N\right] \in \mathbb{R}^{m \times n}$, and a non-degenerate basic solution $\hat{\boldsymbol{x}} = \left(\hat{\boldsymbol{x}}_B^\top, \hat{\boldsymbol{x}}_N^\top\right)^\top$, where $\hat{\boldsymbol{x}}_B = B^{-1}\boldsymbol{b} > \boldsymbol{0}$ and $\hat{\boldsymbol{x}}_N = \boldsymbol{0}$.
  - ***Claim***: Consider the matrix

$$M = \begin{pmatrix} B & N \\ 0 & I \end{pmatrix} \qquad\qquad M\hat{\boldsymbol{x}} = \begin{pmatrix} B & N \\ 0 & I \end{pmatrix}\begin{pmatrix} \hat{\boldsymbol{x}}_B \\ \hat{\boldsymbol{x}}_N \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ 0 \end{pmatrix}$$

The last $n - m$ columns of M^{-1} (ie: from column $m + 1$ onwards) are the directions of the edges of P emanating from the basic feasible solution $\hat{\boldsymbol{x}}$.

Proof: Let $\boldsymbol{\eta}^j$ be the j^{th} column of M^{-1}. Using the fact that since there is no degeneracy, $\boldsymbol{x}_B$ has M nonzero components, and so the row is clearly from the second half of the matrix above, we can write, for $j > m$:

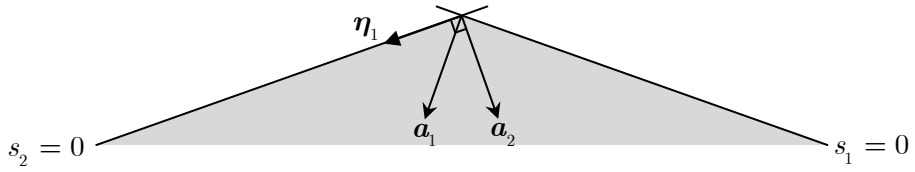
$$\boldsymbol{\eta}^j = M^{-1}e_j = \begin{pmatrix} B^{-1} & -B^{-1}N \\ 0 & I \end{pmatrix} e_j = \begin{pmatrix} -B^{-1}N \\ I \end{pmatrix} e_{j-b} = \begin{pmatrix} -B^{-1}\boldsymbol{A}^{\text{col } j} \\ \vdots \\ 1 \\ \vdots \end{pmatrix} \leftarrow j^{\text{th row}}$$

Now, consider moving in the direction $\boldsymbol{\eta}^j$ by an amount θ; $\boldsymbol{x}(\theta) = \hat{\boldsymbol{x}} + \theta\boldsymbol{\eta}^j$. This point is still on the polyhedron, because

$$\begin{aligned} A\boldsymbol{x}_B(\theta) &= B\boldsymbol{x}_B(\theta) + N\boldsymbol{x}_N(\theta) \\ &= B\left(\hat{\boldsymbol{x}} - \theta B^{-1}\boldsymbol{A}^{\text{col } j}\right) + \theta\boldsymbol{A}^{\text{col } j} \\ &= \boldsymbol{b} - \theta\boldsymbol{A}^{\text{col } j} + \theta\boldsymbol{A}^{\text{col } j} = \boldsymbol{b} \end{aligned}$$

Thus, η_j is indeed an edge of P, and it clearly results from increasing only *one* of the $\boldsymbol{x}_N$.

For a geometric interpretation, consider that the rows of M contain the vectors normal to every active constraint at the BFS, and that $MM^{-1} = I \Rightarrow \boldsymbol{m}_{\text{row } i}\boldsymbol{M}^{-1}_{\text{col } j} = 0 \;\forall i \neq j$. This means that our emanating edges (columns of M^{-1}) are perpendicular to every normal vector save one (along which we're trying to move):



For a final intuitive justification of this result, note that for a problem in standard form, each emanating edge $\boldsymbol{\eta}^j$ involves bringing one variable – say x_j – into the basis and consequently changing the value of the other variables in the basis in order to keep $A\boldsymbol{x} = \boldsymbol{b}$ valid. Now, we established above that the constraint $A\boldsymbol{x} = \boldsymbol{b}$ can be viewed as taking a linear combination of the columns of A to "synthesize" $\boldsymbol{b}$. Now, note that

- If we increase x_j by 1 unit, our "resultant vector" $A\boldsymbol{x}$ increases by $\boldsymbol{A}^{\text{col } j}$.
- If we change the basic variables by $\boldsymbol{\eta}_B^j$, our "resultant vector" $A\boldsymbol{x}$ increases by $B\boldsymbol{\eta}_B^j$.

Now, it makes sense that these two changes should "balance out" so as to keep our "resultant vector" at $\boldsymbol{b}$. Thus, $B\boldsymbol{\eta}_B^j = -\boldsymbol{A}^{\text{col } j}$.

- ***Lemma***: Given a BFS $\hat{\boldsymbol{x}}$, every point $\boldsymbol{y} \in P$ can be expressed as

$$\boldsymbol{y} = \hat{\boldsymbol{x}} + \sum_{j>m} y_j \boldsymbol{\eta}^j$$

Where $\boldsymbol{\eta}^j$ is the j^{th} column of M^{-1}, and where $y_j \geq 0$ for $j > M$.

Proof: Consider that

$$M\left(\boldsymbol{y} - \hat{\boldsymbol{x}}\right) = \begin{pmatrix} B & N \\ 0 & I \end{pmatrix}(\boldsymbol{y} - \hat{\boldsymbol{x}}) = \begin{pmatrix} A\boldsymbol{y} - A\hat{\boldsymbol{x}} \\ \boldsymbol{y}_N - \boldsymbol{0} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{y}_N \end{pmatrix}$$

And so

$$\boldsymbol{y} = \hat{\boldsymbol{x}} + M^{-1}\begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{y}_N \end{pmatrix}$$

- ***Corollary***: If $\hat{\boldsymbol{x}}$ is a BFS, then any point in P is contained in the polyhedral cone generated by the last $n - m$ columns of M^{-1}

$$P \subset C = \left\{\boldsymbol{y} \mid \boldsymbol{y} = \hat{\boldsymbol{x}} + \sum_{j=M+1}^{n} \alpha_j \boldsymbol{\eta}^j, \alpha_j \geq 0 \ \forall j > m\right\}$$

Claim: We claim that

1. $\boldsymbol{\eta}^j$ are the "extreme" recession directions of C.
2. $\boldsymbol{\eta}^j$ are the "edges" (1-dimensional faces) of C. [*Hint*: let $\boldsymbol{c} = \boldsymbol{\sigma}^\top M^{-1}$ and $\boldsymbol{\sigma} = \boldsymbol{e} - \boldsymbol{e}_j$].
3. If $\hat{\boldsymbol{x}}$ is a non-degenerate BFS, then $\boldsymbol{\eta}^j$ is an edge of P.

- ***Background to the Simplex Algorithm***
  - Consider linear program is $\min \boldsymbol{z} = \boldsymbol{c}^\top \boldsymbol{x}$. The directional derivative of $\boldsymbol{z}$ with respect to $\boldsymbol{x}$ in the direction $\boldsymbol{\eta}^j$ is $\boldsymbol{c}^\top \boldsymbol{\eta}^j$. If it is greater than 0, the direction is "uphill", and vice-versa.
  - We call $\overline{c}_j = \boldsymbol{c} \cdot \boldsymbol{\eta}^j = -\boldsymbol{c}_B^\top B^{-1} \boldsymbol{A}^{\text{col } j} + c_j$ the *reduced cost* of direction j. Practically, they can be calculated by
    - Solving $\boldsymbol{\pi}^\top = \boldsymbol{c}_B^\top B^{-1}$
    - Setting, for $j > m$, $\overline{c}_j = c_j - \boldsymbol{\pi} \cdot \boldsymbol{A}^{\text{col } j}$

    Geometrically:
    - $\boldsymbol{\pi}$ is the particular linear combination of equality constraints that gives $\boldsymbol{c}_B$.
    - Each component $\boldsymbol{A}^{\text{col } j}$ is the amount by which a unit change in x_j will "affect" each constraint.
    - Thus, $\boldsymbol{\pi} \cdot \boldsymbol{A}^{\text{col } j}$ is the resulting change in the objective when we move one unit along direction j.
    - Similarly, c_j is the direct change resulting in a unit change of x_j.
  - ***Theorem***: If $\overline{c}_j = \boldsymbol{c} \cdot \boldsymbol{\eta}^j \geq 0$ for all $j > m$, then the current BFS is optimal.

    Proof: Consider that any $\boldsymbol{y} \in P$ can be written

    $$\boldsymbol{y} = \hat{\boldsymbol{x}} + \sum_{j=m+1}^{n} y_j \boldsymbol{\eta}^j$$
    $$\boldsymbol{c} \cdot \boldsymbol{y} = \boldsymbol{c} \cdot \hat{\boldsymbol{x}} + \sum_{j=m+1}^{n} y_j \boldsymbol{c} \cdot \boldsymbol{\eta}^j = \boldsymbol{c} \cdot \hat{\boldsymbol{x}} + \sum_{j=m+1}^{n} y_j \overline{c}_j \geq \boldsymbol{c} \cdot \boldsymbol{x}$$

    Thus, the objective at any point is greater than at $\hat{\boldsymbol{x}}$.

    This theorem has an interesting geometrical explanation, which we discuss below, when we motivate duality.

- ***The Simplex Algorithm***

1. Start with a BFS $\boldsymbol{x}$.
2. Compute the simplex multipliers $\boldsymbol{\pi}$ by solving $B^\top \boldsymbol{\pi} = \boldsymbol{c}_B$, and compute the reduced costs $\overline{c}_j = c_j - \boldsymbol{\pi} \cdot \boldsymbol{A}^{\text{col } j}$ for all $j \notin \text{basis}$.
3. Check for optimality: if $\overline{\boldsymbol{c}}_j \geq 0 \ \forall j \notin \text{basis}$, then the current solution is optimal.
4. Choose a nonbasic variable to enter the basis; ie: choose a "downhill edge" from the set of downhill edges V along which to move (typically, the one with the smallest reduced cost):
$$q \in V = \left\{ j \notin B : c_j < 0 \right\}$$
5. Compute $\boldsymbol{\omega}^q$ for all q by solving $B\boldsymbol{\omega}^q = \boldsymbol{A}^{\text{col } q}$. Note that $\boldsymbol{\omega}^q = -\boldsymbol{\eta}^j$. If $\boldsymbol{\omega}^q \leq 0$, stop: $\boldsymbol{z} \downarrow -\infty$ along $\boldsymbol{\eta}^q$.
6. Otherwise, compute $\theta = \min_{1 \leq i \leq m} \left\{ \frac{x_j}{\omega_j} : \omega_i > 0 \right\}$ (to find the basic variable that should leave the basis).
7. Update the solution and the basis matrix B. Set:
$$x_q \leftarrow \theta$$
$$x_{j_i} \leftarrow x_{j_i} - \theta \omega_i$$

- ***The Full Tableau Simplex***
  - The simplex algorithm outlined above is relatively inefficient, because it involves the inversion of the matrix B at each step. We therefore use a different form of the algorithm which constantly maintains and updates the matrix $B^{-1}\left[A \mid \boldsymbol{b}\right]$. Typically, this information is stored in a *tableau* containing an extra row:

$B^{-1}A$	$B^{-1}\boldsymbol{b}$
$\boldsymbol{c}^\top - \boldsymbol{c}_B^\top B^{-1}A$	$-\boldsymbol{c}_B^\top B^{-1}\boldsymbol{b}$

Or, in more detail:

$\vdots$		$\vdots$	$x_{B,1}$
$B^{-1}\boldsymbol{A}^{\text{col } 1}$	$\cdots$	$B^{-1}\boldsymbol{A}^{\text{col } n}$	$\vdots$
$\vdots$		$\vdots$	$x_{B,m}$
$\overline{c}_1$	$\cdots$	$\overline{c}_n$	$-\boldsymbol{c}_B^\top \boldsymbol{x}_B$

[Note that the columns of the tableau corresponding to basic variables will contain the columns of the identity, because $B^{-1}B = I$]. We describe the operations involved in simplex in terms of the matrix a:

(a_{ij})	$\boldsymbol{x}_B$
$\bar{\boldsymbol{c}}^{\top}$	$-z$

- ***Set up the tableau***: The first step is to find a basic feasible solution; typically, this can be done by setting all slack variables to b and all the "real" variables to 0 (see later for cases where this doesn't work). In that case, $B = B^{-1} = I$ (where some of the entries might be negative, depending on the type of inequality), and the tableau can easily be filled in:

	x_1	x_2	…	z_m	$\boldsymbol{b}$
z_1 basic	a_{11}				b_1
⋮					⋮
z_m basic				a_{mn}	b_m
	c_1	c_2	⋯	0	0

The last row contains the reduced costs; in this case, since the objective function does not contain any slack variables, $\boldsymbol{c}_B = \boldsymbol{0}$, and so the reduced costs are simply equal to the costs for the nonbasic variables.

In the case of a more complex basic feasible solution, it is still an easy matter to work out the reduced costs:

- The matrix a contains $B^{-1}A$. Simply multiply each rows by the corresponding objective function coefficient (for example, multiply the first row above by c_{z_1})…
- …and add all the rows to get to get $\boldsymbol{c}_B^{\top}B^{-1}A$. Subtract this from $\boldsymbol{c}$ to get $\bar{\boldsymbol{c}} = \boldsymbol{c} - \boldsymbol{c}_B^{\top}B^{-1}A$.

In terms of our matrix:

$$\bar{c}_j = c_j - \sum_i a_{ij}c_i$$

✓ If $\bar{\boldsymbol{c}} > \boldsymbol{0}$, then there is no improving direction; we're done!

- ***Find pivot column***: Choose the *pivot column* with the smallest reduced cost; or, in the case of ties, the one with the smallest j. This variable will *enter the basis*:

$$j = \text{Entering basis} = \text{Pivot column} = \underset{j}{\operatorname{argmin}} \left\{ \bar{c}_j : \bar{c}_j < 0 \right\}$$

- ***Find pivot row***: Now, consider that the pivot column contains $\boldsymbol{a}^{\text{col } j} = B^{-1}\boldsymbol{A}^{\text{col } j}$. Very conveniently this is none other than the negative of the emanating direction corresponding to j from our BFS, $-\boldsymbol{\eta}^j$.
  - ✓ If every item in the pivot column, $\boldsymbol{a}^{\text{col } j}$ is negative, then every component of $\boldsymbol{\eta}^j$ is positive – we can move along this direction without ever becoming infeasible. The problem is unbounded.

  Assuming the problem is bounded, find θ, the maximum amount we can move in direction $\boldsymbol{\eta}^j$ before the problem becomes infeasible, and i, the variable that leaves the basis when this happens:

$$\theta = \min \left\{ \frac{x_i}{\eta_i^j} : \eta_i^j < 0, i \in B \right\} \qquad i = \underset{}{\operatorname{argmin}} \left\{ \frac{x_i}{\eta_i^j} : \eta_i^j < 0, i \in B \right\}$$

  In terms of our j^{th} column, this looks like

$$i = \text{Leaving basis} = \text{Pivot row} = \underset{i}{\operatorname{argmin}} \left\{ \frac{x_i}{a_{ij}} : a_{ij} > 0 \right\}$$

$$\theta = \min_i \left\{ \frac{x_i}{a_{ij}} : a_{ij} > 0 \right\}$$

  - ✓ If the minimum above is attained at two values of i, the entering basis is degenerate. See the discussion below for anti-cycling rules.

- ***Pivot***: We now pivot on the element a_{ij}. Because of the structure of our tableau, the only thing that needs to change is the vector $\boldsymbol{c}_B$ and the matrix B, which needs to change from B to $\bar{B}$, where

$$B = \begin{bmatrix} A^{\text{col } ?} & \cdots & \boldsymbol{A}^{\text{col } i} & \cdots & A^{\text{col } ?} \end{bmatrix}$$
$$\bar{B} = \begin{bmatrix} A^{\text{col } ?} & \cdots & \boldsymbol{A}^{\text{col } j} & \cdots & A^{\text{col } ?} \end{bmatrix}$$

  To work out how to update the tableau, consider that

$$B^{-1}\overline{B} = \left[I^{\text{col }?} \quad \cdots \quad B^{-1}\boldsymbol{A}^{\text{col } j} \quad \cdots \quad I^{\text{col }?} \right]$$

Now, imagine we found a matrix Q such that

$$QB^{-1}\overline{B} = Q\left[I^{\text{col }?} \quad \cdots \quad B^{-1}\boldsymbol{A}^{\text{col } j} \quad \cdots \quad I^{\text{col }?} \right] = I$$

Then we would have

$$QB^{-1} = \overline{B}^{-1}$$

So once we find the mysterious matrix Q, all we need to do is apply it to our tableau to update it.

Instead of thinking of Q as a matrix, it is helpful to think about it in terms of row operation – all we need is the series of row operations that will turn $B^{-1}\overline{B}$ into I and apply them to our tableau. These operations are:

- Divide the pivot row by the pivot element, to get a 1 in there.
- For each other row, subtract appropriate multiples of the pivot row to make every other element in the pivot column zero.

In terms of our tableau

$$\overline{a}_{\alpha\beta} = \begin{cases} a_{\alpha\beta} / a_{ij} & \text{for the pivot row (ie: } \alpha = i) \\ a_{\alpha\beta} - a_{\alpha j} a_{i\beta} & \text{for every other row (ie: } \alpha \neq i) \end{cases}$$

It turns out that the rule above also applies to the last row of the tableau. To see why, consider that originally, the last row consists of

$$\left[\boldsymbol{c}^{\top} \mid 0 \right] - \boldsymbol{c}_B^{\top} B^{-1} \left[A \mid \boldsymbol{b} \right]$$

Adding a multiple of the pivot row to this row involves adding a linear combination of $\left[A \mid \boldsymbol{b} \right]$, and so the result will be of the form

$$\left[\boldsymbol{c}^{\top} \mid 0 \right] - \boldsymbol{T} \left[A \mid \boldsymbol{b} \right]$$

But consider that after these row operations

- The last element of the pivot column contains a 0, by design.
- The last element of every other column that stays in the basis will also contain a 0, because

- The original value there was 0, being the reduced cost of a basic variable.
- The column corresponding to the basic variable k is $B^{-1}\boldsymbol{A}^{\text{col } k} = \boldsymbol{I}^{\text{col } k}$, and since the we are only considering variables that remain in the basis, $k \neq \text{pivot row}$; thus, the entry in the pivot row for that column is 0.

This implies that once the row operations have been carried out

$$\boldsymbol{c}_{\bar{B}}^{\top} - \boldsymbol{T}\bar{B} = 0 \Rightarrow \boldsymbol{T} = \boldsymbol{c}_{\bar{B}}^{\top}\bar{B}^{-1}$$

And so we end up with a bottom row of

$$\left[\boldsymbol{c}^{\top} \mid 0\right] - \boldsymbol{c}_{B}^{\top}\bar{B}^{-1}\left[A \mid \boldsymbol{b}\right]$$

As required.

- ***Finding an initial basic feasible solution***
  - In some cases, it is not so easy to find an original basic feasible solution. In this section, we explore two different methods.
  - ***Constructing an auxiliary program***
    - The first method is to solve an auxiliary linear program. If our original problem is $\min \boldsymbol{c}\cdot\boldsymbol{x}$ s.t. $A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}$, we construct an auxiliary program

$$\begin{array}{ll} \min & y_1 + \cdots + y_m \\ \text{s.t.} & A\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \\ & \boldsymbol{y} \geq \boldsymbol{0} \end{array}$$

Initialization of this problem is easy; we simply let $\boldsymbol{x} = \boldsymbol{0}$ and all the $\boldsymbol{y}$ be basic.

If $\boldsymbol{x}^*$ if a feasible solution to the original problem, then $(\boldsymbol{x}^*, \boldsymbol{0})$ is an optimal zero-cost solution to the auxiliary problem. We conclude that the original problem is feasible if and only if the auxiliary problem has 0 optimal cost; in which case $\boldsymbol{x}$ is a feasible solution of our original problem.

- If the auxiliary problem terminates with only original variables in the basis, everything is nice and dandy; we simply delete the columns corresponding to artificial variables, and go from there.
- It might be the case, however, that the problem ends with a zero-cost solution $(\boldsymbol{x}^*, \boldsymbol{0})$ and basis B that has one or more of the y_i in the basis (at 0 level).

  We note, however, that since A has rank m, it is theoretically possible to find m columns of A that are linearly independent and form a matrix $\bar{B}$. It is also clear that once we have found this new basis matrix, the *same* solution $\boldsymbol{x}^*$ is valid for that basis. Why? Because the only components of $\boldsymbol{x}^*$ that were non-zero previously are still in the basis, since $\mathcal{B} \subseteq \bar{\mathcal{B}}$, and the components we drive out of the basis are necessarily 0, since we only drive out y variables.
- It remains to discuss how we can determine, from the final tableau of the auxiliary problem, which variables to bring into the basis. Our technique will be as follows:
  - Choose one of the y to leave the basis – say it is the ℓ^{th} basic variable. This will be our leaving row.
  - Look for a non-zero entry of $B^{-1}A$ in that row – say entry j. We claim that $\boldsymbol{A}^{\text{col } j}$ is linearly independent of the other columns of A in the basis.

    To see why, consider that the columns of the basic variables form the identity matrix. Thus, since our row corresponds to an artificial variable, it is clear that every row corresponding to a "real" variable that is in the basis will have a 0 in that row. Thus, any column with a non-zero entry in that position, it is linearly independent.

- We now simply pivot, driving ℓ out of the basis, and bringing j^{th} in. (Note that the pivot element might be negative, unlike in the simplex method).
- [Note that our assumption A is full rank precludes that possibility that an entire row of $B^{-1}A$ is 0; thus, the method above always works. If such a row occurs, it can be eliminated].

- ***The Big-M Method***
  - An alternative method is to create a problem of the form above, but minimize a function of the form $\boldsymbol{c}\cdot\boldsymbol{x} + M\sum_i y_i$, where M is kept as a large undetermined parameter.
  - For a sufficiently large choice of M, if the original problem is feasible and its optimal cost is finite, all the artificial variables will eventually be driven to 0.

- ***Anti-cyling Rules***
  - If a vertex is degenerate, it is possible that $\theta^* = 0$ at that vertex. In other words, we move bases without changing vertices. Sometimes, this can happen many times before we leave a vertex, and in some cases, the process returns us to the basis we started with – in that case, the simplex algorithm *cycles*. This is clearly undesirable, and we now explore ways to avoid this phenomenon.
  - ***Definition***: $\boldsymbol{u}$ is *lexicographically larger* than $\boldsymbol{v}$ if the first nonzero component of $\boldsymbol{u} - \boldsymbol{v}$ is positive.
  - ***Lexicographic pivoting rule***:
    - Choose any variable to enter the basis, provided its reduced cost is negative.
    - For our entering variable, look at the rows which will need to be *reduced* by the operation (ie: rows for which the pivot column is positive). Divide each of these rows (including $\boldsymbol{x}$) by the pivot amount, and choose the lexicogrpahically smallest row to enter the basis.

- It can be shown (p 110 of B&T) that under this pivoting rule, the simplex algorithm will always terminate in a finite number of steps, provided that every row is lexicographically positive at the start of the algorithm (if this is not the base, we can simply re-arrange columns of $B^{-1}A$ to ensure that the first columns of the tableau form the identity matrix).

- ***Bland's Rule***
  - Bland's rule simply states that we should choose the row with the *smallest* value of i to enter the basis.
  - It is also known to result in an algorithm in which cycling never occurs.

Duality

- ***Motivation I***
  - ***Theorem (optimality Theorem)***: The basic non-degenerate feasible solution

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_B \\ \boldsymbol{x}_N \end{pmatrix} = \begin{pmatrix} B^{-1}\boldsymbol{b} \\ \boldsymbol{0} \end{pmatrix}$$

to the linear programming problem

$$\min \ \boldsymbol{c}\cdot\boldsymbol{x} \ \text{ s.t. } \ A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$$

is optimal if, and only if

$$\boldsymbol{c}^\top = \left(\boldsymbol{y}^\top, \boldsymbol{w}\right)\begin{pmatrix} \cdots & A & \cdots \\ 0 & & I \end{pmatrix} = \left(\boldsymbol{y}^\top, \boldsymbol{w}\right)\begin{pmatrix} B & N \\ 0 & I \end{pmatrix} = \left(\boldsymbol{y}^\top, \bar{\boldsymbol{w}}\right)M \qquad \bar{\boldsymbol{w}} \geq \boldsymbol{0}$$

[Note: A is the matrix of normals to equality constraints at $\boldsymbol{x}$, whereas $\begin{pmatrix} 0 & I \end{pmatrix}$ is the matrix of normals to inequality constraints tight at $\boldsymbol{x}$].

Proof: First note that by the definition of a basic solution, the rows of M are linearly independent, and so there is a vector that satisfies the equation above. [More deeply, this means that the rows of M form a basis for $\mathbb{R}^n$].

Now:

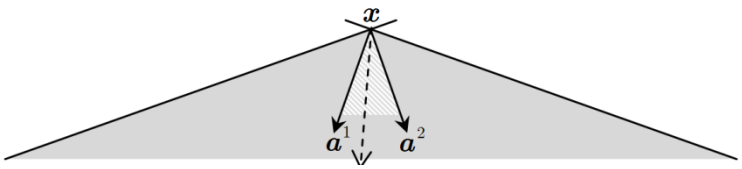
$$\begin{aligned}\left(\boldsymbol{y}^\top, \bar{\boldsymbol{w}}^\top\right) = \boldsymbol{c}^\top M^{-1} &= \left(\boldsymbol{c}_B^\top, \boldsymbol{c}_N^\top\right)\begin{pmatrix} B^{-1} & -B^{-1}N \\ 0 & I \end{pmatrix} \\ &= \left(\boldsymbol{c}_B^\top B^{-1}, \boldsymbol{c}_N^\top - \boldsymbol{c}_B^\top B^{-1}N\right) \\ &= \left(\boldsymbol{\pi}, \bar{\boldsymbol{c}}\right)\end{aligned}$$

Since the reduced costs must be positive, this proves our theorem. Note also that the "if" part works even if our solution is degenerate.

The fact $\boldsymbol{w} \geq \boldsymbol{0}$ has an interesting geometrical explanation. It effectively states that at an optimal solution $\boldsymbol{x}$, $\boldsymbol{c}$ is a

- Linear combination of the normals to the equality constraints at $\boldsymbol{x}$. If this were not the case, we would be able to move along that equality constraint while decreasing the objective function.
- *Non-negative* linear combination of the normals to the inequality constraints that are tight at $\boldsymbol{x}$. If this were not the case, it would be possible to move away from the constraint while decreasing the objective function.

Diagramatically, consider the following polyhedron, and imagine we are at the BFS $\boldsymbol{x}$:



First note that $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$ are the normals to the *inequality constraints satisfied at* $\boldsymbol{x}$. It should be clear that $\boldsymbol{c}$ needs to be contained in the cone defined by the two directions; if it wasn't the case, we'd be able to decrease the objective function by moving in an opposite direction to $\boldsymbol{c}$ without violating the constraints.

This result should not be surprising. We saw above that $\boldsymbol{c}^\top \boldsymbol{\eta}^j \geq 0$ for all nonbasic j at an optimal solution. In other words, $\boldsymbol{c}$ makes an acute angle with every improving direction; logically, it follows that $\boldsymbol{c}$ must lie in the cone formed by the normals (we prove this formally later using Farkas' Lemma).

- Now, consider that the constraint in the previous theorem can be written as

$$\boldsymbol{c}^\top = \left(\boldsymbol{y}^\top, \boldsymbol{w}\right)\begin{pmatrix} B & N \\ 0 & I \end{pmatrix} = \boldsymbol{y}^\top A + \boldsymbol{w}^\top \qquad \boldsymbol{w}^\top = \left(\boldsymbol{0}^\top, \bar{\boldsymbol{w}}^\top\right) \geq \boldsymbol{0}$$

  Now, if we relax the requirement that the first m components of $\boldsymbol{w}$ be 0

$$A\boldsymbol{y} + \boldsymbol{w} = \boldsymbol{c}$$

  This leads us to define the *dual problem*, with constraints $A\boldsymbol{y} \leq \boldsymbol{c}$, and $\boldsymbol{y}$ free:

$$\max \boldsymbol{b} \cdot \boldsymbol{y} \quad \text{s.t.} \quad A\boldsymbol{y} \leq \boldsymbol{c}$$

- This already gives us some insight into *complementary slackness*; the dual is basically equivalent to the primal, provided that $\boldsymbol{w} = \boldsymbol{0}$ for the first m components; ie: those components that are in the basis. We will indeed see that if $\boldsymbol{x}$ and $\boldsymbol{y}$ are primal and dual feasible and $\boldsymbol{x} \cdot \boldsymbol{w} = 0$, then both are optimal, because they both solve identical problems and weak duality (see later) holds.
- The $\boldsymbol{w}$ (ie: the dual slacks) are none other than the reduced costs of the original problem.

- ***Motivation II***
- ***Weak duality***
  - ***Theorem***: If $\boldsymbol{x}$ is primal feasible and $\boldsymbol{y}$ is dual feasible, then $\boldsymbol{b} \cdot \boldsymbol{y} \leq \boldsymbol{c} \cdot \boldsymbol{x}$

    Proof: Consider
    - We have that $A\boldsymbol{x} = \boldsymbol{b}$, and so $\boldsymbol{y}^\top A\boldsymbol{x} = \boldsymbol{y} \cdot \boldsymbol{b}$.
    - We also have that $A\boldsymbol{y} \leq \boldsymbol{c} \Rightarrow \boldsymbol{y}^\top A \leq \boldsymbol{c}^\top$, and since $\boldsymbol{x} \geq 0$, $\boldsymbol{y}^\top A\boldsymbol{x} \leq \boldsymbol{c} \cdot \boldsymbol{x}$.

    The result follows.
  - ***Corollary***: If $\boldsymbol{x}$ is primal feasible, and $\boldsymbol{y}$ is dual feasible, and $\boldsymbol{b} \cdot \boldsymbol{y} = \boldsymbol{c} \cdot \boldsymbol{x}$, then $\boldsymbol{x}$ and $\boldsymbol{y}$ are optimal solutions to their respective linear programs.
  - The question arises, however, of whether there ever arises $\boldsymbol{x}$ and $\boldsymbol{y}$ that satisfy the hypotheses of this corollary. The answer is provided by…
- ***…Strong duality***
  - ***Theorem***:
    - If either the primal problem or the dual problem has a finite optimal solution, then so does the other, and $\min \boldsymbol{c} \cdot \boldsymbol{x} = \max \boldsymbol{b} \cdot \boldsymbol{y}$.

- If either problem has an unbounded objective function value, then the other has no feasible solution.

(Note: the converse of the second statement is not true – it still leaves the possibility that both problems are infeasible).

Proof:

- Suppose the primal has a finite optimal solution

$$\boldsymbol{x}^* = \begin{pmatrix} \boldsymbol{x}_B^* \\ \boldsymbol{x}_N \end{pmatrix} = \begin{pmatrix} B^{-1}\boldsymbol{b} \\ 0 \end{pmatrix}$$

Now, let $\boldsymbol{y} = B^{-\top}\boldsymbol{c}_B$ [the optimal simplex multipliers]. We then have

$$\boldsymbol{c} - A^\top \boldsymbol{y} = \begin{pmatrix} \boldsymbol{c}_B \\ \boldsymbol{c}_N \end{pmatrix} - \begin{pmatrix} B^\top & N^\top \end{pmatrix} \boldsymbol{y} = \begin{pmatrix} \boldsymbol{c}_B - \boldsymbol{c}_B \\ \boldsymbol{c}_N - N^\top B^{-\top} \boldsymbol{c}_B \end{pmatrix} = \begin{pmatrix} 0 \\ \bar{\boldsymbol{c}}_N \end{pmatrix} \geq 0$$

And so the solution is dual feasible [this is a reformulation of the fact we proved above; that we can write $\boldsymbol{c} = A\boldsymbol{y} + \boldsymbol{w}, \boldsymbol{w} \geq \boldsymbol{0}$]. Furthermore,

$$\boldsymbol{c} \cdot \boldsymbol{x} = \boldsymbol{c}_B^\top B^{-1} \boldsymbol{b} \qquad = \qquad \boldsymbol{b} \cdot \boldsymbol{y} = \boldsymbol{b}^\top B^{-\top} \boldsymbol{c}_B$$

By the corollary, this proves our theorem.

- If $\boldsymbol{c} \cdot \boldsymbol{x} \downarrow -\infty$, the dual cannot be feasible, because any feasible $\boldsymbol{y}$ would result in a lower bound on $\boldsymbol{c} \cdot \boldsymbol{x}$.

- The first part of the proof is illuminating in that it shows that the vector of simplex multipliers for the first problem. At every iteration of the simplex method, $\boldsymbol{c} \cdot \boldsymbol{x} = \boldsymbol{b} \cdot \boldsymbol{y}$ still holds, but some of the reduced costs might be negative, and so the problem is not dual feasible.

Theorems of the Alternative

- Duality is useful in proving various Theorems of the Alternative. An example is Farkas' Lemma.
- ***Theorem (Farkas' Lemma)***: Exactly one of the following two problems have a solution:
  1. $A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}$
  2. $\boldsymbol{y}^\top A \leq \boldsymbol{0}^\top, \boldsymbol{b} \cdot \boldsymbol{y} > 0$

  It is first interesting to consider the geometric interpretation of these two statements:

1. The first statement implies that $\boldsymbol{b}$ is in the cone formed by the columns of A.
2. The second statement implies that $\boldsymbol{y}$ makes an *obtuse* angle will all the columns of A, but an *acute* angle with $\boldsymbol{b}$.

Clearly, only one of these can hold.

Proof: Consider the primal-dual pair

	(P)		**(D)**
min	0		
s.t	$A\boldsymbol{x} = \boldsymbol{b}$	max	$\boldsymbol{b} \cdot \boldsymbol{y}$
	$\boldsymbol{x} \geq 0$	s.t.	$A^\top \boldsymbol{y} \leq \boldsymbol{0}$

And note that

- (1) is feasible if and only if (P) has a solution.
- If (D) is unbounded, (2) has a solution. Similarly, if (2) has a solution $\boldsymbol{y}$, multiple of $\boldsymbol{y}$ also lies in the feasible set of (D), and so (D) is unbounded.

Finally, we note that since the dual is feasible ($\boldsymbol{y} = \boldsymbol{0}$ works), then by Strong Duality, (P) is infeasible if and only if (D) is unbounded.

- Note: it is the case in most of these theorems that one direction is easy to prove. In this case, it is easy to show that *both* (1) and (2) cannot have solutions:

$$\boldsymbol{y}^\top (1) \Rightarrow \boldsymbol{y}^\top A\boldsymbol{x} = \boldsymbol{y} \cdot \boldsymbol{b} \qquad (2)\boldsymbol{x} \Rightarrow \boldsymbol{y}^\top A\boldsymbol{x} \leq \boldsymbol{0}^\top$$

This is clearly impossible if $\boldsymbol{b} \cdot \boldsymbol{y} > 0$.

- Farkas' Lemma allows us to confirm the statement we made above – namely that, at an optimal BFS $\boldsymbol{x}$, $\boldsymbol{c}$ must lie in the cone defined by the normals of the *inequality* constraints that are tight at that point. To see why, consider the LP

$$\min \ \boldsymbol{c} \cdot \boldsymbol{x} \ \text{ s.t. } \ A\boldsymbol{x} \geq \boldsymbol{b}$$

And consider a basic feasible solution $\boldsymbol{x}^*$, and let A_{tight} be the matrix of constraints that *happen* to be satisfied there, so that $A_{\text{tight}}\boldsymbol{x}^* = \boldsymbol{b}$ (the other constraints, contained in A_{slack} satisfy $A_{\text{slack}}\boldsymbol{x}^* > \boldsymbol{b}$). Our cone requirement therefore reduces to the fact that

$$\boldsymbol{c} = A_{\text{tight}}^\top \boldsymbol{y}, \boldsymbol{y} \geq \boldsymbol{0} \ \text{ has a solution}$$

By Farkas' Lemme, however, this is only true if and only if

$$A_{\text{tight}}^{\top}\boldsymbol{y} \geq \boldsymbol{0}, \boldsymbol{c}\cdot\boldsymbol{y} < 0 \text{ has no solution}$$

But note that the statement $A_{\text{tight}}^{\top}\boldsymbol{y} \geq \boldsymbol{0}$ simply requires that $\boldsymbol{y}$ be an improving direction (because if it is true, then $\boldsymbol{x}^{*} + \boldsymbol{y}$ will also satisfy $A(\boldsymbol{x}^{*} + \boldsymbol{y}) \geq \boldsymbol{b}$), and the second statement requires that feasible direction to be "downhill".

Thus, Farkas' Lemma tells us that if there are no downhill feasible directions, $\boldsymbol{c}$ must lie in the cone of the normals of the active inequality constraints.

- ***Complementary slackness***
  - Consider the following dual pair

$$(\text{P}): \min \ \boldsymbol{c}\cdot\boldsymbol{x} \ \text{ s.t. } \ A\boldsymbol{x} \geq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}$$
$$(\text{D}): \max \ \boldsymbol{b}\cdot\boldsymbol{y} \ \text{ s.t. } \ A^{\top}\boldsymbol{y} \leq \boldsymbol{c}, \boldsymbol{y} \geq 0$$

We can re-write these in standard form as

$$(\text{P}): \min \ \boldsymbol{c}\cdot\boldsymbol{x} \ \text{ s.t. } \ A\boldsymbol{x} - \boldsymbol{s} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{s} \geq \boldsymbol{0}$$
$$(\text{D}): \max \ \boldsymbol{b}\cdot\boldsymbol{y} \ \text{ s.t. } \ A^{\top}\boldsymbol{y} + \boldsymbol{w} = \boldsymbol{c}, \boldsymbol{y} \geq \boldsymbol{0}, \boldsymbol{w} \geq \boldsymbol{0}$$

  - ***Theorem***: If $\boldsymbol{x}$ is feasible in (P) and $\boldsymbol{y}$ is feasible in (D), then they are both optimal for their respective problems if and only if

$$\boldsymbol{x}\cdot\boldsymbol{w} = \boldsymbol{x}\cdot\left(\boldsymbol{c} - A^{\top}\boldsymbol{y}\right) = 0 \qquad \boldsymbol{y}\cdot\boldsymbol{s} = \boldsymbol{y}\cdot\left(A\boldsymbol{x} - \boldsymbol{b}\right) = \boldsymbol{0}$$

Proof: Our two constraints are

$$A\boldsymbol{x} - \boldsymbol{s} = \boldsymbol{b}$$
$$A^{\top}\boldsymbol{y} + \boldsymbol{w} = \boldsymbol{c}$$

Consider multiplying the first by $\boldsymbol{y}^{\top}$ on the left, and the second by $\boldsymbol{x}^{\top}$ on the left

$$\boldsymbol{y}^{\top}A\boldsymbol{x} - \boldsymbol{y}\cdot\boldsymbol{s} = \boldsymbol{y}\cdot\boldsymbol{b}$$
$$\boldsymbol{x}^{\top}A^{\top}\boldsymbol{y} + \boldsymbol{x}\cdot\boldsymbol{w} = \boldsymbol{x}\cdot\boldsymbol{c}$$

Subtracting the first from the second

$$\boldsymbol{x}\cdot\boldsymbol{w} + \boldsymbol{y}\cdot\boldsymbol{s} = \boldsymbol{x}\cdot\boldsymbol{c} - \boldsymbol{y}\cdot\boldsymbol{b}$$

This quantity is called the *duality gap*, because it is the gap between the primal and dual optimal solutions. Now, consider that:

- LHS of the above = 0

$\Rightarrow$ RHS of the above = 0

$\Rightarrow$ Optimality, by the corollary to weak duality

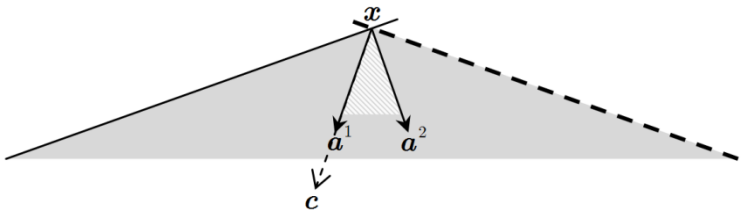
$\Rightarrow$ RHS $= 0$

and the last statement implies the first. Thus, they are all equivalent.

- This makes intuitive sense in light of the above – it is only where the constraints are tight that the Lagrange mutlipliers can be anything apart from 0, so as to "penalize" departure from these constraints.
- This also explains the conditions in the famous Karush-Kuhn-Tucker (KKT) Theorem, which insists that a standard form LP is optimal if and only if there exist $\boldsymbol{y}$ and $\boldsymbol{w}$ such that
  - $A\boldsymbol{x} = \boldsymbol{b}$, with $\boldsymbol{x} \geq \boldsymbol{0}$ [Primal feasibility]
  - $A^{\top}\boldsymbol{y} + \boldsymbol{w} = \boldsymbol{c}$, with $\boldsymbol{w} \geq 0$ [Dual feasibility]
  - $\boldsymbol{x} \cdot \boldsymbol{w} = \boldsymbol{0}$ complementary slackness.

  Or, more general, $\boldsymbol{x}$ is a solution to the problem $\min_x f(\boldsymbol{x})$ s.t. $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$ if and only if
  - $\nabla_x \left\{ f(\boldsymbol{x}) + \boldsymbol{\mu}^{\top}\boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\lambda}^{\top}\boldsymbol{h}(\boldsymbol{x}) \right\} = \boldsymbol{0}$ [Stationarity]
  - $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$ [Primal feasibility]
  - $\boldsymbol{\mu}^{\top} \geq \boldsymbol{0}$ [Dual feasibility]
  - $\mu_i g_i(\boldsymbol{x}) = 0$ for all i [complementary slackness]

- ***Strict complimentary slackness***
  - Consider a situation in which $\boldsymbol{c}$ is a linear combination of a *single* normal at $\boldsymbol{x}$:



  In this case, we have *dual degeneracy* – one of the reduced costs/optimal dual variable slacks is equal to 0.
  - In this case, it is clear that *every* solution along the dotted constraint above are optimal. This implies that *primal non-uniqueness* implies *dual degeneracy*. The converse of this statement, however, is not necessarily true, because even though the reduced cost of moving away along the solid constraint ($\boldsymbol{a}^2$) is 0, there could

have been a third constraint there preventing the move, and therefore preventing non-uniqueness.

- We say a solution satisfies *strict complimentary slackness* if the dual problem is non-degenerate. In other words, provided that *either* $x_i = 0$ *or* $w_i = 0$ for each i, but not both. Every linear program has at least one optimal solution which satisfies strict complimentary slackness (in the case above, any solution along the thick dotted line would do the trick). This is not true of general convex programming.

- ***Economic interpretation of duality***
  - Consider a problem with optimal solution
$$\boldsymbol{x}^* = \begin{pmatrix} \boldsymbol{x}_B^* \\ \boldsymbol{x}_N^* \end{pmatrix} = \begin{pmatrix} B^{-1}\boldsymbol{b} \\ \boldsymbol{0} \end{pmatrix}$$
  - Consider what happens when $\boldsymbol{b}$ is changed to $\hat{\boldsymbol{b}} = \boldsymbol{b} + \delta \boldsymbol{e}_i$. As long as the change is small, the basis remains the same, and the new optimal solution is
$$\hat{\boldsymbol{x}}^* = \begin{pmatrix} B^{-1}(\boldsymbol{b} + \delta \boldsymbol{e}_i) \\ \boldsymbol{0} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_B^* + \delta B^{-1}\boldsymbol{e}_i \\ \boldsymbol{0} \end{pmatrix}$$
    The simplex multipliers do not depend on $\boldsymbol{b}$ and are therefore unchanged; $\hat{\boldsymbol{\pi}} = \boldsymbol{\pi}$
  - Now, consider the new cost
$$\hat{z}^* = \boldsymbol{c}^\top \hat{\boldsymbol{x}}^* = \boldsymbol{c}_B^\top B^{-1}\boldsymbol{b} + \delta \boldsymbol{c}_B^\top B^{-1}\boldsymbol{e}_i = z^* + \delta \pi_i$$
  - It is clear, therefore, that the i^{th} dual variable, π_i, corresponds to the impact on the objective cost of changing the RHS of constraint i by one unit.
  - This also makes sense in terms of geometric interpretation above. We saw that $\boldsymbol{\pi}$ was the particular linear combination of equality constraints that formed $\boldsymbol{c}$. Clearly, if constraint j changes by 1 unit, the amount we move along $\boldsymbol{c}$ is the projection of that constraint onto $\boldsymbol{c}$, which is none other than π_j.
  - Crucially, the above also implies that in a problem in which slack variables were included to turn inequality constraints into equality constraints, the dual variables are simply the reduced costs of these slack variables (or the negative of these reduced costs, depending on the sign of the inequality).

- We can also ascribe an economic interpretation to the dual program itself. Two examples:
  - In a transportation problem where each constraint corresponds to supply at a source or sink, the dual variables can be interpreted as the cost an external contractor would charge to handle the transportation of one unit away from a source or towards a sink. The constraint requires the total cost of transportation of unit material from a given source to a given sink be less than or equal to *our* cost for transportation along that path.
  - In a diet problem, where each problem corresponds to a given nutrient, the dual variables can be interpreted as the cost of a pill containing a unit amount of the said nutrient.

- ***The Dual Simplex Algorithm***
  - The primal simplex algorithm effectively involved jumping from solution to solution while maintaining primal feasibility and complementary slackness and looking for dual feasibility. The dual simplex algorithm does the opposite – it keeps dual feasibility (ie: primal optimality) and complementary slackness and looks for primal feasibility.
  - Consider a basis consisting of m linearly independent columns of A, with the following tableau:

$B^{-1}A$	$B^{-1}\boldsymbol{b}$
$\overline{\boldsymbol{c}}^T$	$-\boldsymbol{c}_B^T B^{-1}\boldsymbol{b}$

Or, in more detail:

$\vdots$		$\vdots$	$x_{B,1}$
$B^{-1}\boldsymbol{A}^{\text{col } 1}$	$\cdots$	$B^{-1}\boldsymbol{A}^{\text{col } n}$	$\vdots$
$\vdots$		$\vdots$	$x_{B,m}$
$\overline{c}_1$	$\cdots$	$\overline{c}_n$	$-\boldsymbol{c}_B^T\boldsymbol{x}_B$

We consider a solution which might be primal infeasible (ie: some of the x_B may be negative) but primal optimal (ie: all the reduced costs are positive).

- The dual simplex method pivots on such a tableau while maintaining dual feasibility. Once again, we describe the operations involved in terms of the array a:

(a_{ij})	$\boldsymbol{x}_B$
$\overline{\boldsymbol{c}}^T$	$-z$

The steps involved are as follows:

- Look at the components of $\boldsymbol{x}_B$. If they are all nonnegative, we're done. Else, pick a negative one; this is the row i, $\boldsymbol{a}^{\text{row } i}$ on which we will pivot; it will *exit* the basis.
- Look along the pivot row; if every component $\boldsymbol{a}_j^{\text{row } i} \geq 0$, the problem is unbounded, with optimal dual cost ∞.
- Otherwise, for each j such that $a_j^{\text{row } i} < 0$, compute $\overline{c}_j / |a_j^{\text{row } i}|$; pick the smallest ratio. The corresponding row will *enter* the basis.
- Perform a pivot operation as for the standard simplex method.

Note that the dual simplex method is identical to the primal simplex method, carried out on the dual (since the dual is the transpose of the primal). [Strictly speaking, this is an ambiguous statement, because the dual is not in standard form – more accurately, it is a version of simplex adapted to problems expressed in terms of inequalities].

- ***Practical issues***

PRIMAL	min	max	DUAL
constraints	$\geq b_i$	≥ 0	variables
	$\leq b_i$	≤ 0	
	$= b_i$	free	
variables	≥ 0	$\leq c_j$	constraints
	≤ 0	$\geq c_j$	
	free	$= c_j$	

	Finite	***Unbounded***	***Infeasible***
Finite	✓		
Unbounded			✓
Infeasible		✓	✓

Sensitivity Analysis

- Consider the linear program

$$\min \ \boldsymbol{c}\cdot\boldsymbol{x} \ \text{ s.t. } \ A\boldsymbol{x}=\boldsymbol{b}, \boldsymbol{x}\geq \boldsymbol{0}$$

And its dual

$$\min \ \boldsymbol{b}\cdot\boldsymbol{y} \ \text{ s.t. } \ A^{\top}\boldsymbol{y}\leq \boldsymbol{c}$$

- We now consider how the solution responds to changes in various problem parameters. It will be important, in doing this, to decide whether a change in a given parameter results in a change in basis or not. If we had an original optimal basis B with an optimal solution $\boldsymbol{x}^{*}$, we require the following two conditions to be met for the basis to remain optimal

$$\begin{aligned} \boldsymbol{x}_B &= B^{-1}\boldsymbol{b}\geq \boldsymbol{0} && \text{[Feasibility]} \\ \boldsymbol{c}^{\top}-\boldsymbol{c}_B^{\top}B^{-1}A &\geq \boldsymbol{0} && \text{[Optimality]} \end{aligned}$$

- ***Adding a new variable***
  - Suppose we add a new variable x_{n+1} together with a corresponding column $\boldsymbol{A}^{\text{col } n+1}$ and objective coefficient c_{n+1}. This gives the new problem

$$\begin{aligned} \min \quad & \boldsymbol{c}^{\top}\boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x}+\boldsymbol{A}^{\text{col } n+1}x_{n+1}=\boldsymbol{b} \\ & \boldsymbol{x}\geq \boldsymbol{0}, x_{n+1}\geq 0 \end{aligned}$$

  - We note that $(\boldsymbol{x}^{*},0)$ is a basic feasible solution, with the same basic matrix B; so we only need to check optimality. In fact, we simply need to check optimality for the new variable:

$$\overline{c}_{n+1}=c_{n+1}-\boldsymbol{c}_B^{\top}B^{-1}\boldsymbol{A}^{\text{col } n+1}\geq 0$$

If this condition is satisfied, our solution is optimal, with the same objective value.

- Otherwise, it is a simple matter to add an extra column to the simplex tableau and perform a few more iterations.

- ***Adding a new inequality constraint***
  - Consider adding a new constraint $\boldsymbol{a}^{\text{row } m+1} \cdot \boldsymbol{x} \geq b_{m+1}$. If $\boldsymbol{x}^*$ satisfies this constraint, then it is an optimal solution to the new problem.
  - If not, we introduce a new nonnegative slack variable, and re-write $\boldsymbol{a}^{\text{row } m+1} \cdot \boldsymbol{x} - x_{n+1} = b_{m+1}$. The matrix A is then replaced by

$$\bar{A} = \begin{bmatrix} A & \boldsymbol{0} \\ \boldsymbol{a}^{\text{row } m+1} & -1 \end{bmatrix}$$

    We introduce a new basis that includes all the variables in B *plus* our new slack variable. This gives

$$\bar{B} = \begin{bmatrix} B & \boldsymbol{0} \\ \boldsymbol{a}_B^{\text{row } m+1} & -1 \end{bmatrix} \qquad \bar{B}^{-1} = \begin{bmatrix} B^{-1} & \boldsymbol{0} \\ \boldsymbol{a}_B^{\text{row } m+1} B^{-1} & -1 \end{bmatrix}$$

    The basic solution is $\left(\boldsymbol{x}^*, \boldsymbol{a}_B^{\text{row } m+1} \cdot \boldsymbol{x}^* - b_{m+1}\right)$ – and it is not feasible, since the original constraint was not satisfied.
  - We want to figure out a way to add this new constraint to our tableau (which, recall, contains $B^{-1}A$). First, consider the problem algebraically. We have that

$$\bar{B}^{-1}\bar{A} = \begin{bmatrix} B^{-1}A & 0 \\ \boldsymbol{a}_B^{\text{row } m+1} B^{-1} A - \boldsymbol{a}^{\text{row } m+1} & 1 \end{bmatrix}$$

    And also that the reduced cost do not change, because the objective coefficient of the new slack variable is 0:

$$\begin{bmatrix} \boldsymbol{c}^\top & 0 \end{bmatrix} - \begin{bmatrix} \boldsymbol{c}_B^\top & 0 \end{bmatrix} \bar{B}^{-1}\bar{A} = \begin{bmatrix} \boldsymbol{c}^\top - \boldsymbol{c}^\top B^{-1} A & 0 \end{bmatrix}$$

  - The above is hardly useful in terms of practically writing the new tableau. More informatively, we be describe the above in terms of row operations:
    - Add a new row to the tableau simply consisting of $\begin{bmatrix} \boldsymbol{a}^{\text{row } m+1} & -1 \end{bmatrix}$
    - Perform the row operations necessary to ensure that the columns of $\bar{B}^{-1}\bar{A}$ that correspond to basic variables form the identity matrix. In particular, this involves:
      - Multiplying the row by –1, to obtain a 1 in the last column.

- Add $\boldsymbol{a}_B^{\text{row } m+1} B^{-1} A$ to the last row – this is equivalent to adding $\left(a_B^{\text{row } m+1}\right)_j$ of the j^{th} row of the tableau to the last row. But remember that
  - The j^{th} row will contain a 1 in the column corresponding to the j^{th} basic variable.
  - $\left(a_B^{\text{row } m+1}\right)_j$ is the current entry in the last row, in the column corresponding to the j^{th} basic variable.

  Thus, these operations are equivalent to ensuring that every component of the last row corresponding to basic variables is 0.

- The only part of the tableau we have not considered in the last column, containing $\boldsymbol{x}_B$. This, however, is rather simple – the old basic variables remain basic, and the slack variable picks up the slack from the inequality.

- ***Adding a new equality constraint***
  - Consider adding a new constraint $\boldsymbol{a}^{\text{row } m+1} \cdot \boldsymbol{x} = b_{m+1}$. If $\boldsymbol{x}^*$ satisfies this constraint, then it is an optimal solution to the new problem.
  - Consider the dual of this problem

$$\begin{aligned} \max \quad & \boldsymbol{p} \cdot \boldsymbol{b} + p_{m+1} b_{m+1} \\ \text{s.t.} \quad & \begin{bmatrix} \boldsymbol{p}^\top & p_{m+1} \end{bmatrix} \begin{bmatrix} A \\ \boldsymbol{a}^{\text{row } m+1} \end{bmatrix} \le \boldsymbol{c}^\top \end{aligned}$$

    If $\boldsymbol{p}^*$ is an optimal basic feasible solution to the original problem, then $\begin{pmatrix} \boldsymbol{p}^* & 0 \end{pmatrix}$ is a feasible solution to the new dual problem.

- ***Changing the vector b***
  - Imagine now that we change the vector $\boldsymbol{b}$ to $\hat{\boldsymbol{b}} = \boldsymbol{b} + \delta \boldsymbol{e}_j$.
  - Changing $\boldsymbol{b}$ does not affect optimality conditions, so all we need to check are feasibility conditions:

$$\begin{aligned} B^{-1}\left(\boldsymbol{b} + \delta \boldsymbol{e}_j\right) &\ge 0 \\ \boldsymbol{x}_B + \delta\left(\boldsymbol{B}^{-1}\right)^{\text{col } j} &\ge 0 \\ \left(x_B\right)_i + \delta\left(B^{-1}\right)_{ij} &\ge 0 \qquad \forall i \end{aligned}$$

Equivalently,

$$\max_{j:\left(B^{-1}\right)_{ij}>0}\left\{-\frac{(x_B)_i}{(B^{-1})_{ij}}\right\} \le \delta \le \min_{j:\left(B^{-1}\right)_{ij}>0}\left\{-\frac{(x_B)_i}{(B^{-1})_{ij}}\right\}$$

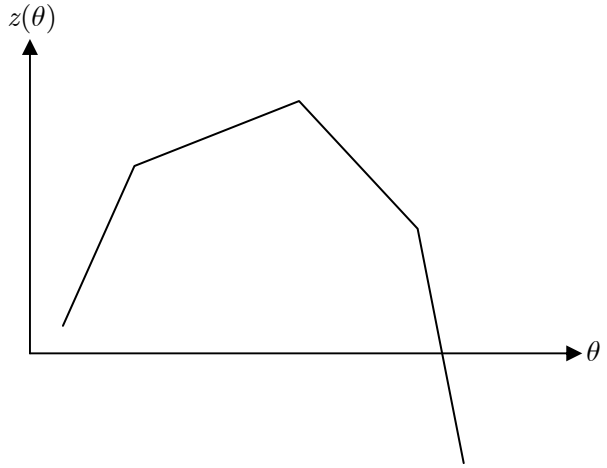
- If δ falls within this range, the current basis is still feasible and optimal. As we saw above, the change in the objective function will be $\delta\pi_j$.
- If δ falls outside the allowed range, the solution is still optimal, but it is primal infeasible. In that case, we simply re-solve the problem using the dual simplex.

- ***Changing the cost vector c***
  - Suppose that we change the vector $\boldsymbol{c}$ to $\hat{\boldsymbol{c}} = \boldsymbol{c} + \delta \boldsymbol{e}_i$. Primal feasibility is clearly not affected. The optimality conditions, however, dictate that
  $$\boldsymbol{c}_B^\top B^{-1} A \le \boldsymbol{c}^\top$$
  - We now consider two options
    - ***c_i is the coefficient of a nonbasic variable*** – in that case, only the i^{th} equation above is modified; we require
    $$\begin{gathered}\hat{c}_i \ge \boldsymbol{c}_B^\top B^{-1}\boldsymbol{A}^{\text{col } i} \\ c_i + \delta \ge -\overline{c}_j + c_i \\ \delta \ge -\overline{c}_j\end{gathered}$$
    - ***c_i is the coefficient of a basic variable*** – in that case, every equation is modified – we require:
    $$\begin{gathered}\left(\boldsymbol{c}_B + \delta \boldsymbol{e}_i\right) B^{-1}\boldsymbol{A}^{\text{col }\alpha} \le c_\alpha \\ \boldsymbol{c}_B B^{-1}\boldsymbol{A}^{\text{col }\alpha} + \delta\left(B^{-1}\boldsymbol{A}^{\text{col }\alpha}\right)_i \le c_\alpha \\ \delta\left(B^{-1}\boldsymbol{A}^{\text{col }\alpha}\right)_i \le \overline{c}_\alpha\end{gathered}$$
  - We can view this in terms of our dual methodology above. Provided the vector $\boldsymbol{c}$ remains within the cone of normals at the basic feasible solution, the solution remains optimal. Otherwise, it "jumps" to an adjacent vertex.

- ***Parametric programming***
  - Consider a linear program of the form
  $$\begin{aligned}\min \quad & (\boldsymbol{c} + \theta \boldsymbol{d}) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \ge 0\end{aligned}$$

We denote the optimal value of this program by $z(\theta)$. Clearly, as θ changes, there will be "breakpoints" at which $\boldsymbol{c}$ leaves of the cone of the current basis, and at which we jump to the next basis.

- ***Claim***: The function $z(\theta)$ is piecewise linear and concave.



As θ gets very large or very small, the problem might become unbounded (in which case $z(\theta) = -\infty$) or it might continue to be bounded, in which case $z(\theta)$ keeps on decreasing linearly indefinitely.

Proof: Two parts:

- Proving linearity is simple; let θ_1 and θ_2 be two values of θ for which the same basis B is optimal. We then have

$$\begin{aligned} z(\theta_2) - z(\theta_1) &= \left[\boldsymbol{c}_B(\theta_2) - \boldsymbol{c}_B(\theta_1)\right] \cdot \boldsymbol{x}_B^* \\ &= (\theta_2 - \theta_1)\boldsymbol{d} \cdot \boldsymbol{x}_B^* \end{aligned}$$

  This is clearly linearly.

- Proving concavity is slightly more involved. Denote the optimal solution at θ by $\boldsymbol{x}^*(\theta)$. Consider any θ_1 and θ_2, and define $\theta_3 = \lambda\theta_1 + (1-\lambda)\theta_2$, where $\lambda \in [0,1]$. Now, it is clear that

$$\begin{aligned} z(\theta_1) &\le \left(\boldsymbol{c} + \theta_1 \boldsymbol{d}\right)^\top \boldsymbol{x}^*(\theta_3) \\ z(\theta_2) &\le (\boldsymbol{c} + \theta_2 \boldsymbol{d})^\top \boldsymbol{x}^*(\theta_3) \end{aligned}$$

  But

$$\begin{aligned} z(\theta_3) &= \left(\boldsymbol{c} + \lambda\theta_1\boldsymbol{d} + (1-\lambda)\theta_2\boldsymbol{d}\right)^\top \boldsymbol{x}^*(\theta_3) \\ &= \lambda\left(\boldsymbol{c} + \theta_1\boldsymbol{d}\right)^\top \boldsymbol{x}^*(\theta_3) + (1-\lambda)\left(\boldsymbol{c} + \theta_2\boldsymbol{d}\right)^\top \boldsymbol{x}^*(\theta_3) \\ &\geq \lambda z(\theta_1) + (1-\lambda) z(\theta_2) \end{aligned}$$

And so z is concave.

Network flow problems

- ***The network flow problem***
  - Let $I(j) = \{(i,j) : (i,j) \in \mathcal{A}\}$ be the set of edges *incoming* into i, and $O(i) = \{(i,j) : (i,j) \in \mathcal{A}\}$ be the set of edges *leaving* node i. Let b_i the *supply* at node i, that enters from the outside. Then the network flow problem is
    $\min \sum_{(i,j)\in\mathcal{A}} c_{ij} f_{ij}$ s.t. $\sum_{(i,j)\in I(j)} f_{ij} - \sum_{(i,j)\in O(i)} f_{ij} = b_j \;\; \forall j \in \mathcal{N}, 0 \leq f_{ij} \leq u_{ij}$
  - The first constraint can concisely be expressed as $A\boldsymbol{f} = \boldsymbol{b}$ where each *row* of A represents a node and each column represents an arc. a_{ij} contains 1 if arc j leads to node i, a –1 if arc j leaves from node i, and a 0 otherwise.
  - To deal with lower bound m_{ij} on flows, simply define $\overline{f}_{ij} = f_{ij} - m_{ij}$, $\overline{u}_{ij} = u_{ij} - m_{ij}$ and $\overline{\boldsymbol{b}} = \boldsymbol{b} - A\boldsymbol{m}$.
  - The dual of the un-capacitated problem is max $\boldsymbol{b}\cdot\boldsymbol{p}$ s.t. $A^\top \boldsymbol{p} \leq \boldsymbol{c}$.
    - Due to the structure of A, we in fact have $p_i - p_j \leq c_{ij} \;\; \forall (i,j) \in \mathcal{A}$. Note that adding or removing a constant from each p_i keeps the solution feasible, and has to effect on the objective (because $\boldsymbol{1}\cdot\boldsymbol{b} = 0$). As such, we can assume $p_n = 0$
    - Complementary slackness requires that $p_i - p_j = c$ for all arcs on which something flows (ie: on which $f_{ij} \neq 0$). These can therefore easily be calculated by setting $p_n = 0$ and backtracking through arcs with flow.
    - The p_i represent shadow prices of increasing b_i by a certain amount.
- ***Network flow algorithms***
  - A *circulation* is a flow vector $\boldsymbol{h}$ such that $A\boldsymbol{h} = \boldsymbol{0}$. The cost of such a circulation is $\boldsymbol{c}\cdot\boldsymbol{h}$, and "pushing" θ units of the flow means setting $\boldsymbol{f} \leftarrow \boldsymbol{f} + \theta\boldsymbol{h}$.

- A *spanning tree solution* $\boldsymbol{f}$ is one that is obtained by (1) picking a set $T \subset \mathcal{A}$ of arcs that form a tree when their direction is ignored (2) partitioning them into two disjoint subsets; set the flow in one subset to 0 (in a capacitated problem, set some to the min flow and some of the max flow), and that in the other subset to satisfy the flow equations, starting from the leafs.

  Theorem: A vector is a *spanning tree solution* if and only if it is a *basic solution* of the minimum-cost flow problem.
- (Another way of finding a starting basis is eliminating all sources and sinks in the system and simply starting with $\boldsymbol{f} = \boldsymbol{0}$).
- Once we have such a solution, we can compute *reduced costs* for each arc not in the spanning tree as $\overline{c}_{ij} = c_{ij} - (p_i - p_j)$. If all are non-negative, we're done. Else, choose a negative one.
- The new arc and the current tree form a cycle – push as much flow as possible around that cycle.
- Note that since A only contains 0s or $\pm$1s, A_B can be rearranged to contain only $\pm$1 on its diagonal. Thus, its determinant is $\pm$1, and by Crammer's Rule, it's inverse contains only integer entries. Finally, $\boldsymbol{f} = A_B^{-1}\boldsymbol{b}$ and $\boldsymbol{\lambda} = \boldsymbol{c}_B^{\top} A_B^{-1}$, and so provided $\boldsymbol{c}$ and $\boldsymbol{b}$ are integer-valued, the primal and dual solutions also are.
- The negative-cycle algorithm creates a *residual network*, in which an arc is created for every "extra bit" of forward and backward capacity in the original network. Finding a negative-cost cycle in the initial network is like finding one in the residual network. The algorithm terminates when there are no negative-cost cycles.

- ***The max-flow problem***
  - ***Example***: m identical machines. Job i requires p_i machine-hours, cannot be processed before time r_i and needs to be completed before d_i. Create a node for each job and a node for each time period (list the r_i and d_i in order, and split time at each point). Arcs into job nodes with capacity p_j indicate how many machine-hours are spent on that job. Arcs from time-periods with capacity

$m \times$ length of time period indicate how much is done in that time period. Arcs from jobs to time periods indicate how much is done of that job in that time period. Our scheduling problem is then a maximum-flow problem. □

- The *Ford-Fulkerson algorithm* proceeds as follows (1) start with a feasible flow (2) search for an augmenting path (a path from s to t in which every forward node is not saturated, and every backward node has flow greater than 0) (3) push as much flow as possible through that path.
- To find an augmenting path, we use a labeling algorithm. Node i is labeled if there is an augmenting path from the source to node i. Start with $I = \{s\}$. Remove a node from I and see if any arc leaving this node could be added to the augmenting path. If so, add all the possible target nodes to I. Repeat until I is empty or contains the sink.
- A cut S is a subset of $\mathcal{N}$ such that $s \in S$ and $t \notin S$. The *capacity* of a cut is given by $C(S) = \sum_{\{(i,j)\in\mathcal{A}: i\in S, j\notin S\}} \text{capacity}_{ij}$. Clearly, the max flow is $\leq$ the min cut. If the Ford-Fulkerson algorithm terminates, the labeled nodes form a cut, and the value of the flow must be equal to the cut (else more nodes would have been labeled). Thus, max flow = min cut.