‘MATH+ECON+CODE’ MASTERCLASS ON MATCHING MODELS, OPTIMAL TRANSPORT AND APPLICATIONS

Alfred Galichon (New York University)

Winter 2019
Monday: Linear programming
Block 2. The min-cost flow problem

- Basic concepts of directed networks
- The min-cost flow problem
- Duality, optimality and equilibrium
- The shortest path problem

- [OTME], Ch.8
- Tolstoi (1930). Methods of finding the minimal total kilometrage in cargo transportation planning in space. *Transportation Planning* [in Russian]
- Koopmans (1949). Optimum utilization of the transportation system. *Econometrica*.
- Schrijver (2002). On the History of the Transportation and Maximum Flow Problems. *Mathematical programming*.

Section 1

MOTIVATION

- In 1930 Tolstoĭ, a Russian engineer, has to optimize the shipping of cement from factories to cities in the Soviet union through railway.
  - each factory produces a fixed number of tons
  - each city needs a fixed number of tons – for now, we'll assume total production=total consumption
  - each factory is connected by rail with a few cities, and the corresponding distance is given
  - how to ship in order to minimize the total distance travelled?
- This problem belongs to the class of *min-cost flow problems*, an important class of linear programming problems, which are the focus of today's lecture. A decade before the invention of linear programming and the work of Kantorovich, Koopmans and Dantzig, Tolstoĭ described a heuristic method for solving the problem, which led to the optimal solution. We'll solve the problem using modern tools (Gurobi), and will see that his solution was right.
- The *shortest path problem*, or how to find the path of minimal distance from a point to another in a network, also belongs in this class. Later in this lecture, you will determine the shortest path between NYU and your place of residence through the NYC subway.

- We will look at three types of data,
  - Tolstoĭ's data, collected by Schrijver (2002), which is in the directory '`02-appli-networks/sovietplanning`'. There are 10 factories and 68 Soviet cities, and 155 links connecting a factory to a city. This yields a sparse 68×10 matrix. Two vectors listing the demand of each city and the supply of each factory are also specified. This is stored in a 69×11 matrix, where we have appended the demand/supply vectors to the right and to the botton of the distance matrix.
  - The data on the NYC subway, available from the mta website, which is in the directory '`02-appli-networks/subway`'. This data is made of two files. The file 'nodes.csv' lists the stations: each station is indexed by the line number; each line has the name of the station, and its spatial coordinates. This is a 472×3 array. The file 'arcs.csv' lists the links between stations: each link specifies the index of the origin, the index of the destination, and the length of the segment. This is a 1230×3 array.
  - The data of the NYC roads ('`02-appli-networks/NYCroads`'), which is available as a 'shapefile' but can be essentially described as the NYC subway above – except that the network is much larger, as there are 67,316 nodes and 169,399 arcs.

Section 2

SETTING

- We start by defining the directed graph on which transportation takes place.

DEFINITION

A (directed) *graph* $(\mathcal{X}, \mathcal{A})$ is a set of *nodes* (cities) $\mathcal{X}$, along with a set of *arcs* $\mathcal{A} \subseteq \mathcal{X}^2$ which are pairs (x, y) where $x, y \in \mathcal{X}$.

Our definition allows an arc to have the same origin and destination. Note that for a dense network, $\mathcal{A} = \mathcal{X}^2$. For a line, $|\mathcal{A}| = |\mathcal{X}| - 1$.

- Next, we define the gradient matrix.

DEFINITION

We define the *gradient matrix* (also called an 'edge-node matrix') as the matrix with general term ∇_{ax}, $a \in \mathcal{A}$, $x \in \mathcal{X}$, such that

$$\begin{aligned} \nabla_{ax} &= -1 \text{ if } a \text{ is out of } x, \\ &= +1 \text{ if } a \text{ is into } x, \\ &= 0 \text{ else.} \end{aligned}$$

Hence, if $f \in \mathbb{R}^{\mathcal{X}}$, then $\nabla f \in \mathbb{R}^{\mathcal{V}}$, and $(\nabla f)_{xy} = f_y - f_x$.
We shall denote $\nabla^\intercal$ the transpose of the gradient matrix. It is the network analog of the $-$div differential operator.

- Next, we define paths and loops

DEFINITION

Given two nodes x and y, a *path* from x to y is a sequence $x_1, x_2, ..., x_K$ in $\mathcal{X}$ where $x_1 = x$, $x_K = y$, and for every $1 \leq k \leq K - 1$, $(x_k, x_{k+1}) \in \mathcal{A}$. A *loop* (also called 'cycle') is a path from a node x to itself.

- A vector $c \in \mathbb{R}^{\mathcal{V}}$ defined transportation costs. That is, for $xy \in \mathcal{V}$, c_{xy} is the transportation cost associated to arc xy. c can also be thought of as the length of arc xy. The cost of moving the good from node x to node y along path $x_1, x_2, ..., x_K$ is

$$\sum_{k=1}^{K-1} c_{x_k x_{k+1}}.$$

- No arbitrage conditions: there is no loop of negative cost:

ASSUMPTION

There is no profitable loop, which means that there is no sequence $x_1, ..., x_K$ in $\mathcal{X}$ such that $x_K = x_1$, $(x_k, x_{k+1}) \in \mathcal{A}$, and $\sum_{k=1}^{K-1} c_{x_k x_{k+1}} < 0$.

In particular, there is no profitable loop if $c \geq 0$.

- Let n_x be the *net demand*, which is the flow of goods disappearing from the graph. The set of nodes defined by

$$\mathcal{X}_0 = \{x \in \mathcal{X} : n_x < 0\} \text{, and } \mathcal{X}_1 = \{x \in \mathcal{X} : n_x > 0\}$$

  are called the *supply nodes* and *demand nodes* respectively. A node which is neither a supply node, neither a demand node is called a *transit node*.
- Total supply is $-\sum_{x \in \mathcal{X}_0} n_x$, total demand is $\sum_{y \in \mathcal{X}_1} n_y$.

ASSUMPTION (BALANCEDNESS)

Assume that total supply equals total demand on the network, that is

$$\sum_{x\in\mathcal{X}_0} n_x + \sum_{y\in\mathcal{X}_1} n_y = 0.$$

ASSUMPTION (CONNECTEDNESS)

Assume the set of supply nodes $\mathcal{X}_0$ is strongly connected to the set of demand nodes $\mathcal{X}_1$, i.e. for every $x \in \mathcal{X}_0$ and $y \in \mathcal{X}_1$, there is a path from x to y.

The specification of the graph, the net demand vector, and the surplus vector defines a network.

DEFINITION

A directed graph $(\mathcal{X}, \mathcal{A})$ endowed with a net demand vector $(n_z)_{z \in \mathcal{X}}$ and a cost vector $(c_a)_{a \in \mathcal{A}}$ is called a *network* $(\mathcal{X}, \mathcal{A}, n, c)$. If Assumptions 1 (No profitable loop), 2 (total supply equals total demand) and 3 (supply is strongly connected to demand) all hold, the network is called *regular*.

Without mention of the contrary we shall assume that the network under consideration is regular.

The flow of mass disappearing at x equals the flow arriving from other nodes minus the flow shipping to other nodes

$$n_x = \sum_{z:(z,x)\in\mathcal{A}} \pi_{zx} - \sum_{z:(x,z)\in\mathcal{A}} \pi_{xz}$$

and this equation can be rewritten as $\nabla^\intercal \pi = n$. This motivates the following definition:

DEFINITION

The set of feasible flows, denoted $\mathcal{M}\left(n\right)$, or $\mathcal{M}$ when there is no ambiguity, is defined as the set of flows $\pi \geq 0$ that verify conservation equation

$$\nabla^\intercal \pi = n. \tag{1}$$

Let ϕ_x be the price of the commodity at x. Consider the strategy which consists in purchasing the good at x, shipping to y, and selling at y. The profit of this strategy is

$$\phi_y - \phi_x - c_{xy} = \left(\nabla\phi - c\right)_{xy}$$

and hence there is no arbitrage opportunity if $\phi_y - \phi_x - c_{xy} \leq 0$ for every arc xy, that is

$$\nabla\phi \leq c.$$

Section 3

The minimum cost flow problem

Consider the *minimum cost flow problem*

$$\min_{\pi \geq 0} \sum_{(x,y)\in\mathcal{A}} \pi_{xy} c_{xy} \tag{2}$$
$$s.t.\ \nabla^{\mathsf{T}} \pi = n$$

which is a Linear Programming problem whose dual is

$$\max_{\phi\in\mathbb{R}^{\mathcal{X}}} \sum_{x\in\mathcal{X}} n_x \phi_x \tag{3}$$
$$s.t.\ \nabla\phi \leq c.$$

PROPOSITION

(i) Under Assumption 1 (No profitable loop), the dual problem (3) is feasible, which means that there is a vector $\phi \in \mathbb{R}^{\mathcal{X}}$ such that $\nabla \phi \leq c$; and the value of Problem (2) is strictly less than $+\infty$.
(ii) Under Assumptions 2 (total supply equals total demand) and 3 (Supply is strongly connected to demand), the primal problem (2) is feasible, which means that there is a flow $\pi \geq 0$ such that $\nabla^{\intercal} \pi = n$; and the value of Problem (3) is strictly greater than $-\infty$.

Assume that $(\mathcal{X}, \mathcal{A}, n, c)$ is a regular network. Then the value of the primal problem (2) coincides with the value of its dual (3), and both problems have solutions. Further, if π is a solution to the primal and ϕ is a solution to the dual, then $\pi_{xy} > 0$ implies $\phi_y - \phi_x = c_{xy}$.

Section 4

SPECIAL CASES

- Assume there is only one source node $s \in \mathcal{X}$ and one target node $t \in \mathcal{X}$, each associated with unit flow. That is, $n_x = 1\{x = t\} - 1\{x = s\}$. Then the problem boils down to how to push one unit of mass from s to t. If we interpret c_{xy} as the distance along arc xy, the solution of this problem corresponds to the shortest path from s to t. This is why this problem is called shortest path problem. (More generally, this problem extends to the case when c does not have a negative loop).
- The dual problem is then

$$\max_{\phi} \phi_t - \phi_s$$
$$s.t.\ \phi_y - \phi_x \leq c_{xy}\ \forall xy \in \mathcal{A}$$

and we can impose normalization $\phi_s = 0$, so that along the travelled path, ϕ_x interprets as the distance travelled thus far.

- We have advocated for the use of Gurobi as a black box in this problem, but there exists a direct method to find out the minimal cost path by dynamic programming.
  - The crucial remark here is that if there is a minimal cost path from s to t, then there is one of length at most $|\mathcal{X}|$.
- For $z \in \mathcal{X}$, and $t \in \mathbb{N}$, let C_{sz}^k be the minimal cost of the path from s to z among paths of length at most k, with the convention that $C_{sz}^k = +\infty$ if there is no such path. One has:
  - $C_{ss}^0 = 0$ and $C_{sz}^0 = +\infty$ for all $z \neq s$, and
  - for $t \geq 1$, C_{sz}^k satisfies the Bellman equation:

$$C_{sz}^k = \min\left\{C_{sz}^k, \min_{x \in \mathcal{Z}: sz \in \mathcal{A}} \left\{c_{sx} + C_{xz}^{k-1}\right\}\right\}.$$

- It is easy to see that $C_{st}^{|\mathcal{X}|}$ is the minimal cost of any path from s to t. Shortest-paths algorithms (Dijikstra and Bellman-Ford) implement this idea.

- Assume the problem is bipartite, that is $\mathcal{X} = \mathcal{S} \cup \mathcal{T}$ and $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{T}$. That is, there are no intermediate nodes, and an arc can only go directly from a source to a target.
- Note that any min-cost flow problem can be recast in this form, by taking the shortest distance between any source node and any target node.
- The dual problem is then

$$\max_{\phi} \sum_{x \in \mathcal{X}} n_x \phi_x$$
$$s.t.\ \phi_t - \phi_s \leq c_{st}\ \forall s \in \mathcal{S}, t \in \mathcal{T}$$

which is our first encounter with optimal transport (more on this tomorrow).