Transport problem

Evgeniy Poltavtsev
Optimization Class Project. MIPT

Introduction

The transport problem is one of the most common linear programming problems. This task will be relevant until humanity discovers a way to deliver any kind of resources with zero cost. The idea of the project is to implement python code, which, according to the input data from the mathematical formulation of the transport problem (The Monge — Kantorovich transport problem), will return the optimal transportation plan.

Mathematical formulation

Needs to do transportation of cargo from n shipping points $A_1,\ A_2,\ ...A_n$ in m destination points $B_1,\ B_2,\ ...B_m$. It is possible to take out a_k units of cargo from the point A_k and it is necessary to deliver b_j units of cargo to the destination B_j. Transportation from point A_k point B_j costs c_{kj} units of some currency. It is required to ensure such transportation so that the minimum amount of money is spent and the necessary amount of cargo is delivered to each destination.

- Denote by x_{kj} the units of cargo transported from point A_k to point B_j and get the following task.

$$\sum_{k=1}^{n}\sum_{j=1}^{m} c_{kj}x_{kj} \to min,\ x_{kj} \geq 0,\ k = 1, ...n;\ j = 1, ...m;$$
$$\sum_{j=1}^{m} x_{kj} = a_k,\ k = 1, ...n;$$
$$\sum_{k=1}^{n} x_{kj} = b_j,\ j = 1, ...m;$$

- After redefining variables and defining the matrix A:

$$c^* = (c_{11}, c_{12}, \dots, c_{1m}, \dots, c_{n1}, c_{n2}, \dots, c_{nm})$$
$$x = (x_{11}, x_{12}, \dots, x_{1m}, \dots, x_{n1}, x_{n2}, \dots, x_{nm})^T$$
$$A = \begin{pmatrix} 1 & 1 & \dots & 1 & 0 & 0 & \dots & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 1 & 1 & \dots & 1 & \dots & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & 1 & 1 & \dots & 1 \\ 1 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & \dots & 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & 0 & 1 & \dots & 0 & \dots & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 & 0 & \dots & 1 & \dots & 0 & 0 & \dots & 1 \end{pmatrix}$$
$$\overline{b} = (a_1, a_2 \dots, a_n, b_1, b_2, \dots, b_m)^T$$

we get the following linear programming problem:

$$c^{top} \cdot x \to min,\ Ax = \overline{b},\ x \geq 0.$$

- We consider a closed transport problem: total amount of cargo at departure points is equal to the total amount of cargo at destination points.
- We need to find a solution x - which will be the optimal transportation plan
- We looking for solution by using the method of potentials

Algorithm

We solve this problem using the next few steps:

- Bring the problem to a closed model (all we need to do is add a fictitious destination with a zero transportation cost)
- Find the initial transportation plan x, which will be the extreme point of the set of acceptable points.
- Use the method of potentials based on the formulation of the dual problem to transport.

Initial transportation plan

A number of methods are used to find the initial extreme point (our initial transportation plan). Below is an algorithm of the matrix minimum method:

given $a_k,\ k = 1, ..., n;$
$b_j,\ j = 1, ..., m;$
$c_{kj},\ j = 1, ..., m,\ k = 1, ...n;$
repeat
1. Let $c_{kj} = min$ of C matrix.
2. **if** $a_k < bj$
   $b_j - = a_k$ and $x_{kj} = a_k$
   delete k column of C
3. **if** $a_k > bj$
   $a_k - = b_j$ and $x_{kj} = b_j$
   delete j stroke of C
3. **else**
   $x_{kj} = b_j$
   delete k column
   delete j stroke of C

return C matrix.

	b_1	b_2	...	b_m
a_1	c_{11}	c_{12}	...	c_{1m}
a_2	c_{21}	c_{22}	...	c_{2m}
...	...	...	...	...
a_n	c_{n1}	c_{n2}	...	c_{nm}

***C* matrix**

The method of potentials

- To use this method, we need to define a problem dual to ours. Using statements from Mathematical formulation we have next problem:

$$y^* \cdot \overline{b} \to max,\ y^*A \leq c$$

where y^* - is stroke with n+m dimension.

- We can rewrite $y^* = (u_1, ...u_n, v_1, ...v_m)$, u_k and v_j - called potentials.

Below is an algorithm:

given C matrix, x_{kj}, (from previous algorithm)
repeat
1. Let $k[n]$, $j[m]$ be some non zero indexes of this components
2. **create** u[size(k)], v[size(j)] - ours potentials
   $u_k + v_j = c_{kj}$ - system
3. **Solve the system**
4. **create** $\overline{C} = \{\overline{c_{kj}}\} = u_k + v_j$
   $\Delta = C - \overline{C}$
5. **if** $\Delta_{kj} \geq 0$ (for all $j = 1, ..., m,\ k = 1, ...n;$)
   return x_{kj} - solution
6. **recreate** x_{kj} - recreation based on position of min Δ_{kj}

return x_{kj} - solution

Numerical example

We consider the data for our task in the form of the following table:

		B_1	B_2	B_3	B_4
		330	480	500	575
A_1	800	1	2	5	3
A_2	295	1	6	5	2
A_3	200	6	3	1	4
A_4	290	3	4	3	6
A_5	300	1	2	3	1

- We consider the results of the potentials method and the simplex method (the transport problem is a linear programming problem).
- The implementation of the simplex method is done using the python function **scipy.optimize.linprog**. With the corresponding field (method $= 'simplex'$).
- The implementation of the potentials method is done using this code.

Results

Both methods produce the same optimal transportation plan.

- The transportation plan is a table in which the cells correspond to the quantity of goods:

	B_1	B_2	B_3	B_4
A_1	320	480		
A_2	10			285
A_3			200	
A_4			290	
A_5			10	290

Total cost of this plan is: 3250 (units of some currency)

With function **time.time()** we can see the difference between the work of these methods. According to the result of the **simplex method** in this example it takes approximately 0.01 sec to solve our problem. While **method of potentials** works in 0.0009 sec (in some cases, **time.time()** does not recognize the operating time due to its smallness).

Acknowledgements

This material is supported by pro-tips from Daniil Merkulov and my mom.

References

[1] Textbook Osipenko K.Y. "Convex analysis"
[2] Optimization methods. MIPT 2021-2022
[3] Website (GitHub source) with **method of potentials** realization.
[4] Linear programming problems: **python** solution.
[5] Repository with project code.