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	$\dots$	b_m
a_1	c_{11}	c_{12}	$\dots$	c_{1m}
a_2	c_{21}	c_{22}	$\dots$	c_{2m}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
a_n	c_{n1}	c_{n2}	$\dots$	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 are considering the following situation:

- Taxi Rhythm has 9 taxi companies in Moscow (addresses can be found on their website) - these will be our departure points.
- In the Moscow region there are 6 train stations and 3 airports where, according to statistics, most taxi orders occur - these will be our destinations.
- The cost matrix will be compiled by analyzing the driver's expenses to get from the taxi company to the train station or airport (initially, the matrix will contain the distances in kilometrs)

		B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9
		3	9	3	6	3	12	36	24	12
A_1	13	16	19	15	21	24	44	34	56	19
A_2	7	15	17	14	19	21	31	32	62	22
A_3	14	15	17	13	19	20	26	42	62	22
A_4	10	22	24	21	20	15	21	53	44	23
A_5	15	15	14	18	8	11	32	54	37	16
A_6	13	6	6	8	13	12	37	46	43	6
A_7	9	26	22	30	20	25	47	79	22	23
A_8	15	15	13	30	11	19	58	67	32	16
A_9	12	15	10	18	17	22	46	56	49	10

- 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.
Total cost of this plan is: 12447 ₽

With **timeit** 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 45.4 ms to solve our problem (best loop).
- **method of potentials** works in $\approx$ 6.21 ms (best loop).

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] Repository with project code.