Sheet 10

Alexander Bespalov, Philipp Köhler

Exercise 1

a)

The total cost of the transport plan is:

$$\text{total cost } = \langle \mathbf{T}, \mathbf{C} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{T}_{ij} \mathbf{C}_{ij}$$

The optimal transport plan is given my the following linear program:

$$\begin{aligned} \underset{\mathbf{T}}{\text{minimize}} \quad & \langle \mathbf{T}, \mathbf{C} \rangle \\ \text{subject to} \quad & \textstyle\sum_{j=1}^{n} \mathbf{T}_{ij} = a_i \forall i \in \{1, \ldots, n\} \\ & \textstyle\sum_{i\ \ 1}^{n} \mathbf{T}_{ij} = b_j \forall j \in \{1, \ldots, n\} \\ & \mathbf{T}_{ij} \geq 0 \forall (i, j) \in \{1, \ldots, n\} \times \{1, \ldots, n\} \end{aligned}$$

Assuming linear transportation cost, the mass should be transported along straight direct lines as this minimizes the total cost. At its core, this is due to the triangle inequality:

$$\|x - z\| \leq \|x - y\| + \|y - z\|.$$

which states that the direct straight-line distance between two points (x and z) is always less than or equal to the sum of distances along any other path passing through an intermediate point (y). As the cost scales linearly with the path length the same argument follows for the cost. As any arbitrary curve can be described by many infinitely small triangles stichted together, this arguments holds for any curve.

b)

Define flattened version of T_{ij}, such that:

$$\mathbf{x} = [T_{11}, T_{12}, \ldots, T_{1n}, T_{21}, \ldots, T_{mn}]^T$$

Define cost vector $\mathbf{c}$ as:

$$\mathbf{c} = [c_{11}, c_{12}, \ldots, c_{1n}, c_{21}, \ldots, c_{mn}]^T$$

where $c_{ij} = \|x_i - y_j\|$.

Constraints: For each source i, the total transported mass is:

$$\sum_{j=1}^{n} T_{ij} = a_i, \quad \forall i$$

In matrix form:

$$\mathbf{A}_{\text{row}}\,\mathbf{x} = \mathbf{a}$$

where $\mathbf{A}_{\text{row}} \in \mathbb{R}^{m \times mn}$. 2. Demand constraint: For each sink j, the total transported mass is:

$$\sum_{i=1}^{m} T_{ij} = b_j, \quad \forall j$$

In matrix form:

$$\mathbf{A}_{\text{col}}\mathbf{x} = \mathbf{b}$$

where $\mathbf{A}_{\text{col}} \in \mathbb{R}^{n \times mn}$. 3. Non-negativity constraint:

$$\mathbf{x} \geq 0$$

Stacking $\mathbf{A}_{\text{row}}$ and $\mathbf{A}_{\text{col}}$:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{\text{row}} \\ \mathbf{A}_{\text{col}} \end{bmatrix},$$

and combining $\mathbf{a}$ and $\mathbf{b}$:

$$\mathbf{d} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}$$

the constraints become:

$$\mathbf{A}\mathbf{x} = \mathbf{d}, \quad \mathbf{x} \geq 0$$

The linear program can now be written as:

$$\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x}, \quad \text{subject to: } \mathbf{A}\mathbf{x} = \mathbf{d}, \quad \mathbf{x} > \mathbf{0}$$

c)

Very good

In [3]:
```
import numpy as np
from scipy.optimize import linprog
from scipy.spatial.distance import cdist

d = 5
num_sources = 10
num_sinks = 20

np.random.seed(42)
mass_sources = np.random.random(num_sources)
mass_sinks = np.random.random(num_sinks)
mass_sources /= np.sum(mass_sources)
mass_sinks /= np.sum(mass_sinks)
```

```
coords_sources = np.random.rand(num_sources, d)
coords_sinks = np.random.rand(num_sinks, d)

# Cost matrix
cost_matrix = cdist(coords_sources, coords_sinks, metric="euclidean")
cost_vector = cost_matrix.flatten()  # Flatten the cost matrix into a vector

# Constraints for the LP problem
A_supply = np.kron(np.eye(num_sources), np.ones(num_sinks))
A_demand = np.kron(np.ones(num_sources), np.eye(num_sinks))
A_eq = np.vstack([A_supply, A_demand])
b_eq = np.hstack([mass_sources, mass_sinks])
bounds = [(0, None)] * (num_sources * num_sinks)
```

```
# Solve the linear programming problem
result = linprog(cost_vector, A_eq=A_eq, b_eq=b_eq, bounds=bounds, method="highs
```

```
# Extract the final transportation cost
final_cost = result.fun if result.success else None
print("The final cost is: ", final_cost)
```

```
The final cost is:  0.6707468352771471
```

d)

Suppose we have two sources, x_1 (bottom left) and x_2 (top left), with supplies a_1 and a_2, and two sinks, y_1(bottom right) and y_2(top right), with demands b_1 and b_2. It is $\|x_1 - y_1\| \leq \|x_1 - y_2\|$ and similarily $\|x_2 - y_2\| \leq \|x_2 - y_1\|$. The right hand side of both inequalities is the case of intersecting routes which is as we can see the longer route, thus it is suboptimal.

What if one sink can't be exactly compensated by a source?