CraterGrader Transport Planner

Alex Pletta, Ben Younes, John Harrington, Russell Wong, Ryan Lee

October 4, 2022

Contents

1 Problem Description

The Transport Planner solves the problem of minimal-energy terrain manipulation to convert an original topography to a design topography. The planner uses nodes defined with a certain amount of volume and planar coordinates within a worksite as shown in Figure 1. A set of "source" nodes are the position of volumes that are above a design topography and a set of "sink" nodes are the position of volumes that are below a design topography. The planner adopts concepts of volume in a distribution and distance moved between

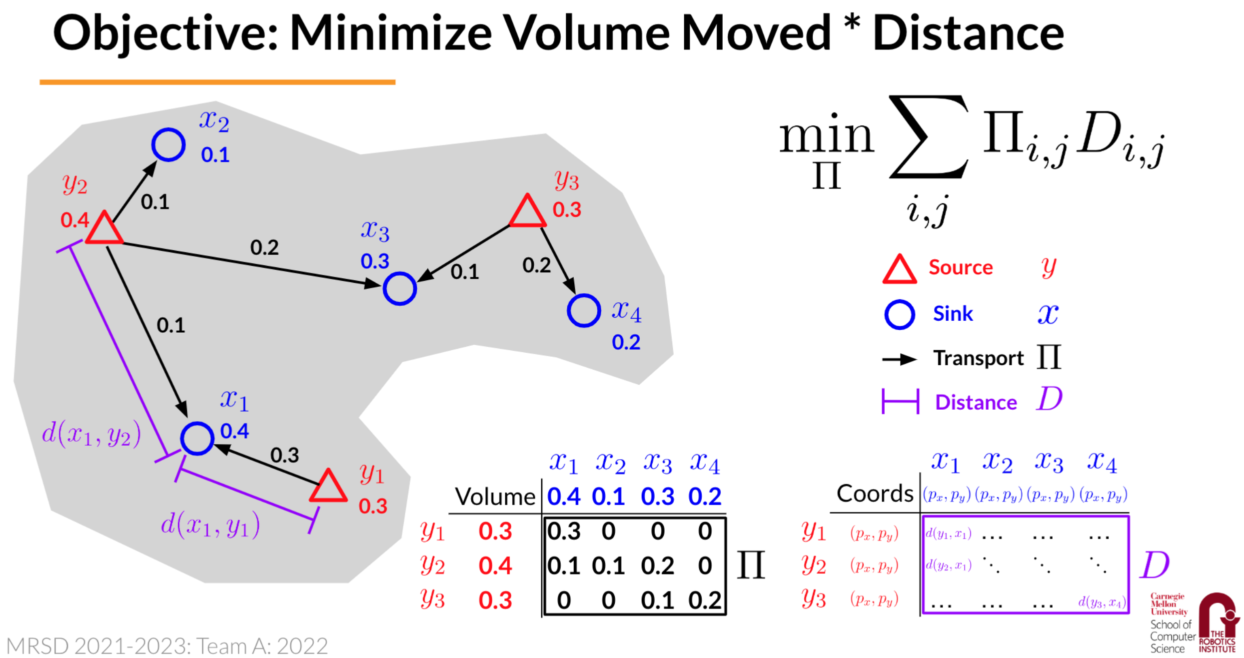


Figure 1. Optimization problem illustration.

distributions from the Earth Mover's Distance to solve for a plan of how much volume to move from each source node to each sink node while minimizing the total sum of volume moved multiplied by the distances traveled; analogous to mechanical work, ignoring transport rate. The transport matrix Π and distance matrix D are both $n \times m$ matrices for n source nodes and m sink nodes.

The set of n source nodes is denoted as $\bar{Y}$ and the set of m sink nodes is denoted as $\bar{X}$. Nodes are defined by planar worksite position coordinates (p_x, p_y) and source volume y or sink volume x. For example, source node Y_i is defined by $< p_{x,i}, p_{y,i}, y_i >$ and sink node X_j is defined by $< p_{x,j}, p_{y,j}, x_j >$. Note that this volume can also be specified as the height at the node location; in this case the height is essentially a "1D volume". In the case of a discretized 2.5D height grid, the height can stand-in for actual volume since the height and volume are just linearly related by the grid cell resolution.

Both source and sink volumes are defined as positive in order to simplify the optimization formulation. The source volume is "extra" volume that should not be present in the final design topography, and the sink volume is "missing" volume that must be filled in order to match the desired design topography.

$$
\begin{aligned}
\bar{Y} =& \{Y_1, Y_2, ..., Y_n\} \\
=& \{< p_{x,1}, p_{y,1}, y_1 >, < p_{x,2}, p_{y,2}, y_2 >, ..., < p_{x,n}, p_{y,n}, y_n >\}
\end{aligned} \tag{1}
$$

$$
\begin{aligned}
\bar{X} =& \{X_1, X_2, ..., X_m\} \\
=& \{< p_{x,1}, p_{y,1}, x_1 >, < p_{x,2}, p_{y,2}, x_2 >, ..., < p_{x,n}, p_{y,n}, x_m >\}
\end{aligned} \tag{2}
$$

where

$$
|\bar{Y}| = n \quad , \quad |\bar{X}| = m \tag{3}
$$

2 Generalized Formulation

First a distance matrix 4 is calculated between all nodes as the euclidean distance, using the planar positions of the nodes in the worksite. Note that this distance could be squared to speed up computation, though the bulk of planner computation time comes from solving the optimization problem itself so the euclidean distance is kept to more closely match the objective value to "mechanical work" (ignoring transport rate).

$$
D_{i,j} = d(Y_i, X_j) \quad \forall \quad i = 1, ..., n \text{ and } j = 1, ..., m \tag{4}
$$

where

$$
d(Y_i, X_j) = ||Y_{i,p} - X_{j,p}||_2^2 = ||\underbrace{\begin{bmatrix} p_{x,i} \\ p_{y,i} \end{bmatrix}}_{Y_{i,p}} - \underbrace{\begin{bmatrix} p_{x,j} \\ p_{y,j} \end{bmatrix}}_{X_{j,p}}||_2^2 \tag{5}
$$

The overall goal is to calculate the transport matrix Π dictating which volumes should be moved between source nodes and sink nodes. The objective 6 is to minimize total effort expended to move volumes from the source nodes to the sink nodes. A first constraint 7 is applied to make all transport volumes positive.

2.1 Equal Source and Sink Volumes

Additional constraints are made on volume moved between the nodes 8, 9. If the sink volume exactly equals the source volume, then the constraints are simply that the sum of volume moved from each source and to each sink total each source and sink respectively. In other words, all sources are completely depleted and all sinks are completely filled. The summation of volume moved is performed using one's vectors ($\mathbf{1}$) of length m and n for sources y and sinks x respectively.

Note that going forward, the vectors y and x contain only the volume/height values.

$$
\begin{aligned}
\min_{\Pi} \quad & \sum_{\substack{i=1,2,\ldots,n \\ j=1,2,\ldots,m}} \Pi_{i,j} D_{i,j} && \text{Minimize total effort expenditure} && (6)\\
\text{s.t.} \quad & \Pi_{i,j} \geq 0 \quad \forall i, j && \text{All transports defined as positive} && (7)\\
& \Pi \mathbf{1}_m - y = 0 && \text{Empty all sources completely} && (8)\\
& \mathbf{1}_n^T \Pi - x = 0 && \text{Fill all sinks completely} && (9)
\end{aligned}
$$

where

$$
\underset{n\times 1}{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad , \quad \underset{m\times 1}{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}
$$

2.2 Cases of Arbitrary Volumes

The constraints on the source volumes 8 and sink volumes 9 will fail in the basic formulation if the source volume is not equal to the sink volume. For example, if there is more source volume than sink, then the sink volume will be filled and 9 will be satisfied but there will be left-over source volume and 8 will fail. Conversely, if there is more sink volume than source, all source volume will be moved so 8 will be satisfied but some sink volume will be unfilled so 9 will fail.

Extending these constraints to support solving for volume inequalities makes the planner much more robust to node input, such as for the cases of mapping inaccuracies or an underlying site topography with non-equal source and sink volumes. One simple approach to make this extension could be to ignore some volume from the larger of the source and sink volumes, but the question then becomes which volume to ignore. Instead, the constraints can be re-written as mixed integer constraints to account for these volume differences while ignoring the optimal amount and location of volume.

The two cases are:

1. **Case 1: more source than sink;** $\sum(y) > \sum(x)$

   The sink constraint 9 will be satisfied, and the source constraint 8 will have left-over volume. So:

$$
\Pi \mathbf{1}_m - y < 0 \tag{10}
$$

$$
\mathbf{1}_n^T \Pi - x = 0 \tag{11}
$$

2. **Case 2: more sink than source;** $\sum(y) < \sum(x)$ The source constraint 9 will be satisfied, and the sink constraint will have un-filled volume. Then $\mathbf{1}_n^T \Pi < x$ so:

$$\Pi \mathbf{1}_m - y = 0 \tag{12}$$

$$\mathbf{1}_n^T \Pi - x < 0 \tag{13}$$

To make the transport optimal, the constraints then simply have to enforce that either the sink or source will be zero for case 1 or case 2, respectively. Since the other constraint for extra volume will be negative, then the constraint is that the maximum of the source and sink transport volume differences is equal to zero.

$$\max \left\{ \Pi \mathbf{1}_m - y, \mathbf{1}_n^T \Pi - x \right\} = 0 \tag{14}$$

2.3 Mixed-Integer Representation for Maximization Constraint

We now need to re-write the new maximization constraint 14 as individual expressions for use with the optimization problem. More generally, we want a set of constraints to enforce $X = \max\{x_1, x_2\}$ for some threshold constant X and expressions x_1, x_2. We can find these constraints by introducing a binary decision variable b and upper-bound constant M that satisfy the following conditions:

$$b = \begin{cases} 0, & \text{if } x_1 < x_2 \\ 1, & \text{otherwise (i.e. } x_1 \geq x_2 \text{)} \end{cases} \quad , \quad M \geq x_1, x_2$$

The definition of b and M give the following constraints:

$$x_1 - x_2 \leq Mb \tag{15}$$

$$x_2 - x_1 \leq M(1 - b) \tag{16}$$

Which make the following cases:

1. **Case 1:** $b = 0$

   Substituting $b = 0$ the constraints become:

$$x_1 - x_2 \leq 0 \qquad \text{Satisfied by definition of } b = 0 \text{, since } x_1 < x_2 \tag{17}$$

$$x_2 - x_1 \leq M \qquad \text{Satisfied by definition of } M \text{, since } x_2 - x_1 \leq x_2 \text{ and } x_2 \leq M \tag{18}$$

   In this case, x_2 is the larger expression. Looking at equations 17 and 18 we can see that the upper bound on x_2 then comes from rearranging 16:

$$x_2 \leq x_1 + M(1 - b) \tag{19}$$

2. **Case 2:** $b = 1$

   Substituting $b = 1$ the constraints become:

$$x_1 - x_2 \leq M \quad \text{Satisfied by definition of } M\text{, since } x_1 - x_2 \leq x_1 \text{ and } x_1 \leq M \tag{20}$$
$$x_2 - x_1 \leq 0 \quad \text{Satisfied by definition of } b = 1\text{, since } x_1 \geq x_2 \tag{21}$$

   In this case, x_1 is the larger, or equal, expression. Looking at equations 20 and 21 we can see that the upper bound on x_1 then comes from rearranging 15:

$$x_1 \leq x_2 + Mb \tag{22}$$

The maximization equality can then be enforced by lower and upper bounds on the threshold constant X:

1. **Lower bound constraints on threshold constant** X

   Since the equality constraint is a maximization, then one of the expressions must equal the threshold and the other must be less than the threshold.

$$x_1 \leq X \tag{23}$$
$$x_2 \leq X \tag{24}$$

2. **Upper bound constraints on threshold constant** X

   When the binary variable is zero, the upper bound is driven by equation 19. Similarly, when the binary variable is one, the upper bound is driven by equation 22. So, the threshold constant upper bounds are:

$$x_1 + M(1-b) \geq X \quad \text{From Case 1 upper bound } (b = 0) \tag{25}$$
$$x_2 + Mb \geq X \quad \text{From Case 2 upper bound } (b = 1) \tag{26}$$

So in summary, we can take a maximization equality constraint

$$X = \max\{x_1, x_2\} \tag{27}$$

And re-write as the following constraints, using binary decision variable b and upper-bound constant M:

$$x_1 \leq X \tag{28}$$
$$x_2 \leq X \tag{29}$$
$$x_1 + M(1-b) \geq X \tag{30}$$
$$x_2 + Mb \geq X \tag{31}$$

2.4 Transport Optimization with Mixed-Integer Constraints

Using the mixed-integer representation, we can now re-write the volume transport maximization constraint 14 by making the following substitutions:

$$
\begin{aligned}
&X \to 0 \\
&x_1 \to \mathbf{1}_n^T \Pi - x \\
&x_2 \to \Pi \mathbf{1}_m - y \\
&M \to \max \left\{ \sum(y), \sum(x) \right\}
\end{aligned}
$$

So, the constraints become:

$$
\begin{aligned}
\mathbf{1}_n^T \Pi - x \leq& 0 && (32)\\
\Pi \mathbf{1}_m - y \leq& 0 && (33)\\
\mathbf{1}_n^T \Pi - x + M(1-b) \geq& 0 && (34)\\
\Pi \mathbf{1}_m - y + Mb \geq& 0 && (35)
\end{aligned}
$$

With binary decision variable $b \in 0, 1$.

2.5 Final Generalized Form

The final optimization problem can then be summarized as the following.

$$
\begin{aligned}
\min_{\Pi} \quad & \sum_{\substack{i=1,2,\ldots,n \\ j=1,2,\ldots,m}} \Pi_{i,j} D_{i,j} && (36)\\
\text{s.t.} \quad & -\Pi_{i,j} \leq 0 \quad \forall i, j && (37)\\
& \mathbf{1}_n^T \Pi - x \leq 0 && (38)\\
& \Pi \mathbf{1}_m - y \leq 0 && (39)\\
& -\mathbf{1}_n^T \Pi + x - M(1-b) \leq 0 && (40)\\
& -\Pi \mathbf{1}_m + y - Mb \leq 0 && (41)\\
& b \in \{0, 1\} && (42)
\end{aligned}
$$

where

$$
\underset{n \times 1}{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \underset{m \times 1}{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}, \quad M = \max \left\{ \sum(y), \sum(x) \right\}
$$

Note that the sink constraints 38 and 40 become tight to $= 0$ when there is more source than sink ($\sum(y) \geq \sum(x)$, $b = 1$) and constraints 39 and 41 become tight to $= 0$ when there is less source than sink ($\sum(y) < \sum(x)$, $b = 0$).

With this observation, although this is technically a mixed integer linear program because of the binary decision variable b, the case for b can be checked prior to solving by comparing the volumes of source vs. sink. In the case of non-equal volumes, the optimal solution will use all of the lesser volume. So, b can be solved online as:

$$b = \begin{cases} 0, & \text{if } \sum(y) < \sum(x)\text{; more sink than source} \\ 1, & \text{otherwise (i.e. } \sum(y) \geq \sum(x)\text{); more source than sink (or equal)} \end{cases} \tag{43}$$

3 Formulation for `CVXOPT` Implementation

`CVXOPT` (https://cvxopt.org/) is a python optimization library that was used for initial planner prototyping. This section shows an example of how to implement the Transport Planner optimization problem into linear programming matrices for the `CVXOPT` library (https://cvxopt.org/userguide/coneprog.html#linear-programming).

The `CVXOPT` library for linear programming can solve optimization problems with the following form:

$$\begin{aligned} \text{minimize} \quad & c^T \hat{x} \\ \text{subject to} \quad & G\hat{x} \leq h \\ & A\hat{x} = b \end{aligned}$$

Our problem is already nicely formatted for using the dot product objective $c^T \hat{x}$ and inequality constraint $G\hat{x} \leq h$. We will disregard the equality constraint $A\hat{x} = b$.

3.1 Objective

To convert the objective, we simply need to turn the sum $\sum_{\substack{i=1,2,\ldots,n \\ j=1,2,\ldots,m}} \Pi_{i,j} D_{i,j}$ into a dot product of a vector c of constant cost terms and a vector $\hat{x}$ of the optimization variables. The distance matrix can be converted into the constant cost vector and the transport matrix can be converted into the optimization variable vector using row major order as follows:

$$\underset{nm\times 1}{c} = \begin{bmatrix} d(Y_1, X_1) \\ d(Y_1, X_2) \\ \vdots \\ d(Y_2, X_1) \\ d(Y_2, X_2) \\ \vdots \\ d(Y_n, X_m) \end{bmatrix}, \quad \underset{nm\times 1}{\hat{x}} = \begin{bmatrix} \Pi_{1,1} \\ \Pi_{1,2} \\ \vdots \\ \Pi_{2,1} \\ \Pi_{2,2} \\ \vdots \ \Pi_{m,n} \end{bmatrix} \tag{44}$$

3.2 Constraints

To convert the constraints, we need to make all inequalities less than a vector of constant values h and create a linear matrix G to multiply the optimization variable vector $\hat{x}$ by. Note that the following formulations assume row-major order.

To replicate the $\Pi \mathbf{1}_m$ term in the matrix G, we can make a one's block matrix $B_{1,m}$ packed with the one's vector $\mathbf{1}_m$ along the "diagonal". There need to be n block "columns" in total to have zeros applied to the rest of the optimization variable vector $\hat{x}$, and n block rows to sum for all n source nodes y, making the final dimensions $B_{1,m} \in \mathbb{R}^{n\times(m\times n)}$.

$$\underset{n\times mn}{B_{1,m}} = \begin{bmatrix} \mathbf{1}_m & 0 & \dots & 0 \\ 0 & \mathbf{1}_m & \dots & 0 \\ \vdots & \dots & \ddots & 0 \\ \vdots & \dots & \dots & \mathbf{1}_m \end{bmatrix} \tag{45}$$

Similarly, to replicate the $\mathbf{1}_n^T \Pi$ term in the matrix G, we can make a one's block matrix $B_{1,n}$ packed with n identity matrices. This then applies zeros to all of the optimization variable vector $\hat{x}$ except for what would be each column in the transport matrix.

$$\underset{m\times mn}{B_{1,n}} = \begin{bmatrix} \underset{m\times m}{I} & \underset{m\times m}{I} & \cdots & \underset{m\times m}{I} \end{bmatrix} \tag{46}$$

These block matrices $B_{1,m}$ and $B_{1,n}$ can be used for constraints 38, 39, 40, 41. The last constraint 37 is then to enforce that all transport volumes are positive or zero, which can be done with an identity matrix having a diagonal of length $(n\times m)$ equal to a zero's vector $\mathbf{0}$ of length nm.

$$\underset{nm\times nm}{I_{\Pi}} = \underset{nm\times 1}{\mathbf{0}} \tag{47}$$

The last step is then to form G by combining the block matrices with the respective rows in h.

$$
\begin{aligned}
-I_{\Pi}\hat{x} \leq& 0 && \text{Replaces constraint 37} && (48)\\
B_{1,n}\hat{x} - x \leq& 0 && \text{Replaces constraint 38} && (49)\\
B_{1,m}\hat{x} - y \leq& 0 && \text{Replaces constraint 39} && (50)\\
-B_{1,n}\hat{x} + x - M(1-b) \geq& 0 && \text{Replaces constraint 40} && (51)\\
-B_{1,m}\hat{x} + y - Mb \geq& 0 && \text{Replaces constraint 41} && (52)
\end{aligned}
$$

Rearranging, we then have:

$$
\begin{aligned}
-I_{\Pi}\hat{x} \leq& \mathbf{0} && (53)\\
B_{1,n}\hat{x} \leq& x && (54)\\
B_{1,m}\hat{x} \leq& y && (55)\\
-B_{1,n}\hat{x} \leq& -x + M(1-b) && (56)\\
-B_{1,m}\hat{x} \leq& -y + Mb && (57)
\end{aligned}
$$

So, the combined matrices G and h are:

$$
\underset{(nm+2n+2m)\times nm}{G} = \begin{bmatrix} -I_{\Pi} \\ B_{1,n} \\ B_{1,m} \\ -B_{1,n} \\ -B_{1,m} \end{bmatrix}, \quad \underset{(nm+2n+2m)\times 1}{h} = \begin{bmatrix} \mathbf{0} \\ x \\ y \\ -x + M(1-b) \\ -y + Mb \end{bmatrix} \tag{58}
$$

Note that in this formulation the source and sink vectors x and y contain only the volume/height values; their relative distances from the distance matrix are encoded already in the constant cost vector c.

3.3 Final Matrices

The final optimization problem as solved by the `CVXOPT` linear programming library can be summarized as the following:

$$
\begin{aligned}
\text{minimize} \quad & c^T\hat{x} \\
\text{subject to} \quad & G\hat{x} \leq h \\
& A\hat{x} = b
\end{aligned}
$$

where

$$
\underset{nm\times 1}{c} = \begin{bmatrix} d(Y_1, X_1) \\ d(Y_1, X_2) \\ \vdots \\ d(Y_2, X_1) \\ d(Y_2, X_2) \\ \vdots \\ d(Y_n, X_m) \end{bmatrix}, \quad \underset{(nm+2n+2m)\times nm}{G} = \begin{bmatrix} -I_\Pi \\ B_{1,n} \\ B_{1,m} \\ -B_{1,n} \\ -B_{1,m} \end{bmatrix}, \quad \underset{(nm+2n+2m)\times 1}{h} = \begin{bmatrix} \mathbf{0} \\ x \\ y \\ -x + M(1-b) \\ -y + Mb \end{bmatrix} \tag{59}
$$

The final transport solution is then contained in the optimization variable vector $\hat{x}$, which can be converted back to the full transport matrix Π by simply reshaping to $n \times m$ using row-major order.

4 Verification Results with `CVXOPT`

The following results verify the planner using the implementation for the `CVXOPT` linear programming library.

1. **Equal source and sink volumes**

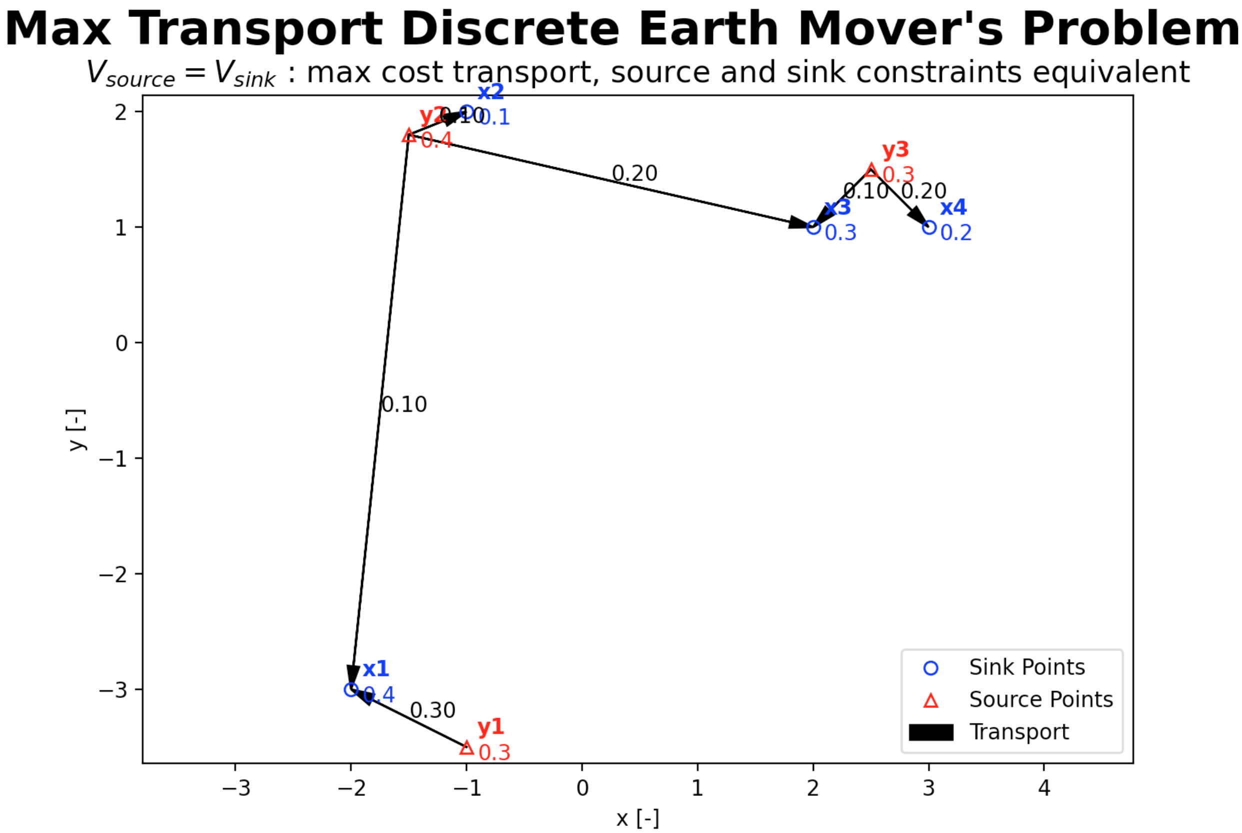
   The first case in Figure 2 verifies the original problem description, with equal source and sink volumes. Note that in this case the transport is at the maximum objective value, because all volume is moved.

2. **More source volume than sink volume**

   The second case in Figure 3 verifies the planner with more source volume than sink volume by removing node x_4. This means there will be left-over source volume that is not moved. The solution removes one of the long transports from y_2 to x_3 and leaves left-over volume at node y_2. The transport cost is also lower than for the equal-volume case, because some of the volume is not moved.

3. **More sink volume than source volume**

   The third case in Figure 4 verifies the planner with more sink volume than source volume by removing node y_3. This means there will be left-over sink volume that is not filled. The solution removes the long transport from y_2 to x_1 and instead sends the majority of y_2 volume to the closer node at x_3, leaving x_1 not completely filled. The node at x_4 is completely unfilled, because all the source volume is exhausted to closer nodes. Again, the transport cost is lower than for the equal-volume case, because not as much volume is moved overall.



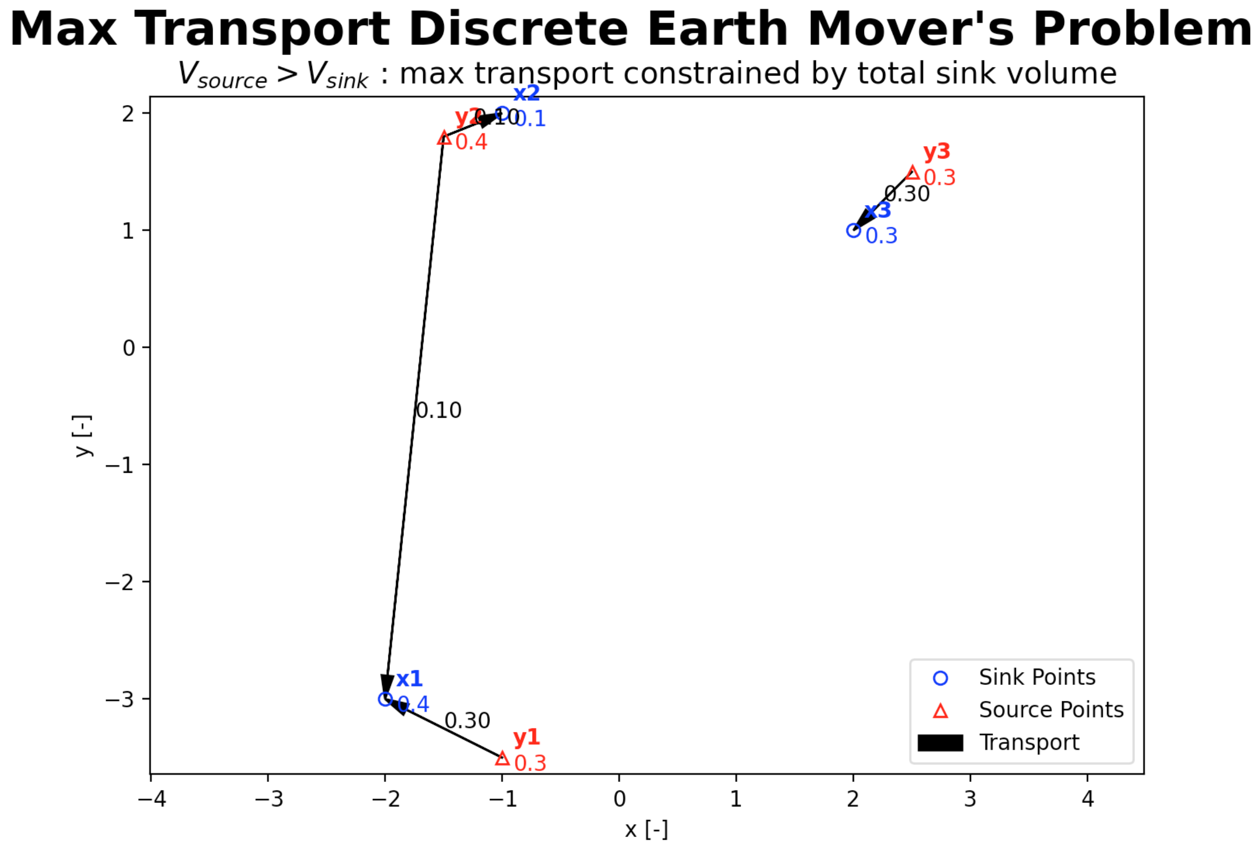
(a) Illustration of transport plan.

[INFO] Total transport cost: 1.80

	x1: 0.4	x2: 0.1	x3: 0.3	x4: 0.2
y1: 0.3	0.30	-0.00	-0.00	-0.00
y2: 0.4	0.10	0.10	0.20	0.00
y3: 0.3	0.00	-0.00	0.10	0.20

(b) Transport plan matrix.

Figure 2. Optimal transport plan solved with `CVXOPT` linear programming library for equal source and sink volumes.

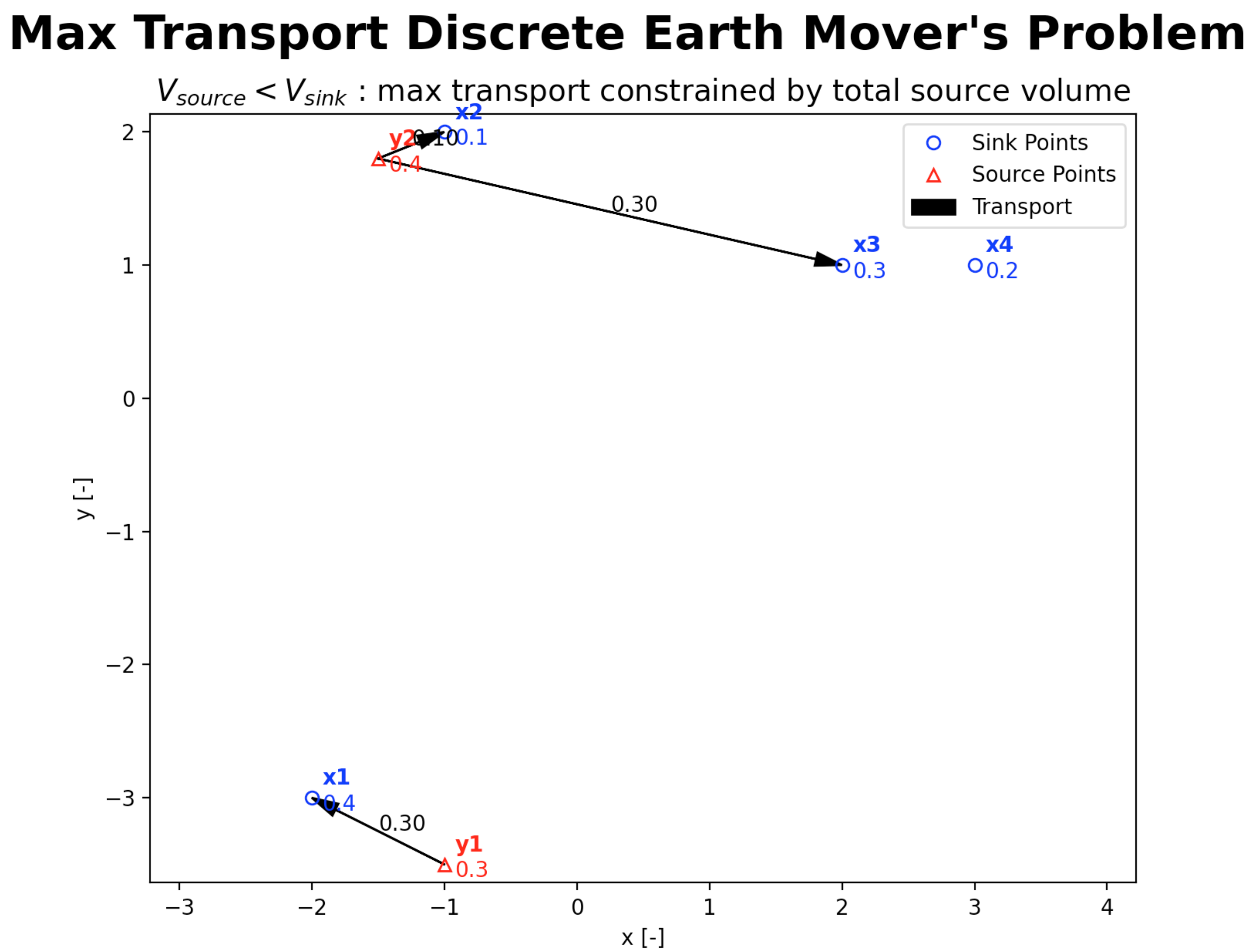


(a) Illustration of transport plan.

```
[INFO] Total transport cost: 1.08
+---------+---------+---------+---------+
|         | x1: 0.4 | x2: 0.1 | x3: 0.3 |
+---------+---------+---------+---------+
| y1: 0.3 |   0.30  |  -0.00  |  -0.00  |
| y2: 0.4 |   0.10  |   0.10  |   0.00  |
| y3: 0.3 |   0.00  |  -0.00  |   0.30  |
+---------+---------+---------+---------+
```

(b) Transport plan matrix.

Figure 3. Optimal transport plan solved with `CVXOPT` linear programming library for more source volume than sink volume.



(a) Illustration of transport plan.

[INFO] Total transport cost: 1.47

	x1: 0.4	x2: 0.1	x3: 0.3	x4: 0.2
y1: 0.3	0.30	-0.00	-0.00	-0.00
y2: 0.4	-0.00	0.10	0.30	0.00

(b) Transport plan matrix.

Figure 4. Optimal transport plan solved with `CVXOPT` linear programming library for more sink volume than source volume.