1 Notations

Given a set of points \( X = \{x_i \in \mathbb{R}\}_{i=1..m} \) and \( Y = \{y_j \in \mathbb{R}\}_{i=1..n} \) on the real line, \( m < n \), the goal is to find an injective assignment \( a : \mathbb{N} \rightarrow \mathbb{N} \) minimizing \( \sum (x_i - y_{a(i)})^2 \).

For the sake of simplicity, we assume that points of \( X \) (resp. \( Y \)) are distinct. All these results hold without this assumption by defining a total order on points with multiplicity (i.e. when \( x_i = x_j \)).

2 Warm-up

2.1 Miscellaneous

For convex cost, no such crossings can occur in the assign map (resolving the crossing will always lead to lower cost).

2.2 Injective nearest neighbor assignment

We consider the nearest neighbor assignment \( t : \mathbb{N} \rightarrow \mathbb{N} \) between \( X \) and \( Y \).

The nearest neighbor assignment can be obtained by a simple 2-sweep algorithm in \( O(n + m) \).

**Proposition 1.** If the \( t \) assignment \( X \rightarrow Y \) is injective, then the optimal assignment \( a \) is given by \( t \).

**Proof.** The nearest neighbor match already minimizes \( \sum (x_i - y_{t(i)})^2 \). Hence, if \( t \) is injective, it trivially corresponds to the optimal assignment \( a \). \( \square \)
3 Reducing the ranges of $X$ and $Y$

Let us consider the example:

Since $x_1 < y_1$, the assignment $x_1 \rightarrow y_1$ (thus setting $a(1) = 1$) is optimal since any other assignment of $x_1$ than $y_1$ would have higher cost. We can then proceed with $X = X \setminus \{x_1\}$ and $Y = Y \setminus \{y_1\}$:

We can repeat this process on both sides of $X$ and $Y$ ($x_1 < y_1$ on the left and $x_m > y_n$ on the right). We finally end up with a smaller optimal assignment problem:

The problem is not solved yet but reducing the ranges allows us to considerably reduce the size the problem using trivial optimal assignments.

**Proposition 2.** If $x_1 < y_1$, the optimal assignment of $X$ to $Y$ is given by setting $a(1) = 1$ and solving the optimal assignment of $X \setminus \{x_1\}$ to $Y \setminus \{y_1\}$. Similarly, if $x_m > y_n$, we can set $a(m) = n$ and solve the problem of $X \setminus \{x_m\}$ to $Y \setminus \{y_n\}$.

**Proof.** Let us prove the first assert, the proof for the second one is similar. By contradiction, let us assume that $x_1 \rightarrow y_k$ with $k > 1$ (i.e. $a(1) = k$). Since no crossing can occur in the optimal assignment, the point $y_1$ is not assigned to any point in $X$. Since $x_1 < y_1 < y_k$, $(x_1 - y_1)^2 < (x_1 - y_k)^2$ which contradicts the fact that the assignment as minimal cost. So we necessarily have $x_1 \rightarrow y_1$ in the optimal transport.

The algorithm iterates from the left to right until $x_i > y_i$, construct the assignment $\{x_j \rightarrow y_j\}$ of all $1 \leq j < i$, and performs the same from the right. This preprocessing step can be done in $O(m)$.

4 Case $n = m + 1$

We consider two assignments: The first one is from left to right, $A : (x_1, \ldots, x_m) \rightarrow (y_1, \ldots, y_m)$, and the second one is from right to left, $B : (x_1, \ldots, x_m) \rightarrow (y_2, \ldots, y_{m+1})$ (resp. in bold and dashed lines). At a point $x_i$, we define the two costs as

$$C_A(j) = \sum_{i=1}^{j} (x_i - y_i)^2 \quad \text{and} \quad C_B(j) = \sum_{i=m}^{j} (x_i - y_{i+1})^2.$$

We are looking for the point $y_j$ in $Y$ that is not assigned to any point in $X$ by the optimal assignment $a$. E.g.
Let \( y_j \) be such that
\[
j = \arg\min_{1 \leq i \leq m} (C_A(i) + C_B(i)).
\]

**Proposition 3.** The optimal assignment \( a \) is obtained by assignments from \( A \) on the left of \( y_j \) and from \( B \) on its right.

**Proof.** First, by definition of \( y_j \), \((C_A(j) + C_B(j))\) is exactly the cost of the optimal assignment we are looking for.

By contradiction, let suppose that we have a point \( x_k \) with \( k \geq j \) associated with some point \( y_{k'} \) with \( k' < j \). As the optimal assignment being a one-to-one and onto mapping from \( X \) to \( Y \setminus \{y_j\} \), there is a point \( x_l \) with \( l < k \) that is associated with some \( y_{l'} \) with \( l' > j \). By convexity of the cost function (Sec. 2.1), we have a contradiction since assignments \( x_k \rightarrow y_{k'} \) and \( x_l \rightarrow y_{l'} \) are crossing.

The algorithm is now very simple and can be done in \( O(m) \): For all \( j \), compute the \( C_A(j) \) and \( C_B(j) \), get the index \( j \) minimizing the sum of \( C_A(j) \) and \( C_B(j) \), construct \( a \).

Note: this algorithm is similar to a substep of the Hirschberg’s algorithm [?] to solve dynamic problems in quadratic time but linear space.

## 5 Reduction \( Y \) using \( m \)

We describe here a simple preprocessing that can reduce the set \( Y \) in \( O(\log n) \). Let us consider the following situation (the nearest neighbor assignment \( t \) is given in magenta):

We denote by \( y_l \) (resp. \( y_r \)) the nearest neighbor of \( x_1 \) (resp. of \( x_m \)). Then \( Y \) can be shrunk to obtain the following \( Y' \):

**Lemma 1.** In the optimal assignment, \( x_m \) cannot be associated to a point \( y_k \) with \( k < l \). Similarly, \( x_1 \) cannot be associated to a point \( y_k \) with \( k > r \).

**Proof.** Let us suppose that we have \( x_m \rightarrow y_k \) with \( k < l \) in the optimal assignment. If \( k < m \) we have a first contradiction (as no crossings may occur in the optimal map, the assignment must be injective between points of \( X \) and \( \{y_1 \ldots y_k\} \), which is only possible if \( k \geq m \)). If \( k \geq m \), we have a second contradiction on the cost of the assignment: Since \( t(1) = l \), \((x_1 - y_l)^2 \leq (x_m - y_l)^2\) (and by definition, \( x_1 < x_m \)), we have \( x_m \geq y_l \) and \((x_m - y_k)^2 = (x_m - y_l)^2 + (y_l - y_k)^2 > (x_m - y_l)^2\). Hence, the assignment \( x_m \rightarrow y_l \) would reduce the cost, which is a contradiction. The proof is similar to \( x_1 \) that cannot be associated to a point \( y_k \) with \( k > r \).

**Proposition 4.** The optimal assignment problem of \( X \) to \( Y \) can be reduced to an assignment problem of \( X \) to \( \{y_{\max(0, l-m+1)} \ldots y_{\min(n, r+m-1)}\} \).
Proof. Using Lemma 1, \( x_r \) is associated in the optimal assignment to some \( y_k \) with \( k \geq l \). Let us consider \( x_{r-1} \), it is necessarily associated \( y_{k'} \) with \( k' \geq l - 1 \). Indeed, by contradiction, let assume that \( x_r \to y_l \) and that \( x_{r-1} \to y_{k'} \) with \( k' < l - 1 \), since assignments cannot cross, points \{\( y_{k+1}, \ldots, y_{l-1} \}\) are not optimally assigned to any point in \( X \). But as \( t(r-1) > l \) (nearest neighbor assignments cannot cross too), we have \((x_{r-1} - y_{k'})^2 > (x_{r-1} - y_{l-1})^2\). So \( x_{r-1} \to y_{k'} \) is not optimal which contradicts the hypothesis. Similar arguments hold for points \( x_{r-2} \) to \( x_l \). Finally, we have that \( x_l \) is necessarily associated to a point \( y_{k'} \) with \( k' \geq \max(0, l - m + 1) \). Again, \( x_l \) is necessarily associated to a point \( y_{k'} \) with \( k' \leq \min(n, l + m - 1) \), which ends the proof. \( \square \)

This proposition can be used as a preprocessing step and only requires to have the nearest neighbor assignment \( t(1) \) and \( t(m) \) of \( x_1 \) and \( x_m \), which can be done in \( O(\log n) \). Note that or the other preprocessings, we may already have the complete nearest neighbor assignment in \( O(n + m) \) which makes this preprocessing be in \( O(1) \).

6 Splitting the problem (Algorithm 2)

We prove in this section the correctness of Algorithm 2. First, let us suppose that we always extend \( Y_{k'} \) on both sides (lines 16-17). Then, Algorithm 2 corresponds to Algorithm 1 where both options are applied (without the need of computing the option costs). Hence, by correctness of Algorithm 1 at each step of Alg.2, for each sub-problem \((X_k, Y_k)\), the optimal assignment of \( X_k \) will only consider points in \( Y_k \). Furthermore, by construction, all \{\( X_k \)\} form a disjoint partition of \( X \). Hence \{\( A_k \)\} sub-problems are independent and can be solved in parallel.

We now consider the special case of lines 13-14 whose objective is to only extend \( Y_k \) on one side, leading to shorter sub-problems. We thus are in the situation where the nearest neighbor of \( x_m', y_{t(m')}' \). Since \( y_{t(m')} \in Y_k \), \( f_{y_{t(m')}} \) is \textit{false} but there is no point in \( X_k \) whose nearest neighbor is \( t(m') \) (for \( k \leq m' \), nearest neighbors cannot cross so we only need to check \( t(m' - 1) \)):

\[
\begin{array}{c}
\text{x}_r \quad \text{x}_{m-1} \quad \text{x}_{m'} \\
\hline
\text{X}_k \quad \text{Y}_k \\
\hline
\text{s}_k \quad \text{f}_{m'} \quad \text{y}_{t(m')} \quad \ell_k
\end{array}
\]

We claim that we just need to extend \( Y_k \) with \( y_{t_k} \) instead of adding both \( y_{t_k} \) and \( y_{t_k} \), to obtain:

\[
\begin{array}{c}
\text{x}_r \quad \text{x}_{m-1} \quad \text{x}_{m'} \\
\hline
\text{X}_k \cup \{x_{m'}\} \quad \text{Y}_k \\
\hline
\text{s}_k \quad \text{f}_{m'} \quad \text{y}_{t(m')} \quad \ell_k
\end{array}
\]

Let us suppose by induction that \((X_k, Y_k)\) is a valid optimal assignment sub-problem, meaning that in the optimal assignment of \( X \) to \( Y \), all points if \( X_k \) are assigned to some points of \( Y_k \). We say that \( Y_k \) is \textit{tight} if \(|X_k| = |Y_k|\).

- Let suppose that \( Y_k \) is tight
  - Let us suppose that optimal assignment is the nearest neighbor one \((i.e. \ a = t)\).
    \[ \Rightarrow \] By hypothesis, we have \( t(m') \neq t(m' - 1) \), so \( t(m') \) is in fact greater or equal to \( \ell_k \). Hence \( x_{m'} \) can only be optimally assigned to \( y_{t_k} \) in the sub-problem \( X_k \cup \{x_{m'}\} \) to \( Y \) (the only free spot is \( y_{t_k} \) and for \( y_j \) with \( j > \ell_k \) the cost would be higher). So the sub-problem \((X_k \cup \{x_{m'}\}, Y_k \cup \{y_{t_k}\})\) is a valid sub-problem (we can just extend \( Y_k \) to its right).
Proposition 5. The optimal assignment of $X$ to $Y$ can be reduced to the optimal assignment problem of $X$ to $Y' = \{y_j\}_{j=\max(1, t(1)-p) \ldots \min(t(m)+p, n)}$.

Proof. The proof uses similar arguments as for Algorithm 2. Let us consider two sets $X$ and $Y$. If $m = 1$, the proposition is true, $a$ corresponds to the nearest neighbor assignment $t$. By induction, we suppose that the proposition is true for $X$ and $Y$, and we add an extra point $x_{m+1}$.

1. there exists $x_i$ in $X$ such that $t(i) = t(m+1)$ (i.e., we have a collision). As $t$ cannot cross, such $i$ is necessarily equal to $m$ (i.e., $t(m) = t(m+1)$ hereafter).

We define $p' := p + 1$. Hence, $p'$ is the new number on non-injectivity in $t$ for the problem ($X \cup \{x_{m+1}\}, Y$). Furthermore, if we denote $Y''$ the extension of $Y'$ by on point in $Y$ on both sides. From the correctness proof of Algorithm 2, we know that if the optimal assignment points of $X$ belongs to $Y'$, then $Y''$ contains the optimal assignments of points $X \cup \{x_{m+1}\}$. By construction, we have $Y'' = \{y_j\}_{j=\max(1, t(1)-p') \ldots \min(t(m)+1+p', n)}$ which proves the proposition by induction ($t(1)$ remains the same and $t(m+1) = t(m)$).

2. There is no such $x_i$. This means that the non-injectivity counter $p'$ when considering $x_m$ equals $p$. Furthermore, as no crossing occurs in $t$, we necessarily have $t(m+1) = \min(t(m)+1, n)$. By induction, the optimal assignments $a$ of $X$ to $Y$ consider points in $Y''$. From Algorithm 2, we know that extending $Y'$ by one only on its right, denotes $Y''$, contains the optimal assignment of $X \cup \{x_{m+1}\}$ (we are in the case of lines 13-14). As $t(1)$ is unchanged and $t(m+1) = \min(t(m)+1, n)$, we have $Y'' = \{y_j\}_{j=\max(1, t(1)-p', \min(t(m)+1+p', n))}$, and $Y''$ contains the optimal assignments of $X \cup \{x_m\}$, which completes the proof by induction.

7 Reduction $Y$ using non-injectivity counters

We prove the reduction of $Y$ using the number of times the nearest neighbor assignment $t$ between $X$ and $Y$ is non-injective.

Let $p = \text{card}\{t(i) = t(i+1) \land i < m\}$. Then,

Proposition 5. The optimal assignment of $X$ to $Y$ can be reduced to the optimal assignment problem of $X$ to $Y' = \{y_j\}_{j=\max(1, t(1)-p) \ldots \min(t(m)+p, n)}$.

Proof. The proof uses similar arguments as for Algorithm 2. Let us consider two sets $X$ ($|X| = m$) and $Y$. If $m = 1$, the proposition is true, $a$ corresponds to the nearest neighbor assignment $t$. By induction, we suppose that the proposition is true for $X$ and $Y$, and we add an extra point $x_{m+1}$.

- If $t(m+1) > \min(t(m) + p, n)$, there is no collision in the nearest neighbor assignment of $X \cup \{x_{m+1}\}$ to $Y$. Thus, $p$ remains the same, $x_{m+1}$ is optimally assigned to $y_{t(m+1)}$ (any other assignment of $x_{m+1}$ would have higher cost). Hence, the optimal assignment of $X \cup \{x_{m+1}\}$ to $Y$ can be reduced to $Y' = \{y_j\}_{j=\max(1, t(1)-p) \ldots \min(t(m)+1+p, n)}$.
- Otherwise, we have two possibilities.

1. There exists $x_i$ in $X$ such that $t(i) = t(m+1)$ (i.e., we have a collision). As $t$ cannot cross, such $i$ is necessarily equal to $m$ (i.e., $t(m) = t(m+1)$ hereafter).

We define $p' := p + 1$. Hence, $p'$ is the new number on non-injectivity in $t$ for the problem ($X \cup \{x_{m+1}\}, Y$). Furthermore, if we denote $Y''$ the extension of $Y'$ by on point in $Y$ on both sides. From the correctness proof of Algorithm 2, we know that if the optimal assignment points of $X$ belongs to $Y'$, then $Y''$ contains the optimal assignments of points $X \cup \{x_{m+1}\}$. By construction, we have $Y'' = \{y_j\}_{j=\max(1, t(1)-p') \ldots \min(t(m)+1+p', n)}$ which proves the proposition by induction ($t(1)$ remains the same and $t(m+1) = t(m)$).

2. There is no such $x_i$. This means that the non-injectivity counter $p'$ when considering $x_m$ equals $p$. Furthermore, as no crossing occurs in $t$, we necessarily have $t(m+1) = \min(t(m)+1, n)$. By induction, the optimal assignments $a$ of $X$ to $Y$ consider points in $Y''$. From Algorithm 2, we know that extending $Y'$ by one only on its right, denotes $Y''$, contains the optimal assignment of $X \cup \{x_{m+1}\}$ (we are in the case of lines 13-14). As $t(1)$ is unchanged and $t(m+1) = \min(t(m)+1, n)$, we have $Y'' = \{y_j\}_{j=\max(1, t(1)-p') \ldots \min(t(m)+1+p', n)}$, and $Y''$ contains the optimal assignments of $X \cup \{x_m\}$, which completes the proof by induction.