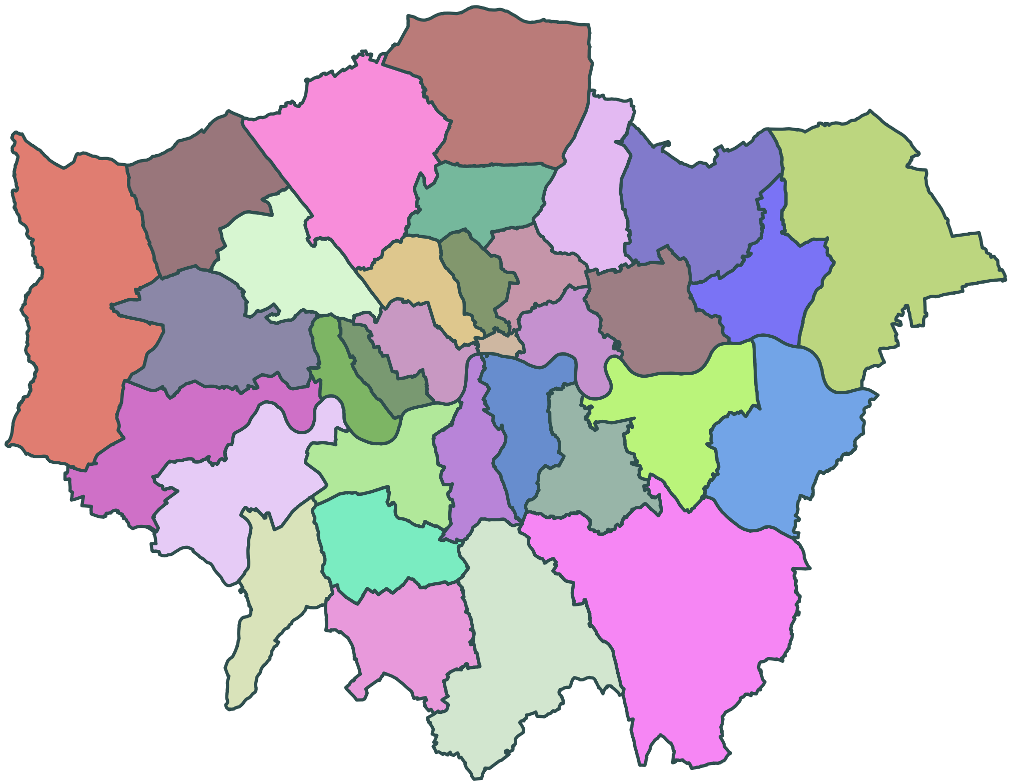
One-to-one Point Set Matchings for Grid Map Layout

David Eppstein
Marc van Kreveld
Bettina Speckmann
Frank Staals

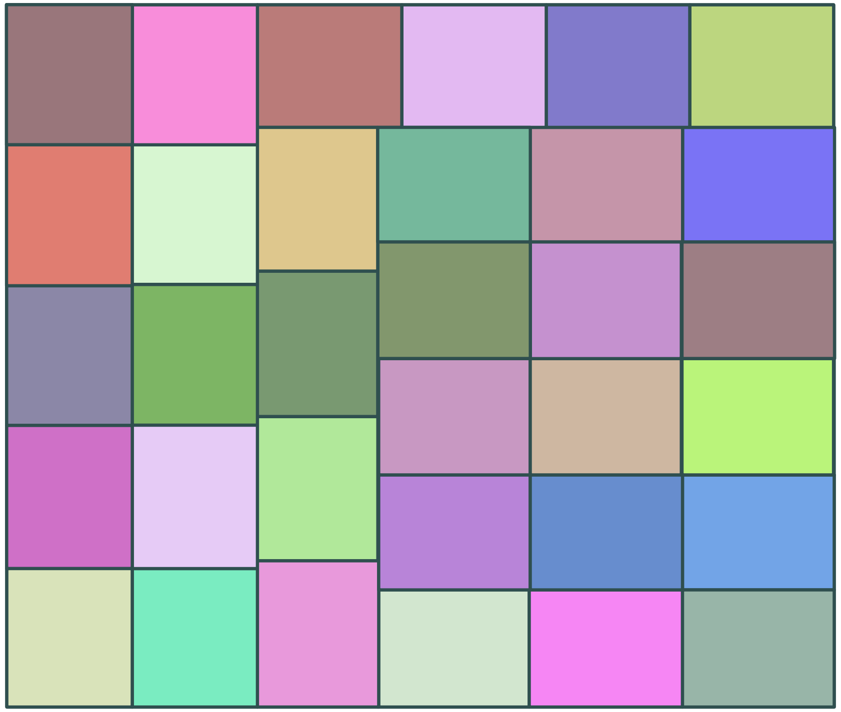
University of California, Irvine,
Utrecht University, TU Eindhoven

Visualising Geographic Data



Given a map with n regions we want to visualise some data for each region.

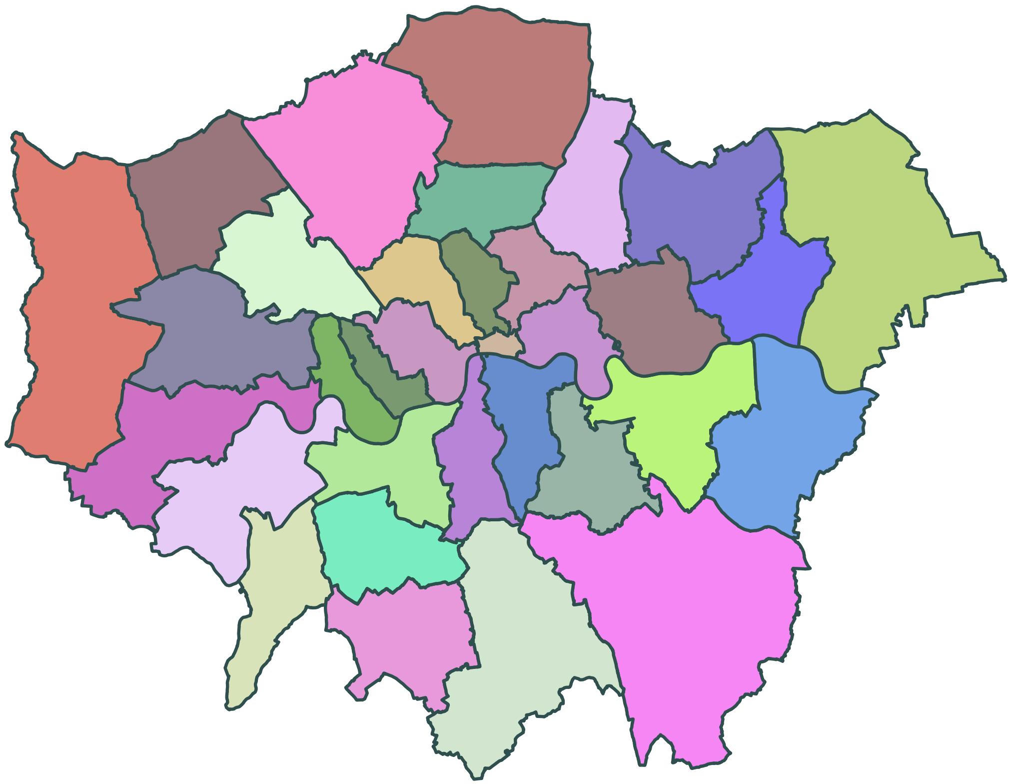
Visualising Geographic Data

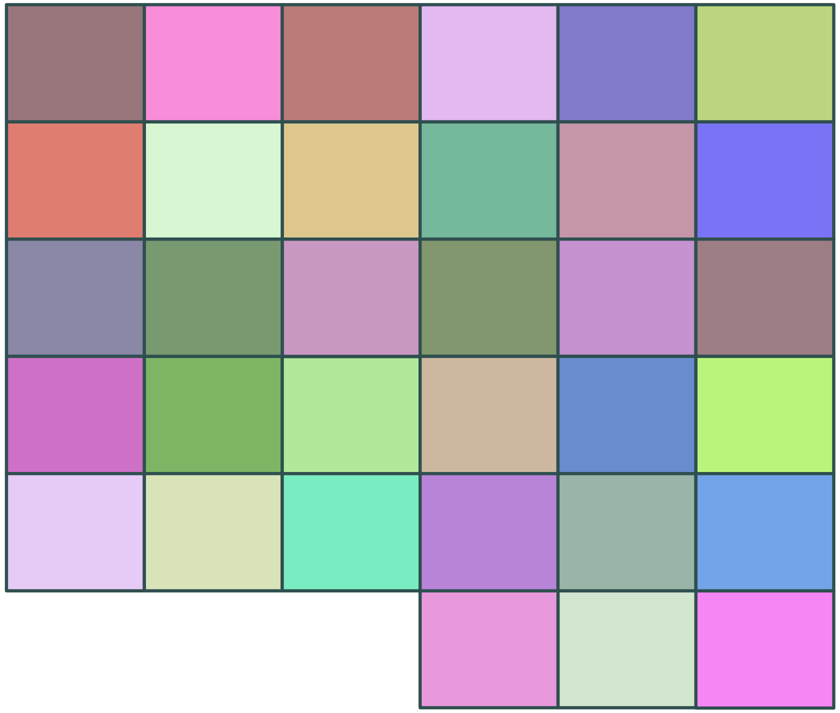


Given a map with n regions we want to visualise some data for each region.

Options: Symbol map, Cartogram, Spatial Treemap (Wood and Dykes 2008)

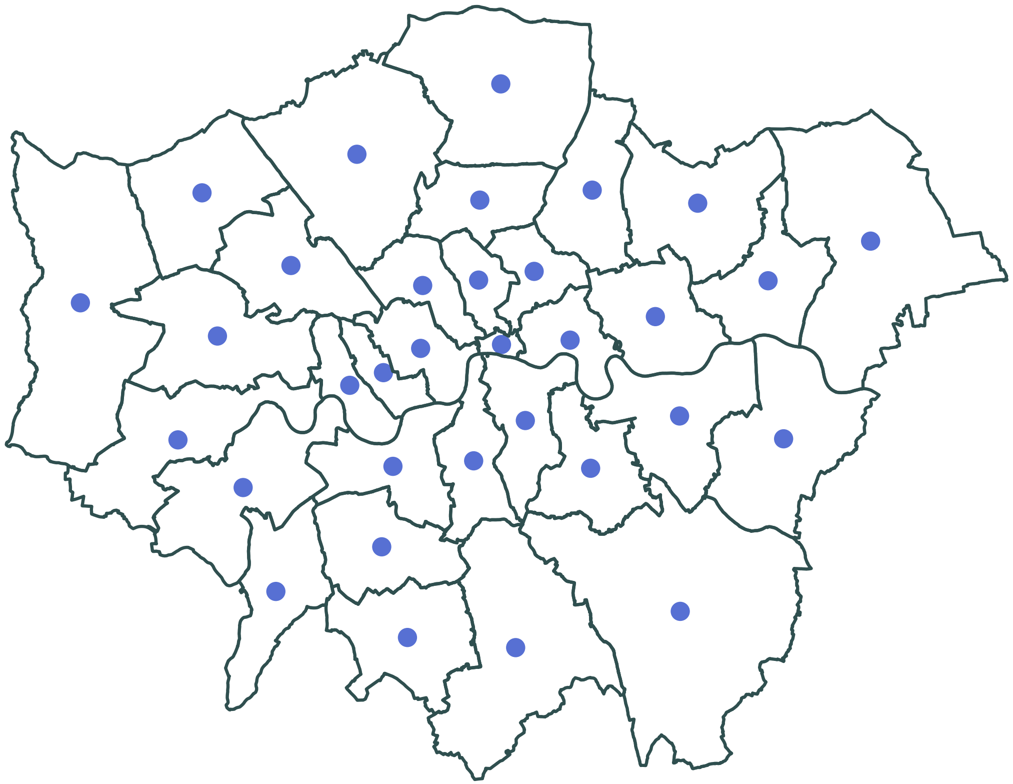
Visualising Geographic Data

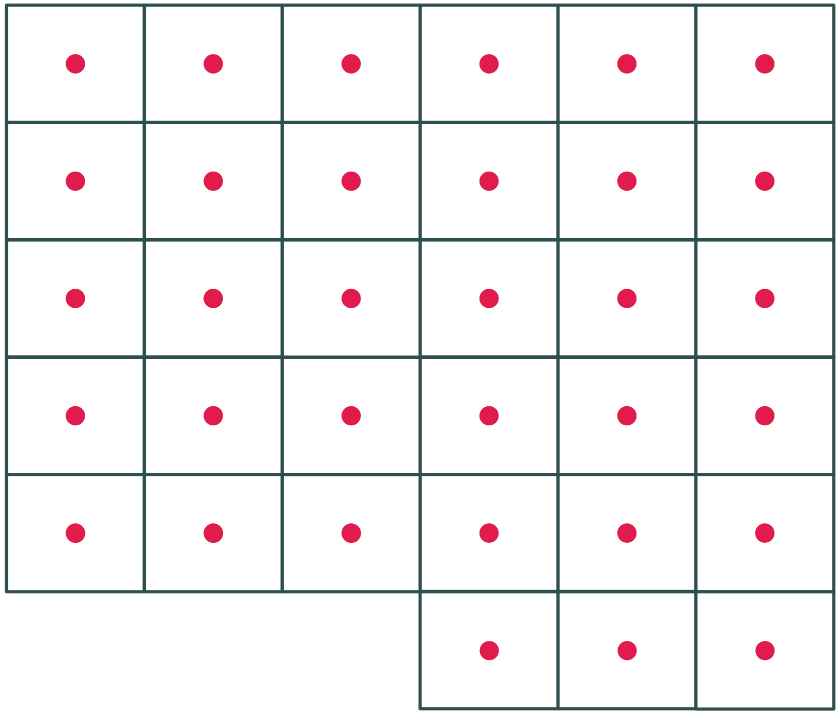




Given a map with n regions we want to visualise some data for each region.

Visualising Geographic Data



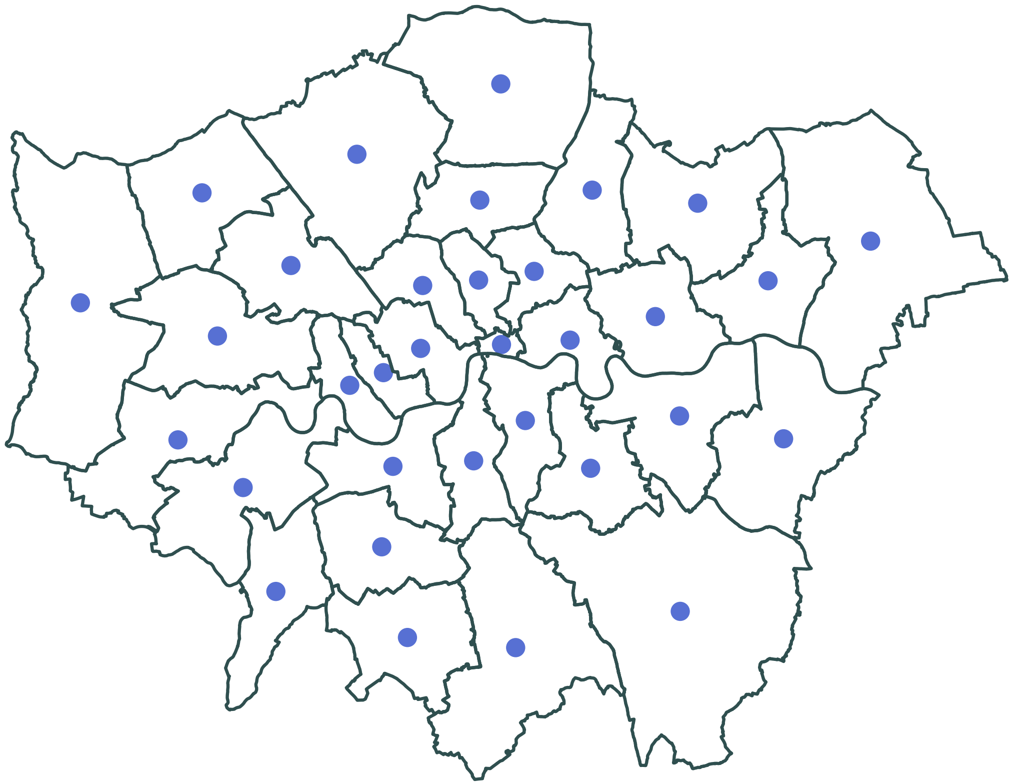


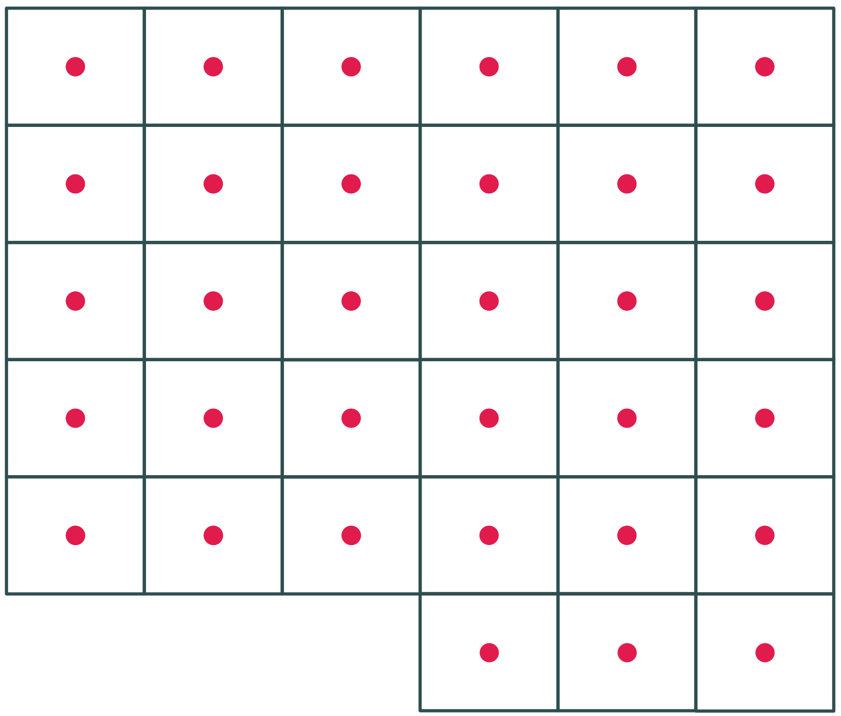
One-to-one Point Set Matching Problem

Represent the regions by a set A blue points.

Represent the grid by a set B blue points.

Visualising Geographic Data





One-to-one Point Set Matching Problem

Represent the regions by a set A blue points.

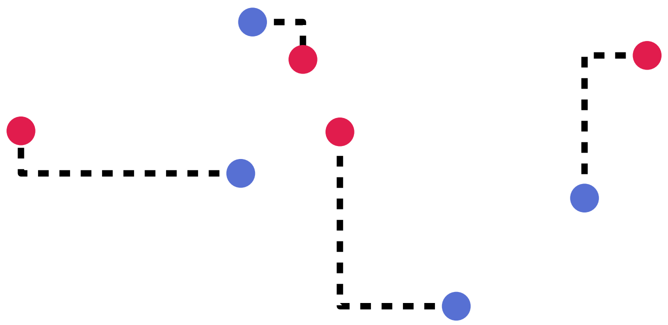
Represent the grid by a set B blue points.

Goal: find the best matching $\phi : A \to B$

Optimisation Criteria

What is the “best” matching?

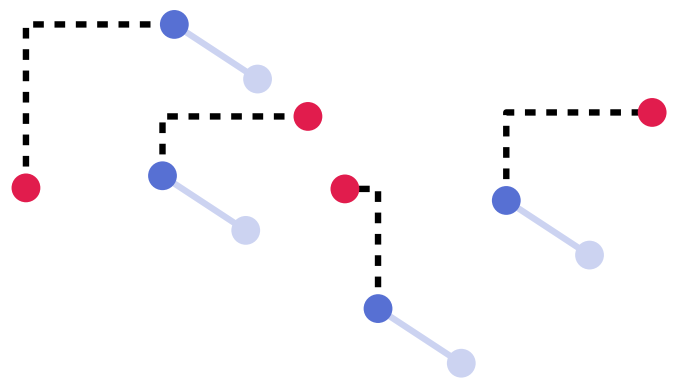
- Minimise the total L_1 distance.



Optimisation Criteria

What is the "best" matching?

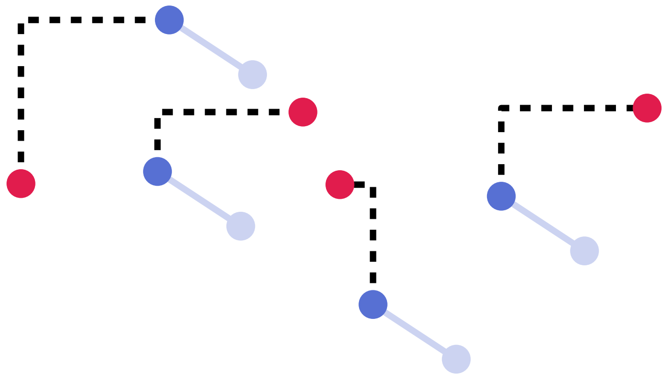
- Minimise the total L_1 distance.



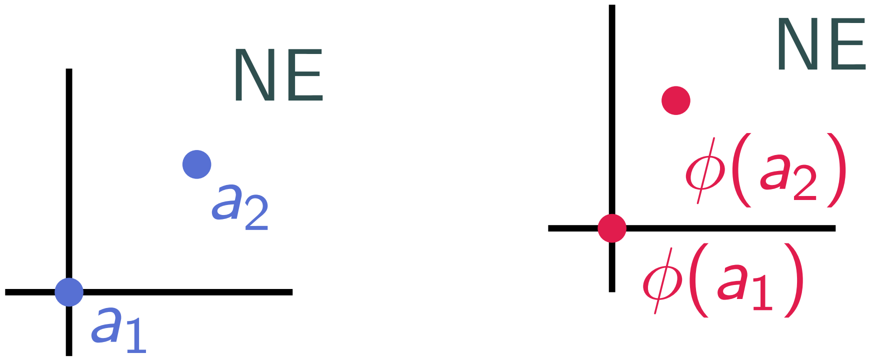
Optimisation Criteria

What is the "best" matching?

- Minimise the total L_1 distance.



- Maximise the number of pairs with the correct directional relation.



Minimising L_1-distance

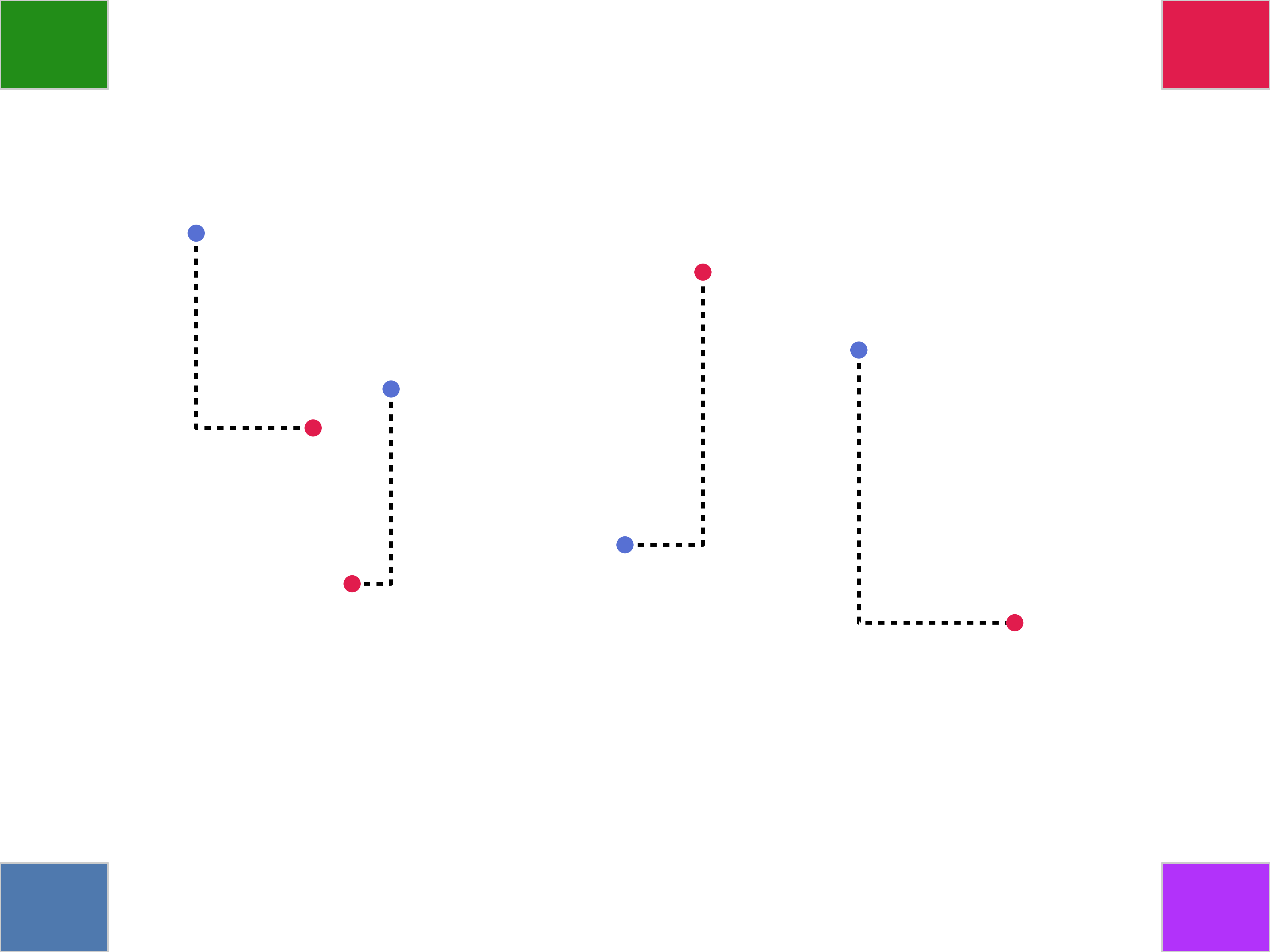
We want to find a matching ϕ^*, translation t^*, and scaling λ^* that minimise

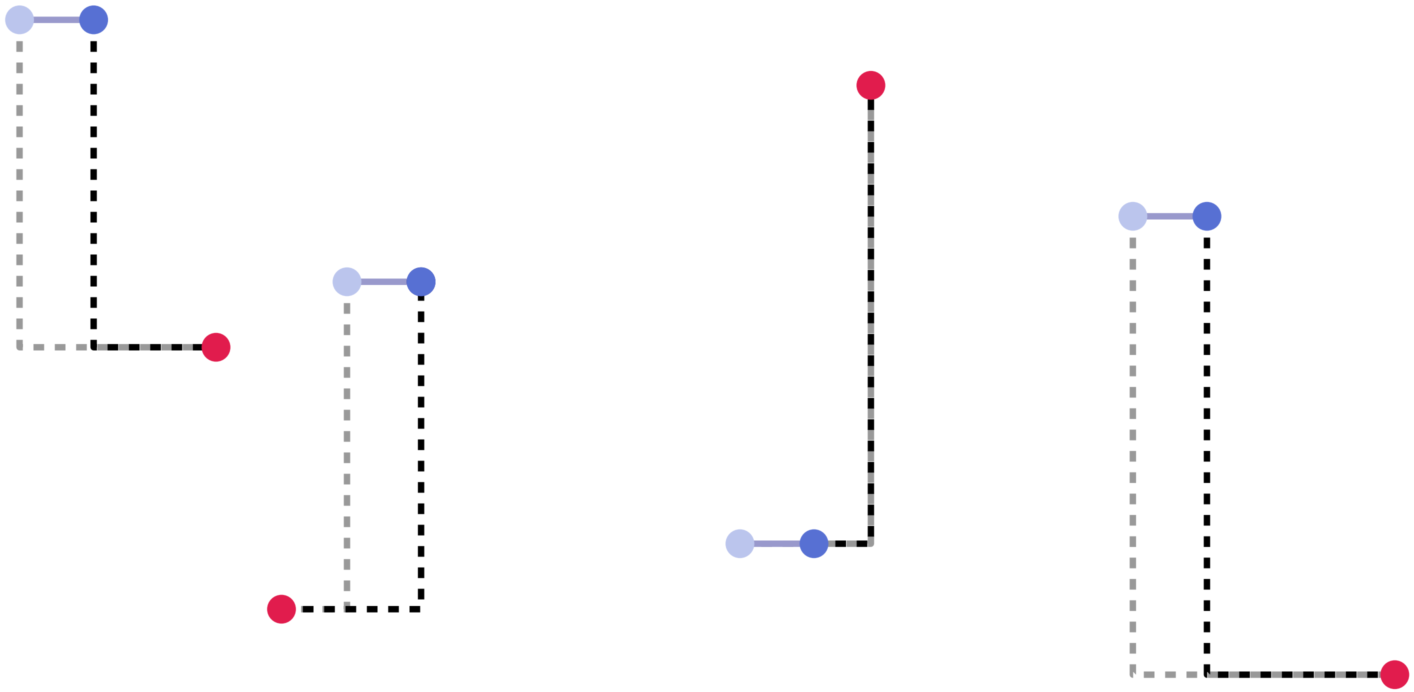
$$D(\phi, t, \lambda) = \sum_{a \in A} d(\lambda a + t, \phi(a)).$$

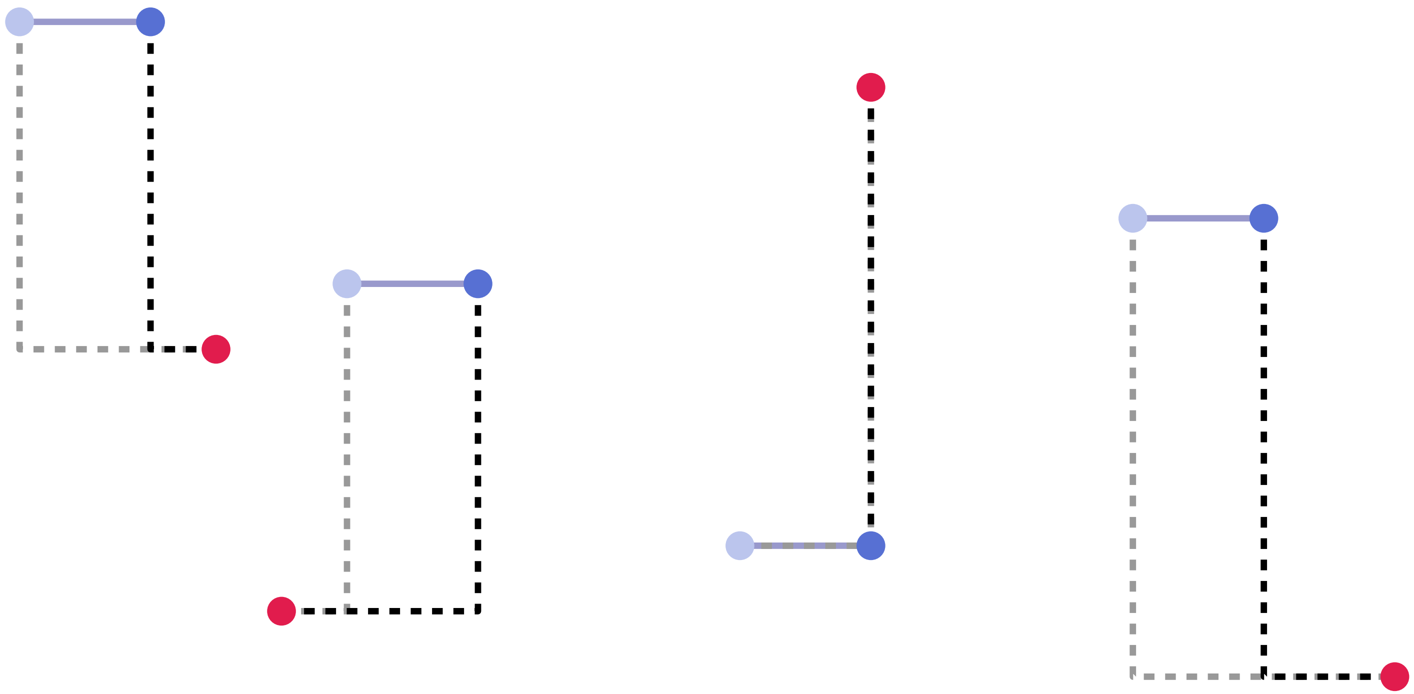
Minimising L_1-distance

We want to find a matching ϕ^* and translation t^* that minimise

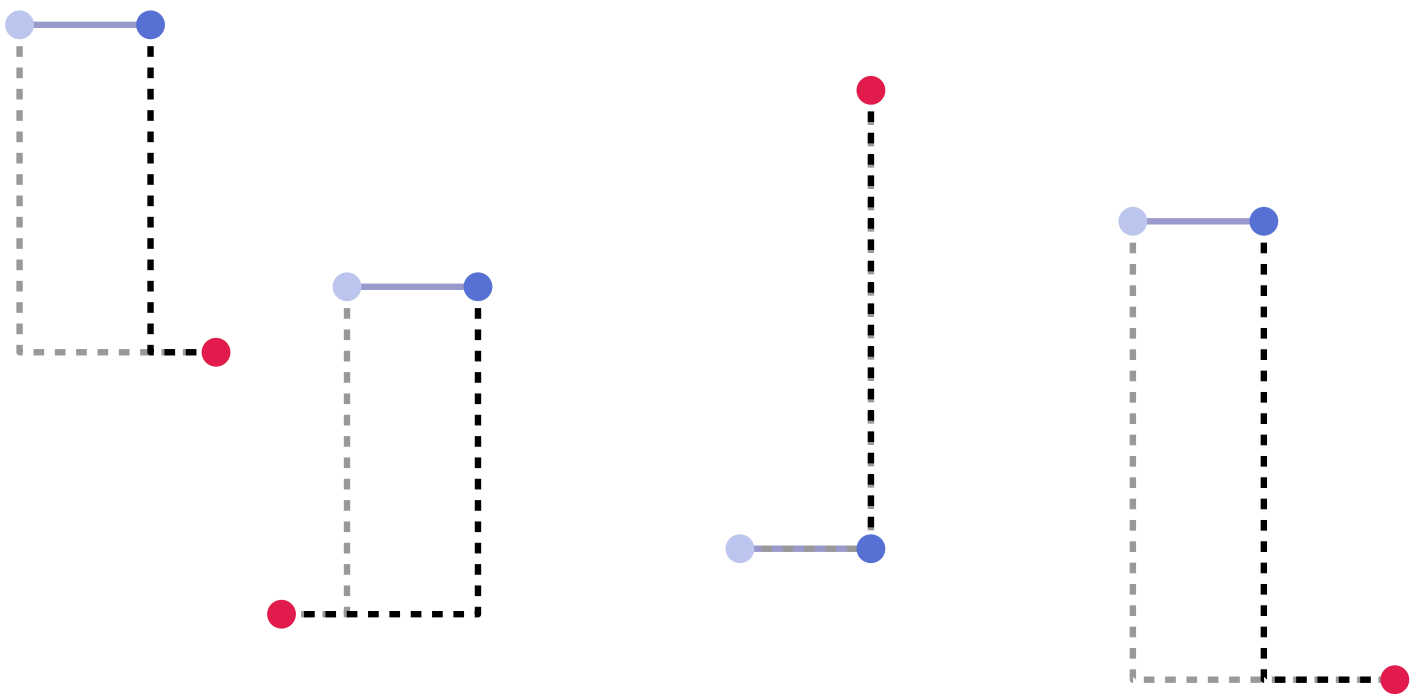
$$D_{\mathcal{T}}(\phi, t) = \sum_{a \in A} d(a + t, \phi(a)).$$







Aligning A and B decreases $D_{\mathcal{T}}$



Lemma 1. *For any matching ϕ, there is a t that x-aligns A and B and minimises $D_{\mathcal{T}}(\phi, \cdot)$.*

Minimising $D_{\mathcal{T}}$

There is an optimal matching at an x-alignment.

Same trick for y-alignment.

Minimising $D_{\mathcal{T}}$

There is an optimal matching at an x-alignment.

Same trick for y-alignment.

There is an optimal matching at an x- and y-alignment.

$\implies$ There are at most n^4 such alignments.

Minimising $D_{\mathcal{T}}$

There is an optimal matching at an x-alignment.

Same trick for y-alignment.

There is an optimal matching at an x- and y-alignment.

$\implies$ There are at most n^4 such alignments.

Theorem 1. *A ϕ^* and t^* that minimise $D_{\mathcal{T}}$ can be computed in $O(n^4 \cdot n^2 \log^3 n) = O(n^6 \log^3 n)$ time.*

Uses the matching algorithm by Vaidya (1988)

Minimising D_Λ and D

Minimum distance matching under scaling?

Use exactly the same approach.

Minimising D_Λ and D

Minimum distance matching under scaling?

Use exactly the same approach.

Minimum distance matching under translation and scaling?

Minimising D_Λ and D

Minimum distance matching under scaling?

Use exactly the same approach.

Minimum distance matching under translation and scaling?

Same idea: x-align (y-align) two pairs of points.

Minimising D_Λ and D

Minimum distance matching under scaling?

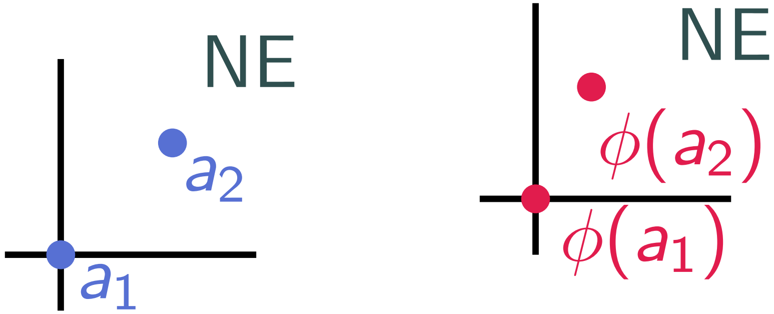
Use exactly the same approach.

Minimum distance matching under translation and scaling?

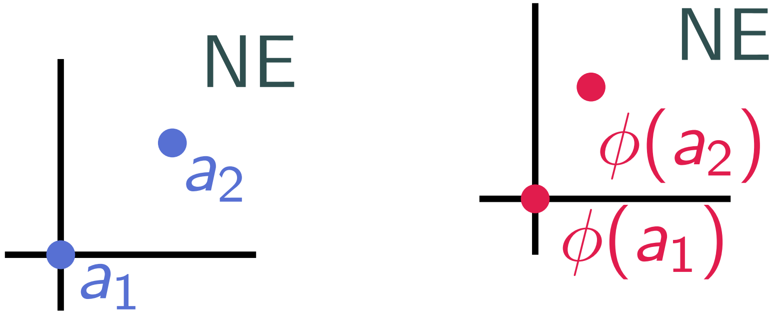
Same idea: x-align (y-align) two pairs of points.

Theorem 2. *A ϕ^*, t^*, and λ^* that minimise D can be computed in $O(n^8 \cdot n^2 \log^3 n) = O(n^{10} \log^3 n)$ time.*

Preserving directional relations



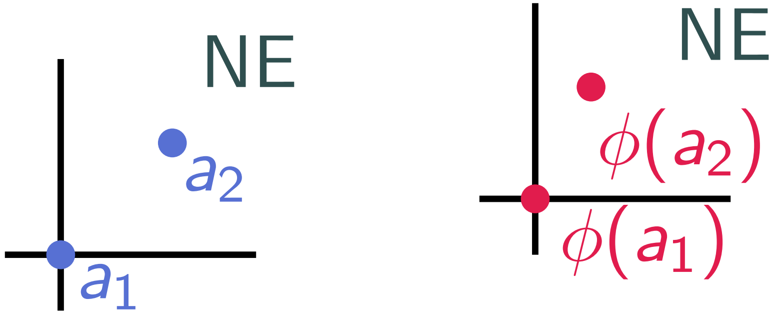
Preserving directional relations



Minimising the number of out-of-order pairs

$$W(\phi) = |\{(a_1, a_2) \mid (a_1, a_2) \in A \times A \wedge dir(a_1, a_2) \neq dir(\phi(a_1), \phi(a_2))\}|.$$

Preserving directional relations



Minimising the number of out-of-order pairs

$$W(\phi) = |\{(a_1, a_2) \mid (a_1, a_2) \in A \times A \wedge \\ dir(a_1, a_2) \neq dir(\phi(a_1), \phi(a_2))\}|.$$

Translation and scaling do not influence W.

A 4-approximation algorithm

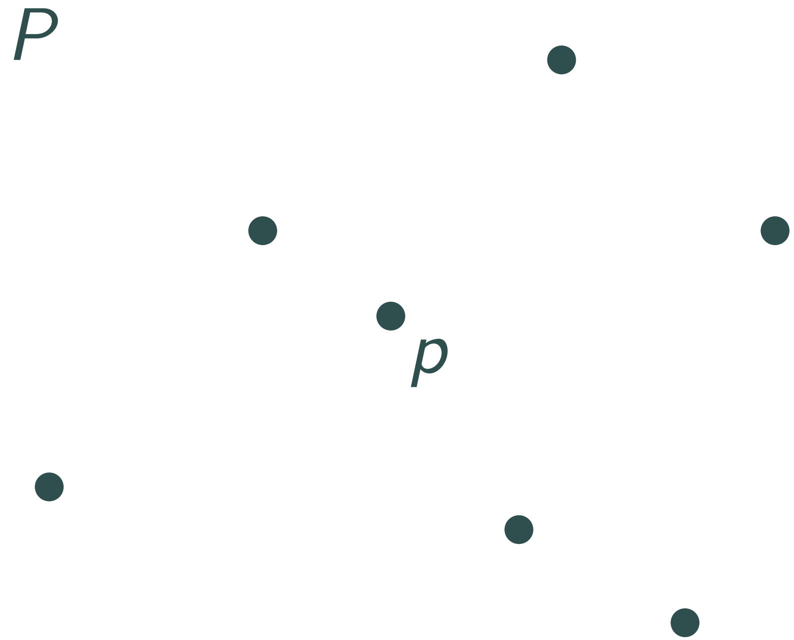
Compute a minimum distance matching with distance measure

$$w(a, b) = |x\text{-}rank_A(a) - x\text{-}rank_B(b)| + |y\text{-}rank_A(a) - y\text{-}rank_B(b)|.$$

A 4-approximation algorithm

Compute a minimum distance matching with distance measure

$$w(a, b) = |x\text{-}rank_A(a) - x\text{-}rank_B(b)| + |y\text{-}rank_A(a) - y\text{-}rank_B(b)|.$$



$x\text{-}rank_P(p) = 3$

$y\text{-}rank_P(p) = 4$

A 4-approximation algorithm

Compute a minimum distance matching with distance measure

$$w(a, b) = |x\text{-}rank_A(a) - x\text{-}rank_B(b)| + |y\text{-}rank_A(a) - y\text{-}rank_B(b)|.$$

$w(a, b)$ is the L_1-distance in terms of ranks.

A 4-approximation algorithm

Compute a minimum distance matching with distance measure

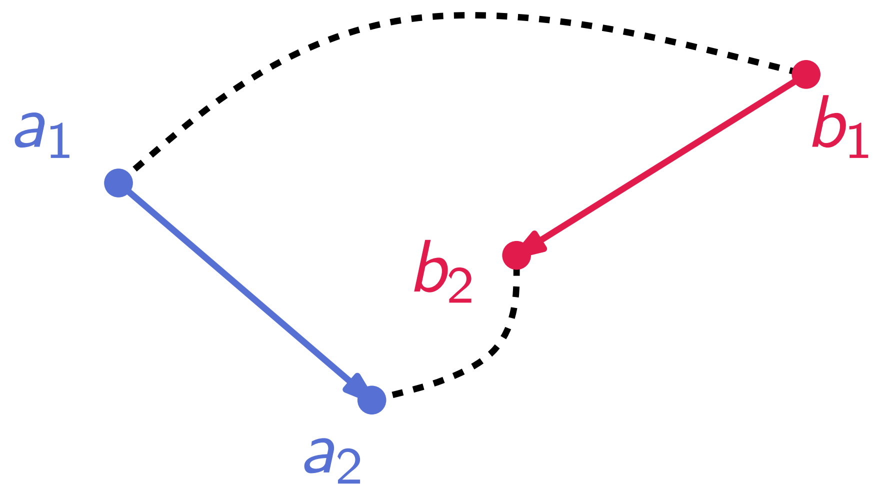
$$w(a, b) = |x\text{-}rank_A(a) - x\text{-}rank_B(b)| + |y\text{-}rank_A(a) - y\text{-}rank_B(b)|.$$

$w(a, b)$ is the L_1-distance in terms of ranks.

So compute an optimal matching using Vaidya's Algorithm.

Theorem 3. *A 4-approximation for minimising W can be computed in $O(n^2 \log^3 n)$.*

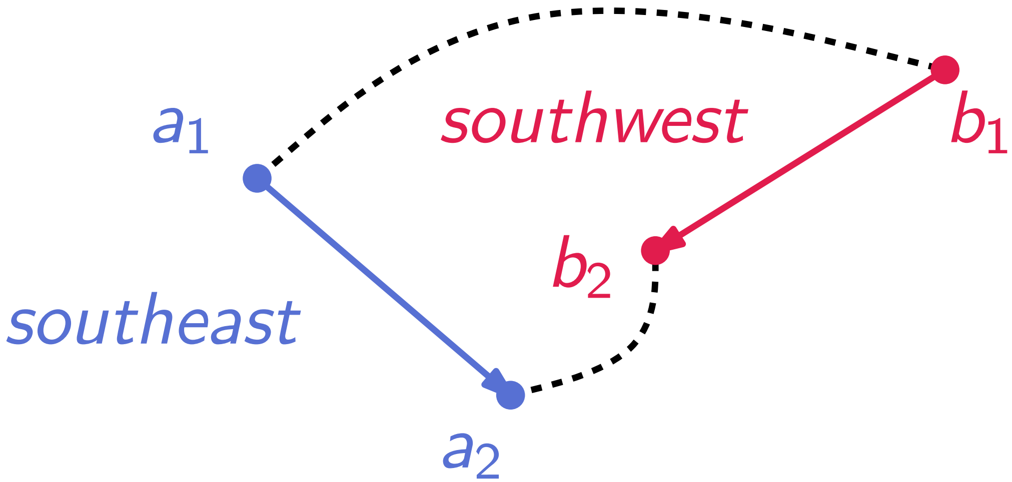
Inversions vs Directions



$x\text{-}rank_A(a_1) < x\text{-}rank_A(a_2)$ and
$x\text{-}rank_B(b_1) > x\text{-}rank_B(b_2)$

(a_1, a_2) is an inversion.

Inversions vs Directions



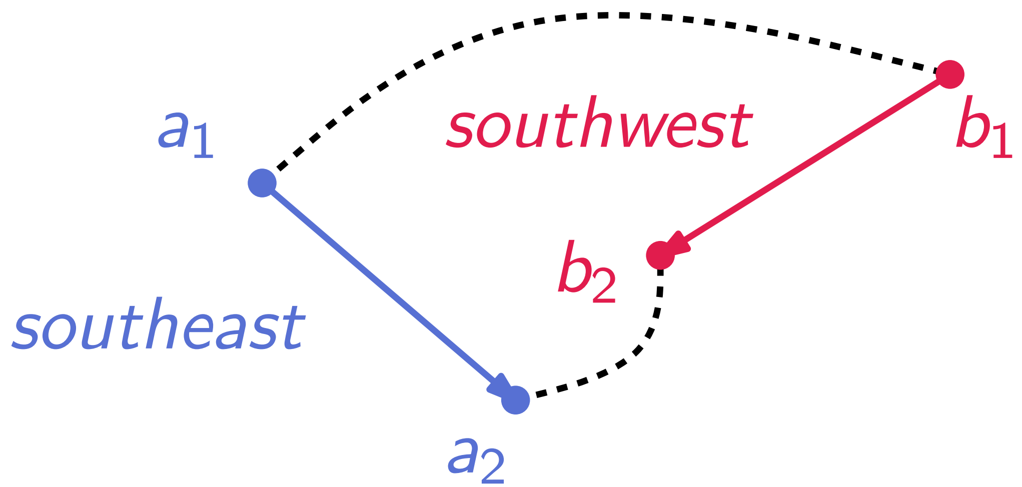
$x\text{-}rank_A(a_1) < x\text{-}rank_A(a_2)$ and
$x\text{-}rank_B(b_1) > x\text{-}rank_B(b_2)$

(a_1, a_2) is an inversion.

$\Updownarrow$

(a_1, a_2) is an out-of-order pair

Inversions vs Directions



$x\text{-}rank_A(a_1) < x\text{-}rank_A(a_2)$ and
$x\text{-}rank_B(b_1) > x\text{-}rank_B(b_2)$

(a_1, a_2) is an inversion.

$$\Updownarrow$$

(a_1, a_2) is an out-of-order pair

So $W(\phi) = \#\text{inversions} = I(\phi)$.

Inversions vs Ranks

Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.

Inversions vs Ranks

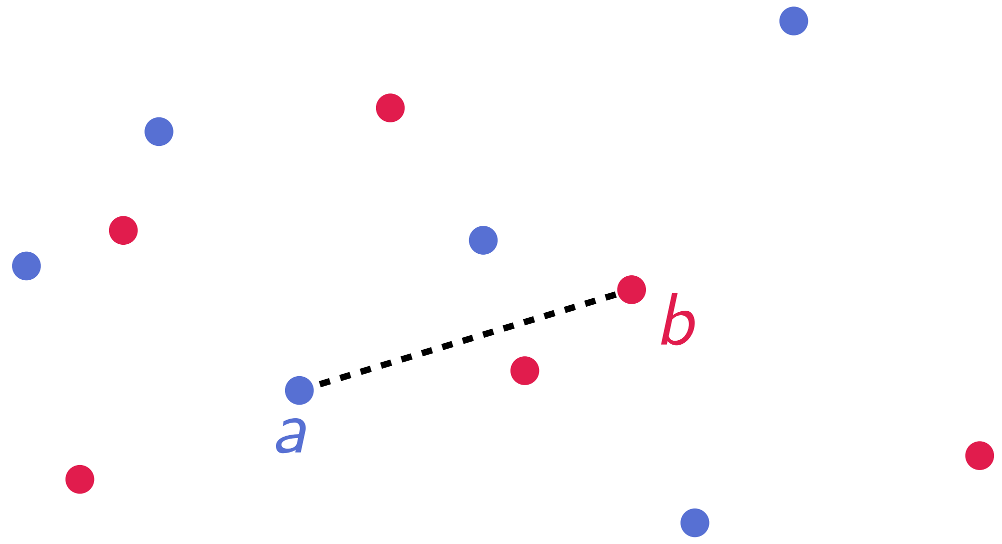
Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.

Lemma 3. $I_y(\phi) \leq Y(\phi) \leq 2I_y(\phi)$.

This leads to a 4-approximation algorithm.

Inversions vs Ranks

Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.

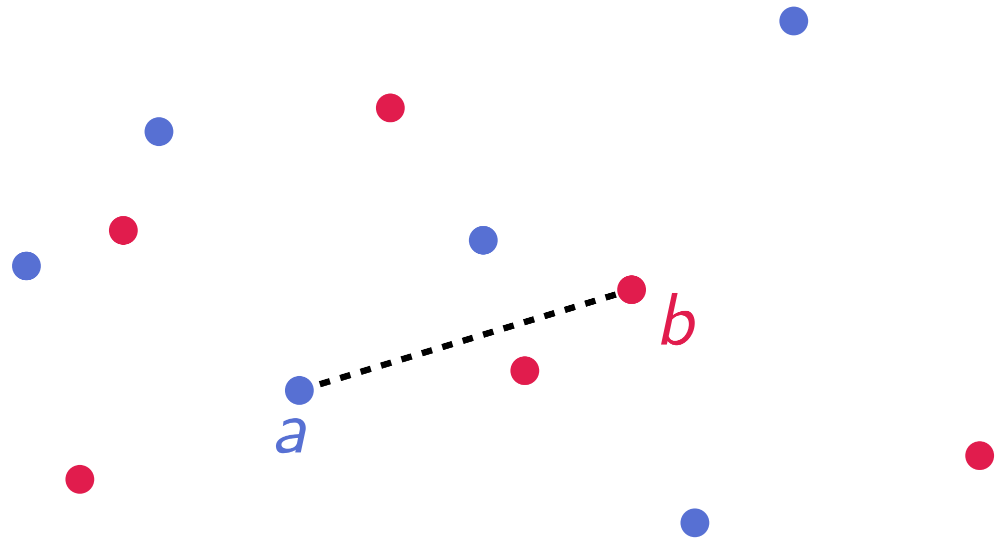


$$x\text{-}rank_A(a) = i = 3$$

$$x\text{-}rank_B(b) = j = 5$$

Inversions vs Ranks

Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.



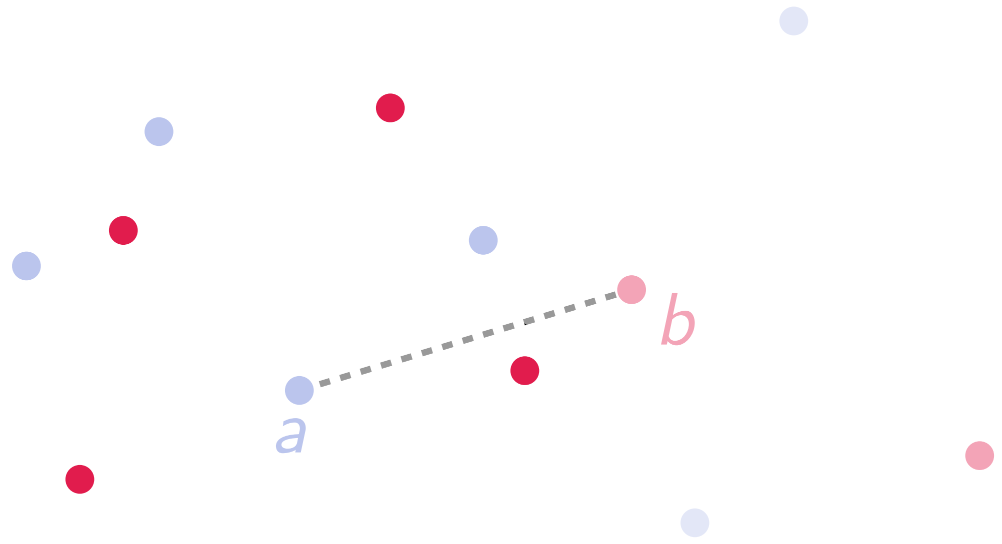
$x\text{-}rank_A(a) = i = 3$

$x\text{-}rank_B(b) = j = 5$

The matching has at least $j - i$ x-inversions.

Inversions vs Ranks

Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.



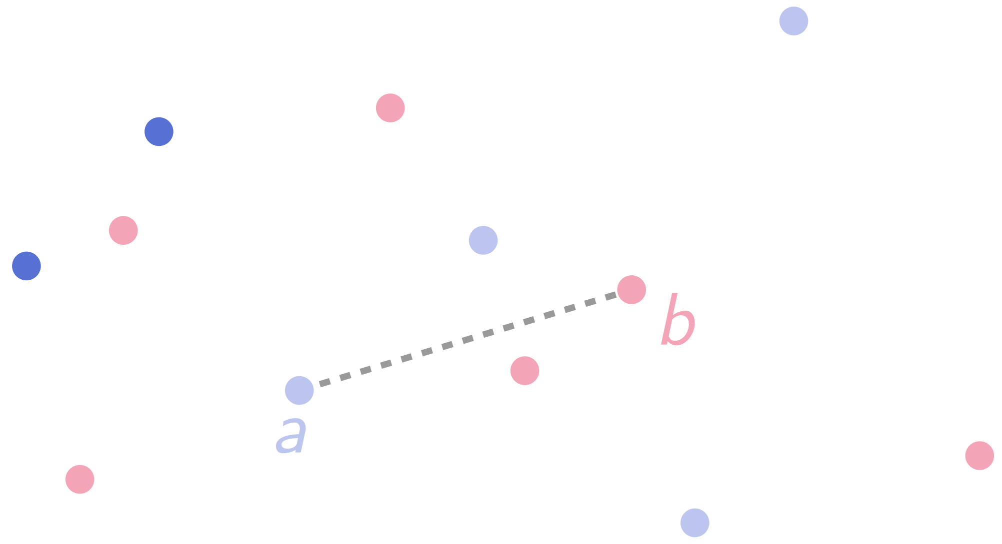
$x\text{-}rank_A(a) = i = 3$

$x\text{-}rank_B(b) = j = 5$

The matching has at least $j - i$ x-inversions.

Inversions vs Ranks

Lemma 2. $I_x(\phi) \leq X(\phi) \leq 2I_x(\phi)$.

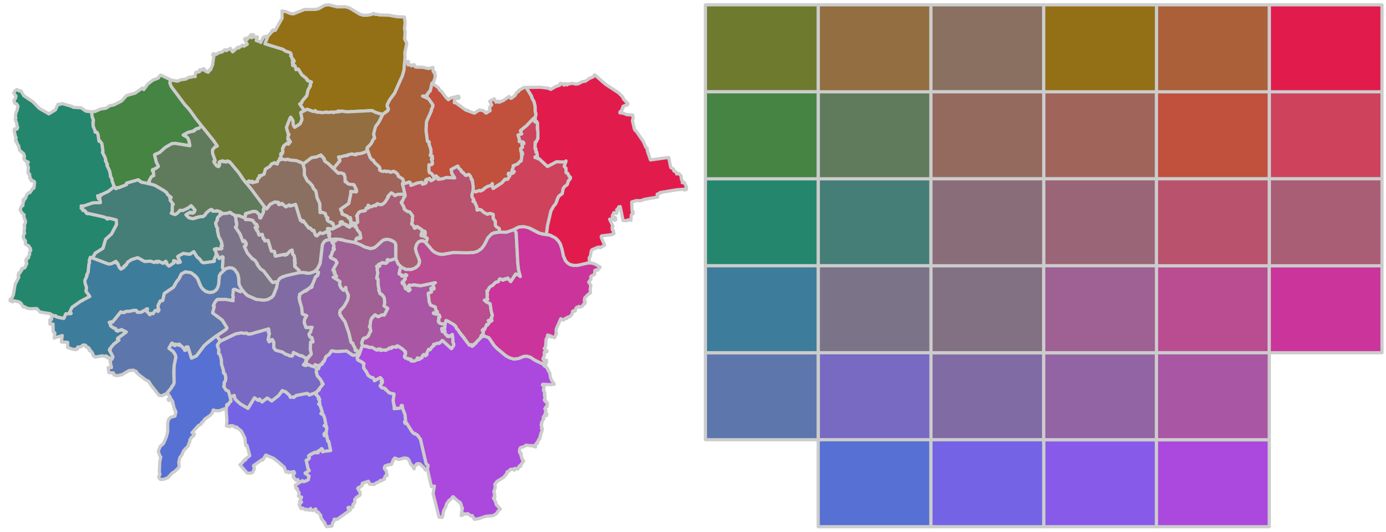


$x\text{-}rank_A(a) = i = 3$

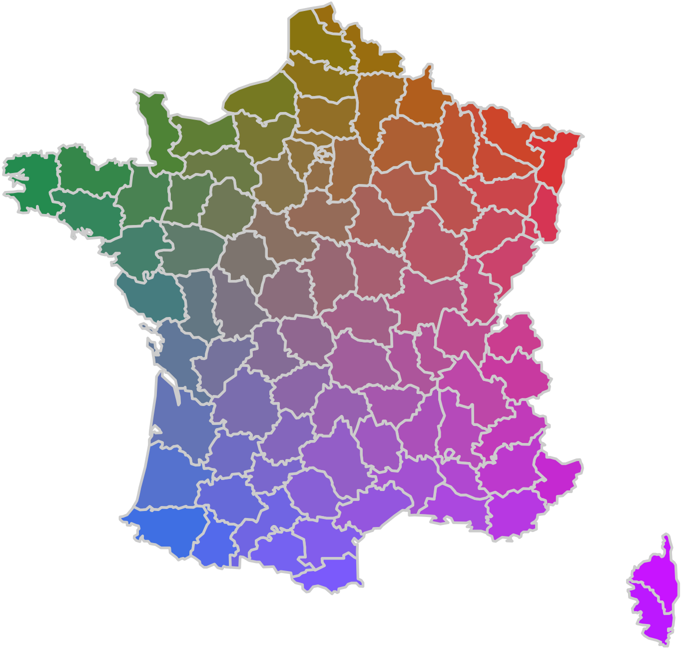
$x\text{-}rank_B(b) = j = 5$

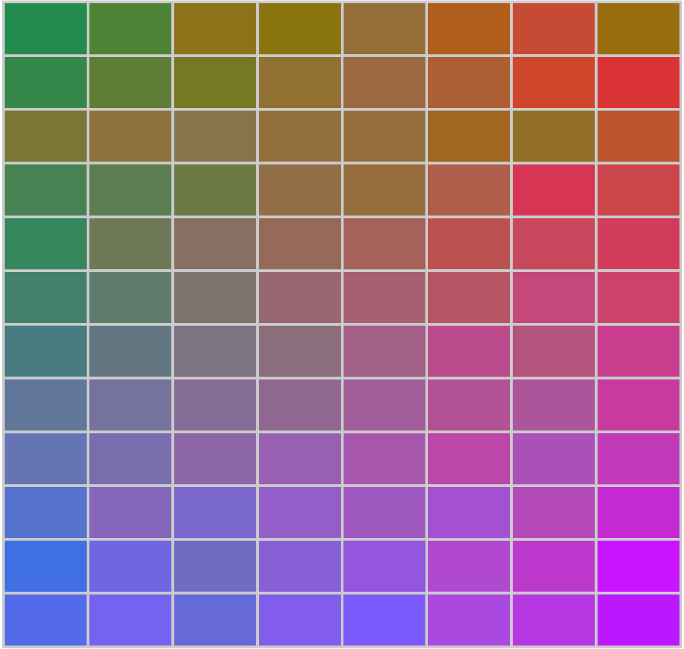
The matching has at least $j - i$ x-inversions.

Concluding Remarks

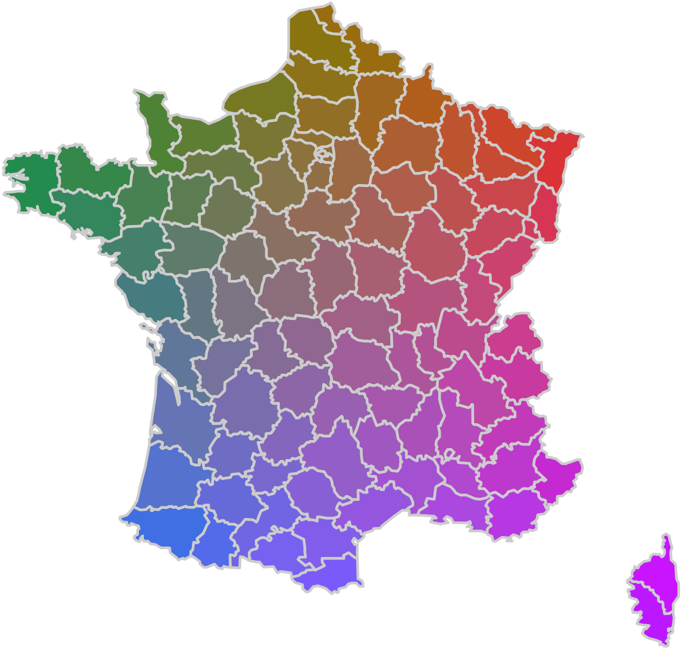


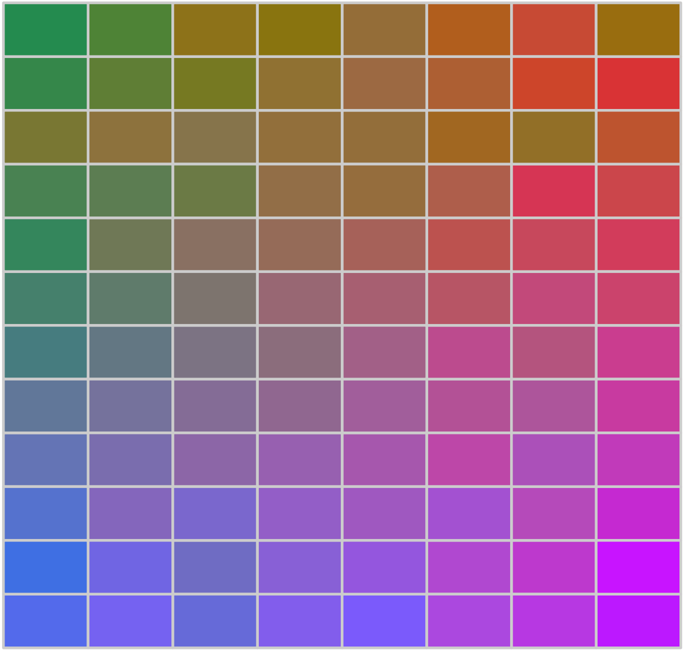
Concluding Remarks





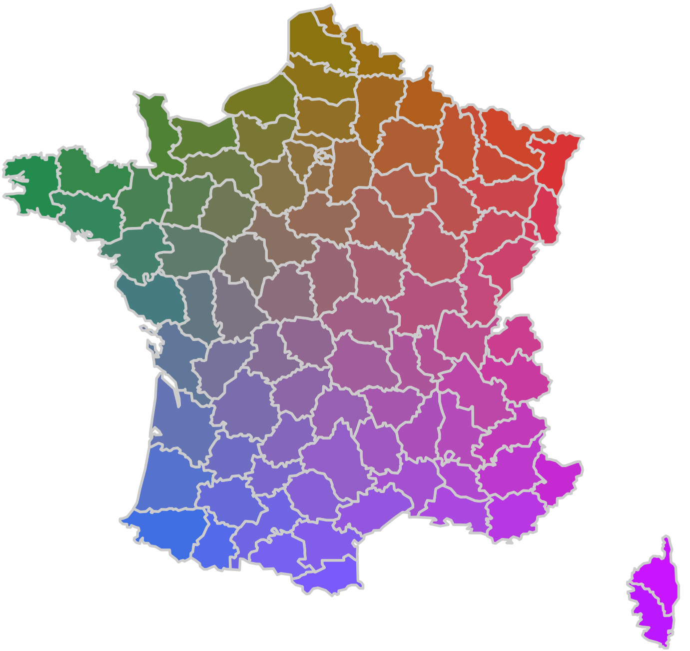
Concluding Remarks

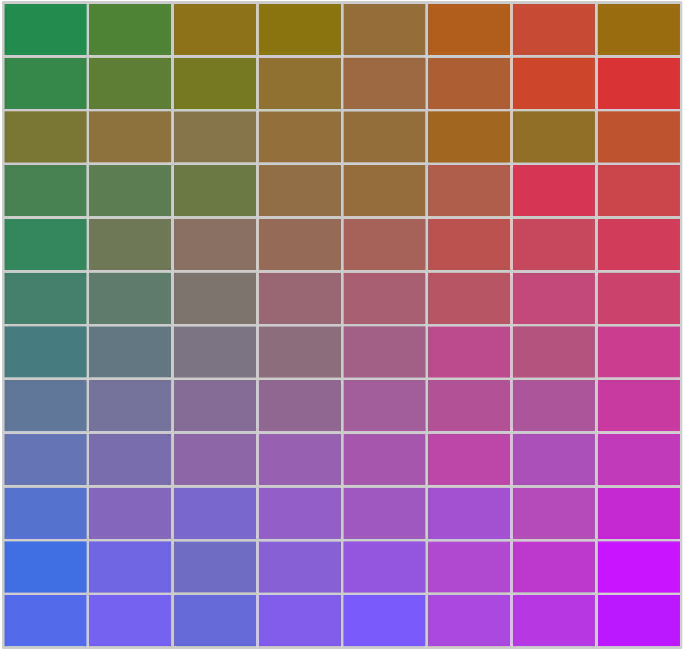




- Faster algorithm to minimise $D_{\mathcal{T}}$, D_{Λ}, and D?

Concluding Remarks

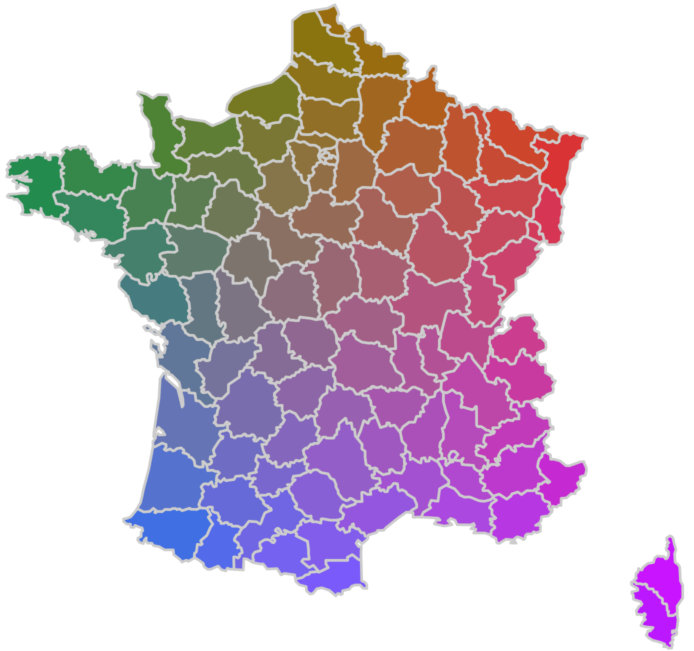


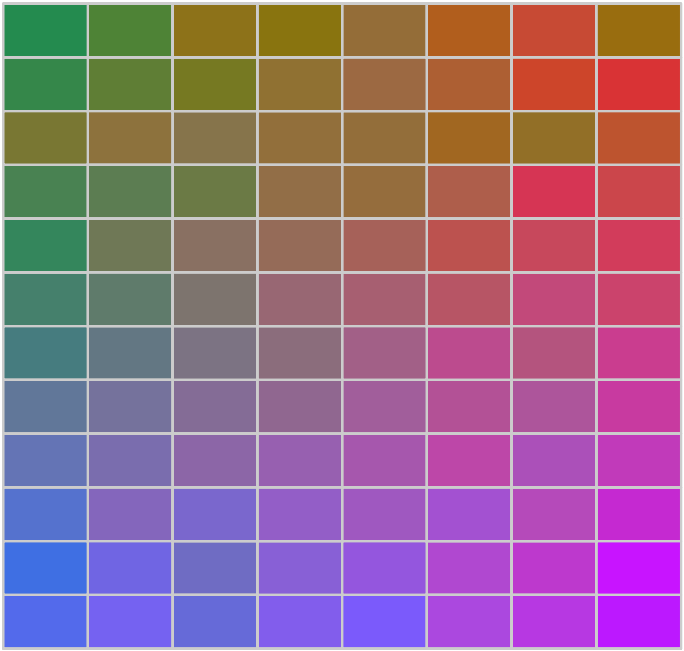


- Faster algorithm to minimise $D_{\mathcal{T}}$, D_{Λ}, and D?
- Exact or algorithm to preserve directional relations?

  $(1+\epsilon)$ approximation?

Concluding Remarks





- Faster algorithm to minimise $D_{\mathcal{T}}$, D_{Λ}, and D?
- Exact or algorithm to preserve directional relations?

  $(1+\epsilon)$ approximation?

Thank you!