Geographic Grid Embeddings

Frank Staals

Eindhoven University of Technology

July 18, 2011

Visualising Data

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

Visualising Data

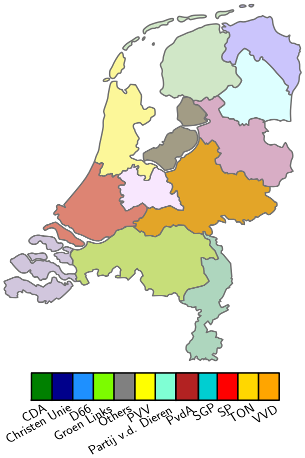
Given a map with *n* regions we want to show data for each region.

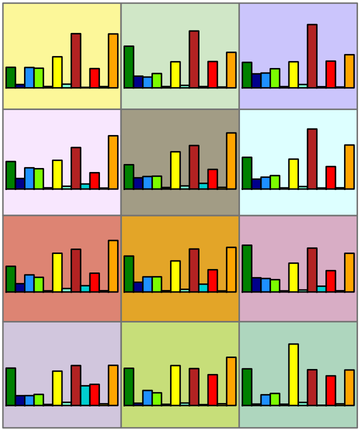
Desired properties:

- easy to read,
- easy to find the data for a given region,
- easy to compare for multiple regions, and
- the usable for both scalar values and multi-variate data.

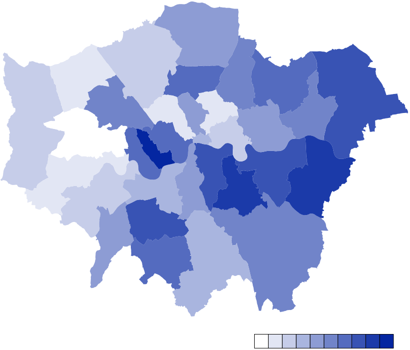
Visualising Data

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



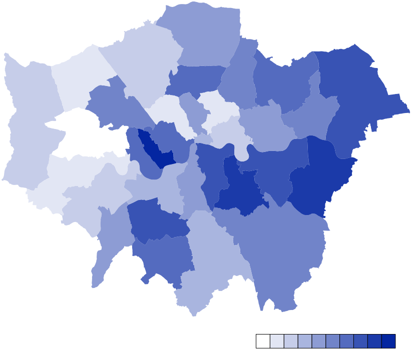


Visualisation Techniques



Use a choropleth map: colour the regions according to the data.

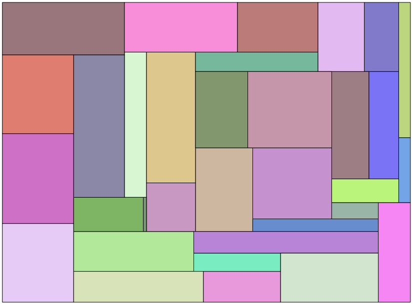
Visualisation Techniques



Use a choropleth map: colour the regions according to the data.

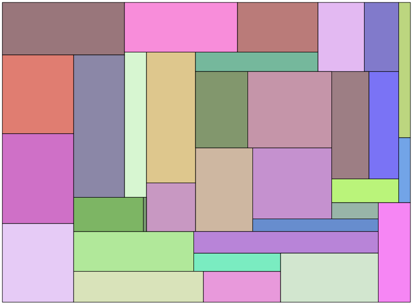
Problem: Difficult to compare.

Visualisation Techniques



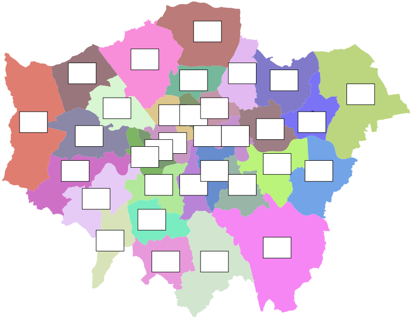
Use a cartogram: scale the region according to the data.

Visualisation Techniques



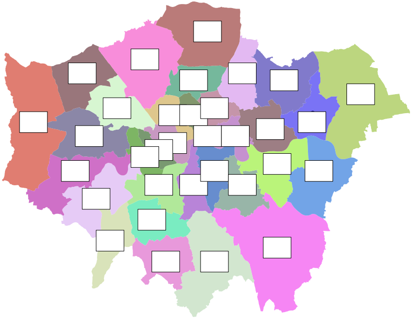
Use a cartogram: scale the region according to the data.

Problems: Hard to recognise regions, difficult to compare.



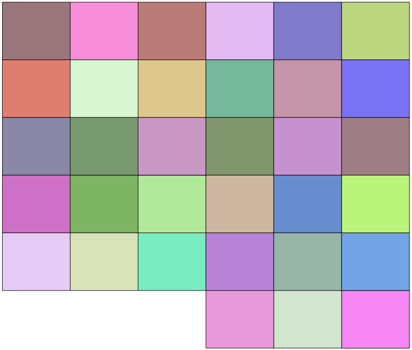
Use a symbol map: show a symbol/graphic to represent the data.

Visualisation Techniques



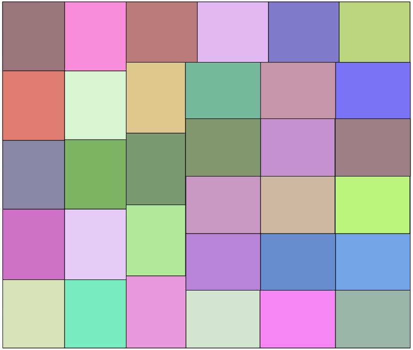
Use a symbol map: show a symbol/graphic to represent the data.
Problem: Adding symbols clutters the view.

Visualisation Techniques



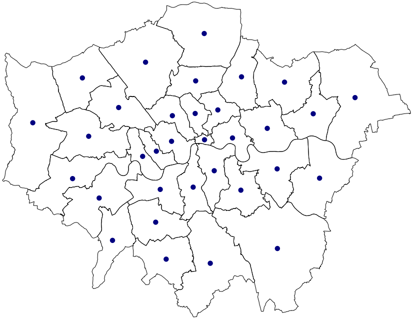
Idea: Add the symbols/graphics in a regular grid. The position in the grid corresponds with geographic location in the map.

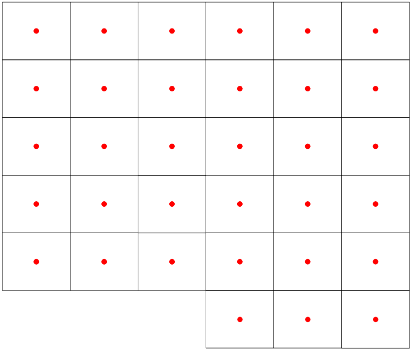
Visualisation Techniques



Related work: Spatially Ordered Treemaps **wood2008spatially**

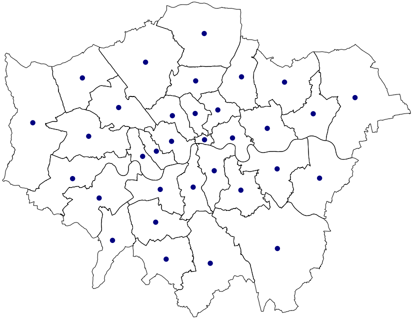
Visualisation Techniques

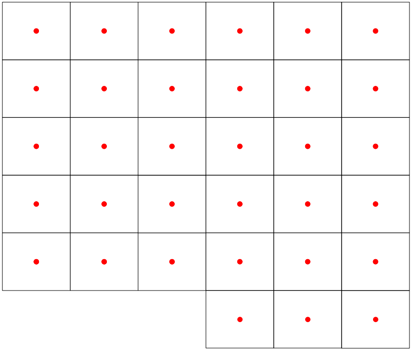




Model the problem as a Pointset Matching Problem.

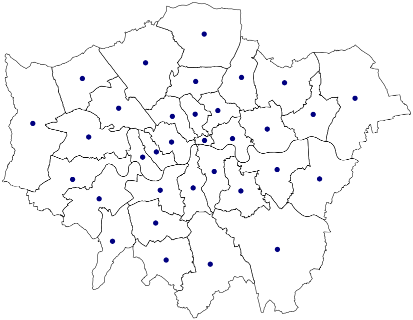
Visualisation Techniques

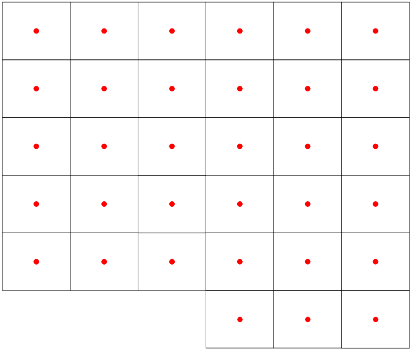




We represent each region in the map by blue point. This yields the set of blue points A.

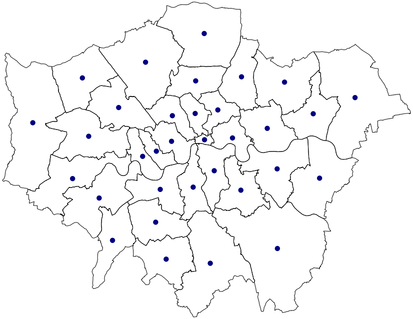
Visualisation Techniques

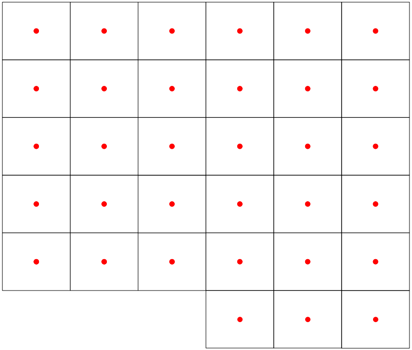




We represent each grid cell by a red point. This yields the set of red points B.

Visualisation Techniques





Goal: Find the "best" 1-1 matching $\phi : A \rightarrow B$.

Finding the “best” matching

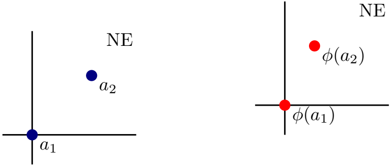
The matching ϕ should:

- minimise the total distance,

Finding the "best" matching

The matching ϕ should:

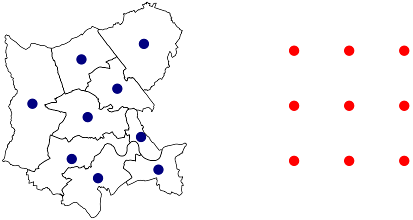
- minimise the total distance,
- preserve the directional relation, and



Finding the "best" matching

The matching ϕ should:

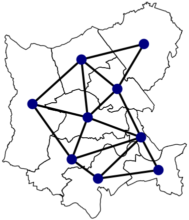
- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies

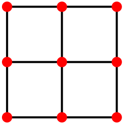


Finding the “best” matching

The matching ϕ should:

- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies

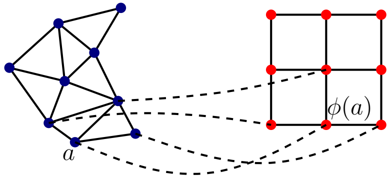




Finding the "best" matching

The matching ϕ should:

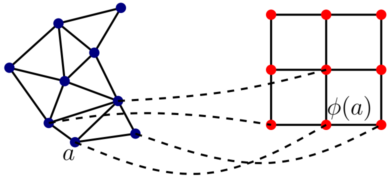
- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies



Finding the "best" matching

The matching ϕ should:

- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies



To get the best matching we allow translation and scaling of the pointset A.

Finding the "best" matching

The matching ϕ should:

- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies

To get the best matching we allow translation and scaling of the pointset A.

Modelling distance

For a 1-1 matching ϕ between A and B we define:

$$D_I(\phi) = \sum_{a \in A} d(a, \phi(a))$$

where d is a distance metric.

Modelling distance

For a 1-1 matching ϕ between A and B we define:

$$D_I(\phi) = \sum_{a \in A} d(a, \phi(a))$$

where d is a distance metric. We will consider $d = L_1$, and $d = L_2^2$.

Modelling distance

For a 1-1 matching ϕ between A and B we define:

$$D_I(\phi) = \sum_{a \in A} d(a, \phi(a))$$

where d is a distance metric. We will consider $d = L_1$, and $d = L_2^2$.

Let Φ be the set of all 1-1 matchings between A and B.

We then want a 1-1 matching ϕ^* such that:

$$D_I(\phi^*) = \min_{\phi \in \Phi} D_I(\phi)$$

Minimising D_I

Question: How can we find a matching ϕ that minimises D_I?

Answer: Use Linear Programming. Let f_{ab} denote the flow from a to b.

Minimising D_I

Question: How can we find a matching ϕ that minimises D_I?

Answer: Use Linear Programming. Let f_{ab} denote the flow from a to b.

$$\text{minimize} \sum_{a \in A} \sum_{b \in B} f_{ab} d(a, b)$$

subject to:

$$\sum_{b \in B} f_{ab} = 1 \qquad \forall a \in A$$

$$\sum_{a \in A} f_{ab} = 1 \qquad \forall b \in B$$

$$0 \leq f_{ab} \leq 1 \qquad \forall a \in A, b \in B$$

Minimising D_I

Analysis: This LP is an instance of the assignment problem.

Theorem

Given two sets A and B of n points in the plane, a one-to-one matching ϕ that minimises D_I can be computed in $O(n^3)$ time.

Minimising D_I

Analysis: This LP is an instance of the assignment problem.

Theorem

Given two sets A and B of n points in the plane, a one-to-one matching ϕ that minimises D_I can be computed in $O(n^3)$ time.

Theorem (vaidya1988geometry)

Given two sets A and B of n points in the plane, a one-to-one matching ϕ that minimises D_I with $d = L_1$ can be computed in $O(n^2(\log n)^3)$ time.

Modelling distance II

For a 1-1 matching ϕ between A and B we define:

$$D_I(\phi) = \sum_{a \in A} d(a, \phi(a))$$

Modelling distance II

For a 1-1 matching ϕ between A and B and a translation t we define:

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

Modelling distance II

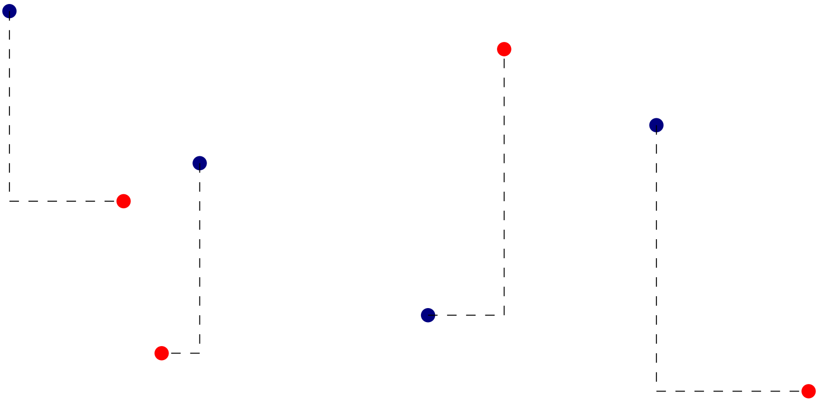
For a 1-1 matching ϕ between A and B and a translation t we define:

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

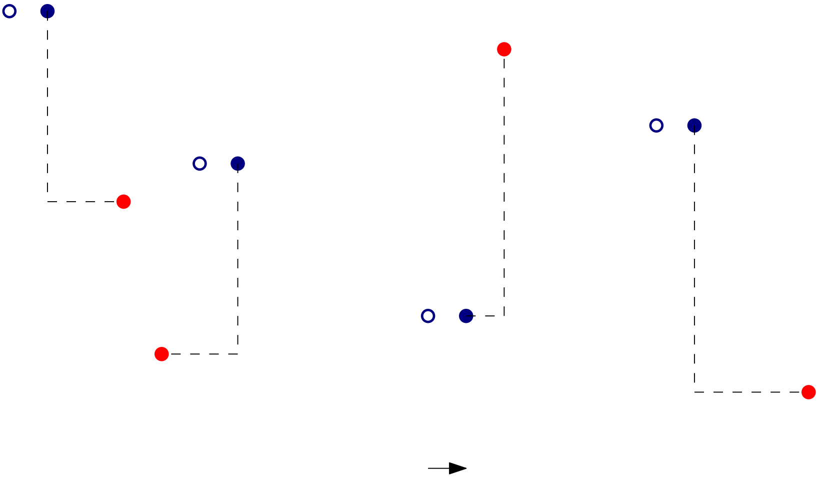
We now want a 1-1 matching ϕ^* and a translation t^* such that:

$$D_{\mathcal{T}}(\phi^*, t^*) = \min_{\phi \in \Phi, t \in \mathcal{T}} D_{\mathcal{T}}(\phi, t)$$

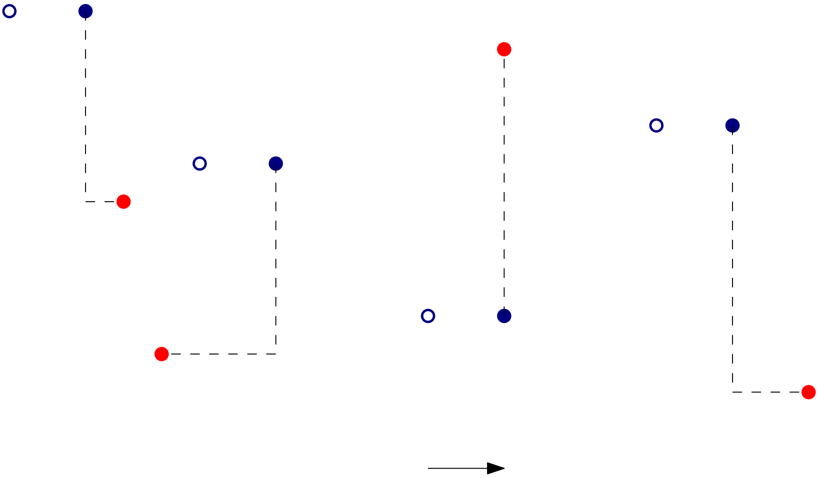
Minimising $D_{\mathcal{T}}$ with the L_1 distance



Minimising $D_{\mathcal{T}}$ with the L_1 distance



Minimising $D_{\mathcal{T}}$ with the L_1 distance



Minimising $D_{\mathcal{T}}$ with the L_1 distance

Lemma

Let A and B be two non x-aligned sets of n points in the plane, and let ϕ be a one-to-one matching between A and B. Then there is a horizontal translation t^* such that $A^* = \{a + t^* \mid a \in A\}$ and B are x-aligned and $D_{\mathcal{T}}(\phi, t^*) \leq D_I(\phi)$.

Minimising $D_{\mathcal{T}}$ with the L_1 distance

Lemma

Let A and B be two non x-aligned sets of n points in the plane, and let ϕ be a one-to-one matching between A and B. Then there is a horizontal translation t^* such that $A^* = \{a + t^* \mid a \in A\}$ and B are x-aligned and $D_{\mathcal{T}}(\phi, t^*) \leq D_I(\phi)$.

Symmetrically we can show that there is a vertical translation t^* that y-aligns two point sets A and B

Minimising $D_{\mathcal{T}}$ with the L_1 distance

Lemma

Let A and B be two non x-aligned sets of n points in the plane, and let ϕ be a one-to-one matching between A and B. Then there is a horizontal translation t^* such that $A^* = \{a + t^* \mid a \in A\}$ and B are x-aligned and $D_{\mathcal{T}}(\phi, t^*) \leq D_{\boldsymbol{I}}(\phi)$.

Symmetrically we can show that there is a vertical translation t^* that y-aligns two point sets A and B

There is a translation t that both x-aligns and y-aligns A and B and minimises $D_{\mathcal{T}}$.

Minimising $D_{\mathcal{T}}$ with the L_1 distance

Lemma

Let A and B be two non x-aligned sets of n points in the plane, and let ϕ be a one-to-one matching between A and B. Then there is a horizontal translation t^* such that $A^* = \{a + t^* \mid a \in A\}$ and B are x-aligned and $D_{\mathcal{T}}(\phi, t^*) \leq D_I(\phi)$.

Symmetrically we can show that there is a vertical translation t^* that y-aligns two point sets A and B

There is a translation t that both x-aligns and y-aligns A and B and minimises $D_{\mathcal{T}}$.

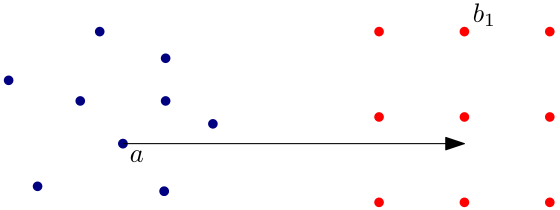
$\Longrightarrow$ We have to consider at most n^4 translations.

Minimising $D_{\mathcal{T}}$ with the L_1 distance

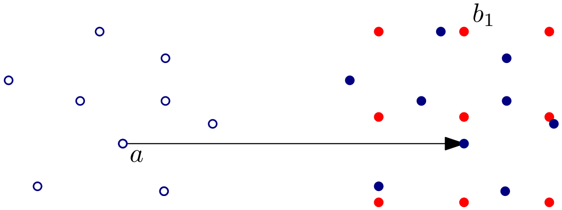
Theorem

Given two sets A and B of n points in the plane, a one-to-one matching ϕ and translation t that minimise $D_{\mathcal{T}}$ can be computed in $O(n^6(\log n)^3)$ time.

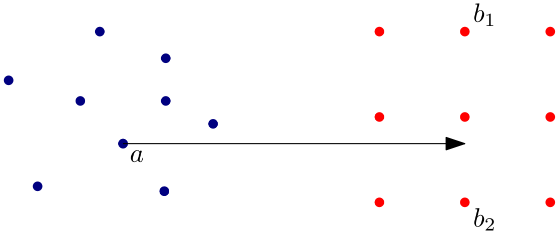
Minimising $D_{\mathcal{T}}$ with the L_1 distance



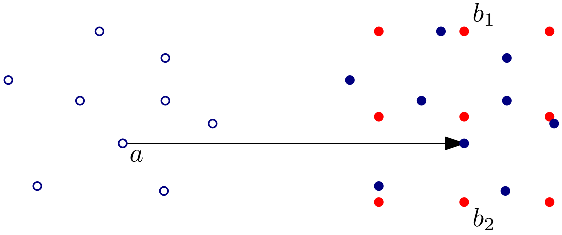
Minimising $D_{\mathcal{T}}$ with the L_1 distance



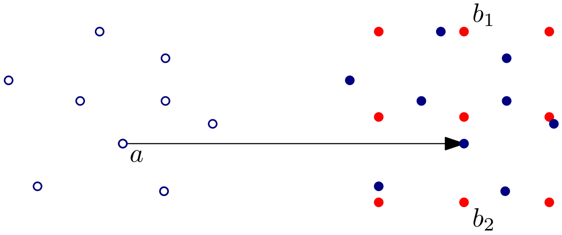
Minimising $D_{\mathcal{T}}$ with the L_1 distance



Minimising $D_{\mathcal{T}}$ with the L_1 distance



Minimising $D_{\mathcal{T}}$ with the L_1 distance



Corollary

Given a set A of n points in the plane and a set B of n grid points in an $R \times C$ grid, a one-to-one matching ϕ and translation t that minimise $D_{\mathcal{T}}$ can be computed in $O(nCnR \cdot n^2(\log n)^3) = O(n^5(\log n)^3)$ time.

Modelling distance III

For a 1-1 matching ϕ between A and B and a translation t we define:

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

Modelling distance III

For a 1-1 matching ϕ between A and B and a scaling λ we define:

$$D_{\Lambda}(\phi, \lambda) = \sum_{a \in A} d(\lambda a, \phi(a))$$

Modelling distance III

For a 1-1 matching ϕ between A and B and a scaling λ we define:

$$D_\Lambda(\phi, \lambda) = \sum_{a \in A} d(\lambda a, \phi(a))$$

We now want a 1-1 matching ϕ^* and a scaling λ^* such that:

$$D_\Lambda(\phi^*, \lambda^*) = \min_{\phi \in \Phi, \lambda \in \Lambda} D_\Lambda(\phi, \lambda)$$

Minimising D_Λ with the L_1 distance

Idea: Use the same approach as with translation...

Theorem

Given two sets A and B of n points in the plane, a one-to-one matching ϕ and scaling λ that minimise D_Λ can be computed in $O(n^6(\log n)^3)$ time.

Minimising D_Λ with the L_1 distance

Idea: Use the same approach as with translation...

Theorem

Given two sets A and B of n points in the plane, a one-to-one matching ϕ and scaling λ that minimise D_Λ can be computed in $O(n^6(\log n)^3)$ time.

Corollary

Given a set A of n points in the plane and a set B of n grid points in an $R \times C$ grid, a one-to-one matching ϕ and scaling λ that minimise D_Λ can be computed in $O(nCnR \cdot n^2(\log n)^3) = O(n^5(\log n)^3)$ time.

Modelling distance IIII

For a 1-1 matching ϕ between A and B, a translation t, and a scaling λ we define:

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

Modelling distance IIII

For a 1-1 matching ϕ between A and B, a translation t, and a scaling λ we define:

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

We now want a 1-1 matching ϕ^*, a translation t^*, and a scaling λ^* such that:

$$D_\Lambda(\phi^*, t^*, \lambda^*) = \min_{\phi \in \Phi, t \in \mathcal{T}, \lambda \in \Lambda} D(\phi, t, \lambda)$$

Minimising D with the L_1 distance

Idea: If one transformation can x-align one pair of points then two transformations can x-align two pairs of points.

Minimising D with the L_1 distance

Idea: If one transformation can x-align one pair of points then two transformations can x-align two pairs of points.

$$t_x = |b_x - a_x|$$
$$\lambda_x = |b'_x - a'_x|$$

Minimising D with the L_1 distance

Idea: If one transformation can x-align one pair of points then two transformations can x-align two pairs of points.

$$t_x = |b_x - a_x|$$
$$\lambda_x = |b'_x - a'_x|$$

Problem: There is only a unique solution if a_x, a'_x, b_x, and b'_x are independent. This is not necessarily the case in our setting!

Minimising $D_{\mathcal{T}}$ and D_{Λ} with the L_2^2 distance

Theorem (By cohen1999earth)

The translation that aligns the centroids of A and B minimises $D_{\mathcal{T}}$.

So only one translation to consider.

Minimising $D_{\mathcal{T}}$ and D_{Λ} with the L_2^2 distance

Theorem (By cohen1999earth)

The translation that aligns the centroids of A and B minimises $D_{\mathcal{T}}$.

So only one translation to consider.

Question: How about minimising D_{Λ}?

Minimising $D_{\mathcal{T}}$ and D_Λ with the L_2^2 distance

Theorem (By cohen1999earth)

The translation that aligns the centroids of A and B minimises $D_{\mathcal{T}}$.

So only one translation to consider.

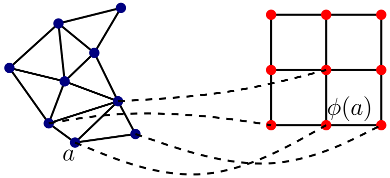
Question: How about minimising D_Λ?

Answer: Does not work...

Finding the “best” matching

The matching ϕ should:

- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies



To get the best matching we allow translation and scaling of the pointset A.

Problem Definition

Given the dual graph of the regions $G = (A, E)$ and the dual graph of the grid $H = (B, Z)$ find an embedding $\phi : G \hookrightarrow H$ that maximises the number of preserved adjacencies from G.

Problem Definition

Given the dual graph of the regions $G = (A, E)$ and the dual graph of the grid $H = (B, Z)$ find an embedding $\phi : G \hookrightarrow H$ that maximises the number of preserved adjacencies from G.

Given two graphs $G = (A, E)$ and $H = (B, Z)$, find the maximal size subsets $E' \subseteq E$ and $Z' \subseteq Z$ such that (A, E') and (B, Z') are isomorphic.

Problem Definition

Given the dual graph of the regions $G = (A, E)$ and the dual graph of the grid $H = (B, Z)$ find an embedding $\phi : G \hookrightarrow H$ that maximises the number of preserved adjacencies from G.

MAXIMUM COMMON EDGE SUBGRAPH

Given two graphs $G = (A, E)$ and $H = (B, Z)$, find the maximal size subsets $E' \subseteq E$ and $Z' \subseteq Z$ such that (A, E') and (B, Z') are isomorphic.

Problem: MAXIMUM COMMON EDGE SUBGRAPH is NP-complete.

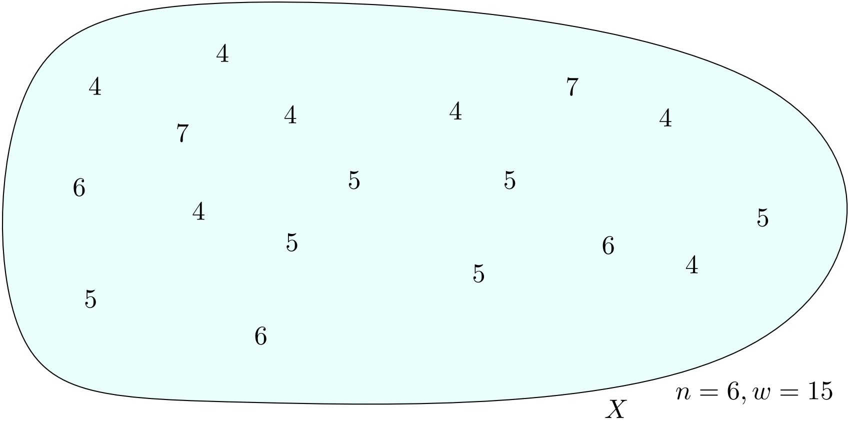
Problem Definition

ADJACENCY PRESERVING GRID EMBEDDING

Given a planar graph $G = (V, E)$ and a grid graph $H = (N, Z)$ with $|V| = |N|$, is it possible to find an embedding $\phi : G \hookrightarrow H$ that preserves at least k adjacencies from G?

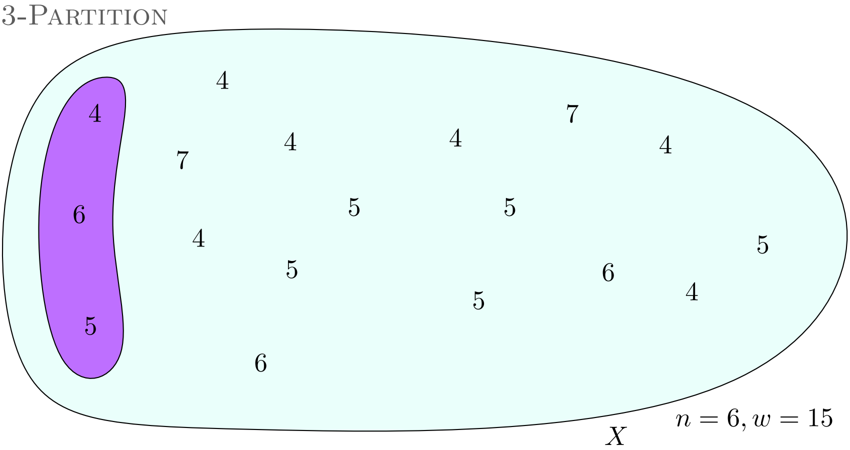
3-Partition

3-PARTITION

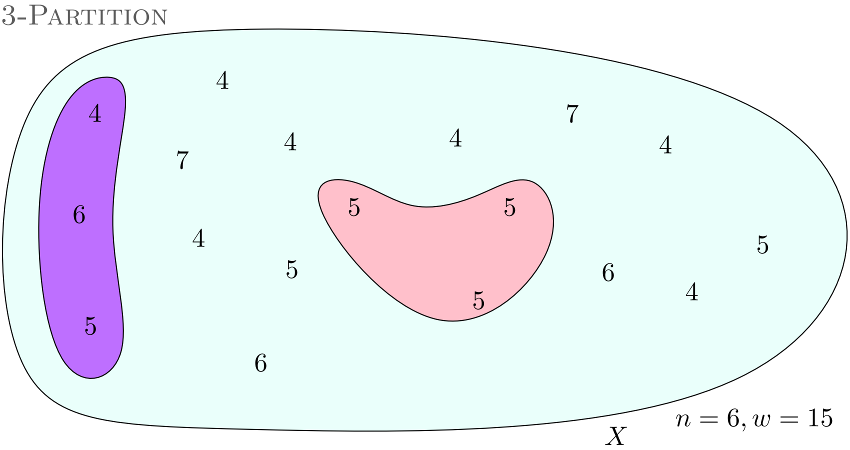


$n = 6, w = 15$

3-Partition

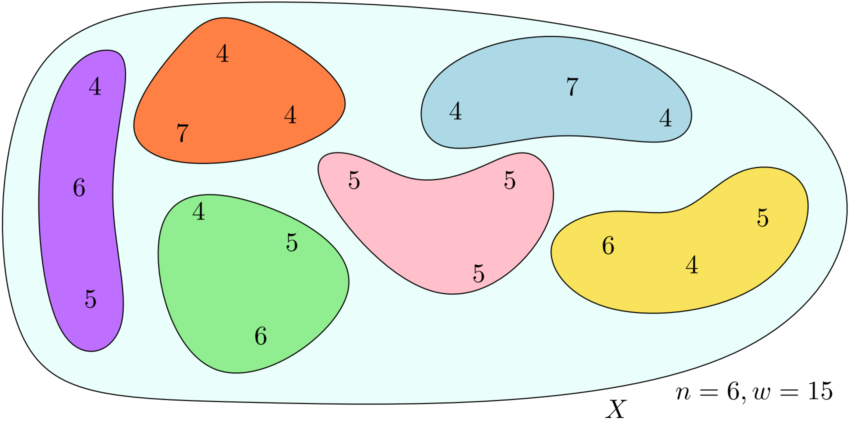


3-Partition



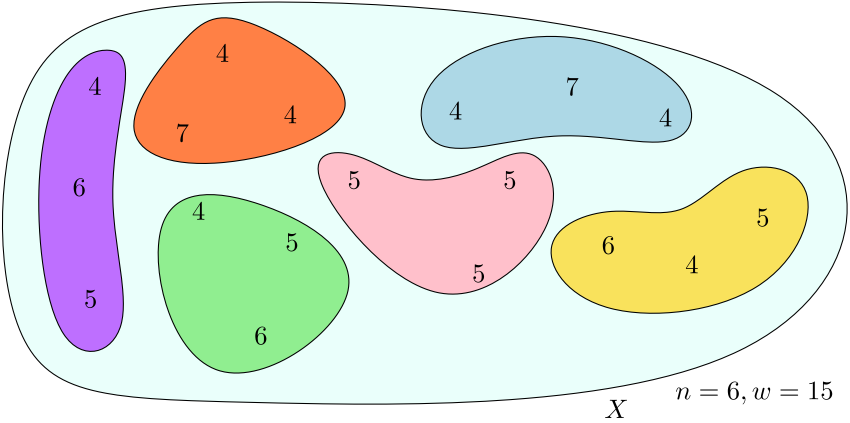
3-Partition

3-PARTITION



3-Partition

3-PARTITION



3-PARTITION is strongly NP-complete:

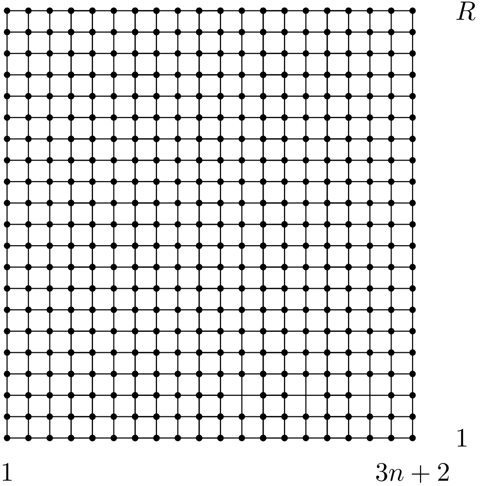
NP-hard even if all $x \in X$ bounded by a polynomial in n.

NP-Completeness Proof

Given X, construct a grid graph H and a graph $G = (V, E)$ such that:

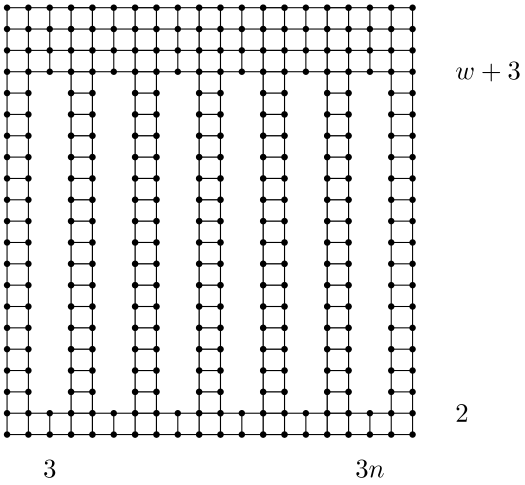
$$\phi \text{ preserves } |E| \text{ edges} \iff X \text{ has a 3-partition}$$

NP-Completeness Proof



H is a grid graph with $3n+2$ columns and $R = \max(w+4, 3n+3)$ rows.

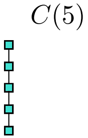
NP-Completeness Proof



G consists of $3n+1$ components:

- a separator S
- for each $x \in X$ a chain $C(x)$

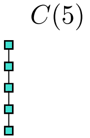
NP-Completeness Proof



G consists of $3n+1$ components:

- a separator S
- for each $x \in X$ a chain $C(x)$

NP-Completeness Proof

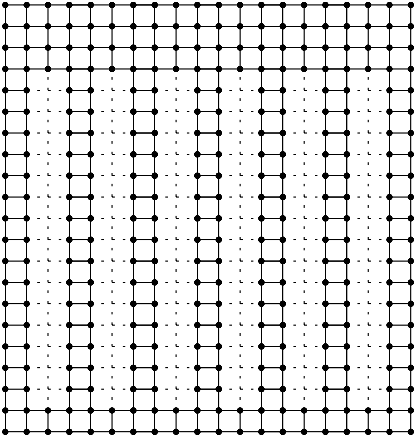


G consists of $3n + 1$ components:

- a separator S
- for each $x \in X$ a chain $C(x)$

G is polynomial in size.

NP-Completeness Proof

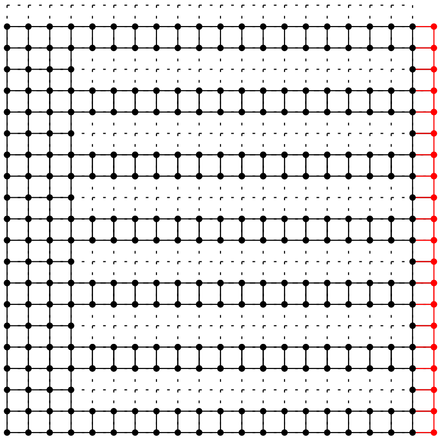


Suppose ϕ preserves all $|E|$ edges.

$\implies$

There are only 2 placements possible for S.

NP-Completeness Proof

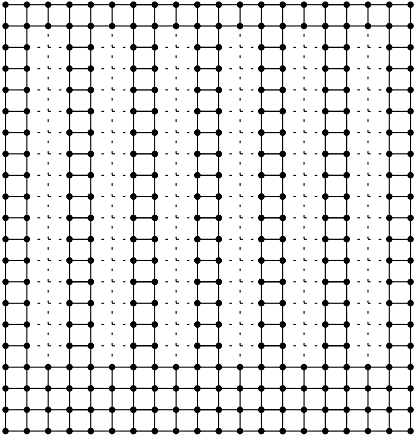


Suppose ϕ preserves all $|E|$ edges.

$\implies$

There are only 2 placements possible for S.

NP-Completeness Proof

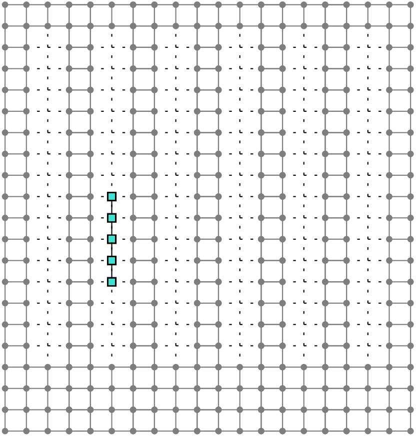


Suppose ϕ preserves all $|E|$ edges.

$\implies$

There are only 2 placements possible for S.

NP-Completeness Proof

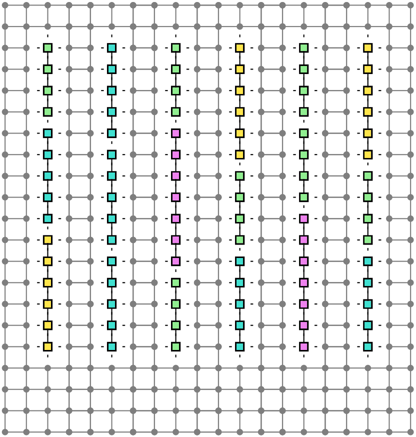


Suppose ϕ preserves all $|E|$ edges.

$\Longrightarrow$

$C(x)$ placed in a single column.

NP-Completeness Proof



Suppose ϕ preserves all $|E|$ edges.

$\implies$

ϕ yields a valid 3-partition of X.

Algorithms for preserving adjacencies

- Use an MAXIMUM COMMON SUBGRAPH algorithm. For example **mcgregor1982backtrack**'s algorithm.

Algorithms for preserving adjacencies

- Use an MAXIMUM COMMON SUBGRAPH algorithm. For example **mcgregor1982backtrack**'s algorithm. $\implies$ Too slow!

Algorithms for preserving adjacencies

- Use an MAXIMUM COMMON SUBGRAPH algorithm. For example **mcgregor1982backtrack**'s algorithm. $\implies$ Too slow!
- Approximate.

Algorithms for preserving adjacencies

- Use an MAXIMUM COMMON SUBGRAPH algorithm. For example **mcgregor1982backtrack**'s algorithm. $\implies$ Too slow!
- Approximate.

Problem: kann1992approximability shows a lot of MAXIMUM SOMMON SUBGRAPH problems are NP-hard to approximate.

Algorithms for preserving adjacencies

- Use an MAXIMUM COMMON SUBGRAPH algorithm. For example **mcgregor1982backtrack**'s algorithm. $\implies$ Too slow!
- Approximate.

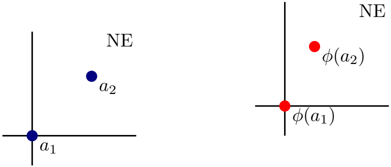
Problem: kann1992approximability shows a lot of MAXIMUM SOMMON SUBGRAPH problems are NP-hard to approximate.

We designed a 4-approximation algorithm to embed a planar graph into a grid graph.

Finding the “best” matching

The matching ϕ should:

- minimise the total distance,
- preserve the directional relation, and
- preserve adjacencies



To get the best matching we allow translation and scaling of the pointset A.

Approximating directional relations

Algorithm DirRel-Preserve

1. Let $w(a, b)$ denote the number of pairs involving a with the wrong directional relation if we match a to b.
2. Compute a minimal distance matching using w as distance measure.

Approximating directional relations

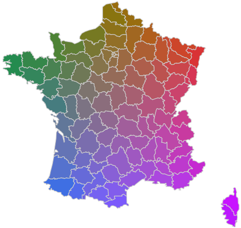
Algorithm DirRel-Preserve

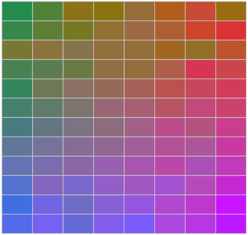
1. Let $w(a, b)$ denote the number of pairs involving a with the wrong directional relation if we match a to b.
2. Compute a minimal distance matching using w as distance measure.

Conjecture

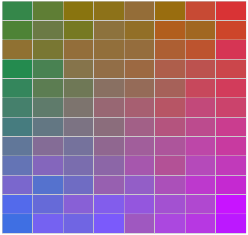
DirRel-Preserve *is a 4-approximation algorithm.*

Results for different measures





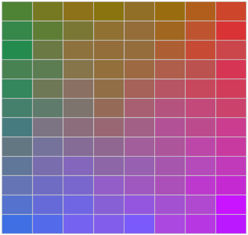
L1_trans



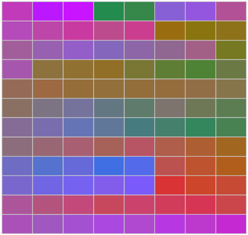
L1_scale

L22:
- directional relation $\approx 79\%$
- adjacencies $\approx 54\%$

adjacency:
- directional relation $\approx 13\%$
- adjacencies $\approx 36\%$

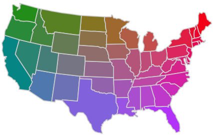


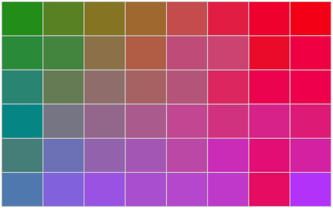
L22



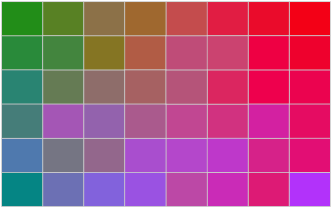
adjacency

Results for different measures

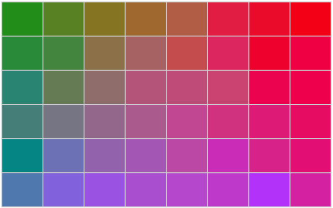




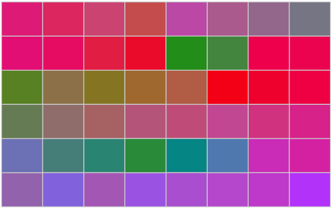
L1_trans



L1_scale

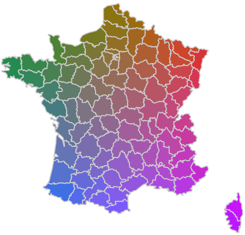


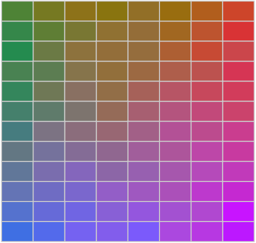
L22



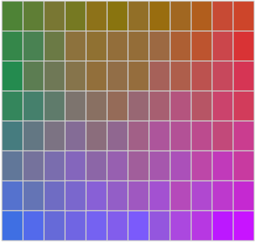
adjacency

Results for different grid sizes

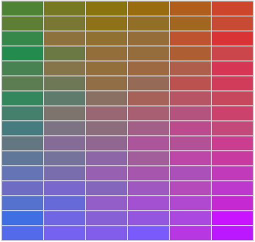




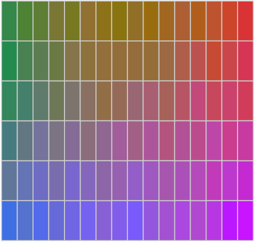
12×8



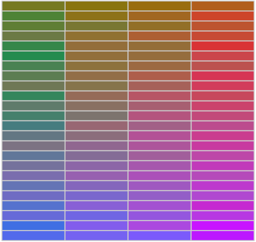
8×12



16×6

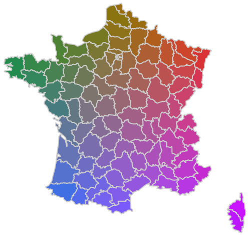


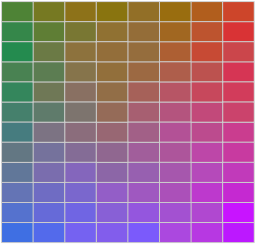
6×16



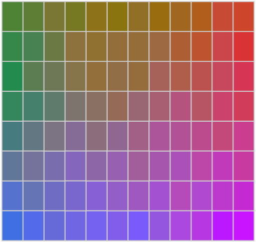
24×4

Results for different grid sizes

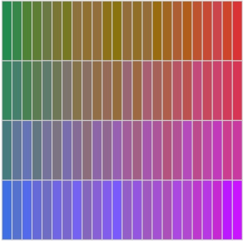




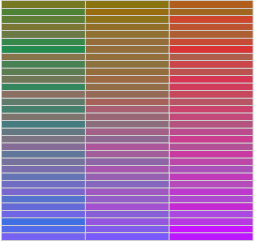
12×8



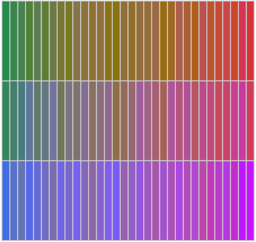
8×12



4×24

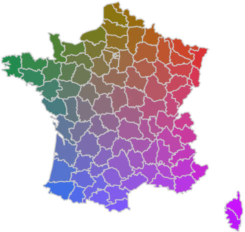


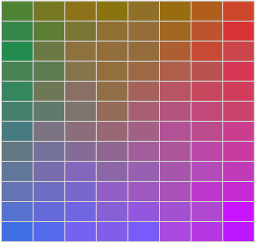
32×3



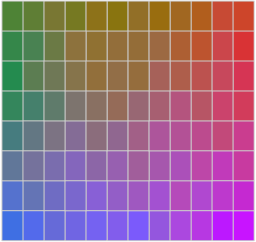
3×32

Results for different grid sizes

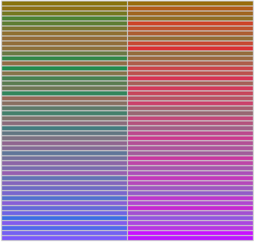




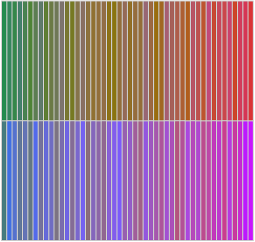
12×8



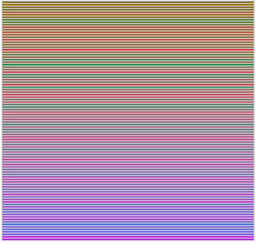
8×12



48×2



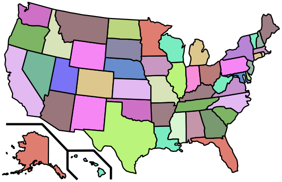
2×48



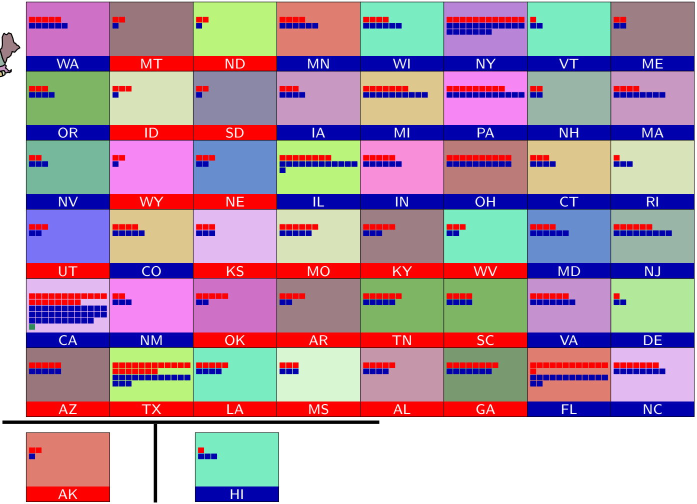
96×1

Examples

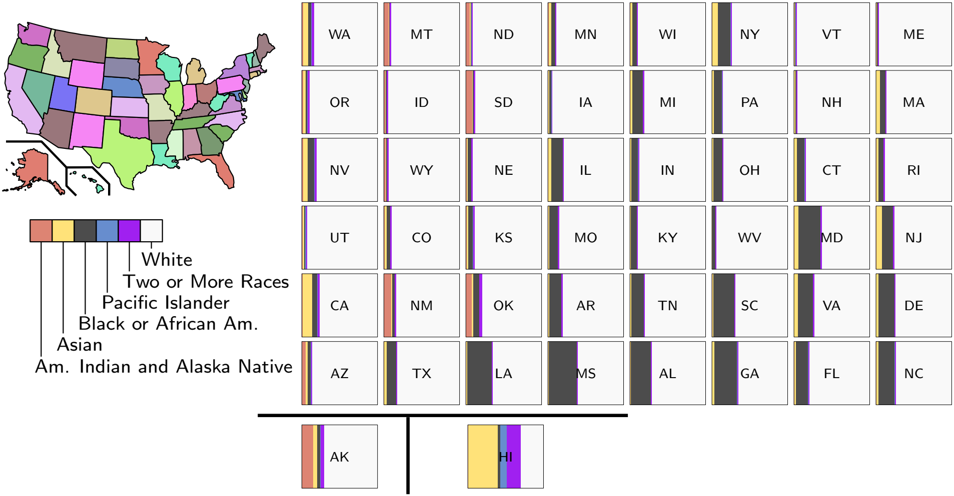
U.S. Presidential Elections 2008



- 1 electoral vote
- McCain
- Obama
- other



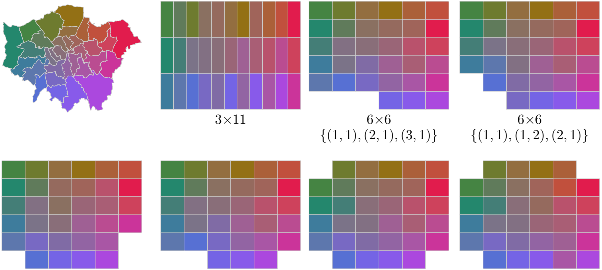
U.S. Population estimates in 2009 per race



Future Work

- Directional Relations proof
- How to optimise all three criteria?
- How to pick a suitable set of grid cells?

- Directional Relations proof
- How to optimise all three criteria?
- How to pick a suitable set of grid cells?



3×11

6×6 $\{(1,1),(2,1),(3,1)\}$

6×6 $\{(1,1),(1,2),(2,1)\}$

6×6 $\{(1,1),(1,6),(2,6)\}$

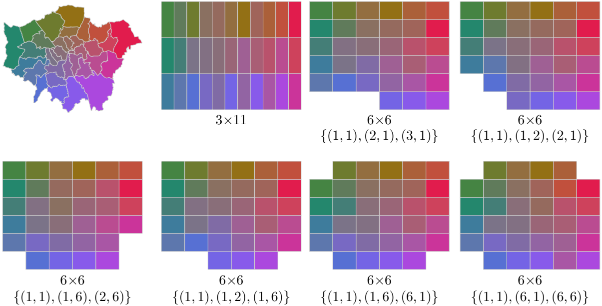
6×6 $\{(1,1),(1,2),(1,6)\}$

6×6 $\{(1,1),(1,6),(6,1)\}$

6×6 $\{(1,1),(6,1),(6,6)\}$

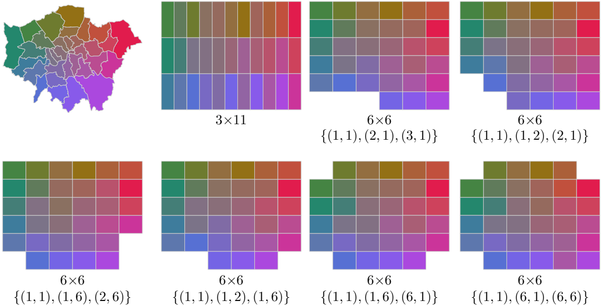
Future Work

- Directional Relations proof
- How to optimise all three criteria?
- How to pick a suitable set of grid cells?
- ...



Future Work

- Directional Relations proof
- How to optimise all three criteria?
- How to pick a suitable set of grid cells?
- ...



Thank you! Questions?

References I