All you need is a tree

2022-07-18

Setting

Let $\boldsymbol{c} \in \mathcal{P}_N$ be an ordered partition of $\{1, \ldots, N\}$, that is:

$$\boldsymbol{c} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\} \qquad \bigcup_k \boldsymbol{c}_k = [N] \qquad \boldsymbol{c}_k \bigcap \boldsymbol{c}_l = \emptyset \tag{1}$$

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be a dataset[1]. We suppose that all datapoints belonging to the same element of the partition are independent and identically distributed, *i.e.* conditional independence:

$$p(\mathbf{x}|\mathbf{c}) = \prod_k \int_\theta \prod_{i \in c_k} p(\mathbf{x}_i|\theta) p(\theta) d\theta$$

[1] This can be extended to a case where $\mathbf{X}$ is an adjacency matrix.

Setting

$$\mathscr{L}^{obs}(\mathbf{X}, \boldsymbol{c}_k) = \log\left(\int_\theta \prod_{i \in c_k} p(\mathbf{x}_i|\theta)p(\theta)d\theta\right)$$

In case of exponential Familly observations :

$$p(\mathbf{x}|\theta) = \exp\left(\gamma(\theta).T(\mathbf{x}) - A(\theta) + B(\mathbf{x})\right)$$

and $\mathscr{L}^{obs}(\mathbf{X}, \boldsymbol{c}_k) = \phi(\sum_{i \in \boldsymbol{c}_k} T(\mathbf{x}_i))$

Setting, MAP clustering

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} p(\mathbf{x}, \boldsymbol{c}) = \arg\max_{\boldsymbol{c}} p(\mathbf{x}|\boldsymbol{c})p(\boldsymbol{c})$$

We need to define a prior for $\boldsymbol{c}$: $p(\boldsymbol{c})$?

Chinese restaurant process

Process :

- $\frac{\alpha}{n+\alpha}$ new table
- $\frac{n_k}{n+\alpha}$ at table k

Distribution

$$p(\mathbf{c}|\alpha) = \frac{\Gamma(\alpha)\alpha^K}{\Gamma(\alpha + N))} \prod_k \Gamma(n_k)$$

Uniform

Completely uniform: $p(\boldsymbol{c}) = \frac{1}{\sum_{k=1}^{N} \mathcal{B}_n^k k!}$

Uniform in k and then uniform over the partitions with k elements:

$$\begin{aligned} p(K) &= \frac{1}{N} && (2) \\ p(\boldsymbol{c}|K) &= \frac{1}{\mathcal{B}_n^k k!} && (3) \end{aligned}$$

Uniform

Uniform in k, then uniform over possible counts and then uniform over the partitions with k elements and the specific counts:

$$
\begin{aligned}
p(K) &= \frac{1}{N} && (4)\\
p(\mathbf{n}|K) &= \binom{N-1}{K-1}^{-1} && (5)\\
p(\mathbf{c}|\mathbf{n}) &= \frac{1}{N!}\prod_k n_k! \times \mathbf{1}_{\{c/\mathbf{n_c}=\mathbf{c}\}} \times \mathbf{1}_{\{c/|c|=K\}} && (6)
\end{aligned}
$$

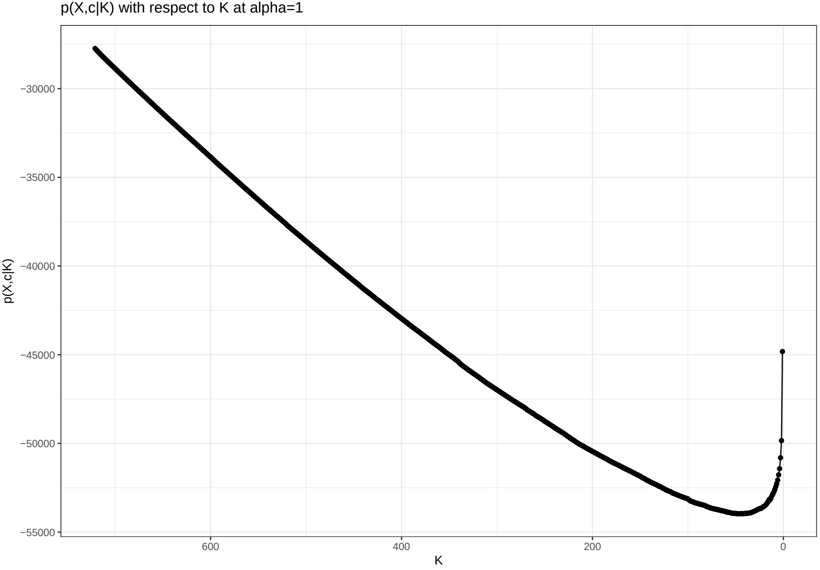
Truncated geometric over K

$$
\begin{aligned}
p(K) &= \frac{1}{1-\alpha^N}\alpha^{K-1}(1-\alpha) && (7)\\
p(\mathbf{n}|K) &= \binom{N-1}{K-1}^{-1} && (8)\\
p(\mathbf{c}|\mathbf{n}) &= \frac{1}{N!}\prod_k n_k! \times \mathbf{1}_{\{c/\mathbf{n_c}=\mathbf{c}\}} \times \mathbf{1}_{\{c/|c|=K\}} && (9)
\end{aligned}
$$

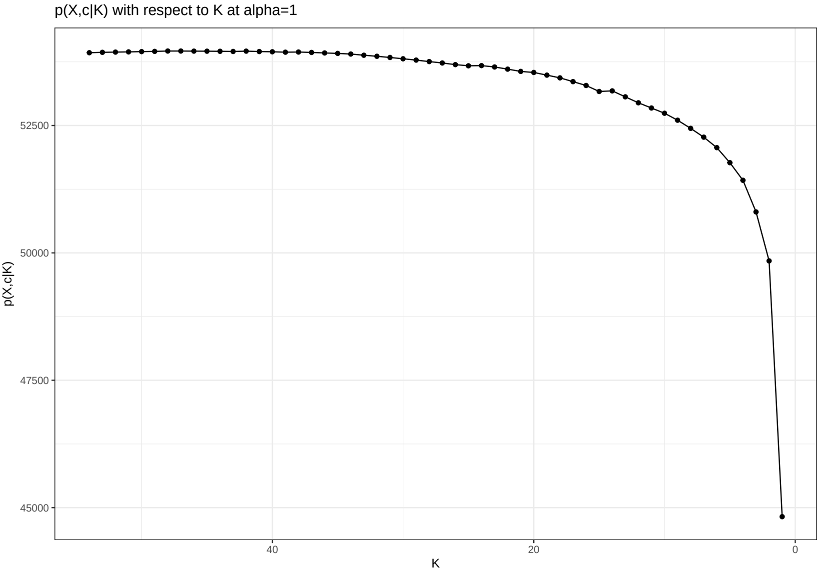
$\alpha = 1$ is equivalent to uniform, $\alpha = 0$ to a dirac at $k = 1$.

$$
p(\mathbf{c}|\alpha) = p(\mathbf{c}, \mathbf{n_c}, K_\mathbf{c}) = \binom{N-1}{K-1}^{-1} \frac{1}{1-\alpha^N}\alpha^{K-1}(1-\alpha)\frac{1}{N!}\prod_k n_k!
$$

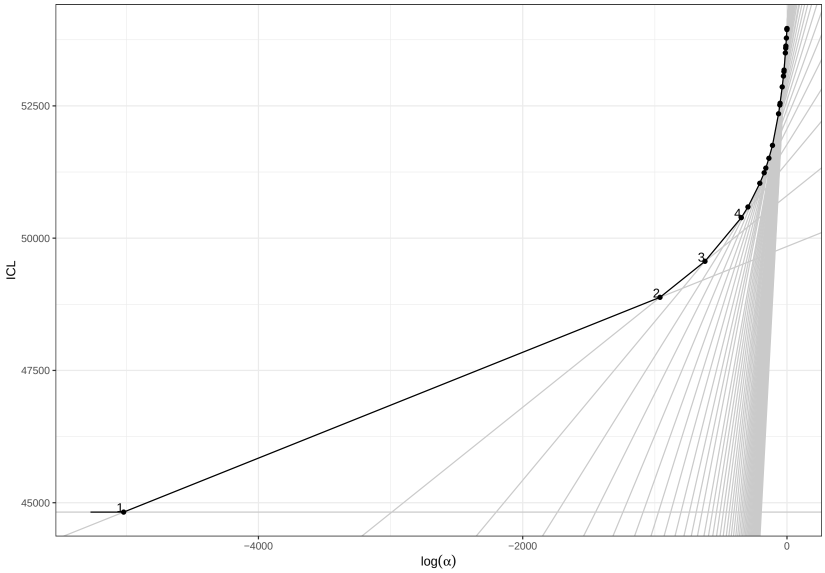
Truncated geometric over K and dendograms



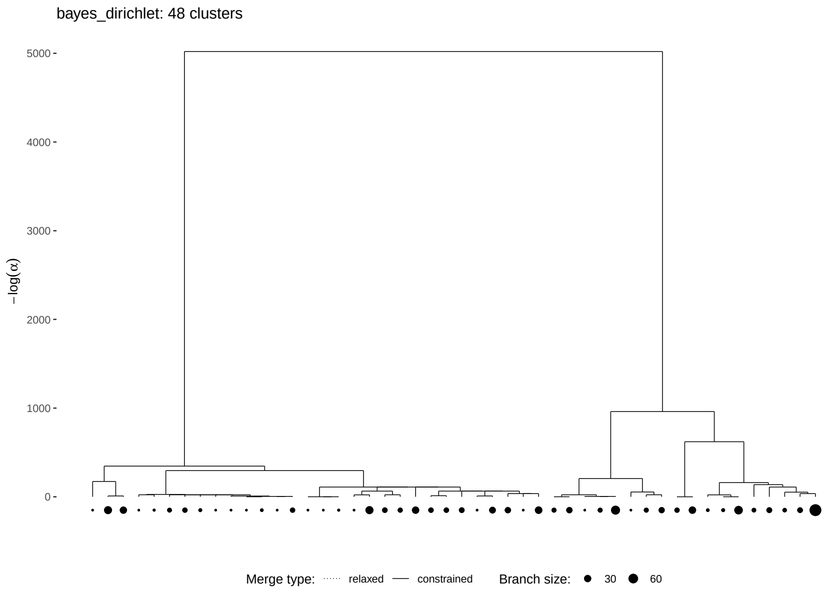
Truncated geometric over K and dendograms



Truncated geometric over K and dendograms



Truncated geometric over K and dendograms



Contiguity constrained clustering

Contiguity constrained clustering

We get a graph G and clusters must be connected in the graph.

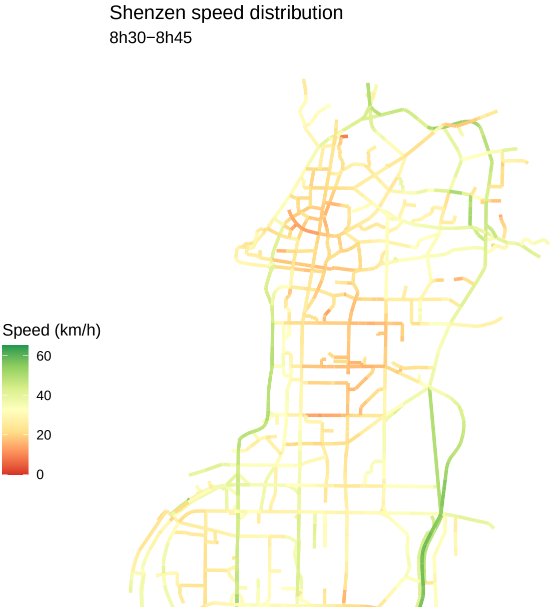
$$\Leftrightarrow$$

A clustering can be obtained by cutting $K-1$ edges in a spanning tree of G.

Pbr how to count the number of possible partition with k elements with theses constraints ? Easy for trees and lines : their are C_{k-1}^{N-1} possible compatible partitions. But for generic graphs ?

Contiguity constrained clustering

Example :



Spanning tree prior

- $\boldsymbol{t}$ a spanning tree of G,
- $\mathcal{T}_G$ the set of all spanning tree of G,
- a partition $\boldsymbol{c}$ is compatible with a spanning tree $\boldsymbol{t}$, noted $\boldsymbol{c} \prec \boldsymbol{t}$; if $\boldsymbol{c}$ corresponds to the set of connected components obtained by pruning $|\boldsymbol{c}| - 1$ edges from $\boldsymbol{t}$,
- if this is not the case $\boldsymbol{c}$ is not compatible with a spanning tree $\boldsymbol{t}$ and we denote this property by $\boldsymbol{c} \not\prec \boldsymbol{t}$.

Spanning tree prior

Sample spanning tree and cut them

$$
\begin{aligned}
p(K|\alpha) &= \frac{1}{1-\alpha^N}\alpha^{K-1}(1-\alpha) \\
p(\boldsymbol{t}) &= \frac{1}{|\mathcal{T}_G|} \\
p(\boldsymbol{c}|\boldsymbol{t},K) &= \frac{1}{C_{K-1}^{N-1}}\mathbf{1}_{\{\boldsymbol{c}/\boldsymbol{c}\prec\boldsymbol{t}\}}
\end{aligned}
$$

Spanning tree prior

If we marginalize out the sampling of the spanning trees we get :

$$
\begin{aligned}
p(\boldsymbol{c}|K) &= \sum_{\boldsymbol{t}\in\mathcal{T}_G} p(\boldsymbol{c}|\boldsymbol{t},K)p(\boldsymbol{t}) \\
&= \frac{1}{|\mathcal{T}_G|C_{K-1}^{N-1}} \sum_{\boldsymbol{t}\in\mathcal{T}_G} \mathbf{1}_{\{\boldsymbol{c}/\boldsymbol{c}\prec\boldsymbol{t}\}} \\
&= \frac{|\{\boldsymbol{t}/\boldsymbol{c}\prec\boldsymbol{t}\}|}{|\mathcal{T}_G|C_{K-1}^{N-1}}
\end{aligned}
$$

We need to compute $|\mathcal{T}_G|$ and $|\{\boldsymbol{t}/\boldsymbol{c}\prec\boldsymbol{t}\}|$?

Kirchhoff's theorem

Kirchhoff's theorem:

$$|\mathcal{T}_G| = \frac{1}{N}\lambda_1 ... \lambda_{N-1},$$

with λ_i be the non-zero eigenvalues of Laplacian matrix L of G. Practically, we compute $\log(|\mathcal{T}|)$ as $\log(det(L_{\text{-}1,\text{-}1}))$. If the graph is sparse, this can be solved in a reasonable amount of time via using a sparse $LU/$ Cholevsky decomposition.

Cuset theorem

- $G[\boldsymbol{c}_k]$ the subgraph of G induced by $\boldsymbol{c}_k$,
- $\boldsymbol{cutset}(G, \boldsymbol{c}_g, \boldsymbol{c}_h) = \{(u, v) \in E / u \in \boldsymbol{c}_g, v \in \boldsymbol{c}_h\}$, the set of edges of G between element g and h of $\boldsymbol{c}$,
- $G \diamond \boldsymbol{c}$ the multigraph $(\{1, ..., K\}, \{(g, h, |\boldsymbol{cutset}(G, \boldsymbol{c}_g, \boldsymbol{c}_h)|)/g \neq h\})$ obtained by counting the number of links in G between each pairs of different elements of $\boldsymbol{c}$.

Cuset theorem

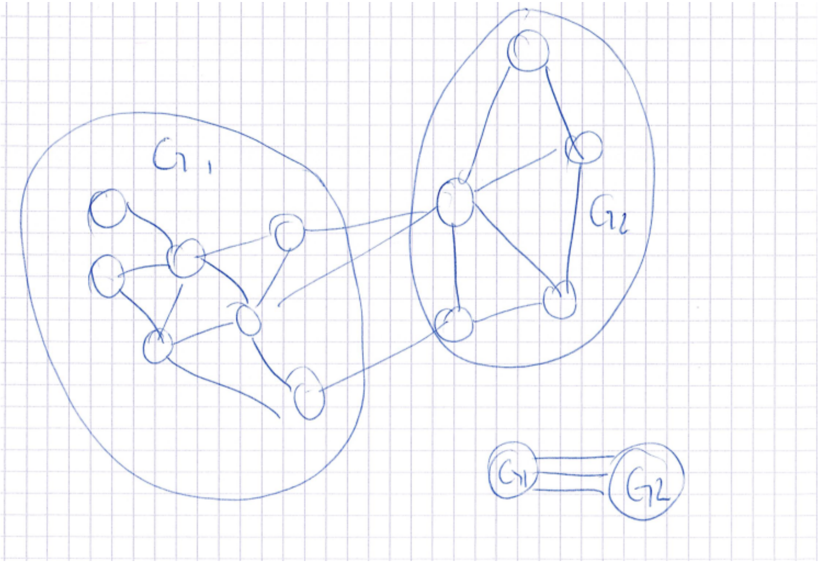
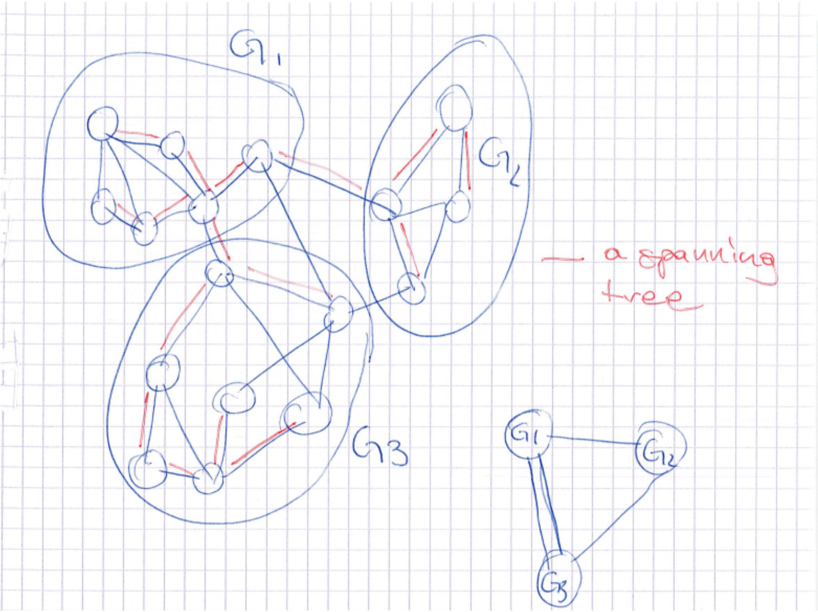


Figure 1: Cut set

Cuset theorem



Cuset theorem

$$\log(|\{\boldsymbol{t}/\boldsymbol{c} \prec \boldsymbol{t}\}|) = \overbrace{\sum_{k=1}^{K} \log(|\mathcal{T}_{G[\boldsymbol{c}_k]}|)}^{\text{intra-cluster spanning trees}} + \underbrace{log(|\mathcal{T}_{G\diamond\boldsymbol{c}}|)}_{\text{inter-clusters spanning trees}}$$

Cuset theorem

When the partition is a simple cut $\boldsymbol{c} = \{\boldsymbol{c}_1, \boldsymbol{c}_2\}$, this reduce to :

$$|\{\boldsymbol{t}/\boldsymbol{c} \prec \boldsymbol{t}\}| = |\mathcal{T}_{G[\boldsymbol{c}_1]}||\mathcal{T}_{G[\boldsymbol{c}_2]}||\boldsymbol{cutset}(G, \boldsymbol{c}_1, \boldsymbol{c}_2)|$$

Marginalizing one more time

$$\Delta_{g \cup h} = \log\left(\sum_{\mathbf{b}/\exists i:\mathbf{b}_i=\boldsymbol{c}_g\cup\boldsymbol{c}_h} p(\mathbf{b},\mathbf{X},K)\right) - \log\left(\sum_{\mathbf{b}/\exists i,j:\mathbf{b}_i=\boldsymbol{c}_g,\mathbf{b}_j=\boldsymbol{c}_h} p(\mathbf{b},\mathbf{X},K)\right)$$

So we have marginalized outside of $\boldsymbol{c}_g \cup \boldsymbol{c}_h$. By doing so we get a simpler formula which only involve $|\boldsymbol{cutset}(G, \boldsymbol{c}_g, \boldsymbol{c}_h)|$ and avoid the computation of $|\mathcal{T}_{G \diamond \boldsymbol{c}}|$.

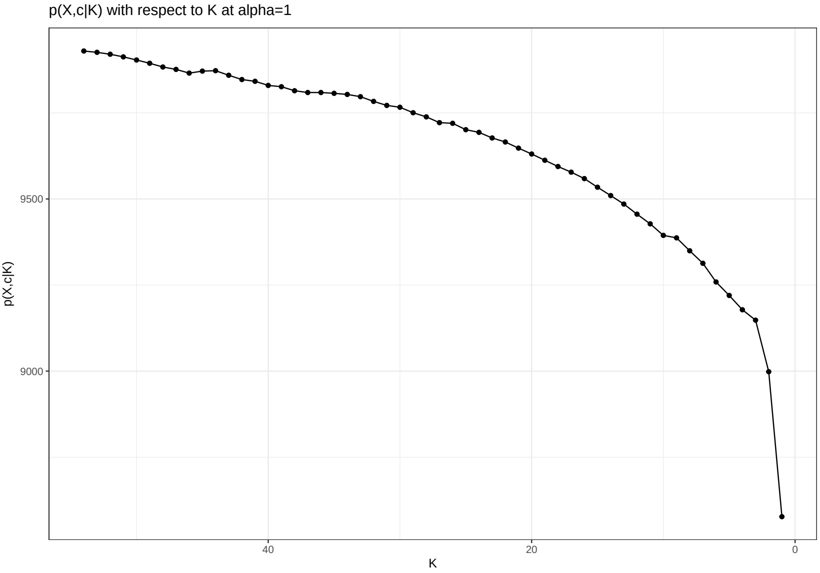
Marginalizing one more time

$$\Delta_{g \cup h} = \mathscr{L}^{obs}(\mathbf{X}, \boldsymbol{c}_g \cup \boldsymbol{c}_h) - \mathscr{L}^{obs}(\mathbf{X}, \boldsymbol{c}_g) - \mathscr{L}^{obs}(\mathbf{X}, \boldsymbol{c}_h) + \log\left(\frac{|\mathcal{T}_{G[\boldsymbol{c}_h \cup \boldsymbol{c}_h]}|}{|\boldsymbol{cutset}(G, \boldsymbol{c}_g, \boldsymbol{c}_h)||\mathcal{T}_{G[\boldsymbol{c}_h]}||\mathcal{T}_{G[\boldsymbol{c}_g]}|}\right) \tag{10}$$

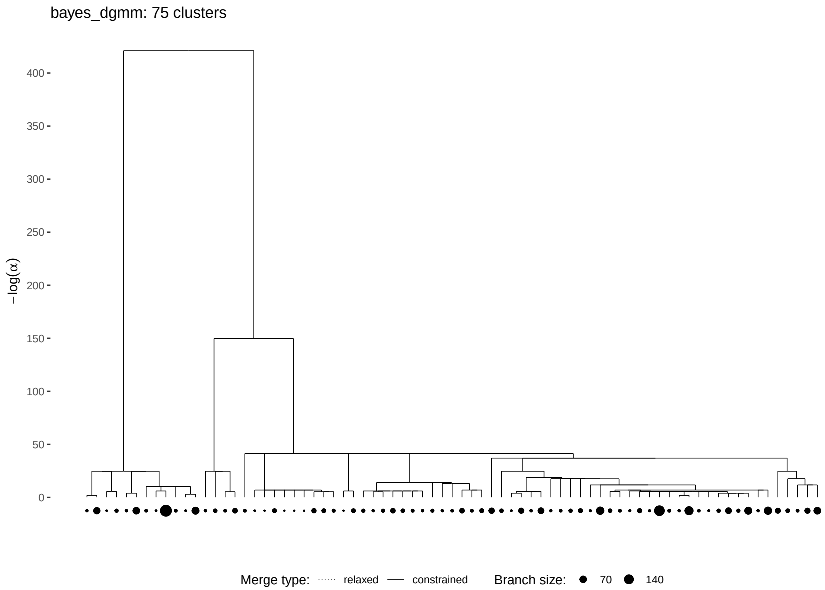
Rank one updates

$$L(G[\boldsymbol{c}_g \cup \boldsymbol{c}_h]) = \begin{pmatrix} L(G[\boldsymbol{c}_g]) & 0 \\ 0 & L(G[\boldsymbol{c}_h]) \end{pmatrix} + \sum_{u,v \in \boldsymbol{cutset}(G,\boldsymbol{c}_g,\boldsymbol{c}_h)} l(u,v)^t l(u,v)$$

Results shenzen

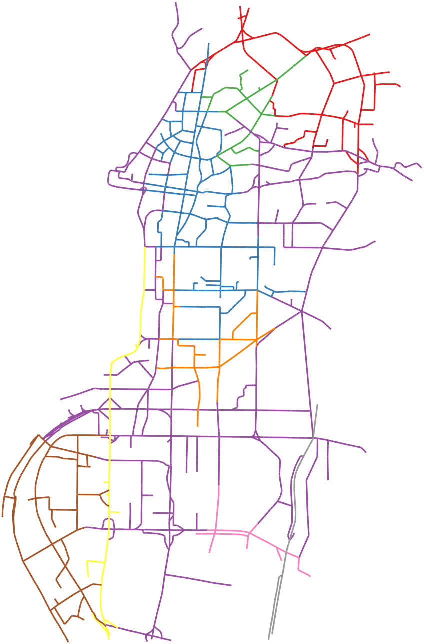


Results shenzen



Shenzen clustering results

8h30−8h45



Shenzen clustering results

8h30−8h45