# Self-organizing maps for labelled graphs

Nathalie Villa-Vialaneix[1,2], Madalina Olteanu[2], Christine Cierco-Ayrolles[1]

http://www.nathalievilla.org

nathalie.villa@toulouse.inra.fr, madalina.olteanu@univ-paris1.fr, christine.cierco@toulouse.inra.fr

EGC'2013 TOULOUSE - Atelier FGG (2013/01/29)

(1) INRA (2) SAMM

# Outline

# Notations and examples

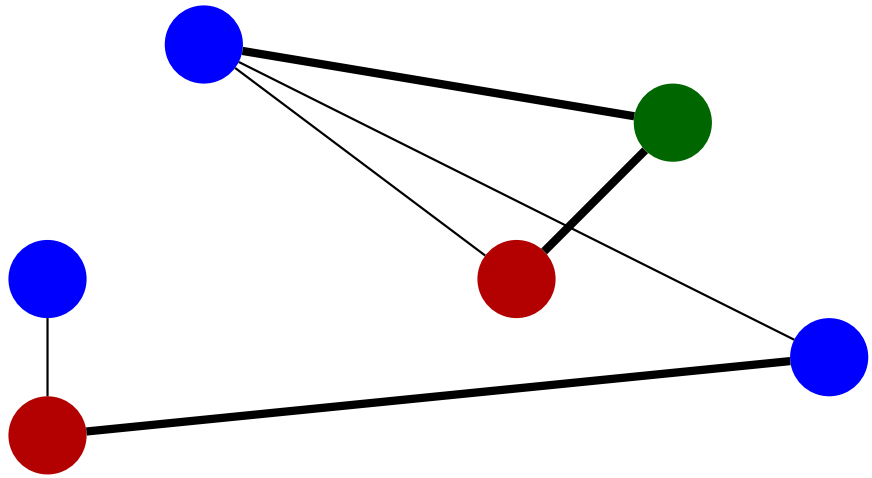
**Data**: A weighted undirected **network** represented by a graph $\mathcal{G}$ with $n$ nodes $x_1, \ldots, x_n$ with **weight matrix** $W$: $W_{ij} = W_{ji}$ and $W_{ii} = 0$.

# Notations and examples

**Data**: A weighted undirected **network** represented by a graph $\mathcal{G}$ with $n$ nodes $x_1, \ldots, x_n$ with **weight matrix** $W$: $W_{ij} = W_{ji}$ and $W_{ii} = 0$.
For every node, **additional information** is provided: either numerical variables, factors, textual information... or a combination of all of them.

# Notations and examples

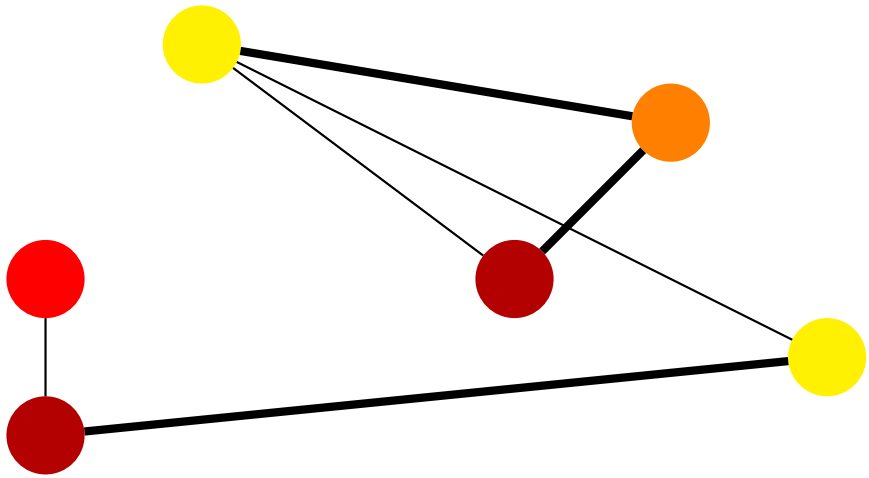
**Data**: A weighted undirected **network** represented by a graph $\mathcal{G}$ with $n$ nodes $x_1, \ldots, x_n$ with **weight matrix** $W$: $W_{ij} = W_{ji}$ and $W_{ii} = 0$.
For every node, **additional information** is provided: either numerical variables, **factors**, textual information... or a combination of all of them.



**Examples**: Gender in a social network, Functional group of a gene in a gene interaction network...

# Notations and examples

**Data**: A weighted undirected **network** represented by a graph $\mathcal{G}$ with $n$ nodes $x_1, \ldots, x_n$ with **weight matrix** $W$: $W_{ij} = W_{ji}$ and $W_{ii} = 0$.
For every node, **additional information** is provided: either **numerical information**, factors, textual information... or a combination of all of them.



**Examples**: Weight of people in a social network, Number of visits of a web page in WWW...

# Purpose

**Node clustering**: find communities, i.e., groups of “close” nodes in the graph; close meaning:

- **densely connected** and sharing (comparatively) a few links with the other groups (“communities”);

# Purpose

**Node clustering**: find communities, i.e., groups of "close" nodes in the graph; close meaning:

- **densely connected** and sharing (comparatively) a few links with the other groups ("communities");
- but also **having similar labels**.

# Purpose

**Node clustering**: find communities, i.e., groups of "close" nodes in the graph; close meaning:

- **densely connected** and sharing (comparatively) a few links with the other groups ("communities");
- but also **having similar labels**.

Here: **self-organizing map** approach to produce a map of the nodes (clustering+visualization).

# Purpose

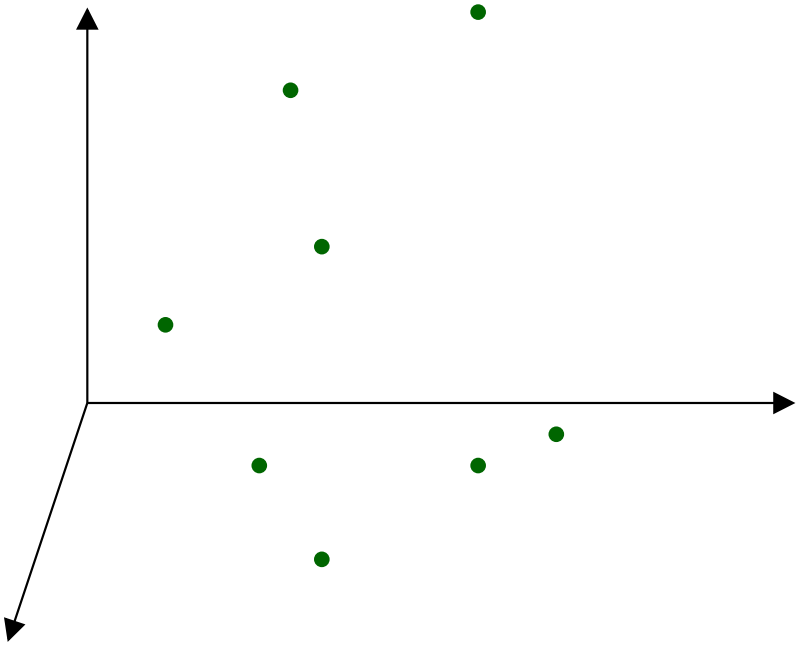
**Node clustering**: find communities, i.e., groups of "close" nodes in the graph; close meaning:

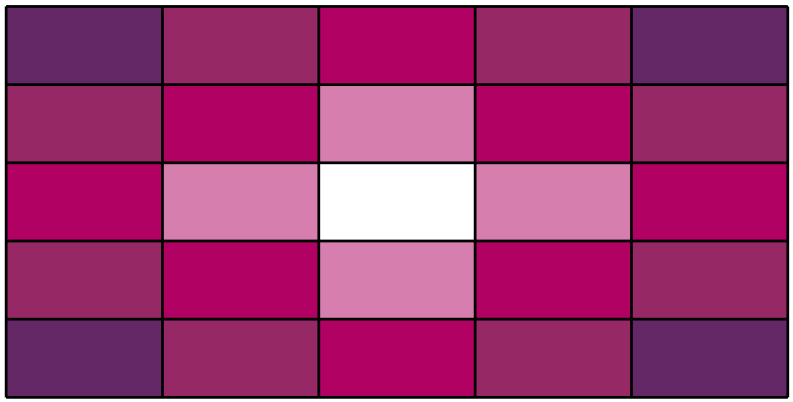
- **densely connected** and sharing (comparatively) a few links with the other groups ("communities");
- but also **having similar labels**.

Here: **self-organizing map** approach to produce a map of the nodes (clustering+visualization). Multiple sources of information are handled by combining **kernels**.

# Outline

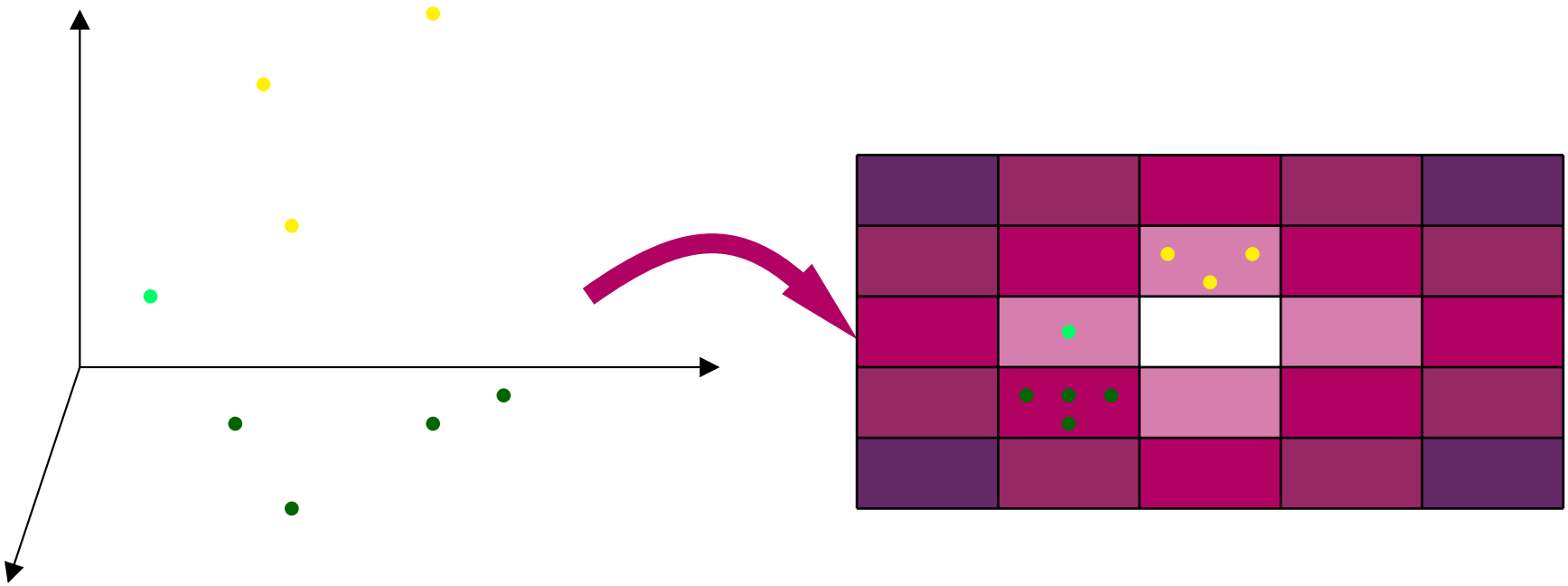
# Basics on SOM





Project the data on a squared grid (each square of the grid is a cluster)

# Basics on SOM



Project the data on a squared grid (each square of the grid is a cluster) st:

- the observations in a same cluster are “close” in the original space;
- the observations in two close clusters are also (less) “close” in the original space;
- the observations in two distant clusters are distant in the original space.

# Basics on SOM

- the map is made of **neurons** (visually symbolized by, e.g., rectangles), 1...$M$, each associated to a **prototype** $p_i$ (a prototype is a "representer" of the neuron in the original dataset);

# Basics on SOM

- the map is made of **neurons** (visually symbolized by, e.g., rectangles), 1...$M$, each associated to a **prototype** $p_i$ (a prototype is a "representer" of the neuron in the original dataset);
- the map is equipped with a **neighborhood relationship**, i.e., a "distance" (actually a dissimilarity) between neurons, $\delta$;

# Basics on SOM

- the map is made of **neurons** (visually symbolized by, e.g., rectangles), 1...$M$, each associated to a **prototype** $p_i$ (a prototype is a "representer" of the neuron in the original dataset);
- the map is equipped with a **neighborhood relationship**, i.e., a "distance" (actually a dissimilarity) between neurons, $\delta$;
- goal: find the best mapping $f(x_i) \in \{1, \ldots, M\}$ of the observations $x_i$ on the map **minimizing the energy**

$$\mathcal{E} = \sum_{i=1}^{n} \sum_{j=1}^{M} h(\delta(f(x_i), j)) \|x_i - p_i\|^2.$$

i.e., each data is assigned to a neuron so that $x_i$ is:

- "close" to the prototype of $f(x_i)$;
- "close" (to a lesser extent) to the prototypes of the neighboring neurons of $f(x_i)$;
- "distant" to the prototypes of the neurons that are distant of $f(x_i)$.

**(topology preservation)**

# on-line SOM

**Data**: $x_1, \ldots, x_n \in \mathbb{R}^d$

1: **Initialization**: randomly set $p_1^0,...,p_M^0$ in $\mathbb{R}^d$
2: **for** $t = 1 \to T$ **do**
3: Randomly choose $i \in \{1, \ldots, n\}$
4: **Assignment**

$$f^t(x_i) \leftarrow \arg \min_{j=1,...,M} \|x_i - p_j^{t-1}\|_{\mathbb{R}^d}$$

5: **for all** $j = 1 \to M$ **do** **Representation**
6: $p_j^t \leftarrow p_j^{t-1} + \mu_t h^t(\delta(f^t(x_i), j)) \left(\mathbf{1}_i - p_j^{t-1}\right)$
7: **end for**
8: **end for**

where $h^t$ is a decreasing function which reduces the neighborhood when $t$ increases and $\mu_t$ is generally of order $1/t$.

# on-line SOM

**Data**: $x_1, \ldots, x_n \in \mathbb{R}^d$

1: **Initialization**: randomly set $p_1^0,...,p_M^0$ in $\mathbb{R}^d$
2: **for** $t = 1 \to T$ **do**
3: Randomly choose $i \in \{1, \ldots, n\}$
4: **Assignment**

$$f^t(x_i) \leftarrow \arg \min_{j=1,...,M} \|x_i - p_j^{t-1}\|_{\mathbb{R}^d}$$

5: **for all** $j = 1 \to M$ **do** **Representation**
6: $p_j^t \leftarrow p_j^{t-1} + \mu_t h^t(\delta(f^t(x_i), j)) \left(\mathbf{1}_i - p_j^{t-1}\right)$
7: **end for**
8: **end for**

where $h^t$ is a decreasing function which reduces the neighborhood when $t$ increases and $\mu_t$ is generally of order $1/t$.

**Problem with non numerical data**: definitions of $\|.\|$ and $p_j$???

# Kernels/Multiple kernel

## What is a kernel?

$(x_i) \in \mathcal{G}$, $(K(x_i, x_j))_{ij}$ st: $K(x_i, x_j) = K(x_j, x_i)$ and
$\forall (\alpha_i)_i,\ \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) \geq 0$. In this case **[Aronszajn, 1950]**,

$$\exists\ (\mathcal{H}, \langle ., . \rangle)\ ,\ \Phi : \mathcal{G} \rightarrow \mathcal{H} \text{ st :} \qquad K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

# Kernels/Multiple kernel

## What is a kernel?

$(x_i) \in \mathcal{G}$, $(K(x_i, x_j))_{ij}$ st: $K(x_i, x_j) = K(x_j, x_i)$ and
$\forall (\alpha_i)_i,\ \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) \geq 0$. In this case **[Aronszajn, 1950]**,

$$\exists\ (\mathcal{H}, \langle ., . \rangle)\ ,\ \Phi : \mathcal{G} \to \mathcal{H} \text{ st :} \qquad K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

**Examples:**

- **nodes of a graph**: Heat kernel $K = e^{-\beta L}$ or $K = L^+$ where $L$ is the Laplacian **[Kondor and Lafferty, 2002, Smola and Kondor, 2003, Fouss et al., 2007]**
- **numerical variables**: Gaussian kernel $K(x_i, x_j) = e^{-\beta \|x_i - x_j\|^2}$;
- **text**: **[Watkins, 2000]**...

# Kernels/Multiple kernel

## What is a kernel?

$(x_i) \in \mathcal{G}$, $(K(x_i, x_j))_{ij}$ st: $K(x_i, x_j) = K(x_j, x_i)$ and
$\forall (\alpha_i)_i,\ \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) \geq 0$. In this case **[Aronszajn, 1950]**,

$$\exists\, (\mathcal{H}, \langle ., . \rangle)\,,\ \Phi : \mathcal{G} \to \mathcal{H} \text{ st}: \qquad K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

**Examples:**

- **nodes of a graph**: Heat kernel $K = e^{-\beta L}$ or $K = L^+$ where $L$ is the Laplacian **[Kondor and Lafferty, 2002, Smola and Kondor, 2003, Fouss et al., 2007]**
- **numerical variables**: Gaussian kernel $K(x_i, x_j) = e^{-\beta \|x_i - x_j\|^2}$;
- **text**: **[Watkins, 2000]**...

Combining kernels (**[Rakotomamonjy et al., 2008]** for SVM):

$$K = \sum_{d=1}^{D} \alpha_d K_d, \qquad \alpha_d \geq \text{ and } \sum_d \alpha_d = 1$$

# multiple kernel SOM

**Prototypes initialization**: randomly set $p_j^0 = \sum_{i=1}^n \gamma_{ij}^0 \Phi(x_i)$ st $\gamma_{ij}^0 \geq 0$ and $\sum_i \gamma_{ij}^0 = 1$

**Kernel initialization**: set $(\alpha_d^0)$ st $\alpha_d^0 \geq 0$ and $\sum_d \alpha_d = 1$ (e.g., $\alpha_d^0 = 1/D$); $K^0 \leftarrow \sum_d \alpha_d^0 K_d$

**for** $t = 1 \rightarrow T$ **do**

  Randomly choose $i \in \{1, \ldots, n\}$

  **Assignment**

$$f^t(x_i) \leftarrow \arg \min_{j=1,\ldots,M} \|\Phi(x_i) - p_j^{t-1}\|^2_{\mathcal{H}(K^t)}$$

  **for all** $j = 1 \rightarrow M$ **do** **Representation**

    $\gamma_j^t \leftarrow \gamma_j^{t-1} + \mu_t h^t(\delta(f^t(x_i), j)) \left(\mathbf{1}_i - \gamma_j^{t-1}\right)$

  **end for**

**end for**

# multiple kernel SOM

1: **Prototypes initialization**: randomly set $p_j^0 = \sum_{i=1}^n \gamma_{ij}^0 \Phi(x_i)$ st $\gamma_{ij}^0 \geq 0$ and $\sum_i \gamma_{ij}^0 = 1$
2: **Kernel initialization**: set $(\alpha_d^0)$ st $\alpha_d^0 \geq 0$ and $\sum_d \alpha_d = 1$ (e.g., $\alpha_d^0 = 1/D$); $K^0 \leftarrow \sum_d \alpha_d^0 K_d$
3: **for** $t = 1 \rightarrow T$ **do**
4: Randomly choose $i \in \{1, \ldots, n\}$
5: **Assignment**

$$f^t(x_i) \leftarrow \arg \min_{j=1,\ldots,M} \sum_{ll'} \gamma_{lj}^{t-1} \gamma_{l'j}^{t-1} K^t(x_l, x_{l'}) - 2 \sum_l \gamma_{lj}^{t-1} K^t(x_l, x_i)$$

6: **for all** $j = 1 \rightarrow M$ **do Representation**
7: $\gamma_j^t \leftarrow \gamma_j^{t-1} + \mu_t h^t(\delta(f^t(x_i), j)) \left(\mathbf{1}_i - \gamma_j^{t-1}\right)$
8: **end for**
9:
10: **end for**

# multiple kernel SOM

1: **Prototypes initialization**: randomly set $p_j^0 = \sum_{i=1}^n \gamma_{ij}^0 \Phi(x_i)$ st $\gamma_{ij}^0 \geq 0$ and $\sum_i \gamma_{ij}^0 = 1$
2: **Kernel initialization**: set $(\alpha_d^0)$ st $\alpha_d^0 \geq 0$ and $\sum_d \alpha_d = 1$ (e.g., $\alpha_d^0 = 1/D$); $K^0 \leftarrow \sum_d \alpha_d^0 K_d$
3: **for** $t = 1 \rightarrow T$ **do**
4: Randomly choose $i \in \{1, \ldots, n\}$
5: **Assignment**

$$f^t(x_i) \leftarrow \arg \min_{j=1,\ldots,M} \sum_{ll'} \gamma_{lj}^{t-1} \gamma_{l'j}^{t-1} K^t(x_l, x_{l'}) - 2 \sum_l \gamma_{lj}^{t-1} K^t(x_l, x_i)$$

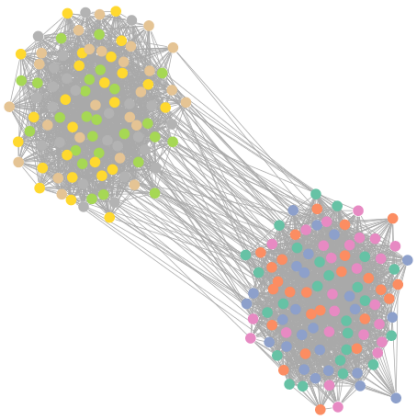
6: **for all** $j = 1 \rightarrow M$ **do Representation**
7: $\gamma_j^t \leftarrow \gamma_j^{t-1} + \mu_t h^t(\delta(f^t(x_i), j)) \left(\mathbf{1}_i - \gamma_j^{t-1}\right)$
8: **end for**
9: **Kernel update** $\alpha_d^t \leftarrow \alpha_d^{t-1} + \nu_t \frac{\partial \mathcal{E}|_{x_i}}{\partial \alpha_d}$ and $K^t \leftarrow \sum_d \alpha_d^t K_d$
10: **end for**

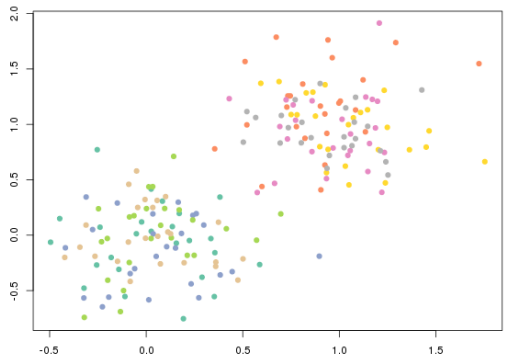
# Outline

# Example1: simulated data

## Graph with 200 nodes classified in 8 groups:

- **graph**: Erdös Reyni models: groups 1 to 4 and groups 5 to 8 with intra-group edge probability 0.3 and inter-group edge probability 0.01;
- **numerical data**: nodes labelled with 2-dimensional Gaussian vectors: odd groups $\mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0.3 & 0\\0 & 0.3\end{pmatrix}\right)$;
- **factor** with two levels: groups 1, 2, 5 and 7: first level; other groups: second level.






Only the knownledge on the three datasets can discriminate all 8 groups.

# Experiment

**Kernels**: graph: $L^+$, numerical data: Gaussian kernel; factor: another Gaussian kernel on the disjunctive recoding

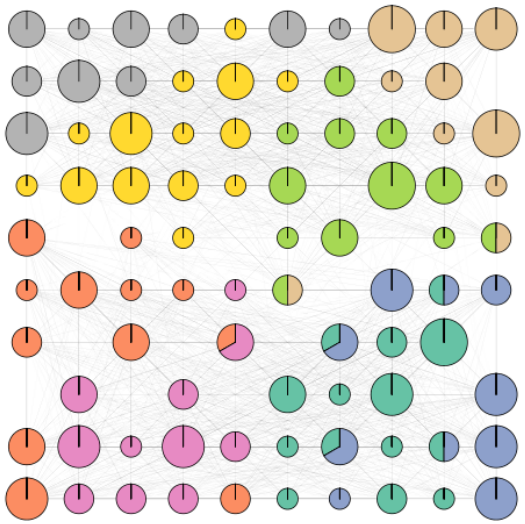
# Experiment

**Kernels**: graph: $L^+$, numerical data: Gaussian kernel; factor: another Gaussian kernel on the disjunctive recoding
**Comparison**: on 100 randomly generated datasets as previously described:

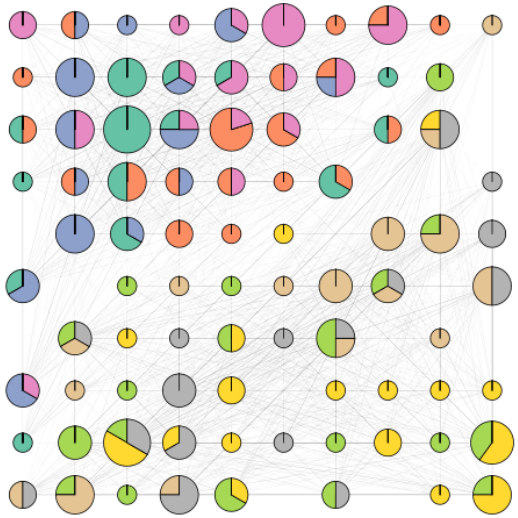
- multiple kernel SOM with all three data;
- kernel SOM with a single dataset;
- kernel SOM with two datasets or all three datasets in a single (Gaussian) kernel.

# An example

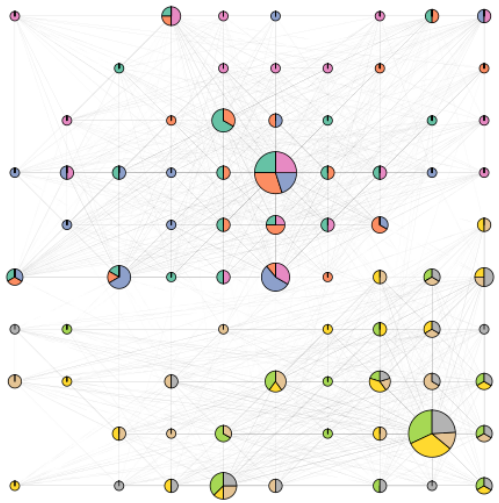
MK-SOM



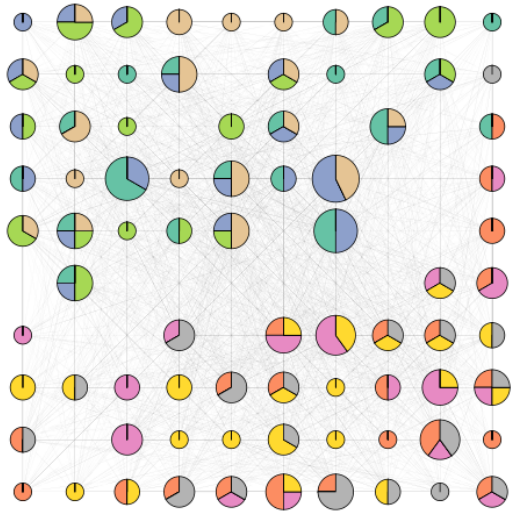
All in one kernel



Graph only



Numerical variables only



# Numerical comparison

(over 100 simulations) with **mutual information**

$$\sum_{ij} \frac{|C_i \cap \tilde{C}_j|}{200} \log \frac{|C_i \cap \tilde{C}_j|}{|C_i| \times |\tilde{C}_j|}$$

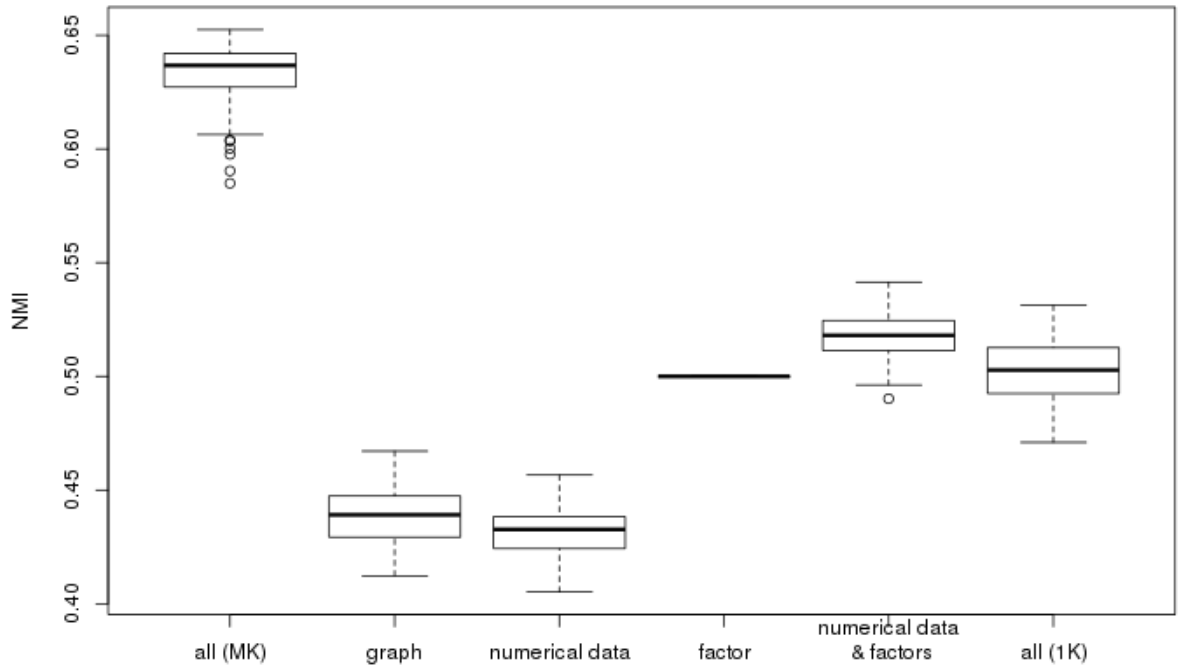
(adjusted version, equal to 1 if partitions are identical **[Danon et al., 2005]**).

# Numerical comparison

(over 100 simulations) with **mutual information**

$$\sum_{ij} \frac{|C_i \cap \tilde{C}_j|}{200} \log \frac{|C_i \cap \tilde{C}_j|}{|C_i| \times |\tilde{C}_j|}$$

(adjusted version, equal to 1 if partitions are identical **[Danon et al., 2005]**).



# Example 2: Data coming from a medieval corpus

## Example from [Boulet et al., 2008]

`http://graphcomp.univ-tlse2.fr/` In the "Archive départementales du Lot" (Cahors, France), big corpus of 5000 transactions (mostly land charters)

- coming from 4 "seigneuries" (about 25 little villages) in South West of France;
- being established between 1240 and 1520 (just before and after the hundred years' war).

# Graph description

**Graph**:

- nodes: 1446 individuals directly involved in the transactions (1 446 individuals);
- edges: the two individuals are involved in the same transaction (3 192 edges).

# Graph description

**Graph**:

- nodes: 1446 individuals directly involved in the transactions (1 446 individuals);
- edges: the two individuals are involved in the same transaction (3 192 edges).

**Labels** on nodes:

- **numerical**: average date of activity;
- **text**: family name of the individual.

# Graph description

**Graph**:

- nodes: 1446 individuals directly involved in the transactions (1 446 individuals);
- edges: the two individuals are involved in the same transaction (3 192 edges).

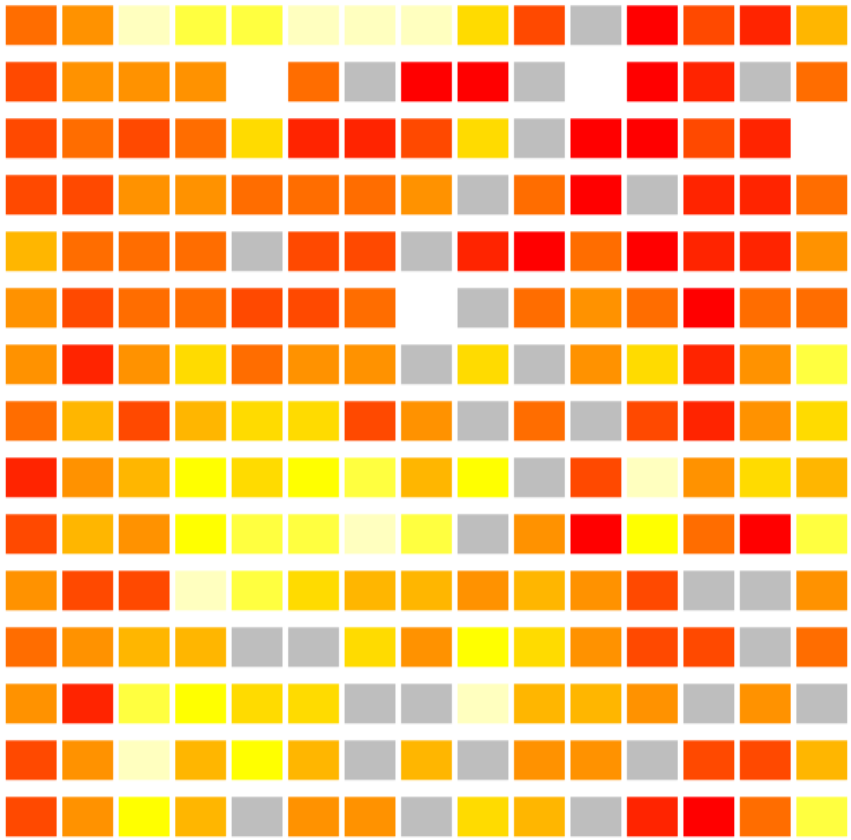
**Labels** on nodes:

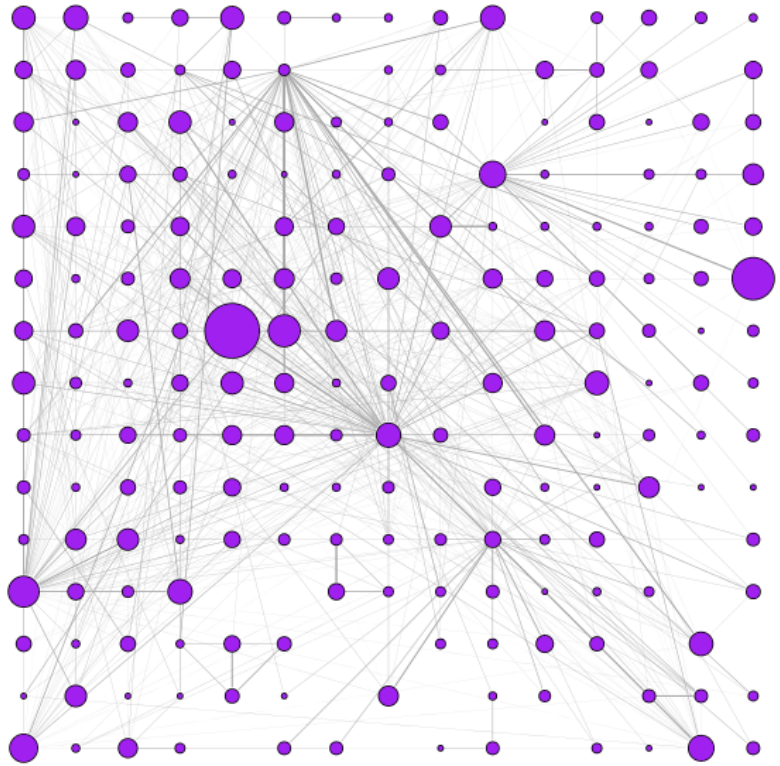
- **numerical**: average date of activity;
- **text**: family name of the individual.

**Kernels**:

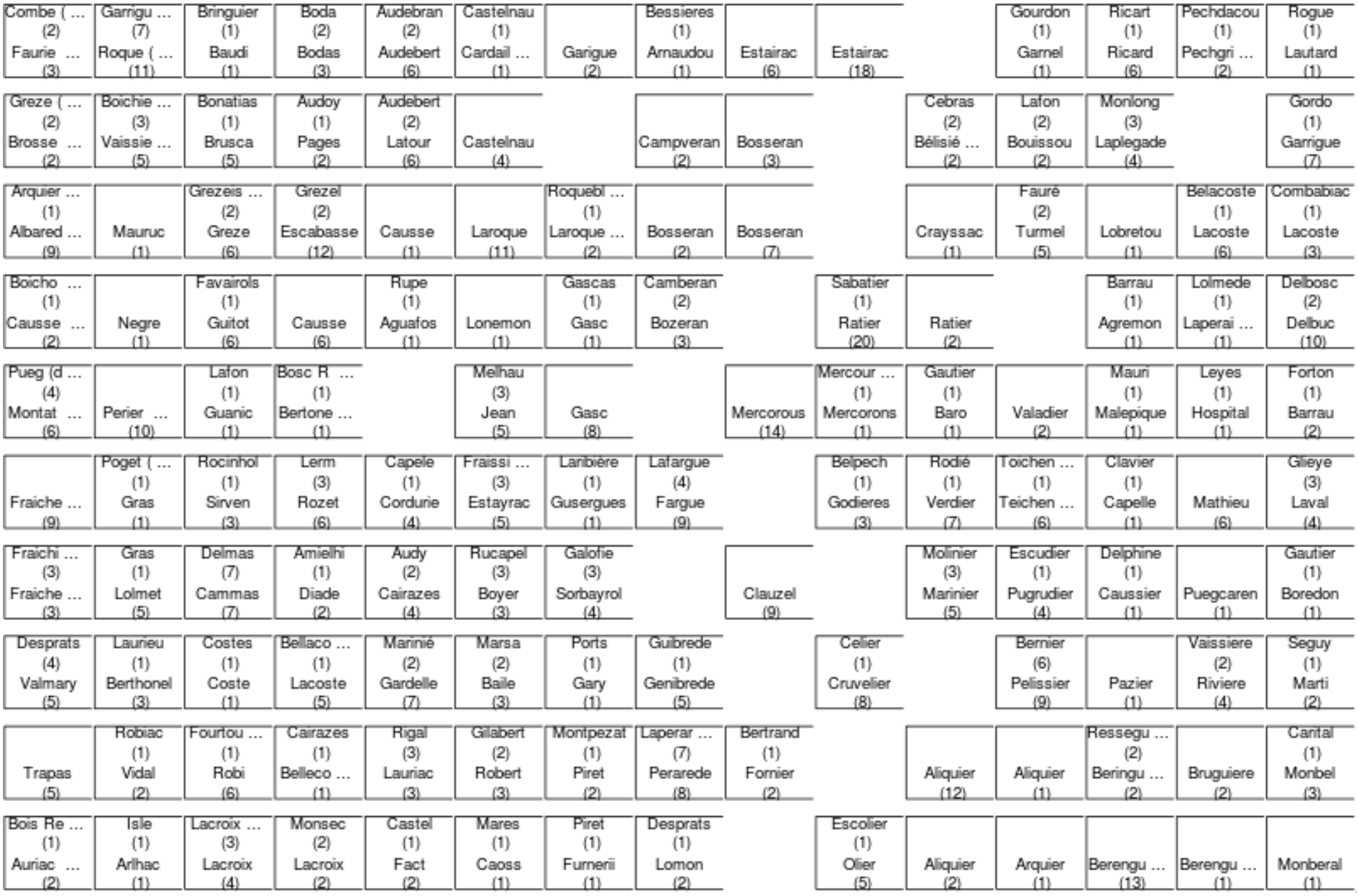
- $L^+$ for the graph;
- linear kernel for the dates;
- spectral string kernel for the family names **[Karatzoglou and Feinerer, 2010]** (uses common strings having a length larger than 4).

# Date and graph maps





# Name map


Combe ( ... (2) Faurie ... (3)
Garrigu ... (7) Roque ( ... (11)
Bringuier (1) Baudi (1)
Boda (2) Bodas (3)
Audebran (2) Audebert (6)
Castelnau (1) Cardail ... (1)
Garigue (2)
Bessieres (1) Arnaudou (1)
Estairac (6)
Estairac (18)
Gourdon (1) Garnel (1)
Ricart (1) Ricard (6)
Pechdacou (1) Pechgri ... (2)
Rogue (1) Lautard (1)
Greze ( ... (2) Brosse ... (2)
Boichie ... (3) Vaissie ... (5)
Bonatias (1) Brusca (5)
Audoy (1) Pages (2)
Audebert (2) Latour (6)
Castelnau (4)
Campveran (2)
Bosseran (3)
Cebras (2) Bélisié ... (2)
Lafon (2) Bouissou (2)
Monlong (3) Laplegade (4)
Gordo (1) Garrigue (7)
Arquier ... (1) Albared ... (9)
Mauruc (1)
Grezeis ... (2) Greze (6)
Grezel (2) Escabasse (12)
Causse (1)
Laroque (11)
Roquebl ... (1) Laroque ... (2)
Bosseran (2)
Bosseran (7)
Crayssac (1)
Fauré (2) Turmel (5)
Lobretou (1)
Belacoste (1) Lacoste (6)
Combabiac (1) Lacoste (3)
Boicho ... (1) Causse ... (2)
Negre (1)
Favairols (1) Guitot (6)
Causse (6)
Rupe (1) Aguafos (1)
Lonemon (1)
Gascas (1) Gasc (1)
Camberan (2) Bozeran (3)
Sabatier (1) Ratier (20)
Ratier (2)
Barrau (1) Agremon (1)
Lolmede (1) Laperai ... (1)
Delbosc (2) Delbuc (10)
Pueg (d ... (4) Montat ... (6)
Perier ... (10)
Lafon (1) Guanic (1)
Bosc R ... (1) Bertone ... (1)
Melhau (3) Jean (5)
Gasc (8)
Mercorous (14)
Mercour ... (1) Mercorons (1)
Gautier (1) Baro (1)
Valadier (2)
Mauri (1) Malepique (1)
Leyes (1) Hospital (1)
Forton (1) Barrau (2)
Fraiche ... (9)
Poget ( ... (1) Gras (1)
Rocinhol (1) Sirven (3)
Lerm (3) Rozet (6)
Capele (1) Cordurie (4)
Fraissi ... (3) Estayrac (5)
Lanbière (1) Gusergues (1)
Lafargue (4) Fargue (9)
Belpech (1) Godieres (3)
Rodié (1) Verdier (7)
Toichen ... (1) Teichen ... (6)
Clavier (1) Capelle (1)
Mathieu (6)
Glieye (3) Laval (4)
Fraichi ... (3) Fraiche ... (3)
Gras (1) Lolmet (5)
Delmas (7) Cammas (7)
Amielhi (1) Diade (2)
Audy (2) Cairazes (4)
Rucapel (3) Boyer (3)
Galofre (3) Sorbayrol (4)
Clauzel (9)
Molinier (3) Marinier (5)
Escudier (1) Pugrudier (4)
Delphine (1) Caussier (1)
Puegcaren (1)
Gautier (1) Boredon (1)
Desprats (4) Valmary (5)
Laurieu (1) Berthonel (3)
Costes (1) Coste (1)
Bellaco ... (1) Lacoste (5)
Mariné (2) Gardelle (7)
Marsa (2) Baile (3)
Ports (1) Gary (1)
Guibrede (1) Genibrede (5)
Celier (1) Cruvelier (8)
Bernier (6) Pelissier (9)
Pazier (1)
Vaissiere (2) Riviere (4)
Seguy (1) Marti (2)
Trapas (5)
Robiac (1) Vidal (2)
Fourtou ... (1) Robi (6)
Cairazes (1) Belleco ... (1)
Rigal (3) Lauriac (3)
Gilabert (2) Robert (3)
Montpezat (1) Piret (2)
Laperar ... (7) Perarede (8)
Bertrand (1) Fornier (2)
Aliquier (12)
Aliquier (1)
Ressegu ... (2) Beringu ... (2)
Bruguiere (2)
Carital (1) Monbel (3)
Bois Re ... (1) Auriac ... (2)
Isle (1) Arlhac (1)
Lacroix ... (3) Lacroix (4)
Monsec (2) Lacroix (2)
Castel (1) Fact (2)
Mares (1) Caoss (1)
Piret (1) Furnerii (1)
Desprats (1) Lomon (2)
Escolier (1) Olier (5)
Aliquier (2)
Arquier (1)
Berengu ... (13)
Berengu ... (1)
Monberal (1)

# Conclusion

- finding communities in graph, while taking into account labels;
- uses multiple kernel and automatically tunes the combination;
- the method gives relevant communities according to all sources of information and a well-organized map.

# Conclusion

- finding communities in graph, while taking into account labels;
- uses multiple kernel and automatically tunes the combination;
- the method gives relevant communities according to all sources of information and a well-organized map.

**Thank you for your attention...**

# References


Aronszajn, N. (1950).

Theory of reproducing kernels.
*Transactions of the American Mathematical Society*, 68(3):337–404.

Boulet, R., Jouve, B., Rossi, F., and Villa, N. (2008).

Batch kernel SOM and related laplacian methods for social network analysis.
*Neurocomputing*, 71(7-9):1257–1273.

Danon, L., Diaz-Guilera, A., Duch, J., and Arenas, A. (2005).

Comparing community structure identification.
*Journal of Statistical Mechanics*, page P09008.

Fouss, F., Pirotte, A., Renders, J., and Saerens, M. (2007).

Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation.
*IEEE Trans Knowl Data En*, 19(3):355–369.

Karatzoglou, A. and Feinerer, I. (2010).

Kernel-based machine learning for fast text mining in R.
*Comput Statist Data Anal*, 54:290–297.

Kondor, R. and Lafferty, J. (2002).

Diffusion kernels on graphs and other discrete structures.
In *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322.

Rakotomamonjy, A., Bach, F., Canu, S., and Grandvalet, Y. (2008).

SimpleMKL.
*J Mach Learn Res*, 9:2491–2521.

Smola, A. and Kondor, R. (2003).

Kernels and regularization on graphs.
In Warmuth, M. and Schölkopf, B., editors, *Proceedings of the Conference on Learning Theory (COLT) and Kernel Workshop*, Lecture Notes in Computer Science, pages 144–158.

Watkins, C. (2000).

Dynamic alignment kernels.

In Smola, A., Bartlett, P., Schölkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 39–50, Cambridge, MA, USA. MIT P.