Spectral graph wavelets: a tool for multiscale community mining in graphs

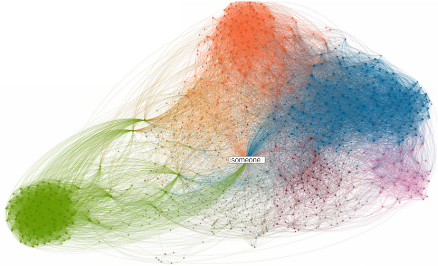
Nicolas Tremblay

under the supervision of: Pierre Borgnat

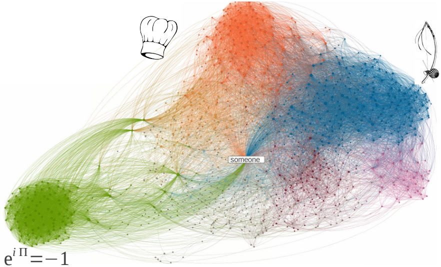
Laboratoire de Physique de l'ENS Lyon
CNRS UMR 5672

LIP6, Paris
April 11th 2013

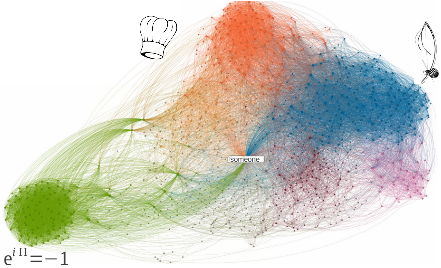
Purpose of community detection?



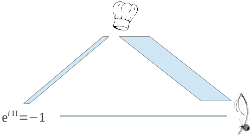
Purpose of community detection?



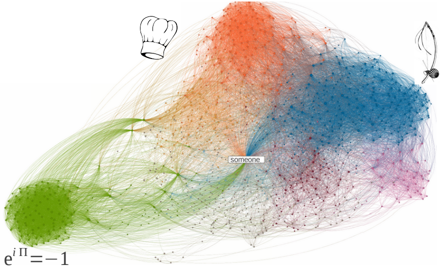
Purpose of community detection?



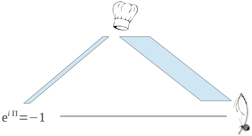
1) Gives us a sketch:



Purpose of community detection?



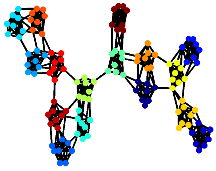
1) Gives us a sketch:



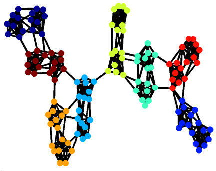
2) Gives us intuition:

Multiscale community structure in a graph

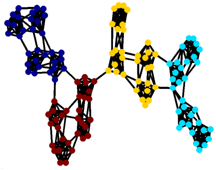
finest scale (16 com.):



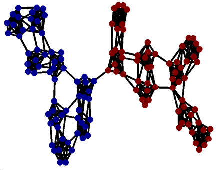
coarser scale (8 com.):



even coarser scale (4 com.):

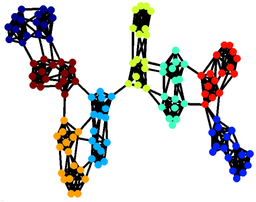


coarsest scale (2 com.):



Multiscale community structure in a graph

Classical community detection algorithm do not have this "scale-vision" of a graph. Modularity optimisation finds:

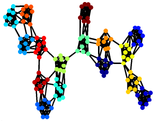


Where the modularity function reads:

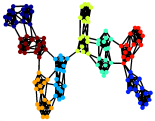
$$Q = \frac{1}{2N} \sum_{ij} \left[A_{ij} - \frac{d_i d_j}{2N} \right] \delta(c_i, c_j)$$

Multiscale community structure in a graph

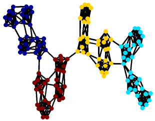
Q=0.80 :



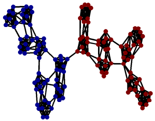
Q=0.83 :



Q=0.74 :



Q=0.50 :



All representations are correct but
modularity optimisation favours one.

Related work

- Lambiotte, "Multiscale modularity in complex networks" (2010)
- Schaub et al., "Markov dynamics as a zooming lens for multiscale community detection: non clique-like communities and the field-of-view limit" (2012)
- Arenas et al., "Analysis of the structure of complex networks at different resolution levels" (2008)
- Reichardt et al., "Statistical Mechanics of Community Detection" (2006)
- Mucha et al., "Community Structure in Time-Dependent, Multiscale, and Multiplex Networks" (2010)

Purpose of this work

Develop a scale dependent community mining tool

General Ideas

- Take advantage of local information encoded in Graph Wavelets
- Cluster together nodes whose local environments are similar

Notations

$\mathcal{G} = (V, E, w)$	a weighted graph	
$N = \|V\|$	number of nodes	
A	adjacency matrix	$A(i,j) = w_{ij}$
d	vector of strengths	$d_i = \sum_{j \in V} w_{ij}$

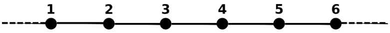
Laplacian matrix

L	laplacian matrix	$L = diag(d) - A$
(λ_i)	L's eigenvalues	$0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{N-1}$
(χ_i)	L's normalized eigenvectors	$L\,\chi_i = \lambda_i\,\chi_i$

Objective

f : signal defined on V $\longleftrightarrow$ $\hat{f}$: Fourier transform of f

A simple example: the straight line



$$\longleftrightarrow \quad L = \begin{pmatrix} \cdots & & & & & & \\ \cdots & -1 & 0 & 0 & 0 & 0 & \\ \cdots & 2 & -1 & 0 & 0 & 0 & \\ & -1 & 2 & -1 & 0 & 0 & \\ & 0 & -1 & 2 & -1 & 0 & \\ & 0 & 0 & -1 & 2 & -1 & \\ & 0 & 0 & 0 & -1 & 2 & \cdots \\ & 0 & 0 & 0 & 0 & -1 & \cdots \\ & & & & & & \cdots \end{pmatrix}$$

On the regular line, L is the 1-D classical laplacian operator (i.e. double derivative operator): its eigenvectors are the Fourier vectors, and its eigenvalues the associated (squared) frequencies.

Fundamental analogy

On *any* graph, the eigenvectors χ_i of the Laplacian matrix L will be considered as the Fourier vectors, and its eigenvalues λ_i the associated (squared) frequencies.

The graph Fourier transform

- $\hat{f}$ is obtained from f's decomposition on the eigenvectors χ_i :

$$\hat{f} = \begin{pmatrix} < \chi_0, f > \\ < \chi_1, f > \\ < \chi_2, f > \\ \dots \\ < \chi_{N-1}, f > \end{pmatrix}$$

Define $\boldsymbol{\chi} = (\chi_0|\chi_1|\dots|\chi_{N-1})$: $\boxed{\hat{f} = \boldsymbol{\chi}^\top f}$

- Reciprocally, the inverse Fourier transform reads: $\boxed{f = \boldsymbol{\chi} \hat{f}}$
- The Parseval theorem is valid: $\forall (g, h) \quad < g, h > = < \hat{g}, \hat{h} >$

Filtering

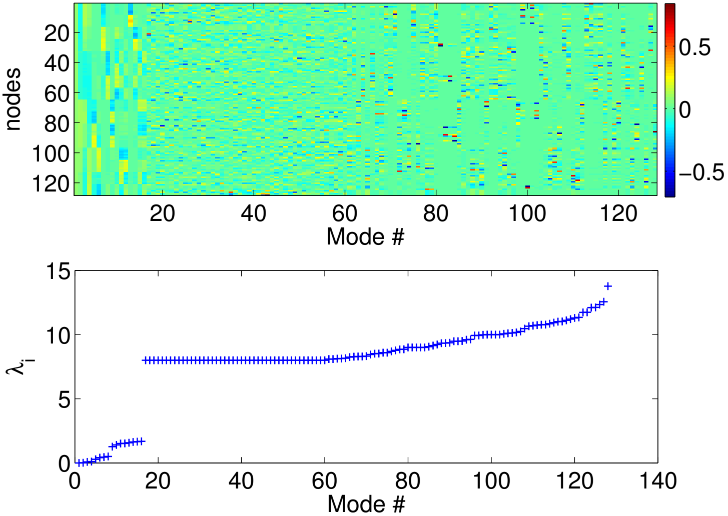
Definition of graph filtering

We define a filter function g in the Fourier space.
It is discrete and defined on the eigenvalues $\lambda_i \rightarrow g(\lambda_i)$.

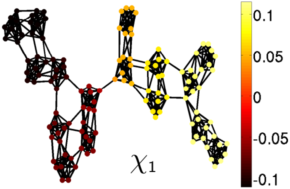
$$\hat{f}^g = \begin{pmatrix} \hat{f}(0)\, g(\lambda_0) \\ \hat{f}(1)\, g(\lambda_1) \\ \hat{f}(2)\, g(\lambda_2) \\ \cdots \\ \hat{f}(N-1)\, g(\lambda_{N-1}) \end{pmatrix} = \hat{\mathbf{G}}\, \hat{f} \text{ with } \hat{\mathbf{G}} = \begin{pmatrix} g(\lambda_0) & 0 & 0 & \dots & 0 \\ 0 & g(\lambda_1) & 0 & \dots & 0 \\ 0 & 0 & g(\lambda_2) & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & g(\lambda_{N-1}) \end{pmatrix}$$

In the node-space, the filtered signal f^g can be written: $f^g = \boldsymbol{\chi}\, \hat{\mathbf{G}}\, \boldsymbol{\chi}^\top f$

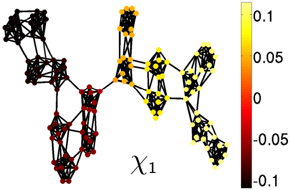
Spectral analysis: the χ_i and λ_i of the multi scale toy graph

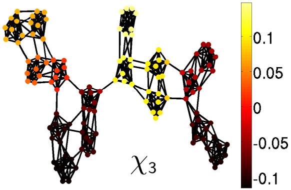


Some Fourier modes

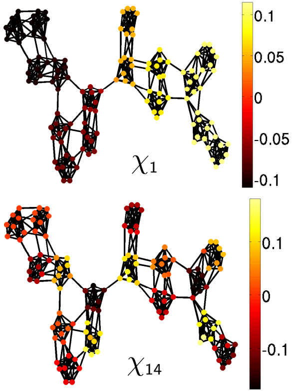


Some Fourier modes

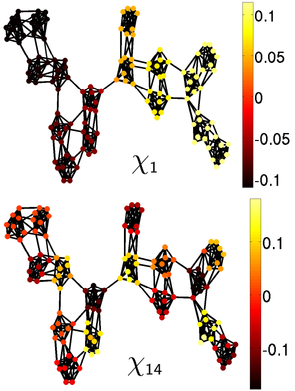




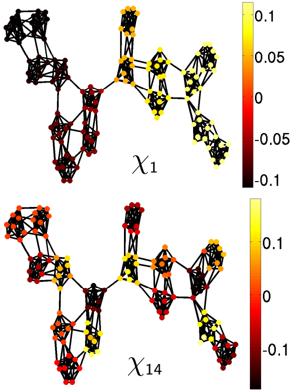
Some Fourier modes



Some Fourier modes



Some Fourier modes



The first few eigenvectors are very important for community detection

Graph Wavelets

- Fourier is a global analysis. Fourier modes (eigenvectors of the laplacian) are used in classical spectral clustering, but do not enable a scale dependent analysis: we need wavelets.
- Classical wavelets $\xrightarrow{\textit{by analogy}}$ Graph wavelets

The classical wavelets

Each wav. $\psi_{s,a}$ is derived by translating and scaling a mother wav. ψ:

$$\psi_{s,a}(x) = \frac{1}{s}\psi\left(\frac{x-a}{s}\right)$$

Equivalently, in the Fourier domain:

$$\begin{aligned}
\hat{\psi}_{s,a}(\omega) &= \int_{-\infty}^{\infty} \frac{1}{s}\psi\left(\frac{x-a}{s}\right)\exp^{-i\omega x}\,dx \\
&= \exp^{-i\omega a}\int_{-\infty}^{\infty} \frac{1}{s}\psi\left(\frac{X}{s}\right)\exp^{-i\omega X}\,dX \\
&= \exp^{-i\omega a}\int_{-\infty}^{\infty} \psi\left(X'\right)\exp^{-i\omega X'}\,dX' \\
&= \hat{\delta}_a(\omega)\,\hat{\psi}(s\omega) \qquad \text{where} \quad \delta_a = \delta(x-a)
\end{aligned}$$

One possible definition: $\psi_{s,a}(x) = \int_{-\infty}^{\infty} \hat{\delta}_a(\omega)\hat{\psi}(s\omega)\exp^{i\omega x}\,d\omega$

The classical wavelets

$$\psi_{s,a}(x) \;=\; \int_{-\infty}^{\infty} \hat{\delta}_a(\omega)\hat{\psi}(s\omega)\exp^{i\omega x}\, d\omega$$

- In this definition, $\hat{\psi}(s\omega)$ acts as a filter bank defined by scaling by a factor s a *filter kernel function* defined in the Fourier space: $\hat{\psi}(\omega)$
- The filter kernel function $\hat{\psi}(\omega)$ is necessarily a *bandpass filter* with:
  - $\hat{\psi}(0) = 0$: the mean of ψ is by definition null
  - $\lim_{\omega \to +\infty} \hat{\psi}(\omega) = 0$: the norm of ψ is by definition finite

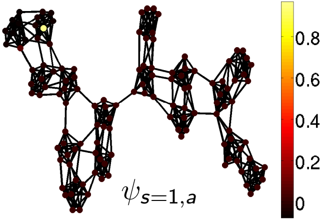
Classical wavelets $\xrightarrow{\text{by analogy}}$ Graph wavelets (Hammond '11)

	Classical (continuous) world	Graph world
Real domain variable	x	node a
Fourier domain variable	ω	eigenvalues λ_i
Filter kernel function	$\hat{\psi}(\omega)$	$g(\lambda_i) \Leftrightarrow \hat{\mathbf{G}}$
Filter bank	$\hat{\psi}(s\omega)$	$g(s\lambda_i) \Leftrightarrow \hat{\mathbf{G}}_\mathbf{s}$
Fourier modes	$\exp^{-i\omega x}$	eigenvectors χ_i
Fourier transform of f	$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) \exp^{-i\omega x}\, dx$	$\hat{f} = \boldsymbol{\chi}^\top f$

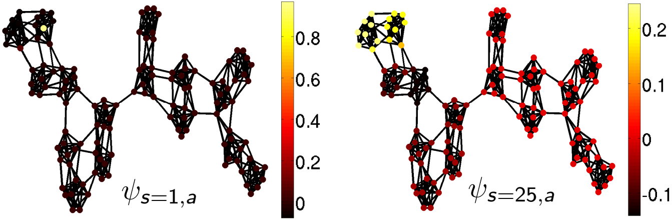
The wavelet at scale s centered around node a is given by:

$$\psi_{s,a}(x) = \int_{-\infty}^{\infty} \hat{\delta}_a(\omega)\hat{\psi}(s\omega) \exp^{i\omega x}\, d\omega \longrightarrow \boxed{\psi_{s,a} = \boldsymbol{\chi}\, \hat{\mathbf{G}}_\mathbf{s} \hat{\delta}_a = \boldsymbol{\chi}\, \hat{\mathbf{G}}_\mathbf{s}\, \boldsymbol{\chi}^\top \delta_a}$$

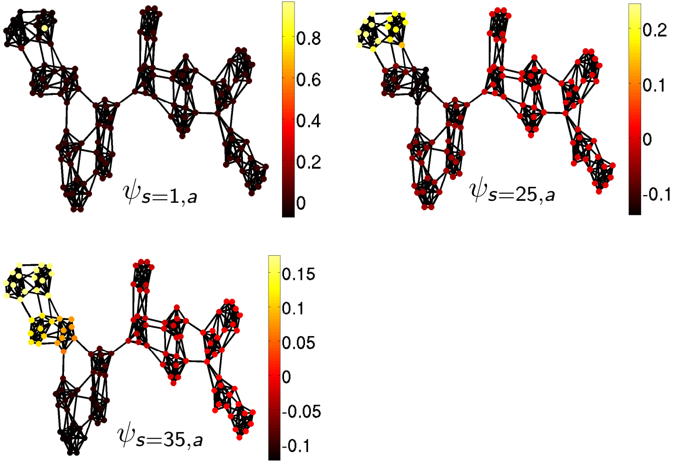
Examples of wavelets



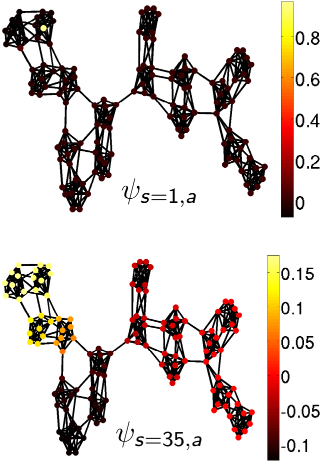
Examples of wavelets



Examples of wavelets



Examples of wavelets



The graph scaling functions

- Consider the following *lowpass filter kernel h*:

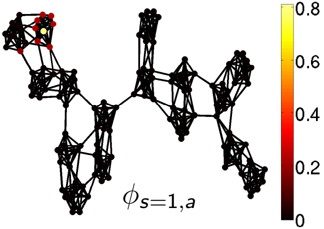
$$h(\omega) = \left(\int_{\omega}^{\infty} \frac{|g(\omega')|^2}{\omega'} d\omega' \right)^{1/2}$$

  Classically, if g corresponds to a wavelet filter kernel, this equation defines the associated scaling function filter kernel.

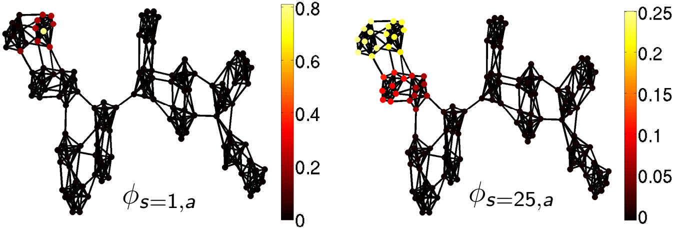
- With the same arguments as previously, we define the graph scaling function at scale s centered around a as:

$$\boxed{\phi_{s,a} = \boldsymbol{\chi}\, \hat{\mathbf{H}}_{\mathbf{s}} \hat{\delta}_a = \boldsymbol{\chi}\, \hat{\mathbf{H}}_{\mathbf{s}}\, \boldsymbol{\chi}^\top\, \delta_a}$$

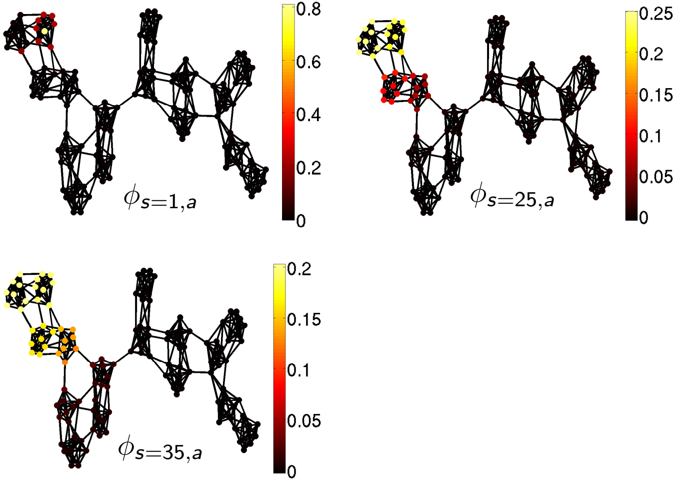
Examples of scaling functions



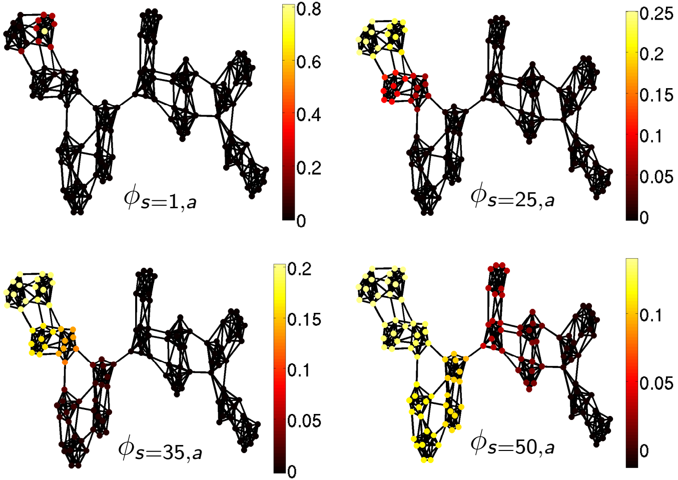
Examples of scaling functions



Examples of scaling functions



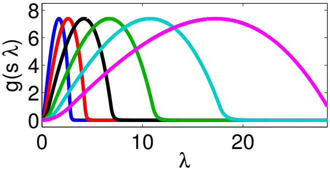
Examples of scaling functions

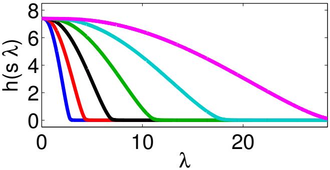


Example of filters

For each graph under study, we automatically find the good filter parameters for g by imposing:

- The coarsest scale needs to be focused on the first mode χ_1.
- All scales need at least to keep some information from χ_1.
- The finest scale needs to use the information from all modes.





Important note

In the following, we will not *actually* perform a Wavelet Transform of any signal: we will rather focus on the wavelets ψ_i and take advantage of the topological information encoded in them

Application to detection of communities

The three key points of clustering

Any clustering technique is based on the choice of:

1. feature vectors for each node
2. a distance to measure two given vectors' closeness
3. a clustering algorithm to separate nodes in clusters

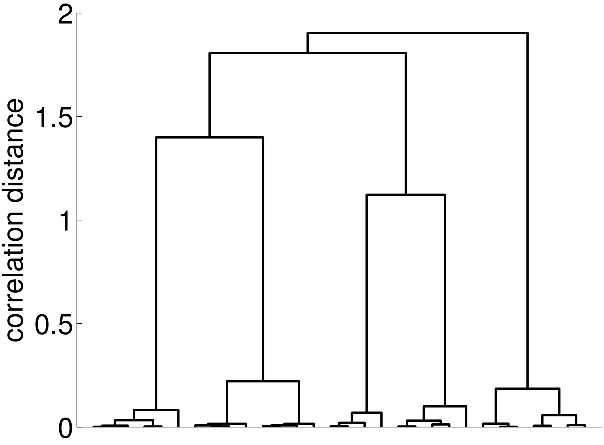
We choose to use:

1. wavelets (resp. scaling functions) as feature vectors
2. the correlation distance
3. the complete linkage clustering algorithm

Complete linkage clustering

- It is a bottom to top hierarchical algorithm: it starts with as many clusters as nodes and works its way up to fewer clusters (by linking subclusters together) until it reaches one global cluster.
- To compute the distance between two subclusters under examination : all possible pairs of nodes, taking one from each cluster, are considered. The *maximum* possible node-to-node distance is declared to be the cluster-to-cluster closeness.
- Outputs a dendrogram (from Greek dendron "tree" and gramma "drawing").

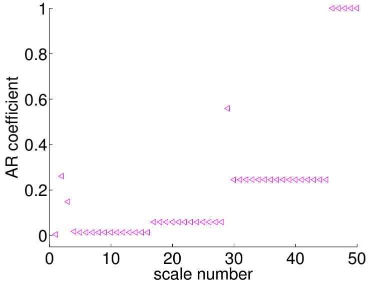
Example of a dendrogram **at a given scale s**



The big question: where should we cut the dendrogram?

With prior knowledge

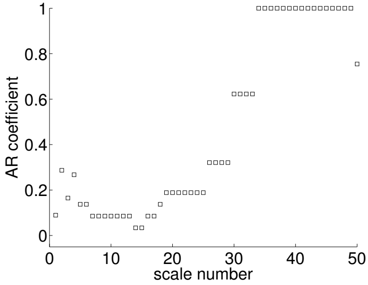
Let us cheat by using prior knowledge on the number of communities we are looking for.
If we cut each dendrogram in two clusters



Using wavelets

With prior knowledge

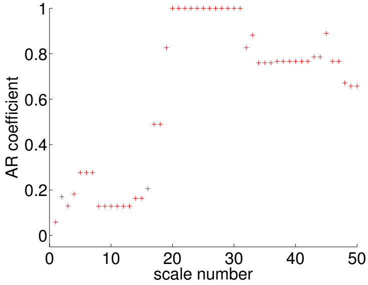
Let us cheat by using prior knowledge on the number of communities we are looking for.
If we cut each dendrogram in four clusters



Using wavelets

With prior knowledge

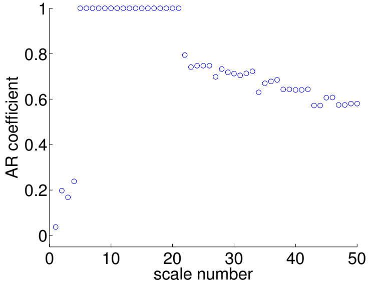
Let us cheat by using prior knowledge on the number of communities we are looking for.
If we cut each dendrogram in eight clusters



Using wavelets

With prior knowledge

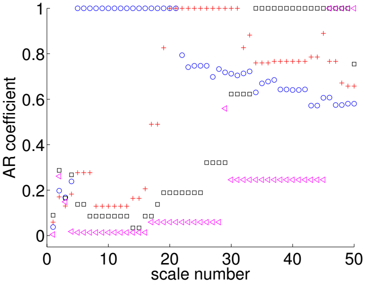
Let us cheat by using prior knowledge on the number of communities we are looking for.
If we cut each dendrogram in sixteen clusters



Using wavelets

With prior knowledge

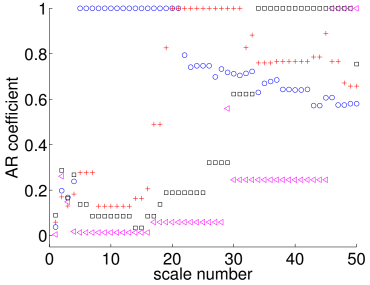
Let us cheat by using prior knowledge on the number of communities we are looking for.
The four levels of communities.



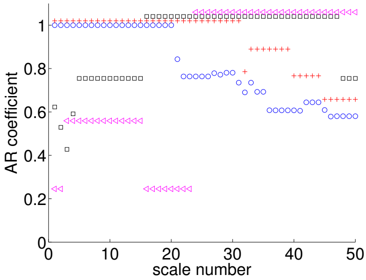
Using wavelets

With prior knowledge

Let us cheat by using prior knowledge on the number of communities we are looking for.
The four levels of communities.



Using wavelets

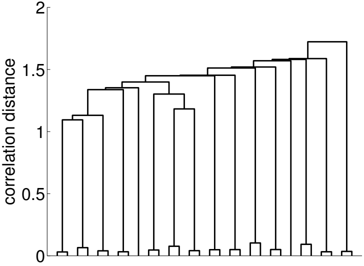


Using scaling functions

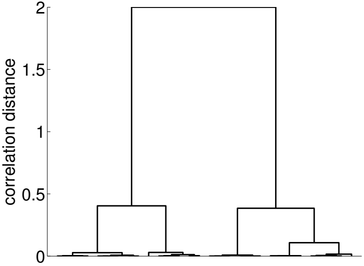
Without prior knowledge

We cut the dendrogram at its maximal gap.

At small scale:



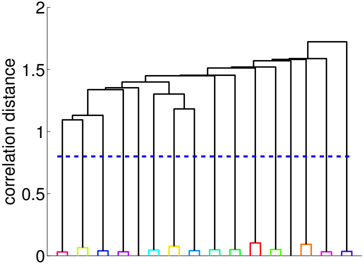
At large scale:



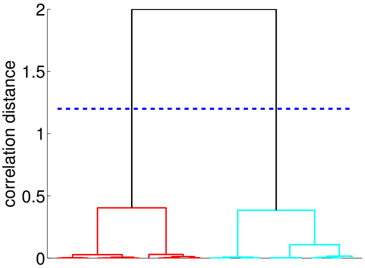
Without prior knowledge

We cut the dendrogram at its maximal gap.

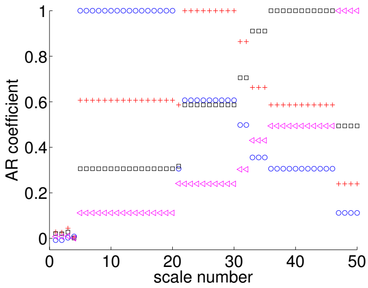
At small scale:



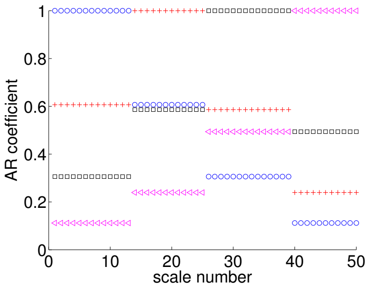
At large scale:



Without prior knowledge

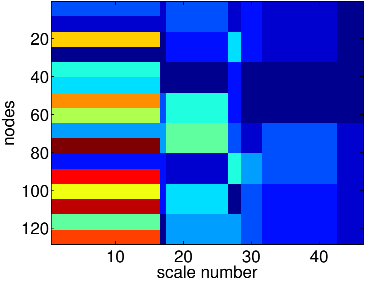


Using wavelets

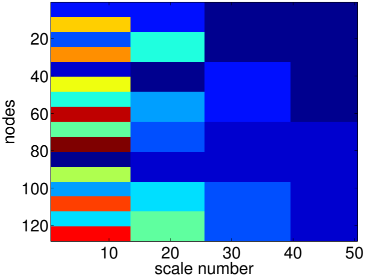


Using scaling functions

Without prior knowledge

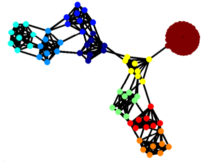


Using wavelets

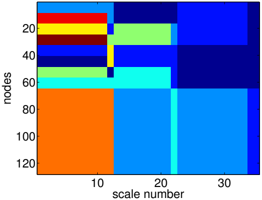


Using scaling functions

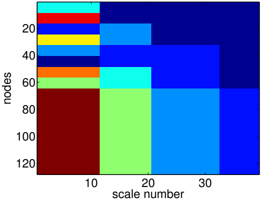
Another toy graph



Another toy graph



Using wavelets



Using scaling functions

The filtered modularity

We define the filtered adjacency matrices at scale s:

- recall that $A = D^{\frac{1}{2}}\chi(I - \Lambda)\chi^\top D^{\frac{1}{2}}$
- $A_s^g = A + D^{\frac{1}{2}}\chi\hat{G}_s\chi^\top D^{-\frac{1}{2}}A$
- $A_s^h = D^{\frac{1}{2}}\chi\hat{H}_s\chi^\top D^{-\frac{1}{2}}A$

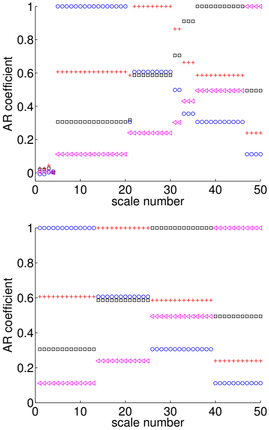
The classical modularity reads: $B(A) = \frac{1}{2m}(A + \frac{dd^\top}{2m})$
where d is the strength vector and $2m = \sum d(i)$

We define the filtered modularity matrices at scale s:

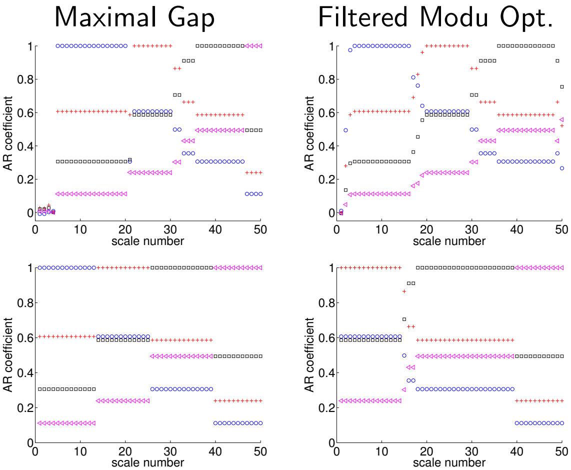
$$B_s^g = B(A_s^g) \text{ and } B_s^h = B(A_s^h)$$

Maximize filtered modularity

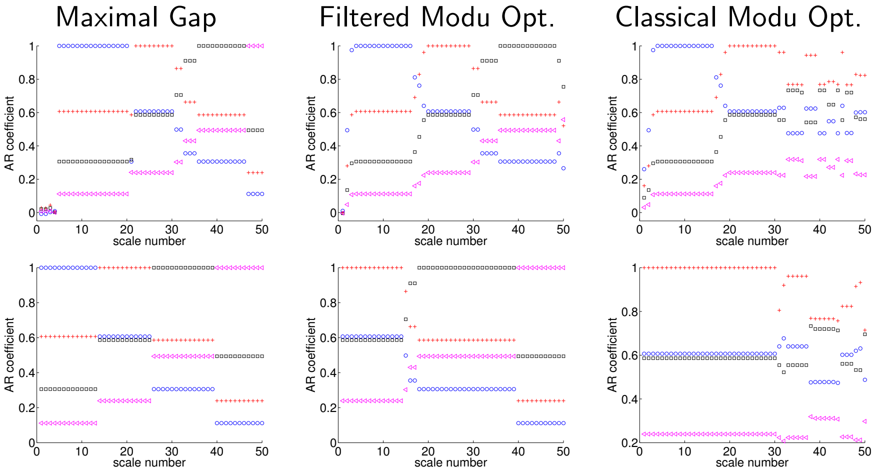
Maximal Gap



Maximize filtered modularity



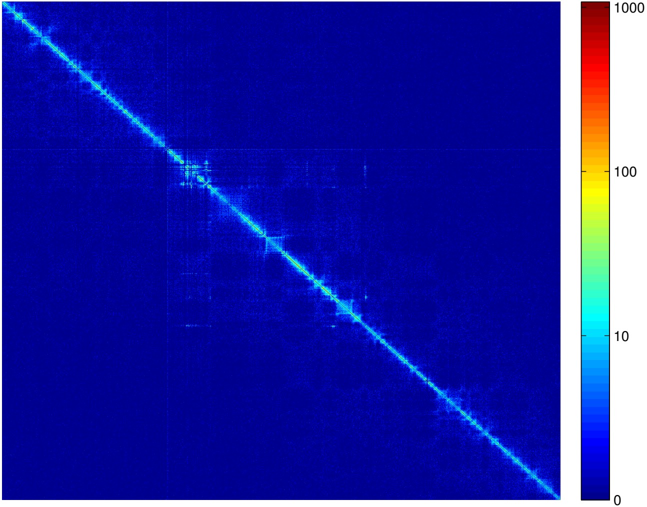
Maximize filtered modularity



Two real-world graphs

Intra-chromosomic interaction data

Collaboration with R. Boulos, B. Audit (ENS Lyon)



Evolution of the correlation matrix of the wavelets with respect to scale

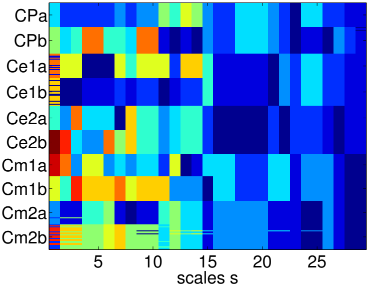
Collaboration with R. Boulos, B. Audit (ENS Lyon)

The dynamic social network of a primary school

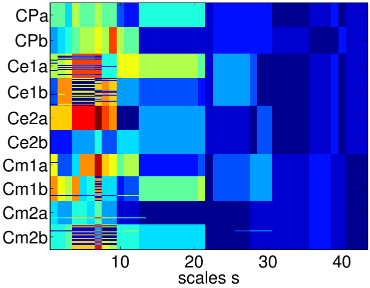
Collaboration with A. Barrat (CPT Marseille)

Multi-scale Communities in Primary School

Collaboration with A. Barrat (CPT Marseille)



Using wavelets



Using scaling functions

Conclusion

- Wavelet $\psi_{s,a}$ gives an "egocentered view" of the network seen from node a at scale s
- Correlation between these different views gives us a distance between nodes at scale s
- This enables multi-scale clustering of nodes in communities

I did not mention:

- the design of the filters
- the scale boundaries of the parameter "s"
- how we choose the relevant scales (we use a notion of stability of each partition)

Thank you for your attention!

The Adjusted Rand Index

Let:

- $\mathcal{C}$ and $\mathcal{C}'$ be two partitions we want to compare.
- a be the # of pairs of nodes that are in the same community in $\mathcal{C}$ *and* in the same community in $\mathcal{C}'$
- b be the # of pairs of nodes that are in different communities in $\mathcal{C}$ *and* in different communities in $\mathcal{C}'$
- c be the # of pairs of nodes that are in the same community in $\mathcal{C}$ *and* in different communities in $\mathcal{C}'$
- d be the # of pairs of nodes that are in different communities in $\mathcal{C}$ *and* in the same community in $\mathcal{C}'$

$a + b$ is the number of "agreements" between $\mathcal{C}$ and $\mathcal{C}'$.
$c + d$ is the number of "disagreements" between $\mathcal{C}$ and $\mathcal{C}'$.

The Adjusted Rand Index

The Rand index, R, is:

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}$$

The Adjusted Rand index AR is the corrected-for-chance version of the Rand index:

$$AR = \frac{R - ExpectedIndex}{MaxIndex - ExpectedIndex}$$

The filtered modularity

$$A_s^g = A + D^{\frac{1}{2}} \chi \hat{G}_s \chi^\top D^{-\frac{1}{2}} A$$

Consider d the vector of strengths of A and $2m$ the sum of the strengths. The classical modularity reads:

$$B = \frac{A}{2m} - \frac{dd^\top}{(2m)^2}$$

Consider d' the vector of strengths of A_s^g and $2m'$ the sum of the strengths. We can show that:

$$\frac{dd^\top}{(2m)^2} = \frac{d'd'^\top}{(2m')^2}$$

Moreover, if $g_s(1) = 0$ (which is the case), the filtered modularity reads:

$$B_s^g = \frac{A + D^{\frac{1}{2}} \chi \hat{G}_s \chi^\top D^{-\frac{1}{2}} A}{2m} - \frac{dd^\top}{(2m)^2}$$

The filtered modularity

$$B_s^g = \frac{A + D^{\frac{1}{2}}\chi\hat{G}_s\chi^\top D^{-\frac{1}{2}}A}{2m} - \frac{dd^\top}{(2m)^2}$$

Recall that modularity compares the actual normalised weight $\frac{A_{ij}}{2m}$ to the expected weight if the graph was a random Chung-Lu graph: $\frac{d_i d_j}{(2m)^2}$.

The filtered modularity does not change the expected weight but rather changes the actual normalised weigth: it adds or retrieve value to $\frac{A_{ij}}{2m}$. At small scale, it will increase the weights important for small scale structures and decrease the weights important for superstructures.

The filtered modularity

It can be written:

$$B_s^g = \frac{1}{2m}\sum_{i=2}^{N}(1+g_s(i))(1-\lambda_i)D^{\frac{1}{2}}\chi_i\chi_i^\top D^{\frac{1}{2}}$$

To compare to Delvenne's filtered modularity:

$$B_t = \frac{1}{2m}\sum_{i=2}^{N}(1-\lambda_i)^t D^{\frac{1}{2}}\chi_i\chi_i^\top D^{\frac{1}{2}}$$

And Arenas' version: (here for regular networks)

$$B_\alpha = \frac{1}{2m}\sum_{i=2}^{N}(1-\frac{\lambda_i}{\alpha})D^{\frac{1}{2}}\chi_i\chi_i^\top D^{\frac{1}{2}}$$