On a strategy for Spectral clustering with parallel computation

Sandrine Mouysset
Joseph Noailles
Daniel Ruiz
Ronan Guivarch

APO Team
IRIT-ENSEEIHT, University of Toulouse

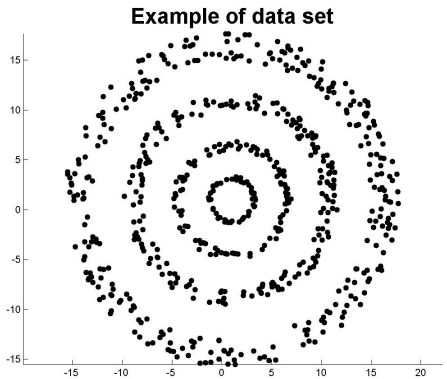
VECPAR'10 - 9th International Meeting High Performance Computing for Computational Science

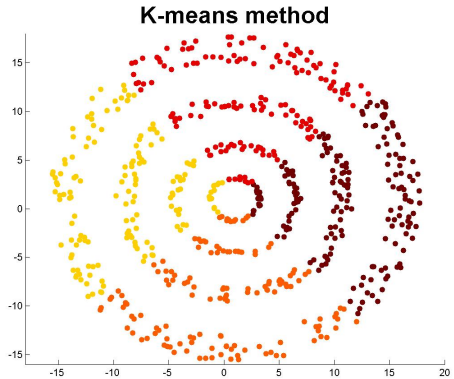
Clustering

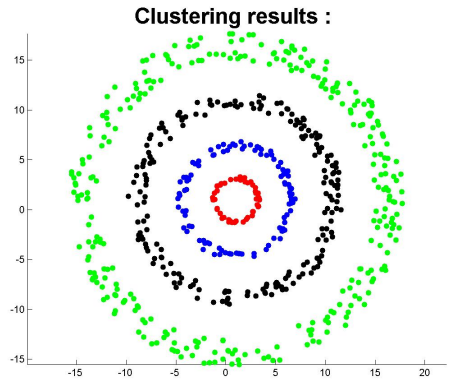
- Goal:

  Partition a $n \times p$ data set in K clusters to obtain larger within-cluster affinity and lower between-clusters affinity

- Some clustering methods based on:
  - geometrical properties: K-means...
  - spectral properties: Spectral clustering...







Spectral Clustering: introduction

Spectral Clustering

select dominant eigenvectors of a parametrized affinity matrix A in order to build a low-dimensional data space wherein data points are grouped into clusters

Main difficulties :

- How to (automatically) separate clusters one from the other?
  $\longrightarrow$ Look for some full-unsupervising process
- How to perform clustering on large datasets (image segmentation)?
  $\longrightarrow$ Parallelization using domain decomposition

Algorithm *Ng, Jordan and Weiss*

→ **Spectral Clustering : theoretical points and through a parallel implementation**

1. Form the Gaussian affinity matrix $A \in \mathbb{R}^{m \times m}$ defined by:

$$A_{ij} = \begin{cases} exp(-\|x_i - x_j\|^2 / 2\sigma^2) \text{ if } i \neq j, \\ 0 \text{ otherwise} \end{cases}$$

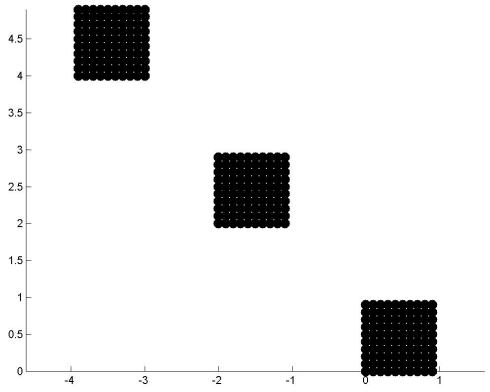
2. Construct the normalized matrix : $L = D^{-1}A$ with $D_{i,i} = \sum_{j=1}^{m} A_{ij}$

3. Construct the matrix $X = [X_1 X_2 .. X_k] \in \mathbb{R}^{m \times k}$ by stacking the k "largest" eigenvectors of L. (k to be defined)

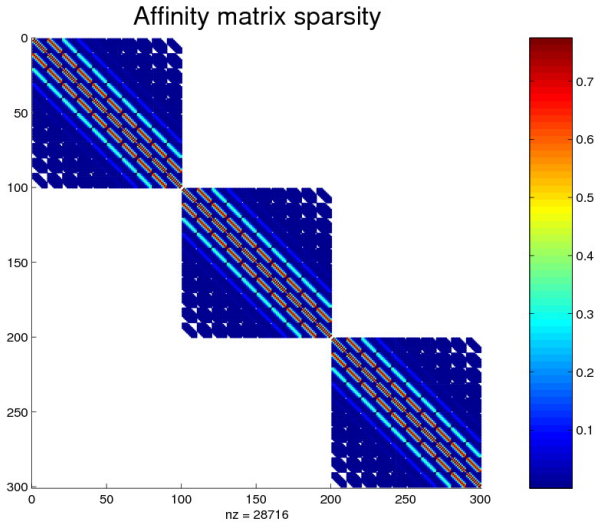
4. Form the matrix Y by normalizing each of the X's rows, and treat each row of Y as a point in $\mathbb{R}^k$ and cluster them in k clusters via *K-means* method

5. Assign the original point x_i to cluster j if and only if row i of the matrix Y was assigned to cluster j.

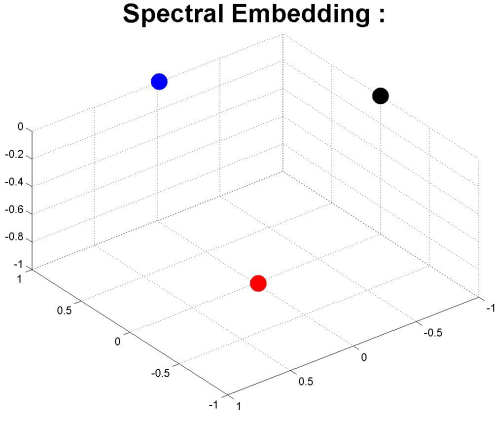
Spectral Clustering: example (ideal case)



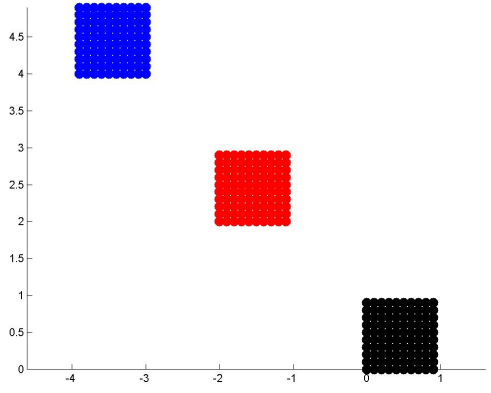
(a) Data set (n=300)



(b) Near block-diagonal affinity matrix (step 1)



(c) Y's rows (step 4)



(d) 3 well-separated clusters (step 5)

Interpretation of Gaussian affinity matrix as discretization of Heat kernel

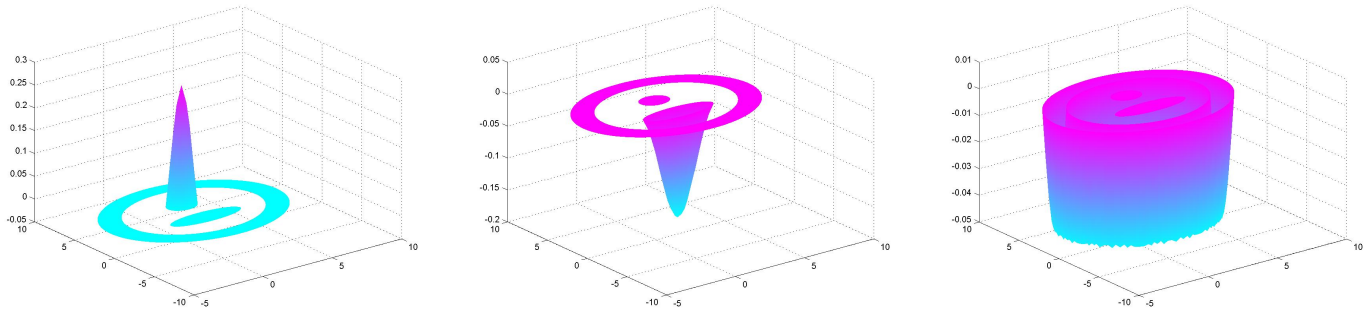
Affinity between two data points x_i and x_j

$$A_{ij} = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$$

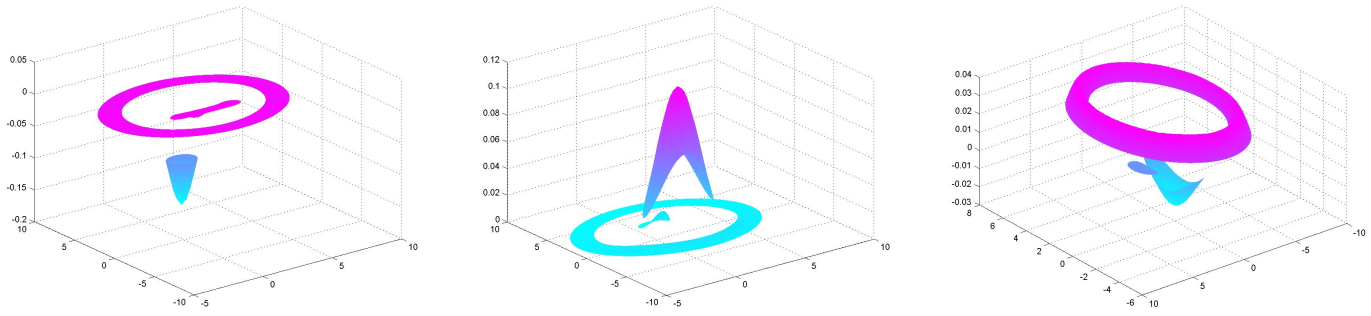
Heat kernel in free space

$$K_t(x-y) = (4\pi t)^{-\frac{p}{2}} \exp\left(-\frac{\|x-y\|^2}{4t}\right)$$

Eigenfunctions for Heat equation with Dirichlet boundary conditions:



Eigenvectors of the affinity matrix A:



Interpretation of Gaussian affinity matrix as discretization of Heat kernel

We can prove that:

1. eigenfunctions for bounded and free space Heat equation are asymptotically close when t goes to 0,
2. difference between eigenvectors of A and discretized eigenfunctions of K_t is in $\mathcal{O}$ of the distance between points inside the same cluster.

Conclusion: Spectral Clustering as a "connected components" method

Applying Spectral Clustering into subdomains resumes in restricting the support of L^2 particular eigenfunctions.

Two main problems arise:

- Choice of the Gaussian affinity parameter σ
- Estimating the number of clusters k

1. **Choice of the Gaussian affinity parameter σ**

Given a data set of points $S = \{x_i, 1 \leq i \leq n\}$, every element of S is included in a p-dimensional box of edge $D_{max} = \max_{1 \leq i,j \leq n} \|x_i - x_j\|$.

Let δ the reference distance defined by:

$$\delta = \frac{D_{max}}{n^{1/p}}$$

where n is the number of data points and p the data dimension.

Global heuristic parameter

Estimation of parameter σ: $\sigma^2 = \frac{\delta^2}{2}$.

(homogeneity of σ with respect to δ obtained by previous theoretical analysis)

$\rightarrow$ Automatic estimation of affinity parameter is performed.

2. **Estimating the number of clusters** k

In general cases, A's off-diagonal blocks are non-zero so, with $k = 3$:

$$\hat{L} = \begin{bmatrix} L^{(11)} & L^{(12)} & L^{(13)} \\ L^{(21)} & L^{(22)} & L^{(23)} \\ L^{(31)} & L^{(32)} & L^{(33)} \end{bmatrix}$$

Evaluate the ratio between off-diagonal-blocks in Frobenius norm and diagonal-blocks one, for $i \neq j$ and $i, j \in 1, .., k$:

$$r_{ij} = \frac{\|L^{(ij)}\|_F}{\|L^{(ii)}\|_F}$$

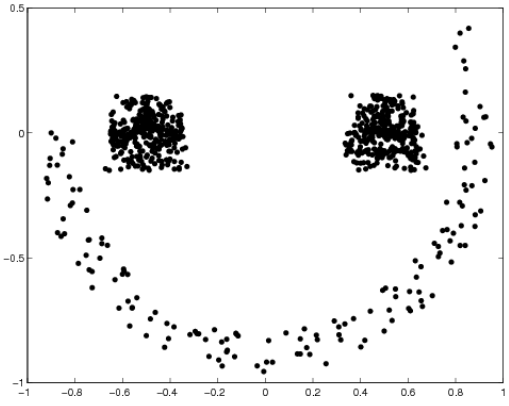
Criterion for determining k

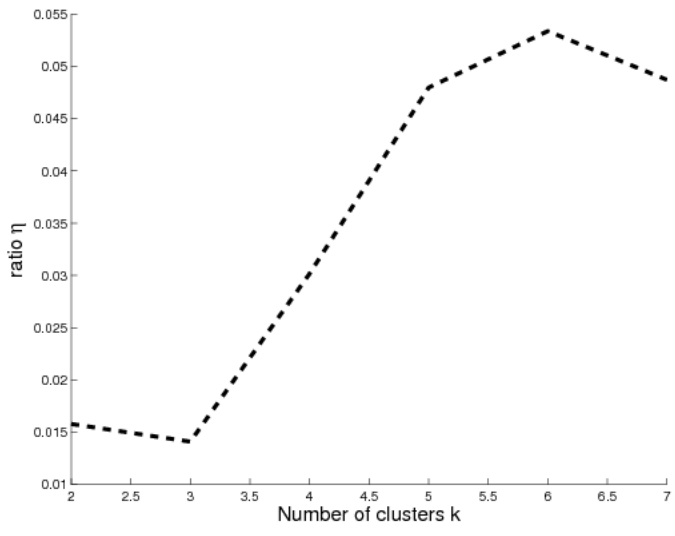
$$k = arg \min_{k'} \frac{2}{k'(k'-1)} \sum_{\substack{i=1 \\ j=i+1}}^{k'} r_{ij}. \tag{1}$$

$\rightarrow$ Automatic determination for the number of clusters k

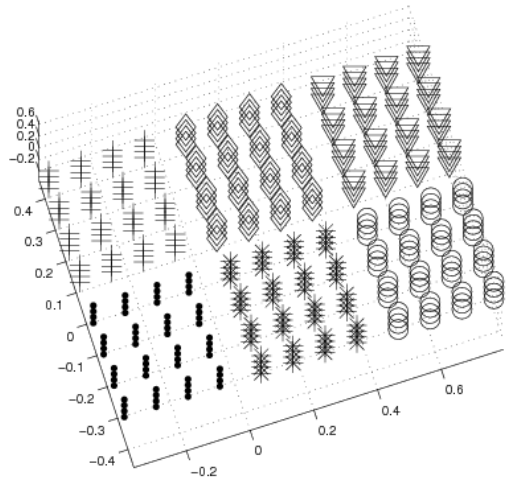
Estimation of the number of clusters: examples

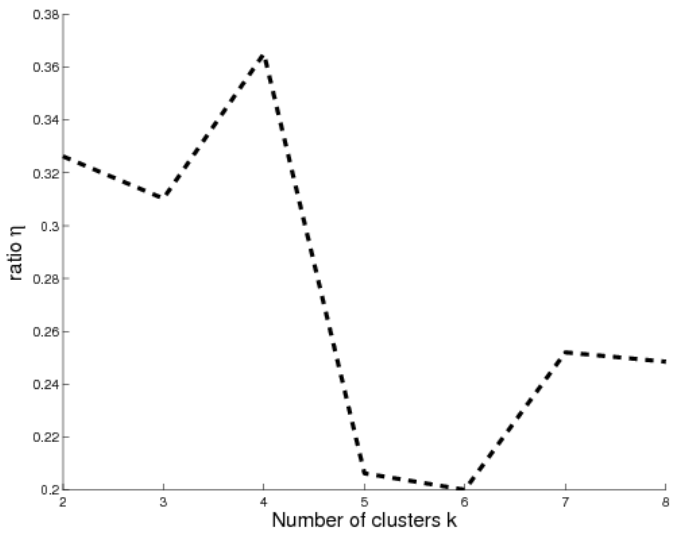
Example smiley:





Example with multiple close clusters:





→ Minimum for ratio function reached for optimal k

Domain decomposition strategy: implementation

→ Parallel strategy 1: disjointed subdomains with interface coupling

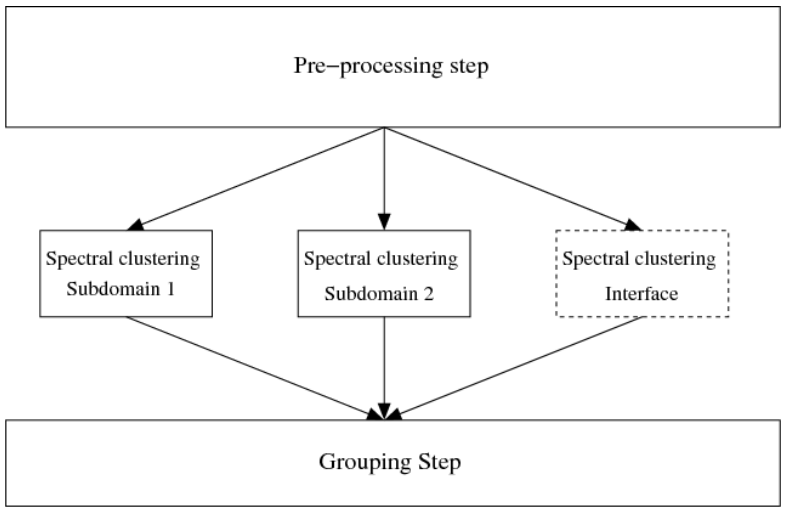


Figure: Principle of parallel Spectral clustering for $q = 2$

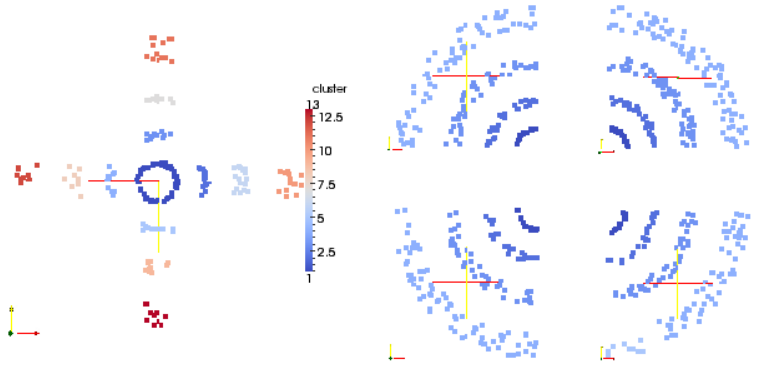


Figure: Target example: interface and subdomains

→ Total number of processes = q+1.

Domain decomposition strategy: parallel experiments

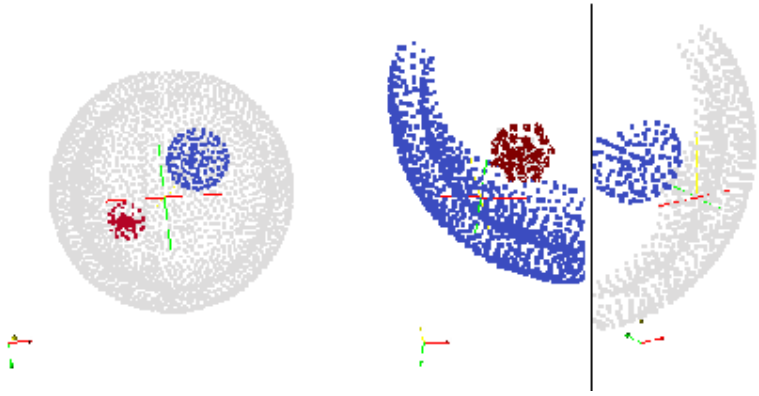


Figure: Geometrical example and zooms: $n = 4361$

n	Number of processors	Number of data in the interface	Total Time (sec)	% of Total Time for spectral clustering
9700	1	-	2930.3	99.9
	5	3601	214.47	99.4
	9	4868	354.77	99.3
	13	5738	628.81	99.6
15247	1	-	> 3h	-
	3	5532	695.41	99.6
	9	7531	1289.43	99.6
	13	8950	2394.01	99.8

→ Test on Hyperion supercomputer with 352 nodes (bi-Intel "Nehalem" EP quad-core), 4.5GB per core, 33TFlops

Domain decomposition strategy: parallel experiments

Observations

- *Main part of algorithm is dedicated to spectral clustering on subdomains;*
- *Speed-up is larger than the ratio between total number of points to the maximum data on one subdomain;*
- *Spectral clustering on subdomains is faster than considering the whole data set.*

Limitations

- Computation of a particular Gaussian parameter for the interface;
- Interface becomes the most time consuming computational task in case of larger number of subdomains.

→ Parallel strategy 2: decomposition with overlaps

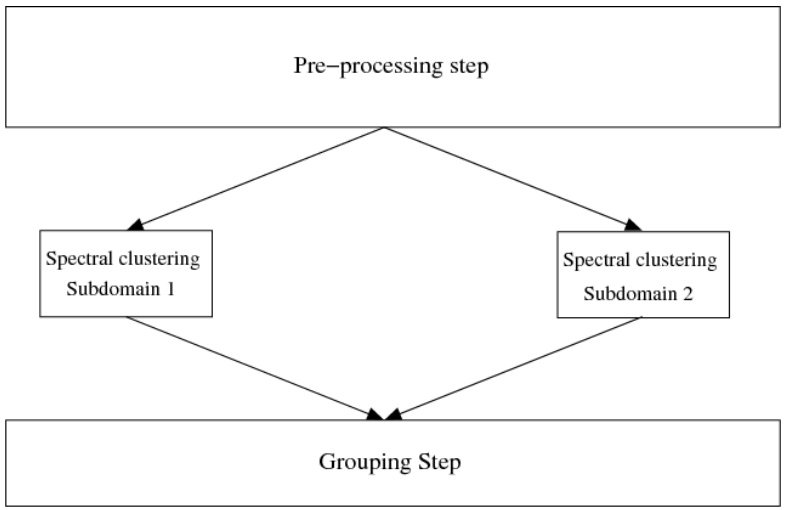


Figure: Principle of parallel Spectral clustering for $q = 2$

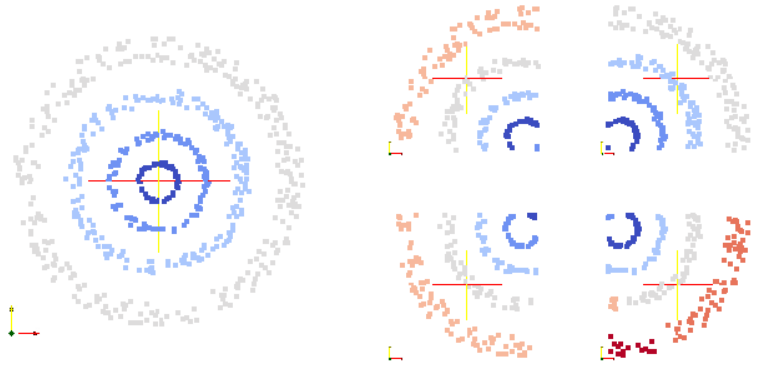


Figure: Target example: intersection and subdomains

→ Total number of processes = q.

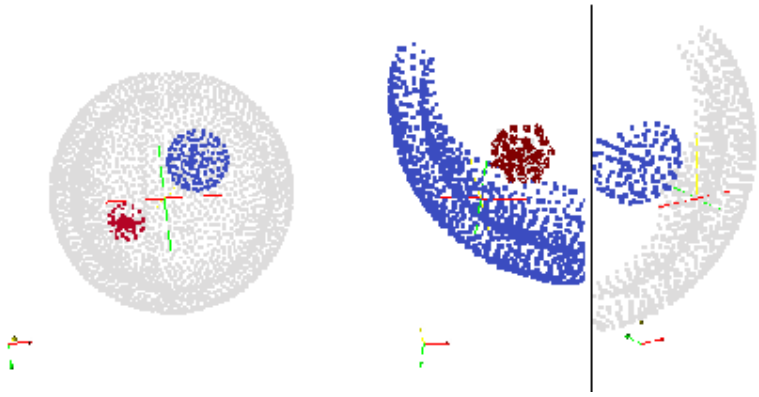


Figure: Geometrical example on Hyperion: $n = 4361$

n	Number of processors	Maximum of data by processor	Total Time (sec)	% of Total Time for spectral clustering
9700	1	-	2930.3	99.9
	4	3712	304.71	99.6
	8	2265	70.35	98.1
	12	2283	67.27	96.6
15247	1	-	$>$ 3h	-
	4	5760	1034.09	99.8
	8	3531	247.16	98.9
	12	3517	231.71	97.9

Domain decomposition strategy: parallel experiments

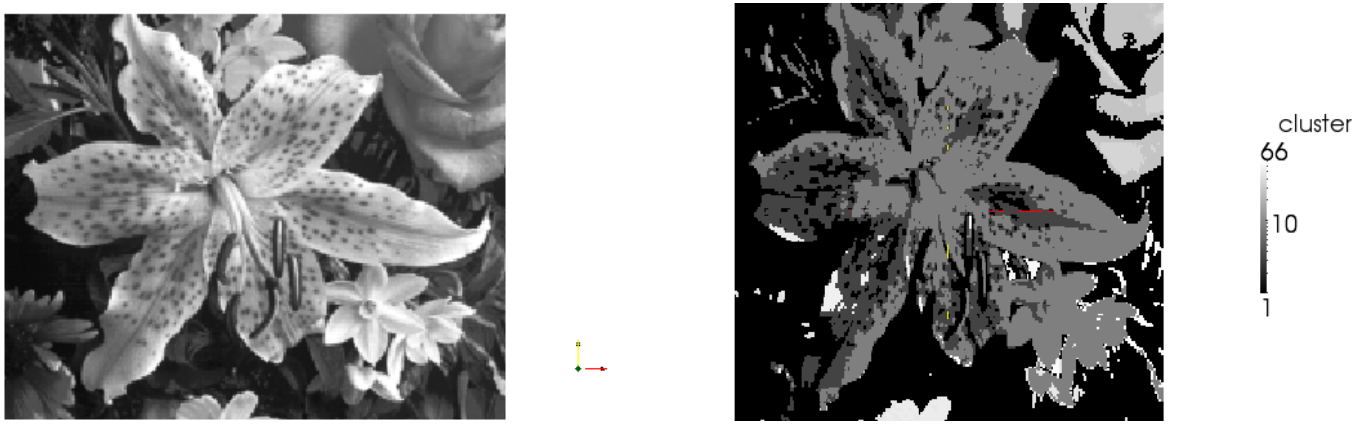
Summary:

- Speed-up » (total number of data) / (the maximum number of data on a subdomain);
- Overlapping strategy faster than interface one for equivalent number of processors;
- Computational time decreases with the maximal number of data points on a processor.

$\longrightarrow$ Overlapping strategy more relevant for image segmentation applications

→ Application on image segmentation

Application 1: Image segmentation including both geometrical and brightness in affinity definition



(a) Original data set (b) Clustering result

Figure: Example of image segmentation tested on Hyperion

Total time: 675.67 seconds for $n = 42780$ points

Application 2: larger dataset with more constrasts

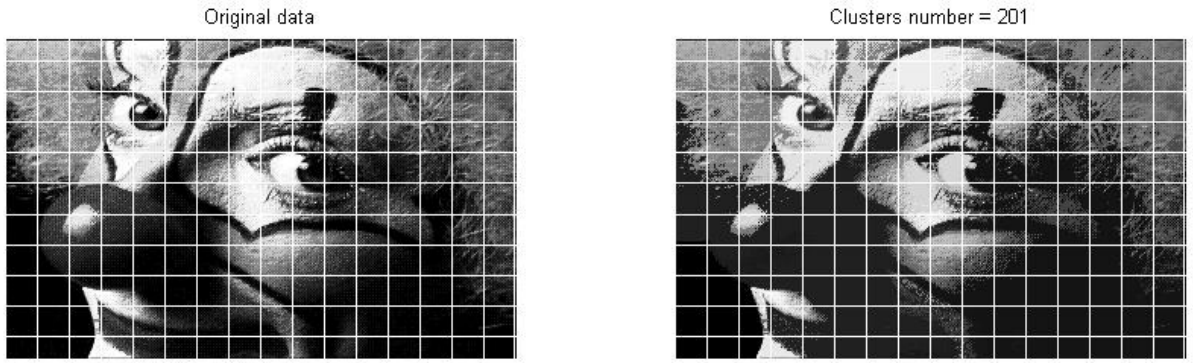


Figure: Example: original data (left) and clustering result (right)

Total time: 5145.05 seconds for $n = 64000$ points

Conclusion and further investigations

Conclusion

- *Parallel strategies of Spectral clustering proposed and tested on geometrical and imaging examples;*
- *Method fully unsupervised.*

Perspectives

- Study of the robustness : sparsification techniques, techniques for distributing uniformly the data per processor;
- Image segmentation : study descriptive parameters in affinity definition (brightness, color, geometrical information...);
- Genomic perspective : how to divide into subdomains for time-dependent applications?

Thank you for your attention.