Rocco Langone, Raghvendra Mall, Carlos Alzate, Johan A. K. Suykens

Kernel Spectral Clustering and applications

Chapter Contribution to the book: Unsupervised Learning Algorithms

March 17, 2015

Springer

Contents

1 Kernel Spectral Clustering and applications 1

1.1 Introduction 1

1.2 Notation 3

1.3 Kernel Spectral Clustering (KSC) 3

1.3.1 Mathematical formulation 3

1.3.2 Soft Kernel Spectral Clustering (SKSC) 6

1.3.3 Hierarchical Clustering 9

1.3.4 Sparse Clustering Models 10

1.4 Applications 15

1.4.1 Image Segmentation 15

1.4.2 Scientific Journal Clustering 16

1.4.3 Power Load Clustering 18

1.4.4 Big data 19

1.5 Conclusions 21

References 23

Chapter 1
Kernel Spectral Clustering and applications

Abstract In this chapter we review the main literature related to kernel spectral clustering (KSC), an approach to clustering cast within a kernel-based optimization setting. KSC represents a least-squares support vector machine based formulation of spectral clustering described by a weighted kernel PCA objective. Just as in the classifier case, the binary clustering model is expressed by a hyperplane in a high dimensional space induced by a kernel. In addition, the multi-way clustering can be obtained by combining a set of binary decision functions via an Error Correcting Output Codes (ECOC) encoding scheme. Because of its model-based nature, the KSC method encompasses three main steps: training, validation, testing. In the validation stage model selection is performed to obtain tuning parameters, like the number of clusters present in the data. This is a major advantage compared to classical spectral clustering where the determination of the clustering parameters is unclear and relies on heuristics. Once a KSC model is trained on a small subset of the entire data, it is able to generalize well to unseen test points. Beyond the basic formulation, sparse KSC algorithms based on the Incomplete Cholesky Decomposition (ICD) and L_0, $L_1, L_0 + L_1$, Group Lasso regularization are reviewed. In that respect, we show how it is possible to handle large scale data. Also, two possible ways to perform hierarchical clustering and a soft clustering method are presented. Finally, real-world applications such as image segmentation, power load time-series clustering, document clustering and big data learning are considered.

1.1 Introduction

Spectral clustering (SC) represents the most popular class of algorithms based on graph theory (Chung 1997). It makes use of the Laplacian's spectrum to partition a graph into weakly connected sub-graphs. Moreover, if the graph is constructed based on any kind of data (vector, images etc.), data clustering can be performed[1]. SC began to be popularized when Shi and Malik introduced the Normalized Cut criterion to handle image segmentation (Shi & Malik 2000). Afterwards, Ng and Jordan (Ng et al. 2002) in a theoretical work based on matrix perturbation theory have shown conditions under which a good performance of the algorithm is expected. Finally, in the tutorial by Von Luxburg the main literature related to SC has been exhaustively summarized (von Luxburg 2007). Although very successful in a number of applications, SC has some limitations. For instance, it cannot handle big data without using approximation methods like the Nyström algorithm (Fowlkes et al. 2004, Williams & Seeger 2001), the power iteration method (Lin & Cohen 2010), or linear algebra based methods (Ning et al. 2010, Dhanjal et al. 2013, Frederix & Van Barel 2013). Furthermore, the generalization to out-of-sample data is only approximate.

These issues have been recently tackled by means of a spectral clustering algorithm formulated as weighted kernel PCA (Alzate & Suykens 2010). The technique, named kernel spectral clustering (KSC), is based on solving a constrained optimization problem in a primal-dual setting. In other words, KSC is a Least Squares Support Vector Machine (LS-SVM (Suykens et al. 2002)) model used for clustering instead of classification[2]. By casting SC in a learning framework, KSC allows to rigorously select tuning parameters such as the natural number of clusters which are present in the data. Also, an accurate prediction of the cluster memberships for unseen points can be easily done by projecting test data in the embedding eigenspace learned during training. Furthermore, the algorithm can be tailored to a given application by using the most appropriate kernel function. Beyond that, by using sparse formulations and a fixed-size (Suykens et al. 2002, De Brabanter et al. 2010) approach, it is possible to readily handle big data. Finally, by means of adequate adaptations of the core algorithm, hierarchical clustering and a soft clustering approach have been proposed.

All these topics will be detailed in the next Sections. Precisely, after presenting the basic KSC method, the soft KSC algorithm will be summarized. Next, two possible ways to accomplish hierarchical clustering will be explained. Afterwards, some sparse formulations based on the Incomplete Cholesky Decomposition (ICD) and L_0, $L_1, L_0 + L_1$, Group Lasso regularization will be described. Lastly, various interesting applications in different domains such as computer vision, power-load consumer profiling, information retrieval and big data clustering will be illustrated. All these examples assume a static setting. Concerning other applications in a dynamic scenario the interested reader can refer to (Langone, Alzate, De Ketelaere & Suykens 2013, Langone et al. 2015) for fault detection, to (Langone, Agudelo, De Moor & Suykens 2014) for incremental time-series clustering, to (Langone, Alzate & Suykens 2013, Langone & Suykens 2013, Langone,

[1] In this case the given data points represent the node of the graph and their similarity the corresponding edges.

[2] This is a considerable novelty, since SVMs are typically known as classifiers or function approximation models rather than clustering techniques.

Mall & Suykens 2014) in case of community detection in evolving networks and (Peluffo et al. 2013) in relation to human motion tracking.

1.2 Notation

x^T	Transpose of the vector x
A^T	Transpose of the matrix A
I_N	$N \times N$ Identity matrix
1_N	$N \times 1$ Vector of ones
$\mathscr{D}_{\mathrm{tr}} = \{x_i\}_{i=1}^{N_{\mathrm{tr}}}$	Training sample of N_{tr} data points
$\varphi(\cdot)$	Feature map
$\mathscr{F}$	Feature space of dimension d_h
$\{\mathscr{A}_p\}_{p=1}^{k}$	Partitioning composed of k clusters
$\mathscr{G} = (\mathscr{V}, \mathscr{E})$	Set of N vertices $\mathscr{V} = \{v_i\}_{i=1}^{N}$ and m edges $\mathscr{E}$ of a graph
$\lvert \cdot \rvert$	Cardinality of a set

1.3 Kernel Spectral Clustering (KSC)

1.3.1 Mathematical formulation

1.3.1.1 Training problem

The KSC formulation for k clusters is stated as a combination of $k-1$ binary problems (Alzate & Suykens 2010). In particular, given a set of training data $\mathscr{D}_{\mathrm{tr}} = \{x_i\}_{i=1}^{N_{\mathrm{tr}}}$, the primal problem is:

$$\begin{aligned} \min_{w^{(l)}, e^{(l)}, b_l} \quad & \frac{1}{2}\sum_{l=1}^{k-1} w^{(l)^T} w^{(l)} - \frac{1}{2}\sum_{l=1}^{k-1} \gamma_l e^{(l)^T} V e^{(l)} \\ \text{subject to} \quad & e^{(l)} = \Phi w^{(l)} + b_l 1_{N_{\mathrm{tr}}}, l = 1, \ldots, k-1. \end{aligned} \tag{1.1}$$

The $e^{(l)} = [e_1^{(l)}, \ldots, e_i^{(l)}, \ldots, e_{N_{\mathrm{tr}}}^{(l)}]^T$ are the projections of the training data mapped in the feature space along the direction $w^{(l)}$. For a given point x_i, the model in the primal form is:

$$e_i^{(l)} = w^{(l)^T} \varphi(x_i) + b_l. \tag{1.2}$$

The primal problem (1.1) expresses the maximization of the weighted variances of the data given by $e^{(l)^T} V e^{(l)}$ and the contextual minimization of the squared norm of the vector $w^{(l)}$, $\forall l$. The regularization constants $\gamma_l \in \mathbb{R}^+$ mediate the model complexity expressed by $w^{(l)}$ with the correct representation of the training data. $V \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}$ is the weighting matrix and Φ is the $N_{\mathrm{tr}} \times d_h$ feature matrix

$\Phi = [\varphi(x_1)^T;\ldots;\varphi(x_{N_{\text{tr}}})^T]$, where $\varphi : \mathbb{R}^d \to \mathbb{R}^{d_h}$ denotes the mapping to a high-dimensional feature space, b_l are bias terms.

The dual problem corresponding to the primal formulation (1.1), by setting $V = D^{-1}$ becomes[3]:

$$D^{-1}M_D\Omega\alpha^{(l)} = \lambda_l\alpha^{(l)} \tag{1.3}$$

where Ω is the kernel matrix with ij-th entry $\Omega_{ij} = K(x_i,x_j) = \varphi(x_i)^T\varphi(x_j)$. $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ means the kernel function. The type of kernel function to utilize is application-dependent, as it is outlined in Table 1.1. The matrix D is the graph degree matrix which is diagonal with positive elements $D_{ii} = \sum_j \Omega_{ij}$, M_D is a centering matrix defined as $M_D = I_{N_{\text{tr}}} - \frac{1}{1^T_{N_{\text{tr}}}D^{-1}1_{N_{\text{tr}}}}1_{N_{\text{tr}}}1^T_{N_{\text{tr}}}D^{-1}$, the $\alpha^{(l)}$ are vectors of dual variables, $\lambda_l = \frac{N_{\text{tr}}}{\gamma_l}$, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the kernel function. The dual clustering model for the i-th point can be expressed as follows:

$$e_i^{(l)} = \sum_{j=1}^{N_{\text{tr}}} \alpha_j^{(l)} K(x_j,x_i) + b_l, j = 1,\ldots,N_{\text{tr}}, l = 1,\ldots,k-1. \tag{1.4}$$

The cluster prototypes can be obtained by binarizing the projections $e_i^{(l)}$ as $\text{sign}(e_i^{(l)})$. This step is straightforward because, thanks to presence of the bias term b_l, both the $e^{(l)}$ and the $\alpha^{(l)}$ variables get automatically centred around zero. The set of the most frequent binary indicators form a code-book $\mathscr{CB} = \{c_p\}_{p=1}^k$, where each code-word of length $k-1$ represents a cluster.

Application	Kernel Name	Mathematical Expression
Vector data	RBF	$K(x_i,x_j) = \exp(-\|\|x_i - x_j\|\|_2^2/\sigma^2)$
Images	RBF$_{\chi^2}$	$K(h^{(i)},h^{(j)}) = \exp(-\frac{\chi^2_{ij}}{\sigma^2_{\chi}})$
Text	Cosine	$K(x_i,x_j) = \frac{x_i^T x_j}{\|\|x_i\|\|\|\|x_j\|\|}$
Time-series	RBF$_{\text{cd}}$	$K(x_i,x_j) = \exp(-\|\|x_i - x_j\|\|_{\text{cd}}^2/\sigma^2_{\text{cd}})$

Table 1.1: **Types of kernel functions for different applications**. In this Table RBF means Radial Basis Function, σ denotes the bandwidth of the kernel. The symbol $h^{(i)}$ indicates a color histogram representing the $i-$th pixel of an image, and to compare two histograms $h^{(i)}$ and $h^{(j)}$ the χ^2 statistical test is used (Puzicha et al. 1997). Regarding time-series data, the symbol *cd* means correlation distance (Liao 2005), and $||x_i - x_j||_{\text{cd}} = \sqrt{\frac{1}{2}(1-R_{ij})}$, where R_{ij} can indicate the Pearson or Spearman's rank correlation coefficient between time-series x_i and x_j.

Interestingly, problem (1.3) has a close connection with SC based on a random walk Laplacian. In this respect, the kernel matrix can be considered as a weighted graph $\mathscr{G} = (\mathscr{V},\mathscr{E})$ with the nodes $v_i \in \mathscr{V}$ represented by the data points x_i. This graph has a corresponding random walk in which the probability of leaving a ver-

[3] By choosing $V = I$, problem (1.3) is identical to kernel PCA (Suykens et al. 2003, Schölkopf et al. 1998, Mika et al. 1999).

tex is distributed among the outgoing edges according to their weight: $p_{t+1} = Pp_t$, where $P = D^{-1}\Omega$ indicates the transition matrix with the ij-th entry denoting the probability of moving from node i to node j in one time-step. Moreover, the stationary distribution of the Markov Chain describes the scenario where the random walker stays mostly in the same cluster and seldom moves to the other clusters (Meila & Shi 2001*b*, Meila & Shi 2001*b*, Meila & Shi 2001*a*, Delvenne et al. 2010).

1.3.1.2 Generalization

Given the dual model parameters $\alpha^{(l)}$ and b_l, it is possible to assign a membership to unseen points by calculating their projections onto the eigenvectors computed in the training phase:

$$e^{(l)}_{\text{test}} = \Omega_{\text{test}}\alpha^{(l)} + b_l 1_{N_{\text{test}}} \tag{1.5}$$

where Ω_{test} is the $N_{\text{test}} \times N$ kernel matrix evaluated using the test points with entries $\Omega_{\text{test,ri}} = K(x^{\text{test}}_r, x_i)$, $r = 1, \ldots, N_{\text{test}}$, $i = 1, \ldots, N_{\text{tr}}$. The cluster indicator for a given test point can be obtained by using an Error Correcting Output Codes (ECOC) decoding procedure:

- the score variable is binarized
- the indicator is compared with the training code-book $\mathscr{C}\mathscr{B}$ (see previous Section), and the point is assigned to the nearest prototype in terms of Hamming distance.

The KSC method, comprising training and test stage, is summarized in algorithm 1, and the related Matlab package is freely available on the Web[4].

Algorithm 1: KSC algorithm (Alzate & Suykens 2010)

```
Data: Training set $\mathscr{D}_{\text{tr}} = \{x_i\}_{i=1}^{N_{\text{tr}}}$, test set $\mathscr{D}_{\text{test}} = \{x_m^{\text{test}}\}_{m=1}^{N_{\text{test}}}$ kernel function
      $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ positive definite and localized ($K(x_i, x_j) \to 0$ if $x_i$ and $x_j$ belong to
      different clusters), kernel parameters (if any), number of clusters $k$.
Result: Clusters $\{\mathscr{A}_1, \ldots, \mathscr{A}_k\}$, codebook $\mathscr{C}\mathscr{B} = \{c_p\}_{p=1}^{k}$ with $\{c_p\} \in \{-1, 1\}^{k-1}$.
compute the training eigenvectors $\alpha^{(l)}$, $l = 1, \ldots, k-1$, corresponding to the $k-1$ largest
  eigenvalues of problem (1.3)
let $A \in \mathbb{R}^{N_{\text{tr}} \times (k-1)}$ be the matrix containing the vectors $\alpha^{(1)}, \ldots, \alpha^{(k-1)}$ as columns
binarize $A$ and let the code-book $\mathscr{C}\mathscr{B} = \{c_p\}_{p=1}^{k}$ be composed by the $k$ encodings of
  $Q = \text{sign}(A)$ with the most occurrences
$\forall i$, $i = 1, \ldots, N_{\text{tr}}$, assign $x_i$ to $A_{p^*}$ where $p^* = \text{argmin}_p d_H(\text{sign}(\alpha_i), c_p)$ and $d_H(.,.)$ is the
  Hamming distance
binarize the test data projections $\text{sign}(e_m^{(l)})$, $m = 1, \ldots, N_{\text{test}}$, and let $\text{sign}(e_m) \in \{-1, 1\}^{k-1}$
  be the encoding vector of $x_m^{\text{test}}$
$\forall m$, assign $x_m^{\text{test}}$ to $A_{p^*}$, where $p^* = \text{argmin}_p d_H(\text{sign}(e_m), c_p)$.
```

[4] *http://www.esat.kuleuven.be/stadius/ADB/alzate/softwareKSClab.php*

1.3.1.3 Model selection

In order to select tuning parameters like the number of clusters k and eventually the kernel parameters, a model selection procedure based on grid search is adopted. First, a validation set $\mathscr{D}_{\text{val}} = \{x_i\}_{i=1}^{N_{\text{val}}}$ is sampled from the whole dataset. Then, a grid of possible values of the tuning parameters is constructed. Afterwards, a KSC model is trained for each combination of parameters and the chosen criterion is evaluated on the partitioning predicted for the validation data. Finally, the parameters yielding the maximum value of the criterion are selected. Depending on the kind of data, a variety of model selection criteria have been proposed:

- *Balanced Line Fit (BLF)*. It indicates the amount of collinearity between validation points belonging to the same cluster, in the space of the projections. It reaches its maximum value 1 in case of well separated clusters, represented as lines in the space of the $e_{\text{val}}^{(l)}$ (see for instance the bottom left side of Figure 1.1)
- *Balanced Angular Fit or BAF* (Mall et al. 2013*b*). For every cluster, the sum of the cosine similarity between the validation points and the cluster prototype, divided by the cardinality of that cluster, is computed. These similarity values are then summed up and divided by the total number of clusters.
- *Average Membership Strength abbr. AMS* (Langone, Mall & Suykens 2013). The mean membership per cluster denoting the mean degree of belonging of the validation points to the cluster is computed. These mean cluster memberships are then averaged over the number of clusters.
- *Modularity* (Newman 2006). This quality function is well suited for network data. In the model selection scheme, the Modularity of the validation sub-graph corresponding to a given partitioning is computed, and the parameters related to the highest Modularity are selected (Langone et al. 2011, Langone et al. 2012).
- *Fisher Criterion*. The classical Fisher criterion (Bishop 2006) used in classification has been adapted to select the number of clusters k and the kernel parameters in the KSC framework (Alzate & Suykens 2012). The criterion maximizes the distance between the means of the two clusters while minimizing the variance within each cluster, in the space of the projections $e_{\text{val}}^{(l)}$.

In Figure 1.1 an example of clustering obtained by KSC on a synthetic dataset is shown. The BLF model selection criterion has been used to tune the bandwidth of the RBF kernel and the number of clusters. It can be noticed how the results are quite accurate, despite the fact that the clustering boundaries are highly nonlinear.

1.3.2 Soft Kernel Spectral Clustering (SKSC)

Soft kernel spectral clustering (SKSC) makes use of algorithm 1 in order to compute a first hard partitioning of the training data. Next, soft cluster assignments are performed by computing the cosine distance between each point and some cluster prototypes in the space of the projections $e^{(l)}$. In particular, given the projec-

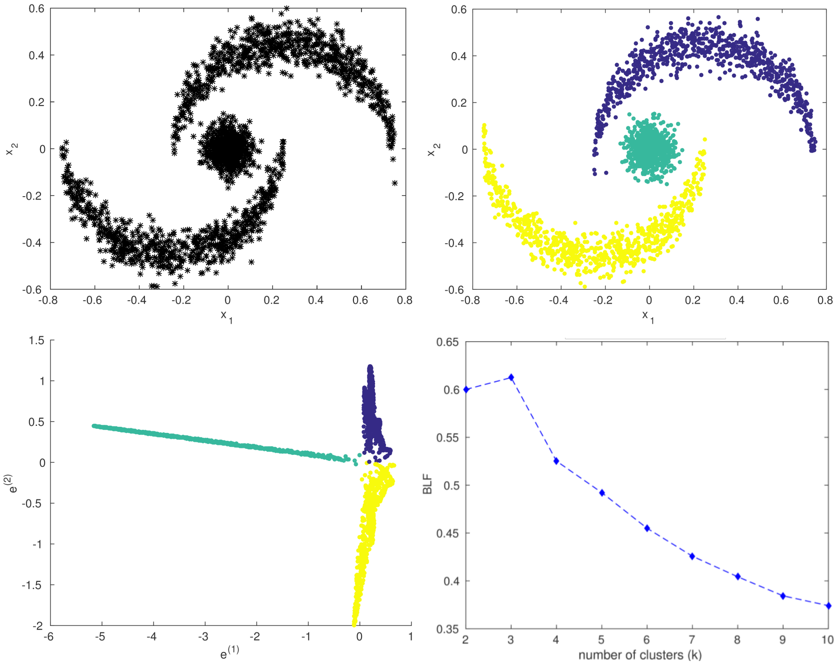


Fig. 1.1: **KSC partitioning on a toy dataset**. **(Top)** Original dataset consisting of 3 clusters (left) and obtained clustering results (right). **(Bottom)** Points represented in the space of the projections $[e^{(1)}, e^{(2)}]$ (left), for an optimal choice of k (and $\sigma^2 = 4.36 \cdot 10^{-3}$) suggested by the BLF criterion (right). We can notice how the points belonging to one cluster tend to lie on the same line. A perfect line structure is not attained due to a certain amount of overlap between the clusters.

tions for the training points $e_i = [e_i^{(1)}, \dots, e_i^{(k-1)}]$, $i = 1, \dots, N_{\text{tr}}$ and the corresponding hard assignments q_i^p we can calculate for each cluster the cluster prototypes $s_1, \dots, s_p, \dots, s_k$, $s_p \in \mathbb{R}^{k-1}$ as:

$$s_p = \frac{1}{n_p} \sum_{i=1}^{n_p} e_i \tag{1.6}$$

where n_p is the number of points assigned to cluster p during the initialization step by KSC. Then the cosine distance between the i-th point in the projections space and a prototype s_p is calculated by means of the following formula:

$$d_{ip}^{\cos} = 1 - e_i^T s_p / (||e_i||_2 ||s_p||_2). \tag{1.7}$$

The soft membership of point i to cluster q can be finally expressed as:

$$sm_i^{(q)} = \frac{\prod_{j \neq q} d_{ij}^{\cos}}{\sum_{p=1}^{k} \prod_{j \neq p} d_{ij}^{\cos}} \tag{1.8}$$

with $\sum_{p=1}^{k} sm_i^{(p)} = 1$. As pointed-out in (Ben-Israel & Iyigun 2008), this membership represents a subjective probability expressing the belief in the clustering assignment.

The out-of-sample extension on unseen data consists simply of calculating eq. (1.5) and assigning the test projections to the closest centroid.

An example of soft clustering performed by SKSC on a synthetic dataset is depicted in Figure 1.2. The AMS model selection criterion has been used to select the bandwidth of the RBF kernel and the optimal number of clusters. The reader can appreciate how SKSC provides more interpretable outcomes compared to KSC.

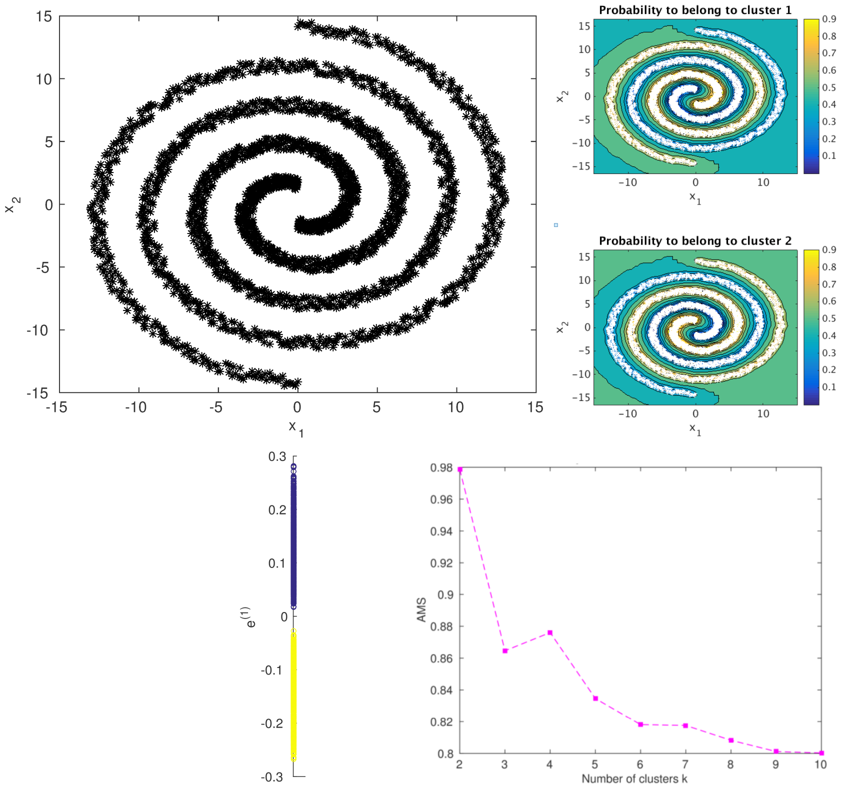


Fig. 1.2: **SKSC partitioning on a synthetic dataset**. **(Top)** Original dataset consisting of 2 clusters (left) and obtained soft clustering results (right). **(Bottom)** Points represented in the space of the projection $e^{(1)}$ (left), for an optimal choice of k (and $\sigma^2 = 1.53 \cdot 10^{-3}$) as detected by the AMS criterion (right).

The SKSC method is summarized in algorithm 2 and a Matlab implementation is freely downloadable[5].

[5] *http://www.esat.kuleuven.be/stadius/ADB/langone/softwareSKSClab.php*

Algorithm 2: SKSC algorithm (Langone, Mall & Suykens 2013)

```
Data: Training set $\mathscr{D}_{\text{tr}} = \{x_i\}_{i=1}^{N_{\text{tr}}}$ and test set $\mathscr{D}_{\text{test}} = \{x_m^{\text{test}}\}_{m=1}^{N_{\text{test}}}$, kernel function
      $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ positive definite and localized ($K(x_i, x_j) \to 0$ if $x_i$ and $x_j$ belong to
      different clusters), kernel parameters (if any), number of clusters $k$.
Result: Clusters $\{\mathscr{A}_1, \ldots, \mathscr{A}_p, \ldots, \mathscr{A}_k\}$, soft cluster memberships $sm^{(p)}, p = 1, \ldots, k$, cluster
        prototypes $\mathscr{S}\mathscr{P} = \{s_p\}_{p=1}^{k}$, $s_p \in \mathbb{R}^{k-1}$.
Initialization by solving eq. (1.4).
Compute the new prototypes $s_1, \ldots, s_k$ (eq. (1.6)).
Calculate the test data projections $e_m^{(l)}$, $m = 1, \ldots, N_{\text{test}}$, $l = 1, \ldots, k-1$.
Find the cosine distance between each projection and all the prototypes (eq. (1.7)) $\forall m$,
assign $x_m^{\text{test}}$ to cluster $A_p$ with membership $sm^{(p)}$ according to eq. (1.8).
```

1.3.3 Hierarchical Clustering

In many cases, clusters are formed by sub-clusters which in turn might have substructures. As a consequence, an algorithm able to discover a hierarchical organization of the clusters provides a more informative result, incorporating several scales in the analysis. The flat KSC algorithm has been extended in two ways in order to deal with hierarchical clustering.

1.3.3.1 Approach 1

This approach, named hierarchical kernel spectral clustering (HKSC), was proposed in (Alzate & Suykens 2012) and exploits the information of a multi-scale structure present in the data given by the Fisher criterion (see end of Section 1.3.1.3). A grid search over different values of k and σ^2 is performed to find tuning parameter pairs such that the criterion is greater than a specified threshold value. The KSC model is then trained for each pair and evaluated at the test set using the out-of-sample extension. A specialized linkage criterion determines which clusters are merging based on the evolution of the cluster memberships as the hierarchy goes up. The whole procedure is summarized in algorithm 3.

1.3.3.2 Approach 2

In (Mall, Langone & Suykens 2014*b*) and (Mall, Langone & Suykens 2014*a*) an alternative hierarchical extension of the basic KSC algorithm was introduced, for network and vector data respectively. In this method, called agglomerative hierarchical kernel spectral clustering (AH-KSC), the structure of the projections in the eigenspace is used to automatically determine a set of increasing distance thresholds. At the beginning, the validation point with maximum number of similar points within the first threshold value is selected. The indices of all these points represent

Algorithm 3: HKSC algorithm (Alzate & Suykens 2012)

```
Data: Training set $\mathscr{D}_{\mathrm{tr}} = \{x_i\}_{i=1}^{N_{\mathrm{tr}}}$, Validation set $\mathscr{D}_{\mathrm{val}} = \{x_i\}_{i=1}^{N_{\mathrm{val}}}$ and test set
    $\mathscr{D}_{\mathrm{test}} = \{x_m^{\mathrm{test}}\}_{m=1}^{N_{\mathrm{test}}}$, RBF kernel function with parameter $\sigma^2$, maximum number of
    clusters $k_{\max}$, set of $R$ $\sigma^2$ values $\{\sigma_1^2, \ldots, \sigma_R^2\}$, Fisher threshold $\theta$.
Result: Linkage matrix $Z$
For every combination of parameter pairs $(k, \sigma^2)$ train a KSC model using algorithm 1,
predict the cluster memberships for validation points and calculate the related Fisher
criterion
$\forall k$, find the maximum value of the Fisher criterion across the given range of $\sigma^2$ values. If the
maximum value is greater than the Fisher threshold $\theta$, create a set of these optimal $(k_*, \sigma_*^2)$
pairs.
Using the previously found $(k_*, \sigma_*^2)$ pairs train a clustering model and compute the cluster
memberships for the test set using the out-of-sample extension.
Create the linkage matrix $Z$ by identifying which clusters merge starting from the bottom of
the tree which contains max $k_*$ clusters.
```

the first cluster at level 0 of hierarchy. These points are then removed from the validation data matrix, and the process is repeated iteratively until the matrix becomes empty. Thus, the first level of hierarchy corresponding to the first distance threshold is obtained. To obtain the clusters at the next level of hierarchy the clusters at the previous levels are treated as data points, and the whole procedure is repeated again with other threshold values. This step takes inspiration from (Blondel et al. 2008). The algorithm stops when only one cluster remains. The same procedure is applied in the test stage, where the distance thresholds computed in the validation phase are used. An overview of all the steps involved in the algorithm is depicted in Figure 1.3. In Figure 1.4 an example of hierarchical clustering performed by this algorithm on a toy dataset is shown.

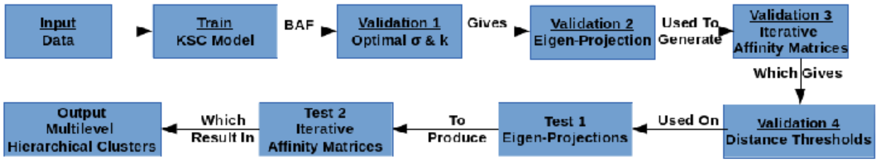


Fig. 1.3: **AH-KSC algorithm**. Steps of AH-KSC method as described in (Mall, Langone & Suykens 2014*b*) with addition of the step where the optimal σ and k are estimated.

1.3.4 Sparse Clustering Models

The computational complexity of the KSC algorithm depends on solving the eigenvalue problem (1.3) related to the training stage and computing eq. (1.5) which

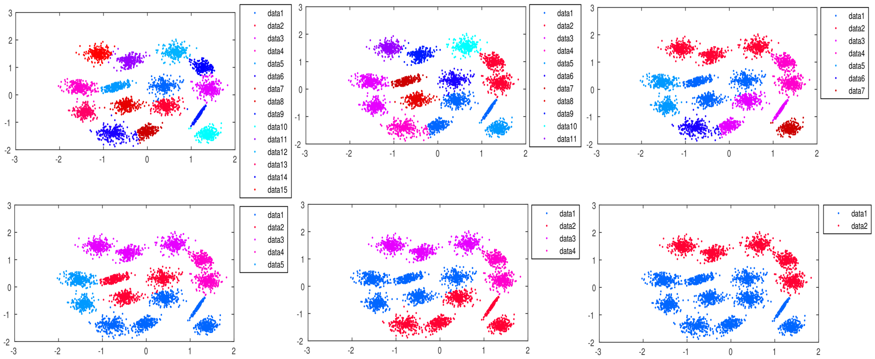


Fig. 1.4: **AH-KSC partitioning on a toy dataset.** Cluster memberships for a toy dataset at different hierarchical levels obtained by the AH-KSC method.

gives the cluster memberships of the remaining points. Assuming that we have N_{tot} data and we use N_{tr} points for training and $N_{\text{test}} = N_{\text{tot}} - N_{\text{tr}}$ as test set, the runtime of algorithm 1 is $O(N_{\text{tr}}^2) + O(N_{\text{tr}} N_{\text{test}})$. In order to reduce the computational complexity, it is then necessary to find a reduced set of training points, without loosing accuracy. In the next Sections two different methods to obtain a sparse KSC model, based on the Incomplete Cholesky Decomposition (ICD) and L_1 and L_0 penalties respectively, are discussed. In particular, thanks to the ICD, the KSC computational complexity for the training problem is decreased to $O(R^2 N_{\text{tr}})$ (Novak et al. 2015), where R indicates the reduced set size.

1.3.4.1 Incomplete Cholesky Decomposition

One of the KKT optimality conditions characterizing the Lagrangian of problem (1.1) is:

$$w^{(l)} = \Phi^T \alpha^{(l)} = \sum_{i=1}^{N_{\text{tr}}} \alpha_i^{(l)} \varphi(x_i). \tag{1.9}$$

From eq. (1.9) it is evident that each training data point contributes to the primal variable $w^{(l)}$, resulting in a non-sparse model. In order to obtain a parsimonious model a reduced set method based on the Incomplete Cholesky Decomposition (ICD) was proposed in (Alzate & Suykens 2011, Novak et al. 2015). The technique is based on finding a small number $R \ll N_{\text{tr}}$ of points $\mathscr{R} = \{\hat{x}_r\}_{r=1}^{R}$ and related coefficients $\zeta^{(l)}$ with the aim of approximating $w^{(l)}$ as:

$$w^{(l)} \approx \hat{w}^{(l)} = \sum_{r=1}^{R} \zeta_r^{(l)} \varphi(\hat{x}_r). \tag{1.10}$$

As a consequence, the projection of an arbitrary data point x into the training embedding is given by:

$$e^{(l)} \approx \hat{e}^{(l)} = \sum_{r=1}^{R} \zeta_r^{(l)} K(x, \hat{x_r}) + \hat{b_l}. \tag{1.11}$$

The set $\mathscr{R}$ of points can be obtained by considering the pivots of the ICD performed on the kernel matrix Ω. In particular, by assuming that Ω has a small numerical rank, the kernel matrix can be approximated by $\Omega \approx \hat{\Omega} = GG^T$, with $G \in \mathbb{R}^{N_{\mathrm{tr}} \times R}$. If we plug in this approximated kernel matrix in problem (1.3), the KSC eigenvalue problem can be written as:

$$\hat{D}^{-1} M_{\hat{D}} U \Psi^2 U^T \hat{\alpha}^{(l)} = \hat{\lambda}_l \hat{\alpha}^{(l)}, l = 1, \ldots, k \tag{1.12}$$

where $U \in \mathbb{R}^{N_{\mathrm{tr}} \times R}$ and $V \in \mathbb{R}^{N_{\mathrm{tr}} \times R}$ denotes the left and right singular vectors deriving from the singular value decomposition (SVD) of G, and $\Psi \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}$ is the matrix of the singular values. If now we pre-multiply both sides of eq. (1.12) by U^T and replace $\hat{\delta}^{(l)} = U^T \hat{\alpha}^{(l)}$, only the following eigenvalue problem of size $R \times R$ must be solved:

$$U^T \hat{D}^{-1} M_{\hat{D}} U \Psi^2 \hat{\delta}^{(l)} = \hat{\lambda}_l \hat{\delta}^{(l)}, l = 1, \ldots, k. \tag{1.13}$$

The approximated eigenvectors of the original problem (1.3) can be computed as $\hat{\alpha}^{(l)} = U \hat{\delta}^{(l)}$, and the sparse parameter vector can be found by solving the following optimization problem:

$$min_{\zeta^{(l)}} \parallel w^{(l)} - \hat{w}^{(l)} \parallel_2^2 = min_{\zeta^{(l)}} \parallel \Phi^T \alpha^{(l)} - \chi^T \zeta^{(l)} \parallel_2^2 . \tag{1.14}$$

The corresponding dual problem can be written as follows:

$$\Omega^{\chi\chi} \delta^{(l)} = \Omega^{\chi\phi} \alpha^{(l)}, \tag{1.15}$$

where $\Omega_{rs}^{\chi\chi} = K(\tilde{x}_r, \tilde{x}_s)$, $\Omega_{ri}^{\chi\phi} = K(\tilde{x}_r, x_i)$, $r, s = 1, \ldots, R, i = 1, \ldots, N_{tr}$ and $l = 1, \ldots, k-1$. Since the size R of problem (1.13) can be much smaller than the size N_{tr} of the starting problem, the sparse KSC method[6] is suitable for big data analytics.

1.3.4.2 Using Additional Penalty terms

In this part we explore sparsity in the KSC technique by using an additional penalty term in the objective function (1.14). In (Alzate & Suykens 2011), the authors used an L_1 penalization term in combination with the reconstruction error term to introduce sparsity. It is well known that the L_1 regularization introduces sparsity as shown in (Zhu et al. 2003). However, the resulting reduced set is neither the sparsest nor the most optimal w.r.t. the quality of clustering for the entire dataset. In

[6] A *C* implementation of the algorithm can be downloaded at: *http://www.esat.kuleuven.be/stadius/ADB/novak/softwareKSCICD.php*

(Mall, Mehrkanoon, Langone & Suykens 2014), we introduced alternative penalization techniques like Group Lasso (Yuan & Lin 2006) and (Friedman et al. 2010), L_0 and $L_1 + L_0$ penalizations. The Group Lasso penalty is ideal for clusters as it results in groups of relevant data points. The L_0 regularization calculates the number of non-zero terms in the vector. The L_0-norm results in a non-convex and NP-hard optimization problem. We modify the convex relaxation of L_0-norm based on an iterative re-weighted L_1 formulation introduced in (Candes et al. 2008, Huang et al. 2010). We apply it to obtain the optimal reduced sets for sparse kernel spectral clustering. Below we provide the formulation for Group Lasso penalized objective (1.16) and re-weighted L_1-norm penalized objectives (1.17).

The Group Lasso (Yuan & Lin 2006) based formulation for our optimization problem is:

$$\min_{\beta \in \mathbb{R}^{N_{tr} \times (k-1)}} \quad \|\Phi^\intercal \alpha - \Phi^\intercal \beta\|_2^2 + \lambda \sum_{l=1}^{N_{tr}} \sqrt{\rho_l} \|\beta_l\|_2, \tag{1.16}$$

where $\Phi = [\phi(x_1), \ldots, \phi(x_{N_{tr}})]$, $\alpha = [\alpha^{(1)}, \ldots, \alpha^{(k-1)}]$, $\alpha \in \mathbb{R}^{N_{tr} \times (k-1)}$ and $\beta = [\beta_1, \ldots, \beta_{N_{tr}}]$, $\beta \in \mathbb{R}^{N_{tr} \times (k-1)}$. Here $\alpha^{(i)} \in \mathbb{R}^{N_{tr}}$ while $\beta_j \in \mathbb{R}^{k-1}$ and we set $\sqrt{\rho_l}$ as the fraction of training points belonging to the cluster to which the l^{th} training point belongs. By varying the value of λ we control the amount of sparsity introduced in the model as it acts as a regularization parameter. In (Friedman et al. 2010), the authors show that if the initial solutions are $\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_{N_{tr}}$ then if $\|X_l^\intercal (y - \sum_{i \neq l} X_i \hat{\beta}_i)\| < \lambda$, then $\hat{\beta}_l$ is zero otherwise it satisfies: $\hat{\beta}_l = (X_l^\intercal X_l + \lambda / \|\hat{\beta}_l\|)^{-1} X_l^\intercal r_l$ where $r_l = y - \sum_{i \neq l} X_i \hat{\beta}_i$.

Analogous to this, the solution to the group lasso penalization for our problem can be defined as: $\|\phi(x_l)(\Phi^\intercal \alpha - \sum_{i \neq l} \phi(x_i) \hat{\beta}_i)\| < \lambda$ then $\hat{\beta}_l$ is zero otherwise it satisfies: $\hat{\beta}_l = (\Phi^\intercal \Phi + \lambda / \|\hat{\beta}_l\|)^{-1} \phi(x_l) r_l$ where $r_l = \Phi^\intercal \alpha - \sum_{i \neq l} \phi(x_i) \hat{\beta}_i$. The Group Lasso penalization technique can be solved by a blockwise co-ordinate descent procedure as shown in (Yuan & Lin 2006). The time complexity of the approach is $O(\text{maxiter} * k^2 N_{tr}^2)$ where maxiter is the maximum number of iterations specified for the co-ordinate descent procedure and k is the number of clusters obtained via KSC. From our experiments we observed that on an average 10 iterations suffice for convergence.

Concerning the re-weighted L_1 procedure, we modify the algorithm related to classification as shown in (Huang et al. 2010) and use it for obtaining the reduced set in our clustering setting:

$$\begin{aligned} \min_{\beta \in \mathbb{R}^{N_{tr} \times (k-1)}} \quad & \|\Phi^\intercal \alpha - \Phi^\intercal \beta\|_2^2 + \rho \sum_{i=1}^{N_{tr}} \varepsilon_i + \|\Lambda \beta\|_2^2 \\ \text{such that} \quad & \|\beta_i\|_2^2 \leq \varepsilon_i, i = 1, \ldots, N_{tr} \\ & \varepsilon_i \geq 0, \end{aligned} \tag{1.17}$$

where Λ is matrix of the same size as the β matrix i.e. $\Lambda \in \mathbb{R}^{N_{tr} \times (k-1)}$. The term $\|\Lambda\beta\|_2^2$ along with the constraint $\|\beta_i\|_2^2 \leq \varepsilon_i$ corresponds to the L_0-norm penalty on β matrix. Λ matrix is initially defined as a matrix of ones so that it gives equal chance to each element of β matrix to reduce to zero. The constraints on the optimization problem forces each element of $\beta_i \in \mathbb{R}^{(k-1)}$ to reduce to zero. This helps to overcome the problem of sparsity per component which is explained in (Alzate & Suykens 2011). The ρ variable is a regularizer which controls the amount of sparsity that is introduced by solving this optimization problem.

In Figure 1.5 an example of clustering obtained using the group lasso formulation (1.16) on a toy dataset is depicted. We can notice how the sparse KSC model is able to obtain high quality generalization using only 4 points in the training set.

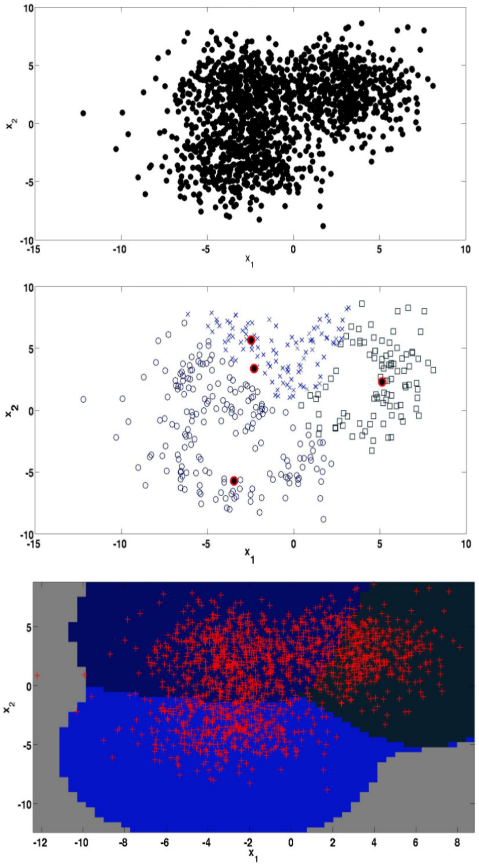


Fig. 1.5: **Sparse KSC on toy dataset**. **(Top)** Gaussian mixture with three highly overlapping components **(Center)** Clustering results, where the reduced set points are indicated with red circles **(Bottom)** Generalization boundaries.

1.4 Applications

The KSC algorithm has been successfully used in a variety of applications in different domains. In the next Sections we will illustrate various results obtained in different fields such as computer vision, information retrieval and power load consumer segmentation.

1.4.1 Image Segmentation

Image segmentation relates to partitioning a digital image into multiple regions, such that pixels in the same group share a certain visual content. In the experiments performed using KSC only the color information is exploited in order to segment the given images[7]. More precisely, a local color histogram with a 5×5 pixels window around each pixel is computed using minimum variance color quantization of 8 levels. Then, in order to compare the similarity between two histograms $h^{(i)}$ and $h^{(j)}$, the positive definite χ^2 kernel $K(h^{(i)}, h^{(j)}) = \exp(-\frac{\chi^2_{ij}}{\sigma^2_\chi})$ has been adopted (Fowlkes et al. 2004). The symbol χ^2_{ij} denotes the χ^2_{ij} statistical test used to compare two probability distributions (Puzicha et al. 1997), σ_χ as usual indicates the bandwidth of the kernel. In Figure 1.6 an example of segmentation obtained using the basic KSC algorithm is given.

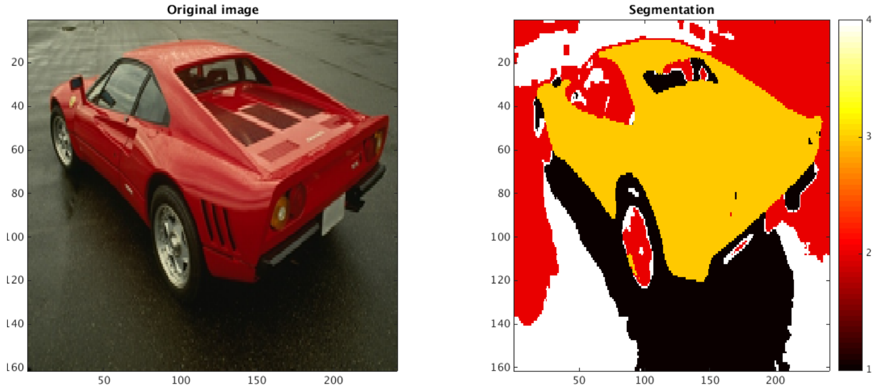


Fig. 1.6: **Image segmentation**. **(Left)** Original image **(Right)** Segmentation given by KSC.

[7] The images have been extracted from the Berkeley image database (Martin et al. 2001).

1.4.2 Scientific Journal Clustering

We present here an integrated approach for clustering scientific journals using KSC. Textual information is combined with cross-citation information in order to obtain a coherent grouping of the scientific journals and to improve over existing journal categorizations. The number of clusters k in this scenario is fixed to 22 since we want to compare the results with respect to the 22 essential science indicators (ESI) shown in Table 1.2.

Field	Name	Field	Name
1	Agricultural sciences	12	Mathematics
2	Biology and biochemistry	13	Microbiology
3	Chemistry	14	Molecular biology & genetics
4	Clinical medicine	15	Multidisciplinary
5	Computer science	16	Neuroscience & behavior
6	Economics and business	17	Pharmacology & toxicology
7	Engineering	18	Physics
8	Environment/Ecology	19	Plant & animal science
9	Geosciences	20	Psychology / Psychiatry
10	Immunology	21	Social sciences
11	Materials sciences	22	Space science

Table 1.2: The 22 science fields according to the essential science indicators (ESI)

The data correspond to more than six million scientific papers indexed by the Web of Science (WoS) in the period $2002-2006$. The type of manuscripts considered is article, letter, note and review. Textual information has been extracted from titles, abstracts and keywords of each paper together with citation information. From these data, the resulting number of journals under consideration is $8,305$.

The two resulting datasets contain textual and cross-citation information and are described as follows:

- **Term/Concept by Journal dataset:** The textual information was processed using the term frequency - inverse document frequency (TF-IDF) weighting procedure (Baeza-Yates & Ribeiro-Neto 1999). Terms which occur only in one document and stop words were not considered into the analysis. The Porter stemmer was applied to the remaining terms in the abstract, title and keyword fields. This processing leads to a term-by-document matrix of around six million papers and $669,860$ term dimensionality. The final journal-by-term dataset is a $8,305\times 669,860$ matrix. Additionally, latent semantic indexing (LSI) (Deerwester et al. 1990) was performed on this dataset to reduce the term dimensionality to 200 factors.
- **Journal cross-citation dataset:** A different form of analyzing cluster information at the journal level is through a cross-citation graph. This graph contains aggregated citations between papers forming a journal-by-journal cross-citation

matrix. The direction of the citations is not taken into account which leads to an undirected graph and a symmetric cross-citation matrix.

The cross-citation and the text/concept datasets are integrated at the kernel level by considering the following linear combination of kernel matrices[8]:

$$\Omega^{\text{integr}} = \rho\Omega^{\text{cross-cit}} + (1-\rho)\Omega^{\text{text}}$$

where $0 \leq \rho \leq 1$ is a user-defined integration weight which value can be obtained from internal validation measures for cluster distortion[9], $\Omega^{\text{cross-cit}}$ is the cross-citation kernel matrix with ij-th entry $\Omega_{ij}^{\text{cross-cit}} = K(x_i^{\text{cross-cit}}, x_j^{\text{cross-cit}})$, $x_i^{\text{cross-cit}}$ is the i-th journal represented in terms of cross-citation variables, Ω^{text} is the textual kernel matrix with ij-th entry $\Omega_{ij}^{\text{text}} = K(x_i^{\text{text}}, x_j^{\text{text}})$, x_i^{text} is the i-th journal represented in terms of textual variables and $i, j = 1, \ldots, N$.

The KSC outcomes are depicted in Tables 1.3 and 1.4. In particular, Table 1.3 shows the results in terms of internal validation of cluster quality, namely mean silhouette value (MSV) (Rousseeuw 1987) and Modularity (Newman & Girvan 2004, Newman 2006), and in terms of agreement with existing categorizations (adjusted rand index or ARI (Hubert & Arabie 1985) and normalized mutual information (NMI (Strehl & Ghosh 2002)). Finally, Table 1.4 shows the top 20 terms per cluster, which indicate a coherent structure and illustrate that KSC is able to detect the text categories present in the corpus.

	Internal validation				External validation		
	MSV textual	MSV cross-cit.	MSV integrated	Modularity cross-cit.	Modularity ISI 254	ARI 22 ESI	NMI 22 ESI
22 ESI fields	0.057	0.016	0.063	0.475	0.526*	1.000	1.000
Cross-citations	0.093	0.057	0.189	**0.547**	0.442	0.278	0.516
Textual (LSI)	0.118	0.035	0.130	0.505	0.451	0.273	0.516
Hierarch. Ward's method $\rho = 0.5$	0.121	0.055	0.190	**0.547**	**0.488**	0.285	0.540
Integr. Terms+Cross-citations $\rho = 0.5$	0.138	**0.064**	**0.201**	*0.533*	*0.465*	0.294	*0.557*
Integr. LSI+Cross-citations $\rho = 0.5$	*0.145*	*0.062*	*0.197*	0.527	*0.465*	0.308	**0.560**

Table 1.3: **Text clustering quality.** Spectral clustering results of several integration methods in terms of mean Silhouette value (MSV), modularity, adjusted Rand index (ARI) and normalized mutual information (NMI). The first four rows correspond to existing clustering results used for comparison. The last two rows correspond to the proposed spectral clustering algorithms. For external validation, the clustering results are compared with respect to the 22 ESI fields and the ISI 254 subject categories. The highest value per column is indicated in bold while the second highest value appears in italic. For MSV, a standard t-test for the difference in means revealed that differences between highest and second highest values are statistically significant at the 1% significance level (p-value $< 10^8$). The selected method for further comparisons is the integrated LSI+Cross-citations approach since it wins in external validation with one highest value (NMI) and one second highest value (Modularity).

[8] Here we use the cosine kernel described in Table 1.1.

[9] In our experiments we used the mean silhouette value (MSV) as an internal cluster validation criterion to select the value of ρ which gives more coherent clusters.

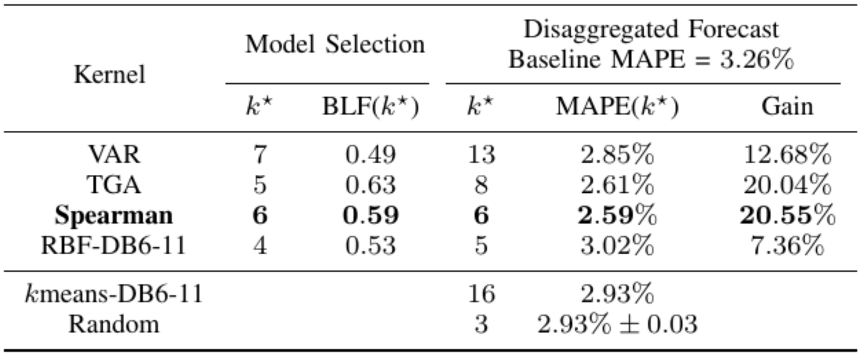
	Best 20 terms
Cluster 1	diabet therapi hospit arteri coronari physician renal hypertens mortal syndrom cardiac nurs chronic infect pain cardiovascular symptom serum cancer pulmonari
Cluster 2	polit war court reform parti legal gender urban democraci democrat civil capit feder discours economi justic privat liber union welfar
Cluster 3	diet milk fat intak cow dietari fed meat nutrit fatti chees vitamin ferment fish dry fruit antioxid breed pig egg
Cluster 4	alloi steel crack coat corros fiber concret microstructur thermal weld film deform ceram fatigu shear powder specimen grain fractur glass
Cluster 5	infect hiv vaccin viru immun dog antibodi antigen pathogen il pcr parasit viral bacteri dna therapi mice bacteria cat assai
Cluster 6	psycholog cognit mental adolesc emot symptom child anxieti student sexual interview school abus psychiatr gender attitud mother alcohol item disabl
Cluster 7	text music polit literari philosophi narr english moral book essai write discours philosoph fiction ethic poetri linguist german christian religi
Cluster 8	firm price busi trade economi invest capit tax wage financi compani incom custom sector bank organiz corpor stock employ strateg
Cluster 9	nonlinear finit asymptot veloc motion stochast elast nois turbul ltd vibrat iter crack vehicl infin singular shear polynomi mesh fuzzi
Cluster 10	soil seed forest crop leaf cultivar seedl ha shoot fruit wheat fertil veget germin rice flower season irrig dry weed
Cluster 11	soil sediment river sea climat land lake pollut wast fuel wind ocean atmospher ic emiss reactor season forest urban basin
Cluster 12	algebra theorem manifold let finit infin polynomi invari omega singular inequ compact lambda graph conjectur convex proof asymptot bar phi
Cluster 13	pain surgeri injuri lesion muscl bone brain ey surgic nerv mri ct syndrom fractur motor implant arteri knee spinal stroke
Cluster 14	rock basin fault sediment miner ma tecton isotop mantl volcan metamorph seismic sea magma faci earthquak ocean cretac crust sedimentari
Cluster 15	web graph fuzzi logic queri schedul semant robot machin video wireless neural node internet traffic processor retriev execut fault packet
Cluster 16	student school teacher teach classroom instruct skill academ curriculum literaci learner colleg write profession disabl faculti english cognit peer gender
Cluster 17	habitat genu fish sp forest predat egg nest larva reproduct taxa bird season prei nov ecolog island breed mate genera
Cluster 18	star galaxi solar quantum neutrino orbit quark gravit cosmolog decai nucleon emiss radio nuclei relativist neutron cosmic gaug telescop hole
Cluster 19	film laser crystal quantum atom ion beam si nm dope thermal spin silicon glass scatter dielectr voltag excit diffract spectra
Cluster 20	polym catalyst ion bond crystal solvent ligand hydrogen nmr molecul atom polymer poli aqueou adsorpt methyl film spectroscopi electrod bi
Cluster 21	receptor rat dna neuron mice enzym genom transcript brain mutat peptid kinas inhibitor metabol cancer mrna muscl ca2 vitro chromosom
Cluster 22	cancer tumor carcinoma breast therapi prostat malign chemotherapi tumour surgeri lesion lymphoma pancreat recurr resect surgic liver lung gastric node

Table 1.4: **Text clustering results.** Best 20 terms per cluster according to the integrated results (LSI+cross-citation) with $\rho = 0.5$. The terms found display a coherent structure in the clusters.

1.4.3 Power Load Clustering

Accurate power load forecasts are essential in electrical grids and markets particularly for planning and control operations (Alzate et al. 2009). In this scenario, we apply KSC for finding power load smart meter data that are similar in order to aggregate them and improve the forecasting accuracy of the global consumption signal. The idea is to fit a forecasting model on the aggregated load of each cluster (aggregator). The k predictions are summed to form the final disaggregated prediction. The number of clusters and the time series used for each aggregator are determined via KSC (Alzate & Sinn 2013). The forecasting model used is a periodic autoregresive model with exogenous variables (PARX) (Espinoza et al. 2005).

Table 1.7 (taken from (Alzate & Sinn 2013) shows the model selection and disaggregation results. Several kernels appropriate for time series were tried including a Vector Autoregressive (VAR) kernel [Add: Cuturi, Autoregressive kernels for time series, arXiv], Triangular Global Alignment (TGA) kernel [Add: Cuturi, Fast Global Alignment Kernels, ICML 2011] and an RBF kernel with Spearman's distance. The results show an improvement of 20.55% with the similarity based on Spearman's corrleation in the forecasting accuracy compared to not using clustering at all (i.e., aggregating all smart meters). The BLF was also able to detect the number of clusters that maximize the improvement (6 clusters in this case).



Kernel	Model Selection		Disaggregated Forecast Baseline MAPE = 3.26%		
	$k^\star$	BLF($k^\star$)	$k^\star$	MAPE($k^\star$)	Gain
VAR	7	0.49	13	2.85%	12.68%
TGA	5	0.63	8	2.61%	20.04%
Spearman	**6**	**0.59**	**6**	**2.59%**	**20.55%**
RBF-DB6-11	4	0.53	5	3.02%	7.36%
$k$means-DB6-11			16	2.93%	
Random			3	2.93% ± 0.03	

Fig. 1.7: **Kernel comparisons for power load clustering data**. Model selection and forecasting results in terms of the mean absolute percentage error (MAPE). RBF-DB6-11 refers to using the RBF kernel on the detail coefficients using wavelets (DB6, 11 levels). The winner is the Spearman-based kernel with a improvement of 20.55%. For this kernel, the number of clusters *k* found by the BLF also coincides with the number of aggregators needed to maximize the improvement.

1.4.4 Big data

KSC has been shown to be effective in handling big data at a desktop PC scale. In particular, in (Mall et al. 2013*b*), we focused on community detection in big networks containing millions of nodes and several million edges, and we explained how to scale our method by means of three steps[10]. First, we select a smaller subgraph that preserves the overall community structure by using the FURS algorithm (Mall et al. 2013*a*), where hubs in dense regions of the original graph are selected via a greedy activation-deactivation procedure. In this way the kernel matrix related to subgraph fits the main memory and the KSC model can be quickly trained by

[10] A *Matlab* implementation of the algorithm can be downloaded at: *http://www.esat.kuleuven.be/stadius/ADB/mall/softwareKSCnet.php*

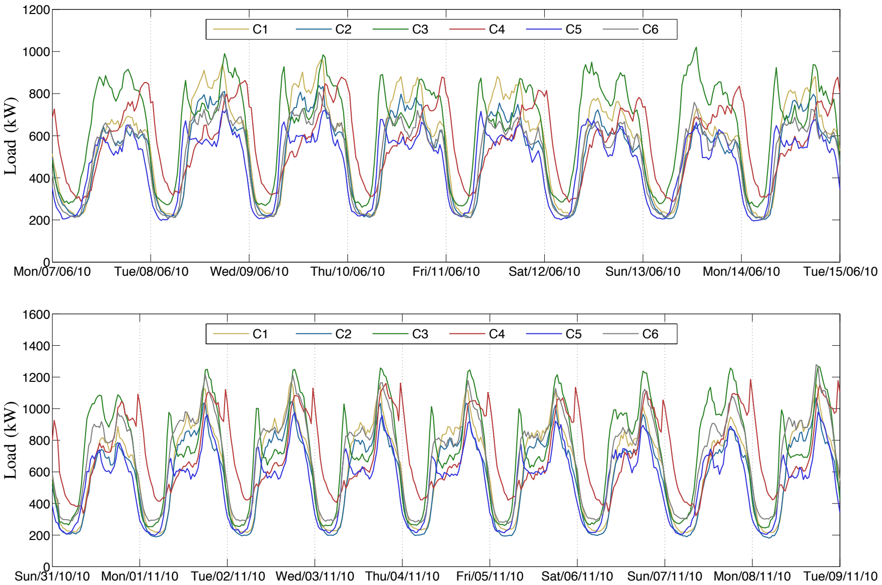


Fig. 1.8: **Power load clustering results**. Visualization of the 6 clusters obtained by KSC. **(Top)** Aggregated load in summer. **(Bottom)** Aggregated load in winter. The daily cycles are clearly visible and the clusters capture different characteristics of the consumption pattern. This clustering result improves the forecasting accuracy by 20.55%

solving a smaller eigenvalue problem. Then the BAF criterion described in Section 1.3.1.3, which is memory and computationally efficient, is used for model selection[11]. Finally, the out-of-sample extension is used to infer the cluster memberships for the remaining nodes forming the test set (which is divided into chunks due to memory constraints).

In (Mall, Langone & Suykens 2014*b*) the hierarchical clustering technique summarized in Section 1.3.3.2 has been used to perform community detection in real-life networks at different resolutions. The method has been shown to be able to detect complex structures at various hierarchical levels, by not suffering of any resolution limit. An example of results obtained on the *Cond-mat* network of collaborations between authors of papers submitted to Condense Matter category in *Arxiv* (Leskovec et al. 2007) is shown in Figure 1.9.

Finally, in (Mall, Jumutc, Langone & Suykens 2014), we propose a deterministic method to obtain subsets from big vector data which are a good representative of the inherent clustering structure. We first convert the large scale dataset into a sparse undirected k-NN graph using a Map-Reduce framework. Then, the FURS method is used to select a few representative nodes from this graph, corresponding to certain

[11] In (Mall et al. 2013*c*) this model selection step has been eliminated by proposing a self tuned method where the structure of the projections in the eigenspace is exploited to automatically identify an optimal cluster structure.

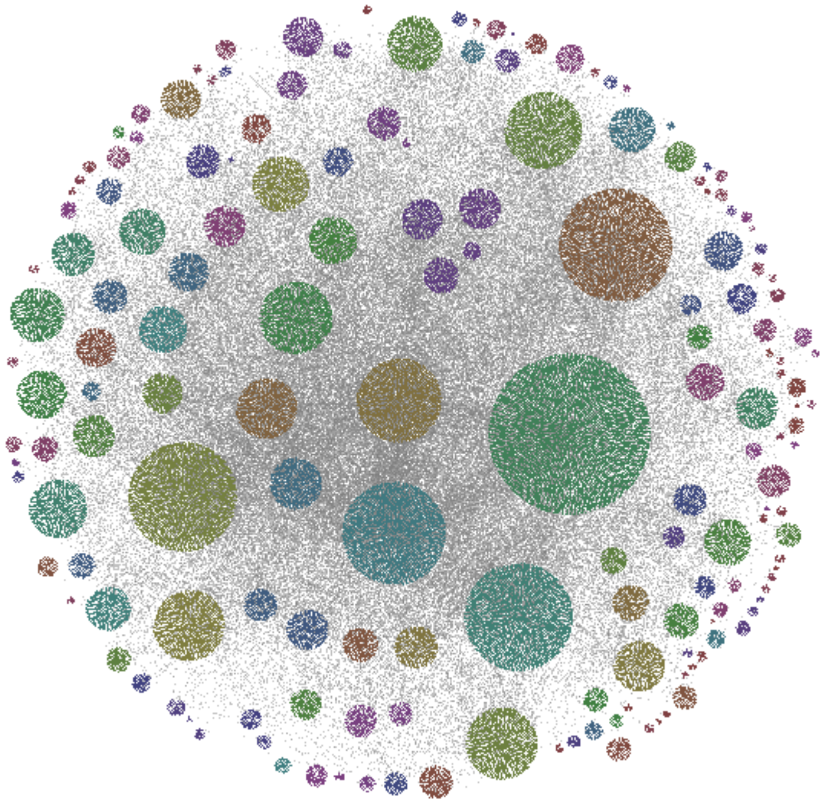


Fig. 1.9: **Large scale community detection**. Community structure detected at one particular hierarchical level by the AH-KSC method summarized in Section 1.3.3.2, related to the *Cond-Mat* collaboration network.

data points in the original dataset. These points are then used to quickly train the KSC model, while the generalization property of the method is exploited to compute the cluster memberships for the remainder of the dataset. In Figure 1.10 a summary of all these steps is sketched.

1.5 Conclusions

In this chapter we have discussed the kernel spectral clustering (KSC) method, which is cast in an LS-SVM learning framework. We have explained that, like in the classifier case, the clustering model can be trained on a subset of the data with

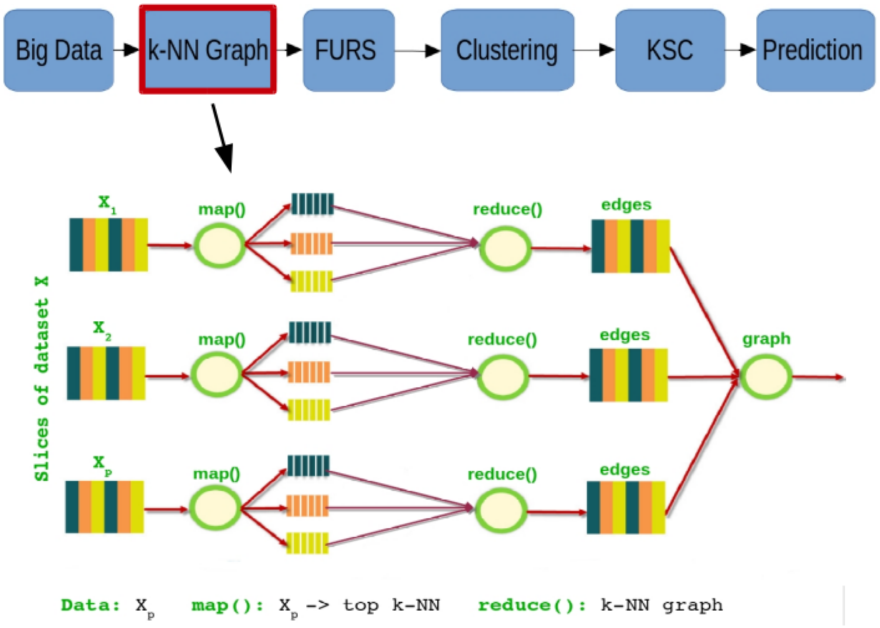


Fig. 1.10: **Big data clustering**. **(Top)** Illustration of the steps involved in clustering big vector data using KSC. **(Bottom)** Map-Reduce procedure used to obtain a representative training subset by constructing a k-NN graph.

optimal tuning parameters, found during the validation stage. The model is then able to generalize to unseen test data thanks to its out-of-sample extension property. Beyond the core algorithm, some extensions of KSC allowing to produce probabilistic and hierarchical outputs have been illustrated. Furthermore, two different approaches to sparsify the model based on the Incomplete Cholesky Decomposition (ICD) and L_1 and L_0 penalties have been described. This allows to handle large scale data at a desktop scale. Finally, a number of applications in various fields ranging from computer vision to text mining have been examined.

Acknowledgements EU: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC AdG A-DATADRIVE-B (290923). This chapter reflects only the authors' views, the Union is not liable for any use that may be made of the contained information. Research Council KUL: GOA/10/09 MaNet, CoE PFV/10/002 (OPTEC), BIL12/11T; PhD/Postdoc grants. Flemish Government: FWO: projects: G.0377.12 (Structured systems), G.088114N (Tensor based data similarity); PhD/Postdoc grants. IWT: projects: SBO POM (100031); PhD/Postdoc grants. iMinds Medical Information Technologies SBO 2014. Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017.)

References

Alzate C, Espinoza M, De Moor B & Suykens J A K 2009 *in* 'Proceedings of the 19th International Conference on Neural Networks (ICANN 2009)' pp. 315–324.

Alzate C & Sinn M 2013 *in* 'ICDM' pp. 943–948.

Alzate C & Suykens J A K 2010 *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**(2), 335–347.

Alzate C & Suykens J A K 2011 *Neurocomputing* **74**(9), 1382–1390.

Alzate C & Suykens J A K 2012 *Neural Networks* **35**, 21–30.

Baeza-Yates R & Ribeiro-Neto B 1999 *Modern Information Retrieval* Addison-Wesley.

Ben-Israel A & Iyigun C 2008 *J. Classif.* **25**(1), 5–26.

Bishop C M 2006 *Pattern Recognition and Machine Learning (Information Science and Statistics)* Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Blondel V D, Guillaume J L, Lambiotte R & Lefebvre E 2008 *Journal of Statistical Mechanics: Theory and Experiment* **2008**(10), P10008.

Candes E J, Wakin M B & Boyd S 2008 *Journal of Fourier Analysis and Applications, special issue on sparsity* **14**(5), 877–905.

Chung F R K 1997 *Spectral Graph Theory* American Mathematical Society.

De Brabanter K, De Brabanter J, Suykens J A K & De Moor B 2010 *Comput. Stat. Data Anal.* **54**(6), 1484–1504.

Deerwester S C, Dumais S T, Landauer T K, Furnas G W & Harshman R A 1990 *Journal of the American Society for Information Science* **41**(6), 391–407.

Delvenne J C, Yaliraki S N & Barahona M 2010 *Proceedings of the National Academy of Sciences* **107**(29), 12755–12760.

Dhanjal C, Gaudel R & Clemenccon S 2013 *arXiv/1301.1318* .

Espinoza M, Joye C, Belmans R & De Moor B 2005 *IEEE Transactions on Power System* **20**(3), 1622–1630.

Fowlkes C, Belongie S, Chung F & Malik J 2004 *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**(2), 214–225.

Frederix K & Van Barel M 2013 *J. Comput. Appl. Math.* **237**(1), 145–161.

Friedman J, Hastie T & Tibshirani R 2010 *arXiv:1001.0736* .

Huang K, Zheng D, Sun J, Hotta Y, Fujimoto K & Naoi S 2010 *Pattern Recognition Letters* **31**(13), 1944–1951.

Hubert L & Arabie P 1985 *Journal of Classification* **1**(2), 193–218.

Langone R, Agudelo O M, De Moor B & Suykens J A K 2014 *Neurocomputing* **139**(0), 246–260.

Langone R, Alzate C, De Ketelaere B & Suykens J A K 2013 *in* 'IEEE Symposium Series on Computational Intelligence and data mining SSCI (CIDM) 2013' pp. 39–45.

Langone R, Alzate C, De Ketelaere B, Vlasselaer J, Meert W & Suykens J A K 2015 *Engineering Applications of Artificial Intelligence* **37**, 268–278.

Langone R, Alzate C & Suykens J A K 2011 *in* 'Proc. of the International Joint Conference on Neural Networks (IJCNN 2011)' pp. 1849–1856.

Langone R, Alzate C & Suykens J A K 2012 *in* 'Proc. of the International Joint Conference on Neural Networks (IJCNN 2012)' pp. 2596–2603.

Langone R, Alzate C & Suykens J A K 2013 *Physica A: Statistical Mechanics and its Applications* **392**(10), 2588–2606.

Langone R, Mall R & Suykens J A K 2013 *in* 'Proc. of the International Joint Conference on Neural Networks (IJCNN 2013)' pp. 1–8.

Langone R, Mall R & Suykens J A K 2014 *SSCI (CIDM) 2014* pp. 1–8.

Langone R & Suykens J A K 2013 *Journal of Physics: Conference Series* **410**(1), 012100.

Leskovec J, Kleinberg J & Faloutsos C 2007 *ACM Trans. Knowl. Discov. Data* **1**(1).

Liao T W 2005 *Pattern Recognition* **38**(11), 1857 – 1874.

Lin F & Cohen W W 2010 *in* 'ICML' pp. 655–662.

Mall R, Jumutc V, Langone R & Suykens J A K 2014 *in* 'IEEE International Conference on Big Data' pp. 37–42.

Mall R, Langone R & Suykens J 2013*a* *Social Network Analysis and Mining* **3**(4), 1–21.

Mall R, Langone R & Suykens J A K 2013*b* *Entropy (Special Issue on Big Data)* **15**(5), 1567–1586.

Mall R, Langone R & Suykens J A K 2013*c* *in* 'IEEE International Conference on Big Data'.

Mall R, Langone R & Suykens J A K 2014*a* *in* 'Symposium Series on Computational Intelligence (SSCI-CIDM)' pp. 1–8.

Mall R, Langone R & Suykens J A K 2014*b* *PLoS ONE* **9**(6), e99966.

Mall R, Mehrkanoon S, Langone R & Suykens J A K 2014 *in* 'Proc. of the International Joint Conference on Neural Networks (IJCNN 2014)' pp. 2436–2443.

Martin D, Fowlkes C, Tal D & Malik J 2001 *in* 'Proc. 8th Int'l Conf. Computer Vision' Vol. 2 pp. 416–423.

Meila M & Shi J 2001*a* *in* T. K Leen, T. G Dietterich & V Tresp, eds, 'Advances in Neural Information Processing Systems 13' MIT Press.

Meila M & Shi J 2001*b* *in* 'Artificial Intelligence and Statistics AISTATS'.

Mika S, Schölkopf B, Smola A J, Müller K R, Scholz M & Rätsch G 1999 *in* M. S Kearns, S. A Solla & D. A Cohn, eds, 'Advances in Neural Information Processing Systems 11' MIT Press.

Newman M E J 2006 *Proc. Natl. Acad. Sci. USA* **103**(23), 8577–8582.

Newman M E J & Girvan M 2004 *Physical Review E* **69**(2).

Ng A Y, Jordan M I & Weiss Y 2002 *in* T. G Dietterich, S Becker & Z Ghahramani, eds, 'Advances in Neural Information Processing Systems 14' MIT Press Cambridge, MA pp. 849–856.

Ning H, Xu W, Chi Y, Gong Y & Huang T S 2010 *Pattern Recogn.* **43**(1), 113–127.

Novak M, Alzate C, langone R & Suykens J A K 2015 *Internal Report 14-119, ESAT-SISTA, KU Leuven (Leuven, Belgium)* .

Peluffo D, Garcia S, Langone R, Suykens J A K & Castellanos G 2013 *in* 'Proc. of the International Joint Conference on Neural Networks (IJCNN 2013)' pp. 1085 – 1090.

Puzicha J, Hofmann T & Buhmann J 1997 *in* 'Computer Vision and Pattern Recognition' pp. 267–272.

Rousseeuw P J 1987 *Journal of Computational and Applied Mathematics* **20**(1), 53–65.

Schölkopf B, Smola A J & Müller K R 1998 *Neural Computation* **10**, 1299–1319.

Shi J & Malik J 2000 *IEEE Trans. Pattern Anal. Machine Intell.* **22**(8), 888–905.

Strehl A & Ghosh J 2002 *Journal of Machine Learning Research* **3**, 583–617.

Suykens J A K, Van Gestel T, De Brabanter J, De Moor B & Vandewalle J 2002 *Least Squares Support Vector Machines* World Scientific, Singapore.

Suykens J A K, Van Gestel T, Vandewalle J & De Moor B 2003 *IEEE Transactions on Neural Networks* **14**(2), 447–450.

von Luxburg U 2007 *Statistics and Computing* **17**(4), 395–416.

Williams C K I & Seeger M 2001 *in* 'Advances in Neural Information Processing Systems 13' MIT Press.

Yuan M & Lin Y 2006 *Journal of Royal Statistical Society* **68**(1), 49–67.

Zhu J, Rosset S, Hastie T & Tibshirani R 2003 *in* 'Neural Information Processing Systems' Vol. 16.