# Clustering data over time using kernel spectral clustering with memory

Rocco Langone[1], Raghvendra Mall[1], Johan A. K. Suykens[1]
[1]Department of Electrical Engineering (ESAT), STADIUS, KU Leuven, B-3001 Leuven Belgium
Email: {rocco.langone,raghvendra.mall,johan.suykens}@esat.kuleuven.be

***Abstract*—This paper discusses the problem of clustering data changing over time, a research domain that is attracting increasing attention due to the increased availability of streaming data in the Web 2.0 era. In the analysis conducted throughout the paper we make use of the kernel spectral clustering with memory (MKSC) algorithm, which is developed in a constrained optimization setting. Since the objective function of the MKSC model is designed to explicitly incorporate temporal smoothness, the algorithm belongs to the family of evolutionary clustering methods. Experiments over a number of real and synthetic datasets provide very interesting insights in the dynamics of the clusters evolution. Specifically, MKSC is able to handle objects leaving and entering over time, and recognize events like continuing, shrinking, growing, splitting, merging, dissolving and forming of clusters. Moreover, we discover how one of the regularization constants of the MKSC model, referred as the smoothness parameter, can be used as a change indicator measure. Finally, some possible visualizations of the cluster dynamics are proposed.**

## I. Introduction

In many practical applications such as community detection of dynamic networks [1], tracking moving objects [2], online fault detection in industrial machines [3] etc., we deal with clustering in dynamic scenarios. This is a challenging problem where the clusters evolve with time via long-term drifts and short-term variations due to noise.

In order to produce a meaningful clustering result at each time step robust to noise, the evolutionary clustering framework was proposed in [4]. This work is based on the intuition that if the new data does not deviate from the recent history the clustering should be similar to that performed for the previous data. However, if the data changes significantly, the clustering must be modified to reflect the new structure. This temporal smoothness between clusters in successive time-steps is also the main principle behind the methods introduced in [5], [6] and [7]. In particular, in [5] the evolutionary spectral clustering algorithm (ESC) has been proposed, which aims to optimize the cost function $J_{tot} = \eta J_{\text{temp}} + (1-\eta) J_{\text{snap}}$. $J_{\text{snap}}$ describes the classical spectral clustering objective [8], [9], [10] related to each snapshot of an evolving data set. $J_{\text{temp}}$ measures the cost of applying the partitioning found at time $t$ to the snapshot at time $t-1$, penalizing then clustering results that disagree with the recent past. In [7] an evolutionary clustering framework that adaptively estimates the optimal smoothing parameter using shrinkage estimation is presented. The method, called AFFECT, allows to extend a number of static clustering algorithms into evolutionary clustering techniques.

In this paper we use kernel spectral clustering with memory (MKSC) [11], [12] to perform clustering of evolving data. The technique has been developed in the Least Squares Support Vector Machines (LS-SVMs [13]) primal-dual optimization setting, where the temporal smoothness between clusters in successive time steps is incorporated at the primal level[1]. Moreover, its nature of being cast in a learning framework allows a precise model selection scheme and the out-of-sample extension to new data points. In [11] we have already shown that MKSC was able to produce consistent, smooth and high quality partitions over time. However, the method was limited to the case where both the objects to be clustered and the number of clusters was not varying over time.

The specific contributions of this paper can be summarized as follows:

- dealing with a variable number of objects and clusters over time. By properly rearranging the data matrices and the solution vectors $\alpha^{(l)}$ of our model, we allow MKSC to recognize a bigger variety of events (see Section III-A).
- matching the clusters in successive time steps. In Section III-B we introduce a novel methodology to perform one-to-one, many-to-one and one-to-many cluster matching.
- we show how the regularization constant $\nu$ can be used as a change detection measure, in order to reveal important change points in the data. As a consequence, unlike in [11], the clustering result are smoothed automatically only when it is needed[2].
- model selection. The tuning of the number of clusters, the kernel hyper-parameters (if any) and the regularization constant at each time step is described in Section III-C.
- visualization of the clusters. We provide two ways of visualizing the clusters evolution over time: by using a 3D embedding in the space of the dual variables $\alpha^{(l)}$ and by means of the adjacency matrices related to the networks constructed in our tracking mechanism.

The rest of this paper is organized in the following way. Section II recalls the MKSC model. In Section III the cluster

[1]For this reason, it belongs to the family of evolutionary clustering algorithms.

[2]In [11] we were considering the smoothness of the clustering results as a prior knowledge to be fulfilled at each time stamp, thus the regularization or smoothness parameter $\nu$ was fixed to 1.

matching problem, the appearance and disappearance of objects over time and the model selection issue are discussed. Section IV describes the data sets that have been used in the experiments. The simulations, together with a discussion related to the computational complexity of our technique, are illustrated in Section V. Finally we draw some concluding remarks and we suggest future research directions.

## II. The MKSC model

Let $\mathcal{S} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_i\}_{i=1}^{T}$ be a sequence of snapshots (data matrices or networks[3]) over the time period $T$. The problem of dynamic clustering can be described as the task of obtaining a partition of the objects at each time step.

Given that at the current time $t$ the data set to be clustered comprises $N$ data points of dimension $d$, the primal problem of the MKSC model, where $N_{\text{Tr}}$ data points are used for training, can be stated as follows in matrix notation [11]:

$$\begin{aligned} \min_{w_t^{(l)}, e_t^{(l)}, b_t^l} \quad & \frac{1}{2}\sum_{l=1}^{k-1} {w_t^{(l)}}^T w_t^{(l)} - \frac{\gamma_t}{2N_{\text{Tr}}}\sum_{l=1}^{k-1} {e_t^{(l)}}^T D_{\text{Mem}}^{-1} e_t^{(l)} \\ & - \nu_t \sum_{l=1}^{k-1} {w_t^{(l)}}^T \sum_{i=1}^{M} w_{t-i}^{(l)} \\ \text{subject to} \quad & e_t^{(l)} = \Phi_t w_t^{(l)} + b_t^l 1_{N_{\text{Tr}}}. \end{aligned} \tag{1}$$

The first term in the objective (1) indicates the minimization of the model complexity, while the second term casts the clustering problem in a weighted kernel PCA formulation as in [14]. The third term describes the correlation between the actual and the previous models, which we want to maximize. In this way it is possible to introduce temporal smoothness in our formulation, such that the current partition does not deviate too dramatically from the recent past. The subscript *Mem* refer to time steps $t-1, \ldots, t-M$, where $M$ indicates the memory, that is the past information we want to consider when performing the clustering at the actual time step $t$. The symbols have the following meaning[4]:

- $e^{(l)}$ represent the $l$-th binary clustering model for the $N_{\text{Tr}}$ points and are referred interchangeably as projections, latent variables or score variables.
- the index $l = 1, \ldots, k-1$ indicates the score variables needed to encode the $k$ clusters to find via an Error Correcting Output Codes (ECOC) encoding-decoding procedure. In other words, $e_i^{(l)} = {w^{(l)}}^T \varphi(x_i) + b_l$ are the latent variables of a set of $k-1$ binary clustering indicators given by $\text{sign}(e_i^{(l)})$. The binary indicators are combined to form a codebook $\mathcal{CB} = \{c_p\}_{p=1}^{k}$, where each codeword is a string of length $k-1$ representing a cluster.
- $w^{(l)} \in \mathbb{R}^{d_h}$ and $b_l$ are the parameters of the (primal) model at time $t$.
- $D^{-1} \in \mathbb{R}^{N_{\text{Tr}} \times N_{\text{Tr}}}$ is the inverse of the degree matrix $D$ related to the actual kernel matrix $\Omega$, i.e. $D_{ii} = \sum_j \Omega_{ij}$, while $D_{\text{Mem}}^{-1} \in \mathbb{R}^{N_{\text{Tr}} \times N_{\text{Tr}}}$ is the inverse of the degree matrix $D_{\text{Mem}} = D + \sum_{i=1}^{M} D_{\text{t-i}}$.
- $\Phi$ is the $N_{\text{Tr}} \times d_h$ feature matrix $\Phi = [\varphi(x_1)^T; \ldots; \varphi(x_{N_{\text{Tr}}})^T]$ which expresses the relationship between each pair of data objects in a high dimensional feature space $\varphi : \mathbb{R}^d \to \mathbb{R}^{d_h}$.
- $\gamma \in \mathbb{R}^+$ and $\nu \in \mathbb{R}^+$ are regularization constants. In particular, $\nu$ can be thought as a smoothness parameter, because it enforces the current model to resemble the old models developed for the previous $M$ snapshots.

The dual solution to problem (1) becomes [11]:

$$(D_{\text{Mem}}^{-1} M_{D_{\text{Mem}}} \Omega_t - \frac{I}{\gamma_t})\alpha_t^{(l)} = -\nu_t D_{\text{Mem}}^{-1} M_{D_{\text{Mem}}} \sum_{i=1}^{M} \Omega_{t-i}\alpha_{t-i}^{(l)} \tag{2}$$

where

- $\Omega_t$ indicates the current kernel matrix with $ij$-th entry $\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$. $\Omega_{t-i}$ captures the similarity between the objects of the current snapshot and the ones of the previous $M$ snapshots.
- $M_{D_{\text{Mem}}}$ is the centering matrix equal to $M_{D_{\text{Mem}}} = I_{N_{\text{Tr}}} - \frac{1}{1_{N_{\text{Tr}}}^T D_{\text{Mem}}^{-1} 1_{N_{\text{Tr}}}} 1_{N_{\text{Tr}}} 1_{N_{\text{Tr}}}^T D_{\text{Mem}}^{-1}$.

The cluster indicators for the training data are:

$$\text{sign}(e_t^{(l)}) = \text{sign}(\Omega_t \alpha_t^{(l)} + \nu_t \sum_{i=1}^{M} \Omega_{t-i}\alpha_{t-i}^{(l)} + b_t^l 1_{N_{\text{Tr}}}). \tag{3}$$

The score variables for test points are defined as follows:

$$e_t^{(l),\text{test}} = \Omega_t^{\text{test}} \alpha_t^{(l)} + \nu_t \sum_{i=1}^{M} \Omega_{t-i}^{\text{test}} \alpha_{t-i}^{(l)} + b_t^l 1_{N_{\text{test}}}. \tag{4}$$

Thus, once we have properly trained our model, the cluster memberships of new points can be predicted by projecting the test data into the solution vectors $\alpha_t^{(l)}, \ldots, \alpha_{t-M}^{(l)}$ via eq. (4).

## III. Clustering evolving data using MKSC

In a general scenario, at each time step $t$ the number of objects to group can be different from the previous time steps, and the same can happen for the community structure. These problems are discussed in the forthcoming Sections, together with the model selection issue.

### A. Objects appearing and leaving over time

When performing the clustering for the actual data snapshot present at time $t$, two possible situations can arise: new data points are introduced or some existing objects may be disappeared. To cope with the first scenario, the rows of the old data matrices corresponding to the new points can be set to zero, as well as the related components of the solution vectors $\alpha_{t-1}^{(l)}, \ldots, \alpha_{t-M}^{(l)}$. In this way, when solving problem (2), the components of $\alpha_t^{(l)}$ related to the new objects have no influence from the past. On the other hand, data points that were present in the previous snapshots but not in the actual one can simply be removed.

[3]Along the paper we assume that if the objects to group are nodes of a network, the current data is represented by an $N \times N$ adjacency matrix, otherwise by an $N \times d$ data matrix, where $N$ indicates the number of objects and $d$ the dimensionality of the data.

[4]For the sake of clarity we omit the time index $t$.

### B. Tracking the clusters

Several events that happen during the evolution of clusters are continuing, shrinking, growing, splitting, merging, dissolving and forming of clusters. In order to recognize these circumstances a tracking algorithm that matches the partitions found at each time step is needed, like the ones proposed in [15] and [16]. In this realm, we introduce the following tracking method.

We generate a directed weighted network $W_N^t$ from the clusters at two consecutive timestamps $t$ and $t+1$. Thus, if we have $T$ timestamps we generate a set $\mathcal{W}_\mathcal{N} = \{W_N^1, \ldots, W_N^{T-1}\}$ of directed weighted networks. Each directed weighted network $W_N^t$ creates a map between the clusters at time stamp $t$ i.e $C^t$ and time stamp $t+1$ i.e. $C^{t+1}$, which form the edges of the network. The weight $v^t(j,k)$ of an edge between two clusters is equivalent to the fraction of nodes in cluster $C_j$ at time stamp $t$ which are assigned to cluster $C_k$ at time stamp $t+1$. An edge exists between two clusters $C_j$ and $C_k$ only if $v^t(j,k) > 0$. Thus, if the number of edges going out of a node of $W_N^i$ is greater than 1 it indicates a split whereas if the number of edges entering a node is greater than 1 then it indicates a merge. In case $v^t(j,k) = 1.0$ then the cluster remains unchanged between the 2 time intervals $t$ and $t+1$. In order to tackle the birth and death of clusters, we add a $C_0$ cluster for each time stamp $t$. For the network $W_N^t$, if $C_0^t$ is isolated then no new clusters were generated at time $t$ and if $C_0^{t+1}$ is isolated then none of the clusters present at time stamp $t$ dissolved in the next snapshot. However, if we have outgoing edges from $C_0^t$ then there was birth of new clusters. Similarly, if we have incoming edges to $C_0^{t+1}$ in $W_N^t$ then some clusters dissolved at time $t$. The whole procedure is summarized in algorithm 1, and in Figure 1 an example of the matching mechanism is given.

### C. Model selection

A right way of choosing the tuning parameters in a kernel-based model is critical to ensure good performance in a given task. To perform the model selection for MKSC we use a grid search approach. Unlike the model selection algorithm described in [11], now the number of clusters $k$ is not fixed but it is tuned and $\nu$ is selected instead of $\gamma$. In the experimental results reported in Section V we utilize the Average Membership Strength (AMS) criterion [17], the Silhouette index [18] and the Modularity quality function [19], [20] to perform model selection. Moreover, for network data the Fast and Unique Representative Subset Selection (FURS [21]) is used to select the training and validation sets, otherwise the Renyi entropy method is chosen [13]. Finally, the proportion of training and validation data is set to $15\%$ and $30\%$ respectively.

The complete algorithm to perform dynamic clustering and track the clusters over time[5] can be resumed in algorithm 2.

[5]Unlike the basic algorithm described in [11], which could not cope with objects entering and leaving over time and could not handle a variable number of clusters, now it is possible to cope with these scenarios. Moreover, a tracking algorithm matching the clusters in successive time steps is also present.

**Algorithm 1:** Cluster Tracking Algorithm

**Data**: At a given time stamp $t$, take the clustering information of time stamp $t-1$ and $t$ i.e. $C^{t-1}$ and $C^t$.
**Result**: A weighted directed network $W_N^t$ tracking the relationship between the clusters at time stamps $t-1$ and $t$.
**foreach** $C_j^{t-1} \subset C^{t-1}$ **do**
  **foreach** $C_k^t \subset C^t$ **do**
    **if** *Nodes with labels* $c_j^{t-1}$ *at* $t-1$ *have the label* $c_k^t$ *at* $t$ **then**
      Create temporary edge $v^t(j,k)$ between $C_j^{t-1}$ and $C_k^t$.
      $n(j,k)$ = Number of nodes with labels $c_j^{t-1}$ at $t-1$ which have the label $c_k^t$ at $t$.
      Weight of the edge $v^t(j,k) = \frac{n(j,k)}{|C_j^t|}$
    **end**
  **end**
**end**
Keep the edge which has maximum weight (in case of multiple edges keep all) w.r.t. $C_j^{t-1}$.
Add this weighted edge to the graph $W_N^t$.
**foreach** $C_k^t \subset C^t$ **do**
  **if** $C_k^t$ *is isolated* **then**
    Select the edge $v^t(j,k)$ which had maximum incoming weight w.r.t. $C_k^t$. Add this weighted edge to the graph $W_N^t$.
    `/* This is done in order to prevent isolated nodes in` $W_N^t$. `*/`
  **end**
**end**

## IV. Description of the data sets

In the experiments discussed in the next Section we have used both synthetic and real-life datasets.

The artificial benchmarks consist of evolving networks generated by the software related to [15]:

- **MergesplitNet**: an initial network of 1000 nodes formed by 7 communities evolves over 5 time steps. At each time step there are 2 splitting events and 1 merging event. The number of nodes remains unchanged.
- **BirthdeathNet**: a starting network with 13 communities experiences at each time one cluster death and one cluster generation, while the number of nodes decreases from 1000 to 866 as time increases from 1 to 5.
- **HideNet**: at each time step a community of an initial network with 1000 nodes and 7 communities dissolves, and the number of nodes also varies over time.

To analyse these data we use the cosine or normalized linear kernel defined as $\Omega_{ij} = \frac{x_i^T x_j}{||x_i||||x_j||}$, which is parameter-free. So at each time step, when performing the model selection, we only have to detect the optimal number of clusters $k$ and

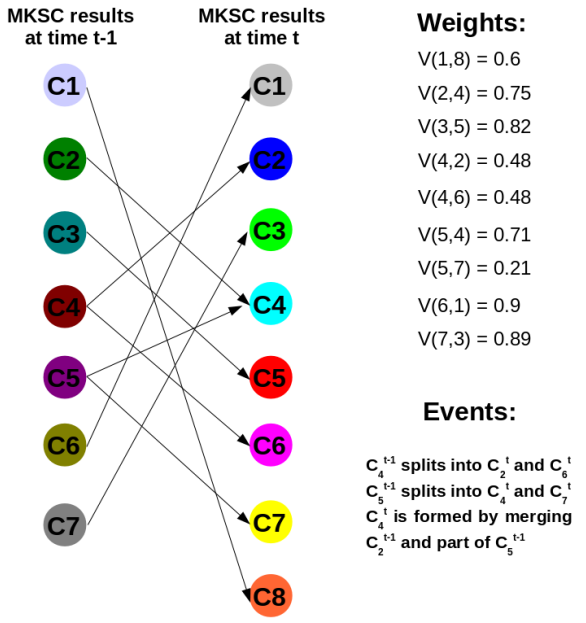



Fig. 1: *Illustrative example of the cluster matching procedure*. Since the labeling at each time step is arbitrary, the clusters found by MKSC at successive time stamps have to be matched to keep track of their evolution. In this specific case, for instance, it is clear that cluster 3 at time $t$ should be labeled as cluster 7, cluster 5 as cluster 3 and so on.

tune the smoothness parameter $\nu$.

The real-world datasets can be described as follows:

- **RealityNet**: this dataset records the cellphone activity for students and staff from two different labs in MIT [22]. It is constructed on users whose cellphones periodically scan for nearby phones over Bluetooth at five minute intervals. The similarity between two users is related to the number of intervals where they were in physical proximity. Each graph snapshot is a weighted network corresponding to 1 week activity, and a total of 46 snapshots[6] covering the entire 2005 academic year are present. In total there are 94 nodes, but not all the nodes are present in every snapshot. The smallest network comprises 21 people and the largest has 88 nodes.
- **NASDAQ**: this time-evolving dataset consists of the daily prices of stocks listed in the NASDAQ stock exchange in 2008 [23]. Each data point is a 15 dimensional vector where each coordinate is the difference between the opening prices at time $t+1$ and at time $t$, and it is normalized to have zero mean and unitary standard deviation. Basically, this feature vector corresponds to the normalized derivatives of the opening prices over a 15 day period, as in [24]. A total of 16 snapshots is present, and the number of data points varies from 2049 to 2095.

[6]However, we noticed that in some snapshots a clear community structure is absent (extremely low values of Modularity). So in the experimental Section we only illustrates the results related to 32 snapshots.

**Algorithm 2:** Clustering evolving data

**Data**: Training sets $\mathcal{D} = \{x_i\}_{i=1}^{N_{\text{Tr}}}$ and $\mathcal{D}_{\text{old}} = \{x_i^{\text{old}}\}_{i=1}^{N_{\text{Tr}}}$, test sets $\mathcal{D}^{\text{test}} = \{x_m^{\text{test}}\}_{m=1}^{N_{\text{test}}}$ and $\mathcal{D}_{\text{old}}^{\text{test}} = \{x_m^{\text{test,old}}\}_{m=1}^{N_{\text{test}}}$, $\alpha_{t-1}^{(l)}, \ldots, \alpha_{t-M}^{(l)}$ (the $\alpha^{(l)}$ calculated for the previous $M$ snapshots), positive definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ such that $K(x_i, x_j) \rightarrow 0$ if $x_i$ and $x_j$ belong to different clusters, kernel parameters (if any), number of clusters $k$, regularization constants $\gamma_t$ and $\nu_t$ found using the tuning algorithm.

**Result**: Clusters $\{\mathcal{C}_1^t, \ldots, \mathcal{C}_p^t\}$, cluster codeset $\mathcal{CB} = \{c_p\}_{p=1}^k$, $c_p \in \{-1, 1\}^{k-1}$.

**if** *t*==*1* **then**
  Initialization by using kernel spectral clustering (KSC [14]).
**else**
  **For every snapshot from $t-1$ to $t-M$ rearrange the data matrices and the solution vectors $\alpha^{(l)}$ as explained in Section III-A.**
  Compute the solution vectors $\alpha_t^{(l)}$, $l = 1, \ldots, k-1$, related to the linear systems described by eq. (2)
  Binarize the solution vectors: $\text{sign}(\alpha_{t,i}^{(l)})$, $i = 1, \ldots, N_{\text{Tr}}$, $l = 1, \ldots, k-1$, and let $\text{sign}(\alpha_{t,i}) \in \{-1, 1\}^{k-1}$ be the encoding vector for the training data point $x_i$.
  Count the occurrences of the different encodings and find the $k$ encodings with most occurrences. Let the codeset be formed by these $k$ encodings: $\mathcal{CB} = \{c_p\}_{p=1}^k$, $c_p \in \{-1, 1\}^{k-1}$.
  $\forall i$, assign $x_i$ to $C_{p^*}$ 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 $\text{sign}(e_{t,m}^{(l)})$, $m = 1, \ldots, N_{\text{test}}$, $l = 1, \ldots, k-1$ and let $\text{sign}(e_{t,m}) \in \{-1, 1\}^{k-1}$ be the encoding vector of $x_m^{\text{test}}$.
  $\forall m$, assign $x_m^{\text{test}}$ to $C_{p^*}^t$ using an ECOC decoding scheme, i.e. $p^* = \text{argmin}_p d_H(\text{sign}(e_m), c_p)$.
  **Match the actual clusters $\{\mathcal{C}_1^t, \ldots, \mathcal{C}_p^t\}$ with the previous partition $\{\mathcal{C}_1^{t-1}, \ldots, \mathcal{C}_q^{t-1}\}$ using the tracking scheme described in Section III-B and summarized in algorithm 1.**
**end**

To cluster the first dataset we use the cosine kernel as for the computer generated networks described earlier. Concerning the *NASDAQ* data, since each feature vector $x_i$ represents a 15 days time-series we utilize the RBF kernel with the correlation distance [25]. This kernel has been recently used for time-series clustering in [26] and is defined as: $K(x_i, x_j) = \exp(-||x_i - x_j||_{\text{cd}}^2/\sigma^2)$, where $||x_i - x_j||_{\text{cd}} = \sqrt{\frac{1}{2}(1 - R_{ij})}$, with $R_{ij}$ indicating the Pearson correlation coefficient between time-series $x_i$ and $x_j$.

## V. Experiments

In this Section an exhaustive study about the ability of the MKSC algorithm to perform dynamic clustering is performed. First we discuss the model selection issue: different criteria are contrasted and the outcomes are analysed. Then the clustering results are evaluated according to a number of cluster quality measures and the MKSC method is compared with the AFFECT algorithm [7] and the ESC [5] technique. Finally, two types of visualization of the clusters evolution over time are

presented[7].

### A. Model selection

Here the AMS criterion [17], the mean Silhouette value [18] and the Modularity criterion [20], [27] are compared for tuning the number of clusters $k$ and the smoothness parameter $\nu$ for the network data. The same procedure is applied when selecting $k$, $\nu$ and the $\sigma$ of the RBF kernel for the *NASDAQ* data.

Concerning the selection of the optimal number of clusters $k$, due to space constraints we only depict the results related to the real-life data in Figure 2. In case of the *RealityNet* only AMS mostly selects $k = 2$ along all the time period, in agreement with the ground truth suggested in [22] and in [28]. In the cited works it has been proposed to group students and staff at MIT according to their belonging to the Sloan business school or being co-workers in the same building. Regarding the *NASDAQ* dataset, Silhouette and AMS are contrasted. The former mainly suggests $k = 2$, the latter $k = 3$ for almost all the 16 snapshots. Thus it seems that the grouping of the feature vectors in terms of the sectors of the stocks (which are 12 in total) is not valid in this case. Probably the model selection criteria are distinguishing between small and big capitalization stocks and small, medium and big cap respectively. In case of the synthetic networks we observed that in general Modularity, AMS or both are able to suggest the right number of communities.

Regarding the smoothness parameter $\nu$, in case of the synthetic networks AMS and Modularity produce the same result only for *HideNet*. Like before, we show only the results related to the real-life datasets, which are plotted in Figure 3. In case of *RealityNet* the regularization constant has some small peaks around important dates like beginning of fall and winter term and end of winter term. Concerning the *NASDAQ* dataset, $\nu$ has a peak around $t = 13$, which corresponds to the market crash happened at the end of September 2008. This behaviour can be explained by considering that, when there is a significant change in the data, the memory effect should activate in order to smooth the clustering results. Moreover, also the RBF kernel parameter $\sigma$ is able to detect this change: it has a sudden drop at $t = 13$. To summarize, it seems that both $\nu$ and $\sigma$ are behaving as a kind of change indicator measures.

### B. Evaluation of the results

In this Section the clustering results are evaluated according to the Adjusted Rand Index (ARI [29]) when the true memberships are available, the smoothed Conductance (Cond_sm) and the smoothed Silhouette (SIL_sm).

The smoothed version of standard cluster quality measures like Conductance [30], Modularity [19] etc. have been introduced in [11]. The new measures are the weighted sum of the snapshot quality and the temporal quality. The former only measures the quality of the current clustering with respect to the current data, while the latter measures the temporal smoothness in terms of the ability of the actual model to cluster the historic data. For a given cluster quality criterion CQ, we can define its smoothed version as:

$$\mathrm{CQ}_{\mathrm{sm}}(X_t, \mathcal{G}_t) = \eta \mathrm{CQ}(X_t, \mathcal{G}_t) + (1-\eta)\mathrm{CQ}(X_t, \mathcal{G}_{t-1}).$$

The symbol $X_t$ indicates the partition found at time $t$. With $\eta$ we denote a user-defined parameter which takes values in the range $[0, 1]$ and reflects the emphasis given to the snapshot quality and the temporal smoothness, respectively.

The MKSC algorithm is compared with Adaptive Evolutionary Clustering (AFFECT [7]) in case of synthetic data

[7]In this case, due to space limits, we only consider the *MergesplitNet* data.

| **MergeplitNet** | | |
|---|---|---|
| **MEASURE** | **MKSC** | **AFFECT [7]** |
| **ARI** | $\mathbf{0.90 \pm 0.03}$ (MOD) | $0.73 \pm 0.01$ (MOD) |
| **Cond_sm** | $0.0112 \pm 0.0001$ (AMS) | $\mathbf{0.0038 \pm 0.0005}$ (SIL) |
| **BirthdeathNet** | | |
| **MEASURE** | **MKSC** | **AFFECT [7]** |
| **ARI** | $\mathbf{0.80 \pm 0.02}$ (MOD) | $0.76 \pm 0.03$ (MOD) |
| **Cond_sm** | $\mathbf{0.036 \pm 0.003}$ (AMS) | $0.052 \pm 0.002$ (MOD) |
| **BirthdeathNet** | | |
| **MEASURE** | **MKSC** | **AFFECT [7]** |
| **ARI** | $\mathbf{0.97 \pm 0.01}$ (AMS, MOD) | $0.85 \pm 0.03$ (MOD) |
| **Cond_sm** | $0.011 \pm 0.001$ (AMS, MOD) | $\mathbf{0.005 \pm 0.001}$ (SIL) |

TABLE I: *Clustering results synthetic datasets*. Both ARI and Cond_sm values represent an average over time, i.e. 5 snapshots. Moreover, for each snapshot the smoothed Conductance is obtained by taking the mean over 5 possible values of the parameter $\eta$ in the range $[0, 1]$. Between parenthesis we indicate with which model selection criterion the related result has been obtained. Although the model selection problem is out of the scope of [7], the authors provide two possible way of tuning the number of clusters, using Silhouette (SIL) or Modularity (MOD).

| **RealityNet** | | | |
|---|---|---|---|
| **MEASURE** | **MKSC** | **AFFECT [7]** | **ESC [5]** |
| **ARI** | $\mathbf{0.861 \pm 0.051}$ (AMS) | $0.763 \pm 0.001$ | – |
| **RI** | $\mathbf{0.943 \pm 0.040}$ (AMS) | 0.893 | 0.861 |
| **Cond_sm** | $\mathbf{0.0035 \pm 0.0001}$ (AMS) | $0.0048 \pm 0.0001$ | – |
| **NASDAQ** | | | |
| **MEASURE** | **MKSC** | **AFFECT [7]** | **ESC [5]** |
| **ARI** | $0.034 \pm 0.001$ (AMS) | $\mathbf{0.058 \pm 0.001}$ | – |
| **RI** | $0.745 \pm 0.001$ (AMS) | $\mathbf{0.808 \pm 0.000}$ | $0.806 \pm 0.000$ |
| **SIL_sm** | $\mathbf{0.21 \pm 0.01}$ (SIL) | $0.08 \pm 0.02$ | – |

TABLE II: *Clustering results real-life datasets*. We compare MKSC with AFFECT and ESC by reporting the results shown in Tables 5 and 6 of [7]. Regarding ESC, the mean between the best results related to the PCQ and PCM frameworks are considered. Moreover, for AFFECT and ESC the number of clusters has been fixed to $k = 2$ and $k = 12$ for *RealityNet* and *NASDAQ* respectively, for the aforementioned comparison purposes. On the other hand, $k$ is fine tuned in case of MKSC. The best performer in case of the *RealityNet* is MKSC. Concerning the NASDAQ data, AFFECT gives the best results. However, it must be noticed that the values of ARI are low for all the methods, indicating that the grouping based on the 12 industrial sectors, acting as the ground truth, is not appropriate.

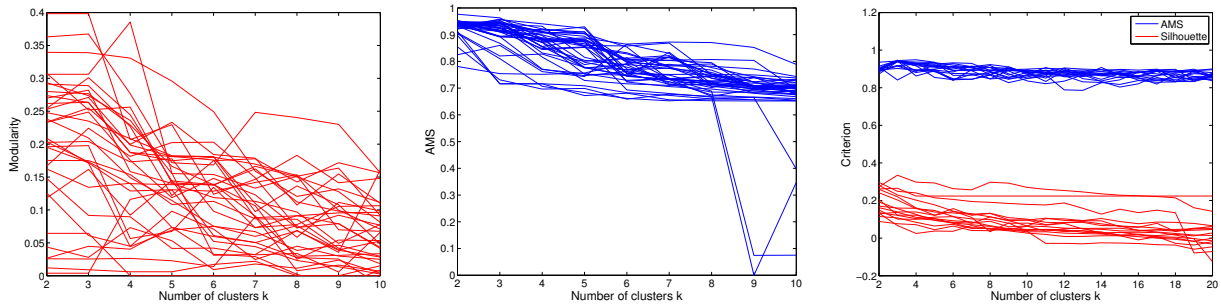



Fig. 2: *Real-life datasets: selection of the number of clusters*. Left and center: *RealityNet*. For this network it has been suggested to consider $k = 2$ as the ground-truth for the entire period of 46 weeks [22]. This is due to the fact that the people representing the nodes of the network belong to 2 different departments at MIT. However we have noticed that the network does not have a clear community structure in every snapshot (for instance in some snapshots the maximum value of the Modularity quality function for $k = 2$ approaches zero). Moreover, both Modularity (left) and AMS (right) do not always select $k = 2$: the former mainly detects $k = 4, 5, 6$, and the latter $k = 2$ but also $k = 3$. Right: NASDAQ. Regarding this dataset, the mean Silhouette value (red) and the AMS criterion (blue) are compared. The former selects mainly $k = 2$ for all the 16 snapshots, while the latter mostly $k = 3$. Probably the criteria are suggesting to group the 15 days time-series of stocks into small and big cap and small, medium and big cap respectively.

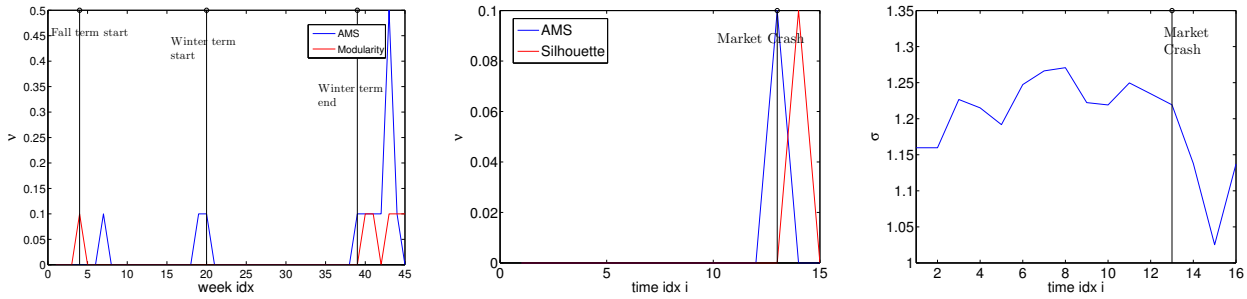



Fig. 3: *Real-life datasets: selection of the smoothness parameter* $\nu$. Left: *RealityNet*. The regularization constant $\nu$ selected by AMS (blue) and Modularity (red). In both cases some peaks are present around important dates which are labeled in the plot. Center and right: *NASDAQ*. The regularization constant $\nu$ selected by AMS (blue) and Silhouette (red). Interestingly, in both cases there is a peak around $t = 13$, which corresponds to the market crash occurred in late September 2008. *Right*: optimal bandwidth of the RBF kernel (AMS and Silhouette produce an identical outcome). Also in this case the market crash is detected ($\sigma$ has a sudden drop around $t = 13$).

and AFFECT and ESC (Evolutionary Spectral Clustering [5]) for *RealityNet* and the *NASDAQ* data. In most of the datasets MKSC produces clustering results closer to the ground truth memberships (higher ARI), as it is shown in Tables I and II. Regarding the *NASDAQ* dataset, AFFECT performs the best, but the ARI values are low for all the algorithms, suggesting that the chosen ground truth memberships are not suitable.

### C. Visualizing the clusters evolution

In this Section we show the results obtained by our tracking mechanism on the *MergesplitNet* as an example. Initially, we use the first 3 dual solution vectors of problem (2), i.e. $\alpha^{(1)}$, $\alpha^{(2)}$, $\alpha^{(3)}$, to visualize the clusters evolution in 3D. In order to explicitly show the growth and shrinkage events we plot the clusters as spheres, centered around the mean of all the points in that cluster. The radius is equivalent to the fraction of points belonging to that cluster at that time stamp. Each sphere is given a unique colour at time stamp $t = 1$. As the clusters grow or shrink the size of the sphere changes. In case of a split the colour and label of that cluster is transferred to all the clusters obtained as a result of the split. In case of a merge we assign the average colour of the clusters which merge together at time $t$ to the new cluster at time $t + 1$. In case of birth of a new cluster we allocate it a new colour and all the nodes which have disappeared at time interval $t$ are depicted as a blue-coloured sphere centered at the origin (dump). Another possible visualization consists of depicting the adjacency matrices representing the networks constructed in the tracking mechanism.

The proposed visualization tools are illustrated in Figure 4. Due to space limitations, only the first three time stamps are shown. However, we observed a major change in the data at time $t = T_4$, when the size of cluster $C5$ increased at the expense of the remaining clusters. In fact, at this time stamp,

the tuning scheme selected $\nu = 0.1$, while in the other time steps $\nu = 0$. Thus, the memory effect got activated to smooth the clustering results.

Finally, we have observed that for all the synthetic networks (although not perfectly) MKSC is able to grasp the main events occurring at each time step. Regarding the real-life datasets, thanks to the proposed visualization tools, it is possible for the user to have an idea of the clusters evolution discovered by the MKSC model.

### D. Computational complexity

The time required to train the MKSC model using a training set of $N_{\text{Tr}}$ data points is $O(N_{\text{Tr}}^3)$, which is needed to solve problem (2). Considering as test set the entire dataset of $N$ points, and supposing that $N_{\text{Tr}} \ll N$, the main contribution to the runtime of the algorithm is given by the out-of-sample extension performed via eq. (4). So the complexity is $O(N_{\text{Tr}}N)$. Moreover, in case the matrices $\Omega_t^{\text{test}}, \ldots, \Omega_{t-M}^{\text{test}}$ cannot fit the memory we can divide the test set into blocks and perform the testing operations iteratively on a single computer or in parallel in a distributed environment, as shown in [31], [32] for kernel spectral clustering (KSC).

## VI. Conclusions

In this paper we have discussed the problem of clustering data over time. Using some synthetic and real-life data, we have shown that our technique MKSC was able to handle a varying number of data points and track the cluster evolution. Also the model selection issue has been investigated. Moreover, we discovered how one of the regularization constants of the MKSC model, the smoothness parameter $\nu$, can be used as a change indicator measure. In fact, in case of the *RealityNet* $\nu$ showed some small peaks around important dates like fall and winter term begin and winter term end. Regarding the *NASDAQ* data, $\nu$ was capable to detect the market crash of late September 2008. Also a comparison with two state-of-the-art techniques, namely AFFECT and ESC, evidenced a very competitive performance. Finally, we proposed two possible visualizations of the cluster dynamics. Future work should be directed to implement a more efficient version of the algorithm and then perform experiments on large datasets. The possibility to scale the MKSC algorithm to big data, together with a systematic model selection procedure and the generalization ability, represent the main advantages of our method compared to the other aforementioned technique.

## 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 paper 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). Johan Suykens is a professor at the KU Leuven, Belgium. The scientific responsibility is assumed by its authors.

## References

[1] C. Tantipathananandh, T. Y. Berger-Wolf, and D. Kempe, "A framework for community identification in dynamic social networks." in *KDD '07*. ACM, 2007, pp. 717–726.

[2] Y. Li, J. Han, and J. Yang, "Clustering moving objects," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '04, 2004, pp. 617–622.

[3] R. Langone, C. Alzate, B. De Ketelaere, and J. A. K. Suykens, "Kernel spectral clustering for predicting maintenance of industrial machines," in *IEEE Symposium Series on Computational Intelligence (SSCI) 2013*, 2013, pp. 39–45.

[4] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 554–560.

[5] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness." in *KDD '07*, 2007, pp. 153–162.

[6] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, "Analyzing communities and their evolutions in dynamic social networks," *ACM Trans. Knowl. Discov. Data*, vol. 3, no. 2, 2009.

[7] K. S. Xu, M. Kliger, and A. O. Hero III, "Adaptive evolutionary clustering," *Data Mining and Knowledge Discovery*, pp. 1–33, 2013.

[8] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[9] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[10] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 849–856.

[11] R. Langone, C. Alzate, and J. A. K. Suykens, "Kernel spectral clustering with memory effect," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 10, pp. 2588 – 2606, 2013.

[12] R. Langone and J. A. K. Suykens, "Community detection using kernel spectral clustering with memory," *Journal of Physics: Conference Series*, vol. 410, no. 1, p. 012100, 2013.

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

[14] C. Alzate and J. A. K. Suykens, "Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 335–347, February 2010.

[15] D. Greene, D. Doyle, and P. Cunningham, "Tracking the evolution of communities in dynamic social networks," in *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, ser. ASONAM '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 176–183.

[16] P. Brodka, S. Saganowski, and P. Kazienko, "Ged: the method for group evolution discovery in social networks," *Social Network Analysis and Mining*, vol. 3, no. 1, pp. 1–14, 2013.

[17] R. Langone, R. Mall, and J. A. K. Suykens, "Soft kernel spectral clustering." in *Proc. of the International Joint Conference on Neural Networks (IJCNN 2013)*, 2013, pp. 1028 – 1035.

[18] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, no. 1, pp. 53–65, 1987.

[19] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Natl. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, 2006.

[20] R. Langone, C. Alzate, and J. A. K. Suykens, "Modularity-based model selection for kernel spectral clustering," in *Proc. of the International Joint Conference on Neural Networks (IJCNN 2011)*, 2011, pp. 1849–1856.

[21] R. Mall, R. Langone, and J. A. K. Suykens, "FURS: Fast and unique representative subset selection retaining large scale community structure," *Social Network Analysis and Mining*, vol. 3, no. 4, pp. 1–21, 2013.

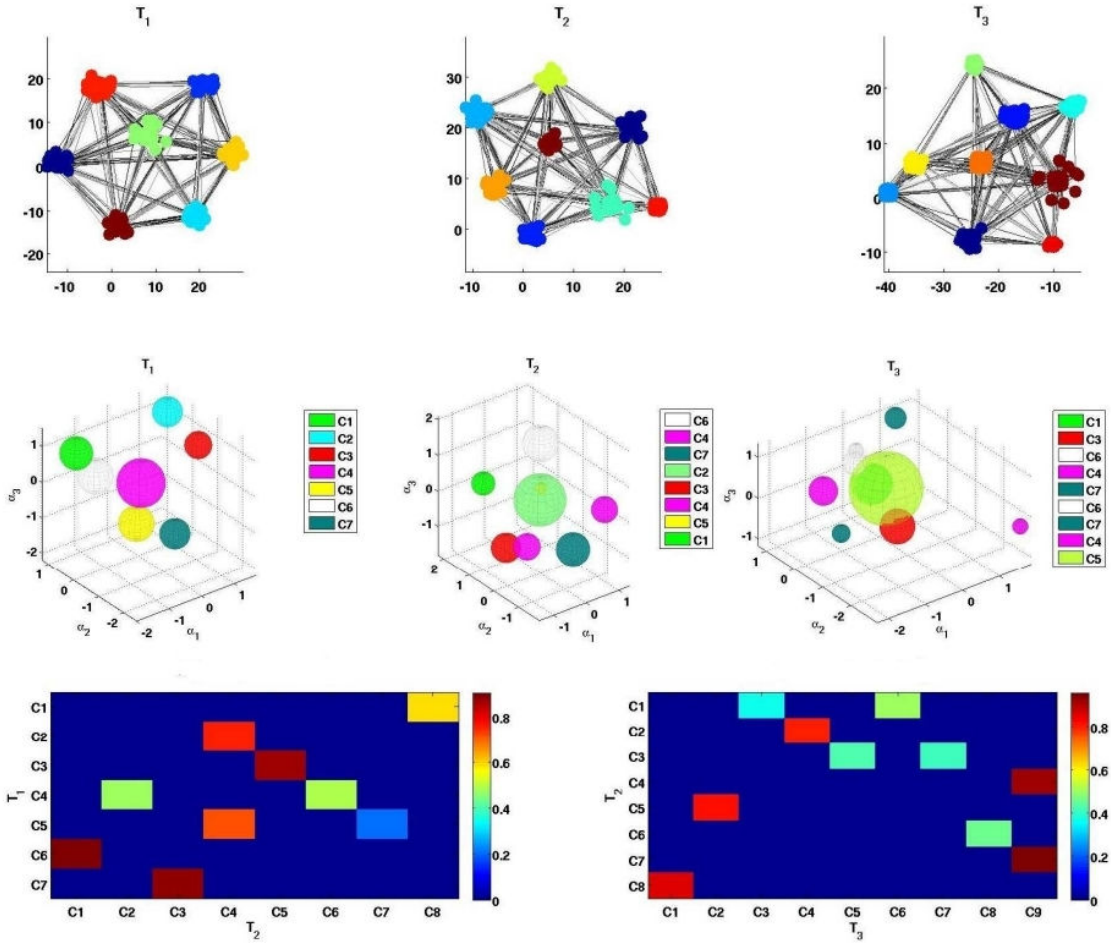



Fig. 4: *MergesplitNet: communities found over time by the MKSC algorithm.* (Top) Standard visualization in terms of nodes and edges (Center) Proposed 3D visualization (Bottom) Sequence of directed weighted networks mapping the clusters at two consecutive time stamps, as explained in Section III-B. **Explanation**: the *MergesplitNet* dataset has 7 clusters at time stamp $T_1$. At time stamp $T_2$ cluster $C4$ splits into 2 clusters, major part of cluster $C5$ merges with cluster $C2$. At time stamp $T_3$ cluster $C6$, cluster $C7$ splits into 2 clusters and clusters $C2$ and $C5$ merge to cluster $C5$. At time stamp $T_4$ clusters $C4$ and $C7$ further split to have 3 clusters each. Cluster $C3$ combines with cluster $C5$. In the final time interval cluster $C5$ splits into 2 clusters.


[22] N. Eagle, A. S. Pentland, and D. Lazer, "Inferring social network structure using mobile phone data," *PNAS*, vol. 106, no. 1, pp. 15 274–15 278, 2009.

[23] "http://www.infochimps.com/datasets/nasdaq-exchange-daily-1970-2010-open-close-high-low-and-volume."

[24] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani, "Mining the stock market (extended abstract): Which measure is best?" in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '00, 2000, pp. 487–496.

[25] T. W. Liao, "Clustering of time series data - a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857 – 1874, 2005.

[26] C. Alzate and M. Sinn, "Improved electricity load forecasting via kernel spectral clustering of smart meters," in *ICDM*, 2013, pp. 943–948.

[27] R. Langone, C. Alzate, and J. A. K. Suykens, "Kernel spectral clustering for community detection in complex networks." in *IJCNN*. IEEE, 2012, pp. 2596–2603.

[28] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu, "Graphscope: parameter-free mining of large time-evolving graphs," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '07. New York, NY, USA: ACM, 2007, pp. 687–696.

[29] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 1, no. 2, pp. 193–218, 1985.

[30] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 631–640.

[31] R. Mall, R. Langone, and J. A. K. Suykens, "Kernel spectral clustering for big data networks," *Entropy*, vol. 15, no. 5, pp. 1567–1586, 2013.

[32] ——, "Self-Tuned Kernel Spectral Clustering for Large Scale Networks," in *IEEE International Conference on Big Data (2013)*, 2013, pp. 385–393.