Clustering by Synchronization

Christian Böhm
University of Munich
Munich, Germany
boehm@dbs.ifi.lmu.de

Claudia Plant
Florida State University
Tallahassee, FL, USA
cplant@fsu.edu

Junming Shao
University of Munich
Munich, Germany
shao@dbs.ifi.lmu.de

Qinli Yang
University of Edinburgh
Edinburgh, UK
q.yang@ed.ac.uk

ABSTRACT

Synchronization is a powerful basic concept in nature regulating a large variety of complex processes ranging from the metabolism in the cell to social behavior in groups of individuals. Therefore, synchronization phenomena have been extensively studied and models robustly capturing the dynamical synchronization process have been proposed, e.g. the Extensive Kuramoto Model. Inspired by the powerful concept of synchronization, we propose *Sync*, a novel approach to clustering. The basic idea is to view each data object as a phase oscillator and simulate the interaction behavior of the objects over time. As time evolves, similar objects naturally synchronize together and form distinct clusters. Inherited from synchronization, *Sync* has several desirable properties: The clusters revealed by dynamic synchronization truly reflect the intrinsic structure of the data set, *Sync* does not rely on any distribution assumption and allows detecting clusters of arbitrary number, shape and size. Moreover, the concept of synchronization allows natural outlier handling, since outliers do not synchronize with cluster objects. For fully automatic clustering, we propose to combine *Sync* with the Minimum Description Length principle. Extensive experiments on synthetic and real world data demonstrate the effectiveness and efficiency of our approach.

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Data mining

General Terms

Algorithms, Design, Reliability

Keywords

Synchronization, Clustering, Kuramoto Model

KDD'10, July 25–28, 2010, Washington, DC, USA.

1. INTRODUCTION

Clustering is essential for knowledge discovery in a large variety of applications. During the last decades, clustering has therefore attracted a huge volume of attention, with multiple books, e.g. [18], surveys, e.g. [23] and research papers, e.g. [3, 4, 11, 31] to mention a few. Many approaches have been proposed to address the clustering problem from different points of view, and each cluster notion comes with specific advantages and drawbacks. As an example, consider the wide-spread Expectation Maximization (EM) algorithm [11]. The result of EM, a mixture model of multivariate Gaussians, is very useful for interpretation in many applications. However, EM only yields good results if the data at least approximately follows the model assumption, i.e. consists of Gaussian clusters. Moreover, the number of clusters needs to be specified as an input parameter and the result of EM is very sensitive w.r.t. outliers.

In this paper, we consider clustering from a novel different point of view: *synchronization*. Synchronization is the phenomenon that a group of events spontaneously come into co-occurrence with a common rhythm, despite of the differences between individual rhythms of the events. Synchronous behavior has attracted a large volume of interest in physics, biology, ecology, sociology, communication and other fields of science and technology. It is known that synchrony is rooted in human life from the metabolic processes in our cells to the highest cognitive tasks we perform as a group of individuals [5]. We will demonstrate that clustering by synchronization has many attractive benefits, but let us first illustrate the basic idea.

1.1 Basic Idea

Synchronization phenomena are prevalent in every day life, as an example consider opinion formation. In the beginning, each person has its own view about a certain problem. After *interaction* by conversation or discussion, some people with similar background, such as education or hobby, will easily group together and form a common opinion. As time evolves, in the phase of *local synchronization*, groups with diverse opinions will be formed. After further conversation with different groups, all people may finally achieve a uniform opinion (*global synchronization*).

Inspired by these synchronization phenomena, we exploit the synchronization dynamics to obtain a natural clustering of a given data set. The key idea is to regard each data object as a phase oscillator. Each object interacts dynamically

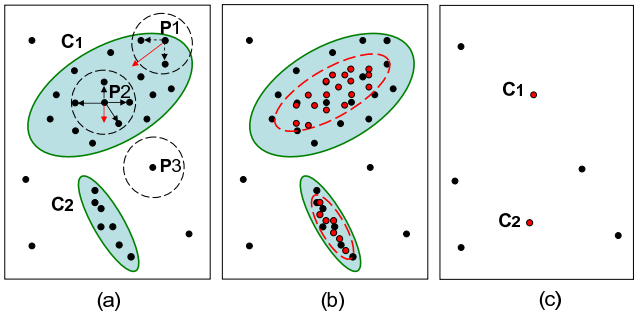


Figure 1: Clustering by synchronization. (a) The initial state of the objects: the back arrows indicate the mutual interaction and red arrows indicate the direction of movement for some sample objects. (b) The comparison of objects states before and after one time step: black points represent the initial state, red points indicate the new state. (c) The final state of objects in local synchronization: synchronized clusters and outliers.

with similar objects. We will elaborate a formal description of the dynamical object interaction patterns by a reformulated Kuramoto model in Section 3.2 and now provide the intuition of clustering by synchronization. The process of the dynamical clustering involves the following stages: First, starting from initial conditions, each object runs independently at its own phase. As time evolves, those objects with highest similarity will synchronize first. Then, in a sequential process, more and more objects synchronize together and clusters are produced driven by the intrinsic structure of the data set. Finally, the whole population is split into several stable distinct synchronized clusters. Outliers are effectively detected due to their different dynamic behavior hardly synchronizing with any of the cluster objects.

Figure 1 displays 3 snapshots of the simulated dynamical movement. For simplicity, 2-dimensional data are displayed. Consider the 2 objects ($P1$,$P2$, $P3$), where $P1$ and $P2$ are cluster objects and $P3$ is an outlier. For border object $P1$, its neighborhood contains less objects and all neighbors are located towards the inside of the cluster. Therefore, the border object is driven towards the inside of the cluster through the non-linear dynamical interaction with its neighbors (Figure 1(a)). The larger the distance between an object and its neighbors, the stronger is the interaction imposed, which implies that neighbors with larger distances have a stronger impact on the object. For the center objects, e.g. $P2$, the neighborhood contains more objects, however, all these neighbors locate around the object, causing a relatively slow movement of $P2$ in comparison to the border objects, such as $P1$. The movement of objects depends on the structure of their overall neighborhood, which implies that the objects will move towards the main direction of their neighborhood. Through the non-linear interaction, all objects with similar attributes finally synchronize together. In contrast, outlier objects remain isolated from all other objects, e.g. $P3$. Figure 1(b) displays the old and the new positions of the objects for comparison. The objects with high similarity gradually synchronize together through mutual coupling. The initial points (black color) are replaced by the red points after one time stamp. Figure 1(c) depicts the final states of the objects. The data set is split into clusters and outliers, where cluster objects are synchronized together and have the same phase while outlier objects remain isolated over time.

1.2 Contributions

The major benefits of our algorithm *Sync* for Clustering by **Sync**hronization, can be summarized as follows:

1. *Clusters detected by SynC truly reflect the intrinsic data structure.* Rooted in the concept of synchronization, *Sync* allows detecting clusters of arbitrary number, shape, size and object density without requiring any distribution assumptions.

2. *Sync is a powerful technique for clustering and outlier detection.* Based on the different dynamical behavior of objects during the process of synchronization, outliers are naturally and effectively detected.

3. *Sync is robust against parameter settings and fully automatic by the combination with Minimum Description Length.* Relying on synchronization, already the basic version of our algorithm is relatively robust against parameter settings in comparison to other clustering paradigms. For automatic clustering, we combine *Sync* with the Minimum Description Length principle.

The remainder of this paper is organized as follows: In the following section, we briefly survey related work. Section 3 presents our algorithm in detail. Section 4 contains an extensive experimental evaluation and Section 5 concludes the paper.

2. RELATED WORK

During the past several decades, many algorithms were proposed for clustering, such as EM [11], CURE [16], CLIQUE [3], BIRCH [31], CLARANS [25], DBSCAN [13], etc. Due to space limitations, we can only provide a very brief survey on some important major research directions. In addition, we briefly introduce related work on synchronization phenomena.

PDF-based Clustering. The key idea of these methods is to detect clusters by using a model of probability density functions (PDFs) to describe the data structure. The most fundamental techniques in this line are K-means [22] and EM [11]. These methods are suitable to detect spherically Gaussian clusters and the number of clusters K needs to be specified by the user. The algorithm X-means [26] extends K-means by a technique for automatically detecting K founded on information theory. The algorithm G-means [17] additionally provides to detect non-spherical Gaussian clusters. Another information-theoretic method, RIC [8], has been designed as a postprocessing step to improve an initial clustering of an arbitrary conventional clustering algorithm. The cluster model comprises a rotation matrix determined by PCA and a PDF assigned to each coordinate selected from a set of predefined PDFs. Clusters with similar characteristics are finally merged. However, the result strongly depends on the quality of the initial clustering and the cluster model is limited to linear attribute correlations and a predefined set of PDFs. Recently, OCI [9] is proposed to detect clusters using Independent Components. This algorithm uses the Exponential Power Distribution as cluster model and applies the Independent Component Analysis for

determining the main directions inside a cluster as well as for finding split planes in a top-down clustering approach. All methods in this category tend to fail if the data distribution does not correspond to the cluster model. An alternative largely avoiding restricting assumptions on the data distribution is the idea of Parzen Windows. Local information on the object density is collected by histograms over local windows in the feature space or by local kernel functions, e.g. Gaussians. Thereby, at a global level, arbitrary data distributions can be captured. The algorithm MeanShift [10] combines this idea with clustering: During the run of the algorithm, the windows move towards the direction of the highest object density until convergence. However, the window size needs to be suitably selected and our experiments demonstrate that this algorithm tends to degrade in performance in the presence of outliers and noise points.

Density-Based and Spectral Clustering. In density-based clustering, clusters are regarded as regions of high object density in the data space which are separated by regions of low object density (noise). The algorithm DBSCAN [13] formalizes this idea using two parameters : the neighborhood of a given radius ϵ has to contain at least a minimum number of objects ($MinPts$). Arbitrarily shaped clusters can be easily detected by DBSCAN, however the choice of a suitable settings of ϵ and $MinPts$ is often difficult, especially since the parameters are correlated. Conceptually closely related, spectral clustering refers to a class of techniques which rely on the Eigenstructure of a similarity matrix to partition objects into disjoint clusters. These algorithms, e.g. [24, 7] detect arbitrarily shaped clusters by considering the clustering problem from a graph-theoretic perspective. A clustering is obtained by removing the weakest edges between highly connected subgraphs, which is formally defined by the normalized cut or similar objective functions. Similar to K-means, the problem of these approaches is to choose a suitable number of clusters and they are sensitive w.r.t. outliers. The approach [7] partially overcomes these problems by proposing a new cost function for spectral clustering based on the error between a given graph partitioning and a clustering result. However, this approach requires sample data with a known partitioning which is not available in most cases. The approach of Affinity Propagation [14] is closely related to spectral and density-based clustering. Each data point is regarded as a node in a network. The algorithm performs clustering by letting the data points exchange messages along the edges of the networks. By message passing, certain data points emerge as so-called exemplars (cluster representatives) and the other data points establish their cluster membership. As input parameter for each data point a real value must be specified characterizing the probability that this particular point is selected as an exemplar. The number of clusters is controlled by this parameter but also by the structure of the network.

Kuramoto Model and Synchronization. Synchronization phenomena in large populations of interacting elements are the subject of intense research efforts in physical, biological, chemical, and social systems. The extensive Kuramoto model [20, 21, 1] is one of the most successful approaches to explore synchronization phenomena. Seliger et al. [27] discuss mechanisms of learning and plasticity in networks of phase oscillators through a generalized Kuramoto model. Arenas et al. [6] apply the Kuramoto model for network analysis, and study the relationship between topological scales and dynamic time scales in complex networks. This analysis provides a useful connection between synchronization dynamics, network topology and spectral graph analysis. From bioinformatics, Kim et al.[19] propose a strategy to find groups of genes by analyzing the cell cycle specific gene expression with a modified Kuramoto model. Aeyels et al. [2] introduce a mathematical model for the dynamics of chaos system. They characterize the data structure by a set of inequalities in the parameters of the model and apply it to a system of interconnected water basins.

In summary, previous approaches mainly focus on the synchronization phenomena of a dynamic system from a global perspective. Moreover, to our best knowledge, the synchronization principle and the Kuramoto model have not been investigated in the data mining community. Inspired by ideas from dynamical system analysis, we propose a novel clustering technique based on local synchronization.

3. CLUSTERING BY SYNCHRONIZATION

Synchronization phenomena are prevalent in nature. For example, the effect of synchrony has been described in experiments of people conversation, song or rhythm, or of groups of children interacting to an unconscious beat. In all cases the purpose of the common wave length or rhythm is to strengthen the group bond. The lack of such synchrony can index unconscious tension, when goals cannot be identified nor worked towards because the members are "out of sync". In this section, we introduce *Sync*, to explore clusters by synchronization. We start with an introduction of the Kuramoto Model and then develop a variant suitable for clustering. In Section 3.3 we discuss the algorithm *Sync* in detail.

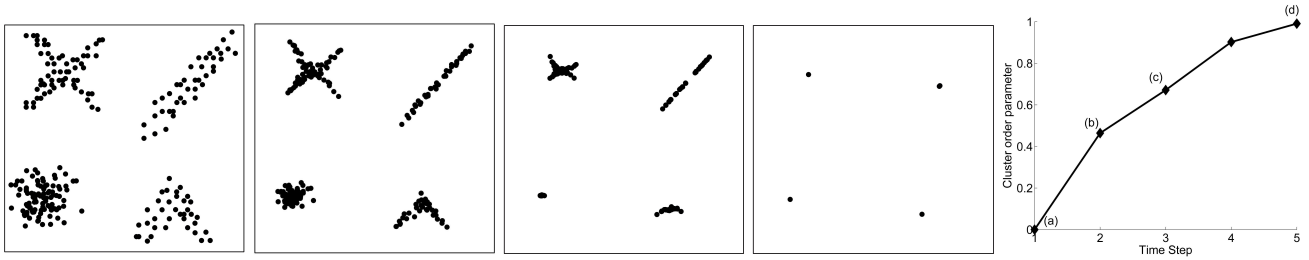
3.1 Kuramoto Model

One of the most successful attempts to understand collective synchronization phenomena was due to Kuramoto [20], [21], who analyzed a model of phase oscillators which run at arbitrary intrinsic frequencies but are coupled through the sine of their phase differences. The *Kuramoto model* (KM) consists of a population of N coupled phase oscillators where the phase of the *i-th* unit, denoted by θ_i, evolves in time according to the following dynamics:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{S}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i), (i = 1, \ldots, N), \qquad (1)$$

where ω_i stands for its natural frequency and S describes the coupling strength between units. There are two major tendencies in this model: the coupling strength S whose effect tends to synchronize the oscillators versus the variance of these natural frequencies, ω_i, the source of disorder which drives them to stay away from each other. When the coupling strength $S = 0$, each oscillator tries to run independently at its own frequency. However, if the coupling is strong enough, all oscillators will freeze into synchrony, which called global synchronization. To characterize the level of synchronization between oscillators, a global order parameter is defined as:

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\theta_j}, \qquad (2)$$



(a) $t = 0$ (b) $t = 1$ (c) $t = 2$ (d) $t = 4$ (e) Cluster order parameter.

Figure 2: The Process of Dynamical Clustering with Time Evolution and Corresponding cluster order parameter: Diagrams (a)-(d) describe the detailed process of dynamical clustering towards synchronization. Due to the space limitation, the states of objects at the time step $t = 0, 1, 2, 4$ are presented. Diagram (e) illustrates the corresponding cluster order parameter during dynamical clustering.

Table 1: Table of symbols.

Symbol	Definition
KM	Kuramoto model.
$\mathcal{D}$	The data set.
x	A data object in $\mathcal{D}$.
x_i	The i-th dimension of the data object x .
N	The number of objects in the data set.
d	The dimensionality of the data set.
K	The number of clusters in the data set.
$x(t)$	The renewal value of object x at time step t.
S	The coupling strength of the KM.
$Nb_\epsilon(x)$	ϵ-neighborhood of the object x.
r_c	Cluster order parameter.
C_i	The i-th cluster of the data set.
$\|C_i\|$	The number of object in the i-th cluster.
$\|p_i\|$	The number of free parameters.

where $0 \leq r(t) \leq 1$, measures the coherence of the oscillator population, and $\psi(t)$ is the average phase.

The Kuramoto model well describes the global synchronization behavior of all coupled phase oscillators, which implies that all the oscillators will be in phase finally through mutual coupling. This situation rarely occurs in real-life systems. Phase locking or partial synchronization is observed more frequently. This is the case a local ensemble of oscillators are synchronized together, where the whole oscillators are split into several clusters of mutually synchronized oscillators.

3.2 Reformulation of the Kuramoto Model for Clustering

To apply KM for clustering, it is necessary to extensively reformulate *Eq.*(1). We need to formally introduce the concept of *local synchronization* of phase oscillators to reflect the intrinsic data structure. Table 1 gives a list of symbols used in the following. To introduce local synchronization, we need to define the notion of ϵ-neighborhood.

Definition 1(ϵ-neighborhood of an object x): The ϵ-neighborhood of object x, which denoted by $Nb_\epsilon(x)$, is defined as:

$$Nb_\epsilon(x) = \{y \in \mathcal{D} | dist(y, x) \leq \epsilon\}, \tag{3}$$

where $dist(y, x)$ is a metric distance function.

Definition 2(Extensive Kuramoto model for Clustering): Let $x \in \mathbb{R}^d$ be an object in the data set $\mathcal{D}$ and x_i be the i-th dimension of the data object x respectively. We regard each object x as a phase oscillator, according to Eq.(1), with an ϵ-neighborhood interaction. The dynamics of each dimension x_i of the object x is governed by:

$$\frac{dx_i}{dt} = \omega_i + \frac{S}{|Nb_\epsilon(x)|} \sum_{y \in Nb_\epsilon(x)} \sin(y_i - x_i). \tag{4}$$

Let $dt = \Delta t$, then:

$$x_i(t+1) = x_i(t) + \Delta t \omega_i + \frac{\Delta t \cdot S}{|Nb_\epsilon(x(t))|} \sum_{y \in Nb_\epsilon(x(t))} \sin(y_i(t) - x_i(t)) \tag{5}$$

It is essential that all objects have a common frequency ω since different individual frequencies would disturb or even prevent cluster formation. However, the particular choice of omega has no effect on the clustering result. The term $\Delta t \cdot \omega_i$ can thus be safely ignored. $\Delta t \cdot S$ is a constant and simply set to 1. Finally the dynamics of each dimension x_i of the object x over time is provided by:

$$x_i(t+1) = x_i(t) + \frac{1}{|Nb_\epsilon(x(t))|} \cdot \sum_{y \in Nb_\epsilon(x(t))} \sin(y_i(t) - x_i(t)). \tag{6}$$

The object x at time step $t = 0$: $x(0) = (x_1(0), ..., x_d(0))$ represents the initial phase of the object (the original location of object x). The $x_i(t+1)$ describes the renewal phase value of i-th dimension of object x at the $t = (0, \ldots, T)$ time evolution.

To characterize the level of synchronization between oscillators during the synchronization process, an order parameter needs be defined. The order parameter of Eq.(2) is suitable for global synchrony. However, it is not effective to identify local dynamic effects. In particular it does not give information about the route to the synchronization in terms of local clusters which is so important to identify compact clusters of synchronized objects. For this reason, instead of considering a global observable, we define a cluster order parameter r_c, measuring the coherence of the local oscillator population.

Definition 3(Cluster Order Parameter): The cluster order parameter r_c characterizing the degree of local synchronization is provided by:

$$r_c = \frac{1}{N} \sum_{i=1}^{N} \sum_{y \in Nb_\epsilon(x)} \mathbf{e}^{-||y-x||}. \tag{7}$$

The value of r_c increases as more neighbors synchronize together with time evolution. The dynamical clustering will terminate when $r_c(t) \to 1$, which indicates the local phase

coherence, also called *local perfect synchronization*. At this moment, all local objects (within in a cluster) have the same phase (location).

3.3 The Sync Algorithm

In this section, we present the *SynC* algorithm based on our reformulated extensive Kuramoto model.

3.3.1 Dynamical Clustering

The basic idea of *Sync* is to regard each object as a non-identical phase oscillator which changes its phase (location) dynamically with time evolution according to *Eq.*(6). The process of the dynamical clustering involves the following steps:

1. At initial time ($t = 0$), without any interaction, all objects in the data set have their own phase, which are represented by different values (feature vectors). Thus, there are N disconnected sets (N clusters).
2. As time evolves, each object interacts with its ϵ- neighborhood, cf. Definition 1. The interaction strength which each neighbor is determined proportional to the similarity (Eq.(6)). Objects with similar attributes synchronize together and form several synchronized clusters gradually following the intrinsic structure of a data set.
3. Finally, the cluster order parameter (Def. 3), which characterizes the level of local synchronization, is used to determine the termination of the dynamical clustering. ($r_{local} \to 1$).

During the dynamical process toward synchronization, all objects change their locations through the interaction with similar objects according to *Eq.*(6). This means that also the neighborhood of each object changes dynamically with time evolution. The traces of all objects are in line with the main direction of the local data structure. Finally, all objects in each cluster synchronize at a certain common phase. Figure 2 (a)-(e) shows the detailed process of dynamical clustering of 2-dimensional points from $t = 0$ to $t = 4$. $t = 0$ indicates the original data set at the initial time. From that moment on, all objects with similar attributes start to synchronize together through the dynamical interaction and finally, all objects in the data set synchronize at 4 different phases after 4 time steps. The level of local synchronization with time evolution is characterized by the cluster order parameter, which indicated in Figure 2 (e). When $r_c \to 1$, all objects in a cluster synchronize together (coherence or phase locking).

3.3.2 Outlier Handling

Depending on their dynamical behavior with time evolution, all objects can be divided into two types: *synchronized objects* (objects in several synchronized clusters) and *non-synchronized objects* (outliers). Most objects synchronize with other objects and form synchronized clusters. Objects which remain isolated all the time are regarded as outliers. In order to illustrate the process of handling outliers, we use two simple one-dimensional data sets: Without outliers $D_1 = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0, 1.1, 1.3, 1.5\}$, and with outliers $D_2 = \{-0.5, 0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0, 1.1, 1.3, 2.0\}$. As displayed in Figure 3, in D_1, all objects synchronize to different clusters and achieve the phase lock. However, in D_2, owing to the different movement dynamics and directions, outlier objects can not synchronize with other objects

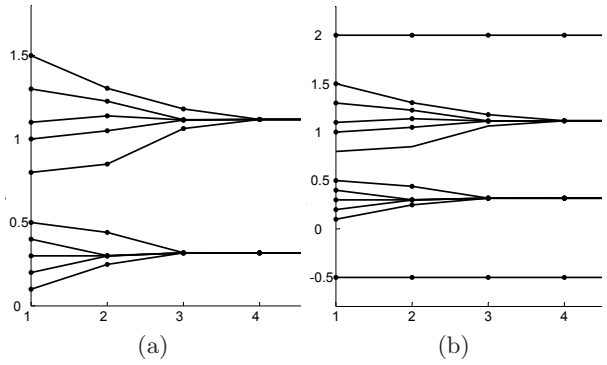


Figure 3: Object dynamics plot displaying the different dynamics of cluster and outlier objects. (a): data set without outliers, (b): data set with outliers.

and keep their original location during the whole time evolution. Moreover, the plots of dynamic object movement are a concise visualization for the intrinsic data structure including cluster points and outliers.

3.3.3 MDL-based Clustering Model Selection

To explore clusters based on synchronization, an appropriate initial size of the neighborhood for mutual interaction needs to be determined. Although our experiments demonstrated that the clustering result is quite robust against the initial neighborhood size ϵ, we propose to combine *Sync* with ideas from Minimum Description Length [15] for fully automatic clustering. More specifically, the MDL principle is applied to compress a set of candidate clustering models M^l, where in our case different models correspond to the clustering results with various ϵ.

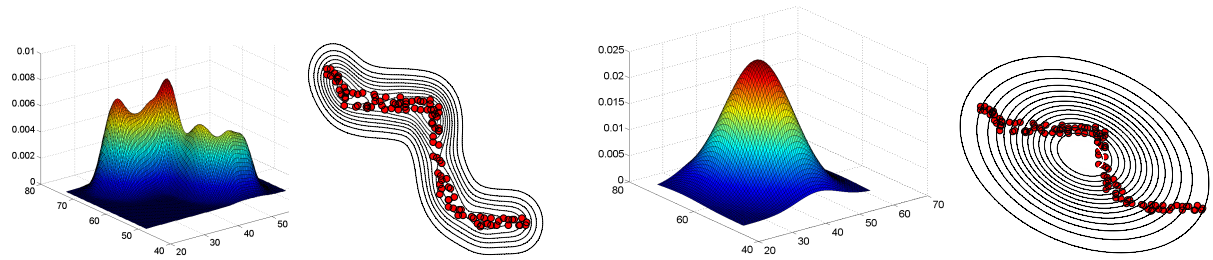
To generate a suitable set of models, we apply the following heuristic: To guarantee a stable interaction of each object, we initiate ϵ with the average value of the k-nearest neighbor distance determined from a sample from the data set for a small k. We increase ϵ stepwise until all objects synchronize in a cluster. A reasonable step size is determined by the difference between average $(k+1)$-nearest neighbor distance and average k-nearest neighbor distance. We recommend to choose k sufficiently small and $k = 3$ nearest neighbors were applied in all experiments. From all candidate models, we select the model resulting in the minimum description length. Pseudocode the *Sync* algorithm is provided in Figure 4. In the following we explain in detail how to compress a candidate model.

Given a data set $\mathcal{D}$ and a clustering model M, the fundamental idea of MDL is to exploit the regularities in the data described by M for effective compression of the data. To avoid overly complex models, the overall description length includes not only the cost for coding the data exploiting the model, denoted by $L(\mathcal{D}|M)$, but also the cost $L(M)$ for coding the model M itself.

Assuming there are K clusters, the coding cost or description length for the data $L(D|M)$ is provided by:

$$L(D|M) = -\sum_{i=1}^{K} \sum_{x \in C_i} \log_2(pdf(x)). \quad (8)$$

where the $pdf(x)$ means the probability of object x in each cluster. To allow for arbitrarily shaped clusters, we propose



(a) Kernel density estimation (density and contour plot). (b) Gaussian estimation (density and contour plot).

Figure 5: Kernel density estimation vs. Gaussian estimation.

```
algorithm M* = Sync(D)
  l = 0, ε^0=NN(k)  //Initialization
  while(no global synchronization)
    [M^l]=DynamicalClustering(D, ε^l);  //Clustering
    if(M^l.clusterSize==1)
      global synchronization = true;
    end
    Compute the coding cost L(D, M^l)
    ε^l = ε^l + Δε;      // Increase ε: Δε = (NN(k+1) - NN(k))
    l = l + 1;
  end for
  M* = argmin_l L(D, M^l)
Return M*;

//Dynamical clustering by synchronization.
[M^l]=DynamicalClustering(D, ε_l)
  while(loopFlag=true)
    for(each object p ∈ D)
      Compute Nb(p);
      Obtain new value of object p using Eq.(6); //Update value
      Compute local order r_p using Eq.(7);
    end for
    Compute local order r_local of all objects; //Local order parameter
    if(r_local > λ) //λ =1-1e-3 in this work
      loopFlag=false;
      C^l = synCluster(D^s); // D^s: New Data by Synchronization
      O^l = \(D, C^l); //Outliers
      M^l = (C^l, O^l);
    end if
  end while
return [M^l];
```

Figure 4: Pseudocode of the *Sync* algorithm.

to estimate $pdf(x)$ by non-parametric kernel density estimation, a widely used technique with numerous applications in machine learning, pattern recognition and computer vision [30].

For coding the cluster model $L(M)$, we need to specify the cluster assignment as well as the model parameters, which is given as:

$$L(M) = \sum_{i=1}^{K} \sum_{j=1}^{|C_i|} \log_2(\frac{N}{|C_i|}) + \sum_{i=1}^{K} \frac{p_i}{2} \log_2(|C_i|). \quad (9)$$

where the first term represents the coding cost for the cluster assignment and the second term is dedicated for coding the parameters. Here p_i is the number of bandwidths for kernel density estimation, which equals the dimensionality of the data set.

Finally, the total coding cost for a clustering model M is:

$$L(D, M) = L(M) + L(D|M). \quad (10)$$

The objective is to find the best cluster model minimizing the total coding cost.

$$M^j = \underset{j}{\operatorname{argmin}}\, L(D, M^j). \quad (11)$$

Kernel Density Estimation. The multivariate kernel density estimation is defined as follows:

$$\hat{f}(x) = \frac{1}{N} \sum_{y \in \mathcal{D}} \frac{1}{h^d} K(\frac{x - y}{h}). \quad (12)$$

where $h = (h_1, \cdots, h_d)^T$ is the bandwidth and the term $K(\cdot)$ is a $d-$dimensional kernel function that is non-negative and integrates to one. As a common choice we use the Gaussian kernel with mean 0 and variance 1. More specifically:

$$\hat{f}(x) = \frac{1}{N} \sum_{y \in \mathcal{D}} \Big(\prod_{i=1}^{d} \frac{1}{h_i} K(\frac{x_i - y_i}{h_i}) \Big), \quad (13)$$

where $K(x) = (2\pi)^{-1/2} \exp(-x^2/2)$. The bandwidth h is selected using an established heuristic which is proven to work well in various applications, even in the case of outliers and bi-modal distributions [28]:
$h_i = 0.9 \cdot N^{-1/(d+4)} \min(\sigma_j, IQR_i/1.34)$, where σ_i is the variance and IQR_i is the interquartile range of dimension i.

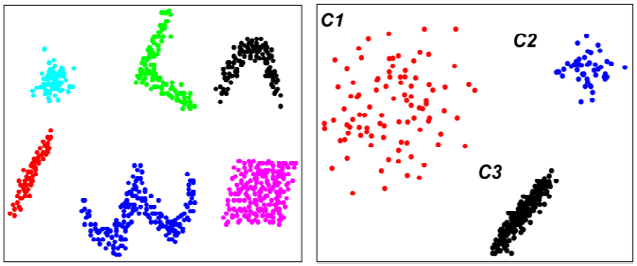
Finally, the probability of each object x is given as:

$$pdf(x) = \frac{\hat{f}(x)}{\sum_{y \in \mathcal{D}} \hat{f}(y)}. \quad (14)$$

Unlike parametric statistics, kernel density estimation allows to robustly estimate the probability for any unknown distribution. Consider for instance the cluster in Figure 5. Kernel density estimation provides an exact and concise representation of the data. In contrast, the Gaussian model is inappropriate for this cluster.

3.4 Runtime Complexity

For each dynamical clustering by synchronization, the runtime complexity with respect to the number of data objects is $O(T \cdot N^2)$, where N is the number of objects and T is the time evolution. In most cases, T is small with $5 \leq T \leq 20$. If there exists an efficient index, the complexity reduces to $O(T \cdot N \log N)$. With clustering model selection based on MDL principle, the final complexity of *Sync* is $O(L \cdot T \cdot N \log N)$ and L is the number of different clustering models.



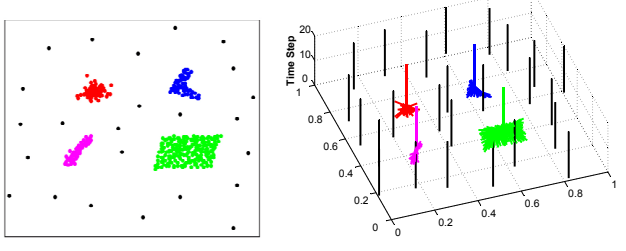
(a) Arbitrarily shaped clusters. (b) Clusters of various density.

Figure 6: Clustering driven by the intrinsic data structure.

4. EXPERIMENTAL EVALUATION

To extensively study the performance of *Sync*, we compare its performance to representatives of various clustering paradigms on synthetic and real-world data: We selected the iterative partitioning algorithm K-Means [22], the density-based algorithm DBSCAN [13], spectral clustering (SC) [24], the algorithm MeanShift [10] and the affinity propagation algorithm (AP) [14]. For the comparison methods we tried several parameter settings and present the best result. Moreover, we also compare to several state-of-the-art parameter-free clustering algorithms: To X-Means [26], which extends K-Means by estimation of the number of clusters based on MDL and to RIC [8] and OCI [9] which have been recently proposed for parameter-free and outlier-robust clustering. For X-Means, RIC and OCI, no parameter settings are required. In all experiments, we used *Sync* in combination with MDL as introduced in Section 3.3.3 which also does not require any input parameters.

We implemented spectral clustering and *Sync* in Java and obtained source code of RIC and OCI from the authors. The source code of the MeanShift algorithm is available at: `http://www.mathworks.com/matlabcentral/fileexchange/10161-mean-shift-clustering` and that of AP at: `http://www.psi.toronto.edu/affinitypropagation/`. K-Means, DBSCAN and X-Means are implemented in WEKA available at `http://www.cs.waikato.ac.nz/ml/weka`. All experiments have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM. Comparing the result of different clustering algorithms with respect to effectiveness is a non-trivial problem, especially if different algorithms produce results with different numbers of clusters. To provide an objective comparison of effectiveness, we report EC, an information-theoretic cluster validity measure proposed in [12]. Intuitively, EC corresponds to the number of bits required to encode the class labels when the cluster labels are known, the smaller EC the better is the clustering. Recently, 3 more robust information-theoretic measures for cluster quality have been proposed in [29]: Adjusted Mutual Information (AMI), Adjusted Variation of Information (AVI), and Normalized Variation of Information (NVI). For these measures, higher values represent better clusterings.



(a) Outlier-robust clustering. (b) Object dynamics plot.

Figure 7: Natural outlier handling.

4.1 Synthetic Data

We start the evaluation with two-dimensional synthetic data to facilitate presentation and demonstrate the benefits of *Sync* mentioned in Section 1.2.

4.1.1 *Clusters Reflecting the Intrinsic Structure*

We first evaluate the performance of *Sync* to detect natural clusters in difficult settings, starting with clusters of arbitrary shape and data distribution. The data set displayed in Figure 6(a) consists of 6 clusters: one Gaussian cluster, one correlation clusters and 3 other arbitrarily shaped clusters. *Sync* successfully detects all types of clusters without requiring any input parameters. Moreover, *Sync* allows detecting clusters of very different object density, as Figure 6(b) demonstrates.

4.1.2 *Powerful Clustering and Outlier Detection*

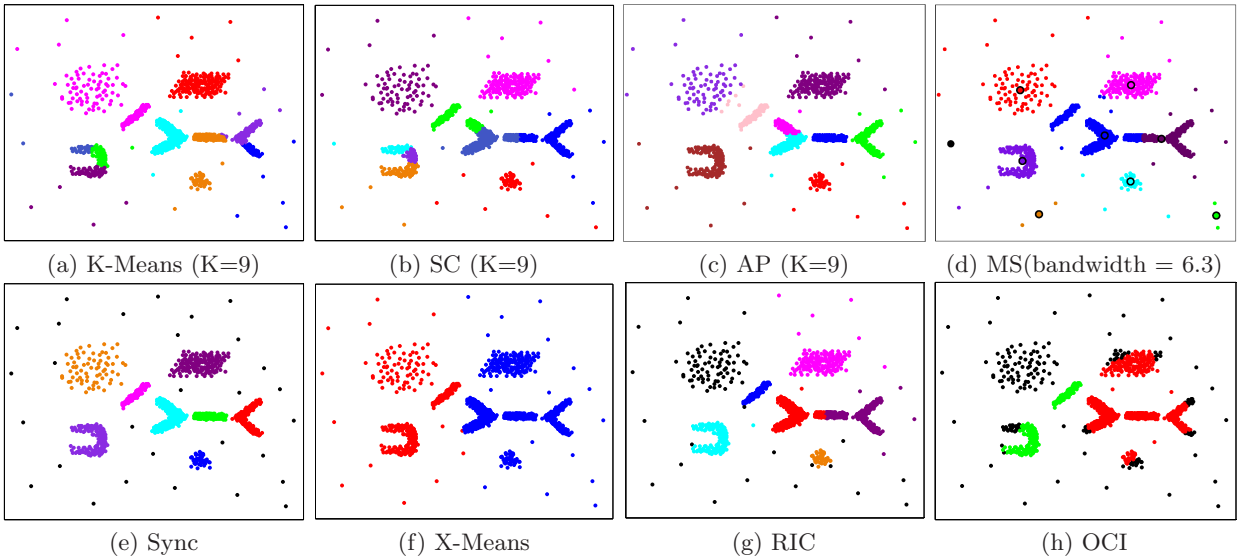
The performance of *Sync* does not degrade in the presence of outliers or noise points. Regardless if the data set contains only single outliers, such as the example in Section 3.3.2, or more noise, such as the data set displayed in Figure 7(a), the clustering result remains stable. In addition, the object dynamics plot provides interesting information on the different kinds of objects: border cluster points, local as well as global outliers.

4.1.3 *Performance on High-dimensional Data*

To assess the impact of the dimensionality of the data space on *Sync*, we created a 5-dimensional synthetic data set (1,000 objects, 5 clusters) and successively increased the dimensionality by adding noise dimensions with uniform data distribution. Table 2 presents the results. The performance of *Sync* does not degrade up to an dimensionality of 8 as demonstrated by a constant EC of 0.09. Also on the 10-dimensional data set, which corresponds to a fraction of one third noise dimensions, 5 clusters are detected but one instance is regarded as a noise point resulting to a slight increase of EG to 0.092. When the dimensionality increases to 12, more and more objects are viewed as noise but *Sync* still obtains the correct clusters (EC 0.44). *Sync* detects 4 clusters with dimensionality of 15. Two clusters are merged together and some objects are regarded as noise (EC 0.8). In summary, the *Sync* performs very well on high-dimensional data with a fraction of approximately up to 50% noise dimensions.

Table 2: Performance of *Sync* on High-dimensional data.

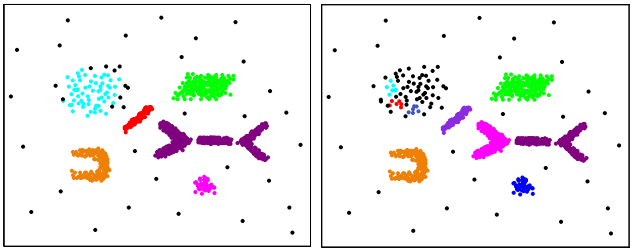
Data set	#Dim./#Noise Dim.	No. of Clusters	Clusters Detected	EC
DS1	5/0	5	5	0.090
DS2	8/3	5	5	0.090
DS3	10/5	5	5	0.092
DS4	12/7	5	5	0.439
DS5	15/10	5	4	0.801



(a) K-Means (K=9) (b) SC (K=9) (c) AP (K=9) (d) MS(bandwidth = 6.3)

(e) Sync (f) X-Means (g) RIC (h) OCI

Figure 8: Comparison with various clustering algorithms: K-Means, Spectral Clustering (SC), Affinity Propagation (AP), MeanShift (MS), X-Means, RIC and OCI.



(a) DBSCAN(ϵ=0.035) (b) DBSCAN(ϵ=0.025)

Figure 9: The result of DBSCAN with different parameters on the synthetic data.

4.1.4 Comparison to State-of-the-art Clustering

For comparison, we created a 2-d data set consisting of 8 arbitrarily shaped clusters plus noise points, cf. Figure 8. For K-Means and spectral clustering, best results have been achieved for $K = 9$ clusters. K-Means can not successfully detect the clusters in this data set which are not limited to Gaussian or spherical data distribution, cf. Figure 8(a). Spectral clustering, which in principle has the potential to detect arbitrarily shaped clusters also performs poorly on the data set cf. Figure 8(b). The reason lies in the fact that the performance of spectral clustering is severely affected by outliers or noise objects. Parameterized as recommended in [14], affinity propagation yields 341 clusters. However, the implementation also allows manually selecting the number of clusters. For comparison with the other algorithms, the result with $K = 9$ clusters is displayed in Figure 8(c). As for spectral clustering, the performance of AF degrades in the presence of noise and outliers. For MeanShift clustering, best results have been achieved setting the window size to 6.3. In Figure 8(d) the areas of estimated major object density are marked by black circles. Similar to SC and AF, the density estimation in Meanshift is disturbed by outliers. For DBSCAN, we tried a wide range of different settings for the parameters $MinPts$ and ϵ. However, to find two suitable parameters is not a trivial task as they are correlated. Fixing $MinPts = 6$ (the default value in Weka) and varying ϵ yielded the best clustering results, displayed in Figure 9(a-b). Since the data includes clusters of various object density, DBSCAN can not identify all these clusters at the same time. In comparison with these established clustering algorithms, *Sync* automatically and correctly detects all clusters and noise points driven by the synchronization principle, cf. Figure 8(e).

In the experiments displayed in Figure 8(f-h), we further compared the performance of *Sync* to various parameter-free clustering algorithms. As evident from Figure 8(f), X-Means fails to detect the clusters since they are not Gaussian and the data set contains noise points. The cluster model of RIC is limited to linear attribute correlations and a predefined set of PDFs, which does not fit to our example data set. Therefore, RIC does not perform very well on this example, cf. Figure 8(g). Also the performance of OCI is not satisfying since again, the assumption of the data distribution, in this case Exponential Power Distribution, does not fit to our example data set, cf. Figure 8(h).

4.2 Real World Data

In this section, we evaluate the performance of *Sync* on real-world data publicly available at the UCI machine learning repository (`http://archive.ics.uci.edu/ml`). Due to space limitation and difficult parametrization, we limit the comparison to the parameter-free clustering algorithms.

4.2.1 Wisconsin Data

The Wisconsin data set deriving from a study on breast cancer consists of 683 instances which are labeled to the classes malignant (M: 239 instances) and benign (B: 444 instances) (16 instances with missing values have been removed from the original data set). Each instance is described by 9 numerical attributes.

Sync detects two clusters successfully. The first cluster with 433 objects represents the class benign, the second

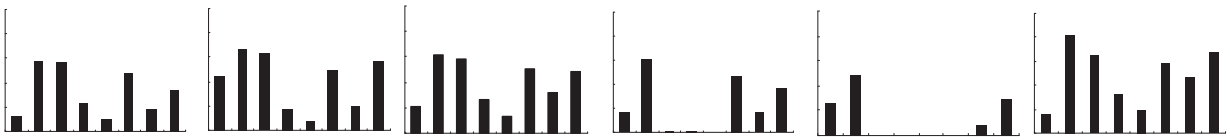


Figure 10: Illustration of the result of *Sync* on diabetes data: each bar in each of the 6 clusters indicates the mean value of different factors and is scaled to [0,1]. The eight factors correspond to: number of pregnancies, plasma glucose concentration, diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin, body mass index, diabetes pedigree function and age, respectively.

cluster with 250 objects the class malignant. In total, 23 instances have been wrongly clustered which results in an EC of 0.154. X-Means detects 3 clusters (C1:197(M:23,B:174), C2:261(B:261), C3:225(M:216 B:9)). 32 instances have been wrongly clustered with EC of 0.183. With RIC, five clusters are obtained with EC of 0.182. On the data set, also the OCI algorithm obtains a good result comprising 13 clusters with good class purity, as reported in [9]. This clustering result has an equally well EC(0.154) as the result of *Sync*. However, the model complexity of *Sync* corresponds with 2 clusters to the expert rating that there are 2 classes in the data set.

In total, we notice that *Sync* achieves all we wanted: (1) *Sync* automatically finds the correct number of clusters; (2) *Sync* detects natural clusters in the data (with high EC value); (3) *Sync* discovers almost all objects of each cluster with high recall (96.2% and 97.5%); (4) all instances in each cluster match with corresponding type (with highest precision of 98.6% and 93.2%).

4.2.2 Diabetes Data

The Pima Indian diabetes database is a collection of medical diagnostic reports of 768 samples from a population living near Phoenix, Arizona, USA. Each sample consists of eight significant risk factors which were chosen for forecasting the onset of diabetes, including e.g. the number of pregnancies, the diastolic blood pressure, the diabetes pedigree function, age, etc. These samples are labeled to 2 classes (Positive:268; Negative:500), namely whether the patient is tested positive for diabetes or not according to World Health Organization criteria. *Sync* detects 6 clusters on the data set. Figure 10 summarizes the cluster contents. Each bin represents the scaled average of each attribute within the cluster. The first cluster consists of 477 samples and mainly hosts the healthy subjects (365 samples). This group is characterized with the relative low value of these eight risk factors, especially for number of pregnancies and the body mass index which is characteristic for young people. *Sync* assigns 118 out of 216 samples of the patient group to cluster 2, where are mainly middle-aged people who have a high number of pregnancies, and a high plasma glucose concentration. Cluster 3 and Cluster 6 are mainly hosting subjects with diabetes, which are characterized by high diabetes pedigree function, skin fold thickness and age. Subjects in cluster 4 are characterized by a high plasma glucose concentration a 2 hours in an oral glucose tolerance test and missing measurements for blood pressure, skin fold thickness and 2-hour serum insulin. Moreover, Cluster 5 hosts people who have no measurement of blood pressure, skin fold thickness, 2-jour serum insulin and body mass index. In total, the clustering results in an EC of 0.625. X-Means merges all instances in one big cluster. RIC and OCI detect 3 and 4 clusters, respectively and corresponding EC values are 0.661 and 0.635. In summary, *Sync* detects meaningful clusters and yields best result with the lowest EC value. For a detailed comparison of the clustering results, Tables 3-6 provide a summary of the quality criteria introduced in [29]. As with EC, *SynC* clearly outperforms the comparison methods on most data sets.

Table 3: Performance of *Sync* on high-dimensional data.

Data set	#Dim./#Noise Dim.	No. of Clusters	Clusters Detected	NMI	AMI	AVI
DS1	5/0	5	5	1	1	1
DS2	8/3	5	5	1	1	1
DS3	10/5	5	5	0.996	0.996	0.998
DS4	12/7	5	5	0.785	0.783	0.811
DS5	15/10	5	4	0.557	0.555	0.569

Table 5: Performances on Wisconsin data.

Algorithms	SynC	X-Means	OCI	RIC
NMI	0.7767	0.3238	0.2742	0.3441
AMI	0.7765	0.3223	0.2716	0.3428
AVI	0.7821	0.464	0.4112	0.4747

5. CONCLUSION

In this paper, we introduced *Sync*, a novel clustering algorithm based on synchronization. The basic idea is to regard each data object as a phase oscillator and simulate the dynamical object interaction over time. To characterize the object interaction, we proposed an adapted variant of the Kuramoto model suitable for clustering. Our extensive experiments demonstrated that the *Sync* algorithm based on synchronization shows several desirable properties:

- *Sync* can detect arbitrarily shaped clusters driven by the true topological structure of the data without assuming any data distribution;
- outliers are naturally separated from cluster points depending on their dynamical behaviors;
- in combination with Minimum Description Length (MDL), *Sync* allows fully automatic clustering.

In ongoing and future work, we focus on exploiting the powerful concept of synchronization for hierarchical clustering. In addition, we investigate on data visualization techniques based on the simulated object movement. As a long term goal we want to closely integrate simulation into the data mining process to design robust algorithms.

Table 4: Performances on synthetic data.

Algorithms	SynC	K-Means	SC	X-MEANS	OCI	RIC	DBSCAN(0.035)	DBSCAN(0.025)	AP	MS
NMI	1	0.8493	0.7702	0.2882	0.3241	0.8342	0.7318	0.9	0.8837	0.7034
AMI	1	0.8472	0.767	0.287	0.3219	0.8325	0.729	0.8985	0.8822	0.6999
AVI	1	0.8527	0.7732	0.4422	0.4427	0.8828	0.8364	0.9338	0.9103	0.7957

Table 6: Performances on Diabetes data.

Algorithms	SynC	X-Means	OCI	RIC
NMI	0.0514	0.0512	0.0319	0.0109
AMI	0.0481	0.0503	0.0307	0.009
AVI	0.0582	0.0508	0.038	0.0111

6. REFERENCES

[1] J. A. Acebron, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Rev. of Modern Physics*, 77(2):137–185, Jan. 2005.

[2] D. Aeyels, and F. D. Smet. A mathematical model for the dynamics of clustering. *Physica D: Nonlinear Phenomena*, 273(19):2517–2530, 2008.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD Conf.*, pages 94–105, 1998.

[4] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. *SIGMOD Conf.*, pages 49–60, 1999.

[5] A. Arenas, A. Diaz-Guilera, J. Kurths, Y. Moreno and C. S. Zhou. Synchronization in complex networks. *Phys. Rep. 469*, pages 93–153, 2008.

[6] A. Arenas, A. Diaz-Guilera, and C. J. Perez-Vicente. Synchronization reveals topological scales in complex networks. *Phys. Rev. Lett.*, 96:114102, 2006.

[7] F. Bach and M. Jordan. Learning spectral clustering. *NIPS Conf.*. MIT Press, 2004.

[8] C. Böhm, C. Faloutsos, J.-Y. Pan, and C. Plant. Robust information-theoretic clustering. *KDD Conf.*, page 65–75, 2006.

[9] C. Böhm, C. Faloutsos, and C. Plant. Outlier-robust clustering using independent components. *SIGMOD Conf.*, pages 185–198, 2008.

[10] D. Comaniciu, and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Patt. Analy. Mach. Intell.* 24(5): 603–619, 2002.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–31, 1977.

[12] B. Dom. An information-theoretic external cluster-validity measure. *Technical Report RJ10219*, IBM, 2001.

[13] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD Conf.*, pages 226–231 1996.

[14] B. J. Frey, D. Dueck. Clustering by passing messages between data points. *Science*, 315: 972–976, 2007.

[15] P. Grünwald. A tutorial introduction to the minimum description length principle. *Advances in Minimum Description Length: Theory and Applications*, 2005.

[16] S. Guha, R. Rastogi and K. Shim. CURE: An Efficient Clustering Algorithm for Large Databases, *SIGMOD Conf.*, pages 73–84, 1998.

[17] G. Hamerly and C. Elkan. Learning the k in k-means. *NIPS Conf.*, 2003.

[18] A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice-Hall. 1988.

[19] C. S. Kim, C. S. Bae, and H. J. Tcha. A phase synchronization clustering algorithm for identifying interesting groups of genes from cell cycle expression data. *BMC Bioinformatics*, 9(56), 2008.

[20] Y. Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. *International Symposium on Mathematical Problems in Theoretical Physics, Lecture Notes in Physics*, pages 420–422, 1975.

[21] Y. Kuramoto. Chemical oscillations, waves, and turbulence. *Springer-Verlag*, New York,NY,USA, 1984.

[22] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. *5-th Berkeley Symposium on Math. Stat. and Prob. Vol. 1*, University of California Press, pages 281–297, 1967.

[23] F. Murtagh. A Survey of Recent Advances in Hierarchical Clustering Algorithms. *Comput. J.* 26(4): 354–359, 1983.

[24] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *In Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[25] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. *VLDB Conf.*, pages 144–155. 1994.

[26] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. *ICML Conf.*, pages 727–734, 2000.

[27] P. Seliger, S. C. Young, and L. S. Tsimring. Plasticity and learning in a network of coupled phase oscillators. *Phys. Rev. E*, 65:137–185, Jan. 2002.

[28] B. Silverman. Density Estimation for Statistics and Data Analysis. *CHAPMAN and HALL*, 1986.

[29] N. X. Vinh, J. Epps, and J. Bailey. Information Theoretic Measures for Clusterings Comparison: Is a Correction for Chance Necessary? *ICML Conf.*, pages 1073–1080, 2009.

[30] M. P. Wand and M. C. Jones. Kernel Smoothing. *Chapman and Hall*, London, 1995.

[31] T. Zhang, R. Ramakrishnan, and M. Livny. An efficient data clustering method for very large databases. *SIGMOD Conf.*, pages 103–114, 1996.