Dynamic Cluster Formation using Level Set Methods [*]

Andy M. Yip[‡][†] Chris Ding[‡] Tony F. Chan[†]

Abstract

Density-based clustering has the advantages for (i) allowing arbitrary shape of cluster and (ii) not requiring the number of clusters as input. However, when clusters touch each other, both the cluster centers and cluster boundaries (as the peaks and valleys of the density distribution) become fuzzy and difficult to determine. In higher dimension, the boundaries become wiggly and over-fitting often occurs.

We introduce the notion of *cluster intensity function* (CIF) which captures the important characteristics of clusters. When clusters are well-separated, CIFs are similar to density functions. But as clusters touch each other, CIFs still clearly reveal cluster centers, cluster boundaries, degree of membership of each data point to the cluster that it belongs, and, whether a certain data point is an outlier or not. Clustering through bump hunting and valley seeking based on these functions are more robust than that based on kernel density functions which are often oscillatory or oversmoothed. These problems of kernel density estimation are resolved using level set methods and related techniques.

Keywords: Clustering Algorithms, Level Set Methods, Cluster Intensity Functions, Unsupervised Learning.

1 Introduction

Recent computer, internet and hardware advances produce massive data which are accumulated rapidly. Applications include sky surveys, genomics, remote sensing, pharmacy, network security and web analysis. Undoubtedly, knowledge acquisition and discovery from such data become an important issue. One common technique to analyze data is clustering which aims at grouping entities with similar characteristics together so that main trends or unusual patterns may be discovered. See [9, 7] for examples of clustering techniques.

Among various classes of clustering algorithms, density-based methods are of special interest for their connections to statistical models which are very useful in many applications. Density-based clustering has the advantages for (i) allowing arbitrary shape of cluster and (ii) not requiring the number of clusters as input, which is usually difficult to determine. Examples of density-based algorithms can be found in [5, 2, 8, 1].

There are several basic approaches for density-based clustering. (A1) The most common approach is so-called bump-hunting, i.e., first find the density peaks or "hot spots" and then expand the cluster boundaries outward, until they meet somewhere, presumably in the valley regions (local minimums) of density contours [1].

(A2) Another direction is to start from valley regions and gradually work uphill to connect data points in low-density regions to clusters defined by density peaks [6, 8].

(A3) A recent approach is to compute reachability from some seed data and then connect those "reachable" points to their corresponding seed [5, 2].

When clusters are well-separated, density-based methods work well because the peak and valley regions are well-defined and easy to detect. When clusters touch each other, which is often the case in real situations, both the cluster centers and cluster boundaries (as the peaks and valleys of the density distribution) become fuzzy and difficult to determine. In higher dimension, the boundaries become wiggly and over-fitting often occurs.

Level Set Methods We recognize that the key issue in density-based approach is how to advance the boundary either from peak regions outward towards valley regions, or the other way around.

In this paper, we introduce level set methods to resolve the boundary advancing problem. Level set methods are widely used in applied mathematics community. They are originally introduced to solve the problem of front propagation of substances such as fluids, flame and crystals where an elegant representation of boundaries is essential [13]. Level set methods have well-established mathematical foundations and have been successfully applied to solve a variety of problems in image processing, computer vision, computational fluid dynamics, optimal design and material science, see [15, 12] for details.

In image processing, one typically interested in detecting sharp edges in an image; a smooth front advanced via level set methods can easily capture these edges. The methods can be modified [Chan and Vese

[*]This work was partially supported by grants from DOE under contract DE-AC03-76SF00098, NSF under contracts DMS-9973341 and ACI-0072112, ONR under contract N00014-02-1-0015 and NIH under contract P20 MH65166.

[†]Department of Mathematics, University of California, Los Angeles, 405 Hilgard Avenue, Los Angeles, CA 90095-1555. Email: {*mhyip,chan*}*@math.ucla.edu*

[‡]Computational Research Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720. Email: *chqding@lbl.gov*

in [3]] to detect not-so-sharp boundaries which is close to clustering 2-dimensional data points. However, these methods are mainly designed for image segmentation which is not suitable for data clustering in general.

An important advantage of level set method is that the boundaries in motion can be made smooth conveniently and smoothness can be easily controlled by a parameter that characterizes surface tension. Furthermore, the advancing of boundaries is achieved naturally within the framework of partial differential equation (PDE) which governs the dynamics of the boundaries. Using level set methods, boundary advancing, especially when boundaries need to be split or merged , can be easily done in a systematical way. This feature is very important in data clustering as clusters can be merged or split in an automatic fashion.

We may use level set methods strictly as an effective mechanism for advancing boundaries. For example, in the above approach (A1), once the density peaks are detected, we may advance cluster boundaries towards low-density regions using level set methods. This would be a level set-based bump hunting approach.

However, it turns out that utilizing level set methods we can further develop a new and useful concept of *cluster intensity function.* A suitably modified version of level set methods becomes an effective mechanism to formulate cluster intensity functions in a dynamic fashion. Therefore our approach goes beyond the three approaches described earlier.

Cluster Intensity Functions We introduce the notion of "cluster intensity function" (CIF) which captures the important characteristics of clusters. When clusters are well-separated, CIFs become similar to density functions. But as clusters touch each other, CIFs still clearly describe the cluster structure whereas density functions and hence cluster structure become blurred. In this sense, CIFs are a better representation of clusters than density functions.

A number of clustering algorithms are based on kernel density functions (KDFs) obtained through kernel density estimation [4]. KDFs possess many nice properties which are good for clustering purposes. However, they also have some drawbacks which limit their use for clustering (see the subsection Kernel Density Estimation).

CIFs, however, resolve the problems of KDFs while advantages of KDFs are inherited. Although CIFs are also built on the top of KDFs, they are cluster-oriented so that only information contained in KDFs that is useful for clustering is kept while other irrelevant information is filtered out. We have shown that such a filtering process is very important in clustering especially when the clusters touch each other. On the other hand, it is well-known that when the clusters are well-separated, then valley seeking on KDFs results in very good clusterings. Since the valleys of CIFs and KDFs are very similar, if not identical, when the clusters are well-separated, clustering based on CIFs is as good as that based on KDFs. However, advantages of CIFs over KDFs become very significant when the clusters touch each other.

Kernel Density Estimation In density-based approach, a general philosophy is that clusters are high density regions separated by low density regions. We particularly consider the use of kernel density estimations [4, pp.164–174] (also known as the Parzen-window approach), a non-parametric technique to estimate the underlying probability density from samples. More precisely, given a set of data $\{\mathbf{x}_i\}_{i=1}^N \subset R^p$, the KDF used to estimate density is defined to be

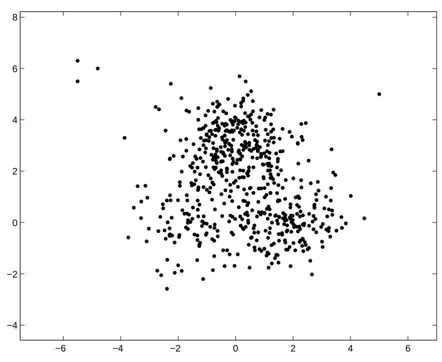
$$\hat{f}_N(\mathbf{x}) := \frac{1}{N h_N^p} \sum_{i=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_N}\right) \tag{1.1}$$

where $K(\mathbf{x})$ is a positive kernel and h_N is a scale parameter. Clusters may then be obtained according to the partition defined by the valleys of $\hat{f}_N(\mathbf{x})$. Efficient valley seeking algorithm is also available [6] which does not require finding the valleys explicitly.

There are a number of important advantages of kernel density approach. Identifying high density regions is independent of the shape of the regions. Smoothing effects of kernels make density estimations robust to noise. Kernels are localized in space so that outliers do not affect the majority of the data. The number of clusters is automatically determined from estimated density functions.

Despite the numerous advantages of kernel density methods, there are some drawbacks which deteriorate the quality of the resulting clusterings. KDFs are very often oscillatory (uneven) since they are constructed by adding many kernels together. Such oscillatory nature may lead to the problem of over-fitting, for instance, when clusters touch each other, a smooth cluster boundary between the clusters are usually preferred than an oscillatory one. Last but not least, valleys and peaks of KDFs are often very vague especially when clusters touch each other.

In Figure 1, we show a data set drawn from a mixture of three Gaussian components and the estimated KDF $\hat{f}_N(\mathbf{x})$. We observe that the valleys and peaks correspond to the two smaller large clusters of the KDF are very vague or may even not exist. It is non-trivial to see that three large clusters exist. Thus, the performance of applying valley seeking algorithm based on the KDF is poor.


(a)

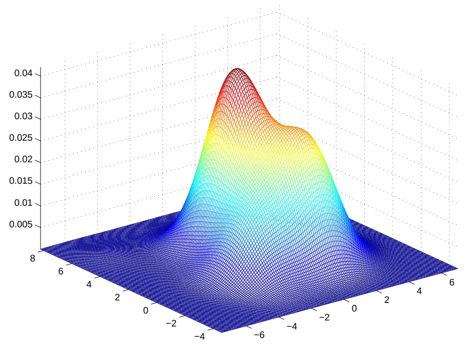

(b)

Figure 1: (a) Data set consisting of a mixture of three Gaussian distributions. (b) Estimated density function $\hat{f}_N(\mathbf{x})$ using Gaussian kernel with window size $h = 1$. In (b), peaks and valleys corresponding to the two smaller large clusters are very vague that it is non-trivial to see that three large clusters exist.

The organization of the rest of the paper is as follows. In §2, we outline our method. In §3, some theoretical results are presented to justify our method. Finally, experiments are presented in §4.

2 Cluster Formation

In this section, we describe our methodology to construct clusters using level set methods.

We start by introducing some terms that will be used throughout the rest of the paper. A *cluster core contour* is a closed surface surrounding the core part/density peak of a cluster at which density is relatively high. A *cluster boundary* refers to the interface between two clusters, i.e., a surface separating two clusters. A cluster core contour is usually located near a density peak while a cluster boundary is located at the valley regions of a density distribution. Here, a point $\mathbf{x}$ is said to belong to a valley region of $\hat{f}_N(\mathbf{x})$ if there exists a direction along which $\hat{f}_N(\mathbf{x})$ is a local minimum.

Our method consists of the following main steps which will be elaborated in details in the next subsections:

1. Initialize cluster core contours to obtain a rough outline of density peaks;

2. Advance the cluster core contours using level set methods to find density peaks;

3. Apply valley seeking algorithm on the CIF constructed from the final cluster core contours to obtain clusters.

2.1 Initialization of Cluster Core Contours In this subsection, we describe how to construct an initial cluster core contours Γ effectively. The basic idea is to locate the contours at which $\hat{f}_N(\mathbf{x})$ has a relatively large (norm of) gradient. In this way, regions inside Γ would contain most of the data points — we refer these regions as cluster regions. Similarly, regions outside Γ would contain no data point at all and we refer them as non-cluster regions.

To construct an interface which divides the space into cluster regions and non-cluster regions reasonably, we construct the initial cluster core contours Γ as follows.

Definition 2.1. *An initial cluster core contours Γ is defined to be the set of zero crossings of $\Delta\hat{f}_N(\mathbf{x})$, the Laplacian of $\hat{f}_N(\mathbf{x})$. Here, a point $\mathbf{x}$ is a zero crossing if $\Delta\hat{f}_N(\mathbf{x}) = 0$ and within any arbitrarily small neighborhood of $\mathbf{x}$, there exist $\mathbf{x}^+$ and $\mathbf{x}^-$ such that $\Delta\hat{f}_N(\mathbf{x}^+) > 0$ and $\Delta\hat{f}_N(\mathbf{x}^-) < 0$.*

We note that Γ often contains several closed surfaces. The idea of using the set of zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ is that it outlines the shape of data sets very well and that for many commonly used kernels (e.g. Gaussian and cubic B-spline) the sign of $\Delta\hat{f}_N(\mathbf{x})$ indicates whether $\mathbf{x}$ is inside or outside Γ.

Reasons for using zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ to outline the shape of data sets are several folds: (a) the solution is a set of surfaces at which $\|\nabla\hat{f}_N(\mathbf{x})\|$ is relatively large; (b) the resulting Γ is a set of closed surfaces; (c) Γ well captures the shape of clusters; (d) the Laplacian operator is an isotropic operator which does not bias towards certain directions; (e) the equation is simple and easy to solve; (f) it coincides with the definition of edge in the case of image processing. In fact, zero crossings of Laplacian of image intensity functions are often used for edge detection to outline an object in image processing [10].

In Figure 2, we show the cluster core contour defined based on zero crossings of $\Delta \hat{f}_N(\mathbf{x})$ juxtaposed with the underlying data set (the data set in Figure 1(a)). We observe that the set of zero crossings of $\Delta \hat{f}_N(\mathbf{x})$ captures the shape of the data set very well.

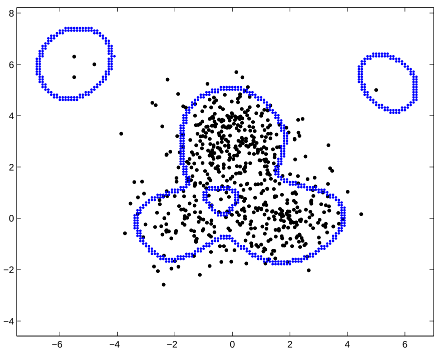


Figure 2: Cluster core contours defined based on zero crossings of $\Delta \hat{f}_N(\mathbf{x})$ capture the shape of the data set very well.

2.2 Advancing Cluster Core Contours Next, we discuss how to advance the initial cluster core contours to obtain peak regions through hill climbing in a smooth way. We found that this is a key issue in density-based approaches and this is also how ideas from level set methods come into play. More precisely, we employ PDE techniques to advance contours in an elegant way.

Since each cluster core contour Γ_i in the initial set of cluster core contours Γ changes its shape when we evolve it, we parameterize such a family of cluster core contours by a time variable t, i.e., the i-th cluster core contour at time t is denoted by $\Gamma_i(t)$. We also define the *mean curvature* κ of a contour Γ_i to be

$$\begin{aligned}\kappa(\mathbf{x},t) &= \nabla \cdot \left(\frac{\nabla\phi(\mathbf{x},t)}{\|\nabla\phi(\mathbf{x},t)\|} \right) \\ &= \frac{\phi_y^2\phi_{xx} - 2\phi_x\phi_y\phi_{xy} + \phi_x^2\phi_{yy}}{(\phi_x^2+\phi_y^2)^{3/2}}.\end{aligned}$$

In level set methods, if we want to evolve a closed surface Γ embedded in a level set function $\phi(\mathbf{x},t)$ with speed $\beta(\mathbf{x},t)$, then the equation is given by

$$\frac{\partial\phi}{\partial t} = \beta\|\nabla\phi\|$$

which is known as the level set equation [12]. Our equation also takes this form.

Given an initial contour $\Gamma_i(0)$, the time dependent PDE that we employ for hill climbing on density functions is given by

$$\frac{\partial\phi}{\partial t} = \left(\frac{1}{1+\|\nabla \hat{f}_N\|} + \alpha\kappa \right) \|\nabla\phi\| \tag{2.2}$$

with initial condition given by a level set function constructed from $\Gamma_i(0)$. This equation is solved independently for each component cluster core contour in $\Gamma(t)$. Evolution is stopped when a stopping criterion is satisfied. In fact, we stop evolution if a contour becomes convex or if a contour becomes stable in sense that it is not split.

The aim of the factor $1/(1+\|\nabla \hat{f}_N\|)$ is to perform hill climbing to look for density peaks. Moreover, the factor also adjusts the speed of each point on the cluster core contour in such a way that the speed is lower if $\|\nabla \hat{f}_N\|$ is larger so that cluster core contours stays in steep regions of $\hat{f}_N(\mathbf{x})$ where peak regions are defined better. In the limiting case where $\hat{f}_N$ has a sharp jump, the cluster core contour actually stops moving at the jump. We remark that in traditional steepest descent methods for solving minimization problems, the speed (step size) is usually higher if $\|\nabla \hat{f}_N\|$ if larger, which is opposite to what we do. This is because our goal is to locate steep regions of $\hat{f}_N$ rather than local minimums.

The curvature term κ exerts tension to the cluster core contour such that the contour is smooth. This mechanism resolves the problem of over-fitting of KDFs. In fact, if $\phi(\mathbf{x},t)$ is kept to be a signed distance function for all t, i.e., $\|\phi(\mathbf{x},t)\| \equiv 1$, then $\kappa = \Delta\phi(\mathbf{x},t)$ so that $\phi(\mathbf{x},t)$ is smoothed out by Gaussian filtering. In variational point of view, the curvature term exactly corresponds to minimization of the length (surface area) of the cluster core contour.

The scalar $\alpha \geq 0$ controls the amount of tension added to the surface and will be adjusted dynamically during the course of evolution. At the beginning of evolution of each $\Gamma_i(0)$, we set $\alpha = 0$ in order to prevent smoothing out of important features. After a contour is split into pieces, tension is added and is gradually decreased to 0. In this way, spurious oscillations can be removed without destroying other useful features.

In summary, the PDE simply (i) moves the initial cluster core contour uphill in order to locate peak regions; (ii) adjusts the speed according to the slope of the KDF; (iii) removes small oscillations of cluster core contours by adding tension so that hill climbing is more robust to the unevenness of the KDF. Of course, the use of level set methods allows the initial cluster core contour to be split and merged easily.

In the following, we apply the PDE to the cluster core contours in Figure 2. In Figure 3, we show the cluster core contours during the course of evolution until the contours become nearly convex and the evolution terminates. In fact, before evolution starts, the two cluster core contours correspond to outliers are convex and hence they are freezed. We observe that the contours are attracted to density peaks. Moreover, when a contour is split into several contours, the pieces are not very smooth near the splitting points. Since tension is added in such cases, the contours are straighten out quickly.

2.3 Cluster Intensity Functions In non-parametric modelling, one may obtain clusters by employing valley seeking on KDFs. However, as mentioned in §1, such methods perform well only when the clusters are well-separated and of approximately the same density in which case peaks and valleys of the KDF are clearly defined. On the other hand, even though we use the density peaks identified by our PDE (2.2) as a starting point. If we expand the cluster cores outward according to the KDF, we still have to face the problems of the KDF; we may still get stuck in local optimum due to its oscillatory nature.

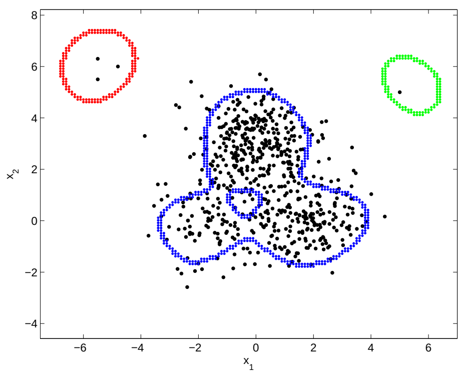
In this subsection, we further explore cluster intensity functions which are a better representation of clusters than that by KDFs. Due to the advantages of CIFs, we propose to perform valley seeking on CIFs to construct clusters, rather than on KDFs. Here, CIFs are constructed based on the final cluster cluster cores obtained by solving the PDE (2.2).

CIFs capture the essential features of clusters and inherit advantages of KDFs while information irrelevant to clustering contained in KDFs is filtered out. Moreover, peaks and valleys of CIFs stand out clearly which is not the case for KDFs. The principle behind is that clustering should not be done solely based on density, rather, it is better done based on density and distance. For example, it is well-known that the density-based algorithm DBSCAN [5] cannot separate clusters that are closed together even though their densities are different.

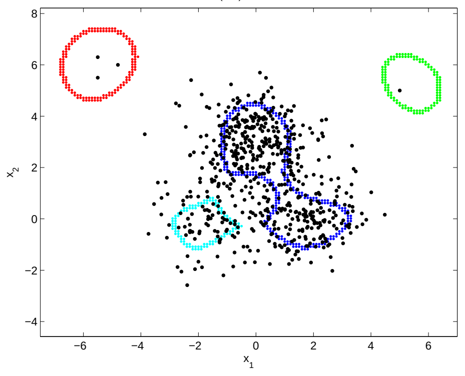
CIFs, however, are constructed by calculating *signed distance* from cluster core contours (which are constructed based on density). Thus, CIFs combine both density and distance information about the data set. We remark that signed distance functions have been widely used as level set functions in level set methods for they are meaningful physically and possess many properties that make computations efficient and accurate, see [12, 15].

The definition of a CIF is as follows. Given a set of closed hypersurfaces Γ (zero crossings of $\Delta \hat{f}_N(\mathbf{x})$ or its refined version), the CIF $\phi(\mathbf{x})$ with respect to Γ is defined to be the signed distance function

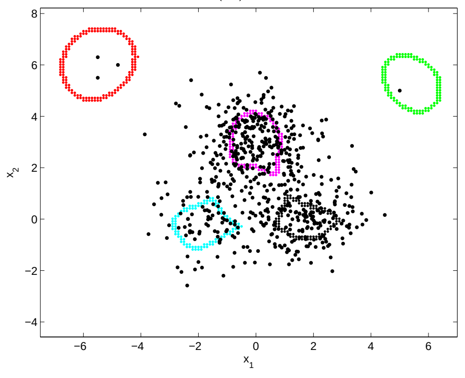
$$(2.3)\quad \phi(\mathbf{x}) = \begin{cases} \min\limits_{\mathbf{y}\in\Gamma} \|\mathbf{x}-\mathbf{y}\| & \text{if } \mathbf{x} \text{ lies inside } \Gamma \\ -\min\limits_{\mathbf{y}\in\Gamma} \|\mathbf{x}-\mathbf{y}\| & \text{if } \mathbf{x} \text{ lies outside } \Gamma \end{cases} .$$



(a)



(b)



(c)

Figure 3: Evolution of Γ in Figure 2 using bump hunting PDEs (2.2). (a) Initial boundary. (b) After 400 iterations. (c) After 800 iterations (converged). We observe that the resulting boundaries capture the hot spots of the data set very well.

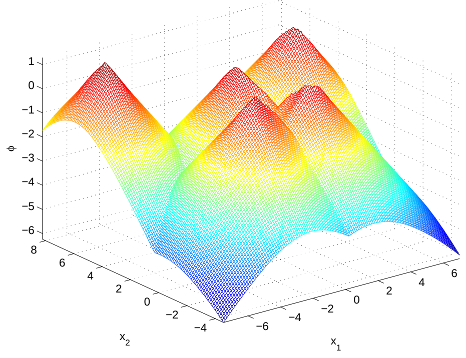


Figure 4: CIF constructed from the contours in Figure 3(c). Peaks corresponding to the three large clusters are clearly seen.

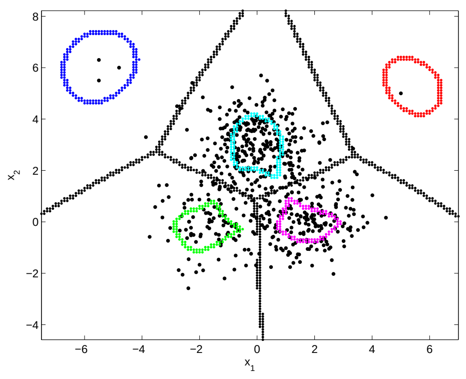


Figure 5: Valleys of the CIF in Figure 4. We observe that the three core clusters are well-discovered.

The value of a CIF at $\mathbf{x}$ is simply the distance between $\mathbf{x}$ and Γ with its sign being positive if $\mathbf{x}$ lies inside Γ and negative if $\mathbf{x}$ lies outside Γ. Roughly speaking, a large positive (respectively negative) value indicates that the point is deep inside (respectively outside) Γ while a small absolute value indicates that the point lies close to the interface Γ.

In Figure 4, the CIF constructed from the cluster core contours in Figure 3(c) is shown. The peaks correspond to the three large clusters can be clearly seen which shows that our PDE is able to find cluster cores effectively.

2.4 Valley Seeking The final step to obtain clusters is to apply valley seeking (see [6]) on the new CIF constructed based on the final cluster core contours according to (2.3). Essentially, we partition the space according to the valleys of the CIF.

The use of signed distance functions as CIFs has a property that their valleys are nothing but the equidistance surfaces between the cluster core contours. Moreover, cluster core contours play a similar role as cluster centers in the k-means algorithm. Thus, our method may be treated as a generalization of the k-means algorithm in the sense that a "cluster center" may be of arbitrary shape instead of just a point.

In Figure 5, we show the valleys of the CIF juxtaposed with the data set and the final cluster core contours. We observe that the three large clusters are well-discovered and the outliers are also separated. We may also observe that the value of a CIF indicates the degree of membership (cluster intensity) of a point to the cluster to which it belongs (measured based on distance).

Under level set methods framework, valleys and peaks are easily obtained. The valleys are just the singularities of the level set function (i.e. CIF) having negative values. On the other hand, the singularities of the level set function having positive values are the peaks or ridges of the CIF (also known as skeleton).

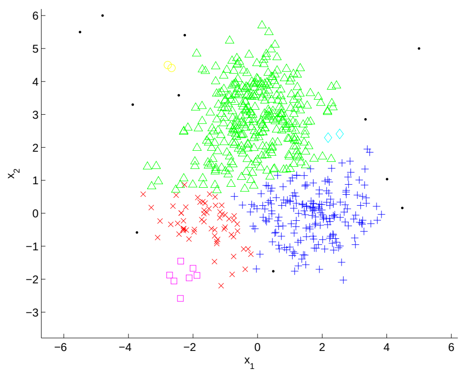
We remark that (i) when applying the valley seeking algorithm, we do not need to find the valleys explicitly — the valleys are shown for visualization purposes only; (ii) the valleys are independent of the choice of the kernel at all — different choices of kernel may result in a slightly different shape of the cluster core contours but the valleys will be quite stable; (iii) one may imagine that if the outliers are removed, then the valleys away from the three large clusters in Figure 4 will be very different, however, the valleys in between the three large clusters will remain the same and hence the final clusterings will be the same (except for the outliers of course).

We now further illustrate how the problem of over-fitting (or under-fitting) of KDFs is resolved using our method. In Figure 6, we show the clustering results of applying valley seeking algorithm on the KDF directly, using the scale parameter $h = 0.6$ and $h = 0.7$. As expected, one can hardly discover the three large cluster using such a method because the valleys are either too vague or too oscillatory. In contrast our method resolves these problems by (i) outlining the shape of the data set well while keeping the cluster core contours smooth; (ii) using curvature motion to smooth out oscillations due to unevenness of KDFs.

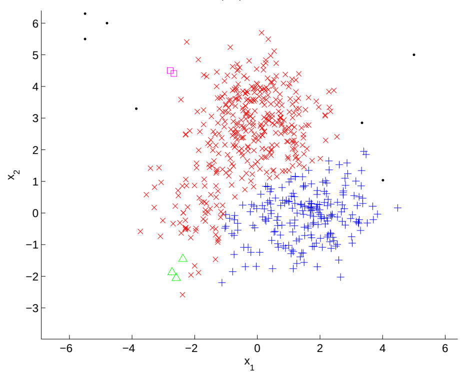
3 Theoretical Results of the Method

In this section, we present some mathematical results to justify our method.

First, we state some fundamental properties of the cluster core contours constructed to justify that the use



(a)



(b)

Figure 6: (a) Clustering result of applying valley seeking algorithm on the KDF with scale parameter $h = 0.6$. (b) Clustering result of applying valley seeking algorithm on the KDF with scale parameter $h = 0.7$. We observed that in (a), due to over-fitting, 18 clusters are discovered with the three large clusters split into pieces. In (b), 11 clusters are found, under-fitting causes some of the regions of the KDF between the three large clusters have no valleys. Hence, two large clusters are merged into one.

of zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ as cluster core contours. To begin, some assumptions on $K(\mathbf{x})$ are needed:

A1. $K(\mathbf{x})$ is at least two times continuously differentiable;

A2. there exist $0 < L_1 < L_2$ such that $K(\mathbf{x})$ is strictly concave for all $\|\mathbf{x}\| < L_1$, strictly concave for all $L_1 < \|\mathbf{x}\| \le L_2$ and concave for all $\|\mathbf{x}\| \ge L_2$.

We remark that Gaussian kernel possesses all these properties with $L_1 = 1$ and $L_2 = \infty$. The following proposition follows from the assumptions A1 and A2.

Proposition 3.1. *If the data set $X \subset R^p$ is non-empty and the kernel satisfies the above assumptions, then zero crossings of $\Delta\hat{f}_N$ exist. Moreover, the set of zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ is a set of bounded closed surfaces in R^p.*

The following proposition states that the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ contain all edges in the infinite samples case.

Proposition 3.2. *If $N \to \infty$ (infinite samples) and if $\hat{f}_N(\mathbf{x})$ is discontinuous on $\tilde{\Gamma}$, then the zero crossings Γ of $\Delta\hat{f}_N(\mathbf{x})$ contains $\tilde{\Gamma}$.*

It is well-known that if the clusters are well-separated, then applying valley seeking algorithm on the KDF will give the correct clustering. Our next proposition states that this is also true for the CIF constructed from the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$. We remark that if the clusters touch each other, then valley seeking on KDFs may result in poor clusterings while our method performs better.

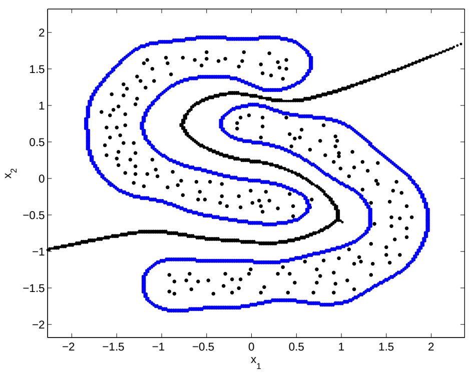
Proposition 3.3. *If all clusters are well-separated, then the valleys of the CIF constructed from the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ give the correct clustering.*

The above proposition follows from the fact that if clusters are well-separated, then each cluster will be roughly surrounded by one cluster core contour. Since these cluster core contours are also well-separated, the valleys of the CIF defined on the top of them will correctly partition the data set.

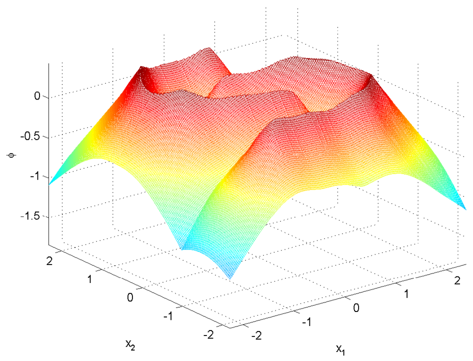
4 Experiments

In addition to the examples shown in Figures 1–6, we give examples to further illustrate the usefulness of the cluster intensity function and the level set techniques. For visualization of cluster intensity functions which is one dimension higher than the data sets, two dimensional data sets are used while the theories presented above apply to any number of dimensions. The PDE (2.2) is solved on a regular grid using finite difference methods. An upwind scheme is used, see [14, pp.80–81] for details. When moving the cluster core contours, we also employ the narrow band version [15, pp.77–85] of level set methods so that only a band of few grid points wide around the cluster core contours is considered. Time step is chosen according to the CFL condition [12, p.44]. CIFs are built by using fast marching methods [15] efficiently (fast sweeping methods [11] may also be used).

Example 1. We illustrate the valleys of CIFs having complicated shape. In the figure, we may see that the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ capture the shape of the data very well while the use of the valleys of the CIF allows us to separate clusters of complicated shape.



(a)



(b)

Figure 7: (a) Two "C" shape clusters juxtaposed with the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$ and the valleys of the cluster intensity function. (b) The cluster intensity function constructed from the zero crossings of $\Delta\hat{f}_N(\mathbf{x})$. In (a), the valleys of cluster intensity function clearly separate the two clusters.

Example 2. In this experiment, we show that our method when applying to the data set in Figure 1(a) recovers the underlying mean of the Gaussian component very well. The true means of the three Gaussian components are $(1.7, 0)$, $(-1.7, 0)$ and $(0, 2.9445)$ while the ones estimated by our algorithm are $(1.6775, -0.0005)$, $(-1.8056, -0.2209)$ and $(0.0294, 2.8979)$. This shows that each peak of the final CIF is very close to the center of the corresponding Gaussian component. Thus, CIFs describe clusters very well.

Example 3. Our next example uses text documents data from three newsgroups. The results are shown in Figure 8. We observe that the clustering results agree with the true clustering very well.

5 Concluding Remarks

In the paper, we introduced level set methods to identify density peaks and valleys in density landscape for data clustering. The method relies on advancing contour to form cluster cores. One key point is that during front advancement, smoothness is enforced via level set methods. Another point is that important features of clusters are captured by cluster intensity functions. The usual problem of roughness of density functions is overcome. The method is shown to be much more robust and reliable than traditional methods that perform bump hunting or valley seeking on density functions.

Our method can also identify outliers effectively. After the initial cluster core contours are constructed, outliers are clearly revealed and can be easily identified. In this method, different contours evolv independently. Thus outliers do not affect normal cluster formation via contour advancing; This nice property does not hold for clustering algorithms such as the k-means where several outliers could skew the clustering.

Our method for front advancement (2.2) is based on the dynamics of front propagation in level set methods. A more elegant approach is to recast the cluster core formation as a minimization problem where the from advancement can be derived from first principles which will be presented in a later paper.

Acknowledgements

A. Yip would like to thank Michael K. Ng (The Univ. of Hong Kong) for his long discussion on level set methods and data clustering and would like to thank Russell Caflisch, Stanley Osher and Stott Parker (UCLA) for their valuable comments and suggestions.

References

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, *Automatic subspace clustering of high dimensional data for data mining applications*, Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data, pp. 94–105, Seattle, WA, 1998.

[2] M. Ankerst, M. Breunig, H. P. Kriegel, and J. Sander, *OPTICS: Ordering points to identify the clustering structure*, Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data, pp. 49–60, Philadelphia, PA, 1999.

[3] T. F. Chan and L. Vese, *Active contours without edges*, IEEE Transactions on Image Processing, 10 (2001), pp. 266–277.

[4] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*, 2nd Ed., New York: Wiley-Interscience, 2001.

[5] M. Ester, H. Kriegel, J. Sander, and X. Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, Proceedings of 2nd Int. Conf. On Knowledge Discovery and Data Mining, Portland, OR, pp.226–231, 1996.

[6] K. Fukunaga, *Introduction to statistical pattern recognition*, 2nd Ed., Boston Academic Press, 1990.

[7] J. Han and M. Kamber, *Data mining: concepts and techniques*, San Francisco: Morgan Kaufmann Publishers, 2001.

[8] A. Hinneburg and D. A. Keim, *An efficient approach to clustering in large multimedia databases with noise*, Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining,pp. 58–65, New York, 1998.

[9] A. K. Jain, M. N. Murty, and P. J. Flyn, *Data clustering: a review*, ACM Computing Surveys, 31:3 (1999), pp. 264–323.

[10] A. K. Jain, *Fundamentals of digital image processing*, Prentice Hall, Englewood Cliffs, NJ, 1988.

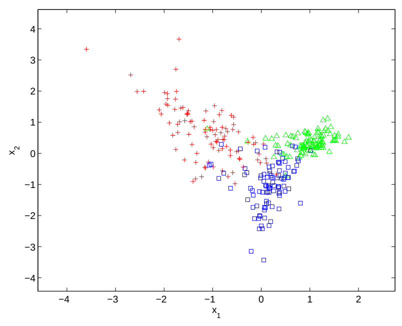
[11] C. Y. Kao, S. Osher, and Y. Tsai, *Fast sweeping methods for Hamilton-Jacobi equations*, submitted to SIAM Journal on Numerical Analysis, 2002.

[12] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*, New York: Spring Verlag, 2003.

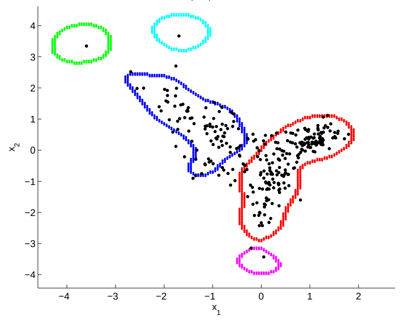
[13] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: algorithms based on Hamiton-Jacobi formulations*, Journal of Computational Physics, 79 (1988), pp. 12–49.

[14] G. Sapiro, *Geometric partial differential equations*, New York: Cambridge University Press, 2001.

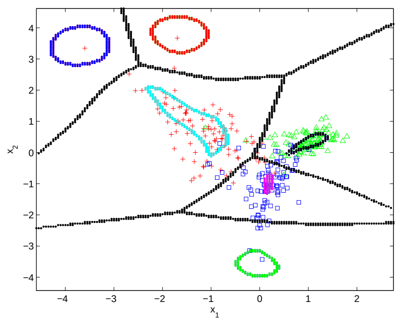
[15] J. A. Sethian, *Level set methods and fast marching methods, 2nd Ed*, New York: Cambridge University Press, 1999.



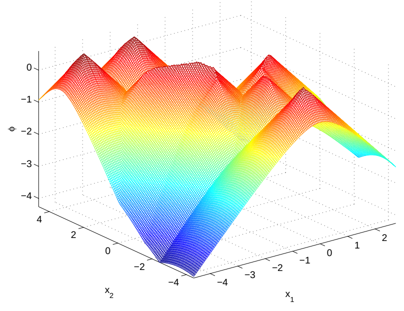
(a)



(b)



(c)



(d)

Figure 8: Clustering results of a newsgroup data set. (a) Data set obtained from three newsgroups having size 100, 99 and 99 respectively. Data points (articles) in the same newsgroup are displayed with the same symbol. (b) The set of zero crossings of $\Delta \hat{f}_N(\mathbf{x})$. (c) Clustering results where the lines are valleys of the final CIF and the closed curves are the final cluster core contours enclosing the core part of the clusters. (d) Cluster intensity function.