Efficient algorithms for non-parametric clustering with clutter

Weng-Keen Wong
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
wkw@cs.cmu.edu

Andrew Moore
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
awm@cs.cmu.edu

Abstract

Detecting and counting overdensities in data is a common problem in the physical and geographic sciences. One of the most successful of recent algorithms for the counting version of the problem was introduced by Cuevas, Febrero and Fraiman [Cuevas et al., 2000], which will be referred to as the CFF algorithm. This algorithm first determines the subset of data points that are in high density regions using a non-parametric density estimator. A clustering step follows where such high density points are agglomerated. While this algorithm was originally intended to estimate the number of clusters, it can also be used to perform non-parametric clustering against a noisy background. However, the algorithm proposed by CFF is too computationally expensive to work on large datasets with greater than two dimensions. We propose an alternative implementation of the CFF algorithm producing exactly the same results but addressing the computational problems in both the density estimation and in the agglomeration step. We will then illustrate the effectiveness of our approach on large multi-dimensional astrophysics datasets.

1 Introduction

The most common kinds of clustering algorithms tend to fall into two categories – mixture models and graph theoretic approaches such as hierarchical single linkage clustering. The best choice is task and data dependent. However, cases do exist where both of these styles perform poorly, such as in Figure 2, which has too much noise for the clustering algorithms to find the appropriate clusters. When single linkage clustering was asked to find five clusters in the example dataset, it joined together nearly all the points, as shown in Figure 3. A mixture of Gaussians, optimized by EM, was also unsuccessful, as the results in Figure 4 indicate. Even though we permitted a uniform background component and started the iteration with hand picked Gaussians, EM was unable to correctly model the shape of the overdense area. Furthermore, seven Gaussians were required by EM, which is more than the three clusters that make up the region of overdensity. On the other hand, the Cuevas-Febrero-Fraiman algorithm, to be described shortly, successfully finds the correct number of clusters as shown in Figure 5.

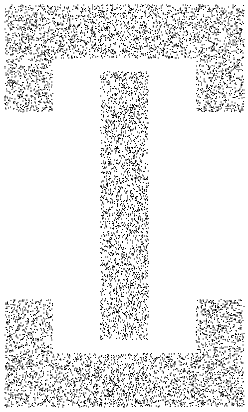

Figure 1: The anchor shape

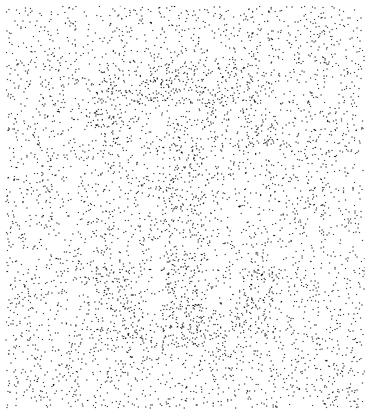

Figure 2: Example dataset with a central anchor shape made of 3 components and 90% clutter

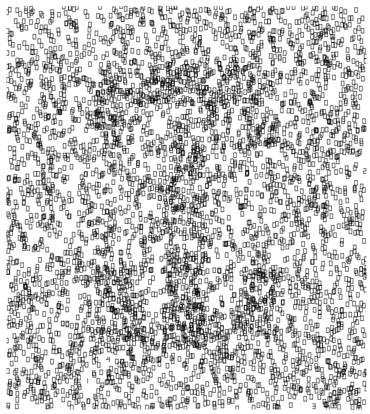

Figure 3: Clusters as found by Single Linkage Clustering. All points marked with a square are in one big cluster. Clusters 2 through 5 are tiny and not in useful places

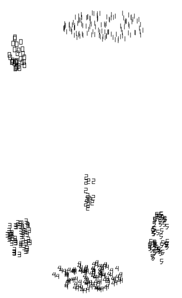

Figure 4: Clusters as found by a mixture of Gaussians with a uniform background component for clutter removal

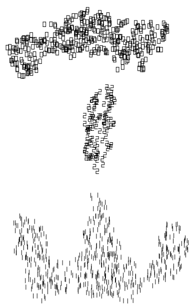

Figure 5: Clusters as found by CFF algorithm

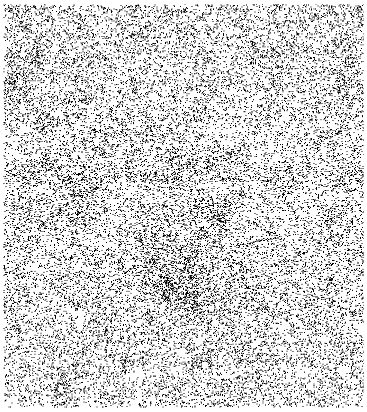

Figure 6: A dataset of galaxies from the Sloan Digital Sky Survey

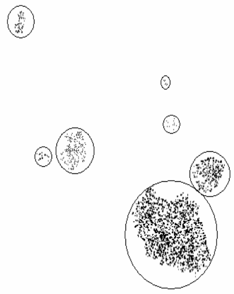

Figure 7: The seven clusters found by CFF and manually circled

Situations with a combination of noisy backgrounds and clusters that can not be modeled by parametric forms such as Gaussians are common enough in the physical sciences, such as astrophysics [Reichart et al., 1999], that a new approach is needed. As an example, consider the dataset shown in Figure 6. This dataset is a two dimensional projection taken from the Early Data Release of the Sloan Digital Sky Survey [Sloan Digital Sky Survey, 2000]. Each point is a galaxy containing many billions of stars. Astrophysicists are interested in clustering galaxies, but a noisy background of field galaxies interferes with traditional clustering techniques. Figure 7 demonstrates the clusters found by using the Cuevas-Febrero-Fraiman algorithm. Other examples include detection of minefields [Dasgupta and Raftery, 1998], detection of seismic faults [Dasgupta and Raftery, 1998], robust estimation of covariance [Byers and Raftery, 1998], and identification of foci of activation in the brain [Pereira et al., 2001].

The algorithm proposed by Cuevas, Febrero and Fraiman [Cuevas et al., 2000], while originally developed to estimate the number of clusters, can also be applied to the problem of finding clusters against a noisy background as we will show. We will hereafter abbreviate the Cuevas, Febrero and Fraiman algorithm to be the CFF algorithm.

2 The CFF Algorithm

One common definition, originated by Hartigan [Hartigan, 1975], of the number of clusters in a dataset is the number of connected components of the level set $\{f > c\}$, where f is a density function on $\Re^d$ and c is a positive constant. In [Cuevas et al., 2000], CFF describes a two step algorithm that takes three parameters, a bandwidth h, a density threshold c, and a link-length ϵ:

- **Step One:** Find the set of high density datapoints $\{X_i \ \epsilon \ \text{Dataset} \mid \hat{f}_h(X_i) > c\}$, where $\hat{f}_h$ is a kernel density estimator of f with bandwidth h.
- **Step Two:** Construct the graph in which the vertices are the high density datapoints and in which the nodes X_i and X_j are linked together if and only if $|X_i - X_j| \ \leq \epsilon$, where $|X_i - X_j|$ is the Euclidean distance. Find the connected components of this graph.

The second step forms a set of spanning trees with the X_is in which the edges are of length ϵ or less. In fact, the algorithm proposed by CFF forms a Minimum Spanning Tree since in that algorithm the next point to be considered is the closest point to the members of the current connected component.

While the algorithm is conceptually simple, it suffers from computational issues, especially when there are many datapoints and more than one dimension. Nonparametric density estimation is expensive. Euclidean Minimum Spanning Trees (EMSTs) can be determined in $O(nlogn)$ time in two dimensions using Delauney triangulations [Preparata and Shamos, 1985], where n is the number of datapoints. However, this technique does not scale well as the dimensionality increases. We propose a fast version of the CFF algorithm by addressing the computational issues in both of these steps.

2.1 Scaling Step One: Nonparametric Density Estimation

This paper will pay relatively little attention to Step One for two reasons. First, although it is critical to the scalability of the method, it can be addressed using technology to be reported in a separate paper [Gray and Moore, 2002]. That paper describes how two ideas can allow near linear time detection of high density points. The first idea uses a dual tree algorithm similar to that described in [Gray and Moore, 2001] while the second idea cuts off the search early without computing exact densities in a manner similar to the single ball-tree approach of [Moore, 2000]

The second reason we will pay no more attention to Step One is that frequently the CFF algorithm is executed with a sequence of hundreds of different values of c and ϵ but with the same bandwidth h and thus the same density estimate $\hat{f}$. For this scenario the speed of Step One is much less critical than Step Two, since provided we can obtain numerical density estimates for all datapoints (again, using a new all-kernel algorithm to be described in [Gray and Moore, 2002]), we must run Step 2 hundreds of times for each execution of Step One.

2.2 Scaling Step Two: Connected Components

The CFF algorithm for finding connected components does not address the issue of obtaining the distances between points when forming the Minimum Spanning Tree. The initial improvement over the CFF method is straightforward. We simply exploit recent results in computational geometry for efficient EMSTs. This is based on the GeoMST2 algorithm described in [Narasimhan et al., 2000]. One of the key concepts behind GeoMST2 is the use of Well-Separated Pair Decompositions, which were developed by Callahan [Callahan, 1995], [Callahan and Kosaraju, 1995]. We will now outline the GeoMST2 algorithm, after which we will introduce two additional speedups that are specific to the CFF algorithm and gain at least one extra order of magnitude speed, especially in higher dimensions.

3 Well-Separated Pair Decomposition

Figure 14 illustrates a Well-Separated Pair. Let A and B be point sets in $\Re^d$ with bounding hyper-rectangles R_A and R_B respectively. The notation Diam(R_A) indicates the diameter of the hyper-rectangle R_A. MargDistance(A,B) is defined to be the minimum distance between the hyper-rectangles R_A and R_B. The point sets A and B are considered to be well-separated [Callahan and Kosaraju, 1995] if MargDistance(A,B) $\geq \max\{\mathrm{Diam}(R_A), \mathrm{Diam}(R_B)\}$ [Narasimhan et al., 2000]. The *interaction product* between two point sets A and B is defined as:

$$A \otimes B = \{\{p, p'\} |\ p \in A, p' \in B, \text{ and } p \neq p'\}$$

A set of pairs $(A_1, B_1), \ldots, (A_k, B_k)$ is said to be a well-separated realization of $A \otimes B$ if the following properties [Callahan, 1995] hold:

1. $A_i \subseteq A$ and $B_i \subseteq B$ for all $i = 1, \ldots, k$.
2. $A_i \cap B_i = \emptyset$ for all $i = 1, \ldots, k$.
3. $(A_i \otimes B_i) \cap (A_j \otimes B_j) = \emptyset$ for all i, j such that $i \neq j$
4. $A \otimes B = \bigcup_{i=1}^{k} A_i \otimes B_i$
5. A_i and B_i are well-separated for all $i = 1, \ldots, k$.

Notice that $A_1 \otimes B_1, \ldots, A_k \otimes B_k$ is a partitioning of the large set consisting of all pairs of points $(a \in A, b \in B)$. Every pair of points exists in exactly one (A_i, B_i). And in addition, every (A_i, B_i) has the property of being well-separated.

If P is a point set in $\Re^d$ then a Well-Separated Pair Decomposition, hereafter abbreviated to WSPD, of P is a set of pairs $(A_i, B_i), \ldots, (A_k, B_k)$ which is a well-separated realization of $P \otimes P$. Somewhat astonishingly, a WSPD of a point set P of size n can be constructed with only $O(n)$ elements and in only $O(nlogn)$ time [Callahan and Kosaraju, 1995]. This decomposition is performed using a fair split tree [Callahan and Kosaraju, 1995], which is essentially a k-d tree.

Figures 9 to 12 illustrate a WSPD of the data set with five datapoints shown in Figure 8. This decomposition was formed using a fair split tree. The WSPD contains four pairs. The members in each pair of the WSPD are drawn using a large black circle and a light gray circle. The bounding hyper-rectangles of the pairs in the WSPD are also shown.

4 GeoMST2

The GeoMST2 algorithm [Narasimhan et al., 2000] is an improvement on the GeoMST algorithm [Callahan and Kosaraju, 1993] and [Eppstein, 1999]. Both algorithms use WSPDs in order to address the issue of forming EMSTs in higher dimensions. GeoMST2 differs from its predecessor by reducing the number of Bichromatic Closest Pair (BCP) distance calculations. The BCP distance of two point sets A_i and B_i is defined to be the shortest distance between a and b where $a \in A_i$ and $b \in B_i$. GeoMST2 has an expected running time of $O(nlogn)$.

The algorithm is best explained by looking at the pseudocode in Figure 13. Line 1 forms the WSPD of a point set S. The edges of the final EMST are selected from among the pairs of the WSPD. In lines 3-5, GeoMST2 inserts each pair of the WSPD into a priority queue, using the MargDistance as the priority. In order to form a EMST correctly, lines 6-14 must remove the pairs of the priority queue in the order of ascending BCP distance. However, in line 4, the priority used for the queue is the MargDistance, which is a lower bound for the BCP distance and it is

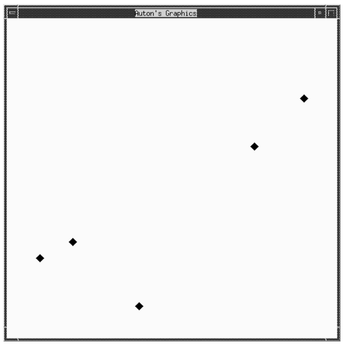

Figure 8: The original dataset

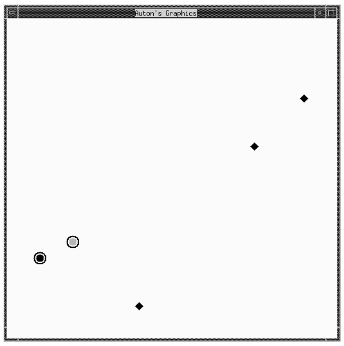

Figure 9: WSPD Pair 1

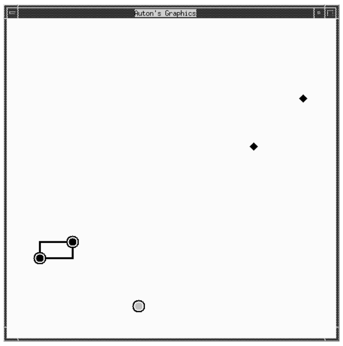

Figure 10: WSPD Pair 2

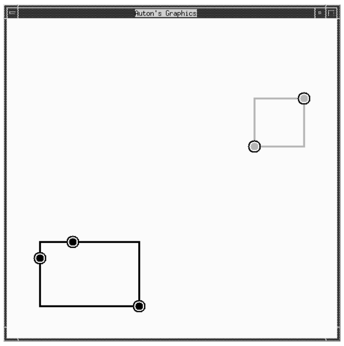

Figure 11: WSPD Pair 3

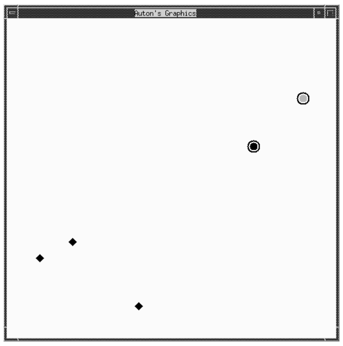

Figure 12: WSPD Pair 4

```
Line #
0......GeoMST2(S)
1......   {(A1,B1),...,(Am,Bm)} = WSPD(S)
2......   E = 0, PQ = 0
3......   for i = 1 to m do
4......      PQ.Insert(i,MargDistance(Ai,Bi))
5......      ComputedBCP[i] = FALSE
6......   while(not PQ.Empty()) do
7......      i = PQ.ExtractMin()
8......      if( Ai and Bi are not connected )
9......         if( ComputedBCP[i] )
10.....            E = E Union {(ai,bi)}
11.....         else
12.....            {ai,bi,D} = BCP(Ai,Bi,Infinity)
13.....            PQ.Insert(i,D)
14.....            ComputedBCP[i] = true
15.....   T = (S,E)
16.....   return T
```

Figure 13: The GeoMST2 algorithm

also much cheaper to compute. We will shortly explain how GeoMST2 does indeed produce a EMST.

Let the element currently removed be the pair (A_i, B_i). Line 8 determines if A_i and B_i are already connected. If they are connected, then the pair is ignored and we have possibly saved ourselves a BCP calculation. Otherwise, line 9 determines if the BCP distance has been calculated. In the case that the BCP distance has not been computed, Lines 12-14 calculate the BCP pair and distance. The Well-Separated Pair is reinserted into the priority queue with the actual BCP distance as the priority. Line 14 flips the ComputedBCP flag to be true for that pair. If the BCP distance has been computed, as shown in line 10, an edge is formed between the element $a \epsilon A_i$ and the element $b \epsilon B_i$ that yields the BCP distance.

There are two crucial points for the correctness of GeoMST2. First of all, Line 13 reinserts a pair into the priority queue when the BCP distance has been calculated; this BCP distance will be at least as large as the MargDistance. Secondly, an edge is only added to the MST if the actual BCP distance has been computed. These two points guarantee that an edge is added to the EMST only when all shorter edges have already been handled.

4.1 CFF-specific optimizations

A naive implementation would use GeoMST2 to calculate the minimum spanning tree until the distance obtained from the priority queue was greater than ϵ. This is wasteful for us because all we need are ϵ-clusters, i.e. clusters formed by joining all points that are within ϵ distance or less from each other. A point may be joined to a cluster through any link of length less than ϵ—not necessarily the link that would appear in the EMST. This observation can be exploited by modifying the step that finds the WSPD while leaving the rest of the GeoMST2 algorithm intact. If we can reduce the number of pairs that are in the WSPD, then GeoMST2 will be much faster due to less elements in the priority queue, as will be shown in Figures 18 and 19.

4.1.1 Optimization 1: Ignoring links that are $> \epsilon$

The main purpose of the WSPD is to reduce the number of edges under consideration when forming the MST. Every pair (A_i, B_i) in the set of pairs of the WSPD becomes an edge in the EMST unless it joins two already connected components. If the minimum distance separating the bounding hyper-rectangles of the sets A_i and B_i is greater than ϵ, as illustrated in Figure 15, then an edge of length ϵ or less cannot possibly exist between a point in A_i to a point in B_i. While forming the WSPD, we need not include any (A_i, B_i) IPs with separation distance greater than ϵ.

4.1.2 Optimization 2: Joining all elements that are within ϵ distance of each other

If the maximum distance separating the bounding hyper-rectangles of A_i and B_i is less than ϵ, then the points in A_i and B_i must belong to the same ϵ-cluster. We can simply join all the points in A_i and B_i if they are not already connected. Since we need not form a EMST, we can connect the points in any order. Furthermore, we need not add such pairs that have this property to the WSPD.

4.1.3 Summary

These two optimizations speed up two crucial areas of the GeoMST algorithm. First of all, they reduce the amount of time required to produce the WSPD. Secondly, the number of well-separated pairs is reduced, thereby speeding up the later half of the GeoMST2 algorithm by reducing the size of the priority queue.

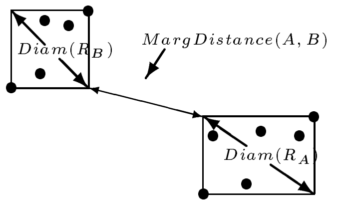

Figure 14: A Well-Separated Pair

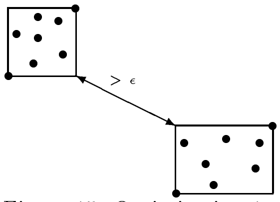

Figure 15: Optimization 1

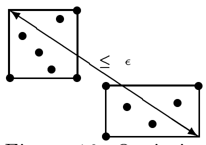

Figure 16: Optimization 2

5 Results

Experiments were conducted on a subset of data taken from the Sloan Digital Sky Survey [Sloan Digital Sky Survey, 2000], which is a digital map of the sky. The dataset used had 7 columns, consisting of four colors, two sky coordinates, and a redshift value. We performed most of our experiments on the four color values.

Figure 17 compares the time taken to perform the clustering step of the CFF algorithm by using Kruskal's MST algorithm, GeoMST2, and our new implementation called ϵ-clustering. Modifications were made to both Kruskal's algorithm and GeoMST2 to terminate the algorithm the moment an edge that was greater than ϵ was encountered. The experiments were run on 4 dimensional datasets with 200 to 1000 records, in increments of 200. Since Kruskal's algorithm is non-competitive as the number of datapoints are above 10000, we compare the speeds of GeoMST2 and ϵ-clustering on datasets of size 20000 to 100000, in increments of 20000 records. These experiments were run on 3 and 4 dimensional data. Figures 18 and 19 show the results. ϵ-clustering outperforms GeoMST2 in all the cases.

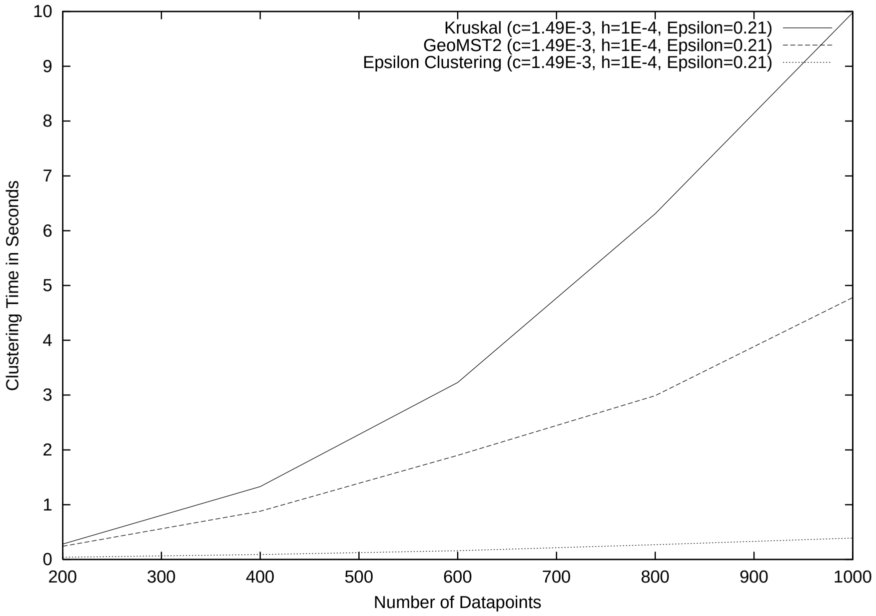

Figure 17: GeoMST2 vs. ϵ-Clustering vs Kruskal in 4D

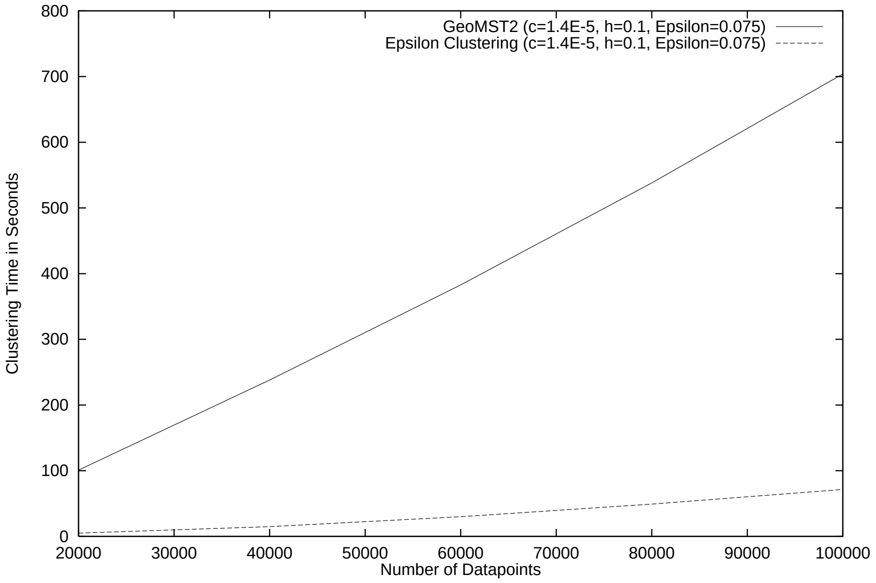

Figure 18: GeoMST2 vs. ϵ-Clustering in 3D

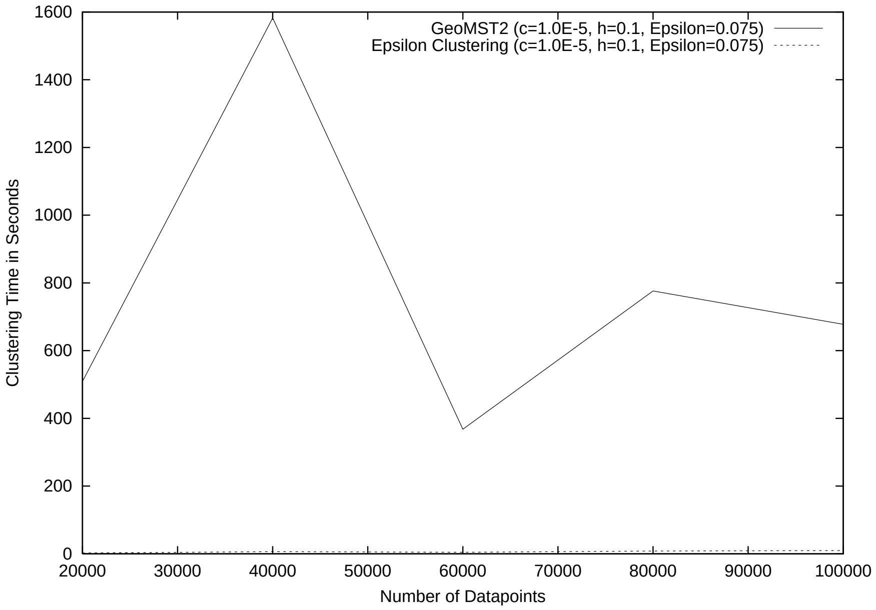

Figure 19: GeoMST2 vs. ϵ-Clustering in 4D

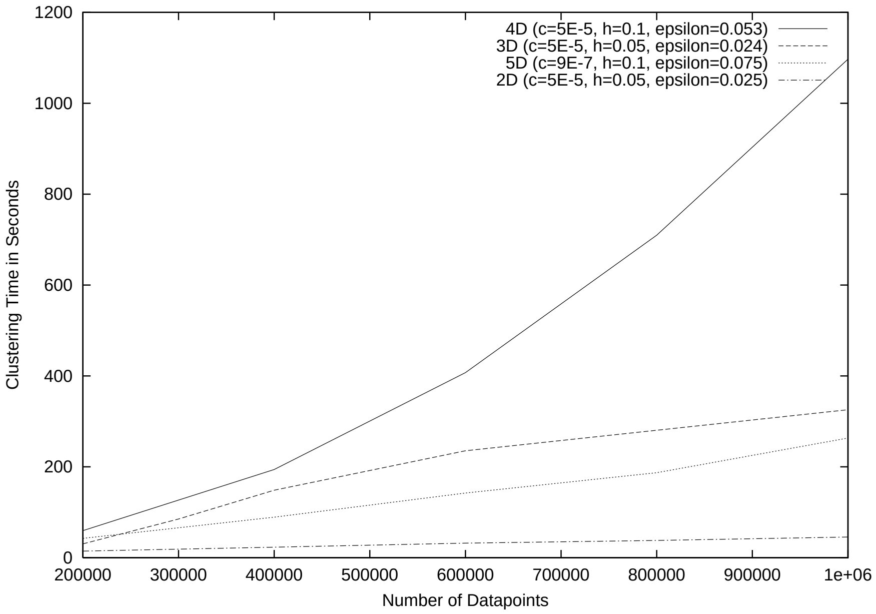

Figure 20: Change in time as Number of Datapoints increases

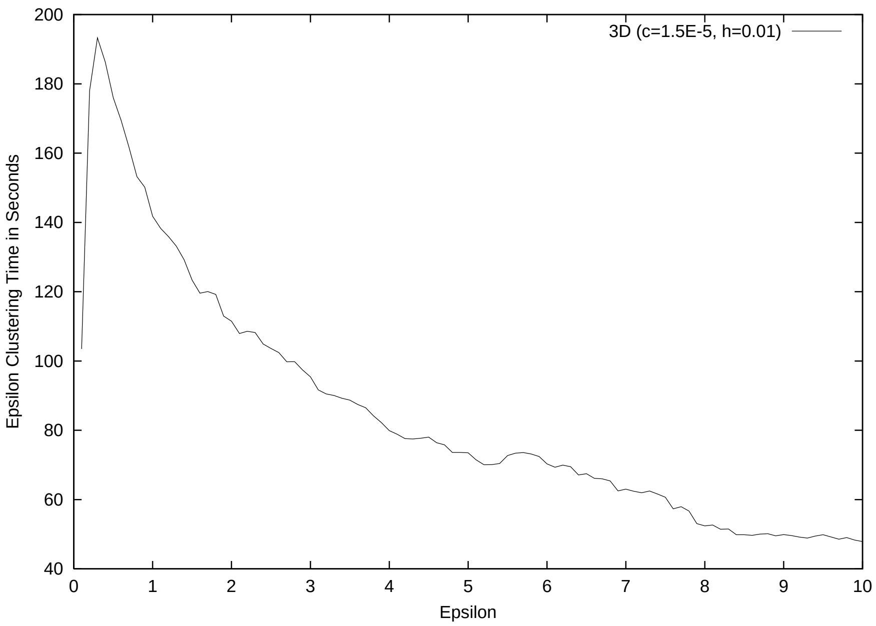

Figure 21: Change in Time as a Function of ϵ for 3D data

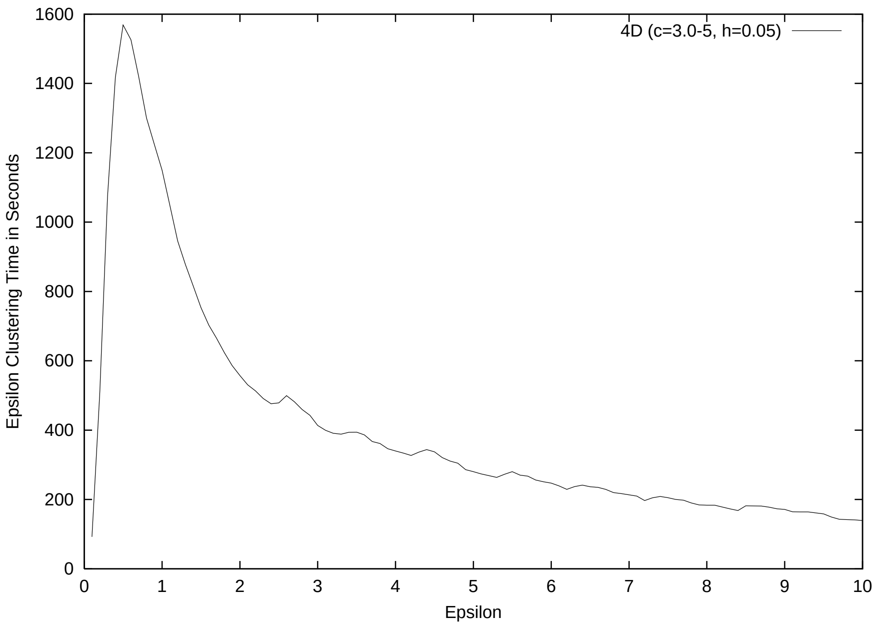

Figure 22: Change in Time as a Function of ϵ for 4D data

Figure 20 shows the effects of increasing the number of datapoints on the time taken by ϵ-clustering. We ran experiments on datasets of size 200000 to 1000000, in increments of 200000. The experiments were performed on data in 2 to 5 dimensions. For certain settings of the parameters, the time increases linearly, while for others, time appears to increase on the order of $O(nlogn)$.

These experiments indicate that the bandwidth h, threshold c, and ϵ have an impact on the running time. h and c affect the number of high density datapoints. The ϵ parameter affects the time spent in the WSPD step. If ϵ is very large, the WSPD step ends early as all the points in the fair split tree node can be connected. Similarly, a small value of ϵ results in more pairs that are separated by a distance greater than ϵ. Figures 21 and 22 illustrate the effect of varying ϵ on time on a 3-dimensional and a 4-dimensional dataset with 100000 records each. The time seems to peak at about a value of 0.5 for ϵ for both plots.

6 Related Work

Banfield and Raftery [Banfield and Raftery, 1993] use a mixture-model approach to perform clustering on d-dimensional data with clutter. The data is assumed to be generated from a mixture of Gaussians together with a homogeneous spatial Poisson process which produces the clutter. The clusters are assumed to be highly linear or piecewise linear, where linear means that they are concentrated along their first principal component. Clustering is performed by maximizing the parameters and the partitioning of the data over the classification likelihood. Their approach, however, requires the user to provide information about the shape of the clusters and the algorithm's approximation of the number of clusters in the data performs poorly when the clutter makes up a majority of the data. Dasgupta and Raftery [Dasgupta and Raftery, 1998] extend on the work by Banfield and Raftery. The technique proposed by [Dasgupta and Raftery, 1998] iterates over a range of possible number of clusters for the data. In each iteration, an initial clustering is performed using the method proposed by Banfield and Raftery. Then, an EM step follows in which the maximized mixture likelihood is obtained upon convergence and this value is used to compute the Bayesian Information Criterion score for the model. The EM step is also modified to estimate the shape of the clusters from the data. The best model, along with its clustering, is selected.

Byers and Raftery [Byers and Raftery, 1998] propose a further algorithm that is not restricted by the shape or the number of the clusters. Once again, the clutter is assumed to be a homogeneous Poisson point process, but the features are also considered as a Poisson process on top of the clutter. This approach computes, for every data point, the kth nearest neighbor distance, where k is a parameter supplied by the user. This distance is modeled as a mixture of two gamma distributions, one for the clutter and the other for the features. The next step of the algorithm applies EM to estimate the parameters of the mixture. From the results of EM, a data point can be classified as clutter if its probability of belonging to the clutter component is higher than that of belonging to the feature component and vice versa.

7 Conclusion

ϵ-clustering is a computationally efficient algorithm for determining the connected components in Step 2 of the CFF algorithm. Our results show that it outperforms

GeoMST2, one of the fastest EMST algorithms, by nearly an order of magnitude in higher dimensions. Combining ϵ-clustering with the nonparametric density estimation algorithms in [Gray and Moore, 2002] will yield an effective algorithm for identifying clusters against a noisy background with massive amounts of data.

References

[Banfield and Raftery, 1993] Banfield, J. and Raftery, A. E. (1993). Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821.

[Byers and Raftery, 1998] Byers, S. and Raftery, A. E. (1998). Nearest-neighbour clutter removal for estimating features in spatial point processes. *Journal of the American Statistical Association*, 93:577–584.

[Callahan, 1995] Callahan, P. (1995). *Dealing with higher dimensions: the well-separated pair decomposition and its applications.* PhD thesis, Johns Hopkins University, Baltimore, Maryland.

[Callahan and Kosaraju, 1995] Callahan, P. and Kosaraju, S. (1995). A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*, 62(1):67–90.

[Callahan and Kosaraju, 1993] Callahan, P. and Kosaraju, S. R. (1993). Faster algorithms for some geometric graph problems in higher dimensions. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 291–300.

[Cuevas et al., 2000] Cuevas, A., Febrero, M., and Fraiman, R. (2000). Estimating the number of clusters. *The Canadian Journal of Statistics*, 28(2):367–382.

[Dasgupta and Raftery, 1998] Dasgupta, A. and Raftery, A. E. (1998). Detecting features in spatial point processes with clutter via model-based clustering. *Journal of the American Statistical Association*, 93:294–302.

[Eppstein, 1999] Eppstein, D. (1999). Spanning trees and spanners. In Sack, J.-R. and Urrutia, J., editors, *Handbook of Computational Geometry*, pages 425–461. Elsevier Science Publishers B.V., North-Holland,Amsterdam.

[Gray and Moore, 2001] Gray, A. and Moore, A. W. (2001). N-body problems in statistical learning. In Leen, T. K., Dietterich, T. G., and Tresp, V., editors, *Advances in Neural Information Processing Systems 13 (December 2000)*. MIT Press.

[Gray and Moore, 2002] Gray, A. and Moore, A. W. (2002). Efficient kernel density algorithms. In Preparation.

[Hartigan, 1975] Hartigan, J. (1975). *Clustering Algorithms.* John Wiley, New York.

[Moore, 2000] Moore, A. W. (2000). The Anchors Hierarchy: Using the triangle inequality to survive high dimensional data. In *Twelfth Conference on Uncertainty in Artificial Intelligence*. AAAI Press.

[Narasimhan et al., 2000] Narasimhan, G., Zhu, J., and Zachariasen, M. (2000). Experiments with computing geometric minimum spanning trees. In *Proceedings of ALENEX'00*, Lecture Notes in Computer Science, pages 183–196. Springer-Verlag.

[Pereira et al., 2001] Pereira, F., Just, M., and Mitchell, T. (2001). Distinguishing natural language processes on the basis of fmri-measured brain activation. In Raedt, L. D. and Siebes, A., editors, *Principles of Data Mining and Knowledge Discovery*, pages 374–385. Springer-Verlag.

[Preparata and Shamos, 1985] Preparata, F. and Shamos, M. (1985). *Computational Geometry: an Introduction*. Springer-Verlag, New York.

[Reichart et al., 1999] Reichart, D., Nichol, R., Castander, F., Burke, D., Romer, A., Holden, B., Collins, C., and Ulmer, M. (1999). A deficiency of high-redshift, high-luminosity x-ray clusters: Evidence for a high value of omega matter? *The Astrophysical Journal*, 518(2):521–532.

[Sloan Digital Sky Survey, 2000] Sloan Digital Sky Survey (2000). The Sloan Digital Sky Survey. http://www.sdss.org.