DENCLUE 2.0: Fast Clustering based on Kernel Density Estimation

Alexander Hinneburg[1] and Hans-Henning Gabriel[2]

[1] Institute of Computer Science
Martin-Luther-University Halle-Wittenberg, Germany
hinneburg@informatik.uni-halle.de
[2] Otto-von-Guericke-University Magdeburg, Germany
Hans-Henning.Gabriel@web.de

Abstract. The DENCLUE algorithm employs a cluster model based on kernel density estimation. A cluster is defined by a local maximum of the estimated density function. Data points are assigned to clusters by hill climbing, i.e. points going to the same local maximum are put into the same cluster. A disadvantage of DENCLUE 1.0 is, that the used hill climbing may make unnecessary small steps in the beginning and never converges exactly to the maximum, it just comes close.
We introduce a new hill climbing procedure for Gaussian kernels, which adjusts the step size automatically at no extra costs. We prove that the procedure converges exactly towards a local maximum by reducing it to a special case of the expectation maximization algorithm. We show experimentally that the new procedure needs much less iterations and can be accelerated by sampling based methods with sacrificing only a small amount of accuracy.

1 Introduction

Clustering can be formulated in many different ways. Non-parametric methods are well suited for exploring clusters, because no generative model of the data is assumed. Instead, the probability density in the data space is directly estimated from data instances. Kernel density estimation [15,14] is a principled way of doing that task. There are several clustering algorithms, which exploit the adaptive nature of a kernel density estimate. Examples are the algorithms by Schnell [13] and Fukunaga [5] which use the gradient of the estimated density function. The algorithms are also described in the books by Bock [3] and Fukunaga [4] respectively. The DENCLUE framework for clustering [7,8] builds upon Schnells algorithm. There, clusters are defined by local maxima of the density estimate. Data points are assigned to local maxima by hill climbing. Those points which are assigned to the same local maximum are put into a single cluster.

However, the algorithms use directional information of the gradient only. The step size remains fixed throughout the hill climbing. This implies certain disadvantages, namely the hill climbing does not converges towards the local maximum, it just comes close, and the number of iteration steps may be large

due to many unnecessary small steps in the beginning. The step size could be heuristically adjusted by probing the density function at several positions in the direction of the gradient. As the computation of the density function is relatively costly, such a method involves extra costs for step size adjustment, which are not guaranteed to be compensated by less iterations.

The contribution of this article is a new hill climbing method for kernel density estimates with Gaussian kernels. The new method adjusts the step size automatically at no additional costs and converges towards a local maximum. We prove this by casting the hill climbing as a special case of the expectation maximization algorithm. Depending on the convergence criterium, the new method needs less iterations as fixed step size methods. Since the new hill climbing can be seen as an EM algorithm, general acceleration methods for EM, like sparse EM [11] can be used as well. We also explore acceleration by sampling. Fast Density estimation [17] can be combined with our method as well but is not tested in this first study.

Other density based clustering methods beside DENCLUE, which would benefit from the new hill climbing, have been proposed by Herbin et al [6]. Variants of density based clustering are DBSCAN [12], OPTICS [1], and followup versions, which, however, do not use a probabilistic framework. This lack of foundation prevents the application of our new method there.

Related approaches include fuzzy c-means [2], which optimized the location of cluster centers and uses membership functions in a similar way as kernel functions are used by DENCLUE. A subtle difference between fuzzy c-means and DENCLUE is, that in c-means the membership grades of a point belonging to a cluster are normalized, s.t. the weights of a single data point for all clusters sum to one. This additional restriction makes the clusters competing for data points. DENCLUE does not have such restriction. The mountain method [16] also uses similar membership grades as c-means. It finds clusters by first discretizing the data space into a grid, calculates for all grid vertices the mountain function (which is comparable to the density up to normalization) and determines the grid vertex with the maximal mountain function as the center of the dominant cluster. After effects of the dominant cluster on the mountain function are removed, the second dominant cluster is found. The method iterates until the heights of the clusters drop below a predefined percentage of the dominant cluster. As the number of grid vertices grow exponentially in high dimensional data spaces, the method is limited to low dimensional data. Niche clustering [10] uses a non-normalized density function as fitness function for prototype-based clustering in a genetic algorithm. Data points with high density (larger than a threshold) are seen as core points, which are used to estimate scale parameters similar to the smoothing parameter h introduced in the next section.

The rest of the paper is structured as follows. In section 2, we briefly introduce the old DENCLUE framework and in section 3 we propose our new improvements for that framework. In section 4, we compare the old and the new hill climbing experimentally.

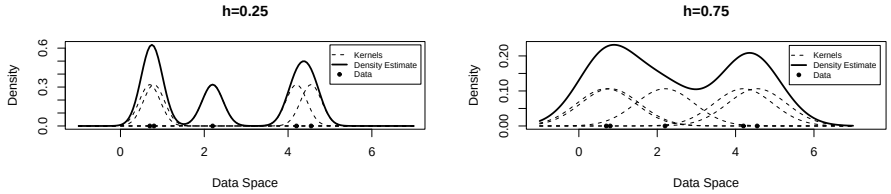

Fig. 1. Kernel density estimate for one-dimensional data and different values for the smoothing parameter h.

2 DENCLUE 1.0 framework for clustering

The DENCLUE framework [8] builds on non-parametric methods, namely kernel density estimation. Non-parametric methods are not looking for optimal parameters of some model, but estimate desired quantities like the probability density of the data directly from the data instances. This allows a more direct definition of a clustering in contrast to 'parametric methods, where a clustering corresponds to an optimal parameter setting of some high-dimensional function. In the DENCLUE framework, the probability density in the data space is estimated as a function of all data instances $\boldsymbol{x}_t \in X \subset \mathbb{R}^d, d \in \mathbb{N},\ t = 1, \ldots, N$. The influences of the data instances in the data space are modeled via a simple kernel function, e.g. the Gaussian kernel $K(\boldsymbol{u}) = (2\pi)^{-\frac{d}{2}} \cdot \exp\left[-\frac{\boldsymbol{u}^2}{2}\right]$. The sum of all kernels (with suitable normalization) gives an estimate of the probability at any point $\boldsymbol{x}$ in the data space $\hat{p}(\boldsymbol{x}) = {}^1/_{(Nh^d)} \sum_{t=1}^{N} K\left({}^{\boldsymbol{x}-\boldsymbol{x}_t}/_h\right)$. The estimate $\hat{p}(\boldsymbol{x})$ enjoys all properties like differentiability like the original kernel function. The quantity $h > 0$ specifies to what degree a data instance is smoothed over data space. When h is large, an instance stretches its influence up to more distant regions. When h is small, an instance effects only the local neighborhood. We illustrate the idea of kernel density estimation on one-dimensional data as shown in figure 1.

A clustering in the DENCLUE framework is defined by the local maxima of the estimated density function. A hill-climbing procedure is started for each data instance, which assigns the instance to a local maxima. In case of Gaussian kernels, the hill climbing is guided by the gradient of $\hat{p}(\boldsymbol{x})$, which takes the form

$$\nabla \hat{p}(\boldsymbol{x}) = \frac{1}{h^{d+2}N} \sum_{t=1}^{N} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_t}{h}\right) \cdot (\boldsymbol{x}_t - \boldsymbol{x}). \tag{1}$$

The hill climbing procedure starts at a data point and iterates until the density does not grow anymore. The update formula of the iteration to proceed from $\boldsymbol{x}^{(l)}$ to $\boldsymbol{x}^{(l+1)}$ is

$$\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)} + \delta \frac{\nabla \hat{p}(\boldsymbol{x}^{(l)})}{\|\nabla \hat{p}(\boldsymbol{x}^{(l)})\|_2}. \tag{2}$$

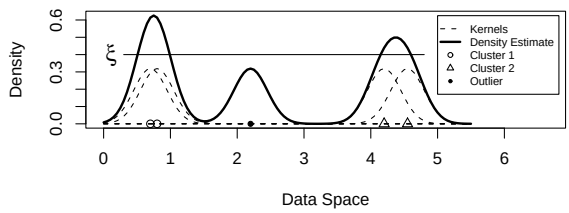

Fig. 2. Example of a DENCLUE clustering based on a kernel density estimate and a noise threshold ξ.

The step size δ is a small positive number. In the end, those end points of the hill climbing iteration, which are closer than 2δ are considered, to belong to the same local maximum. Instances, which are assigned to the same local maximum, are put into the same cluster.

A practical problem of gradient based hill climbing in general is the adaptation of the step size. In other words, how far to follow the direction of the gradient? There are several general heuristics for this problem, which all need to calculate $\hat{p}(\boldsymbol{x})$ several times to decide a suitable step size.

In the presence of random noise in the data, the DENCLUE framework provides an extra parameter $\xi > 0$, which treats all points assigned to local maxima $\hat{\boldsymbol{x}}$ with $\hat{p}(\hat{\boldsymbol{x}}) < \xi$ as outliers. Figure 2 sketches the idea of a DENCLUE clustering.

3 DENCLUE 2.0

In this section, we propose significant improvements of the DENCLUE 1.0 framework for Gaussian kernels. Since the choice of the kernel type does not have large effects on the results in the typical case, the restriction on Gaussian kernels is not very serious. First, we introduce a new hill climbing procedure for Gaussian kernels, which adjust the step size automatically at no extra costs. The new method does really converge towards a local maximum. We prove this property by casting the hill climbing procedure as an instance of the expectation maximization algorithm. Last, we propose sampling based methods to accelerate the computation of the kernel density estimate.

3.1 Fast Hill Climbing

The goal of a hill climbing procedure is to maximize the density $\hat{p}(\boldsymbol{x})$. An alternative approach to gradient based hill climbing is to set the first derivative of $\hat{p}(\boldsymbol{x})$ to zero and solve for $\boldsymbol{x}$. Setting (1) to zero and rearranging we get

$$\boldsymbol{x} = \frac{\sum_{t=1}^{N} K\big(\frac{\boldsymbol{x}-\boldsymbol{x}_t}{h}\big)\boldsymbol{x}_t}{\sum_{t=1}^{N} K\big(\frac{\boldsymbol{x}-\boldsymbol{x}_t}{h}\big)} \tag{3}$$

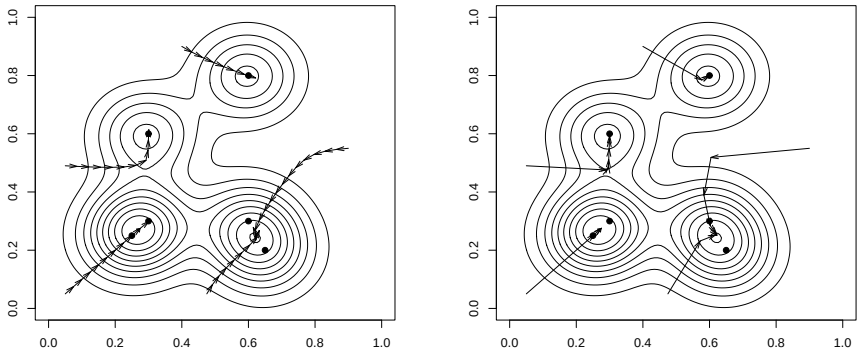

Fig. 3. (left) Gradient hill climbing as used by DENCLUE 1.0, (right) Step size adjusting hill climbing used by DENCLUE 2.0.

Obviously, this is not a solution for $\boldsymbol{x}$, since the vector is still involved into the righthand side. Since $\boldsymbol{x}$ influences the righthand side only through the kernel, the idea is to compute the kernel for some fixed $\boldsymbol{x}$ and update the vector on the lefthand side according to formula (3). This give a new iterative procedure with the update formula

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t=1}^{N} K\big(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h}\big)\boldsymbol{x}_t}{\sum_{t=1}^{N} K\big(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h}\big)} \tag{4}$$

The update formula can be interpreted as a normalized and weighted average of the data points and the weights of the data points depend on the influence of their kernels on the current $\boldsymbol{x}^{(l)}$. In order to see that the new update formula makes sense it is interesting to look at the special case $N = 1$. In that case, the estimated density function consists just of a single kernel and the iteration jumps after one step to $\boldsymbol{x}^1$, which is the maximum.

The behavior of DENCLUEs 1.0 hill climbing and the new hill climbing procedure is illustrated in figure 3. The figure shows that the step size of the new procedure is adjusted to the shape of the density function. On the other hand, an iteration of the new procedure has the same computational costs as one of the old gradient based hill climbing. So, adjusting the step size comes at no additional costs. Another difference is, that the hill climbing of the new method really converges towards a local maximum, while the old method just comes close.

Since the new method does not need the step size parameter δ, the assignment of the instances to clusters is done in a new way. The problem is to define a heuristic, which automatically adjusts to the scale of distance between the converged points.

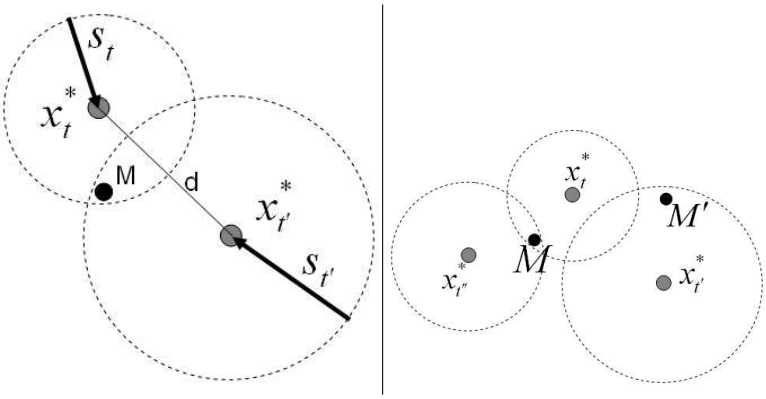

Fig. 4. (left) Assignment to a local maximum, (right) Ambiguous assignment. The points M and M' denote the true but unknown local maxima.

A hill climbing is started at each data point $\boldsymbol{x}_t \in X$ and iterates until the density does not change much, i.e. $[\hat{f}(\boldsymbol{x}_t^{(l)}) - \hat{f}(\boldsymbol{x}_t^{(l-1)})] / \hat{f}(\boldsymbol{x}_t^{(l)}) \leq \epsilon$. An end point reached by the hill climbing is denoted by $\boldsymbol{x}_t^* = \boldsymbol{x}_t^{(l)}$ and the sum of the k last step sizes is $s_t = \sum_{i=1}^{k} \|\boldsymbol{x}_t^{(l-i+1)} - \boldsymbol{x}_t^{(l-i)}\|_2$. The integer k is parameter of the heuristic. We found that $k = 2$ worked well for all experiments. Note, that the number of iterations may vary between the data points, however, we restricted the number of iterations to be larger than k. For appropriate $\epsilon > 0$, it is safe to assume that the end points $\boldsymbol{x}_t^*$ are close to the respective local maxima. Typically, the step sizes are strongly shrinking before the convergence criterium is met. Therefore, we assume that the true local maximum is within a ball around $\boldsymbol{x}_t^*$ of radius s_t. Thus, the points belonging to the same local maximum have end points $\boldsymbol{x}_t^*$ and $\boldsymbol{x}_{t'}^*$, which are closer than $s_t + s_{t'}$. Figure 4 left illustrates that case.

However, there might exists rare cases, when such an assignment is not unique. This happens when for three end points $\boldsymbol{x}_t^*, \boldsymbol{x}_{t'}^*$ and $\boldsymbol{x}_{t''}^*$ hold the following conditions $\|\boldsymbol{x}_t^* - \boldsymbol{x}_{t'}^*\| \leq s_t + s_{t'}$ and $\|\boldsymbol{x}_t^* - \boldsymbol{x}_{t''}^*\| \leq s_t + s_{t''}$ but not $\|\boldsymbol{x}_{t'}^* - \boldsymbol{x}_{t''}^*\| \leq s_{t'} + s_{t''}$. In order to solve the problem, the hill climbing is continued for all points, which are involved in such situations, until the convergence criterium is met for some smaller ϵ (a simple way to reduce ϵ is multiply it with a constant between zero and one). After convergence is reached again, the ambiguous cases are rechecked. The hill climbing is continued until all such cases are solved. Since further iterations causes the step sizes to shrink the procedure will stop at some point. The idea is illustrated in figure 4 right.

However, until now it is not clear why the new hill climbing procedure converges towards a local maximum. In the next section, we prove this claim.

3.2 Reduction to Expectation Maximization

We prove the convergence of the new hill climbing method by casting the maximization of the density function as a special case of the expectation maximization framework [9]. When using the Gaussian kernel we can rewrite the kernel density estimate $\hat{p}(\boldsymbol{x})$ in the form of a constrained mixture model with Gaussian components

$$p(\boldsymbol{x}|\boldsymbol{\mu},\sigma) = \sum_{t=1}^{N} \pi_t \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma) \tag{5}$$

and the constraints $\pi_t = 1/N$, $\boldsymbol{\mu}_t = \boldsymbol{x}_t$ ($\boldsymbol{\mu}$ denotes a vector consisting of all concatenated $\boldsymbol{\mu}_t$), and $\sigma = h$. We can think of $p(\boldsymbol{x}|\boldsymbol{\mu},\sigma)$ as a likelihood of $\boldsymbol{x}$ given the model determined by $\boldsymbol{\mu}$ and σ. Maximizing $\log p(\boldsymbol{x}|\boldsymbol{\mu},\sigma)$ wrt. $\boldsymbol{x}$ is not possible in a direct way. Therefore, we resort to the EM framework by introducing a hidden bit variable $\boldsymbol{z} \in \{0,1\}^N$ with $\sum_{t=1}^{N} z_t = 1$ and

$$z_t = \begin{cases} 1 & \text{if the density at } \boldsymbol{x} \text{ is explained by } \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma) \text{ only} \\ 0 & \text{else} \end{cases}. \tag{6}$$

The complete log-likelihood is $\log p(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{\mu},\sigma) = \log p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\mu},\sigma)p(\boldsymbol{z})$ with $p(\boldsymbol{z}) = \prod_{t=1}^{N} \pi_t^{z_t}$ and $p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\mu},\sigma) = \prod_{t=1}^{N} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma)^{z_t}$.

In contrast to generative models, which use EM to determine parameters of the model, we maximize the complete likelihood wrt. $\boldsymbol{x}$. The EM-framework ensures that maximizing the complete log-likelihood maximizes the original log-likelihood as well. Therefore, we define the quantity

$$\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)}) = E[\log p(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{\mu},\sigma)|\boldsymbol{\mu},\sigma,\boldsymbol{x}^{(l)}] \tag{7}$$

In the E-step the expectation $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)})$ is computed wrt. to $\boldsymbol{z}$ and $\boldsymbol{x}^{(l)}$ is put for $\boldsymbol{x}$, while in the M-step $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)})$ is taken as a function of $\boldsymbol{x}$ and maximized. The E-step boils down to compute the posterior probability for the z_t:

$$E[z_t|\boldsymbol{\mu},\sigma,\boldsymbol{x}^{(l)}] = p(z_t = 1|\boldsymbol{x}^{(l)},\boldsymbol{\mu},\sigma) \tag{8}$$

$$= \frac{p(\boldsymbol{x}^{(l)}|z_t = 1,\boldsymbol{\mu},\sigma)p(z_t = 1|\boldsymbol{\mu},\sigma)}{\sum_{t=1}^{N} p(\boldsymbol{x}^{(l)}|z_t = 1,\boldsymbol{\mu},\sigma)p(z_t = 1|\boldsymbol{\mu},\sigma)} \tag{9}$$

$$= \frac{1/N \cdot \mathcal{N}(\boldsymbol{x}^{(l)}|\boldsymbol{\mu}_t,\sigma)}{\sum_{t'=1}^{N} 1/N \cdot \mathcal{N}(\boldsymbol{x}^{(l)}|\boldsymbol{\mu}_{t'},\sigma)} \tag{10}$$

$$= \frac{1/N \cdot K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})}{\hat{p}(\boldsymbol{x}^{(l)})} = \theta_t \tag{11}$$

In the M-step, z_t is replaced by the fixed posterior θ_t, which yields $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)}) = \sum_{t=1}^{N} \theta_t[\log 1/N + \log \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma)]$. Computing the derivative wrt. $\boldsymbol{x}$ and setting it to zero yields $\sum_{t=1}^{N} \theta_t \sigma^{-2}(\boldsymbol{x} - \boldsymbol{\mu}_t) = 0$ and thus

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t=1}^{N} \theta_t \mu_t}{\sum_{t=1}^{N} \theta_t} = \frac{\sum_{t=1}^{N} K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})\boldsymbol{x}_t}{\sum_{t=1}^{N} K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})} \tag{12}$$

By starting the EM with $\boldsymbol{x}^{(0)} = \boldsymbol{x}_t$ the method performs an iterative hill climbing starting at data point $\boldsymbol{x}_t$.

3.3 Sampling based Acceleration

As the hill climbing procedure is a special case of the expectation maximization algorithm, we can employ different general acceleration techniques known for EM to speed up the the DENCLUE clustering algorithm.

Most known methods for EM, try to reduce the number of iterations needed until convergence [9]. Since the number of iterations is typically quite low, that kind of techniques yield no significant reduction for the clustering algorithm.

In order to speed up the clustering algorithm, the costs for the iterations itself should be reduced. One option is sparse EM [11], which still converges to the true local maxima. The idea is to freeze small posteriors for several iterations, so only the $p\%$ largest posteriors are updated in each iteration. As the hill climbing typically needs only a few iterations we modify the hill climbing starting at the single point $\boldsymbol{x}^{(0)}$ as follows. All kernels $K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})$ are determined in the initial iteration and $\boldsymbol{x}^{(1)}$ is determined as before. Let be U the index set of the $p\%$ largest kernels and L the complement. Then, in the next iterations the update formula is modified to

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t\in U} K(\frac{\boldsymbol{x}^{(l)}-\boldsymbol{x}_t}{h})\boldsymbol{x}_t + \sum_{t\in L} K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})\boldsymbol{x}_t}{\sum_{t\in U} K(\frac{\boldsymbol{x}^{(l)}-\boldsymbol{x}_t}{h}) + \sum_{t\in L} K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})} \tag{13}$$

The index set U and L can be computed by sorting. The disadvantage of the method is, that the first iteration is still the same as in the original EM.

The original hill climbing converges towards a true local maximum of the density function. However, we does not need the exact position of such a maximum. It is sufficient for the clustering algorithm, that all points of a cluster converge to the same local maximum, regardless where that location might be. In that light, it makes sense to simplify the original density function by reducing the data set to a set of $p\%$ representative points. That reduction can be done in many ways. We consider here random sampling and k-means. So the number of points N is reduced to a much smaller number of representative points N', which are used to construct the density estimate.

Note that random sampling has much smaller costs as k-means. We investigate in the experimental section, whether the additional costs by k-means pay off by less needed iterations or by cluster quality.

4 Experimental Evaluation

We compared the new step size adjusting (SSA) hill climbing method with the old fixed step size hill climbing. We used synthetic data with normally distributed 16-dimensional clusters with uniformly distributed centers and approximately same size. Both methods are tuned to find the perfect clustering in the most

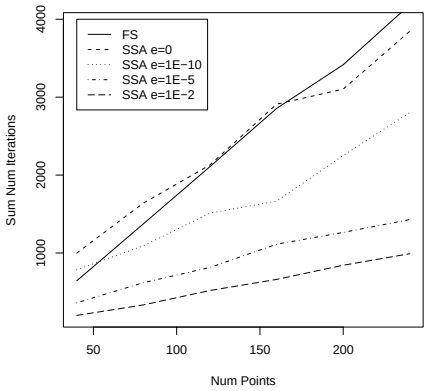

Fig. 5. Number of data points versus the total sum of numbers of iterations.

efficient way. The total sum of numbers of iterations for the hill climbings of all data points is plotted versus the number of data points. SSA was run with different values for ϵ, which controls the convergence criterium of SSA. Figure 5 clearly shows that SSA ($\epsilon = 0.01$) needs only a fraction of the number of iterations of FS to achieve the same results. The costs per iterations are the same for both methods.

Next, we tested the influence of different sampling methods on the computational costs. Since the costs per iteration differ for sparse EM, we measure the costs in number of kernel computations versus sample size. Figure 6(left) shows that sparse EM is more expensive than random sampling and k-means based data reduction. The difference between the two latter methods is negligible, so the additional effort of k-means during the data reduction does not pay off in less computational costs during the hill climbing. For sample size 100% the methods converge to the original SSA hill climbing.

For random sampling, we tested sample size versus cluster quality measured by normalized mutual information (NMI is one if the perfect clustering is found). Figure 6(right) shows that the decrease of cluster quality is not linear in sample size. So, a sample of 20% is still sufficient for a good clustering when the dimensionality is $d = 16$. Larger dimensionality requires larger samples as well as more smoothing (larger h), but the clustering can still be found.

In the last experiment, we compared SSA, its sampling variants, and k-means with the optimal k on various real data sets from the machine learning repository wrt. cluster quality. Table 1 shows average values of NMI with standard deviation for k-means and sampling, but not for SSA which is a deterministic algorithm.

SSA has better or comparable cluster quality as k-means. The sampling variants degrade with smaller sample sizes $(0.8, 0.4, 0.2)$, but k-means based data reduction suffers much less from that effect. So, the additional effort of k-means based data reduction pays off in cluster quality.

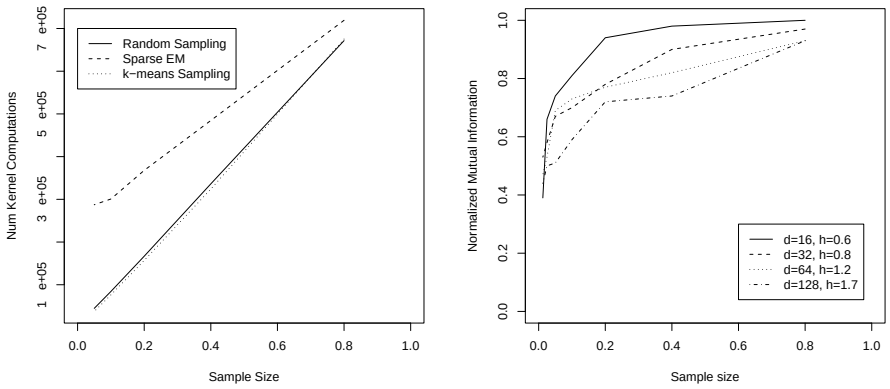

Fig. 6. (left) Sample size versus number of kernel computations, (right) sample size versus cluster quality (normalized mutual information, NMI).

Table 1. NMI values for different data and methods, the first number in the three rightmost columns shows the sample size.

	k-means	**SSA**	**Random Sampling**	**Sparse EM**	**k-means Sampling**
iris	0.69±0.10	0.72	0.8: 0.66±0.05	0.8: 0.68±0.06	0.8: 0.67±0.06
			0.4: 0.63±0.05	0.4: 0.60±0.06	0.4: 0.65±0.07
			0.2: 0.63±0.06	0.2: 0.50±0.04	0.2: 0.64±0.07
ecoli	0.56±0.05	0.67	0.8: 0.65±0.02	0.8: 0.66±0.00	0.8: 0.65±0.02
			0.4: 0.62±0.06	0.4: 0.61±0.00	0.4: 0.65±0.04
			0.2: 0.59±0.06	0.2: 0.40±0.00	0.2: 0.65±0.03
wine	0.82±0.14	0.80	0.8: 0.71±0.06	0.8: 0.72±0.07	0.8: 0.70±0.11
			0.4: 0.63±0.10	0.4: 0.63±0.00	0.4: 0.70±0.05
			0.2: 0.55±0.15	0.2: 0.41±0.00	0.2: 0.58±0.21

In all experiments, the smoothing parameter h was tuned manually. Currently, we are working on methods to determine that parameter automatically. In conclusion, we proposed a new hill climbing method for kernel density functions, which really converges towards a local maximum and adjusts the step size automatically. We believe, that our new technique has some potential for interesting combinations with parametric clustering methods.

References

1. M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *Proceedings SIGMOD'99*, pages 49–60. ACM Press, 1999.
2. J. Bezdek. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing.* Kluwer Academic Pub, 1999.

3. H. H. Bock. *Automatic Classification.* Vandenhoeck and Ruprecht, 1974.
4. K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.
5. K. Fukunaga and L. Hostler. The estimation of the gradient of a density function, with application in pattern recognition. *IEEE Trans. Info. Thy.*, 21:32–40, 1975.
6. M. Herbin, N. Bonnet, and P. Vautrot. Estimation of the number of clusters and influence zones. *Pattern Recognition Letters*, 22:1557–1568, 2001.
7. A. Hinneburg and D. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Proceedings KDD'98*, pages 58–65. AAAI Press, 1998.
8. A. Hinneburg and D. A. Keim. A general approach to clustering in large databases with noise. *Knowledge and Information Systems (KAIS)*, 5(4):387–415, 2003.
9. G. J. McLachlan and T. Krishnan. *EM Algorithm and Extensions.* Wiley, 1997.
10. O. Nasraoui and R. Krishnapuram. The unsupervised niche clustering algorithm: extension tomultivariate clusters and application to color image segmentation. *IFSA World Congress and 20th NAFIPS International Conference,*, 3, 2001.
11. R. M. Neal and G. E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. MIT Press, 1999.
12. J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1997.
13. P. Schnell. A method to find point-groups. *Biometrika*, 6:47–48, 1964.
14. D. Scott. *Multivariate Density Estimation.* Wiley, 1992.
15. B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman & Hall, 1986.
16. R. Yager and D. Filev. Approximate clustering via the mountain method. *IEEE Transactions on Systems, Man and Cybernetics*, 24(8):1279–1284, 1994.
17. T. Zhang, R. Ramakrishnan, and M. Livny. Fast density estimation using cf-kernel for very large databases. In *Proceedings KDD'99*, pages 312–316. ACM, 1999.