BNL-108231-2015-JA



# Density-aware Clustering based on Aggregated Heat Kernel and Its Transformation

Hao Huang, Computer Science Department, Stony Brook University
Shinjae Yoo, Computational Science Center, Brookhaven National Laboratory
Dantong Yu, Computational Science Center, Brookhaven National Laboratory
Hong Qin, Computer Science Department, Stony Brook University

Current spectral clustering algorithms suffer from the sensitivity to existing noise, and parameter scaling, and may not be aware of different density distributions across clusters. If these problems are left untreated, the consequent clustering results cannot accurately represent true data patterns, in particular, for complex real world datasets with heterogeneous densities. This paper aims to solve these problems by proposing a diffusion-based Aggregated Heat Kernel (AHK) to improve the clustering stability, and a Local Density Affinity Transformation (LDAT) to correct the bias originating from different cluster densities. AHK statistically models the heat diffusion traces along the entire time scale, so it ensures robustness during clustering process, while LDAT probabilistically reveals local density of each instance and suppresses the local density bias in the affinity matrix. Our proposed framework integrates these two techniques systematically. As a result, not only does it provide an advanced noise-resisting and density-aware spectral mapping to the original dataset, but also demonstrates the stability during the processing of tuning the scaling parameter (which usually controls the range of neighborhood). Furthermore, our framework works well with the majority of similarity kernels, which ensures its applicability to many types of data and problem domains. The systematic experiments on different applications show that our proposed algorithms outperform state-of-the-art clustering algorithms for the data with heterogeneous density distributions, and achieve robust clustering performance with respect to tuning the scaling parameter and handling various levels and types of noise.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data Mining*; I.5.1 [**Pattern Recognition**]: Models—*Clustering*

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Aggregated Heat Kernel, Local Density Affinity Transformation

**ACM Reference Format:**
Hao Huang, Shinjae Yoo, Dantong Yu, and Hong Qin, 2014. Density-aware Clustering based on Aggregated Heat Kernel and Its Transformation. *ACM Trans. Knowl. Discov. Data, Vol.* 9, No. 4, Article 29 (June 2015), 34 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

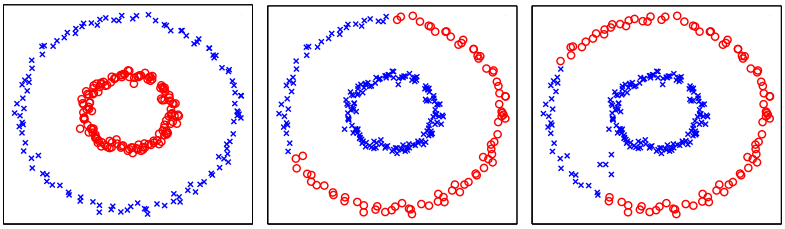
## 1. INTRODUCTION

Clustering, the task of discovering natural groupings based on the input data patterns, has been one of the most active research topics in machine learning and knowledge discovery. As a powerful unsupervised data analysis technique, clustering is especially desirable for modeling large datasets because the tedious and often inconsistent manual classification and labeling process can be avoided. While many traditional clustering algorithms have been developed over the past few decades [Jain and Dubes 1988] [Duda et al. 2001], some popular ones that emerged over the last decade generate promising results on various challenging tasks. Among them, spectral clustering [Ng et al. 2002] [Shi and Malik 2000] [Zha et al. 2001] [Chi et al. 2009] [Buhler and Hein 2009] [Wauthier et al. 2012] demonstrates excellent performance to detect clusters with complex shapes and complicated input space distributions.

### 1.1. Motivations

Despite their earlier success, most of spectral clustering methods still suffer from the following real world challenges:

— The clustering results can be radically different when the scaling parameters of the algorithms are slightly modified or there is some noise perturbation among clusters. We call such a susceptibility **the sensitivity to parameter tuning and noise**.
— Most of these methods tend to assign medium similarity between the boundary instances among clusters with different densities. Therefore **they fail to quantify local density well**, which may result in poor manifold reconstruction and undesirable clustering results.
— Most of the existing density-aware algorithms are only applicable on the Euclidean space. Therefore **their capabilities are significantly constrained in handling today's various types of data**, such as social networks and text datasets.



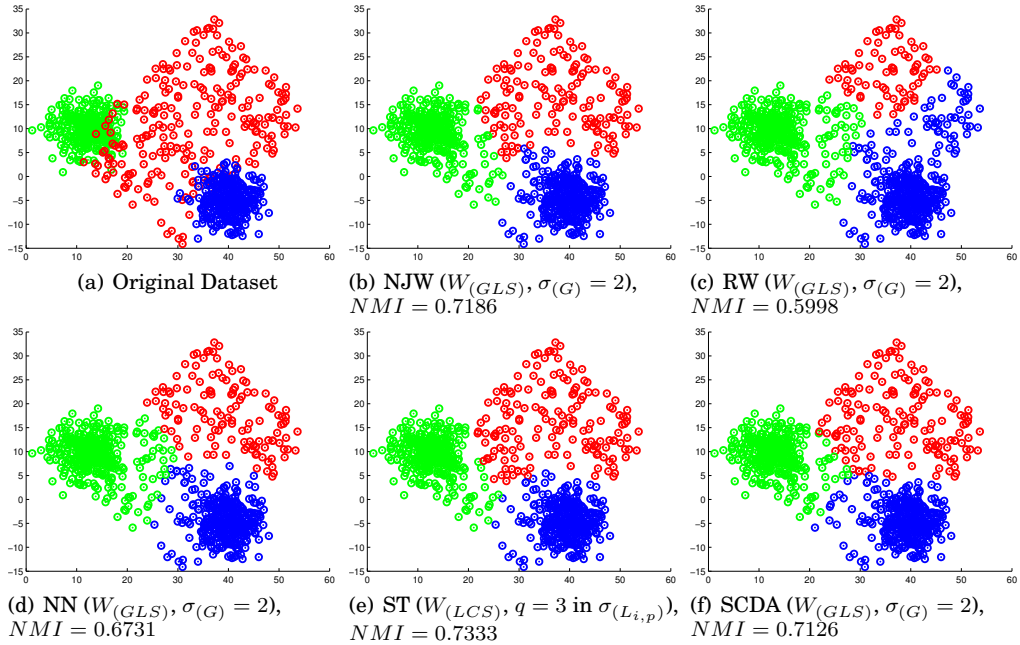
(a) $\sigma = 0.22$ (b) $\sigma = 0.23$ (c) $\sigma = 0.22$ with noises

Fig. 1. The sensitivity example of NJW [Ng et al. 2002], one of the traditional spectral clustering algorithms, with respect to different Gaussian scaling parameter $\sigma$ and noise appearance. The two output clusters are colored with red or blue. A small variation to $\sigma$ or data points (noise) leads to radically-different results. Such an instability becomes an issue to traditional spectral clustering algorithms.

Robustness is one of the most desirable properties of clustering algorithms, however, here it becomes an essential challenge for spectral clustering. As shown in Figure 1, it is a well known problem that the scaling parameter $\sigma$ of Gaussian kernel (see Equation 3 for details) for the affinity matrix has significant impacts on discovering embedded structure because $\sigma$ determines whether two points are considered similar (neighbors) or not [Perona and Freeman 1998]. Although several methods were proposed to address this problem (e.g., [Valizadegan and Jin 2007], [Zelnik-Manor and Perona 2004]), it remains challenging to find a certain range for $\sigma$ which is optimal to maintain stable yet desirable performance. Another aspect of robustness in spectral clustering is the clustering quality with respect to noise data. As noted in [Luxburg 2007], spectral clustering is less sensitive to data perturbation than the popular k-means algorithm. However, given different application domains and/or inappropriate data preprocessing techniques, spectral clustering can still be susceptible to noise [Verma and Meila 2001], which tends to complicate the clustering parameter selection, especially when making use of scaling parameter $\sigma$ of Gaussian kernel. In summary, since parameter selection can be significantly affected by the noise level of data (as shown in Figure 1(c)), we must address robust spectral clustering in terms of parameter selection and noise appearance simultaneously.

In this paper, the robustness of clustering algorithms should be measured in the following aspects: (1) not sensitive to small parameter changes; (2) not sensitive to existing noise; (3) stable performance even under a significant noise level or suboptimal parameter settings; and (4) competitive and comparable results when comparing with those less-robust clustering algorithms without any data perturbation and with correct parameter settings. With these robustness properties, we can reliably analyze data and conduct other data-driven tasks in subsequent analysis steps. The robustness property is equally significant for domain experts who do not have strong ma-

chine learning background as they become much more comfortable in utilizing robust algorithms for their domain data analysis. Therefore it is imperative to develop robust clustering algorithms [Chi et al. 2009].



(a) Original Dataset

(b) NJW ($W_{(GLS)}, \sigma_{(G)} = 2$), $NMI = 0.7186$

(c) RW ($W_{(GLS)}, \sigma_{(G)} = 2$), $NMI = 0.5998$

(d) NN ($W_{(GLS)}, \sigma_{(G)} = 2$), $NMI = 0.6731$

(e) ST ($W_{(LCS)}, q = 3$ in $\sigma_{(L_{i,p})}$), $NMI = 0.7333$

(f) SCDA ($W_{(GLS)}, \sigma_{(G)} = 2$), $NMI = 0.7126$

Fig. 2. Clustering results of different algorithms on a synthetic dataset with heterogeneous density distributions. Figure 2(a) shows the original dataset, where the green and blue clusters with Gaussian distributions have higher density than the red cluster with a uniform distribution. The clustering results of NJW (Figure 2(b)), RWC (Figure 2(c)) and NN (Figure 2(d)) are shown respectively, which are not capable of capturing the density variation. For the localized method, ST (Figure 2(e)) has better result since it has a locally adaptive scaling parameter (in Equation 4), while SCDA (Figure 2(f)) reveals a similar density-awareness as NJW. In short, none of the above methods provides a desirable separation that is aware of both density change and manifold structures across clusters.

Another requirement of real world clustering application is to discern the different density distribution among clusters. The traditional spectral clustering algorithms (such as NJW [Ng et al. 2002] and RWC [Shi and Malik 2000]) assume uniform sampling distribution inside the input dataset to approximate the continuous Laplace operators on Riemannian manifold, and tend to assign medium level affinity on the boundary between low and high density areas. These problems cause the inferior manifold reconstruction especially around cluster boundaries.

As an example, Figure 2 shows the clustering results from different graph Laplacians built upon Gaussian kernels. The synthetic dataset in Figure 2(a) contains three clusters: the blue and green clusters with a denser Gaussian distribution and the red one with a uniform and sparser distribution within a rectangular area. Figure 2(b) to 2(d) show the results from three conventional spectral clusterings: NJW [Ng et al. 2002] with symmetric Laplacian ($L_{sym}$), RWC [Meila and Shi 2001] with random walk Laplacian ($L_{rw}$), and NN [Luxburg 2007] without Laplacian normalization ($L_{nn}$) . Since $L_{sym}$ has a more balanced view, NJW performs better than RWC, and demonstrates better density-awareness. However, if we take density distribution into account, all of them fail to separate the clusters appropriately.

Some localized approaches have been focusing on solving these problems, e.g., Self-tuning Spectral Clustering (ST) [Zelnik-Manor and Perona 2004] and Local Density Adaptive Similarity (SCDA)[Zhang et al. 2011]. Nevertheless, they could not effectively capture the local density on the affinity matrix since additional parameters which are very sensitive to heterogeneous density distributions are required. Therefore such approaches may also fail to quantify local density well, and cause undesired clustering results (as shown in Figure 2(e) and 2(f)). Moreover, these algorithms are built upon Gaussian kernel, and could only be applicable to the applications in the Euclidean space. Their capabilities are therefore significantly constrained in handling today's big complex data. One example is network dataset, including social networks, computer networks, and biological networks. Network dataset can be naturally represented as affinity matrix themselves because they are already of graph-structure (see details in Equation 1). Another example is text data which often uses the cosine kernel to measure similarity (see details in Equation 2). Although Gaussian kernel is popularly used in many applications, we also need to handle the aforementioned datasets with diverse characteristics. Therefore, to be more practical and adaptive in different real world situations, local-density-aware clustering algorithms need to work well with any form of similarity kernels.

### 1.2. Contributions

In this research we propose a heat-diffusion-based framework which provides not only competitive average performance, but also robustness to scaling parameter, noise appearance and different density distributions across clusters. Our framework has the following contributions:

1. We derive a robust kernel function by integrating heat kernel along the entire time scale (Section 3.1), and combine it with Laplace-Beltrami Normalization (LBN, Section 3.3). We call this algorithm as Aggregated Heat Kernel (AHK, Section 3.4). As a result, we **provide a robust clustering algorithm while reducing the negative influences on stability by scaling parameter tuning and noise appearance**. We also discuss the connections of AHK to the other popular and related approaches (Section 3.2).
2. We design a probability-based local density affinity transformation (LDAT, Section 4) that aims to **reduce different density effects across clusters in the affinity matrix**. It is a simple and effective enhancement to local density awareness especially around the cluster boundary area. It is not only based on affinity matrix, **so it works well with any type of similarity kernels**. These features distinguish our proposed framework from other candidate approaches in handling diverse and complex real world problems.
3. Our novel framework (Section 5), systematically combining AHK and LDAT together, delivers robust clustering results in terms of different scaling parameter, noise level and divergent density distribution across different clusters.
4. We thoroughly evaluate the proposed framework with several closely-related baseline algorithms on a number of synthetic and benchmark datasets (Section 6). The experimental results confirm that the proposed framework, even under suboptimal parameter settings, outperforms existing approaches for datasets with noise and heterogeneous density distribution, using different similarity kernels.

## 2. BACKGROUND INTRODUCTION

Our framework is based upon spectral clustering and heat diffusion. In this section we briefly review the basic ideas of spectral clustering, affinity matrix constructions, diffusion maps and heat equation, and analyze the weakness of existing approaches.

### 2.1. Spectral Embeddings and Clustering

Spectral clustering already gained increasing popularity in the last decade because of its ability to discover embedding data structure. It has a strong connection with graph cut, i.e., it uses eigenspace to solve a relaxed form of the balanced graph partitioning problem [Ng et al. 2002]. Its second desirable aspect is that, with nonlinear kernels it can capture the nonlinear structure of data, which is difficult for $k$-means [Hartigan and Wong 1978] or other linear clustering algorithms.

**ALGORITHM 1:** SpectralClustering($X$, $c$)

**Input**: $X \in R^{n \times m}$ where $n$ is #instances, $m$ is #features, and $c$ is #clusters.
**Output**: Cluster assignments of $n$ instances.

1 Construct the affinity matrix $W \in R^{n \times n}$;
2 Compute the diagonal matrix $D \in R^{n \times n}$ where $D(i,i) = \sum_{j=1}^{n} W(i,j)$ and $D(i,j) = 0$ if $i \neq j$ ;
3 Apply the graph Laplacian $L$ using $L_{nn} = D - W$, $L_{rw} = I - D^{-1}W$ or $L_{sym} = I - D^{-1/2}WD^{-1/2}$ where $I$ is an identity matrix ;
4 Extract the first $c$ nontrivial eigenvectors $\psi$ of $L$, $\psi = \{\psi_1, \psi_2, \ldots, \psi_c\}$ ;
5 Re-normalize the rows of $\psi \in R^{n \times c}$ into $Y_i(j) = \psi_i(j)/(\sum_l \psi_i(l)^2)^{1/2}$ ;
6 Run $k$-means with $c$ and $Y \subset R^{n \times c}$.

Spectral clustering, as shown in Algorithm 1, usually starts with local information encoded in a weighted graph which is constructed from certain similarity kernels on input data, and clusters according to the global eigenvectors of the corresponding (normalized) affinity matrix. However, it has a few limitations as follows:

— The selection of the scaling parameter (if any) of similarity kernel could affect the clustering results radically (as shown in Figure 1(a) and 1(b)) because the scaling parameter usually determines each instance's neighborhood scope.
— The clustering results are sensitive to noise. For instance in Figure 1(c), with only a few noisy instances, the clustering result is quite different and the optimal range of scaling parameter also varies.
— The reconstructed embedding structures may fail to represent the diversity of density across clusters, which leads to clustering results with a poor quality (as shown in Figure 2).

The above problems are partly due to the fact that the similarity kernels (or affinity matrix construction) used in the spectral clustering are sensitive to the parameter scaling and noise appearance [Zelnik-Manor and Perona 2004]. In Section 2.2, we will describe some popularly used similarity kernels. In Section 3 we will propose a robust clustering algorithm against parameter and noise sensitivity.

The second reason for the aforementioned problems is that (normalized) affinity matrix cannot take the local density information into consideration, in particular for those data points between two clusters with heterogeneous densities. A synthetic example of such problem is demonstrated in Figure 2. Section 4 will introduce a simple and effective way of correcting the local density bias.

### 2.2. Affinity Matrix Construction

Affinity matrix construction is the first step of spectral clustering with significant influence to the final results. In practice it is derived from certain similarity kernels. We denote the input dataset as $X$ which is a $n \times m$ matrix with $n$ instances and $m$ features. Its global similarity matrix $W$, a $n \times n$ matrix, represents the pair-wise likeness of instances usually considering the whole feature space. How to choose an appropriate

similarity kernel is a critical step in spectral clustering as different types of datasets might have different preference for similarity measurements. There are several popular ways to measure the similarity $W(i,j)$ between any two instances $x(i)$ and $x(j)$. In this paper we focus on three of them: network connectivity, cosine similarity, and Gaussian kernel.

**Network Connectivity.** Network datasets such as social networks, computer networks and biological networks, define affinity matrix based on their dataset representations, which are often modeled as undirected graphs. Each edge has a weight to describe the relationship between the two related nodes representing data instances in the dataset. In a network dataset the simplest edge weight is usually defined as the connectivity between two nodes. The simplest network connectivity $W_{(NET)}(i,j)$ can be defined as:

$$W_{(NET)}(i,j) = \begin{cases} 1, & \text{if } x(i) \text{ and } x(j) \text{ are connected,} \\ 0, & \text{if } x(i) \text{ and } x(j) \text{ are unconnected.} \end{cases} \tag{1}$$

In addition, we can model network datasets as directed graphs as well. A twitter network with followers and followees is a good example of directed graph.

**Cosine Similarity.** A popular measurement for text dataset is the cosine angle between two vectors [Andrews and Fox 2007]. The cosine similarity is represented using dot product and magnitude as:

$$W_{(COS)}(i,j) = \frac{x(i) \cdot x(j)}{\| x(i) \|_2 \cdot \| x(j) \|_2}. \tag{2}$$

For text matching, the vectors $x(i)$ and $x(j)$ are usually the term frequency vectors of the documents. Since term frequency is always positive, the resulting similarity ranges from $0$ meaning independence, to $1$ meaning exactly the same, and in-between values indicating intermediate similarity. The cosine similarity can be seen as a method of normalizing length during comparison, with denominator normalizing each vector to compare different text sizes.

**Gaussian Kernel.** One of the most commonly used similarity measurements in the clustering application is the Gaussian kernel, of which traditional form is defined as follows:

$$W_{(GLS)}(i,j) = exp(\frac{- \| x(i) - x(j) \|^2}{2{\sigma_{(G)}}^2}), \tag{3}$$

where $\sigma_{(G)}$ controls the width of neighborhood [Luxburg 2007] with a globally fixed value. So we call this kernel as the global Gaussian kernel $W_{(GLS)}$. It is widely used because it works well on most datasets. However, its biggest challenge is how to choose $\sigma_{(G)}$'s value, which affects the clustering results significantly [Zelnik-Manor and Perona 2004]. In other words, clustering result is very sensitive of tuning $\sigma_{(G)}$. Besides, $\sigma_{(G)}$ is not adaptive to local density change.

Instead of selecting one globally-fixed parameter $\sigma_{(G)}$, Zelnik-Manor et al. proposed to calculate a local scaling parameter $\sigma_{(L_{i,k})}$ for each data point in their self-tuning spectral clustering algorithm (ST) [Zelnik-Manor and Perona 2004]:

$$W_{(LCS)}(i,j) = exp(\frac{- \| x(i) - x(j) \|^2}{\sigma_{(L_{i,k})}\sigma_{(L_{j,k})}}), \tag{4}$$

where the parameter $\sigma_{(L_{i,k})}$ is the Euclidean distance between $x(i)$ and its $k$-th nearest neighbor ($k$-nn). This kernel uses $k$-nn distance to approximate the local density, which is similar to the idea of Local Outlier Factor (LOF) [Breunig et al. 2000]. Therefore it can adaptively recognize the local density difference to some extent. However it is

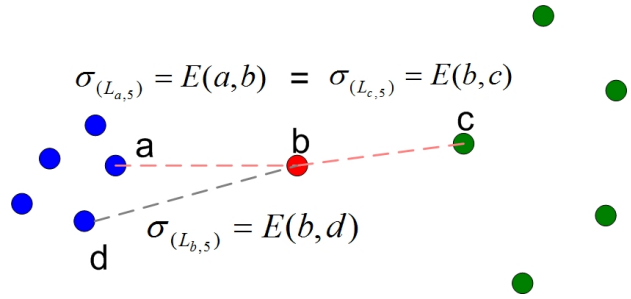



Fig. 3. Due to the similar density distribution, point $b$ should belong to the green cluster. However, both global and local scaling Gaussian kernels fail to classify point $b$ due to their non-awareness of local density statistics.

extremely important to determine the value of $k$ to faithfully reveal the local density, as shown in [Huang et al. 2012]. On one hand, $k$ cannot be too large to capture the local distribution. On the other hand, an overly small $k$ can lead to statistical error without a sufficient neighborhood scope. This is to say, compared with global Gaussian kernel $W_{(GLS)}$, local kernel $W_{(LCS)}$ shifts the degree of freedom, or sensitivity, from $\sigma_{(G)}$ to $k$, which is still hard for users to specify.

Particularly, both Equation 3 and 4 may fail miserably on the boundary area among clusters with different densities. Figure 3 shows a synthetic example that both kernels fail to classify the red data point $b$. Here the blue cluster is relatively denser than the green one. The point $b$ lies between these two clusters with the same Euclidean distance to the closest data point in each cluster (namely $E(b,a) = E(b,c)$, where $E(i,j)$ denotes the Euclidean distance between $x(i)$ and $x(j)$). When the density distribution is considered, point $b$ should belong to the green cluster because its local density is more similar to the density of the green cluster. Therefore, the ideal similarity kernel should return $W(b,a) < W(b,c)$ and $W(a,d) > W(a,b)$.

Nevertheless, the global Gaussian kernel, with a single scaling parameter, only obtains the right result on $W(a,d) > W(a,b)$ because of $E(a,d) < E(a,b)$, while on the other hand returns $W_{(GLS)}(b,a) = W_{(GLS)}(b,c)$ because of $E(b,a) = E(b,c)$. For the local scaling Gaussian kernel, we need to tune $k$ very carefully. If $k = 5$ there is $\sigma_{(L_{a,5})} = E(a,b)$ and $\sigma_{(L_{c,5})} = E(b,c)$, which also leads to $W_{(LCS)}(b,a) = W_{(LCS)}(b,c)$. Only if $k < 5$ it gives $W_{(LCS)}(b,a) < W_{(LCS)}(b,c)$ because of $\sigma_{(L_{a,k})} < \sigma_{(L_{b,k})} \simeq \sigma_{(L_{c,k})}$. However, it will bring another issue: it leads to $W_{(LCS)}(a,d) < W_{(LCS)}(a,b)$ due to $\sigma_{(L_{d,k})} < \sigma_{(L_{b,k})}$ when $k < 5$. The same problem also shows up in Figure 2(e): although $k$ is best tuned and ST has additional adjusting steps to further boost the performance [Zelnik-Manor and Perona 2004], it does not improve performance substantially compared with NJW (only about $2\%$ improvement in NMI). In other words, both the global and local scaling Gaussian kernel cannot stably and accurately differentiate the similarity differences with regards to local density.

Recently, Zhang et al. proposed a local density adaptive similarity kernel (SCDA) [Zhang et al. 2011] which is defined as:

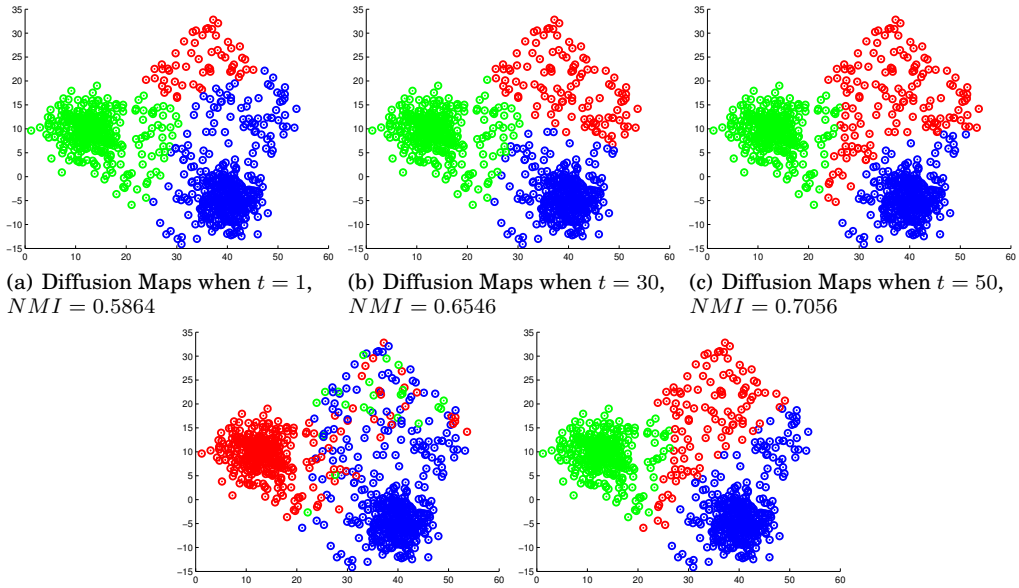
$$W_{(DA)}(i,j) = exp(\frac{- \parallel x(i) - x(j) \parallel^2}{2\sigma^2_{(G)}(f_\epsilon(i,j)+1)}), \tag{5}$$

where $f_\epsilon(i,j)$ is the number of instances in the joint region of the $\epsilon$-neighborhoods around instance $x(i)$ and $x(j)$, and $\epsilon$ is the specified radius of the sphere neighborhood region. It is claimed that $f_\epsilon(i,j)$ can represent the local density between $x(i)$ and $x(j)$, and therefore $W_{(DA)}$ can be used to distinguish inter-cluster instances. However, the

way of choosing $\epsilon$ in [Zhang et al. 2011] is by a linear regression with the input parameters such as maximum and variance of all instance pairs in the test dataset. Not surprisingly, it is highly unstable when SCDA is used on the other datasets in an unsupervised way. Correa et al. proposed a similar idea using empty region [Correa and Lindstrom 2012] which also suffers instability by a slight perturbation to the radius of region, especially on the complex high-dimensional datasets, due to the curse of dimensionality. Figure 2(f) shows that SCDA performs quite similarly to NJW, and can neither provide a convincing correction to the density bias.

### 2.3. Diffusion Distance and Diffusion Maps

Embedding reconstruction in spectral clustering (Step 1 to 5 in Algorithm 1) are very sensitive to noise appearance and scaling parameter tuning (if it is built upon the Gaussian kernels). Diffusion maps was proposed by Coifman et al. in [Coifman and Lafon 2006] to solve these problems.



(a) Diffusion Maps when $t = 1$, $NMI = 0.5864$

(b) Diffusion Maps when $t = 30$, $NMI = 0.6546$

(c) Diffusion Maps when $t = 50$, $NMI = 0.7056$

(d) Diffusion Maps when $t = 100$, $NMI = 0.6140$

(e) Multiscale Diffusion Maps, $NMI = 0.6486$

Fig. 4. Clustering results of Diffusion Maps (DM) and Multiscale Diffusion Maps (MDM) on the synthetic dataset in Figure 2(a). The global Gaussian kernel is used here with $\sigma_{(G)} = 2$. Figure 4(a) to 4(d) show the results of DM from $t = 1$ to $t = 100$. Although DM with $t = 50$ obtains better separation in the boundary area among the three clusters, it is hard to guess the best range of $t$ unsupervisedly. MDM, in spite of the elimination of parameter $t$, easily gets over-diffusion without perception of density change (see Figure 4(e)).

Diffusion maps (DM) is a Markov-transition-based projection hinged on diffusion process. The non-negativity property of the original affinity matrix $W$ allows to normalize it into a Markov transition matrix $P = D^{-1}W$. The states of the corresponding Markov process are data points, which enables us to analyze it as (positive) random walk. It is straightforward to calculate the transition probability, $p_t(i, j)$ (the probability of transition from $x(i)$ to $x(j)$ after $t$ steps) using entries from $P$. Thus the diffusion

distance $\mathcal{D}_t(i,j)$ between $x(i)$ to $x(j)$ at the time scale of $t$ can be defined as:

$$\mathcal{D}_t(i,j) = (\sum_k \frac{(p_t(i,k) - p_t(j,k))^2}{\phi_1(k)})^{\frac{1}{2}}, \tag{6}$$

where $\phi_1$ is the stationary distribution of the positive random walk (trivial left eigenvectors). So the diffusion map at the time scale of $t$ projects the data points to $n$ dimensional eigenspace as:

$$\Psi_t : x \to [\lambda_1^t \psi_1(x), \lambda_2^t \psi_2(x), ..., \lambda_n^t \psi_n(x)], \tag{7}$$

where $\lambda_i$ are eigenvalues and $\psi_i$ are the corresponding right eigenvectors of $P$ [Nadler et al. 2005]. In this way the diffusion distance between $x(i)$ and $x(j)$ becomes:

$$\mathcal{D}_t(i,j) = (\sum_{k=1}^{n} [\lambda_k^{2t} (\psi_k(i) - \psi_k(j))^2])^{\frac{1}{2}}. \tag{8}$$

By projecting the data to the diffusion space, 1) the sensitivity to noise is minimized due to the theory of random walk, 2) the effect of scaling parameter $\sigma$ (if there is any) is reduced. However, another scaling parameter $t$ is still essential because it controls the transitive connectivity. Besides, how to tune the value of $t$ is perplexing because even less clues for tuning it exist than that of $\sigma$ in the Gaussian kernels. Here we use experiments to reveal its effect, as shown in Figure 4(a) to 4(d).

In 2009, Richards et al. proposed multiscale diffusion maps (MDM) [Richards et al. 2009], which considers all possible paths between each instance pair across all the discrete time scales $t$ in the diffusion space. In practice $\lambda_i^t$ in Equation (7) is replaced by:

$$\sum_{t=1}^{\infty} \lambda_i^t = \frac{\lambda_i}{1 - \lambda_i}. \tag{9}$$

So the multiscale diffusion maps is defined as:

$$\Psi_{(M)} : x \to [\frac{\lambda_1}{1-\lambda_1}\psi_1(x), \frac{\lambda_2}{1-\lambda_2}\psi_2(x), ..., \frac{\lambda_n}{1-\lambda_n}\psi_n(x)]. \tag{10}$$

Multiscale diffusion maps is claimed to be more robust [Richards et al. 2009] by eliminating the effect of $t$. The quantity of MDM involves summing over all paths of all discrete time scales connecting $x(i)$ to $x(j)$. As a consequence, this projection should be very robust to noise perturbation in theory, unlike the geodesic distance or Euclidean distance. From the prospect of machine learning, this observation allows us to conclude that this projection is appropriate for designing inference algorithms based on the majority: it takes into account all the evidences relating $x(i)$ to $x(j)$.

Although diffusion maps [Coifman and Lafon 2006] and multiscale diffusion maps [Richards et al. 2009] provide more stable descriptions with a strong probabilistic interpretation, and therefore reduce the instability incurred by Gaussian scaling parameters and noise appearance, they still suffer from the lack of density-awareness as shown in Figure 4.

### 2.4. Heat Equation and Heat Kernel

Our proposed Aggregated Heat Kernel (AHK) in Section 3 is strongly inspired by heat diffusion theory [Hsu 2002] because of its capability to provide intrinsic and robust similarity measurement that is aware of manifold structure, and other attractive properties such as symmetric, positive semi-definite, and stable under noise appearance.

Heat diffusion can be interpreted as the transition density function of Brownian motion [Sun et al. 2009], a fundamental continuous Markov process in the time domain. Thereby in practice the heat equation is often coupled with a random walk graph Laplacian $L_{rw} = I - D^{-1}W$, which describes a stochastic process that randomly jumps from instance to adjacent instance. Heat equation therefore can be defined by:

$$\frac{\partial H_t}{\partial t} = -L_{rw} H_t, \tag{11}$$

where $H_t = e^{-tL_{rw}}$ is the heat kernel on Riemannian manifold $\mathcal{M}$ and $t$ is the time scaling parameter [Grigoryan 1999]. For $L_{rw} = \psi' \lambda \psi$ ($\psi$ and $\lambda$ are the eigenvectors and eigenvalues of $L_{rw}$), the heat kernel can be approximated through:

$$H_t(i,j) = \sum_{k=1}^{N} [e^{-\lambda_k t} \psi_k(i) \psi_k(j)], \tag{12}$$

where $H_t(i,j)$ represents the amount of heat being transferred from $x(i)$ to $x(j)$ in time $t$ given a unit heat source at $x(i)$ in the very beginning (when $t = 0$). The scaling parameter $t$ here is used to control the transitive connectivity: a small $t$ can slightly improve the connectivity of a loosely-connected graph, while a large $t$ makes the graph even more strongly-connected.

### 2.5. Other Related Works

Towards robustness, researchers have explored various techniques, including robust statistics [Black et al. 1998] [Huber and Ronchetti 2009], noise insensitive regression [Cherkassky and Ma 2005] [Camps-Valls et al. 2006], noise resistant transformation [Weiss 1993], and noise robust clustering [Dave and Krishnapuram 1997] [Guha et al. 2000] [Li et al. 2007] . However, robust clustering approaches that are adaptive to both parameter tuning and noise sensitivity are rather rare. In fact, as shown in Figure 1, scaling parameter and noise perturbation are correlated to each other. Mean shift clustering [Comaniciu and Meer 2002] and noise robust spectral clustering [Li et al. 2007] also fail in considering these two simultaneously and systematically. In [Chang and Yeung 2008], M-estimation robust statistics is used in a robust path-based similarity measurement which requires no local parameters to be set manually, nonetheless, prior knowledge of data domain is required, which is not our research target in this paper.

Towards density-driven clustering, some non-spectral clustering algorithms such as DBSCAN [Ester et al. 1996] [Tran et al. 2013] and OPTICS [Ankerst et al. 1999] start from the estimated density distributions of corresponding nodes. Some researches approached density through updating similarity information, such as shared nearest neighbors (SNN) [Jarvis and Patrick 1973] and [Guha et al. 2000]. The similarity between two points is confirmed by their shared (or common) nearest neighbors. Later some advanced techniques [Ertoz et al. 2002] [Steinbach et al. 2003] based on SNN have also been proposed. But their performance suffer significantly from the curse of dimensionality and the sensitivity of neighborhood scaling parameters [Ertoz et al. 2002], since their metrics are usually based on Euclidean space. Moreover they cannot cluster datasets well when the density distributions vary significantly [Hinneburg and Keim 1999].

There are some existing graph-based techniques which built upon hierarchical modeling. Chameleon [Karypis et al. 1999] defines affinity from relative inter-connectivity and closeness which are based on a min-cut bisection of clusters. But its computation requires a high computational cost. Recently Graph Degree Linkage [Zhang et al. 2012] was designed with easy implementation and high computational efficiency. How-

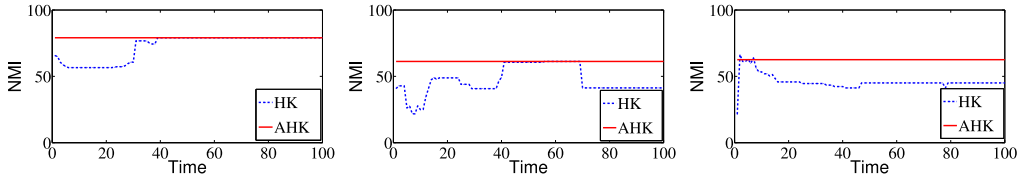
ever its performance is very sensitive to the perturbation of similarity result from scaling parameter tuning.

In the next several sections, we propose a new framework that corrects the undesired effects from the aforementioned limitations. It is built upon advanced diffusion space which is stable to scaling parameter tuning and noise perturbation. Also our proposed method is aware of local density change across clusters, therefore it behaves well under non-uniform density distribution. Moreover, compared with the other popular algorithms, our method is more universally applicable since it can use the Gaussian kernels, as well as any other kernel to construct affinity matrix.

## 3. AGGREGATED HEAT KERNEL AND ITS USE IN CLUSTERING

This section proposes a probabilistic clustering method based on heat diffusion theory. The reason we resort to heat diffusion is to minimize the negative influence of both scaling parameter tuning and noise appearance. Since we concentrate on global distribution for data clustering, the embedded structure must be invariant to local perturbation (noise or outliers), and they should be determined only by "visible" neighborhood while avoiding negative effects from changing scaling parameters. The Heat kernel (HK), as the fundamental solution of heat diffusion, offers a statistical description of random walk, so it can be employed to build a diffusion map. Here we integrate spectral clustering and the heat diffusion theory together and show that the integrated approach improves the robustness to both scaling parameter tuning and noise appearance.

### 3.1. Aggregated Heat Kernel




(a) $\sigma = 0.15$ (b) $\sigma = 0.16$ (c) $\sigma = 0.28$

Fig. 5. The sensitivity of heat kernel (HK, Equation 12) to time scaling parameter $t$ on Iris dataset clustering (measured by NMI). Experiments in Figure 5(a) to 5(c) are built upon global Gaussian kernel with different $\sigma$. We can see that AHK outperforms HK in most cases and it doesn't require tuning $t$. We use the random walk Laplacian in this experiment.

In this subsection, we describe and analyze the construction of Aggregated Heat Kernel (AHK). Similar to the conventional heat kernel in Equation 12, AHK is also built upon the eigen-decomposition of (Laplacian-normalized) affinity matrix $W$. As discussed in Section 2.4, a heat kernel (HK) is multiscale. The function $H_t(i,*)$ is mainly determined by the nearby neighborhood of $x(i)$, and this area grows bigger as $t$ increases. In other words, for a small $t$, $H_t(i,*)$ only represents local properties of the area around $x(i)$, but a large $t$ can capture the properties from a larger area or even the entire data space. Yet this additional degree of freedom makes it difficult to determine the value of $t$ (Figure 5) because we have little clue about how to find the best $t$ value, which is similar to the time scaling parameter in diffusion maps and the scaling parameter $\sigma$ in Gaussian kernels. In other words, the clustering result become sensitive to the time parameter selection in heat kernel.

We propose a robust kernel function by integrating the entire continuous time scale on heat kernel, and name it as **Aggregated Heat Kernel** (AHK):

$$\mathcal{H}(i,j) = \int_0^\infty H_t(i,j)dt = \sum_{k=1}^{n}[\frac{1}{\lambda_i}\psi_k(i)\psi_k(j)]. \tag{13}$$

Specifically, the derivation of this function can be explained in the form of Laplace transform [Arendt 2011]:

$$F(s) = \int_0^\infty e^{-st}f(t)dt, \tag{14}$$

where parameter $s$ is a complex number and $f(t)$ is "degeneralized" to a constant function one. Here the Laplace transform is interpreted as a transformation from the time domain, in which inputs and outputs are functions of time ($t$), to the frequency domain, where the inputs and outputs are the functions of frequency ($\lambda$). Therefore this transform provides an alternative functional description that simplifies the process of analyzing the heat system behavior, and synthesizes a new comprehensive system with a set of properties inherited from the original heat kernel:

— Symmetric: $\mathcal{H}(i,j) = \mathcal{H}(j,i)$.
— Semigroup identity: $\mathcal{H}(i,j) = \int_M \mathcal{H}(i,k)\mathcal{H}(j,k)dk$.
— Positive semi-definite: $\sum_{p,q}\mathcal{H}(i,j)c_pc_q \geq 0$, where $c_1, c_2, ..., c_n$ are real numbers.

With the pure and applied probability term, AHK can be explained as an expected value of heat kernel. If we interpret $t$ as a random variable with the probability density function $\mathscr{F}$, then AHK, or the Laplace transform of $f$ is given by the expectation:

$$\mathcal{L}\mathscr{F}(\lambda) = E[e^{-\lambda t}]. \tag{15}$$

So AHK, to some degree, is a weighted average of all possible heat diffusion processes across the entire continuous time-domain.

The definition of AHK can also be elucidated by Fredholm theory [Driver 2003], a theory of integral equations, where the actual function space is determined by the eigenfunctions of the differential operator; that is, by the solutions to $L\psi(i) = \lambda\psi(i)$. The set of eigenvectors $\psi$ here spans a Hilbert space since there is a natural inner product. Therefore the kernel $\mathcal{H}(i,j)$ is a realization of the Fredholm operator or the Fredholm kernel. It follows from the completeness of the basis of the Hilbert space, namely, that one has:

$$\delta(x(i) - x(j)) = \sum_k[\psi_k(i)\psi_k(j)], \tag{16}$$

where $\delta(x)$ is the Dirac delta function (a generalized function defined in the real space $R$, such that its value is zero everywhere except at origin $0$) since the eigenvectors $\psi$ associated with $L$ are assumed to be complete and orthogonal to each other.

From Figure 5 we observe that in the original heat kernel (HK) the time scaling parameter $t$ is also correlated with Gaussian scaling parameter $\sigma$, and $t$ needs to be tuned carefully. Moreover, Figure 5 shows that AHK performs better than the original HK on almost all times $t$ regardless the value of $\sigma$. Comparatively, AHK is capable of providing more comprehensive and stable probabilistic affinity information.

By the definition of heat diffusion, AHK is naturally associated with the random walk normalization, $L_{rw}$, but we could also generalize AHK on other Laplacians such as $\mathcal{H}_{sym}$ on symmetric $L_{sym}$ or $\mathcal{H}_{nn}$ on unnormalized $L_{nn}$. In Section 3.3 we will analyze the best Laplacian for constructing AHK.

### 3.2. Connections to AHK

In this subsection we build theoretical connections from AHK to the other existing popular techniques.

***Inverse Laplacian***. AHK can be viewed as a pseudo inverse or Moor-Penrose inverse [Gutman and Xiao 2004]. By doing so, we achieve multiscale heat diffusion. Instead of doing pseudo inverse, we could directly inverse graph Laplacian matrix [Li et al. 2007] as:

$$(I + \beta L_{sym})^{-1}, \tag{17}$$

where $\beta$ is the positive regularization parameter and $I$ allows us to invert Laplacian matrix always. Note that, [Li et al. 2007] used this direct inversion to design noise robust spectral clustering.

***Commute Distance***. Commute distance $C(i,j)$ between $x(i)$ and $x(j)$ is defined by the expected random walk round trip travel time. AHK is also known as Green's function [Qiu and Hancock 2007], which is closely related to the commute distance (CD) or resistance distance. The Green's function is a left inverse operator of Laplace operator, $\mathcal{H}_{rw} \cdot L_{rw} = I$. For $\mathcal{H}_{nn}$ constructed on unnormalized $L_{nn}$, commute distance can be reformulated as:

$$C(i,j) = vol(\mathcal{H}_{nn}(i,i) + \mathcal{H}_{nn}(j,j) - 2\mathcal{H}_{nn}(i,j)), \tag{18}$$

where $vol = \sum_{i=1}^{n} D(i,i)$. Just like AHK, commute distance also considers all possible length, paths and their weights, which is more robust than the shortest path or geodesic distance. Note that, commute distance can also be expressed by the random walk $L_{rw}$ or symmetric graph Laplacian $L_{sym}$ [Qiu and Hancock 2007].

***Diffusion Distance***. Commute distance is also related to diffusion distance. By integrating Equation (13) into the above equation, we get:

$$C(i,j) = vol \sum_{k=2}^{n} [(1/\lambda_k)(\psi_k(i) - \psi_k(j))^2], \tag{19}$$

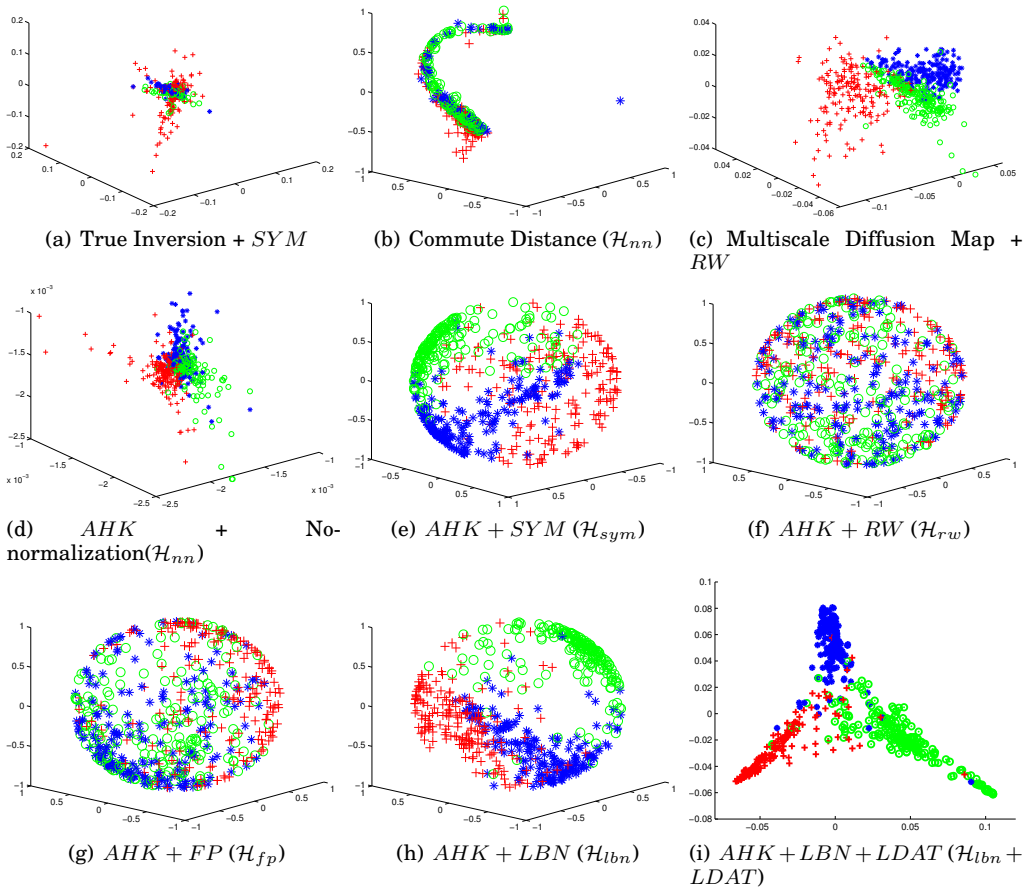
and also multiscale diffusion distance can be defined by:

$$\sum_{t=1}^{\infty} D_t^2(i,j) = \sum_{k=1}^{n} [1/(1-\lambda_k^2)(\psi_k(i) - \psi_k(j))^2]. \tag{20}$$

Both commute distance and diffusion distance look similar but they have different eigenvalue weighting and different Laplacian normalization.

Diffusion distance [Richards et al. 2009] can also be represented by $1/\lambda_i$, which shares the same weighting with $\mathcal{H}$ but it is for distance weighting. If the time summation starts from $t = 1$, then it is exactly the same as the multiscale diffusion distance (MDM) of Equation (9). Both of eigenvalue weighting (starting from $t = 0$ or $t = 1$) will show quite similar weighting distribution anyway for $0.5 \leq \lambda \leq 2$, which is common for most of the graph Laplacians.

***Laplace Transform and Fourier Transform***. As previously analyzed, AHK can be explained as a degeneralized form of Laplace transform [Arendt 2011]. Laplace transform is related to Fourier transform, but whereas the Fourier transform expresses a function or signal as a series of modes of vibration (frequencies), the Laplace transform resolves a function into its moments. Like the Fourier transform, in our derivation of AHK, the Laplace transform is used for solving differential and integral equations. But the equivalence relation between Laplace Transform and Fourier



(a) True Inversion + $SYM$ (b) Commute Distance ($\mathcal{H}_{nn}$) (c) Multiscale Diffusion Map + $RW$

(d) $AHK$ + No-normalization($\mathcal{H}_{nn}$) (e) $AHK + SYM$ ($\mathcal{H}_{sym}$) (f) $AHK + RW$ ($\mathcal{H}_{rw}$)

(g) $AHK + FP$ ($\mathcal{H}_{fp}$) (h) $AHK + LBN$ ($\mathcal{H}_{lbn}$) (i) $AHK + LBN + LDAT$ ($\mathcal{H}_{lbn} + LDAT$)

Fig. 6. Different ways of manifold reconstruction on 20ngB dataset.

Transform is not valid in our AHK derivation because the region of convergence (ROC) of $F(s)$ in Equation 14 contains no imaginary component.

### 3.3. Different Laplacians and Their Comparison

Even though we made proper connections among relative approaches, most of them used different Laplacians without thorough evaluation. Therefore it is not yet clear what is the best graph Laplacian for our proposed $\mathcal{H}$. It is shown in [Lafon et al. 2006] that if we assume uniform sampling of data points from a sub-manifold $\mathcal{M}$, the eigenvectors of $L_{rw}$ with $\sigma \to 0$ and $n \to \infty$ tend to approximate Laplace-Beltrami operator on $\mathcal{M}$, which guarantees manifold reconstruction. However, in reality, the sampling rate of data points tends to be non-uniform and it shows skewed density distributions, resulting in a manifold reconstruction with a poor quality in AHK. The following two additional normalizations are used to improve the density awareness of Laplacians:

$$W^{(\kappa)} = D^{-\kappa} W D^{-\kappa}, \tag{21}$$

$$L^{(\kappa)} = I - {D^{(\kappa)}}^{-1} W^{(\kappa)}, \tag{22}$$

where $\kappa$ is a normalization factor and $D^{(\kappa)}$ is the diagonal matrix with the sum of $W^{(\kappa)}$ row weight.

— If $\kappa = 0$, $L^{(0)} = L_{rw}$ which is exactly the random walk (RW) Laplacians.
— If $\kappa = 1/2$, then it is called Fokker-Planck (FP) diffusion.
— If $\kappa = 1$, it is called Laplace-Beltrami Normalization (LBN).

The relations among the three normalizations are well described in [Coifman and Lafon 2006]. Depending on $\kappa$, LBN can also be reduced to the random walk normalization or Fokker-Planck diffusion. In particular, we focus on LBN because it removes the negative influence of the dataset density and recovers manifold structures on $\mathcal{M}$ with the condition of $\sigma \to 0$ and $n \to \infty$ [Coifman and Lafon 2006]. In other words, the additional re-normalization of affinity matrix $W$ enables us to reconstruct manifold structures under non-uniform density distribution. Another advantage of LBN is that the consequent clustering results can be less sensitive to noise and scaling parameter tuning.

Figure 6 shows the effects of different approaches and Laplacians on 20 newsgroup text data (20ngB) (see Section 6 for more details). True inversion (Figure 6(a)) and commute distance (Figure 6(b)) show the worst results in separating three topics. Although they share the same Laplacian matrix inversion, the results are quite different. Interestingly multiscale diffusion map (MDM, Figure 6(c)) shows the best separation among all the non-AHK approaches. In the case of AHK, most of Laplacian approaches (except unnormalized Laplacian) reconstruct the topic distribution as a sphere shape. AHK with unnormalized Laplacian (Figure 6(d)) appears to have the ability of separation but the distance among documents are very close to each other compared with other Laplacians. Symmetric Laplacian (Figure 6(e)) shows very good separation and sphere shape reconstruction but it is not anisotropic transition. The original random walk (RW) normalization (Figure 6(f)) shows the most mixture of three topics but once we add the additional normalization of Equation (21), we reconstruct better manifold structures. LBN shows the best coherent and condensed structure (Figure 6(h)) among all different Laplacians. For our future experiments we mainly focus on LBN, but we provide further and more detailed analysis regarding different approaches with different Laplacians in Section 6.

### 3.4. Combination of AHK and LBN and Discussion

After investigating the nice properties of AHK and LBN, we now present our robust spectral clustering algorithm that combines these two techniques, and thereby is less sensitive to the scaling parameter selection and noise appearance. For the notational simplicity, we call the integrated algorithm as the AHK Clustering, or AHK directly. Let $X$ be the input dataset of size $n \times m$, where $n$ is the number of data points and $m$ is the number of dimensions (features), our algorithm is detailed in Algorithm 2.

Algorithm 2 undergoes two times of data warping: In Step 1, the original affinity matrix $W$ is constructed using a similarity kernel as appropriate (please refer to Section 2.2 for the kernel selection) according to the specific data characteristics. Then we use LBN as Laplacian normalization on $W$ (Step 2 and 3, check Section 3.3 for more details) and utilize the derived eigendecomposition to construct AHK (Step 4 and 5, check Section 3.1 for more details). After that it comes to the second data warping by extracting the first $c$ nontrivial eigenvectors from the AHK affinity matrix (Step 6). Finally we perform $k$-means on the normalized eigenvectors and label the projected data points. The smoothing parameter $\gamma$ in Step 5 is added to avoid the eigenvalue $\lambda$ from being to small and thereby stabilize the AHK affinity matrix computation. In practice we set $\gamma = 0.001$ by default.

Regarding the computational complexity, eigenvalue decomposition is the most time consuming step and dominates the computation. There are many iterative methods to conduct eigenvalue decomposition (e.g., power iteration [Badeau et al. 2005]), but

**ALGORITHM 2:** AHKClustering($X$,$c$,$\gamma$)

**Input**: $X \in R^{n\times m}$ where $n$ is #instances, $m$ is #features, $c$ is #clusters, and $\gamma$ is an eigenvalue smoothing parameter.
**Output**: Cluster assignments of $n$ instances.

1 Construct the affinity matrix $W \in R^{n\times n}$;
2 Compute the diagonal matrix $D \in R^{n\times n}$ where $D(i,i) = \sum_{j=1}^{n} W(i,j)$ and $D(i,j) = 0$ if $i \neq j$ ;
3 Apply Laplace-Beltrami Normalization $L_{\text{lbn}}$ using Equation 21 and 22 with $\kappa = 1$ ;
4 Extract generalized eigenvectors $\psi(i)$ and corresponding eigenvalues $\lambda_i$, $i = 1, 2, ..., n$ ;
5 Construct $\mathcal{H}_{\text{lbn}}$ matrix with $\psi(i)$ and $\lambda_i$ using $\mathcal{H}(i,j) = \sum_{k=1}^{n}[\frac{1}{\lambda_i+\gamma}\psi_k(i)\psi_k(j)]$ ;
6 Extract the first $c$ nontrivial eigenvectors $\psi'$ of $\mathcal{H}_{\text{lbn}}$, $\psi' = \{\psi'_1, \psi'_2, \ldots, \psi'_c\}$ ;
7 Re-normalize the rows of $\psi' \in R^{n\times c}$ into $Y_i(j) = \psi'_i(j)/(\sum_l \psi'_i(l)^2)^{1/2}$ ;
8 Run $k$-means with $c$ and $Y \in R^{n\times c}$.

in general finding the eigenvalues reduces to matrix multiplications by computing a symbolic determinant, in which the running time is $O(n^3 + n^2 log^2 n)$ [Pan and Chen 1999].

It is worth to notice that AHK has the following significant benefits: 1) It is a stronger form of random walk process by taking all possible paths in entire continuous time scales into consideration, therefore it is more robust and less sensitive to noise or artifacts than other regular kernels. 2) To mitigate the biased contribution of the denominator from some extremely small eigenvalues $\lambda$, AHK introduces a smoothing term $\gamma$ to make computation more stable. 3) To relieve the bias to non-uniform density distribution, AHK employs Laplace-Beltrami normalization (LBN) which can recover the Riemannian manifold under skewed density distribution. In other words, AHK enables better and more stable manifold reconstruction, especially under noise, parameter disturbance, and non-uniform density distribution. Therefore in theory it guarantees the strong adjacency (similarity) among intra-cluster instances even under suboptimal conditions. In the next section, we introduce an affinity transformation to give the clustering algorithm a better insight into the separation between adjacent/overlapping clusters with different density distributions.

## 4. LOCAL DENSITY AFFINITY TRANSFORMATION (LDAT)

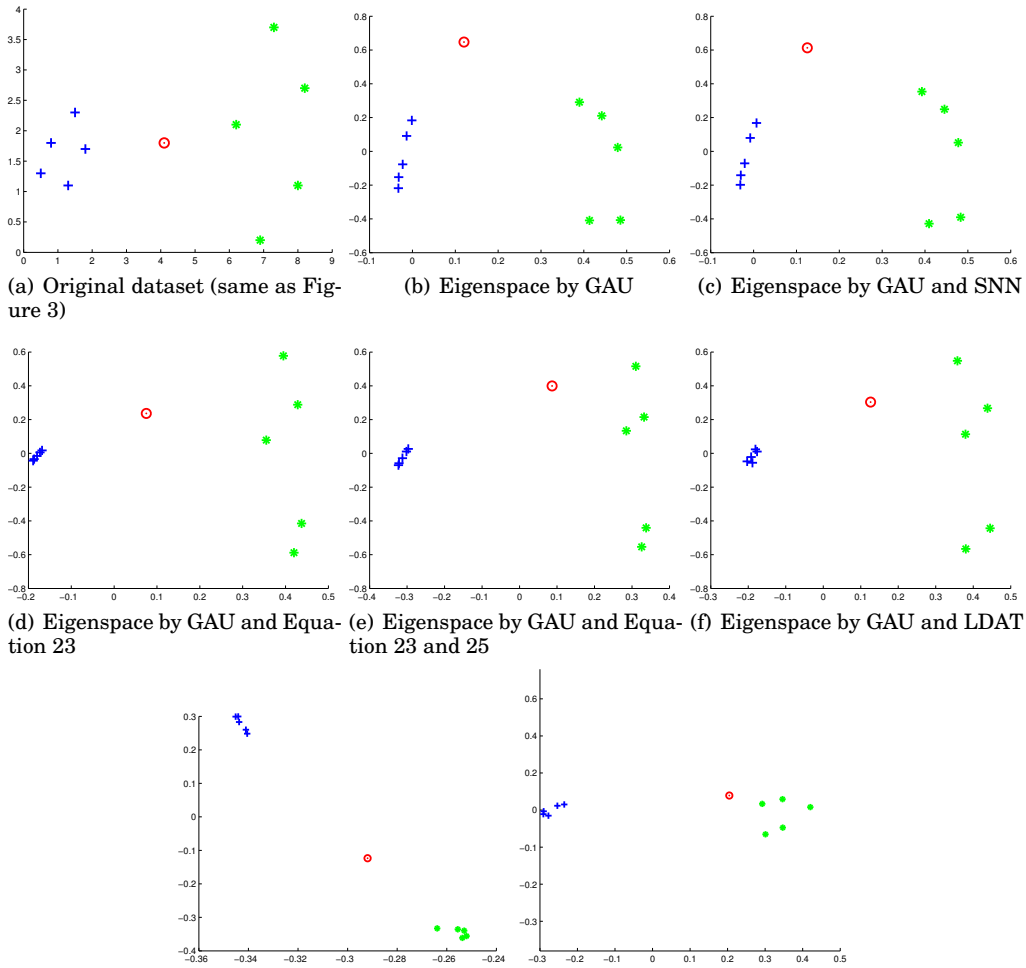
As we discussed in Section 2.2, there are numerous similarity measurements ranging from network connectivity to Gaussian kernels. Unfortunately, few existing approaches took local density into consideration. Some exceptions, such as [Zhang et al. 2011] [Yang et al. 2011] are based on simple approximations of local density that fail to provide the stability against neighborhood perturbation.

In this section we propose a Local Density Affinity Transformation (LDAT) with the following attractive properties: 1) It reveals local density differences for the purpose of correcting density bias; 2) It can be applied on any similarity kernel; 3) It works quite stably with a solid probabilistic interpretation.

**Stage 1 (Step $2$ in Algorithm $3$).** In our research we measure the local density on affinity matrix with a positive random walk normalization as our first step. It provides both probabilistic and local density information by involving degree or volume of each instance, and brings the advantage from the difference between $P(i,j)$ and $P(j,i)$:

$$P(i,j) = \frac{W(i,j)}{\sum_k W(i,k)}, \tag{23}$$

where $P(i,j)$ is the transition probability from $x(i)$ to $x(j)$ and $\sum_k W(i,k)$ is the local volume of $x(i)$ if we quantify $k$ in a certain neighborhood ($W(i,k)$ is non-zero if $x(k)$



(a) Original dataset (same as Figure 3)
(b) Eigenspace by GAU
(c) Eigenspace by GAU and SNN
(d) Eigenspace by GAU and Equation 23
(e) Eigenspace by GAU and Equation 23 and 25
(f) Eigenspace by GAU and LDAT
(g) Eigenspace by AHK
(h) Eigenspace by AHK and LDAT

Fig. 7. 2D Eigenspace derived from the (transformed) affinity matrix of the previous synthetic example (Figure 7(a)), with only the first two non-trivial eigenvectors being plotted. Here we only focus on the relative distances. The eigenspace derived from Gaussian similarity (GAU) is shown in Figure 7(b), while the one from shared nearest neighbors (SNN) on GAU is shown in Figure 7(c). The relative density between blue and green cluster doesn't change much since the projection has no probabilistic transition. Figure 7(d) to 7(f) show the effect of the three steps in our proposed LDAT built GAU. The blue cluster becomes denser after probabilistic transition. Our proposed AHK in Figure 7(g) makes the inner-cluster points even more condense. The combination of AHK+LDAT in Figure 7(h) draws the red point into the green cluster.

is inside $x(i)$'s $k$-nn neighborhood). It means that we only maintain the connections within the $k$ nearest neighborhood ($k$-nn) and remove other distant affinity information. If $W$ faithfully describes the real affinity information, $\sum_k W(i,k)$, as local volume of $x(i)$, is a simple and effective approximation of $x(i)$'s local density. Intuitively, the larger the local volume is, the denser $x(i)$'s local neighborhood is. In general, $P(i,j)$ is different from $P(j,i)$ if the local density distribution between instances $x(i)$ and $x(j)$ is different. Intuitive speaking, if the manifold structure and associated data points can be properly recovered by the positive random walk normalized affinity matrix, a data

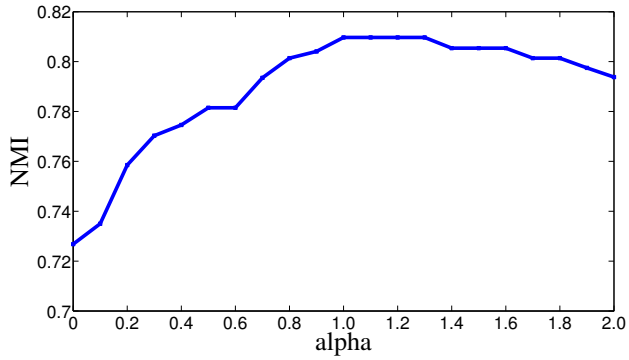



Fig. 8. RWC+LDAT performance on the dataset of Figure 2(a) with reduction factor $\alpha \in [0, 2]$ in Equation 24.

point set with high density before normalization would become even more condense (comparatively) afterward, which can be observed from the blue cluster Figure 7(d).

**Stage 2 (Step $3$ in Algorithm $3$).** Ideally, the transition probability between two points within the same cluster should be larger than two boundary points across two (neighboring) clusters with different densities. The difference between the transition probability must be captured in order to accurately separate different clusters. For example in Figure 3, there should be $P(a, d) \simeq P(d, a) \gg P(a, b)$, and $P(a, b) \ll P(b, a)$ if we consider the local volume difference. In other words, as far as point $a$ is concerned, its affinity to $b$ is relatively smaller compared with its affinity to any other point in the blue cluster. We call the difference between $P(i, j)$ and $P(j, i)$ ***local density bias***.

Our goal is to fix this local density bias by making point $b$ in Figure 3 to be further away from point $a$ than from any point in the green cluster, and thereby assimilate $b$ into the green cluster. We achieve this goal by reducing $P(i, j)$ (if $P(i, j) > P(j, i)$) :

$$\mathcal{P}(i, j) = max[P(i, j) - \alpha(P(i, j) - P(j, i)), 0], \qquad \text{if } P(i, j) > P(j, i), \tag{24}$$

where $\alpha \in [0, \inf]$ is used to control how much reduction is applied to $\mathcal{P}(i, j)$. When $\alpha = 0$, $\mathcal{P}(i, j)$ is the same as a positive random walk normalization $P(i, j)$. When $\alpha > 0$, the local density bias is taken into account. When $\alpha = 1$, $\mathcal{P}(i, j) = P(j, i)$, which translates into $\mathcal{P}(b, a) = P(a, b) < \mathcal{P}(b, c)$ in Figure 3, so that point $b$ can be classified into the green cluster.

Because our goal is to rectify the local density bias, $\alpha = 1$ is a simple and natural choice and our experiments on Figure 2(a) dataset also confirmed that when $\alpha = 1$, it shows the best performance, as shown in Figure 8. Therefore, Equation 24 can be simplified as:

$$\mathcal{P}(i, j) = min(P(i, j), P(j, i)). \tag{25}$$

Although this step looks simple, it actually contributes a lot on classifying the boundary points by "assimilating" them to the point set with similar density. Figure 7(e) shows the effect of Equation 25, where the relative distance between the red point and green cluster become shorter compared with Figure 7(d). From the perspective of any blue point, the red one is farther away than the blue species. Intuitively, even though the red point may initially treat blue points as closer neighbors than the green points, the blue points will "push" it away.

**Stage 3 (Step $4$ in Algorithm $3$).** After applying Equation 25, we employ another positive random walk normalization, which again endows our method with a probabilistic interpretation. Figure 7(f) shows the effect of this second normalization on top of Figure 7(e): the red point is still far away from blue cluster and close to green cluster.

We term the whole process of this affinity transformation as ***Local Density Affinity Transformation*** **(LDAT)**. The entire procedure is documented in Algorithm 3.

**ALGORITHM 3:** LDAT($W$, $k$)

**Input**: Input affinity matrix $W \in R^{n \times n}$ where $n$ is #instances, and $k$ is the neighborhood size.
**Output**: LDAT affinity matrix $W^{(\mathrm{LDAT})}$.

1 For each instances, only keep the $k$-nn affinity information and set all the other as zero in $W$;
2 Apply a positive random walk normalization $P$ on $W$ (Equation 23) ;
3 Construct the reduced $\mathcal{P}$ (Equation 25) ;
4 Employ another positive random walk normalization $W^{(\mathrm{LDAT})}$ on the reduced $\mathcal{P}$.

Superficially speaking, LDAT is similar to SNN (shared nearest neighbors [Jarvis and Patrick 1973] [Ertoz et al. 2002] [Steinbach et al. 2003]) except for the first random walk normalization in Algorithm 3. But in fact this step is of great importance. If there is no random walk normalization before Equation 25, most part of the matrix is still symmetric (considering $k$ is not very small in step 1). In this case Equation 25 would not have real impact on the final performance, as shown in the comparison between Figure 7(b) and 7(c). However in our proposed LDAT, the first random walk normalization between step 1 and step 3 delivers awareness of density difference. Therefore the subsequent reduced $\mathcal{P}$ is capable to correct the bias originating from different cluster densities. Another positive side-effect of the first random walk normalization in LDAT is that it supplies stability with different setting of $k$, while SNN suffers a lot from such perturbation (Figure 14 in Section 6).

We now analyze the effect of LDAT in theory, which is closely connected to NCut (normalized cut). Suppose there are only two point sets $X$ and $Y$ in the entire dataset $V$ where $V = X \cup Y$, the corresponding NCut is defined as follows [Luxburg 2007]:

$$NCut(X,Y) = \frac{C(X,Y)}{assoc(X,V)} + \frac{C(Y,X)}{assoc(Y,V)}, \tag{26}$$

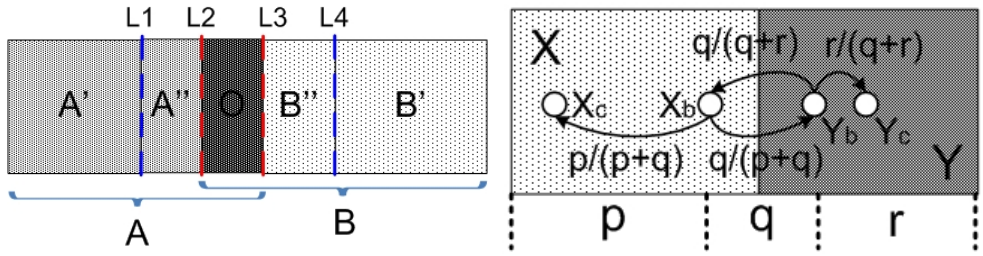
where $C(X,Y) = \sum_{i \in X, j \in Y} W_{ij}$, and $assoc(X,V) = \sum_{i \in X, j \in V} W_{ij}$. If we restraint the connections of each node in $k$-nn neighborhood, Equation 26 can be rewritten as:

$$NCut(X,Y) = \frac{C(X,Y)}{v(X)} + \frac{C(Y,X)}{v(Y)} = P(X|Y) + P(Y|X), \tag{27}$$

where $v(X)$ is the summation of volume of all the instances in $X$, and $P(X|Y)$ is the transition probability from any instance in cluster $Y$ to any instance in cluster $X$. The minimization of NCut actually seeks a cut through the graph such that a random walk seldom transitions from $X$ into $Y$ or vice versa. However NCut has strong density bias when dealing with datasets with heterogeneous density distributions, which can be proven as follows.

PROPOSITION 1. *Let graph $\mathcal{G}$ be $k$-nn connected and non bi-partite. And the affinity have been normalized by the positive random walk Laplacian. For two overlapping sets $A, B \subset V$ and $A \cup B = V$, given that $A$ is denser than $B$, but intra-cluster density is uniform and the number of nodes are very similar. NCut may fail to provide the best cut due to the local density bias.*

**Proof:** As shown in Figure 9(a), define $O$ as the overlapping area, $A''$ as the area inside $A$ that is close to $O$, and $B''$ as the area inside $B$ that is close to $O$. Assume that under certain $k$-nn constraint, connections only exist between two adjacent area.




(a) (b)

Fig. 9. In Figure 9(a), A and B are two overlapping subsets and A is denser than B. O is the overlapping part and apparently has the highest density. Suppose there is only one cut, L3 would be the best choice to maintain uniform inner-cluster density distribution. Traditional NCut fails to cut along L3 as proven in Proposition 1. But LDAT can correct the density bias of NCut and cut along L3, which is proven in Proposition 2. Figure 9(b) shows the connections in the boundary area between two adjacent sets X and Y. The average density of X and Y are $p$ and $r$ respectively. The density of boundary area is $q$. The change of connection weight before and after LDAT is analyzed in the proof of Proposition 1 and 2.

In other words, there is no connection between $A'$ and $O$, $A''$ and $B''$, and $B'$ and $O$. Apparently the overlapping area $O$ has the highest density.

Now let's firstly "zoom-in" to analyze the cutting area. Suppose a cut separate $V$ into two adjacent sets $X$ and $Y$, where $X \cup Y = V$ and $X \cap Y = \emptyset$. Apparently there is:

$$v(V) = v(X) + v(Y), \tag{28}$$

and since the affinity has been normalized by positive random walk normalization, there is:

$$v(X) = |X|, \tag{29}$$

where $|X|$ is the number of instance in $X$. Suppose $|X| \sim |Y| \sim T$, there are:

$$\begin{aligned} NCut(X,Y) &= \frac{C(X,Y)}{v(X)} + \frac{C(Y,X)}{v(Y)} \\ &= \frac{1}{T}(C(X,Y) + C(Y,X)). \end{aligned} \tag{30}$$

Now we only focus on the value of $C(X,Y) + C(Y,X)$ under different density distributions. Figure 9(b) shows the connection between $X$ and $Y$. We suppose the simplest 2-nn neighborhood and the average density of $X$ and $Y$ are $p$ and $r$ respectively. The average density of the boundary area is $q$. $X_b$ are the points in the boundary area of $X$ close to $Y$, similarly $Y_b$ are the points in the boundary area of $Y$ close to $X$. $X_c$ and $Y_c$ are the other points inside $X$ and $Y$. There is $C(X,Y) = C(X_b, Y_b)$ and $C(Y,X) = C(Y_b, X_b)$. We can assume $C(Y_b, X_b) = q\eta/(q+r)$ and $C(Y_b, Y_c) = r\eta/(q+r)$, and $C(X_b, Y_b) = q\eta/(p+q)$ and $C(X_b, X_c) = p\eta/(p+q)$, where $\eta$ is a connection factor. $C(X,Y) + C(Y,X)$ will change under different density distributions.

(1) If $p \sim r$, $q$ would also have similar value. Then $C(X_b, Y_b) + C(Y_b, X_b) = \eta$;
(2) Suppose $p < r$, then we have $p < q < r$ (since $X \cap Y = \emptyset$). There is:

$$C(X_b, Y_b) + C(Y_b, X_b) = \frac{q\eta}{q+r} + \frac{q\eta}{p+q} = \frac{(2q^2 + pq + qr)\eta}{q^2 + pq + qr + pr}. \tag{31}$$

If $q^2 > pr$, there is $C(X_b, Y_b) + C(Y_b, X_b) > \eta$.

In short, if $|X|$ is similar to $|Y|$, the value of Equation 30 is dominated by $C(X_b, Y_b) + C(Y_b, X_b)$. And the minimization of conventional NCut makes it less likely to cut along

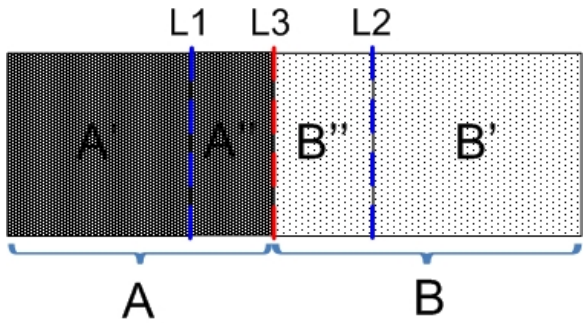



Fig. 10. Special case: A and B are two adjacent but non-overlapping subsets and A is denser than B. In order to maintain uniform inner-cluster density distribution, L3 is the best cut. Traditional NCut fails to cut along L3 as proven in Proposition 1. But LDAT can correct the density bias of NCut and cut along L3. The effect of LDAT is proven in Proposition 2.

the boundary where $q^2 > pr$. Since NCut value under the condition of $p \ll r$ or $p \gg r$ is very possible to be greater than that under $p \sim r$, NCut is less likely to cut along $L_3$, since the two sides have the most different density distribution.

A special case is that A and B are adjacent (A is much more denser than B) but there is (almost) no overlapping area, as shown in Figure 10. We make the similar assumption that under certain $k$-nn constraint, connections only exist between two adjacent area. In other words, there is no connection between $A'$ and $B''$, and $A''$ and $B'$. The same deduction of Equation 30 and analysis still holds as in the general case in Figure 9(a). Since density on the two sides of $L3$ changes a lot, the conventional NCut tends to cut along $L1$ or $L2$ rather than $L3$. □

PROPOSITION 2. *LDAT alleviates the density bias of NCut through lowering the NCut value.*

**Proof:** To prove LDAT alleviates the density bias under different density distribution, we suppose $p < r$. In Figure 9(b) (after random walk normalization), $C(Y_b, X_b) < C(X_b, Y_b)$. Step 3 in Algorithm 3 makes $C(X_b, Y_b) \leftarrow C(Y_b, X_b) = q\eta/(q+r)$. After the second random walk normalization (Step 4), there is:

$$\begin{aligned} C(X_b, Y_b) + C(Y_b, X_b) &= \frac{q\eta/(q+r)}{p/(p+q) + q/(q+r)} + \frac{q\eta}{q+r} \\ &< \frac{q\eta/(q+r)}{p/(q+r) + q/(q+r)} + \frac{q\eta}{q+r} = \frac{q\eta}{p+q} + \frac{q\eta}{q+r}, \end{aligned} \tag{32}$$

therefore LDAT lowers the NCut value compared with Equation 31. Furthermore,

$$\begin{aligned} C(X_b, Y_b) + C(Y_b, X_b) &= \frac{q\eta/(q+r)}{p/(p+q) + q/(q+r)} + \frac{q\eta}{q+r} \\ &= \frac{(2pq^2 + q^3 + pqr + rq^2)\eta + (pq^2 + q^3)\eta}{(2pq^2 + q^3 + pqr + rq^2) + (pqr + pr^2)}, \end{aligned} \tag{33}$$

and we have

$$\frac{(pq^2 + q^3)\eta}{pqr + pr^2} = \frac{(pq^2 + q^3)\eta/(pq)}{(pqr + pr^2)/(pq)} = \frac{(q + q^2/p)\eta}{r + r^2/q}. \tag{34}$$

Suppose $q = \beta p$ and $r = (\beta + \triangle)p$, where $\beta > 1$ and $\triangle > 0$, the condition that makes Equation 33 smaller than $\eta$ is

$$\begin{aligned} &\frac{q + q^2/p}{r + r^2/q} = \frac{\beta p + \beta^2 p^2/p}{(\beta + \triangle)p + (\beta + \triangle)^2 p^2/(\beta p)} = \frac{\beta + \beta^2}{(\beta + \triangle) + (\beta + \triangle)^2/\beta} < 1 \\ &\Rightarrow \frac{\triangle^2}{\beta} + 3\triangle - (\beta^2 - \beta) > 0 \\ &\Rightarrow \triangle > \frac{(\sqrt{9 + 4(\beta - 1)} - 3)\beta}{2}. \end{aligned} \tag{35}$$

(1) On the one hand, from the proof of Proposition 1 we know that traditional NCut doesn't cut along L3 when $q^2 > qr$. This condition can be also represented as:

$$\beta^2 > \beta + \triangle. \tag{36}$$

Assume that both Equation 35 and 36 holds, we have:

$$\begin{aligned} &\beta^2 - \beta > \triangle > \frac{(\sqrt{9 + 4(\beta - 1)} - 3)\beta}{2} \\ &\Rightarrow \beta - 1 > \frac{\sqrt{9 + 4(\beta - 1)} - 3}{2} \\ &\Rightarrow \beta > 1, \end{aligned} \tag{37}$$

which always holds according to $p < r$.

(2) On the other hand, traditional NCut cuts along L3 when $q^2 < qr$. This condition can be also represented as:

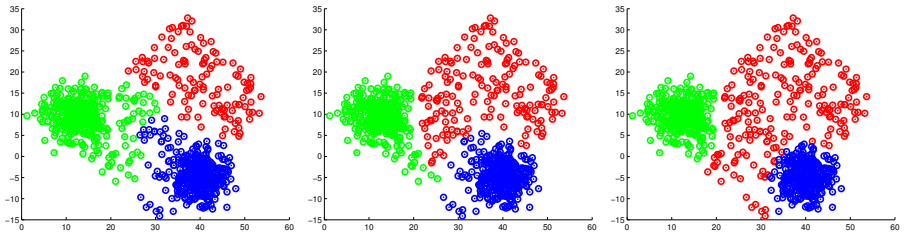
$$\beta^2 < \beta + \triangle \Rightarrow \triangle > \beta^2 - \beta. \tag{38}$$

We also have the condition $\triangle > \beta^2 - \beta > \frac{(\sqrt{9+4(\beta-1)}-3)\beta}{2}$ holds.

Therefore, after LDAT the value of NCut becomes smaller than $\eta$ where exists density difference. The effect of LDAT is not so obvious inside each cluster of the area with similar density. However, if $p << r$, $C(X_b, Y_b) + C(Y_b, X_b)$ becomes much smaller than that in the uniform distribution. In short, if $|X| \sim |Y|$, after LDAT $C(X_b, Y_b) + C(Y_b, X_b)$ between areas with the most different density carries the smallest value. So in Figure 9(a) it leads to the cutting line to be much closer to $L3$. It means that LDAT alleviates the density bias of NCut. In the special case of Figure 10, the cutting line with the most different density distribution on the two sides is $L3$, therefore after LDAT, NCut also tends to cut along $L3$ other than $L1$ and $L2$. □

In Figure 11 the positive effect of LDAT is quite evident when applied to the conventional spectral algorithms (RWC). Compared with the simple RWC result in Figure 2(c), RWC+LDAT boosted performance more than $25\%$ (Figure 11(b)), which will be further verified in Section 6. Note that since LDAT is an straightforward transformation of affinity matrix, it can work well with any type of similarity kernels, which gives rise to a novel feature compared with other methods only rely on Euclidean space [Zhang et al. 2011] [Yang et al. 2011] [Correa and Lindstrom 2012].

However, LDAT has a strong assumption that the affinity matrix $W$ contains sufficient and accurate neighborhood information. Although this requirement can be easily satisfied for those small and simple datasets, it becomes quite challenging when the datasets are large and complex (with high-dimensions). Especially if we only keep $k$-nn neighborhood for each instance, the constrained information will be highly vulnerable to the change of $k$ given only simple kernel function like Gaussian kernel (see blue



(a) RWC+SNN ($W^{(GLS)}$ with $\sigma_{(G)} = 2$), $NMI = 0.6686$ (b) RWC+LDAT ($W^{(GLS)}$ with $\sigma_{(G)} = 2$), $NMI = 0.7555$ (c) AHK+LDAT ($W^{(GLS)}$ with $\sigma_{(G)} = 2$), **NMI=0.8097**

Fig. 11. Figure 11(a) and 11(b) show the effect of shared nearest neighbor (SNN) and our proposed LDAT, both built upon $W^{(GLS)}$ and a positive random walk normalization (RWC). It demonstrates LDAT's advantage of better recognizing density differences among clusters than SNN and other algorithms shown in Figure 2 and 4. The LDAT built upon AHK, shown in 11(c), has the best NMI result through being aware of both density and manifold structures.

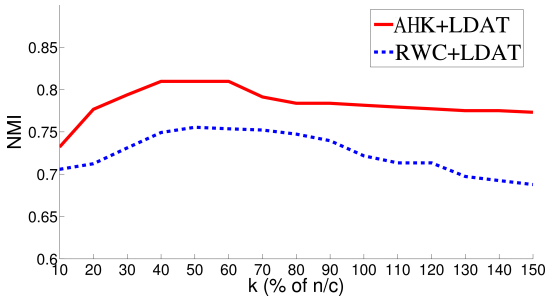



Fig. 12. LDAT performance on the dataset in Figure 2(a) with different neighborhood size $k$. $k$ is set as the percentage of $n/c$ ($n$ is #instances and $c$ is #clusters). AHK+LDAT has better and more stable performance than RWC+LDAT as $k$ changes.

curve in Figure 12). In our research we perform LDAT on top of AHK, in order to reflect both manifold-aware and density-aware structure. In the next section we will describe and further analyze the combined framework.

## 5. THE COMPLETE PROPOSED FRAMEWORK

We have introduced a robust heat-diffusion-based kernel (AHK, Section 3) and a local-density-bias-corrected affinity transformation (LDAT, Section 4). We now incorporate these two techniques into a systematical framework to provide a more effective and powerful clustering algorithm, as documented in Algorithm 4.

Here AHK stably provides sufficient and accurate affinity information of the high-dimensional datasets with complex distribution. Therefore AHK supplies a sturdy platform for the subsequent LDAT. LDAT is also extremely crucial, due to its nice properties of precisely considering the local density and cleanly separating the boundary instances between clusters with diverse densities. It makes corrections and improvements to the traditional NCut-based spectral clustering. By systematically integrating AHK and LDAT, we set up a robust manifold-and-density-aware clustering algorithm. Its running time is $O(n^3 + n^2 log^2 n)$ when the eigen-decomposition method [Pan and Chen 1999] is used.

Although AHK is quite effective in recognizing manifolds even under skewed conditions, it has a negative side-effect of getting over-connection due to the natural properties of random walk. This side-effect might lead to an undesirable uniform affinity

**ALGORITHM 4:** AHK+LDAT-Clustering($X$, $c$, $k$)

**Input**: Input data $X \in R^{n \times m}$ where $n$ is #instances and $m$ is #features, $c$ is #clusters, and $k$ is the neighborhood size.
**Output**: Cluster assignments of $n$ instances.

1 Construct the AHK affinity matrix $\mathcal{H}_{\text{lbn}}$ as in Algorithm 2;
2 Set the diagonal of $\mathcal{H}_{\text{lbn}}$ to be zero to avoid over-diffusion ;
3 Only keep the $k$-nn affinity information for each instances ;
4 Transform the $\mathcal{H}_{\text{lbn}}$ using LDAT as in Algorithm 3 ;
5 Compute the first $c$ eigenvectors $\psi$, $\psi = \{\psi_1, \psi_2, \ldots, \psi_c\}$ ;
6 Extract the first $c$ nontrivial eigenvectors $\psi$, $\psi = \{\psi_1, \psi_2, \ldots, \psi_c\}$ ;
7 Re-normalize the rows of $\psi \in R^{n \times c}$ into $Y_i(j) = \psi_i(j)/(\sum_l \psi_i(l)^2)^{1/2}$ ;
8 Run $k$-means with $c$ and $Y \in R^{n \times c}$.

since an infinite number of pathes between any two instances tend to draw them deceivably close to each other without proper control. To mitigate this problem, we only use the off-diagonal terms while ignoring the diagonal term of AHK (Step 2 of Algorithm 4). It avoids the infinite diffusion getting lost on those instances with a lot of connections, as their degrees are enormous [Luxburg et al. 2010].

In order to show the positive effect of AHK and LDAT respectively, we perform case study separately.

(1) First of all, AHK helps to improve the manifold reconstruction quality of LDAT. Compared with RWC+LDAT, we test AHK+LDAT on the synthetic example of Figure 2(a), the result is shown in Figure 11(c). It can be observed that AHK can directly helps the subsequent LDAT to gain more cluster-aware separation (by clearly separating the blue and green clusters from the red one), and obtain $7\%+$ performance increment compared with RWC+LDAT. Moreover, AHK+LDAT demonstrates more stable performance than RWC+LDAT as the neighborhood size $k$ changes, which is shown in Figure 12.
(2) Secondly, to confirm AHK+LDAT's superiority on AHK alone, we test AHK+LDAT on the 20ngB dataset and Figure 6(i) shows the corresponding manifold reconstruction. Clearly LDAT helps to make the intra-cluster instances more condense to their cluster center. It boost the performance more than $7\%$ compared with AHK alone.

Supplementary experiments on real benchmark datasets will be presented in Section 6 to further support the above discovery.

## 6. EXPERIMENTAL RESULTS AND QUANTITATIVE ANALYSIS

In this section we analyze and verify our proposed AHK, LDAT, and the combination of AHK+LDAT in terms of clustering effectiveness and robustness.

### 6.1. Experimental Setup

**Datasets.** We verify the effectiveness of our proposed methods by evaluating on eighteen benchmark datasets (Table I) with three types of affinity constructions. Gaussian kernel is used for the first seven datasets from UCI. Network connectivity with undirected edges (all weighted as $1$) are applied to the next five datasets ranging from political blogs to scientific paper citation domains. The last six text datasets use cosine similarity: the first three are the subsets of 20 Newsgroup [Joachims 1996] and the last three are the subsets of RVC1 [Lewis et al. 2004]. To reduce sampling bias, we randomly samples from different clusters to make each (sub)datasets 20 times, and record the average clustering performances. 20ngA contains $200$ documents from misc.forsale and soc.religion.christian. 20ngB adds $200$ documents from talk.politics.guns to 20ngA.

Table I. Statistics of our evaluation datasets.

| | Data Set | # inst. | # attr. | # clus. |
|---|---|---|---|---|
| 1 | Wine | 178 | 13 | 3 |
| 2 | Glass | 214 | 9 | 6 |
| 3 | Vehicle | 846 | 18 | 4 |
| 4 | Vowel | 990 | 11 | 11 |
| 5 | Yeast | 1484 | 8 | 10 |
| 6 | Images | 2100 | 19 | 7 |
| 7 | Pendigits | 3498 | 16 | 10 |
| 8 | Polbooks | 105 | 105 | 3 |
| 9 | UMBC | 404 | 404 | 2 |
| 10 | MSP | 1067 | 1067 | 2 |
| 11 | Citeseer | 2114 | 2114 | 6 |
| 12 | Cora | 2485 | 2485 | 7 |
| 13 | 20ngA | 400 | 400 | 2 |
| 14 | 20ngB | 600 | 600 | 3 |
| 15 | 20ngC | 800 | 800 | 4 |
| 16 | RCV1-2 | 1600 | 29992 | 2 |
| 17 | RCV1-3 | 2400 | 29992 | 3 |
| 18 | RCV1-4 | 3200 | 29992 | 4 |

20ngC adds $200$ documents from rec.sport.baseball to 20ngB. RCV1-2 contains $1600$ documents, among them $800$ from C15 and another $800$ from ECAT. We add $800$ documents from GCAT to RCV1-2 to create RCV1-3. RCV1-4 has $800$ more documents from MCAT upon RCV1-3.

**Baselines.** Eight popularly used clustering algorithms are chosen for comparison: symmetric normalized spectral clustering (NJW) [Ng et al. 2002] and random walk spectral clustering (RWC) [Meila and Shi 2001] are chosen since they are the classic spectral clustering algorithms. RWC+SNN is RWC algorithm built upon shared nearest neighbors (SNN) [Jarvis and Patrick 1973] [Ertoz et al. 2002] to update similarity. Additionally, we choose Graph Degree Linkage (GDL) [Zhang et al. 2012] as the representative of recent graph-based methods with agglomerative (or hierarchical) modeling. Density-based spatial clustering of applications with noise (DBSCAN) [Ester et al. 1996] [Tran et al. 2013] is a density-driven clustering algorithm because it finds a number of clusters starting from the estimated density distribution of corresponding instances. Self-Tuning (ST) spectral clustering [Zelnik-Manor and Perona 2004] and local density adaptive similarity clustering (SCDA) [Zhang et al. 2011] are two locally adaptive clustering methods which adjust classification according to different neighborhood density measurements. We also select the $k$-nn Diffusion Maps clustering (kDM) [Coifman and Lafon 2006] as another candidate since it is a diffusion-based algorithm with robustness on noise perturbation.

**Evaluation Metrics.** We use normalized mutual information (NMI) [Strehl and Ghosh 2003] as the evaluation metric due to its popularity and its information-theoretical interpretation. Suppose $S \in R^{n\times 1}$ is the result label vector for all data instances generated by one particular clustering algorithm and $T \in R^{n\times 1}$ is the true label vector. The NMI score is calculated as follows:

$$NMI(S,T) = \frac{I(S;T)}{\sqrt{H(S)\times H(T)}}, \tag{39}$$

where $H(S)$ and $H(T)$ are the entropies, and $I(S;T)$ is the mutual information between $S$ and $T$. The NMI score is normalized by their entropies and it ranges from zero to one where the larger score indicates the better clustering result.

**Parameter Settings.** As most of the spectral clustering algorithms assume that the number of clusters $c$ is known a priori, so do our algorithms. Our proposed methods

have three parameters: Gaussian scaling factor $\sigma$ (if Gaussian kernel is used to construct AHK), the size of neighborhood $k$ (to control $k$-nn connections), and the reduction parameter $\alpha$ in LDAT.

Gaussian scaling factor $\sigma$ is also used in the other included clustering competitors. To obtain an adaptive parameter and at the same time preserve local density information, we compute the average distance between each instance to its $q$-nearest neighbors, and use this value (noted as $\sigma_q$) to set the Gaussian scaling parameter. In the remaining section, we will test the algorithm performance with different $q$ in the range of $[2, 50]$, with $1$ as the step size. When we test the stability of algorithms against the other two parameters $k$ and $\alpha$ (Section 6.5 and 6.6), we fix $q = 2$ by default.

For a proper neighborhood size $k$, we set its value as half of the average cluster size $k = n/(2c)$, where $n$ is the number of instances in a dataset and $c$ is the number of clusters. We assume that this is a safe choice for each instance to assemble its true local density. In Section 6.5 we will further test the algorithm sensitivity to $k$ in the range of $[10\%, 100\%]$ of $\frac{n}{c}$ with $10\%$ as step size to verify the rationality of $k = 50\%$.

In Section 4, we already discussed the effect of $\alpha$ on LDAT. In our general experiments, we use $\alpha = 1$ by default but we will also test RWC+LDAT and AHK+LDAT with different value of $\alpha \in [0, 2]$ in Section 6.6.

For DBSCAN experiments, we set $Eps$, the neighborhood radius, in the same way as we set $\sigma_q$. We assign $minPts$, minimal number of instances considered as a cluster, in the range of $[10, min(n/c, 300)]$, and only record the best result among them.

Table II. Comparison of NMI between AHK+LDAT and the other seven methods on twelve datasets. Experiments with the first seven datasets (from Wine to Pendigits) make use of the Gaussian kernel with $\sigma_q$ ($q \in [2, 50]$), and the best score across all the $q$ settings is shown for each algorithm and dataset. Experiments with the datasets from Polbooks to Cora use network connectivity. The bold-faced numbers indicate the best method for a particular dataset. The numbers in parentheses are the rankings of the corresponding methods. $AVG^{(GAU)}$ and $AVG^{(NET)}$ are the average NMI of the algorithms using Gaussian kernel and network connectivity respectively.

| Dataset | RWC | kDM | ST | SCDA | NJW | DBSCAN | GDL | AHK+LDAT |
|---|---|---|---|---|---|---|---|---|
| Wine | 0.4355 (6) | 0.4421 (4) | **0.4604** (1) | 0.4179 (7) | 0.4375 (5) | 0.2341 (8) | **0.4604** (1) | 0.4493 (3) |
| Glass | 0.3615 (8) | 0.3858 (5) | 0.3813 (6) | 0.4054 (3) | 0.3686 (7) | **0.4445** (1) | 0.4018 (4) | 0.4325 (2) |
| Vehicle | 0.1876 (3) | 0.1492 (4) | 0.0964 (5) | 0.0901 (7) | 0.1968 (2) | 0.0745 (8) | 0.0964 (5) | **0.2476** (1) |
| Vowel | 0.4109 (5) | 0.4249 (2) | 0.4094 (6) | 0.3196 (7) | 0.4205 (4) | 0.0983 (8) | 0.4206 (3) | **0.4351** (1) |
| Yeast | **0.3019** (1) | 0.2876 (2) | 0.2694 (5) | 0.2701 (4) | 0.2619 (6) | 0.0587 (8) | 0.2591 (7) | 0.2811 (3) |
| Images | 0.5581 (4) | 0.5958 (2) | 0.4321 (7) | 0.4981 (5) | 0.5887 (3) | 0.3296 (8) | 0.4466 (6) | **0.6746** (1) |
| Pendigits | 0.7131 (4) | 0.7129 (5) | 0.5972 (7) | 0.7056 (6) | 0.7315 (3) | 0.5046 (8) | 0.7858 (2) | **0.8787** (1) |
| $AVG^{(GAU)}$ | 0.4241 (4) | 0.4283 (3) | 0.3780 (7) | 0.3867 (6) | 0.4294 (2) | 0.2492 (8) | 0.4101 (5) | **0.4856** (1) |
| Polbooks | **0.5862** (1) | 0.5745 (2) | 0.5629 (3) | — | 0.5423 (4) | 0.4325 (6) | — | 0.5402 (5) |
| UMBC | 0.0241 (6) | **0.7488** (1) | 0.7375 (2) | — | 0.7375 (2) | 0.1136 (5) | — | 0.7151 (4) |
| MSP | 0.0114 (5) | 0.0114 (5) | 0.0231 (4) | — | 0.0541 (3) | 0.0887 (2) | — | **0.2004** (1) |
| Citeseer | 0.3139 (5) | 0.3628 (3) | 0.3524 (4) | — | 0.3728 (2) | 0.2073 (6) | — | **0.3871** (1) |
| Cora | 0.1434 (6) | 0.2551 (3) | 0.2051 (5) | — | 0.3966 (2) | 0.2103 (4) | — | **0.4325** (1) |
| $AVG^{(NET)}$ | 0.2158 (5) | 0.3905 (3) | 0.3762 (4) | — | 0.4207 (2) | 0.2103 (6) | — | **0.4551** (1) |

### 6.2. General Comparison with Different Affinity Constructions

Table II and III summarize the clustering performance of seven algorithms: RWC, kDM, ST, SCDA, NJW, DBSCAN, GDL, and our proposed AHK+LDAT. Specifically, for the first seven datasets using Gaussian kernel, we document the best performance across $q \in [2, 50]$ for each algorithm. SCDA and GDL are only defined on Euclidean distance/space and therefore could not work with network connection and cosine similarity.

Generally speaking, AHK+LDAT outperforms those selected algorithms across the three commonly-used affinity construction methods. In Table II our AHK+LDAT shows

Table III. Comparison on text datasets, each of which has an increasing number of clusters. The bold-faced numbers indicate the best method for a particular dataset. The numbers in parentheses are the rankings of the corresponding methods. $AVG^{(COS)}$ is the average performance scores.

| Dataset | RWC | kDM | ST | SCDA | NJW | DBSCAN | GDL | AHK+LDAT |
|---|---|---|---|---|---|---|---|---|
| 20ngA | 0.0235 (5) | 0.0235 (5) | 0.0497 (4) | — | 0.1002 (3) | 0.3456 (2) | — | **0.6916** (1) |
| 20ngB | 0.3321 (5) | 0.4625 (2) | 0.3435 (4) | — | 0.3487 (3) | 0.2429 (6) | — | **0.7213** (1) |
| 20ngC | 0.2438 (5) | 0.3368 (2) | 0.2958 (3) | — | 0.2927 (4) | 0.2058 (6) | — | **0.7076** (1) |
| RCV1-2 | 0.2204 (5) | 0.2249 (4) | **0.4039** (1) | — | 0.0417 (6) | 0.3134 (2) | — | 0.2341 (3) |
| RCV1-3 | 0.2172 (6) | 0.5220 (2) | 0.4039 (4) | — | 0.4134 (3) | 0.3523 (5) | — | **0.5467** (1) |
| RCV1-4 | 0.4546 (3) | 0.3638 (4) | 0.2889 (6) | — | 0.5136 (2) | 0.3478 (5) | — | **0.5438** (1) |
| $AVG^{(COS)}$ | 0.2486 (6) | 0.3223 (2) | 0.2635 (5) | — | 0.2851 (4) | 0.3013 (3) | — | **0.5742** (1) |

the best average score. 1) **Gaussian kernel**: AHK+LDAT obtains $0.4856$ average NMI which is $13.09\%$ higher than the second best algorithm NJW, and $16.14\%$ better than RWC. 2) **Network connectivity**: the average NMI of AHK+LDAT reaches $0.4551$, which is $8.18\%$ higher than the second best algorithm (NJW), and $110.89\%$ better than RWC. Table III shows that for datasets with an increasing number of clusters, our AHK+LDAT on **cosine similarity** has better and more stable performance than the other algorithms. In particular, AHK+LDAT outperforms the second best method kDM by $78.16\%$.

Our AHK+LDAT either has the best or ranks top three over all the candidate algorithms for each dataset. The only two exceptions are Polbooks and UMBC. However, for UMBC our AHK+LDAT score is more than $95\%$ of the best score from kDM. As for Polbooks, though the NMI score of AHK+LDAT is only about $92\%$ of the best performance, the dataset size is quite small. Intuitively, the density distribution and its variation on such a small dataset is not obvious due to the small sample size v.s. that of a larger one.

Compared with the other algorithms, DBSCAN fails miserably since it is mainly defined in Euclidean space and suffers from the "curse of dimensionality" and lack of manifold awareness. GDL, ST and SCDA, although based on the theory that supports local density adaptation, are unable to maintain desirable performance across all the datasets, which is mainly caused by their suboptimal local density approximations. Originated from diffusion equations, kDM shows its stability on all the three types of datasets/kernel functions. NJW has comparable performance on Table II but not Table III, partially due to that it does not have any correction for local density bias.

### 6.3. Respective Effect of AHK and LDAT

To verify the effect of AHK and LDAT respectively, we document the experiments of RWC, RWC+SNN, RWC+LDAT, AHK, and AHK+LDAT in Table IV.

(1) For the data experiments using Gaussian kernel, AHK increases NMI by $5.43\%$ compared with RWC, while LDAT boosts up $13.90\%$. Compared with RWC+SNN, RWC+LDAT raise $3.12\%$. Therefore here the effect of LDAT is more obvious than that of AHK. Therefore although both using LDAT, AHK+LDAT is only $2\%$ better than RWC+LDAT. And AHK+LDAT outperforms AHK by about $20\%$.
(2) For those network datasets, AHK increases $93.47\%$ while LDAT only raises about $2\%$ over RWC. It means AHK helps a lot here: AHK+LDAT raises $9\%$ compared with AHK only, but on the other hand AHK+LDAT is $107\%$ better than RWC+LDAT.
(3) In the experiments of text dataset, AHK enhances performance by $99.68\%$ and LDAT $86.89\%$ when both of them are compared with RWC. AHK+LDAT outperforms RWC+LDAT by $23.59\%$, and outperforms RWC+SNN by $26.23\%$. On the other hand, AHK+LDAT also outperforms AHK by about $15.67\%$.

Table IV. Comparison between RWC, RWC+SNN, RWC+LDAT and AHK, AHK+SNN and AHK+LDAT. Experiments with the first seven datasets (from Wine to Pendigits) make use of the Gaussian kernel with $\sigma_q$ ($q \in [2, 50]$), and the best score across all the $q$ settings is shown for each algorithm and dataset. Experiments with the datasets from Polbooks to Cora use network connectivity. Cosine kernel is applied on the last six text datasets. The bold-faced numbers indicate the best method for a particular dataset. The numbers in parentheses are the rankings of the corresponding methods. $AVG^{(GAU)}$, $AVG^{(NET)}$ and $AVG^{(COS)}$ are the average NMI of the algorithms using Gaussian kernel, network connectivity and cosine kernel respectively.

| Dataset | RWC | RWC+SNN | RWC+LDAT | AHK | AHK+SNN | AHK+LDAT |
|---|---|---|---|---|---|---|
| Wine | 0.4355 (5) | 0.4375 (3) | 0.4375 (3) | 0.4244 (6) | 0.4417 (2) | **0.4493** (1) |
| Glass | 0.3615 (6) | 0.4575 (2) | **0.4618** (1) | 0.4266 (4) | 0.4067 (5) | 0.4325 (3) |
| Vehicle | 0.1876 (6) | 0.2031 (4) | 0.2173 (3) | 0.2262 (2) | 0.1880 (5) | **0.2476** (1) |
| Vowel | 0.4109 (4) | 0.3725 (6) | 0.4321 (3) | **0.4432** (1) | 0.4008 (5) | 0.4351 (2) |
| Yeast | **0.3019** (1) | 0.2974 (2) | 0.2763 (4) | 0.2571 (6) | 0.2756 (5) | 0.2811 (3) |
| Images | 0.5581 (6) | 0.6577 (3) | **0.6926** (1) | 0.5751 (5) | 0.5991 (4) | 0.6746 (2) |
| Pendigits | 0.7131 (6) | 0.8068 (3) | 0.8157 (2) | 0.7328 (5) | 0.7701 (4) | **0.8787** (1) |
| $AVG^{(GAU)}$ | 0.4241 (6) | 0.4618 (3) | 0.4762 (2) | 0.4408 (4) | 0.4403 (5) | **0.4856** (1) |
| Polbooks | **0.5862** (1) | 0.5745 (3) | 0.5667 (4) | 0.5833 (2) | 0.5401 (6) | 0.5402 (5) |
| UMBC | 0.0241 (5) | **0.7488** (1) | 0.0241 (5) | 0.5927 (4) | 0.5942 (3) | 0.7151 (2) |
| MSP | 0.0114 (6) | 0.0756 (4) | 0.0124 (5) | 0.1383 (3) | 0.1580 (2) | **0.2004** (1) |
| Citeseer | 0.3139 (5) | 0.3644 (3) | 0.3324 (4) | 0.3697 (2) | 0.2873 (6) | **0.3871** (1) |
| Cora | 0.1434 (6) | 0.2550 (4) | 0.1640 (5) | 0.4037 (2) | 0.3522 (3) | **0.4325** (1) |
| $AVG^{(NET)}$ | 0.2158 (6) | 0.4037 (3) | 0.2200 (5) | 0.4175 (2) | 0.3864 (4) | **0.4551** (1) |
| 20ngA | 0.0235 (6) | **0.7587** (1) | 0.7581 (2) | 0.7175 (3) | 0.7002 (4) | 0.6916 (5) |
| 20ngB | 0.3321 (6) | 0.4609 (4) | 0.4428 (5) | 0.6724 (2) | 0.6501 (3) | **0.7213** (1) |
| 20ngC | 0.2438 (6) | 0.2754 (5) | 0.3588 (4) | 0.4659 (3) | 0.4699 (2) | **0.7076** (1) |
| RCV1-2 | 0.2204 (5) | 0.2221 (4) | 0.2249 (2) | 0.2234 (3) | 0.1290 (6) | **0.2341** (1) |
| RCV1-3 | 0.2172 (6) | **0.5487** (1) | 0.5327 (3) | 0.3876 (5) | 0.4982 (4) | 0.5467 (2) |
| RCV1-4 | 0.4546 (6) | 0.4635 (3) | 0.4701 (5) | 0.5117 (4) | 0.5307 (2) | **0.5438** (1) |
| $AVG^{(COS)}$ | 0.2486 (6) | 0.4549 (5) | 0.4646 (4) | 0.4964 (2) | 0.4964 (2) | **0.5742** (1) |

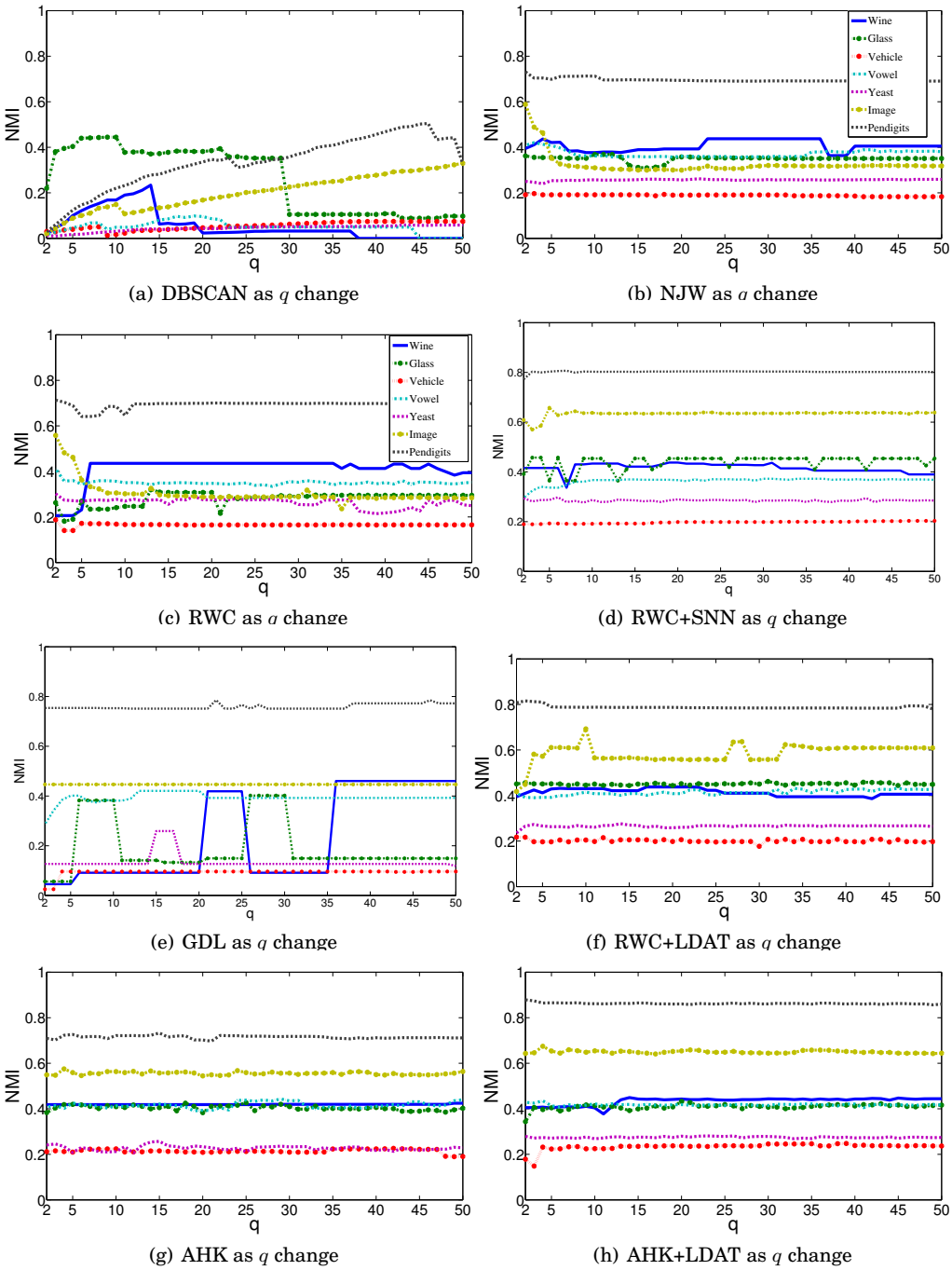
In short, both AHK and LDAT can improve RWC, but in different aspects: AHK alleviates the clustering sensitivity to the scaling parameter and data perturbation, while LDAT provides more insight to the density change across different clusters. Table IV shows that RWC+LDAT and AHK all increase RWC's performance. When both AHK and LDAT are applied, AHK+LDAT obtains the best performance in general.

### 6.4. Robustness on Adaptive Scaling Parameter ($q$)

To systematically demonstrate the superior robustness of AHK and AHK+LDAT across different Gaussian scaling parameter $q$, we test several algorithms on seven datasets: wine, glass, vehicle, vowel, yeast, image, and pendigits datasets. The test range of $q$ is $[2, 50]$ with $one$ as the step size. Figure 13 shows the performance of eight algorithms.

DBSCAN (Figure 13(a)) and GDL (Figure 13(e)) are extremely unstable and there are little clue about how to tune $q$ in an unsupervised way. With manifold awareness, NJW (Figure 13(b)) shows better and more stable clustering performance. Compared with RWC (Figure 13(c)), RWC+LDAT (Figure 13(f)) achieves higher quality of performances, especially with glass datasets ($41.15\% \uparrow$), yeast ($13.04\% \uparrow$), image ($54.25\% \uparrow$) and pendigits ($7.25\% \uparrow$). On the other hand, compared with RWC+SNN, RWC+LDAT has better performance on wine ($3.03\% \uparrow$), glass ($5.54\% \uparrow$) and vowel ($7.81\% \uparrow$). This confirms that LDAT helps to improve clustering results by providing density awareness in the cluster overlapping area. AHK (Figure 13(g)) enhances the clustering stability across $q$ especially on wine, glass and image datasets, but it doesn't necessarily retain the best performance.

Overall, the combination of AHK+LDAT has the best result on average and performs consistently across different values of $q$. The stability of AHK+LDAT inherently originates from the AHK with LBN, and its outperformance partly comes from LDAT.




(a) DBSCAN as $q$ change

(b) NJW as $q$ change

(c) RWC as $q$ change

(d) RWC+SNN as $q$ change

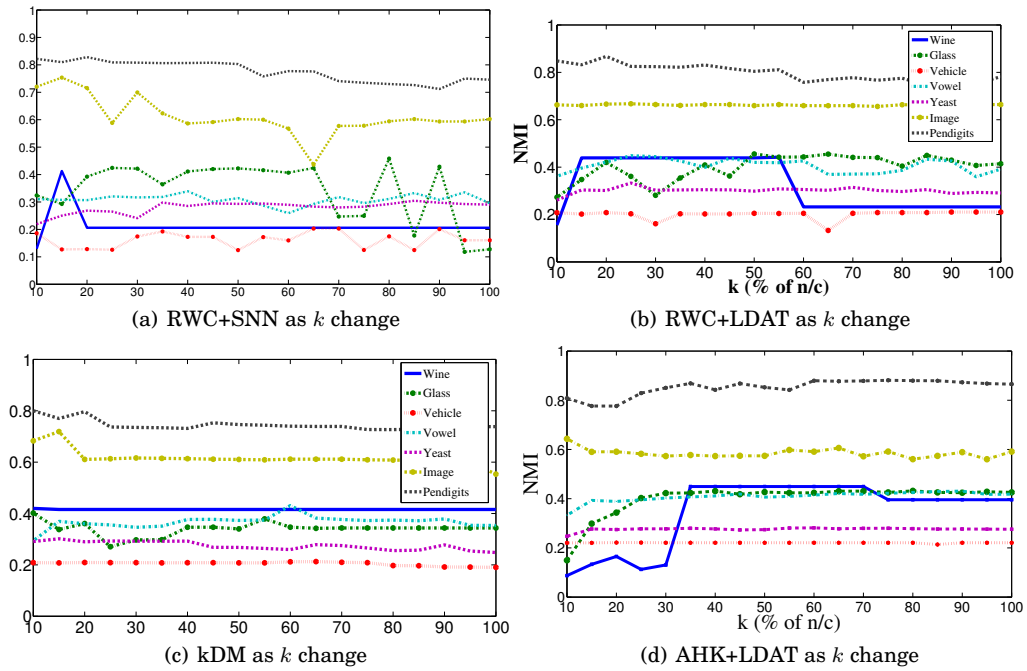
(e) GDL as $q$ change

(f) RWC+LDAT as $q$ change

(g) AHK as $q$ change

(h) AHK+LDAT as $q$ change

Fig. 13. Stability with different adaptive scaling parameter $q$.




(a) RWC+SNN as $k$ change (b) RWC+LDAT as $k$ change

(c) kDM as $k$ change (d) AHK+LDAT as $k$ change

Fig. 14. Stability under different neighborhood size $k$.

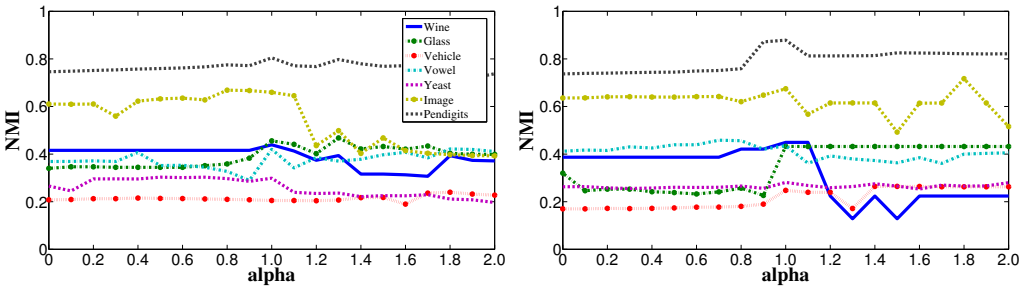
### 6.5. Robustness on Neighborhood Size ($k$)

To reveal the stability with respect to the neighborhood scaling parameter $k$ across datasets, we test $k \in [10\%, 100\%]$ of $\frac{n}{c}$ on kDM, RWC+SNN, RWC+LDAT and AHK+LDAT, the algorithms that built upon $k$-nn neighborhood. We also report results on wine, glass, vehicle, vowel, yeast, images, and pendigits datasets.

Although it doesn't provide the best performance, kDM (Figure 14(c)) indeed has a stable performance across different $k$ since it builds on diffusion map. The peak performance of RWC+LDAT (Figure 14(b)) on each dataset mostly surpass the peak performance of kDM because of the power of LDAT. However it fails to sustain stable result across different $k$. But still RWC+LDAT is more stable than RWC+SNN in that LDAT updates the similarity on a probabilistic interpretation rather than the direct (original) similarity. From the same point of view, AHK+LDAT obviously has better stability than RWC+LDAT since AHK contributes a lot on the manifold-awareness and therefore the whole algorithm has stronger support from statistics. Figure 14(d) shows that our AHK+LDAT demonstrates consistently better performance when we choose the neighborhood size $k = 50\% \times \frac{n}{c}$.

### 6.6. Robustness on Reduction Degree ($\alpha$) in LDAT

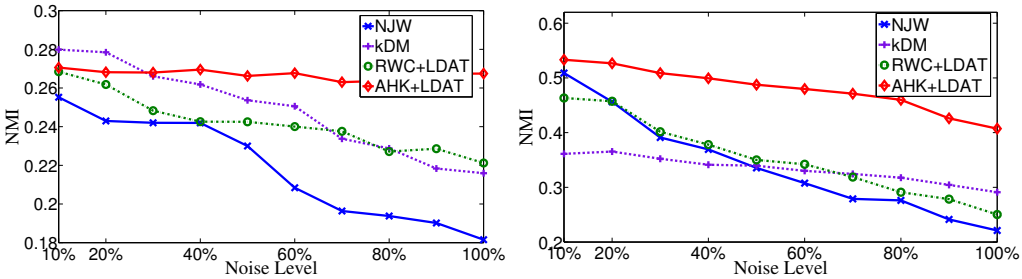
Figure 15 shows the stability of RWC+LDAT and AHK+LDAT when $\alpha$ is being tuned. The integration along all the time scales and LBN provide AHK an advanced random walk process, which leads to better average performance of AHK+LDAT. The only two exceptions are glass and wine, where RWC+LDAT outperforms AHK+LDAT. But if the datasets are getting larger, more sampling points will support a better manifold reconstruction (like image and pendigits).




(a) RWC+LDAT as $\alpha$ change (b) AHK+LDAT as $\alpha$ change

Fig. 15. Stability under different reduction factor $\alpha$.

Generally speaking, $\alpha \in [0.9, 1.1]$ makes LDAT sustainably performs better than RWC (LDAT when $\alpha = 0$) for both RWC+LDAT and AHK+LDAT. Therefore by default we recommend $\alpha$ to be one.




(a) Yeast with different noise percentages (b) RCV1-4 with different noise percentages

Fig. 16. Algorithm performance on different noise levels.

### 6.7. Robustness to Different Noise Levels

We conduct experiments on controlled noisy datasets to examine the performance of our algorithms and make comparison with the other algorithms. The yeast and RCV1 datasets are used for this experiment. We added uniformly-distributed noise to create more datasets, each of which has a different percentage of noise, i.e. $0\%$, $10\%$, $20\%$, $\cdots$, $100\%$. To study the robustness against noise comprehensively and avoid tuning scaling parameter $q$ for the best case scenario, we iteratively evaluate with all possible values for the scaling parameter $q$ and record the average performance result. The experimental results are displayed in Figure 16.

Overall, AHK+LDAT shows both robust and better performance across different noise conditions. Although RWC+LDAT shows better performance than NJW for most cases, its effectiveness decreases dramatically and demonstrates a similar trend to that of NJW. Similar to AHK+LDAT, kDM shows stable performance across different noise percentages, occasionally even outperforms AHK+LDAT in the case of yeast dataset with a small percentage of noise.

### 6.8. Scalability Analysis

This section analyzes the algorithm scalability. The experiment was done on a $2.3$ GHz Intel Core $i7$ processor with $8$ GB $1600$ MHz DDR3 memory.

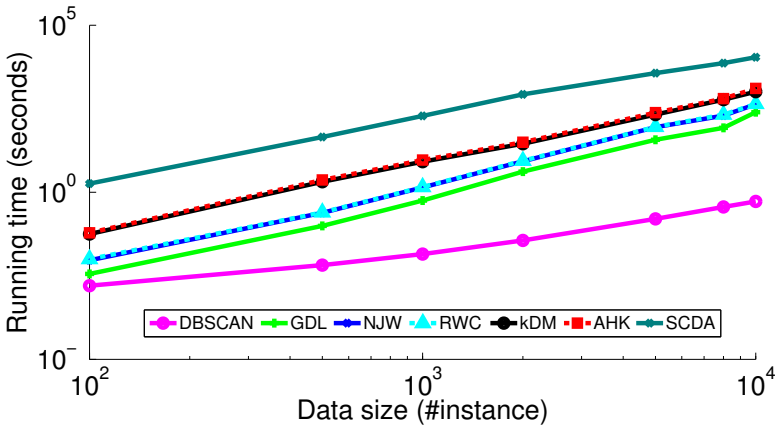



Fig. 17. Scalability Analysis.

Figure 17 shows that SCDA is the most time-consuming algorithm, which is not surprising since it requires more time to compute the joint region of the $\epsilon$-neighborhoods. On the other hand, DBSCAN is the most efficient algorithm in time with R* tree and a non-matrix based implementation. NJW and RWC are in need of eigen-decomposition, therefore their running time is more than GDL. Our proposed AHK, similar to kDM, also requires second construction of similarity matrix and another eigen-decomposition, so their scalabilities are worse than NJW and RWC, but still better than SCDA.

## 7. CONCLUSION

This paper presented a novel clustering algorithm that seamlessly integrated two robust and effective techniques, i.e. Aggregated Heat Kernel(AHK) and Local Density Affinity Transformation (LDAT). Consequently, our proposed approach achieved remarkable performance improvements for those datasets with heterogeneous density distributions. Three primary advantages of our work are: (1) Its manifold reconstruction is robust to the scaling parameter tuning and noise appearance; (2) It alleviates local density bias in Normalized Cut; and (3) It functions well with any affinity measurement, and is universally applicable. Our comprehensive experiments validate that the proposed algorithm outperforms the majority of the existing clustering algorithms. Future work will focus on analyzing the interaction effect between local density awareness and manifold reconstruction.

## 8. ACKNOWLEDGEMENTS

This research is supported in part by NSF grants IIS-0949467, IIS-1047715, and IIS-1049448. It is also supported by United States Department of Energy, Grant No. DE-SC0003361, funded through the American Recovery and Reinvestment Act of 2009. In addition, this project is also supported in part by DOE Systems Biology Knowledgebase (DE-AC02-98CH10886).

## REFERENCES

N. O. Andrews and E. A. Fox. 2007. Recent Developments in Document Clustering. *Technical Report* (2007).

M. Ankerst, M. M. Breunig, H. P. Kriegel, and J. Sander. 1999. OPTICS: Ordering Points To Identify the Clustering Structure. *ACM SIGMOD* (1999), 49–60.

W. Arendt. 2011. Vector-valued Laplace transforms and Cauchy problems. *Springer* 96 (2011).

R. Badeau, B. David, and G. Richard. 2005. Fast Approximated Power Iteration Subspace Tracking. *IEEE Signal Processing* (2005), 2931–2941.

M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger. 1998. Robust anisotropic diffusion. *IEEE Image Processing* (1998), 421–432.

M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. 2000. LOF: Identifying Density-based Local Outliers. *ACM SIGMOD* (2000), 93–104.

T. Buhler and M. Hein. 2009. Spectral Clustering based on the Graph p-Laplacian. *ICML* (2009), 81–88.

G. Camps-Valls, L. Bruzzone, J. L. Rojo-Alvarez, and F. Melgani. 2006. Robust support vector regression for biophysical variable estimation from remotely sensed images. *Geoscience and Remote Sensing Letters, IEEE* (2006), 339–343.

H. Chang and D. Y. Yeung. 2008. Robust Path-based Spectral Clustering. *Pattern Recognition* 41 (2008), 191–203.

V. Cherkassky and Y. Ma. 2005. Multiple model regression estimation. *Neural Networks, IEEE* (2005), 785–798.

Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. 2009. On Evolutionary Spectral Clustering. *ACM TKDD* 3, 4 (2009), 1–30.

R. R. Coifman and S. Lafon. 2006. Diffusion maps. *Applied and Computational Harmonic Analysis* 21, 1 (2006), 5–30.

D. Comaniciu and P. Meer. 2002. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence* (2002), 603–619.

C. D. Correa and P. Lindstrom. 2012. Locally-scaled Spectral Clustering using Empty Region Graphs. *KDD* (2012), 1330–1338.

R. N. Dave and R. Krishnapuram. 1997. Robust clustering methods: a unified view. *Fuzzy Systems, IEEE* (1997), 270–293.

K. B. Driver. 2003. Compact and Fredholm Operators and the Spectral Theorem. *Analysis tools with applications, Lecture Notes* 35 (2003), 579–600.

R. O. Duda, P. E. Hart, and D. G. Stork. 2001. Pattern Classification. (2001).

L. Ertoz, M. Steinbach, and V. Kumar. 2002. A new shared nearest neighbor clustering algorithm and its applications. *In Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining* (2002), 105–115.

M. Ester, H. Kriegel, J. Sander, and X. Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD* (1996), 226–231.

A. Grigoryan. 1999. Estimates of Heat Kernels on Riemannian Manifolds. (1999), 140–225.

S. Guha, R. Rastogi, and K. Shim. 2000. ROCK: A robust clustering algorithm for categorical attributes. *Information systems* (2000), 345–366.

I. Gutman and W. Xiao. 2004. The generalized inverse of the laplacian matrix and some applications. *Bulletin, Sciences mathematiques* 29 (2004), 1–9.

J. A. Hartigan and M. A. Wong. 1978. Algorithm as 136: a K-means Clustering Algorithm. *Appl. Stat.* 28 (1978), 100–108.

A. Hinneburg and D. A. Keim. 1999. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. *VLDB* (1999).

E. Hsu. 2002. Stochastic Analysis on Manifolds. *Graduate Studies in Mathematics* (2002).

H. Huang, H. Qin, S. Yoo, and D. Yu. 2012. Local anomaly descriptor: a robust unsupervised algorithm for anomaly detection based on diffusion space. *ACM CIKM* (2012), 405–414.

H. Huang, S. Yoo, H. Qin, and D. Yu. 2011. A Robust Clustering Algorithm based on Aggregated Heat Kernel Mapping. *IEEE ICDM* (2011), 270–279.

P. J. Huber and E. M. Ronchetti. 2009. Robust statistics. *NJ: Wiley. doi 10* (2009).

A. K. Jain and R. C. Dubes. 1988. Algorithms for Clustering Data. (1988).

R. A. Jarvis and E. A. Patrick. 1973. Clustering using a similarity measure based on shared nearest neighbors. *In IEEE Transactions on Computers* (1973), 1025–1034.

T. Joachims. 1996. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. *Computer Science Technical Report CMU-CS-96-118* (1996).

G. Karypis, E. H. Han, and V. Kumar. 1999. Chameleon: Hierarchical Clustering using Dynamic Modeling. *In IEEE Transactions on Computers* (1999), 68–74.

S. Lafon, Y. Keller, and R. R. Coifman. 2006. Data fusion and multicue data matching by diffusion maps. *Trans. on Pattern Analysis and Machine Intelligence* 28 (2006), 1784–1797.

D. D. Lewis, Y. Yang, T. G. Rose, F. Li, and G. Dietterich. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* 5 (2004), 361–397.

Z. Li, J. Liu, S. Chen, , and X. Tang. 2007. Noise robust spectral clustering. *IEEE ICCV* (2007), 1–8.

U. V. Luxburg. 2007. A Tutorial on Spectral Clustering. *Statistics and Computing* 17, 4 (2007), 395–416.

U. V. Luxburg, A. Radl, and M. Hein. 2010. Getting Lost in Space: Large Sample Analysis of the Resistance Distance. *NIPS* (2010), 2622–2630.

M. Meila and J. Shi. 2001. A Random Walks View of Spectral Segmentation. *8th International Workshop on Artificial Intelligence and Statistics* (2001).

B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis. 2005. Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators. *NIPS* (2005).

A. Ng, M. Jordan, and Y. Weiss. 2002. On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing Systems* 14 (2002), 846–856.

V. Y. Pan and Z. Q. Chen. 1999. The Complexity of the Matrix Eigenproblem. *STOC'99 Proceedings of the thirty-first annual ACM symposium on Theory of computing* (1999), 507–516.

P. Perona and W. T. Freeman. 1998. A factorization approach to grouping. *ECCV* (1998), 655–670.

H. Qiu and E. R. Hancock. 2007. Clustering and Embedding using Commute Times. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 11 (2007), 1873–1890.

J. W. Richards, P. E. Freeman, A. B. Lee, and C. M. Schafer. 2009. Accurate parameter estimation for star formation history in galaxies using sdss spectra. *MNRAS* (2009), 1044–1057.

J. Shi and J. Malik. 2000. Normalized Cuts and Image Segmentation. *IEEE TPAMI* 22, 8 (2000), 888–905.

M. Steinbach, P. Tan, V. Kumar, S. Klooster, and C. Potter. 2003. Discovery of Climate Indices using Clustering. *KDD* (2003), 446–455.

A. Strehl and J. Ghosh. 2003. Cluster Ensembles - a Knowledge Reuse Framework for Combining Multiple Partitions. *J. Mach. Learn. Res.* (2003), 583–617.

J. Sun, M. Ovsjanikov, and L. Guibas. 2009. A Concise and Provably Informative Multi-scale Signature based on Heat Diffusion. *SGP* (2009), 1383–1392.

T. N. Tran, K. Drab, and M. Daszykowski. 2013. Revised DBSCAN algorithm to cluster data with dense adjacent clusters. *Chemom. Intell. Lab. Syst. 120* (2013), 92–96.

H. Valizadegan and R. Jin. 2007. Generalized maximum margin clustering and unsupervised kernel learning. *NIPS* (2007), 1417–1424.

D. Verma and M. Meila. 2001. Comparison of spectral clustering methods. *UW CSE Technical report* (2001).

F. L. Wauthier, N. Jolic, and M. I. Jordan. 2012. Active Spectral Clustering via Iterative Uncertainty Reduction. *KDD* (2012).

I. Weiss. 1993. Noise-resistant invariants of curves. *Pattern Analysis and Machine Intelligence, IEEE* (1993), 943–948.

P. Yang, Q. Zhu, and B. Huang. 2011. Spectral Clustering with Density Sensitive Similarity Function. *Knowledge-Based Systems* (2011), 621–628.

L. Zelnik-Manor and P. Perona. 2004. Self-tuning Spectral Clustering. *Advances in Neural Information Processing Systems* 17 (2004), 1601–1608.

H. Zha, X. He, C. Ding, H. D. Simon, and M. Gu. 2001. Spectral relaxation for k-means clustering. *In Advances in neural information processing systems* (2001), 1057–1064.

W. Zhang, X. Wang, D. Zhao, and X. Tang. 2012. Graph Degree Linkage: Agglomerative Clustering on a Directed Graph. *ECCV Lecture Notes in Computer Science* (2012), 428–441.

X. Zhang, J. Li, and H. Yu. 2011. Local Density Adaptive Similarity Measurement for Spectral Clustering. *Pattern Recognition Letters* (2011), 352–358.