Fast Approximate Spectral Clustering

Donghui Yan
Department of Statistics
University of California
Berkeley, CA 94720

Ling Huang
Intel Research Lab
Berkeley, CA 94704

Michael I. Jordan
Departments of EECS and Statistics
University of California
Berkeley, CA 94720

Technical Report
March 2009

Abstract

Spectral clustering refers to a flexible class of clustering procedures that can produce high-quality clusterings on small data sets but which has limited applicability to large-scale problems due to its computational complexity of $O(n^3)$, with n the number of data points. We extend the range of spectral clustering by developing a general framework for fast approximate spectral clustering in which a distortion-minimizing local transformation is first applied to the data. This framework is based on a theoretical analysis that provides a statistical characterization of the effect of local distortion on the mis-clustering rate. We develop two concrete instances of our general framework, one based on local k-means clustering (KASP) and one based on random projection trees (RASP). Extensive experiments show that these algorithms can achieve significant speedups with little degradation in clustering accuracy. Specifically, our algorithms outperform k-means by a large margin in terms of accuracy, and run several times faster than approximate spectral clustering based on the Nyström method, with comparable accuracy and significantly smaller memory footprint. Remarkably, our algorithms make it possible for a single machine to spectral cluster data sets with a million observations within several minutes.

1 Introduction

Clustering is a problem of primary importance in data mining, statistical machine learning and scientific discovery. An enormous variety of methods have been developed over the past several decades to solve clustering problems [15, 20]. A relatively recent area of focus has been *spectral clustering*, a class of methods based on eigendecompositions of affinity, dissimilarity or kernel matrices [21, 29, 33]. Whereas many clustering methods are strongly tied to Euclidean geometry, making explicit or implicit assumptions that clusters form convex regions in Euclidean space, spectral methods are more flexible, capturing a wider range of geometries. They often yield superior empirical performance when compared to competing algorithms such as k-means, and they have been successfully deployed in numerous applications in areas such as computer vision, bioinformatics, and robotics. Moreover, there is a substantial theoretical literature supporting spectral clustering [21, 37].

Despite these virtues, spectral clustering is not widely viewed as a competitor to classical algorithms such as hierarchical clustering and k-means for large-scale data mining problems. The reason is easy to state – given a data set consisting of n data points, spectral clustering algorithms form an $n \times n$ affinity matrix and compute eigenvectors of this matrix, an operation that has a computational complexity of $O(n^3)$. For applications with n on the order of thousands, spectral clustering methods begin to become infeasible, and problems with n in the millions are entirely out of reach.

In this paper we focus on developing fast approximate algorithms for spectral clustering. Our approach is not fundamentally new. As in many other situations in data mining in which a computational bottleneck is involved, we aim to find an effective preprocessor that reduces the size of the data structure that is input to that bottleneck (see, e.g., [26, 28]). There are many options that can be considered for this preprocessing step. One option is to perform various forms of subsampling of the data, selecting data points at random or according to some form of stratification procedure. Another option is to replace the original data set with a small number of points (i.e., "representatives") that aim to capture relevant structure. Another approach that is specifically available in the spectral clustering setting is to exploit the literature on low-rank matrix approximations. This last approach has been particularly prominent in the literature; in particular, several researchers have proposed using the Nyström method for rank reduction [9, 38, 11]. While it is useful to define such preprocessors, simply possessing a knob that can adjust computational complexity does not constitute a solution to the problem of fast spectral clustering. What is needed is an explicit connection between the amount of data reduction that is achieved by a preprocessor and the subsequent effect on the clustering. Indeed, the motivation for using spectral methods is that they can provide a high-quality clustering, and if that high-quality clustering is destroyed by a preprocessor then we should consider other preprocessors (or abandon spectral clustering entirely). In particular, it is not satisfactory to simply reduce the rank of an affinity matrix so that an eigendecomposition can be performed in a desired time frame, unless we have an understanding of the effect of this rank reduction on the clustering.

In this paper we propose a general framework for fast spectral clustering and conduct an end-to-end theoretical analysis for our method. In the spirit of rate-distortion theory, our analysis yields a relationship between an appropriately defined notion of distortion at the input and some notion of clustering accuracy at the output. This analysis allows us to argue that the goal of a preprocessor should be to minimize distortion; by minimizing distortion we minimize the effect of data reduction on spectral clustering.

To obtain a practical spectral clustering methodology, we thus make use of preprocessors that minimize distortion. In the current paper we provide two examples of such preprocessors. The first is classical k-means, used in this context as a local data reduction step. The second is the Random Projection tree (RP tree) of [8]. In either case, the overall approximate spectral clustering algorithm takes the following form: (1) coarsen the affinity graph by using the preprocessor to collapse neighboring data points into a set of local "representative points," (2) run a spectral clustering algorithm on the set of representative points, and (3) assign cluster memberships to the original data points based on those of the representative points.

Our theoretical analysis is a perturbation analysis, similar in spirit to those of [21] and [29] but different in detail given our focus on practical error bounds. It is also worth noting that this analysis has applications beyond the design of fast approximations to spectral clustering. In particular, as discussed by [19], our perturbation analysis can be used for developing distributed versions of spectral clustering and for analyzing robustness to noise.

The remainder of the paper is organized as follows. We begin with a brief overview of spectral clustering in Section 2, and summarize the related work in Section 3. In Section 4 we describe our framework for fast approximate spectral clustering and discuss two implementations of this framework – "KASP," which is based on k-means, and "RASP," which is based on RP trees. We evaluate our algorithms in Section 5, by comparing both KASP and RASP with Nyström approximation and k-means. We present our theoretical analysis in Section 6. In particular, in that section, we provide a bound for the mis-clustering rate that depends linearly on the amount of perturbation to the original data. We then turn to an analysis of the performance of our approximate algorithms in Section 7. Finally, we conclude in Section 8.

2 Spectral clustering

Given a set of n data points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, with each $\mathbf{x}_i \in \mathbb{R}^d$, we define an *affinity graph* $\mathcal{G} = (V, E)$ as an undirected graph in which the i^{th} vertex corresponds to the data point $\mathbf{x}_i$. For each edge $(i, j) \in E$, we associate a weight a_{ij} that encodes the affinity (or similarity) of the data points $\mathbf{x}_i$ and $\mathbf{x}_j$. We refer to the matrix $A = (a_{ij})_{i,j=1}^n$ of affinities as the *affinity matrix*.

The goal of spectral clustering is to partition the data into m disjoint classes such that each $\mathbf{x}_i$ belongs to one and only one class. Different spectral clustering algorithms formalize this partitioning problem in different ways [33, 27, 29, 39]. In the current paper we adopt the *normalized cuts* (Ncut) formulation [33].[1] Define $W(V_1, V_2) = \sum_{i \in V_1, j \in V_2} a_{ij}$ for two (possibly overlapping) subsets V_1 and V_2 of V. Let $V =$

[1]We use Ncut only for concreteness; our methodology applies immediately to other spectral clustering formulations.

```
Algorithm 1 SpectralClustering $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$
Input:    $n$ data points $\{\mathbf{x}_i\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^d$
Output: Bipartition $S$ and $\bar{S}$ of the input data

1. Compute the affinity matrix $A$ with elements:
        $a_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \quad i,j = 1, \ldots, n$
2. Compute the diagonal degree matrix D with elements:
        $d_i = \sum_{j=1}^n a_{ij}$
3. Compute the normalized Laplacian matrix:
        $L = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$
4. Find the second eigenvector $\mathbf{v}_2$ of $L$
5. Obtain the two partitions using $\mathbf{v}_2$:
        $S = \{i : (\mathbf{v}_2)_i > 0\}, \ \ \bar{S} = \{i : (\mathbf{v}_2)_i \leq 0\}$
```

$(V_1, \ldots, V_m)$ denote a partition of V, and consider the following optimization criterion:

$$\text{Ncut } = \sum_{j=1}^{m} \frac{W(V_j, V) - W(V_j, V_j)}{W(V_j, V)}. \tag{1}$$

In this equation, the numerator in the j^{th} term is equal to the sum of the affinities on edges leaving the subset V_j and the denominator is equal to the total degree of the subset V_j. Minimizing the sum of such terms thus aims at finding a partition in which edges with large affinities tend to stay within the individual subsets V_j and in which the sizes of the V_j are balanced.

The optimization problem in (1) is intractable and spectral clustering is based on a standard relaxation procedure that transforms the problem into a tractable eigenvector problem. In particular, the relaxation for Ncut is based on rewriting (1) as a normalized quadratic form involving indicator vectors. These indicator vectors are then replaced with real-valued vectors, resulting in a generalized eigenvector problem that can be summarized conveniently in terms of the (normalized) graph Laplacian L of A defined as follows:

$$L = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = I - L^0, \tag{2}$$

where $D = diag(d_1, ..., d_n)$ with $d_i = \sum_{j=1}^n a_{ij}, i = 1, \ldots, n$, and where the final equality defines L^0.

Ncut is based on the eigenvectors of this normalized graph Laplacian. The classical Ncut algorithm focuses on the simplest case of a binary partition [33], and defines multiway partitions via a recursive invocation of the procedure for binary partitions. In the case of a binary partition, it suffices to compute the second eigenvector of the Laplacian (i.e., the eigenvector with the second smallest eigenvalue). The components of this vector are thresholded to define the class memberships of the data points. Although spectral clustering algorithms that work directly with multiway partitions exist [4, 39], in the current paper we will focus on the classical recursive Ncut algorithm. We assume that the number of clusters is given a priori and we run the recursion until the desired number of clusters is reached. See Algorithm 1 for a specific example of a spectral bipartitioning algorithm where a Gaussian kernel is used to define the pairwise affinities.

3 Related Work

An influential line of work in graph partitioning approaches the partitioning problem by reducing the size of the graph by collapsing vertices and edges, partitioning the smaller graph, and then uncoarsening to construct a partition for the original graph [17, 23]. Our work is similar in spirit to this multiscale approach; we provide rigorous theoretical analysis for a particular kind of coarsening and uncoarsening methodology. More generally, our work is related to a tradition in the data mining community of using data preprocessing techniques to overcome computational bottlenecks in mining large-scale data. Examples include [28], who proposed a nonparametric data reduction scheme based on multiscale density estimation, and [5], who proposed a fast algorithm to extract small "core-sets" from the input data, based on which $(1+\epsilon)$-approximation algorithms for the k-center clustering have been developed.

Our work is also related to the literature on kernel-based learning, which has focused principally on rank reduction methods as a way to attempt to scale to large data sets. Rank reduction refers to a large class of methods in numerical linear algebra in which a matrix is replaced with a low-rank approximation. These methods have been widely adopted, particularly in the context of approximations for the support vector machine (SVM) [9, 38, 10, 34]. The affinity matrix of spectral clustering is a natural target for rank reduction. In particular, [11] have used the Nyström approximation, which samples columns of the affinity matrix and approximates the full matrix by using correlations between the sampled columns and the remaining columns. A variety of sampling procedures can be considered. [38] use uniform sampling without replacement, and [11] use a similar strategy in applying the Nyström method to image segmentation. A drawback of these procedures is that they do not incorporate any information about the affinity matrix in choosing columns to sample; moreover, they do not come with performance guarantees.

[9] replace the uniform sampling step with a judiciously chosen sampling scheme in which columns of the Gram matrix are sampled with probability proportional to their norms. While this yields a rigorous bound on the approximation error of Gram matrix, this method may need to select a large number of columns to achieve a small approximation error. It is shown that with probability at least $1-\delta$

$$||G-\tilde{G}_k||_F \leq ||G-G_k||_F + \epsilon \sum_{i=1}^{n} G_{ii}^2, \tag{3}$$

where G_k is the best rank-k approximation to G. This yields a rigorous bound for approximation of the Gram matrix. However, as the number of sampled columns n from G is on the order of $O\left(\log(\frac{1}{\delta})\cdot\frac{1}{\epsilon^4}\right)$, the algorithm has a computational complexity on the order of $O((\log(\frac{1}{\delta})/\epsilon^4)^3)$. The right hand side of (3) indicates that a very small ϵ might be required in order to obtain a small approximation error, such that the number of rows to be selected will be large. For example, when the Gaussian kernel is used, the term $\sum_{i=1}^{n} G_{ii}^2$ may grow on the order of $O(n)$. Thus the number of columns sampled is expected to be $O(n)$. As an example of this scaling, Fig. 1 plots the growth of $\frac{1}{n}\sum_{i=1}^{n} G_{ii}^2$ with data sets generated from a two-component Gaussian mixture, $\frac{1}{2}N(\mu,\Sigma)+\frac{1}{2}N(-\mu,\Sigma)$, with $\mu=(1,1)$ and $\Sigma=[1,\ 0.5; 0.5,\ 1]$, where G is the Laplacian matrix of the pairwise affinity matrix for the data.

Although the Nyström method reduces the rank of the kernel matrix, its working memory requirement can be very high. For example, a data set of size 100,000 may require more than 6GB of memory while a data set of size 1,000,000 may require more than 17GB of memory. Another issue with the Nyström method is that in data sets that are unbalanced the number of observations selected by the sampling procedure from the small clusters may be small (if not zero), which can cause small clusters to be missed and may potentially lead to problems with numerical stability.

4 Fast spectral clustering

In this section we present our algorithmic framework for fast spectral clustering. Our approach reposes on the theoretical analysis of spectral clustering that we present in Section 6. In that section we establish a quantitative relationship between the mis-clustering rate at the output of a spectral clustering algorithm and the distortion in the input. This motivates our interest in algorithms that invoke a distortion-minimizing transformation on the original data before performing spectral clustering.

Our algorithm consists of a data preprocessing step and the spectral clustering step. In the current section we present two different ways of achieving the first step: one is based on k-means and the other is based

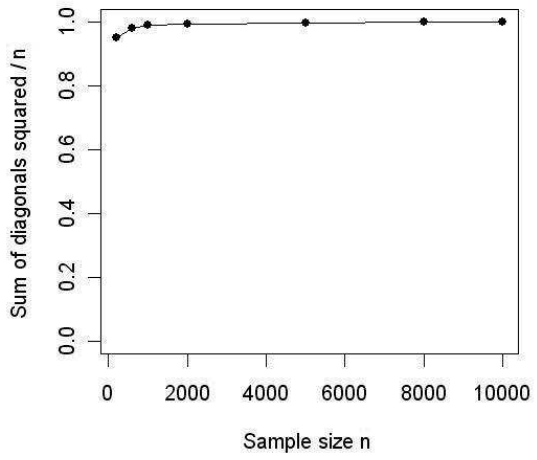

Figure 1: The growth of $\frac{1}{n}\sum_{i=1}^{n} G_{ii}^2$ for data generated from the Gaussian mixture $\frac{1}{2}N(\mu,\Sigma)+\frac{1}{2}N(-\mu,\Sigma)$ with $\mu=(1,1)$ and $\Sigma=[1,\ 0.5; 0.5,\ 1]$.

Symbol	Meaning
m	Number of clusters for partitioning input data
n, d	Size and dimension of input data
$\mathbf{x}_i, \tilde{\mathbf{x}}_i$	Input data point i and its perturbed version
$\mathbf{y}_1, \ldots, \mathbf{y}_k$	k representative points
ϵ, ϵ_i	Perturbation error
ρ	Mis-clustering rate
$G, \tilde{G}$	Distribution of input data and its perturbed version
$A, \tilde{A}$	Affinity matrix and its perturbed version
$L, \tilde{L}$	Laplacian matrix and its perturbed version
L^0, L_A^0, L_B^0	Shorthand for $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, with varying affinity matrix
d_i	Sum of row i of affinity matrix
$\lambda_2, \lambda_B, \mathbf{v}_2, \mathbf{u}_B$	The second eigenvalue and eigenvector of Laplacian matrix
g, g_0	Eigengap of Laplacian matrix

Table 1: Notation.

on random projection trees. We have chosen these two approaches because of their favorable computational properties and the simplicity of their implementation. Table 1 summarizes our notation.

4.1 Fast spectral clustering with k-means

Vector quantization is the problem of choosing a set of representative points that best represent a data set in the sense of minimizing a distortion measure [13]. When the distortion measure is squared error, the most commonly used algorithm for vector quantization is k-means, which has both theoretical support and the virtue of simplicity. The k-means algorithm employs an iterative procedure. At each iteration, the algorithm assign each data point to the nearest centroid, and recalculates the cluster centroids. The procedure stops when the total sum of squared error stabilizes.

The use that we make of k-means is as a preprocessor for spectral clustering. In particular, we propose a "k-means-based approximate spectral clustering" (KASP) algorithm that has the form in Algorithm 2.

The computational complexity of step 1, k-means, is $O(knt)$, where t is the number of iterations[2]. Given

[2]There also exist approximate k-means algorithms (e.g., the $(1+\epsilon)$ k-means in [24]) with a running time of $O(nt)$.

Algorithm 2 KASP $(\mathbf{x}_1, \ldots, \mathbf{x}_n, k)$

```
Input:   $n$ data points $\{\mathbf{x}_i\}_{i=1}^n$, number of representative points $k$
Output: $m$-way partition of the input data

1. Perform $k$-means with $k$ clusters on $\mathbf{x}_1, \ldots, \mathbf{x}_n$ to:
   a) Compute the cluster centroids $\mathbf{y}_1, \ldots, \mathbf{y}_k$ as the $k$ representative points.
   b) Build a correspondence table to associate each $\mathbf{x}_i$ with the nearest cluster centroid $\mathbf{y}_j$.
2. Run a spectral clustering algorithm on $\mathbf{y}_1, \ldots, \mathbf{y}_k$ to obtain an $m$-way cluster membership
   for each of $\mathbf{y}_i$.
3. Recover the cluster membership for each $\mathbf{x}_i$ by looking up the cluster membership
   of the corresponding centroid $\mathbf{y}_j$ in the correspondence table.
```

that the complexity of step 2 is $O(k^3)$ and the complexity of step 3 is $O(n)$, the overall computational complexity of KASP is $O(k^3) + O(knt)$. In the evaluation section we compare KASP to the alternative of simply running k-means on the entire data set.

4.2 Fast spectral clustering with RP trees

RP trees are an alternative to k-means in which a distortion-reducing transformation is obtained via random projections [8]. An RP tree gives a partition of the data space, with the center of the mass in each cell of the partition used as the representative for the data points in that cell. RP trees are based on k-d trees, which are spatial data structures that partition a data space by recursively splitting along one coordinate at a time [2]. Rather than splitting along coordinate directions, RP tree splits are made according to randomly chosen directions. All points in the current cell are projected along the random direction and the cell is then split. While classical k-d trees scale poorly with dimensionality of the data space due to the restriction to axis-parallel splits, RP trees more readily adapt to the intrinsic dimensionality of the data.

Using the RP tree as a local distortion-minimizing transformation, we obtain the "RP-tree-based approximate spectral clustering" (RASP) algorithm by replacing step 1 in Algorithm 2 with:

- Build an h-level random projection tree on $\mathbf{x}_1, \ldots, \mathbf{x}_n$; compute the centers of mass $\mathbf{y}_1, \ldots, \mathbf{y}_k$ of the data points in the leaf cells as the k representative points.

The total computational cost of this method is $O(k^3) + O(hn)$, where the $O(hn)$ term arises from the cost of building the h-level random projection tree.

5 Evaluation

Before turning to our theoretical analysis of KASP and RASP, we present a comparative empirical evaluation of these algorithms. We have conducted experiments with data sets of various sizes taken from the UCI machine learning repository [3]; an overview is given by Table 2.

The original USCI (US Census Income) data set has 299,285 instances with 41 features. We excluded instances that contain missing items, and removed features #26, #27, #28 and #30, as they have too many missing instances. We were left with 285,799 instances with 37 features, with all categorical variables converted to integers. The Poker Hand data set consists of 10 classes with a total of 1,000,000 instances. However, the original data set is extremely unbalanced – there are 6 classes which together comprise less than 1% of the total number of instances. We merged small classes together while leaving the large classes untouched. We obtained 3 final classes which correspond to about 50.12%, 42.25% and 7.63% of the total number of instances, respectively. We normalized the Connect-4 and USCI data sets so that all features have mean 0

Data set	# Features	# instances	# classes
Medium size			
ImageSeg	19	2,100	7
Musk	166	6,598	2
penDigits	16	10,992	10
mGamma	10	19,020	2
Large size			
Connect-4	42	67,557	3
USCI	37	285,779	2
Poker Hand	10	1,000,000	3

Table 2: UCI data sets used in our experiments.

and standard deviation 1. For spectral clustering, we set kernel bandwidths via a cross-validatory search in the range $[0, 200]$ (with step size 0.1) for each data set.

Spectral algorithms have not previously been studied on data sets as large as one million data points; the largest experiment that we are aware of for spectral algorithms involves the MNIST data set, which consists of 60,000 handwritten digits. In particular, [14] reported experiments using this data set, where a total running time of about 30 hours was required when using a fast iterative algorithm.

5.1 Evaluation metrics

We used two quantities to assess the clustering performance: the running time and the clustering accuracy as measured by using the true class labels associated with each of the data sets. Our experiments were performed on a Linux machine with 2.2 GHz CPU with 32 GB main memory. The running time was taken as the elapsed time (wall clock time) for clustering. Clustering accuracy was computed by counting the fraction of labels given by a clustering algorithm that agree with the true labels. This requires a search over permutations of the classes. Let $\mathbf{z} = \{1, \ldots, k\}$ denote the set of class labels, and $\theta(\cdot)$ and $f(\cdot)$ denote the true label and the label given by the clustering algorithm of a data point, respectively. Formally, the clustering accuracy β is defined as

$$\beta(f) = \max_{\tau \in \Pi_{\mathbf{z}}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{\tau\left(f(\mathbf{x}_i)\right) = \theta(\mathbf{x}_i)\} \right\}, \tag{4}$$

where $\mathbb{I}$ is the indicator function and $\Pi_{\mathbf{z}}$ is the set of all permutations on $\mathbf{z}$. When the number of classes is large, computing (4) exactly becomes infeasible. In that case we sampled from the set $\Pi_{\mathbf{z}}$ and computed the best match over the sample as an estimate of β. In particular, in our experiments, we exhaustively enumerated $\Pi_{\mathbf{z}}$ if $k \leq 7$ and otherwise sampled $10,000$ instances from $\Pi_{\mathbf{z}}$.

5.2 Competing algorithms

We compare the performance of KASP and RASP with two competing algorithms: k-means clustering and Nyström approximation based spectral clustering (referred to simply as Nyström henceforth) as implemented in [11]. Unless other specified, all algorithms were implemented in R code.

The existing work on spectral clustering has focused principally on rank reduction methods as a way to scale to large-size data. We thus compare KASP and RASP algorithms to the rank reduction approach, focusing on the Nyström approximation. The idea of Nyström is to sparsify the Laplacian matrix by random sampling and then to take advantage of the fact that eigendecomposition is usually much faster on a sparse matrix. There are several variants available for Nyström-based spectral clustering, and we choose the Matlab implementation due to Fowlkes et al. [11].

	KM-2	KM-1 (20, 200)	KM-1 (20, 1000)	KM-1 (50, 200)	BF (0.1)	BF (0.05)	BF (0.01)
ImageSeg	50.98 1	51.15 1	50.27 1	48.23 1	42.00 1	43.07 1	47.03 1
Musk	54.02 1	53.99 6	53.99 7	53.99 13	53.99 8	53.99 2	54.01 1
penDigits	51.61 1	52.85 3	52.72 5	51.90 7	51.85 3	51.88 1	51.86 1
mGamma	64.91 1	64.91 4	64.91 4	64.91 5	64.91 5	64.91 2	64.91 1

Table 3: Evaluation of k-means on medium-size data sets with different initialization methods. Numbers right below the initialization methods are parameters: (n_{rst}, n_{it}) for KM-1 and α for BF. Each result cell contains two numbers: the top one is clustering accuracy and the bottom one is the running time in second. All results are averaged over 100 runs.

The performance of k-means can vary significantly depending on the initialization method. Recently a variety of approaches have been proposed for the initialization of k-means [22, 1, 31, 25, 6]. We chose to study three initialization methods, based on their documented favorable performance [6, 30, 31, 25], as well as their relatively straightforward implementation: the Hartigan-Wong algorithm (KM-1) [16], the sampling-based two-stage algorithm (KM-2) (i.e., the Matlab implementation of k-means with the "cluster" option), and the Bradley and Fayyad algorithm (BF) [6]. In reporting a result for k-means results we report the highest level of accuracy attained across these three algorithms for each data set.

KM-1 is simply the R function kmeans() with option "Hartigan-Wong." This function has two parameters, n_{rst} and n_{it}, which denote the number of restarts and the maximal number of iterations during each run, respectively. We ran KM-1 with $(n_{rst}, n_{it}) = (20, 200), (50, 200), (20, 1000)$, respectively.

KM-2 consists of two stages of k-means. The idea is to run k-means in the first stage on a subset of the data to obtain good initial centroids so that substantially fewer iterations are required for k-means in the second stage. In the first stage, we sample 10% (5% for the Poker Hand dataset) of the data uniformly at random, and run k-means with k clusters. In the second stage, we run k-means with the k cluster centers obtained in the first stage as initial centroids. The parameters for k-means were chosen to be $(n_{rst}, n_{it}) = (20, 200)$ for the first stage and $(n_{rst}, n_{it}) = (1, 200)$ for the second stage.

BF consists of three stages of k-means. In the first stage, BF runs k-means several times (we used 10 runs) on randomly selected subsets, using, say, a fraction α of the entire data set. The output centroids from all individual runs constitutes a new data set, on which the second stage k-means runs. The centroids so obtained are used as the initial cluster centers for the third stage of k-means. In our experiment, we fixed the parameters $(n_{rst}, n_{it}) = (20, 200)$ for the first and second stages and $(n_{rst}, n_{it}) = (1, 100)$ for the third stage, while varying $\alpha \in \{0.01, 0.05, 0.1\}$.

The above are the standard settings for our first set of experiments. See below for discussion of an additional set of experiments in which the running time of k-means was matched to that of KASP.

5.3 Evaluation results

Medium-size data. We first evaluated k-means using different initialization methods and parameter configurations on the four medium-size data sets. The complete result is shown in Table 3, from which we choose the best result across all k-means experiments to compare with our method.

Table 4 shows the performance comparison for K-means, Nyström and our KASP method on the medium-size data sets. We run KASP with KM-2 for data preprocessing using different data reduction ratios γ, where γ is the ratio of the size of original data set to the reduced data set. As expected, we see that k-means runs

	K-means	Nyström	$\gamma = 1$	$\gamma = 4$	$\gamma = 8$
ImageSeg	51.15	51.10	54.76	58.95	53.66
	1	4	200	11	8
	0.03	0.13	0.47	0.06	0.04
Musk	54.02	84.45	84.97	83.18	84.31
	1	386	3884	567	162
	0.07	0.42	3.2	0.32	0.17
penDigits	52.85	54.40	51.63	53.36	53.02
	3	593	14188	381	132
	0.04	1.0	7.7	0.73	0.22
mGamma	64.91	70.97	68.60	70.61	70.36
	1	2510	71863	1116	272
	0.05	3.4	22.0	1.6	0.52

Table 4: Evaluation of k-means, Nyström and KASP on medium-size data sets. Each cell contains three numbers: the clustering accuracy, the running time in seconds, and the amount of memory used in units of GB. The parameter γ denotes the data reduction ratio for KASP; for Nyström sampling the data reduction ratio is fixed at $\gamma = 8$.

	KM-2	KM-1	KM-1	KM-1	BF	BF	BF
		(200,20)	(1000,20)	(200,50)	(0.1)	(0.05)	(0.01)
Connect-4	49.63	49.04	48.34	49.68	51.56	53.52	65.33
	6	69	184	146	78	19	3
USCI	63.47	63.48	63.47	63.47	63.47	63.47	63.47
	26	169	465	310	187	44	11
Poker Hand	35.55	35.64	35.58	35.52	35.57	35.56	35.56
	44	524	1612	1126	331	243	35

Table 5: Evaluation of k-means on large-size data sets with different initialization methods. Numbers right below the initialization methods are parameters: (n_{rst}, n_{it}) for KM-1 and α for BF. Each result cell contains two numbers: the top one is clustering accuracy and the bottom one is the running time in second. All results are averaged over 100 runs.

the fastest among the three methods; KASP runs faster and requires less working memory than Nyström when both of them use the same data reduction ratio ($\gamma = 8$). In terms of accuracy, KASP and Nyström are comparable with each other, and both are better than k-means, particularly on the data sets Musk and mGamma. From Table 4 we also see that the running time and working memory required by KASP decrease substantially as the data reduction ratio γ increases, while incurring little loss in clustering accuracy. In fact, we see that sometimes clustering accuracy increases when we use the reduced data. (This is presumably due to the regularizing effect of the pre-grouping, where neighboring observations are forced into the same final clusters.)

Large-size data. We now turn to the three large-size data sets. We first evaluated k-means with different initialization methods and parameter configurations, and report the result in Table 5.

We present the results for the KASP algorithm in Table 6, where we note that we have used relatively large data reduction ratios γ due to the infeasibility of running spectral clustering on the original data. For each data set, we observe that when we increase the data reduction ratio γ, there is little degradation in clustering accuracy while both computation time and working memory decrease substantially.

In our experiments on RASP, we used the C++ implementation of Nakul Verma to build the RP tree [8] and used this as input to our spectral clustering algorithm (implemented in R). We varied the tree depth and

Connect-4	$\gamma = 20$ 65.70 628 1.6	$\gamma = 50$ 65.69 138 0.35	$\gamma = 100$ 65.70 71 0.28	$\gamma = 200$ 65.69 51 0.20
USCI	$\gamma = 100$ 94.04 796 1.2	$\gamma = 200$ 93.97 661 0.92	$\gamma = 300$ 94.03 554 0.91	$\gamma = 500$ 94.03 282 0.90
Poker Hand	$\gamma = 500$ 50.03 2500 0.77	$\gamma = 1000$ 50.01 1410 0.56	$\gamma = 2000$ 50.01 510 0.50	$\gamma = 3000$ 49.84 310 0.44

Table 6: Evaluation of KASP on the three large-size data sets with different data reduction ratios. The values in each cell denote the data reduction ratio (γ), the clustering accuracy, the running time and the memory usage.

required that each leaf node in the tree contains at least 50 data points. The running time of RASP consists of three parts – the construction of the tree, spectral clustering on the reduced set, and the cluster membership recovery. The results for RASP are shown in Table 7. Here we again see that accuracy does not decrease over this range of data reduction values. Comparing Table 7 and Table 6, we see that RASP is roughly comparable to KASP in terms of both speed and accuracy.[3]

In Table 8 we compare our methods (using the largest values of the reduction ratio) to k-means, again using different initialization methods and parameter configurations for k-means and reporting the best result as the third column of the table. We again see the significant improvement in terms of accuracy over k-means for two of the data sets. We also compared to Nyström, where the memory requirements of Nyström forced us to restrict our experiments to only the largest values of the data reduction ratios studied for KASP and RASP. We see that KASP and Nyström have comparable clustering accuracy. As for the running time, we see from Table 8 that KASP (and RASP) are 3-5 times faster than Nyström. (Note also that KASP and RASP were implemented in R and Nyström runs in Matlab; the slowness of R relative to Matlab suggests that we are underestimating the difference.) Another difficulty with Nyström is the memory requirement, which is of order $O(n^2)$. The actual memory usages were approximately 4GB, 12GB and 17GB, respectively, for the three large data sets, while the working memory required by KASP was less than 1GB.

Given the large size of these data sets we are not able to assess the loss in clustering accuracy due to data reduction in KASP and RASP relative to the original data set (because we are unable to run spectral clustering on the original data). Instead, to provide a rough upper bound, we treat the clustering problem as a classification problem and present results from a state-of-the-art classification algorithm, the Random Forests (RF) algorithm [7]. These results suggest that the data reduction in KASP and RASP have not seriously degraded the clustering accuracy.

We also performed a further comparison of k-means and our methods in which we increased the number of restarts and iterations for k-means so that the running time matches that of KASP on the large data sets. For these experiments we used the BF implementation of k-means, and report the results in Table 9. Our results show that the longer runs of k-means did not yield significant improvements in accuracy to the results we have reported here; k-means continued to fall significantly short of KASP and Nyström on USCI and Poker Hand.

[3] Due to the random nature of RASP during the tree construction stage, we are not able to match the data reduction ratio in RASP to that of KASP. Hence only a rough comparison is possible between KASP and RASP.

Connect-4	$\gamma = 144$	$\gamma = 136$	$\gamma = 207$
	65.72	63.27	63.95
	107	78	67
	0.14	0.14	0.13
USCI	$\gamma = 170$	$\gamma = 267$	$\gamma = 516$
	92.99	93.66	92.09
	1229	922	418
	0.79	0.70	0.67
Poker hand	$\gamma = 509$	$\gamma = 977$	$\gamma = 3906$
	50.11	50.03	49.70
	1440	710	215
	0.91	0.67	0.45

Table 7: Evaluation of RASP on the three large-size data sets with different data reduction ratios. The values in each cell denote the data reduction ratio (γ), the clustering accuracy, the running time and the memory usage.

	RF	K-means	Nyström	KASP	RASP
Connect-4	75.00	65.33	65.82	65.69	63.95
		3	181	51	67
		0.19	4.0	0.20	0.13
USCI	95.27	63.47	93.88	94.03	92.09
		11	1603	282	418
		0.65	12.0	0.90	0.67
Poker Hand	60.63	35.56	50.24	49.84	49.70
		35	1047	310	215
		0.42	17.0	0.44	0.45

Table 8: Comparison of Random Forests (RF) classification, k-means, Nyström, KASP and RASP on the three large data sets. For RF classification, we set the training set sizes to be 7557, 28578 and 25010, and the test set sizes to be 60000, 257201, and 1000000, respectively.

6 Perturbation analysis for spectral clustering

In this section we present a theoretical analysis of the effect on spectral clustering of a perturbation to the original data. Section 7 shows how this analysis applies to the specific examples of KASP and RASP. It is worth noting that our analysis is a general one, applicable to a variety of applications of spectral clustering. In particular, perturbations arise when the original data are truncated, compressed, filtered, quantized or distorted in some way. These degradations may be unavoidable consequences of a noisy channel, or they may arise from design decisions reflecting resource constraints, computational efficiency or privacy considerations.

Data perturbation can be modeled in several different ways, including contaminated distribution models [36], measurement error models [12] and mixture modeling. We choose to work with an additive noise model, due to its simplicity and its proven value in a number of problem areas such as data filtering, quantization and compression.

We assume that the original data $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are independently and identically distributed (i.i.d.) according to a probability distribution G, and we treat data perturbation as adding a noise component ϵ_i to $\mathbf{x}_i$:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i + \epsilon_i, \tag{5}$$

for each $i = 1, \ldots, n$, and we denote the distribution of $\tilde{\mathbf{x}}$ by $\tilde{G}$. To make the analysis tractable, we further

	Stage 1	Stage 2	Stage 3	Running time	Accuracy
Connect-4	(1000, 2000)	(100, 1000)	(1, 1000)	40	65.68
USCI	(2000, 1000)	(100, 200)	(1, 1000)	248	63.47
Pokerhand	(400, 1000)	(100, 200)	(1, 1000)	280	35.55

Table 9: Evaluation of k-means with long running time. Results are obtained with the Bradley and Fayyad (BF) implementation of 3 stages of k-means, and are averaged over 100 runs. We fixed $\alpha = 0.01$, and used the pairs of number in the parenthesis for (n_{rst}, n_{it}) in each stage.

assume that: (1) ϵ_i is independent of $\mathbf{x}_i$, which is a good approximation for many real applications [13]; (2) the ϵ_i are i.i.d. according to a symmetric distribution with mean zero and bounded support; (3) the variance of ϵ_i is small relative to that of the original data, a natural assumption in our setting in which we control the nature of the data transformation.

We aim to investigate the impact on the clustering performance of the perturbation. Specifically, we wish to assess the difference between the clustering obtained on the original $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and that obtained on the perturbed data $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$. We quantify this difference by the *mis-clustering rate*, which is defined as

$$\rho = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{I_i \neq \tilde{I}_i\}, \tag{6}$$

where $\mathbb{I}$ is the indicator function, $I = (I_1, \ldots, I_n)$ being a vector indicating the cluster membership for $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and $\tilde{I} = (\tilde{I}_1, \ldots, \tilde{I}_n)$ for $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$.

Our approach to quantifying (i.e., upper bounding) the mis-clustering rate ρ consists of two components: (1) a bound that relates ρ to the perturbation of the eigenvector used in spectral clustering (see Section 6.1); (2) a perturbation bound on the matrix norm of the Laplacian in terms of the amount of data perturbation (see Section 6.2).

6.1 Mis-clustering rate via the 2^{nd} eigenvector

Let $\tilde{A}$ and $\tilde{L}$ denote the affinity matrix and the Laplacian matrix, respectively, on the perturbed data. We wish to bound the mis-clustering rate ρ in terms of the magnitude of the perturbation $\epsilon = \tilde{\mathbf{x}} - \mathbf{x}$. In our early work we derived such a bound for two-class clustering problems [19]. The bound is expressed in terms of the perturbation of the second eigenvector of the Laplacian matrix. We begin by summarizing this result. Letting $\mathbf{v}_2$ and $\tilde{\mathbf{v}}_2$ denote the unit-length second eigenvectors of L and $\tilde{L}$, respectively, we can bound the mis-clustering rate of a spectral bipartitioning algorithm (a spectral clustering algorithm that forms two classes) as follows.

Theorem 1 ([19])**.** *Under the assumptions discussed in [19], the mis-clustering rate ρ of a spectral bipartitioning algorithm on the perturbed data satisfies*

$$\rho \leq \|\tilde{\mathbf{v}}_2 - \mathbf{v}_2\|^2. \tag{7}$$

There are two limitations to this result that need to be overcome to be able to use the result in our design of a fast spectral clustering algorithm. First, the bound needs to be extended to the multiway clustering problem. We achieve that by considering recursive bipartitionings. Second, we need to estimate the amount of perturbation to the second eigenvector of the Laplacian matrix. In [19] this was done by assuming availability of the perturbed data, an assumption which is reasonable for applications that involve resource constraints in a distributed computing environment, but which is not appropriate here. We instead approach this problem via a model-based statistical analysis, to be discussed in Section 6.2. That analysis allows us to bound the perturbation of the Laplacian matrix expressed in terms of a Frobenius norm. To connect that analysis to Theorem 1, we make use of the following standard lemma.

Lemma 2 ([35]). *Let g denote the eigengap between the second and the third eigenvalues of L. Then the following holds:*

$$\|\tilde{\mathbf{v}}_2 - \mathbf{v}_2\| \leq \frac{1}{g}\|\tilde{L} - L\| + O\left(\|\tilde{L} - L\|^2\right).$$

With these links in the chain of the argument in place, we turn to a discussion of these two remaining problems, that of dealing with multiway clusterings and that of bounding the norm of the perturbation of the Laplacian matrix.

Our approach to obtaining theoretical bounds for multiway spectral clustering is a relatively simple one that is based on recursive bipartitioning. Although it may be possible to obtain a direct perturbation bound of the form of Theorem 1 for the multiway case, the problem is challenging, and in our current work we have opted for a simple approach.

Theorem 3. *Assume that: 1) the assumptions of Theorem 1 hold throughout the recursive invocation of the Ncut algorithm, 2) the smallest eigengap g_0 along the recursion is bounded away from zero, and 3) the Frobenius norm of the perturbation on Laplacian matrices along the recursion is bounded by $c||\tilde{L} - L||_F^2$ for some constant $c \geq 1$. Then the mis-clustering rate for an m-way spectral clustering solution can be bounded by (ignoring the higher order term on the right hand side):*

$$\rho \leq \frac{m}{g_0^2} \cdot c||\tilde{L} - L||_F^2.$$

This theorem provides an upper bound on ρ via the perturbation of the Laplacian matrix.

Proof. Let the sequence of Frobenius norms of the perturbation on the Laplacian matrices and the eigengaps along the recursion of Ncut be denoted by L_i and g_i, respectively, for $i = 1, \ldots, m-1$. By definition, $g_0 = \min\{g_i : i = 1, \ldots, m-1\}$. Let $n_1, \ldots, n_m$ denote the size of clusters returned from the Ncut algorithm, and $r_1, \ldots, r_{m-1}$ denote the number of mis-clustered instances within each cluster (at the last step of Ncut, assume that all errors go to the $(m-1)^{th}$ cluster). Then, by repeatedly applying Theorem 1 and Lemma 2, we get (ignoring the high-order terms on the right-hand side of Lemma 2):

$$r_1 \leq n \cdot \frac{L_1^2}{g_1^2}, \quad \text{and} \quad r_i \leq (n - n_{i-1}) \cdot \frac{L_i^2}{g_i^2},\ i = 2, \ldots, m-1.$$

Thus the final error rate ρ can be bounded by

$$\begin{aligned}
\rho &= \frac{1}{n}\sum_{i=1}^{m-1} r_i \leq \frac{1}{n}\left(n \cdot \frac{L_1^2}{g_1^2} + \sum_{i=2}^{m-1}(n - n_{i-1}) \cdot \frac{L_i^2}{g_i^2}\right) \\
&\leq \frac{1}{n}\left(n \cdot \frac{||\tilde{L} - L||_F^2}{g_0^2} + \sum_{i=2}^{m-1}(n - n_{i-1}) \cdot \frac{c||\tilde{L} - L||_F^2}{g_0^2}\right) \\
&\leq m \cdot \frac{c||\tilde{L} - L||_F^2}{g_0^2}.
\end{aligned}$$

□

6.2 Perturbation on Laplacian matrix

In this section, we develop a bound for the Frobenius norm of the perturbation on the Laplacian matrix. Let $\tilde{A} = A + \Lambda$ and $\tilde{D} = D + \Delta$. Based on a Taylor expansion, we have the following approximation for $\|\tilde{L} - L\|_F$.

Lemma 4. *If ϵ is small compared (element-wise) to A in the sense that $||\Delta D^{-1}||_2 = o(1)$, then*

$$\tilde{L} - L = -D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}} - \left(\frac{1}{2} + o(1)\right)\left[D^{-\frac{1}{2}}AD^{-\frac{3}{2}}\Delta - \Delta D^{-\frac{3}{2}}AD^{-\frac{1}{2}}\right].$$

Moreover, using standard properties of the matrix Frobenius norm [35], this yields the following inequality for the norms:

$$||\tilde{L} - L||_F \leq ||D^{-\frac{1}{2}} \Lambda D^{-\frac{1}{2}}||_F + (1 + o(1)) ||\Delta D^{-\frac{3}{2}} A D^{-\frac{1}{2}}||_F. \tag{8}$$

In the remainder of this section we use this inequality to work out perturbation bounds using Taylor series expansions for $||D^{-\frac{1}{2}} \Lambda D^{-\frac{1}{2}}||_F$ and $||\Delta D^{-\frac{3}{2}} A D^{-\frac{1}{2}}||_F$.

At this stage of our argument we need to introduce a statistical model for the original data. Specifically, we need to introduce a model for data that fall into two clusters. To obtain a tractable analysis, we model distribution G as a two-component mixture model:

$$G = (1 - \pi) \cdot G_1 + \pi \cdot G_2, \tag{9}$$

where $\pi \in \{0, 1\}$ with $\mathcal{P}(\pi = 1) = \eta$. The effect of data perturbation is to transform this model into a new mixture model specified by $\tilde{G} = (1 - \pi) \cdot \tilde{G}_1 + \pi \cdot \tilde{G}_2$, where $\tilde{G}_1$ and $\tilde{G}_2$ are obtained through Eq. (5).

The perturbation to the affinity between $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$\delta_{ij} = \exp\left(-\frac{\|\mathbf{x}_i + \epsilon_i - \mathbf{x}_j - \epsilon_j\|^2}{2\sigma^2}\right) - \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right).$$

We can simplify δ_{ij} by a Taylor expansion of the function $f(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{a}+\mathbf{x}\|^2}{2\sigma^2}\right)$ around $\mathbf{x} = 0$:

$$\delta_{ij} = \frac{(\mathbf{x}_i - \mathbf{x}_j)^T (\epsilon_i - \epsilon_j)}{\sigma^2} \cdot \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) + R_{ij}. \tag{10}$$

For the univariate case, the remainder term satisfies $|R_{ij}| \leq R_{max} \cdot (\epsilon_i - \epsilon_j)^2$ for some universal constant R_{max}, since $|f''|$ is uniformly bounded. A similar result holds for multivariate case. Based on this result, we are then able to prove the following Lemma (see the Appendix for details):

Lemma 5. *Assuming that: 1)* $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$ *are generated i.i.d. from* (9) *such that* $\inf_{1 \leq i \leq n} d_i/n > c_0$ *holds in probability for some constant* $c_0 > 0$*, 2) the distribution of the components of* ϵ *is symmetric about zero with bounded support, and 3)* $||\Delta D^{-1}||_2 = o(1)$*, then*

$$||D^{-\frac{1}{2}} \Lambda D^{-\frac{1}{2}}||_F^2 \quad \leq_p \quad c_1 \sigma_\epsilon^{(2)} + c_2 \sigma_\epsilon^{(4)}, \tag{11}$$

$$||\Delta D^{-\frac{3}{2}} A D^{-\frac{1}{2}}||_F^2 \quad \leq_p \quad c_3 \sigma_\epsilon^{(2)} + c_4 \sigma_\epsilon^{(4)}. \tag{12}$$

for some universal constants c_1, c_2, c_3, c_4 *as* $n \to \infty$*, where* $\sigma_\epsilon^{(2)}$ *and* $\sigma_\epsilon^{(4)}$ *denote the second and fourth moments of* $\|\epsilon\|$*, respectively, and "*$\leq_p$*" indicates that inequality holds in probability.*

Combining Eqs. (8), (11) and (12), we have the following theorem for the perturbation bound on the Laplacian matrix.

Theorem 6. *Assuming that: 1)* $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$ *are generated i.i.d. from* (9) *such that* $\inf_{1 \leq i \leq n} d_i/n > c_0$ *holds in probability for some constant* $c_0 > 0$*, 2) the distribution of components in* ϵ *is symmetric about* 0 *with bounded support, and 3)* $||\Delta D^{-1}||_2 = o(1)$*, then*

$$||\tilde{L} - L||_F^2 \leq_p c_1 \sigma_\epsilon^{(2)} + c_2 \sigma_\epsilon^{(4)}$$

for some universal constants c_1 *and* c_2 *as* $n \to \infty$*, where* $\sigma_\epsilon^{(2)}$ *and* $\sigma_\epsilon^{(4)}$ *denote the second and fourth moments of* $\|\epsilon\|$*, respectively, and "*$\leq_p$*" indicates that inequality holds in probability.*

The result of Theorem 6 holds when there are more than two clusters. By combining Theorems 3 and 6, we have obtained a perturbation bound for the mis-clustering rate under suitable conditions.

Remarks. i) The fourth moment is often negligible compared to the second moment. In such cases the main source of perturbation in the matrix norm comes from the second moment of ϵ. ii) We assume $\epsilon_1, ..., \epsilon_n$ i.i.d.

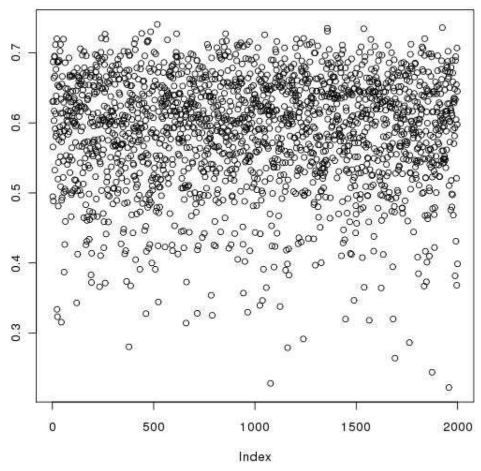

Figure 2: Scatter plot of d_i/n for 2000 observations generated i.i.d. from the Gaussian mixture $\frac{1}{2}N(-(0.3,...,0.3)^T, \Sigma_{10\times 10}) + \frac{1}{2}N((0.3,...,0.3)^T, \Sigma_{10\times 10})$. The matrix $\Sigma_{10\times 10}$ has all diagonals 1 and other entries generated i.i.d. uniform from $[0, 0.5]$ subject to symmetry.

for simplicity; Theorem 6 remains true when the ϵ_i are resulted from KASP and RASP by a similar proof. iii) The assumption that d_i/n's are bounded away from zero is a technical assumption that substantially simplifies our proof. We believe that the theorem holds more generally. Figure 2 is the scatter plot of d_i/n's for data generated from a Gaussian mixture, which suggests that it is reasonable to assume that the d_i/n are bounded away from zero.

7 Analysis of KASP and RASP

In this section we show how the analysis described in the previous section can be applied to KASP and RASP. In this analysis the noise component models the difference between the original data and their corresponding representative points. With either k-means or RP tree preprocessing, the variance of perturbation on original data can be made small according to Theorem 9 and Theorem 11, which satisfies the requirement of the model.

In the rest of this section, we first present a set of *embedding lemmas*, which establish the connection between the cluster membership of the representative points and those of the original data. We then present a performance analysis for KASP and RASP.

7.1 The embedding lemmas

Let S denote the set of representative data points (with repetitions) that correspond to each original data point:

$$S = \{\mathbf{y}_1, \ldots, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_2, \ldots, \mathbf{y}_k, \ldots, \mathbf{y}_k\}, \tag{13}$$

with repetition counts (i.e., the number of points sharing the same representative point) denoted by $r_1, r_2, \ldots, r_k$ such that $\sum_{i=1}^{k} r_i = n$. Let $S_1 = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$ denote the set of unique representative points. We show that the second eigenvector of the Laplacian matrix corresponding to the data set S can be computed from that of S_1. Since the Laplacian matrix of set S_1 can be made much smaller than that of set S, a significant reduction in computational cost can be achieved.

Lemma 7. *Let $\mathbf{v}_2$ denote the second eigenvector of the Laplacian matrix corresponding to the data set S. Then $\mathbf{v}_2$ can be written in the following form:*

$$\mathbf{v}_2 = [v_1, \ldots, v_1, v_2, \ldots, v_2, \ldots, v_k, \ldots, v_k]^T, \tag{14}$$

where the number of repetitions of each v_i is exactly r_i.

Proof. It is sufficient to consider L_A^0 in the Laplacian matrix L_A (defined in Eq. (2)) for the data set S with the first two data points being the same, i.e., $S = \{\mathbf{y}_1, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$. It is easy to see that the affinity matrix of the data set S is given by $A = [\mathbf{a}_1, \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k]$, which yields $L_A^0 = [\mathbf{b}_1, \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k]$, where the first two rows and two columns are the same in both matrices. Letting $\mathbf{v}_2 = [v_1, v_2, \ldots, v_{k+1}]^T$, then we have

$$\begin{cases} b_{11}v_1 + b_{11}v_2 + b_{12}v_3 + \cdots + b_{1k}v_{k+1} = (1-\lambda_2)v_1 \\ b_{11}v_1 + b_{11}v_2 + b_{12}v_3 + \cdots + b_{1k}v_{k+1} = (1-\lambda_2)v_2 \end{cases},$$

which implies $v_1 = v_2$. □

Lemma 8. *Let the $k \times k$ matrix B have the form*

$$B = [r_1 \cdot \mathbf{a}_1, r_2 \cdot \mathbf{a}_2, \ldots, r_k \cdot \mathbf{a}_k],$$

where matrix $[\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k]$ is the affinity matrix computed from S_1, the set of unique representative points. Let λ_B and $\mathbf{u}_B = [u_1, u_2, \ldots, u_k]^T$ be the second eigenvalue and eigenvector of the Laplacian matrix of B, respectively. Then the following equality holds (up to scaling):

$$v_1 = u_1, v_2 = u_2, \ldots, v_k = u_k, \tag{15}$$

where the v_i are the components of $\mathbf{v}_2$ in Eq. (14)*.*

Proof. It is sufficient to consider L_B^0 in the Laplacian matrix L_B. Clearly, L_B^0 has the form $L_B^0 = [r_1\mathbf{b}_1, r_2\mathbf{b}_2, \ldots, r_k\mathbf{b}_k]$. Thus, for each $i = 1, \ldots, k$, it is true that

$$r_1 b_{i1} u_1 + r_2 b_{i2} u_2 + \cdots + r_k b_{ik} u_k = (1-\lambda_B)u_i. \tag{16}$$

By Lemma 7 the second eigenvector of L_A satisfies

$$b_{i1}v_1 + \cdots + b_{i1}v_1 + b_{i2}v_2 + \cdots + b_{i2}v_2 + \cdots + b_{ik}v_k + \cdots + b_{ik}v_k = (1-\lambda_2)x_i,$$

for each $i = 1, \ldots, k$, which after re-arrangement becomes

$$r_1 b_{i1} v_1 + r_2 b_{i2} v_2 + \cdots + r_k b_{ik} v_k = (1-\lambda_2)v_i. \tag{17}$$

Since both (16) and (17) solve the same system of linear equations and both correspond to the second eigenvalue, we must have $v_i = u_i$ (up to scaling) for $i = 1, \ldots, k$. □

Remark. Empirically we find that $r_1, \ldots, r_k$ are not very different in data sets preprocessed by KASP and RASP, so in practice we do not perform the scaling; i.e., we set all $r_1, \ldots, r_k$ in the matrix B equal to 1.

Based on Lemma 7 and Lemma 8, the second eigenvector of the $n \times n$ Laplacian matrix L_A corresponding to the large data set $S = [\mathbf{y}_1, , \ldots, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_2, \ldots, \mathbf{y}_k, \ldots, \mathbf{y}_k]$ can be exactly computed from a reduced $k \times k$ Laplacian matrix L_B, after proper scaling. In the case that $k \ll n$ (which usually occurs in practice), a substantial computational speedup can be achieved (as demonstrated in Section 5). The remaining issue is how to approximate the original data set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ (with small distortion) using the reduced representative data set $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$. With this achieved, Theorem 6 then ensures that the resulting mis-clustering rate will be small, and will tend to zero in probability. In the remainder of this section, we show that the distortion can be made small if the representative points are computed by k-means or by the RP tree quantization method.

7.2 Numerical example of the embedding lemmas

As an illustration of the idea of embedding lemmas, we provide here a simple example with three unique data points, $\mathbf{x}_1 = [-1, 0]^T$, $\mathbf{x}_2 = [2, 0]^T$, $\mathbf{x}_3 = [0, 3]^T \in \mathbb{R}^2$. Let $S = \{\mathbf{x}_1, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_3, \mathbf{x}_3\}$ and $S_1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$. Using a Gaussian kernel with bandwidth $\sigma = \sqrt{3}$, we obtain the following matrices for data set S:

$$A = \left[\begin{array}{cc:cc:ccc} 1.00 & 1.00 & 0.22 & 0.22 & 0.19 & 0.19 & 0.19 \\ 1.00 & 1.00 & 0.22 & 0.22 & 0.19 & 0.19 & 0.19 \\ \hdashline 0.22 & 0.22 & 1.00 & 1.00 & 0.11 & 0.11 & 0.11 \\ 0.22 & 0.22 & 1.00 & 1.00 & 0.11 & 0.11 & 0.11 \\ \hdashline 0.19 & 0.19 & 0.11 & 0.11 & 1.00 & 1.00 & 1.00 \\ 0.19 & 0.19 & 0.11 & 0.11 & 1.00 & 1.00 & 1.00 \\ 0.19 & 0.19 & 0.11 & 0.11 & 1.00 & 1.00 & 1.00 \end{array}\right]$$

$$L_A^0 = \left[\begin{array}{cc:cc:ccc} 0.33 & 0.33 & 0.08 & 0.08 & 0.06 & 0.06 & 0.06 \\ 0.33 & 0.33 & 0.08 & 0.08 & 0.06 & 0.06 & 0.06 \\ \hdashline 0.08 & 0.08 & 0.36 & 0.36 & 0.04 & 0.04 & 0.04 \\ 0.08 & 0.08 & 0.36 & 0.36 & 0.04 & 0.04 & 0.04 \\ \hdashline 0.06 & 0.06 & 0.04 & 0.04 & 0.28 & 0.28 & 0.28 \\ 0.06 & 0.06 & 0.04 & 0.04 & 0.28 & 0.28 & 0.28 \\ 0.06 & 0.06 & 0.04 & 0.04 & 0.28 & 0.28 & 0.28 \end{array}\right]$$

and the eigenvector of interest is given by

$$\mathbf{v}_2 = [-0.194 \ \ -0.194 \ \ -0.475 \ \ -0.475 \ \ 0.397 \ \ 0.397 \ \ 0.397]. \tag{18}$$

The affinity matrix for data set S_1 is given by

$$[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3] = \begin{bmatrix} 1.00 & 0.22 & 0.19 \\ 0.22 & 1.00 & 0.11 \\ 0.19 & 0.11 & 1.00 \end{bmatrix}.$$

The matrix B (cf. Lemma 8) and L_B^0 are given by

$$B = \begin{bmatrix} 2 \times 1.00 & 2 \times 0.22 & 3 \times 0.19 \\ 2 \times 0.22 & 2 \times 1.00 & 3 \times 0.11 \\ 2 \times 0.19 & 2 \times 0.11 & 3 \times 1.00 \end{bmatrix},$$

$$L_B^0 = \begin{bmatrix} 2 \times 0.33 & 2 \times 0.08 & 3 \times 0.06 \\ 2 \times 0.08 & 2 \times 0.36 & 3 \times 0.04 \\ 2 \times 0.06 & 2 \times 0.04 & 3 \times 0.28 \end{bmatrix},$$

and the eigenvector of interest is given by

$$\mathbf{u}_B = [-0.299 \ \ -0.732 \ \ 0.612].$$

By scaling by the factor 0.649, $\mathbf{u}_2$ becomes

$$\mathbf{u}_B = [-0.194 \ \ -0.475 \ \ 0.397]. \tag{19}$$

Comparing A with B, L_A^0 with L_B^0, and (18) with (19), we verify the claims stated in Lemma 7 and Lemma 8 as well as in their proofs.

7.3 Performance analysis for KASP

Existing work from vector quantization [40, 13] allows us to characterize precisely the amount of distortion when the representative points are computed by k-means clustering if the probability distribution of the original data is given.

Let a quantizer Q be defined as $Q : \mathbb{R}^d \mapsto \{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$ for $\mathbf{y}_i \in \mathbb{R}^d$. For $\mathbf{x}$ generated from a random source in $\mathbb{R}^d$, let the distortion of Q be defined as $\mathcal{D}(Q) = \mathbb{E}\|\mathbf{x} - Q(\mathbf{x})\|^s$, which is the mean square error for $s = 2$. Let $R(Q) = \log_2 k$ denote the rate of the quantizer. Define the distortion-rate function $\delta(R)$ as

$$\delta(R) = \inf_{Q:\ R(Q) \le R} \mathcal{D}(Q).$$

Then $\delta(R)$ can be characterized in terms of the source density of G and constants d, s by the following theorem.

Theorem 9 ([40]). *Let f be the density function for G (defined in Eq.* (9)*) in $\mathbb{R}^d$. Then, for large rates R, the distortion-rate function of fixed-rate quantization has the following form:*

$$\delta_d(R) \cong b_{s,d} \cdot ||f||_{d/(d+s)} \cdot k^{-s/d},$$

where $\cong$ means the ratio of the two quantities tends to 1, $b_{s,d}$ is a constant depending on s and d, and

$$||f||_{d/(d+s)} = \left(\int f^{d/(d+s)}(x)dx \right)^{(d+s)/d}.$$

Thus, by Theorems 3 and 6, we arrive at the following characterization of the mis-clustering rate of KASP.

Theorem 10. *Let the data be generated from a distribution with density f. Let the assumptions for Theorem 3 and Theorem 6 hold. Then the mis-clustering rate ρ can be computed as:*

$$\rho = c \cdot b_{2,d} \cdot ||f||_{d/(d+2)} \cdot k^{-2/d} + O\left(k^{-4/d}\right), \tag{20}$$

where c is a constant determined by the number of clusters, the variance of the original data, the bandwidth of the Gaussian kernel and the eigengap of Laplacian matrix (or minimal eigengap of the Laplacian of all affinity matrices used in Ncut).

7.4 Performance analysis for RASP

We now briefly discuss the case of RASP, where the distortion-minimizing transformation is given by an RP tree instead of by k-means. By combining our perturbation analysis for spectral clustering with quantization results from [8], we obtain an analysis for RASP. Define the average diameter of input data $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ as [8]

$$\Delta_A^2(X) = \frac{1}{|X|^2} \sum_{\mathbf{x},\mathbf{y} \in X} ||\mathbf{x} - \mathbf{y}||^2.$$

It is clear that if $\mathbf{u}(X)$ is the center of mass for the data set X, then $\Delta_A^2(X) = \frac{2}{|X|} \sum_{\mathbf{x} \in X} ||\mathbf{x} - \mathbf{u}(X)||^2$. A set $X \subset \mathbb{R}^d$ is said to have local covariance dimension (d', ϵ, r) if its restriction to any ball of radius r has a covariance matrix whose largest d' eigenvalues satisfy

$$\sigma_1^2 + \cdots + \sigma_{d'}^2 \ge (1 - \epsilon) \cdot (\sigma_1^2 + \cdots + \sigma_d^2).$$

The quantization error of the RP tree is characterized in terms of the local covariance dimension as follows.

Theorem 11 ([8]). *Suppose an RP tree is built using data set $X \subset \mathbb{R}^d$, then there exist constants $0 < c_1, c_2 < 1$ with the following property. Consider any cell C of radius r such that $X \cap C$ has local covariance dimension (d', ϵ, r) with $\epsilon < c_1$. Pick a point $\mathbf{x} \in X \cap C$ at random, and let C' be the cell that contains $\mathbf{x}$ at the next level down. Then*

$$\mathbb{E}[\Delta_A^2(X \cap C')] \leq \left(1 - \frac{c_2}{d'}\right) \Delta_A^2(X \cap C),$$

where the expectation is taken over the randomization in splitting C and the choice of $\mathbf{x} \in X \cap C$.

Theorem 11 shows that the vector quantization error of RP tree behaves as $e^{-O(h/d')}$ with h the depth of the tree and d' the intrinsic dimension of the data. Thus the quantization error can be made small as the tree depth grows, and a result similar to Theorem 10 holds for RASP.

8 Conclusion

We have proposed a general framework and presented two fast algorithms for approximate spectral clustering. Our algorithms leverage k-means and RP tree methods to pre-group neighboring points and produce a set of reduced representative points for spectral clustering. These algorithms significantly reduce the expense of the matrix computation in spectral clustering, while retaining good control on the clustering accuracy. Evaluation on a set of real data sets shows that a significant speedup for spectral clustering can be achieved with little degradation in clustering accuracy. Remarkably, our approximate algorithms enable a single machine to perform spectral clustering for a large dataset – the Poker Hand dataset – which consists of one million instances.

We also presented a theoretical analysis of our approximate spectral clustering algorithms using statistical perturbation theory. Our perturbation bound reveals that the mis-clustering rate is closely related to the amount of data perturbation – one can make the mis-clustering rate small by reducing the amount of perturbation. We show that the mis-clustering rate converges to zero as the number of representative points grows. These results provide a theoretical foundation for our algorithms and also have potentially wider applicability. In particular, a natural direction to pursue in future work is the use of other local data reduction methods (e.g., data squashing and condensation methods) for preprocessing; we believe that our bounds can be extended to these methods. We also plan to explore other methods for assigning clustering membership to the original data according to the membership of the representative data based on local optimization and edge-swapping methods.

References

[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete algorithms (SODA)*, 2007.

[2] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45:891–923, 1998.

[3] A. Asuncion and D. Newman. UCI Machine Learning Repository, Department of Information and Computer Science. http://www.ics.uci.edu/ mlearn/MLRepository.html, 2007.

[4] F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

[5] M. B$\overline{\text{a}}$doiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, 2002.

[6] P. S. Bradley and U. M. Fayyad. Refining initial points for k-means clustering. In *Proceedings of the 15th International Conference on Machine Learning (ICML)*, 1998.

[7] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[8] S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In *Fortieth ACM Symposium on Theory of Computing (STOC)*, 2008.

[9] P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. In *Proceedings of COLT*, pages 323–337, 2005.

[10] S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

[11] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2), 2004.

[12] W. A. Fuller. *Measurement Error Models*. Wiley, New York, 1987.

[13] R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Transactions of Information Theory*, 44(6):2325–2383, 1998.

[14] S. Gunter, N. N. Schraudolph, and A. V. N. Vishwanathan. Fast iterative kernel principal component analysis. *Journal of Machine Learning Research*, 8:1893–1918, 2007.

[15] J. A. Hartigan. *Clustering Algorithms*. Wiley, New York, 1975.

[16] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[17] B. Hendrickson and R. Leland. A multilevel algorithm for partitioning graphs. In *Proceedings of Supercomputing*, 1995.

[18] W. Hoeffding. The strong law of large numbers for U-statistics. Technical Report 302, University North Carolina Institute of Statistics Mimeo Series, 1961.

[19] L. Huang, D. Yan, M. I. Jordan, and N. Taft. Spectral clustering with perturbed data. In *Advances in Neural Information Processing Systems (NIPS)*, December 2008.

[20] A. Jain, M. Murty, and P. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[21] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.

[22] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[23] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1999.

[24] A. Kumar, Y. Sabbarwal, and S. Sen. A simple linear time $(1+\epsilon)$ -approximation algorithm for k-means clustering in any dimensions. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 2004.

[25] J. F. Lu, J. B. Tang, Z. M. Tang, and J. Y. Yang. Hierarchical initialization approach for k-means clustering. *Pattern Recognition Letters*, 29:787–795, 2008.

[26] D. Madigan, I. Raghavan, W. Dumouchel, M. Nason, C. Posse, and G. Ridgeway. Likelihood-based data squashing: a modeling approach to instance construction. *Data Mining and Knowledge Discovery*, 6(2):173–190, 2002.

[27] M. Meila and J. Shi. Learning segmentation with random walk. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[28] P. Mitra, C. A. Murthy, and S. K. Pal. Density-based multiscale data condensation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):1–14, 2002.

[29] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[30] J. M. Pena, J. A. Lozano, and P. Larranaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 14(1):1027–1040, 1999.

[31] S. J. Redmond and C. Heneghen. A method for initialising the k-means clustering algorithm using kd-trees. *Pattern Recognition Letters*, 28:965–973, 2007.

[32] R. J. Serfling. *Approximation Theorems of Mathematical Statistics.* Wiley, New York, 1980.

[33] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[34] A. Smola and B. Scholkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the 17th International Conference on Machine Learning*, 2000.

[35] G. Stewart. *Introduction to Matrix Computation*. Academic Press, 1973.

[36] J. W. Tukey. A survey of sampling from contaminated distributions. In I. Olkin, editor, *Contributions to Probability and Statistics*, pages 448–485. Stanford University Press, 1960.

[37] U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Annals of Statistics*, 36(2):555–586, 2008.

[38] C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2001.

[39] S. Yu and J. B. Shi. Multiclass spectral clustering. In *Proceedings of ICCV*, 2003.

[40] P. L. Zador. Asymptotic quantization error of continuous signals and the quantization dimension. *IEEE Transactions of Information Theory*, 28:139–148, 1982.

9 Appendix

In this Appendix we provide more detailed analysis and proofs that are omitted in the main body of the paper.

9.1 Proof of Lemma 4

We have

$$L(\tilde{A}) - L(A) = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} - (I + \Delta D^{-1})^{-\frac{1}{2}} D^{-\frac{1}{2}} (A + \Lambda) D^{-\frac{1}{2}} (I + \Delta D^{-1})^{-\frac{1}{2}}.$$

Since $||\Delta D^{-1}||_2 = o(1)$, a Taylor expansion to $f(X) = (I + X)^{-\frac{1}{2}}$ around $X = \mathbf{0}_{n \times n}$ yields

$$(I + \Delta D^{-1})^{-\frac{1}{2}} = I - \frac{1}{2} \Delta D^{-1} + O((\Delta D^{-1})^2).$$

It follows that

$$\begin{aligned}
L(\tilde{A}) - L(A) &= -D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}} - \frac{1}{2}D^{-\frac{1}{2}}(A+\Lambda)D^{-\frac{3}{2}}\Delta - \frac{1}{2}\Delta D^{-\frac{3}{2}}(A+\Lambda)D^{-\frac{1}{2}} \\
&\quad +D^{-\frac{1}{2}}(A+\Lambda)D^{-\frac{1}{2}}O((\Delta D^{-1})^2) \\
&= -D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}} - \left(\frac{1}{2}+o(1)\right)\left[D^{-\frac{1}{2}}AD^{-\frac{3}{2}}\Delta - \Delta D^{-\frac{3}{2}}AD^{-\frac{1}{2}}\right].
\end{aligned}$$

9.2 Proof of Lemma 5

Theorem 6 is implied by Lemma 5, the proof of which we present here. The proof involves an application of the theory of U-statistics. We provide a detailed proof for the univariate case while only stating the results for the multivariate case. Before we proceed, we present the definition and a classical result on U-statistics.

Definition [32]. Let $\mathbf{x}_1, ..., \mathbf{x}_n$ be i.i.d. drawn from some probability distribution $\mathbb{P}$. A U-statistic is defined as

$$U_n = U(\mathbf{x}_1, ..., \mathbf{x}_n) = \frac{1}{\binom{n}{m}} \sum_{(i_1,...,i_m)\in\Pi_m} h(\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_m}),$$

where h is a symmetric kernel and Π_m is the set of all m-subsets of $\{1, ..., n\}$. Let $\theta \triangleq \mathbb{E}_{\mathbb{P}} h(\mathbf{x}_1, ..., \mathbf{x}_m)$. The following due to Hoeffding [18] is used in our proof.

Lemma 12 ([18]). *Let $h(\mathbf{x}_1, ..., \mathbf{x}_m)$ be a symmetric kernel. If $\mathbb{E}_{\mathbb{P}}|h| < \infty$, then*

$$U_n \rightarrow_{a.s.} \theta.$$

Univariate case. For univariate variables $x_1, ..., x_n$ generated i.i.d. from (9), let $d_i/n > c_0 > 0$ for some universal constant c_0 for $i = 1, ..., n$. Then we have

$$\begin{aligned}
||D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}}||_F^2 &= \sum_{i=1}^n\sum_{j=1}^n \frac{\epsilon_{ij}^2}{d_i d_j} \leq_p \frac{1}{c_0^2 n^2}\sum_{i=1}^n\sum_{j=1}^n \delta_{ij}^2 \\
&= \frac{1}{c_0^2 n^2}\sum_{i=1}^n\sum_{j=1}^n c\left[\frac{(x_i-x_j)^2}{\sigma^4}\exp\left(-\frac{(x_i-x_j)^2}{\sigma^2}\right)(\epsilon_i-\epsilon_j)^2 + R_{ij}^2\right] \\
&\leq \frac{1}{c_0^2 n^2}\sum_{i=1}^n\sum_{j=1}^n(\epsilon_i-\epsilon_j)^2 + \frac{1}{c_0^2 n^2}\sum_{i=1}^n\sum_{j=1}^n R_{max}^2(\epsilon_i-\epsilon_j)^4 \\
&= I_{51} + I_{52}.
\end{aligned}$$

For U-statistics with symmetric kernel $h = (\epsilon_1 - \epsilon_2)^2$ and $h = (\epsilon_1 - \epsilon_2)^4$, we can easily show that $\mathbb{E}_{\mathbb{P}}|h| < \infty$ if the ϵ_i have bounded fourth moments. Note that our quantity differs from the U-statistic by a scaling constant that tends to 1 (it is known as a *V-statistic*). By Lemma 12, we have

$$I_{51} \rightarrow_{a.s.} c_1\sigma_\epsilon^{(2)}, \quad I_{52} \rightarrow_{a.s.} c_2\sigma_\epsilon^{(4)},$$

where $\sigma_\epsilon^{(2)}$ and $\sigma_\epsilon^{(4)}$ denote the second and fourth moments of ϵ, respectively. Thus we have shown that

$$||D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}}||_F^2 \leq_p c_1\sigma_\epsilon^{(2)} + c_2\sigma_\epsilon^{(4)},$$

for some universal constant c_1 and c_2 as $n \to \infty$.

We have

$$||\Delta D^{-\frac{3}{2}}AD^{-\frac{1}{2}}||_F^2 = \sum_{i=1}^n\sum_{j=1}^n \frac{a_{ij}^2}{d_i^3 d_j}\epsilon_{i.}^2 \leq_p \frac{1}{c_0^4 n^4}\sum_{i=1}^n\sum_{j=1}^n \epsilon_{i.}^2 = \frac{1}{c_0^4 n^3}\sum_{i=1}^n \epsilon_{i.}^2,$$

where $\epsilon_{i.}$ is defined as and bounded by

$$\begin{aligned}\epsilon_{i.}^2 &= \left[\sum_{k=1}^{n}\left(\frac{(x_i-x_k)}{\sigma^2}\exp\left(-\frac{(x_i-x_k)^2}{2\sigma^2}\right)(\epsilon_i-\epsilon_k)+R_{ik}\right)\right]^2 \\ &\leq n\sum_{k=1}^{n}\left[c(\epsilon_i-\epsilon_k)^2+R_{max}^2(\epsilon_i-\epsilon_k)^4\right].\end{aligned}$$

Thus, using the same set of U-statistics, we get

$$\begin{aligned}||\Delta D^{-\frac{3}{2}}AD^{-\frac{1}{2}}||_F^2 &\leq_p \frac{1}{c_0^4 n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\left[c(\epsilon_i-\epsilon_k)^2+R_{max}^2(\epsilon_i-\epsilon_k)^4\right] \\ &\leq_{a.s.} c_3\sigma_\epsilon^{(2)}+c_4\sigma_\epsilon^{(4)}.\end{aligned}$$

Multivariate case. For multivariate variables, we assume that the second and fourth moments of the noise on the k^{th} components are given by $\sigma_{k\epsilon}^{(2)}$ and $\sigma_{k\epsilon}^{(4)}$, respectively. Then similar to the univariate case, we have the following result for the multivariate case for Lemma 5. If the assumptions of Lemma 5 hold, then as $n\to\infty$,

$$\begin{aligned}||D^{-\frac{1}{2}}\Lambda D^{-\frac{1}{2}}||_F^2 &\leq_p \sum_{k=1}^{d}\left(c_k\sigma_{k\epsilon}^{(2)}+c_k'\sigma_{k\epsilon}^{(4)}\right)+\sum_{i\neq j=1}^{d}c_{ij}\left(\sigma_{i\epsilon}^{(2)}\sigma_{j\epsilon}^{(2)}\right)^{\frac{1}{2}} \\ &= c_1\sigma_\epsilon^{(2)}+c_2\sigma_\epsilon^{(4)}, \\ ||\Delta D^{-\frac{3}{2}}AD^{-\frac{1}{2}}||_F^2 &\leq_p \sum_{k=1}^{d}\left(c_k\sigma_{\epsilon}^{(2)}+c_k'\sigma_{k\epsilon}^{(4)}\right)+\sum_{i\neq j=1}^{d}c_{ij}\left(\sigma_{i\epsilon}^{(4)}\sigma_{j\epsilon}^{(4)}\right)^{\frac{1}{2}} \\ &= c_3\sigma_\epsilon^{(2)}+c_4\sigma_\epsilon^{(4)},\end{aligned}$$

where $\sigma_\epsilon^{(2)}$ and $\sigma_\epsilon^{(4)}$ denote the second and fourth moments of $\|\epsilon\|$, respectively.