Approximate Spectral Clustering*

Liang Wang, Christopher Leckie, Kotagiri Ramamohanarao, and James Bezdek

Department of Computer Science and Software Engineering
The University of Melbourne, Parkville, Victoria 3010, Australia
{lwwang, caleckie, rao}@csse.unimelb.edu.au, jbezdek@cs.uwf.edu

Abstract. While spectral clustering has recently shown great promise, computational cost makes it infeasible for use with large data sets. To address this computational challenge, this paper considers the problem of approximate spectral clustering, which enables both the feasibility (of approximately clustering in very large and unloadable data sets) and acceleration (of clustering in loadable data sets), while maintaining acceptable accuracy. We examine and propose several schemes for approximate spectral grouping, and make an empirical comparison of those schemes in combination with several sampling strategies. Experimental results on several synthetic and real-world data sets show that approximate spectral clustering can achieve both the goals of feasibility and acceleration.

Keywords: Spectral clustering, scalability, matrix approximation, sampling.

1 Introduction

As an exploratory data analysis tool, clustering aims to group objects of a similar kind into their respective categories (see [1] for a comprehensive survey). Given a data set $\mathcal{O}$ comprising n objects $\{o_1, o_2, \cdots, o_n\}$, (crisp) clustering partitions the data into c groups $G_1, G_2, \cdots, G_c$, so that $G_i \cap G_j = \emptyset$ if $i \neq j$ and $G_1 \cup G_2 \cup \cdots \cup G_c = \mathcal{O}$. In particular, pairwise grouping methods, such as spectral clustering [2], present an appealing alternative to traditional central grouping techniques (such as K-means), because 1) they are applicable to situations in which the objects are not naturally representable in terms of feature vectors; and 2) they avoid the assumption that all examples in a cluster must be close to a prototype. This means they are amenable to irregular-shaped clusters.

Spectral clustering algorithms usually rely on the eigendecomposition of a $n \times n$ similarity matrix (where n is the number of examples), which generally takes $\mathrm{O}(n^3)$ time and $\mathrm{O}(n^2)$ space complexity. In addition, to obtain such a similarity matrix it is necessary to compare all possible pairs of examples, which is computationally expensive for a large data set. These limitations make spectral clustering methods impractical (or computationally infeasible) when handling large data sets. Additional strategies are thus required to adapt to growing

* This work was supported by ARC Discovery Project DP0663196.

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 134–146, 2009.

data sizes while maintaining both cluster quality and speed. A spectral grouping approach based on the Nyström approximation was proposed in [3], which first solves a small-scale eigendecomposition problem on randomly chosen sample data, then computes approximated eigenvectors via extrapolation. Another incremental spectral clustering algorithm was proposed to handle "dynamic" evolving data in [4] by introducing the incidence vector/matrix. In [5], the spectral clustering algorithm is parallelized across distributed machines. However, these two methods either sacrifice accuracy or require distributed computing infrastructure. In contrast, this paper considers approximate spectral clustering (ASC) for "static" data without the use of distributed computation.

The motivations of approximate spectral clustering can be described as follows. When the data set is large and unloadable on the available computing platform, ASC provides an approximate solution to clustering (i.e., making clustering feasible), whereas it is impossible to use a literal clustering approach in batch mode on such data. If the data set is small, medium, or merely large but still loadable, then ASC may offer an approximation comparable to the literal solution but at a significantly reduced computational cost. In summary, the benefits of an approximate clustering scheme are "feasibility" for very large data sets and "acceleration" for manageably-sized data sets. To this end, this paper proposes two new methods for approximate spectral clustering, as well as the examination of an existing method described in [3]. One of them is based on matrix approximation, and the other uses a "sampling plus extension" approach. Our major contributions are as follows: 1) we present two different schemes for approximate spectral clustering; 2) we provide a comprehensive quantitative comparison of several approximate spectral clustering algorithms, together with a comparison of several sampling schemes; and 3) we provide extensive experimental results on synthetic and real data sets, and several meaningful conclusions are highlighted.

The rest of this paper is organized as follows. Section 2 gives a brief review of spectral clustering. Section 3 details several approximate algorithms. Section 4 introduces four kinds of sampling schemes. The results are presented in Section 5, prior to discussion and conclusion in Section 6.

2 Spectral Clustering

Spectral methods for clustering, e.g., normalized cut [6] and max-min cut [7], are based on the eigenvectors and eigenvalues of a symmetric positive semidefinite (SPSD) matrix of size $n \times n$ derived from a given data set. Let the symmetric $W \in \mathcal{R}^{n \times n}$ denote the weighted adjacency matrix for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ representing the n objects in $\mathcal{O}$ to be analyzed and edges $\mathcal{E}$ whose weights capture pairwise affinities between objects. Let D be a diagonal matrix with entries $D_{ii} = d_i$, where $d_i = \sum_j W_{ij}$ denotes the degree of the ith node, then the graph Laplacian matrix is defined as $L = D - W$ [8].

Let C_1 and C_2 be a bipartition of $\mathcal{V}$, i.e., $C_1 \cap C_2 = \emptyset$ and $C_1 \cup C_2 = \mathcal{V}$, and the volume of a set as the sum of the degrees within the set, i.e., $\mathrm{vol}(C_j) = \sum_{i \in C_j} d_i$.

The normalized cut between sets C_1 and C_2 is defined as [6]

$$\text{ncut}(C_1, C_2) = \frac{2 \cdot \text{cut}(C_1, C_2)}{\text{vol}(C_1) \parallel \text{vol}(C_2)} \tag{1}$$

where $\parallel$ denotes the harmonic mean and $\text{cut}(C_1, C_2) = \sum_{i \in C_1, j \in C_2} W_{ij}$. To minimize (1), Shi and Malik [6] showed that an approximate solution may be obtained by thresholding the eigenvector corresponding to the second smallest eigenvalue of the normalized Laplacian matrix $\mathcal{L}$, i.e.,

$$\mathcal{L} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2} \tag{2}$$

The matrix $\mathcal{L}$ is positive semidefinite, even when W is indefinite.

Extensions to multiple groups are possible, e.g., using multiple eigenvectors [9]. In this work we adopt this approach by computing the leading eigenvectors V from $(D^{-1/2} W D^{-1/2})V = V\Lambda_\lambda$. These eigenvectors induce an embedding of the objects in a low-dimensional subspace, in which the K-means clustering algorithm is then used to discover final partitions by grouping columns of V.

3 Approximate Spectral Clustering Algorithms

To address the complexity of spectral decomposition for large n, we now outline several approximate approaches to the problem. Spectral clustering generally deals with an $n \times n$ SPSD matrix, say M, which can be decomposed as $M = U\Sigma U^T$ with Σ the eigenvalues of M and U the associated eigenvectors. Suppose $m \ll n$ columns of M are sampled without replacement. Let A be the $n \times m$ matrix of these sampled columns, and S be the $m \times m$ matrix consisting of the intersection of these m columns with the corresponding m rows. Without loss of generality, we can rearrange the columns and rows of M such that

$$M = \begin{pmatrix} S & B \\ B^T & C \end{pmatrix} \text{ with } A = \begin{pmatrix} S \\ B^T \end{pmatrix} \tag{3}$$

where $B \in \mathcal{R}^{m \times (n-m)}$ contains the elements from the samples to the rest of the objects, and $C \in \mathcal{R}^{(n-m) \times (n-m)}$ contains the elements between all of the remaining objects. In the case of $m \ll n$, C is usually large.

3.1 nSPEC

The Nyström approximation has recently been studied in the machine learning community, e.g., for fast approximate Gaussian process classification [10] and low-rank approximation to the kernel matrix [11]. A spectral grouping method based on the Nyström approximation was proposed for image segmentation in [3]. We refer to this as nSPEC (Nyström-based Spectral Clustering).

The Nyström approximation uses S and A to approximate M as

$$M \approx \tilde{M} = AS^+ A^T \tag{4}$$

where '+' is the pseudoinverse. The Nyström approximation models C by BS^+B^T, and the resulting approximate eigenvalues and eigenvectors of M are

$$\tilde{\Sigma} = (\frac{n}{m})\Sigma_S \text{ and } \tilde{U} = \sqrt{\frac{m}{n}} A U_S \Sigma_S^+ \tag{5}$$

where $S = U_S \Sigma_S U_S^+$ [10].

The eigenvectors generated from the Nyström approximation are not exactly orthogonal because they are extrapolated from the eigenvectors of S, losing orthogonality in this process. If S is positive definite, Fowlkes et al. [3] used a one-shot method to solve for the orthogonalized approximate eigenvectors. Define $Q = S + S^{-1/2}BB^TS^{-1/2}$ and diagonalize it as $Q = U_Q \Lambda_Q U_Q^T$, then $\tilde{M}$ can be diagonalized as $\tilde{M} = \hat{U}\Lambda_Q\hat{U}^T$ with

$$\hat{U} = AS^{-1/2}U_Q\Lambda_Q^{-1/2} \tag{6}$$

3.2 cSPEC

An alternative column-sampling technique has been analyzed in the theoretical computer science community [12], which can also be used to approximate spectral decomposition of a large matrix using a subset of columns. The column-sampling method was initially introduced to approximate SVD (Singular Value Decomposition) for any rectangular matrix [12,13]. However, it has not yet explicitly been used in spectral clustering. Here we use it to approximate the spectral decomposition of M, and call the resulting method cSPEC (Column-sampling Spectral Clustering).

The column-sampling technique approximates the eigendecomposition of M by using the SVD of A directly. Suppose $A = U_A \Sigma_A V_A^T$, then the approximate eigenvectors and eigenvalues of M are given by the left singular vectors of A and the corresponding scaled singular values, i.e.,

$$\tilde{\Sigma} = \sqrt{\frac{n}{m}}\Sigma_A \text{ and } \tilde{U} = U_A = AV_A\Sigma_A^+ \tag{7}$$

Accordingly the approximation of M can be written as [14]

$$M \approx \tilde{M} = A\left(\sqrt{\frac{m}{n}}(A^TA)^{1/2}\right)^+ A^T \tag{8}$$

which has a very similar form to (4). When n is very large, the SVD on A directly is still quite demanding. Fortunately, we have

$$A^TA = V_A\Sigma_A^TU_A^TU_A\Sigma_AV_A^T = V_A\Sigma_A^2V_A^T \tag{9}$$

It is thus easy to obtain U_A and Σ_A by computing the SVD on $A^TA \in \mathcal{R}^{m\times m}$. Relatively, A^TA can be easily computed even for large n.

3.3 eSPEC

Another way to attack the scalability problem is "extensibility" (e.g., [15,16]). An extended scheme applies a clustering algorithm to a representative sample set, and then extends the sample result to obtain (approximate) clusters for the remaining data. Also, the extended clustering schemes can be effectively applied in cases in which data are collected sequentially. Here we propose a "sampling, clustering plus extension" solution, called eSPEC (Extensible Spectral Clustering), for approximate spectral clustering.

After the sample data S is obtained, we first use a literal spectral clustering algorithm to group them. Next we address the problem of out-of-sample extension, i.e., to assign each of the remaining $n-m$ objects to one of the c previously determined groups. We regard the whole $m \times n$ matrix (i.e., A^T) as a semantically-meaningful vectorial representation (*i.e.*, each object has m attributes, each of which corresponds to a similarity relation between the object and one of the m sample objects, leading to a virtual data set $\{x_i\}_{i=1}^n$). For learning the cluster-preserving embedding space from S, we adopt computationally-efficient locality preserving projection (LPP) [17], which is very similar to the mapping procedure used in spectral clustering algorithms.

Let $\mathcal{G}_\mathcal{S}$ denote a graph with m nodes corresponding to the m labeled samples $\{x_j^S\}_{j=1}^m$. An edge occurs between nodes i and j if x_i^S and x_j^S are "close" according to k-nearest neighbors ($k \in \mathcal{N}$). Let W_S be a symmetric $m \times m$ matrix, whose element W_S^{ij} is the weight of the edge joining nodes i and j, and is 0 if there is no such edge (W_S is thus sparse). To obtain the embedding space, we solve the generalized eigenvector problem

$$SL_S S^T f = \lambda S D_S S^T f \tag{10}$$

where L_S and D_S are the corresponding Laplacian matrix and diagonal degree matrix of $\mathcal{G}_\mathcal{S}$. Let the column vectors $f_1, \cdots, f_l$ be the solutions of the eigenvectors in (10), ordered according to their eigenvalues, $\lambda_1 < \cdots < \lambda_l$ ($l \leq m$). Thus, the embedding of x_i in the l-dimensional spectral space is represented as

$$y_i = F^T x_i \text{ with } F = [f_1, f_2, \cdots, f_l] \tag{11}$$

Out-of-sample extension can then be treated as a prediction problem in this embedding space. For each $x_j^e (j = m+1, m+2, \cdots, n)$ in B to be extended, we use F to project x_j^e to y_j^e in the learned embedding space. Together with the embedding $\{y_j^S\}_{j=1}^m$ of the m labeled samples o_j^S, we use the k-nearest neighbor classifier to assign the object o_j^e to the class label with the maximum votes from its k nearest neighbors measured in the spectral domain.

4 Sampling Schemes

How to effectively sample a small set of representative samples is critical to encoding the structure of the whole data set. In this work, we focus on the following four sampling schemes in our empirical comparison:

- *Random sampling* (RS) uses a uniform distribution $\{p_i = 1/n\}_{i=1}^{n}$ to choose columns of M. This is a simple and commonly used method, e.g., [3,15,14].
- *Selective sampling* (SS) [18] was shown to be superior to progressive sampling in [16]. This method first selects h distinguished objects using a max-min farthest point strategy. Then each object in $\mathcal{O}$ is grouped to its nearest distinguished object. Finally a small number of samples are randomly selected from each of the h groups to form the sample set.
- *K-means sampling* (KS) suggested in [11] simply chooses the sample points as the K-means cluster centers. This method is inapplicable where the objects cannot be represented by feature vectors. In addition, the computation cost depends greatly on the data size n and the feature dimension.
- *Probabilistic sampling* (PS) [13] uses the probability distribution $\{p_i\}_{i=1}^{n}$ to choose columns of M with $p_i = |M^{(i)}|^2/\|M\|_F^2$, where $|M^{(i)}|$ is the length of the ith column of M, and $\|M\|_F$ is the Frobenius norm of M. This method needs the whole matrix to compute p_i, making it impractical for large n.

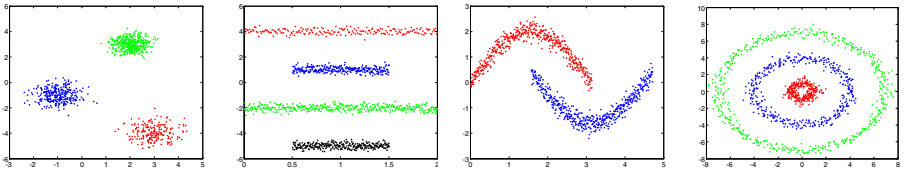
5 Experiments

In order to test these approximate algorithms, we carried out a number of experiments on several artificially generated and real-world data sets. Unless otherwise mentioned, in the following experiments the (Euclidean) distance matrix was computed in the original attribute space, which was then transformed to the affinity matrix by the Gaussian function, i.e., $W_{i,j} = \exp(-\|o_i - o_j\|^2/2\sigma^2)$. The number of clusters c was chosen manually, since choosing c is a difficult model-selection problem which lies outside of the scope of this work. All experiments were implemented in a Matlab 7.2 environment on a PC with an Intel 2.4GHz CPU and 2GB memory running Windows XP.

An accuracy metric AC has been widely used for clustering performance evaluation [4,19]. Suppose that l_i^c is the clustering label of object o_i and l_i^g is the ground truth label, AC is defined as $\max_{map} \sum_{i=1}^{n} \delta(l_i^g, map(l_i^c))/n$, where $\delta(l_1, l_2)$ is the delta function that equals 1 if and only if $l_1 = l_2$ and 0 otherwise, and map is the mapping function that permutes clustering labels to match equivalent ground-truth labels. The Kuhn-Munkres algorithm is usually used to obtain the best mapping [20]. In addition to accuracy, we also measure computational efficiency. For each experiment, we performed these approximate algorithms multiple times, and reported results in terms of the average accuracy (AAC) and the average computation time (ACT). Note that our programs have not been optimized for run-time efficiency.

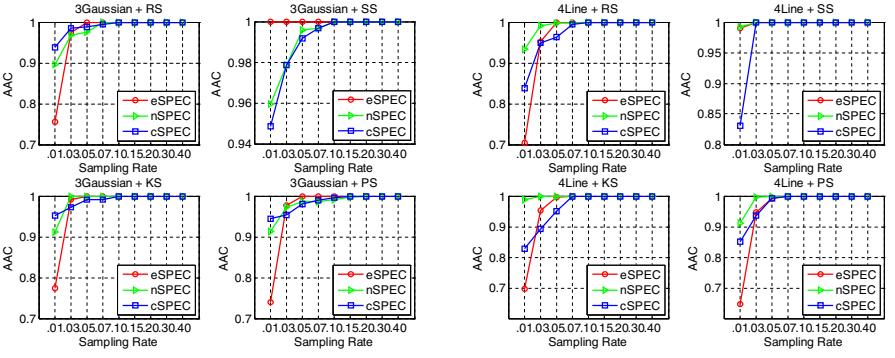
5.1 Results on Synthetic Data Sets

We begin with four synthetic data sets of different types and sizes (i.e., 3Gaussian, 4Line, 2HalfMoon, and 3Circle), whose scatter plots are shown in Figure 1, in which each color represents a cluster. The '3Gaussian' is a simple case in which even central grouping techniques can perform well. The later three cases

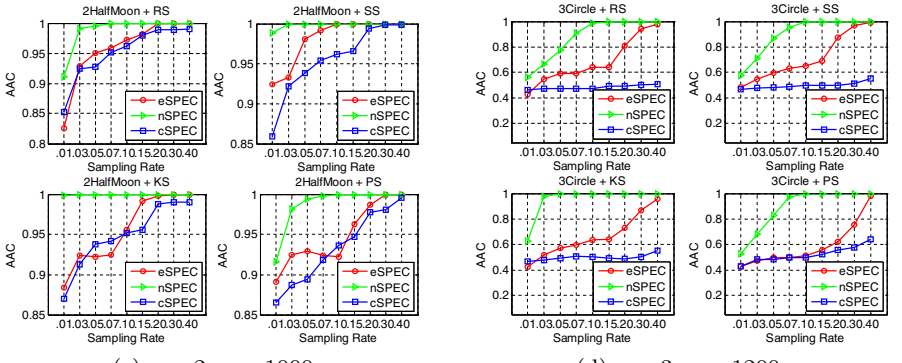


(a) 3Gaussian (b) 4Line (c) 2HalfMoon (d) 3Circle

Fig. 1. Synthetic data with different sizes and structures



(a) $c = 3, n = 900$ (b) $c = 4, n = 1200$

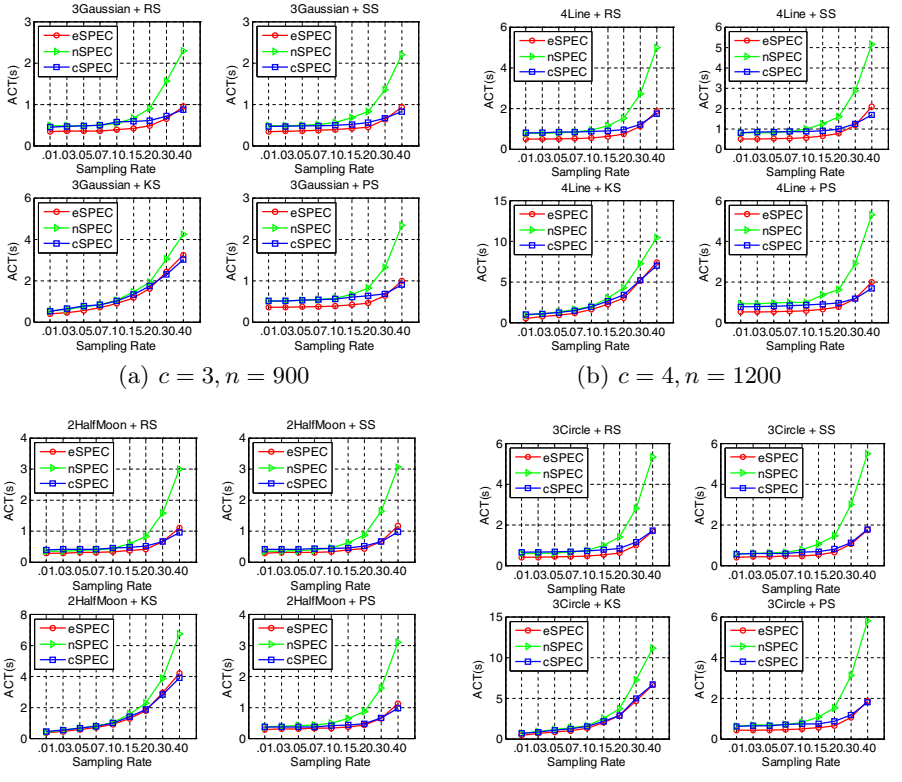


(c) $c = 2, n = 1000$ (d) $c = 3, n = 1200$

Fig. 2. Average accuracy of approximate clustering algorithms

are generally hard for central grouping, but easy for spectral clustering. These synthetic 2D data sets are relatively small so that we can perform both literal clustering and approximate clustering to measure the approximation error. We include these progressively harder data sets to test these approximate algorithms, though these synthetic cases are not necessarily realistic in practice.

First we performed literal spectral clustering on these data sets, and obtained fully correct clustering results at the computational cost of about 11s for



(a) $c = 3, n = 900$ (b) $c = 4, n = 1200$

(c) $c = 2, n = 1000$ (d) $c = 3, n = 1200$

Fig. 3. Average computation time of approximate clustering algorithms

'3Gaussian', 15s for '4Line', 27s for '2HalfMoon' and 27s for '3Circle'. Then, we applied eSPEC, nSPEC and cSPEC. We tried multiple sampling rates (i.e., m/n)[1]. For each sampling rate, 50 trials were made, then we computed the average clustering accuracy and the average computation time consumed by the clustering procedure. The results for the AACs and ACTs are respectively shown in Figures 2 and 3, from which it can be seen that:

– The approximate algorithms can achieve a good approximation, sometimes with the same accuracy as the literal solution when the sample size is sufficient. Moreover, these estimates are obtained using only a small fraction of the data and in much less time than the literal problem.

[1] What we are really concerned with is the sample size m, not m/n, since the ideal m generally depends just on the data structure (i.e., the number of clusters c and their distributions) than the data size n. That is, the necessary sample size m is basically fixed if the data structure is unchanged regardless of the real data size n.

- For complex-shaped clusters (e.g., 2HalfMoon and 3Circle), more samples are generally required to obtain stable results. When the number of samples is sufficient, the accuracy curve remains flat as the sample size increases further, but the computation time increases quickly.
- For simple data sets such as 3Gaussian and 4Line, the three algorithms perform similarly. But for more complex data sets such as 2HalfMoon and 3Circle, nSPEC performs better than the other two.
- In terms of accuracy, nSPEC performs the best overall, then eSPEC, and finally cSPEC. But in terms of computation time, nSPEC is the most expensive, and eSPEC is slightly cheaper than cSPEC.
- In terms of overall accuracy, SS performs best, then KS, and finally RS and PS. The computation time of RS, PS and SS is similar, though PS and SS are a little higher than RS. However, KS spends the most time, even in these small 2D data sets.

These numerical experiments on synthetic data sets suggest that the strategy of approximate spectral clustering can obtain comparable results to the literal approach but in much less time. In addition, overall nSPEC performs best but is most expensive in time. SS achieves the best tradeoff between accuracy and computation efficiency. Considering that synthetic data sets with controllable structures are designed only for simulation and are not realistic, real-world data sets with unknown data distributions would speak louder. Therefore, we will further evaluate these algorithms on several real data sets.

5.2 Results on Real Data Sets

We first considered two medium-sized data sets. 1) The multiple features (MF) data set from the UCI consists of binary image features of handwritten numerals ('0' $\sim$ '9') extracted from a collection of Dutch utility maps. Each class has 200 patterns, thus there are $n = 2000$ patterns in total. These digits are represented as a 649-dimensional vector in terms of 6 feature sets. We set the number of clusters $c = 10$ corresponding to 10 different numerals. 2) The Yale-B face data set[2] contains single light source images of 10 individuals, each seen under 585 viewing conditions (9 poses $\times$ 65 illumination conditions) [21]. Hence, the total number of images is $n = 5850$. Each original image was down-sampled to 30×40 pixels, leading to a 1200-dimensional vector representation. We set the number of clusters $c = 10$ corresponding to 10 different subjects.

For each data set, we applied the algorithms 25 times for each of several sampling rates with the selective sampling scheme. The AACs and ACTs are summarized in Table 1, where *ane* means the average number of examples for each cluster in the sample set. Table 1 shows that 1) the computation time of nSPEC is most expensive, then cSPEC, and finally eSPEC, which is consistent with the results on the synthetic data sets. 2) On the MF data set, nSPEC obtained the best accuracy, then eSPEC, and finally cSPEC, which is basically

[2] http://markus-breitenbach.com/machine_learning_data.php

Table 1. Summary of the results on the MF and Yale-B data sets

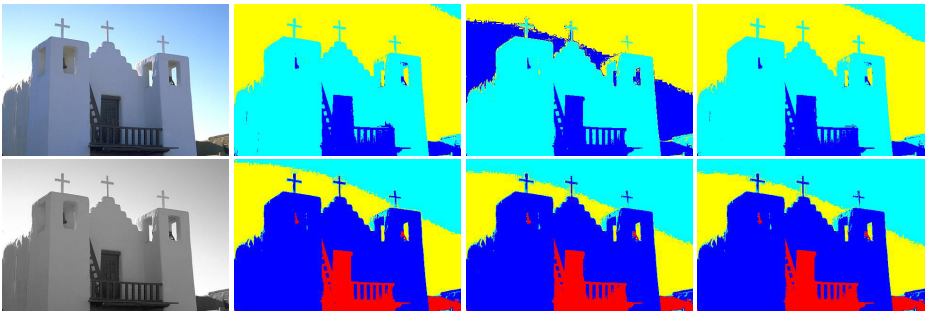
Algorithms			nSPEC		cSPEC		eSPEC	
Data sets	m/n	ane	AAC(%)	ACT(s)	AAC(%)	ACT(s)	AAC(%)	ACT(s)
MF	0.03	6	80.10	5.39	73.34	5.09	72.75	1.83
$c = 10$	0.05	10	80.51	5.65	73.76	5.27	75.29	2.08
$n = 2000$	0.07	14	80.57	5.74	76.99	5.33	78.85	2.39
Yale-B	0.01	≈ 6	80.21	13.89	82.03	12.89	72.25	4.05
$c = 10$	0.03	≈ 18	81.97	15.20	84.39	13.09	78.07	6.13
$n = 5850$	0.05	≈ 29	82.11	18.97	85.56	14.46	77.16	8.79

consistent with the results on the synthetic data sets (with similar data sizes). However, it is interesting to see that on the Yale-B data, cSPEC performed better than the other two algorithms. Note that cSPEC performs SVD on a larger submatrix of M than does the Nyström method ($A_{n\times m}$ versus $S_{m\times m}$). This could be a reason why cSPEC performs better than nSPEC in such a relatively large real-world problem.

We also applied these clustering algorithms to the problem of high-resolution image segmentation (where it is generally infeasible to use literal spectral clustering). Different features (such as intensity, color, texture, and proximity) can be used to compute the similarities between image pixels, e.g., locally-windowed color and texture histograms were used in [3]. We just used the intensity feature since our main concern is to demonstrate the feasibility of these approximate algorithms in the context of image segmentation, but not purely for image segmentation. Figure 4 shows segmentation results on three 481×321 images[3], in which pixels with the same color represent one group. We set $c = 3$ (or 4) for these images according to the number of visually meaningful components.

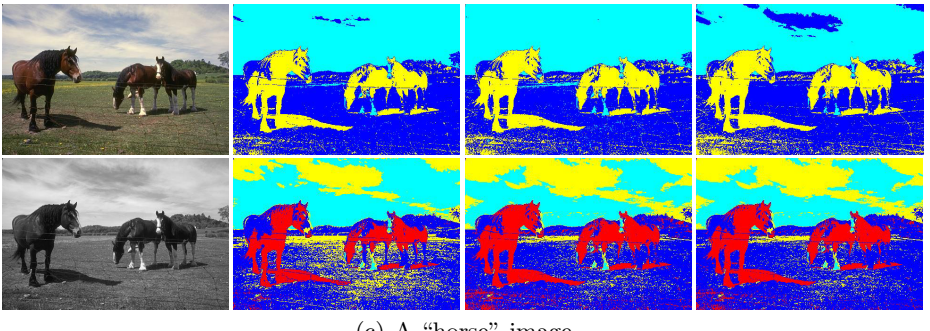
Running a literal spectral clustering algorithm on the whole image (which contains $n = 481 \times 321 = 154,401$ pixels) would be simply impossible in the Matlab environment. For these images, the number of sampled pixels was empirically chosen to be 150 (less than 0.1% of the number of total pixels), considering that there are far fewer coherent groups (*i.e.,* $c \ll n$) in a scene than pixels. We cannot measure the clustering error in this case because literal spectral clustering cannot be performed and we lack any form of ground truth. So, the best we can do for evaluation here is to resort to visual inspection of the segmentation results. In these three cases, all algorithms partitioned the images into meaningful components when $c = 4$ regardless of slight differences. More interesting, when $c = 3$, nSPEC gave results that were inconsistent with human perception of the intensity values in the images (i.e., a tendency to over-segmentation), whereas cSPEC and eSPEC performed similarly well. This seems to demonstrate again that cSPEC could be superior to nSPEC on these larger image data sets.

[3] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/



(a) A “house” image

(b) A “stone” image



(c) A “horse” image

Fig. 4. Intensity-based image segmentation results. For each of (a), (b) and (c), the left-most column is original color image (top) and corresponding intensity image (bottom); and the right three columns are respectively the segmentation results using eSPEC, nSPEC and cSPEC with $c = 3$ (top row) and $c = 4$ (bottom row).

6 Discussion and Conclusion

This paper has examined several approximate spectral clustering approaches. Extensive experiments on synthetic and real-world data sets show that these algorithms are not only feasible for very large data sets, but also provide acceleration on large data sets with comparable accuracy compared to the literal

solution. In particular, in terms of memory, the matrix (i.e., A, or S and B) needed in these three approximate algorithms can be simply computed on demand. This greatly reduces the memory requirements for very large-scale problems.

Accuracy and efficiency are two important factors in data clustering. Comparative results on the synthetic data sets have shown that nSPEC performs best in terms of accuracy, but this is not always the case in real-world data sets. We cannot thus say that a specific algorithm is always superior to the others. The computation time of nSPEC is consistently highest among all of the three algorithms, which may be due to its additional strategy for computing approximated orthogonalized eigenvectors. Relatively, eSPEC is cheapest, which makes out-of-sample extension more appealing when a large number of samples have been accumulated. To summarize, just as discussed in [1], there is no clustering algorithm that can be universally used to solve all problems. It is more likely that the performance of each clustering method depends strongly on the real characteristics of the data sets used. In this sense, it is not rational to claim a "best" in the context of clustering algorithms, though comparison on a wider variety of data is possible. However, among the four compared sampling schemes, selective sampling provides the best choice in terms of accuracy, efficiency and applicability for various types of data.

References

1. Xu, R., Wunsch II, D.: Survey of clustering algorithms. IEEE Trans. Neural Networks 16(3), 645–678 (2005)
2. Luxburg, U.: A tutorial on spectral clustering. Technical report, Max Planck Institute for Biological Cybernetics, Germany (2006)
3. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the Nyström method. IEEE Trans. Pattern Analysis and Machine Intelligence 26(2), 214–225 (2004)
4. Ning, H., Xu, W., Chi, Y., Gong, Y., Huang, T.: Incremental spectral clustering with application to monitoring of evolving blog communities. In: SIAM Conference on Data Mining (2007)
5. Miao, G., Song, Y., Zhang, D., Bai, H.: Parallel spectral clustering algorithm for large-scale community data mining. In: International Conference on WWW (2008)
6. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
7. Ding, C., He, X., Zha, H., Gu, M., Simon, H.: A min-max cut algorithm for graph partitioning and data clustering. In: International Conference on Data Mining, pp. 107–114 (2001)
8. Chung, F.: Spectral Graph Theory. American Mathematical Society (1997)
9. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems (2001)
10. Williams, C., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems, pp. 682–688 (2000)
11. Zhang, K., Tsang, I.W., Kwok, J.T.: Improved Nyström low-rank approximation and error analysis. In: International Conference on Machine Learning (2008)

12. Deshpande, A., Rademacher, L., Vempala, S., Wang, G.: Matrix approximation and projective clustering via volume sampling. In: Symposium on Discrete Algorithms (2006)
13. Drineas, P., Kannan, R., Mahoney, M.: Fast Monte Carlo algorithms for matrices II: computing a low-rank approximation to a matrix. SIAM Journal on Computing 36(1), 158–183 (2006)
14. Talwalkar, A., Kumar, S., Rowley, H.: Large-scale manifold learning. In: International Conference on Computer Vision and Pattern Recognition (2008)
15. Pavan, M., Pelillo, M.: Efficient out-of-sample extension of dominant-set clusters. In: Advances in Neural Information Processing Systems (2004)
16. Bezdek, J., Hathaway, R., Huband, J., Leckie, C., Kotagiri, R.: Approximate clustering in very large relational data. International Journal of Intelligent Systems 21(8), 817–841 (2006)
17. He, X., Niyogi, P.: Locality preserving projections. In: Advances in Neural Information Processing Systems (2003)
18. Wang, L., Bezdek, J.C., Leckie, C., Kotagiri, R.: Selective sampling for approximate clustering of very large data sets. International Journal of Intelligence Systems 23(3), 313–331 (2008)
19. Cai, D., He, X., Han, J.: Document clustering using locality preserving indexing. IEEE Trans. Knowledge and Data Engineering 17(2), 1637–1642 (2005)
20. Lovasz, L., Plummer, M.: Matching Theory. Akademiai Kiado. North Holland, Budapest (1986)
21. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Analysis and Machine Intelligence 23(6), 643–660 (2001)