Determining the Number of Clusters via Iterative Consensus Clustering [*]

Carl Meyer[†] Shaina Race [‡] Kevin Valakuzhy [§]

January 25, 2013

Abstract

We use a cluster ensemble to determine the number of clusters, k, in a group of data. A consensus similarity matrix is formed from the ensemble using multiple algorithms and several values for k. A random walk is induced on the graph defined by the consensus matrix and the eigenvalues of the associated transition probability matrix are used to determine the number of clusters. For noisy or high-dimensional data, an iterative technique is presented to refine this consensus matrix in way that encourages a block-diagonal form. It is shown that the resulting consensus matrix is generally superior to existing similarity matrices for this type of spectral analysis.

1 Introduction

Ensemble Methods have been used in various data mining fields to improve the performance of a single algorithm or to combine the results of several algorithms. In data clustering, these same strategies have been implemented, and the techniques are commonly referred to as consensus methods [15, 22]. Since no single algorithm will work best in any given class of data, it is a natural approach to use several algorithms to solve clustering problems. However, the vast majority of clustering algorithms in the literature require the user to specify the number of clusters, k, for the algorithm to create. In applied data mining, the problem is that it is unusual for the user to know this information before hand. In fact, the number of distinct groups in the data may be the very question that the data miner is attempting to answer.

This paper proposes a solution to this fundamental problem by using multiple algorithms with multiple values for k to determine the most appropriate value for the number of clusters. We begin with a brief theoretical motivation and an example which provides the intuition behind our basic approach. We will follow this discussion with results on real datasets which demonstrate the effectiveness of our iterated approach.

1.1 Data

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ be an $m \times n$ matrix of column data. For the particular implementation of our iterated consensus clustering (ICC) approach outlined herein, we assume the data in $\mathbf{X}$ is nonnegative and relatively noisy. Neither of these conditions are necessary for the general scheme of ICC but one of our preferred algorithms for dimension reduction is nonnegative matrix factorization (NMF), which, as the name suggests requires nonnegative data. Our main focus falls in the realm of document clustering, but we demonstrate that our method works equally well on other types of data. In document clustering the data matrix $\mathbf{X}$ is a term-by-document matrix where $\mathbf{X}_{ij}$ represents the frequency of term i in document j. The data in $\mathbf{X}$ are normalized and weighted according to term-weighting schemes like those described in [3, 25, 9].

1.2 Similarity Matrices

A similarity matrix $\mathbf{S}$ is an $n \times n$ symmetric matrix of pairwise similarities for the data in $\mathbf{X}$, where $\mathbf{S}_{ij}$ measures some notion of similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. Many clustering algorithms, particularly those of the spectral variety rely on a similarity matrix to cluster data points [26, 19, 28, 6, 16]. While many types of similarity functions exist, the most commonly used function in the literature is the Gaussian similarity function, $\mathbf{S}_{ij} = \exp(-\frac{\|x_i - x_j\|}{2\sigma^2})$, where σ is a parameter,

[*] This research was supported in part by grants DMS-1127914, NSA REU H9823-10-1-0252, NSF REU DMS-01063010

[†] North Carolina State University, Mathematics, Institute of Advanced Analytics & SAMSI (meyer@ncsu.edu)

[‡] North Carolina State University, Mathematics & Operations Research (slrace@ncsu.edu)

[§] University of North Carolina Chapel Hill, Mathematics (kevin.valakuzhy@gmail.com)

set by the user. We will discuss our own similarity matrix, known as a consensus matrix, in Section 2. The goal of clustering is to create clusters of objects that have high intra-cluster similarity and low inter-cluster similarity. Thus any similarity matrix, once rows and columns are ordered by cluster, should have a nearly block-diagonal structure.

1.3 Nearly Uncoupled Markov Chains

Any similarity matrix, $\mathbf{S}$, can be viewed as an adjacency matrix for nodes on an undirected graph. The n data points act as nodes on the graph and edges are drawn between nodes with weights from the similarity matrix. Figure 1 illustrates such a graph, using the thickness of an edge to indicate its weight. While edges may exist between nodes in separate clusters, we expect the weights of such edges to be far less than the weights within the clusters.

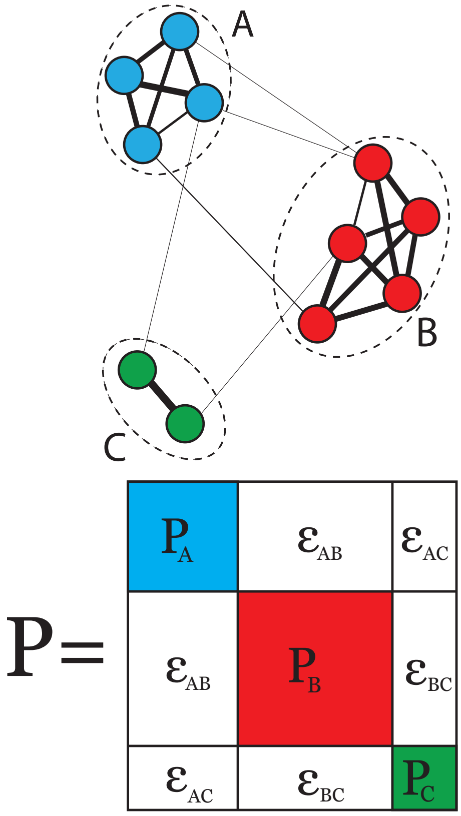


Figure 1: A Nearly Uncoupled Markov Chain

A random walk is induced on the graph and a transition probability matrix, $\mathbf{P}$, is created from the similarity matrix, $\mathbf{S}$, as $\mathbf{P} = \mathbf{D}^{-1}\mathbf{S}$ where $\mathbf{D} = diag(\mathbf{Se})$, and $\mathbf{e}$ is a vector of ones. It is easily verified that a steady-state distribution of this Markov chain is given by $\pi^T = \frac{\mathbf{e}^T\mathbf{D}}{\mathbf{e}^T\mathbf{De}}$. Let $\mathbf{Q} = diag(\pi) = \frac{\mathbf{D}}{\mathbf{e}^T\mathbf{De}}$. $\mathbf{P}$ represents a *reversible* Markov chain because it satisfies the detailed balance equations, $\mathbf{QP} = \mathbf{P}^T\mathbf{Q}$ [7, 27]. This condition guarantees that the eigenvalues of $\mathbf{P}$ are real since $\mathbf{Q}^{1/2}\mathbf{P}\mathbf{Q}^{-1/2} = \mathbf{Q}^{-1/2}\mathbf{P}^T\mathbf{Q}^{1/2}$ indicates that $\mathbf{P}$ is similar to a symmetric matrix. In fact, this symmetric matrix, $\mathbf{Q}^{1/2}\mathbf{P}\mathbf{Q}^{-1/2}$, is precisely $\mathbf{I} - \mathscr{L}$ where $\mathscr{L}$ is the **normalized Laplacian** matrix used in many spectral clustering algorithms [28]. For computational considerations, we use this symmetric matrix to compute the spectrum of $\mathbf{P}$ in our algorithm.

Let $\sigma(\mathbf{P}) = \{1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n\}$ be the spectrum of $\mathbf{P}$. A block diagonally dominant Markov Chain is said to be *nearly uncoupled* if the diagonal blocks of $\mathbf{P}$ are themselves *nearly stochastic*, meaning $\mathbf{P}_i\mathbf{e} \approx 1$ for each i (for a more precise definition, see [17]). A nearly uncoupled Markov chain with real eigenvalues will have exactly k eigenvalues near 1 where k is the number of blocks on the diagonal. This cluster of eigenvalues, $[\lambda_1, \ldots, \lambda_k]$, near 1 is known as the **Perron cluster** [20, 17, 29]. Moreover, if there is no further decomposition (or meaningful subclustering) of the diagonal blocks, a relatively large gap between the eigenvalues λ_k and λ_{k+1} is expected [20, 17, 29]. It has previously been suggested that this gap be observed to determine the number of clusters in data [28]. However, as we will demonstrate in Section 6, the most common similarity matrices in the literature do not impart the level of uncoupling that is necessary for a visible Perron cluster. The main goal of our algorithm is to construct a nearly uncoupled Markov chain using a similarity matrix from a cluster ensemble. The next section fully motivates our approach.

2 The Consensus Similarity Matrix

We will build a similarity matrix using the results of several, say N, different clustering algorithms. As previously mentioned, most clustering algorithms require the user to input the number of desired clusters. We will choose 1 or more values for k, denoted by $\tilde{k} = [\tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_J]$, and use each of the N algorithms to partition the data into $\tilde{k}_i$ clusters, for $i = 1, \ldots, J$. The result is a set of JN clusterings. These clusterings are recorded in a **consensus matrix**, $\mathbf{M}$, by setting M_{ij} equal to the number of times observation i was clustered with observation j. Such a matrix has become popular for ensemble methods, see for example [22, 15]. We will then observe the eigenvalues of the transition probability matrix of the random walk on the graph associated with the consensus matrix.

To motivate our approach, we'll look at a brief fabricated example. We will use the vertices from the graph in Figure 1 which are clearly separated into 3 clusters. Figure 2 illustrates (a) two different clusterings of these points (each with $\tilde{k} = 5$ clusters), (b) the consensus similarity matrix resulting from these two clusterings, and (c) the first few eigenvalues of the transition probability matrix, sorted by magnitude. Using an incorrect guess of $\tilde{k} = 5$ and 2 clusterings, the correct value of k is discovered by counting the number of eigenvalues in the Perron cluster.

Our use of this consensus similarity matrix relies on the following assumptions about our underlying clustering algorithms:

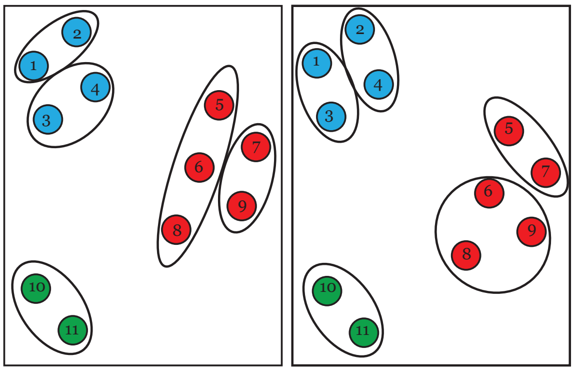
- If there are truly k distinct clusters in a given dataset, and a clustering algorithm is set to find $\tilde{k} > k$ clusters, then the original k clusters will be broken apart into smaller clusters to make $\tilde{k}$ total clusters.
- Further, if there is no clear "subcluster" structure, meaning the original k clusters do not further break down into meaningful components, then different algorithms will break the clusters apart in different ways.

Before discussing adjustments made to this basic approach, we provide a brief description of the clustering algorithms used herein.

3 Clustering Algorithms

The authors have chosen four different algorithms to form the consensus matrix: principal direction divisive partitioning (PDDP) [4], k-means, and expectation-maximization with Gaussian mixtures (EMGM) [21]. For each round of clustering, k-means is run twice, once initialized randomly and once initialized with the centroids of the clusters found by PDDP. This latter hybrid, "PDDP-k-means", is considered the 4^{th} algorithm. For text data sets and the clustering of symmetric matrices, spherical k-means is used as opposed to Euclidean k-means.

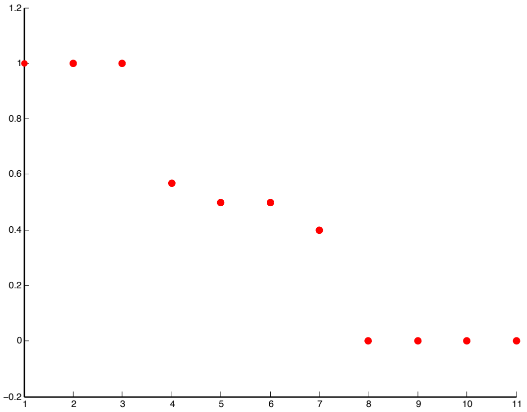
Three different dimension reductions are used to alter the data input to each of these algorithms. Our motivation for this is focused along three objectives. The first objective is merely to reduce the size of the data matrix, which speeds the computation time of the clustering algorithms. The second objective is to reduce noise in the data. The final objective is to decompose the data into components that reveal underlying patterns or features. The first dimension reduction is the ever popular Principal Components Analysis (PCA) [8]. The second dimension reduction is a simple truncated Singular Value Decomposition (SVD) [18]. We use Larson's PROPACK software to efficiently compute both the SVD and PCA [13]. The third dimension reduction is a nonnegative matrix factorization (NMF) [14, 11]. The NMF algorithm used is the alternating constrained least squares (ACLS) algorithm [12] with sparsity parameters



(a) Two Different Clusterings with $\tilde{k} = 5$

$$\begin{array}{c|cccc|ccccc|cc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\
\hline
1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 1 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
5 & 0 & 0 & 0 & 0 & 2 & 1 & 1 & 1 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 1 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 \\
8 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 1 & 0 & 0 \\
9 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 \\
\hline
10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 \\
11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2
\end{array}$$

(b) Resulting Consensus Matrix



(c) $k = 3$ Eigenvalues in Perron Cluster

Figure 2: A Simple Motivating Example

$\lambda_W = \lambda_H = 0.5$, and initialization of factor W with the Acol approach outlined in [12]. For further explanation on how and why these techniques are used for dimension reduction, see the complete discussion in [24].

All of the above dimension reduction techniques require the user to input the level of the dimension reduction, r. The choices for this parameter can provide hundreds of different clusterings for a single algorithm. Here, we choose three different values for r: r_1, r_2, r_3. For smaller datasets where it is feasible to compute the complete SVD/PCA of the data matrix, r_1, r_2, r_3 were chosen to be the number of principal components required to capture 60%, 75% and 90% of the variance in the data respectively. We require that the values of r_1, r_2, and r_3 be unique. For larger document datasets ($n \geq 3000$ documents), where it is unwieldy to compute the entire SVD of the data matrix, values for r are chosen such that $r_1 \approx 0.01n$, $r_2 \approx 0.05n$, and $r_3 \approx 0.1n$.

Using our four different algorithms, 4 representations of the data (raw data plus three dimension reductions), three ranks of dimension reduction we can create up to $N = 40$ different clusterings for each value of $\tilde{k}$.

4 Our Method

The base version of our method is simple and works well on datasets with well-defined, well-separated clusters. Section 5 discusses enhancements that provide an exploratory method for larger, noisier datasets.

ALGORITHM 4.1. (BASIC METHOD)

Input: Data Matrix $\mathbf{X}$ and a sequence $\tilde{k} = \tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_J$

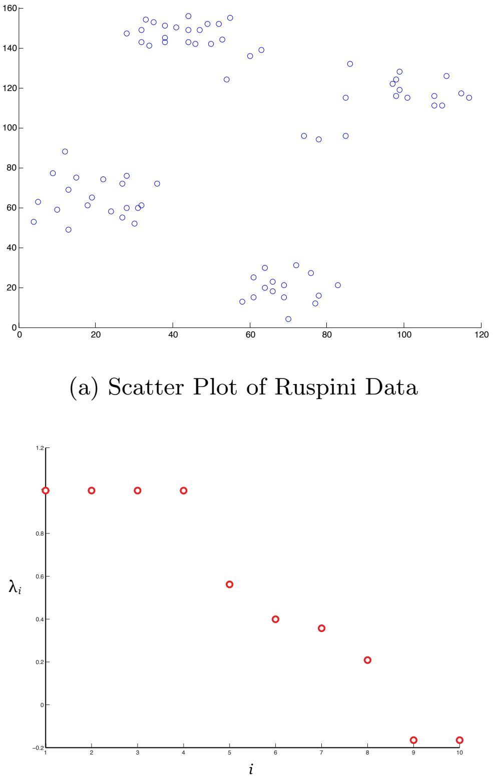
1. Using each clustering method $i = 1, \ldots, N$, partition the data into $\tilde{k}_j$ clusters, $j = 1, \ldots, J$

2. Form a consensus matrix, $\mathbf{M}$ with the JN different clusterings determined in step 1. Let $\mathbf{D} = \text{diag}(\mathbf{Me})$

3. Compute the eigenvalues of $\mathbf{P}$ using the symmetric matrix $\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{M}\mathbf{D}^{-1/2}$ and identify the Perron cluster.

Output: The number of eigenvalues, k, contained in the Perron cluster.

To demonstrate the effectiveness of our base method on a simple synthetic dataset, we employed it on the Ruspini dataset, a two-dimensional dataset that has commonly been used to validate clustering methods and metrics. We simply used our four different algorithms and five different values for $\tilde{k} = 6, \ldots, 10$. The resulting eigenvalue plot, displayed in Figure 3, clearly shows the correct number, $k = 4$, of eigenvalues in the Perron cluster.



(a) Scatter Plot of Ruspini Data

(b) Eigenvalues of Probability Matrix

Figure 3: Results on Ruspini Dataset

For the purposes of comparison Figure 4 shows the eigenvalues of the Markov chain induced by the Gaussian similarity matrix. In all of our experiments we set the parameter $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n} \|\mathbf{x}_i - \mu\|_2^2$ where $\mu = \frac{\mathbf{Xe}}{n}$ is the mean. While there are 4 relatively large eigengaps in Figure 4, the largest gap occurs after the first eigenvalue and there is little indication of the block-diagonal dominance (uncoupling) illustrated in Figure 1.

5 Adjustments to the Algorithm

This section discusses two adjustments to our algorithm, each of which are meant to combat the effect of noise in large datasets, particularly document sets. In document clustering, although the underlying topics that define individual clusters may be quite distinct, the spatial concept of "well-separated" clusters becomes convoluted

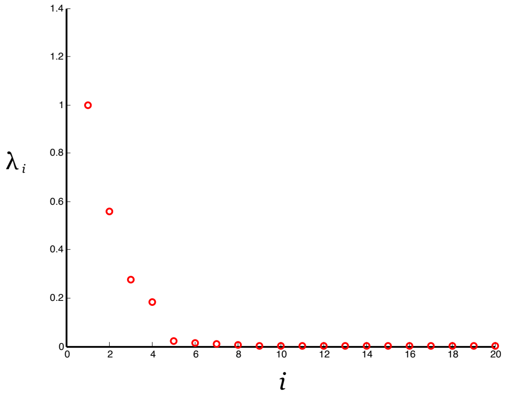


Figure 4: Eigenvalues from Gaussian Similarity Matrix for Ruspini Dataset

in high dimensions. Thus the nearly uncoupled structure depicted in Figure 1 is rare in practice. The adjustments presented in this section are meant to refine the data in an iterative way to encourage such uncoupling.

5.1 Drop Tolerance, τ There will necessarily be some similarity between documents from different clusters. As a result, clustering algorithms make errors. However, if the algorithms are independent, it is reasonable to expect that the majority of algorithms will not make the *same* error. To this end, we introduce a *drop tolerance*, τ, $0 \le \tau < 0.5$ for which we will drop (set to zero) entries $\mathbf{M}_{ij}$ in the consensus matrix if $\mathbf{M}_{ij} < \tau JN$. For example $\tau = 0.1$ means that when x_i and x_j are clustered together in fewer than 10% of the clusterings they are disconnected in the graph.

5.2 Iteration In the basic algorithm previously outlined, we use several clusterings to transform our original data matrix, $\mathbf{X}$, into a similarity matrix, $\mathbf{M}$. The rows/columns of $\mathbf{M}$ are essentially a new set of variables describing our original observations. Thus $\mathbf{M}$ can be used as the data input to our clustering algorithms, and the procedure can be iterated as follows:

Algorithm 5.1. (Iterated Method (ICC))
Input: Data Matrix $\mathbf{X}$, drop-tolerance τ, and sequence $\tilde{k} = \tilde{k}_1, \tilde{k}_2, \dots, \tilde{k}_J$

1. Using each clustering method $i = 1, \dots, N$, partition the data into $\tilde{k}_j$ clusters, $j = 1, \dots, J$
2. Form a consensus matrix, $\mathbf{M}$ with the JN different clusterings determined in step 1.
3. Set $\mathbf{M}_{ij} = 0$ if $\mathbf{M}_{ij} < \tau JN$.
4. Let $\mathbf{D} = \text{diag}(\mathbf{Me})$. Compute the eigenvalues of $\mathbf{P}$ using the symmetric matrix $\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{M}\mathbf{D}^{-1/2}$.
5. If the Perron Cluster is clearly visible, stop and output the number of eigenvalues in the Perron cluster, k. Otherwise, repeat steps 1-5 using $\mathbf{M}$ as the data input in place of $\mathbf{X}$.

While the uncoupling benefit of the drop tolerance should be clear from the graph in Figure 1, the benefit of iteration may not be apparent to the reader until the result is visualized. In the next section, we will use noisy datasets to illustrate.

6 Results on Noisy Data

In order to demonstrate the uncoupling effect of iteration, we use three datasets that are difficult to cluster because of their inherent noise.

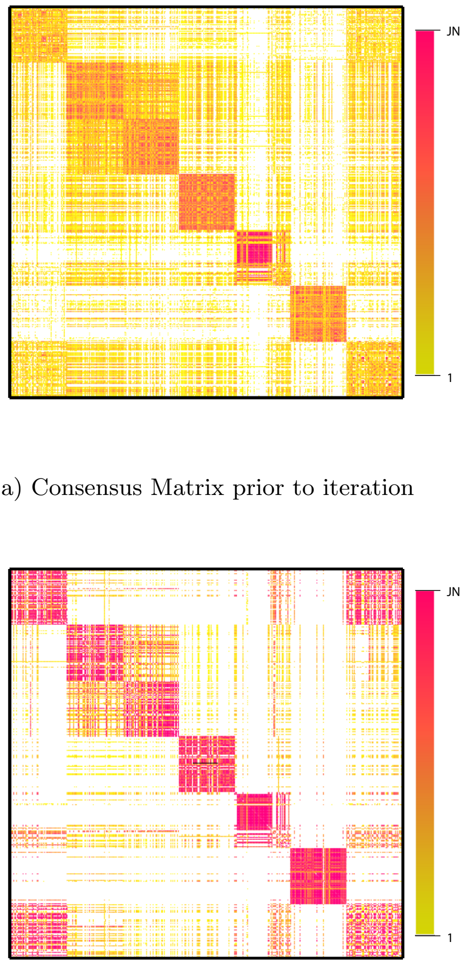
6.1 Newsgroups Dataset Our Newsgroups dataset is a subset of 700 documents, 100 from each of $k = 7$ clusters, from the 20 Newsgroups dataset [1]. The topic labels from which the documents were drawn can be found in Table 1.

Alt: Atheism
Comp: Graphics
Comp: OS MS Windows Misc.
Rec: Sport Baseball
Sci: Medicine
Talk: Politics Guns
Talk: Religion Misc.

Table 1: Topics for Newsgroups Dataset

The documents were clustered using $\tilde{k} = [10, 11, \dots, 20]$ clusters and a drop tolerance of $\tau = 0.1$. David Gleich's VISMATRIX tool allows us to visualize our matrices as heat maps. In Figure 5, observe the difference between the consensus matrix prior to iteration (after the drop tolerance enforced) and after just two iterations. Each non-zero entry in the matrix is represented by a colored pixel. The colorbar on the right indicates the magnitude of the entries by color.

After 2 iterations, the magnitude of intra-cluster similarites are clearly larger and the magnitudes of inter-cluster similarities (noise) are noticeably diminished. Note the strong similarities between clusters 1 and 7, and some weaker



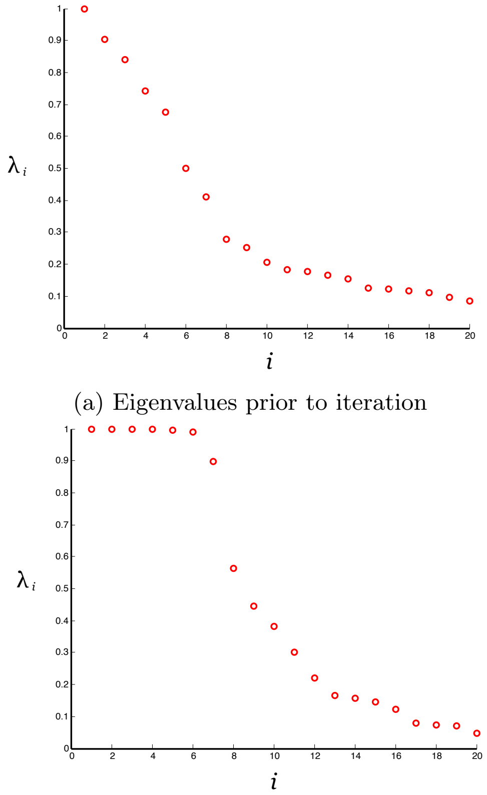
(a) Consensus Matrix prior to iteration

(b) Consensus Matrix after 2 iterations

Figure 5: The Uncoupling Effect of Iteration

similarities between clusters 2 and 3. This is due to the meaningful subcluster structure of the document collection: the categorical topics for clusters 1 and 7 are "atheism" and "misc. religion" respectively and the topics for clusters 2 and 3 are "computers- graphics" and "computers- OS MS windows misc". In fact, one of the beautiful aspects of our algorithm is its ability to detect this "subcluster" structure of data.

In Figure 6 we visualize the uncoupling effect of iteration by observing the difference in the eigenvalues of the transition probability matrix. Prior to iteration, the Perron cluster of eigenvalues is not apparent because there is still too much inter-cluster noise in the matrix. However, after 2 iterations, the Perron cluster



(a) Eigenvalues prior to iteration

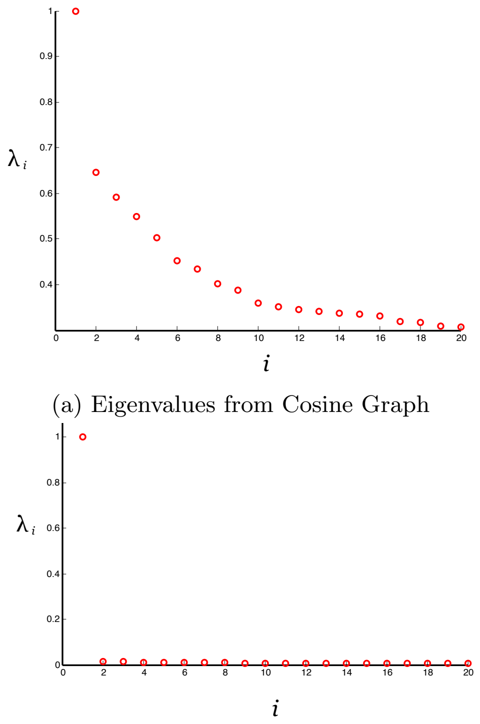
(b) Eigenvalues after 2 iterations

Figure 6: Newsgroups Dataset - ICC Results

is clearly visible, and contains the "correct" number of $k = 7$ eigenvalues. Furthermore, the 7^{th} eigenvalue belonging to the Perron-cluster is smaller in magnitude ($\lambda_7 = 0.89$) than the others ($\lambda_6 = 0.99$). This type of effect in the eigenvalues should cause the user to consider a subclustering situation like the one caused by the topics labels "atheism" and "misc. religion" where one topic could clearly be considered a subtopic of another.

To complete our discussion of this text dataset, we present in Figure 7 the eigenvalue plots of the Markov chains induced by two other similarity matrices, the cosine matrix and the Gaussian matrix. It is evident from this illustration that these measures of similarity fail to yield a nearly uncoupled Markov chain.

6.2 PenDigits17 PenDigits17 is a dataset, a subset of which was used in [6], which consists of coordinate observations made on handwritten digits. There are roughly 1000 instances each of $k = 2$ digits, '1's and '7's, drawn by 44 writers. This is considered a difficult dataset because of the similarity of the two digits and the number of ways to draw each. The complete PenDigits



(a) Eigenvalues from Cosine Graph

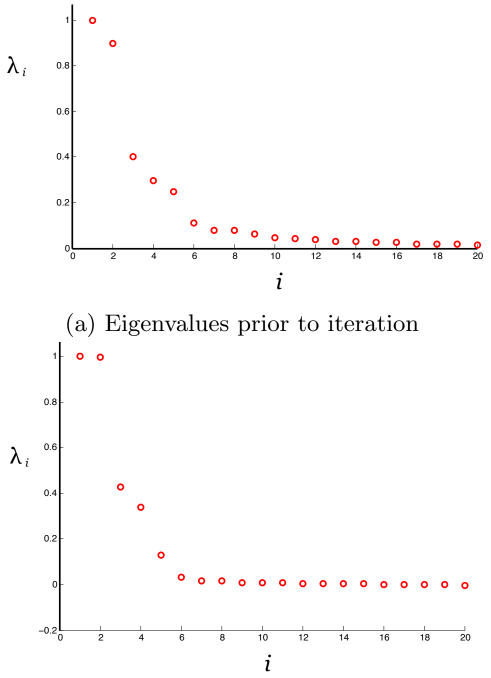
(b) Eigenvalues from Gaussian Graph

Figure 7: Newsgroups dataset

dataset is available from the UCI machine learning repository [2]. For our experiments we used the sequence $\tilde{k} = [3, 4, 5, 6]$ and a drop-tolerance $\tau = 0.1$. As seen in Figure 8 the Perron-cluster is convincing prior to iteration, and the system is almost completely uncoupled after 6 iterations.

For the purposes of comparison, we observe the eigenvalues of the transition probability matrices associated with the graphs defined by the cosine similarity matrix (used for clustering of this dataset in [6]) and the Gaussian similarity matrix. In Figure 9 it is again clear that these similarity matrices are inadequate for determining the number of clusters.

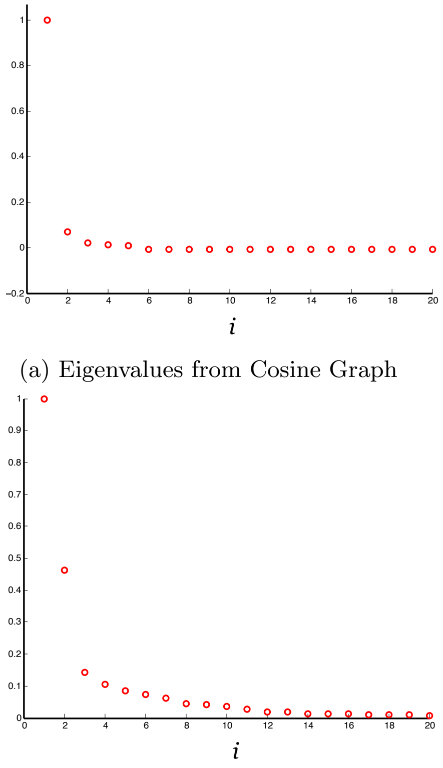
6.3 AGblog is an undirected hyperlink network mined from 1222 political blogs. This dataset was used in [6] and is described in [10]. It contains $k = 2$ clusters pertaining to the liberal and conservative division. We set our algorithms to find $\tilde{k} = [2 : 7]$ clusters with a drop tolerance of $\tau = 0.2$. The resulting eigenvalue plots are displayed in Figure 10.



(a) Eigenvalues prior to iteration

(b) Eigenvalues after 6 iterations

Figure 8: PenDigits17 Dataset - ICC Results

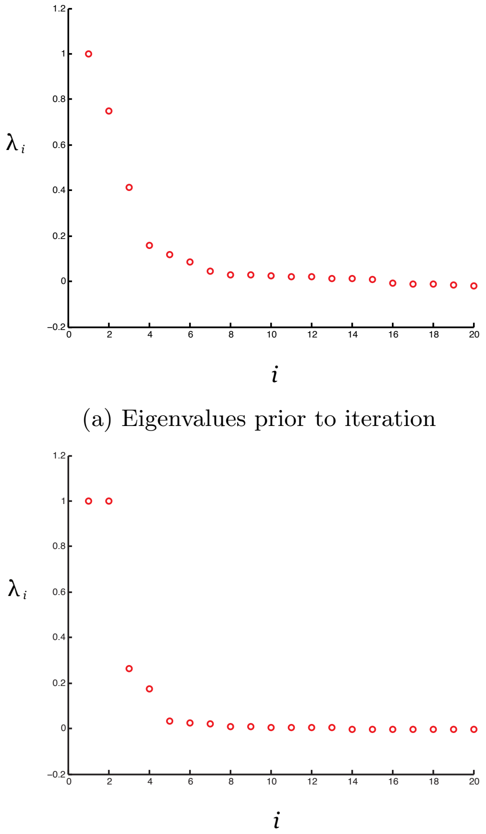


(a) Eigenvalues from Cosine Graph

(b) Eigenvalues from Gaussian Graph

Figure 9: PenDigits17 Dataset

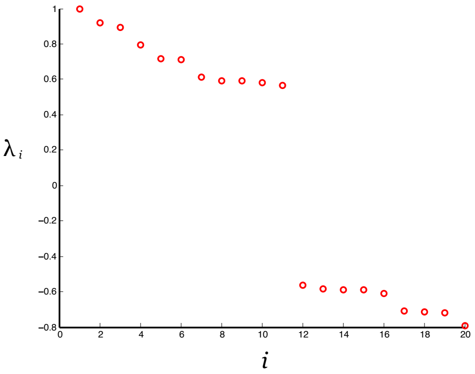
Figure 11 displays the eigenvalues from the Markov chain imposed by the similarity matrix used in [6], which was simply the original hyperlink matrix. This plot is particularly interesting because it does contain what appears to be a Perron cluster as defined by the large gap after the 11^{th} eigenvalue. There is, however, no indi-



(a) Eigenvalues prior to iteration

(b) Eigenvalues after 6 iterations

Figure 10: AGBlog Dataset - ICC Results



(a) Eigenvalues from Hyperlink Graph

Figure 11: AGBlog Dataset

cation in any type of analysis that suggests there are 11 communities in this dataset. We encourage readers to check out the visualizations of this graph at www4.ncsu.edu/~slrace which support this hypothesis. We believe this "uncoupling" is due, in fact, to rather isolated blogs that do not link to other sites with the frequency that others do. This example warns that outliers can have a misleading effect on the eigenvalues of an affinity matrix.

It may be unreasonable to expect a graph clustering algorithm like *Normalized Cut* (NCut) [26], *Power Iteration* (PIC) [6], or that of Ng, Jordan and Weiss (NJW) [19] to accurately divide a graph into two clusters when the eigenvalues of the associated Markov chain indicate eleven potential groups to be formed rather than two. In fact, we see the accuracy or purity of the clusters found by these algorithms may increase dramatically when the consensus matrix is used in place of the original matrix. Table 2 shows the purity of the clusterings found by these three spectral algorithms which were compared in [6]. The consensus matrix makes the true clusters more obvious to the first two algorithms, and has little to no effect on the third. The cluster solutions from the three algorithms on the consensus matrix are identical. This type of agreement is important in practice because it gives the user an additional level of confidence in the cluster solution [24].

Similarity Matrix	NCut	NJW	PIC
Undirected Hyperlink	0.52	0.52	0.96
Consensus Matrix	0.95	0.95	0.95

Table 2: Comparison of purity measurements for spectral algorithms on two similarity matrices for AGBlog Data

7 Conclusions

This paper demonstrates the effectiveness of Iterated Consensus Clustering (ICC) at the task of determining the number of clusters, k, in a dataset. Our main contribution is the formation of a consensus matrix from multiple algorithms and dimension reductions without prior knowledge of k. This consensus matrix is superior to other similarity matrices for determining k due to the nearly uncoupled structure of its associated graph. If the graph of the initial consensus matrix is not nearly uncoupled, then the adjustments of iteration and drop tolerance outlined in Section 5 will encourage such a structure.

Once the number of clusters is known, ICC has been previously been shown to obtain excellent clustering results by encouraging the underlying algorithms to agree upon a common solution through iteration [23]. Here we have demonstrated that using the consensus similarity matrix instead of existing similarity matrices can improve the performance of existing spectral clustering algorithms as suggested in [15].

ICC is a flexible, exploratory method for determining the number of clusters. Its framework can be adapted to use any clustering algorithms or dimension reductions preferred by the user. This flexibility allows for scalability, given that the computation time of our method is dependent only upon the computation time of the algorithms used. The drop tolerance, τ can be changed to reflect the confidence the user has with their chosen clustering algorithms based upon the level of noise in the data. The range of values specified for $\tilde{k}$ and the level of dimension reduction (if any) can also changed for the purposes of investigation.

References

[1] 20 Newsgroups Dataset. http://qwone.com/ jason/20Newsgroups/

[2] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[3] Michael W. Berry, editor. *Computational Informational Retrieval*. SIAM, 2001.

[4] Daniel Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, December 1998.

[5] Miguel Á. Carreira-Perpiñán. Fast nonparametric clustering with Gaussian blurring mean shift. *Proceedings of International Conference of Machine Learning,* 2006

[6] W. W. Cohen F. Lin. Power iteration clustering. *Proceedings of the 27th International Conference of Machine Learning*, 2010.

[7] J. Laurie Snell John G. Kemeny. *Finite Markov Chains*. Springer, 1976.

[8] I.T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistices. Springer, 2nd edition, 2002.

[9] Jacob Kogan. *Introduction to Clustering Large and High-Dimensional Data*. Cambridge University Press, Cambridge, New York, 2007.

[10] N. Glance L. Adamic. The political blogoshpere and the 2004 u.s. election: Divided they blog. In *Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem*, 2005.

[11] Amy Langville, Michael W. Berry, Murray Browne, V. Paul Pauca, and Robert J. Plemmons. Algorithms and applications for the approximate nonnegative matrix factorization. *Computational Statistics and Data Analysis*, 2007.

[12] Amy Langville, Carl D. Meyer, Russell Albright, James Cox, and David Duling. Initializations, algorithms, and convergences for the nonnegative matrix factorization. Preprint.

[13] Rasmus Munk Larsen. Lanczos bidiagonalization with partial reorthogonalization, 1998.

[14] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–91, October 1999.

[15] Franklina de Toledo, Maria Nascimento and Andre Carvalho. Consensus clustering using spectral theory. *Advances in Neuro-Information Processing*, 461-468, 2009

[16] Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. 2001.

[17] C. D. Meyer and C. D. Wessell. Stochastic Data Clustering. *ArXiv e-prints*, August 2010.

[18] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2nd edition, 2001.

[19] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm, 2001.

[20] A. Fischer Ch. Schutte P. Deuflhard, W. Huisinga. Identification of almost invariant aggregates in reversible nearly uncoupled markov chains. *Linear Algebra and its Applications*, 315:39–59, 2000.

[21] Vipin Kumar Pang-Ning Tan, Michael Steinbach. *Introduction to Data Mining*. Pearson, 2006.

[22] Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52:91–118, 2003.

[23] Shaina Race. Clustering via dimension-reduction and algorithm aggregation. Master's thesis, North Carolina State University, 2008.

[24] Shaina Race. *Iterated Consensus Clustering*. PhD thesis, North Carolina State University, 2013 (Pending).

[25] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

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

[27] William J. Stewart. *Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Modeling*. Princeton University Press, 2009.

[28] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:4, 2007.

[29] Chuck Wessell. *Stochastic Data Clustering*. PhD thesis, North Carolina State University, 2011.