Efficient Eigen-updating for Spectral Graph Clustering

Charanpal Dhanjal[a], Romaric Gaudel[b], Stéphan Clémençon[c]

[a]*LIP6, UPMC, 4 Place Jussieu, 75252 Paris Cedex 05, France*
[b]*Université Lille, Domaine Universitaire du Pont de Bois, 59653 Villeneuve d'Ascq Cedex, France*
[c]*Telecom ParisTech, 46 rue Barrault, 75634 Paris Cedex 13, France*

Abstract

Partitioning a graph into groups of vertices such that those within each group are more densely connected than vertices assigned to different groups, known as *graph clustering*, is often used to gain insight into the organisation of large scale networks and for visualisation purposes. Whereas a large number of dedicated techniques have been recently proposed for static graphs, the design of *online* graph clustering methods tailored for evolving networks is a challenging problem, and much less documented in the literature. Motivated by the broad variety of applications concerned, ranging from the study of biological networks to the analysis of networks of scientific references through the exploration of communications networks such as the World Wide Web, it is the main purpose of this paper to introduce a novel, computationally efficient, approach to graph clustering in the evolutionary context. Namely, the method promoted in this article can be viewed as an incremental eigenvalue solution for the spectral clustering method described by Ng. *et al.* (2001). The incremental eigenvalue solution is a general technique for finding the approximate eigenvectors of a symmetric matrix given a change. As well as outlining the approach in detail, we present a theoretical bound on the quality of the approximate eigenvectors using perturbation theory. We then derive a novel spectral clustering algorithm called *Incremental Approximate Spectral Clustering* (IASC). The IASC algorithm is simple to implement and its efficacy is demonstrated on both synthetic and real datasets modelling the evolution of a HIV epidemic, a citation network and the purchase history graph of an e-commerce website.

Keywords: spectral graph clustering, eigen-decomposition, unsupervised learning, normalized Laplacian

*Corresponding author. Tel: +33 1 44 27 43 32
Email address: `charanpal.dhanjal@lip6.fr` (Charanpal Dhanjal)

1. Introduction

Graph-mining has recently received increasing attention in the machine-learning literature, motivated by application domains such as the Internet, social networks, epidemics of transmissible infectious diseases, sensor and biological networks. A crucial task in exploratory analysis and data visualization is *graph clustering* [31, 10], which aims to partition the vertices in a graph into groups or clusters, with dense internal connections and few connections between each other. There is a large body of work on graph clustering. A possible approach is to consider a certain measure that quantifies community structure and formulate the clustering issue as an optimisation problem (which is generally NP-hard), for which fairly good solutions can be obtained recursively or by using adequate metaheuristics, see [23, 21, 9, 30] for example. A variety of approaches to graph clustering exist, such as those based on modularity maximisation for instance, see [23, 22]. In this paper, focus is on the *spectral clustering* approach [36], which performs well empirically, is often simple to implement and benefits computationally from the availability of fast linear algebra libraries. The general principle of spectral clustering is to compute the smallest eigenvectors of some particular matrix $\mathbf{L}$ (refer to Section 2 for further details) and then cluster the vertices based on their representation in the corresponding eigen-space. The popularity of this approach arises from the fact that the obtained clustering of vertices is closely connected to the spectral relaxation of the minimization of the normalised cut criterion, see [33].

In many applications such as communications networks (e.g. the Web and Internet), biological networks (of proteins, metabolic reactions, *etc.*), social networks or networks of scientific citations for instance, the graphs of interest slowly change with time. A naive approach to this incremental problem is to cluster each graph in the sequence separately, however this is computationally expensive for spectral clustering as the cost of solving the eigenvalue problem is $\mathcal{O}(n^3)$ at each iteration, where n is the number of vertices. There has been some previous work on the incremental spectral clustering problem, for example [35, 26, 25, 15] however only [26, 25, 15] update the eigen-system. In this paper we propose an efficient method for clustering a sequence of graphs which leverages the eigen-decomposition and the clustering on the previous graph to find the new clustering. Firstly, a fast approximation of a rank-k eigen-decomposition of $\mathbf{L}_{t+1}$ knowing that of $\mathbf{L}_t$ is derived from the Singular Value Decomposition (SVD) updating approach used for Latent Semantic Indexing in [40]. Here, the update is efficient to compute when the change (defined in the subsequent analysis) between $\mathbf{L}_t$ and $\mathbf{L}_{t+1}$ is small. Secondly the clustering of vertices is updated accordingly to the new eigen-space. The efficiency of the complete approach, in terms of clustering accuracy, is demonstrated on synthetic and real data. We point out that the clustering approach was first outlined in [5]. Here we provide a theoretical analysis on the quality of the eigen-approximation, as well as a more extensive empirical study of the algorithm.

The paper is organised as follows. Standard spectral clustering and SVD updating approaches are recalled in Sections 2 and 3. Then Section 4 details

the proposed eigen-decomposition update and Section 5 studies the accuracy of the resulting approximate eigenvectors using perturbation theory. In Section 6 we show how the eigen-decomposition updates can be applied to spectral clustering. Numerical results are gathered in Section 7, and the paper ends with Section 8 discussing results and planned future work.

Notation: A bold uppercase letter represents a matrix, e.g. $\mathbf{X}$, and a column vector is displayed using a bold lowercase letter e.g. $\mathbf{x}$. The transpose of a matrix or vector is written $\mathbf{X}^T$. The concatenation of the columns of two matrices $\mathbf{A}$ and $\mathbf{B}$ is written $[\mathbf{A} \quad \mathbf{B}]$. $\mathbf{A}[I, I]$ represent the submatrix of $\mathbf{A}$ formed using the row and columns indexed by I, and $\mathbf{A}[I, :]$ and $\mathbf{A}[:, I]$ are submatrices formed using the rows and columns I respectively. The matrix $\mathbf{A}_k$ is that formed using the largest k eigenvectors and eigenvalues of $\mathbf{A}$, however $\mathbf{I}_p$ is the $p \times p$ identity matrix.

2. Graph Clustering

Consider an undirected graph $G = (V, E)$, composed of a set of vertices $V = \{v_1, \ldots, v_n\}$ and edges $E \subseteq V \times V$ such that for every edge (v_i, v_j) there is also an edge (v_j, v_i). One way of representing the edges of G is using an *adjacency matrix* $\mathbf{A} \in \{0, 1\}^{n \times n}$ which has $\mathbf{A}_{ij} = 1$ if there is an edge from the i-th to the j-th vertex and $\mathbf{A}_{ij} = 0$ otherwise. More generally, the *weight matrix* $\mathbf{W} \in \mathbb{R}^{n \times n}$ allows one to assign nonzero numerical values on edges, and thus $\mathbf{W}_{ij} \neq 0$ when there is an edge from the i-th to the j-th vertex.

In the perspective of spectral clustering, a useful way of representing G is through its Laplacian matrix [3]. There are several definitions of the Laplacian matrix, however we are interested in the *normalised Laplacian matrix*, which is symmetric and positive semi-definite. We recall its definition below for clarity.

Definition 2.1. *The* unnormalised Laplacian matrix *of a graph G is defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ where $\boldsymbol{D}$ is the degree matrix with supposedly nonzero diagonal entries $\boldsymbol{D}_{ii} = deg(v_i)$, denoting by $deg(v_i) = \sum_j \boldsymbol{W}_{ij}$ the degree of the i-th vertex, and zeros elsewhere, and $\boldsymbol{W}$ is the weight matrix. The normalised Laplacian matrix of a graph G is then defined as*

$$\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}}.$$

The normalised Laplacian matrix is used in the spectral clustering approach of Ng et al. [24]. The algorithm computes the Laplacian and then finds the k smallest eigenvectors which are used for clustering in conjunction with the k-means algorithm, see Algorithm 1. The Laplacian matrix is often sparse and one can use power or Krylov subspace methods such as the Lanczos method to find the eigenvectors. There are several variants of Algorithm 1, such as that of [33] which uses the so-called random walk Laplacian and clusters the smallest eigenvectors in a similar way. One of the motivations for these clustering methods is from the spectral relaxation of the minimisation of the normalised cut criterion.

Algorithm 1 Spectral Clustering using the Normalised Laplacian [24]

Require: Graph G with weight matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$, number of clusters k
  1: Find k smallest eigenvectors $\mathbf{V}_k = [\mathbf{v}_1, \ldots, \mathbf{v}_k]$ of normalised Laplacian $\tilde{\mathbf{L}}$
  2: Normalise the rows of $\mathbf{V}_k$, i.e. $\mathbf{V}_k \leftarrow \text{diag}(\mathbf{V}_k \mathbf{V}_k^T)^{-\frac{1}{2}} \mathbf{V}_k$
  3: Cluster rows of $\mathbf{V}_k$ with the k-means algorithm
  4: **return** Cluster membership vector $\mathbf{c} \in \{1, \ldots, k\}^n$

A naive approach to spectral clustering on a sequence of graphs has a large update cost due to the computation of the eigen-decomposition of $\tilde{\mathbf{L}}_t$ at each iteration t. A more efficient approach is to update relevant eigenvectors from one iteration to the next. In [35] the authors use the spectral clustering of [24] however they do not update the eigenvectors incrementally but instead the clustering directly. The iterative clustering approach of *Ning et al.* [26, 25] uses the spectral clustering method given in [33] and updates eigenvectors incrementally. The algorithm incrementally updates the solution of the generalised eigenproblem $\mathbf{L}\mathbf{v} = \lambda \mathbf{D}\mathbf{v}$ by finding the derivatives on the eigenvalues/vectors with perturbations in all of the quantities involved. An iterative refinement algorithm is given for the eigenvalues and eigenvectors given a change in the edges or vertices of a graph. One then clusters the resulting k smallest eigenvectors using k-means clustering. In order to limit errors which can build up cumulatively the authors recompute the eigenvectors after every R-th graph in the sequence. A modification of the algorithm is proposed in [15] which improves efficiency by modelling clusters using a set of representative points (a similar strategy is used in [39] for example to cluster points in $\mathbb{R}^d$).

A disadvantage of the approach of [26, 25] lies in the fact that, to update an eigenvector, one must invert a small matrix for each weight change in the graph which makes updates costly. The size of this matrix is proportional to the number of neighbours of the vertices incident to the changed edge. In contrast, the approach presented in our paper has a smaller update cost for a set of vertex or edge weight changes between $\mathbf{L}_t$ and $\mathbf{L}_{t+1}$, since changes are considered in a batch fashion, as will later become clear. A further problem with the *Ning et al.* approach is that eigenvectors are updated independently of one another and hence one loses the orthogonality $\mathbf{v}_i^T \mathbf{D}\mathbf{v}_j = \delta(i,j)$, where δ is the Kronecker delta function (taking the value 1 if $i = j$ and 0 otherwise), and vectors can become correlated for example.

Another way of improving the efficiency of spectral clustering is to compute approximate eigen-decompositions at each iteration, for example by using the Nyström approach [38]. The Nyström method is used for spectral graph clustering in [11] in conjunction with image segmentation. To estimate the eigenvalues and eigenvectors of $\mathbf{A} \in \mathbb{R}^{n \times n}$, one first finds a matrix $\mathbf{A}[I, I]$ in which $I \in \{1, \ldots n\}^m$ is a set of indices selected uniformly at random. If we assume that $\mathbf{A}[I, I]$ is positive definite, we can take its square root $\mathbf{A}[I, I]^{1/2}$. One then defines

$$\mathbf{S} = \mathbf{A}[I, I] + \mathbf{A}[I, I]^{-1/2} \mathbf{A}[I, \bar{I}] \mathbf{A}[I, \bar{I}]^T \mathbf{A}[I, I]^{-1/2},$$

4

where $\bar{I}$ is the complement of I, and diagonalises $\mathbf{S}$ using the eigen-decomposition $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$. Let $\mathbf{V} = \mathbf{A}[:, I]\mathbf{A}[I, I]^{-1/2}\mathbf{U}\boldsymbol{\Lambda}^{-1/2}$, then it can be shown that the Nyström approximation of $\mathbf{A}$, given by $\tilde{\mathbf{A}}_I = \mathbf{A}[:, I]\mathbf{A}[I, I]^{-1}\mathbf{A}[I, :]$, is equivalent to $\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$. If $\mathbf{A}$ is indefinite a more complicated two-step procedure is required, see [11] for details. The resulting approximation is applied to find the first few eigenvectors of the normalised Laplacian matrix at a total cost of $\mathcal{O}(nm^2 + m^3)$. Several efficiency improvements based on this approach are proposed in [19] which finds the largest k approximate eigenvectors of $\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$ at a reduced time and space complexity.

The quality of the resulting approximation is determined by the extent the submatrix $\mathbf{A}[\bar{I}, \bar{I}]$ is spanned by $\mathbf{A}[I, :]$. One would naturally expect a good approximation for example when $\mathbf{A}[I, :]$ spans the space of $\mathbf{A}$. The choice of sampling of I also effects approximation quality, and empirical and theoretical work in [16] suggests random sampling without replacement over the non-uniform sampling in [7, 6]. Hence for our later empirical work, we will use uniform random sampling without replacement. In addition to [16], error bounds for the Nyström method are presented in [7, 20] in terms of the matrix approximation error using the Frobenius or spectral norm (given by $\|\mathbf{A}\|_F = \sqrt{\mathrm{tr}(\mathbf{A}^T\mathbf{A})}$ and $\|\mathbf{A}\|_2 = \sqrt{\lambda_{max}(\mathbf{A}^T\mathbf{A})}$ respectively). One of the disadvantages of applying the Nyström based approaches of [11, 19] on graphs is that by sampling only a subset of columns of the normalised Laplacian one can exclude important edges which help define clusters. These approaches are naturally more effective when a set of points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^d$ is mapped into a weighted graph using for example the Gaussian weighted distance $\mathbf{W}_{i,j} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2)/2\sigma^2$, for $\sigma \in \mathbb{R}^+$, which is very informative about the relative positions of the points.

3. SVD Updating

An important aspect of the incremental clustering method described later lies in its ability to efficiently compute the eigenvectors of a matrix from the eigenvectors of a submatrix, and for clarity's sake, we outline the SVD-updating algorithm of [40]. The SVD of $\mathbf{A} \in \mathbb{R}^{m \times n}$ is the decomposition

$$\mathbf{A} = \mathbf{P}\boldsymbol{\Sigma}\mathbf{Q}^T,$$

where $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_m]$, $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_n]$ and $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$ are respectively the orthogonal matrices of left and right singular vectors and a diagonal matrix of singular values $\sigma_1 \geq \sigma_2 \geq, \ldots, \geq \sigma_r$, with $r = \min(m, n)$.

In [40], the authors use the SVD of $\mathbf{A}$ to approximate the SVD of $[\mathbf{A} \quad \mathbf{B}]$, with $\mathbf{B} \in \mathbb{R}^{m \times p}$, without recomputing the SVD of the new matrix. It is known that the best k-rank approximation of $\mathbf{A}$, using the Frobenius norm error, is given by its SVD, namely

$$\mathbf{A}_k = \mathbf{P}_k \boldsymbol{\Sigma}_k \mathbf{Q}_k^T,$$

where $\mathbf{P}_k$, $\mathbf{Q}_k$, $\mathbf{\Sigma}_k$ correspond to the k largest singular values. The general idea of the algorithm is to write from $\mathbf{A}_k$'s SVD as

$$[\mathbf{A}_k \quad \mathbf{B}] = \mathbf{S}\mathbf{\Lambda}\mathbf{R}^T,$$

in which $\mathbf{S}$ and $\mathbf{R}$ are matrices with orthonormal columns. One then takes the rank-k SVD $\mathbf{\Lambda} = \mathbf{G}_k\mathbf{\Gamma}_k\mathbf{H}_k^T$ and $(\mathbf{S}\mathbf{G}_k)\mathbf{\Gamma}_k(\mathbf{R}\mathbf{H}_k)^T$ is the rank-k approximation for $[\mathbf{A} \quad \mathbf{B}]$. The dimensionality of $\mathbf{\Lambda}$ is generally much smaller than that of $[\mathbf{A}_k \quad \mathbf{B}]$ and hence the corresponding SVD approximation is computationally inexpensive.

This approach is more accurate that those presented in [2, 27], however it comes at additional computational cost. Furthermore, the authors prove that when the matrix $[\mathbf{A} \quad \mathbf{B}]^T[\mathbf{A} \quad \mathbf{B}]$ has the form $\mathbf{X} + \alpha^2\mathbf{I}$ in which $\mathbf{X}$ is symmetric positive semi-definite with rank-k then the rank-k approximation of $[\mathbf{A} \quad \mathbf{B}]$ is identical to that of $[\mathbf{A}_k \quad \mathbf{B}]$.

4. Incremental Eigen-approximation

In this section we address three types of updating problem upon the largest k eigenvectors of a symmetric matrix. The updates are general operations, however they will be explained in the context of spectral clustering later in Section 6. Assuming that $\mathbf{Y}_1, \mathbf{Y}_2 \in \mathbb{R}^{n \times p}$ and one does not have direct access to $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{m \times p}$ but only to the matrices $\mathbf{C} = \mathbf{A}^T\mathbf{A}$, $\mathbf{A}^T\mathbf{B}$ and $\mathbf{B}^T\mathbf{B}$, the updates are:

1. Addition of a low-rank symmetric matrix $\mathbf{C} \to \mathbf{C} + \mathbf{U}$ where $\mathbf{U} = \mathbf{Y}_1\mathbf{Y}_2^T + \mathbf{Y}_2\mathbf{Y}_1^T$
2. Addition of rows and columns $\mathbf{A}^T\mathbf{A} \to [\mathbf{A} \quad \mathbf{B}]^T[\mathbf{A} \quad \mathbf{B}]$
3. Removal of rows and columns $[\mathbf{A} \quad \mathbf{B}]^T[\mathbf{A} \quad \mathbf{B}] \to \mathbf{A}^T\mathbf{A}$

We have written the updates above in terms of a symmetric matrix $\mathbf{A}^T\mathbf{A}$ to improve notation. Note that any positive semi-definite symmetric matrix $\mathbf{M}$ can be decomposed into the form $\mathbf{M} = \mathbf{A}^T\mathbf{A}$, where $\mathbf{A}$ has real entries, for example by using an eigen-decomposition or incomplete Cholesky factorisation.

4.1. Addition of a Low-rank Symmetric Matrix

The first type of eigen-approximation we are interested in is the addition of a low-rank symmetric matrix. One computes the eigen-decomposition of $\mathbf{C}$ and then approximates the rank-k decomposition of $\mathbf{C}_k + \mathbf{U}$ where $\mathbf{U} = \mathbf{Y}_1\mathbf{Y}_2^T + \mathbf{Y}_2\mathbf{Y}_1^T$ and $\mathbf{C}_k$ is the approximation of $\mathbf{C}$ using the k largest eigenvectors (also known as the best k-rank approximation of $\mathbf{C}$). A similar but not applicable update is considered for the SVD case in [40] in which the rank-k approximation of $\mathbf{A}_k + \hat{\mathbf{Y}}_1\hat{\mathbf{Y}}_2^T$ is found from $\mathbf{A}_k$ in which $\hat{\mathbf{Y}}_1 \in \mathbb{R}^{m \times j}$ and $\hat{\mathbf{Y}}_2 \in \mathbb{R}^{n \times j}$ for some j. The general idea in our case is to find a matrix with orthonormal columns $\tilde{\mathbf{Q}}$ such that $\mathbf{C}_k + \mathbf{U} = \tilde{\mathbf{Q}}\mathbf{\Delta}\tilde{\mathbf{Q}}^T$ for a square matrix $\mathbf{\Delta}$. To this purpose, we first project the columns of $\mathbf{Y}_1$ into the space orthogonal to the k largest eigenvectors $\mathbf{Q}_k$ of

C (note the deviation from standard notation), a process known as *deflation*. Assuming that eigenvectors have unit norm, the matrix $\mathbf{Y}_1$ is thus deflated as follows:

$$\bar{\mathbf{Y}}_1 = (\mathbf{I} - \mathbf{Q}_k \mathbf{Q}_k^T)\mathbf{Y}_1,$$

at a cost of $\mathcal{O}(npk)$. Note that $\bar{\mathbf{Y}}_1 \mathbf{\Theta}_1$ for some $\mathbf{\Theta}_1$, is orthogonal to $\mathbf{Q}_k$ since $\mathbf{Q}_k^T \bar{\mathbf{Y}}_1 \mathbf{\Theta}_1 = (\mathbf{Q}_k^T \mathbf{Y}_1 - \mathbf{Q}_k^T \mathbf{Y}_1)\mathbf{\Theta}_1 = \mathbf{0}$. If we take the SVD $\bar{\mathbf{Y}}_1 = \bar{\mathbf{P}}_1 \bar{\mathbf{\Sigma}}_1 \bar{\mathbf{Q}}_1^T$ then $\bar{\mathbf{P}}_1$ is orthogonal to $\mathbf{Q}_k$ since $\bar{\mathbf{P}}_1 = \bar{\mathbf{Y}}_1 \bar{\mathbf{Q}}_1 \bar{\mathbf{\Sigma}}_1^{-T}$ assuming $\bar{\mathbf{\Sigma}}_1$ has nonzero diagonal entries.

At the next stage we would like to orthogonalise the columns of $\mathbf{Y}_2$ with respect to both $\mathbf{Q}_k$ and $\bar{\mathbf{P}}_1$. Hence we deflate $\mathbf{Y}_2$ in the following way:

$$\bar{\mathbf{Y}}_2 = (\mathbf{I} - \bar{\mathbf{P}}_1 \bar{\mathbf{P}}_1^T - \mathbf{Q}_k \mathbf{Q}_k^T)\mathbf{Y}_2,$$

at cost $\mathcal{O}(npk)$, where we have used the fact that $\bar{\mathbf{P}}_1$ is orthogonal to $\mathbf{Q}_k$. Proved in a similar way to the step used earlier, the matrix in the column space of $\bar{\mathbf{Y}}_2$, $\bar{\mathbf{Y}}_2 \mathbf{\Theta}_2$ for some $\mathbf{\Theta}_2$, is orthogonal to $\mathbf{Q}_k$ and $\bar{\mathbf{P}}_1$. Hence, we compute the SVD $\bar{\mathbf{Y}}_2 = \bar{\mathbf{P}}_2 \bar{\mathbf{\Sigma}}_2 \bar{\mathbf{Q}}_2^T$ and note that the matrices $\bar{\mathbf{P}}_1$, $\bar{\mathbf{P}}_2$ and $\mathbf{Q}_k$ are mutually orthogonal and span the space spanned by $\mathbf{C}_k + \mathbf{U}$. This allows one to write

$$\mathbf{C}_k + \mathbf{U} = \tilde{\mathbf{Q}}\mathbf{\Delta}\tilde{\mathbf{Q}}^T$$

as required with $\tilde{\mathbf{Q}} = [\mathbf{Q}_k \ \bar{\mathbf{P}}_1 \ \bar{\mathbf{P}}_2] \in \mathbb{R}^{n \times (k+2p)}$ and $\mathbf{\Delta} = \tilde{\mathbf{Q}}^T(\mathbf{C}_k + \mathbf{U})\tilde{\mathbf{Q}}$, or equivalently

$$\mathbf{\Delta} = \begin{bmatrix} \mathbf{\Omega}_k + \mathbf{Q}_k^T \mathbf{U} \mathbf{Q}_k & \mathbf{Q}_k^T \mathbf{U} \bar{\mathbf{P}}_1 & \mathbf{Q}_k^T \mathbf{Y}_1 \bar{\mathbf{Q}}_2 \bar{\mathbf{\Sigma}}_2 \\ \bar{\mathbf{P}}_1^T \mathbf{U} \mathbf{Q}_k & \bar{\mathbf{P}}_1^T \mathbf{U} \bar{\mathbf{P}}_1 & \bar{\mathbf{\Sigma}}_1 \bar{\mathbf{Q}}_1^T \bar{\mathbf{Q}}_2 \bar{\mathbf{\Sigma}}_2 \\ \bar{\mathbf{\Sigma}}_2 \bar{\mathbf{Q}}_2^T \mathbf{Y}_1^T \mathbf{Q}_k & \bar{\mathbf{\Sigma}}_2 \bar{\mathbf{Q}}_2^T \bar{\mathbf{Q}}_1 \bar{\mathbf{\Sigma}}_1 & \mathbf{0} \end{bmatrix},$$

in which $\mathbf{\Delta} \in \mathbb{R}^{(k+2p)\times(k+2p)}$. We take the rank-$k$ eigen-decomposition $\mathbf{\Delta}_k = \mathbf{H}_k \mathbf{\Pi}_k \mathbf{H}_k^T$ and then the final eigen-approximation is given by $(\tilde{\mathbf{Q}}\mathbf{H}_k)\mathbf{\Pi}_k(\tilde{\mathbf{Q}}\mathbf{H}_k)^T$ in which it is easy to verify that the columns of $\tilde{\mathbf{Q}}\mathbf{H}_k$ are orthonormal.

The deflation and SVD of $\bar{\mathbf{Y}}_1$ and $\bar{\mathbf{Y}}_2$ cost $\mathcal{O}(npk)$ and $\mathcal{O}(np^2)$ respectively and the eigen-decomposition of $\mathbf{\Delta}$ is $\mathcal{O}((k+2p)^3)$. In order to compute $\mathbf{\Delta}$ one can reuse the computations of $\mathbf{Q}_k^T \mathbf{Y}_1$, $\mathbf{Q}_k^T \mathbf{Y}_2$, $\bar{\mathbf{P}}_1^T \mathbf{Y}_2$ which are used for deflations and also the matrices used for the SVD decompositions. Thus $\mathbf{\Delta}$ is found in $\mathcal{O}(p^3 + p^2k + pk^2)$, and the overall complexity of this algorithm is $\mathcal{O}((k^2 + p^2)(p+k) + np(p+k))$. Of note here is that n scales the complexity in a linear fashion, however costs are cubically related to k and p.

4.2. Addition of Rows and Columns

In correspondence with the SVD-updating method given above we consider the case in which one has the eigen-decomposition of **C** and then wants to find the rank-k approximation of $\mathbf{E} = [\mathbf{A} \ \ \mathbf{B}]^T[\mathbf{A} \ \ \mathbf{B}]$. Such a process is useful not just in incremental clustering but also in incrementally solving kernel Principal

Components Analysis (KPCA, [32]) for example. This update can be written in terms of that described above. This is seen by writing the former update $\mathbf{A}^T\mathbf{A} \to [\mathbf{A} \quad \mathbf{B}]^T[\mathbf{A} \quad \mathbf{B}]$ in terms of the latter:

$$\begin{bmatrix} \mathbf{A}^T\mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \mathbf{A}^T\mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{A}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{bmatrix}.$$

The second term on the right hand side can be written as $\mathbf{Y}_1\mathbf{Y}_2^T + \mathbf{Y}_2\mathbf{Y}_1^T$ where $\mathbf{Y}_1 = [\mathbf{0} \quad \mathbf{I}_p]^T$ and $\mathbf{Y}_2 = [\mathbf{B}^T\mathbf{A} \quad \frac{1}{2}\mathbf{B}^T\mathbf{B}]^T$. The eigenvectors of the first matrix on the right-hand side are found from those of $\mathbf{A}^T\mathbf{A}$ by simply adding p zero rows to the existing eigenvectors, and the corresponding eigenvalues are identical. Additional eigenvectors are standard unit vectors spanning the p new rows with corresponding eigenvalues as zero. A useful insight is that the deflated matrix $\bar{\mathbf{Y}}_1 = \mathbf{Y}_1$ and hence its SVD decomposition can be written directly as $\bar{\mathbf{Y}}_1 = [\mathbf{0} \quad \mathbf{I}_p]^T\mathbf{I}_p\mathbf{I}_p$.

4.2.1. Alternative Approach

Here we outline a simpler and more direct approach for the addition of rows and columns to a matrix. First let $\mathbf{C} = \mathbf{Q}\boldsymbol{\Omega}\mathbf{Q}^T$ in which $\mathbf{Q}$ is a matrix of eigenvectors and $\boldsymbol{\Omega}$ is a diagonal matrix of eigenvalues. Note that $\hat{\mathbf{E}} = [\mathbf{A}_k \quad \mathbf{B}]^T[\mathbf{A}_k \quad \mathbf{B}]$ can be written as $\tilde{\mathbf{Q}}\boldsymbol{\Delta}\tilde{\mathbf{Q}}^T$ for a square matrix $\boldsymbol{\Delta}$. In our case we have

$$\tilde{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_p \end{bmatrix} \text{ and } \boldsymbol{\Delta} = \begin{bmatrix} \boldsymbol{\Omega}_k & \mathbf{Q}_k^T\mathbf{A}_k^T\mathbf{B} \\ \mathbf{B}^T\mathbf{A}_k\mathbf{Q}_k & \mathbf{B}^T\mathbf{B} \end{bmatrix},$$

noting that $\mathbf{A}_k\mathbf{Q}_k\mathbf{Q}_k^T = \mathbf{Q}_k\boldsymbol{\Omega}_k\mathbf{Q}_k^T\mathbf{Q}_k\mathbf{Q}_k^T = \mathbf{A}_k$ since $\mathbf{Q}_k^T\mathbf{Q}_k = \mathbf{I}$. Furthermore, note that $\mathbf{Q}_k^T\mathbf{A}^T\mathbf{B} = \mathbf{Q}_k^T\mathbf{Q}\boldsymbol{\Sigma}\mathbf{P}^T\mathbf{B} = \mathbf{Q}_k^T\mathbf{Q}_k\boldsymbol{\Sigma}_k\mathbf{P}_k^T\mathbf{B} = \mathbf{Q}_k^T\mathbf{A}_k^T\mathbf{B}$ and calculating $\mathbf{Q}_k^T\mathbf{A}_k^T\mathbf{B}$ is $\mathcal{O}(npk)$. It follows that $\boldsymbol{\Delta} \in \mathbb{R}^{(k+p)\times(k+p)}$ can be found using $\mathbf{Q}_k$, $\mathbf{A}^T\mathbf{A}$, and $\mathbf{A}^T\mathbf{B}$ and $\mathbf{B}^T\mathbf{B}$. In the final step, and analogously to the SVD case we take the rank-k eigen-decomposition $\boldsymbol{\Delta} = \mathbf{H}_k\boldsymbol{\Pi}_k\mathbf{H}_k^T$ at a cost of $\mathcal{O}((k+p)^3)$ and then the rank-k eigen approximation of $\hat{\mathbf{E}}$ is given by $(\tilde{\mathbf{Q}}\mathbf{H}_k)\boldsymbol{\Pi}_k(\tilde{\mathbf{Q}}\mathbf{H}_k)^T$.

A similar eigen-update is considered in [17] and used in conjunction with PCA [14] and KPCA in [17]. In this method the authors phrase the problem as a series of SVD updates, and we use a more direct approach. Furthermore, the eigen-approximation above is identical to that at the start of Section 4.2 with $\mathbf{Y}_1 = [\mathbf{0} \quad \mathbf{I}_p]^T$ and $\mathbf{Y}_2 = [\mathbf{B}^T\mathbf{A}_k \quad \frac{1}{2}\mathbf{B}^T\mathbf{B}]^T$. The difference between the method above and that of Section 4.2 is the former uses $\mathbf{A}_k$ as opposed to $\mathbf{A}$ in $\mathbf{Y}_2$ which results in a greater error.

4.3. Removing Rows and Columns

Observe that removing rows and columns is equivalent to zeroing the corresponding rows/columns:

$$\hat{\mathbf{C}} = \left[\begin{array}{cc} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{array} \right] \rightarrow \left[\begin{array}{cc} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{array} \right] - \left[\begin{array}{cc} \mathbf{0} & \mathbf{A}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{array} \right],$$

and in this form one can see the connection to Section 4.2. Again, one can write the second term on the right hand side as $\mathbf{Y}_1\mathbf{Y}_2^T + \mathbf{Y}_2\mathbf{Y}_1^T$ where $\mathbf{Y}_1 = [\mathbf{0} \quad \mathbf{I}_p]^T$ and $\mathbf{Y}_2 = -[\hat{\mathbf{B}}^T\hat{\mathbf{A}} \quad \frac{1}{2}\hat{\mathbf{B}}^T\hat{\mathbf{B}}]^T$ where $\hat{\mathbf{B}}^T\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}^T\hat{\mathbf{B}}$ are found using the rank-k approximation of $\hat{\mathbf{C}}$. Since we are updating the rank-k approximation of $\hat{\mathbf{C}}$, the final eigen-approximation will have zero elements in the eigenvectors at rows corresponding to those row/columns that are deleted.

4.4. Interpretation

A common theme in the updates outlined above is that one can write them all in terms of the addition of a low-rank symmetric matrix $\mathbf{U}$ to a positive semi-definite matrix $\mathbf{C}$. Our approximation method computes $\mathbf{C}_k + \mathbf{U}$ via its expression as $\tilde{\mathbf{Q}}\mathbf{H}_k\mathbf{\Pi}_k\mathbf{H}_k^T\tilde{\mathbf{Q}}^T$ where $\tilde{\mathbf{Q}}$ is a matrix with orthogonal columns spanning $\mathbf{C}_k + \mathbf{U}$ and $\mathbf{H}_k$ and $\mathbf{\Pi}_k$ represent the largest k eigenvector and eigenvalues of a matrix $\mathbf{\Delta} = \tilde{\mathbf{Q}}^T(\mathbf{C}_k + \mathbf{U})\tilde{\mathbf{Q}}$. The following lemma shows the consequence of this approach.

Lemma 4.1. *Decompose a matrix $\mathbf{Z} = \mathbf{F}\mathbf{\Lambda}\mathbf{F}^T$ where $\mathbf{F}$ is* any *matrix with orthonormal columns $\mathbf{F}^T\mathbf{F} = \mathbf{I}$. For some k find the best rank-k eigen-approximation $\mathbf{\Lambda}_k = \mathbf{H}_k\mathbf{\Pi}_k\mathbf{H}_k^T$ in which $\mathbf{H}_k$ and $\mathbf{\Pi}_k$ are the largest k eigenvectors and eigenvalues of $\mathbf{\Lambda}$, and let $\hat{\mathbf{Z}} = \mathbf{F}\mathbf{H}_k\mathbf{\Pi}_k\mathbf{H}_k^T\mathbf{F}^T$. Then the best rank-$k$ eigen-decompostion of $\mathbf{Z}$ is given by:*
$$\mathbf{Z}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{U}_k^T = \hat{\mathbf{Z}}.$$

Proof. Note that $\mathbf{\Lambda} = \mathbf{F}^T\mathbf{Z}\mathbf{F}$ due to the orthogonality of $\mathbf{F}$. Let $\mathbf{u}$, s be the eigenvectors and eigenvalues of $\mathbf{Z}$, and define $\mathbf{u} = \mathbf{F}\mathbf{v}$ for some $\mathbf{v}$ then $\mathbf{F}^T\mathbf{Z}\mathbf{F}\mathbf{v} = s\mathbf{v}$. This implies that the eigenvalues of $\mathbf{\Lambda}$ are the same as those of $\mathbf{Z}$ and the eigenvectors are related by $\mathbf{U} = \mathbf{F}\mathbf{V}$. Hence we have $\mathbf{V} = \mathbf{H}$ and $\mathbf{S} = \mathbf{\Pi}$ which implies $\hat{\mathbf{Z}} = \mathbf{F}\mathbf{H}_k\mathbf{\Pi}_k\mathbf{H}_k^T\mathbf{F}^T = \mathbf{U}_k\mathbf{S}_k\mathbf{U}_k^T = \mathbf{Z}_k$ as required. The only condition on $\mathbf{F}$ is that the eigenvectors of $\mathbf{Z}$ are in the column space of $\mathbf{F}$. This must be the case however since $\mathbf{Z} = \mathbf{U}\mathbf{S}\mathbf{U}^T = \mathbf{F}\mathbf{V}\mathbf{S}\mathbf{V}^T\mathbf{F}^T$. $\square$

Hence, the update $\mathbf{C}_k + \mathbf{U}$ of Section 4.1 is identical to the best rank-k approximation of $\mathbf{C}_k + \mathbf{U}$.

5. Eigen-approximation Quality

This section aims to bound the quality of the proposed eigen-approximation approach. As mentioned in Section 4.4, the proposed approach replaces the expected best rank-k approximation of a matrix $\mathbf{A} + \mathbf{B}$ by the best rank-k approximation of $\mathbf{A}_k + \mathbf{B}$, where $\mathbf{A}_k$ denotes the best rank-k approximation

of $\mathbf{A}$. Hence, our objective is to control the difference between $(\mathbf{A} + \mathbf{B})_k$ and $(\mathbf{A}_k + \mathbf{B})_k$.

The result in Section 4.4 gives us a first insight into the approximation error of the updates described: the residual matrix is that corresponding to the eigenvectors and eigenvalues after k. One can see that for any matrix $\mathbf{C}$

$$\|\mathbf{C} - \mathbf{C}_k\|_F^2 = \|\mathbf{C}_{k\perp}\|_F^2 = \sum_{i=k+1}^n \omega_i^2,$$

where ω_i is the ith eigenvalue of $\mathbf{C}$ (unless otherwise stated eigenvalues are always given in descending order) and $\mathbf{C}_{k\perp}$ is the approximation of $\mathbf{C}$ using eigenvectors/eigenvalues after k. This implies that $\mathbf{C}$ is well approximated by the largest k eigenvectors if the sum of the square of the remaining eigenvalues is small.

This certainly gives us insight into when our eigen-updating approach will be accurate, however the kind of matrices we will work with do not have this property in general. We now turn to matrix perturbation theory [34] in order to learn more about the approximated eigenvectors of the updated matrix. In a nutshell, it lies in controlling the *angle* between two *invariant subspaces*. Before giving the corresponding theorem, the following subsection introduces the necessary notions of invariant subspaces of a matrix and of the canonical angles between subspaces.

5.1. Invariant Subspaces and Canonical Angles

We begin by introducing the simple concept of an invariant subspace.

Definition 5.1. *The subspace $\mathcal{X}$ is an* invariant subspace *of $\boldsymbol{A}$ if $\boldsymbol{A}\mathcal{X} \subset \mathcal{X}$.*

It can also be shown that if the columns of $\mathbf{X}$ form a basis for $\mathcal{X}$ of $\mathbf{A}$ then there is a unique matrix $\mathbf{L}$ such that $\mathbf{AX} = \mathbf{XL}$. The matrix $\mathbf{L}$ is a representation of $\mathbf{A}$ with respect to the basis $\mathbf{X}$, and it has identical eigenvalues to $\mathbf{A}$. A useful decomposition in perturbation theory is to reduce $\mathbf{A}$ to a block diagonal form. Let $\mathbf{X}_1$ be an orthogonal matrix which spans the invariant subspace of $\mathbf{A}$, $\mathcal{X}_1$, and assume that we have a matrix $\mathbf{Y}_2$ such that $[\mathbf{X}_1\mathbf{Y}_2]$ is unitary and $\mathbf{Y}_2$ spans the space orthogonal to $\mathcal{X}_1$, then this allows us to write

$$[\mathbf{X}_1\mathbf{Y}_2]^T\mathbf{A}[\mathbf{X}_1\mathbf{Y}_2] = \begin{bmatrix} \mathbf{L}_1 & \mathbf{H} \\ \mathbf{0} & \mathbf{L}_2 \end{bmatrix}, \tag{1}$$

in which $\mathbf{L}_1 = \mathbf{X}_1^T\mathbf{A}\mathbf{X}_1$, $\mathbf{L}_2 = \mathbf{Y}_2^T\mathbf{A}\mathbf{Y}_2$ and $\mathbf{H} = \mathbf{X}_1^T\mathbf{A}\mathbf{Y}_2$. The above equation is known as the *reduced form* of $\mathbf{A}$ with respect to $[\mathbf{X}_1\mathbf{Y}_2]$. The proof of why the bottom left block of this matrix is zero is straightforward, see [34] for details. This gives us the knowledge to define a *simple invariant subspace*.

Definition 5.2. *Let $\mathcal{X}$ be an invariant subspace of $\boldsymbol{A}$ and consider the reduced form of Equation* (1)*, then $\mathcal{X}$ is a* simple invariant subspace *of $\boldsymbol{A}$ if there are no common eigenvalues between $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$.*

Notice that a simple invariant subspace has a complementary space, defined as follows.

Definition 5.3. *Let the simple invariant subspace $\mathcal{X}_1$ have the reduced form of Equation* (1) *with respect to the orthogonal matrix* $[\boldsymbol{X}_1 \, \boldsymbol{Y}_2]$. *Then there exist* $\boldsymbol{X}_2$ *and* $\boldsymbol{Y}_1$ *such that* $[\boldsymbol{X}_1 \, \boldsymbol{X}_2]^{-1} = [\boldsymbol{Y}_1 \, \boldsymbol{Y}_2]^T$ *and*

$$\boldsymbol{A} = \boldsymbol{X}_1 \boldsymbol{L}_1 \boldsymbol{Y}_1^T + \boldsymbol{X}_2 \boldsymbol{L}_2 \boldsymbol{Y}_2^T, \tag{2}$$

where $\boldsymbol{L}_i = \boldsymbol{Y}_i^T \boldsymbol{A} \boldsymbol{X}_i$, $i = 1, 2$. *This form of* $\boldsymbol{A}$ *is known as the* spectral resolution *of* $\boldsymbol{A}$ *along* $\mathcal{X}_1$ *and* $\mathcal{X}_2$.

This allows us to introduce a theorem essential to our main result. However, first we must define the notion of angle between two subspaces.

Theorem 5.1. *Let* $\boldsymbol{X}_1, \boldsymbol{Y}_1 \in \mathbb{R}^{n \times \ell}$ *with* $\boldsymbol{X}_1^T \boldsymbol{X}_1 = \boldsymbol{I}$ *and* $\boldsymbol{Y}_1^T \boldsymbol{Y}_1 = \boldsymbol{I}$. *If* $2\ell \leq n$, *there are unitary matrices* $\boldsymbol{Q}, \boldsymbol{U}_{11}, \boldsymbol{V}_{11}$ *such that*

$$\boldsymbol{Q} \boldsymbol{X}_1 \boldsymbol{U}_{11} = \begin{pmatrix} \boldsymbol{I}_\ell \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix} \text{ and } \boldsymbol{Q} \boldsymbol{Y}_1 \boldsymbol{V}_{11} = \begin{pmatrix} \boldsymbol{\Gamma} \\ \boldsymbol{\Sigma} \\ \boldsymbol{0} \end{pmatrix},$$

in which $\boldsymbol{\Gamma} = diag(\gamma_1, \ldots, \gamma_\ell)$ *with* $0 \leq \gamma_1 \leq \ldots \leq \gamma_\ell$, $\boldsymbol{\Sigma} = diag(\sigma_1, \ldots, \sigma_\ell)$ *with* $\sigma_1 \geq \cdots \geq \sigma_\ell \geq 0$, *and* $\gamma_i^2 + \sigma_i^2 = 1$, $i = 1, \ldots, \ell$. *If* $2\ell > n$ *then* $\boldsymbol{Q}, \boldsymbol{U}_{11}, \boldsymbol{V}_{11}$ *can be chosen so that*

$$\boldsymbol{Q} \boldsymbol{X}_1 \boldsymbol{U}_{11} = \begin{pmatrix} \boldsymbol{I}_{n-\ell} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{2\ell-n} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \text{ and } \boldsymbol{Q} \boldsymbol{Y}_1 \boldsymbol{V}_{11} = \begin{pmatrix} \boldsymbol{\Gamma} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{2\ell-n} \\ \boldsymbol{\Sigma} & \boldsymbol{0} \end{pmatrix},$$

in which $\boldsymbol{\Gamma} = diag(\gamma_1, \ldots, \gamma_{n-\ell})$ *with* $0 \leq \gamma_1 \leq \ldots \leq \gamma_{n-\ell}$, $\boldsymbol{\Sigma} = diag(\sigma_1, \ldots, \sigma_{n-\ell})$ *with* $\sigma_1 \geq \cdots \geq \sigma_{n-\ell} \geq 0$, *and* $\gamma_i^2 + \sigma_i^2 = 1$, $i = 1, \ldots, n - \ell$.

Geometrically, let $\mathcal{X}_1$ and $\mathcal{Y}_1$ be subspaces of dimension ℓ and $\mathcal{Q}$ be a unitary transformation. Then the matrices $\boldsymbol{Q} \boldsymbol{X}_1 \boldsymbol{U}_{11}$ and $\boldsymbol{Q} \boldsymbol{Y}_1 \boldsymbol{V}_{11}$, with $\boldsymbol{X}_1 \in \mathcal{X}_1$, $\boldsymbol{X}_2 \in \mathcal{X}_2$, $\boldsymbol{Q} \in \mathcal{Q}$, form bases of $\mathcal{Q} \mathcal{X}_1$ and $\mathcal{Q} \mathcal{Y}_1$ and σ_i and γ_i can be regarded as sines and cosines of the angles between the bases. We now define a measure of similarity between subspaces.

Definition 5.4. *Let* $\mathcal{X}$ *and* $\mathcal{Y}$ *be subspaces of the same dimension, then the canonical angles* between the subspaces are the diagonal entries of the matrix $\sin^{-1} \boldsymbol{\Sigma} = diag(\sin^{-1}(\sigma_1), \ldots, \sin^{-1}(\sigma_{n-\ell}))$ *where* $\boldsymbol{\Sigma}$ *is the matrix defined in Theorem 5.1.*

It follows that $\boldsymbol{\Sigma}$ is a measure of how two subspaces differ.

5.2. Angle Between Exact and Updated rank-k Approximations

We are now ready to state the main theorem required for our result.

Theorem 5.2. *Let $\boldsymbol{A}$ be a Hermitian matrix with spectral resolution given by $[\boldsymbol{X}_1 \boldsymbol{X}_2]^T \boldsymbol{A} [\boldsymbol{X}_1 \boldsymbol{X}_2] = diag(\boldsymbol{L}_1, \boldsymbol{L}_2)$ where $[\boldsymbol{X}_1 \boldsymbol{X}_2]$ is unitary. Let $\boldsymbol{Z} \in \mathbb{R}^{n \times k}$ have orthonormal columns, $\boldsymbol{M}$ be any Hermitian matrix of order k and define the residual matrix as $\boldsymbol{R} = \boldsymbol{A}\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{M}$. Let $\lambda(\boldsymbol{A})$ represent the set of eigenvalues of $\boldsymbol{A}$ and suppose that $\lambda(\boldsymbol{M}) \subset [\alpha, \beta]$ and for some $\delta > 0$, $\lambda(\boldsymbol{L}_2) \subset \mathbb{R} \setminus [\alpha - \delta, \beta + \delta]$. Then, for any unitary invariant norm, we have:*

$$\| \sin \Theta(\mathcal{R}(\boldsymbol{X}_1), \mathcal{R}(\boldsymbol{Z})) \| \leq \frac{\|\boldsymbol{R}\|}{\delta}, \tag{3}$$

where $\mathcal{R}(\cdot)$ is the column space of a matrix.

Before we introduce the main result we present a result by Weyl [37] which characterises the perturbation in the eigenvalues of a matrix.

Theorem 5.3 (Weyl, [37])**.** *Define $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and let $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}$ be its perturbation. The eigenvalues of $\boldsymbol{A}$ and $\boldsymbol{E}$ are given by λ_i and ϵ_i, $i = 1, \ldots, n$, respectively. Then the eigenvalues of $\tilde{\boldsymbol{A}}$ are, for $i = 1, \ldots, n$, $\tilde{\lambda}_i \in [\lambda_i + \epsilon_n, \lambda_i + \epsilon_1]$.*

We can now present our main result which is closely related to the Davis-Kahan theorem [4].

Theorem 5.4. *Consider a positive semi-definite matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ with eigenvalues $\omega_1, \ldots, \omega_n$ and corresponding eigenvectors $\boldsymbol{Q} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$. Let $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ be symmetric with eigenvalues ϵ_i and $(\boldsymbol{A} + \boldsymbol{B})$ be positive semi-definite with eigen-decomposition $\boldsymbol{U} \boldsymbol{\Gamma} \boldsymbol{U}^T$ where γ_i are eigenvalues, $i = 1, \ldots, n$. Fix integer k, let $\boldsymbol{A}_k + \boldsymbol{B}$ have decomposition $\boldsymbol{V} \boldsymbol{\Pi} \boldsymbol{V}^T$ with eigenvalues $\pi_1, \ldots, \pi_n$ and assume $\gamma_k \neq \gamma_{k+1}$. Then the following bounds hold on the canonical angles between the subspaces defined by $\boldsymbol{U}_k$ and $\boldsymbol{V}_k$, assuming $\pi_k > \hat{\gamma}_{k+1}$,*

$$\| \sin \Theta(\mathcal{R}(\boldsymbol{U}_k), \mathcal{R}(\boldsymbol{V}_k)) \|_F \leq \frac{\sqrt{tr(\boldsymbol{V}_k^T \boldsymbol{A}_{k\perp}^2 \boldsymbol{V}_k)}}{\pi_k - \hat{\gamma}_{k+1}}, \tag{4}$$

$$\| \sin \Theta(\mathcal{R}(\boldsymbol{U}_k), \mathcal{R}(\boldsymbol{V}_k)) \|_2 \leq \frac{\sqrt{\lambda_{max}(\boldsymbol{V}_k^T \boldsymbol{A}_{k\perp}^2 \boldsymbol{V}_k)}}{\pi_k - \hat{\gamma}_{k+1}}, \tag{5}$$

where $\hat{\gamma}_{k+1} = \omega_{k+1} + \pi_{k+1}$.

Proof. We will start by considering the first bound. It is clear that $\mathbf{U}_k$ is a simple invariant subspace for $(\mathbf{A} + \mathbf{B})$. The spectral resolution of $\mathbf{A} + \mathbf{B}$ is given by $[\mathbf{U}_k \mathbf{U}_{k\perp}]$ since this matrix is unitary and we have $\mathbf{A} + \mathbf{B} = \mathbf{U}_k \mathbf{U}_k^T (\mathbf{A} + \mathbf{B}) \mathbf{U}_k \mathbf{U}_k^T + \mathbf{U}_{k\perp} \mathbf{U}_{k\perp}^T (\mathbf{A} + \mathbf{B}) \mathbf{U}_{k\perp} \mathbf{U}_{k\perp}^T$. In the reduced form $\mathbf{L}_1 = \mathbf{\Gamma}_k$ and $\mathbf{L}_2 = \mathbf{\Gamma}_{k\perp}$. Furthermore, we set $\mathbf{M} = \mathbf{\Pi}_k$ and $\mathbf{Z} = \mathbf{V}_k$. The residual matrix is given by $\mathbf{R} = (\mathbf{A} + \mathbf{B})\mathbf{V}_k - \mathbf{V}_k \mathbf{\Pi}_k = \mathbf{A}_{k\perp} \mathbf{V}_k$ and

$$\|\mathbf{R}\|_F = \sqrt{tr(\mathbf{V}_k^T \mathbf{A}_{k\perp}^2 \mathbf{V}_k)}. \tag{6}$$

We know that the eigenvalues of $\mathbf{M}$ fall within the range $[\pi_k, \pi_1]$ and those of $\mathbf{L}_2$ are bounded using Theorem 5.3 in the range $\gamma_i \in [\pi_i + \omega_n, \pi_i + \omega_{k+1}]$ for $i = k+1, \ldots, n$. Considering also the perturbation of eigenvalues of $\mathbf{A}$ we can write $\gamma_i \leq \hat{\gamma}_i = \min(\omega_{k+1} + \pi_{k+1}, \omega_{k+1} + \epsilon_1) = \omega_{k+1} + \pi_{k+1}$ given $\pi_{k+1} \leq \omega_{k+1} + \epsilon_1$. It follows that $\delta = \pi_k - \hat{\gamma}_{k+1}$ and plugging into Theorem 5.2 gives the required result.

The second bound is proved similarly except in this case we have

$$\|\mathbf{R}\|_2 = \sqrt{\lambda_{max}(\mathbf{V}_k^T \mathbf{A}_{k\perp}^2 \mathbf{V}_k)}.$$

$\square$

Thus we have a bound on the angle between the subspace of the first k eigenvectors of $\mathbf{A}_k + \mathbf{B}$ and the corresponding eigenvectors of it perturbation $\mathbf{A} + \mathbf{B}$ without explicitly requiring the eigen-decomposition of $\mathbf{A} + \mathbf{B}$. Provided the eigenvalues of $\mathbf{V}_k^T \mathbf{A}_{k\perp}^2 \mathbf{V}_k$ are small and the eigengap $\pi_k - \hat{\gamma}_{k+1}$ is large one can be sure that the two subspaces have small canonical angles. One can see that under small perturbations $\mathbf{V}_k$ is close to $\mathbf{Q}_k$ and hence $\mathbf{V}_k^T \mathbf{Q}_{k\perp}$ is small, resulting in tight bounds in the angles. In the case that the matrices involved correspond to normalised Laplacians this result corresponds well with similar results outlining a perturbation-based motivation of spectral cluster (see e.g. [36]) which state that if the value of $\gamma_k - \gamma_{k+1}$ is large then one might reasonably expect a good clustering. The bound becomes loose when this eigengap is small, however in this case the clusters are less distinct even when computing the exact eigenvectors.

6. Incremental Cluster Membership

We now return to the eigenproblem of Algorithm 1, $\tilde{\mathbf{L}}\mathbf{v} = \lambda\mathbf{v}$, in which we are interested in the eigenvectors with the smallest eigenvalues. Define the *shifted Laplacian* as

$$\hat{\mathbf{L}} = 2\mathbf{I} - \tilde{\mathbf{L}} = \mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}},$$

which is positive semi-definite since $\tilde{\mathbf{L}}$ is positive semi-definite with largest eigenvalue 2. Note that by negating a matrix one negates the eigenvalues, leaving the eigenvectors the same, and similarly an addition of $\sigma\mathbf{I}$ increases the eigenvalues by σ leaving the eigenvectors intact. Since we are interested in the smallest eigenvectors of $\tilde{\mathbf{L}}$ they correspond exactly to the maximum eigenvectors of $\hat{\mathbf{L}}$ and we can use the eigen-update methods described above. Observe that the shifted Laplacian is a normalised version of the *signless Laplacian* [13] defined as $\mathbf{L}^+ = \mathbf{D} + \mathbf{W}$, and this can be seen from $\hat{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D} + \mathbf{W})\mathbf{D}^{-\frac{1}{2}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}^+\mathbf{D}^{-\frac{1}{2}}$.

Putting the ingredients together allows us to outline an efficient incremental method for performing graph clustering called Incremental Approximate Spectral Clustering (IASC), see Algorithm 2. At a high level the algorithm is quite simple: in the initialisation steps one computes the shifted Laplacian matrix

for the first graph $\hat{\mathbf{L}}_1$ and then performs k-means clustering using the largest k eigenvectors of this matrix. For the t-th successive graph, $t > 1$, we use the eigen-update methods above to approximate the largest eigenvectors of $\hat{\mathbf{L}}_t$ using the approximate eigenvectors computed at the previous iteration in step 10. For these updates, it is simple to recover the matrices $\mathbf{Y}_1$ and $\mathbf{Y}_2$ given a change in edge weights. Note that we use the first method of Section 4.2 to compute eigenvector upon the addition of row and columns. Furthermore, we store the first ℓ eigenvectors of $\hat{\mathbf{L}}_1$, with $\ell \geq k$, and recompute eigenvectors every T iterations in order to reduce cumulative errors introduced in the loop at the expense of increased computation. When $\ell = n$ one recovers the exact eigenvectors at each iteration and Algorithms 2 and 1 become nearly equivalent. In the case that there is a significant eigengap between eigenvalues, one can fix both ℓ and k according to Theorem 5.4.

Algorithm 2 Incremental Approximate Spectral Clustering

Require: Graphs $G_1, \ldots, G_T$ of sizes $n_1, \ldots, n_T$, no. clusters k, approximation rank $\ell \geq k$, eigen-decomposition recomputation step R
1: Compute the shifted Laplacian for G_1, $\hat{\mathbf{L}}_1$
2: Find ℓ largest eigenvectors of $\hat{\mathbf{L}}_1$, $\mathbf{V}_\ell^{(1)}$, let $\mathbf{V}_k^{(1)}$ be the first k cols of $\mathbf{V}_\ell^{(1)}$
3: Normalise the eigenvector rows, $\mathbf{V}_k^{(1)} \leftarrow \mathrm{diag}(\mathbf{V}_k^{(1)}(\mathbf{V}_k^{(1)})^T)^{-\frac{1}{2}} \mathbf{V}_k^{(1)}$
4: Use k-means on rows of $\mathbf{V}_k^{(1)}$ and store indicators $\mathbf{c}_1 \in \{1, \ldots, k\}^{n_1}$
5: **for** $t = 2 \rightarrow T$ **do**
6:    Compute shifted Laplacian for G_t, $\hat{\mathbf{L}}_t$
7:    **if** $i \mathbin{\%} R == 0$ **then**
8:      Recompute eigenvectors of $\hat{\mathbf{L}}_t$
9:    **else**
10:     Use rank-ℓ eigen-approximation of Section 4
11:    **end if**
12:    Normalise rows of $\mathbf{V}_k^{(t)}$, $\mathbf{V}_k^{(t)} \leftarrow \mathrm{diag}(\mathbf{V}_k^{(t)}(\mathbf{V}_k^{(t)})^T)^{-\frac{1}{2}} \mathbf{V}_k^{(t)}$
13:    Use k-means on $\mathbf{V}_k^{(t)}$ with initial centroids $\mathbf{c}_{t-1}$, store $\mathbf{c}_t \in \{1, \ldots, k\}^{n_t}$
14: **end for**
15: **return** Cluster membership $\mathbf{c}_1 \in \{1, \ldots, k\}^{n_1}, \ldots, \mathbf{c}_T \in \{1, \ldots, k\}^{n_T}$

The complexity of Algorithm 2 is dictated by the sparsity of the graphs and the extent of the change between successive graphs. We ignore the steps before the for-loop since in general they do not impact the overall complexity. At iteration t and step 6 one can compute $\hat{\mathbf{L}}_t$ from the weight and degree matrix at a cost of $\mathcal{O}(n_t + |E_t|)$. In the following step if there is a change between edges incident to vertices $S = \{v_{I_1}, \ldots, v_{I_\ell}\}$ then the rows and columns corresponding to the union of the neighbours of S, $n(S)$, will change in the corresponding shifted Laplacian. In this case $p = |n(S)|$ in $\mathbf{Y}_1, \mathbf{Y}_2 \in \mathbb{R}^{n \times p}$ and if the neighbourhood of the vertices with changed edges $n(S)$ is small, this update can be efficiently computed as outlined in Section 4. The cost of k-means is $\mathcal{O}(k^2 n s)$ where s is the number of iterations required for convergence.

7. Computational Results

In this section we study the clustering quality of four incremental strategies: the naive one which computes the exact eigenvectors at each iteration, denoted `Exact`, Ning et al.'s method (`Ning`) since it is a competing incremental strategy, `IASC`, and `Nyström` which uses the Nyström eigen-decomposition approximation of the shifted Laplacian, since it is often used in spectral clustering. First, however we observe the quality of the eigenvectors found using the Nyström approximation and our eigen-updating approach.

7.1. Quality of Approximate Eigenvectors

We make a comparison between the Nyström approximation and our eigen-updating methods on a synthetic dataset generated in the following way: The initial graph contains 4 clusters of size 250 which are generated using an Erdös-Rényi [8] process with edge probability $p = 0.1$. The Erdös-Rényi process creates edges independently randomly with a fixed probability p for all pairs of vertices. For each successive graph we then add 50 random edges to simulate "noise" in the clusters for a total of 100 graphs. We then compute the largest $k = 4$ eigenvectors of the shifted Laplacian using the full eigen-decomposition, the Nyström method and the eigen-updating approach of Section 4. For the Nyström method we use $m = \{600, 900\}$ randomly selected columns and for the eigen-updating approach we update based on the approximate eigenvectors and eigenvalues found for the previous graph. In order to measure the quality of the approximations we apply a slightly different form of Theorem 5.2 in which $\delta = \min |\lambda(\mathbf{L}_2) - \lambda(\mathbf{M})|$ and one uses the Frobenius norm, see Theorem V.3.4 of [34] for details. In this case $\mathbf{L}_2$ is a diagonal matrix of the smallest $n - k$ eigenvalues $\omega_{k+1}, \ldots, \omega_n$ of the Laplacian, $\mathbf{Z}$ is the first k approximate eigenvectors and $\mathbf{M} = \mathbf{Z}^T \mathbf{A} \mathbf{Z}$ in the notations of the theorem. The experiment is repeated with results averaged over 20 iterations.

Observe that the approximations given by the Nyström method are generally worse than that of eigen-approximation, see Figure 1(a). For example, with $m = 900$ the norm of $\mathbf{\Sigma}$ is 0.739 compared to 0.253 with the eigen-updating approach with 4 eigenvectors for the final graph. We also tested the Nyström approximation in conjunction with the normalised Laplacian and found the canonical angles were worse still. Notice that the canonical angles become larger over time and this is due the increasing difficulty in separating the first eigenvectors from the remaining ones. Furthermore, when $m = 600$ the Nyström eigenvectors separate their angles with the true ones at a faster rate than when $m = 900$.

7.1.1. Perturbation Bound

We now turn to demonstrating the effectiveness of the bound of Theorem 5.4 on a synthetic dataset containing 150 vertices. The initial graph contains 3 clusters of size 50 which are generated using an Erdös-Rényi process with edge probability $p = 0.3$, resulting in 1092 edges. For each successive graph we add 10 random edges and in total there is a sequence of 80 graphs. We then compute the bound of Theorem 5.4 to measure the difference between the canonical

angles of the real and approximated eigenvectors of the shifted Laplacian. We compare the results to the bound using the real eigenvalues of the Laplacian, i.e. $\delta = \pi_k - \gamma_{k+1}$. This process is repeated 50 times with different random seeds and the results are averaged.
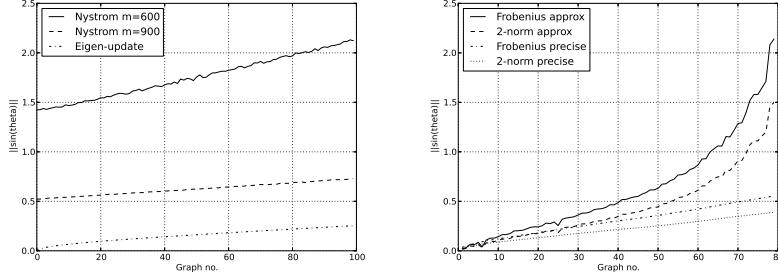

Figure 1: The left plot compares the canonical angles of the largest eigenvectors found using the approximation methods with the equivalent real eigenvectors. On the right we compare the perturbation bounds of Theorem 5.4. By "precise" we mean that we use $\delta = \pi_k - \gamma_{k+1}$ in the theorem.

Figure 1(b) shows the resulting bounds for this sequence of graphs. The bound diverges slowly from the precise bound for many of the initial graphs, for example $\| \sin \Theta(\mathcal{R}(\mathbf{U}_k), \mathcal{R}(\mathbf{V}_k)) \|_F$ is bounded by 0.616 ± 0.106 versus 0.352 ± 0.024 at the 50th graph and the equivalent results for the 2-norm are 0.434 ± 0.007 and 0.248 ± 0.02. Soon after this point, we see a large divergence in the precise and approximate bounds, although the approximate bounds become trivial after the 78th graph for the Frobenius norm and the 73rd for the 2-norm corresponding to the addition of 770 and 730 edges respectively to the initial graph. When we look at the precise bound, one can see that at the last graph $\| \sin \Theta(\mathcal{R}(\mathbf{U}_k), \mathcal{R}(\mathbf{V}_k)) \|_2 \leq 0.39$ despite nearly doubling the number of edges which points to the precision of the eigen-approximation in this case.

7.2. Clustering on Synthetic Data

To evaluate the clustering approaches, two separate synthetic datasets are considered, both of which are generated by an Erdös-Rényi process. The first dataset, called `3clust`, is based on a graph of 3 clusters of 60 vertices each. This dataset allows us to compare the clustering quality as the clusters become more/less distinct and also the addition and removal of edges. In the corresponding graph, any possible edge between two points from the same cluster occurs with probability $p_c = 0.3$ and the probability of edges between vertices in different clusters is selected from $p_g \in \{0.1, 0.2\}$. To generate a sequence of graphs we first allow only 20 vertices per cluster and then add 5 vertices to each cluster at a time until each one is of size 60. We then reverse the process so that the 10th graph is the same as the 8th, and the 11th is the same as the 7th etc., which allows us to test the clustering methods upon the removal of vertices.
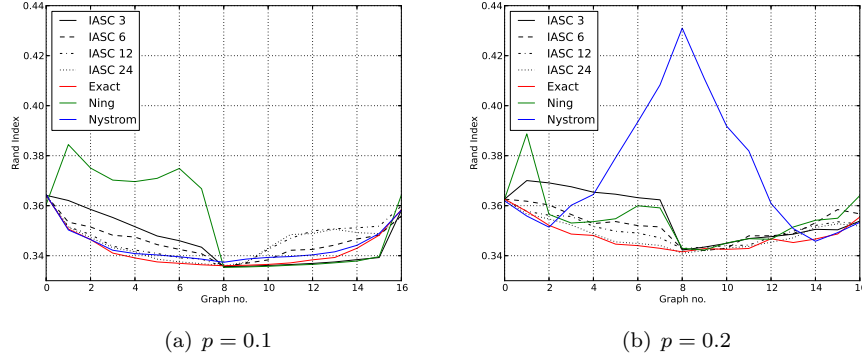
16

(a) $p = 0.1$ (b) $p = 0.2$

Figure 2: The mean Rand Index error of the learned clustering on `3clust`. The numbers after `IASC` denote the number of eigenvectors computed.

The second dataset, `discrepancy`, aims to examine the clustering methods on a more complex set of hierarchical clusters. Here 180 vertices are split into 3 clusters of equal size, and inside each cluster there are 3 sub-clusters. The initial graph is empty, and at each iteration $t \in \{1, \ldots, T\}$, $T = 23$, an edge $\{v_i, v_j\}$ is added with a probability $p_\varepsilon + (p - p_\varepsilon)\ell t/T$, where ℓ is set to 1 if v_i and v_j are in the same subgroup, 0.5 if v_i and v_j are in the same group, and 0 otherwise. In our case we set $p_\varepsilon = 0.0005$ and $p = 0.01$.

For both datasets and the clustering approaches, k-means is run with k corresponding to the number of clusters (3 and 9 for `3clust` and `discrepancy` respectively). For IASC we fix the number of eigenvectors $\ell \in \{3, 6, 12, 24\}$ with the `3clust` dataset and $\ell \in \{9, 18, 72\}$ with `discrepancy` in order to test the approximation quality as this parameter varies. With `Nyström` we sample $m = 90$ columns of the Laplacian matrix to find the approximate eigenvectors on `3clust` and select $m \in \{9, 18, 36, 72\}$ for `discrepancy`. On `discrepancy`, the approximation methods start with the 3rd graph in the sequence to allow the initial graph to contains enough edges. The experiments are repeated 50 times with different random graphs constructed using the methods described above and the results are averaged. Clustering accuracy is measured through the *Rand Index* [28] between the finest true clustering $\mathcal{C}$ and the learned one $\widehat{\mathcal{C}}$. Rand Index corresponds to the proportion of true answers to the question "*Are vertices v_i and v_j in the same cluster ?*". More formally, it is given by

$$\text{RandIndex}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{\left| \left\{ v_i, v_j \in V : \ v_i \neq v_j, \ \delta(\mathcal{C}(v_i), \mathcal{C}(v_j)) \neq \delta(\widehat{\mathcal{C}}(v_i), \widehat{\mathcal{C}}(v_j)) \right\} \right|}{\left| \{ v_i, v_j \in V : \ v_i \neq v_j \} \right|},$$

where $\mathcal{C}(v_i)$ stands for the cluster index of vertex v_i after clustering $\mathcal{C}$, δ is the Kronecker delta function and $|\mathcal{E}|$ denotes the cardinality of any finite set $\mathcal{E}$.

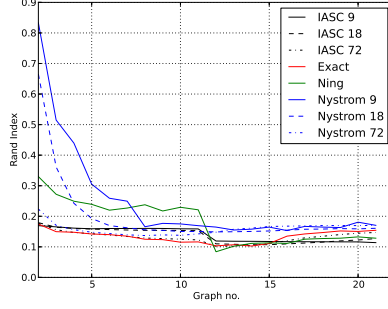The errors on `3clust` are shown in Figure 2. The first point to note is

17

Figure 3: The mean Rand Index of the learned clusterings on `discrepancy`. For `IASC` and `Nyström` the numbers in the legend indicates the number of eigenvectors and columns sampled at each iteration.

that the errors decrease as the cluster size increases up until graph 8 and then increase again as vertices are removed. This is explained by that fact that as the cluster size increases there are more edges within each cluster relative to those between clusters, hence they becomes easier to identify. Notice also that the methods which use the shifted Laplacian have identical results for the first graph and the approximation methods perform worse than `exact` which is what one might expect. When $p = 0.1$ `IASC` $\ell = 24$ remains close to the exact eigen-computation, while `Ning` becomes sharply worse. The Nyström approach is only slightly worse than `exact` despite using only 90 columns to approximate the eigen-decomposition. Note that we recompute eigenvectors for `IASC` and `Ning` at graph 8, and upon removal of vertices it is `IASC` that results in the largest errors except when $\ell = 3$. `Ning` works well with vertex removal, remaining close to `exact`.

When $p = 0.2$ the clusters are less well defined and the Nyström approach performs badly with an error of 0.43 versus 0.34 with `exact` at graph 8. If we compare the eigengap between the 3rd and 4th eigenvalues then they are 0.11 and 0.04 for $p = 0.1$ and $p = 0.2$ respectively. `IASC` is more comparable to `exact` in this case, however again we observe that `Ning` clusters badly when vertices are added, though better upon vertex removal. It is worth noting that since ℓ is a fixed value for `IASC`, the approximation of the largest eigenvectors of the shifted Laplacian represents a smaller fraction of the total sum of eigenvalues as the cluster size increases. When using 24 eigenvectors for the approximation on the 2nd largest graph one requires approximately 14.5% of the dimensionality of the Laplacian, yet the rand index is close to that of `exact`.

The results on `discrepancy` (Figure 3) show that `Ning` generally does not generate accurate clustering until the 12th graph when eigenvectors are recomputed, however it becomes competitive after this point. This is partly due to the fact that one solves a different eigensystem to the other methods which is not as effective at clustering for the initial graphs. As we increase the number

of columns used for the Nyström approach the rand indices improve for the first few graphs in which the clusters are generally not well defined. In contrast, `IASC` results are relatively accurate even for these initial graphs, and surprisingly at the final graph we see that with just 9 eigenvectors `IASC` results in the most accurate clustering. At this point more eigenvectors seem to make the solution worse, with the exact eigen-decomposition being less accurate than `IASC` with 72 eigenvectors. One explanation is that an accurate eigen-decomposition fits "noise" in the Laplacian which is excluded with `IASC` and a low value of ℓ.

7.3. Real-world Graphs

We now apply the clustering methodology analysed previously on several real datasets.

7.3.1. Setup

Here we use the clustering methods on three real datasets. The first one represents individuals in Cuba who are detected as HIV positive between the period 1986 and 2004, see [1] for details of the related database. Edges in the graph indicate the occurrence of a sexual contact between two individuals as determined using *contact tracing*, whereby contacts of an infected person are identified and tested. The full sexual contact graph at the end of 2004 consists of 5389 people however it is strongly disconnected and we consider the growth of the largest component of size 2387. We find graphs of detected individuals at 1 month intervals, starting when the graph contains at least 500 vertices. At any particular time point, we consider the component containing first person detected in the largest component at the end of the recorded epidemic.

The next dataset is the high energy physics theory citation network from the Arxiv publications database [12] for the period February 1992 to March 2002. The full dataset contains 27,770 papers with 352,807 edges and a graph is generated as follows: if a paper cites another, an edge is made from the former to the latter. Not all of the papers present in the dataset have publication dates and hence we use only those with dates and label cited papers with the date of the oldest citing paper. Taking the largest component, the final resulting graph consists of 15,112 vertices and 193,826 edges. To track the evolution of the connected component we start with the oldest paper, and consider the graphs of connected papers at 1 month intervals, starting with a graph of at least 500 vertices.

The final dataset, called `Bemol`, was first introduced in [29] and corresponds to the purchase history of an e-commerce website over a period of almost two years. The initial dataset is a bipartite graph between users and products composed of more than 700,000 users and 1,200,000 products. In the current experiment we focus on the first 10,000 users, and a graph is constructed between users with edge weights corresponding to the number of commonly purchased products between two users. Taking a maximum of 500 purchases per iteration in the graph sequence we focus on graphs 500 to 600.

To test the clustering methods we run each on the sequences of evolving graphs under a variety of parameters. As the selection of the number of clusters

is a complex issue and outside the scope of this paper and we manually choose this value for each dataset. The experiment is run using $k = 25$ clusters for `HIV`, $k = 50$ for `Citation` and $k = 100$ for `Bemol`. For `HIV` and `IASC`, $\ell \in \{25, 50, 100\}$ and $R = 10$ and for `Ning` we recompute exact eigenvectors after every 10 iterations. When using `Citation` and `Bemol`, eigenvector recomputations are performed every 20 iterations and $\ell \in \{50, 100, 200, 500\}$ and $\ell \in \{100, 200, 500\}$. With `Nyström` the number of columns is chosen from $m \in \{500, 1000, 1500\}$ for `HIV` and $m \in \{1000, 2000, 5000\}$ for the other datasets. Note that in our implementation of k-means clustering, k represents an upper bound on the number of clusters found. To evaluate the learned clusters we use measures of *modularity* and *k-way normalised cut*. Let $\mathbf{d}_i = \sum_j \mathbf{W}_{ij}$ and $r = \sum_i \mathbf{d}_i$, then the modularity is defined as

$$Q = \frac{1}{2r} \sum_{i,j} \left(\mathbf{W}_{ij} - \frac{\mathbf{d}_i \mathbf{d}_j}{2r} \right) \delta(\mathbf{c}_i, \mathbf{c}_j),$$

where $\mathbf{c} \in \mathbb{R}^n$ is the cluster indicator vector and δ is the Kronecker delta function. Intuitively modularity is the difference in the sum of edges weights within a cluster and the expected edge weights assuming the same weight distribution $\mathbf{d}$ for each vertex. The k-way normalised cut is

$$N = \frac{1}{k} \sum_{\ell=1}^{k} \frac{\sum_{ij} \mathbf{W}_{ij} \delta(\mathbf{c}_i, \mathbf{c}_\ell)(1 - \delta(\mathbf{c}_j, \mathbf{c}_\ell))}{\sum_{ij} \mathbf{W}_{ij} \delta(\mathbf{c}_i, \mathbf{c}_\ell)}.$$

A cut between two clusters A and B is the sum of the weights between the clusters and the normalised cut is this sum divided by the sum of the weights of all edges incident to vertices in cluster A. Hence the k-way normalised cut is the mean normalised cut between each cluster and its complementary vertices. To summarise, the greater the modularity, the better, and the lower the k-way normalised cut the better.

All experimental code is written in Python and we use an Intel Core i7-2600K at 3.40GHz with 16GB of RAM to conduct the simulations. The Laplacian matrices are stored in compressed sparse row representation and eigenvectors are found using Implicitly Restarted Lanczos Method in ARPACK [18] which computes only the required eigenvectors and not the full eigen-decomposition.

7.3.2. Results

Figure 4 show the resulting modularities and k-way normalised cuts for all datasets however we begin by studying `HIV`. For both `IASC` and `Ning`, since eigenvectors are recomputed every 10 iterations, this can manifest itself as sudden changes in the modularities and k-way normalised cuts. These changes are more pronounced with `Ning`. We see a close correspondence of `IASC` and `exact` for both the modularity and normalised cut. As we observed with the toy datasets, a lower value of ℓ seems to improve results with the final graph having a cut of 0.09 with `IASC` $\ell = 100$ versus 0.10 for `exact`. Notice that `IASC` matches or improves results on `exact`, while keeping only 5% or fewer of the
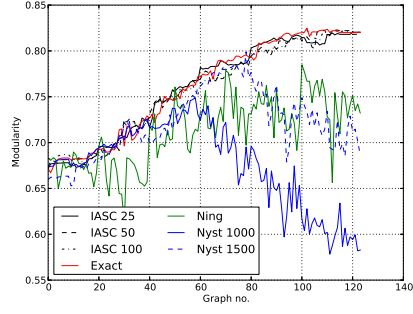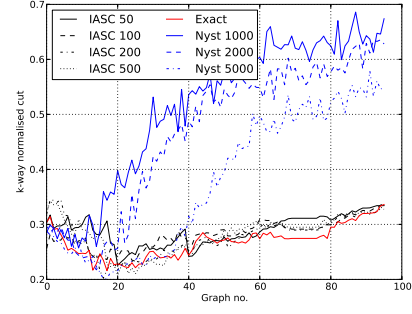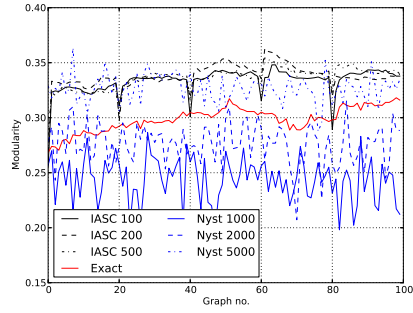
(a) HIV modularity
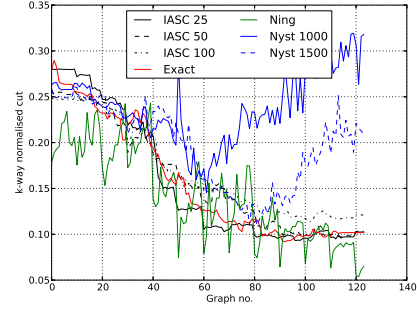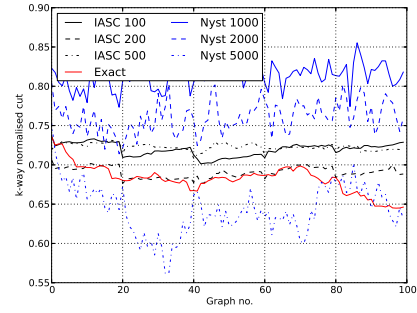
(b) HIV k-way normalised cut
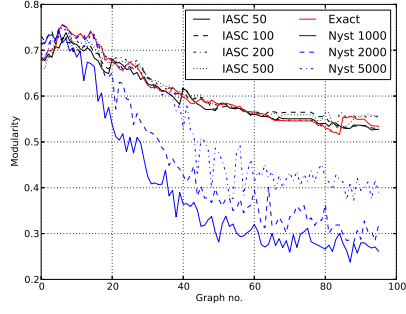
(c) Citation modularity

(d) Citation k-way normalised cut

(e) Bemol modularity

(f) Bemol k-way normalised cut

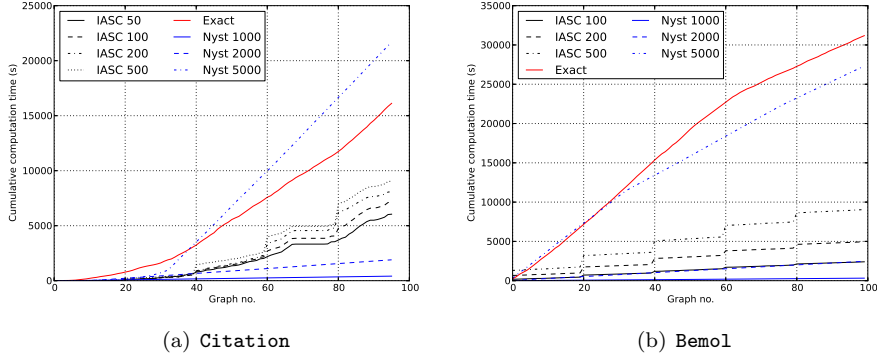Figure 4: The performance of the clustering methods on the sequences of changing graphs.

21

Figure 5: The cumulative times taken by the eigenvector computations of the clustering methods.

final number of eigenvectors. `Ning` fares badly in terms of the modularity of the resulting clustering with a value of 0.73 versus 0.82 for the `exact` approach at the final graph. However, with the cut measure `Ning` provides the best clustering, albeit with more unstable curve than the other methods. `Nyström` does not provide a good approximation of the largest eigenvectors when the rank of the Laplacian exceeds the number of columns sampled.

A similar picture emerges with `Citation` and we see again that `IASC` is close to `exact` in terms of both measures. Note that it was too costly to compute clustering using `Ning` on this and `Bemol`. On `Citation`, when $\ell \in \{200, 500\}$ we obtain a close match to `exact` in general. The results are impressive when we consider the change in the graphs between eigenvector recomputations: the first graph is of size 555, and the 19th is 2855, an increase of 2300. With the 60th graph there are 10063 vertices and 12135 at the 79th. Looking at the `Nyström` curves, we again observe poor clustering performance even when $m = 5000$.

Finally consider the `Bemol` graphs in which is difficult to find clear clusters, although they become more distinct over time. This is evident when looking at the `exact` curves for example: the modularity increases slightly from 0.27 to 0.32 whereas the k-way normalised falls from 0.74 to 0.65 from beginning to end. In contrast to the other datasets `exact` is improved upon by both `IASC` and `Nyström` (when $m = 5000$) respectively. One of the reasons that the Nyström method is effective on this data is because there are many edges and one can sample them out without affecting the clustering significantly. Note however that as we have seen in the other plots, `Nyström` is rather unstable compared to the other methods. Furthermore, the eigenvector updates every 20 iterations are noticeable in the cluster measures with `IASC`.

To conclude the analysis, Figure 5 shows the timings of the eigenvector computations of the clustering methods for `Citation` and `Bemol`. Clearly the Nyström approach costs the least in term of computation when $m \in \{1000, 2000\}$

22

however exceeds the time taken for `exact` when $m = 5000$. In this case the time is dominated by the eigen-decomposition of a matrix in $\mathbb{R}^{m \times m}$. `IASC` improves over `exact` over the whole sequence of graphs as one does not recompute the eigenvectors at every iteration. Notice that the "staircase" effect in the `IASC` curves correspond the computation of the exact eigenvectors. Observe that on `Bemol`, `IASC` $\ell = 200$ takes 4,956 seconds in total for eigenvector computations versus 31,182 for `exact`, a speedup factor of 6.29 for a similar cluster quality. The equivalent improvement is 2.26 on `Citation`.

8. Discussion

We have presented a novel incremental method for spectral graph clustering which updates the eigenvectors of the Laplacian in a computationally efficient way. Such an algorithm is useful for finding clusterings in time evolving graphs such as biological and social networks, and the Internet. A key part of the algorithm uses the principles behind the SVD updating approach of [40] to derive a general way to approximate the first k eigenvectors of a perturbed symmetric matrix given the eigenvectors of the original matrix. The resulting clustering algorithm, IASC, can be easily implemented using a standard linear algebra library.

We analysed IASC in both theoretical and empirical respects. Using perturbation theory, we showed when the canonical angles between the real and approximate subspaces generated by our update algorithm are close. Furthermore, IASC is examined empirically relative to the computation of exact eigenvectors for each graph, the method of Ning et al., and the Nyström approach. On 2 toy and 3 real datasets we show that IASC can often match the cluster accuracy of the exact approach using a small fraction of the total number of eigenvectors and at a much reduced computational cost.

This work has opened up several perspectives for further study. The first is the analysis of the update of eigenvectors for a modularity matrix and other cluster quality criteria. As we have shown, the quality of the updates would depend on the spectrum of the matrices in question. Another interesting line of research is the issue of how to choose the number of clusters in the time evolving graphs.

References

[1] B. Auvert, H. de Arazoza, S. Clémençon, J. Perez, and R. Lounes. The HIV/AIDS epidemic in Cuba: description and tentative explanation of its low HIV prevalence. *BMC Infectious Diseases*, 7(30), November 2007.

[2] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

[3] F.R.K. Chung. *Spectral graph theory*. Amer Mathematical Society, 1997.

[4] C. Davis and W.M. Kahan. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

[5] Charanpal Dhanjal, Romaric Gaudel, and Stéphan Clémençon. Incremental spectral clustering with the normalised laplacian. In *3rd NIPS Workshop on Discrete Optimization in Machine Learning*, 2011.

[6] P. Drineas, R. Kannan, and M.W. Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006.

[7] P. Drineas and M.W. Mahoney. On the nyström method for approximating a gram matrix for improved kernel-based learning. *The Journal of Machine Learning Research*, 6:2153–2175, 2005.

[8] Paul Erdős and Alfréd Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[9] G.W. Flake, R.E. Tarjan, and K. Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1(4):385–408, 2004.

[10] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[11] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nystrom method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):214–225, 2004.

[12] J. Gehrke, P. Ginsparg, and J. Kleinberg. Overview of the 2003 kdd cup. *ACM SIGKDD Explorations Newsletter*, 5(2):149–151, 2003.

[13] W.H. Haemers and E. Spence. Enumeration of cospectral graphs. *European Journal of Combinatorics*, 25(2):199–211, 2004.

[14] Harold Hotelling. Analysis of a complex of statistical variables into principle components. *Journal of Educational Psychology*, 24:417–441 and 498–520, 1933.

[15] T. Kong, Y. Tian, and H. Shen. A fast incremental spectral clustering for large data sets. In *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2011 12th International Conference on*, pages 1–5. IEEE, 2011.

[16] S. Kumar, M. Mohri, and A. Talwalkar. Sampling techniques for the nyström method. In *Conference on Artificial Intelligence and Statistics*, pages 304–311, 2009.

[17] James T. Kwok and Zhao Haitao. Incremental eigen-decomposition . In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 270–273, 2003.

[18] R.B. Lehoucq, D.C. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, volume 6. Siam, 1998.

[19] M. Li, X.C. Lian, J.T. Kwok, and B.L. Lu. Time and space efficient spectral clustering via column sampling. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2297–2304. IEEE, 2011.

[20] M. Mahdavi, T. Yang, and R. Jin. An improved bound for the nystrom method for large eigengap. *arXiv preprint arXiv:1209.0001*, 2012.

[21] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, Jun 2004.

[22] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[23] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[24] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14: Proceeding of the 2001 Conference*, pages 849–856, 2001.

[25] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. Huang. Incremental spectral clustering with application to monitoring of evolving blog communities. In *SIAM Int. Conf. on Data Mining*. Citeseer, 2007.

[26] H. Ning, W. Xu, Y. Chi, Y. Gong, and T.S. Huang. Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition*, 43(1):113–127, 2010.

[27] G.W. O'Brien. Information management tools for updating an svd-encoded indexing scheme. *Master's thesis, The University of Knoxville, Tennessee, Knoxville, TN*, 1994.

[28] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[29] Emile Richard, Nicolas Baskiotis, Theodoros Evgeniou, and Nicolas Vayatis. Link discovery using graph feature tracking. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS'10)*, pages 1966–1974, 2010.

[30] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 737–746. ACM, 2009.

[31] S.E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[32] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

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

[34] G.W. Stewart and J. Sun. *Matrix perturbation theory*, volume 175. Academic press New York, 1990.

[35] C. Valgren, T. Duckett, and A. Lilienthal. Incremental spectral clustering and its application to topological mapping. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4283–4288. IEEE, 2007.

[36] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[37] H. Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, 1912.

[38] Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688, Cambridge, MA, 2000. MIT Press.

[39] D. Yan, L. Huang, and M.I. Jordan. Fast approximate spectral clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 907–916. ACM, 2009.

[40] H. Zha and H.D. Simon. On updating problems in latent semantic indexing. *SIAM Journal on Scientific Computing*, 21:782, 1999.