Graph Approximation and Clustering on a Budget

Ethan Fetaya
Weizmann Institute of Science

Ohad Shamir
Weizmann Institute of Science

Shimon Ullman
Weizmann Institute of Science

Abstract

We consider the problem of learning from a similarity matrix (such as spectral clustering and low-dimensional embedding), when computing pairwise similarities are costly, and only a limited number of entries can be observed. We provide a theoretical analysis using standard notions of graph approximation, significantly generalizing previous results, which focused on spectral clustering with two clusters. We also propose a new algorithmic approach based on adaptive sampling, which experimentally matches or improves on previous methods, while being considerably more general and computationally cheaper.

1 Introduction

Many unsupervised learning algorithms, such as spectral clustering [18], [2] and low-dimensional embedding via Laplacian eigenmaps and diffusion maps [3],[16], need as input a *matrix of pairwise similarities W* between the different objects in the dataset. In some cases, obtaining the full matrix can be a costly matter. For example, W_{ij} may be based on some expensive-to-compute metric such as W2D [5]; based on some physical measurement (such as in certain computational biology applications); or is given by a human annotator. In such cases, we would like to have a good approximation of the initially unknown matrix, while querying only a limited number of entries. An alternative but equivalent viewpoint is the problem of approximating an unknown weighted undirected graph, by querying a limited number of edges.

The problem of using machine learning algorithms on partially sampled matrices has previously received attention in works such as [17] and [9], under the assumption that two distinct clusters indeed exists. Namely, they assume a large gap between the second and third eigenvalues of the Laplacian matrix. In this work we consider, both theoretically and algorithmically, the question of query-based graph approximation more generally, obtaining results relevant beyond two clusters and beyond spectral clustering.

When considering graph approximations, the first question is what definition of approximation to consider. One important definition is *cut approximation* [14], where we wish for every cut in the approximated graph to have a weight close to the weight of the cut in the original graph up to a multiplicative factor. Many machine learning algorithms (and many more general algorithms) such as cut based clustering [11], energy minimization [22], and many others [1] are based on cuts, so this definition of approximation is natural for these uses. An alternative definition is *spectral approximation* [19], where we wish to uniformly approximate the quadratic form defined by the Laplacian up to a multiplicative factor. This approximation is important for algorithms such as spectral clustering [2], Laplacian eigenmaps [3], diffusion maps [16], etc. that use the connection between the spectral properties of the Laplacian matrix and the graph.

We first consider the simple and intuitive strategy of sampling edges uniformly at random and obtain results for both cut and spectral approximations under various assumptions. We then show how these results can be applied to the problem of clustering. We note that these results are considerably more general than the theoretical analysis in [17], which focuses on the behavior of the 2nd eigenvector of the Laplacian matrix, and crucially rely on a large eigengap between the second and third eigenvectors.

Our approximation results build on techniques for

Appearing in Proceedings of the 18^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2015, San Diego, CA, USA. JMLR: W&CP volume 38.

graph sparsification [14] [19], in which the task is to find a sparse approximation to a given graph G. This is somewhat similar to our task, but with an important difference: We do not have access to the full graph, whereas in graph sparsification the graph is given, and this full knowledge is used by algorithms, e.g. using the sum of edge weights associated with each node.

We then consider how adaptive sampling may be used to reduce the number of edges queried. We extend our guarantees to adaptive sampling strategies, and design a generic framework as well as a new adaptive sampling algorithm for clustering (CLUS2K). Compared to previous approaches, the algorithm is much simpler and avoid a costly full eigen-decomposition at each iteration. We conclude by presenting experimental comparison to previous work and show that the proposed algorithm achieves equal or even better performance on a range of datasets.

2 A General Graph Approximation Guarantee

In this section we derive our general approximation theorem for spectral approximation. We will also present a lower bound proving that, for a family of graphs, the bound is tight up to a log factor.

We consider full graphs defined by a set of vertices V and a weight matrix W, with zero weights indicate a missing edge. We start with a few basic definitions.

Definition 2.1. *Let $G = (V, W)$ be a weighted graph and $S \subset V$ a subset of vertices, then the cut defined by S, $|\partial_G S|$, is the sum of all the weights of edges that have exactly one endpoint in S.*

Definition 2.2. *Let $G = (V, W)$ and $\tilde{G} = (V, \tilde{W})$ be two graphs on the same set of vertices. $\tilde{G}$ is an ϵ-cut approximation of G if for any $S \subset V$ we have $(1-\epsilon)|\partial_G S| \leq |\partial_{\tilde{G}} S| \leq (1+\epsilon)|\partial_G S|$*

Definition 2.3. *Let G be a weighted graph. The graph Laplacian L_G is defined as $L_G = D - W$ where D is a diagonal matrix with values $D_{ii} = \sum\limits_{1 \leq j \leq n} W_{ij}$. The normalized graph Laplacian $\mathcal{L}_G$ is defined as $\mathcal{L}_G = D^{-1/2}(D - W)D^{-1/2} = D^{-1/2} L_G D^{-1/2}$.*

The Laplacian holds important information about the graph [4]. In particular, the quadratic form defined by the Laplacian relates to the graph through the equation

$$x^T L_G x = \frac{1}{2} \sum_{i,j=1}^{n} W_{ij}(x_i - x_j)^2 \qquad (1)$$

When $x_i \in \{0, 1\}$ this is easily seen to be the value of the cut defined by x. Many spectral graph techniques, such as spectral clustering, can be seen as a relaxation of such a discrete problem to $x \in \mathbb{R}^n$.

Definition 2.4. *A graph $\tilde{G}$ is an $\epsilon-$spectral approximation of G if*

$$\forall x \in \mathbb{R}^n \quad (1-\epsilon) x^T L_{\tilde{G}} x \leq x^T L_G x \leq (1+\epsilon) x^T L_{\tilde{G}} x$$

We note that this is different than requiring $||L_G - L_{\tilde{G}}|| \leq \epsilon$ using the matrix 2-norm, as we can view it as a multiplicative error vs. an additive error term. In particular, it implies approximation of eigenvectors (using the min-max theorem [4]), which is relevant to many spectral algorithms, and includes the approximation of the 2nd eigenvector, the focus of the analysis in [17], as a special case. Moreover, it implies cut approximation via equation 1, and is in fact strictly stronger (see [19] for a simple example of a cut approximation which is not a spectral approximation). We will focus more on spectral approximation in our theoretical results.

Our initial approximation strategy will be to uniformly at random sample a subset $\tilde{E}$ of m edges, i.e. pick m edges without replacement and construct a graph $\tilde{G} = (V, \tilde{W})$ with weights $\tilde{w}_{ji} = \tilde{w}_{ij} = \frac{w_{ij}}{p}$ for any $(i, j) \in \tilde{E}$ and zero otherwise, where $p = m/\binom{n}{2}$ is the probability any edge is sampled. It is easy to see that $\mathbb{E}[\tilde{W}] = W$.

We begin by providing a bound on m which ensures an ϵ-spectral approximation. It is based on an adaptation of the work in [19], in which the author considered picking each edge independently. This differs from our setting, where we are interested in picking m edges without replacement, since in this case the probabilities of picking different edges are no longer independent. While this seems like a serious complication, it can be fixed using the notion of negative dependence:

Definition 2.5. *The random variables $X_1, ..., X_n$ are said to be* negatively dependent *if for all disjoint subset $I, J \subset [n]$ and all nondecreasing functions f and g, $\mathbb{E}[f(X_i, i \in I)g(X_j, j \in J)] \leq \mathbb{E}[f(X_i, i \in I)]\mathbb{E}[g(X_j, j \in J)]$.*

Intuitively, a group of random variables are negatively dependent if when some of them have a high value, the others are more probable to have lower values. If we pick m edges uniformly, each edge that has been picked lowers the chances of the other edges to get picked, so intuitively the probabilities are negatively dependent. The probabilities of picking edges have indeed been shown to be indeed negatively dependent in [17].

An important application of negative dependence is the Chernoff-Hoeffding bounds, which hold for sums of independent random variables, also hold for negatively dependent variables. See supplementary material for details.

We can now state the general spectral approximation theorem:

Theorem 2.1. *Let G be a graph with weights $w_{ij} \in [0,1]$ and $\tilde{G}$ its approximation after sampling m edges uniformly. Define λ as the second smallest eigenvalue of $\mathcal{L}_G$ and $k = \max\{\log(\frac{3}{\delta}), \log(n)\}$. If $m \geq \binom{n}{2}\frac{(12k/\epsilon\lambda)^2}{\min D_{ii}}$ then the probability that $\tilde{G}$ is not an ϵ-spectral approximation is smaller then δ.*

Proof outline. The proof is based on an adaptation of part of theorem 6.1 from [19]. The two main differences are that we use negative dependence instead of independence and a weighted graph instead of an unweighted graph. The proof uses the following lemma

Lemma 2.1. *Let $\mathcal{L}_G$ be the normalized Laplacian of G with second eigenvalue λ. If $||D^{-1/2}(L_G - L_{\tilde{G}})D^{-1/2}|| \leq \epsilon$ then $\tilde{G}$ is an σ-spectral approximation for $\sigma = \frac{\epsilon}{\lambda-\epsilon}$.*

The next part is to bound $||D^{-1/2}(L_G - L_{\tilde{G}})D^{-1/2}||$ using a modified version of the trace method [21]. See the supplementary material for more details. □

Stating theorem 2.1 in a simplified form, we have that if $\min D_{ii} = \Omega(n^{\alpha})$, then one gets an ϵ-approximation guarantee using $m = \mathcal{O}\left(n^{2-\alpha}\left(\frac{\log(n)+\log(1/\delta)}{\epsilon\lambda}\right)^2\right)$ sampled edges.

The main caveat of theorem 2.1 is that it only leads to a non-trivial guarantee ($m \ll n^2$) when $\alpha > 0$ and λ is not too small. Most algorithms, such as spectral clustering, assume that the graph has $k \geq 2$ relatively small eigenvalues, in the ideal case (more then one connected component) we even have $\lambda = 0$. We will now show that this is unavoidable, and that the bound above is essentially optimal, up to log factors, for graphs with bounded $\lambda > C > 0$, i.e. expanders.

Since spectral approximation implies cut approximation, we will use this to find simple bounds on the number of edges needed for both approximations. We will show that a necessary condition for any approximation is that the minimal cut is not too small, the intuition being that even finding a single edge for connectedness, on that cut can be hard, and get a lower bound on the number of samples needed. For this we will need the following lemma (which follows directly from the linearity of expectation)

Lemma 2.2. *Let X be a finite set, and $Y \subset X$. If we pick a subset Z of size m uniformly at random then $\mathbb{E}\left[|Z \cap Y|\right] = \frac{m\cdot|Y|}{|X|}$*

We will now use this to prove a lower bound on the number of edges sampled for binary weighted graphs (i.e. unweighted graph) $w_{ij} \in \{0,1\}$.

Theorem 2.2. *Let G be an binary weighted graph with minimal cut weight c>0. Assume $\tilde{G}$ was constructed by sampling m<$\binom{n}{2}\frac{(1-\delta)}{c}$ edges, then for any ϵ<1, the probability that $\tilde{G}$ is not an ϵ-cut approximation of G is greater then δ.*

Proof. Let Y be all the edges in a minimal cut and let $\tilde{E}$ be the edges sampled. Since the weights are binary, the weight of this cut in $\tilde{G}$ is the number of edges in $Y \cap \tilde{E}$. From lemma 2.2 we know that $\mathbb{E}\left[|Y \cap \tilde{E}|\right] = mc/\binom{n}{2} < 1-\delta$. From Markov's inequality we get that $P\left(|Y \cap \tilde{E}| \geq 1\right) < 1-\delta$. If $|Y \cap \tilde{E}| < 1$ then the intersection is empty and we do not have an ϵ-approximation for any $\epsilon < 1$ proving $P\left(\tilde{G} \text{ is an } \epsilon\text{-cut approximation of } G\right) < 1-\delta$. □

This theorem proves that in order to get any reasonable approximation with a small budget m (at least with uniform sampling) the original graph's minimal cut cannot be too small and that $\Omega(n^2/c)$ samples are needed. Comparing this to theorem 2.1 (noticing $\min_i D_{ii} \geq c$) we see that, for graphs with a lower bound on λ, sampling a logarithmic factor of this lower bound is sufficient to ensure not only a good cut approximation, but spectral approximation as well.

In the next section, we show how a few reasonable assumptions allows us to recover non-trivial guarantees even in the regime of small eigenvalues.

3 Clusterable Graphs

Clustering algorithms assume a certain structure of the graph. In general they assume k strongly connected components, the clusters, with weak connections between them. The precise assumptions vary from algorithm to algorithm. While this is a bad setting for approximation, as this normally means a small minimal cut, and for spectral clustering a small λ, we will show how the basic assumptions for clustering ensure approximation can be used on the inner-cluster graphs to obtain useful results. We provide two results, one geared towards spectral approximation, and the other towards cut approximation.

3.1 Spectral Clustering

In this section, we show how the eigenspace corresponding to the k clusters can be approximated, and give a tradeoff between the number of edges sampled and the error.

Definition 3.1. *Assume a graph $G = (V, W)$ consists of k clusters, define W^{in} as the block diagonal matrix consisting of the similarity scores between same-cluster elements.*

$$\mathbf{W^{in}} = \begin{bmatrix} \mathbf{W}^1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}^2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{W}^k \end{bmatrix}$$

and $W^{out} = W - W^{in}$ the off-diagonal elements.

Assumption 3.1. *Define $\lambda^{in} = \min\limits_{1 \le i \le k} \lambda_2(\mathcal{L}^i)$, the smallest of all the second normalized eigenvalues over all $\mathcal{L}^i = \mathcal{L}_{W^i}$. Assume $\lambda^{in} > C > 0$ for some constant C.*

Assumption 3.2. $\min D^{in}_{ii} = \Omega(n^\alpha)$ *where $D^{in}_{ii} = \sum_j W^{in}_{ij}$ for some $\alpha > 0$.*

Assumption 3.3. *Assume that $||W^{out}|| = \mathcal{O}(n^\beta)$ for some $\beta < \alpha$.*

Assumption 3.1 implies well connected clusters, while assumption 3.2 excludes sparse, well-connected graphs, which we have already shown earlier to be hard to approximate. Assumption 3.3 essentially requires the between-cluster connections to be relatively weaker than the within-cluster connections.

Under these assumptions, we can approximate the zero eigenspace of $L^{in} = D^{in} - W^{in}$ which corresponds to the k connected components, i.e. the clusters. More rigorously:

Theorem 3.1. *Let P be the zero eigenspace of $L^{in} = D^{in} - W^{in}$ corresponding to the k clusters, $\tilde{L}$ the Laplacian of the graph we get by sampling $m = \tilde{\mathcal{O}}(n^{2-\gamma})$ edges for $\beta \le \gamma \le \alpha$, and Q the space spanned by the first k eigenvectors of $\tilde{L}$. Under previous assumptions, $||\sin(\Theta(P, Q))|| = \mathcal{O}(\frac{n^\beta + n^\gamma}{n^\alpha})$.*

We simplified the statement in order not to get overwhelmed by notation. $\Theta(P, Q)$ is a diagonal matrix whose diagonal values correspond to the canonical angles between the subspaces P and Q, and $||\sin(\Theta(P, Q))||$ is a common way to measure distance between subspaces.

Proof outline. If $\gamma = 0$, i.e. Q was spanned by eigenvectors of the full L, then the theorem would be true by the sin-theta theorem [6] under our assumptions. We need to show that this theorem can be used with $\tilde{L}$. The sin-theta theorem states that $||\sin(\Theta(P, Q))|| \le \frac{||\tilde{L}^{out}||}{\mu_2}$ where $||\tilde{L}^{out}||$ the "noise" factor, and $\tilde{\mu}_2$ the unnormalized second eigenvalue of $\tilde{L}^{in}$ the "signal" factor. Using theorem 2.1 and our first two assumptions we can approximate each L_{W^i} and use to show that $\tilde{\mu}_2 = \Omega(n^\alpha)$. We now only need to show $||\tilde{L}^{out}|| = \mathcal{O}(n^\beta + n^\gamma)$. This can be done using the matrix Chernoff inequality [20], by applying a result in [12] that shows how it can be adapted to sampling without replacements. We note that the result in [12] is limited to sampling without replacements as negative dependence has no obvious extension to random matrices. For further details see the supplementary material □

This gives us a tradeoff between the number of edges sampled and the error. The theoretical guarantee from the sin-theta theorem for the complete graph is $\mathcal{O}(n^\beta / n^\alpha)$ so for $\gamma = \beta$ we have the same guarantee as though we had used we used the full graph. For n large enough one can get $||\sin(\Theta(P, Q))||$ as small as desired by using $\gamma = \alpha - \epsilon$.

3.2 Cut Clustering

Cut based clustering, such as [11], has a different natural notion of "clusterable". We will assume nothing on eigenvalues, making this more general than the previous section.

We will show that after a sufficient number of edges sampled, the cuts between clusters are smaller then cuts in clusters.

Assumption 3.4. *Assume G can be partitioned into k clusters, within which the minimal cut is at least c_{in}. Furthermore, assume that any cut separating between the clusters of G, i.e. not splitting same cluster elements, is smaller then c_{out}, and that $c_{in} > 4c_{out}$.*

These assumptions basically require the inner-cluster connections to be relatively stronger than between-cluster connections.

Theorem 3.2. *Let G be a graph with weights $w_{ij} \in [0,1]$ and $\tilde{G}$ its approximation after observing m edges. Under previous assumptions if $m = \tilde{\Omega}\left(\frac{n^2}{c_{in}} k \ln(\frac{1}{\delta})\right)$ then the cuts separating the clusters are smaller then any cut that cuts into one of the clusters.*

Proof outline. We can use cut approximation for the clusters themselves so $\tilde{c}_{in} \geq c_{in}/2$. Using the Chernoff bound and union bound for the 2^k cuts between clusters we get that none of them is greater then $c_{in}/2$. See the supplementary material for full proof. □

In the supplementary material, we provide a more in-depth analysis of cut approximation including an analog of theorem 2.1.

4 Adaptive Sampling and the Clus2K Algorithm

Theorem 2.2 states that, with uniform sampling and no prior assumptions on the graph structure, we need to sample at least $\Omega(n^2/c)$ edges where c is the weight of the smallest cut. What if we had an adaptive algorithm instead of just uniform sampling? It is easy to see that for some graphs the same lower bound holds, up to a constant. Consider a graph with $2n$ vertices, consisting of two cliques that have c randomly chosen edges connecting them. Further assume that an oracle told us which vertex is in which clique, so any sensible algorithm would sample only edges connecting the cliques. As the edges are random, it would take $\Theta\left(n^2/c\right)$ tries just to hit one edge needed for any good approximation. Nevertheless, in some cases an adaptive scheme can reduce the number of samples needed, as we now turn to discuss it in the context of clustering.

Consider a similar toy problem - we have a graph which is known to consist of two connected components, each a clique of size n, and we wish to find these clusters. We can run the uniform sampling algorithm until we have only two connected components and return them. How many edges will we need to sample until we get only two connected components? If we look only at one clique, then basic results in random graph theory [8] show that with high probability, the number of edges added before we get a connected graph is $\Theta(n\log(n))$ which lower bounds the number of samples needed. To improve on this we can use an adaptive algorithm with the following scheme: at each iteration, pick an edge at random connecting the smallest connected component to some other connected component. At each step we have at least a probability of $\frac{1}{3}$ to connect two connected components. This is because there are n nodes in the wrong cluster, and at least $\frac{n}{2}$ in the right cluster (since we pick the smallest connected component). Therefore with high probability the number of steps needed to decrease the number of connected components from $2n$ to two is $\Theta(n)$.

This argument leads us to consider adaptive sampling schemes, which iteratively sample edges according to a non-uniform distribution. Intuitively, such a distribution should place more weight on edges which may be more helpful in approximating the structure of the original graph. We first discuss how we can incorporate *arbitrary* non-uniform distributions into our framework. We then propose a specific non-uniform distribution, motivated by the toy example above, leading to a new algorithm for our setting in the context of clustering.

One approach to incorporate non-uniform distributions is by unbiased sampling, where we re-scale the weights according to the sampling probability. This means that the weights are unbiased estimates of the actual weights. One can show that whatever the non-uniform distribution, a simple modification (adding with probability half a uniform sample) suffices for cut approximation to hold. Unfortunately, we found this approach to work poorly in practice, as it was unstable and oscillated between good and bad clustering long after a good clustering is initially found.

Due to these issues, we considered a *biased* sampling approach, where we mix the non-uniform distribution with a uniform distribution (as proposed earlier) on unseen edges, but do not attempt to rescale weights. More specifically, consider any adaptive sampling algorithm which picks an unseen edge at step $i+1$ with probability $p(e;\tilde{G}_i)$ that depends on the graph $\tilde{G}_i$ seen so far. We will consider a modified distribution that with probability 0.5 picks an unseen edge uniformly, and with probability 0.5 picks it according to $p(e;\tilde{G}_i)$.

4.1 Adaptive biased sampling

While biased sampling can ruin approximation guarantees, we show similar results to theorem 3.2 (under stronger conditions) for any adaptive sampling scheme.

First, note that for a specific known graph one can always design a bad biased sampling scheme. Consider an adversarial scheme that always samples the largest weight edge between two constant clusters, it is easy to see that this can lead to bad cut clustering. To circumvent this we will consider graphs where the edge weights between the clusters, which we regard as noise, are picked randomly.

Assumption 4.1. *Assume G can be partitioned into k clusters of size $\Omega(n)$, within which the minimal cut is at least $c_{in} = \Omega(n^{\alpha})$.*

Assumption 4.2. *Assume that the weights of edges between the clusters are 0, besides $c_{out} = o(n^{\alpha})$ edges chosen uniformly at randomly (without replacement) between any two clusters that have weight 1.*

Theorem 4.1. *Let $\tilde{G}$ be the graph after sampling $m = \tilde{\Omega}\left(n^{2-\beta} k \ln(\frac{1}{\delta})\right)$ edges without replacements (with probability $1/2$ of sampling uniformly) with $\beta < \alpha$. Let $\tilde{c}_{in}$ and $\tilde{c}_{out}$ be the minimal cut weight inside any cluster and the maximal cut weight between clusters, under previous assumptions the probability that $\tilde{c}_{in} < \tilde{c}_{out}$ is smaller then δ.*

Proof. *Using cut approximation theorem (**??** in the supplementary material) on the edges sampled uniformly (remembering that the biased sampling can only increase the cut weight) we get that with probability greater then $\delta/2$, $\tilde{c}_{in} = \tilde{\Omega}\left(\frac{m}{n^2} c_{in}\right) = \tilde{\Omega}(n^{\alpha-\beta})$. If we consider the weight of any cut between clusters, then the key observation is that because the edges are picked uniformly at random, then whatever the algorithm does is equivalent to running a uniform sampling of a constant edge set. We then get that the expected minimal cut weight is $\tilde{\mathcal{O}}(\frac{m \cdot c_{out}}{n^2}) = o(n^{\alpha-\beta})$ using lemma 2.2 (the upper bound is by looking as if all edges where picked from this cut). We can now use the Markov inequality to show $P(\tilde{c}_{out}/\tilde{c}_{in} < 1) = \frac{o(n^{\alpha-beta})}{\Omega(n^{\alpha-\beta})} < \delta/2$.*

It is simple to generalize this theorem to any uniform weighting that has $o(c_{out})$ expected cut weights.

4.2 CLUS2K Algorithm

We now turn to consider a specific algorithmic instantiation, in the context of clustering. Motivated by the toy example presented earlier, we consider a non-uniform distribution which iteratively attempts to connect clusters in the currently-observed graph, by picking edges between them. These clusters are determined by the clustering algorithm we wish to use on the approximated graph, and are incrementally updated after each iteration. Inspired by the common practice in computer vision of over-segmentation, we use more clusters than the desired number of clusters k ($2k$ in our case). Moreover, as discussed earlier, we mix this distribution with a uniform distribution. The resulting algorithm, which we denote as CLUS2K, appears as Algorithm 1 below.

Algorithm 1 CLUS2K

```
Input: budget $b$, number of clusters $k$
Initialize: $S = \{(i,j) \in \{1,...,n\}^2 : i < j\}$, $\tilde{W}$ the zero matrix.
for $t = 1, ..., b$ do
    With probability 1/2 pick $(i,j) \in S$ uniformly;
    Otherwise:
        $C_1, ..., C_{2k} \leftarrow$cluster $\tilde{W}$ into $2k$ clusters;
        pick two distinct clusters $C_l$ and $C_m$ uniformly at random;
        pick $(i,j) \in S$ connecting $C_l$ and $C_m$ uniformly at random;
    Set $\tilde{w}_{ij} = \tilde{w}_{j,i} = w_{ij}; S = S \backslash (i,j)$;
end for
```

For the setting of budget-constrained clustering, the two most relevant algorithms we are aware of are the algorithm of [17] (hereby denoted as S&T), and the IU_RED algorithm of [9]. These algorithms are somewhat similar to our approach, in that they interleave uniform sampling and a non-uniform sampling scheme. However, the sampling scheme is very different than ours and focuses on finding the edge to which the derivative of the 2nd Laplacian eigenvector is most sensitive. This has two drawbacks. First, it is specifically designed for spectral clustering and the case of $k = 2$ clusters, which is based on the 2nd Laplacian eigenvector. Extending this to more than 2 clusters requires either recursive partitioning (which can be suboptimal), or considering sensitivity w.r.t. $k - 1$ eigenvectors, and it is not clear what is the best way to do so. Second, computing eigenvector derivatives requires a full spectral decomposition at each iteration, which can be quite costly or impractical for large matrices. In contrast, our algorithm does not compute derivatives. Therefore, when used with spectral clustering methods, which require only the smallest $2k$ eigenvectors, we have a significant improvement.

It is possible to speed up implementation even further, in the context of spectral clustering. Since only a single edge is added per iteration, one can use the previously computed eigenvectors as an initial value for fast iterative eigenvector solvers (although restarting every couple of steps is advised). Another possible option is to pick several edges from this distribution at each step, which makes this process parallelizable.

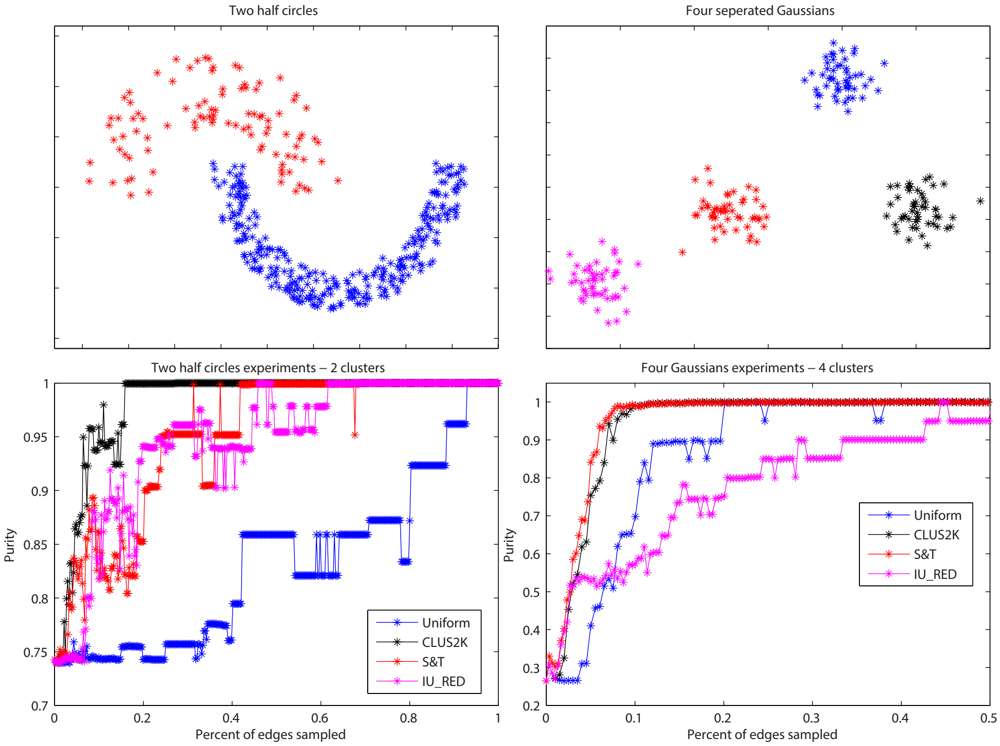

Figure 1: Synthetic datasets results

We note that generalizing theorem 3.1 for any adaptive sampling algorithm is impossible, but we will give intuition as to why the CLUS2K algorithm preserves spectral clustering.

The reason spectral clustering can fail is if $||\tilde{L}^{out}||$ is too large. Using the Gershgorin circle theorem we can show that if each node is sampled $\theta(m/n)$ times then the biased samples do not add a large deviation to $||\tilde{L}^{out}||$. As long as the clusters (at least for most of the running time) are of size $\theta(n/(2k)$ this holds and we will get the same guarantees as uniform sampling. In practice, much better performance results are achieved.

5 Experiments

We tested our CLUS2K algorithm on several datasets, and compared it to the S&T and IU_RED discussed earlier (other alternatives were tested in [17] and shown to be inferior). It is important to note that S&T and IU_RED were designed specifically for $k = 2$ and spectral clustering using the unnormalized Laplacian L_G, while we also tested for various values of k, and using the normalized Laplacian $\mathcal{L}_G$ [18] as well . The IU_RED performed badly (perhaps because it relies substantially on the $k = 2$ assumption) in these cases while S&T performed surprisingly well (yet still inferior to CLUS2K).

Clustering was measured by cluster purity, a standard measure for clustering performance . The purity of a single cluster is the percent of the most frequent class in the cluster. The purity of a clustering is a weighted average of its single cluster purity, weighted by the number of elements in each cluster.

As all algorithms are random, we ran each experiment 5 times and averaged the purity over the runs.

5.1 Synthetic Data

The synthetic experiments were performed on two datasets - the two half circles dataset, and a dataset comprising of four well separated Gaussians. Both experiments used unnormalized spectral clustering (see figure 5.1) using a gaussian weight matrix $w_{ij} = \exp(-||x_i - x_j||/t)$. The two half circles is a classic clustering dataset with $k = 2$ clusters. The Gaussian dataset shows how the various algorithms handle an easy $k > 2$ dataset. The IU_RED performs worse than uniform sampling in this case.

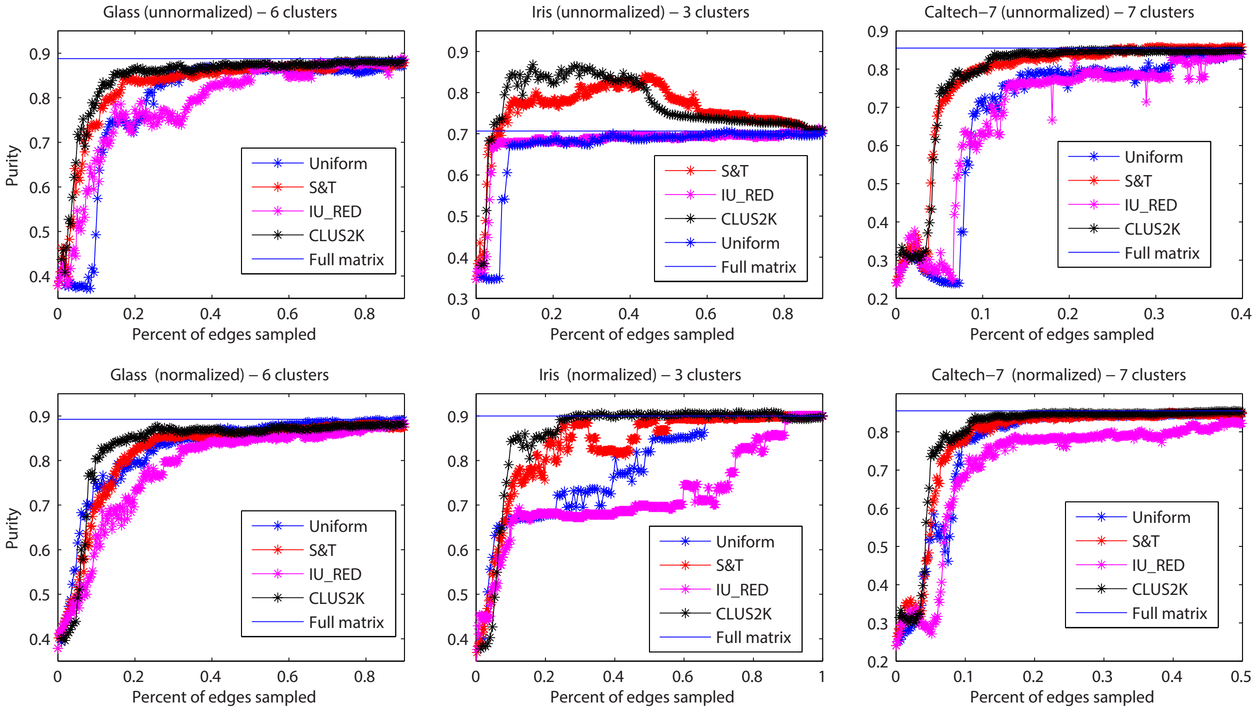

Figure 2: UCI datasets results

5.2 Real Data

We tested on three datasets - the iris and glass UCI datasets, both with k>2 clusters, using a Gaussian weight matrix and the Caltech-7 dataset, a subset of the Caltech-101 images datasets with 7 clusters gathered by [15], using the similarity matrix suggested by [10]. We tested each dataset using both the normalized and unnormalized Laplacian for clustering. The results are presented in figure 5.1

Overall, the experiments show that the `CLUS2K` algorithm performs as good as or better than previous algorithms for budget-constrained clustering, while being significantly computationally cheaper as it avoids doing a full eigen-decomposition.

6 Summary

We have shown that well connected graphs can be approximated by uniform sampling and we derived a tight (up to log factor) bound on the number of edges needed. We later showed that while clusterable graphs are not well connected, their structure suffices to ensure that the clusters can be retrieved while sampling a relative small number of edges. We discussed how adaptive sampling can lower the number of edges sampled, and we introduced a new adaptive sampling algorithm the `CLUS2K` algorithm. This algorithm performs as well as or superior to previous algorithms on various datasets while being computationally cheaper and can scale on larger graphs.

Acknowledgements: OS was supported by an Israel Science Foundation grant 425/13 and an FP7 Marie Curie CIG grant. SU was supported by ERC Advanced Grant Digital Baby and by the Center for Minds, Brains and Machines (CBMM), funded by NSF STC award CCF-1231216.

References

[1] Y. Boykov and O. Veksler. Graph cuts in vision and graphics: Theories. *Handbook of Mathematical Models of Computer Vision*, 2006.

[2] M. I. Jordan A. Y. Ng and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2001.

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. 2001.

[4] F. R. K Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

[5] C. Gollan D. Keysers, T. Deselaers and H. Ney. Deformation models for image recognition. *Transactions on Pattern Analysis and machine Intelligence*, 2007.

[6] C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. iii. *Journal on Numerical Analysis*, 1970.

[7] D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 2012.

[8] P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae*, pages 290–297, 1959.

[9] N. Jojic F. Wauthier and M. Jordan. Active spectral clustering via iterative uncertainty reduction. *SIGKDD Conference on Knowledge Discovery and Data Mining*, 2012.

[10] A. Faktor and M. Irani. clustering by composition unsupervised discovery of image categories. *European Conference on Computer Vision*, 2013.

[11] R. Tarjan G. Flake and K.Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 2003.

[12] D. Gross and V. Nesme. Note on sampling without replacing from a finite collection of matrices. *CoRR, abs/1001.2738*, 2010.

[13] D. Karger. Global min-cuts in $\mathcal{RNC}$ and other ramifications of a simple mincut algorithm. In *SODA*, 1993.

[14] D. R. Karger. using randomized sparsification to approximate minimum cuts. *In Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 424–432, 1994.

[15] Y.J. Lee and K. Grauman. Foreground focus: Unsupervised learning from partially matching. *International Journal of Computer Vision*, 2009.

[16] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *NIPS*, 2005.

[17] O. Shamir and N. Tishby. Spectral clustering on a budget. *Journal of Machine Learning Research*, 2011.

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. *Transactions on Pattern Analysis and Machine Intelligance*, 2000.

[19] D. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40, 2011.

[20] J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 2012.

[21] V. Vu. Spectral norm of random matrices. *Combinatorica,*, 2007.

[22] O. Veksler Y. Boykov and R. Zabih. Fast approximate energy minimization via graph cuts. *Transactions on Pattern Analysis and machine Intelligence*, 2001.