Fast Approximate Spectral Clustering for Dynamic Networks

Lionel Martin [1] **Andreas Loukas** [1] **Pierre Vandergheynst** [1]

Abstract

Spectral clustering is a widely studied problem, yet its complexity is prohibitive for dynamic graphs of even modest size. We claim that it is possible to reuse information of past cluster assignments to expedite computation. Our approach builds on a recent idea of sidestepping the main bottleneck of spectral clustering, i.e., computing the graph eigenvectors, by a polynomial-based randomized sketching technique. We show that the proposed algorithm achieves clustering assignments with quality approximating that of spectral clustering and that it can yield significant complexity benefits when the graph dynamics are appropriately bounded. In our experiments, our method clusters 30k node graphs 3.9× faster in average and deviates from the correct assignment by less than 0.1%.

1. Introduction

Spectral clustering (SC) is one of the most utilized methods for clustering multivariate data (Von Luxburg, 2007; Fortunato, 2010). However, because of its inherent dependence on the spectrum of some large graph, SC is computationally expensive. Let n and k be the number of nodes and clusters, respectively. Clustering a graph takes $O(n^3)$ operations if a full eigendecomposition is performed and $O(kn^2)$ if the Lanczos method is used. This has motivated a surge of research focusing in reducing its complexity, for example using matrix sketching (Fowlkes et al., 2004; Li et al., 2011; Gittens et al., 2013), coarsening (Loukas and Vandergheynst, 2018), and compressive sampling (Ramasamy and Madhow, 2015; Tremblay et al., 2016), attaining a complexity reduction by roughly a factor of n.

Yet, computation is still an issue for dynamic networks, where the edge set is a function of time. Temporal dynamics constitute an important aspect of many network datasets and should be taken into account in the algorithmic design and analysis. Unfortunately, SC is poorly suited to this setting as eigendecomposition –its main computational bottleneck– has to be recomputed from scratch whenever the graph is updated, or at least periodically (Ning et al., 2007). This is a missed opportunity since the clustering assignments of many real networks change slowly with time, suggesting that successive algorithmic runs wastefully repeat similar computations.

This paper proposes an algorithm that reuses information of past cluster assignments to expedite computation. Different from previous work on dynamic clustering, our objective is *not* to improve the clustering quality, e.g., by enforcing a temporal-smoothness hypothesis (Chakrabarti et al., 2006; Chi et al., 2007) or by using tensor decompositions (Gauvin et al., 2014; Tu et al., 2016). Similar to recent work by (Dhanjal et al., 2014), we focus entirely on reducing the complexity while producing assignments that are *provably* close to those of SC.

Our work starts from the recent idea of sidestepping eigendecomposition by utilizing as features random vectors filtered by Chebyshev polynomials of the graph Laplacian (Tremblay et al., 2016). We notice that, in the dynamic setting, there are ample opportunities to reuse information from previous cluster assignments, both in terms of approximating the k-th eigenvalue (a necessary step of Chebyshev filter design), as well as in terms of computing the features themselves. When the consecutive graphs are appropriately similar, these ideas lead to complexity reductions.

Concretely, we provide the following contributions:

1. In Section 3 we refine the analysis of compressive spectral clustering (CSC) presented in (Tremblay et al., 2016). Our goal is to move from assertions about feature approximation to guarantees about the quality of the solution of CSC itself. We prove that w.h.p. the quality of the clustering assignments of CSC and SC differ by $O(2k/\sqrt{d})$, and thus $d \propto k^2$ filtered vectors are sufficient to obtain a good approximation. Importantly, our analysis does not make restricting assumptions about the graph structure, such as assuming a stochastic block model (Pydi and Dukkipati, 2017).

2. In Section 4, we focus on dynamic graphs and propose dynamic CSC (dCSC), an algorithm that reuses informa-

[1]École Polytechnique Fédérale de Lausanne, Switzerland. Correspondence to: Lionel Martin <lionel.martin@epfl.ch>, Andreas Loukas <andreas.loukas@epfl.ch>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

tion of past cluster assignments to expedite computation. We discover that the algorithm's ability to reuse features is determined by a measure of spectral similarity ρ between consecutive graphs: we prove that, when pd features are reused (i.e., each new instance of the dynamic graph is clustered using pd features of the previous graph and $(1-p)d$ new features, where $0 < p \leq 0.5$), w.h.p. the clustering quality of dCSC approximates that of CSC up to an additive term in the order of $p\rho$.

The paper concludes with a proof of concept comparison against SotA approximation algorithms for Spectral Clustering (Section 5). Our experiments confirm that dCSC yields computational benefits when the graph dynamics are bounded. A case in point: we can cluster 30'000 node graphs 3.9× faster than SC and 1.5× faster than CSC in average. Due to space constraints, certain proofs and implementation details are presented as an appendix in a supplementary document.

2. Background

We start by summarizing the standard method for spectral clustering as well as the idea behind the more recent accelerated methods. Due to space constraints, our exposition is brief; the reader is encouraged to refer to the original works for a more comprehensive discussion.

2.1. Spectral clustering (SC)

To determine the best node-to-cluster assignment, spectral clustering solves a k-means problem with the eigenvectors of the graph Laplacian as features (Shi and Malik, 2000; Ng et al., 2002).

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ be a weighted undirected graph with n nodes $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, and m edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. The graph Laplacian is defined as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is a diagonal matrix whose entries are the degree of the nodes in the graph (i.e. the sum of the weighed edges adjacent to each node). We denote the eigendecomposition of the Laplacian by $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, with the eigenvalues contained in $\mathbf{\Lambda}$ sorted in non-decreasing order, such that $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

Spectral clustering consists of computing the first k eigenvectors of $\mathbf{L}$ arranged in matrix $\mathbf{U}_k$ and subsequently computing a k-means assignment of the n vectors of size k found in the rows of $\mathbf{U}_k$. Formally, if $\mathbf{\Phi} \in \mathbb{R}^{n\times d}$ is the feature matrix (here $\mathbf{\Phi} = \mathbf{U}_k$ and $d = k$), and k is a positive integer denoting the number of clusters, the k-means clustering problem finds the indicator matrix $\mathbf{X} \in \mathbb{R}^{n\times k}$ which satisfies

$$\mathbf{X}_\Phi = \arg\min_{\mathbf{X}\in\mathcal{X}} \|\mathbf{\Phi} - \mathbf{X}\mathbf{X}^\top\mathbf{\Phi}\|_F, \tag{1}$$

with associated cost $C_\Phi = \|\mathbf{\Phi} - \mathbf{X}_\Phi\mathbf{X}_\Phi^T\mathbf{\Phi}\|_F$. Symbol $\mathcal{X}$ denotes the set of all $n \times k$ indicator matrices $\mathbf{X}$. These matrices indicates the cluster membership of each data point by setting

$$\mathbf{X}_{i,j} = \begin{cases} \frac{1}{\sqrt{s_j}} & \text{if data point } i \text{ belongs to cluster } j \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where s_j is the size of cluster j, also equals to the number of non-zero elements in column j. Note that the cost described in eq. (1) is the square root of the more traditional definition expressed with the distances to the cluster centers (Cohen et al., 2015, Sec 2.3). We refer the reader to the work by (Boutsidis et al., 2015) and references therein for more details.

2.2. Compressive spectral clustering (CSC)

To reduce the computational cost of spectral clustering, (Tremblay et al., 2016) proposed to approximate $\mathbf{U}_k$ using a filtering of random vectors (a similar idea was also examined by (Boutsidis et al., 2015)). The former work also introduced the benefits of compressed sampling techniques reducing the total cost down to $O(k^2\log^2(k) + cn(\log(n) + k))$, where c is the order of the polynomial approximation. Their algorithm consists of two steps:

Step 1. Approximate features. Feature matrix $\mathbf{U}_k$ is approximated by the projection of a random matrix over the same subspace. In particular, let $\mathbf{R} \in \mathbb{R}^{N\times d}$ be a random (gaussian) matrix with centered i.i.d. entries, each having variance $\frac{1}{d}$. We can project $\mathbf{R}$ onto *span*$\{\mathbf{U}_k\}$ by multiplying each one of its columns by a projector $\mathbf{H}$ defined as

$$\mathbf{H} = \mathbf{U}\begin{pmatrix} \mathbf{I}_k & 0 \\ 0 & 0 \end{pmatrix}\mathbf{U}^\top. \tag{3}$$

It is then a simple consequence of the Johnshon-Lindenstrauss lemma that the rows $\boldsymbol{\psi}_i^\top$ of matrix $\mathbf{\Psi} = \mathbf{H}\mathbf{R}$ can act as a replacement of the features used in spectral clustering, i.e., the rows $\boldsymbol{\phi}_i^\top$ of $\mathbf{\Phi} = \mathbf{U}_k$.

Theorem 2.1 (adapted from (Tremblay et al., 2016))**.** *For every two nodes v_i and v_j the restricted isometry relation*

$$(1-\varepsilon)\|\boldsymbol{\phi}_i - \boldsymbol{\phi}_j\|_2 \leq \|\boldsymbol{\psi}_i - \boldsymbol{\psi}_j\|_2 \leq (1+\varepsilon)\|\boldsymbol{\phi}_i - \boldsymbol{\phi}_j\|_2 \tag{4}$$

holds with probability larger than $1 - n^{-\beta}$, as long as the dimension is $d > \frac{4+2\beta}{\varepsilon^2/2 - \varepsilon^3/3}\log(n)$.

We note that, even though $\mathbf{HR}$ is also expensive to compute exactly, it can be easily approximated by applying the graph filter $h(\mathbf{L})$ on each column of $\mathbf{R}$, which entails $O(dc)$ sparse matrix-vector multiplications (each costing $O(m)$) using graph Chebychev polynomials (Shuman et al., 2011a; Hammond et al., 2011) or rational graph filters (Isufi et al., 2017; Loukas et al., 2015) (c relates to the quality of the

approximation and is usually below 100). A more elaborate discussion on the approximation of $\mathbf{HR}$ can be found in the appendix.

Step 2. Compressive k-means. The complexity is reduced further by computing the k-means step for only a subset of the nodes. The remaining cluster assignments are then inferred by solving a graph Tikhonov regularized interpolation problem involving k additional graph filtering operations, each with a cost linear in cm. To guarantee a good approximation, it is sufficient to sample $O(k\log(k))$ nodes using variable density sampling (Puy et al., 2016). For simplicity, in the following, we present our theoretical results w.r.t. the non-compressed version of their algorithm. The proofs can be generalized using similar arguments as in (Tremblay et al., 2016).

3. The Approximation Quality of Static CSC

Before delving to the dynamic setting, we refine the analysis of compressive spectral clustering. Our objective is to move from assertions about distance preservation currently known (see Thm. 2.1) to guarantees about the quality of the solution of CSC itself. Formally, let

$$\mathbf{X}_\Psi = \arg\min_{\mathbf{X}\in\mathcal{X}} \|\mathbf{\Psi} - \mathbf{X}\mathbf{X}^\top\mathbf{\Psi}\|_F \tag{5}$$

be the clustering assignment obtained from using k-means with $\mathbf{\Psi}$ as features (CSC assignment), and define the *CSC cost* C_Ψ as

$$C_\Psi = \|\mathbf{\Phi} - \mathbf{X}_\Psi\mathbf{X}_\Psi^\top\mathbf{\Phi}\|_F. \tag{6}$$

The question we ask is: *how close is C_Ψ to the cost C_Φ of the same problem, where the assignment has been computed using $\mathbf{\Phi}$ as features, i.e., the SC cost corresponding to (1)?* Note that, as in previous work (Boutsidis et al., 2015), we express the approximation quality in terms of the difference of clustering assignment costs and not of the distance between the assignments themselves. We are not aware of any analysis that would allow us to characterize (the perhaps more intuitive goal of) how well $\mathbf{X}_\Psi$ approximates $\mathbf{X}_\Phi$, which is a combinatorial objective. Yet, our approach exhibits the benefit of not penalizing approximation algorithms that choose alternative assignments of the same or similar quality[1].

Our central theorem asserts that with high probability the assignments of SC and CSC have similar costs.

Theorem 3.1. *The SC cost C_Φ and the CSC cost C_Ψ are related by*

$$C_{\mathbf{\Phi}} \le C_\Psi \le C_{\mathbf{\Phi}} + 2\sqrt{\frac{k}{d}}(\sqrt{k}+\varepsilon), \tag{7}$$

with probability at least $1-\exp(-\varepsilon^2/2)$.

This result emphasizes the importance of the number of random vectors d and directly links it to the distance with the optimal assignment for the spectral features. Indeed, one can see that the difference between the two costs vanishes when d is sufficiently large. Importantly, $d \propto k^2$ is sufficient to guarantee a small error.

3.1. The approximation quality of CSC

The first step in proving Thm. 3.1 is to establish the relation between C_Φ and C_Ψ. The following lemma relates the two costs by an additive error term that depends on the feature's differences $\|\mathbf{\Psi} - \mathbf{\Phi}\mathbf{I}_{k\times d}\mathbf{Q}\|_F$ when $d \ge k$.Since $\mathbf{\Phi}$ and $\mathbf{\Psi}$ have different sizes we introduced the multiplication by a unitary matrix $\mathbf{Q}$. We will first show that any unitary $\mathbf{Q}$ can be picked in Lem. 3.1 and then derive the optimal $\mathbf{Q}$, the one minimizing the additive term, in Thm. 3.2.

Lemma 3.1. *For any unitary matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$, the SC cost C_Φ and the CSC cost C_Ψ are related by*

$$C_\Phi \le C_\Psi \le C_\Phi + 2\|\mathbf{\Psi} - \mathbf{\Phi}\mathbf{I}_{k\times d}\mathbf{Q}\|_F, \tag{8}$$

where, the matrix $\mathbf{I}_{\ell\times m}$ of size $\ell \times m$ above contains only ones on its diagonal and serves to resize matrices.

Being able to show that the additive term is small encompasses the result of Thm. 2.1, ensuring distance preservation. However, this statement is stronger than the previous one as our lemma is not necessarily true under distance preservation only.

The remaining of this section is devoted to bounding the Frobenius error $\|\mathbf{\Psi} - \mathbf{\Phi}\mathbf{I}_{k\times d}\mathbf{Q}\|_F$ between the features of SC and CSC. In order to prove this result, we will first express our Frobenius norm exclusively in terms of the singular values of the random matrix $\mathbf{R}$ and then in a second step we will study the distribution of these singular values.

Our next result, which remarkably is an equality, reveals that the achieved error is exactly determined by how close a Gaussian matrix is to a unitary matrix.

Theorem 3.2. *There exists a $d\times d$ unitary matrix $\mathbf{Q}$, such that*

$$\|\mathbf{\Psi} - \mathbf{\Phi}\mathbf{I}_{k\times d}\mathbf{Q}\|_F = \|\mathbf{\Sigma} - \mathbf{I}_{k\times d}\|_F, \tag{9}$$

where $\mathbf{\Sigma}$ is the diagonal matrix holding the singular values of $\mathbf{R}' = \mathbf{I}_{k\times n}\mathbf{U}^\top\mathbf{R}$.

Before presenting the proof, let us observe that $\mathbf{R}'$ is an i.i.d. Gaussian random matrix of size $k\times d$ and its entries have zero mean and the same variance as that of $\mathbf{R}$. We use

[1] The k-means objective is a non convex objective and has multiple minima. For instance, any of the $k!$ re-labelings of the optimal assignment are valid solutions with the same cost.

this fact in the following to control the error by appropriately selecting the number of random vectors d.

To bound the feature error further, we will use the following result by Vershynin, whose proof is not reproduced.

Corollary 3.1 (adapted from Cor. 5.35 (Vershynin, 2010))**.** *Let $\mathbf{N}$ be an $d \times k$ matrix whose entries are independent standard normal random variables. Then for every $\varepsilon, i \geq 0$, with probability at least $1 - \exp(-\varepsilon^2/2)$ one has*

$$\sigma_i(\mathbf{N}) - \sqrt{d} \leq \sqrt{k} + \varepsilon, \tag{10}$$

where $\sigma_i(\mathbf{N})$ is the ith singular value of $\mathbf{N}$.

Exploiting this result, the following corollary of Thm. 3.2 reveals the relation of the feature error and the number of random vectors d.

Corollary 3.2. *There exists a $d \times d$ unitary matrix $\mathbf{Q}$, such that, for every $\varepsilon \geq 0$, one has*

$$\|\mathbf{\Psi} - \mathbf{\Phi}\mathbf{I}_{k\times d}\mathbf{Q}\|_F \leq \sqrt{\frac{k}{d}}\,(\sqrt{k} + \varepsilon), \tag{11}$$

with probability at least $1 - \exp(-\varepsilon^2/2)$.

Finally, Cor. 3.2 combined with Lem. 3.1 provide the direct proof of Thm. 3.1 that we introduced earlier.

Before proceeding, we would like to make some remarks about the tightness of the bound. First, guaranteeing that the feature error is small is a stronger condition than distance preservation (though necessary for a complete analysis of CSC). For this reason, the bound derived can be larger than that of Thm. 2.1. Nevertheless, we should stress it is tight: the main inequality in our analysis stems from bounding the k largest singular values of the random matrix by Vershynin's tight bound of the maximal singular value.

3.2. Practical aspects

The study presented above assumes that $\mathbf{H}$ is defined as in eq. (3), namely that it is a projector on the subspace spanned by the first k eigenvectors of $\mathbf{L}$. However, as discussed in the appendix, to be computationally efficient we choose to compute $\mathbf{H}$ by an application of a polynomial function h on $\mathbf{L}$ (Shuman et al., 2011a). More specifically, we select a polynomial that approximates the ideal low-pass response (Allen-Zhu and Li, 2016). As long as λ_k is known, the approximated projector $h(\mathbf{L})$ can be designed to be very close to $\mathbf{H}$: using the arguments of (Shuman et al., 2011b, Proposition 3) and (Laurent and Massart, 2000, Lemma 1) it is easy to prove that w.h.p. using $h(\mathbf{L})$ instead of $\mathbf{H}$ does not add more than $O(c^{-c}\sqrt{n})$ error to $C_{\mathbf{\Psi}}$, where c is the polynomial order (the proof is omitted due to space limitations).

Moreover, we need to estimate λ_k accurately (the design of h involves finding a polynomial which takes the value 1 when the input is smaller than λ_k and 0 otherwise). Towards this goal, we refer the readers to (Di Napoli et al., 2016; Paratte and Martin, 2016) and their respective *eigencount* techniques that allow to approximate the filter in $O(cm\log((\lambda_{k+1} - \lambda_k)^{-1}))$ operations.

4. Compressive Clustering of Dynamic Graphs

In this section, we consider the problem of spectral clustering a sequence of graphs. We focus on graphs $\mathcal{G}_t$ where $t \in \{1, \ldots, \tau\}$, composed of a static node set $\mathcal{V}$ and evolving edge sets $\mathcal{E}_t$. Identifying each assignment from scratch (using SC or CSC) is a computationally demanding task, as the complexity increases linearly with the number of time-steps. However, when consecutive graphs are "appropriately similar", we should be able to cut on this cost by reusing information. We will utilize two alternative similarity measures:

Definition 4.1 (Measures of graph similarity)**.** *Two graphs $\mathcal{G}_{t-1}$ and $\mathcal{G}_t$ are:*

- *(ρ, k)-**spectrally similar** if the spaces spanned by their first k eigenvectors are almost aligned: $\|\mathbf{H}_t - \mathbf{H}_{t-1}\|_F \leq \rho$.*
- *ρ-**edge similar** if the edge-wise difference of their Laplacians is bounded: $\|\mathbf{L}_t - \mathbf{L}_{t-1}\|_F \leq \rho$.*

Both measures are relevant in the context of dynamic clustering. Two spectrally similar graphs might have different connectivity, but possess similar clustering assignments. On the other hand, assuming that two graphs are edge similar is a stronger condition that postulates fine-grained similarities between them. It is however more intuitive and computationally economical to ascertain (see Section 4.3).

4.1. Algorithm

We now present an accelerated method for spectral clustering an evolving network. Without loss of generality, suppose that we need to compute the assignment for $\mathcal{G}_t$ while knowing already that of $\mathcal{G}_{t-1}$ and possessing the features $\mathbf{\Psi}_{t-1}$ used to compute it.

Component 1. We reuse a portion of features $\mathbf{\Psi}_{t-1}$ to cluster $\mathcal{G}_t$. Let p be a number between 0 and 0.5, and set $q = 1 - p$.[2] Instead of recomputing $\mathbf{\Psi}_t$ from scratch running a new CSC routine, we construct a feature matrix $\mathbf{\Theta}_t$ which consists of dq new features (corresponding to $\mathcal{G}_t$)

[2] Although in practice p can go up to 1, the analysis only considers the case when reused features are only from $\mathcal{G}_{t-1}$ (and not from earlier graphs).

Algorithm 1 Dynamic CSC

Input: $(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_\tau), p, d$
Output: $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_\tau)$

Determine λ_k and filter h_1 for $\mathcal{G}_1$.
Find $\mathbf{X}_1$ for $\mathcal{G}_1$ using CSC with $\mathbf{\Psi}_1 = h_1(\mathbf{L}_1)\mathbf{R}$.
for t from 2 to τ **do**
  Select dp features from $\mathbf{\Psi}_{t-1}$ and call them $\mathbf{\Psi}_t^{(1)}$.
  Generate $d(1-p)$ features $\mathbf{\Psi}_t^{(2)}$ by filtering as many random vectors by $h_{t-1}(\mathbf{L}_t)$.
  Test whether $\lambda_k \in [\lambda_k(\mathbf{L}_t), \lambda_{k+1}(\mathbf{L}_t)]$ with eigencount and using $\mathbf{\Psi}_t^{(2)}$.
  if the test fails **then**
    Determine λ_k for $\mathcal{G}_t$ using eigencount.
    Update h_t and recompute $\mathbf{\Psi}_t^{(2)}$ based on $h_t(\mathbf{L}_t)$.
  end if
  Find assignment $\mathbf{X}_t$ by applying the compressive k-means to features $\mathbf{\Theta}_t = [\mathbf{\Psi}_t^{(1)}, \mathbf{\Psi}_t^{(2)}]$.
  Set $\mathbf{\Psi}_t = \mathbf{\Psi}_t^{(2)}$.
end for

and dp randomly selected features of $\mathcal{G}_{t-1}$:

$$\mathbf{\Theta}_t = [\mathbf{H}_{t-1}\mathbf{R}_{dp},\ \mathbf{H}_t\mathbf{R}_{dq}] = \mathbf{\Psi}_{t-1}\mathbf{S}_{dp}^d + \mathbf{\Psi}_t\overline{\mathbf{S}_{dp}^d} \quad (12)$$

Above, we use the sub-identity matrix $\mathbf{S}_{dp}^d = \mathbf{I}_{d\times dp}\mathbf{I}_{dp\times d}$ and its complement $\overline{\mathbf{S}_{dp}^d} = \mathbf{I}_{d\times d} - \mathbf{S}_{dp}^d$.

Component 2. An important part of the complexity of CSC stems from using the eigencount algorithm to estimate λ_k and construct the Chebyshev polynomials (step 1 of their algorithm). To avoid recomputing λ_k, we start by assuming that the estimated value for λ_k at $t-1$ is a also good candidate for t and proceed to use the same polynomial in order to filter the qd new random vectors $\mathbf{R}_{dq}$ in $\mathcal{G}_t$. Notice that the eigencount method requires exactly these new features to determine if λ_k was correctly estimated and thus to validate our assumption. If our assumption is invalid, i.e., the λ_k has changed from $t-1$ to t, then we rerun the eigencount method from scratch as in (Di Napoli et al., 2016) but providing λ_k as an initial estimate. The final set of features generated in the eigencount now serves as $\mathbf{\Psi}_t$. When the assumption is valid, we proceed as is.

Complexity analysis. There are two steps where the complexity is reduced with respect to CSC. First, the optimization proposed for the determination of λ_k avoids computing steps of dichotomy for every graph. Spectrally similar graphs generally possess close spectrum and close values for λ_k. One could then expect to recompute λ_k only intermittently, in which cases he/she would also benefit from a reduced number of iterations due to a good initialization[3]. If S are the total number of eigencount steps gained, the total gain is $O(Scm)$. Second, since we reuse random features from one graph to the next, the total number of computed random vectors will necessarily be reduced compared to the use of τ independent CSC calls. The gain here is $O(cmdp)$ per time-step.

All reductions applied through compression can also benefit to our dynamic method. Indeed, we theoretically showed that reusing features from the past can replace the creation of new random vectors. Thus, sampling the combination of old and new vectors can be applied exactly as defined in CSC. Then, the result of the sub-assignment can be interpolated also as defined in (Tremblay et al., 2016).

[3]Though this trend has been confirmed by our numerical experiments, a formal proof remains elusive.

4.2. Analysis of dynamic CSC

Similarly to the static case, our objective is to provide probabilistic guarantees about the approximation quality of the proposed method. Let

$$\mathbf{X}_{\Theta_t} = \arg\min_{\mathbf{X}\in\mathcal{X}} \|\mathbf{\Theta}_t - \mathbf{X}\mathbf{X}^\top\mathbf{\Theta}_t\|_F. \quad (13)$$

be the clustering assignment obtained from k-means with $\mathbf{\Theta}_t$ as features, and define the *dynamic CSC cost* C_{Θ_t} as

$$C_{\Theta_t} = \|\mathbf{\Phi} - \mathbf{X}_{\Theta_t}\mathbf{X}_{\Theta_t}^\top\mathbf{\Phi}\|_F. \quad (14)$$

As the following theorem claims, the graph evolution introduces an additional error term that is a function of the graph similarity (spectral- or edge- wise).

Theorem 4.1. *At time t, the dynamic CSC cost C_{Θ_t} and the SC cost C_{Φ_t} are related by*

$$C_{\Phi_t} \le C_{\Theta_t} \le C_{\Phi_t} + 2\sqrt{\frac{k}{d}}(\sqrt{k}+\varepsilon) + (1+\delta)p\,\gamma, \quad (15)$$

with probability at least

$$1 - \exp\left(-\frac{\varepsilon^2}{2}\right) - \exp\left(2\log(n) - dp\left(\frac{\delta^2}{4} - \frac{\delta^3}{6}\right)\right),$$

where $0 < \delta \le 1$. Above, γ depends only on the similarity of the graphs in question. If graphs $\mathcal{G}_{t-1}$ and $\mathcal{G}_t$ are

- *(ρ, k)-spectrally similar, then $\gamma = \rho$,*
- *ρ-edge similar, then $\gamma = (\sqrt{2}\,\rho)/\alpha$, where $\alpha = \min\{\lambda_k^t, \lambda_{k+1}^{(t-1)} - \lambda_k^t\}$ is the Laplacian eigen-gap.*

Proof. Let $\mathbf{X}_{\Phi_t}$ and $\mathbf{X}_{\Theta_t}$ be respectively the optimal SC and dCSC clustering assignments at time t, and denote $\mathbf{E} = \mathbf{\Theta}_t - \mathbf{\Phi}_t\mathbf{I}_{k\times d}\mathbf{Q}$. We have that,

$$C_{\Theta_t} \le C_{\mathbf{\Phi}_t} + 2\|\mathbf{\Theta}_t - \mathbf{\Phi}_t\mathbf{I}_{k\times d}\mathbf{Q}\|_F, \quad (16)$$

following the exact same steps as in the proof of Lemma 3.1. By completing the matrices containing the filtering of both graphs, we can see that the error term can be

rewritten as

$$\begin{aligned}\|\mathbf{E}\|_F &= \|\mathbf{\Psi}_{t-1}\mathbf{S}^d_{dp} + \mathbf{\Psi}_t\overline{\mathbf{S}^d_{dp}} - \mathbf{\Phi}_t\mathbf{I}_{k\times d}\mathbf{Q}\|_F \qquad (17)\\ &= \|(\mathbf{\Psi}_{t-1}-\mathbf{\Psi}_t)\mathbf{S}^d_{dp} + \mathbf{\Psi}_t - \mathbf{\Phi}_t\mathbf{I}_{k\times d}\mathbf{Q}\|_F\\ &\leq \|(\mathbf{\Psi}_t-\mathbf{\Psi}_{t-1})\mathbf{S}^d_{dp}\|_F + \|\mathbf{\Psi}_t - \mathbf{\Phi}_t\mathbf{I}_{k\times d}\mathbf{Q}\|_F.\end{aligned}$$

The rightmost term of eq. (17) corresponds to the effects of random filtering and has been studied in depth in Thm. 3.2 and Cor. 3.2. The rest of the proof is devoted to studying the leftmost term.

We apply the Johnson-Lindenstrauss lemma (Johnson and Lindenstrauss, 1984) on the term of interest. Setting $\mathbf{R}' = \frac{1}{\sqrt{p}}\mathbf{R}\mathbf{I}_{d\times dp}$, we have that

$$\begin{aligned}\|(\mathbf{\Psi}_t-\mathbf{\Psi}_{t-1})\mathbf{S}^d_{dp}\|_F^2 &= \|(\mathbf{H}_t-\mathbf{H}_{t-1})\mathbf{R}\mathbf{I}_{d\times dp}\|_F^2\\ &= p\sum_{i=1}^{n}\|\mathbf{R}'^{\top}(\mathbf{H}_t-\mathbf{H}_{t-1})^{\top}\delta_i\|_2^2.\end{aligned}$$

Matrix $\mathbf{R}' = p^{-1/2}\mathbf{R}\mathbf{I}_{d\times dp}$ has $n\times dp$ Gaussian i.i.d. entries with zero-mean and variance $1/dp$. It follows from the Johnson-Lindenstrauss lemma that

$$\begin{aligned}\|(\mathbf{\Psi}_t-\mathbf{\Psi}_{t-1})\mathbf{S}^d_{dp}\|_F^2 &\leq p(1+\delta)\sum_{i=1}^{n}\|(\mathbf{H}_t-\mathbf{H}_{t-1})^{\top}\delta_i\|_2^2\\ &\leq p(1+\delta)\|\mathbf{H}_t-\mathbf{H}_{t-1}\|_F^2,\end{aligned}$$

with probability at least $1-n^{-\beta}$ and for $dp \geq \frac{4+2\beta}{\delta^2(\frac{1}{2}-\frac{\delta}{3})}\log(n)$. Coupling the two together we obtain a probability at least equal to $1-\exp(2\log(n)-\frac{dp\delta^2}{2}(\frac{1}{2}-\frac{\delta}{3}))$, where δ can be set between 0 and 1. A loose bound gives $2p\|\mathbf{H}^{(2)}-\mathbf{H}^{(1)}\|_F^2$ with probability $1-\exp(2\log(n)-\frac{dp}{12})$.

This concludes the part of the proof concerning spectrally similar graphs. The result for edge-wise similarity follows from Cor. 5.1 found in the appendix. □

4.3. Controlling the approximation error

Theorem 4.1 can be used to adaptively determine p at each time-step with the objective of attaining a bounded error for the term $(1+\delta)p\rho$. For instance, if the graph did not evolve much between the last two time-steps, more signals should be reused than otherwise. For that, one needs to be able to approximate the graph similarity, without computing the spectral basis.

This can be quite easily achieved for the edge similarity measure, by simply computing the Frobenius norm of the edge weight difference between the two graph Laplacian matrices. As we show next, the spectral similarity measure ρ can also be estimated as follows:

$$\hat{\rho} := \|\mathbf{H}_t\mathbf{R} - \mathbf{H}_{t-1}\mathbf{R}\|_F \qquad (18)$$

Algorithm 2 Dynamic CSC with adaptive p

Input: $(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_\tau), d, k, \delta, \epsilon$
Output: $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_\tau)$

Determine λ_k and filter h_1 for $\mathcal{G}_1$.
Find $\mathbf{X}_1$ for $\mathcal{G}_1$ using CSC with $\mathbf{\Psi}_1 = h_1(\mathbf{L}_1)\mathbf{R}$.
for t from 2 to τ **do**
  Split features $\mathbf{\Psi}_{t-1} = [\mathbf{\Psi}^{(1)}_{t-1}, \mathbf{\Psi}^{(2)}_{t-1}]$ s.t. $\mathbf{\Psi}^{(1)}_{t-1}$ contains $\frac{d}{2}$ vectors. Let $\mathbf{R}^{(1)}$ be random vectors used to generate $\mathbf{\Psi}^{(1)}_{t-1}$.
  Set $h_t = h_{t-1}$ and compute $\mathbf{\Psi}^{(1)}_t = h_t(\mathbf{L}_t)\mathbf{R}^{(1)}$.
  Test whether $\lambda_k \in [\lambda_k(\mathbf{L}_t), \lambda_{k+1}(\mathbf{L}_t)]$ with eigencount and using $\mathbf{\Psi}^{(1)}_t$.
  if the test fails **then**
    Determine λ_k for $\mathcal{G}_t$ using eigencount.
    Update h_t and recompute $\mathbf{\Psi}^{(1)}_t = h_t(\mathbf{L}_t)\mathbf{R}^{(1)}$.
  end if
  Set $p = \min\left(\frac{1}{2}, \frac{\varepsilon_2}{1+\delta}\|\mathbf{\Psi}^{(1)}_t - \mathbf{\Psi}^{(1)}_{t-1}\|_F^{-1}\right)$.
  Select dp features from $\mathbf{\Psi}^{(2)}_{t-1}$ and call them $\mathbf{\Psi}^{(2)}_t$.
  Generate $d(\frac{1}{2}-p)$ features $\mathbf{\Psi}^{(3)}_t = h_t(\mathbf{L}_t)\mathbf{R}^{(3)}$, where $\mathbf{R}^{(3)}$ are new random vectors.
  Find assignment $\mathbf{X}_t$ by applying the compressive k-means to features $\mathbf{\Theta}_t = [\mathbf{\Psi}^{(1)}_t, \mathbf{\Psi}^{(2)}_t, \mathbf{\Psi}^{(3)}_t]$.
  Set $\mathbf{\Psi}_t = [\mathbf{\Psi}^{(1)}_t, \mathbf{\Psi}^{(3)}_t]$.
end for

To motivate this, observe that $\tilde{\rho}^2$ is an unbiased estimator:

$$\begin{aligned}\mathbb{E}\left[\hat{\rho}^2\right] &= \mathbb{E}\left[\mathrm{tr}\left(\mathbf{R}^{\top}(\mathbf{H}_t-\mathbf{H}_{t-1})^{\top}(\mathbf{H}_t-\mathbf{H}_{t-1})\mathbf{R}\right)\right]\\ &= \mathrm{tr}\left((\mathbf{H}_t-\mathbf{H}_{t-1})\,\mathbb{E}\left[\mathbf{R}\mathbf{R}^{\top}\right](\mathbf{H}_t-\mathbf{H}_{t-1})^{\top}\right)\\ &= \mathrm{tr}\left((\mathbf{H}_t-\mathbf{H}_{t-1})(\mathbf{H}_t-\mathbf{H}_{t-1})^{\top}\right) = \rho^2, \qquad (19)\end{aligned}$$

which implies that $\tilde{\rho}$ approaches ρ as d grows.

With this in place, we proceed to modify Alg. 4.1 so as to include the estimation of ρ and the adaptive estimation of p. The detailed procedure is summarized in Alg. 2, focusing on the case of spectral similarity (i.e., $\gamma = \rho$). Since Thm. 4.1 proved an additive error of at most $(1+\delta)p\rho$, we fix an upper bound on the error that we tolerate ε_2 and δ that controls the probability of success, and set $p = \frac{\varepsilon_2}{(1+\delta)\tilde{\rho}}$.

Though the adaptive algorithm features the same complexity, it is slightly more involved than Alg. 4.1. The main difference is that ρ is estimated based on features $\mathbf{\Psi}^{(1)}_t$ and $\mathbf{\Psi}^{(1)}_{t-1}$ that correspond to the same random vectors $\mathbf{R}^{(1)} \in \mathbb{R}^{n\times d}$ filtered on two consecutive graphs (i.e., $\mathcal{G}_t$ and $\mathcal{G}_{t-1}$). Features $\mathbf{\Psi}^{(1)}_t$ are combined with the pd reused features $\mathbf{\Psi}^{(2)}_t$ and the $d(1/2-p)$ new features $\mathbf{\Psi}^{(3)}_t$ to identify assignment $\mathbf{X}_t$.

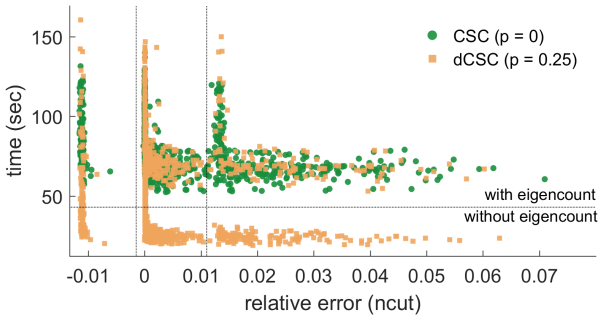

Figure 1. Error-complexity trade-off achieved by compressive clustering methods. The points correspond to different number of features d and reperitions. Three behaviors are highlighted: the majority of the runs tend to the same quality than SC with provided enough features; the rest is relatively close (1% deviation) and can be of better or worse quality; dCSC benefits from a significant complexity reduction when λ_k is not recomputed (up to a factor 3).

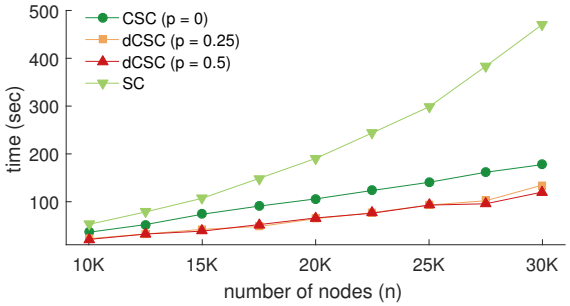

Figure 2. Scaling capabilities of SC and its approximations. dCSC consistently outperforms CSC in terms of complexity. Their almost linear complexity make them good candidates for big data analysis compared to SC.

5. Experiments

This section complements the theoretical results described in Section 4. All our experiments are designed using the GSPBox (Perraudin et al., 2014).

5.1. Experimental setup

As is common practice, we use the Stochastic Block Model (SBM) to evaluate the efficiency of our spectral clustering method (e.g., Görke et al., 2013; Tremblay et al., 2016). In SBM, data are clustered in k classes and the n nodes are connected at random with edge-wise probability that depends if the two extremities belong to the same cluster (q_1) or not (q_2 with $q_2 \ll q_1$). In the following, we qualify the SBM parameters in terms of the nodes' average degree $\bar{\delta}$ and the ratio q_2/q_1 that captures the graph clusterability (Decelle et al., 2011). Following the recommendations of the former work, we set $\frac{q_2}{q_1} = \frac{\bar{\delta}-\sqrt{\bar{\delta}}}{2(\bar{\delta}+\sqrt{\bar{\delta}}(k-1))}$ to construct non-trivially clusterable graphs.

We compare the quality and complexity of our dynamic method (dCSC) against the algorithm of Tremblay et al. (CSC) and an optimized spectral clustering (Ng et al., 2002) that uses the Lanczos algorithm to compute the first k-eigenvectors (this is significantly faster than doing the entire eigendecomposition while introducing negligible error). We use relative error measures to compare the achieved clustering accuracy of CSC and dCSC with that of SC (i.e., $|C_A - C_{SC}|/C_{SC}$, where C_A is the cost of algorithm A and C_{SC} the cost of SC). We considered two cost measures: the k-means cost (eq. (6)) and the normalized cut (ncut) cost. Since the obtained results were almost identical, we only report the results for ncut in the rest of this section (except for Table 1). After all, the k-means cost of the spectral features is a relaxation of the ncut cost.

Our analysis highlights the importance of the spectral similarity between consecutive graphs. It is thus important to define how the graph changes between consecutive steps. Starting from a SBM, we perform two types of perturbations: *edge redrawing* and *node reassignment*. Edge redrawing consists of removing some edges at random from the original graph and then adding the same number following the probabilities defined by the graph model (using q_1 and q_2). In node reassignment, one selects nodes, removes all edges that share at least one end with the nodes previously picked, reassigns those nodes to any other class at random and reconnects these nodes with new edges using again the same probabilities q_1 and q_2. Both perturbations are combined in the synthetic graph that we are studying. We replicate the construction of 100 different SBM with the same parameters, then we alter each with 1% of node reassignment and 1% of edges modifications. The modified graph is used for the evaluation of all methods.

5.2. When does reusing features pay off?

We first study the error-complexity trade-off achieved by the compressive clustering methods as a function of d. We set $n = 15000, k = 25, \bar{\delta} = 60$. Each point in Figure 1 corresponds to a single graph being clustered. For each of the two methods, there are 1600 points resulting from 100 repetitions when the number of features is $d \in [6, 200]$ with logarithmic increments. To comprehend the results, it is helpful to consider each of the six sextants in the figure separately. The top-middle sextant shows that when d is large enough (left side), the relative error of CSC and dCSC is close to zero. Increasing d reduces the error but increases the time required for the computation, following the elbow from right to left. The top-right and top-left sextants occur because Lloyd's algorithm (despite being rerun 100 times) sometimes fails to retrieve the optimal solution to the k-means problem: the top-left (resp. right) sextant corresponds to cases when Lloyd's algorithm produces a suboptimal assignment for SC (resp. CSC/dCSC). The bot-

Table 1. Timing and accuracy comparison with state-of-the-art for various sizes, $k = 25, d = 50, \bar{\delta} = 60, \varepsilon = \frac{\varepsilon_c}{2}$.
SC: Spectral Clustering using the Lanczos method. IASC: Incremental Approximate Spectral Clustering (Dhanjal et al., 2014). CSC: Compressive Spectral Clustering (Tremblay et al., 2016). dCSC: our method (Alg. 4.1).

		SC (Lanczos)	IASC	CSC (p=0)	dCSC (p=0.25)	dCSC (p=0.5)
n = 1'000	time (sec)	**1.27** ($\pm$0.09)	1.37 ($\pm$0.20)	2.81 ($\pm$0.56)	2.53 ($\pm$0.54)	2.86 ($\pm$0.56)
	k-means	**5.42** ($\pm$0.06)	6.19 ($\pm$0.28)	6.16 ($\pm$6.40)	5.49 ($\pm$1.35)	5.48 ($\pm$5.02)
	ncut	**18.96** ($\pm$0.01)	19.20 ($\pm$0.08)	19.18 ($\pm$2.00)	18.98 ($\pm$0.44)	18.99 ($\pm$1.39)
n = 10'000	time (sec)	48.29 ($\pm$5.41)	806.29 ($\pm$72.91)	41.21 ($\pm$3.11)	23.33 ($\pm$9.29)	**21.98** ($\pm$11.33)
	k-means	6.24 ($\pm$0.01)	6.61 ($\pm$0.31)	6.31 ($\pm$0.61)	6.28 ($\pm$0.40)	**6.26** ($\pm$0.40)
	ncut	18.80 ($\pm$0.00)	18.91 ($\pm$0.10)	18.82 ($\pm$0.21)	**18.81** ($\pm$0.13)	**18.81** ($\pm$0.13)

tom three sextants correspond to cases when dCSC did not have to recompute λ_k (step 7 of Alg. 4.1). In these cases, dCSC is up to $2\times$ faster than CSC. Though the frequency of this phenomenon depends on many factors, such as the size of the eigengap and the spectral similarity of consecutive graphs, we report that in our experiment dCSC could avoid recomputing λ_k, roughly 50% of the times.

In summary, reusing features produces a clear computational benefit with a reasonable loss of accuracy. Most benefit comes from λ_k estimation (component 2) that can be often avoided when consecutive graphs are spectrally similar, especially for well-clusterable graphs (where the gap $\lambda_{k+1} - \lambda_k$ is large). To quantify the benefit of reusing a portion of features (component 1), we compare here the execution time of CSC and dCSC, excluding the time for λ_k estimation. Increasing p by 0.25 saved 0.97 and 9.98 seconds respectively when $n = 10'000$ and $50'000$—the later corresponds to a speedup of 1.29x w.r.t. feature estimation.

5.3. Comparison with state-of-the-art

To evaluate the efficiency of dCSC, we varied the number of nodes n (while fixing $k = 25, d = 50, \bar{\delta} = 60$). Figure 2 shows the results. As expected, the difference of complexity between spectral clustering using the partial eigen-decomposition and dCSC is clearly visible. Increasing p from 0.25 to 0.5 incurs a non-negligible computational benefit for larger n (14.5 seconds when n =30'000, corresponding to a 12% improvement). We also report that the achieved relative error for both methods remained consistently below 0.1% and did not grow as n increased. We do not present values of n above 30'000 as, for such cases, SC took too long to complete. For example, with 64Gb of RAM, SC took one hour to process the graph and return an assignment when $n = 50'000$. For the same graph, dCSC run in 6.5 minutes and resulted in a similar k-means cost.

Table 1 further compares our proposed method to SC, CSC and IASC, the state-of-the-art method for spectral clustering suitable for dynamic graphs (Dhanjal et al., 2014). Note that there is a long list of heuristic-based clustering algorithms optimized for speed (Dhillon et al., 2007; Karypis and Kumar, 1998), but we only consider here algorithms that provably approximate spectral clustering. We can see that dCSC achieves a significant improvement in timing when n is large enough. Note that IASC results were obtained by running the optimized and parallel implementation kindly provided by the original authors[4]. Our hypothesis is that the poor complexity of IASC is attributed to the fact that, in our tests (and as is frequently the case) the eigengap was not particularly large.

6. Conclusion and Future Work

The major contribution of this paper has been the presentation of a clustering algorithm for dynamic graphs that achieves solution quality *provably* approximating that of Spectral Clustering. Numerical experiments suggest that our method is faster than previous approximation methods.

Recent advances in spectral clustering, including this work, can provide a huge complexity gain. Nevertheless, practitioners must pay attention because these works require a proper setup. In particular, n must be large for the approximated method to make sense. Moreover, when working with dynamic networks, the spectral similarity should remain bounded for our algorithm to perform best.

We highlight in this paper two open directions of research. It appears clearly in the experiments that the majority of the remaining complexity lies in the estimation of $\mathbf{H}$ and more precisely the capability to reuse the previous determination of λ_k. Finally, expressing the approximation error in function of the assignment instead of ncut could produce a more insightful explaination of the impact of the various factors.

Acknowledgements

We would like to acknowledge the reviewers for their valuable comments. Their suggestions helped us clarify and improve our work.

[4] Available at https://github.com/charanpald/sandbox

References

Allen-Zhu, Z. and Li, Y. (2016). Faster principal component regression via optimal polynomial approximation to sgn (x). *arXiv preprint arXiv:1608.04773*.

Boutsidis, C., Gittens, A., and Kambadur, P. (2015). Spectral clustering via the power method-provably. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*.

Chakrabarti, D., Kumar, R., and Tomkins, A. (2006). Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM.

Chi, Y., Song, X., Zhou, D., Hino, K., and Tseng, B. L. (2007). Evolutionary spectral clustering by incorporating temporal smoothness. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 153–162. ACM.

Cohen, M. B., Elder, S., Musco, C., Musco, C., and Persu, M. (2015). Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 163–172. ACM.

Decelle, A., Krzakala, F., Moore, C., and Zdeborová, L. (2011). Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106.

Dhanjal, C., Gaudel, R., and Clémençon, S. (2014). Efficient eigen-updating for spectral graph clustering. *Neurocomputing*, 131:440–452.

Dhillon, I. S., Guan, Y., and Kulis, B. (2007). Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11).

Di Napoli, E., Polizzi, E., and Saad, Y. (2016). Efficient estimation of eigenvalue counts in an interval. *Numerical Linear Algebra with Applications*.

Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3):75–174.

Fowlkes, C., Belongie, S., Chung, F., and Malik, J. (2004). Spectral grouping using the nystrom method. *IEEE transactions on pattern analysis and machine intelligence*, 26(2):214–225.

Gauvin, L., Panisson, A., and Cattuto, C. (2014). Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach. *PloS one*, 9(1):e86028.

Gittens, A., Kambadur, P., and Boutsidis, C. (2013). Approximate spectral clustering via randomized sketching. *Ebay/IBM Research Technical Report*.

Görke, R., Maillard, P., Schumm, A., Staudt, C., and Wagner, D. (2013). Dynamic graph clustering combining modularity and smoothness. *Journal of Experimental Algorithmics (JEA)*, 18:1–5.

Hammond, D. K., Vandergheynst, P., and Gribonval, R. (2011). Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150.

Isufi, E., Loukas, A., Simonetto, A., and Leus, G. (2017). Autoregressive moving average graph filtering. *IEEE Transactions on Signal Processing*, 65(2):274–288.

Johnson, W. B. and Lindenstrauss, J. (1984). Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1.

Karypis, G. and Kumar, V. (1998). A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392.

Laurent, B. and Massart, P. (2000). Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pages 1302–1338.

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

Loukas, A., Simonetto, A., and Leus, G. (2015). Distributed autoregressive moving average graph filters. *Signal Processing Letters*, 22(11):1931–1935.

Loukas, A. and Vandergheynst, P. (2018). Spectrally approximating large graphs with smaller graphs. In *Interenational Conference on Machine Learning (ICML)*.

Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856.

Ning, H., Xu, W., Chi, Y., Gong, Y., and Huang, T. (2007). Incremental spectral clustering with application to monitoring of evolving blog communities. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 261–272. SIAM.

Paratte, J. and Martin, L. (2016). Fast eigenspace approximation using random signals. *arXiv preprint arXiv:1611.00938*.

Perraudin, N., Paratte, J., Shuman, D., Martin, L., Kalofolias, V., Vandergheynst, P., and Hammond, D. K. (2014). GSPBOX: A toolbox for signal processing on graphs. *ArXiv e-prints*.

Puy, G., Tremblay, N., Gribonval, R., and Vandergheynst, P. (2016). Random sampling of bandlimited signals on graphs. *Applied and Computational Harmonic Analysis*.

Pydi, M. S. and Dukkipati, A. (2017). Spectral clustering via graph filtering: Consistency on the high-dimensional stochastic block model. *arXiv preprint arXiv:1702.03522*.

Ramasamy, D. and Madhow, U. (2015). Compressive spectral embedding: sidestepping the svd. In *Advances in Neural Information Processing Systems*, pages 550–558.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

Shuman, D. I., Vandergheynst, P., and Frossard, P. (2011a). Chebyshev polynomial approximation for distributed signal processing. In *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–8. IEEE.

Shuman, D. I., Vandergheynst, P., and Frossard, P. (2011b). Distributed signal processing via chebyshev polynomial approximation. *arXiv preprint arXiv:1111.5239*.

Tremblay, N. T., Puy, G., Gribonval, R., and Vandergheynst, P. (2016). Compressive Spectral Clustering. In *33rd International Conference on Machine Learning*, New York, United States.

Tu, K., Ribeiro, B., Swami, A., and Towsley, D. (2016). Detecting cluster with temporal information in sparse dynamic graph. *arXiv preprint arXiv:1605.08074*.

Vershynin, R. (2010). Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*.

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