2021 60th IEEE Conference on Decision and Control (CDC)
December 13-15, 2021. Austin, Texas

Progressive Graph Partitioning Based on Information Diffusion

Christos N. Mavridis and John S. Baras

2021 60th IEEE Conference on Decision and Control (CDC) | 978-1-6654-3659-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CDC45484.2021.9682799

***Abstract*— We propose an online deterministic annealing algorithm for progressive graph partitioning based on the spectral information of the underlying graph Laplacian matrix. Online deterministic annealing is a prototype-based unsupervised learning algorithm that progressively adjusts the number of prototypes used with respect to a performance-complexity trade-off. Due to the online nature of the proposed learning algorithm, the structure of the graph need not be known a priori. In this regard, we construct a distributed approximation algorithm to estimate the spectral information of the graph Laplacian, bypassing the exact computation of its eigenvectors. By propagating an impulse through the graph via a diffusion equation, we show that each node can construct a local learning representation which can be used for spectral clustering. As a result, the proposed approach is suitable for large graphs, requires minimal hyper-parameter tuning, and provides online control over the complexity-accuracy trade-off. We illustrate the properties and evaluate the performance of the proposed methodology in graph partition and image segmentation applications.**

I. INTRODUCTION

Analysis of large interconnected systems, such as communication and power networks, swarms of autonomous agents, and social networks, has been a topic of increasing interest for decades [1], [2], [3]. Graph partitioning, in particular, is considered a problem of both theoretical and practical importance, as it appears as an integral part of network analysis in multiple applications, including image processing [4], analysis of protein sequences [5], and resource allocation and routing in 5G networks [6].

Networked systems are mathematically described and studied via graph theoretic methods. Spectral properties of the Laplacian matrix L associated with such graphs, have been shown to provide useful information for their analysis and design [7]. In graph clustering, spectral methods use the eigenvectors of L as unsupervised learning features for standard clustering algorithms, such as k-means [8]. This procedure approximates a global solution to the NP complete problem of creating a partition that minimizes the ratio of the inter-connection strength to the size of individual clusters [9]. However, traditional clustering algorithms, including k-means, are offline algorithms that require the entire dataset to be known a priori, while they heavily depend on three major design parameters: (a) the number of prototypes, which, defines the complexity of the model, (b) the initial conditions, that affect the performance of the algorithm, and (c) the proximity measure used to quantify the similarity between two vectors in the data space. As a result, the development of an online clustering algorithm that avoids poor local minima while progressively adjusting the number of clusters used, is of significant interest [10], [11].

At the same time, an online clustering algorithm would suggest that the structure of the graph need not be known a priori. This gives rise to the use of distributed algorithms that estimate the eigenvectors of the graph Laplacian L, which would typically require the eigendecomposition of L, which in case of large graphs is computationally expensive, and assumes that the graph Laplacian is readily available a priori. Numerous distributed algorithms have been proposed for the estimation of the eigenvectors of L [12], [13], [14]. In [12] the nodes of the graph perform a local iterative projection algorithm to estimate the eigenvectors of L, during which, however, the nodes are required to communicate (via a random walk algorithm) and reach a consensus multiple times. In [13] and [14] on the other hand, a discretized wave equation is used to propagate a test signal, while frequency-based Fourier methods are used to analyze the response of the nodes and approximate the eigenvalues of the graph Laplacian L. While this is a more direct and fast approach towards the computation of the spectral properties of a graph, the solution of the wave equation does not reach a consensus, which can greatly limit the network's bandwidth. More importantly, the solution approximation depends on finding peak values of the discrete Fourier transform, which, by itself, is an unstable process, and requires observations over a large time window, especially for low frequencies which is desired.

In this work, we develop an online deterministic annealing algorithm for spectral clustering. Online deterministic annealing is a prototype-based clustering algorithm that, due to its annealing nature, is robust with respect to the initial conditions, while progressively increasing the number of prototypes used, with respect to a performance-complexity trade-off. Moreover, it makes use of dissimilarity measures that belong to the family of Bregman divergences [15], [10], which can greatly improve the performance of learning algorithms [16]. In order to perform spectral clustering, we adopt a direct approach to approximate the spectral information of the graph in a distributed fashion, similar to the one in [13]. A low-bandwidth signal is propagated through the graph via a diffusion equation, while the local node responses are used to construct a useful representation of the spectral information of L, thus bypassing the exact computation of the eigenvectors. Similar to traditional spectral clustering [9], the locally computed representations

The authors are with the Electrical and Computer Engineering Department and the Institute for Systems Research, University of Maryland, College Park, USA. emails:{mavridis, baras}@umd.edu

Research partially supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00111990027, by ONR grant N00014- 17-1-2622, and by a grant from Northrop Grumman Corporation.

are used as unsupervised learning features for the online deterministic annealing algorithm, which acquires observations online, by direct communication with the nodes, one at a time, in order to progressively build a graph partition that approximately solves the minimal ratio graph clustering problem. As a result, the proposed approach is suitable for large graphs, requires minimal hyper-parameter tuning, and offers online control over the complexity-accuracy trade-off. We illustrate the properties and evaluate the performance of the proposed methodology in graph partition and spectral clustering applications.

II. Distributed Spectral Feature Extraction

In this section we briefly review some of the main concepts of graph theory, and introduce a distributed algorithm to compute learning representations based on the spectral information of the graph Laplacian.

A. Graph Theory and Notation

Let $G = (V, E)$ be an undirected graph with vertex set V of cardinality $|V| = N$, and edge set $E \subseteq V \times V$. Without loss of generality we assume that $V = \{1, \dots, N\}$. In general, G is assumed weighted, with the weights $W_{ij} \geq 0$, associated with each edge $(i, j) \in E$, forming the weighted adjacency matrix W of dimension $N \times N$. The weights W_{ij} represent the connection strength with the understanding that $W_{ij} = 0$ if and only if $(i, j) \notin E$. We define the random-walk normalized Laplacian matrix L by:

$$L_{ij} = \begin{cases} 1, & \text{if } i = j \\ -W_{ij}/\sum_{k=1}^{N} W_{ij}, & \text{if } (i,j) \in E \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

which is equivalent to $L = \mathbb{I} - D^{-1}W$ where D is the diagonal degree matrix with the row sums of W. The normalized graph Laplacian L is strongly connected to the Laplacian operator ∇^2 found in the parabolic heat PDEs and to the evolution of random walks on the graph [9].

Given the Laplacian matrix L, spectral clustering methods partition the graph G with respect to the elements of the eigenvectors $\left\{v^{(i)}\right\}_{i=1}^{N}$ of L. Assuming that the eigenvalues $\{\lambda_i\}_{i=1}^{N}$ of L are ordered such that $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$, it is easy to show that $\lambda_1 = 0$ and $v^{(1)} = \mathbb{1}$. Moreover, it is well known that the multiplicity of λ_1 is equal to the number of connected components in the graph [9], and that $0 \leq \lambda_j \leq 2, \forall j \in \{1, \dots, N\}$, which is a direct consequence of the Gershgorin's theorem. In the following, we assume that $\lambda_1 < \lambda_2$, i.e. that G does not have disconnected clusters. This is not a restrictive assumption, as will be shown in Section IV. We also we assume that there exist unique cuts that divide the graph into K clusters, i.e. that there exist K distinct eigenvalues close to zero, albeit for unknown K.

B. Distributed Spectral Feature Extraction via Information Diffusion

The cluster that each node (i) belongs to is decided by the values $\left\{v_i^{(m)}\right\}_{m \in M}$, where $M \subseteq \{2, \dots, N\}$, while the second eigenvector $v^{(2)}$, called the Feidler vector, is believed to provide most of the information needed [7]. If the matrix L is known a priori and is relatively small, one can compute the eigenvectors $\left\{v^{(i)}\right\}_{i=1}^{N}$ offline by standard matrix decomposition algorithms, and feed the values $\left\{v_i^{(m)}\right\}_{m \in M}$ to a clustering algorithm, e.g. k-means [8]. Distributed methods have also been proposed such that each node (i) can locally compute the corresponding elements of the eigenvectors $\left\{v_i^{(m)}\right\}_{m \in M}$. These methods typically include consensus algorithms or some other information-sharing mechanisms [12], [13], [14].

We simulate the evolution of the heat equation $\frac{\partial u}{\partial t} = \nabla^2 u$ on the graph G, which is given by the discretized equation [17]:

$$u_i(t+1) = u_i(t) - \sum_{j \in \mathcal{N}(i)} L_{ij} u_j(t), \quad i, j \in V \quad (2)$$

where $t \geq 0$ and $\mathcal{N}(i) = \{j : W_{ij} \neq 0\}$ is the set of the neighbors of node (i). The scalar values u_i form the signal $u := [u_0, \dots, u_N]^{\mathrm{T}}$ which is initialized such that $u_k(0) = 1$ for some $k \in \{1, \dots, N\}$ and $u_j(0) = 0$ for $j \neq k$. Therefore, the above iteration requires only local communication with the neighbors of each node (i), while only a single scalar value is being broadcasted.

Equation (2) forms a discrete dynamical system which can be written in matrix form as:

$$u(t+1) = (\mathbb{I} - L)u(t) \quad (3)$$

The solution of the heat equation (3) can be expanded in terms of the eigenvalues and eigenvectors of the matrix $(\mathbb{I} - L)$ which are closely related to those of matrix L. In fact, it is easy to see that the eigenvalues of $(\mathbb{I} - L)$ are given by $\{(1 - \lambda_i)\}_{i=1}^{N}$ and the eigenvectors of $(\mathbb{I} - L)$ and L coincide. Therefore, the solution $u(t)$ of (3) is given by

$$u(t) = c_1(1 - \lambda_1)^t v^{(1)} + \ldots + c_N(1 - \lambda_N)^t v^{(N)} \quad (4)$$

where the constants c_i, $i \in \{1, \dots, N\}$, depend on the initial condition $u(0)$ and the eigenvectors of the graph Laplacian L, and are given by $c_i = u(0)^{\mathrm{T}} \tilde{v}^{(i)}$, where $\tilde{v}^{(i)}$ represents the i-th row of the matrix V^{-1} with V being the matrix with columns corresponding to the eigenvectors $\left\{v^{(i)}\right\}_{i=1}^{N}$. Since c_i are constants, the products $\hat{v}^{(i)} := c_i v^{(i)}$, $i = 1, \dots, N$, are also eigenvectors of both $(\mathbb{I} - L)$ and L. Therefore, (4) can be written as:

$$u(t) = (1 - \lambda_1)^t \hat{v}^{(1)} + \ldots + (1 - \lambda_N)^t \hat{v}^{(N)} \quad (5)$$

and a similar equation holds locally for every node (i):

$$u_i(t) = c_1 + (1 - \lambda_2)^t \hat{v}_i^{(2)} + \ldots + (1 - \lambda_N)^t \hat{v}_i^{(N)} \quad (6)$$

since $(1 - \lambda_1) = 1$ and $v_i^{(1)} = 1$, $\forall i = 1 \dots, N$. Therefore, since every node (i) can keep track of its own response $u_i(t)$ for $t \geq 0$, a non-linear system of equations can be solved to reconstruct the $2N-2$ unknowns λ_j, $v_i^{(j)} = 1$, $j = 2, \dots, N$.

However, such an approach would not be robust for two main reasons. First, the solution is not unique. Secondly, the

values of λ_j, $j = 2, \ldots, N$, need to be the same in all N solutions that are locally computed, a constraint that cannot be imposed without further communication between the nodes. To counteract that, we observe that the information to reconstruct the eigenvectors $\left\{v^{(i)}\right\}_{i=1}^{N}$ lies within the vector of observations $\{u_i(t)\}_{t=0}^{N_c}$, and that, for spectral clustering, it may be possible to bypass the exact computation of the eigenvectors $\left\{v^{(i)}\right\}_{i=1}^{N}$, and use $\{u_i(t)\}_{t=0}^{N_c}$ directly. Here we have used $N_c \geq 2N - 2$ to represent the point of practical convergence to the consensus value $u_i^{\infty} = c_1, \quad \forall i \in \{1, \ldots, N\}$ Recall that for $i = 2, \ldots, N$, $0 < \lambda_i \leq 2$, and, making the reasonable assumption that the equality does not hold, i.e. that $\lambda_N < 2$, we get that $|(1 - \lambda_i)| < 1$, for $i = 2, \ldots, N$. Therefore, from (6) and the geometric series convergence, we get:

$$\sum_{t=0}^{\infty} u_i(t) - u_i^{\infty} = \sum_{j=2}^{N} \frac{1}{\lambda_j} \hat{v}_i^{(j)} \tag{7}$$

which is a linear combination of the eigenvectors of the graph Laplacian L, in which, notably, more weight is given to the first eigenvectors, since $0 < \lambda_2 \leq \ldots \leq \lambda_N < 2$. Moreover, the infinite sum $\sum_{t=0}^{\infty} u_i(t) - u_i^{\infty}$ can be approximated by

$$\hat{x}_i := \sum_{t=0}^{N_c} u_i(t) - u_i(N_c) \tag{8}$$

where N_c is the time that each node observes practical convergence. Therefore the one-dimensional value $\hat{x}_i$ that is locally computed by each node (i), for all $i = 1, \ldots, N$, can be used as a learning feature for spectral clustering.

We note that clustering with respect to $\hat{x}_i$, similar to the spectral clustering which is done with respect to $\left\{v_i^{(m)}\right\}_{m \in M}$, is a heuristic approach to approximate a solution to the graph cut problem that minimizes the ratio of the inter-connection strength to the size of individual clusters [9]. As a final note, since the time responses of the nodes are being used as learning features, further feature extraction can be applied using time-frequency analysis to reveal hidden patterns that may help in the clustering process. However, to our knowledge, there are currently no results that can connect the choice of these learning features with the resulting graph cuts. This is a similar problem to that of choosing the number M of the eigenvectors to be used in spectral clustering, and is typically answered experimentally.

III. Progressive Clustering with Online Deterministic Annealing

In this section, we introduce the online deterministic annealing algorithm for progressive graph partitioning using either the eigenvectors of the graph Laplacian $\left\{v_i^{(m)}\right\}_{m \in M}$, or the spectral features $\hat{x}_i$ as defined above. A detailed review of the mathematics and the implementation of the algorithm can be found in [10] and the references therein.

A. Towards the Deterministic Annealing Approach

Unsupervised analysis can provide valuable insights into the nature of a data space. In particular, vector quantization [18] can reduce storage needs, reveal structures, such as clusters in the data, or pre-process large datasets for further analysis, through the representation of the data space by a set of prototypes. Given a random variable $X : \Omega \to S$ defined in the probability space $(\Omega, \mathcal{F}, \mathbb{P})$, a quantizer $Q : S \to S$ is defined such that $Q(X) = \sum_{h=1}^{K} \mu_h \mathbb{1}_{[X \in S_h]}$, where $V := \{S_h\}_{h=1}^{K}$ forms a partition of S and $M := \{\mu_h\}_{h=1}^{K}$ represents a set of codevectors such that $\mu_h \in ri(S_h)$, $h \in \{1, \ldots, K\}$. Given a dissimilarity measure $d : S \times ri(S) \to [0, \infty)$ one seeks the optimal M, V in terms of minimum average distortion:

$$\min_{M,V} \; D(Q) := \mathbb{E}\left[d\left(X, Q(X)\right)\right]$$

Vector quantization algorithms assume that Q is a deterministic function of X and are proven to converge to locally optimal configurations even when formulated as online learning algorithms [18]. However, their convergence properties and final configuration depend heavily on two design parameters: (a) the number of clusters (neurons), and (b) their initial configuration. To deal with this phenomenon, we adopt concepts from annealing optimization methods, motivated by annealing processes in physical chemistry.

A particularly interesting approach is the Deterministic Annealing approach [19], which makes use of a probabilistic framework, where input vectors are assigned to clusters in probability, thus dropping the assumption that Q is a deterministic function of X. For the randomized partition, the expected distortion becomes:

$$D = \mathbb{E}\left[d_\phi(X, Q)\right] = \mathbb{E}\left[\mathbb{E}\left[d_\phi(X, Q)|X\right]\right]$$

The central idea of deterministic annealing is to seek the distribution that minimizes D subject to a specified level of randomness, measured by the Shannon entropy

$$H(X, M) = H(X) - \mathbb{E}\left[\mathbb{E}\left[\log p(M|X)|X\right]\right]$$

with $p(\mu|x)$ representing the association probability relating the input vector x with the codevector μ. This is essentially a realization of the Jaynes's maximum entropy principle [20] which states: of all the probability distributions that satisfy a given set of constraints, choose the one that maximizes the entropy. The resulting multi-objective optimization is conveniently formulated as the minimization of the Lagrangian

$$F = D - TH \tag{9}$$

where T is the temperature parameter that acts as a Lagrange multiplier. Clearly, (9) represents the scalarization method for trade-off analysis between two performance metrics. For large values of T we maximize the entropy, and, as T is lowered, we essentially transition from one Pareto point to another in a naturally occurring direction that resembles an annealing process. In this regard, the entropy H, which is closely related to the "purity" of the clusters, acts as a

regularization term which is given progressively less weight as T decreases.

As is the case in vector quantization, we minimize F via a coordinate block optimization algorithm. Minimizing F with respect to the association probabilities $p(\mu|x)$ is straightforward and yields the Gibbs distribution

$$p(\mu|x) = \frac{e^{-\frac{d(x,\mu)}{T}}}{\sum_{\mu} e^{-\frac{d(x,\mu)}{T}}} \tag{10}$$

while, in order to minimize F with respect to the codevector locations μ we set the gradients to zero

$$\frac{d}{d\mu} D = 0 \implies \frac{d}{d\mu} \mathbb{E}\left[\mathbb{E}\left[d(X,\mu)|X\right]\right] = 0 \tag{11}$$

This optimization procedure takes place for decreasing values of the temperature coefficient T such that the solution maintains minimum free energy (thermal equilibrium) while gradually lowering the temperature. Adding to the physical analogy, it is significant that, as the temperature is lowered, the system undergoes a sequence of "phase transitions", which consists of natural cluster splits where the cardinality of the codebook (number of prototypes) increases. This is a bifurcation phenomenon that provides a useful tool for controlling the size of the model relating it to the scale of the solution. At very high temperature ($T \to \infty$) the optimization yields uniform association probabilities $p(\mu|x) = 1/\kappa$, and, all the codevectors are located at the same point. As we lower the temperature, the cardinality of the codebook changes. The bifurcation can be traced by generating a perturbed pair of codevectors for each effective cluster, which, after convergence, can either merge together or get separated, depending on whether a phase transition has occurred [10].

B. Bregman Divergences as Dissimilarity Measures

The proximity measure d need not be a metric, and can be generalized to more general dissimilarity measures inspired by information theory and statistical analysis. In particular, Bregman divergences can offer numerous advantages in learning applications compared to the Euclidean distance alone [15]. Examples of Bregman divergences include the squared Euclidean distance, and the Kullback-Leibler divergence, and formally, they are defined by $d_\phi(x,\mu) = \phi(x) - \phi(\mu) - \frac{\partial \phi}{\partial \mu}(\mu)(x-\mu)$, where $\phi : S \to \mathbb{R}$ is a strictly convex twice F-differentiable function on a vector space $S \subseteq \mathbb{R}^d$, and $x, \mu \in S$. Notably, in the case of deterministic annealing Bregman divergences play an even more important role, since we can show that, if d is a Bregman divergence, the solution to the second optimization step (11) can be analytically computed in a convenient centroid form:

Theorem 1 ([10]): Assuming the conditional probabilities $p(\mu|x)$ are constant, the Langragian F in (9) is minimized with respect to the codevector locations μ by

$$\mu^* = \mathbb{E}\left[X|\mu\right] \tag{12}$$

if $d := d_\phi$ is a Bregman divergence for an appropriately defined function ϕ.

C. The online learning rule

The deterministic annealing algorithm is an offline algorithm, since the approximation of the conditional expectation $\mathbb{E}\left[X|\mu\right]$ is computed by the sample mean of the data points weighted by their association probabilities $p(\mu|x)$. To define an online training rule for the deterministic annealing framework, we formulate a stochastic approximation algorithm to recursively estimate $\mathbb{E}\left[X|\mu\right]$ directly.

Theorem 2 ([10]): Let S be a vector space, $\mu \in S$, and $X : \Omega \to S$ be a random variable defined in a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Let $\{x_n\}$ be a sequence of independent realizations of X, and $\{\alpha(n) > 0\}$ a sequence of stepsizes such that $\sum_n \alpha(n) = \infty$, and $\sum_n \alpha^2(n) < \infty$. Then the random variable $m_n = \sigma_n/\rho_n$, where (ρ_n, σ_n) are sequences defined by

$$\begin{aligned} \rho_{n+1} &= \rho_n + \alpha(n)\left[p(\mu|x_n) - \rho_n\right] \\ \sigma_{n+1} &= \sigma_n + \alpha(n)\left[x_n p(\mu|x_n) - \sigma_n\right], \end{aligned} \tag{13}$$

converges to $\mathbb{E}\left[X|\mu\right]$ almost surely, i.e. $m_n \xrightarrow{a.s.} \mathbb{E}\left[X|\mu\right]$.

As a direct consequence of this theorem, the following corollary provides an online learning rule that solves the optimization problem of the deterministic annealing algorithm.

Corollary 2.1 ([10]): The online training rule

$$\begin{cases} \rho_i(n+1) &= \rho_i(n) + \beta(n)\left[\hat{p}(\mu_i|x_n) - \rho_i(n)\right] \\ \sigma_i(n+1) &= \sigma_i(n) + \beta(n)\left[x_n \hat{p}(\mu_i|x_n) - \sigma_i(n)\right] \end{cases} \tag{14}$$

where $\sum_n \beta(n) = \infty$, $\sum_n \beta^2(n) < \infty$, and the quantities $\hat{p}(\mu_i|x_n)$ and $\mu_i(n)$ are recursively updated as follows:

$$\mu_i(n) = \frac{\sigma_i(n)}{\rho_i(n)}, \quad \hat{p}(\mu_i|x_n) = \frac{\rho_i(n) e^{-\frac{d(x_n,\mu_i(n))}{T}}}{\sum_i \rho_i(n) e^{-\frac{d(x_n,\mu_i(n))}{T}}} \tag{15}$$

converges almost surely to a solution of the block optimization (10), (12).

The learning rule (14), (15) is a stochastic approximation algorithm [21] which can be used for progressive graph partitioning in the vector space defined by the span of the eigenvectors $\left\{v_i^{(m)}\right\}_{m \in M}$ of L, or the spectral features $\hat{x}_i$ as defined in Section II. Notably, the size of $\{\mu_i\}_i$ progressively increases with respect to a trade-off between accuracy and complexity. At every temperature level T, perturbations of the existing codevectors are introduced, which, after convergence (i.e. $d_\phi(\mu_n^i, \mu_{n-1}^i) < \epsilon_c$, $\forall i$, for some $\epsilon_c > 0$), will merge back together (i.e. $d_\phi(\mu^j, \mu^i) < \epsilon_n$, $\forall i, j, i \neq j$, for some $\epsilon_n > 0$) or get separated, signifying bifurcation. At the same time, idle codevectors can also be detected, since, as a natural consequence of the stochastic approximation learning rule, the approximated probability $p(\mu_i)$ becomes negligible (i.e. $p(\mu_i) < \epsilon_r$, for some $\epsilon_r > 0$) if μ_i is largely dissimilar to the majority of the observed data points.

In terms of hyper-parameters, the temperature schedule T, the stepsizes α_n, and the constants ϵ_c, ϵ_n, ϵ_r, δ used to determine practical convergence, are all parameters that may affect the behavior of the algorithm. In particular, the temperature schedule and the parameter ϵ_n which is used to determine if two codevectors are similar enough

to be merged, play an important role in the complexity-accuracy trade-off of the algorithm. The complete Algorithm 1 is shown bellow, and parameter selection is discussed thoroughly in [10].

Algorithm 1 Online Deterministic Annealing [10]

Set d_ϕ, T_{max}, T_{min}, γ, K_{max}, $\{\alpha_n\}$, ϵ_p, ϵ_c, ϵ_n, ϵ_r, δ
Initialize $\{\mu^i\}$, $K=1$, $T=T_{max}$, $p(\mu^i)=1$,
$\sigma(\mu^i)=\mu^i p(\mu_i)$, $\forall i$
while $K<K_{max}$ **and** $T>T_{min}$ **do**
Perturb $\mu^i \leftarrow \{\mu^i+\delta, \mu^i-\delta\}$, $\forall i$
Increment $K \leftarrow 2K$
Update $p(\mu^i)$, $\sigma(\mu^i) \leftarrow \mu^i p(\mu^i)$, $\forall i$
Set $n \leftarrow 0$
repeat
Observe data point x
for $i=1,\dots,K$ **do**
Update:

$$p(\mu^i|x) \leftarrow \frac{p(\mu^i)e^{-\frac{d_\phi(x,\mu^i)}{T}}}{\sum_i p(\mu^i)e^{-\frac{d_\phi(x,\mu^i)}{T}}}$$

$$p(\mu^i) \leftarrow p(\mu^i)+\alpha_n\left[p(\mu^i|x)-p(\mu^i)\right]$$

$$\sigma(\mu^i) \leftarrow \sigma(\mu^i)+\alpha_n\left[xp(\mu^i|x)-\sigma(\mu^i)\right]$$

$$\mu^i \leftarrow \frac{\sigma(\mu^i)}{p(\mu^i)}$$

Increment $n \leftarrow n+1$
end for
until $d_\phi(\mu_n^i, \mu_{n-1}^i) < \epsilon_c$, $\forall i$
Keep effective codevectors:
discard μ^i if $d_\phi(\mu^j, \mu^i) < \epsilon_n$, $\forall i,j, i \neq j$
Remove idle codevectors:
discard μ^i if $p(\mu^i) < \epsilon_r$, $\forall i$
Update K, $p(\mu^i)$, $\sigma(\mu^i)$, $\forall i$
Lower temperature $T \leftarrow \gamma T$
end while

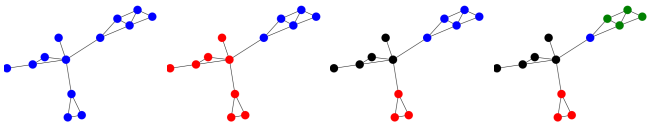
IV. Experiments

We illustrate the properties and evaluate the performance of the proposed methodology in graph partition and image segmentation applications.

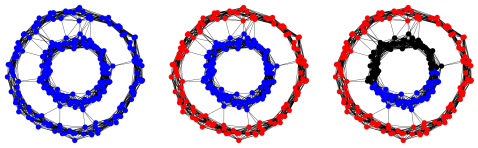
A. Graph Clustering

We showcase how the proposed methodology works, by presenting, in Fig. 1, the output of the proposed graph clustering algorithm in three toy graph examples. Unless stated otherwise, the results presented will correspond to the progressive clustering algorithm Alg. 1 using the spectral features $\{\hat{x}_i\}_i$ of (8), calculated locally by each node via the distributed computation algorithm detailed in Section II.

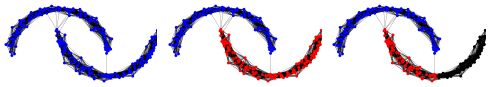
In Fig. 1, the number of clusters K increases from left to right, as the temperature coefficient T decreases. The rightmost partitions are shown only for illustration purposes. They correspond to very low values of T and, under a reasonable choice of T_{min}, Alg. 1 would have stopped before reaching this configuration. Notably, the partitions shown in Fig. 1 are identical to those produced by spectral clustering [9], and by the k-means algorithm trained on the features $\{\hat{x}_i\}_i$ of (8).



(a) Simple graph (13 nodes).



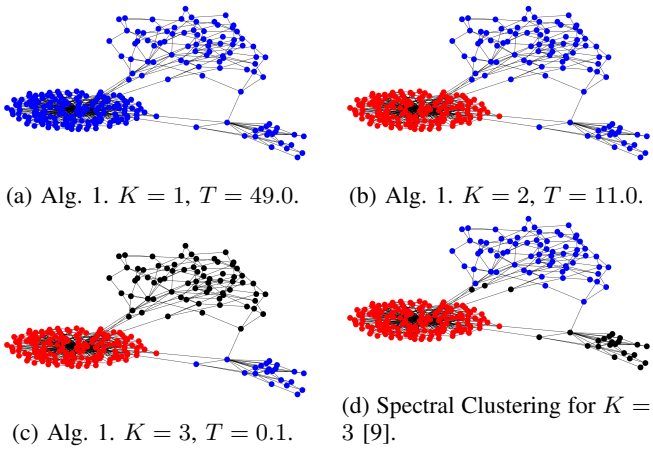
(b) Concentric circles (250 nodes, 25-nn).



(c) Half moons (250 nodes, 25-nn).

Fig. 1: Toy examples of graph clustering using the proposed methodology: (8) and Alg. 1. The number of clusters K increases as temperature T decreases from left to right. The rightmost partition corresponds to very low values of T and is typically discarded. The results align with those of traditional spectral clustering.

In a more interesting example, we use the proposed methodology to partition a community graph from the benchmark introduced in [22]. The results are shown in Fig. 2. Once again, the partitions shown in Fig. 2 are almost identical to those produced by spectral clustering [9], and seem to be a little better. For reference, the results are identical if we use the k-means algorithm trained on the features $\{\hat{x}_i\}_i$ of (8) for $k=3$. However, in k-means, there is no way to know a priori the number of clusters K, especially if the graph is not known and the spectral features are computed in a distributed manner. In contrast, Alg. 1 automatically decides the number K, by progressively adjusting to the observations.



(a) Alg. 1. $K=1$, $T=49.0$.

(b) Alg. 1. $K=2$, $T=11.0$.

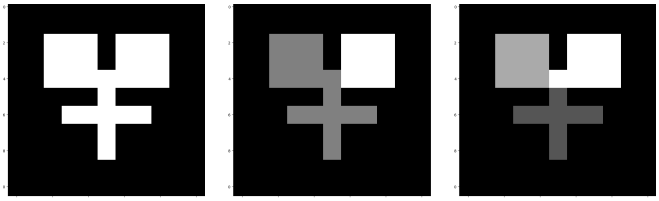
(c) Alg. 1. $K=3$, $T=0.1$.

(d) Spectral Clustering for $K=3$ [9].

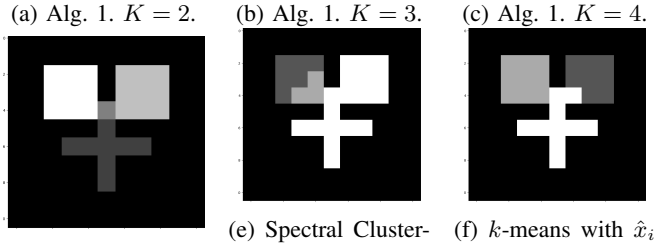
Fig. 2: Example of community graph partitioning [22]. Here 250 nodes form 3 communities.

B. Image Segmentation

A common application of spectral clustering is image segmentation. In Fig. 3, we present a simple multi-class image segmentation task. The picture has been converted to a graph with each pixel assumed adjacent to all pixels that are less than $n = 3$ pixels away in any direction, with a weight W_{ij} depending on the similarity of the intensity of the image. Despite the simplicity of the image, spectral clustering fails to infer the underlying structure. The use of the proposed features (8) with k-means seems to outperform spectral clustering, as does Alg. 1, which progressively infers the number of clusters.



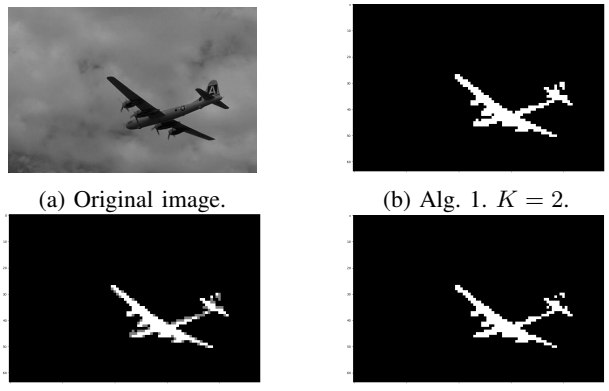
(a) Alg. 1. $K = 2$. (b) Alg. 1. $K = 3$. (c) Alg. 1. $K = 4$.



(d) Alg. 1. $K = 5$. (e) Spectral Clustering for $K = 4$. (f) k-means with $\hat{x}_i$ for $K = 4$.

Fig. 3: Simple multi-class image segmentation.

Lastly, in Fig. 4, we experiment with a real image from the Berkeley Segmentation Dataset [23]. The image depicts an airplane flying on the sky. After downsampling the image to a resolution of 64×96, in order to reduce the computational time, we convert it to a graph in a similar way. Alg. 1 successfully detects the airplane for $K = 2$, and its boundaries for $K = 3$.



(a) Original image. (b) Alg. 1. $K = 2$.

(c) Alg. 1. $K = 3$. (d) Spectral Clustering, $K = 2$.

Fig. 4: Image segmentation [23].

V. Conclusion

We proposed a progressive graph partitioning algorithm based on a distributed approximation of the spectral information of the underlying graph Laplacian matrix and a novel online deterministic annealing algorithm. The experimental results suggest that the proposed approach can be effectively used for progressive partitioning of large graphs and image segmentation applications.

References

[1] C. N. Mavridis, A. Tirumalai, and J. S. Baras, "Learning interaction dynamics from particle trajectories and density evolution," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020.

[2] C. N. Mavridis, N. Suriyarachchi, and J. S. Baras, "Detection of dynamically changing leaders incomplex swarms from observed dynamic data," in *2020 Conference on Decision and Game Theory for Security (GameSec)*, 2020.

[3] Z. Lu, Y. Zhu, W. Li, W. Wu, and X. Cheng, "Influence-based community partition for social networks," *Computational Social Networks*, vol. 1, no. 1, pp. 1–18, 2014.

[4] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *(CVPR'05)*, vol. 2. IEEE, 2005.

[5] A. Paccanaro, J. A. Casbon, and M. A. Saqi, "Spectral clustering of protein sequences," *Nucleic acids research*, vol. 34, no. 5, 2006.

[6] G. Wang, Y. Zhao, J. Huang, Q. Duan, and J. Li, "A k-means-based network partition algorithm for controller placement in software defined network," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.

[7] M. Fiedler, "A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory," *Czechoslovak Mathematical Journal*, vol. 25, no. 4, pp. 619–633, 1975.

[8] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," in *Advances in neural information processing systems*, 1995, pp. 585–592.

[9] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[10] C. Mavridis and J. Baras, "Online deterministic annealing for classification and clustering," 2021.

[11] C. N. Mavridis and J. S. Baras, "Maximum-entropy input estimation for gaussian processes in reinforcement learning," in *Conference on Decision and Control (CDC)*. IEEE, 2021.

[12] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *Journal of Computer and System Sciences*, vol. 74, no. 1, pp. 70–83, 2008.

[13] T. Sahai, A. Speranzon, and A. Banaszuk, "Hearing the clusters of a graph: A distributed algorithm," *Automatica*, vol. 48, no. 1, 2012.

[14] M. Franceschelli, A. Gasparri, A. Giua, and C. Seatzu, "Decentralized estimation of laplacian eigenvalues in multi-agent systems," *Automatica*, vol. 49, no. 4, pp. 1031–1036, 2013.

[15] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with bregman divergences," *Journal of machine learning research*, vol. 6, no. Oct, pp. 1705–1749, 2005.

[16] E. Mwebaze, P. Schneider, F.-M. Schleif, J. R. Aduwo, J. A. Quinn, S. Haase, T. Villmann, and M. Biehl, "Divergence-based classification in learning vector quantization," *Neurocomputing*, 2011.

[17] M. Belkin and P. Niyogi, "Towards a theoretical foundation for laplacian-based manifold methods," *Journal of Computer and System Sciences*, vol. 74, no. 8, pp. 1289–1308, 2008.

[18] C. N. Mavridis and J. S. Baras, "Convergence of stochastic vector quantization and learning vector quantization with bregman divergences," in *21rst IFAC World Congress*. IFAC, 2020.

[19] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, 1998.

[20] E. T. Jaynes, "Information theory and statistical mechanics," *Physical review*, vol. 106, no. 4, p. 620, 1957.

[21] V. S. Borkar, *Stochastic approximation: a dynamical systems viewpoint*. Springer, 2009, vol. 48.

[22] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical review E*, vol. 78, no. 4, p. 046110, 2008.

[23] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 416–423.