Fast and Simple Multiclass Data Segmentation: An Eigendecomposition and Projection-Free Approach

Fast and Simple Multiclass Data Segmentation: An Eigendecomposition and Projection-Free Approach*

Chiara Faccio†, Margherita Porcelli‡, Francesco Rinaldi†, and Martin Stoll§

Abstract. Graph-based machine learning has seen an increased interest over the last decade with many connections to other fields of applied mathematics. Learning based on partial differential equations, such as the phase-field Allen-Cahn equation, allows efficient handling of semi-supervised learning approaches on graphs. The numerical solution of the graph Allen-Cahn equation via a convexity splitting or the Merriman-Bence-Osher (MBO) scheme, albeit being a widely used approach, requires the calculation of a graph Laplacian eigendecomposition and repeated projections over the unit simplex to maintain valid partitions. The computational efficiency of those methods is hence limited by those two bottlenecks in practice, especially when dealing with large-scale instances.

In order to overcome these limitations, we propose a new framework combining a novel penalty-based reformulation of the segmentation problem, which ensures valid partitions (i.e., binary solutions) for appropriate parameter choices, with an eigendecomposition and projection-free optimization scheme, which has a small per-iteration complexity (by relying primarily on sparse matrix-vector products) and guarantees good convergence properties.

Experiments on synthetic and real-world datasets related to data segmentation in networks and images demonstrate that the proposed framework achieves comparable or better accuracy than the CS and MBO methods while being significantly faster, particularly for large-scale problems.

Key words. Frank–Wolfe methods, graph-based learning, greedy algorithms

1. Motivation. Many problems can be formulated as learning problems defined on a graph-domain (see, e.g., [14] for a detailed discussion on this matter). In this paper, a graph $G = (V, E)$ consists of a set of vertices V with $|V| = n$ and an edge set E as a subset of $V \times V$. The graphs we consider are undirected and consist of one component. In the case of multiple components, the methods proposed here can be used for the individual subgraphs. The focus of our work is a semi-supervised learning framework where a subset of the nodes are assigned to a label and one wants to infer the label of the remaining nodes. While in binary classification one assumes to have only two labels, the focus of this work is on assigning each node to one of $K > 2$ labels.

A key ingredient in graph-based learning is the representation of the graph information in terms of linear algebra components. For this, one typically considers the weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{n,n}$ where $w_{i,j} \geq 0$ describes the strength of the connection between the different

*Submitted to the editors August 13, 2025.

Funding: The research of M.P. was partially granted by the Italian Ministry of University and Research (MUR) through the PRIN 2022 "MOLE: Manifold constrained Optimization and LEarning", code: 2022ZK5ME7 MUR D.D. financing decree n. 20428 of Nov. 6th, 2024 (CUP B53C24006410006), and by PNRR - Missione 4 Istruzione e Ricerca - Componente C2 Investimento 1.1, Fondo per il Programma Nazionale di Ricerca e Progetti di Rilevante Interesse Nazionale (PRIN) funded by the European Commission under the NextGeneration EU programme, project "Advanced optimization METhods for automated central veIn Sign detection in multiple sclerosis from magneTic resonAnce imaging (AMETISTA)", code: P2022J9SNP, MUR D.D. financing decree n. 1379 of 1st Sept. 2023 (CUP E53D23017980001). Her work was also partially supported by the INdAM-GNCS project CUP_E53C24001950001. The research of F.R. has been partially funded by the EuropeanUnion - NextGenerationEU under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.1 - Call PRIN 2022 No. 104 of February 2, 2022 of Italian Ministry of University and Research; Project 2022BMBW2A (subject area: PE - Physical Sciences and Engineering)"Large-scale optimization for sustainable and resilient energy systems."

†Department of Mathematics "Tullio Levi-Civita" University of Padova, Italy. C.F. is member of the INdAM Research Group GNCS (rinaldi@math.unipd.it, chiara.faccio@unipd.it)

‡Dipartimento di Ingegneria Industriale (DIEF) Università degli Studi di Firenze, Italy, ISTI–CNR, Italy and member of the INdAM Research Group GNCS (margherita.porcelli@unifi.it)

§Department of Mathematics, Chemnitz University of Technology, Germany (martin.stoll@math.tu-chemnitz.de)

nodes. The weight between two nodes is often modeled using the Gaussian similarity measure

$$w_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right),$$

where σ is a chosen hyperparameter, x_i and x_j are two d dimensional feature vectors associated with the nodes v_i and v_j, respectively. As a measure of importance of the nodes in the network one typically computes the degree d_i for node v_i using $d_i = \sum_{j=1}^{n} w_{ij}$, which corresponds to the row-sum of the weight matrix $\mathbf{W}$. This leads to the definition of the *graph Laplacian*

$$\mathbf{L} = \mathbf{D} - \mathbf{W},$$

where $\mathbf{D}$ is a diagonal matrix collecting the degrees of all nodes. The graph Laplacian has been used extensively and we refer to [16, 37] for excellent reviews on the graph Laplacian and its applications. One often considers versions of the Laplacian, namely, the symmetric normalized Laplacian $\mathbf{L}_s = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ or the random walk Laplacian $\mathbf{L}_w = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}$, where the assumption made on the graph guarantees the invertibility of the matrix $\mathbf{D}$.

In semi-supervised learning, the node labels are given for a potentially small subset of the nodes and we want to learn the labels of the remaining nodes while simultaneously leveraging labeled and unlabeled nodes. To solve this problem, approaches based on partial differential equations (PDEs) have become popular (see, e.g., [7, 20]) and while these often show very good performance, they typically are implemented relying on the a priori computation of a suitably chosen subset of the eigenvalues and eigenvectors of the graph Laplacian to reduce the complexity during the solution process. The computation of eigeninformation typically requires the use of a Krylov subspace, which in turns requires matrix vector products with the graph Laplacian. Low-rank approaches such as the Nyström approximation are often also based on matrix vector products with the graph Laplacian, especially when modern randomized linear algebra methods [24] are employed. This means that the truncated eigendecomposition needed for an efficient implementation of the PDE-based schemes like, e.g., the convexity splitting (CS) [36] and Merriman-Bence-Osher (MBO) [26, 36] approach considered in this paper, have non-negligible startup cost. This cannot be avoided as otherwise the mentioned schemes would require the solution of linear systems involving shifted versions of the large-scale graph Laplacian. While this decomposition can obviously be precomputed, the final accuracy and computational efficiency of the methods strongly depend on the number of retained eigenpairs, an unknown hyperparameter that is very expensive to tune and, as we will see, varies depending on the problem structure. In this work, both the diffuse interface and the MBO scheme are considered as reference methods among PDE-based approaches as these have previously been compared favorably to other graph-based learning schemes in [20]. For certain imaging classification task these methods have been compared to neural network based approaches [4] producing similar accuracy while only requiring a fraction of the computational cost.

Furthermore, these PDE-based schemes require at each iteration a tailored projection step to guarantee that the related iterate represents a valid partition. As we will see later on, such a step has a $\mathcal{O}(nK \log K)$ cost for a problem with n data points and K classes. While this may be still doable for small-scale instances, it becomes prohibitively slow when dealing with large-scale ones, like, e.g., social networks or high-resolution images, where n and K can easily reach millions and thousands, respectively. We hence need a way to get rid of these two bottlenecks in practice.

Contribution. The contribution of this work is threefold. First, leveraging on the graph-based semi-supervised learning model proposed in [7], we define a new and easier to handle

penalty-based continuous formulation of the multiclass data segmentation problem. We analyze its theoretical properties and prove that the model obtains valid partitions (i.e., binary solutions) for a sufficiently small value of the penalty parameter.

Second, we develop a new algorithm in the class of Frank-Wolfe methods [10], named Greedy Frank-Wolfe (GFW), which suitably exploits the structure of the proposed model to keep the per-iteration cost small enough (by relying only on tailored sparse matrix-vector products) and allows us to avoid both the projection steps and eigendecomposition computations, which represent, as we highlighted above, the main bottlenecks for the CS and MBO approaches. We analyze the convergence rate of the method and discuss the conditions under which the method achieves one-shot convergence (i.e., convergence in one iteration). Given appropriate assumptions, we also provide a detailed complexity analysis of the proposed algorithm. Such a result, to the best of our knowledge, is not available for the PDE-based schemes considered in the paper.

Finally, we show the usefulness of the proposed framework. We provide experimental results on both benchmark data sets and real data (real networks and image labeling), demonstrating the effectiveness of our method compared to CS and MBO, both in terms of computational time and quality of the segmentation.

Organization of the paper. The remainder of this paper is organized as follows. In Section 2, we introduce the mathematical models for multiclass data segmentation on graphs. Starting from classical diffuse interface models based on the Ginzburg–Landau energy, we derive the novel penalty-based continuous reformulation of the problem and analyze its theoretical properties. In Section 3, we briefly review two established PDE-inspired methods widely used for multiclass data segmentation, i.e., the convexity splitting (CS) and Merriman-Bence-Osher (MBO) schemes and discuss their computational bottlenecks. Section 4 presents the greedy variant of the Frank-Wolfe algorithm, suitably designed to exploit the structure of our new model, discusses the convergence properties of the method, and analyzes its computational complexity. In Section 5, we report numerical experiments that compare our method with CS and MBO on a variety of synthetic and real datasets, including network data and image segmentation tasks. Finally, Section 6 presents our conclusions and discusses possible directions for future research.

2. Models for multiclass segmentation.

Learning methods based on partial differential equation (PDE) concepts have gained more traction in the recent years, while machine learning methods for PDEs have also become a major topic in computational science and engineering [18]. The objective function for the semi-supervised learning method we propose is derived from diffuse interface methods. Diffuse interface methods are a classical tool in the simulation of materials science problems [2, 13]. The derivation of classical models such as the Allen–Cahn [2] or Cahn–Hilliard equations [13] is typically obtained from a gradient flow of the Ginzburg–Landau energy

$$E_{GL}(u) = \int_\Omega \frac{\mu}{2} |\nabla u|^2 \, dx + \int_\Omega \frac{1}{\mu} \psi(u) dx \tag{2.1}$$

where u is the phase-field function and μ the interface parameter, which is typically assumed to be small and related to the interface width describing the transition from one pure phase to the other within the domain Ω. The function $\psi(u)$ is a potential that forces the phase-field u to take values at either $u \approx -1$ or $u \approx 1$ (assuming two pure phases). We come back to the discussion of the choice of potential as this is one of the contributions of this paper. The minimization of the energy $E_{GL}(u)$ follows a gradient flow, i.e.,

$$\partial_t u = -\text{grad}(E_{GL}(u)).$$

Bertozzi and Flenner [7] formulated the semi-supervised learning problem based on graph quantities. The energy function to minimize is now defined using the description of the underlying graph Laplacian $\mathbf{L}_s$ to give

$$E_{GL}(\mathbf{u}) = \frac{\mu}{2}\mathbf{u}^\top \mathbf{L}_s \mathbf{u} + \sum_{x\in V} \frac{1}{\mu}\psi(\mathbf{u}(x)) + F_\omega(f,\mathbf{u}), \tag{2.2}$$

with the energy contribution $F_\omega(f,\mathbf{u})$ describing the fidelity term that would lead to $\omega(x)(f-\mathbf{u})$ (cf. [7]) in the continuous Allen–Cahn equation often used for image inpainting applications [5, 12]. In practice, often more than two components are of interest. The continuous Ginzburg–Landau energy for two components in (2.1) generalizes to

$$E_{GL}(u) = \int_\Omega \frac{\mu}{2}\sum_{i=1}^{K} |\nabla u_i|^2 dx + \int_\Omega \frac{1}{\mu}\psi(u)dx, \tag{2.3}$$

for $K > 2$ components. Here, $u = (u_1,\dots,u_K)^\top$ is now the vector-valued phase-field, and the potential function $\psi(u)$ has K distinct minima instead of two. Note that with a slight abuse of notation use the subscript GL for all objective functions based on the Ginzburg-Landau energy.

Garcia-Cardona et al. [20] as well as Merkurjev et al. [25] have extended these continuous models to the graph domain. In the following, we briefly summarize their approach. We introduce the matrix $\mathbf{U} = (\mathbf{u}_1,\dots,\mathbf{u}_n)^\top \in \mathbb{R}^{n\times K}$. Here, the kth component of $\mathbf{u}_i \in \mathbb{R}^K$ is the strength for data point i to belong to class k. For each node i, the vector $\mathbf{u}_i$ has to be an element of the unit simplex:

$$\Delta_{K-1}^i := \{\mathbf{u}_i \in \mathbb{R}_+^K : \mathbf{1}^\top \mathbf{u}_i = 1\} = \mathrm{conv}\{\mathbf{e}_j : j \in [1\!:\!K]\},$$

so we have that $\mathbf{U}$ belongs to the set:

$$\Sigma = \{\mathbf{U} \in \mathbb{R}^{n\times K} : \mathbf{u}_i \in \Delta_{K-1}^i,\ i = 1,\dots,n\}. \tag{2.4}$$

In the text $\mathbf{1}$ is either a vector or matrix of all ones of appropriate dimensionality. The Ginzburg–Landau energy functional on graphs in (2.2) generalizes to the multi-class case as

$$E_{GL}(\mathbf{U}) = \frac{\mu}{2}\langle \mathbf{U}, \mathbf{L}_s\mathbf{U}\rangle + \frac{1}{\mu}\psi(\mathbf{U}) + F_\omega(\hat{\mathbf{U}},\mathbf{U}), \tag{2.5}$$

where the graph-cut term from the binary case is found as

$$\langle \mathbf{U}, \mathbf{L}_s\mathbf{U}\rangle = \mathrm{trace}(\mathbf{U}^\top \mathbf{L}_s \mathbf{U})$$

and measures variations in the vector field. The potential term is given via

$$\psi(\mathbf{U}) = \frac{1}{2}\sum_{i\in V}\left(\prod_{k=1}^{K}\frac{1}{4}||\mathbf{u}_i - \mathbf{e}_k||_{L_1}^2\right) \tag{2.6}$$

drives the system closer to the pure phases as the minimum is achieved if $\mathbf{u}_i$ is close to a unit vector, i.e., a pure phase. Finally, the fidelity term incorporating training data is given by

$$F_\omega(\hat{\mathbf{U}},\mathbf{U}) = \sum_{i\in V}\frac{\omega_i}{2}||\hat{\mathbf{u}}_i - \mathbf{u}_i||_{L_2}^2 = \frac{1}{2}\|\mathbf{D}_\omega^{1/2}(\hat{\mathbf{U}} - \mathbf{U})\|_F^2$$

as it enables the encoding of a priori information with $\hat{\mathbf{U}} = (\hat{\mathbf{u}}_1,\dots,\hat{\mathbf{u}}_n)^\top$. Indeed, ω_i is a parameter that takes the value of a positive constant ω_0 if i is a fidelity node and zero

otherwise, and $\hat{\mathbf{u}}_i$ is the known value of the fidelity node i. Here the matrix $\mathbf{D}_\omega$ is a diagonal matrix with elements ω_i on the diagonal.

Note that the authors of [20, 25] use an L_1-norm for the potential term as it prevents an undesirable local minimum from occurring at the center of the simplex, as would be the case with an L_2-norm for large K. One standard way to minimize the above given energies is to follow a gradient flow approach and for the graph case, one can also employ a gradient scheme as was done in [7, 11, 25].

2.1. A new penalty-based model for multiclass data segmentation. In here, we propose a different formulation where the term $\psi(\mathbf{U})$ enforcing the pure phases is replaced with a new penalty term, that is

$$\phi(\mathbf{U}) = \langle \mathbf{U}^\top, \mathbf{1} - \mathbf{U}^\top \rangle = \sum_{i \in V} \mathbf{u}_i^\top (\mathbf{1} - \mathbf{u}_i) = \operatorname{trace}(\mathbf{U}(\mathbf{1} - \mathbf{U}^\top)).$$

The Ginzburg–Landau energy functional on graphs in (2.2) with this new penalty term for the multi-class case can be written as follows

$$E(\mathbf{U}) = \frac{1}{2}\langle \mathbf{U}, \mathbf{L}_s \mathbf{U} \rangle + \frac{1}{\varepsilon}\phi(\mathbf{U}) + F_\omega(\hat{\mathbf{U}}, \mathbf{U}), \tag{2.7}$$

and the problem we have is

$$\operatorname{argmin}_{\mathbf{U} \in \Sigma} E(\mathbf{U}). \tag{2.8}$$

Both parameters μ and ε play a role in enforcing the pure phases, in the objectives (2.5) and (2.7), respectively. The parameter μ originates from materials science modeling and corresponds to a smooth transition from one phase to the other, the parameter ε is purely motivated as a penalty parameter in the objective function. We now calculate the diameter of the feasible set Σ, a technical result that will be useful for the analyses that follow. To improve readability and enhance the presentation flow, all proofs are reported in the Appendix.

Proposition 2.1. *The diameter of the set Σ is $\sqrt{2n}$.*

The new energy function in (2.7) takes the form

$$E(\mathbf{U}) = \frac{1}{2}\operatorname{trace}(\mathbf{U}^\top \mathbf{L}_s \mathbf{U}) + \frac{1}{\varepsilon}\operatorname{trace}(\mathbf{U}(\mathbf{1} - \mathbf{U}^\top)) + \frac{1}{2}\|\mathbf{D}_\omega^{1/2}(\hat{\mathbf{U}} - \mathbf{U})\|_F^2, \tag{2.9}$$

and we then obtain its gradient

$$\nabla E(\mathbf{U}) = \mathbf{L}_s \mathbf{U} + \frac{1}{\varepsilon}(\mathbf{1} - 2\mathbf{U}) - \mathbf{D}_\omega(\hat{\mathbf{U}} - \mathbf{U}). \tag{2.10}$$

It is important to observe that the objective has Lipschitz continuous gradient. In the next proposition, we report a bound on the Lipschitz constant of the gradient.

Proposition 2.2. *The gradient of $E(\mathbf{U})$ is Lipschitz continuous with constant*

$$L \leq \bar{L} = \lambda_{max}(\mathbf{L}_s) + \frac{2}{\varepsilon} + d_\omega^{max}, \tag{2.11}$$

with $\lambda_{max}(\mathbf{L}_s)$ maximum eigenvalue of the matrix $\mathbf{L}_s$ and $d_\omega^{max} = \max_i(\mathbf{D}_\omega)_{ii} = \omega_0$.

It is important to highlight that when ε is sufficiently small we can guarantee that the solution matrix $\mathbf{U}$ is binary, i.e., problem (2.8) is equivalent to

$$\operatorname{argmin}_{U \in \bar{\Sigma}} \bar{E}(\mathbf{U}), \tag{2.12}$$

with

$$\bar{E}(\mathbf{U}) = \frac{1}{2}\langle \mathbf{U}, \mathbf{L}_s \mathbf{U} \rangle + F_\omega(\hat{\mathbf{U}}, \mathbf{U})$$

and $\bar{\Sigma} = \{\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_n)^\top \in \mathbb{R}^{n,K} : \mathbf{1}^\top \mathbf{u}_i = 1,\ \mathbf{u}_i \in \{0,1\}^K,\ \forall\, i \in [1:n]\}$. This equivalence result is stated below.

Theorem 2.3. *Let $\bar{\varepsilon} < 2/\lambda_{max}(\mathbf{L}_s + \mathbf{D}_\omega)$. Then, for all $\varepsilon \in (0, \bar{\varepsilon}]$, we have:*
(i) Every local/global minimizer $\mathbf{U}^*$ of Problem (2.8) is strict and such that $\mathbf{U}^* \in \{0,1\}^{n\times K}$.
(ii) Problems (2.8) and (2.12) have the same global minimizers, that is

$$\operatorname{argmin}_{U\in\Sigma} E(\mathbf{U}) = \operatorname{argmin}_{U\in\bar{\Sigma}} \bar{E}(\mathbf{U})\,.$$

The above result therefore guarantees that even when using a local solver to handle the problem (2.8), we are able to obtain a solution that has binary components in the end. Although our formulation is primarily designed to produce hard-membership solutions, selecting a moderately large value for the parameter ε naturally leads to fractional values. In the same way as with softmax-based scores, these values can therefore be seen as soft memberships or pseudo-confidences. It is also important to note that the upper bound $2/\lambda_{max}(\mathbf{L}_s + \mathbf{D}_\omega)$ on the parameter ε just enforces a theoretical guarantee rather than representing a requirement for practical use. In all of our experiments, the method indeed showed robustness with respect to a broad range of parameter values, and a simple inexpensive tuning procedure was sufficient even on large graphs.

Before introducing our Frank-Wolfe scheme, we briefly recall two of the most popular graph-based methods for multiclass graph segmentation.

3. Existing PDE-based approaches: convexity splitting and MBO scheme. A possible way to address the multi-class problem is by means of a convexity splitting method [6, 11, 20], which involves decomposing the energy function E_{GL} in (2.5) into convex and concave parts:

$$E_{GL}(\mathbf{U}) = E_1(\mathbf{U}) - E_2(\mathbf{U})$$

with

$$E_1(\mathbf{U}) = \frac{\mu}{2}\langle \mathbf{U}, \mathbf{L}_s\mathbf{U}\rangle + \frac{c}{2}\langle \mathbf{U}, \mathbf{U}\rangle, \quad E_2(\mathbf{U}) = -\frac{1}{\mu}\psi(\mathbf{U}) - F(\hat{\mathbf{U}}, \mathbf{U}) + \frac{c}{2}\langle \mathbf{U}, \mathbf{U}\rangle,$$

with $c \in \mathbb{R}$ denoting a constant that is chosen to guarantee the convexity/concavity of the energy terms. Indeed, we require $c \geq \omega_0 + \frac{1}{\mu}$; see [20, p. 6]. The convexity splitting scheme results in

$$\frac{\mathbf{U}^{\ell+1} - \mathbf{U}^\ell}{\tau} + \mu\mathbf{L}_s\mathbf{U}^{\ell+1} + c\mathbf{U}^{\ell+1} = -\frac{1}{2\mu}T(\mathbf{U}^\ell) + c\mathbf{U}^\ell + \mathbf{D}_\omega(\hat{\mathbf{U}} - \mathbf{U}^\ell), \tag{3.1}$$

where the elements T_{ij} of the matrix $T(\mathbf{U}^\ell)$ are given as

$$T_{ij} = \sum_{l=1}^{K} \frac{1}{2}\left(1 - 2\delta_{jl}\right) ||\mathbf{u}_i^\ell - \mathbf{e}_l||_{L_1} \prod_{m=1, m\neq l}^{K} \frac{1}{4}||\mathbf{u}_i^\ell - \mathbf{e}_m||_{L_1}^2 \tag{3.2}$$

and $\tau > 0$ is the time step.

Here, ℓ indicates the old time-step and $\ell+1$ the new time-step. Using the eigendecomposition of the graph Laplacian as $\mathbf{L}_s = \Phi\Lambda\Phi^\top$, $\mathbf{U}$ can be expressed in this basis via $\mathbf{U} = \Phi_k\mathcal{U}$ with $\Phi_k \in \mathbb{R}^{n,k}$ collecting the dominating eigenvalues of the graph Laplacian and $\mathcal{U}$ the coefficients for the expansion in those vectors. By plugging this into the above equation and multiplying on the left by $\Phi_k^\top$ to obtain the reduced equation

$$\mathcal{U}^{\ell+1} = B_k^{-1}\left[(1 + c\tau)\mathcal{U}^\ell - \frac{\tau}{2\mu}\Phi_k^\top T(\mathbf{U}^\ell) + \tau\Phi_k^\top\mathbf{D}_\omega(\hat{\mathbf{U}} - \mathbf{U}^\ell)\right]. \tag{3.3}$$

where $B_k = (1+c\tau)I + \mu\tau\Lambda_k$ and Λ_k is the diagonal matrix of k dominating eigenvalues. Since B_k is a diagonal matrix with positive entries, its inverse is easy to apply. At every iteration,

i.e., time-step, every ith row of the approximation $\mathcal{U}^{\ell+1}$ has to be projected onto the unit simplex Δ^i_{K-1} to avoid explosion of the solution coefficients (see, e.g., Algorithm 3.1).

Alternatively, one can use the Merriman-Bence-Osher (MBO) [27] scheme, which was originally used for the binary classification in [25]. It is based on a two-step procedure for the continuous model based on a *diffusion step:* Obtain $u^{\ell+\frac{1}{2}}$ from the solution of

$$\frac{\partial u}{\partial t} = \Delta u,$$

followed by a *thresholding step:* Compute

$$u^{\ell+1} = \begin{cases} 1 & : u^{\ell+\frac{1}{2}} \geq 0, \\ -1 & : u^{\ell+\frac{1}{2}} < 0. \end{cases}$$

This scheme was later extended to the case of graph-based semi-supervised learning in [7] for the binary classification case and for multiclass case in [20]. There, the authors propose the following procedure. Perform a *diffusion step:* Obtain $\mathbf{U}^{\ell+\frac{1}{2}}$ from

$$\frac{\mathbf{U}^{\ell+\frac{1}{2}} - \mathbf{U}^{\ell}}{\tau} = -\mathbf{L}_s \mathbf{U}^{\ell+\frac{1}{2}} - \mathbf{D}_\omega(\hat{\mathbf{U}} - \mathbf{U}^{\ell}),$$

and then perform a *thresholding step:* Compute

$$\mathbf{u}_i^{\ell+1} = e_k,$$

where e_k is the vertex that is closest to the projection of $\mathbf{u}_i^{\ell+\frac{1}{2}}$ to the simplex. This problem can again be tackled by using a truncated eigendecomposition. We then obtain

$$\mathcal{U}^{\ell+\frac{1}{2}} = (\mathbf{I} + \tau\Lambda_k)^{-1} \left[\mathcal{U}^{\ell} - \tau\Phi_k^\top \mathbf{D}_\omega(\hat{\mathcal{U}} - \Phi_k\mathcal{U}^{\ell})\right]. \tag{3.4}$$

Instead of the updating for one step we can also perform a number of such steps via

- $\mathcal{U}^{\ell+\frac{1}{2}} = (\mathbf{I} + \tau\Lambda_k)^{-1} \left[\mathcal{U}^{\ell} - \tau\Phi_k^\top \mathbf{D}_\omega(\hat{\mathcal{U}} - \Phi_k\mathcal{U}^{\ell})\right]$
- $\mathcal{U}^{\ell+1} = \text{projspx}(\mathcal{U}^{\ell+\frac{1}{2}})$
- $\mathcal{U}^{\ell} = \mathcal{U}^{\ell+1}$,

where projspx denotes the row-wise projection onto the unit simplex (see Algorithm 3.1). Once the reduced eigendecomposition (Λ_k, Φ_k) is available, the computational cost per step of MBO is small as a very small linear system has to be solved.

In both the approaches, CS and MBO, if the cost of the eigendecomposition is negligible, the computational bottleneck is the projection onto the unit simplex.

We conclude this section with the most common procedure to project a vector $y \in \mathbb{R}^K$ onto the simplex Δ^i_{K-1}, commonly used in the implementation of CS and MBO. The algorithm is the sorting-based method (see,e.g., [17] and [21] for further details) summarized in Algorithm 3.1.

Algorithm 3.1 `Projection onto the unit simplex` [15, Algorithm 1 (`projsplx`)]

1 Input $Y = (y_1, \dots, y_K)^\top \in \mathbb{R}^K$.
2 Sort y as $\hat{y}$ in the ascending order as $\hat{y}_1 \leq \dots \leq \hat{y}_K$, and set $i = K - 1$.
3 Compute

$$t_i = \frac{\left(\sum_{j=i+1}^{K} \hat{y}_j\right) - 1}{K - i}.$$

If $t_i \geq \hat{y}_i$ set $\hat{t} = t_i$ and go to Step 5.
Else set $i := i - 1$ and go to Step 3 if $i \geq 1$, or go to Step 4 if $i = 0$.
4 Compute

$$\hat{t} = \frac{\left(\sum_{j=i+1}^{\ell} \hat{y}_j\right) - 1}{K - i}.$$

5 Return $x = (y - \hat{t})_+$ as the projection of y onto the simplex.

At step 5 we have $(y - \hat{t})_+ := \max\left\{0, y - \hat{t}\right\}$, where the maximum is understood in a component-wise fashion. The computational cost of this algorithm is $\mathcal{O}(K \log K)$ for the K-class problem. Note that both the convexity splitting and the MBO method require then $\mathcal{O}(nK \log K)$ for the projection step in every iteration. Other methods with an expected linear time complexity exist (see, e.g., [17]), but they are usually slower in practice.

4. Projection-free methods: a greedy Frank-Wolfe algorithm. In this section, we focus on some first-order methods that have recently garnered significant attention in both the machine learning and optimization communities. These methods make use of a Linear Minimization Oracle (LMO) to minimize a sequence of linear functions over the original feasible set. We will introduce the Frank-Wolfe (FW) approach (the interested reader is referred to, e.g., [10] for further details on this method) and a tailored variant that allow to handle the specific problem we are dealing with.

The classical FW method for minimization of a smooth objective E generates a sequence of feasible points $\{\mathbf{U}_k\}$ following the scheme of Algorithm 4.1. At the iteration k it moves toward a vertex, i.e., an extreme point, of the feasible set minimizing the scalar product with the current gradient $\nabla E(\mathbf{U}_k)$ [1]. It therefore makes use of a linear minimization oracle (LMO) for the feasible set Σ, that is,

$$\texttt{LMO}_\Sigma(\mathbf{G}) \in \operatorname{argmin}_{\mathbf{Z}\in\Sigma} \langle \mathbf{G},\ \mathbf{Z} \rangle\,, \tag{4.1}$$

defining the descent direction as

$$\mathbf{D}_k := \mathbf{S}_k - \mathbf{U}_k, \quad \mathbf{S}_k \in \texttt{LMO}_\Sigma(\nabla E(\mathbf{U}_k))\,. \tag{4.2}$$

In particular, the update at step 6 can be written as

$$\mathbf{U}_{k+1} = \mathbf{U}_k + \alpha_k(\mathbf{S}_k - \mathbf{U}_k) = \alpha_k \mathbf{S}_k + (1 - \alpha_k)\mathbf{U}_k. \tag{4.3}$$

Since $\alpha_k \in [0, 1]$, by induction $\mathbf{U}_{k+1}$ can be written as a convex combination of elements in the set $\mathcal{S}_{k+1} := \{\mathbf{U}_0\} \cup \{\mathbf{S}_i\}_{0\leq i\leq k}$. When $\mathcal{C} = \operatorname{conv}(\mathcal{A})$ for a set $\mathcal{A}$ of matrices with some common property, usually called "elementary atoms", if $\mathbf{U}_0 \in \mathcal{A}$ then $\mathbf{U}_k$ can be written as a convex combination of $k + 1$ elements in $\mathcal{A}$. In our case we have that $\mathcal{A}$ is given by all $n \times K$ matrices having as rows only elements of the canonical basis $e_j^\top$ with $j \in [1 : K]$.

[1] an $n \times K$ matrix whose element (i, j) is $\frac{\partial E(X)}{\partial X_{i,j}}$

Algorithm 4.1 `Frank-Wolfe method` [19]

1 Choose a point $\mathbf{U}_0 \in \Sigma$
2 For $k = 0, \dots$
3 Compute $\mathbf{S}_k \in \texttt{LMO}_\Sigma(\nabla E(\mathbf{U}_k))$
4 If $\mathbf{S}_k$ satisfies some specific condition, then STOP
5 Set $\mathbf{D}_k = \mathbf{S}_k - \mathbf{U}_k$
6 Set $\mathbf{U}_{k+1} = \mathbf{U}_k + \alpha_k \mathbf{D}_k$, with $\alpha_k \in (0, 1]$ a suitably chosen stepsize
7 End for

The LMO for the feasible set (2.4) is defined as follows:

$$\texttt{LMO}_\Sigma(\mathbf{C}) = (\mathbf{e}_{j_1}, \dots, \mathbf{e}_{j_n})^\top,$$

with $\mathbf{C} = (\mathbf{c}_1, \dots, \mathbf{c}_n)^\top$ and $j_i \in \operatorname{argmin}_{j \in [1:K]} (\mathbf{c}_i)_j$. It is easy to see that the LMO cost is $\mathcal{O}(n \cdot K)$.

We now define a greedy variant that better exploits the structure of the problem at hand.

Algorithm 4.2 `Greedy Frank-Wolfe (GFW) method`

1 Choose a point $\mathbf{U}_0 \in \Sigma$
2 For $k = 0, \dots$
3 Compute $\mathbf{S}_k \in \texttt{GLMO}_\Sigma(\nabla E(\mathbf{U}_k))$
4 If $\mathbf{S}_k$ satisfies some specific condition, then STOP
5 Set $\mathbf{D}_k = \mathbf{S}_k - \mathbf{U}_k$
6 Set $\mathbf{U}_{k+1} = \mathbf{U}_k + \alpha_k \mathbf{D}_k$, with $\alpha_k \in (0, 1]$ a suitably chosen stepsize
7 End for

As we can easily see, the structure of the algorithm stays the same apart from the LMO, now replaced by the so-called *Greedy Linear Minimization Oracle* (GLMO), which significantly reduces the per-iteration computational cost, while preserving the convergence properties of the original method. We can define the GLMO as follows:

$$\texttt{GLMO}_\Sigma(\mathbf{G}) = (\mathbf{w}_1, \dots, \mathbf{w}_n)^\top,$$

with $\mathbf{G} = (\mathbf{g}_1, \dots, \mathbf{g}_n)^\top = \nabla E(\mathbf{U})$, $\mathbf{w}_i = \mathbf{e}_{j_i}$, if $i \in \mathcal{I} = \{i \in [1 : n] : \ \mathbf{u}_i \notin \{0, 1\}^K\}$, $j_i \in \operatorname{argmin}_{j \in supp(\mathbf{u}_i)} (\mathbf{g}_i)_j$, and $\mathbf{w}_i = \mathbf{u}_i$ otherwise. Here, $supp(\mathbf{u}_i)$ denotes the indices of the nonzero elements of the vector $\mathbf{u}_i$. More specifically, in the GLMO we solve for each i such that $\mathbf{u}_i \notin \{0, 1\}^K$ the following problem:

$$\min_{\mathbf{y} \in \bar{\Delta}^i_{K-1}} \ \mathbf{g}_i^\top \mathbf{y}, \tag{4.4}$$

with $\bar{\Delta}^i_{K-1} := \{\mathbf{y} \in \mathbb{R}^K_+ : \mathbf{1}^\top \mathbf{y} = 1, \mathbf{y}_j = 0 \ \forall \ j \notin supp(\mathbf{u}_i)\}$, while we do nothing and choose $\mathbf{u}_i$ in case $\mathbf{u}_i \in \{0, 1\}^K$. It is hence easy to see that the GLMO cost is $\mathcal{O}(|\mathcal{I}| \cdot m)$, with $m = \max_{i \in \mathcal{I}} |supp(\mathbf{u}_i)|$.

The overall algorithm is given in Algorithm 4.2 and we now report a detailed analysis of its convergence properties.

Theorem 4.1 (Convergence Rate GFW - General Case). *Consider problem (2.8) with $\varepsilon < \tilde{\varepsilon} = 1/\left[K(\rho^{max} + d_\omega^{max})\right]$, where $\rho^{max} = \max_i \sum_{j=1}^n |(\mathbf{L}_s)_{ij}|$. Let $\{\mathbf{U}_k\}$ be a sequence generated by the GFW algorithm, where the step size α_k satisfies*

$$\alpha_k \geq \bar{\alpha}_k = \min\left(1, \frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right), \tag{4.5}$$

as well as

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha_k \mathbf{D}_k) \geq \rho \bar{\alpha}_k g_k \quad \text{for some fixed } \rho > 0\,, \tag{4.6}$$

with $g_k = -\langle \nabla E(\mathbf{U}_k),\ \mathbf{D}_k \rangle$. *Then for every* $T \in \mathbb{N}$

$$g_T^* \leq \max\left(\sqrt{\frac{L2n(E(\mathbf{U}_0) - E^*)}{\rho T}}, \frac{2(E(\mathbf{U}_0) - E^*)}{T} \right), \tag{4.7}$$

where $g_k^* = \min\limits_{0 \leq i \leq k-1} g_i$ *and* $E^* = \min_{\mathbf{U} \in \Sigma} E(\mathbf{U})$.

It is crucial to observe that the conditions specified in equations (4.5)-(4.6) can be satisfied by employing appropriate line search techniques. For further elucidation, one may refer to the works of Bomze et al. [9], [10], and Rinaldi et al. [29]. Notably, Lemma 4.2 presented herein demonstrates that these conditions are particularly met when using the Armijo line search method, which determines

$$\alpha_k = \delta^j, \tag{4.8}$$

where j is the smallest non-negative integer such that

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha_k \mathbf{D}_k) \geq -\gamma \alpha_k \langle \nabla E(\mathbf{U}_k),\ \mathbf{D}_k \rangle, \tag{4.9}$$

with $\gamma \in (0, 1/2)$ and $\delta \in (0, 1)$ being two fixed parameters.

Lemma 4.2. *If* α_k *is determined by the Armijo line search described above then*

$$\alpha_k \geq \min\left(1, 2\delta(1-\gamma)\frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right) \geq \min\{1, 2\delta(1-\gamma)\}\bar{\alpha}_k\,, \tag{4.10}$$

with $\bar{\alpha}_k = \min\left(1, \frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right)$ *as in* (4.5), *and* (4.6) *holds with* $\rho = \gamma \min\{1, 2\delta(1-\gamma)\} < 1$.

In Lemma 4.2 we prove that

$$\alpha_k \geq \min\left(1, c\frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right) \text{ for some } c > 0\,,$$

for the Armijo line search. When $c \geq 1$ then (4.5) is of course a lower bound for the step size α_k, and when $c < 1$ we can still recover (4.5) by considering $\tilde{L} = \frac{L}{c}$ instead of L as Lipschitz constant.

Now we show that when the parameter ε is sufficiently small, the Greedy Frank-Wolfe algorithm gives a stationary point in one step, thus being equivalent to the simple algorithmic scheme reported below in Algorithm 4.3.

Algorithm 4.3 `One-Shot Frank-Wolfe (OSFW) method`

1 Choose a point $\mathbf{U}_0 \in \Sigma$
2 Compute $\mathbf{S}_0 \in \texttt{GLMO}_\Sigma(\nabla E(\mathbf{U}_0))$
3 Set $\mathbf{U}_1 = \mathbf{S}_0$

Corollary 4.3 (One-Shot Frank-Wolfe Convergence). *Consider problem* (2.8) *with* $\varepsilon < \min\{\bar{\varepsilon}, \tilde{\varepsilon}\}$, *where* $\bar{\varepsilon}$ *and* $\tilde{\varepsilon}$ *are chosen as in Theorem 2.3 and Theorem 4.1, respectively. Then GFW gives a stationary point in one iteration, and it is equivalent to OSFW.*

We can now provide an in-depth analysis of the computational complexity associated with the method introduced in the preceding section. In Theorem 4.1, we established that the duality gap g_T^* exhibits a sublinear convergence rate, specifically $g_T^* \leq \max(c_1 T^{-\frac{1}{2}}, c_2 T^{-1})$, where c_1 and c_2 are suitable constants. From this, complexity outcomes can be directly derived using conventional principles from information-based complexity theory, as detailed in [28]. Specifically, in the context of our study, we demonstrate a worst-case complexity of $\mathcal{O}(\epsilon^{-2})$ concerning the iteration count required to reduce g_T^* below the threshold ϵ. Furthermore, an upper limit on the arithmetic operations performed in each iteration of the GFW algorithm can be easily determined: each iteration requires one gradient evaluation, having a cost $\mathcal{O}(n)$ (this usually depends on the application), and the `GLMO` cost (with the line search having a computational cost of $\mathcal{O}(1)$, provided the Lipschitz constant L is known). In summary, each iteration of the GFW algorithm presents a computational burden of $\mathcal{O}(n)$, implying that $\mathcal{O}(n\epsilon^{-2})$ arithmetic operations are necessary to achieve g_T^* below ϵ.

We finally note that the proposed `GFW` algorithm naturally supports efficient inference on previously unseen data points. When new samples are added to the dataset (i.e., nodes are added to the graph), the way the `GLMO` is defined indeed ensures that, by suitably warm-starting the `GFW` with a point embedding the binary solution related to the original data, only the rows corresponding to the new samples are updated in the algorithm iterates, while the rows related to the old binary solution remain unchanged. Moreover, computing these updates requires just the gradient rows associated with the new nodes, which in turns depend solely on their connections with the other graph nodes. As a result, the new labels can be assigned by simply running a few `GFW` iterations. Since each iteration involves sparse matrix–vector products of size determined by the new nodes, this inference step might be even cheaper than the original training phase, thus making the procedure highly scalable.

5. Numerical experiments.

In this section, we illustrate the performance of the new Greedy Frank-Wolfe (GFW) method when solving multiclass segmentation formulated using the new energy function E in (2.9). In particular, we compare GFW against the classic PDE graph-based Convexity Splitting (CS) method and the Merriman-Bence-Osher (MBO) method on segmentation problems both for networks and images (color segmentation). After introducing some implementation details and our test set, we first perform a preliminary parameter analysis for the considered models (cf. the objectives (2.5) and (2.7)), and then compare the three algorithms on different data sets.

5.1. Implementation issues and test problems.

The CS, MBO and GFW methods have been implemented in MATLAB[2] and are available at https://github.com/mstoll1602/GreedyFrankWolfe. Regarding the details of the implementation of CS and MBO we refer the reader to [7, 26]. The low-rank eigenvalue approximations for both methods are computed with built-in MATLAB functions.

The Greedy Frank Wolfe method has been implemented following Algorithm 4.2. The method is stopped when either 30 iterations are performed or when $g_k = -\langle \nabla E(\mathbf{U}_k),\ \mathbf{D}_k \rangle \leq 10^{-6}$. Moreover, we implemented a classical Armijo line search strategy to compute the step size α_k in Step 6 of Algorithm 4.2 with $\delta = 0.5$ and $\gamma = 10^{-6}$ in (4.8) and (4.9), respectively.

The initial guess $\mathbf{U}_0$ for the algorithms CS and MBO is produced in accordance with the procedures outlined in [7, 26]. More specifically, we begin by drawing random numbers from a standard uniform distribution defined over the interval $(0, 1)$. Subsequently, these randomly generated values are projected row-by-row onto the unit simplex Δ_{K-1}^i. Furthermore, at locations corresponding to fidelity points, these values are adjusted to correspond to pure phases. On the other hand, the initialization $\mathbf{U}_0 = \hat{\mathbf{U}}$ is used for the FW algorithms.

[2]MATLAB R2019b on a Intel Core i7-9700K CPU @ 3.60GHz x 8, 16 GB RAM, 64-bit.

We have tested our algorithms on both networks for data segmentation and images for image labeling, as detailed in Tables 1 and 2, respectively. Tables report the number n of nodes (i.e. pixels for images) in the data set, the number of classes K and the sparsity $\mathrm{sp}(\mathbf{L}_s)$ of the Laplacian matrix $\mathbf{L}_s$ (only for the networks in Table 1).

In detail, we considered the widely used Lancichinetti-Fortunato-Radicchi (LFR) benchmark data set [3] for different numbers of nodes and mixing coefficients (0.1 and 0.2) and the 5 real networks: Twitch, LastFM, Facebook and Amazon (computers and photos). In all the networks, for each class we correctly labeled 1/3 of randomly sampled nodes, 1/10 for LastFM (the unknown nodes are initialized as $1/K$ for each class). For all data sets, the matrix $\mathbf{L}_s$ is retrieved from the available graph adjacency matrix and results in a sparse matrix. Regarding the imaging problems in Table 2[4], the matrix-vector products with the graph Laplacian is performed using an NFFT approach [1]. This function approximates the matrix vector product with the fully connected graph Laplacian while avoiding the complexity of having to deal with a dense matrix and the corresponding numerical cost. The images are resized versions of the original ones. For the *beach*, initial and labeled images are shown in the forthcoming Figure 2, for the others in the Supplementary Materials (SM), Figures SM2, SM4 and SM6.

In the next sections, solvers will be compared mainly using two performance measures: the accuracy ($\mathcal{A}$), computed as the percentage of correctly classified nodes (correctly classified pixels in the case of images), and the elapsed time ($\mathcal{T}$) in seconds. In order to compute $\mathcal{A}$ for image labeling, we generated their ground truths as explained in SM Section SM1.1.

Data set	n	K	$\mathrm{sp}(\mathbf{L}_s)$	Data set	n	K	$\mathrm{sp}(\mathbf{L}_s)$
LFR (1000, 5, 50, 0.1/0.2) [22]	1000	11	$5\,10^{-3}$	Twitch [23, 33]	168114	21	$4\,10^{-4}$
LFR (5000, 5, 50, 0.1/0.2) [22]	5000	62	10^{-3}	LastFM [23, 32]	7624	9	10^{-3}
LFR (10000, 5, 50, 0.1/0.2) [22]	10000	120	$5\,10^{-4}$	Facebook [23, 31]	22470	4	$7\,10^{-4}$
LFR (50000, 5, 50, 0.1/0.2) [22]	50000	426	10^{-4}	Amazon (pc) [34]	13752	10	$2\,10^{-3}$
LFR (100000, 5, 50, 0.1/0.2) [22]	100000	585	$5\,10^{-5}$	Amazon (pic) [34]	7650	8	$4\,10^{-3}$

Table 1: Data set details for data segmentation.

Image	n	K	Image	n	K
Beach [4]	38220	4	*4 geometric figures* [35]	25920	5
3 geometric figures [35]	35828	4	*Sheets of paper* [3, 35]	16500	8

Table 2: Data set details for image labeling.

5.2. Preliminary parameter tuning. In this section, we investigate the role of the weights ω, μ and ε within the objective functions (2.5) and (2.7) and in (3.4), which correspond to different graph-based semi-supervised learning models for multiclass segmentation.

Earlier studies [4, 7, 20] provide limited discussion regarding how the CS and MBO methods rely on the parameters ω_0 and μ. Therefore, we considered the values $\omega_0 = 10^i$, $i \in \{1, 2, 3, 4\}$ and $\mu \in \{5\,10^2, 10^2, 5\,10, 10, 5\,10^{-1}, 10^{-1}, 5\,10^{-2}, 10^{-2}, 5\,10^{-3}\}$ (for CS only) and noticed a variation from 10% to 20% in terms of accuracy in the computed solution.

[3] LFR networks are available at https://github.com/GiulioRossetti/cdlib_datasets. The graph parameters reported in Table 1 correspond to LFR(number of nodes, average degree, min community, mixing coefficient).

[4] The *beach* image can be downloaded from https://github.com/mstoll1602/GreedyFrankWolfe, while the other three images are available at https://github.com/Visillect/colorsegdataset

Therefore, in the experiments, we used the parameters that gave the best accuracy on an average of 10 runs. These best parameters are reported in Table 3. The table also reports the number K_{eig} of the eigenvalues/eigenvectors chosen for each test problem. We note that the accuracy obtained using CS and MBO strongly depends on this choice and that we selected the smallest value that gives the highest accuracy in our experience. Clearly, a large value of K_{eig} (e.g., for Facebook and Amazon networks) may also impact on the overall elapsed time.

	LFR ($n \leq 10000$)	LFR ($n \geq 50000$)	Twitch	LastFM	Facebook	Amazon
ω_0	100	100	100	1000	100	100
μ	100	100	10	50	10	500
K_{eig}	100	200	6	15	100	150

	Beach	*3 geometric figures*	*4 geometric figures*	*Sheets of paper*
ω_0	10^5	10^5	10^5	10^5
μ	10^{-1}	10^{-1}	10^{-1}	10^{-1}
K_{eig}	7 (CS) and 3 (MBO)	6	6	20

Table 3: Best parameter configuration for CS and MBO.

We also considered values $\omega_0 = 10^i$, $i \in \{1, 2, 3, 4\}$ and $\varepsilon \in \{5\,10^2, 10^2, 5\,10, 10, 5\,10^{-1}, 10^{-1}, 5\,10^{-2}, 10^{-2}, 5\,10^{-3}\}$ in the objective (2.7) and analyzed the performance of GFW in terms of both accuracy and elapsed time. Figure 1 shows these values varying ε and ω_0 for the Facebook network as a representative example for all the datasets. The figure shows a double behaviour of GFW: for some choices of the parameters (those corresponding to blue dots) GFW terminates in one step, being in fact OSFW and therefore the fastest version; for some other values of the parameters (those corresponding to red dots) GFW reaches the maximum accuracy. Therefore, in the next section we use the following parameter values

$$\omega_0 = 1000 \quad \text{and} \quad \varepsilon = 50,$$

for experiments on data from networks. For image labeling we used

$$\omega_0 = 10^5 \quad \text{and} \quad \varepsilon = 10^{-1}$$

that guarantee the one-shot behavior for GFW, see Algorithm 4.3 and Corollary 4.3.

5.3. Numerical results. Table 4 collects averaged results obtained using GFW and the competing solvers CS and MBO on network data (Table 1), over 10 runs. The symbol ∞ means that the elapsed time is above 1 hour.

Focusing on LFR problems, we observe that GFW always gives the best accuracy in short time and the gain increases as the number of nodes increases. Moreover, while the elapsed time for the FW solver stays bounded as n and K increase, both CS and MBO take longer time. In details, these runs highlight on the main computational bottleneck of CS and MBO, that is the cost of the projections over simplices that is quite high when the number of vertices is large. Moreover, the large elapsed time of CS is due to the evaluation of the elements $T_{i,j}$ in (3) that is rather time consuming as K increases.

Focusing on the real networks, GFW always roughly computes the solution with 10% of better accuracy wrt CS and MBO in much less time, except for the Twitch network for which MBO is the fastest. In fact, for Twitch MBO takes a short time as it employs only few iterations (and then only few projections). Finally, we remark that for Facebook and Amazon the elapsed time of CS and MBO is affected by the nonnegligible cost of the initial eigendecomposition of, although sparse, large matrices (see Table 3).

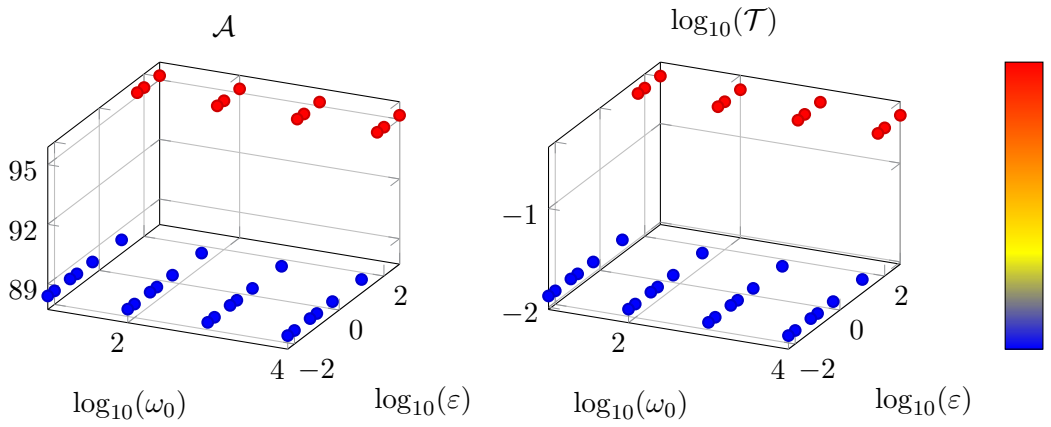


Figure 1: Facebook network: values of accuracy ($\mathcal{A}$) and base-10 logarithm of elapsed time ($\log_{10}(\mathcal{T})$) varying the parameters ω_0 and ε in the objective function (2.9).

	GFW		CS		MBO	
Data set	$\mathcal{T}$	$\mathcal{A}$	$\mathcal{T}$	$\mathcal{A}$	$\mathcal{T}$	$\mathcal{A}$
LFR 1000/0.1	0.02	99.0%	0.32	98.8%	0.39	98.4%
LFR 5000/0.1	0.27	99.2%	85.45	98.9%	5.45	98.9%
LFR 10000/0.1	0.70	99.4%	587.78	91.3%	17.31	96.8%
LFR 50000/0.1	17.23	98.1%	∞	-	2575.31	84.8%
LFR 100000/0.1	62.02	83.7%	∞	-	∞	-
LFR 1000/0.2	0.03	92.5%	0.39	91.1%	0.93	91.2%
LFR 5000/0.2	0.33	93.8%	65.5	92.4%	18.73	92.5%
LFR 10000/0.2	0.68	94.9%	807.03	81.5%	40.99	86.6%
LFR 50000/0.2	18.25	90.9%	∞	-	2749.85	76.5%
LFR 100000/0.2	64.05	77.5%	∞	-	∞	-
Twitch	15.12	94.9%	41.68	84.6%	8.92	84.7%
LastFM	0.12	85.6%	1.42	76.4%	1.71	75.9%
Facebook	0.30	95.2%	19.84	74.7%	20.99	75.7%
Amazon (computer)	0.31	91.8%	40.3	77.7%	51.7	81.1%
Amazon (photo)	0.12	94.0%	25.6	85.9%	30.7	89.2%

Table 4: Network data sets of Table 1: results in terms of accuracy $\mathcal{A}$ and elapsed time $\mathcal{T}$ in seconds for GFW, CS and MBO.

Table 5 shows the elapsed time employed by the GFW (with one-shot behavior), CS and MBO and the accuracy obtained in the segmentation of the images in Table 2. Evidently, the FW method is roughly faster by a factor of 2 (or 3) with respect to CS and MBO. In these runs, CS and MBO greatly suffer from the time-consuming initial eigendecomposition, while the computational effort GFW is mainly due to the matrix-matrix product with $\mathbf{L}_s$. As for the accuracy, GFW returns the highest values in all experiments. In the case of the *beach* image, Figure 2, the four classes are reasonably identified by GFW, while CS predicts as "sea" some "beach" pixels and MBO completely misses the "sea", which is classified as "sky", see the confusion matrices in Figure 3.

	GFW		CS		MBO	
Image	$\mathcal{T}$	$\mathcal{A}$	$\mathcal{T}$	$\mathcal{A}$	$\mathcal{T}$	$\mathcal{A}$
Beach	37.10	95.7%	87.52	91.2%	60.96	84.9%
3 geometric figures	57.95	98.6%	133.41	97.7%	114.22	87.2%
4 geometric figures	35.70	98.4%	98.29	93.4%	78.92	83.3%
Sheets of paper	10.47	97.9%	45.67	83.7%	26.72	93.2%

Table 5: Image labeling: elapsed cpu time $\mathcal{T}$ and accuracy $\mathcal{A}$.

Figures and confusion matrices related to experiments on *3* and *4 geometric figures* and *Sheets of paper* are contained in Section SM1.2 of Supplementary Materials. For *3 geometric figures* image, the accuracy reported in Table 5 of MBO is misleading. Indeed, this image presents a strongly imbalanced class (larger of a factor 6, Figure SM2), consequently, even if MBO completely fails to identify another class, it returns a good value of accuracy, Figure SM3. Also for *4 geometric figures* image, the classes are imbalanced, Figure SM4, and, in this case CS is unsuccessful in the classification of a class, while MBO completely miscarry two minority classes, Figure SM5. In the case of *sheets of paper* image, CS and MBO mixed some classes more, Figures SM6 and SM7. Therefore, even if the accuracy values seem high for CS and MBO, the quality of the segmentation is poor. Conversely, our new approach provides higher accuracy values and better segmentations.

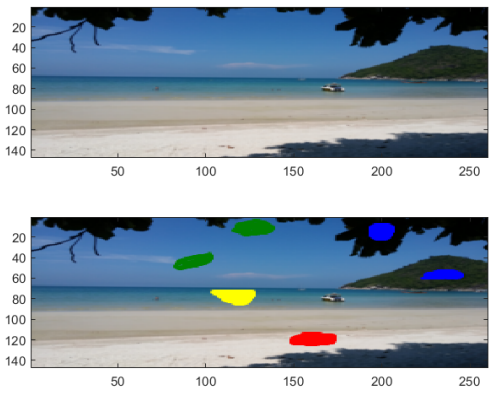
6. Conclusions and Future Work. We considered the graph-based segmentation framework and addressed key computational bottlenecks shared by classic methods in this context: the eigendecomposition of the graph Laplacian and the projection on the unit simplex. We therefore proposed a new model problem that still uses the Ginzburg-Landau energy functional but which ensures valid partitions (i.e., binary solutions) by employing a suitable penalty term. For the minimization of such a model, we defined a tailored Frank-Wolfe algorithm that achieves both theoretical and practical improvements compared to traditional PDE-inspired methods (CS and MBO). We illustrated the effectiveness of our proposal on several test problems, including real networks and images, showing both the validity of our model and the gain in terms of computational time attained with our algorithm.

The approach that has been proposed paves the way for novel and captivating avenues of research. First, an in-depth theoretical and computational analysis related to the penalty parameter for different graph topologies might be carried out, thus strengthening the model's guarantees and giving the possibility to broaden the applicability of the method to dynamic or evolving graphs. Second, the use of acceleration (e.g., momentum-based), decomposition (e.g., block-coordinate Frank-Wolfe-like methods) and stochastic-approximation strategies (e.g., stochastic Frank-Wolfe-like methods) might help to further reduce the computational burden when dealing with massive datasets. Third, extensions that allow one to handle higher-order graph data structures (e.g., multilayer, hyperedges, and so on) might represent another interesting research topic. Finally, exploring distributed implementations could make real-time segmentation viable for large-scale imaging and network analysis.

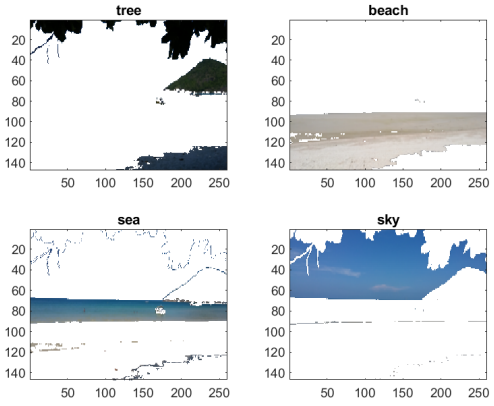
Appendix A. Proof of Proposition 2.1.

Proof. The diameter is defined as the maximum distance between two matrices in the feasible set, that is

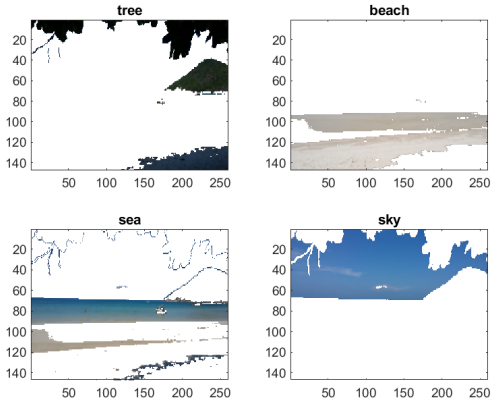
$$diam(\Sigma) = \max_{\bar{\mathbf{U}},\mathbf{U}\in\Sigma} \|\bar{\mathbf{U}} - \mathbf{U}\|_F.$$



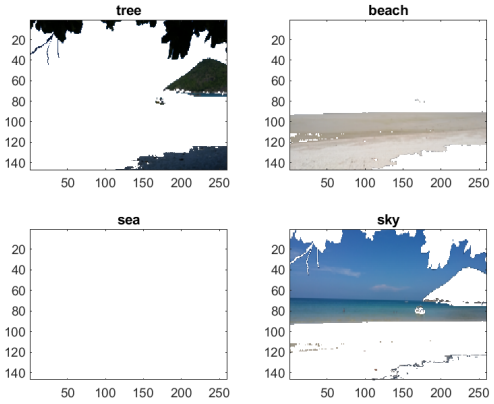
(a) Original (top) and labeled (bottom) image



(b) Segmentation with GFW (OSFW)



(c) Segmentation with CS



(d) Segmentation with MBO

Figure 2: *Beach* image: labeling results with GFW (b), CS (c) and MBO (d).

Since each row is constrained to the unit simplex, we have

$$\max_{\bar{\mathbf{U}},\mathbf{U}\in\Sigma} \|\bar{\mathbf{U}}-\mathbf{U}\|_F^2 = \sum_{i=1}^{n} \max_{\bar{\mathbf{u}}_i,\mathbf{u}_i\in\Sigma^K} \|\bar{\mathbf{u}}_i-\mathbf{u}_i\|_2^2 = 2n,$$

thus proving the desired result. ∎

Appendix B. Proof of Proposition 2.2.

Proof. Let us consider the gradient of the function given in (2.10). We want to prove that

$$\|\nabla E(\bar{\mathbf{U}}) - \nabla E(\mathbf{U})\|_F \le L\|\bar{\mathbf{U}}-\mathbf{U}\|_F.$$

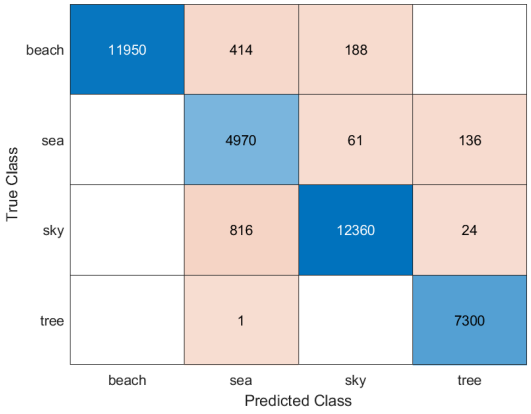
We now separately bound the 3 terms of the gradient in (2.10):

- term $\mathbf{L}_s\mathbf{U}$. Let $\mathbf{V}=\bar{\mathbf{U}}-\mathbf{U}$, so we can write:

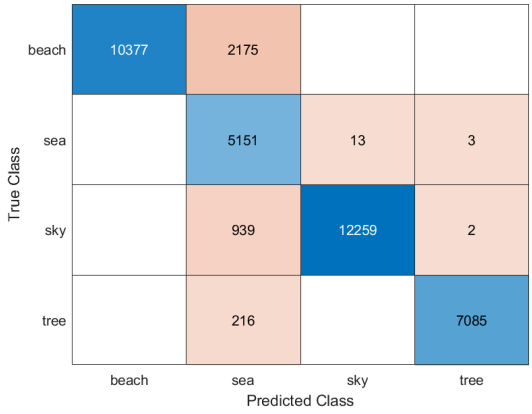
$$\|\mathbf{L}_s\mathbf{V}\|_F^2 = \sum_{j=1}^{p}\|\mathbf{L}_s\mathbf{v}_j\|_2^2 \le \|\mathbf{L}_s\|_2^2\sum_{j=1}^{p}\|\mathbf{v}_j\|_2^2 = \|\mathbf{L}_s\|_2^2\|\mathbf{V}\|_F^2.$$

Taking square roots:

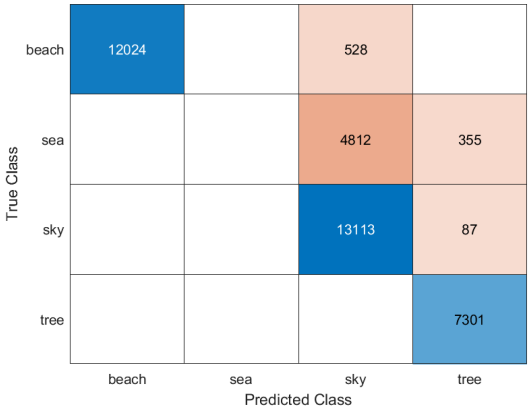
$$\|\mathbf{L}_s(\bar{\mathbf{U}}-\mathbf{U})\|_F = \|\mathbf{L}_s\mathbf{V}\|_F \le \|\mathbf{L}_s\|_2\|\mathbf{V}\|_F = \|\mathbf{L}_s\|_2\|\bar{\mathbf{U}}-\mathbf{U}\|_F.$$



(a) Confusion matrix GFW



(b) Confusion matrix CS



(c) Confusion matrix MBO

Figure 3: *Beach* image: confusion matrix results with GFW (a), CS (b) and MBO (c).

We thus have for this term

$$\|\mathbf{L}_s\bar{\mathbf{U}} - \mathbf{L}_s\mathbf{U}\|_F = \|\mathbf{L}_s(\bar{\mathbf{U}} - \mathbf{U})\|_F \leq \|\mathbf{L}_s\|_2\|\bar{\mathbf{U}} - \mathbf{U}\|_F \leq \lambda_{max}(\mathbf{L}_s)\|\bar{\mathbf{U}} - \mathbf{U}\|_F.$$

- term $\frac{1}{\epsilon}(\mathbf{1} - 2\mathbf{U})$. It is easy to see that

$$\|\frac{2}{\epsilon}(\bar{\mathbf{U}} - \mathbf{U})\|_F \leq \frac{2}{\varepsilon}\|\bar{\mathbf{U}} - \mathbf{U}\|_F.$$

- term $-\mathbf{D}_\omega(\hat{\mathbf{U}} - \mathbf{U})$. Since $\mathbf{D}_\omega$ is diagonal, we have

$$\|\mathbf{D}_\omega(\mathbf{U} - \bar{\mathbf{U}})\|_F \leq d_\omega^{max}\|\bar{\mathbf{U}} - \mathbf{U}\|_F.$$

So the result holds. ∎

Appendix C. Proof of Theorem 2.3.

Proof. If we choose a value $\varepsilon \in (0, \bar{\varepsilon}]$, the Hessian $\nabla^2 E(\mathbf{U}) = \mathbf{I}_K \otimes (\mathbf{L}_s + \mathbf{D}_\omega - \frac{2}{\epsilon}\mathbf{I}_n)$ is negative definite. As a result, the function $E(\mathbf{U})$ is strictly concave and every local (and global) minimum $\mathbf{U}^*$ is strict and is a vertex of the feasible set Σ. In order to see this, by contradiction assume that $\mathbf{U}^*$ is not an extreme point of Σ. Then there exists a neighborhood

$B(\mathbf{U}^*, \rho)$, with $\rho > 0$, of suitable dimension fully contained in the feasible set. We can hence find a direction $\tilde{\mathbf{D}} = \tilde{\mathbf{U}} - \mathbf{U}^*$, with $\tilde{\mathbf{U}} \in B(\mathbf{U}^*, \rho)$, such that both $\tilde{\mathbf{D}}$ and $-\tilde{\mathbf{D}}$ are feasible directions. By first order optimality conditions we can write

$$\langle \nabla E(\mathbf{U}^*), \mathbf{D} \rangle \geq 0,$$

for any feasible direction $\mathbf{D}$. Then $\langle \nabla E(\mathbf{U}^*), \tilde{\mathbf{D}} \rangle = 0$. Thanks to the way we choose ε, we also have

$$\langle \tilde{\mathbf{D}}, \nabla^2 E(\mathbf{U}^*) \tilde{\mathbf{D}} \rangle < 0.$$

Thus we can write

$$E(\tilde{\mathbf{U}}) = E(\mathbf{U}^* + \tilde{\mathbf{D}}) = E(\mathbf{U}^*) + \langle \nabla E(\mathbf{U}^*), \tilde{\mathbf{D}} \rangle + \langle \tilde{\mathbf{D}}, \nabla^2 E(\mathbf{U}^*) \tilde{\mathbf{D}} \rangle < E(\mathbf{U}^*),$$

which contradicts optimality of $\mathbf{U}^*$ (see, e.g., [30] for further details). By using similar arguments, we can easily see that $\mathbf{U}^*$ is strict. Since Σ is defined as in (2.4), then $\mathbf{U}^* \in \{0,1\}^{n \times K}$ and the result (i) is proved.

In order to prove point (ii), we first need to show that any global minimizer of (2.8) is also a global minimizer of (2.12). Since for any global minimizer $\mathbf{U}^*$ of (2.8), we have that $\mathbf{U}^* \in \Sigma \cap \{0,1\}^{n \times K} =: \bar{\Sigma}$, and $E(\mathbf{U}^*) = \bar{E}(\mathbf{U}^*) \leq \bar{E}(\mathbf{U})$ for all $\mathbf{U} \in \bar{\Sigma}$, then the result holds. Now we need to prove that any global minimizer of (2.12) is also a global minimizer of(2.8). By contradiction, assume there exists $\varepsilon \in (0, \bar{\varepsilon}]$ such that

$$E(\hat{\mathbf{U}}) < E(\mathbf{U}^*),$$

with $\hat{\mathbf{U}} \in \Sigma$ global minimizer of (2.8) and $\mathbf{U}^* \in \bar{\Sigma}$ global minimizer of (2.12). Recalling that for all $\varepsilon \in (0, \bar{\varepsilon}]$ we proved that $\hat{\mathbf{U}} \in \bar{\Sigma}$ we then have

$$\bar{E}(\hat{\mathbf{U}}) < \bar{E}(\mathbf{U}^*),$$

which contradicts global optimality of $\mathbf{U}^*$. Thus point (ii) is proved. ■

Appendix D. Proof of Theorem 4.1.

Proof. We first show that $\texttt{GLMO}_\Sigma(\mathbf{G}) = \texttt{LMO}_\Sigma(\mathbf{G})$ where $\mathbf{G} = \nabla E(\mathbf{U})$ and $\mathbf{U} \in \Sigma$, when $\varepsilon \leq \tilde{\varepsilon}$. If $\mathbf{U} \in \Sigma$ is such that $\mathbf{u}_i \in \{0,1\}^K$ for some i, then it is easy to see that $(\mathbf{g}_i)_j \simeq \varepsilon^{-1}$ if $(\mathbf{u}_i)_j = 0$ and $(\mathbf{g}_i)_j \simeq -\varepsilon^{-1}$ otherwise. Since in the LMO we solve for each i the following problem:

$$\min_{\mathbf{y} \in \Delta^i_{K-1}} \mathbf{g}_i^\top \mathbf{y}, \tag{D.1}$$

then we have $\mathbf{y} = \mathbf{u}_i$. If $\mathbf{U} \in \Sigma$ is such that $\mathbf{u}_i \notin \{0,1\}^K$, but $supp(\mathbf{u}_i) < K$, we have that $(\mathbf{g}_i)_j \simeq \varepsilon^{-1}$ for all j such that $(\mathbf{u}_i)_j = 0$. We now show that thanks to the way we set ε there always exists at least one component of the gradient smaller than ε^{-1} and thus we have $\mathbf{y}_j = 0$ in the solution of problem (D.1) when $(\mathbf{u}_i)_j = 0$. Notice that the gradient of the function without penalty term is such that $\|\nabla \bar{E}(\mathbf{U})\|_\infty \leq \rho^{max} + d_\omega^{max}$, furthermore the j-th gradient component related to the penalty is $\varepsilon^{-1}[1 - 2(\mathbf{u}_i)_j]$, so the worst case scenario for us is when all components have the same weight, with an upper bound given by $\varepsilon^{-1}[1 - 2K^{-1}]$. Hence, taking into account the way we choose ε in problem (2.8), we have

$$\varepsilon^{-1} > \varepsilon^{-1}[1 - 2K^{-1}] + 2(\rho^{max} + d_\omega^{max})$$

and a component $(\mathbf{u}_i)_j = 0$ will stay zero when solving problem (D.1) and the result is proved.

We now analyze the convergence rate of the method by considering two different cases.
Case 1. $\bar{\alpha}_k < 1$. It is easy to see that $\bar{\alpha}_k = \frac{g_k}{L\|\mathbf{D}_k\|_F^2}$ in this case and inequality (4.6) becomes

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha_k \mathbf{D}_k) \geq \frac{\rho}{L\|\mathbf{D}_k\|_F^2} g_k^2 \geq \frac{\rho g_k^2}{L2n}\,, \tag{D.2}$$

where the last inequality is obtained by considering Proposition 2.1.
Case 2. $\bar{\alpha}_k = 1$. We start by considering the standard descent lemma [8, Proposition 6.1.2] applied to E with center $\mathbf{U}_k$ and $\alpha = 1$

$$E(\mathbf{U}_{k+1}) = E(\mathbf{U}_k + \mathbf{D}_k) \leq E(\mathbf{U}_k) + \langle \nabla E(\mathbf{U}_k), \mathbf{D}_k \rangle + \frac{L}{2}\|\mathbf{D}_k\|_F^2\,.$$

The condition that characterizes Case 2 gives $\min\left(\frac{-\langle \nabla E(\mathbf{U}_k), \mathbf{D}_k\rangle}{\|\mathbf{D}_k\|_F^2 L}, 1\right) = \alpha_k = 1$, thus we can write

$$\frac{-\langle \nabla E(\mathbf{U}_k), \mathbf{D}_k\rangle}{\|\mathbf{D}_k\|_F^2 L} \geq 1\,, \text{ and } \quad -L\|\mathbf{D}_k\|_F^2 \geq \langle \nabla E(\mathbf{U}_k), \mathbf{D}_k\rangle\,,$$

hence we obtain

$$E(\mathbf{U}_k) - E(\mathbf{U}_{k+1}) \geq -\langle \nabla E(\mathbf{U}_k), \mathbf{D}_k\rangle - \frac{L}{2}\|\mathbf{D}_k\|_F^2 \geq -\frac{\langle \nabla E(\mathbf{U}_k), \mathbf{D}_k\rangle}{2} \geq \frac{g_k}{2}\,. \tag{D.3}$$

Now, taking into account the two cases analyzed above, we can partition the iterations $\{0, 1, \ldots, T-1\}$ into two subsets N_1 and N_2 defined as follows:

$$N_1 = \{k < T \colon \bar{\alpha}_k < 1\}, \quad N_2 = \{k < T \colon \bar{\alpha}_k = 1\}\,.$$

Using (D.2) and (D.3), we have:

$$\begin{aligned}
E(\mathbf{U}_0) - E^* &\geq \sum_{k=0}^{T-1} (E(\mathbf{U}_k) - E(\mathbf{U}_{k+1})) \\
&= \sum_{k\in N_1} (E(\mathbf{U}_k) - E(\mathbf{U}_{k+1})) + \sum_{k\in N_2} (E(\mathbf{U}_k) - E(\mathbf{U}_{k+1})) \\
&\geq \sum_{k\in N_1} \frac{\rho g_k^2}{L2n} + \sum_{k\in N_2} \frac{g_k}{2} \\
&\geq |N_1| \min_{k\in N_1} \frac{\rho g_k^2}{L2n} + |N_2| \min_{k\in N_2} \frac{g_k}{2} \\
&\geq (|N_1| + |N_2|) \min\left(\frac{\rho (g_T^*)^2}{L2n}, \frac{g_T^*}{2}\right) \\
&= T \min\left(\frac{\rho (g_T^*)^2}{L2n}, \frac{g_T^*}{2}\right)\,,
\end{aligned} \tag{D.4}$$

where the last inequality is obtained using the definition of g_T^*.
Now, if $T \min\left(\frac{\rho (g_T^*)^2}{L2n}, \frac{g_T^*}{2}\right) = \frac{T g_T^*}{2}$ we then have

$$g_T^* \leq \frac{2(E(\mathbf{U}_0) - E^*)}{T}\,, \tag{D.5}$$

and otherwise

$$g_T^* \leq \sqrt{\frac{L2n(E(\mathbf{U}_0) - E^*)}{\rho T}}\,. \tag{D.6}$$

The result is thus proved by just considering the max between the right-hand sides in equation (D.5) and (D.6). ∎

Appendix E. Proof of Lemma 4.2.

Proof. Using the standard descent lemma [8, Proposition 6.1.2] we obtain

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha\mathbf{D}_k) \geq \alpha g_k - \alpha^2 \frac{L\|\mathbf{D}_k\|_F^2}{2}\,. \tag{E.1}$$

It follows

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha\mathbf{D}_k) \geq \gamma\alpha g_k \quad \text{for } \alpha \in \left[0, 2(1-\gamma)\frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right]\,, \tag{E.2}$$

and

$$\alpha_k > 2\delta(1-\gamma)\frac{g_k}{L\|\mathbf{D}_k\|_F^2}\,.$$

Therefore

$$\alpha_k \geq \min\left(1, 2\delta(1-\gamma)\frac{g_k}{L\|\mathbf{D}_k\|_F^2}\right) \geq \min\{1, 2\delta(1-\gamma)\}\bar{\alpha}_k\,, \tag{E.3}$$

which proves (4.10). Furthermore, it holds that

$$E(\mathbf{U}_k) - E(\mathbf{U}_k + \alpha_k\mathbf{D}_k) \geq \gamma\alpha_k g_k \geq \gamma\min\{1, 2\delta(1-\gamma)\}\bar{\alpha}_k g_k\,, \tag{E.4}$$

by using the Armijo condition (4.9) in the first inequality and (4.10) in the second. Thus, by $\gamma, \delta \in (0,1)$ and $\gamma(1-\gamma) \leq \frac{1}{4}$, our analysis shows that equation (4.6) is satisfied given that ρ is defined as $\rho = \gamma\min\{1, 2\delta(1-\gamma)\}$ and that this value remains less than 1. ∎

Appendix F. Proof of Corollary 4.3.

Proof. As proved in Theorem 2.3, when $\varepsilon < \bar{\varepsilon}$ the function $E(\mathbf{U})$ is strictly concave. Since the point $\mathbf{S}_0$ obtained through the `GLMO` is a vertex of the feasible set, so we have $\mathbf{S}_0 \in \{0,1\}^{n\times K}$. By exploiting strict concavity, we know that the univariate function $\phi(\alpha) = E(\mathbf{U} + \alpha\mathbf{D}_k)$ has its minimum in $\alpha = 1$, which is the stepsize that a linesearch embedded in GFW would give. By exploiting first order strict concavity we can indeed write

$$E(\mathbf{U}_0) - E(\mathbf{U}_1) > -\langle\nabla E(\mathbf{U}_0), (\mathbf{U}_1 - \mathbf{U}_0)\rangle > 0\,.$$

Therefore $\mathbf{U}_1 \in \{0,1\}^{n\times K}$, and since $\varepsilon < \tilde{\varepsilon}$, we have that $\texttt{GLMO}(\nabla E(\mathbf{U}_1))$ (and the classic `LMO`) would output $\mathbf{U}_1$ meaning that $\mathbf{U}_1$ is a stationary point for problem (2.8). This proves our result. ∎

REFERENCES

[1] D. Alfke, D. Potts, M. Stoll, and T. Volkmer, *NFFT meets Krylov methods: Fast matrix-vector products for the graph Laplacian of fully connected networks*, Frontiers in Applied Mathematics and Statistics, 4 (2018), p. 61.

[2] S. M. Allen and J. W. Cahn, *A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening*, Acta metallurgica, 27 (1979), pp. 1085–1095.

[3] K. Barnard, L. Martin, B. Funt, and A. Coath, *A data set for color research*, Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur, 27 (2002), pp. 147–151.

[4] K. Bergermann, M. Stoll, and T. Volkmer, *Semi-supervised learning for aggregated multilayer graphs using diffuse interface methods and fast matrix-vector products*, SIAM Journal on Mathematics of Data Science, 3 (2021), pp. 758–785.

[5] A. Bertozzi, S. Esedog¯lu, and A. Gillette, *Analysis of a two-scale Cahn–Hilliard model for binary image inpainting*, Multiscale Modeling & Simulation, 6 (2007), pp. 913–936.

[6] A. L. Bertozzi, S. Esedoglu, and A. Gillette, *Inpainting of binary images using the Cahn–Hilliard equation*, IEEE Transactions on image processing, 16 (2006), pp. 285–291.

[7] A. L. Bertozzi and A. Flenner, *Diffuse interface models on graphs for classification of high dimensional data*, Multiscale Model. Simul., 10 (2012), pp. 1090–1118, https://doi.org/10.1137/11083109X.

[8] D. P. Bertsekas, *Convex optimization algorithms*, Athena Scientific, Belmont, 2015, http://www.athenasc.com/convexalgorithms.html (accessed 2020-06-03).

[9] I. M. Bomze, F. Rinaldi, and D. Zeffiro, *Active set complexity of the away-step Frank–Wolfe algorithm*, SIAM Journal on Optimization, 30 (2020), pp. 2470–2500.

[10] I. M. Bomze, F. Rinaldi, and D. Zeffiro, *Frank–Wolfe and friends: a journey into projection-free first-order optimization methods*, 4OR, 19 (2021), pp. 313–345.

[11] J. Bosch, S. Klamt, and M. Stoll, *Generalizing diffuse interface methods on graphs: nonsmooth potentials and hypergraphs*, SIAM Journal on Applied Mathematics, 78 (2018), pp. 1350–1377.

[12] J. Bosch and M. Stoll, *A fractional inpainting model based on the vector-valued Cahn–Hilliard equation*, SIAM Journal on Imaging Sciences, 8 (2015), pp. 2352–2382.

[13] J. W. Cahn and J. E. Hilliard, *Free energy of a nonuniform system. I. Interfacial free energy*, J. Chem. Phys., 28 (1958), pp. 258–267, https://doi.org/10.1063/1.1744102, http://dx.doi.org/10.1063/1.1744102.

[14] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, and K. Murphy, *Machine learning on graphs: A model and comprehensive taxonomy*, Journal of Machine Learning Research, 23 (2022), pp. 1–64.

[15] Y. Chen and X. Ye, *Projection onto a simplex*, arXiv preprint arXiv:1101.6081, (2011).

[16] F. R. K. Chung, *Spectral graph theory*, vol. 92 of CBMS Regional Conference Series in Mathematics, Amer. Math. Soc., Providence, RI, 1997.

[17] L. Condat, *Fast projection onto the simplex and the l1 ball*, Mathematical Programming, 158 (2016), pp. 575–585.

[18] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, *Scientific machine learning through physics–informed neural networks: Where we are and what's next*, Journal of Scientific Computing, 92 (2022), p. 88.

[19] M. Frank and P. Wolfe, *An algorithm for quadratic programming*, Naval Research Logistics Quarterly, 3 (1956), pp. 95–110, https://doi.org/10.1002/nav.3800030109.

[20] C. Garcia-Cardona, E. Merkurjev, A. L. Bertozzi, A. Flenner, and A. G. Percus, *Multiclass data segmentation using diffuse interface methods on graphs*, IEEE Trans. Pattern Anal. Mach. Intell., 36 (2014), pp. 1600–1613, https://doi.org/10.1109/TPAMI.2014.2300478.

[21] M. Held, P. Wolfe, and H. P. Crowder, *Validation of subgradient optimization*, Mathematical programming, 6 (1974), pp. 62–88.

[22] A. Lancichinetti, S. Fortunato, and F. Radicchi, *Benchmark graphs for testing community detection algorithms*, Physical Review E—Statistical, Nonlinear, and Soft Matter Physics, 78 (2008), p. 046110.

[23] J. Leskovec and A. Krevl, *SNAP Datasets: Stanford large network dataset collection*. http://snap.stanford.edu/data, June 2014.

[24] P.-G. Martinsson and J. A. Tropp, *Randomized numerical linear algebra: Foundations and algorithms*, Acta Numerica, 29 (2020), pp. 403–572.

[25] E. Merkurjev, C. Garcia-Cardona, A. L. Bertozzi, A. Flenner, and A. G. Percus, *Diffuse interface methods for multiclass segmentation of high-dimensional data*, Appl. Math. Lett., 33 (2014), pp. 29–34, https://doi.org/10.1016/j.aml.2014.02.008, http://dx.doi.org/10.1016/j.aml.2014.02.008.

[26] E. Merkurjev, T. Kostić, and A. L. Bertozzi, *An MBO scheme on graphs for classification and image processing*, SIAM J. Imaging Sci., 6 (2013), pp. 1903–1930, https://doi.org/10.1137/120886935, http://dx.doi.org/10.1137/120886935.

[27] B. Merriman, J. K. Bence, and S. J. Osher, *Motion of multiple junctions: A level set approach*, Journal of computational physics, 112 (1994), pp. 334–363.

[28] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media, 2003.

[29] F. Rinaldi and D. Zeffiro, *Avoiding bad steps in Frank-Wolfe variants*, Computational Optimization

and Applications, (2022), pp. 1–40.
[30] R. T. Rockafellar, *Convex Analysis*, Princeton Landmarks in Mathematics and Physics, Princeton University Press, 1970.
[31] B. Rozemberczki, C. Allen, and R. Sarkar, *Multi-scale attributed node embedding*, 2019, https://arxiv.org/abs/1909.13021.
[32] B. Rozemberczki and R. Sarkar, *Characteristic Functions on Graphs: Birds of a Feather, from Statistical Descriptors to Parametric Models*, in Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), ACM, 2020, p. 1325–1334.
[33] B. Rozemberczki and R. Sarkar, *Twitch gamers: a dataset for evaluating proximity preserving and structural role-based node embeddings*, 2021, https://arxiv.org/abs/2101.03091.
[34] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, *Pitfalls of graph neural network evaluation*, Relational Representation Learning Workshop, NeurIPS 2018, (2018).
[35] A. Smagina, V. Bozhkova, S. Gladilin, and D. Nikolaev, *Linear colour segmentation revisited*, in Eleventh International Conference on Machine Vision (ICMV 2018), vol. 11041, International Society for Optics and Photonics, 2019, p. 110410F.
[36] Y. Van Gennip, N. Guillen, B. Osting, and A. L. Bertozzi, *Mean curvature, threshold dynamics, and phase field theory on finite graphs*, Milan Journal of Mathematics, 82 (2014), pp. 3–65.
[37] U. von Luxburg, *A tutorial on spectral clustering*, Stat. Comput., 17 (2007), pp. 395–416, https://doi.org/10.1007/s11222-007-9033-z, http://dx.doi.org/10.1007/s11222-007-9033-z.