SIMPLIFIED ENERGY LANDSCAPE FOR MODULARITY USING TOTAL VARIATION[*]

ZACHARY BOYD[†], EGIL BAE[‡], XUE-CHENG TAI[§], AND ANDREA BERTOZZI[¶]

Abstract. Networks capture pairwise interactions between entities and are frequently used in applications such as social networks, food networks, and protein interaction networks, to name a few. Communities, cohesive groups of nodes, often form in these applications, and identifying them gives insight into the overall organization of the network. One common quality function used to identify community structure is *modularity.* In Hu et al. [SIAM J. App. Math., 73(6), 2013], it was shown that modularity optimization is equivalent to minimizing a particular nonconvex total variation (TV) based functional over a discrete domain. They solve this problem—assuming the number of communities is known—using a Merriman, Bence, Osher (MBO) scheme.

We show that modularity optimization is equivalent to minimizing a *convex* TV-based functional over a discrete domain—again, assuming the number of communities is known. Furthermore, we show that modularity has no convex relaxation satisfying certain natural conditions. Despite this, we partially relax the discrete constraint using a Ginzburg Landau functional, yielding an optimization problem that is more nearly convex. We then derive an MBO algorithm with fewer parameters than in Hu et al. and which is 7 times faster at solving the associated diffusion equation due to the fact that the underlying discretization is unconditionally stable. Our numerical tests include a hyperspectral video whose associated graph has 2.9×10^7 edges, which is roughly 37 times larger than was handled in the paper of Hu et al.

Key words. social networks, community detection, data clustering, graphs, modularity, MBO scheme

AMS subject classifications. 65K10, 49M20, 35Q56, 62H30, 91C20, 91D30, 94C15

1. Introduction.

Community detection in complex networks is a difficult problem with applications in numerous disciplines, including social network analysis [52], molecular biology [36], politics [52], material science [5], and many more [53]. There is a large and growing literature on the subject, with many competing definitions of community and associated algorithms [22, 59, 24]. In practice, community detection is used as a way to understand the coarse, or mesoscale, properties of networks. Further investigation into these communities sometimes leads to insights about the processes that formed the network or the dynamics of processes acting on the network.

In this paper, we focus on the task of partitioning the nodes in a complex network into disjoint communities, although many other variations, such as overlapping, fuzzy, and time-dependent communities are also used in the literature. The proper way to understand such communities in small networks has been fairly well-studied, and their role in larger networks is the subject of active research [38].

A great variety of definitions have been proposed to make the partitioning task precise [24], including notions involving edge-counting, random walk trapping, information theory, and—especially recently—generative models such as stochastic block models (SBMs). In this paper, we focus on modularity optimization [55], which is the most well-studied of existing methods. Let G be a non-negatively weighted, undirected, sparse graph with N nodes, weight matrix $W = (w_{ij})$, degree vector k satisfying $k_i = \sum_j w_{ij}$, and $2m = \sum_i k_i$. Modularity-optimizing algorithms seek a

[*] Submitted to the editors DATE.

Funding: Z. Boyd was funded by an NDSEG Fellowship.

[†] Department of Mathematics, UCLA, Los Angeles, CA (zach.boyd@math.ucla.edu).

[‡] Norwegian Defense Research Establishment (FFI), Kjeller, Norway (Egil.Bae@ffi.no).

[§] Department of Mathematics, University of Bergen, Bergen, Norway (tai@math.uib.no).

[¶] Department of Mathematics, UCLA, Los Angeles, CA (bertozzi@math.ucla.edu).

partition $A_1, \cdots, A_{\hat{\text{n}}}$ of the nodes of G which maximizes

$$Q = \frac{1}{2m} \sum_{\ell=1}^{\hat{\text{n}}} \sum_{ij \in A_\ell} w_{ij} - \frac{k_i k_j}{2m}.$$

Intuitively, we are to understand w_{ij} as the observed edge weight and $\frac{k_i k_j}{2m}$ as the expected weight if the edges had been placed at random. Thus, there is an incentive to group those nodes which have an unusually strong connection under the null model.

The results of modularity optimization must be interpreted carefully. For example, the modularity functional, Q, will find communities in a random graph [34]. In addition, many dissimilar partitions may yield near-optimal modularity values [33]. This is to be expected, since the network partitioning problem is very well posed. Real networks are generated by complicated processes with many factors, and thus there are often multiple ways to partition a network that reflect legitimate divisions among the objects being studied [58]. One way to leverage this diversity of high-modularity partitions in practice, as well as prevent the discovery of communities in random graphs, relies on consensus clustering [73]. Another approach is simply to sample many high-modularity partitions, expecting that multiple intuitively-meaningful partitions may be found. Such effects effects have been observed, for instance, in the Zachary Karate Club network, which has both a community structure and leader-follower structure [58].

Modularity also has preferred scale for communities [23, 44]. For this reason, one typically includes a resolution parameter $\gamma > 0$ [60, 4], yielding

$$Q = \frac{1}{2m} \sum_{\ell=1}^{\hat{\text{n}}} \sum_{ij \in A_\ell} w_{ij} - \gamma \frac{k_i k_j}{2m}$$

When γ is nearly zero, the incentive is to place many nodes in the same community, so that the edge weight is included in the sum. When γ is large, few nodes are placed in each community, to avoid including the large penalty term $\gamma \frac{k_i k_j}{2m}$.

A number of heuristics have been proposed to optimize modularity [22, 24], with prominent approaches including spectral [54], simulated annealing [34], and greedy or Louvain algorithms [7]. The optimization problem is NP-hard [8], so it is not expected that a single heuristic will suffice for all situations.

In 2013, Hu, Laurent, Porter, and Bertozzi [37] discovered a connection between the modularity optimization problem in network science and total variation (TV) minimization from image processing. As an application, Hu et al. developed Modularity MBO, a TV-oriented optimization algorithm that effectively optimizes modularity. The present work strengthens both theoretical and algorithmic connections from [37]. Specifically, we make the following contributions:

We provide four additional formulations of modularity, two in terms of TV and two in terms of graph cuts. In addition to being intuitively simpler than the formulation in [37], these formulas place all of the nonconvexity of the problem into a discrete constraint and the choice of the number of classes—the functionals themselves are convex. This new formulation leads quickly to a theorem showing that convex relaxation of modularity is not possible under certain conditions. While many practitioners have observed that modularity optimization seems highly nonconvex, ours is the first result of which we are aware showing this in a rigorous way. We provide an alternative relaxation, using the Ginzburg-Landau functional, that relaxes the discrete constraint in a way that is not convex but is more manageable. As the amount of relaxation goes to zero, solutions of the relaxed problem approach solutions of the discrete problem.

Based on these ideas, and following [37], we develop an MBO-type scheme, Balanced TV, which quickly and accurately optimizes modularity in several examples. This algorithm seems especially well-suited to use on similarity networks that arise in machine learning, where prior knowledge of the approximate number of communities is available, and the number of such communities tends to be small. We provide practical help for implementing Balanced TV that eliminates the need to hand-tune certain parameters, as is necessary in [37]. This includes bounds on the timestep in the MBO iteration and a method to avoid using an inner loop in that iteration. We also show how to discretize the partial differential equation (PDE) part of the MBO iteration in an unconditionally stable, efficient way. We test our algorithm on much larger datasets than are used in [37], giving insight into scalability, which typically is very good for MBO-type algorithms. These examples include hyperspectral video, which makes evaluation of the resulting communities relatively easy. In general, we find that the modularity approach does capture the essence of these machine learning problems, although the simplicity of our approach necessarily means that more specialized algorithms may have an advantage in specific cases. In addition, our algorithm typically optimizes modularity as accurately as the Louvain method, which is one of the best tools available for modularity optimization on large graphs.

The rest of the paper is organized as follows: Section 2 surveys the necessary background in both modularity optimization and TV minimization. Section 3 develops the main theoretical results about the optimization problem itself. Section 4 develops the theory and practical implementation of our algorithm, Balanced TV. Section 5 gives numerical examples. Section 6 concludes. There are also appendices containing additional background and deferred proofs.

2. Total Variation Optimization.

While modularity optimization is normally understood as a combinatorial problem, TV was historically seen as a continuum object, with applications in partial differential equations, physics simulation, and image processing.[1] Given a smooth function f from some domain $U \subset \mathbb{R}^n$ to $\mathbb{R}$, we define the TV of f as

$$|u|_{TV} = \int_U |\nabla f|.$$

In the special case where $n = 1$, this is the total rise and fall of the function, hence the name. An important special case is when $n = 2$ or 3 and f is the indicator function of a region $V \subset U$. In such a case, $|f|_{TV}$ is the perimeter or surface area of V.

Total variation minimization is an important heuristic in image processing, where e.g. a black and white image that is corrupted by noise can be viewed as a function $f : [0,1]^2 \to [0,1]$, where the value of f varies from 0 (black) to 1 (white). A common task is to remove the noise and recover the original image. Since noise is manifest as large gradients in f, early approaches found u as the solution to a minimization problem such as

$$\min_u \int_{[0,1]^2} ||\nabla u||^2 + ||u - f||^2.$$

The solution to such a problem is a smoothed image, which means that the noise is eliminated, but all edges are also erroneously eliminated. The correct approach [61], is to modify the problem as follows

$$\min_u \int_{[0,1]^2} ||\nabla u|| + ||u - f||^2.$$

[1] See [11] for a more complete treatment.

This small change allows the minimization procedure to preserve edges and yields much better results in many applications. The reason is that minimizers of total variation tend to be *piecewise* smooth. Total variation minimization has other applications as well, such as compressed sensing [10] and mean curvature flow [12, 43].

The preceding discussion has assumed that the domain in question is some subset of $\mathbb{R}^n$, but the main ideas above can be carried out on a graph as well [29]. We first define the nonlocal gradient of a function $f : G \to \mathbb{R}$ at node i in the direction of the edge from i to j as

$$\nabla f(i,j) = f(j) - f(i).$$

The 1-norm of ∇f at node i is then given by

$$|\nabla f(i)|_1 = \sum_j w_{ij} |f(j) - f(i)|.$$

Finally, we can use summation to replace integration, yielding the *graph total variation*

$$|f|_{TV} = \frac{1}{2} \sum_{ij} w_{ij} |f(j) - f(i)|. \tag{1}$$

We will actually use a slight generalization of Equation 1; in the case where f is vector-valued, we identify it with an $N \times \hat{\mathrm{n}}$ matrix where the i-th row is $f(i)$ and write

$$|f|_{TV} = \frac{1}{2} \sum_{ij\ell} w_{ij} |f_{i\ell} - f_{j\ell}|.$$

We also want an associated notion of perimeter for graphs, which we will term the graph cut. Let $f : G \to \mathbb{R}$ be the characteristic function of a set of nodes S. Then we can calculate

$$|f|_{TV} = \frac{1}{2} \sum_{ij} w_{ij} |f(i) - f(j)| = \sum_{i \in S, j \in S^c} w_{ij} := \mathrm{Cut}(S, S^c). \tag{2}$$

TV minimization on a graph tends to produce piecewise-constant functions whose corresponding graph cut is small.

3. Equivalence Theorem and its Consequences.

In this section, we prove and explore some consequences of our main result. We will need this definition:

Definition 3.1. *Let $\Pi(G)$ be the set of all partitions of the nodes of G. We identify the nodes of G with the integers 1 through N. Thus if $\{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}} \in \Pi(G)$, then we write $i \in A_\ell$ if node i lies in the ℓ-th component of the given partition.*

To each element of $\Pi(G)$, we identify a matrix in $\mathbb{R}^{N \times \hat{\mathrm{n}}}$, called the partition matrix satisfying, $u_{i\ell} = \delta(i \in A_\ell)$, where δ is the Kronecker delta. Thus, $u \in \Pi(G)$ means that u corresponds to some partition of the nodes of G under the correspondence just given.

3.1. Main Result.

Theorem 3.2 (Equivalent forms of modularity). *For any subset S of the nodes of G, define $\operatorname{vol} S = \sum_{i \in S} k_i$. Then the following optimization problems are all equivalent:*

(3)

Standard formulation: $$\underset{\hat{\mathrm{n}} \in \mathbb{N}, \{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}} \in \Pi(G)}{\operatorname{argmax}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} w_{ij} - \gamma \frac{k_i k_j}{2m}$$

(4)

$$\textit{Balanced cut (I):} \qquad \underset{\hat{\mathrm{n}}\in\mathbb{N},\{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}}\in\Pi(G)}{\operatorname{argmin}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}}\left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m}\,(\mathrm{vol}\,A_\ell)^2\right)$$

(5)

$$\textit{Balanced cut (II):} \qquad \underset{\hat{\mathrm{n}}\in\mathbb{N},\{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}}\in\Pi(G)}{\operatorname{argmin}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}}\left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m}\left(\mathrm{vol}\,A_\ell - \frac{2m}{\hat{\mathrm{n}}}\right)^2\right) + \gamma\frac{2m}{\hat{\mathrm{n}}}$$

(6)

$$\textit{Balanced TV (I):} \qquad \underset{\hat{\mathrm{n}}\in\mathbb{N},u\in\Pi(G)}{\operatorname{argmin}} \quad |u|_{TV} + \frac{\gamma}{2m}\left|\left|k^T u\right|\right|_2^2$$

(7)

$$\textit{Balanced TV (II):} \qquad \underset{\hat{\mathrm{n}}\in\mathbb{N},u\in\Pi(G)}{\operatorname{argmin}} \quad |u|_{TV} + \frac{\gamma}{2m}\left|\left|k^T u - \frac{2m}{\hat{\mathrm{n}}}\right|\right|_2^2 + \gamma\frac{2m}{\hat{\mathrm{n}}}$$

Each of the preceding forms has a different interpretation. The original formulation of modularity was based on comparison with a statistical model and views communities as regions that are more connected than they would be if edges were totally random. The cut formulations represent modularity as favoring sparsely interconnected regions with balanced volumes, and the TV formulation seeks a piecewise-constant partition function u whose discontinuities have small perimeter, together with a balance-inducing quadratic penalty. The cut and TV forms come in pairs. The first form (labelled "I") is simpler to write but harder to interpret, while the second (labelled "II") has more terms, but the nature of the balance term is easy to understand, as it is minimized (for fixed $\hat{\mathrm{n}}$) when each community has volume $\frac{2m}{\hat{\mathrm{n}}}$.

One can compare these equivalent formulations with [37], in which minimizing the functional

$$|u|_{TV} - \gamma||u - \mathrm{mean}(u)||^2_{\ell^2(G)} = |u|_{TV} - \gamma\sum_{i\ell} k_i \left|u_{i\ell} - \frac{1}{2m}\sum_{i'=1}^{N} k_i u_{i'\ell}\right|^2 \tag{8}$$

is shown to be equivalent to modularity optimization, subject to the same constraint as the other TV formulas presented here. Thus, in [37], there are two sources of nonconvexity, namely the balance term and the constraint, while in our formulation, the discrete constraint is the only source of nonconvexity. It is also more clear from our formulation which features of a solution are incentivized by modularity optimization, namely, the two priorities of having a small graph cut and balanced class sizes are the only considerations. The relative weight of these considerations, as well as the number of communities, is governed by γ. The fact that the second term induces balance can be seen from (5), since if each community has exactly $\frac{2m}{\hat{\mathrm{n}}}$ volume, there is minimal penalty from that term. Overall, these theoretical simplifications should make the nonconvexity of the problem easier to navigate, both for humans and computers.

Finally, we note that all of these formulations of modularity provide a convenient way to incorporate metadata into the partitioning process. This can be done by simply incorporating a fidelity term and minimizing the functional

$$|u|_{TV} + \frac{\gamma}{2m}||k^T u||_2^2 + \lambda||\chi * (u - f)||_2^2$$

where $\lambda > 0$ is a parameter, f is a term containing the metadata labels, $*$ is the entry-wise matrix product, and χ is a matrix that is zero except in the entries where labels are known. Including metadata should always be done with care, of course, but this device should prove very convenient, as it has been well-tested in image processing and machine learning applications.

Theorem 3.2. Notice that the cut and TV formulations are really just a change of notation, so that there are two nontrivial equivalences, namely the equivalence of (3) with (4) and the equivalence of (4) and (5). We first show the equivalence of (3) with (4). Fix $\hat{\mathrm{n}}$, and consider an otherwise arbitrary partition $\{A_1, \dots, A_{\hat{\mathrm{n}}}\}$ of G. Then we have

$$Q = \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} w_{ij} - \gamma \frac{k_i k_j}{2m} \tag{9}$$

$$= \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\sum_{i \in A_\ell, j \in \{1,\dots,N\}} w_{ij} - \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} \right) - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{10}$$

$$= \frac{1}{2m} \sum_{ij=1}^{N} w_{ij} - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{11}$$

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{12}$$

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \mathrm{Cut}(A_\ell, A_\ell^c) - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m}. \tag{13}$$

Summing along the j index first yields

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}(A_\ell, A_\ell^c) + \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{i \in A_\ell} k_i \operatorname{vol} A_\ell \right) \tag{14}$$

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}(A_\ell, A_\ell^c) + \frac{\gamma}{2m} (\operatorname{vol} A_\ell)^2 \right) \tag{15}$$

Thus, the maxima of modularity coincide with the minima the functional from (4), as required.

To see that (4) and (5) are equivalent, we calculate:

$$\sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m} \left(\operatorname{vol} A_\ell - \frac{2m}{\hat{\mathrm{n}}} \right)^2 \right) \tag{16}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m} \left((\operatorname{vol} A_\ell)^2 - \frac{4m}{\hat{\mathrm{n}}} \operatorname{vol} A_\ell + \frac{4m^2}{\hat{\mathrm{n}}^2} \right) \right) \tag{17}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m} (\operatorname{vol} A_\ell)^2 \right) - \frac{\gamma}{2m} \frac{8m^2}{\hat{\mathrm{n}}} + \frac{\gamma}{2m} \frac{4m^2}{\hat{\mathrm{n}}} \tag{18}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\,(A_\ell, A_\ell^c) + \frac{\gamma}{2m} (\operatorname{vol} A_\ell)^2 \right) - \gamma \frac{2m}{\hat{\mathrm{n}}} \tag{19}$$

□

3.2. On convex relaxations. The preceding equivalence theorem makes it very tempting to look for a convex relaxation of (6), but this is not really possible,[2]as the following theorem shows. We will use the term *symmetric* to describe a functional that is invariant under column permutations of the input matrix. In the case where the input lies in $\Pi(G)$, this is the same as permuting the group labels.

THEOREM 3.3. *Let*

$$\mathcal{F}(u) = |u|_{TV} + \frac{\gamma}{2m}||k^T u||_2^2$$

have domain $\Pi(G, \hat{\mathrm{n}}) = \Pi(G) \cap \mathbb{R}^{N \times \hat{\mathrm{n}}}$*, and let* $\tilde{\mathcal{F}}$ *be any symmetric, convex extension of* $\mathcal{F}$ *to the convex hull of* $\Pi(G, \hat{\mathrm{n}})$*. Then* $\tilde{\mathcal{F}}$ *has a trivial, global minimizer* $\tilde{u}$ *that has all columns equal to each other, thus yielding no classification information.*

Proof. Consider an arbitrary matrix u in the convex hull of $\Pi(G, \hat{\mathrm{n}})$. For each permutation of the columns $\pi \in S_{\hat{\mathrm{n}}}$, we have $\mathcal{F}(u) = \mathcal{F}(\pi u)$. Let $\tilde{u} = \frac{1}{\hat{\mathrm{n}}!} \sum_{\pi \in S_{\hat{\mathrm{n}}}} \pi u$. Then by Jensen's inequality we have

$$\mathcal{F}(\tilde{u}) = \tilde{\mathcal{F}}\left(\frac{1}{\hat{\mathrm{n}}!} \sum_{\pi \in S_{\hat{\mathrm{n}}}} \pi u\right) \leq \frac{1}{\hat{\mathrm{n}}!} \sum_{\pi \in S_{\hat{\mathrm{n}}}} \tilde{\mathcal{F}}(\pi u) = \tilde{\mathcal{F}}(u).$$

Since u was arbitrary, $\tilde{u}$ is a global minimizer. Finally, all the columns of $\tilde{u}$ are equal, as promised.□

3.3. Ginzburg-Landau Relaxation. In this subsection, we develop a way to relax the modularity problem to a continuum domain, which can make the nonconvexity more manageable. In other TV problems arising in materials science and image processing, discrete constraints similar to modularity's are dealt with using the idea of *phase fields*, where a thin transition layer between discrete-valued regions is allowed, making the problem smooth so that it can be attacked by continuum methods. (See e.g. [66, 20, 2, 6].) As discussed above, TV is used for two of its properties: promoting small perimeter and encouraging binary results. The Ginzburg-Landau relaxation replaces the TV term with two other terms: the Dirichlet energy and a multiwell potential, each of which has one of the aforementioned properties. Thus the Ginzburg-Landau energy in the continuum is given by

$$F_\epsilon(u) = \int_U \epsilon||\nabla u(x)||^2 + \frac{1}{\epsilon}P(u(x))\,dx,$$

where ϵ is a small parameter and P is a multiwell potential with local minima at the corners of the simplex. The exact form of P will not be important for our purposes, but we will give a concrete example in the next theorem. A classical result asserts that for $u : U \subset \mathbb{R} \to \mathbb{R}$ and P having minima at 0 and 1, we have the following convergence[3]result:

$$F_\epsilon(u) \xrightarrow{\Gamma} \begin{cases} \text{const } |u|_{TV} & \text{if } u \text{ is binary} \\ +\infty & \text{otherwise} \end{cases}$$

as $\epsilon \to 0$, under appropriate conditions.

[2] We do note, however, that by means of a different embedding [15] was able to obtain a convex relaxation with solutions which, while not discrete, are also not trivial. Thus, the embedding requirement is a significant part of our theorem. Other related works include [13] and [1].

[3] See the appendices for an overview of Γ-convergence.

In order to arrive at the graph Ginzburg-Landau functional, observe that if we ignore boundary terms, then

$$\int_U ||\nabla u||^2 = \int_U (\nabla u, \nabla u) = \int_U -(\operatorname{div} \nabla u, u) = \int_U -(\Delta u, u), \tag{20}$$

which suggests that we use an appropriate graph Laplacian in our formulation. The Laplacian that is appropriate for our context is the *combinatorial* or *unnormalized Laplacian*, $L = \operatorname{diag}(k) - W$.

In [6], the idea of using a Ginzburg-Landau functional in graph-based optimization first appeared, and it has subsequently been treated in more depth in [63], where much of the continuum theory was successfully extended to graphs. Our approach closely mirrors [37], the main difference in this case simply being that our functionals have different convexity properties. For general $\hat{n}$, a complete theory of convergence on graphs has not been developed yet, although in applications convergence seems not to depend on $\hat{n}$. With $\hat{n} = 2$, we have the following result:

THEOREM 3.4 (Γ-convergence for the binary balanced TV problem). *Assume $\hat{n} = 2$, and let u be a single binary-valued column. Assume furthermore that $P(u_i) = u_i^2(1 - u_i^2)$. Then the functionals*[4]

$$\mathcal{F}_\epsilon = ||\nabla u||_2^2 + \frac{1}{\epsilon}\sum_{i=1}^{N} P(u_i) + \frac{\gamma}{2m}\left((k^T u)^2 + (k^T(1-u))^2\right) \tag{21}$$

$$:= u^T L u + \frac{1}{\epsilon}\sum_{i=1}^{N} P(u_i) + \frac{\gamma}{2m}\left((k^T u)^2 + (k^T(1-u))^2\right), \tag{22}$$

defined over all of R^N, Γ-converge to the functional

$$|u|_{TV} + \frac{\gamma}{2m}\left((k^T u)^2 + (k^T(1-u))^2\right),$$

which is understood to take the value $+\infty$ whenever u is not binary. In particular,

- *for any sequence $\epsilon_n \to 0$, and any corresponding sequence u_ϵ of minimizers of $\mathcal{F}_{\epsilon_n}$, there is a subsequence that converges to a maximizer of (binary) modularity, and*
- *any convergent subsequence of the u_ϵ converges to a maximizer of (binary) modularity.*

The proof is given in the appendices.

Moving forward, we focus on minimizing the relaxed functionals from Theorem 3.4, or more precisely their multi-class analogue:

$$\operatorname{tr}\left(u^T L u\right) + \frac{1}{\epsilon}\sum_{i=1}^{N} P(u_i) + ||k^T u||_2^2, \tag{23}$$

where $u \in \mathbb{R}^{N\times\hat{n}}$, and P is a potential with minima at the standard basis vectors of $\mathbb{R}^{\hat{n}}$. This should give a good approximation of the true modularity maximization problem.

While using the Ginzburg-Landau functional does introduce a Laplacian into our formulation, we stress that this approach is different from spectral heuristics such as spectral clustering—the preceding result on Γ-convergence shows that the real object we are aiming for is TV. Later on, we will see that our approach gives very different results from spectral clustering.

[4] Note that due to the discrete setting, there is no epsilon factor preceding the Laplacian term, see [63].

4. Numerical Scheme.

4.1. MBO iteration.

We minimize the functional from (23) using an adaptation of the graph MBO scheme. We call our approach Balanced TV. The acronym "MBO" stands for Merriman, Bence, and Osher [51], who introduced this algorithm in Euclidean space. It has been widely used as an approach to motion by mean curvature and TV minimization. The connection between graph-based TV and MBO was first made in [49] and [26]. The theoretical study of the algorithm on graphs was initiated in [64]. We sketch the logic of MBO here and refer the reader to [51] for a more complete treatment. The scheme works by approximating the gradient descent flow of the Ginzburg-Landau functional in the case where ϵ is very small. Consider the Ginzburg-Landau gradient descent equation (at fixed $\hat{\text{n}}$)

$$u_t = -Lu - \frac{1}{\epsilon}P'(u) - \frac{\gamma}{m}kk^T u.$$

One way to approximate this flow is by operator splitting [30]. Given u^n one obtains $u^{n+\frac{1}{2}}$ as the solution to

$$u_t = -Lu - \frac{\gamma}{m}kk^T u \tag{24}$$

for $t_n < t < t_n + \frac{1}{2}dt$ and initial data u^n. Then one gets u^{n+1} by solving

$$u_t = -\frac{1}{\epsilon}P'(u) \tag{25}$$

for $t_n + \frac{1}{2}dt < t < t_n + dt$ and initial data $u^{n+\frac{1}{2}}$. The iteration continues until a fixed point is reached. Such operator splitting schemes are typically first-order accurate in time. In the case where ϵ is very small, the second flow is essentially a thresholding operation, pushing all values of u into the nearest well, i.e.

$$u^{n+1}_{i\ell} = \begin{cases} 1 & \ell = \text{argmax}_{\hat{\ell}}\, u^{n+\frac{1}{2}}_{i\hat{\ell}} \\ 0 & \text{otherwise} \end{cases}$$

This gives the MBO scheme:

```
Balanced TV MBO scheme

Initialize $u$ randomly.
Set $n = 0$.
while A stationary point has not been reached do
    $u^{n+\frac{1}{2}} = e^{-dtM}u^n$ where $M = L + \frac{\gamma}{m}kk^T$
    $u^{n+1} = \text{threshold}(u^{n+\frac{1}{2}})$
    $n = n + 1$
end while
```

The most expensive part of this procedure is evaluating the matrix exponential. We accomplish this efficiently using a pseudospectral scheme, which will be described below.

In the foregoing, we have made the decision to treat the forcing term implicitly, which differs from several recent studies, such as [37, 6, 49]. This can be done efficiently because the operator

M is positive semi-definite and can be applied to a vector in linear time, assuming A is sparse. Implicit treatment has the advantage of avoiding an inner loop, which is time-consuming, has a timestep-restriction, and adds another user-set parameter, namely the inner loop timestep. For this reason, the implicit treatment described herein is much easier and faster than the typical nested-loop approach.

As stated, we assume from here on that A is sparse. The case where A is dense could be approached using the Nyström method, as in [6]. Beware, however, that one must find a way to estimate k and $2m$ efficiently, which is not obvious. An alternative is to sparsify the network in preprocessing, which is the approach taken in our examples. This is generally cheap compared to the cost of partitioning the resulting sparse network.

4.2. Treating the matrix exponential. As stated above, the most time-intensive step in the MBO iteration is the matrix exponential, and this step is repeated many times. Therefore, it makes sense to use a pseudospectral scheme, as described in, for instance, [6]. This means that we precompute the eigenvalues and eigenvectors of M, and use them to solve the matrix exponential. By doing the eigenvalue calculation up front, each iteration is greatly accelerated. Here is how the scheme looks:

```
Pseudospectral Balanced TV MBO scheme

Initialize $u$ randomly.
Calculate the eigenvalues of $M$, and form the diagonal matrix $D$ with its diagonals being the eigenvalues.
Also calculate the eigenvectors and form the matrix $V$ whose columns are the eigenvectors.
while a stationary point has not been reached do
    $a^n = V^T u^n$.
    $a^{n+1} = e^{-dtD} a^n$
    $u^{n+\frac{1}{2}} = V a^{n+1}$
    $u^{n+1} = \text{threshold}(u^{n+\frac{1}{2}})$.
end while
```

In practice, it may not be possible to calculate the full spectrum of M, if M is large. In this case, we calculate the N_{eig} smallest eigenvalues and eigenvectors of M. Then instead of changing coordinates using a full matrix, use the $N \times N_{\text{eig}}$ matrix V exactly the same way as before. This is equivalent to projecting onto a subspace generated by these eigenvectors, and it makes the algorithms very efficient.

To understand the effect of computing only a few eigenvectors, recall that M is positive semidefinite. Therefore, it has an orthonormal basis of eigenvectors, and the evolution we are solving, namely $u_t = -Mu$, can be diagonalized as $a_t = -Da$ where $a = V^T u$, and V is the full matrix of eigenvalues, and D is a non-negative, diagonal matrix. Therefore, the evolution occurs in distinct "modes", with rates of decay controlled by the eigenvalues of M. The modes corresponding to small eigenvalues persist longer than those corresponding to large eigenvalues (which experience stiff exponential decay), so that it is not a bad approximation to simply project these components away when it is numerically necessary. Thus, in practice, we collect the smallest eigenvectors of M and the corresponding eigenvectors, neglecting the others.

We use Anderson's iterative Rayleigh-Chebyshev code [3]—which the author kindly provided to us—to get the eigenvalues and eigenvectors. We generally set $N_{\text{eig}} = 5\,\hat{\text{n}}$.

4.3. Determining the number of communities. The preceding algorithm assumes a fixed $\hat{\mathrm{n}}$. In practice, we found three methods of determining the value of $\hat{\mathrm{n}}$:

1. Use domain knowledge—for instance, in two moons, it is known that there are two communities,
2. Try several values of $\hat{\mathrm{n}}$ and take whichever one produces the best modularity—this works best in cases where there are few communities, as in MNIST. Note that the most time consuming part of the MBO scheme, namely computation of eigenvectors need only be done once, so that several different values of $\hat{\mathrm{n}}$ can be tried without incurring much extra cost.
3. Recursively partition the network—this works when many communities are present, as in the LFR networks. The partition is only made at each step if it increases modularity. This approach worked well in our examples, although in the case of LFR, where $O(N)$ communities are present, a lot of recursion is needed. This is compensated for by the fact that the subgraphs grow smaller and smaller near the end.

4.4. Scaling. We expect the scaling of our approach to be roughly linear, as suggested by the following informal argument. The main components of the algorithm are

1. finding eigenvalues and eigenvectors (probably $O(N \log^q N)$ for some q),[5]
2. changing coordinates using only the leading eigenvectors ($O(N)$ per iteration, with empirically $O(1)$ iterations needed to converge),
3. evaluating the exponential of a vector componentwise (also $O(N)$ per iteration), and
4. thresholding ($O(N)$ per iteration).

The preceding estimates all apply in the case where no recursion is needed, i.e. the number of communities is known in advance. If the recursion is done by partitioning the graph into $\hat{\mathrm{n}}$ pieces at each level, then the cost is heuristically on the order of

$$\tilde{O}(N) + \hat{\mathrm{n}}\,\tilde{O}\left(\frac{N}{\hat{\mathrm{n}}}\right) + \hat{\mathrm{n}}^2\,\tilde{O}\left(\frac{N}{\hat{\mathrm{n}}^2}\right) + \cdots + O(N)O(1) = \tilde{O}(N)$$

where $\tilde{O}$ means that logarithmic terms are neglected, and each term in the sum is the product of the number of partitioning problems to be solved with the size of the partitioning problems. This scalability is roughly borne out in our example data sets, although we warn that there are additional complications, based on the varying number of communities to be produced, differences in the efficiency of parallelization at different scales, and possibly other factors.

4.5. On the choice of timestep. Our approach requires the selection of parameters γ, dt, N_{eig}, $\hat{\mathrm{n}}$, and various other parameters and methods. In order to simplify the exploration of this parameter space in practical applications, it is useful to have some theory about the choice of these parameters. Here, we describe how to set dt in the MBO scheme. This is especially useful in the recursive implementation, as the appropriate timestep empirically decreases as the graph gets smaller, and it would be laborious for a human to check at each recursion step.

Our derivations are inspired by those in [64], and proofs are deferred to an appendix. First, we consider a lower bound on the timestep:

PROPOSITION 4.1 (Lower bounds on the timestep). *Let $u_0 \in \Pi(G, \hat{\mathrm{n}})$. If u satisfies $u_t = -Mu$ with initial data u_0, then we have the following bounds:*

[5] There is no rigorous result for the Rayleigh-Chebyshev procedure, but numerical evidence suggests strongly better than quadratic convergence, and $O(N \log^q N)$ is the convergence speed for some similar algorithms.

1. $$||u(\tau) - u_0||_\infty \leq e^{2(\gamma+1)k_{\max}\tau}.$$

2. *In the case where* $\hat{\mathrm{n}} = 2$*, this bound implies that if the MBO timestep* τ *satisfies*

$$\tau < \frac{\log 2}{2(\gamma+1)k_{\max}} \approx \frac{0.15}{(\gamma+1)k_{\max}},$$

then the MBO iteration is stationary.

3. *If* ρ *is the spectral radius of* M*, we also have*

$$||u(\tau) - u_0||_\infty \leq \sqrt{\hat{\mathrm{n}}}||u_0||_2 \left(e^{\tau\rho} - 1\right).$$

4. *If* $\hat{\mathrm{n}} = 2$*, the MBO iteration is guaranteed to be stationary whenever*

$$\tau < \rho^{-1} \log\left(1 + N^{-\frac{1}{2}}\right).$$

Although we had to restrict to $\hat{\mathrm{n}} = 2$ in the above, we used the timestep restriction regardless of $\hat{\mathrm{n}}$—indeed the authors expect that $\hat{\mathrm{n}} = 2$ is the worst case, although we are unable to prove it at present.

The upper bound on the timestep is more delicate. Normally, the upper bound would be determined by convergence theory, using error bounds and stability estimates, the theory of which is incomplete in the graph setting at present. Instead, we use the following heuristic to motivate our bounds: In most cases, M is strictly positive definite, so the evolution $u_t = -Mu$ forces u to decay toward 0. The idea behind MBO is that the diffusion effects give information about curvature on short time scales, and the long time scales give information about more global quantities, which is useless in that context. Therefore, in the graph context, it makes sense to try to understand the time scale that is "long" and set the timestep to be shorter than that. Using the approach to 0 as a convenient notion of long-time behavior, we obtain the following useful bounds:

Proposition 4.2 (Decay estimates for M). *Let* $u_t = -Mu$ *with initial data* $u_0 \in \Pi(G, \hat{\mathrm{n}})$*. Then the following bounds hold:*

1. *Assume* λ_1 *is the smallest eigenvalue of* M*. Then*

$$||u||_2 \leq e^{-\tau\lambda_1}||u_0||_2.$$

2. *Let* M *be nonsingular. Then for any* $\epsilon > 0$*, we have* $||u(\tau)||_\infty < \epsilon$ *if*

$$\tau > \lambda_1^{-1} \log\left(\frac{||u_0||_2}{\epsilon}\right).$$

In practice, setting the timestep as the geometric mean between this upper bound and the lower bound from Proposition 4.1 has produced good results without resorting to hand-tuning of parameters.[6]

[6] We also found empirically that a simple time stepping procedure improved results sometimes: Let the algorithm run to convergence, then continue with a smaller timestep until convergence occurs again.

5. Results. Table 1 summarizes the results of our Balanced TV algorithm on various test datasets. We compared our results to the Modularity MBO algorithm from [37]. We chose conservative timesteps for Modularity MBO, which slowed it down, but this was necessary to get it to work on all of the test datasets. With careful tuning of all the parameters, Modularity MBO can be made more competitive. We also compared to the GenLouvain code from [39], which implements the well-known Louvain method. Note that the timing comparison with GenLouvain is not very important, since there exist much faster Louvain codes, e.g. igraph's C library routines [17]. All three codes tested here are written in MATLAB, with C or C++ calls for the most costly parts, such as eigenvector computation. All three codes are in principle dependent on random number generation, so our numerical tests had to be repeated several times to understand the variance. In all cases, we report the best outcomes for modularity and metadata matching and the medians for the timings.

5.1. Summary of Results. We found that Balanced TV gave the highest-modularity partition on 4 of the 6 types of data. The modularity of these partitions was also better than the metadata ("ground truth") partition in 2 of the 3 experiments where metadata was available, which means that the limiting factor in these cases is not the Balanced TV algorithm, but rather the small amount of noise that enters the problem when reducing to the graph structure and restricting to the modularity optimization framework, which considers only graph cuts and cluster volumes. We emphasize that for this reason, one must be very careful not to interpret the metadata matching as a measure of each algorithm's success, but rather as a measure of the overall success of the entire process, including graph construction, the modularity optimization framework, and the optimization scheme used on that problem. The modularity value obtained is the way to measure the success of the different optimization algorithms.

In terms of timing, we find that Balanced TV is best by far, probably due to a combination of the simplified energy landscape, smarter timestep selection, elimination of the inner loop, and lack of explicitly-treated forcing terms. For the imaging problems, Figures 1 and 2 show that Balanced TV gives results at least as good as other approaches, such as NLTV [74] and spectral clustering [67], and our results are in some respects clearly superior, especially on the plume data.

5.2. Analysis of each experiment. We now describe each test we performed and interpret the results.

Two Moons Two moons consists of 2,000 points, embedded in 100-dimensional space, with variance 0.02, with 98 noisy dimensions. We used a 13-nearest neighbors graph with the edge weights given by a Gaussian law, with locally-determined decay parameters [72]. The number of classes was assumed known. These results demonstrate that both noisy dimensions and nonconvex geometry can be handled. Thirty-five tests were performed for each algorithm.

MNIST MNIST consists of 70,000 28x28-pixel images, each of which contains a single handwritten digit [46]. The task is to identify the digit in each image. The graph was constructed by projecting onto 50 principle components for each image and then using a 10-nearest neighbors graph with self-tuning Gaussian decay [72]. The number of classes was assumed known. As in [37], 11 classes were used, as there are two different ways to write the digit 1. This is the only dataset in which we tuned the timestep parameters by hand, although it only improved the results a small amount, except for the speed of modularity MBO, which was improved by an order of magnitude. 100 tests were performed, except for GenLouvain, which was too slow and could only be tested 24 times. We found this problem particularly difficult, apparently due to the noisy way in which the graph was constructed. For instance, about 25% of the partitions we found had better modularity

		Moons	MNIST	Urban	LFR50k	Plume7	Plume40
	Nodes	2,000	70,000	94,249	50,000	286,720	$1.6 * 10^6$
	Edges	$1.8 * 10^4$	$4.7 * 10^5$	$6.8 * 10^5$	$7.9 * 10^5$	$5.3 * 10^6$	$2.9 * 10^7$
	Communities	2	10	5	2,000	5	5
Modularity	Balanced TV	**0.887**	**.932**	**.944**	.700	.861	**.825**
	Mod. MBO	**0.887**	**.932**	.938	.702	.864	.814
	GenLouvain	**0.887**	.930	.926	**.738**	**.882**	*****
	Ground Truth	0.873	.921	—	.736	—	—
Classification	Balanced TV	98.6	97.3	—	97.4	—	—
	Mod. MBO	98.0	96.8	—	98.3	—	—
	GenLouvain	98.8	97.5	—	96.9	—	—
Iter. Time (sec.)	Balanced TV	**0.05**	**6**	**3**	**57**	**13**	**71**
	Mod. MBO	0.35	18	23	132	96	518
	GenLouvain	0.78	4,860	5,112	112	106,417	*****
Eig. Time (sec.)	Balanced TV	2.0	48	32	—	305	56 min
	Mod. MBO	2.8	52	35	—	303	77 min
Data Corr.		0.87	0.94	—	0.98	—	—

Table 1

Numerical results for algorithms on several data sets. Overall, Balanced TV performed similarly to the other methods but was much faster. Note that the algorithms achieve better modularity than the "ground truth" partition. This indicates that there is a small amount of noise in the way the graph was generated, which makes an exact recovery of the ground truth impossible using modularity alone. The amount of information lost, however, appears to be slight. Timing for the algorithm is split into the eigenvector computation and the main iteration. This is because the eigenvector computation need be done only once per graph, and then the main iteration can be used to quickly sample different high-modularity partitions. The data correlation is simply the correlation between modularity value and classification rate in the partitions recovered. This correlation is high enough to show that modularity is a reasonable model for the ground truth, but low enough to explain the existence of many partitions with slightly better modularity than the ground truth partition. Stars denote tests which were not performed due to excessive expected length, and dashes denote measurements that do not make sense for the given dataset, either because nodes did not come pre-labelled or recursive partitioning was used.

than the ground truth partition, but partitions with a classification accuracy greater than 95% were found only about 4% of the time. This shows how the graph combined with the modularity model did not encode the ground truth perfectly. On the other hand, almost all partitions with modularity better than the ground truth partition also matched the ground truth partition at least 90%, so the issues with the model only seem relevant if very high accuracy is needed.

LFR 50k This is an artificial social network consisting of a ring of 50 1,000-node LFR networks connected together with a small number of random connections [45, 37]. There are about 2,000 communities in these networks. We used the following parameters: average degree of 20, maximum degree of 50, degree distribution exponent of 2, community size distribution exponent of 1, effective mixing parameter of 0.2, maximum community size of 50, minimum community size of 10. Fifteen tests were performed for each method, each on a different random network. We used recursive partitioning to determine the number of communities.

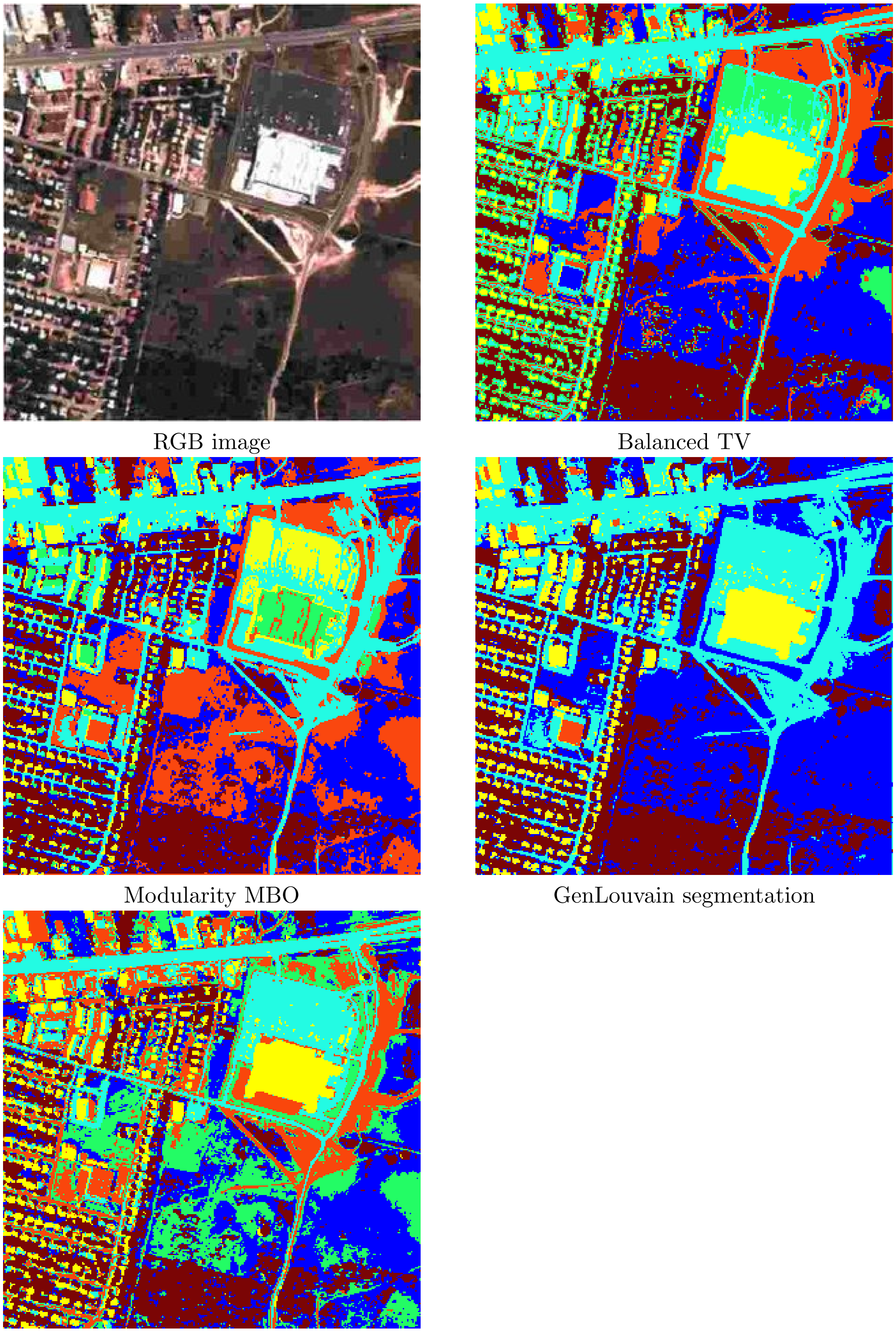
This is the only example in which it is clear that Balanced TV does not get near enough to the global minimizer, and the reason is not totally clear. We do note, however, that when the same tests are performed on a single 1,000-node LFR network, Balanced TV attains better than 99 percent accuracy, even with much higher effective mixing parameter, which suggests that some aspect of the ring structure or greater scale in the LFR50 test makes it more difficult. Another interesting fact is that while Balanced TV and Modularity MBO obtain essentially equal modularity scores, their classification rate is not the same, showing that there can exist problem-specific biases in the minimizers likely to be found by different solvers. These biases are essentially accidents, since both solvers are designed to look for local minima of the modularity functional.

Urban Image The urban hyperspectral image is a 307×307 image of an urban setting, where each pixel encodes the intensity of light at 129 different wavelengths. The classification problem in this setting is to identify the pixels that contain similar materials, such as dirt, road, grass, etc. In past works, authors have tried to identify a small number of classes in this image, see e.g. [74], and we follow that precedent in our own presentation, noting that the image can also be naturally split into many more classes, as there are many materials present in the picture. In fact, this makes the segmentation task very difficult, as large clusters in the graph are really an agglomeration of smaller clusters corresponding to each substance.

The graph representation was computed using “nonlocal means” [9]. This means that for each pixel p, a vector v_p was constructed by concatenating the data in a 3×3 window centered at p. One then uses the weighted cosine distance on these $3 \times 3 \times 162 = 1,458$ component vectors, where the components from the center of the window are given the most weight. For each pixel, we obtained the 10 nearest neighbors in this distance using a k-d tree and the VLFeat software package [65].

We assumed either 5 or 6 communities were present for our tests. We performed 200 tests with the MBO-based methods, giving a large variety of segmentations. Almost all of these were reasonable but corresponded to different interpretations of the scene. As an example of the different interpretations of the scene, observe that in Figure 1, some of the images consider all houses to be of the same class, but others distinguish the brown and white roofs, which seem to be made of different materials. The diversity of outcomes suggests that the results of several segmentations may be combined for more robust insights, but we do not pursue this further in this work. The images selected for inclusion in Figure 1 are those that are most similar to the existing literature. We also performed 10 tests with GenLouvain, of which 9 gave identical outcomes. Finally, we compared with a recent NLTV-based algorithm [74], which is specifically designed for hyperspectral imaging applications.

Overall, we found that Balanced TV was very competitive, with some clear advantages. We cannot remark on every difference between the images in Figure 1 here, but we can point out some of the more striking ones. Comparing with Modularity MBO and NLTV, Balanced TV does a much better job of placing all of the grass into a single class. Comparing with GenLouvain, we see that Balanced TV correctly resolved the difference between pavement and dirt, which is not well-represented in the GenLouvain segmentation. Indeed, looking at the roads in the upper right, we see that Balanced TV gives the sharpest resolution of the roads and the surrounding dirt sections. On the other hand, Balanced TV does have a little trouble compared to GenLouvain when resolving the buildings just below the large road in the upper left corner of the picture, although this is partly due to the fact that the roofs there are clearly made of different materials from most of the houses further down in the image, and NLTV has a similar problem.



RGB image

Balanced TV

Modularity MBO

GenLouvain segmentation

NLTV segmentation [74]

FIGURE 1. *The urban dataset segmented using different methods. Balanced TV does well when separating dirt from roads, resolving the roads in the upper right corner, and placing all of the grass into a single class. It has some difficulty with the buildings in the upper left corner, just below the main road, which are a different material from the other buildings. See the main text for more detailed analysis.*

Plume Hyperspectral Video The gas plume hyperspectral video records a gas plume being released at the Dugway Proving Ground [27, 47, 50].[7] The graph was constructed by the same procedure as the urban dataset, simply concatenating each frame side-by-side into one large image and using nonlocal means to form the graph. Each frame has 320 × 128 pixels with data from 129 wavelengths. Two versions of this dataset were used, one with 7 frames, and another with 40 frames. We have included the segmentation of one frame in Figure 2, together with segmentations produced by competing algorithms. Thirty tests were performed for the 7-frame video, once again generating a variety of interpretations of the scene, most of which were reasonable. We show the results most similar to existing approaches in Figure 2.

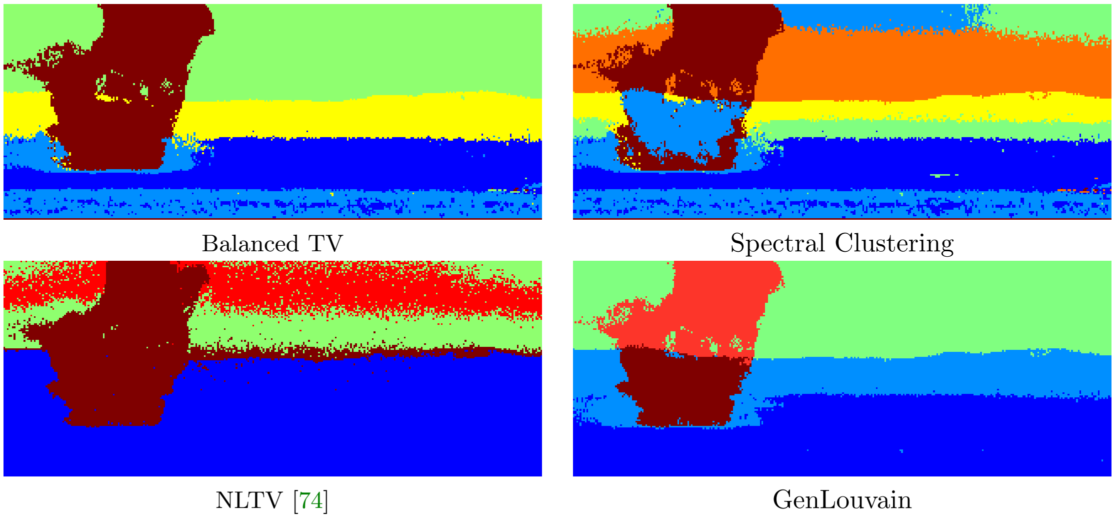


FIGURE 2. *Segmentations of the plume hyperspectral video using different methods. Observe that Balanced TV is the only method that gets the whole plume into a single class without any erroneous additions.*

It seems that Balanced TV gives the best results, since it is the only approach to keep the plume in a single group, while also capturing the details on the edges well. It also has no artificial divisions in the sky, as the other methods do. All of the methods separate the ground into multiple classes, although NLTV nearly avoids this issue (at the expense of making the plume less clear). In general, if the application is to identify the plume, Balanced TV should be preferred.

6. Conclusion. We have shown that modularity optimization can be framed as a balanced TV problem that is convex except for a discrete constraint. This formulation yields an energy landscape is easier to understand by using terms with a ready intuitive meaning and by putting all of the nonconvexity into a simple discrete constraint. We have given a rigorous nonconvexity result and shown how to use the Ginzburg-Landau functional to approximate modularity optimization by more convex problems. We have also proposed an improved modularity optimization scheme, Balanced TV, which works very well even on large graphs and which requires much less hand-tuning. Numerical tests show that our method is competitive in terms of accuracy, while being faster than the other methods tested.

[7] In [50], and semi-supervised MBO-type approach was used. The balance terms in modularity provide a way to avoid trivial minimizers without supervision.

A. Gamma convergence.

The following are some basic facts about Gamma-convergence to aid in understanding the results of this paper. See [63] for more details.

Definition A.1. *Let X be a topological space, and $\mathcal{F}_n$ a sequence of real-valued functionals of X. Then the sequence is said to Γ-converge to a functional $\mathcal{F}$ on X if the following two conditions hold:*

1. *For convergent sequence $x_n \to x$, we have $\liminf_{n\to\infty} \mathcal{F}_n(x_n) \le \mathcal{F}(x)$.*
2. *For every x, there exists a convergent sequence $x_n \to x$ such that $\limsup_{n\to\infty} \mathcal{F}_n(x_n) \ge F(x)$.*

For our purposes, Γ-convergence is primarily a tool for ensuring that the minimizers of $\mathcal{F}_n$ approach the minimizers of $\mathcal{F}$, as guaranteed by the following:

Theorem A.2. *Let $\mathcal{F}_n$ Γ-converge to $\mathcal{F}$, and let x_n be a minimizer of $\mathcal{F}_n$. Then every cluster point of the x_n is a minimizer of $\mathcal{F}$. If $\mathcal{G}$ is continuous, then $\mathcal{F}_n + \mathcal{G}$ Γ-converges to $\mathcal{F} + \mathcal{G}$.*

We end with the proof of 3.4.

Proof. From [63], we know that the graph Ginzburg-Landau functional Γ-converges to the graph TV functional, and since the added term $(k^T u)^2 + (k^T(1-u))^2$ is continuous, the Γ-convergence is retained when this term is added to both the graph TV and graph Ginzburg-Landau functionals. The subsequence condition then arises as a standard result in Γ-convergence theory and the compactness of bounded domains in $\mathbb{R}^N$. □

B. Deferred proofs.

In this section, we prove in detail the bounds presented earlier regarding the timestep selection in the MBO scheme.

Proof of Proposition 4.1. We first get pointwise estimates on $u - u_0$:

$$||u-u_0||_\infty \le ||e^{-\tau M} - I||_\infty ||u_0||_\infty = ||e^{-\tau M} - I||_\infty \le \sum_{k=1}^{\infty} \frac{1}{k!}\tau^k ||M||_\infty^k = e^{\tau ||M||_\infty} - 1 \tag{26}$$

We estimate $||M||_\infty$ as follows:

$$\begin{aligned} ||M||_\infty &= \max_i \sum_j |L_{ij} + \frac{\gamma}{m} k_i k_j| = \max_i \sum_j |k_i \delta_{ij} - w_{ij} + \frac{\gamma}{m} k_i k_j| \\ &\le \max_i k_i + k_i + \frac{\gamma}{m} k_i 2m = 2(1+\gamma) k_{\max} \end{aligned}$$

These computations do not depend on n̂, but in order to get a timestep, we assume that n̂ = 2. In this case, let u^1 and u^2 be the columns of u. We have $u^1_t = -Mu^1$ and $u^2_t = -Mu^2$. Subtracting these, and letting $v = u^1 - u^2$ yields $v_t = -Mv$. Allowing v to evolve until the time of thresholding, we see that node i will switch classes if and only if $v(i)$ has changed sign, that is if $|v - v_0|_i > 1$. The quantity in (26) is less than 1 exactly when $\tau < \frac{\log 2}{2(\gamma+1)k_{\max}} \approx \frac{0.15}{(\gamma+1)k_{\max}}$. This is exactly the bound we sought.

Next, we work on the L^2 bound

$$||u-u_0||_\infty \le \sqrt{\hat{n}}||u-u_0||_2 \le \sqrt{\hat{n}}||e^{-\tau M} - I||_2 ||u_0||_2 \le \sqrt{\hat{n}}||u_0||_2 \sum_{k=1}^{\infty} \frac{1}{k!}\tau^k ||M||_2^k \tag{27}$$

$$= \sqrt{\hat{n}}||u_0||_2 \left(e^{\tau ||M||_2} - 1\right) = \sqrt{\hat{n}}||u_0||_2 \left(e^{\tau\rho} - 1\right) \tag{28}$$

As before, when we let $\hat{n} = 2$, one can subtract the columns to get v, so that no node will switch communities as long as $||v - v_0||_\infty < 1$, which is guaranteed if $\tau < \rho^{-1} \log\left(1 + N^{-\frac{1}{2}}\right)$. □

Proof of Proposition 4.2. To get the bound, we let Λ be a diagonal matrix with the eigenvalues of M on the diagonal. Since M is positive semidefinite, we can write $M = Q\Lambda Q^T$ for some orthogonal matrix Q. Then we have

$$||u(\tau)||_2 = ||e^{-\tau M} u_0||_2 \leq ||e^{-\tau M}||_2 ||u_0||_2 = ||e^{-\tau\Lambda}||_2 ||u_0||_2 = e^{-\tau\lambda_1}||u_0||_2$$

Setting the latter quantity less than ϵ and then solving for τ yields the required bound. □

REFERENCES

[1] G. Agarwal and D. Kempe, *Modularity-maximizing graph communities via mathematica programming*, European Physics Journal B, 66/3 (2008).

[2] L. Ambrodio and V. Tortorelli, *On the approximation of free discontinuity problems*, Boll. Un. Mat. Ital. B, 7 (1992), pp. 105–123.

[3] C. Anderson, *A Rayleigh-Chebyshev procedure for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices*, J. Comput. Phys., 229 (2010), pp. 7477–7487.

[4] A. Arenas, A. Fernández, and S. Gómez, *Analysis of the structure of complex networks at different resolution levels*, New J. Phys., 10 (2008), p. 053039.

[5] D. Bassett, E. Owens, M. Porter, M. Manning, and K. Daniels, *Extraction of force-chain network architecture in granular materials using community detection*, Soft Matter, (2015).

[6] 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.

[7] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, *Fast unfolding of communities in large networks*, J. Stat. Mech. Theory Exp., 2008 (2008), p. P10008.

[8] U. Brandes, D. Delling, M. Gaertler, D. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, *On modularity clustering*, IEEE Trans. on Knowledge Data Engrg., 20 (2008), pp. 172–188.

[9] A. Buades, B. Coll, and J. M. Morel, *A non-local algorithm for image denoising*, in CVPR, vol. 2, June 2005, pp. 60–65.

[10] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. on Inform. Theory, 52 (2006), pp. 489–509.

[11] A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock, *An introduction to total variation for image analysis*, Theoretical Foundations and Numerical Methods for Sparse Recovery, 9 (2010), p. 227.

[12] A. Chambolle and J. Darbon, *On total variation minimization and surface evolution using parametric maximum flows*, Int. J. Comput. Vis., 84 (2009), pp. 288–307.

[13] E. Y. Chan and D.-Y. Yeung, *A convex formulation of modularity maximization for community detection*, in Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Spain, 2011, p. 2218.

[14] T. F. Chan and L. A. Vese, *Active contours without edges*, IEEE Trans. on Image Process., (2001).

[15] Y. Chen, X. Li, and J. Xu, *Convexified modularity for degree-corrected stochastic block models*, CORR, abs/1512.08425 (2015).

[16] F. Chung, *Spectral Graph Theory*, AMS, 1992.

[17] G. Csardi and T. Nepusz, *The igraph software package for complex network research*, InterJournal, Complex Systems (2006), p. 1695, http://igraph.org.

[18] J. Darbon and M. Sigelle, *A fast and exact algorithm for total variation minimization*, in Iberian Conference on Pattern Recognition and Image Analysis, Springer Berlin Heidelberg, 2005, pp. 351–359.

[19] E. Davis and S. Sethuraman, *Consistency of modularity clustering on random geometric graphs*, CORR, abs/1604.03993v1 (2016).

[20] S. Esedoglu and Y.-H. R. Tsai, *Threshold dynamics for the piecewise constant mumford-shah functional*, J. Comput. Phys., 211 (2006), pp. 367–384.

[21] L. Evans, *Convergence of an algorithm for mean curvature motion*, Indiana Univ. Math J., 42 (1993), pp. 553–557.

[22] S. Fortunato, *Community detection in graphs*, Phys. Rep., 486 (2010), pp. 75–174.

[23] S. Fortunato and M. Barthélemy, *Resolution limit in community detection*, Proc. Natl. Acad. Sci., 104 (2007), pp. 36–41.

[24] S. Fortunato and D. Hric, *Community detection in networks: A user guide*, Phys. Rep., 659 (2016), pp. 1 – 44.

[25] H. Gao, M. Guo, R. Li, and L. Xing, *Graphics Processing Unit-Based High Performance Computing in Radiation Therapy*, CRC Press, 2015, ch. 4DCT and 4D Cone-Beam CT Reconstruction Using Temporal Regularization, pp. 63–82.

[26] C. Garcia-Cardona, E. Merkurjev, A. L. Bertozzi, A. Percus, and A. Flenner, *Multiclass segmentation using the Ginzburg-Landau functional and the MBO scheme*, IEEE Trans. Pattern Anal. Mach. Intell., 36 (2014), pp. 1600–1614.

[27] T. Gerhart, J. Sunu, L. Lieu, E. Merkurjev, J.-M. Chang, J. Gilles, and A. L. Bertozzi, *Detection and tracking of gas plumes in LWIR hyperspectral video sequence data*, in SPIE Defense, Security, and Sensing, International Society for Optics and Photonics, 2013, pp. 87430J–87430J.

[28] G. Gilboa, *A total variation spectral framework for scale and texture analysis*, SIAM J. Imag. Sci., 7 (2014), pp. 1937–1961.

[29] G. Gilboa and S. Osher, *Nonlocal operators with applications to image processing*, Multiscale Model. Simul., 7 (2008), pp. 1005–1028.

[30] R. Glowinski, T.-W. Pan, and X.-C. Tai, *Some Facts About Operator-Splitting and Alternating Direction Methods*, Springer International Publishing, Cham, 2016, pp. 19–94.

[31] D. Goldfarb and W. Yin, *Second-order cone programming methods for total variation-based image restoration*, SIAM J. Sci. Comput., 27 (2005), pp. 622–645.

[32] T. Goldstein and S. Osher, *The Split Bregman method for L1-regularized problems*, SIAM J. Img. Sci., 2 (2009), pp. 323–343.

[33] B. H. Good, Y.-A. de Montjoye, and A. Clauset, *Performance of modularity maximization in practical contexts*, Phys. Rev. E, 81 (2010), p. 046106.

[34] R. Guimerè, M. Sales-Pardo, and L. A. N. Amaral, *Modularity from fluxuations in random graphs and complex networks*, Phys. Rev. E, 70 (2004), p. 025101.

[35] A. Harten, *High resulution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.

[36] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. Murray, *From molecular to modular cell biology*, Nature, 402 (1999).

[37] H. Hu, T. Laurent, M. A. Porter, and A. L. Bertozzi, *A method based on total variation for network modularity optimization using the MBO scheme*, SIAM J. Appl. Math., 73 (2013), pp. 2224–2246.

[38] L. G. S. Jeub, P. Balachandran, M. A. Porter, P. J. Mucha, and M. W. Mahoney, *Think locally, act locally: Detection of small, medium-sized, and large communities in large networks*, Phys. Rev. E, 91 (2015), p. 012821.

[39] I. S. Jutla, L. G. S. Jeub, and P. J. Mucha, *A generalized Louvain method for community detectio implemented in MATLAB*. http://netwiki.amath.unc.edu/GenLouvain (2011-2014).

[40] B. W. Kernighan and S. Lin, *An efficient heuristic procedure for partitioning graphs*, Bell System Technical J., 49 (1970), pp. 291–307.

[41] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, *An interior-point method for large-scale l1-regularized least squares*, IEEE J. Selected Topics in Signal Processing, 1 (2007), pp. 606–617.

[42] R. Kohn and P. Sternberg, *Local minimisers and singular perturbations*, Proc. Roy. Soc. Edinburgh Sect. A, (1989).

[43] V. Kolmogorov, Y. Boykov, and C. Rother, *Applications of parametric maxflow in computer vision*, in 2007 IEEE 11th International Conference on Computer Vision, IEEE, 2007, pp. 1–8.

[44] A. Lancichinetti and S. Fortunato, *Limits of modularity maximization in community detection*, Phys. Rev. E, 84 (2011), p. 066122.

[45] A. Lancichinetti, S. Fortunato, and F. Radicchi, *Benchmark graphs for testing community detection algorithms*, Phys. Rev. E., 78 (2008), p. 056117.

[46] Y. LeCun and C. Cortes, *The MNIST database of handwritten digits*, 1998.

[47] D. Manolakis, C. Siracusa, and G. Shaw, *Adaptive matched subspace detectors for hyperspectral imaging applications*, in 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221), vol. 5, 2001, pp. 3153–3156 vol.5.

[48] Z. Meng, A. Koniges, Y. H. He, S. Williams, T. Kurth, B. Cook, J. Deslippe, and A. L. Bertozzi, *OpenMP parallelization and optimization of graph-based machine learning algorithms*, in International Workshop on OpenMP (IWOMP), 2016.

[49] E. Merkujev, T. Kostic, and A. Bertozzi, *MBO scheme on graphs for segmentation and image processing*,

SIAM J. Imag. Sci., 6 (2013), pp. 1903–1930.

[50] E. Merkurjev, J. Sunu, and A. L. Bertozzi, *Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video*, in IEEE International Conference on Image Processing, 2014.

[51] B. Merriman, J. Bence, and S. Osher, *Diffusion generated motion by mean curvature*, Proc. Comput. Crystal Growers Workshop, (1992), pp. 73–83.

[52] P. J. Mucha, T. Richardson, K. Macon, M. Porter, and J.-P. Onnela, *Community structure in time-dependent, multiscale, and multiplex networks*, Science, 328 (2010), pp. 876–878.

[53] M. Newman, *Networks: an Introduction*, 2010.

[54] M. E. Newman, *Modularity and community structure in networks*, Proc. Nat. Acad. Sci., 103 (2006), pp. 8577–8582.

[55] M. E. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E, 69 (2004), p. 026113.

[56] M. E. J. Newman, *Equivalence between modularity optimization and maximum likelihood methods for community detection*, Phys. Rev. E, 94 (2016), p. 052315.

[57] O. A. Oleinik, *Discontinuous solutions of nonlinear differential equations*, Uspekhi Mat. Nauk, 12 (1957), pp. 3–73.

[58] L. Peel, D. B. Larremore, and A. Clauset, *The ground truth about metadata and community detection in networks*, Science Advances, 3 (2017).

[59] M. A. Porter, J.-P. Onnela, and P. J. Mucha, *Communities in networks*, Notic. Amer. Math. Soc., 56 (2009), pp. 1082–1097.

[60] J. Reichardt and S. Bornholdt, *Statistical mechanics of community detection*, Phys. Rev. E, 74 (2006), p. 016110.

[61] L. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation noise removal algorithm*, Phys. D, 60 (1992), pp. 259–268.

[62] H.-W. Shen, X.-Q. Cheng, and J.-F. Guo, *Quantifying and identifying the overlapping community structure in networks*, J. Stat. Mech. Theory Exp., 2009 (2009), p. P07042.

[63] Y. van Gennip and A. L. Bertozzi, *Gamma-convergence of graph Ginzburg-Landau functionals*, Adv. Differential Equations, 17 (2012), pp. 1115–1180.

[64] Y. van Gennip, N. Guillen, B. Osting, and A. L. Bertozzi, *Mean curvature, threshold dynamics, and phase field theory on finite graphs*, Milan J. Math., 82 (2014), pp. 3–65.

[65] A. Vedaldi and B. Fulkerson, *VLFeat: An open and portable library of computer vision algorithms*. http://www.vlfeat.org/, 2008.

[66] B. P. Vollmayr-Lee and A. D. Rutenberg, *Fast and accurate coarsening simulation with an unconditionally stable time step*, Phys. Rev. E, 68 (2003), p. 066703.

[67] U. von Luxborg, *A tutorial on spectral clustering*, Statist. Comput., 17 (2007), pp. 395–416.

[68] Y. Wang, J. Yang, W. Yin, and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM J. Imag. Sci., 1 (2008), pp. 248–272.

[69] J. Yang, Y. Zhang, and W. Yin, *A fast alternating direction method for TVL1-L2 signal reconstruction from partial Fourier data*, Selected Topics in Signal Processing, IEEE J., Special Issue on Compressed Sensing, 4 (2010), pp. 288–297.

[70] J. Yuan, E. Bae, and X.-C. Tai, *A study on continuous max-flow and min-cut approaches*, in CVPR, IEEE, 2010, pp. 2217–2224.

[71] W. W. Zachary, *An information flow model for conflict and fission in small groups*, J. Anthropological Res., 33 (1977), pp. 452–473.

[72] L. Zelnik-Manor and P. Perona, *Self-tuning spectral clustering*, NIPS, 17 (2004), p. 16.

[73] P. Zhang and C. Moore, *Scalable detection of statistically significant communities and hierarchies, using message passing for modularity*, Proc. of the Nat. Acad. Sci., 111 (2014), pp. 18144–18149.

[74] W. Zhu, V. Chayes, A. Tiard, S. Sanchez, D. Dahlberg, A. L. Bertozzi, S. Osher, D. Zosso, and D. Kuang, *Unsupervised classification in hyperspectral imagery with nonlocal total variation and primal-dual hybrid gradient algorithm*, IEEE Transactions on Geoscience and Remote Sensing, PP (2017), pp. 1–13.