SIMPLIFIED ENERGY LANDSCAPE FOR MODULARITY USING TOTAL VARIATION*

ZACHARY M. 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. We therefore, find a manageable non-convex approximation using a Ginzburg Landau functional, which provably converges to the correct energy in the limit of a certain parameter. We then derive an MBO algorithm with fewer hand-tuned 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 [55], molecular biology [38], politics [55], material science [5], and many more [56]. There is a large and growing literature on the subject, with many competing definitions of community and associated algorithms [24, 64, 26]. 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 [40].

A great variety of definitions have been proposed to make the partitioning task precise [26], 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 [58], which is the most well-studied of existing methods. To define it, we need the following terminology, which is used throughout the paper.

†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
¶Department of Mathematics, Hong Kong Baptist University, Kowloon Tong Kowloon, Hong Kong (tai@math.uib.no)
‖Department of Mathematics, UCLA, Los Angeles, CA (bertozzi@math.ucla.edu).
*Submitted to the editors DATE.

DEFINITION 1.1. *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 partition $A_1, \ldots, A_{\hat{\mathrm{n}}}$ of the nodes of G which maximizes

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

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 [36]. In addition, many dissimilar partitions may yield near-optimal modularity values [35]. 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 [63]. 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 [78]. Another approach is simply to sample many high-modularity partitions, expecting that multiple intuitively-meaningful partitions may be found. Such effects have been observed, for instance, in the Zachary Karate Club network, which has both a community structure and leader-follower structure [63].

Modularity also has preferred scale for communities [25, 46]. For this reason, one typically includes a resolution parameter $\gamma > 0$ [65, 4], yielding

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

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 [24, 26], with prominent approaches including spectral [57, 59], simulated annealing [36], and greedy or Louvain algorithms [7]. It can also be interpreted in terms of force-directed layout and optimized using visualization techniques [61]. The modularity 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 [39] 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 [39]. Specifically, we make the following contributions:

We start with derivations of four formulations of modularity, two in terms of TV and two in terms of graph cuts, which inspire the subsequent analysis. In addition to being intuitively simple, these formulas place all of the nonconvexity of the problem into a discrete constraint—the functionals themselves are convex. We prove 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 then provide an alternative relaxation, using the Ginzburg-Landau functional, that smooths the discrete constraint so that it becomes manageable. We end the theory section by showing that solutions of our relaxed problem converge to maximizers of modularity in an appropriate sense.

Based on these ideas, and following [39], we develop an MBO-type scheme, Balanced TV, which quickly and accurately optimizes modularity in several examples. This algorithm seems especially well-suited to similarity networks from machine learning, where prior knowledge of the number of communities is available and the number of such communities tends to be modest. Using the convexity of our formulation of TV, we provide inner- and outer-loop timestep bounds to avoid hand-tuning parameters, as is necessary in [39]. 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 [39]. Finally, we show that this approach can solve semi-supervised problems as well.

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: Continuum and Discrete. 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 [66], is to modify the problem as follows

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

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*

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

smooth. Total variation minimization has other applications as well, such as compressed sensing [10] and mean curvature flow [12, 45].

Network community detection is in some ways analogous to image segmentation, in that both seek a partition into coherent subsets, and one of the main ideas behind the use of total variation in the network context is that it helps us arrive at the "correct" energy to optimize for, as in the image processing context. An important example is spectral approaches, such at those of [57, 59]. In the case of only two communities, we let u be a real-valued function on the nodes of the graph. A partition of the nodes into two communities can be encoded in such a function by letting $u = 1$ on the nodes in one set and $u = -1$ on the others. The modularity can then be written as

$$\frac{1}{4m}\sum_{ij}\left(w_{ij}-\gamma\frac{k_ik_j}{2m}\right)(1+u_iu_j) = \frac{1}{4m}\sum_{ij}\left(w_{ij}-\gamma\frac{k_ik_j}{2m}\right)+\frac{1}{4m}\sum_{ij}\left(w_{ij}-\gamma\frac{k_ik_j}{2m}\right)u_iu_j \tag{1}$$

$$= \text{const}+\frac{1}{4m}\sum_{ij}\left(w_{ij}-\gamma\frac{k_ik_j}{2m}\right)u_iu_j \tag{2}$$

$$= \text{const}+\frac{1}{2}u^TMu \tag{3}$$

where $M_{ij} = w_{ij} - \gamma\frac{k_ik_j}{2m}$ is the *modularity matrix*. Thus, (3) is exactly equal to modularity when u represents a partition but has an obvious extension to all N-vectors. An important idea in spectral approaches is to maximize (3) or related energies over all real vectors and then employ some kind of thresholding on the values of the result to recover a binary partition. Recursive bipartitioning can be used to find partitions into more than two communities. A large number of variations on this idea exist and are widely used. Such approaches are analogous to ideas from section 2, in that the solutions are expected to be smooth because of the quadratic term, which is indeed observed in practice, thus necessitating some kind of thresholding. In contrast, by using a non-quadratic measure of differences in the value of u across edges, it is possible to promote sharp interfaces in the solutions.

We now very briefly give the definition of total variation on a graph, referring to [31, 69] for a complete treatment, where it is shown that these choices are consistent with discrete notions of Riemannian metrics, inner products, divergences, and so forth. The nonlocal gradient of a function $f : G \to \mathbb{R}$ at node i in the direction of the edge from i to j is

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

The *graph total variation* is then given by the 1-norm of ∇f at node i

$$|f|_{TV} = \frac{1}{2}\sum_{ij} w_{ij}|f(j) - f(i)|, \tag{4}$$

where w_{ij} is the i, j entry of the adjacency matrix (see Definition 1.1). We will actually use a slight generalization of (4) to the case where $f : \{1, \cdots, N\} \to \mathbb{R}^{\hat{n}}$ is vector-valued, in which case

$$|f|_{TV} = \sum_{\ell=1}^{\hat{n}} |f_\ell|_{TV}$$

where f_ℓ is the ℓ-th component of f. It is usually convenient in this case to identify f with an $N \times \hat{n}$

matrix where $f_{i\ell} = f_\ell(i)$. Then we have

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

Graph total variation is connected to graph cuts, which correspond roughly to perimeter in Euclidean space.

DEFINITION 2.1. *Let S be a subset of the nodes of G. Then the graph cut associated to S is given by*

$$\mathrm{Cut}(S, S^c) = \sum_{i \in S, j \in S^c} w_{ij}.$$

Let $f : \{1, \cdots, N\} \to \mathbb{R}$ be the characteristic function of a set of nodes S. Then we can calculate

$$|f|_{TV} = \frac{1}{2} \sum_{ij} w_{ij} |f(i) - f(j)| = \sum_{i \in S, j \in S^c} w_{ij} = \mathrm{Cut}(S, S^c). \tag{5}$$

TV minimization on a graph tends to produce piecewise-constant functions whose corresponding graph cut is small [52].

3. Equivalence Theorem and its Consequences. In this section, we derive representations of modularity and explore some consequences. We will need definitions:

DEFINITION 3.1. *A family of sets $S_1, \cdots, S_{\hat{\mathrm{n}}}$ is a partition of a set S if $S = \bigcup_{\ell=1}^{\hat{\mathrm{n}}} S_\ell$ and $S_{\ell_1} \cap S_{\ell_2}$ is empty for each $\ell_1 \neq \ell_2$.*

DEFINITION 3.2. *Let $\Pi(G)$ be the set of all partitions of the nodes of G. For each partition $A_1, \cdots, A_{\hat{\mathrm{n}}}$ in $\Pi(G)$, there is an $N \times \hat{\mathrm{n}}$* partition matrix *defined by,*

$$u_{i\ell} = \begin{cases} 1 & i \in A_\ell \\ 0 & i \in A_\ell^c \end{cases}$$

For a matrix u, we say $u \in \Pi(G)$ when u is the partition matrix of some partition.

DEFINITION 3.3. *For any subset S of the nodes of G, its* volume *is given by* $\mathrm{vol}\, S = \sum_{i \in S} k_i$.

3.1. Formulations of Modularity on Terms of TV and Graph Cuts. We are now ready to give the different formulations of modularity that form the basis for our subsequent analysis.

PROPOSITION 3.4 (Equivalent forms of modularity). *The following optimization problems all have the some solution set:*

$$\text{(6)} \quad \textit{Modularity:} \quad \underset{\hat{\mathrm{n}} \in \mathbb{N}, \{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}} \in \Pi(G)}{\mathrm{argmax}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} w_{ij} - \gamma \frac{k_i k_j}{2m}$$

$$\text{(7)} \quad \textit{Balanced cut (I):} \quad \underset{\hat{\mathrm{n}} \in \mathbb{N}, \{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}} \in \Pi(G)}{\mathrm{argmin}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left(\mathrm{vol}\, A_\ell\right)^2 \right)$$

$$\text{(8)} \quad \textit{Balanced cut (II):} \quad \underset{\hat{\mathrm{n}} \in \mathbb{N}, \{A_\ell\}_{\ell=1}^{\hat{\mathrm{n}}} \in \Pi(G)}{\mathrm{argmin}} \quad \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left(\mathrm{vol}\, A_\ell - \frac{2m}{\hat{\mathrm{n}}} \right)^2 \right) + \gamma \frac{2m}{\hat{\mathrm{n}}}$$

$$\text{(9)}\quad \textit{Balanced TV (I):} \qquad \operatorname*{argmin}_{\hat{\mathrm{n}}\in\mathbb{N},u\in\Pi(G)} \quad |u|_{TV} + \frac{\gamma}{2m}\left|\left|k^T u\right|\right|_2^2$$

$$\text{(10)}\quad \textit{Balanced TV (II):} \qquad \operatorname*{argmin}_{\hat{\mathrm{n}}\in\mathbb{N},u\in\Pi(G)} \quad |u|_{TV} + \frac{\gamma}{2m}\left|\left|k^T u - \frac{2m}{\hat{\mathrm{n}}}\right|\right|_2^2 + \gamma\frac{2m}{\hat{\mathrm{n}}}$$

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 easier to understand, as it is minimized (for fixed $\hat{\mathrm{n}}$) when each community has volume $2m/\hat{\mathrm{n}}$. Furthermore, the third term of the forms labelled II reveals that the incentive to increase the number of communities $\hat{\mathrm{n}}$ can be quantified in terms of an $O(\hat{\mathrm{n}}^{-1})$ penalty term, which is not obvious from other formulations of modularity.

One can compare these equivalent formulations with [39], in which minimizing the functional

$$\text{(11)}\qquad |u|_{TV} - \gamma||u - \mathrm{mean}(u)||^2_{\ell^2(G)} = |u|_{TV} - \gamma\sum_{i\ell} k_i\left|u_{i\ell} - \frac{1}{2m}\sum_{i'=1}^{N} k_i u_{i'\ell}\right|^2$$

is shown to be equivalent to modularity optimization, subject to the same constraint as the other TV formulas presented here. Thus, in [39], 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.[2] It is also clearer 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, via the second and third terms of (10). Overall, these theoretical simplifications make the nonconvexity of the problem easier to navigate.

We note that forms similar to (7)–(10) have appeared in the literature before (see e.g. [65]), although the only previous work to consider any modularity formula in terms of total variation is [39]. To the best of our knowledge, the composition of modularity into the three intuitively meaningful terms in the forms labelled II is also novel. We will see shortly that the total variation perspective on (7)–(10), combined with the convexity of the functionals in (9) and (10) leads to a number of new developments.

Equations (7)–(10) 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

$$\text{(12)}\qquad |u|_{TV} + \frac{\gamma}{2m}||k^T u||_2^2 + \lambda||\chi * (u - f)||_2^2$$

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

[2] To see rigorously that (11) is nonconvex, consider the special case of two nodes connected by a single edge, $\gamma = 1$ and $u = [\lambda\ 0; 0\ 0]$. Then considering (11) as a function of λ immediately shows the nonconvexity. The nonconvexity is actually very general; computing the second derivative of the second term in (11) with respect to any component of u gives a negative value for any connected graph with more than one node. Since the TV term grows asymptotically linearly, it is eventually dominated by the quadratic growth of the second, concave term.

metadata should always be done with care, of course, but the general utility of semisupervised learning is well-attested in image processing and machine learning applications. (See Table 3 for two numerical examples.)

Proof of Proposition 3.4. 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 (6) with (7) and the equivalence of (7) and (8). We first show the equivalence of (6) with (7). Fix $\hat{\mathrm{n}}$, and consider an otherwise arbitrary partition $\{A_1, \dots, A_{\hat{\mathrm{n}}}\}$ of G. Then we have

$$Q = \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} w_{ij} - \gamma \frac{k_i k_j}{2m} \tag{13}$$

$$= \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\sum_{i \in A_\ell, j \in \{1,\dots,N\}} w_{ij} - \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} \right) - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{14}$$

$$= \frac{1}{2m} \sum_{ij=1}^{N} w_{ij} - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{15}$$

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{i \in A_\ell, j \in A_\ell^c} w_{ij} - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m} \tag{16}$$

$$= 1 - \frac{1}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \mathrm{Cut}(A_\ell, A_\ell^c) - \frac{\gamma}{2m} \sum_{\ell=1}^{\hat{\mathrm{n}}} \sum_{ij \in A_\ell} \frac{k_i k_j}{2m}. \tag{17}$$

Summing along the j index first yields

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

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

Thus, the maxima of modularity coincide with the minima the functional from (7), as required.

To see that (7) and (8) are equivalent, we calculate:

$$\sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left(\operatorname{vol} A_\ell - \frac{2m}{\hat{\mathrm{n}}} \right)^2 \right) \tag{20}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left((\operatorname{vol} A_\ell)^2 - \frac{4m}{\hat{\mathrm{n}}} \operatorname{vol} A_\ell + \frac{4m^2}{\hat{\mathrm{n}}^2} \right) \right) \tag{21}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left(\operatorname{vol} A_\ell \right)^2 \right) - \frac{\gamma}{2m} \frac{8m^2}{\hat{\mathrm{n}}} + \frac{\gamma}{2m} \frac{4m^2}{\hat{\mathrm{n}}} \tag{22}$$

$$= \sum_{\ell=1}^{\hat{\mathrm{n}}} \left(\mathrm{Cut}\left(A_\ell, A_\ell^c\right) + \frac{\gamma}{2m} \left(\operatorname{vol} A_\ell \right)^2 \right) - \gamma \frac{2m}{\hat{\mathrm{n}}} \tag{23}$$

∎

3.2. On convex relaxations. The preceding equivalence theorem makes it very tempting to look for a convex relaxation of (9). Recall that, given two sets, $A \subset B$ where A is discrete and a functional $\mathcal{F} : A \to \mathbb{R}$, a relaxation of $\mathcal{F}$ is any function $\bar{\mathcal{F}} : B \to \mathbb{R}$ such that $\mathcal{F} = \bar{\mathcal{F}}$ on A. A relaxation is called exact in the context of minimization if $\min_{x \in A} \mathcal{F} = \min_{x \in B} \bar{\mathcal{F}}$.[3] Finally, a relaxation is called convex if $\bar{\mathcal{F}}$ is convex.

Modularity (6) and balanced TV (9) are both defined only over a discrete domain, and we would like an extension, or relaxation, of these functions to a larger, continuum domain so that they are easier to work with numerically. Ideally, we could arrive at a convex relaxation and have access to the powerful tools of convex optimization. The formulation in (9) indicates one way to proceed. Using (9), we already have a convex functional except for the domain, so one would hope that the obvious relaxation obtained by using formula (9) on all of $\mathbb{R}^{N \times \hat{\mathrm{n}}}$ would be useful. Unfortunately, the next theorem shows that this obvious relaxation is minimized by the constant matrix and is thus not likely to be useful. In fact, it shows that a large class of other convex relaxations will be uninformative. This will force us to look for nonconvex approaches in the next subsection. Before we state the theorem, we include three more definitions:

DEFINITION 3.5. *The symmetric group on* $\hat{\mathrm{n}}$ *symbols,* $S_{\hat{\mathrm{n}}}$*, is the set of all permutations on* $\{1, \cdots, \hat{\mathrm{n}}\}$*. Each element* $\sigma \in S_{\hat{\mathrm{n}}}$ *acts on a matrix* $u \in \mathbb{R}^{N \times \hat{\mathrm{n}}}$ *with columns* $u_1, \cdots, u_{\hat{\mathrm{n}}}$ *by sending* u *to another matrix,* $\sigma(u)$ *with columns* $u_{\sigma(1)}, \cdots, u_{\sigma(\hat{\mathrm{n}})}$*. If* $u \in \Pi(G)$*, then* $\sigma(u)$ *is the same partition with the labels permuted.*

DEFINITION 3.6. *A map* $\mathcal{F}$ *from some set of matrices to the real numbers is* symmetric *if it is invariant under column permutations, i.e.* $\mathcal{F}(u) = \mathcal{F}(\sigma(u))$ *for all* σ *and* u*.*

The balanced TV functional (9) is symmetric, and most natural relaxations of it are symmetric.

DEFINITION 3.7. *Given a set* S *lying in a vector space* V*, the convex hull is the smallest convex set containing* S*.*

It can be shown that in a finite-dimensional vector space, the convex hull exists and is the intersection of all convex sets containing S. For example, if S is given by three noncolinear points in the plane, the convex hull is a triangle.

We now state and prove our theorem on convex relaxations of modularity.

THEOREM 3.8. *Let* $\mathcal{F}$ *be given by* (9) *with 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.*

If the symmetry requirement is dropped, then $\tilde{u}$ *need not be a global minimizer, but will have an objective value at least as good as any* $u \in \Pi(G, \hat{\mathrm{n}})$*.*

Proof. We consider the symmetric case first. Let u lie in the convex hull of $\Pi(G, \hat{\mathrm{n}})$. We will use the symmetry of $\tilde{\mathcal{F}}$ plus convexity to average all the column permutations of u and get a value of $\tilde{\mathcal{F}}$ at least as low as u gives. Let $\tilde{u} = \frac{1}{\hat{\mathrm{n}}!} \sum_{\sigma \in S_{\hat{\mathrm{n}}}} \sigma(u)$. Then by Jensen's inequality we have

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

Since u was arbitrary, $\tilde{u}$ is a global minimizer.

[3] Analogous notions apply to maximization problems, but we are using (9) rather than (6) for the moment.

Finally, all the columns of $\tilde{u}$ are equal,[4] and thus uninformative. To see this, take any $k, \ell \in \{1, \dots, \hat{\mathrm{n}}\}$. Let τ be the permutation that swaps these two values and leaves all the others fixed. Then any $\sigma \in S_{\hat{\mathrm{n}}}$ can be written uniquely as $\tau \circ \sigma'$, with $\sigma' = \tau \circ \sigma$. (Proof: $\tau \circ \tau$ is the identity, so left-multiply by τ.) Thus the k-th column of $\tilde{u}$ is given by

$$\tilde{u}_k = \frac{1}{\hat{\mathrm{n}}!} \sum_{\sigma \in S_{\hat{\mathrm{n}}}} \sigma(u)_k \tag{24}$$

$$= \frac{1}{\hat{\mathrm{n}}!} \sum_{\sigma' \in S_{\hat{\mathrm{n}}}} \tau \circ \sigma'(u)_k \tag{25}$$

$$= \frac{1}{\hat{\mathrm{n}}!} \sum_{\sigma' \in S_{\hat{\mathrm{n}}}} \sigma'(u)_\ell \qquad \text{(Note the change in subscript!)} \tag{26}$$

$$= \tilde{u}_\ell \tag{27}$$

So all columns of $\tilde{u}$ are equal.

The non-symmetric case is similar, except that u must lie in $\Pi(G, \hat{\mathrm{n}})$ since $\tilde{\mathcal{F}}$ is not known to be symmetric. Therefore, in that case, we can only show that the value of $\tilde{\mathcal{F}}$ at $\tilde{u}$ is at least as good as at any point in $\Pi(G, \hat{\mathrm{n}})$. ∎

This means that modularity cannot be convexly relaxed using this embedding of $\Pi(G, \hat{\mathrm{n}})$ in $\mathbb{R}^{N \times \hat{\mathrm{n}}}$.[5] Thus, our only option to make use of smooth optimization techniques is a non-convex relaxation. In the following subsection, we present one such family of relaxations.

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. [71, 22, 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, which is the set of nonnegative vectors whose components sum to 1. 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[6] result:

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

[4] Incidentally, all of the rows are also equal, since row stochasticity is preserved under column permutation.

[5] 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 non-trivial part of our theorem. Other related works include [13] and [1].

Note that our proof does not rely on many specific properties of modularity, and indeed, a similar theorem holds for any symmetric quality function over a discrete domain.

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

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

$$\int_U ||\nabla u||^2 = \int_U \nabla u \cdot \nabla u = \int_U -\operatorname{div} \nabla u \cdot u = \int_U -\Delta u \cdot u, \tag{28}$$

which suggests that we use a 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 [68], where much of the continuum theory was successfully extended to graphs. Our approach closely mirrors [39], the main difference in this case simply being that our functionals have better convexity properties, which allows for different estimates and improved techniques. We begin with a convergence result.

THEOREM 3.9 (Γ-convergence for the balanced TV problem). *Assume* $\frac{P(u_i)}{||u_i||} \to \infty$ *as* $||u_i|| \to \infty$*, where* u_i *is the* i*-th row of* u*. Then the functionals*[7]

$$\mathcal{F}_\epsilon = ||\nabla u||_2^2 + \frac{1}{\epsilon}\sum_{i=1}^N P(u_i) + \frac{\gamma}{2m}||k^T u||_2^2 \tag{29}$$

$$:= u^T L u + \frac{1}{\epsilon}\sum_{i=1}^N P(u_i) + \frac{\gamma}{2m}||k^T u||_2^2, \tag{30}$$

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

$$\begin{cases} |u|_{TV} + \frac{\gamma}{2m}||k^T u||_2^2 & \text{if } u \text{ corresponds to a partition} \\ +\infty & \text{otherwise} \end{cases} \tag{31}$$

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 modularity, and*
- *any convergent subsequence of the* u_ϵ *converges to a maximizer of modularity.*

The proof is given in the appendices.

Moving forward, we focus on minimizing the relaxed functionals from Theorem 3.9. While using the Ginzburg-Landau functional does introduce a Laplacian into our formulation, we stress that this approach is different from spectral approaches, such as those in [57, 59]—the preceding result on Γ-convergence shows that the real object we are aiming for is TV, which, as discussed in the background section, has very different solutions from quadratic optimization problems. In the results section, we will see numerically that the answers are indeed different from one particular spectral method.

4. Numerical Scheme.

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

[7] Note that due to the discrete setting, there is no epsilon factor preceding the Laplacian term, see [68].

4.1. MBO iteration. We minimize the functional from (30) using an adaptation of the graph MBO scheme. We call our approach Balanced TV. The acronym "MBO" stands for Merriman, Bence, and Osher [54], 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 [51] and [28]. The theoretical study of the algorithm on graphs was initiated in [69]. We sketch the logic of MBO here and refer the reader to [54] 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 $\hat{\mathrm{n}}$)

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

One way to approximate this flow is by operator splitting [32, p.22] with time-step dt and $t_n = n * dt, n = 0, 1, 2, \cdots$. Given u^n one obtains $u^{n+\frac{1}{2}}$ as the solution to

$$\begin{aligned} \frac{d}{dt}u_1 &= -Lu_1 - \frac{\gamma}{m}kk^T u_1, \quad t \in [t_n, t_{n+1}], \\ u_1(t_n) &= u^n, u^{n+1/2} = u_1(t_{n+1}). \end{aligned} \tag{32}$$

Then one gets u^{n+1} by solving

$$\begin{aligned} \frac{d}{dt}u_2 &= -\frac{1}{\epsilon}P'(u_2), \quad t \in [t_n, t_{n+1}], \\ u_2(t_n) &= u^{n+1/2}, u^{n+1} = u_2(t_{n+1}). \end{aligned} \tag{33}$$

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.

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

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 $M = L + \frac{\gamma}{m}kk^T$
  $u^{n+1} = \mathrm{threshold}(u^{n+\frac{1}{2}})$
  $n = n + 1$
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.

We treat the forcing term implicitly, which differs from several recent studies, such as [39, 6, 51]. 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 semi-definite. Therefore, it has an orthonormal basis of eigenvectors, and the evolution we are solving, namely $\frac{d}{dt}u = -Mu$, 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 $\hat{\text{n}}$. In practice, we found three methods of determining the value of $\hat{\text{n}}$:

1. Use domain knowledge—for instance, in two moons, it is known that there are two communities,
2. Try several values of $\hat{\text{n}}$ 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 $\hat{\text{n}}$ 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),[8]
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 $\hat{\text{n}}$ pieces at each level, then the cost is heuristically on the order of

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

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}, $\hat{\text{n}}$, 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 [69], and proofs are deferred to an appendix. First, we consider a lower bound on the timestep:

[8]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.

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

1. $$||u(\tau) - u_0||_\infty \leq e^{2(\gamma+1)k_{\max}\tau}.$$
2. *In the case where $\hat{n} = 2$, 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*
$$||u(\tau) - u_0||_\infty \leq \sqrt{\hat{n}}||u_0||_2 \left(e^{\tau\rho} - 1\right).$$
4. *If $\hat{n} = 2$, 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 $\hat{n} = 2$ in the above, we used the timestep restriction regardless of $\hat{n}$—indeed the authors expect that $\hat{n} = 2$ 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 $\frac{d}{dt}u = -Mu$ 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 $\frac{d}{dt}u = -Mu$ with initial data $u_0 \in \Pi(G, \hat{n})$. 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.[9]

[9] 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.

5.1. Summary. Tables 1 and 2 summarize the results of our Balanced TV algorithm on several examples, mostly drawn from machine learning and image processing problems. [10] We compared our method to the Modularity MBO algorithm from Hu et al. [39], as well as three other well-known algorithms: the Louvain method [7], the hierarchical method of Clauset, Newman, and Moore [17], and a classic spectral recursive bipartitioning method of Newman [57]. Our own method and that of Hu et al. were written in MATLAB except for the eigenvector computations, which use Anderson's Rayleigh-Chebyshev code [3], written in C++ with OpenMP support. The three other methods are slight modifications of igraph's C library implementations [18]. In practice, the difference in programming language may make a difference in speed, although the eigenvalue computation is typically the most time-intensive part of the computation. We chose a single conservative timestep for Modularity MBO rather than hand-tuning for each experiment. Our method and that of Hu et al. use a random starting seed, so we ran those codes 20 times and report the best modularity and classification rate and the median time.

Overall, we found that our method is competitive with the state of the art on these data sets. Our method generally found higher-modularity partitions and had faster run times than either the method of Hu et al. or of Newman.[11] The Louvain method and our method often gave similar modularity scores, although the partitions they uncovered were not necessarily similar. For example, on the MNIST example, our method achieved the better modularity score, but the Louvain partition matched the true labels more closely. On the Plume40 example, the opposite effect occurs, with our method achieving the lower modularity score but finding a partition that is closer to the true labeling of the pixels. Such issues are a manifestation of the well-known degeneracy of the modularity energy [35], where a number of dissimilar partitions can receive similarly high modularity scores. It is also an indication that modularity needs to be complemented with supervision, regularization, biased initialization, or some other device in order to reliably find the partition that is most appropriate for the problem. In Figure 3, we illustrate the effectiveness of including a small amount of supervision with our method. (See (12).)

5.2. Analysis of each experiment. We now describe the individual experiments.

Two Moons Two moons consists of 2,000 points in 100-dimensional space, sampled from two half-circles, with Gaussian noise added, see Figure 1. We constructed a 13-nearest neighbors graph with the edge weights given by a Gaussian law, with locally-determined decay parameters [77]. The number of classes was assumed known, where the class of a point is the half-circle to which it originally belonged.

MNIST MNIST consists of 70,000 28x28-pixel images, each of which contains a single hand-written digit [48]. 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 [77]. The number of classes was assumed known. As in [39], 11 classes were used, as there are two different ways to write the digit 1, with or without the top flag and flat base. This modularity landscape was particularly troublesome, with about 25% of the partitions we found having better modularity than the ground truth partition, despite the fact that

[10] We also performed some brief tests of our method on biological and social networks but found that the results were not as encouraging, apparently due to some structural differences from our machine learning networks—it would be interesting to understand this issue more.

[11] We chose this particular spectral method because it was available in igraph. A complete comparison with other spectral methods would be interesting but is beyond the scope of this paper.

		Moons	MNIST	LFR50k	Urban	Plume7	Plume40
	Nodes	2,000	70,000	50,000	94,249	286,720	$1.6 * 10^6$
	Edges	$1.8 * 10^4$	$4.7 * 10^5$	$7.9 * 10^5$	$6.8 * 10^5$	$5.3 * 10^6$	$2.9 * 10^7$
	Communities	2	10	2,000	5	5	5
	Res. Param.	0.2	0.5	15	0.1	1	1
Modularity	Our method	0.84	0.92	0.77	0.95	0.76	0.64
	Hu et al.	0.85	0.91	0.58	0.95	0.74	0.64
	Hierarchical	0.77	0.88	0.88	0.94	0.65	0.92
	Louvain	0.72	0.83	0.89	0.90	0.78	0.97
	Spectral	0.60	0.56	-5.88	0.90	0.30	0.04
	Ref	0.83	0.92	0.89	0.90	0.00	0.00
Classification	Our method	0.97	0.90	0.92	—-	—-	—-
	Hu et al.	0.95	0.80	0.72	—-	—-	—-
	Hierarchical	0.98	0.93	0.80	—-	—-	—-
	Louvain	0.98	0.96	0.87	—-	—-	—-
	Spectral	0.95	0.30	0.09	—-	—-	—-
Time (sec.)	Our method	0.55	59	63	19	135	1284
	Hu et al.	0.80	167	206	42	152	39196
	Hierarchical	0.55	16	6	44	3066	9437
	Louvain	0.38	9	6	14	89	520
	Spectral	0.87	301	1855	24	265	1804

TABLE 1

Results on six data sets. Our method generally does better than that of Newman and Hu et al. It is also notable that the choice of metric matters. For instance, on the MNIST example, the Louvain method gets a worse modularity score than our method but better agreement with the ground truth labels. Conversely, our method gets a lower modularity score than Louvain on the Plume40 example, but the segmentation our method produced for Figure 3 more closely agrees with domain experts' knowledge of how the plume really looks. See Table 3 for an example of how a small amount of supervision with our method reduces this ambiguity. Dashes denote missing entries in cases where metadata was not available. The LFR50k example illustrates the ability of our approach to deal with a large number of small communities using recursive partitioning.

partitions with a classification accuracy greater than 95% were found only about 4% of the time.

LFR 50k This is a well-known ensemble of artificial networks [47]. We used the following parameters to generate it: 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. The large number of small communities makes this a challenging problem—similar experiments on a 1,000-node networks with 40 communities gave near-perfect classification. We use purity to gauge classification accuracy. Given two partitions g_1 and g_2, the purity is defined as $\frac{1}{N}\sum_{\alpha=1}^{\hat{n}} \max_{\beta=1,\ldots,\hat{n}} \#\{i : g_1 = \alpha \text{ and } g_2 = \beta\}$, where # denotes the cardinality.

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 is to identify pixels that contain similar materials, such as dirt, road, grass, etc.

The graph representation was computed using "nonlocal means" [9], which means that for each

		Jas. Rid.	Samson	Cuprite	FLC	Pavia U	Salinas	Salinas 1
	Nodes	19,800	14,820	30,162	208,780	207,400	7,092	111,063
	Edges	$1.1 * 10^5$	$8.3 * 10^4$	$1.6 * 10^5$	$1.5 * 10^6$	$1.6 * 10^6$	$4.7 * 10^5$	$8.4 * 10^5$
	Communities	4	3	12	3	9	6	16
Modularity	Our method	0.99	0.98	0.99	0.94	0.93	0.97	0.96
	Hu et al.	0.99	0.98	0.90	0.94	0.94	0.97	0.96
	Hierarchical	0.98	0.98	0.99	0.93	0.93	0.97	0.96
	Louvain	0.99	0.98	0.99	0.90	0.88	0.95	0.95
	Spectral	0.91	0.90	0.91	0.90	0.90	0.96	0.90
	Ref	0.90	0.90	0.90	0.90	0.90	0.90	0.90
Time (sec.)	Our method	17	13	42	121	160	4.6	96
	Hu et al.	40	27	63	203	270	3.3	117
	Hierarchical	1.5	1.1	2.4	378	411	0.74	66
	Louvain	1.5	1.2	2.7	39	40	0.75	15
	Spectral	28	10	148	38	65	6.5	24

TABLE 2

Results on additional hyperspectral data sets. The resolution parameter was 0.1*, and the reference partition has all nodes in the same community. Our method achieves a top modularity score in each network except for Salinas, where Hu et al.'s method gets slightly higher results. Our method partitioned recursively and initialized with kmeans clustering on leading eigenvectors that had been computed for use in the pseudospectral scheme.*

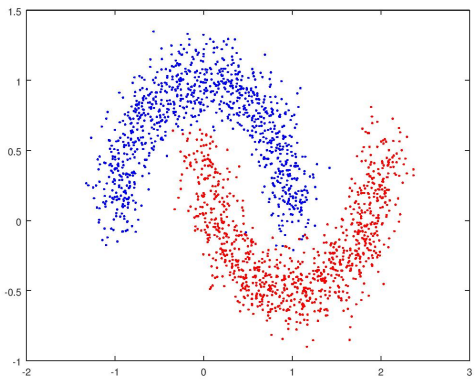

FIGURE 1. *Projection of the two moons example onto two dimensions*

pixel p, a vector v_p was constructed by concatenating the data in a 3×3 window centered at p. One then uses a 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 [70]. The images in Figure 2 were selected from a collection of 200 segmentations as being the most visually appealing. We compared with a recent NLTV-based algorithm [79], which is specifically designed

		Moons	MNIST
Modularity	Unsupervised	0.84	0.91
	10% supervised	0.84	0.92
	Reference	0.83	0.92
Classification	Unsupervised	0.97	0.90
	10% supervised	0.97	0.97
Modularity Consistency	Unsupervised	0.75	0.65
	10% supervised	1.00	1.00
Classification Consistency	Unsupervised	0.75	0.05
	10% supervised	1.00	0.65

TABLE 3

Results of our method using networks constructed from the two moons and MNIST examples with and without 10% supervision. Consistency here denoted percent of cases for which the results were within 2% of the best value achieved. In the two moons example, supervision improves consistent matching to metadata. In the MNIST example, both consistency and peak metadata matching are substantially improved. Note that in both cases, the peak modularity is not changed, indicating that the supervision helps the solver find local maxima that are more relevant to the classification task, thus addressing the well-known degeneracy issues of modularity's energy landscape. The code was run 20 times on each example.

for hyperspectral imaging applications and found our segmentation competitive. We also compared with Modularity MBO and GenLouvain [41] segmentations. For instance, Balanced TV does well at placing the grass into a single class and correctly resolved the difference between pavement and dirt. Balanced TV gives the sharpest resolution of the roads and the surrounding dirt in the upper right. Our method 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 made of different materials from most of the houses further down in the image, and NLTV has a similar problem.

Plume Hyperspectral Video The gas plume hyperspectral video records a gas plume being released at the Dugway Proving Ground [29, 49, 53].[12] 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 3, together with segmentations produced by competing algorithms. Our method is the only one that places the entire plume in a single class. The images shown were chosen as the best out of thirty for visual appeal.

[12] In [53], a semi-supervised MBO-type approach was used.

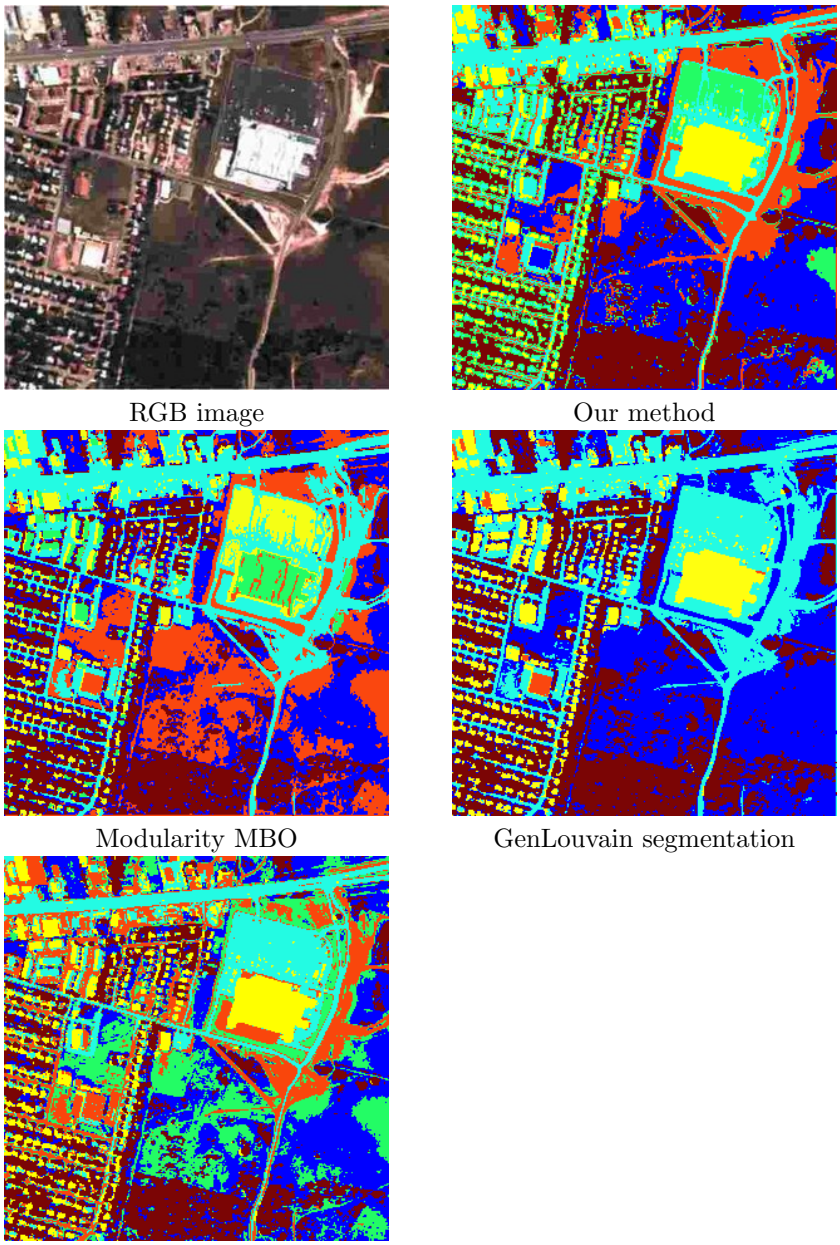

FIGURE 2. *The urban dataset segmented using different methods. Our method effectively separates the dirt from roads, resolving the roads in the upper right corner, and placing all of the grass into a single class. It has some difficulty with the buildings in the upper left corner, just below the main road, which are a different material from the other buildings.*

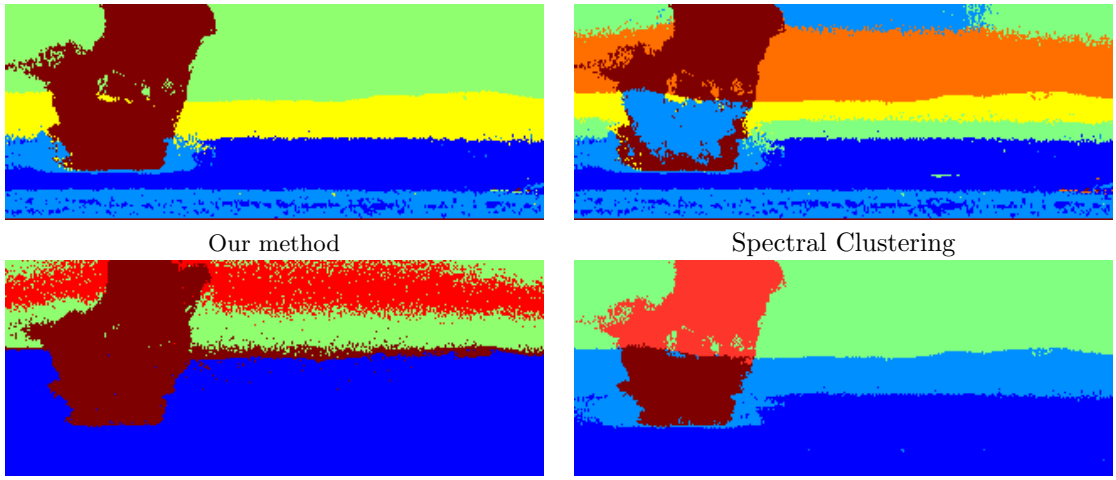

FIGURE 3. *Segmentations of the plume hyperspectral video using different methods. Observe that our method is the only method that gets the whole plume into a single class without any erroneous additions.*

Other Hyperspectral Examples We included seven additional hyperspectral image examples, which are well-known in the image processing community. In each case, we formed the k-nearest neighbor graph using nonlocal means and VLFeat. See Appendix C for more details. Overall, our algorithm performs very competitively on these examples in terms of modularity. The speed is slower than Louvain, but the run time is still very reasonable, and the modularity scores are more consistently good.

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 that 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 its predecessor, Modularity MBO.

Acknowledgements. A. L. Bertozzi and Z. M. Boyd were supported by NSF grants DMS-1417674 and DMS-1118971. X. C. Tai and Z. M. Boyd were supported by ISP-Matematikk (Project no. 2390033/F20) at the University of Bergen. X. C. Tai was additionally supported by the startup grant at Hong Kong Baptist University. Z. M. Boyd was additionally supported by the U.S. Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

A. Gamma convergence. The following are some basic facts about Gamma-convergence to aid in understanding the results of this paper. See [68] 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* $\liminf_{n\to\infty} \mathcal{F}_n(x_n) \leq \mathcal{F}(x)$.
2. *For every x, there exists a convergent sequence $x_n \to x$ such that* $\limsup_{n\to\infty} \mathcal{F}_n(x_n) \geq F(x)$.

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 Theorem 3.9.

Proof. We largely follow [68], generalizing and filling in a minor hole from that proof.

Observe that all of the terms not involving the potential are continuous and independent of ϵ, so they cannot interfere with the Γ-convergence [19]. Therefore, it suffices to prove that $\frac{1}{\epsilon}T$ Γ-converges to

$$\chi(u) = \begin{cases} 0 & \text{if } u \text{ corresponds to a partition} \\ +\infty & \text{otherwise.} \end{cases}$$

To prove the lower bound, let $u_n \to u$ and $\epsilon_n \to 0$. If u corresponds to a partition, then $\chi(u) = 0$, which is automatically less than or equal to $\frac{1}{\epsilon_n}T(u_n)$ for each n. If u does not correspond to a partition, then $\chi(u) = +\infty$. Pick N_1 such that whenever $n > N_1$, the distance from u_n to the nearest feasible point is at least $c > 0$. Letting T_c be the infimum of T on all of $\mathbb{R}^{N\times\hat{\mathrm{n}}}$ minus the balls of radius c surrounding each feasible point (so $T_0 > 0$ in particular). Then we have $\liminf_{n\to\infty} \frac{1}{\epsilon_n}T(u_n) \geq \lim_{n\to\infty} \frac{1}{\epsilon_n}T_0 = +\infty$. Thus, the lower bound always holds.

To prove the upper bound, let u be any $N \times \hat{\mathrm{n}}$ matrix. If u corresponds to a partition, then letting $u_n = u$ for all n gives the required sequence. If u does not correspond to a partition, then $u_n = u$ for all n still satisfies the upper bound requirement.

Thus both the upper and lower bound requirements hold, and we have proved Γ-convergence.□

B. Deferred proofs. In this section, we give proofs of propositions stated earlier in the paper.

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

$$||u - u_0||_\infty \leq ||e^{-\tau M} - I||_\infty ||u_0||_\infty = ||e^{-\tau M} - I||_\infty \leq \sum_{k=1}^{\infty} \frac{1}{k!}\tau^k ||M||_\infty^k = e^{\tau||M||_\infty} - 1 \tag{34}$$

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

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

These computations do not depend on $\hat{\mathrm{n}}$, but in order to get a timestep, we assume that $\hat{\mathrm{n}} = 2$. In this case, let u^1 and u^2 be the columns of u. We have $u_t^1 = -Mu^1$ and $u_t^2 = -Mu^2$. 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 (34) 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

$$||u-u_0||_\infty \leq \sqrt{\hat{\mathrm{n}}}||u-u_0||_2 \leq \sqrt{\hat{\mathrm{n}}}||e^{-\tau M}-I||_2||u_0||_2 \leq \sqrt{\hat{\mathrm{n}}}||u_0||_2 \sum_{k=1}^{\infty} \frac{1}{k!}\tau^k ||M||_2^k \tag{35}$$

$$= \sqrt{\hat{\mathrm{n}}}||u_0||_2 \left(e^{\tau||M||_2}-1\right) = \sqrt{\hat{\mathrm{n}}}||u_0||_2 \left(e^{\tau\rho}-1\right) \tag{36}$$

As before, when we let $\hat{\mathrm{n}} = 2$, 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 semi-definite, 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. ∎

C. Hyperspectral Image Details. In this appendix we collect some basic facts about the images used in Table 2.

- Jasper Ridge: An image of a river area. It has 198 channels and 100x100 pixels. Retrieved from http://www.escience.cn/people/feiyunZHU/Dataset_GT.html.
- Samson: An image of a coastline. It has 156 channels and 952x952 pixels. Retrieved from http://www.escience.cn/people/feiyunZHU/Dataset_GT.html.
- Cuprite: An image of ground near Las Vegas. It has 224 channels and 250x190 pixels. Retrieved from http://www.escience.cn/people/feiyunZHU/Dataset_GT.html.
- FLC: A moderate-dimensional image. It has 12 channels and 949x220 pixels. Available at ftp://www.daba.lv/pub/TIS/atteelu_analiize/MultiSpec/tutorial/ModDimensionDataSet.zip.
- Pavia U: An image of Pavia University in Northern Italy. It has 103 channels and 610x610 pixels. Retrieved from http://lesun.weebly.com/hyperspectral-data-set.html.
- Salinas: An image containing vineyard fields, soils, and vegetation. It has 224 channels and 512x217 pixels. Retrieved from http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes.
- Salinas 1: A subimage of the previous image containing 86x83 pixels.

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] A. Clauset, M. E. J. Newman, and C. Moore, *Finding community structure in very large networks*, Phys. Rev. E, 70 (2004), 066111.
[18] G. Csardi and T. Nepusz, *The igraph software package for complex network research*, InterJournal, Complex Systems (2006), p. 1695, http://igraph.org.
[19] G. Dal Maso, *An Introduction to Gamma-convergence*, Birkhauser, Boston, 1993.
[20] 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.
[21] E. Davis and S. Sethuraman, *Consistency of modularity clustering on random geometric graphs*, CORR, abs/1604.03993v1 (2016).
[22] S. Esedoglu and Y.-H. R. Tsai, *Threshold dynamics for the piecewise constant mumford-shah functional*, J. Comput. Phys., 211 (2006), pp. 367–384.
[23] L. Evans, *Convergence of an algorithm for mean curvature motion*, Indiana Univ. Math J., 42 (1993), pp. 553–557.
[24] S. Fortunato, *Community detection in graphs*, Phys. Rep., 486 (2010), pp. 75–174.
[25] S. Fortunato and M. Barthélemy, *Resolution limit in community detection*, Proc. Natl. Acad. Sci., 104 (2007), pp. 36–41.
[26] S. Fortunato and D. Hric, *Community detection in networks: A user guide*, Phys. Rep., 659 (2016), pp. 1 – 44.
[27] H. Gao, M. Guo, R. Li, and L. Xing, *4DCT and 4D cone-beam ct reconstruction using temporal regularization*, in Graphics Processing Unit-Based High Performance Computing in Radiation Therapy, X. Jia and S. B. Jiang, eds., CRC Press, 2015, pp. 63–82.
[28] 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.
[29] 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.
[30] G. Gilboa, *A total variation spectral framework for scale and texture analysis*, SIAM J. Imag. Sci., 7 (2014), pp. 1937–1961.
[31] G. Gilboa and S. Osher, *Nonlocal operators with applications to image processing*, Multiscale Model. Simul., 7 (2008), pp. 1005–1028.
[32] R. Glowinski, T.-W. Pan, and X.-C. Tai, *Some facts about operator-splitting and alternating direction methods*, in Splitting Methods in Communication, Imaging, Science, and Engineering, R. Glowinski, S. J. Osher, and W. Yin, eds., Springer International Publishing, Cham, 2016, pp. 19–94.
[33] 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.
[34] T. Goldstein and S. Osher, *The Split Bregman method for L1-regularized problems*, SIAM J. Img. Sci., 2 (2009), pp. 323–343.
[35] B. H. Good, Y.-A. de Montjoye, and A. Clauset, *Performance of modularity maximization in practical contexts*, Phys. Rev. E, 81 (2010), p. 046106.
[36] 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.
[37] A. Harten, *High resulution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.
[38] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. Murray, *From molecular to modular cell biology*, Nature, 402 (1999).
[39] 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.
[40] 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.
[41] 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).
[42] B. W. Kernighan and S. Lin, *An efficient heuristic procedure for partitioning graphs*, Bell System Technical J., 49 (1970), pp. 291–307.
[43] 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.
[44] R. Kohn and P. Sternberg, *Local minimisers and singular perturbations*, Proc. Roy. Soc. Edinburgh Sect. A, (1989).
[45] 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.
[46] A. Lancichinetti and S. Fortunato, *Limits of modularity maximization in community detection*, Phys. Rev. E, 84 (2011), p. 066122.
[47] A. Lancichinetti, S. Fortunato, and F. Radicchi, *Benchmark graphs for testing community detection algorithms*, Phys. Rev. E., 78 (2008), p. 056117.
[48] Y. LeCun and C. Cortes, *The MNIST database of handwritten digits*, 1998.
[49] 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.
[50] 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.
[51] 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.
[52] E. Merkurjev, E. Bae, A. L. Bertozzi, and X.-C. Tai, *Global binary optimization on graphs for classification of high-dimensional data*, Journal of Mathematical Imaging and Vision, 52 (2015), pp. 414–435.
[53] 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.
[54] B. Merriman, J. Bence, and S. Osher, *Diffusion generated motion by mean curvature*, Proc. Comput. Crystal Growers Workshop, (1992), pp. 73–83.
[55] 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.
[56] M. Newman, *Networks: an Introduction*, 2010.
[57] M. E. Newman, *Modularity and community structure in networks*, Proc. Nat. Acad. Sci., 103 (2006), pp. 8577–8582.
[58] M. E. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E, 69 (2004), p. 026113.
[59] M. E. J. Newman, *Finding community structure in networks using the eigenvectors of matrices*, Phys. Rev. E, 74 (2006), p. 036104.
[60] M. E. J. Newman, *Equivalence between modularity optimization and maximum likelihood methods for community detection*, Phys. Rev. E, 94 (2016), p. 052315.
[61] A. Noack, *Modularity clustering is force-directed layout*, Phys. Rev. E, 79 (2009), p. 026102.
[62] O. A. Oleinik, *Discontinuous solutions of nonlinear differential equations*, Uspekhi Mat. Nauk, 12 (1957), pp. 3–73.
[63] L. Peel, D. B. Larremore, and A. Clauset, *The ground truth about metadata and community detection in networks*, Science Advances, 3 (2017).
[64] M. A. Porter, J.-P. Onnela, and P. J. Mucha, *Communities in networks*, Notic. Amer. Math. Soc., 56 (2009), pp. 1082–1097.
[65] J. Reichardt and S. Bornholdt, *Statistical mechanics of community detection*, Phys. Rev. E, 74 (2006),

p. 016110.
[66] L. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation noise removal algorithm*, Phys. D, 60 (1992), pp. 259–268.
[67] 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.
[68] Y. van Gennip and A. L. Bertozzi, *Gamma-convergence of graph Ginzburg-Landau functionals*, Adv. Differential Equations, 17 (2012), pp. 1115–1180.
[69] 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.
[70] A. Vedaldi and B. Fulkerson, *VLFeat: An open and portable library of computer vision algorithms*. http://www.vlfeat.org/, 2008.
[71] 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.
[72] U. von Luxborg, *A tutorial on spectral clustering*, Statist. Comput., 17 (2007), pp. 395–416.
[73] 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.
[74] 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.
[75] 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.
[76] W. W. Zachary, *An information flow model for conflict and fission in small groups*, J. Anthropological Res., 33 (1977), pp. 452–473.
[77] L. Zelnik-Manor and P. Perona, *Self-tuning spectral clustering*, NIPS, 17 (2004), p. 16.
[78] 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.
[79] 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.