Modified Cheeger and Ratio Cut Methods Using the Ginzburg-Landau Functional for Classification of High-Dimensional Data

Ekaterina Merkurjev[*], Andrea Bertozzi[**], Xiaoran Yan[***], and Kristina Lerman[***]

[*]Department of Mathematics, University of California, San Diego

[**]Department of Mathematics, University of California, Los Angeles

[***]Information Sciences Institute, University of Southern California, Marina del Rey

Email: emerkurj@ucsd.edu, bertozzi@math.ucla.edu, xiaoran@isi.edu, lerman@isi.edu.

Abstract

Recent advances in clustering have included continuous relaxations of the Cheeger cut problem and those which address its linear approximation using the graph Laplacian. In this paper, we show how to use the graph Laplacian to solve the fully nonlinear Cheeger cut problem, as well as the ratio cut optimization task. Both problems are connected to total variation minimization, and the related Ginzburg-Landau functional is used in the derivation of the methods. The graph framework discussed in this paper is undirected. The resulting algorithms are efficient ways to cluster the data into two classes, and they can be easily extended to the case of multiple classes, or used on a multiclass data set via recursive bipartitioning. In addition to showing results on benchmark data sets, we also show an application of the algorithm to hyperspectral video data.

Keywords: classification, spectral clustering, Cheeger cut, ratio cut, graphs, Ginzburg-Landau functional, total variation, graph Laplacian, Nyström extension method.

1 Introduction

1.1 General Problem and Background

Total variation has become increasingly more central to inverse problems in imaging. The classic paper by Rudin, Osher and Fatemi [41] sparked interest in total variation by applying it very successfully to denoising problems. Later, similar techniques were applied to areas such as inpainting, image restoration and blind deconvolution. For example, in [14], Chan et al. present a blind deconvolution method using total variation (TV), in which they use an alternating minimization implicit iterative scheme. In addition, while [12] proposes a nonlinear primal-dual method for TV-based image restoration, [13] formulates a higher order TV-based restoration algorithm.

More recently, total variation has been used successfully in a graphical framework. In [35], the authors present a global minimization framework for segmentation of high-dimensional data into two classes using graph-based total variation. In [2], the authors propose solving a convex relaxation for a certain subset of graph-based multiclass data segmentation problems involving total variation in a graphical setting.

Moreover, note that, in a graph setting, total variation is exactly the graph cut when applied to an indicator function. Therefore, the minimum cut problem becomes a natural mechanism with which to proceed to achieve accurate data classification. We thus consider the problem of minimizing the cut as a way of partitioning the set X into two clusters. Let $G = (V, E)$ represent an undirected graph with the set of vertices V and set of edges E, and consider a target set X of size n embedded in a graph G. A weight function is defined on the set of edges, and represents a measure of similarity between the vertices it is connecting. As usual, the degree of a vertex x is formulated as $d(x) = \sum_{y \in V} w(x, y)$. Denote by $\mathbf{D}$ the diagonal matrix containing the degrees of vertices as diagonal entries, and let $\mathbf{W}$ denote the similarity matrix containing the weight function values. The minimum cut problem is to find the set $S \subset V$ such that the following value is minimized:

$$cut(S, \bar{S}) = \sum_{x \in S, y \in \bar{S}} w(x, y).$$

Here, $\bar{S}$ indicates the complement. Intuitively, the weights between vertices of different clusters should be small. However, minimizing the cut leads to an undesirable solution, first and foremost because the trivial solution would be to cluster all the points into one set, thus giving the cut the value of zero. Even if a non-trivial partition condition is enforced, the solution tends to isolate individual vertices from the rest of the graph. Usually, what is wanted is for the two clusters to be relatively close in size, and one solution is to include a normalization of the cut.

One normalization is the ratio cut. The problem is to find a subset S of V such that

$$RatioCut(S, \bar{S}) = cut(S, \bar{S})\Big(\frac{1}{|S|} + \frac{1}{|\bar{S}|}\Big)$$

is minimized. This is a NP hard discrete problem [49]. One way to simplify it would be to allow the solution to take arbitrary values in $\mathbb{R}$. A clever reformulation [48] of the original task and relaxation of the binary constraint leads to the problem of finding the argument of the following optimization task:

$$\min_{u \in R^n} \langle u, \mathbf{L}u \rangle, \quad \text{such that } u \perp \mathbf{1}, \quad ||u||^2 = n, \tag{1}$$

where $\mathbf{L}$ is the graph Laplacian $\mathbf{D} - \mathbf{W}$. We emphasize the fact that the above problem obtains a real-valued solution instead of a discrete-valued one. To solve (1), one can apply the Rayleigh-Ritz theorem, and the solution is given by the second eigenvector of the graph Laplacian. To obtain a binary solution, which indicates the binary partition, one can use several methods, the simplest of which is thresholding.

Another normalization of the cut is the normalized cut. If we let $vol(S) = \sum_{x \in S} d(x)$, where $d(x)$ represents the degree of vertex x, the problem is modified to find a subset S of V such that

$$Ncut(S, \bar{S}) = cut(S, \bar{S})\Big(\frac{1}{vol(S)} + \frac{1}{vol(\bar{S})}\Big)$$

is minimized. This is a NP hard discrete problem [49] as well. We simplify it by allowing the solution to take arbitrary values in $\mathbb{R}$. A reformulation of the normalized cut problem and relaxation of the binary constraint lead to the following task of finding the argument of the minimum of the task:

$$\min_{u \in R^n} \langle u, \mathbf{L}u \rangle, \quad \text{such that } Du \perp \mathbf{1}, \quad \langle u, \mathbf{D}u \rangle = vol(V),$$

where $vol(V) = \sum_x d(x)$. One can again apply the Rayleigh-Ritz theorem, and the solution is given by the second eigenvector of the random walk Laplacian $\mathbf{D^{-1}L}$. Thresholding can be used to obtain a binary solution which represents the desired partition.

The last normalization to be discussed is the Cheeger cut, which involves finding a subset S of V that minimizes

$$h(S, \bar{S}) = \frac{cut(S, \bar{S})}{min(|S|, |\bar{S}|)}.$$

If volume of a set is used instead, one obtains the normalization

$$h^v(S, \bar{S}) = \frac{cut(S, \bar{S})}{min(vol(S), vol(\bar{S}))},$$

which is also known as conductance. Let $\phi(G) = \min_{S \in V} h^v(S, \bar{S})$. The Cheeger cut and the second eigenvalue λ_2 of the matrix $\mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$ can be related via the following Cheeger inequality [15]:

$$\frac{\phi(G)^2}{2} \leq \lambda_2 \leq 2\phi(G). \tag{2}$$

The authors of [22] present a generalized version of it.

We have seen that the ratio cut and the normalized cut problems can be formulated in a relaxed setting, with non-binary solutions given by the second eigenvectors of the Laplacian and the random walk Laplacian, respectively, with the binary answer obtained by thresholding. However, for an arbitrary number of classes, this technique is too simple. The method of spectral clustering computes the k clusters using the steps:

1. Formulate the data set in a graph setting.
2. Compute either the Laplacian ($\mathbf{L} = \mathbf{D} - \mathbf{W}$) or the random walk Laplacian ($\mathbf{L_{rw}} = \mathbf{D^{-1}L}$).
3. Compute the first k eigenvectors $\{v_1, v_2, ..., v_k\}$ of the Laplacian (or the random walk Laplacian).
4. Let $\mathbf{U}$ be the matrix containing the vectors $\{v_1, v_2, ..., v_k\}$ as columns.
5. Cluster the rows of the matrix $\mathbf{U}$ with the k-means algorithm into k clusters.

There is a large literature on the binary partitioning problem on graphs. Here, we review work that relates to solving the cut problem under some normalization condition. In [11], the authors present a generalized version of spectral clustering using the graph p-Laplacian. They show that, as $p \to 1$, the cut resulting from thresholding the second eigenvector of the graph p-Laplacian is the solution to the Cheeger cut problem. An efficient scheme to calculate the eigenvector is then introduced. In [26], Hein et al. show that some constrained optimization problems can be formulated as nonlinear eigenproblems. The authors then describe a generalization of the inverse power method which converges to nonlinear eigenvectors. This method is applied to spectral clustering and sparse PCA. Moreover, in [16], Chung describes the heat kernel as the pagerank of a graph, and uses it and the local Cheeger cut inequality to

establish a fast graph partitioning algorithm. In [42], Spielman et al. present nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems.

Recent work by Bresson, Szlam, Laurent, von Brecht, et al. includes several important papers related to clustering. In [44], Szlam et al. present a relaxation of the Cheeger cut problem and then show similarities of the energy of the relaxed problem and well studied energies in image processing. An algorithm based on the split-Bregman method [24] is then developed to minimize the proposed energy. Authors of [6] describe two procedures solving the relaxed Cheeger cut problem. The first algorithm is a novel steepest descent approach, and the second one is a modified inverse power method. Some convergence results are also given. In [8], Bresson et al. develop another version of the method shown in [6] using a new adaptive stopping condition. The result is an algorithm that is monotonic and much more efficient than before. Note that the mentioned methods are unsupervised, which do not incorporate any known data. As opposed to these kinds of algorithms, semi-supervised optimization approaches involve the inclusion of a fidelity term, a fitting term to known data.

Multiclass extensions relating to the cut have also been proposed. One approach is to use recursion and thus solve a collection of binary problems. However, other ideas have also been introduced. The authors of [7] present a framework for multiclass total variation clustering that does not use recursion. They formulate the Cheeger energy in a multiclass setting, and then relax the energy in a continuous setting. This results in an optimization problem involving total variation, which is then solved using the proposed proximal splitting algorithm. In [9], a multiclass extension of the result of [44] is derived. The method deals with learning several classes simultaneously with a set of labels, and is made even more efficient by the usage of fast L^1 solvers, designed for the total variation semi-norm. Finally, we mention the work of Hu et al. [28] in which the authors propose a novel clustering algorithm involving graph-based total variation, which is the cut when applied to an indicator function. In the paper, the authors formulate modularity optimization as a minimization problem using an energy functional that contains a total variation term.

In addition to using the cut as a basis for our method, one technique used in this work is one of spectral methods. Spectral methods are a type of approach used to solve differential equations. Their idea is to write the solution of the differential equation, and perhaps other relevant parts of the problem, as a sum of basis functions. The coefficients of the sum are then calculated so that the differential equation is satisfied (or satisfied to the best ability). In this work, we use the eigenfunctions of a Laplacian as basis functions.

1.2 Our Contributions

Specifically, our contributions are:

1. We show how to use the graph Laplacian to solve the fully nonlinear ratio cut problem. We call this algorithm the Modified Ratio Cut Method (RCCM), and present it in Section 3. Two approaches are presented; one approach links a modified ratio cut formulation to an optimization problem involving total variation, and the other one deals with solving the ratio cut formulation directly.
2. We show how to use the graph Laplacian to solve the fully nonlinear Cheeger cut problem. We call this algorithm the Modified Cheeger Cut Method (MCCM), and present it in Section 4.

3. A main advantage of these methods is efficiency, while being able to maintain the accuracy of that of the state-of-the-art. The efficiency is partly achieved through the use of the Nyström extension method [18, 19], detailed in the Appendix. Moreover, it is possible to modify the methods to be applicable to the multiclass case. Of course, techniques like recursive bipartitioning can be used with the existing procedures.

2 Graphical Framework

2.1 Fundamentals

In this paper, we consider a graphical framework with an undirected graph $G = (V, E)$, where V is the set of vertices and E is the set of edges. Earlier in Section 1, we introduced some important concepts related to the framework, such as the weight function, the degree and the similarity matrix. One advantage of using a graphical framework is that it allows one to take into account the relationship between any two nodes in the data set. In the case of image processing, using nonlocal information makes it easier for one to capture repetitive structure and texture, as shown in [37]. Furthermore, the framework is very general, and can be easily applied to any data set.

It is possible to introduce some common mathematical operators in a graphical setting. In this section, we will only be concerned with the graph version of the differential Laplace (Δ) operator. Although many versions exist, three popular ones are the unnormalized Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$, symmetric Laplacian $\mathbf{L_s} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$, and the random walk Laplacian $\mathbf{L_{rw}} = \mathbf{D^{-1}L}$. The graph Laplacians have the following easily shown properties:

1) $\mathbf{L}$ and $\mathbf{L_s}$ are symmetric.

2) $\mathbf{L}$, $\mathbf{L_s}$ and $\mathbf{L_{rw}}$ are positive semi-definite.

3) $\mathbf{L}$, $\mathbf{L_s}$ and $\mathbf{L_{rw}}$ have non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.

4) The smallest eigenvalue of $\mathbf{L}$ is 0; the corresponding eigenvector is just a constant vector.

5) λ is an eigenvalue of $\mathbf{L_{rw}}$ with eigenvector u if and only if λ is an eigenvalue of $\mathbf{L_s}$ with eigenvector $w = \mathbf{D}^{\frac{1}{2}}u$.

Note that the multiplicity of eigenvalue 0 equals the number of connected components in the graph.

2.2 Graphical Framework, Extended

We outline the graphical framework in more detail here, giving general definitions of other operators. The operators are defined similarly to [25, 47], where the justification for these choices is explained.

Let n be the number of vertices in the graph and let $\mathcal{V} \cong \mathbb{R}^n$ and $\mathcal{E} \cong \mathbb{R}^{\frac{n(n-1)}{2}}$ be Hilbert spaces (associated with the set of vertices and edges, respectively) defined via the inner products:

$$\langle u, \gamma \rangle_{\mathcal{V}} = \sum_x u(x)\gamma(x)d(x)^r,$$

$$\langle \psi, \phi \rangle_{\mathcal{E}} = \frac{1}{2}\sum_{x,y} \psi(x,y)\phi(x,y)w(x,y)^{2q-1},$$

$$\langle u, \gamma \rangle_{\mathcal{L}^2} = \sum_x u(x)\gamma(x),$$

$\forall u, \gamma \in \mathcal{V}$ and $\forall \psi, \phi \in \mathcal{E}$, for some $r \in [0, 1]$ and $q \in [\frac{1}{2}, 1]$. Let us also define:

$$\|u\|_{\mathcal{V}} = \sqrt{\langle u, u \rangle_{\mathcal{V}}} = \sqrt{\sum_x u(x)^2 d(x)^r},$$

$$\|\phi\|_{\mathcal{E}} = \sqrt{\langle \phi, \phi \rangle_{\mathcal{E}}} = \sqrt{\frac{1}{2} \sum_{x,y} \phi(x, y)^2 w(x, y)^{2q-1}},$$

$$\|u\|_{\mathcal{L}^2} = \sqrt{\langle u, u \rangle_{\mathcal{L}^2}} = \sqrt{\sum_x u(x)^2},$$

$$\|\phi\|_{\mathcal{E},\infty} = \max_{x,y} |\phi(x, y)|.$$

The gradient operator $\nabla : \mathcal{V} \to \mathcal{E}$ is formulated as:

$$(\nabla u)_w(x, y) = w(x, y)^{1-q}(u(y) - u(x)). \tag{3}$$

The Dirichlet energy does not include parameters r or q:

$$\frac{1}{2} \|\nabla u\|_{\mathcal{E}}^2 = \frac{1}{4} \sum_{x,y} w(x, y)(u(x) - u(y))^2. \tag{4}$$

The divergence $div_w : \mathcal{E} \to \mathcal{V}$ definition is based on the adjoint of the gradient:

$$(\mathrm{div}_w\, \phi)(x) = \frac{1}{2d(x)^r} \sum_y w(x, y)^q (\phi(y, x) - \phi(x, y)), \tag{5}$$

where we define the adjoint using the following definition: $\langle \nabla u, \phi \rangle_{\mathcal{E}} = \langle u, \mathrm{div}_w\, \phi \rangle_{\mathcal{V}}$.

We now have a set of graph Laplacians $\triangle_r = \mathrm{div}_w \nabla : \mathcal{V} \to \mathcal{V}$:

$$(\triangle_w u)(x) = \sum_y \frac{w(x, y)}{d(x)^r} (u(x) - u(y)). \tag{6}$$

In matrix form, we can write the set of Laplacians as

$$\triangle_w = \mathbf{D}^{1-r} - \mathbf{D}^{-r}\mathbf{W}. \tag{7}$$

The matrices resulting from $r = 0$ and $r = 1$ are called the unnormalized Laplacian and the random walk Laplacian, respectively. Using divergence, a set of anisotropic total variations $TV_w : \mathcal{V} \to \mathbb{R}$ can now be formulated:

$$TV_{w,q}(u) = \max \left\{ \langle \mathrm{div}_w\, \phi, u \rangle_{\mathcal{V}} : \phi \in \mathcal{E}, \|\phi\|_{\mathcal{E},\infty} \leq 1 \right\} = \frac{1}{2} \sum_{x.y} w(x, y)^q |u(x) - u(y)|. \tag{8}$$

The last operator to be defined will involve the Ginzburg-Landau functional [3–5, 20, 29, 36, 37] that our methods are based upon:

$$GL_\epsilon(u) = \|\nabla u\|_{\mathcal{E}}^2 + \frac{1}{\epsilon} \sum_x \mathcal{W}(u(x)), \tag{9}$$

where $\mathcal{W}$ is the double well potential $\mathcal{W}(u) = u^2(u - 1)^2$. Note that, while the first term in the continuous Ginzburg-Landau functional

$$GL_c(u) = \epsilon \int |\nabla u|^2 dx + \frac{1}{\epsilon} \int \mathcal{W}(u) dx$$

is scaled by ϵ, the first term of GL_ϵ does not include ϵ. This is because, unlike the case with the continuous functional, the difference terms of GL_ϵ are finite even for binary functions, and no rescaling of the first term is necessary.

In relation to parameters, we choose $q = 1$ since in [47], it is shown that for any r and $q = 1$, TV_w is the Γ-limit (Gamma convergence) of a sequence of graph-based Ginzburg-Landau (GL)-type functionals:

Theorem 1. $GL_\epsilon \xrightarrow{\Gamma} GL_0$ *as* $\epsilon \to 0$*, where*

$$GL_\epsilon(u) = \|\nabla u\|_{\mathcal{E}}^2 + \frac{1}{\epsilon}\sum_x \mathcal{W}(u(x)) = \frac{1}{2}\sum_{x,y} w(x,y)(u(x)-u(y))^2 + \frac{1}{\epsilon}\sum_x \mathcal{W}(u(x)),$$

$$GL_0(u) = \begin{cases} TV_{w,1}(u) & \textit{for } u \textit{ s.t. } u(x) \in \{0,1\}, \\ \infty & \textit{otherwise}. \end{cases}$$

Proof. See Theorem 3.1 of [47]. ■

Thus, to be consistent with their work, use $q = 1$. The parameter r is allowed to change, since we experiment with different Laplacians. With the parameters so defined, total variation is now formulated as

$$TV_w(u) = \max\left\{\langle \mathrm{div}_w\, \phi, u\rangle_{\mathcal{V}} : \phi \in \mathcal{E}, \|\phi\|_{\mathcal{E},\infty} \le 1\right\} = \frac{1}{2}\sum_{x.y} w(x,y)|u(x)-u(y)|. \tag{10}$$

This is the definition used in the paper.

3 Modified Ratio Cut Method

Let us consider the binary partitioning problem involving the target set X (of size n) embedded in a graphical framework. One way to classify the set into two classes would be to find the partition that minimizes some normalization of the cut. In this section, our derivations are based on

$$RatioCut(S,\bar{S}) = cut(S,\bar{S})\Big(\frac{1}{|S|} + \frac{1}{|\bar{S}|}\Big).$$

The ratio cut is related to the Cheeger cut $h(S,\bar{S})$ via the following inequality:

$$h(S,\bar{S}) \le RatioCut(S,\bar{S}) \le 2h(S,\bar{S}),$$

which can be easily shown using the definitions. We now present two different approaches to the ratio cut problem.

3.1 Approach I

In this section, we do not solve the exact ratio cut formulation, but instead focus on a slightly modified version. Earlier, we discussed the general problem of minimizing the cut to find a desired partition:

$$\min_{S\subset V} \sum_{x\in S, y\in \bar{S}} w(x,y).$$

Since minimizing the cut is equivalent to minimizing its square, we can consider that slightly modified problem. However, some sort of normalization is needed to create a bias towards partitions where the two clusters are of similar

size. Using the regular ratio cut normalization, one obtains the modified ratio cut problem:

$$\min_{S\subset V} cut(S,\bar{S})^2\Big(\frac{1}{|S|}+\frac{1}{|\bar{S}|}\Big), \tag{11}$$

which is equivalent to

$$\min_{S\subset V} cut(S,\bar{S})\sqrt{\frac{1}{|S|}+\frac{1}{|\bar{S}|}} \tag{12}$$

Before giving more details, we note that if $\chi_S : G \to \{0,1\}$ is the indicator function of a subset S of G, then

$$TV_w(\chi_S) = cut(S,\bar{S}). \tag{13}$$

In addition, if the mean of a function is defined as the average of its values, we also have

$$\|\chi_S - mean(\chi_S)\|^2_{\mathcal{L}^2} = \sum_{x\in V}\Big(\chi_S(x)-\frac{|S|}{n}\Big)^2 = |S|\Big(1-\frac{|S|}{n}\Big)^2 + |\bar{S}|\Big(\frac{|S|}{n}\Big)^2 = \frac{|S||\bar{S}|}{n} = \frac{|S||\bar{S}|}{|S|+|\bar{S}|}. \tag{14}$$

Therefore, due to (13) and (14), problem (11) (and (12)) is equivalent to the following problem:

$$\min_{u:u(x)\in\{0,1\}} \frac{TV_w(u)}{\|u-mean(u)\|_{\mathcal{L}^2}}.$$

This is an NP-hard problem, and one way to simplify it would be to relax the binary condition to include a larger set of functions; in our case, we minimize over all functions $u : V \to \mathbb{R}$ with values in $\mathbb{R}$. We now show that there exists a binary minimizer of the relaxed problem, and it represents the solution to the modified ratio cut problem.

Theorem 2. *Let $u : V \to \mathbb{R}$. Consider the problem*

$$\lambda = \min_u \frac{TV_w(u)}{\|u-mean(u)\|_{\mathcal{L}^2}}. \tag{15}$$

There is binary valued minimizer, and

$$\lambda = \min_{S\subset V} cut(S,\bar{S})\sqrt{\frac{1}{|S|}+\frac{1}{|\bar{S}|}}. \tag{16}$$

Proof. The techniques of the proof are similar to the ones in [44]. Suppose that $u^\star$ is the minimizer of (15). We have $TV_w(u^\star) \neq 0$, otherwise $u^\star$ would be constant, and the ratio in (15) would be undefined. Note that the functional in (15) is scale invariant; we can thus rescale $u^\star$ so that $TV_w(u^\star)$=1. In this case, $u^\star$ is a maximizer of the denominator, constrained to the set of functions u such that $TV_w(u) \leq 1$. Let A_1 be the set of indices where $u^\star$ is non-positive, and A_2 be the set where it is positive. Define Z to be the set of functions that are non-positive on the first set and non-negative on the second one, and C to be the TV semi-norm unit ball on Z. Clearly, $u^\star$ is a solution to $\max_C \|u-mean(u)\|_{\mathcal{L}^2}$. Since the set C is convex, and $E(u) = \|u-mean(u)\|_{\mathcal{L}^2}$ is a convex functional on C, a maximum is also attained on at least one of the extreme points. By Lemma 2.2 of [44], any extreme point of C takes the form of a (scaled) indicator function of some set $S^\star$. Therefore, there exists an extreme point $v = \alpha\chi_{S^\star}$ of C, which is also a solution to $\max_C \|u-mean(u)\|_{\mathcal{L}^2}$. Note that due to the binary property of v, and since

$\|u^\star - mean(u^\star)\|_{\mathcal{L}^2} = \|v - mean(v)\|_{\mathcal{L}^2}$ and $TV_w(v) \leq TV_w(u^\star) = 1$,

$$\min_u \frac{TV_w(u)}{\|u - mean(u)\|_{\mathcal{L}^2}} = \lambda = \frac{TV_w(u^\star)}{\|u^\star - mean(u^\star)\|_{\mathcal{L}^2}} \geq \frac{TV_w(v)}{\|v - mean(v)\|_{\mathcal{L}^2}} =$$

$$= cut(S^\star, \bar{S^\star})\sqrt{\frac{1}{|S^\star|} + \frac{1}{|\bar{S^\star}|}} \geq \min_{S\subset V} cut(S, \bar{S})\sqrt{\frac{1}{|S|} + \frac{1}{|\bar{S}|}}. \quad (17)$$

Since

$$\min_{S\subset V} cut(S, \bar{S})\sqrt{\frac{1}{|S|} + \frac{1}{|\bar{S}|}} = \min_{u:u(x)\in\{0,1\}} \frac{TV_w(u)}{\|u - mean(u)\|_{\mathcal{L}^2}} \geq \min_u \frac{TV_w(u)}{\|u - mean(u)\|_{\mathcal{L}^2}},$$

we get equality in (17). ■

Therefore, we consider the problem

$$\min_u \frac{TV_w(u)}{\|u - mean(u)\|_{\mathcal{L}^2}}. \quad (18)$$

Due to the relationship shown in Theorem 1, the idea now is to interchange total variation, which is directly related to the cut, with the Ginzburg-Landau functional. This will result in an easier problem to minimize due to the Dirichlet energy term of the functional, which results in a gradient descent equation involving the Laplacian, for which we use an efficient spectral approach. By contrast, minimizing total variation produces a nonlinear curvature term, which we would generally like to avoid due to some problems it can create such as division by zero, etc..

We later introduce the use of the Nyström extension method [18, 19] - a clever technique that approximates the eigenvectors of a Laplacian using only a small portion of the graph weights. This method is very efficient, and especially useful for very big data sets. Details about the procedure can be found in the Appendix.

Replacing the total variation in (18) by the Ginzburg-Landau functional, we obtain the problem of

$$\min_u E(u) = \min_u \frac{GL_\epsilon(u)}{B(u)} = \min_u \frac{GL_\epsilon(u)}{\|u - mean(u)\|_{\mathcal{L}^2}}. \quad (19)$$

The $\mathcal{V}$-gradient of E is

$$\nabla E(u) = \frac{1}{B(u)}(2\triangle_w u + \frac{1}{\epsilon}\mathcal{W}'(u) - E(u)\nabla B(u)), \quad (20)$$

where

$$B(u) = \|u - mean(u)\|_{\mathcal{L}^2}. \quad (21)$$

Of course, ∇B is a gradient with respect to the $\mathcal{V}-$inner product.

Using gradient descent and calculating the Laplacian implicitly, one obtains the following scheme:

$$\frac{u^{n+1} - u^n}{dt} = -\frac{1}{B(u^n)}(2\triangle_w u^{n+1} + \frac{1}{\epsilon}\mathcal{W}'(u^n) - E(u^n)\nabla B(u^n)) \quad (22)$$

We note that, in the case that $r \neq 0$, the i^{th} term in the $\mathcal{W}'$ expression in (20) and (22), viewed numerically as a vector, should be scaled by d_i^{-r}, where d_i is the degree of node i, to result in correct $\mathcal{V}$-gradient flow.

3.2 Approach II

In this section, we consider the exact ratio cut problem. Let u be a binary function (taking values 0 or 1) denoting the class of each of the nodes. Then one can write the ratio cut problem as

$$\min_{u:u(x)\in\{0,1\}} TV_w(u)\Big(\frac{1}{\langle u, 1\rangle} + \frac{1}{\langle 1 - u, 1\rangle}\Big). \quad (23)$$

where $\langle , \rangle$ is an inner product. Such notation in the definition allows us to be general. If the inner product is just the $\mathcal{V}$-inner product with $r = 0$, then the problem becomes simply the ratio cut. If the $\mathcal{V}$-inner product with $r = 1$ is used, then the problem is the normalized cut.

Similarly to the procedure of the previous section, we relax the binary constraint and replace the total variation term in (23) by the GL functional, with motivation described in Section 3.1.1, so the following problem is formulated:

$$\min_u E(u) = \min_u \frac{GL_\epsilon(u)}{B(u)} = \min_u GL_\epsilon(u)\Big(\frac{1}{\langle u, 1\rangle} + \frac{1}{\langle 1-u, 1\rangle}\Big). \tag{24}$$

To solve (24), we use the scheme (22) with

$$B(u) = \frac{\langle u, 1\rangle\langle 1-u, 1\rangle}{\langle 1, 1\rangle}. \tag{25}$$

Of course, in the case of the ratio cut method, the inner product in the $B(u)$ term is the $\mathcal{V}$-inner product with $r = 0$.

3.3 The Procedure of Modified Ratio Cut Method (MRCM)

We now present two different versions of the algorithm solving (22). The first version computes the gradient term of the Ginzburg-Landau functional explicitly using the graph weights via equation (4). The second version uses

$$\|\nabla u\|_{\mathcal{E}}^2 = -\langle \triangle u, u\rangle_{\mathcal{V}} \tag{26}$$

and the spectral eigencomposition of the Laplacian to compute the Dirichlet's energy.

In the case of a very large data set, computing the entire graph is prohibitive. We approach the problem by using spectral techniques and the Nyström extension method. The latter algorithm is appropriate here and also very efficient because its procedure of calculating eigenvalues and eigenvectors of the Laplacian requires the calculation of only a small (randomly chosen) subset of the weights. In fact, for this method, we only need to compute a very small fraction of the weights, around 0.3 percent, to obtain good accuracy. Now, when r equals zero, the graph Laplacian term is the only other term in (22), in addition to (26), where the graphical structure appears. Due to this, in the case of the second version, one can use the Nyström extension method to efficiently compute the eigenvectors and eigenvalues, which can then be used in (22) to calculate both of the terms, i.e. the Dirichlet's energy (via (30)) and the graph Laplacian. Note that using the Nyström method for the first version is impractical and unnecessary, because the Dirichlet energy is computed via (4), which explicitly uses weights, so all the weights need to be computed anyway, defeating the whole purpose of the Nyström procedure. Overall, the key point is that the second version allows one to proceed very efficiently in the case when the graph is very large, and this is its advantage.

Each of the two versions is designed to be more favorable for a particular kind of graph or data set. When the computation of the entire similarity graph is prohibitive, such as in the case of a very big data set like one involving hyperspectral data, version 2 allows one to use the efficient Nyström extension method to calculate the eigendecomposition of the Laplacian. When the graph is sparse or of a small size, one may choose to calculate the Dirichlet energy directly using the original graph formula (4) for a more numerically accurate calculation. Thus, version 1 is preferred in this case.

For both versions, the goal is to solve (22). To do so, we use the eigendecomposition of a Laplacian and write the terms of (22) as series involving eigenvectors. Let

$$u^n = \sum_k a_k^n \phi_k, \tag{27}$$

where $\{\phi_k(x)\}_k$ are the eigenfunctions of a Laplacian. We use this formula for both versions.

3.3.1 Version 1

To derive version 1, let

$$-\frac{1}{B(u^n)}\Big\{\frac{1}{\epsilon}\mathcal{W}'(u^n) - E(u^n)\nabla B(u^n)\Big\} = \sum_k b_k^n \phi_k, \tag{28}$$

where $\{\phi_k(x)\}_k$ are the eigenfunctions of a Laplacian. Here, we calculate the Dirichlet's energy term (in the GL_ϵ term of $E(u)$) explicitly using (4). This is suited for problems where the data set is small or the known graph is sparse (size of the graph is $O(n)$, where n is the size of the data set), so that the calculation in (4) is not computationally expensive. After plugging (27) and (28) into (22), one obtains

$$a_k^{n+1} = \frac{a_k^n + dtb_k^n}{1 + \frac{2dt\lambda_k}{B(u^n)}}, \tag{29}$$

where λ_k is the k^{th} eigenvalue of a Laplacian, in ascending order. The steps of the method are outlined in Figure 1.

3.3.2 Version 2

Using (26) and the orthonormal property of eigenvectors, one can derive the following equation involving the $||\nabla_w u^n||_{\mathcal{E}}^2$ term:

$$||\nabla_w u^n||_{\mathcal{E}}^2 = \sum_k \lambda_k (a_k^n)^2, \tag{30}$$

where λ_k are the eigenvalues of a Laplacian. In this version, when calculating the right side of (28) to find b^n, we use (30). Thus, in this version, the coefficients b_k^n are defined as:

$$-\frac{1}{B(u^n)}\Big\{\frac{1}{\epsilon}\mathcal{W}'(u^n) - \frac{\nabla B(u^n)}{B(u^n)}(\sum_k \lambda_k (a_k^n)^2 + + \frac{1}{\epsilon}\sum_x \mathcal{W}(u^n))\Big\} = \sum_k b_k^n \phi_k, \tag{31}$$

where we have used (30) to replace the Dirichlet's energy term. Note that Version 2 allows one to handle the situation when computing the entire graph is prohibitive and a low rank approximation of the Laplacian, calculated using a method such as Nyström extension, has to be used. The steps of the method are outlined in Figure 1.

Remark: Note that versions 1 and 2 are theoretically equivalent. In other words, if we write $u^n = \sum_k a_k^n \phi_k(x)$, it can be shown that

$$||\nabla_w u^n||_{\mathcal{E}}^2 = \frac{1}{2}\sum_{x,y} w(x,y)(u^n(x) - u^n(y))^2 = \sum_k \lambda_k (a_k^n)^2. \tag{32}$$

Thus, if the eigenvalues and eigenvectors are computed in the same way, any difference in the answer of the two versions is due to numerical reasons, although the difference is insignificant. Moreover, for both versions, we use

Figure 1: Modified Ratio Cut Method (MRCM)

Modified Ratio Cut Method (MRCM)

Here u will be a function whose thresholded value indicates the class membership. The steps of the algorithm are:

* Organize the data set in a graphical setting, compute eigenvalues and eigenvectors of a chosen Laplacian.
* For Approach I, let $B(u)$ be (21). For Approach II, let $B(u)$ be (25).
* Initialize u^0, calculate a^0 using (27), calculate b^0 using (28) (for Version 1) and (31) (for Version 2).
* Repeat the following steps from $n = 0$ until a stopping criterion is satisfied:
  - Calculate a^{n+1} using (29).
  - Calculate u^{n+1} using (27).
  - For Version 1, calculate b^{n+1} using (28). For Version 2, calculate b^{n+1} using (31). Note the scaling mentioned in the last sentence of section 3.1.
* Threshold the solution to obtain a binary answer, which represents the assigned class.

spectral methods that approximate u as a linear combination of the leading eigenvectors of the graph Laplacian; we use only a small subset of the total number of eigenfunctions for both versions. The only difference in the procedures of the versions is the fact that version 1 uses the *definition*

$$||\nabla_w u^n||_{\mathcal{E}}^2 = \frac{1}{2}\sum_{x,y} w(x,y)(u^n(x) - u^n(y))^2 \tag{33}$$

to calculate the Dirichlet energy, while version 2 uses the following equivalent expression for that task:

$$||\nabla_w u^n||_{\mathcal{E}}^2 = \sum_k \lambda_k (a_k^n)^2. \tag{34}$$

While the last equality in (32) is theoretically true, due to numerics and computer calculations, it might not be exactly fulfilled in practice, although the error will be very small. Therefore, version 1 is included so that the Dirichlet energy is calculated in the most accurate manner. Moreover, each of the two versions is favorable for a particular kind of data or graph. If the data set is very large, version 2 is preferred since it allows one to use the very efficient Nyström procedure to compute the eigenvectors and eigenvalues. If the graph is sparse or of a small size, version 1 provides a way for a more direct and numerically accurate computation by using the original graph formula (4).

4 Modified Cheeger Cut Method

We again consider the binary partitioning problem involving the target set X embedded in a graphical framework. We use similar techniques of the previous sections, but using the Cheeger cut. As described in Section 1, the Cheeger cut problem can be formulated as finding the subset S of X such that the following value is minimized:

$$h(S, \bar{S}) = \frac{cut(S, \bar{S})}{min(|S|, |\bar{S}|)}$$

If u is a binary function (taking values 0 or 1) denoting the class of each of the nodes, one can write the Cheeger problem as

$$\min_{\{u:u(x)\in\{0,1\}\}} \frac{TV_w(u)}{min\big(\langle u,1\rangle,\langle 1-u,1\rangle\big)}. \tag{35}$$

where $\langle,\rangle$ is an inner product. Again, the notation allows us to be general. If the inner product is just the $\mathcal{V}$-inner product with $r=0$, then the problem becomes simply the Cheeger cut. If the $\mathcal{V}$-inner product with $r=1$ is used, then the problem is the conductance.

Unfortunately, this is an NP hard problem, and one approach is to relaxation of constraints. We relax the problem by minimizing over functions with values in $\mathbb{R}$:

$$\min_{u} \frac{TV_w(u)}{min\big(\langle u,1\rangle,\langle 1-u,1\rangle\big)}. \tag{36}$$

Similarly to the procedure of the previous section, we replace the total variation term in (23) by the GL functional, with motivation described in Section 3.1.1, so that the above problem is formulated as:

$$\min_u E(u) = \min_u \frac{GL_\epsilon(u)}{B(u)} = \min_u \frac{GL_\epsilon(u)}{min\big(\langle u,1\rangle,\langle 1-u,1\rangle\big)}. \tag{37}$$

To solve (37), we use the scheme (22) with

$$B(u) = \min\big(\langle u,1\rangle,\langle 1-u,1\rangle\big). \tag{38}$$

In the case of the Cheeger cut method, the inner product in the $B(u)$ term is the $\mathcal{V}$-inner product with $r=0$.

4.1 The Procedure of the Modified Cheeger Cut Method

The two different versions of the modified Cheeger cut method are the same as for the previous method, except, of course, the different definition of $B(u)$. We outline the steps of both versions of the method in Figure 2.

5 Parametrized Laplacian Framework

In [22], Ghosh et al. formulated a general Laplacian framework by considering a dynamic process on graphs:

$$\frac{d\Theta}{dt} = -\mathcal{L}\Theta.$$

Here, Θ is a vector containing the values of the dynamic variable for all vertices, and $\mathcal{L}$ is a positive semi-definite matrix called the spreading operator, which defines the process. The parameterized Laplacian family is defined as

$$\mathcal{L} = T^{-\frac{1}{2}} D'^{-\frac{1}{2}} (D' - BWB) D'^{-\frac{1}{2}} T^{-\frac{1}{2}}.$$

Compared with traditional Laplacian operators, the parametrized Laplacian has two additional parameters: T and B. The diagonal matrix T controls the time delay factors, or local clock rate of a random walk, at each vertex. The bias factors form the other diagonal matrix B. Note that the degree matrix D' is defined in terms of the biased weight matrix: $d'_i = \sum_j [W']_{ij} = \sum_j [BWB]_{ij}$, where we set $W' = BWB$.

Figure 2: Modified Cheeger Cut Method (MCCM)

Modified Cheeger Cut Method (MCCM)

Here u will be a function whose thresholded value indicates the class membership. The steps of the algorithm are:

* Organize the data set in a graphical setting, compute eigenvalues and eigenvectors of a chosen Laplacian.
* Let $B(u)$ be (38).
* Initialize u^0, calculate a^0 using (27), calculate b^0 using (28) (for Version 1) and (31) (for Version 2).
* Repeat the following steps from $n = 0$ until a stopping criterion is satisfied:
  - Calculate a^{n+1} using (29).
  - Calculate u^{n+1} using (27).
  - For Version 1, calculate b^{n+1} using (28). For Version 2, calculate b^{n+1} using (31). Note the scaling mentioned in the last sentence of section 3.1.
* Threshold the solution to obtain a binary answer, which represents the assigned class.

To investigate how different dynamic processes effect our perceptions of the graph structure, we study how the results change when T and B vary. Recall in Section 2.2, we defined a similar family of Laplacians with the parameter r (6). As we will show for the following four special cases, they can be formulated as equivalent operators up to a similarity transformation.

* ***Normalized Laplacian.*** In the case when $T = B = I$, we obtain the symmetric normalized Laplacian:

$$\mathbf{L_s} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}.$$

  The normalized Laplacian is related to the random walk Laplacian ($r = 1$) by a change of basis: $\mathbf{L_s} = D^{\frac{1}{2}}\mathbf{L_{rw}}D^{-\frac{1}{2}}$.

* ***Scaled Graph Laplacian.*** In the case when $B = I$ and $T = d_{max}D^{-1}$ (where d_{max} represents the maximum degree in the degree matrix), the spreading operator is called the scaled graph Laplacian:

$$\mathbf{L_{scl}} = \frac{1}{d_{max}}(D - W).$$

  It is just the matrix (6) with $r = 0$ scaled by $\frac{1}{d_{max}}$.

* ***Unbiased Adjacency Matrix.*** If we let $B = D^{-\frac{1}{2}}$ and $T = d'_{max}D'^{-1}$ (where d'_{max} represents the maximum degree in the degree matrix associated with $W' = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$), we obtain the unbiased adjacency matrix:

$$\mathbf{L_{unb}} = \frac{1}{d'_{max}}(D' - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}) = \frac{1}{d'_{max}}(D' - W').$$

  The matrix is related to (6) with $r = 0$ (using a similarity matrix biased by $D^{-\frac{1}{2}}$) by a factor of $\frac{1}{d'_{max}}$.

* ***Replicator.*** If we let $B = V$, where V is a diagonal matrix whose elements are the components of the eigenvector corresponding to the largest eigenvalue of W, and $T = I$, we obtain

$$\mathbf{L_{rep}} = I - \frac{1}{\lambda_{max}} W,$$

where λ_{max} is the largest eigenvalue of W. The replicator is related to the max entropy random walk by a change of basis: $\mathbf{L_{rep}} = D'^{\frac{1}{2}} D'^{-1}(D' - VWV)D'^{-\frac{1}{2}}$, where $D'^{-1}(D' - VWV)$ is the max entropy random walk.

We utilize these four different formulations of the Laplacian in our paper.

6 Results

In this section, we show results for some benchmark data sets, and then show an application to hyperspectral imagery. We observe that the advantage of the methods introduced in the paper is computational efficiency, while being able to maintain the accuracy of state-of-the-art.

The energy minimization proceeds until a steady state condition is reached. Once the change of the norm of the vector field in subsequent iterations falls below a threshold, the system is no longer evolving and the energy decrement is negligible. In the experiments, obtained on a 2.4 GHz Intel Core i2 Quad computer, the calculation is stopped when

$$\frac{\|u^{n+1} - u^n\|^2_{\mathcal{L}^2}}{\|u^{n+1}\|^2_{\mathcal{L}^2}} < \eta, \tag{39}$$

where the value of η is dependent on the data set.

When choosing a weight function, the goal is to assign a large weight to an edge if the two vertices it is connecting are similar and a small weight otherwise. One popular choice for the weight function is the Gaussian

$$w(x, y) = e^{-\frac{M(x,y)^2}{\sigma^2}}, \tag{40}$$

where $M(x, y)$ is some distance measure between the two vertices x and y, and σ is a parameter to be chosen. For example, if the data set consists of points in $\mathbb{R}^2$, $M(x, y)$ can be chosen to be the Euclidean distance between x and y, since points farther away are less likely to belong to the same cluster than points closer together. For images, $M(x, y)$ can be defined as the weighted L^2-norm of the difference of the feature vectors of pixels x and y, where the feature vector of a pixel can be defined as the set of intensity values in the pixel's neighborhood, as described in [23].

Another choice for the weight function is the Zelnik-Manor and Perona function [40] for sparse matrices:

$$w(x, y) = e^{-\frac{M(x,y)^2}{\sqrt{\tau(x)\tau(y)}}}, \tag{41}$$

using $\tau(x) = M(x, z)^2$, where z is the K^{th} closest vertex to vertex x. The parameter K is to be chosen.

Note that it is not necessary or even desirable to use a fully connected graph setting, which might be a computational burden. Specifically, the fully connected graph can be approximated by a much smaller graph by only including an edge between vertex x and y if x is among the k-nearest neighbors of y or vice versa. This is called a k-nearest

Table 1: Modified Cheeger and ratio cut method results and comparison

MNIST

Method	Accuracy
symmetric Laplacian (MRCM- Approach 1)	98.62%
symmetric Laplacian (MRCM- Approach 2)	98.27%
symmetric Laplacian (MCCM)	98.33%
scaled graph Laplacian (MRCM- Approach 1)	98.45%
scaled graph Laplacian (MRCM- Approach 2)	98.12%
scaled graph Laplacian (MCCM)	98.2%
unbiased adjacency matrix (MRCM- Approach 1)	**98.66%**
unbiased adjacency matrix (MRCM- Approach 2)	98.26%
unbiased adjacency matrix (MCCM)	98.32%
replicator (MRCM- Approach 1)	96.60%
replicator (MRCM- Approach 2)	94.38%
replicator (MCCM)	94.14%
method in [6]	98.36%
Computation of eigenvectors	45 sec.
Timing for our methods	2 to 15 sec.
Timing for method in [6]	52.4 sec.

Two moons

Method	Accuracy
symmetric Laplacian (MRCM- Approach 1)	**98.35%**
symmetric Laplacian (MRCM- Approach 2)	97.80%
symmetric Laplacian (MCCM)	98.00%
scaled graph Laplacian (MRCM- Approach 1)	**98.35%**
scaled graph Laplacian (MRCM- Approach 2)	97.45%
scaled graph Laplacian (MCCM)	97.70%
unbiased adjacency matrix (MRCM- Approach 1)	98.05%
unbiased adjacency matrix (MRCM- Approach 2)	97.55%
unbiased adjacency matrix (MCCM)	97.70%
replicator (MRCM- Approach 1)	97.35%
replicator (MRCM- Approach 2)	95.50%
replicator (MCCM)	95.55%
method in [44]	95.08%
method in [11]	93.5%
method in [26]	95.38%
method in [8]	91.31%
method in [6]	95.86%

neighbor graph. We sparsify the graph in this way, making the final matrix symmetric by setting both $w(x, y)$ and $w(x, y)$ to $\max(w(x, y), w(y, x))$. One can also build a mutual k-nearest neighbor graph by only including an edge between x and y if both of them are k-nearest neighbors of each other, but we do not use this approach. If two vertices x and y are not connected by an edge, the weight between them is set to 0.

With regards to the parameters, they are chosen mostly by trial and error; however, the algorithm seems to be stable for many choices. There also exist many procedures to select parameters [30, 32, 33, 46], although we have not used them in this work. Moreover, we use 0.5 for the threshold, since it is between 0 and 1 (which represent the two classes) and is not "biased" towards a particular class. The results are also not very sensitive to the threshold because they are close to being binary anyway.

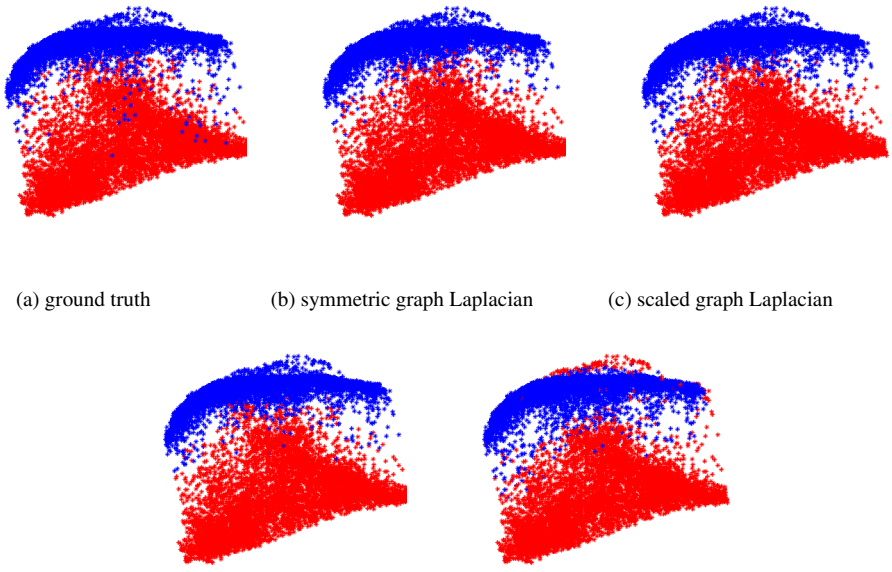
6.1 MNIST Data Set

The MNIST digits data set [31], available at *http://yann.lecun.com/exdb/mnist/*, is a data set of 70000 28×28 images of handwritten digits from $0 - 9$ to be classified into ten classes. Since our method is only binary, we obtained a subset of this set to cluster into two classes; in particular, we chose digits 4 and 9 since these digits are sometimes hard to distinguish, if handwritten. This created a set of 13782 digits, each either 4 or 9. We start with a random initialization of the class, and the goal is to classify each image into either a 4 or 9. The eigenvectors and eigenvalues of the graph

Laplacian were calculated using the Raleigh-Chebyshev procedure of Anderson [1]; we use 100 eigenvectors. The parameters for this data set were $dt = 0.1$ and $\epsilon = 0.1$.

The MNIST data set results of the MCRM (both approaches, version 1) and MCCM methods are shown in Table 1. Our methods are italicized, and the best method is written in bold. From the table, we see that all but the replicator achieve accuracy in the 98^{th} percentile. Using the replicator worsens the accuracy by at least one percentile. The results of MRCM (approach 1, version 1) are visualized in Figure 3. Those of other algorithms are not shown simply because they are also very similar, as are results of the first and second version. Also, note that this data set, the computation of eigenvectors took around 45 seconds, while the minimization part took around 2 to 15 seconds.

To compare to some state-of-the-art work involving the cut with normalization (i.e. Cheeger cut, balanced cut, etc.) or spectral computations, we note the result of 98.36% of Bresson et al. in [6]. We see that our methods achieve accuracy that is very similar to that result, in addition to being very efficient computationally. In fact, after the graph and the eigenvectors were computed, for all algorithms presented in this paper, the minimization took only around 2 to 15 seconds, depending on the number of iterations. The mean time for the computations recorded in [6] is 52.4 seconds, while the algorithm in [8] is twice as fast.



(a) ground truth (b) symmetric graph Laplacian (c) scaled graph Laplacian

(d) unbiased adjacency matrix (e) replicator

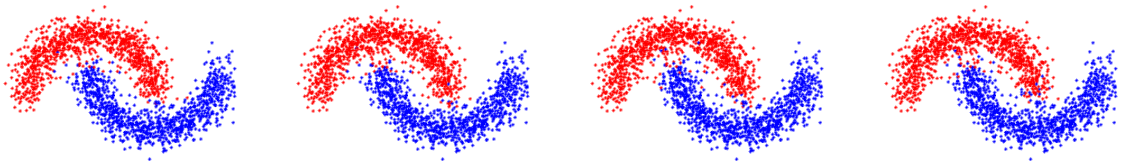
Figure 3: MNIST data set results

6.2 Two Moons Data Set

The data set is constructed from two half circles in $\mathbb{R}^2$ with a radius of one with centers at $(0, 0)$ and $(1, 0.5)$. A thousand uniformly chosen points are sampled from each circle, embedded in $\mathbb{R}^{100}$ and i.d.d. Gaussian noise with standard deviation 0.02 is added to each coordinate, giving a set of two thousand points. Starting from some initial

classification of the points, the goal is to segment the two half circles. Here, we use an initialization involving the thresholded second eigenvector of the chosen Laplacian. The eigenvectors and eigenvalues of the graph Laplacian were calculated using the Raleigh-Chebyshev procedure of Anderson [1]; we use 60 eigenvectors. The parameters for this data set were $dt = 0.7$ and $\epsilon = 0.1$.

The results of the MRCM (both approaches, version 1) and MCCM algorithms are shown in Table 1. The table shows that all but the replicator always achieve accuracies in the 97^{th} or 98^{th} percentile. The results of MRCM (approach 1, version 1) are visualized in Figure 4; those of the rest are not shown because they are very similar, as are results of the first and second version. To compare to some state-of-the-art work involving the cut with normalization (i.e. Cheeger cut, balanced cut, etc.) or spectral computations, we note the 95.08% result of Szlam et al. in [44], the 93.5% result of Buhler et al. in [11], the 95.38% result of Hein et al. in [26], the 91.31% result of Bresson et al. in [8] and the 95.86% result of Bresson et al. in [6]. We see that our method achieves an accuracy that is at least 2% higher. The whole procedure, including the computation of the graph, takes around 4 seconds.



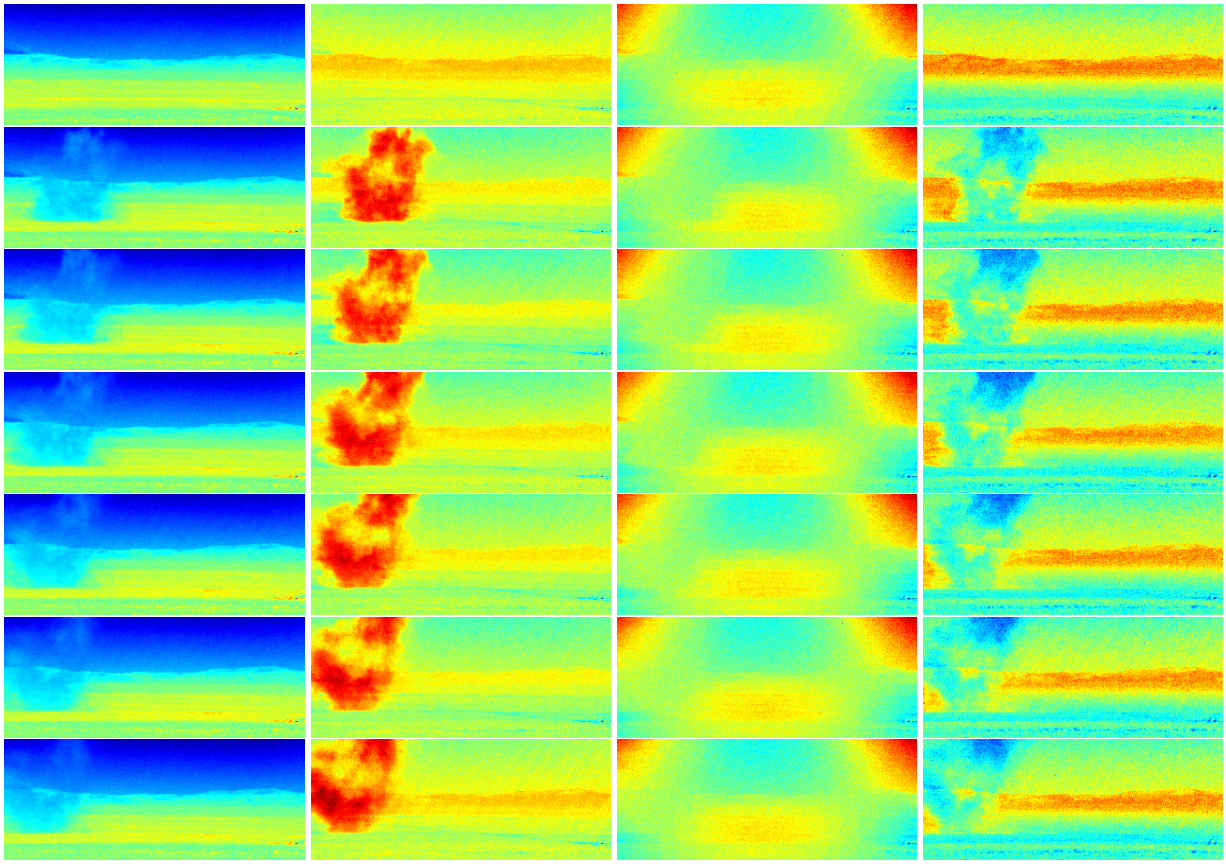
(a) symmetric graph Laplacian (b) scaled graph Laplacian (c) unbiased adjacency matrix (d) replicator

Figure 4: Results on the two moons data set

6.3 Application to Hyperspectral Imagery

We apply our method to hyperspectral data; the goal is to track chemical plumes recorded as a hyperspectral video sequence. The data [10], provided by the Applied Physics Laboratory at John Hopkins University, is a video sequence of plumes released at the Dugway Proving Ground. The images are of dimension $128 \times 320 \times 129$, where the last dimension indicates the number of channels, each depicting a particular frequency from 7,820 nm to 11,700 nm, spaced 30 nm apart. The sets of images were taken from videos captured by three long wave infrared (LWIR) spectrometers, each placed at a different location about 2 km away from the release of plume at an elevation of 1300 feet. One hyperspectral image is captured every five seconds. Other work on this data set can be found in [21], [45] and [43]. Prior work on hyperspectral plume detection using other sensors includes [27] (MWIR) and [34] (HYDICE).

Since our data depends on time, to form a graph on the set, we select k different video frames and then concatenate the points, allowing for the data to be associated over these k frames. Although the resulting graph is very large (for 7 frames, the full graph Laplacian is an $n \times n$ matrix with $n = 286,720$), we use the Nyström extension method to effectively calculate the desired eigenfunctions. The algorithm allows one to calculate an approximation of the eigenfunctions, while only having to compute a small portion of the graph. In this case of a very large graph, we need to use Version 2 of the proposed algorithm, because the goal is to not calculate the Dirichlet's energy terms using the



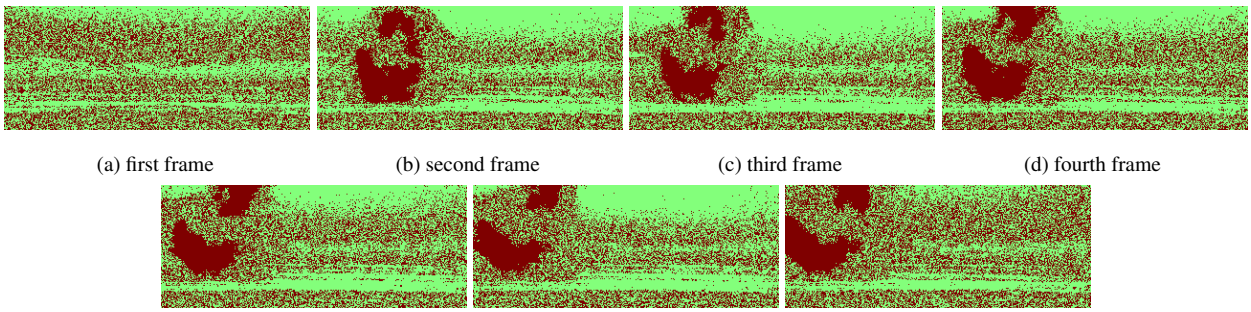
(a) second eigenvector (b) third eigenvector (c) fourth eigenvector (d) fifth eigenvector

Figure 5: Eigenvectors of the symmetric normalized graph Laplacian for a subset of 7 video frames shown in false color.

explicit formula (4), which would be computationally inefficient, but use the eigenfunctions of the Laplacian and (30). We use 100 eigenvectors. The parameters for this data set were $dt = 0.8$ and $\epsilon = 0.1$.

Figure 5 shows a sampling of four different eigenvectors for a subset of 7 frames. Note that each eigenvector highlights a different aspect of the image. To produce a good initialization, we use an operator assisted method involving spectral clustering [38]. In particular, by thresholding appropriately the values of eigenvectors, one obtains information about a particular class. For example, as shown in Figure 5, the third eigenvector highlights the plume. By thresholding its values, one can find the pixels that are most likely part of the plume.

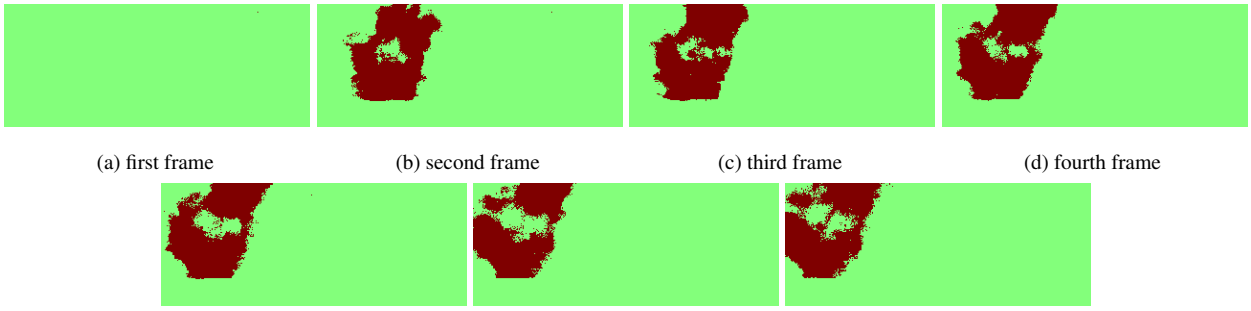
We tested our method on several video frames. The initialization for 7 of the frames is displayed in Figure 6. The final segmentation results (for MCCM, version 2, $r = 0$) for these frames, after 141 iterations, are shown in Figure 7. The plume is outlined in dark red, and the background shown in light green. This experiment took around 23 seconds after the graph was computed.



(a) first frame (b) second frame (c) third frame (d) fourth frame

(e) fifth frame (f) sixth frame (g) seventh frame

Figure 6: Plume data set initialization



(a) first frame (b) second frame (c) third frame (d) fourth frame

(e) fifth frame (f) sixth frame (g) seventh frame

Figure 7: Plume data set results

7 Conclusion

This paper introduces new methods that utilize the graph Laplacian to solve the fully nonlinear Cheeger and (modified) ratio cut problems, with all algorithms making use of the Ginzburg-Landau functional. To make spectral and graph computations efficient, fast numerical routines are used. The experiments, which include an application to hyperspectral data, show that the algorithms produce results that are comparable with or better than some of the best methods, and very efficiently. However, there is more work to be done; we plan on extending the two procedures to the multiclass case, and finding other interesting applications.

Acknowledgment

This work was supported by ONR grant N00014-16-1-2119, AFOSR MURI grant FA9550-10-1-0569, NSF grants DMS-1417674, DMS-1118971 and CIF-1217605, UC Lab Fees Research grant 12-LR-236660 and DARPA W911NF-12-1-0034. Ekaterina Merkurjev is also supported by the UC President's Postdoctoral Fellowship.

Appendix

Calculation of the Eigenvalues/Eigenvalues of the Graph Laplacian

Our method involves the computation of eigenvalues and associated eigenvectors of graph Laplacians. In practice, one needs to compute only a fraction of the eigenvalues and eigenvectors because only a small portion of the nodes are significant. When the graph is sparse and is of moderate size, around 5000×5000 or less, we use a Rayleigh-Chebyshev procedure outlined in [1]. It is a modification of an inverse subspace iteration method that uses adaptively determined Chebyshev polynomials. When the graph is very large, the Nyström extension method [18, 19] can be used. It is a matrix completion method, and is often used in many applications, such as kernel principle component analysis [17] and spectral clustering [39]. This procedure is especially efficient because it uses approximations based on calculations on very small submatrices of the original matrix; therefore, it is useful when the graph is very large.

Here, we show how to apply the Nyström extension method in the case of the symmetric Laplacian, but it can be easily applied to other choices. Note that the eigenvectors of the matrix $\hat{W} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ are the same as those of the symmetric graph Laplacian, and their eigenvalues have a very simple relationship: λ is an eigenvalue of the symmetric Laplacian if and only if $1 - \lambda$ is an eigenvalue of $\hat{W}$. Below, we formulate a method to calculate the eigenvectors and eigenvalues of $\hat{W}$ and thus of L_s.

Let w be the weight function, λ be an eigenvalue of W, and ϕ its associated eigenvector. The Nyström method approximates the eigenvalue equation

$$\int_\Omega w(y,x)\phi(x)dx = \lambda\phi(x)$$

using a quadrature rule, a technique to find weights $\{c_j(y)\}$ and a set of l interpolation points $X = \{x_j\}$ such that

$$\sum_{j=1}^{l} c_j(y)\phi(x_j) = \int_\Omega w(y,x)\phi(x)dx + E(y,X)$$

where $E(y,X)$ represents the error in the approximation. We use $c_j(y) = w(y,x_j)$ and choose the l interpolation points randomly from the vertex set V. Denote the set of l randomly chosen points by $X = \{x_i\}_{i=1}^{l}$ and its complement by Y. Partioning Z into $Z = X \cup Y$ and letting $\phi_k(x)$ be the the k^{th} eigenvector of W and λ_k its associated eigenvalue, we obtain the system of equations

$$\sum_{x_j \in X} w(y_i,x_j)\phi_k(x_j) = \lambda_k\phi_k(y_i) \qquad \forall y_i \in Y, \quad \forall k \in 1,...,l.$$

One cannot solve these equations directly because ϕ_k are unknown. However, the eigenvectors of W are approximated using submatrices of W. Let W_{XY} be defined as

$$\begin{bmatrix} w(x_1,y_1) & \dots & w(x_1,y_{n-l}) \\ \vdots & \ddots & \vdots \\ w(x_l,y_1) & \dots & w(x_l,y_{n-l}) \end{bmatrix}$$

The matrices W_{YX}, W_{XX} and W_{YY} are defined similarly. The matrix W is then

$$\begin{bmatrix} W_{XX} & W_{XY} \\ W_{YX} & W_{YY} \end{bmatrix}$$

We normalize the above weight matrix to calculate the eigenvalues and eigenvectors of $\hat{W}$. Here, $\mathbf{1}_K$ is the K-dimensional unit vector. The matrices d_X, d_Y, $\hat{W}_{XX}$ and $\hat{W}_{XY}$ are defined as

$$\begin{aligned} d_X &= W_{XX}\mathbf{1}_l + W_{XY}\mathbf{1}_{n-l}, \\ d_Y &= W_{YX}\mathbf{1}_l + (W_{YX}W_{XX}^{-1}W_{XY})\mathbf{1}_{n-l}, \\ \hat{W}_{XX} &= W_{XX}./(s_X s_X^T), \\ \hat{W}_{XY} &= W_{XY}./(s_X s_X^Y), \end{aligned}$$

where $A./B$ denotes componentwise division between matrices A and B, v^T denotes the transpose of vector v, $s_X = \sqrt{d_X}$ and $s_Y = \sqrt{d_Y}$. It is shown in [3] that if $\hat{W}_{XX} = B_X D B_X^T$, and if A and Γ are matrices such that $A^T \Gamma A = \hat{W}_{XX} + \hat{W}_{XX}^{-\frac{1}{2}} \hat{W}_{XY} \hat{W}_{YX} \hat{W}_{XX}^{-\frac{1}{2}}$, then the eigenvector matrix V consisting of approximations of l eigenvectors of $\hat{W}$, and thus of the symmetric graph Laplacian, is

$$\begin{bmatrix} B_X D^{\frac{1}{2}} B_X^T A \Gamma^{-\frac{1}{2}} \\ \hat{W}_{YX} B_X D^{-\frac{1}{2}} B_X^T A \Gamma^{-\frac{1}{2}} \end{bmatrix}$$

while $I - \Gamma$ contains the corresponding eigenvalues of the symmetric graph Laplacian in its diagonal entries.

One can see that the Nyström extension method is efficient because it approximates the eigenvalues and eigenvectors of an $n \times n$ matrix (n is large) by calculations using matrices that are much smaller, having dimension no larger than $n \times l$, where l is small. We also note that the method calculates l eigenvalues and corresponding eigenvectors; in practice, only a very small portion (we use around 0.3%) of eigenvalues and eigenvectors of the graph Laplacian are needed to obtain an accurate solution.

References

[1] C. Anderson. A Rayleigh-Chebyshev procedure for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices. *Journal of Computational Physics*, 229:7477–7487, 2010.

[2] E. Bae and E. Merkurjev. Convex variational methods for multiclass data segmentation on graphs. submitted, 2016.

[3] A.L. Bertozzi and A. Flenner. Diffuse interface models on graphs for classification of high dimensional data. *Multiscale Modeling & Simulation*, 10(3):1090–1118, 2012.

[4] F. Bethuel, H. Brezis, and F. Hélein. Asymptotics for the minimization of a Ginzburg-Landau functional. *Calculus of Variations and Partial Differential Equations*, 1(2):123–148, 1993.

[5] F. Bethuel, H. Brezis, and F. Hélein. *Ginzburg-Landau Vortices*, volume 13. Springer Science & Business Media, New York, New York, 2012.

[6] X. Bresson, T. Laurent, D. Uminsky, and J. von Brecht. Convergence and energy landscape for Cheeger cut clustering. *Advances in Neural Information Processing Systems*, 25:1385–1393, 2012.

[7] X. Bresson, T. Laurent, D. Uminsky, and J. von Brecht. Multiclass total variation clustering. *Advances in Neural Information Processing Systems*, 26:1421–1429, 2013.

[8] X. Bresson, T. Laurent, D. Uminsky, and J.H. von Brecht. An adaptive total variation algorithm for computing the balanced cut of a graph. *arXiv preprint arXiv:1302.2717*, 2013.

[9] X. Bresson, X.-C. Tai, T.F. Chan, and A. Szlam. Multi-class transductive learning based on L1 relaxations of Cheeger cut and Mumford-Shah-Potts model. *Journal of Mathematical Imaging and Vision*, 49(1):191–201, 2014.

[10] J.B. Broadwater, D. Limsui, and A.K. Carr. A primer for chemical plume detection using LWIR sensors. Technical report, National Security Technology Department, John Hopkins Applied Physics Laboratory, 2011.

[11] T. Bühler and M. Hein. Spectral clustering based on the graph p-Laplacian. *International Conference on Machine Learning*, pages 81–88, 2009.

[12] T.F. Chan, G.H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM journal on scientific computing*, 20(6):1964–1977, 1999.

[13] T.F. Chan, A. Marquina, and P. Mulet. High-order total variation-based image restoration. *SIAM Journal on Scientific Computing*, 22(2):503–516, 2000.

[14] T.F. Chan and C.-K. Wong. Total variation blind deconvolution. *IEEE Transactions on Image Processing*, 7(3):370–375, 1998.

[15] F. Chung. *Spectral Graph Theory*, volume 92. AMS Bookstore, Providence, Rhode Island, 1997.

[16] F. Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007.

[17] P. Drineas and M.W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *The Journal of Machine Learning Research*, 6:2153–2175, 2005.

[18] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

[19] C. Fowlkes, S. Belongie, and J. Malik. Efficient spatiotemporal grouping using the Nyström method. *Computer Society Conference on Computer Vision and Pattern Recognition*, 1:I–231, 2001.

[20] C. Garcia-Cardona, E. Merkurjev, A.L. Bertozzi, A. Flenner, and A. Percus. Fast multiclass segmentation using diffuse interface methods on graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1600–1613, 2014.

[21] 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. *SPIE Conference on Defense Security and Sensing*, 8743:87430J–87430J, 2013.

[22] R. Ghosh, S.-H. Teng, K. Lerman, and X. Yan. The interplay between dynamics and networks: centrality, communities, and cheeger inequality. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1406–1415, 2014.

[23] G. Gilboa and S. Osher. Nonlocal operators with applications to image processing. *Multiscale Modeling & Simulation*, 7(3):1005–1028, 2008.

[24] T. Goldstein and S. Osher. The split Bregman method for L_1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

[25] M. Hein, J. Audibert, and U. Von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. *International Conference on Computational Learning Theory*, pages 470–485, 2005.

[26] M. Hein and T. Bühler. An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA. *Advances in Neural Information Processing Systems*, 23:847–855, 2010.

[27] M. Hinnrichs, J. O. Jensen, and G. McAnally. Handheld hyperspectral imager for standoff detection of chemical and biological aerosols. *Optical Technologies for Industrial, Environmental, and Biological Sensing*, 5268:67–78, 2004.

[28] 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 Journal of Applied Mathematics*, 73(6):2224–2246, 2013.

[29] R.L. Jerrard and H.M. Soner. Limiting behavior of the Ginzburg-Landau functional. *Journal of Functional Analysis*, 192(2):524–561, 2002.

[30] Ron Kohavi and George H John. Automatic parameter selection by minimizing estimated error. In *ICML*, pages 304–312, 1995.

[31] Y. LeCun and C. Cortes. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

[32] L. Lindawati, H. Lau, and F. Zhu. Instance-specific parameter tuning via constraint-based clustering. *Proc. of the 1st Int. Workshop on Combining Constraint Solving with Mining and Learning*, 2012.

[33] L. Lindawati, H. C. Lau, and D. Lo. Instance-based parameter tuning via search trajectory similarity clustering. In *International Conference on Learning and Intelligent Optimization*, pages 131–145. Springer, 2011.

[34] D. Manolakis, C. Siracusa, and G. Shaw. Adaptive matched subspace detectors for hyperspectral imaging applications. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 5:3153–3156, 2001.

[35] E. Merkurjev, E. Bae, A.L. Bertozzi, and X.-C. Tai. Global binary optimization on graphs for classification of high-dimensional data. *J. Math Imaging Vis.*, 52(3):414–435, 2015.

[36] E. Merkurjev, C. Garcia-Cardona, A.L. Bertozzi, A. Flenner, and A.G. Percus. Diffuse interface methods for multiclass segmentation of high-dimensional data. *Appl. Math. Lett.*, 33:29–34, 2014.

[37] E. Merkurjev, T. Kostic, and A.L. Bertozzi. An MBO scheme on graphs for classification and image processing. *SIAM Journal on Imaging Sciences*, 6(4):1903–1930, 2013.

[38] E. Merkurjev, J. Sunu, and A.L. Bertozzi. Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video. *IEEE International Conference on Image Processing*, pages 689–693, 2014.

[39] M.M. Naeini, G. Dutton, K. Rothley, and G. Mori. Action recognition of insects using spectral clustering. *IAPR Conference on Machine Vision Applications*, pages 1–4, 2007.

[40] P. Perona and L. Zelnik-Manor. Self-tuning spectral clustering. *Advances in Neural Information Processing Systems*, 17:1601–1608, 2004.

[41] L.I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[42] D. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. *STOC '04 Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90, 2004.

[43] J. Sunu, J.M. Chang, and A.L. Bertozzi. Simultaneous spectral analysis of multiple video sequence data for LWIR gas plumes. *SPIE Conference on Defense, Security, and Sensing*, 9088:90880T–90880T, 2014.

[44] A. Szlam and X. Bresson. Total variation, Cheeger cuts. *International Conference on Machine Learning*, pages 1039–1046, 2010.

[45] G. Tochon, J. Chanussot, J. Gilles, M. Dalla Mura, J.-M. Chang, and A.L. Bertozzi. Gas plume detection and tracking in hyperspectral video sequences using binary partition trees. *IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, 2014.

[46] V. Torra, Y. Endo, and S. Miyamoto. Computationally intensive parameter selection for clustering algorithms: The case of fuzzy c-means with tolerance. *International Journal of Intelligent Systems*, 26(4):313–322, 2011.

[47] Y. van Gennip and A.L. Bertozzi. Gamma-convergence of graph Ginzburg-Landau functionals. *Advanced in Differential Equations*, 17(11–12):1115–1180, 2012.

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

[49] D. Wagner and F. Wagner. *Between min cut and graph bisection*. Springer, 1993.