Fast Multiclass Segmentation using Diffuse Interface Methods on Graphs

Cristina Garcia-Cardona, Ekaterina Merkurjev, Andrea L. Bertozzi, Arjuna Flenner and Allon G. Percus

***Abstract*—We present two graph-based algorithms for multiclass segmentation of high-dimensional data. The algorithms use a diffuse interface model based on the Ginzburg-Landau functional, related to total variation compressed sensing and image processing. A multiclass extension is introduced using the Gibbs simplex, with the functional's double-well potential modified to handle the multiclass case. The first algorithm minimizes the functional using a convex splitting numerical scheme. The second algorithm is a uses a graph adaptation of the classical numerical Merriman-Bence-Osher (MBO) scheme, which alternates between diffusion and thresholding. We demonstrate the performance of both algorithms experimentally on synthetic data, grayscale and color images, and several benchmark data sets such as MNIST, COIL and WebKB. We also make use of fast numerical solvers for finding the eigenvectors and eigenvalues of the graph Laplacian, and take advantage of the sparsity of the matrix. Experiments indicate that the results are competitive with or better than the current state-of-the-art multiclass segmentation algorithms.**

***Index Terms*—segmentation, Ginzburg-Landau functional, diffuse interface, MBO scheme, graphs, convex splitting, image processing, high-dimensional data.**

I. Introduction

Multiclass segmentation is a fundamental problem in computer vision and machine learning. We present a multiclass generalization of the graph-based segmentation procedure introduced in [5]. The model applies L_2 gradient flow minimization of the Ginzburg-Landau (GL) diffuse interface energy functional to the case of functions defined on graphs. The algorithm in [5] performs binary segmentations in a semi-superised learning (SSL) framework, but multiclass tasks are solved by recursively applying a sequence of binary segmentations. Our new method rapidly and simultaneously segments multiclass benchmark data with higher accuracy than most previously existing methods, while avoiding the recurrent applications of the minimization procedure.

This new formulation uses a phase-field representation of the GL energy functional: a vector-valued quantity is assigned to every data point such that each of its components represents the fraction of the phase, or class, present in that particular point. The components of the field variable add up to one, so the phase-field vector is constrained to lie on the Gibbs simplex. The phase-field representation is used in the material sciences to study the evolution of multi-phase systems [26] and has been employed before for multiclass image segmentation [36]. Nevertheless, to the best of our knowledge, this is the first time that a vector field GL representation in the context of functions defined on graphs is applied for multiclass semi-supervised classification of high dimensional data.

In addition, we apply this Gibbs simplex idea to the graph-based Merriman-Bence-Osher (MBO) scheme developed in [38]. The MBO scheme [39] is a well-established PDE method for evolving an interface by mean curvature. As with the diffuse interface model, tools for nonlocal calculus [27] are used in [38] to generalize the PDE formulation to the graph setting. By introducing the phase-field representation to the graph-based MBO scheme, we develop another new and highly efficient algorithm for multiclass segmentation in a SSL framework.

Thus, the main contributions of our work are: (i) to introduce two new graph-based methods for multiclass data segmentation, namely a multiclass GL formulation based on the binary representation described in [5] and a multiclass graph-based MBO that extends the model in [38]; and (ii) to present very efficient algorithms to segment multiclass geometric, image, and text data.

The paper is organized as follows. In section II we discuss prior related work. In section III, we discuss the fundamentals of diffuse interface models in machine learning. We then describe our two algorithms in sections IV and V. In section VI, we present experimental results on benchmark data sets, demonstrating the effectiveness of our methods. Finally, in section VII, we conclude and discuss ideas for future work.

II. Previous Work

The discrete graph formulation of GL energy minimization may be understood as an example of the more general form of energy (or cost) functional for data classification used in the context of machine learning,

$$E(\psi) = ||\psi||_a + \mu||\psi - \hat{\psi}||_b^p, \tag{1}$$

where the norm $||\psi||_a$ of the classification function ψ is a regularization term, and $||\psi - \hat{\psi}||_b^p$ is a fidelity term, incorporating most (supervised) or just a few (semi-supervised) of the known values $\hat{\psi}$. The choice of the regularization norm $||\cdot||_a$ has non-trivial consequences in the final classification accuracy. In instances where an L_2 norm is used, the resulting cost functional is a tradeoff between accuracy in the classification of given labels vs. function smoothness. Although this encodes the reasonable assumption that the labels should vary smoothly almost everywhere, it is also important to preserve

C. Garcia-Cardona and A. G. Percus are with the Institute of Mathematical Sciences at Claremont Graduate University. E-mail: {cristina.garciacardona,allon.percus}@cgu.edu.

A. L. Bertozzi and E. Merkurjev are with the Department of Mathematics at University of California, Los Angeles. Email: {kmerkurev,bertozzi}@math.ucla.edu.

A. Flenner is with the Naval Air Warfare Center, China Lake. E-mail: arjuna.flenner@navy.mil.

the sharp discontinuities that may arise in the boundaries between classes. Hence the interest in formulations that can produce piecewise smooth solutions [8].

Graph-based regularization terms, expressed in terms of the discrete Laplace operator on graphs, are often used in simi-supervised learning as a way to exploit underlying similarities in the data set [3], [13], [52], [54]–[56]. Additionally, some of these methods use a matrix representation to generalize eq. (1) to the multiple class case [13], [52], [54], [56]. The rows in the matrix correspond to graph vertices and the columns to indicator functions for class membership: the class membership for vertex i is computed as the column with largest component in the ith row. The resulting minimization procedure is akin to multiple relaxed binary classifications running in parallel. This representation is different from the Gibbs simplex we use, as there is usually no requirement that the elements in the row add up to 1.

An alternative regularization method for the graph-based multiclass setup is presented in [47], where the authors minimize a Kullback-Leibler divergence function between discrete probability measures that translates into class membership probabilities.

Not all the methods deal directly with the multiple classes in the data set. A different approach is to reduce the multiclass case to a series of two-class problems and to combine the sequence of resulting sub-classifications. Strategies employed include recursive partitioning, hierarchical classification and binary encodings, among others. For example, Szlam and Bresson present a method involving Cheeger cuts and split Bregman iteration [28] to build a recursive partitioning scheme in which the data set is repeatedly divided until the desired number of classes is reached. In [29], a pairwise coupling is described, in which each two-class problem is solved and then a class decision is made combining the decisions of all the subproblems. In [19], a binary encoding approach of the class labels is used.

Our methods, on the other hand, have roots in the continuous setting as they are derived via a variational formulation. Our first method comes from a variational formulation of the L_2 gradient flow minimization of the GL functional [5]. Our second method is built upon the MBO classical scheme to evolve interfaces by mean curvature [39]. The latter has connections with the work presented in [22], where an MBO-like scheme is used for image segmentation. The method is motivated by the propagation of the Allen-Cahn equation with a forcing term, obtained by applying gradient descent to minimize the GL functional with a fidelity term.

Alternative variational principles have also been used for image segmentation. In [36], a multiclass labeling for image analysis is carried out by a multidimensional total variation formulation involving a simplex-constrained convex optimization. In that work, a discretization of the resulting PDEs is used to numerically solve for the minimization of the energy. In contrast, the discretization in both of our algorithms is given by posing the problem in a graph setting. We represent the data as nodes in a weighted graph, with each edge assigned a measure of similarity between the vertices it is connecting. Nonlocal calculus, such as that outlined in [27], is the tool used to generalize the continuous formulation to a discrete setting given by functions on graphs. This approach has successfully been used in other areas such as spectral graph theory [15], [40].

As pointed out in [5], there are interesting connections between the GL functional on graphs and normalized graph cuts. Shi and Malik [45] pose the problem of image segmentation as the solution of a generalized eigensystem generated from a graph Laplacian. In [8], graph cuts are used to efficiently find local minima of a wide class of energies with various smoothness constraints for multiclass image restoration. Also, as mentioned earlier, the method in [48] is a recursive graph-based partition scheme. A multiclass algorithm for the transductive learning problem in high-dimensional data classification, described in [10], is based on ℓ^1 relaxation of the Cheeger cut and the piecewise constant Mumford-Shah or Potts models. In [9], rigorous convergence results are presented for two algorithms that solve the relaxed Cheeger cut minimization used for unsupervised data clustering are presented. Our proposed methods are related to some of these approaches, but use the graph Ginzburg-Landau functional framework.

In the continuous setting, it can be shown that the GL is a diffuse interface approximation to the total variational functional [21], [33], and analogous results have recently been proved in the graph setting as well [6]. This function is a natural framework for producing smooth labels everywhere while preserving sharp discontinuities, with the sharpness controlled by a diffuse interface parameter. The advantage of the diffuse interface model is that the energy functional is more tractable, and can be minimized by simpler numerical methods.

III. Segmentation using the Ginzburg-Landau Functional on Graphs

A. *Ginzburg Landau Functional and Diffuse Interface Model*

The classical Ginzburg-Landau (GL) functional, originally proposed to describe physical phenomena such as liquid-gas transitions and superconductivity [46], can be written as

$$GL(u) = \frac{\epsilon}{2}\int |\nabla u|^2 dx + \frac{1}{\epsilon}\int \Phi(u)dx, \qquad (2)$$

where u is a scalar field defined over a space of arbitrary dimensionality and representing the state of the phases in the system, ∇ denotes the spatial gradient operator, $\Phi(u)$ is a double-well potential, such as $\Phi(u) = \frac{1}{4}(u^2-1)^2$ and ϵ is a positive constant. In the next subsection we derive in detail the proper formulation in a graph setting (see eq. (12)).

The functional (2) is composed of two terms: a smoothing term that measures the differences in the components of the field, and a potential term that measures how far each component is from a specific value (± 1 in the example above). Consequently, the minimization of the first term leads to smoother regions, while the minimization of the second penalizes variations from the minima of the double-well potential. Given initial conditions with states $+1$ and states -1 distributed randomly in the domain, the minimization of the GL functional entails an inherent conflict between the two terms in the functional, leading to the generation of a transition region: the diffuse interface. The smoothness of this diffuse interface is regulated by the parameter ϵ. For small ϵ, the state minimizing the functional contains sharp transitions between the minima of the double-well potential, while a large ϵ gives more weight to the smoothing term so that the transitions are more gradual.

It is shown in [33] that the $\epsilon \to 0$ limit of the GL functional, in the sense of Γ-convergence, is the Total Variation (TV) semi-norm, so one can write:

$$GL(u) \to_{\Gamma} ||u||_{TV}. \tag{3}$$

The advantage of the GL functional is that its L_2 gradient flow leads to a linear differential operator, which allows us to use spectral methods for minimization.

The diffuse interface description, with its controlled smoothness transition between phases, is attractive for segmentation problems due to its straightforward way of modeling the separation of a domain into regions or phases. It has been used successfully in image impainting [7], [20] and image segmentation [22]. The standard practice is to introduce an additional term in the energy functional to escape from trivial steady-state solutions (e.g., all labels taking on the same value). This leads to the expression

$$E(u) = GL(u) + F(u, \hat{u}), \tag{4}$$

where F is the additional term, usually called *fidelity*. This fidelity term allows the specification of any known information about the particular task at hand, for example, regions of an image that belong to a certain class.

B. Application to Semi-Supervised Learning (SSL) on Graphs

Inspired in part by the PDE-based imaging community, where variational algorithms combining ideas from spectral methods on graphs with nonlinear edge detection methods are common [27], Bertozzi and Flenner extended in [5] the L_2 gradient flow of the Ginzburg-Landau (GL) energy functional to the domain of functions on a graph.

The energy $E(u)$ in (4) can be minimized in the L_2 sense using gradient descent. This leads to the following dynamic equation (*modified Allen-Cahn equation*):

$$\frac{\partial u}{\partial t} = -\frac{\delta GL}{\delta u} - \mu \frac{\delta F}{\delta u} = \epsilon \Delta u - \frac{1}{\epsilon}\Phi'(u) - \mu \frac{\delta F}{\delta u} \tag{5}$$

where Δ represents the Laplacian operator. A local minimizer of the energy is obtained by evolving this expression to steady state. Note that E is not convex, and may have multiple local minima.

Before continuing further, let us introduce some simple graph concepts we will be using in subsequent sections.

1) Graph Framework for Large Data Sets: Let G be an undirected graph $G = (V, E)$, where V and E are the sets of vertices and edges, respectively. The vertices are the building blocks of the data set, such as points in $\mathbb{R}^n$ or pixels in an image. The similarity between vertices i and j is measured by a weight function $w(i, j)$ that satisfies the symmetric property $w(i, j)$= $w(j, i)$. A large value of $w(i, j)$ indicates that vertices i and j are similar to each other, while a small $w(i, j)$ indicates that they are dissimilar. For example, an often used similarity measure is the Gaussian function $w(i, j) = \exp(-d(i, j)^2/\sigma^2)$, with $d(i, j)$ representing the Euclidean distance between the points associated with vertices i and j, and σ^2 a positive parameter.

Define $\mathbf{W}$ as the matrix $W_{ij} = w(i, j)$, and define the degree of a vertex $i \in V$ as

$$d_i = \sum_{j \in V} w(i, j). \tag{6}$$

If $\mathbf{D}$ is the diagonal matrix with elements d_i, then the graph Laplacian is defined as the matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$.

2) Ginzburg-Landau Functional on Graphs: Instead of the continuous domain of the original functional, the diffuse interface model of Bertozzi and Flenner expresses the L_2 flow for the discrete graph structure. Thus the graph serves as the domain in which the energy minimization takes place.

Nonlocal calculus, such as that outlined in [27], provides a way to generalize the continuous GL formulation to the case of weighted graphs. It also shows that the Laplace operator is related to the graph Laplacian matrix defined above, and that the eigenvectors of the discrete Laplacian converge to the eigenvectors of the Laplacian [5]. However, to guarantee convergence to the continuum differential operator in the limit of large sample size, the matrix $\mathbf{L}$ must be correctly scaled [5]. Although there are several ways of doing this, the two methods used more often are the random walk Laplacian

$$\mathbf{L_{\boldsymbol{w}}} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}, \tag{7}$$

related to Markov processes [51], and the symmetric normalized Laplacian

$$\mathbf{L_{\boldsymbol{s}}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}. \tag{8}$$

Since the symmetric property of $\mathbf{L_{\boldsymbol{s}}}$ allows for more efficient implementations, we will use this operator in all our computations.

Note that $\mathbf{L_{\boldsymbol{s}}}$ satisfies:

$$(\mathbf{L_{\boldsymbol{s}}}\boldsymbol{u})_i = \frac{1}{\sqrt{d_i}} \sum_j w(i, j) \left(\frac{u_i}{\sqrt{d_i}} - \frac{u_j}{\sqrt{d_j}} \right) \tag{9}$$

$$\langle \boldsymbol{u}, \mathbf{L_{\boldsymbol{s}}}\boldsymbol{u} \rangle = \frac{1}{2} \sum_{i,j} w(i, j) \left(\frac{u_i}{\sqrt{d_i}} - \frac{u_j}{\sqrt{d_j}} \right)^2 \tag{10}$$

for all $\boldsymbol{u} \in \mathbb{R}^n$. Here the subscript i refers to the i^{th} coordinate of the vector, and the brackets denote the standard dot product. Note also that $\mathbf{L_{\boldsymbol{s}}}$ has nonnegative, real-valued eigenvalues, including 0.

Likewise, it is important to point out that for tasks such as data classification, the use of a graph representation has the advantages of providing a way to deal with nonlinearly separable classes as well as simplifying the processing of high dimensional data.

The GL functional on graphs is then expressed as

$$GL(\boldsymbol{u}) = \frac{\epsilon}{2}\langle \boldsymbol{u}, \mathbf{L_{\boldsymbol{s}}}\boldsymbol{u} \rangle + \frac{1}{4\epsilon} \sum_{i \in V} \left(u_i^2 - 1 \right)^2, \tag{11}$$

where u_i is the (real-valued) state of node i. The first term replaces the gradient term in (2), and the second term is the double-well potential function with minima at ± 1, appropriate for binary classifications.

3) Semi-Supervised Learning (SSL) on Graphs: In the graph version of the GL functional, and in general in graph-based learning methods, the graph is constructed such that the edges represent the similarities in the data set and the nodes have an associated real state that encodes, with an appropriate thresholding operation, class membership.

In addition, in some data sets, the label of a small fraction of data points is known beforehand. The possibility of using this a priori information considerably improves the learning accuracy, explaining in part the popularity of semi-supervised learning (SSL) methods. The graph generalization of the diffuse interface model handles this condition effortlessly by using the labels of known points. The GL functional for SSL is expressed as:

$$\begin{aligned} E(\boldsymbol{u}) \;&=\; \frac{\epsilon}{2}\langle \boldsymbol{u}, \mathbf{L}_{\boldsymbol{s}}\boldsymbol{u}\rangle + \frac{1}{4\epsilon}\sum_{i\in V}\left(u_i^2-1\right)^2 \\ &\quad + \sum_{i\in V}\frac{\mu_i}{2}\left(u_i-\hat{u}_i\right)^2 . \end{aligned} \tag{12}$$

The final term in the sum is the new fidelity term that enforces those label values that are known beforehand. μ_i is a parameter that takes the value of a positive constant μ if i is a fidelity node and zero otherwise, and $\hat{u}_i$ is the known value of fidelity node i.

Therefore, given an initial state u_i of each vertex i, the problem consists of minimizing the GL functional with fidelity term. The classes are then obtained by thresholding the values of u_i to the closer of either 1 or -1. The result is a very efficient method for binary data segmentation. In the following sections, the modifications introduced to generalize this functional to multiclass segmentation are described.

IV. Multiclass Ginzburg-Landau Approach

A. Extension to Multiclass Segmentation

Given N_D data points, we generalize the label vector $\boldsymbol{u}$ to a label matrix $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_{N_D})^T$. Rather than node i adopting a single state $u_i \in \mathbb{R}$, it now adopts a composition of states expressed by a vector $\mathbf{u}_i \in \mathbb{R}^K$ where the kth component of $\mathbf{u}_i$ is the strength with which it takes on class k. The matrix $\mathbf{U}$ has dimensions $N_D \times K$, where K is the total number of possible classes.

For each node i, we require the vector $\mathbf{u}_i$ to be an element of the Gibbs simplex Σ^K, defined as

$$\Sigma^K := \left\{ (x_1, \ldots, x_K) \in [0,1]^K \;\middle|\; \sum_{k=1}^{K} x_k = 1 \right\}. \tag{13}$$

Vertex k of the simplex is given by the unit vector $\boldsymbol{e}_k$, whose kth component equals 1 and all other components vanish. These vertices correspond to pure phases, where the node belongs exclusively to class k. Note that the simplex formulation has a straightforward probabilistic interpretation, with $\mathbf{u}_i$ representing the probability distribution over the K classes.

The multiclass GL energy functional for the phase field approach on graphs is written as:

$$\begin{aligned} E(\mathbf{U}) \;&=\; \frac{\epsilon}{2}\langle \mathbf{U}, \mathbf{L}_{\boldsymbol{s}}\mathbf{U}\rangle + \frac{1}{2\epsilon}\sum_{i\in V}\left(\prod_{k=1}^{K}\frac{1}{4}\left\|\mathbf{u}_i-\boldsymbol{e}_k\right\|_{L_1}^2\right) \\ &\quad + \sum_{i\in V}\frac{\mu_i}{2}\left\|\mathbf{u}_i-\hat{\mathbf{u}}_i\right\|^2 , \end{aligned} \tag{14}$$

where

$$\langle \mathbf{U}, \mathbf{L}_{\boldsymbol{s}}\mathbf{U}\rangle \;=\; \operatorname{trace}(\mathbf{U}^T\mathbf{L}_{\boldsymbol{s}}\mathbf{U}),$$

and $\hat{\mathbf{u}}_i$ is a vector indicating prior class knowledge of sample i. We set $\hat{\boldsymbol{u}}_i = \boldsymbol{e}_k$ if node i is known to be in class k.

As mentioned before, the first (smoothing) term in the GL functional (14) measures variations in the vector field. The simplex representation has the advantage that, like in Potts-based models but unlike in some other multiclass methods, the penalty assigned to differently labeled neighbors is independent of the integer ordering of the labels. The second (potential) term drives the system closer to the vertices of the simplex, with the use of an L_1 norm preventing the emergence of an undesirable minimum at the center of the simplex, as would happen with an L_2 norm for large K. This potential aims to provide a clear way to calculate class memberships, as the phase composition is purer near the vertices of the simplex. The compromise between the smoothing and potential terms is established through the constant ϵ. The third (fidelity) term enables the encoding of *a priori* information.

B. Energy Minimization

Following [5], we use a convex splitting scheme to minimize the GL functional in the phase field approach. The energy functional (14) is decomposed into convex and concave parts as:

$$\begin{aligned} E(\mathbf{U}) \;&=\; E_{\text{convex}}(\mathbf{U}) + E_{\text{concave}}(\mathbf{U}) \\ E_{\text{convex}}(\mathbf{U}) \;&=\; \frac{\epsilon}{2}\langle \mathbf{U}, \mathbf{L}_{\boldsymbol{s}}\mathbf{U}\rangle + \frac{C}{2}\langle \mathbf{U}, \mathbf{U}\rangle \\ E_{\text{concave}}(\mathbf{U}) \;&=\; \frac{1}{2\epsilon}\sum_{i\in V}\prod_{k=1}^{K}\frac{1}{4}\left\|\mathbf{u}_i-\boldsymbol{e}_k\right\|_{L_1}^2 \\ &\quad + \sum_{i\in V}\frac{\mu_i}{2}\left\|\mathbf{u}_i-\hat{\mathbf{u}}_i\right\|_{L_2}^2 - \frac{C}{2}\langle \mathbf{U}, \mathbf{U}\rangle \end{aligned}$$

with $C \in \mathbb{R}$ denoting a constant that is chosen to guarantee the convexity/concavity of the energy terms. Under the right conditions, this approach results in an unconditionally stable time-discretization scheme for gradient descent [22], [24], [53] of the form

$$U_{ik}^{n+1} + dt\,\frac{\delta E_{\text{convex}}}{\delta U_{ik}}(U_{ik}^{n+1}) = U_{ik}^{n} - dt\,\frac{\delta E_{\text{concave}}}{\delta U_{ik}}(U_{ik}^{n}). \tag{15}$$

Evaluating the functional derivatives, we write this equation in matrix form as

$$\begin{aligned} &\mathbf{U}^{n+1} + dt\left(\epsilon\mathbf{L}_{\boldsymbol{s}}\mathbf{U}^{n+1} + C\mathbf{U}^{n+1}\right) \\ &\qquad = \mathbf{U}^{n} - dt\left(\frac{1}{2\epsilon}\mathbf{T}^{n} + \boldsymbol{\mu}(\mathbf{U}^{n}-\hat{\mathbf{U}}) - C\mathbf{U}^{n}\right), \end{aligned} \tag{16}$$

where

$$T_{ik} = \sum_{l=1}^{K} \frac{1}{2}\left(1 - 2\delta_{kl}\right) \|\mathbf{u}_i - \boldsymbol{e}_l\|_{L_1} \prod_{\substack{m=1\\ m\neq l}}^{K} \frac{1}{4} \|\mathbf{u}_i - \boldsymbol{e}_m\|_{L_1}^2 \,, \tag{17}$$

$\boldsymbol{\mu}$ is a diagonal matrix with elements μ_i, and $\hat{\mathbf{U}} = (\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_{N_D})^T$.

Solving (16) for $\mathbf{U}^{n+1}$ gives the iteration equation

$$\mathbf{U}^{n+1} = \mathbf{B}^{-1}\left[\left(1 + C\,dt\right)\mathbf{U}^n - \frac{dt}{2\epsilon}\mathbf{T}^n - dt\,\boldsymbol{\mu}(\mathbf{U}^n - \hat{\mathbf{U}})\right] \tag{18}$$

where

$$\mathbf{B} = (1 + C\,dt)\mathbf{I} + \epsilon\,dt\,\mathbf{L}_{\boldsymbol{s}}. \tag{19}$$

This implicit scheme allows the evolution of $\mathbf{U}$ to be numerically stable regardless of the time step dt, in spite of the numerical "stiffness" of the underlying differential equations which could otherwise force dt to be impractically small. (Note, though, that stability is a separate issue from accuracy.)

In general, after the update, the phase field is no longer on the Σ^K simplex. Consequently, we use the procedure in [14] to project back to the simplex.

Computationally, the schemes' numerical efficiency is increased by using a low-dimensional subspace spanned by only a small number of eigenfunctions. Let $\mathbf{X}$ be the matrix of eigenvectors of $\mathbf{L}_{\boldsymbol{s}}$ and $\boldsymbol{\Lambda}$ be the diagonal matrix of corresponding eigenvalues. We now write $\mathbf{L}_{\boldsymbol{s}}$ as its eigendecomposition $\mathbf{L}_{\boldsymbol{s}} = \mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^T$,

$$\mathbf{B} = \mathbf{X}\left[(1 + C\,dt)\mathbf{I} + \epsilon\,dt\,\boldsymbol{\Lambda}\right]\mathbf{X}^T, \tag{20}$$

but we approximate $\mathbf{X}$ by a truncated matrix retaining only N_e eigenvectors ($N_e \ll N_D$), to form a matrix of dimension $N_D \times N_e$. The term in brackets is simply a diagonal $N_e \times N_e$ matrix. This allows $\mathbf{B}$ to be calculated rapidly, but more importantly it allows the update step (18) to be decomposed into two significantly faster matrix multiplications (as discussed below), while sacrificing little accuracy in practice.

For initialization, the phase compositions of the fidelity points are set to the vertices of the simplex corresponding to the known labels, while the phase compositions of the rest of the points are set randomly.

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. Consequently, the calculation is stopped when

$$\frac{\max_i \|\mathbf{u}_i{}^{n+1} - \mathbf{u}_i{}^{n}\|^2}{\max_i \|\mathbf{u}_i{}^{n+1}\|^2} < \eta, \tag{21}$$

where η represents a given small positive constant. The final classes are obtained by assigning class k to node i if $\mathbf{u}_i$ is closest to vertex $\boldsymbol{e}_k$ on the Gibbs simplex.

The specific algorithm is outlined in Figure 1. Note that other operator splitting methods have been studied for minimization problems (e.g. [36]). Ours however has the following advantages: (i) it is direct (i.e. it does not require the solution of further minimization problems), (ii) the resolution can be adjusted by increasing the number of eigenvectors N_e used in the representation of the phase field, and (iii) it has low complexity. To see this final point, observe that each iteration of the multiclass GL algorithm has only $O(N_D K N_e)$ operations for the main loop, since matrix $\mathbf{Z}$ in Figure 1 only has dimensions $N_e \times K$, and then $O(N_D K \log K)$ for the projection to the simplex. Usually, $N_e \ll N_D$ and $K \ll N_D$, so the dominant factor is simply the size of the data set N_D. In addition, it is generally the case that the number of iterations required for convergence is moderate (around 200 iterations). Thus, practically speaking, the complexity of the algorithm is linear.

V. MBO Reduction of the Ginzburg-Landau Energy Functional

Given that the modified Allen-Cahn equation (5), resulting from the L_2 gradient flow of the GL energy functional, produces an efficient method for data segmentation when adapted to function estimation on graphs, a natural question is what other adaptations of this formulation can be equally effective when expressed in the graph domain. The immediate answer comes from the relation between the Allen-Cahn equation and the motion by mean curvature.

Let us start by reviewing this connection in the continuous setting. In [39], Merriman, Bence and Osher propose an algorithm to approximate motion by mean curvature, or motion in which normal velocity equals mean curvature, using threshold dynamics. The authors note that if one applies the heat equation to an interface, then the diffusion blunts the sharp points of the boundary, but has very little effect on the flatter regions. Therefore, one can imagine that diffusion creates some sort of motion by mean curvature, providing that we specify the boundaries of the moving set.

Given a phase field $u(z,t)$, consider the basic (unmodified) Allen-Cahn equation, namely equation (5) without the fidelity term:

$$\frac{\partial u}{\partial t} = \epsilon \Delta u - \frac{1}{\epsilon}\Phi'(u). \tag{22}$$

For small values of ϵ, the following time-splitting scheme can be used numerically to evolve the Allen-Cahn equation:

1) The first step is propagation using:

$$\frac{\partial u}{\partial t} = \epsilon \Delta u \tag{23}$$

2) The second step is propagation using:

$$\frac{\partial u}{\partial t} = -\frac{1}{\epsilon}\Phi'(u) \tag{24}$$

Note, however, that in the $\epsilon \to 0$ limit, the second step is simply thresholding [39]. Thus, as $\epsilon \to 0$, the time splitting scheme above consists of alternating between diffusion and thresholding steps.

It has been shown [44] that in the limit $\epsilon \to 0$, the rescaled solutions $u_\epsilon(z, t/\epsilon)$ of (22) yield motion by mean curvature of the interface between the two phases of the solutions. This motivates the two sequential steps of the MBO scheme:

1) *Diffusion.* Let $u^{n+\frac{1}{2}} = S(\delta t)u^n$ where $S(\delta t)$ is the propagator (by time δt) of the standard heat equation:

$$\frac{\partial u}{\partial t} = \Delta u. \tag{25}$$

Fig. 1: Multiclass GL algorithm

```
Require: $\epsilon, dt, N_D, N_e, K, \boldsymbol{\mu}, \hat{\mathbf{U}}, \mathbf{\Lambda}, \mathbf{X}$
Ensure: $\text{out} = \mathbf{U}^{\mathbf{end}}$
  $C \leftarrow \mu + \frac{1}{\epsilon}$
  $\mathbf{Y} \leftarrow \left[(1 + C\,dt)\mathbf{I} + \epsilon\,dt\mathbf{\Lambda}\right]^{-1} \mathbf{X}^T$
  for $i = 1 \to N_D$ do
    $U_{ik}^{\;0} \leftarrow rand((0,1)),\ U_{ik}^0 \leftarrow projectToSimplex(\mathbf{u}_i{}^0)$.   If $\mu_i > 0,\ U_{ik}^{\;0} \leftarrow \hat{U}_{ik}^{\;0}$
  end for
  $n \leftarrow 1$
  while Stop criterion not satisfied do
    for $i = 1 \to N_D,\ k = 1 \to K$ do
      $T_{ik}^{\;n} \leftarrow \sum_{\beta=1}^{K} \frac{1}{2}\left(1 - 2\delta_{k\beta}\right) \left\|\mathbf{u}_i{}^n - \boldsymbol{e}_\beta\right\|_{L_1} \prod_{\gamma=1,\gamma\neq\beta}^{K} \frac{1}{4} \left\|\mathbf{u}_i{}^n - \boldsymbol{e}_\gamma\right\|_{L_1}^2$
    end for
    $\mathbf{Z} \leftarrow \mathbf{Y}\left[(1 + C\ dt)\,\mathbf{U}^n - \frac{dt}{2\epsilon}\,\mathbf{T}^n - dt\,\boldsymbol{\mu}(\mathbf{U}^n - \hat{\mathbf{U}})\right]$
    $\mathbf{U}^{n+1} \leftarrow \mathbf{X}\mathbf{Z}$
    for $i = 1 \to N_D$ do
      $\mathbf{u}_i{}^{n+1} \leftarrow projectToSimplex(\mathbf{u}_i{}^{n+1})$
    end for
    $n \leftarrow n + 1$
  end while
```

2) *Thresholding.* Let

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

Barles [2] and Evans [23] have proven rigorously that this scheme approximates motion by mean curvature.

A. Graph Formulation

The motion by mean curvature of the MBO scheme can be generalized to the case of functions on a graph in much the same way as the procedure followed for the modified Allen-Cahn equation (5) in [5]. Merkurjev et al. have pursued this idea in [38], where a modified MBO scheme on graphs has been applied to the case of binary segmentation. The authors apply a two-step time splitting scheme to (5) so that the second step is the same as the one in the original MBO scheme, and then replace the Δu term with a more general graph term $-\mathbf{L}_{\boldsymbol{s}}\boldsymbol{u}$. The discretized version of the algorithm is:

1) Heat equation with forcing term:

$$\frac{\boldsymbol{u}^{n+\frac{1}{2}} - \boldsymbol{u}^n}{dt} = -\mathbf{L}_{\boldsymbol{s}}\boldsymbol{u}^n - \boldsymbol{\mu}(\boldsymbol{u}^n - \hat{\boldsymbol{u}}). \tag{26}$$

2) Thresholding:

$$u_i^{n+1} = \begin{cases} 1, & \text{if } u_i^{n+\frac{1}{2}} > 0, \\ -1, & \text{if } u_i^{n+\frac{1}{2}} < 0. \end{cases}$$

Here, after the second step, u_i^n can take only two values of 1 or -1; thus, this method is appropriate for binary segmentation. Note that the fidelity term scaling can be different from the one in (5).

In practice, it can be productive to repeat the diffusion step a number of times before thresholding. In order to keep the convention that one iteration of the diffusion-thresholding procedure corresponds to one time step, we divide dt by the number of diffusion steps per iteration, which we denote N_S.

B. Extension to Multiclass Segmentation

Using the standard Gibbs-simplex Σ^K defined in Section IV, the multiclass extension of the algorithm in [38] is straightforward. The notation is the same as in the previous section: we use a matrix $\mathbf{U}$ to represent the phase composition of nodes. The second step of the algorithm is modified, however, so that the thresholding is converted to the displacement of the vector field variable towards the closest vertex in the Gibbs simplex. In other words, now in the second step the row vector $\mathbf{u}_i{}^{n+\frac{1}{2}}$ of step 1 is projected back to the simplex (using the approach outlined in [14] as before) and then a pure phase given by the vertex in the Σ^K simplex closest to $\mathbf{u}_i{}^{n+\frac{1}{2}}$ is assigned to be the new phase composition of node i.

In summary, the new algorithm consists of alternating between the following two steps to obtain approximate solutions $\mathbf{U}^n$ at discrete times:

1) Heat equation with forcing term:

$$\frac{\mathbf{U}^{n+\frac{1}{2}} - \mathbf{U}^n}{dt} = -\mathbf{L}_{\boldsymbol{s}}\mathbf{U}^n - \boldsymbol{\mu}(\mathbf{U}^n - \hat{\mathbf{U}}). \tag{27}$$

2) Thresholding:

$$\mathbf{u}_i{}^{n+1} = \boldsymbol{e}_k, \tag{28}$$

where vertex $\boldsymbol{e}_k$ is the vertex in the simplex closest to $projectToSimplex(\mathbf{u}_i{}^{n+\frac{1}{2}})$.

As for the multiclass GL algorithm, when a label is known, it is represented by the corresponding vertex in the Σ^K simplex. The final classification is achieved by assigning node i to class k if if the kth component of $\mathbf{u}_i$ is one. Again, as in the binary

Fig. 2: Multiclass MBO Algorithm

```
Require: $dt, N_D, N_e, N_S, K, \boldsymbol{\mu}, \hat{\mathbf{U}}, \boldsymbol{\Lambda}, \mathbf{X}$
Ensure: out $= \mathbf{U}^{\mathbf{end}}$
  $\mathbf{Y} \leftarrow \left(\mathbf{I} + \frac{dt}{N_S}\boldsymbol{\Lambda}\right)^{-1} \mathbf{X}^T$
  for $i = 1 \to N_D$ do
    $U_{ik}^{\ 0} \leftarrow rand((0,1)),\ \mathbf{u}_i^{\ 0} \leftarrow projectToSimplex(\mathbf{u}_i^{\ 0})$.  If $\mu_i > 0,\ U_{ik}^{\ 0} \leftarrow \hat{U}_{ik}^{\ 0}$
  end for
  $n \leftarrow 1$
  while Stop criterion not satisfied do
    for $s = 1 \to N_S$ do
      $\mathbf{Z} \leftarrow \mathbf{Y}\left[\mathbf{U}^n - \frac{dt}{N_S}\,\boldsymbol{\mu}(\mathbf{U}^n - \hat{\mathbf{U}})\right]$
      $\mathbf{U}^{n+1} \leftarrow \mathbf{XZ}$
    end for
    for $i = 1 \to N_D$ do
      $\mathbf{u}_i^{\ n+1} \leftarrow projectToSimplex(\mathbf{u}_i^{\ n+1})$
      $\mathbf{u}_i^{\ n+1} \leftarrow \boldsymbol{e}_k$, where $k$ is closest simplex vertex to $\mathbf{u}_i^{\ n+1}$
    end for
    $n \leftarrow n + 1$
  end while
```

case, the diffusion step can be repeated a number of times before thresholding and when that happens, dt is divided by the number of diffusion iterations N_S.

As in the previous section, we use an implicit numerical scheme. For the MBO algorithm, the procedure involves modifying (27) to apply $\mathbf{L_s}$ to $\mathbf{U}^{n+\frac{1}{2}}$ instead of to $\mathbf{U}^n$. This gives the diffusion step

$$\mathbf{U}^{n+\frac{1}{2}} = \mathbf{B}^{-1}\left[\mathbf{U}^n - dt\,\boldsymbol{\mu}(\mathbf{U}^n - \hat{\mathbf{U}})\right] \tag{29}$$

where

$$\mathbf{B} = \mathbf{I} + dt\,\mathbf{L_s}. \tag{30}$$

As before, we use the eigendecomposition $\mathbf{L_s} = \mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^T$ to write

$$\mathbf{B} = \mathbf{X}\left(\mathbf{I} + dt\,\boldsymbol{\Lambda}\right)\mathbf{X}^T, \tag{31}$$

which we approximate using the first N_e eigenfunctions. The initialization procedure and the stopping criterion are the same as in the previous section.

The multiclass MBO algorithm is summarized in Figure 2. Its complexity is $O(N_D K N_e N_S)$ operations for the main loop, $O(N_D K \log K)$ operations for the projection to the simplex and $O(N_D K)$ operations for thresholding. As for the multiclass GL algorithm, $N_e \ll N_D$ and $K \ll N_D$. Furthermore N_S needs to be set to three, and due to the thresholding step, we find that extremely few iterations (e.g., 6) are needed to reach steady state. Thus, in practice, the complexity of this algorithm is linear as well, and typical runtimes are very rapid as shown in Table III.

VI. Experimental Results

We have tested our algorithms on synthetic data, grayscale and color images, and the MNIST, COIL and WebKB benchmark data sets. In all cases, we compute the symmetric normalized graph Laplacian matrix $\mathbf{L_s}$, of expression (8), using N-neighborhood graphs: in other words, vertices i and j are connected only if i is among the N nearest neighbors of j or if j is among the N nearest neighbors of i. Otherwise, we set $w(i,j) = 0$. This results in a sparse matrix, making calculations and algorithms more tractable. In addition, for the similarity function we use the local scaling weight function of Zelnik-Manor and Perona [43], defined as:

$$w(i,j) = \exp\left(-\frac{d(i,j)^2}{\sqrt{\tau(i)\tau(j)}}\right) \tag{32}$$

where $d(i,j)$ is some distance measure between vertices i and j, such as the L_2 distance, and $\sqrt{\tau(i)} = d(i,k)$ defines a local value for each vertex i, parametrized by M, with k being the index of the Mth closest vertex to i.

With the exception of the images, all the results and comparisons with other published methods are summarized in Tables I and II. Due to the arbitrary selection of the fidelity points, our reported values correspond to averages obtained over 10 runs with different random selections. The timing results and number of iterations of the two methods are shown in Tables III and IV, respectively. The methods are labeled as “multiclass GL” and “multiclass MBO”. These comparisons show that our methods exhibit a performance that is competitive with or better than the current state-of-the-art segmentation algorithms.

Parameters are chosen to be compatible between the methods. For the multiclass GL method, the convexity constant used is: $C = \mu + \frac{1}{\epsilon}$. This is the minimum constant that guarantees the convexity and concavity of the terms in the energy partition of the convex splitting strategy employed. For the multiclass MBO method, as discussed in the previous section, the diffusion step can be repeated a number of times before thresholding. In all of our results, we run the diffusion step three times before any thresholding is done ($N_S = 3$).

To compute the eigenvectors and eigenvalues of the symmetric graph Laplacian, we use fast numerical solvers. As we only need to calculate a portion of the eigenvectors to get good results, we use the Rayleigh-Chebyshev procedure of [1] for computing all the eigendecompositions. This numerical solver is especially efficient for producing a few of the smallest eigenvectors of a sparse symmetric matrix. For example, for the MNIST data set of 70,000 images, it was only necessary to calculate 300 eigenvectors, which is less than 0.5% of the data set size. This is one of the factors that makes our methods very efficient.

A. Synthetic Data

The synthetic data set we tested our method against is the three moons data set. It is constructed by generating three half circles in $\mathbb{R}^2$. The two half top circles are unit circles with centers at $(0, 0)$ and $(3, 0)$. The bottom half circle has radius 1.5 and the center at $(1.5, 0.4)$. Five hundred points from each of those three half circles are sampled and embedded in $\mathbb{R}^{100}$ by adding Gaussian noise with standard deviation of 0.14 to each of the 100 components of each embedded point. The dimensionality of the data set, together with the noise, makes segmentation a significant challenge.

The weight matrix of the graph edges was calculated using $N = 10$ nearest neighbors and local scaling based on the 17^{th} closest point ($M = 17$). The fidelity term was constructed by labeling 25 points per class, 75 points in total, corresponding to only 5% of the points in the data set.

The multiclass GL method used the following parameters: 20 eigenvectors, $\epsilon = 1$, $dt = 0.1$, $\mu = 30$, $\eta = 10^{-7}$. The method was able to produce an average of 98.4% of correct classification, with a corresponding computation time of 0.16 s per run on a 2.4 GHz Intel Core i2 Quad without any parallel processing.

Analogously, the multiclass MBO method used the following parameters: 20 eigenvectors, $dt = 0.1$, $\mu = 30$, $\eta = 10^{-7}$. It was able to segment an average of 99.12% of the points correctly over 10 runs with only 3 iterations and about 0.01 s of computation time. One of the results obtained is shown in Figure 3.

Table I gives published results from other related methods, for comparison. Note that the results for p-Laplacians [11], Cheeger cuts [48] and binary GL are for the simpler binary problem of two moons (also embedded in $\mathbb{R}^{100}$). While, strictly speaking, these are unsupervised methods, they all incorporate prior knowledge such as a mass balance constraint. We therefore consider them comparable to our SSL approach. The "tree GL" method [25] uses a scalar multiclass GL approach with a tree metric. It can be seen that our methods achieve the highest accuracy on this test problem.

B. Image Segmentation

a) Grayscale (single channel) Image: We tested our algorithms on the image of figures shown in Figure 4a. This is a 191 × 196 pixel grayscale image, to be divided into five classes: black, dark gray, medium gray, light gray and white. To construct the weight matrix, we use feature vectors corresponding to a pixel's x-coordinate, y-coordinate, and intensity. For the fidelity term, 1,500 or 4% of the points were randomly chosen.

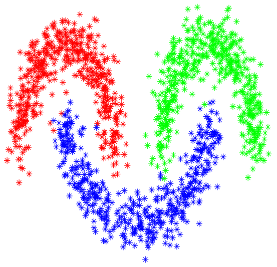


Fig. 3: Segmentation of three moons using multiclass MBO (98.4667% correct).

A local scaling graph with $N = 30$ and $M = 30$ is constructed and used by both our algorithms.

The multiclass Ginzburg-Landau method was applied using the following parameters: 10 eigenvectors, $\epsilon = 1$, $dt = 0.5$, $\mu = 50$ and $\eta = 10^{-7}$. It was able to segment the classes perfectly, with an average time of 4.1 s. The original image as well as the segmentation are included in Figure 4. In each segmentation, the white regions correspond to the identified class.

The multiclass MBO method used the following parameters: 10 eigenvectors, $dt = 0.5$, $\mu = 50$, $\eta = 10^{-7}$. The algorithm was able to segment all the points correctly in 2 iterations and 0.232 s.

We compare our result to the method of Li and Kim [37], which also segments the image perfectly. However, their method requires additional information, such as the densities of each class, that we do not need in our method.

b) Color (multichannel) Image: We then tested our algorithms on the color image of cows shown in Figure 5a. This is a 213 × 320 color image, to be divided into four classes: sky, grass, black cow and red cow.

To construct the weight matrix, we use feature vectors defined as the set of intensity values in the neighborhood of a pixel. The neighborhood is a patch of size 5 × 5. Red, green and blue channels are appended, resulting in a feature vector of dimension 75. A local scaling graph with $N = 30$ and $M = 30$ is constructed for both algorithms. For the fidelity term, 2.6% of labeled pixels are used.

The multiclass Ginzburg-Landau method used the following parameters: 30 eigenvectors, $\epsilon = 1$, $dt = 0.1$, $\mu = 50$ and $\eta = 10^{-7}$. The average time for segmentation using different fidelity sets was 19.9 s.

The multiclass MBO method used the following parameters: 30 eigenvectors, $dt = 0.1$, $\mu = 50$, $\eta = 10^{-7}$. The average time for segmentation over 10 runs was around 1.2 s, and there were 11 iterations.

One of the results of each of our two methods (using the same fidelity set) is depicted in Figure 5. It can be seen that both methods are able to identify all the classes, with almost no perceptible difference between the two results. Most of the

TABLE I: Results for benchmark data sets: Moons, MNIST, COIL and WebKB

Two/Three moons

Method	Accuracy
spectral clustering [25]	80%
p-Laplacian [11]	94%
Cheeger cuts [48]	95.4%
tree GL [25]	97.4%
binary GL [5]	97.7%
multiclass GL	98.4%
multiclass MBO	99.12%

MNIST

Method	Accuracy
p-Laplacian [11]	87.1%
multicut normalized 1-cut [30]	87.64%
linear classifiers [34], [35]	88%
Cheeger cuts [48]	88.2%
boosted stumps [32], [35]	92.3-98.74%
transductive classification [49]	92.6%
tree GL [25]	93.0%
k-nearest neighbors [34], [35]	95.0-97.17%
neural/convolutional nets [16], [34], [35]	95.3-99.65%
nonlinear classifiers [34], [35]	96.4-96.7%
multiclass GL	96.8%
multiclass MBO	96.91%
SVM [18], [34]	98.6-99.32%

COIL

Method	Accuracy
k-nearest neighbors [47]	83.5%
LapRLS [3], [47]	87.8%
sGT [31], [47]	89.9%
SQ-Loss-I [47]	90.9%
MP [47]	91.1%
multiclass GL	91.4%
multiclass MBO	91.46%

WebKB

Method	Accuracy
vector method [12]	64.47%
k-nearest neighbors ($k = 10$) [12]	72.56%
centroid (normalized sum) [12]	82.66%
naive Bayes [12]	83.52%
SVM (linear kernel) [12]	85.82%
multiclass GL	87.3%
multiclass MBO	88.48%

TABLE II: WebKB results with varying fidelity percentage

Method	10%	15%	20%	25%	30%
WebKB results for Multiclass GL (% correct)	81.5%	84.2%	85.4%	86.7%	87.3%
WebKB results for Multiclass MBO (% correct)	83.71%	85.75%	86.81%	87.74%	88.48%

TABLE III: Comparison of timings (in seconds)

Data set	three moons	grayscale image	color image	MNIST	COIL	WebKB
Graph Calculation	0.771	19.96	645.34	6183.1	0.95	399.35
Eigenvector Calculation	0.331	210.10	190.93	1683.5	0.19	64.78
Multiclass GL	0.163	4.08	19.92	811.5	2.31	6.97
Multiclass MBO	0.013	0.23	1.20	15.4	0.03	0.05

TABLE IV: Comparison of number of iterations

Data set	three moons	grayscale image	color image	MNIST	COIL	WebKB
Multiclass GL	140	90	200	460	700	275
Multiclass MBO	3	2	11	7	6	7

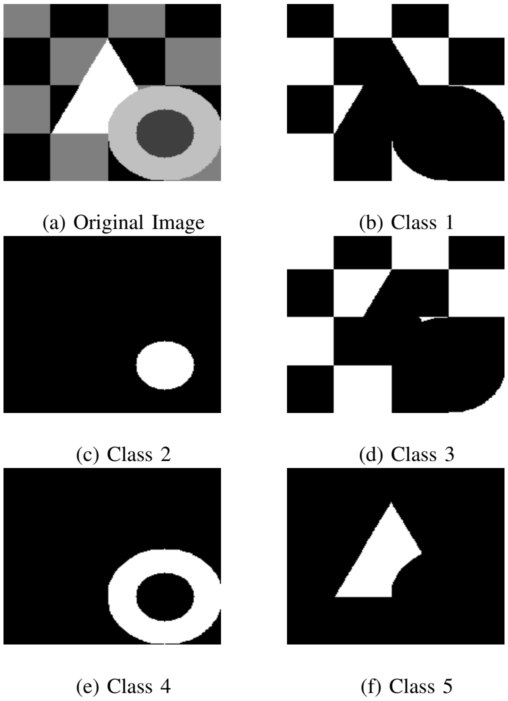
TABLE V: Confusion Matrix for the MNIST Data Segmentation - Multiclass GL

Obtained/True	0	1	2	3	4	5	6	7	8	9
0	6844	1	43	4	3	16	21	2	19	19
1	6	7809	36	8	35	2	14	62	58	15
2	5	22	6733	45	2	4	1	27	19	7
3	0	3	20	6882	1	91	0	1	89	92
4	1	16	6	2	6626	4	7	15	28	75
5	9	0	3	75	0	6072	28	2	125	14
6	31	5	11	3	23	65	6802	0	31	4
7	2	16	108	45	11	7	0	7078	19	110
8	1	2	22	42	4	15	3	2	6365	20
9	4	3	8	35	119	37	0	104	72	6602

TABLE VI: Confusion Matrix for the MNIST Data Segmentation - MBO Scheme

Obtained/True	0	1	2	3	4	5	6	7	8	9
0	6844	20	41	3	3	15	21	1	20	17
1	5	7789	32	8	34	1	14	63	51	14
2	5	22	6731	42	2	4	1	23	19	8
3	0	3	20	6890	1	86	0	1	81	90
4	1	17	6	2	6625	3	7	12	28	67
5	9	0	3	70	0	6077	28	2	109	14
6	31	5	11	3	22	69	6800	0	29	5
7	2	16	117	44	12	9	0	7093	20	101
8	2	2	21	46	4	17	5	2	6398	22
9	4	3	8	33	121	32	0	96	70	6620

mistakes made correspond to identifying some borders of the red cow as part of the black cow.


(a) Original Image

(b) Class 1

(c) Class 2

(d) Class 3

(e) Class 4

(f) Class 5

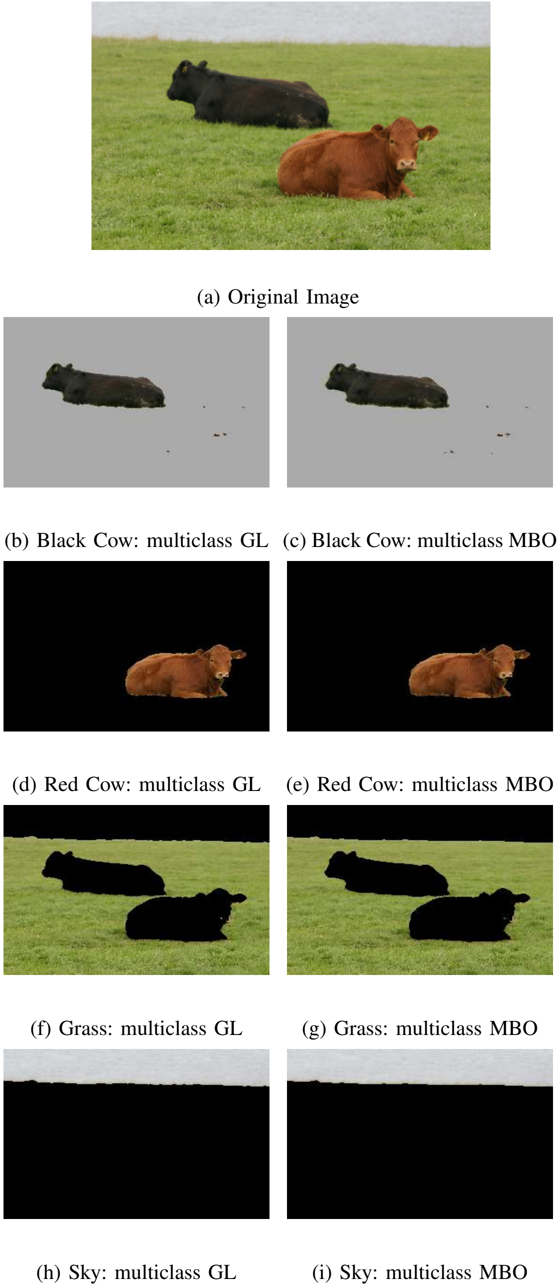
Fig. 4: Grayscale image segmentation

C. MNIST Data

The MNIST data set [35] is composed of $70,000$ 28×28 images of handwritten digits 0 through 9. Examples of entries can be found in Figure 6. The task is to classify each of the images into the corresponding digit. The images include digits from 0 to 9; thus, this is a 10 class segmentation problem.

To construct the weight matrix, we used $N = 8$ nearest neighbors with local scaling based on the 8^{th} closest neighbor ($M = 8$). Note that we perform no preprocessing, i.e. the graph is constructed using the 28×28 images. For the fidelity term, 250 images per class (2500 images corresponding to 3.6% of the data) are chosen randomly.

The multiclass GL method used the following parameters: 300 eigenvectors, $\epsilon = 1$, $dt = 0.15$, $\mu = 50$ and $\eta = 10^{-7}$. The complete set of 70,000 images was segmented with an


(a) Original Image

(b) Black Cow: multiclass GL

(c) Black Cow: multiclass MBO

(d) Red Cow: multiclass GL

(e) Red Cow: multiclass MBO

(f) Grass: multiclass GL

(g) Grass: multiclass MBO

(h) Sky: multiclass GL

(i) Sky: multiclass MBO

Fig. 5: Color image segmentation

average accuracy of 96.8% of the digits classified correctly in an average time of 811 s. The averages are obtained over 10 runs. The confusion matrix for the best result obtained is included in Table V. Most of the mistakes were in distinguishing digits 4 and 9, and digits 5 and 8.

The multiclass MBO method used the following parameters: 300 eigenvectors, $dt = 0.15$, $\mu = 50$, $\eta = 10^{-7}$. The algorithm was able to segment an average of 96.91% of the digits correctly over 10 runs in only 4 iterations and 15.382 s. We display the confusion matrix in Table VI. Note that most of the mistakes were in distinguishing digits 4 and 9, and digits 2 and 7.

Table I compares our results with those from other methods in the literature. As with the three moon problem, some of these are based on unsupervised methods but incorporate enough prior information that they can fairly be compared with SSL methods. The methods of linear/nonlinear classifers, k-nearest neighbors, boosted stumps, neural and convolutional nets and SVM are all supervised learning approaches, taking 60,000 of the digits as a training set and 10,000 digits as a testing set [35], in comparison to our SSL approaches where we take only 3.6% of the points for the fidelity term. Our algorithms are nevertheless competitive with, and in most cases outperform, these supervised methods. Moreover, we perform no preprocessing or initial feature extraction on the image data, unlike most of the other methods we compare with (we did exclude from the comparison, however, methods that explicitly deskewed the image). While there is a computational price to be paid in forming the graph when data points use all 784 pixels as features (see graph calculation time in Table III), this is a one-time operation that conceptually simplifies our approach.

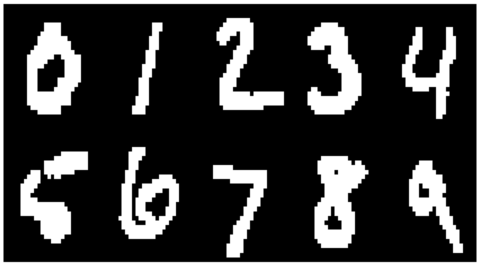


Fig. 6: Examples of digits from the MNIST data base

D. COIL dataset

We evaluated our performance on the benchmark COIL data set [13], [41]. This is a set of color 128×128 images of 100 objects, taken at different angles. The red channel of each image was then downsampled to 16×16 pixels by averaging over blocks of 8×8 pixels. Then 24 of the objects were randomly selected and then partitioned into six classes. Discarding 38 images from each class leaves 250 per class, giving a data set of 1500 data points.

To construct the weight matrix, we used $N = 4$ nearest neighbors with local scaling based on the 4^{th} closest neighbor ($M = 4$). The fidelity term was constructed by labeling 10% of the points, selected at random.

For multiclass GL, the parameters were: 50 eigenvectors, $\epsilon = 1$, $dt = 0.2$, $\mu = 100$ and $\eta = 10^{-7}$. This resulted in 91.4% of the points classified correctly (average) in 2.3 s.

For multiclass MBO, the parameters were: 50 eigenvectors, $dt = 0.2$, $\mu = 100$, $\eta = 10^{-7}$. We obtained an accuracy of 91.46%, averaged over 10 runs. The procedure took 6 iterations and 0.03 s.

Comparative results reported in [47] are shown in Table I. These are all SSL methods (with the exception of k-nearest neighbors which is supervised), using 10% fidelity just as we do. Our results are of comparable or greater accuracy.

E. WebKB dataset

Finally, we tested our methods on the task of text classification on the WebKB data set [17]. This is a collection of webpages from Cornell, Texas, Washington and Wisconsin universities, as well as other miscellaneous pages from other universities. The webpages are to be divided into four classes: project, course, faculty and student. The data set is preprocessed as described in [12].

To construct the weight matrix, we used 575 nearest neighbors. Tfidf term weighting [12] is used to represent the website feature vectors. They were then normalized to unitary length. The weight matrix points are calculated using cosine similarity.

For the multiclass GL method, the parameters were: 250 eigenvectors, $\epsilon = 1$, $dt = 1$, $\mu = 50$ and $\eta = 10^{-7}$. The average accuracies obtained are: 81.5%, 84.2%, 85.4%, 86.7% and 87.3% over fidelity sets of 10%, 15%, 20%, 25% and 30% of the points, respectively. The average computation time is 6.97 s.

For the multiclass MBO method, the parameters were: 250 eigenvectors, $dt = 1$, $\mu = 4$, $\eta = 10^{-7}$. We obtained average accuracies of: 83.71%, 85.75%, 86.81%, 87.74% and 88.48% over fidelity sets of 10%, 15%, 20%, 25% and 30% of the points, respectively. The procedure took an average of 0.05 s and 7 iterations.

We compare our results with those of several supervised learning methods reported in [12], shown in Table I. For these methods, two thirds of the data was used for training, and one third for testing. Our SSL methods obtain higher accuracy, using only 20% fidelity (for multiclass MBO). Note also that a larger sample of points for the fidelity term reduces the error in the classification results, as shown in Table II. Nevertheless, the accuracy is high even for the smallest fidelity sets. Therefore, the methods appear quite adequate for the SSL setting where only a few labeled data points are known beforehand.

Multiclass GL and MBO: All the results reported point out that both multiclass GL and multiclass MBO perform well in terms of data segmentation accuracy. While the ability to tune multiclass GL can be an advantage, multiclass MBO is simpler and, in our examples, displays even better performance in terms of its greater accuracy and tiny number of iterations required. The relative strength and speed of multiclass MBO may not always hold, but the avoidance of a nonconvex functional minimization that takes place in multiclass GL may explain the accuracy and speed increase. Exploring the underlying connections of the energy evolution of these methods and the energy landscape for the relaxed Cheeger cut minimization recently established in [9] are to be explored in future work.

VII. Conclusions

We have presented two graph-based algorithms for multiclass classification of high dimensional data. The two algorithms are based on the diffuse interface model using the Ginzburg-Landau functional, and the multiclass extension is obtained using the Gibbs simplex. The first algorithm minimizes the functional using gradient descent and a convex-splitting scheme. The second algorithm executes a simple scheme based on an adaptation of the classical numerical MBO method. It uses fewer parameters than the first algorithm, and while this may in some cases make it more restrictive, in our experiments it was highly accurate and efficient. Both of these algorithms demonstrate how methods motivated by the PDE literature can be productively adapted to graphs, producing effective multiclass data segmentation methods.

Testing the algorithms on synthetic data, images and benchmark data sets shows that the results are competitive with or better than some of the most recent and best published algorithms in the literature. In addition, our methods have several advantages. First, they are simple and efficient, avoiding the need for intricate function minimizations or heavy preprocessing of data. Second, a relatively small proportion of fidelity points is needed for producing an accurate result. For most of our data sets, we used at most 10% of the data points for the fidelity term; for synthetic data and the two images, we used no more than 5%. Furthermore, the minimization obtained is not sensitive to the initial state. As long as the fidelity set contains samples of all classes in the problem, a random initialization is enough to produce good multiclass segmentation results. Finally, our methods do not use one-vs-all or sequences of binary segmentations that are needed for some other multiclass methods. We therefore avoid the bias and extra processing that is often inherent in those methods.

Our algorithms benefit from the sparsity of the neighborhood graphs generated by the local scaling procedure of Perona and Zelnik-Manor [43], as well as from the fast numerical Rayleigh-Chebyshev method of Anderson [1] used to find a subset of eigenvalues and eigenvectors of the resulting graph Laplacian matrices.

In all of the data sets we have studied, multiclass MBO performed better than multiclass GL in terms of accuracy and convergence time. Nevertheless, we anticipate that more intricate geometries could impair its effectiveness. In those cases, multiclass GL might still perform well, due to the additional control provided by tuning ϵ to increase the thickness of the interfaces, producing smoother decision functions.

Acknowledgment

The authors would like to thank Chris Anderson for providing the code for the Rayleigh-Chebyshev procedure of [1]. This work was supported by ONR grants N000141210838, N000141210040, N0001413WX20136, AFOSR MURI grant FA9550-10-1-0569, NSF grants DMS-1118971 and DMS-0914856. Ekaterina Merkurjev is also supported by an NSF graduate fellowship.

References

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

[2] G. Barles and C. Georgelin, "A simple proof of convergence for an approximation scheme for computing motions by mean curvature," *SIAM Journal on Numerical Analysis*, vol. 32, no. 2, pp. 484–500, 1995.

[3] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *The Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, 2006.

[4] L. Bertelli, S. Chandrasekaran, F. Gibou, and B. S. Manjunath, "On the length and area regularization for multiphase level set segmentation," *Int. J. Comput. Vis.*, no. 90, pp. 267–282, 2010.

[5] A. Bertozzi and A. Flenner, "Diffuse interface models on graphs for classification of high dimensional data," *Multiscale Modeling & Simulation*, vol. 10, no. 3, pp. 1090–1118, 2012.

[6] A. Bertozzi and Y. van Gennip, "Gamma-convergence of graph Ginzburg-Landau functionals," *Advanced in Differential Equations*, vol. 17, no. 11–12, pp. 1115–1180, 2012.

[7] A. Bertozzi, S. Esedoḡlu, and A. Gillette, "Impainting of binary images using the Cahn-Hilliard equation," *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 285–291, 2007.

[8] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222 –1239, 2001.

[9] X. Bresson, T. Laurent, D. Uminsky, and J. H. von Brecht, "Convergence and energy landscape for Cheeger cut clustering," *Advances in Neural Information Processing Systems*, 2012.

[10] X. Bresson, X.-C. Tai, T. F. Chan, and A. Szlam, "Multi-class transductive learning based on ℓ^1 relaxations of Cheeger cut and Mumford-Shah-Potts model," *UCLA CAM Report 12-03*, 2012.

[11] T. Bühler and M. Hein, "Spectral clustering based on the graph p-Laplacian," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 81–88.

[12] A. Cardoso, "Datasets for single-label text categorization." [Online]. Available: http://www.ist.utl.pt/˜acardoso/datasets/

[13] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006. [Online]. Available: http://www.kyb.tuebingen.mpg.de/ssl-book

[14] Y. Chen and X. Ye, "Projection onto a simplex," *arXiv preprint arXiv:1101.6081*, 2011.

[15] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997, vol. 92.

[16] D. Cireşan, U. Meier, J. Masci, L. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Two*. AAAI Press, 2011, pp. 1237–1242.

[17] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to extract symbolic knowledge from the world wide web," in *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*. AAAI Press, 1998, pp. 509–516. [Online]. Available: http://www.cs.cmu.edu/˜webkb

[18] D. Decoste and B. Schölkopf, "Training invariant support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 161–190, 2002.

[19] T. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *arXiv preprint cs/9501101*, 1995.

[20] J. A. Dobrosotskaya and A. L. Bertozzi, "A wavelet-Laplace variational technique for image deconvolution and inpainting," *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 657–663, 2008.

[21] ——, "Wavelet analogue of the Ginzburg-Landau energy and its Γ-convergence," *Interfaces and Free Boundaries*, vol. 12, no. 2, pp. 497–525, 2010.

[22] S. Esedoḡlu and Y. Tsai, "Threshold dynamics for the piecewise constant Mumford–Shah functional," *Journal of Computational Physics*, vol. 211, no. 1, pp. 367–384, 2006.

[23] L. C. Evans, "Convergence of an algorithm for mean curvature motion," *Indiana University Mathematics Journal*, vol. 42, no. 2, pp. 533–557, 1993.

[24] D. J. Eyre, “An unconditionally stable one-step scheme for gradient systems,” http://www.math.utah.edu/˜eyre/research/methods/papers.html, 1998.

[25] C. Garcia-Cardona, A. Flenner, and A. G. Percus, “Multiclass diffuse interface models for semi-supervised learning on graphs,” in *Proceedings of the 2th International Conference on Pattern Recognition Applications and Methods*. SciTePress, 2013.

[26] H. Garcke, B. Nestler, B. Stinner, and F. Wendler, “Allen-Cahn systems with volume constraints,” *Mathematical Models and Methods in Applied Sciences*, vol. 18, no. 8, 2008.

[27] G. Gilboa and S. Osher, “Nonlocal operators with applications to image processing,” *Multiscale Modeling & Simulation*, vol. 7, no. 3, pp. 1005–1028, 2008.

[28] T. Goldstein and S. Osher, “The split Bregman method for L_1-regularized problems,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[29] T. Hastie and R. Tibshirani, “Classification by pairwise coupling,” *The Annals of Statistics*, vol. 26, no. 2, pp. 451–471, 1998.

[30] M. Hein and S. Setzer, “Beyond spectral clustering - tight relaxations of balanced graph cuts,” in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 2366–2374.

[31] T. Joachims *et al.*, “Transductive learning via spectral graph partitioning,” in *International Conference on Machine Learning*, vol. 20, no. 1, 2003, p. 290.

[32] B. Kégl and R. Busa-Fekete, “Boosting products of base classifiers,” in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 497–504.

[33] R. Kohn and P. Sternberg, “Local minimizers and singular perturbations,” *Proc. Roy. Soc. Edinburgh Sect. A*, vol. 111, no. 1-2, pp. 69–84, 1989.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[35] Y. LeCun and C. Cortes, “The MNIST database of handwritten digits.” [Online]. Available: http://yann.lecun.com/exdb/mnist/

[36] J. Lellmann, J. H. Kappes, J. Yuan, F. Becker, and C. Schnörr, “Convex multi-class image labeling by simplex-constrained total variation,” IWR, University of Heidelberg, Technical Report, October 2008. [Online]. Available: http://www.ub.uni-heidelberg.de/archiv/8759/

[37] Y. Li and J. Kim, “Multiphase image segmentation using a phase-field model,” *Computers & Mathematics with Applications*, vol. 62, no. 2, pp. 737–745, 2011.

[38] E. Merkurjev, T. Kostic, and A. Bertozzi, “An MBO scheme on graphs for segmentation and image processing,” *Submitted*, 2013.

[39] B. Merriman, J. K. Bence, and S. J. Osher, “Motion of multiple functions: a level set approach,” *J. Comput. Phys.*, vol. 112, no. 2, pp. 334–363, 1994. [Online]. Available: http://dx.doi.org/10.1006/jcph.1994.1105

[40] B. Mohar, “The Laplacian spectrum of graphs,” *Graph Theory, Combinatorics, and Applications*, vol. 2, pp. 871–898, 1991.

[41] S. Nene, S. Nayar, and H. Murase, “Columbia Object Image Library (COIL-100),” *Technical Report CUCS-006-96*, 1996. [Online]. Available: http://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php

[42] K. Ni, B. Hong, S. Soatto, and T. Chan, “Unsupervised multiphase segmentation: A recursive approach,” *Computer Vision and Image Understanding*, vol. 113, no. 4, pp. 502–510, 2009.

[43] P. Perona and L. Zelnik-Manor, “Self-tuning spectral clustering,” *Advances in Neural Information Processing Systems*, vol. 17, pp. 1601–1608, 2004.

[44] J. Rubinstein, P. Sternberg, and J. Keller, “A simple proof of convergence for an approximation scheme for computing motions by mean curvature,” *SIAM Journal of Applied Mathematics*, vol. 49, no. 1, pp. 116–133, 1989.

[45] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[46] B. Simons, *Phase Transitions and Collective Phenomena*, http://www.tcm.phy.cam.ac.uk/ bds10/phase.html, University of Cambridge, 1997.

[47] A. Subramanya and J. Bilmes, “Semi-supervised learning with measure propagation,” *Journal of Machine Learning Research*, vol. 12, pp. 3311–3370, 2011.

[48] A. Szlam and X. Bresson, “A total variation-based graph clustering algorithm for Cheeger ratio cuts,” in *Proceedings of the 27th International Conference on Machine Learning*. Citeseer, 2010, pp. 1039–1046.

[49] A. D. Szlam, M. Maggioni, and R. R. Coifman, “Regularization on graphs with function-adapted diffusion processes,” *Journal of Machine Learning Research*, vol. 9, pp. 1711–1739, 2008.

[50] L. Vese and T. Chan, “A multiphase level set framework for image segmentation using the Mumford and Shah model,” *International Journal of Computer Vision*, vol. 50, no. 3, pp. 271–293, 2002.

[51] U. Von Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[52] J. Wang, T. Jebara, and S. Chang, “Graph transduction via alternating minimization,” in *Proceedings of the 25th international conference on Machine learning*. Citeseer, 2008, pp. 1144–1151.

[53] A. L. Yuille and A. Rangarajan, “The concave-convex procedure (CCCP),” *Neural Computation*, vol. 15, no. 4, pp. 915–936, 2003.

[54] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, “Learning with local and global consistency,” in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004, pp. 321–328.

[55] D. Zhou and B. Schölkopf, “A regularization framework for learning from graph data,” in *Workshop on Statistical Relational Learning*. Banff, Canada: International Conference on Machine Learning, 2004.

[56] X. Zhu, “Semi-supervised learning literature survey,” University of Wisconsin-Madison, Technical Report 1530, Computer Sciences, 2005.

Cristina Garcia-Cardona is a PhD student in the Joint Degree Program in Computational Science at Claremont Graduate University and San Diego State University. She obtained her Bachelor’s degree in Electrical Engineering from Universidad de Los Andes in Colombia and her Master’s degree in Emergent Computation from Universidad Central de Venezuela. She is currently working under the supervision of Prof. Allon Percus and Dr. Arjuna Flenner. Her research interests include energy minimization and graph algorithms.

Ekaterina Merkurjev is a third year graduate student at the UCLA Department of Mathematics. She obtained her Bachelors and Masters degrees in Applied Mathematics from UCLA in 2010. She is currently working on a PhD under the supervision of Prof. Andrea Bertozzi. Her research interests include image processing and segmentation.

Andrea L. Bertozzi received the BA, MA, and PhD degrees in mathematics from Princeton University, Princeton, NJ, in 1987, 1988, and 1991 respectively. She was on the faculty of the University of Chicago, Chicago, IL, from 1991-1995 and Duke University, Durham, NC, from 1995-2004. During 1995-1996, she was the Maria Goeppert-Mayer Distinguished Scholar at Argonne National Laboratory. Since 2003, she has been with the University of California, Los Angeles, as a Professor of Mathematics and currently serves as the Director of Applied Mathematics. In 2012 she was appointed the Betsy Wood Knapp Chair for Innovation and Creativity. Her research interests include image inpainting, image segmentation, cooperative control of robotic vehicles, swarming, and fluid interfaces, and crime modeling. Prof. Bertozzi is a Fellow of both the Society for Industrial and Applied Mathematics and the American Mathematical Society; she is a member of the American Physical Society. She has served as a Plenary/Distinguished Lecturer for both SIAM and AMS and is an Associate Editor for the SIAM journals Multiscale Modelling and Simulation, Mathematical Analysis. She also serves on the editorial board of Interfaces and Free Boundaries, Applied Mathematics Research Express, Nonlinearity, Appl. Math. Lett., Math. Mod. Meth. Appl. Sci. (M3AS), J. Nonlinear Sci, J. Stat. Phys., Comm. Math. Sci., Nonlinear Anal. Real World Appl., and Adv. Diff. Eq. Her past honors include a SloanFoundation Research Fellowship, the Presidential Career Award for Scientists and Engineers, and the SIAM Kovalevsky Prize in 2010.

Arjuna Flenner received his Ph.D. in Physics at the University of Missouri-Columbia in 2004. His major emphasis was mathematical physics. Arjuna Flenner's research interests at the Naval Air Weapons Centre at China Lake include image processing, machine learning, statistical pattern recognition, and computer vision. In particular, he has investigated automated image understanding algorithms for advanced naval capabilities. His main research areas are non-local operators, geometric diffusion, graph theory, non-parametric Bayesian analysis, and a-contrario hypothesis testing methods. Arjuna Flenner was a US Department of Energy GAANN Fellowship in 1997-2001, and currently is also a visiting research professor at Claremont Graduate University.

Allon G. Percus received his BA in physics from Harvard in 1992 and his PhD from the Université Paris-Sud, Orsay in 1997. He was a member of the scientific staff at Los Alamos National Laboratory in the Division of Computer and Computational Sciences, and from 2003 to 2006 he served as Associate Director of the Institute for Pure and Applied Mathematics at UCLA. Since 2009, he has been Associate Professor of Mathematics at Claremont Graduate University. His research interests combine discrete optimization, combinatorics and statistical physics, exploiting physical models and techniques to study the performance of algorithms on NP-hard problems.