Title
Global Binary Optimization on Graphs for Classification of High Dimensional Data

Authors
Merkurjev, Ekaterina
Bae, Egil
Bertozzi, Andrea L
et al.

Publication Date
2014-09-01

DOI
10.21236/ada610270

Peer reviewed

(will be inserted by the editor)

Global binary optimization on graphs for classification of high dimensional data

Ekaterina Merkurjev · Egil Bae · Andrea L. Bertozzi · Xue-Cheng Tai

Received: date / Accepted: date

Abstract This work develops a global minimization framework for segmentation of high dimensional data into two classes. It combines recent convex optimization methods from imaging with recent graph based variational models for data segmentation. Two convex splitting algorithms are proposed, where graph-based PDE techniques are used to solve some of the subproblems. It is shown that global minimizers can be guaranteed for semi-supervised segmentation with two regions. If constraints on the volume of the regions are incorporated, global minimizers cannot be guaranteed, but can often be obtained in practice and otherwise be closely approximated. Experiments on benchmark data sets show that our models produce segmentation results that are comparable with or outperform the state-of-the-art algorithms. In particular, we perform a thorough comparison to recent MBO (Merriman-Bence-Osher) [49] and phase field methods, and show the advantage of the algorithms proposed in this paper.

Keywords classification · graphs · combinatorial optimization · convex optimization

E. Bae is supported by the Norwegian Research Council eVita project 214889. E. Merkurjev is supported by the National Science Foundation Graduate Fellowship (NSF). X.-C. Tai is supported by the Christian Michelsen Research (CMR), Bergen. This work was supported by AFOSR MURI grant FA9550-10-1-0569, NSF grant DMS-111897, ONR grants N000141210040 and N000141210838, and the W. M. Keck Foundation.

Ekaterina Merkurjev · Egil Bae · Andrea Bertozzi
University of California, Los Angeles
405 Hilgard Avenue, 90095, Los Angeles, USA
Fax: (310) 206-6673
E-mail: {kmerkurev,ebae,bertozzi}@math.ucla.edu

Xue-Cheng Tai
University of Bergen
Johaness Brunsgate 12, 5007 Norway
E-mail: tai@mi.uib.no

1 Introduction

We consider the problem of clustering general high-dimensional data into two classes. The data points are viewed as nodes on a graph and the similarity between them is represented by the weight function defined on the edges between the nodes. Recently, there has been a growing interest in formulating such problems as variational or optimization problems, where the non-local total variation term plays a fundamental role in constructing the cost function.

A graphical framework is often used to exploit underlying similarities in the data [3,17,59,63–65]. For example, spectral graph theory [19, 50] uses this approach to perform various tasks in imaging and data clustering. The graph Laplacian, one of its fundamental concepts, is described in section 2.

Graph-based formulations have been used extensively for image processing applications [6, 21, 22, 25, 34, 35, 37, 46, 54]. A typical framework involves the similarity graph where each two vertices are given a weight measuring their similarity. Buades et al. in [13] introduce a new non-local means algorithm for image denoising and compare it to some of the best methods. In [35], Grady describes a random walk algorithm for image segmentation using the solution to a Dirichlet problem. Elmoataz et al. present generalizations of the graph Laplacian [25] for image denoising and manifold smoothing. Couprie et al. in [21] propose a parameterized graph-based energy function that unifies graph cuts, random walker, shortest paths and watershed optimizations. We use a non-local calculus formulation [28] to generalize the continuous formulation to a (non-local) discrete setting, while other non-local versions for weighted graphs are described in [25]. A comprehensive reference

about casting continuous PDEs in graph form is found in [36].

Our work involves semi-supervised clustering, where the labeling of a small set of the data points is provided in advance. Such problems have been studied in [4,48] in a variational framework, where a Ginzburg-Landau functional was defined on the graph and minimized by PDE techniques, such as phase field [4] and the MBO (Merriman-Bence-Osher) scheme [48]. MBO was originally formulated in the Euclidean space as a numerical scheme for solving evolution equations involving mean curvature motion of a region boundary [49] and has also been used for image segmentation [58]. Since the energy functional is non-convex and the PDEs evolve in the steepest descent direction, these approaches may potentially get stuck in unwanted local minima.

On the other hand, in unsupervised clustering, there are no a priori knowledge of the labeling of some of the data points. In such cases, some knowledge of the sizes of each class is typically incorporated in order to prevent the trivial solution where all data points are assigned to the same class. The normalized cut and the Cheeger ratio cut [18,54] are two popular such cost functions that favor clusterings with classes of equal size. Recent work has been focused on greatly simplifying the energy landscape [10, 12, 39, 55, 56] in such problems by writing them as constrained binary optimization problems involving the total variation on graphs. Even though the simplified problems are not completely convex, experiments demonstrate that they can often avoid bad local minima. Supervised constraints can theoretically also be incorporated into the framework [9], although the experiments were focused on undersupervised clustering. Another interesting paper [47] generalized well known variational image segmentation models consisting of a fidelity and regularization term to graphs, paying particular attention to the Chan-Vese model [15] in experiments, and generalized recent convex relaxation methods [16] for the graph versions of such models.

It is well known in combinatorial optimization that certain graph partition problems can be formulated as min-cut problems [27], which aim to find the minimal separation of the graph into two sets, one of them containing a predefined source node and the other a predefined sink node. The max-flow problem is the equivalent dual problem and can be globally optimized by classical combinatorial algorithms such as Ford-Fulkerson [27] or the push-relabel method [31]. Specialized versions of such algorithms have recently become popular for solving certain optimization problems in image processing and computer vision [5, 6, 43].

There are some fundamental differences between the imaging applications and more general clustering problems on graphs. In imaging, most of the data is incorporated in a strong fidelity term, which measures how well each pixel fits to each region. Edges are also defined between neighboring data points on a regular grid, but they are mainly used for smoothing purposes. In more general clustering problems, the data is mainly incorporated on edges between pairs of data points, and the fidelity term is zero at the majority of the data points.

The combinatorial max-flow algorithm of [5] was developed with specific imaging problems and a regular grid in mind. We anticipate that it is not easily transferable to graph clustering while still maintaining a high efficiency; one reason being that the paths between the source node and sink node are much longer. Such combinatorial algorithms also do not exploit PDE techniques to approximate the large graph, like the approximate eigendecomposition of the graph Laplacian as in [4, 48].

This paper proposes an efficient global optimization framework for semi-supervised clustering problems, formulated in the same variational form as [4, 48]. Instead of applying classical combinatorial algorithms, we build on more recent work from imaging, which formulates two class partition problems as convex variational problems [8, 16, 32] or variational min-cut/max-flow problems [61, 62]. Convex optimization algorithms were used in [8, 32, 61, 62] to split the problems into simpler subproblems, each of which could be solved by PDE techniques. In this paper, we describe the extension of the variational min-cut/max-flow duality in [61, 62] and of the algorithm in [32, 60] to a more general graph setting to solve a more general clustering problem. Here, the two global minimization methods are referred to as "max-flow" and "primal augmented Lagrangian" algorithms, respectively. The new subproblems are solved by graph-based PDE techniques. We also show how constraints on the size of each class can be incorporated by a small modification of the max-flow algorithm.

Our global minimization algorithms are tested on several benchmark data sets, and we compare them with the phase field [4] and MBO [48] algorithms as well as with each other. One notable finding is that if the known data points are not distributed relatively uniformly among the entire data set, the local minimization methods may have difficulty finding the correct solution.

The paper is organized as follows. In section 2, we review the graphical framework and some previous related work. In section 3, we present our novel graph-based clustering methods. The results are shown in section 4. We conclude in section 5.

2 The Graphs Framework

We consider the data as vertices on a graph. Let G be an undirected graph $G = (V, E)$, where V and E are the sets of vertices and edges, respectively. Each edge is equipped with a weight denoted by the weight function $w(x, y)$, which satisfies the symmetric property and measures the similarity between vertices x and y. A big value of $w(x, y)$ indicates that nodes x and y are very similar, while a small value of $w(x, y)$ indicates that they are dissimilar, and thus less likely to belong to the same class. The challenge is to use the right weight function- the one that measures the important classification attributes of each pair of vertices.

One popular choice for the weight function is the Gaussian

$$w(x, y) = e^{-\frac{d(x,y)^2}{\sigma^2}}, \tag{1}$$

where $d(x, y)$ 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 $\mathbb{R}^2$, $d(x, y)$ can be the Euclidean distance between point x and point y, since points farther away are less likely to belong to the same cluster than points closer together. For images, $d(x, y)$ can be defined as the weighted 2-norm of the difference of the feature vectors of pixels x and y, where the feature vector of a node consists of intensity values of pixels in its neighborhood, as described in [30].

Another choice for the similarity function used in this work is the Zelnik-Manor and Perona weight function [52] for sparse matrices:

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

where the local parameter $\tau(x) = d(x, z)^2$, and z is the M^{th} closest vertex to vertex x.

Note that it is not necessary 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 edges in E between the M nearest neighbors for each node in V, i.e. only the edges with large weight w. In this paper, we make use of such an approximation; our edge set includes only edges between vertices that are near to each other. Specifically, we include an edge between vertices x and y only if x is among M nearest neighbors of y or vice versa.

The matrix $\mathbf{W}$ is defined as $W_{xy} = w(x, y)$ and the degree of a vertex $x \in V$ as

$$d(x) = \sum_{y \in V} w(x, y). \tag{3}$$

We let $\mathbf{D}$ to be the diagonal matrix with elements $d(x)$ on the diagonal.

We use a graphical framework because it simplifies the processing of high-dimensional data and provides a way to deal with nonlinearly separable classes.

2.1 Well known operators in graph form

We define operators on graphs in a similar fashion as done in [28, 38], where the justification for these choices is shown.

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

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

for some $r \in [0, 1]$ and $q \in [\frac{1}{2}, 1]$. Let us also define the following norms:

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

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

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

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

$$(\nabla\lambda)_w(x, y) = w(x, y)^{1-q}(\lambda(y) - \lambda(x)). \tag{4}$$

The Dirichlet energy does not depend on r or q:

$$\frac{1}{2}\|\nabla\lambda\|_{\mathcal{E}}^2 = \frac{1}{4}\sum_{x,y} w(x, y)(\lambda(x) - \lambda(y))^2. \tag{5}$$

The divergence $div : \mathcal{E} \to \mathcal{V}$ is defined as the adjoint of the gradient:

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

(6)

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 family of graph Laplacians $\triangle_r = \mathrm{div}_w\, \dot{\nabla} : \mathcal{V} \to \mathcal{V}$:

$$(\triangle_w \lambda)(x) = \sum_y \frac{w(x,y)}{d(x)^r}(\lambda(y) - \lambda(x)). \tag{7}$$

Viewing λ as a vector in $\mathbb{R}^m$, we can write

$$-\triangle_w \lambda = (\mathbf{D}^{1-r} - \mathbf{D}^{-r}\mathbf{W})\lambda. \tag{8}$$

The case with $r = 0$ is the unnormalized Laplacian

$$\mathbf{L} = \mathbf{D} - \mathbf{W}. \tag{9}$$

However, the matrix $\mathbf{L}$ is usually scaled to guarantee convergence to the continuum differential operator in the limit of large sample size [4]. Although several versions exist, two popular versions are the symmetric Laplacian

$$\mathbf{L_s} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}} \tag{10}$$

and the random walk Laplacian ($r = 1$)

$$\mathbf{L_{rw}} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D^{-1}}\mathbf{W}. \tag{11}$$

The advantage of the former formulation is its symmetric property which allows for more efficient implementations.

A family of anisotropic total variations $TV_w : \mathcal{V} \to \mathbb{R}$ can now be defined:

$$\begin{aligned} TV_w(\lambda) &= \max\left\{\langle \mathrm{div}_w\, \phi, \lambda\rangle_{\mathcal{V}} : \phi \in \mathcal{E}, \|\phi\|_{\mathcal{E},\infty} \le 1\right\} \\ &= \frac{1}{2}\sum_{x.y} w(x,y)^q |\lambda(x) - \lambda(y)|. \end{aligned} \tag{12}$$

It remains to choose the parameters q and r. We choose $q = 1$ as in [28], where it is shown that for any r, 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*

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

$$GL_0(\lambda) = \begin{cases} TV_w(\lambda) \textit{with } q\textit{=1} & \textit{for } \lambda \textit{ s.t. } \lambda(x) \in \{0,1\} \\ \infty & \textit{otherwise} \end{cases}$$

Proof. See Theorem 3.1 of [28]. □

Here W is the double-well potential $W(u) = u^2(u-1)^2$ having two zeros (in our case 0 and 1), and ϵ is a small positive number. It is also shown in the paper (specifically Theorem 3.6) that *the addition of a fidelity term is compatible with Γ-convergence*. Since one of the algorithms we compare our methods to deals with the Ginzburg-Landau functional directly, to be consistent, we use the above definitions with $q = 1$ in our formulations.

Remark. *It is noted in [28] that although the first term in the continuous Ginzburg-Landau functional*

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

is scaled by ϵ, the first term of GL_ϵ contains no ϵ. This occurs because the Dirichlet energy in GL_c is unbounded for functions λ of bounded variation and taking on two values of the minima of the double-well potential (almost everywhere). However, the difference terms of GL_ϵ are finite even in the case of binary functions, and no rescaling of the first term is necessary.

We choose $r = 1$ because it results in a normalized random walk Laplacian and the eigenvectors as well as the corresponding eigenvalues of the matrix can be efficiently calculated. Although the random walk Laplacian matrix itself is not symmetric, spectral graph theory described in [19] shows that the eigenvectors of the random walk Laplacian can be directly computed from knowing the diagonal matrix $\mathbf{D}$ and the eigenvectors of the symmetric graph Laplacian (which is a symmetric matrix) $\mathbf{L_s}$. In particular, 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 $w = \mathbf{D}^{\frac{1}{2}}u$. This is proved by multiplying the eigenvalue equation $\mathbf{L_{rw}}u = \lambda u$ by $\mathbf{D}^{\frac{1}{2}}$ from the left and then substituting $w = \mathbf{D}^{\frac{1}{2}}u$, obtaining $\mathbf{L_s}w = \lambda w$.

We take advantage of this property by calculating the eigenvalues and eigenvectors of the symmetric graph Laplacian (since symmetric matrices allow for more efficient implementations) and then using this information to calculate the same for the random walk Laplacian.

To summarize, we use the above operator definitions with $q = 1$ and $r = 1$.

In this work, we use the notation $\lambda(x)$ to denote the value of λ at node $x \in V$ that provides information about the class membership of the node. Specifically, we use $\lambda(x) = 0$ to denote the fact that node x belongs to class 1, and $\lambda(x) = 1$ to denote that it belongs to class 2.

2.2 Partition problems on graphs

In this work, we are interested in solving partition problems of the form

$$\min_{S\subset V} \sum_{(x,y)\in E\,:\,x\in S,\ y\in V\setminus S} w(x,y), \tag{13}$$

which is the formulation of the minimum cut problem, under supervised constraints

$$S \supseteq V^f, \quad V\setminus S \supseteq V^b \tag{14}$$

and an optional volume constraint

$$|S| = a|V|,$$

where $0 < a < 1$. $V^f \subset V$ is a set of nodes that are known a priori to belong to the region S and $V^b \subset V$ is a set of nodes that are known to belong to region $V\setminus S$. Variations of this problem have been vastly explored in literature. For example, in [11], the authors describe an algorithm which minimizes a normalized version of the cut, specifically the balanced cut. A multiclass version of this method is introduced in [9].

By defining a binary function

$$\lambda(x) := \begin{cases} 1, & x \in S \\ 0, & x \in V\setminus S \end{cases}$$

the above problem can be expressed as

$$\min_{\lambda\in\mathcal{B}} E^P(\lambda) = TV_w(\lambda) + \sum_{x\in V} f(\lambda(x), x), \tag{15}$$

where

$$TV_w(\lambda) = \frac{1}{2}\sum_{x.y} w(x,y)|\lambda(x) - \lambda(y)|$$

as defined earlier (with $q = 1$) and

$$\mathcal{B} = \{\lambda : V \mapsto \{0,1\}\} \tag{16}$$

is the set of binary functions indicating the partition. Here, $f(\lambda(x), x)$ is a fidelity term which incorporates the supervised constraints (14). It typically takes the form of

$$f(\lambda(x), x) = \eta(x)|\lambda(x) - \lambda^0(x)|^2, \tag{17}$$

where λ^0 is a binary function taking value 1 in V^f and 0 in V^b, and $\eta(x)$ is a function that takes on a large constant value η on fidelity points $V^f \cup V^b$ and zero elsewhere. If η is chosen sufficiently large, it can be guaranteed that the solution λ satisfies the supervised constraints. In [47], the authors propose solving a similar minimization problem by introducing nonlocal global minimizers of active contour models on graphs.

In addition, when the size of the two classes is known, the volume of the regions may be enforced to satisfy a constraint of the form

$$\sum_{x\in V} \lambda(x) = a|V|, \tag{18}$$

where a is the fraction of the nodes belonging to class 2, e.g. $a = \frac{1}{2}$ enforces partitions of equal volume. The goal of this is to create an algorithm that requires a much smaller fidelity set (to produce an accurate classification) than otherwise, because we also have the information about class size.

In previous work [4], the problem (15) was formulated as the minimization of a Ginzburg-Landau (GL) functional on graphs with a fidelity term

$$GL_\epsilon(\lambda) = \|\nabla\lambda\|_{\mathcal{E}}^2 + \frac{1}{\epsilon}\sum_x W(\lambda(x)) + f(\lambda(x), x), \tag{19}$$

where

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

as defined before. Note that as $\epsilon \to 0$, in the limit of gamma convergence, the sum of first two terms of the energy converge to the total variation term, making the energy exactly the same as one in (15). The problem is solved using gradient descent and an efficient convex splitting scheme. This method will be referred to as "binary GL" in the paper, and we compare it to our work.

In [48], (19) is solved numerically by a variation of the MBO scheme [49], a method to approximate motion by mean curvature. To make everything consistent with the notation and theorems stated in the paper, we include an extra scaling in our implementation of the method in [48], and the justification is described shortly. We note that this change in the method did not exacerbate the results as compared to those of the original method; in fact, it produced very little change in any simulation. This algorithm will be referred to as "binary MBO" in the paper, and we compare it to our new algorithms. The discretized version of the algorithm is the following:

Starting with some initial classification $\lambda \in \{0,1\}$, alternate between the following two steps until the stopping criterion is satisfied:

1. Heat equation with forcing term:

$$\frac{\lambda^{n+\frac{1}{2}} - \lambda^n}{dt} = 2\triangle_w\lambda^{n+1} - \frac{1}{d(x)^r}\frac{\partial f(\lambda(x), x)}{\partial\lambda}. \tag{20}$$

2. Thresholding:

$$\lambda^{n+1}(x) = \begin{cases} 1, & \text{if } \lambda^{n+\frac{1}{2}}(x) \geq 0.5, \\ 0, & \text{if } \lambda^{n+\frac{1}{2}}(x) < 0.5. \end{cases} \quad (21)$$

Here, after the second step, $\lambda^{n+1}(x)$ can take only two values of 1 or 0; thus, this method is appropriate for binary segmentation.

Following [48], (20) is solved by a semi-implicit scheme, where the Laplacian term is calculated implicitly, and the terms are considered as a linear combination of the eigenvectors of the random walk Laplacian.

To show the general idea of the derivation, we start with the Ginzburg-Landau (GL) functional on graphs (19). One can rewrite it using inner product notation:

$$GL_\epsilon(\lambda) = \|\nabla\lambda\|_{\mathcal{E}}^2 + \frac{1}{\epsilon}\langle D^{-r}W(\lambda(x)), 1\rangle_{\mathcal{V}} + \langle D^{-r}f(\lambda(x), x), 1\rangle_{\mathcal{V}}, \quad (22)$$

where $(D^{-r}W(\lambda))(x) = d(x)^{-r}W(\lambda(x))$. The factor $d(x)^{-r}$ is needed to cancel the factor $d(x)^r$ in the $\mathcal{V}$- inner product.

The Allen-Cahn equation can then be derived using the $\mathcal{V}$-gradient flow associated with GL_ϵ. We have

$$\frac{d}{dt}GL_\epsilon(\lambda + t\gamma)|_{t=0} = -2\langle \triangle_w\lambda, \gamma\rangle_{\mathcal{V}} + \frac{1}{\epsilon}\langle D^{-r}W'(\lambda(x)), \gamma\rangle_{\mathcal{V}} + \langle D^{-r}\frac{\partial f}{\partial\lambda}(\lambda(x), x), \gamma\rangle_{\mathcal{V}}. \quad (23)$$

The Allen-Cahn equation is then

$$\dot{\lambda}(x) = 2\triangle_w\lambda - \frac{1}{\epsilon d(x)^r}W'(\lambda(x)) - \frac{1}{d(x)^r}\frac{\partial f}{\partial\lambda}(\lambda(x), x). \quad (24)$$

The above equation can be solved using a time-splitting scheme, where the splitting occurs so that the double-well potential term is separated. The first step is

$$\dot{\lambda}(x) = 2\triangle_w\lambda - \frac{1}{d(x)^r}\frac{\partial f}{\partial\lambda}(\lambda(x), x) \quad (25)$$

and the second step (with the double-well potential) is just thresholding in the $\epsilon \to 0$ limit. By alternating between these two steps, one can form an approximate solution of the Allen-Cahn equation (24).

Such approaches (binary GL and binary MBO methods) converge to the nearest local minimizer from a given initialization. In general, one cannot guarantee that the desired global minimizer is obtained. The subject of this work is to develop a convex optimization framework for minimizing (15) which is guaranteed to obtain the global minimizer. Various algorithms are developed for solving the convex problems.

3 Global optimization for partition problems on graphs

The problem (15) is non-convex because the binary side constraints (16) are non-convex. We show that the binary constraints can be replaced by their convex hull $[0, 1]$ to obtain an exact convex formulation as was shown in [16] for images. Define first the functions

$$C_s(x) = f(0, x), \quad C_t(x) = f(1, x), \quad \forall x \in V,$$
$$g(\phi(x), x) = C_t(x)\phi(x) + C_s(x)(1 - \phi(x)), \ \forall x \in V. \quad (26)$$

The problem

$$\min_{\lambda\in\mathcal{B}} E^P(\lambda) = TV_w(\lambda) + \sum_{x\in V} g(\lambda(x), x) \quad (27)$$

is equivalent to the formulation (15). The proof is obvious as $g(\phi(x), x) = f(\phi(x), x)$ for all binary ϕ.

The convex relaxed problem is formulated as follows:

$$\min_{\lambda\in\mathcal{B}'} E^P(\lambda) = TV_w(\lambda) + \sum_{x\in V} g(\lambda(x), x), \quad (28)$$

where

$$\mathcal{B}' = \{\lambda : V \mapsto [0, 1]\}. \quad (29)$$

In case of images and local differential operators, it was shown in [16] theorem 1 that the minimizer of the convex problem can be thresholded to yield a binary global minimizer of the original problem. Generalizations were proposed in [7] to continuous manifolds arising from patch based non-local operators. A generalization of the theorem to discrete graphs was proposed in [47], although a formal proof was not included. Here we state the same result as in [47] and give a complete proof.

Theorem 2 *Let λ^* be a minimizer of (28). Denote by $\lambda^\ell : V \mapsto \{0, 1\}$ the binary function*

$$\lambda^\ell(x) = \begin{cases} 1, & \text{if } \lambda^*(x) \geq \ell \\ 0, & \text{if } \lambda^*(x) < \ell \end{cases}. \quad (30)$$

Then for almost every $\ell \in (0, 1]$, λ^ℓ is a global minimizer of the non-convex problem (27).

Proof. For any function $\lambda \in \mathcal{B}'$ and for any $x \in V$, if $\lambda(x) \in [0,1]$, then $\int_0^1 \lambda^\ell(x)\, d\ell = \lambda(x)$. Therefore, for each $x \in V$,

$$\begin{aligned}&\int_0^1 g(\lambda^\ell(x), x)\, d\ell = \\ &\int_0^1 \lambda^\ell(x) C_t(x) + (1-\lambda^\ell(x)) C_s(x)\, d\ell \\ &\quad = \lambda(x) C_t(x) + (1-\lambda(x)) C_s(x) = g(\lambda(x), x).\end{aligned} \tag{31}$$

By the coarea formula, we have that

$$\int_0^1 TV_w(\lambda^\ell)\ d\ell = TV_w(\lambda)\,.$$

For a proof of the coarea formula on graphs, see Appendix B of [29].

Combining the above properties, we obtain that for any $\lambda \in \mathcal{B}'$,

$$\begin{aligned}\int_0^1 E^P(\lambda^\ell)\, d\ell = \sum_{x\in V} \int_0^1 (TV_w(\lambda^\ell) + g(\lambda^\ell(x), x)) d\ell = \\ \sum_{x\in V} TV_w(\lambda) + g(\lambda(x), x) = E^P(\lambda).\end{aligned} \tag{32}$$

For a λ that minimizes the energy, clearly $E^P(\lambda) \leq E^P(\lambda^\ell)$ for any $\ell \in (0,1]$. However, equality (32) can then only be true provided $E^P(\lambda) = E^P(\lambda^\ell)$ for almost every $\ell \in (0,1]$. In other words, λ^ℓ also minimizes the energy for almost every $\ell \in (0,1]$. □

In order to solve (28), we consider two main algorithms. The first is based on solving a dual formulation of the problem, which can be identified as a maximum-flow problem, by convex optimization techniques. It will be referred to as the "max-flow" method in this paper. We present three versions of this algorithm: one without hard supervised constraints (Algorithm 1), one with hard supervised constraints (Algorithm 1s), and one with balancing constraints (Algorithm 1b). The second algorithm (Algorithm 2) solves the primal problem by the augmented Lagrangian technique, and will be denoted the "primal augmented Lagrangian" method in this paper.

3.1 Max-flow algorithm without balancing constraints

In graph theory, the max-flow problem [27] aims to maximize the flow from a source node s to a sink node t, which are both connected by edges to the nodes in V. We let $p_s, p_t : V \mapsto \mathbb{R}$ denote the flow variables on sink and source edges and $p : E \mapsto \mathbb{R}$ represent the flow on edges between pairwise points in V, where $E \subset V \times V$. The upper capacities on the source edges are denoted by C_s and on sink edges by C_t, and there is no lower bound on the capacities. The flows $p(x,y)$ on the edges (x,y) are bounded by $|p(x,y)| \leq w(x,y)$. The amount of flow in the graph can be expressed as the amount of flow on the source edges, which we want to maximize under flow capacity and flow conservation constraints. In this section, we describe two max-flow problems. The first is dual to the problem (15) with fidelity term, and consequently solves the original problem (13) provided the penality parameter η is high enough. The second max-flow problem incorporates the supervised constraints directly without the need for a very large penalty term. The following derivations extend the continuous max-flow problem [61,62] from images to general graphs.

Max-flow formulation with supervised constraints as fidelity term

The following problem can be interpreted as a max-flow problem over the graph and is shown to be dual to the convex partition problem (28).

$$\max_{p_s, p_t, p} \left\{ P(p_s, p_t, p) = \sum_{x\in V} p_s(x) \right\} \tag{33}$$

subject to

$$|p(x,y)| \leq w(x,y), \quad (x,y) \in E; \tag{34}$$

$$p_s(x) \leq C_s(x), \quad \forall x \in V; \tag{35}$$

$$p_t(x) \leq C_t(x), \quad \forall x \in V; \tag{36}$$

$$\mathrm{div}_w\, p(x) - p_s(x) + p_t(x) = 0, \quad \forall\, x \in V. \tag{37}$$

where (34) is the flow capacity constraint on edges between paiwise nodes, (35) and (36) are flow capacities on the source and sink edges, and (37) is the flow conservation constraint. The objective function (33) measures the total amount of flow on the graph. Due to constraint (35), the maximization problem (33) is bounded above by $\int_\Omega C_s(x)$, which is finite provided $f(\phi(x), x)$ is bounded (true for the data terms considered in this work).

It is well known that the maximum flow problem is equivalent to the min-cut problem, where the goal is to find a partition that minimizes the sum of the weights between vertices of the two regions. In classical max-flow min-cut theory, to obtain the final classification by solving the maximum-flow problem, one can use the information of the flow on the source and sink edges. If for $x \in V$, there is a non-saturated path between s and x,

then x is in class 1. If there is a non-saturated path between x and t, then x is in class 2.

In this paper, we instead solve the max-flow problem (33) by continuous optimization. The solution to the min-cut problem can be obtained directly from the Lagrange multiplier for the flow conservation constraint (37). Introducing such a Lagrange multiplier for the flow conservation constraint (37) leads to the the primal-dual formulation of (33):

$$\min_{\lambda} \max_{p_s,p_t,p} \Big\{ E(p_s,p_t,p;\lambda) = \sum_{x\in V} p_s(x) + \sum_{x\in V} \lambda(x)\big(\operatorname{div}_w p - p_s + p_t\big) \Big\} \quad (38)$$

subject to

$$|p(x,y)| \leq w(x,y), \quad (x,y)\in E; \quad (39)$$
$$p_s(x) \leq C_s(x), \quad \forall x \in V; \quad (40)$$
$$p_t(x) \leq C_t(x), \quad \forall x \in V; \quad (41)$$

Rearranging the terms, we obtain

$$\min_{\lambda} \max_{p_s,p_t,p} \Big\{ E(p_s,p_t,p;\lambda) = \sum_{x\in V} \big\{ (1-\lambda)p_s + \lambda p_t + \lambda \operatorname{div}_w p \big\} \Big\} \quad (42)$$

subject to (39), (40) and (41).

The integrand of the first two terms of (42) can be rewritten for each $x \in \Omega$ as

$$\sup_{p_s(x)\leq C_s(x)} ((1-\lambda)p_s)(x) = \begin{cases} ((1-\lambda)C_s)(x) & \text{if } \lambda(x) \leq 1 \\ \infty & \text{if } \lambda(x) > 1 \end{cases} \quad (43)$$

$$\sup_{p_t(x)\leq C_t(x)} \lambda(x)p_t(x) = \begin{cases} (\lambda C_t)(x) & \text{if } \lambda(x) \geq 0 \\ \infty & \text{if } \lambda(x) < 0. \end{cases} \quad (44)$$

Note that (42) is bounded above. This can be seen by defining the zero function $\emptyset(x) = 0\ \forall x \in \Omega$. From constraints (40) it follows that $\inf_\lambda P(\lambda) \leq P(\emptyset) = \int_\Omega C_s(x)\,dx$, which is finite since C_s is uniformly bounded. From (43) and (44), an optimal variable λ must therefore satisfy the constraints

$$\lambda(x) \in [0,1]\ \forall\ x \in \Omega. \quad (45)$$

Otherwise, the primal-dual energy (42) would be infinite, contradicting boundedness from above.

The last term of (42) can be rewritten using the dual formulation of total variation (12) as

$$\max_{\|p\|_{\mathcal{E},\infty}\leq 1} \langle \operatorname{div}_w p, \lambda\rangle_{\mathcal{V}} = TV_w(\lambda)\,. \quad (46)$$

Combined with the observation (45), this implies that an optimal variable λ must be contained in the set $\mathcal{B}'$.

By combining (43), (44) and (46), we see that by maximizing the above problem for p_s, p_t and p, we obtain the closed form expression (28) subject to the constraint (29). Existence of dual and primal-dual solutions follows by the minimax theorem, Prop. 2.4 of [24] Chapter VI.

Max-flow formulation with hard supervised constraints

We also describe another formulation of the problem, which avoids using a fidelity term that is forced to take on a very large value of η to enforce that λ satisfies the supervised constraints. Define first the binary functions

$$v^f(x) = \begin{cases} 1, & x \in V^f \\ 0, & \text{otherwise} \end{cases}, \quad v^b(x) = \begin{cases} 0, & x \in V^b \\ 1, & \text{otherwise} \end{cases},$$

and consider the following modification of the max-flow problem (75):

$$\max_{p_s,p_t,p} \sum_{x\in V} (v^b p_s - v^f p_t) \quad (47)$$

subject to

$$|p(x,y)| \leq w(x,y), \quad (x,y)\in E; \quad (48)$$
$$p_s(x) \leq 0, \quad \forall x \in V; \quad (49)$$
$$p_t(x) \leq 0, \quad \forall x \in V; \quad (50)$$
$$\operatorname{div}_w p(x) - p_s(x) + p_t(x) = 0, \quad \forall\ x \in V. \quad (51)$$

Introducing the Lagrange multiplier λ for constraint (37), we obtain the following Lagrangian formulation after rearrangement of the terms:

$$\min_{\lambda} \max_{p_s,p_t,p} \Big\{ E(p_s,p_t,p;\lambda) = \sum_{x\in V} \big\{ (v^b-\lambda)p_s + (\lambda - v^f)p_t + \lambda \operatorname{div}_w p \big\} \Big\} \quad (52)$$

As there are no lower bounds on p_s and p_t, it can be observed that optimal solutions λ must satisfy the constraints

$$v^f \leq \lambda \leq v^b, \quad (53)$$

otherwise the energy could be made arbitrarily large. By maximizing for the flows p_s, p_t and p, we therefore obtain the primal problem

$$\min_{\lambda\in B'} TV_w(\lambda) \quad (54)$$

subject to (53). If λ^* is a solution to (54), then λ^* is obviously also a solution to (28) provided the penalty parameter η in the fidelity term of (28) is chosen sufficiently high.

Algorithms

The dual problems (33) or (47) are solved by the augmented Lagrangian method as in [61, 62]. To solve (33), construct first the augmented Lagrangian function corresponding to (33):

$$L_c(p_s, p_t, p, \lambda) = \sum_{x\in V} \left\{ p_s + \lambda(\mathrm{div}_w\, p - p_s + p_t) \right\} - \frac{c}{2} \left\| \mathrm{div}_w\, p - p_s + p_t \right\|_2^2 , \quad (55)$$

where $\|s\|_2^2 = \sum_{x\in V} |s(x)|_2^2$. An augmented Lagrangian method can be applied by alternatively maximizing L_c for the dual variables p_s, p_t and p with constraints (39)-(41) and updating the Lagrange multiplier λ. The max-flow algorithm for (33) with supervised constraints as a fidelity term is outlined as "Algorithm 1". The max-flow algorithm for (47) with hard supervised constraints is outlined as "Algorithm 1s"

Algorithm 1 Max-flow Algorithm

Initialize p_s^1, p_t^1, p^1 and λ^1. For $k = 1, ...$ until convergence:

- Optimize p flow

$$p^{k+1} = \underset{|p(e)|\le W(e)\ \forall e\in E}{\arg\max} \; -\frac{c}{2} \left\| \mathrm{div}_w\, p - F^k \right\|_2^2 , \quad (56)$$

  where $F^k = {p_s}^k - {p_t}^k + \frac{\lambda^k}{c}$ is fixed.
- Optimize source flow p_s

$$p_s^{k+1} = \underset{p_s(x)\le C_s(x)\ \forall x\in V}{\arg\max} \sum_{x\in V} p_s - \frac{c}{2} \left\| p_s - G^k \right\|_2^2 , \quad (57)$$

  where $G^k = {p_t}^k + \mathrm{div}_w\, p^{k+1} - \frac{\lambda^k}{c}$ is fixed.
- Optimize sink flow p_t

$$p_t^{k+1} = \underset{p_t(x)\le C_t(x)\ \forall x\in V}{\arg\max} \; -\frac{c}{2} \left\| p_t - H^k \right\|_2^2 , \quad (58)$$

  where $H^k = {p_s}^{k+1} - \mathrm{div}_w\, p^{k+1} + \frac{\lambda^k}{c}$ is fixed.
- Update λ

$$\lambda^{k+1} \;=\; \lambda^k - c\,(\mathrm{div}_w\, p^{k+1} - p_s^{k+1} + p_t^{k+1}) .$$

To solve (47), construct the augmented Lagrangian function:

$$L_c(p_s, p_t, p, \lambda) = \sum_{x\in V} \left\{ v^b p_s - v^f p_t + \lambda(\mathrm{div}_w\, p - p_s + p_t) \right\} - \frac{c}{2} \left\| \mathrm{div}_w\, p - p_s + p_t \right\|_2^2 . \quad (59)$$

The augmented Lagrangian method for (47) becomes:

Algorithm 1s Supervised Max-flow Algorithm

Initialize p_s^1, p_t^1, p^1 and λ^1. For $k = 1, ...$ until convergence:

- Optimize p flow

$$p^{k+1} = \underset{|p(e)|\le W(e)\ \forall e\in E}{\arg\max} \; -\frac{c}{2} \left\| \mathrm{div}_w\, p - F^k \right\|_2^2 , \quad (60)$$

  where $F^k = {p_s}^k - {p_t}^k + \frac{\lambda^k}{c}$ is fixed.
- Optimize source flow p_s

$$p_s^{k+1} = \underset{p_s(x)\le C_s(x)\ \forall x\in V}{\arg\max} \sum_{x\in V} v^b p_s - \frac{c}{2} \left\| p_s - G^k \right\|_2^2 , \quad (61)$$

  where $G^k = {p_t}^k + \mathrm{div}_w\, p^{k+1} - \frac{\lambda^k}{c}$ is fixed.
- Optimize sink flow p_t

$$p_t^{k+1} = \underset{p_t(x)\le C_t(x)\ \forall x\in V}{\arg\max} \; -\sum_{x\in V} v^f p_t - \frac{c}{2} \left\| p_t - H^k \right\|_2^2 , \quad (62)$$

  where $H^k = {p_s}^{k+1} - \mathrm{div}_w\, p^{k+1} + \frac{\lambda^k}{c}$ is fixed.
- Update λ

$$\lambda^{k+1} \;=\; \lambda^k - c\,(\mathrm{div}_w\, p^{k+1} - p_s^{k+1} + p_t^{k+1}) .$$

Due to the relation between problem (52) and problem (28), the output λ at convergence will be a solution to (28). Similarly, if η is chosen sufficiently high in (28), then solution λ to (42) will also be a solution to (28).

By Theorem 1, one can obtain a partition which solves (27) by the thresholding procedure described in (30).

The subproblems (56) and (60) for updating p can either be solved by inexactly a few iterations of Chambolle's algorithm [14], or in one gradient ascent step as follows:

$$p^{k+1} = \Pi_W \left(p^k + c\nabla_w(\mathrm{div}_w p^k - F^k) \right) . \quad (63)$$

Above, Π_W is a projection operator which is defined as

$$\Pi_W(p(x,y)) = \begin{cases} p(x,y) & \text{if } |p(x,y)| \le 1, \\ \mathrm{sgn}(p(x,y))W(x,y) & \text{if } |p(x,y)| > 1, \end{cases} \quad (64)$$

where sgn is the sign function. There are extended convergence theories the augmented Lagrangian method in case one of the subproblems are solved

inexactly, see e.g. [26, 32]. In our experience, one gradient ascent iteration leads to the fastest overall speed of convergence.

The subproblems (57) and (58) can be solved by

$$p_s(x) = \min(G^k(x) + \frac{1}{c}, C_s(x)); \tag{65}$$

$$p_t(x) = \min(H^k(x), C_t(x)). \tag{66}$$

The subproblems (61) and (62) can be solved by

$$p_s(x) = \min(G^k(x) + \frac{v^b}{c}, C_s(x)); \tag{67}$$

$$p_t(x) = \min(H^k(x) - \frac{v^f}{c}, C_t(x)). \tag{68}$$

Note that the gradient and divergence operators in the algorithm are constructed using the graphical framework, as shown by equations (4) and (6).

3.2 Max-flow algorithm with balancing constraints

This section demonstrates how to incorporate balancing constraints of the form (18), which take into account the size of the classes. Volume constraints have been proposed for image segmentation models in a convex framework in [42, 51]. We propose an efficient algorithm for incorporating the hard volume constraint (18) on graphs by slightly modifying the dual max-flow problem using a new variable $\rho : V \to \mathbb{R}$ as follows:

$$\max_{p_s,p_t,p,\rho} \Big\{ P(p_s, p_t, p) = \sum_{x\in V} (p_s(x) - a\rho) \Big\} \tag{69}$$

subject to

$$|p(x, y)| \leq w(x, y), \quad (x, y) \in E; \tag{70}$$

$$p_s(x) \leq C_s(x), \quad \forall x \in V; \tag{71}$$

$$p_t(x) \leq C_t(x), \quad \forall x \in V; \tag{72}$$

$$\mathrm{div}_w\, p(x) - p_s(x) + p_t(x) + \rho = 0, \quad \forall x \in V; \tag{73}$$

$$\rho \text{ is a constant function.} \tag{74}$$

Introducing a Lagrange multiplier λ for the constraint (73) yields the primal-dual model

$$\min_{\lambda} \max_{p_s,p_t,p} \Big\{ E(p_s, p_t, p, \rho; \lambda) = \sum_{x\in V} (p_s(x) - a\rho) + \sum_{x\in V} \lambda(x)\big(\mathrm{div}_w\, p - p_s + p_t + \rho\big) \Big\} \tag{75}$$

subject to

$$|p(x, y)| \leq w(x, y), \quad (x, y) \in E; \tag{76}$$

$$p_s(x) \leq C_s(x), \quad \forall x \in V; \tag{77}$$

$$p_t(x) \leq C_t(x), \quad \forall x \in V; \tag{78}$$

$$\rho \text{ is a constant function.} \tag{79}$$

Rearranging the terms, we obtain

$$\min_{\lambda} \max_{p_s,p_t,p,\rho} \Big\{ E(p_s, p_t, p, \rho; \lambda) = \sum_{x\in V} \big\{ \big(1 - \lambda\big)p_s + \lambda p_t + \rho\big(\lambda - a\big) + \lambda\, \mathrm{div}_w\, p \big\} \Big\} \tag{80}$$

subject to (76) - (79).

The intuition of having the above model lies in the following. Following the same arguments as in Section 3.1, we observe that if $\lambda \notin \mathbb{B}'$, the energy can be arbitrarily large by choosing p_s or p_t arbitrarily small, contradicting boundedness from above. From the second last term of (80), it follows that if the balancing constraint (18) is not satisfied, the energy can be made arbitrarily large by choosing ρ arbitrarily high or low. Therefore, by maximizing for p_s, p_t, p and ρ, we obtain the closed form expression (28) subject to the constraints (29) and the balancing constraint (18).

In contrast to the case with the model without balancing constraints, it cannot be guaranteed in advance that a global minimizer is obtained by the thresholding procedure described in Theorem 2. If the solution is binary, it must also be a global minimizer of the binary constrained problem, since the convex set $\mathbb{B}'$ contains the binary set $\mathbb{B}$. In the experiments, the solution tends to be binary or very close to binary, indicating that a global minimizer, or close approximation, can be obtained.

We again construct the augmented Lagrangian functional

$$L_c(p_s, p_t, p, \lambda) = \sum_{x\in V} -a\rho + p_s + \lambda(\mathrm{div}_w\, p - p_s + p_t + \rho) - \frac{c}{2} \left\| \mathrm{div}_w\, p - p_s + p_t + \rho \right\|_2^2, \tag{81}$$

which is exactly (59) if ρ is zero.

We have the following primal augmented Lagrangian algorithm for minimizing the above functional, where we alternate between maximizing L_c for the dual variables and updating the Lagrange multiplier λ:

Algorithm 1b Balancing Constraints

Initialize p_s^1, p_t^1, p^1 and λ^1. For $k = 1, ...$ until convergence:

- Optimize p flow

$$p^{k+1} = \underset{\|p(e)\| \leq W(e)\ \forall e \in E}{\arg\max} \; - \frac{c}{2} \left\| \operatorname{div}_w p - F^k \right\|_2^2, \tag{82}$$

where $F^k = {p_s}^k - {p_t}^k + \frac{\lambda^k}{c} - \rho^k$ is fixed.

- Optimize source flow p_s

$$p_s^{k+1} = \underset{p_s(x) \leq C_s(x)\ \forall x \in V}{\arg\max} \sum_{x \in V} p_s - \frac{c}{2} \left\| p_s - G^k \right\|_2^2, \tag{83}$$

where $G^k = {p_t}^k + \operatorname{div}_w p^{k+1} - \frac{\lambda^k}{c} + \rho^k$ is fixed.

- Optimize sink flow p_t

$$p_t^{k+1} := \underset{p_t(x) \leq C_t(x)\ \forall x \in V}{\arg\max} \; - \frac{c}{2} \left\| p_t - H^k \right\|_2^2, \tag{84}$$

where $H^k = {p_s}^{k+1} - \operatorname{div}_w p^{k+1} + \frac{\lambda^k}{c} - \rho^k$ is fixed.

- Optimize ρ

$$\rho^{k+1} = \arg\max_{\rho} \sum_{x \in V} a\rho - \frac{c}{2} \left\| \rho - J^k \right\|_2^2, \tag{85}$$

where $J^k = -p_t^{k+1} - \operatorname{div}_w p^{k+1} + \frac{\lambda^k}{c} + p_s^{k+1}$ is fixed.

- Update λ

$$\lambda^{k+1} = \lambda^k - c\left(\operatorname{div}_w p^{k+1} - p_s^{k+1} + p_t^{k+1} + \rho^{k+1}\right).$$

The optimization problem (82) for p can be solved by one step of the projected gradient method as follows:

$$p^{k+1} = \Pi_W(p + c\nabla_w(\operatorname{div}_w p^k - F^k)), \tag{86}$$

where Π_W is the projection defined in (64).

The subproblems (83) and (84) can be solved by

$$p_s(x) = \min(G^k(x) + \frac{1}{c}, C_s(x)); \tag{87}$$

$$p_t(x) = \min(H^k(x), C_t(x)). \tag{88}$$

We solve (85) using

$$\rho^{k+1} = \operatorname{mean}(-{p_s}^{k+1} + {p_t}^{k+1} + \operatorname{div}_w p^{k+1} + \rho^k - \frac{\lambda^k}{c} - \frac{a}{c}). \tag{89}$$

In step (89), the constraint that ρ should be constant is imposed exactly by computing the average of the pointwise unconstrained maximizers $\rho(x)$ for $x \in V$, and then the average value is assigned to $\rho(x)\ \forall\ x \in V$.

Just like in the previous cases, the final classification is obtained by thresholding λ.

3.3 Extension of primal augmented Lagrangian method to graphs

In this section, we describe another algorithm for solving the convex problem (27), by extending the Split-Bregman algorithm [33] for geometric problems [32] to general graphs. It has recently been shown [53, 60] that the Split-Bregman algorithm is equivalent to solving a specialized decomposition of total variation regularized problems by the augmented Lagrangian method. We use the augmented Lagrangian notation when describing the algorithm, since this notation has already been introduced in Section 3.1 when deriving the max-flow algorithms.

Consider the general minimization problem

$$\min_{\lambda}\ TV_w(\lambda) + \sum_{x \in V} g(\lambda(x), x). \tag{90}$$

The ROF model is a special case of (90) in the case that V is a regular image domain, λ^0 is the noisy input image and $g(\lambda(x), x) = |\lambda(x) - \lambda^0(x)|^2$.

In our case, we wish to choose g according to (26) and impose the constraint $\lambda \in \mathbb{B}'$. The idea is to solve

$$\min_{\lambda, q} \|q\|_1 + \sum_{x \in V} g(\lambda(x), x)$$
$$\text{s.t. } q = \nabla_w \lambda, \tag{91}$$

where $\|s\|_1 = \sum_{x \in V} |s(x)|$, by the augmented Lagrangian method.

We introduce a Lagrange multiplier ϕ for the constraint (91). This results in the augmented Lagrangian functional

$$L_c(\lambda, q, \phi) = \|q\|_1 + \sum_{x \in V} \left\{ g(\lambda(x), x) + \phi \cdot (q - \nabla_w \lambda) \right\} + \frac{c}{2} \|q - \nabla_w \lambda\|_2^2 \tag{92}$$

where c is a constant and $\|s\|_2^2 = \sum_{x \in V} |s(x)|^2$. We want to find a saddle point of (92) over λ, q and ϕ:

$$\max_{\phi} \min_{\lambda, q} L_c(\lambda, q, \phi) \tag{93}$$

by alternating between minimizing for λ and q

$$(\lambda^k, q^k) = \arg\min_{\lambda, q} L_c(\lambda, q, \phi^k) \tag{94}$$

and updating the Lagrange multiplier by one step of gradient ascent:

$$\phi^{k+1} = \phi^k + c(q^{k+1} - \nabla \lambda^{k+1}). \tag{95}$$

The minimization problem (94) can be separated into two subproblems:

$$\min_{\lambda} \sum_{x \in V} \left\{ g(\lambda(x), x) - \phi^k \cdot \nabla_w \lambda \right\} + \frac{c}{2} \|q - \nabla_w \lambda\|_2^2;$$

(96)

$$\min_q \|q\|_1 + \sum_{x\in V} \phi^k \cdot q + \frac{c}{2}\|q - \nabla_w \lambda\|_2^2. \quad (97)$$

Therefore, the algorithm is the following:

Algorithm 2 Augmented Lagrangian Algorithm

Initialize ϕ^1, q^1 and λ^1. For $k = 1, ...$ until convergence:

- Optimize λ

$$\lambda^{k+1} = \min_\lambda \sum_{x\in V} \{g(\lambda(x), x) - \phi^k \cdot \nabla_w \lambda\} + \frac{c}{2}\left\|q^k - \nabla_w \lambda\right\|_2^2. \quad (98)$$

- Optimize q

$$q^{k+1} = \min_q \|q\|_1 + \sum_{x\in V} \phi^k \cdot q + \frac{c}{2}\left\|q - \nabla_w \lambda^{k+1}\right\|_2^2. \quad (99)$$

- Update Lagrange multipliers

$$\phi^{k+1} = \phi^k + c(q^{k+1} - \nabla\lambda^{k+1}). \quad (100)$$

Again, as in the max-flow algorithm, the final binary classification is obtained by thresholding λ to either 0 or 1.

The subproblem (98) gives the Euler-Lagrange equation:

$$\frac{\partial g}{\partial \lambda} + c\,\mathrm{div}_w(q^k - \nabla_w \lambda) + \mathrm{div}_w(\phi^k) = 0, \quad (101)$$

where in this case $\frac{\partial g}{\partial \lambda} = C_t - C_s$.

We solve the above subproblem using one step of forward Euler:

$$\frac{\lambda^{k+1} - \lambda^k}{dt} = -(C_t - C_s + c\,\mathrm{div}_w(q^k - \nabla_w \lambda^k) + \mathrm{div}_w(\phi^k)). \quad (102)$$

This becomes

$$\frac{\lambda^{k+1} - \lambda^k}{dt} = -(C_t - C_s + c\,\mathrm{div}_w(q^k) - c\triangle_w \lambda^k) + \mathrm{div}_w(\phi^k)). \quad (103)$$

All the operators, as was stated before, are formulated in a graph setting.

We solve subproblem (99) in the same way as it is done in [60]:

$$q^{k+1}(x) = \begin{cases} \frac{1}{c}(1 - \frac{1}{|b(x,y)|})b(x,y), & \text{if } |b(x,y)| > 1, \\ 0, & \text{if } |b(x,y)| \le 1, \end{cases}$$

(104)

where $b = |c\nabla\lambda - \phi^k|$.

We have tried solving the subproblem (98) above in another way. A similar scheme is used, except the Laplace operator is calculated implicitly, and we proceed further by considering its terms as a linear combination of the eigenvectors of the random walk Laplacian, in a similar way as in [4] and [48]. Only a small fraction of the eigenvectors are used. This way of solving the subproblem turns out to be several times faster than the one previously discussed, but because it does not perform well in the case of non-random fidelity, we did not use it. A disadvantage of this method is that it only uses a small fraction of the eigenvectors, which might not contain enough information to result in an accurate classification, as was the case with experiments with non-random fidelity. However, it also has some advantages, which are discussed in the next section.

Avoiding trivial global minimizers

If the number of supervised points $V^f \cup V^b$ are very low, the global minimizer of (13) may just be the trivial solution $S = V^f$ or $S = V^b$. This was the case for a small number of our experiments. In order to avoid this problem, the cost of these trivial solutions can be increased by increasing the number of edges incident to the supervised points $V^f \cup V^b$, which amounts to adding nonlocal behavior. Non-supervised points in the graph are connected by edges to their M nearest neighbors. Supervised points can instead be connected to their K nearest neighbors, where $K > M$, thereby increasing the cost of the partitions $S = V^f$ and $S = V^b$.

An interesting observation is that if the subproblem (98) in the primal max-flow algorithm is solved via the approximate eigendecomposition, the algorithm does not result in the trivial solution $S = V^f$ and $S = V^b$, even when the number of supervised points are very low. The reason for this seems to be that the approximation resulting from not using all the eigenfunctions erases the unwanted trivial global minimizers from the energy landscape.

Note that the second eigenvector of the Laplacian already provides a solution to a cut using a spectral clustering approximation approach. Although we experiment with using that approximation as an initialization, the methods work just as well when random initialization is used.

4 Results

The results for several data sets are summarized in Table 2, with those of the best method highlighted. In all experiments, we have constructed the graph using the M nearest neighbors and approximated the eigendecomposition of the Laplacian using only the M largest eigenvalues. By "random fidelity", we mean choosing supervised points randomly. By "corner fidelity", we mean choosing supervised points in a certain portion of the data set only, in this case in the "corner" portion of the eigenvector graph. The technique described in Section 3.3 for avoiding trivial global minimizers, was used on the two moons data set in Fig. 2 in case of less than 3.25 % supervised points. Below, we provide details about each of the data sets that we used. The results were computed on a 2.4 GHz Intel Core i2 Quad.

In the case when we need to compute the eigenvectors and eigenvalues of the random walk graph Laplacian, we first use a fast numerical solver called the Rayleigh-Chebyshev procedure [1] to compute those of the symmetric graph Laplacian. One can then use the previously described relationship between the eigendecomposition of the symmetric graph Laplacian and that of the random walk graph Laplacian. The Rayleigh-Chebyshev procedure itself is a modification of an inverse subspace iteration method using adaptively determined Chebyshev polynomials. It is also a robust method that converges rapidly and that can handle cases when there are eigenvalues of multiplicity greater than one. The calculations are made even more efficient by the fact that only a small portion of the eigenvectors need to be calculated, as the most significant nodes contain enough information to produce accurate results. To have a fair comparison, we use the same number of eigenvectors per data set for all methods.

We have used

$$f(\lambda(x), x) = \eta(x)|\lambda(x) - \lambda_0(x)|^2 \tag{105}$$

for all our computations. Here, λ_0 is the initial value of λ, and $\eta(x)$ is a function that takes on a value of a constant η on fidelity points and zero elsewhere.

Below, we provide more detail on the results for each of the benchmark data sets, as well as a description of the data set itself. In addition, we provide a comparison of the results to those of some of the best methods, including the binary MBO and GL algorithms.

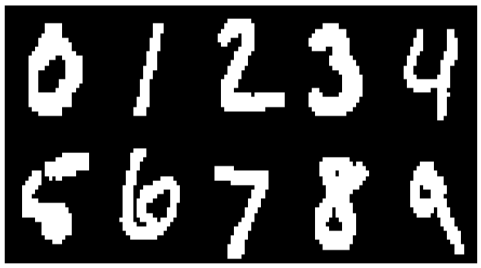


Fig. 1 Examples of digits from the MNIST data base

4.1 MNIST

The MNIST digits data set [45], available at *http://yann.lecun.com/exdb/mnist/*, is a data set of 70000 28×28 images of handwritten digits from $0-9$. However, since our method is only binary, we obtained a subset of this set to classify, in particular, 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. Starting from some initial classification of the points and using only a small fraction of the set as fidelity, the goal is to classify each image into being either a 4 or 9. We used $M = 10$.

Using random initialization and random fidelity, the max-flow method obtained an accuracy of around 98.48% averaged over 100 runs with different fidelity sets of 500 randomly chosen points (or only 3.62% of the set). The primal augmented Lagrangian method's accuracy was around 98.44%. The accuracy of binary MBO graph method from [48] and the binary GL graph method from [4] was slightly lower than that of our methods; the algorithms were able to achieve an average accuracy of around 98.37% and 98.29%, respectively. Table 2 summarizes the above and also shows results in the case when the initialization is constructed using the thresholded second eigenvector of the Laplacian or when the fidelity region is chosen nonrandomly by only considering points that give values in the corners of leftmost image in Figure 4(a). More detail on the latter information will be given in Section 4.6. The parameters for Algorithm 1 were: $c = 0.5$, $\eta = 50$. For Algorithm 2, they were: $c = 0.008$, $\eta = 400$, $dt = 0.032$.

To compare with other methods, we note a recent result by Hu et al. [40], which is an unsupervised method. The authors of the paper also tested only digits 4 and 9 and obtained a purity score (measures the fraction of the nodes that have been assigned to the correct community) of 0.977. The GenLouvain algorithm obtained a purity score of 0.975. In addition, many other algorithms have used the full MNIST data set with all 10 digits.

For example, Cheeger cuts [56], boosted stumps [41, 45], and transductive classification [57] have obtained accuracies of 88.2%, 92.3%-98.74%, and 92.6%, respectively. Also, papers on k-nearest neighbors [44, 45], neural or convolutional nets [20, 44, 45], nonlinear classifiers [44, 45] and SVM [23, 44] report accuracies of 95.0%-97.17%, 95.3%-99.65%, 96.4%-96.7%, and 98.6%-99.32%, respectively. The aforementioned approaches are supervised learning methods using 60,000 out of 70,000 digits (or about 85.71% of the whole data set) as a training set. Morever, we compare our method with [9], which obtains 98.05% accuracy by knowing 10% of the labels, 97.78% by knowing 5% of the labels, and 97.72% by knowing 2.5% of the labels. Our algorithms, taking only 3.6% of the data set as fidelity, obtain around 98.5% accuracy, and thus are competitive with, and in most cases outperform, these methods. Moreover, we have not performed any preprocessing or initial feature extraction on the data set, unlike most of the mentioned algorithms.

4.2 Banknote Authentication Data Set

The banknote authentication data set, from the UCI machine learning repository [2], is a data set of 1372 features extracted from images (400×400 pixels) of genuine and forged banknotes. Wavelet transform was used to extract the features from the images. The goal is to segment the banknotes into being either genuine or forged. We used $M = 15$.

The results are shown in Table 2. With the max-flow method, for a 5.1% fidelity set, we were able to obtain an average accuracy (over 100 different fidelity sets) of around 99.09%, while the primal augmented Lagrangian method achieved a similar accuracy of 98.75%. The results did not deteriorate much for a smaller fidelity set of 3.6%, with the two methods achieving an accuracy of 98.83% and 98.29%, respectively. The parameters for Algorithm 1 were: $c = 0.15$, $\eta = 250$. For Algorithm 2, they were: $c = 0.08$, $\eta = 50$, $dt = 0.5$.

We compare this result to the binary MBO algorithm, which achieved a lower accuracy of 95.43% and 93.48% for 5.1% and 3.6% fidelity sets, respectively. For the binary GL method, the results were also not as good- 97.76% and 96.10%, for 5.1% and 3.6% fidelity sets, respectively.

4.3 Two Moons

This data set is constructed from two half circles in $\mathbb{R}^2$ with a radius of one. The centers of the two half circles are 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. Therefore, the set consists of two thousand points. Starting from some initial classification of the points, the goal is to segment the two half circles. We used $M = 10$.

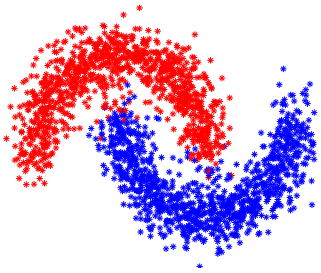


Fig. 2 Two moons example with max-flow method

For the max-flow method, in the case of 65 or lower number of fidelity points (3.25 %), we increased the number of edges of supervised points to others to avoid the trivial global minimizer where all points but the supervised ones are classified as one class.

Using random initialization and random fidelity, for the max-flow method, we obtained an average accuracy (over 100 different fidelity sets) of 97.10% and 97.05% in the case of 100 and 50 fidelity points, respectively. An example of a solution is shown in Figure 2, with the two classes colored in red and blue. The primal augmented Lagrangian method achieved an accuracy of around 97.07% for 100 fidelity points and around 96.78% for 50 fidelity points. The parameters for Algorithm 1 were: $c = 0.5$, $\eta = 50$. For Algorithm 2, they were: $c = 0.32$, $\eta = 100$, $dt = 0.008$.

To compare this with binary MBO, the method obtained 98.41% and 97.53% accuracy for 100 and 50 fidelity points, respectively, which is very similar to the results of the binary GL graph method.

4.4 Rods

We have also tested this algorithm on two other synthetic data sets created using the rods pictured in Figure 3. Around two thousand uniformly chosen points were sampled from each image, and then embedded in $\mathbb{R}^{100}$. Finally, noise was added to each of the points, much like the case with the two moons data set. We used $M = 25$.

In the case of random fidelity region, we obtained accuracy in the 98^{th} or 99^{th} percentile, no matter what initialization. In the case of fidelity region in the corner, we obtained interesting global

minimizers for the two data sets. The comparison of our results with the binary MBO and GL method is detailed in the next section. The parameters for Algorithm 1 were: $c = 0.01$, $\eta = 50$. For Algorithm 2, they were: $c = 0.016$, $\eta = 500$, $dt = 0.512$.

4.5 Comparison of the balancing constraints max-flow method to the regular max-flow method

We tested our balancing constraints max-flow method on several data sets using random initialization and fidelity, and compared it to the regular max-flow meethod. It handles the case of a small fidelity region better than the original max-flow method (Algorithm 1) and gives higher accuracy everywhere. We used the same M as described in the previous section. The results are displayed in Table 1, and those of the best method (compared to Algorithm 1) are highlighted. In general, the solution is very close to binary, with some small differences that may be explained by the finite stopping criterion of the algorithm, indicating that close approximations to global minimizers are obtained. To measure how close the solution λ is to binary, we have computed the norm $\sum_{x\in V} \frac{|\lambda(x)-\lambda^\ell(x)|}{|V|}$, where λ^ℓ is defined in (30) and $\ell = 0.5$. The norm should be 0 if λ is binary. In the experiments, the values of the norm range from 0.0005 to $6 * 10^{-18}$.

For the two moons example, starting from 20 fidelity points, we obtained very reasonable results. For 50, 40, 30 and 20 fidelity points, we obtained 97.19%, 97.12%, 97.11% and 96.11% average accuracy (over 100 different fidelity sets), respectively. While the results of the binary MBO method (without any zero means constraint) for the two moons data set achieves slightly better accuracy for 50 and 100 fidelity points (being of 97.53% and 98.41%, respectively), we noticed that if the number of fidelity points is too low, the method is unable to perform well, as the results vary not insignificantly depending on the fidelity set. The same is the case with the max-flow method with no balancing constraint. For example, for 20 fidelity points, the average accuracy accuracy we obtained for the method was 88.22%. However, with the balanced method, we still obtain a good result (96.11%) for a fidelity set containing as little as 20 points. Thus, the advantage of the method is that it performs well with even a very small fidelity region. The results are summarized in Table 1.

For the MNIST data set, we obtained very good results even for a small number of fidelity points. For 500, 400, 300 and 210 fidelity points (or 3.6%, 2.9%, 2.2% and 1.5% of the data, respectively), we obtained an average accuracy (over 100 different fidelity sets) of 98.59%, 98.48%, 98.45% and 98.41%, respectively. A comparison to the results of the regular max-flow method is in Table 1. Note that in addition to giving at least a slightly higher accuracy everywhere, it handles the case of a small number of fidelity points better than the original method. For example, for 210 fidelity points, the method obtained an accuracy of 98.41%, while the regular max-flow method achieved a much lower accuracy of 93.68%.

For the banknote authentication data set from the UCI Machine Learning Repository [2], we obtained reasonable results for as little as 20 fidelity points. The results are shown in Table 1. The balancing constraints method obtains better results than the original max-flow method, achieving an average accuracy (over 100 different fidelity sets) of 98.55% for only 20 fidelity points, as opposed to the accuracy of 96.78% of the original max-flow method.

For the first rod data set, we obtained reasonable results starting from around 10 fidelity points out of around two thousand that are in the rods data set. For 10 to 20 fidelity points, the accuracy was around around 96%. Testing 50 and 100 fidelity points, we obtained around 99% accuracy.

4.6 Comparison of our convex algorithms to binary MBO and GL methods

After comparing the results of our convex algorithms to the binary MBO [48] and binary GL [4] graph methods, we have reached the following conclusions based on our work:

- As long as the fidelity points are well represented for each class (meaning the fidelity points represent a whole variety of points in the class), the binary MBO method and the binary GL method have no trouble finding the correct minimizer or something very close. The initialization might not matter; even with a bad initialization, the local minimizer will still be found. Our convex methods find the local minimizer easily.
- Problems occur when the fidelity is not chosen randomly. In this case, even if the initialization is random, the convergence might not occur for the binary MBO and GL methods. In all our experiments with the rods data sets and MNIST, the local minimizer was not found by the two methods. However, our convex algorithms still found the correct local minimizer.

Table 1 Comparison of balancing constraints max-flow method and regular max-flow method

Number of fidelity points	50	40	30	20
Two moons- Max-flow method (regular)	97.05%	96.92%	96.86%	88.22%
Two moons- Max-flow method (balancing constraints)	**97.19%**	**97.12%**	**97.11%**	**96.11%**
Number of fidelity points	500	400	300	210
MNIST- Max-flow method (regular)	98.44%	98.40%	98.36%	93.68%
MNIST- Max-flow method (balancing constraints)	**98.59%**	**98.48%**	**98.45%**	**98.41%**
Number of fidelity points	50	40	30	20
Banknote Authentication Data Set (regular)	**98.83%**	98.72%	98.21%	96.78%
Banknote Authentication Data Set (balancing constraints)	**98.83%**	**98.91%**	**98.72%**	**98.55%**

Table 2 Comparison of methods

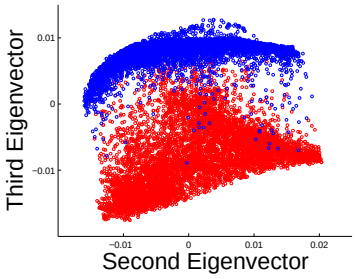
	max-flow	primal augmented Lagrangian	binary MBO	binary GL
MNIST (3.6% fidelity) random initialization, random fidelity	**98.48%**	98.44%	98.37%	98.29%
MNIST (3.6% fidelity) 2nd eigenvector initialization, random fidelity	**98.48%**	98.43%	98.36%	98.25%
MNIST (3.6% fidelity) random initialization, corner fidelity	**98.47%**	98.40%	62.35%	64.39%
MNIST (3.6% fidelity) 2nd eigenvector initialization, corner fidelity	**98.46%**	98.40%	63.87%	63.19%
Banknote Authentication Data Set (5.1% fidelity)	**99.09%**	98.75%	95.43%	97.76%
Banknote Authentication Data Set (3.6% fidelity)	**98.83%**	98.29%	93.48%	96.10%
two moons (5% fidelity)	97.10%	97.07%	**98.41%**	98.31%
two moons (2.5% fidelity)	97.05%	96.78%	97.53%	**98.15%**

Fig. 3 Rod 1 and Rod 2

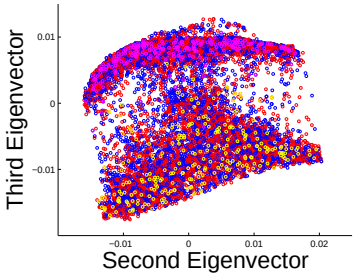
These conclusions are supported by the work done on the MNIST digits data set, using digits 4 and 9 only. The second vs. third eigenvector of the symmetric graph Laplacian are graphed in Figure 4(a), with one digit represented by blue and another by red. The results of experiments on this data set are found in Figure 4. Each row represents a different experiment: first two rows contain experiments with random initialization, while last two rows contain experiments with fidelity in a constrained area. The initialization is random for experiments in first and third row, and is constructed by thresholding the second eigenvector of the Laplacian for the results in the second and fourth row. The first column represents the initialization, while the second and third columns are results for the max-flow and binary MBO algorithm, respectively. The fidelity points are marked by yellow and magenta for the two classes.

We see that if the fidelity region is well represented in the data set, no matter what initialization, none of the algorithms have a problem finding a close to perfect solution (accuracy is between 98 and 99 percent- see Table 2). However, when the fidelity region is not random (in this case constrained to only the nodes whose corresponding entry in the second or third eigenvector of the Laplacian match a certain range), we see that the binary MBO and GL algorithms fail to obtain an accurate solution; its accuracy is below 70 percent. However, the max-flow algorithm and the primal augmented Lagrangian methods achieve an almost perfect solution of 98.47% and 98.40% accuracy, respectively.

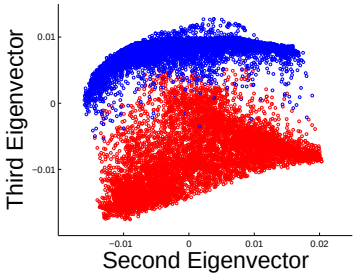
The two conclusions are also supported by the work done on the two rod data sets created from images displayed in Figure 3. Experiments with the first and second rod image are shown in Figures 5 and 6, respectively. The first two rows represent cases with a random fidelity set, while the last two rows are experiments with corner fidelity. Cases with random initialization are in first and third rows, and second eigenvector initialization is used in experiments in the second and fourth



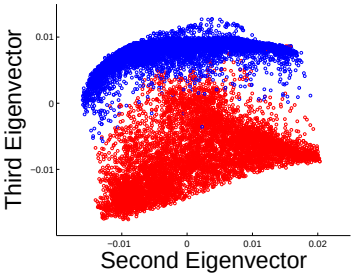
(a) Ground Truth



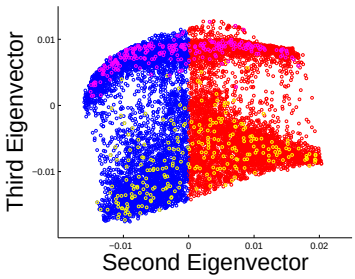
(b) Random initialization and fidelity



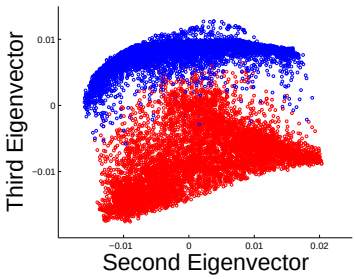
(c) max-flow result



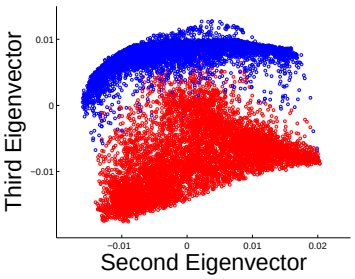
(d) binary MBO result



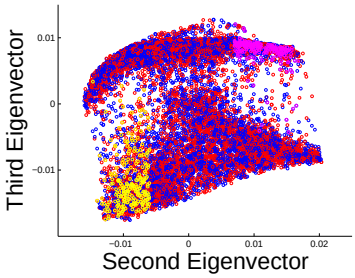
(e) 2nd eigenvector initialization, random fidelity



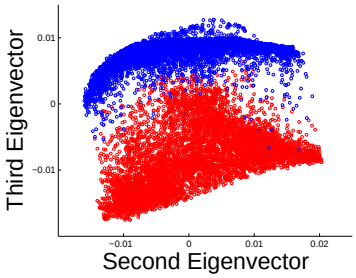
(f) max-flow result



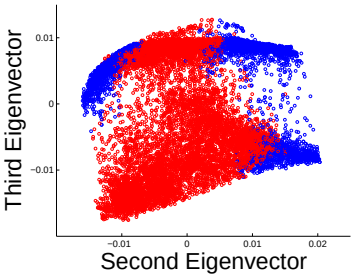
(g) binary MBO result



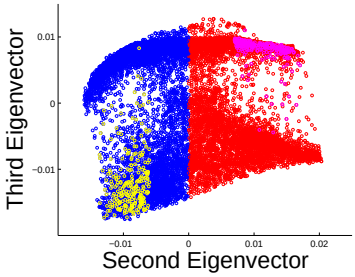
(h) Random initial., corner fidelity



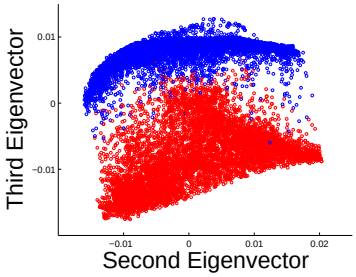
(i) max-flow result



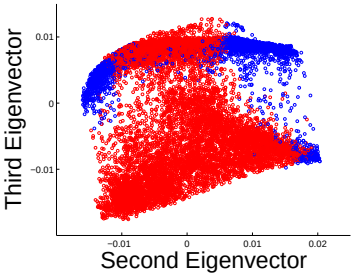
(j) binary MBO result



(k) 2nd eigenvector initialization, corner fidelity



(l) max-flow result



(m) binary MBO result

Fig. 4 MNIST. Left: initialization, supervised points are marked in yellow and magenta. Middle: max-flow algorithm result. Right: binary MBO result

row. The first column is the initialization, second column is the max-flow algorithm result, and third column is the binary MBO result.

For rod 1, we found that when the fidelity is chosen randomly, the minimizer divides the bottom two rods from the rest of the image. When the fidelity is chosen at the corners, the minimizer is shown in the bottom two rows of the second column. The convex max-flow and primal augmented Lagrangian algorithms are able to attain these minimizers, while the binary MBO and GL algorithms struggle in the case of non-random fidelity. The situation is similar with rod 2.

Note about MBO algorithm

As noted in [48], the first step of the MBO algorithm (heat equation with an extra term) was solved using the eigenvalue and eigenvector decomposition of the Laplacian. However, a disadvantage of solving equations using this method is that, for it to be efficient, only a fraction of eigenvectors are used, and they might not contain enough information to result in an accurate classification. Naturally, the more eigenvectors one computes, the longer the process will take.

As an alternative way of solving the first step (20) of the MBO algorithm, we have tried using just the simple forward heat equation solver. However, this did not result in an accurate segmentation in the case of non-random fidelity, thus not improving the results from the original way of solving it. This shows that the algorithm is getting stuck in a local minimum, since the problem is clearly not the lack of information encoded within the small number of eigenvectors used.

4.7 Comparison of convergence, speed, and energy

The stopping criterion used for all algorithms was taken to be the point at which the square of the relative L2 norm between the current and previous iterate is negligible, or below a certain constant α. With the exception of the MNIST data set (where $\alpha = 5*1e-10$), the max-flow, binary MBO and GL algorithms stabilize around $\alpha = 1e-17$ or $1e-16$. The primal augmented Lagrangian method stabilizes around $\alpha = 1e-08$ or $1e-09$.

Table 3 includes information about the number of iterations needed to reach stability, and also the timing results for each data set.

We have also computed the initial and final energy for each data set. The energy was calculated using

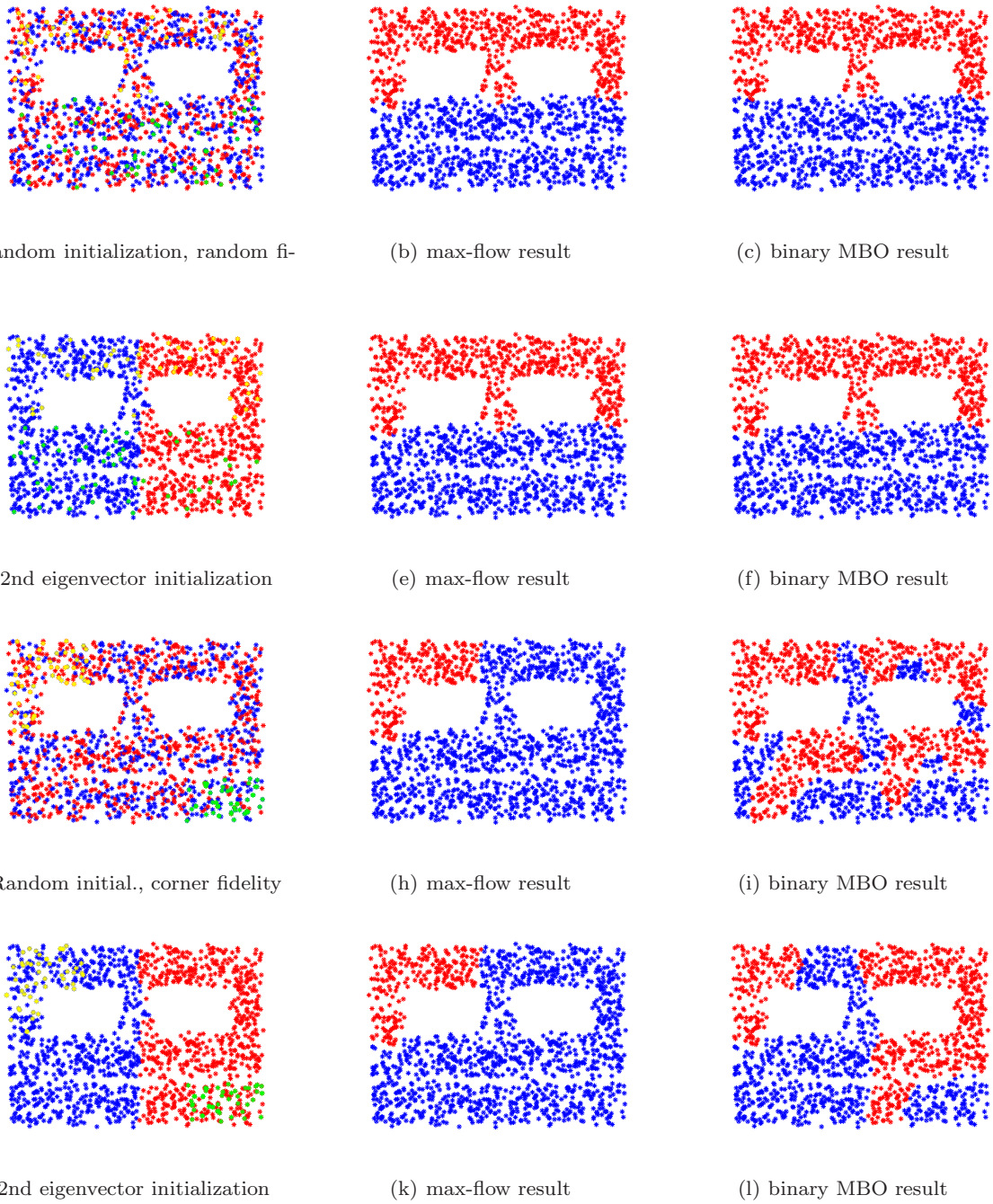
$$E(\lambda) = \frac{1}{2} \sum_{x,y \in V} w(x,y)|\lambda(x) - \lambda(y)|, \tag{106}$$

where $\lambda(x)$ is 0 if node x was classified to be in the first class, and 1 if it was classified to be in the second class. Note that the energy here is exactly $TV_w(\lambda)$. Table 4 includes information about the initial and final energy for each method. We see that the max-flow algorithm is able to obtain the lowest energy in each case. In general, one can see that the convex algorithms are able to obtain the global minimizer in all cases, while the binary MBO and GL algorithms struggle in the case of non-random fidelity.

It can be observed that the max-flow method obtains marginally lower energy than the primal augmented Lagrangian method. The reason for this is that the max-flow method stabilizes around a lower precision in terms of relative L2 difference between successive iterations, as explained above. We believe this difference is caused by the pointwise projection step of λ onto the set $[0, 1]$ each iteration in the primal augmented Lagrangian, which is avoided in the max-flow algorithm.

5 Conclusion

We have described two convex methods for data segmentation using a graphical framework. The first solves a dual maximum flow problem by continuous optimization techniques, and the second method solves the primal problem directly. It was proved the algorithms were guaranteed to produce global minimizers for semi-supervised data segmentation problems with two classes. In case where the class sizes are known precisely or approximately, the first model could be slightly modified to produce more stable and accurate results by incorporating constraints on the class sizes. Simulations showed that the methods were comparable with or outperformed the state-of-the-art algorithms. In fact, our convex models had the advantage over non-convex methods in that the latter could occasionally get stuck in local minima. Moreover, a thorough comparison to a non-convex binary MBO and GL method [48] revealed that the latter may not produce an accurate result in case when the fidelity region is not chosen randomly, but that did not affect the proposed convex methods. To speed up the timing of the algorithms, we made use of a fast numerical solver described in [1] to solve some of the subproblems involving graph-based PDEs. Future work includes an extension to multiple classes and experimentation with other ways to incorporate knowledge about class sizes.



(a) Random initialization, random fidelity (b) max-flow result (c) binary MBO result

(d) 2nd eigenvector initialization (e) max-flow result (f) binary MBO result

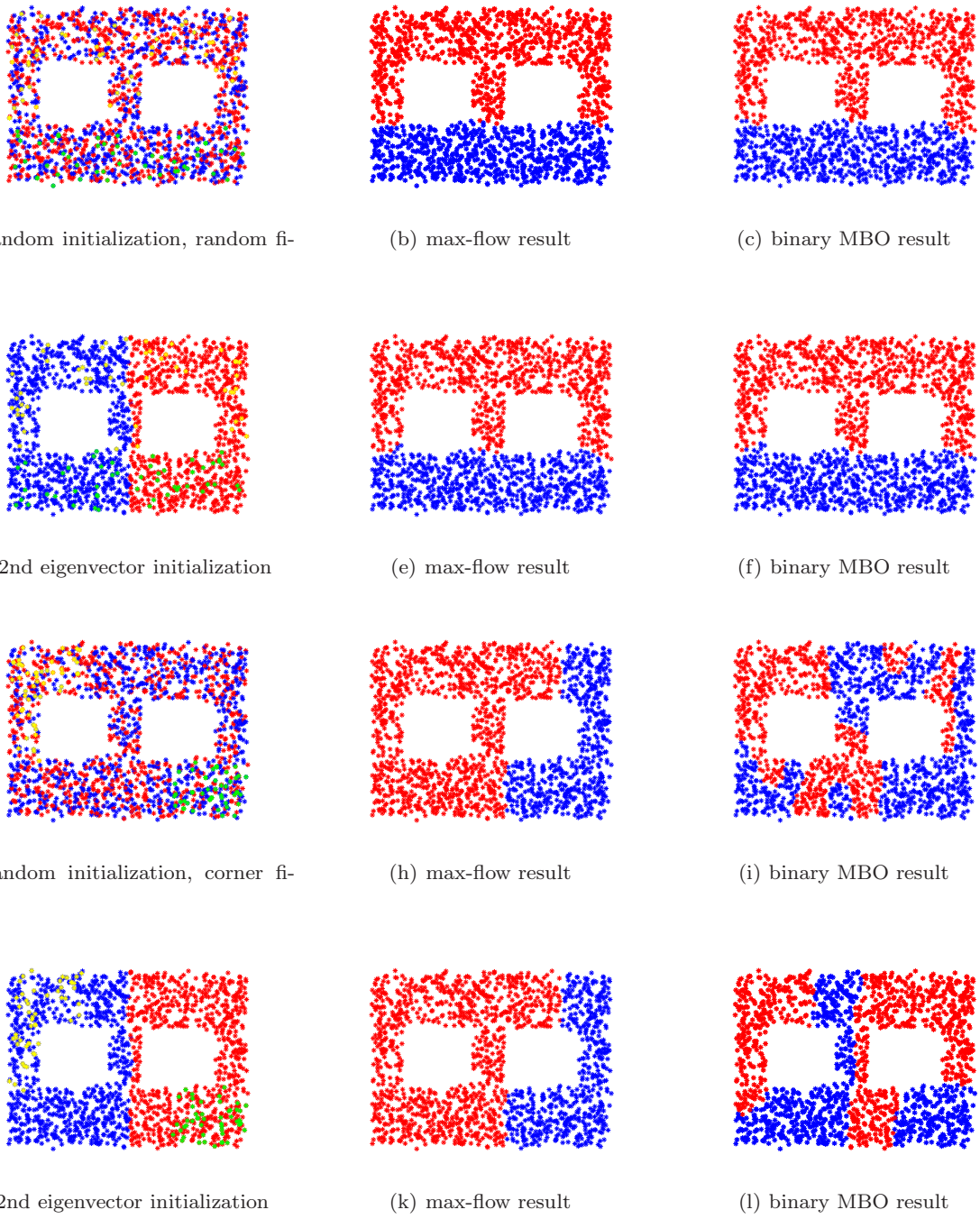
(g) Random initial., corner fidelity (h) max-flow result (i) binary MBO result

(j) 2nd eigenvector initialization (k) max-flow result (l) binary MBO result

Fig. 5 Results for Rod 1. Left: initialization, supervised points are marked in yellow and green. Middle: max-flow algorithm result. Right: binary MBO result

References

1. Anderson, C.: A Rayleigh-Chebyshev procedure for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices. J. Comput. Phys. **229**, 7477–7487 (2010)
2. Bache, K., Lichman, M.: UCI machine learning repository (2013). URL http://archive.ics.uci.edu/ml
3. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. **7**, 2399–2434 (2006)
4. Bertozzi, A., Flenner, A.: Diffuse interface models on graphs for classification of high dimensional data. Multiscale Model. Simul. **10**(3), 1090–1118 (2012)
5. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**, 359–374 (2001)
6. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23**(11), 1222 – 1239 (2001)



(a) Random initialization, random fidelity (b) max-flow result (c) binary MBO result

(d) 2nd eigenvector initialization (e) max-flow result (f) binary MBO result

(g) Random initialization, corner fidelity (h) max-flow result (i) binary MBO result

(j) 2nd eigenvector initialization (k) max-flow result (l) binary MBO result

Fig. 6 Results for Rod 2. Left: initialization, supervised points are marked in yellow and green. Middle: max-flow algorithm result. Right: binary MBO result

7. Bresson, X., Chan, T.F.: Non-local unsupervised variational image segmentation models. Tech. rep., UCLA, cam-report 08-67 (2008)
8. Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J., Osher, S.: Fast global minimization of the active contour/snake model. J. Math Imaging Vis. **28**(2), 151–167 (2007)
9. Bresson, X., Laurent, T., Uminsky, D., von Brecht, J.: Multiclass total variation clustering. In: Advances in Neural Information Processing Systems, pp. 1421–1429 (2013)
10. Bresson, X., Laurent, T., Uminsky, D., von Brecht, J.H.: Convergence and energy landscape for Cheeger cut clustering. Adv. Neural Inf. Process. Syst. **25**, 1394–1402 (2012)
11. Bresson, X., Laurent, T., Uminsky, D., von Brecht, J.H.: An adaptive total variation algorithm for computing the balanced cut of a graph. arXiv preprint arXiv:1302.2717 (2013)
12. Bresson, X., Tai, X.C., Chan, T.F., Szlam, A.: Multi-class transductive learning based on ; 1 relaxations of cheeger cut and mumford-shah-potts model. Journal of Mathematical Imaging and Vi-

Table 3 Number of Iterations and Timing

Number of iterations	max-flow	primal augmented Lagrangian	binary MBO	binary GL
MNIST	426	2709	10	52
Banknote Authentication Data Set	314	725	7	449
two moons	1031	451	8	108
Timing (s)	**max-flow**	**primal augmented Lagrangian**	**binary MBO**	**binary GL**
MNIST [a]	2.88	43.21	0.52	0.78
Banknote Authentication Data Set	1.21	3.76	0.90	0.95
two moons	4.13	5.23	2.30	2.98

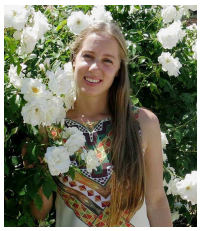
[a] This is the timing of the method using already computed weights and eigenvalues/eigenvectors of the random walk Laplacian.

Table 4 Comparison of Final Energy

Data Set	initial energy	max-flow final energy	primal augmented Lagrangian final energy	binary MBO final energy	binary GL final energy
MNIST (random fid)	23223	789	789	798	804
MNIST (non-random fid)	23223	791	792	2167	5363
Banknote Authentication	3308	30	37	51	42
two moons	3802	533	535	538	548
rod 1 (random fid)	4159	146	148	163	159
rod 1 (non-random fid)	4159	88	89	825	391
rod 2 (random fid)	4528	171	176	186	184
rod 2 (non-random fid)	4528	101	105	709	421

sion **49**(1), 191–201 (2014)
13. Buades, A., Coll, B., Morel, J.M.: A review of image denoising algorithms, with a new one. Multiscale Model. Simul. **4**(2), 490–530 (2005)
14. Chambolle, A.: An algorithm for total variation minimization and applications. Journal of Mathematical Imaging and Vision **20**(1), 89–97 (2004)
15. Chan, T., Vese, L.: Active contours without edges. IEEE Image Proc., 10, pp. 266-277 (2001)
16. Chan, T.F., Esedoḡlu, S., Nikolova, M.: Algorithms for finding global minimizers of image segmentation and denoising models. SIAM J. Appl. Math. **66**(5), 1632–1648 (2006)
17. Chapelle, O., Schölkopf, B., Zien, A.: Semi-Supervised Learning, vol. 2. MIT Press (2006)
18. Cheeger, J.: A lower bound for the smallest eigenvalue of the Laplacian. Princeton University Press (1970)
19. Chung, F.: Spectral Graph Theory, vol. 92. American Mathematical Society (1997)
20. Cireşan, D., Meier, U., Masci, J., Gambardella, L., Schmidhuber, J.: Flexible, high performance convolutional neural networks for image classification. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 1237–1242 (2011)
21. Couprie, C., Grady, L., Najman, L., Talbot, H.: Power watershed: A unifying graph-based optimization framework. IEEE Trans. Pattern Anal. Mach. Intell. **33**(7), 1384–1399 (2011)
22. Couprie, C., Grady, L., Talbot, H., Najman, L.: Combinatorial continuous maximum flow. SIAM J. Imaging Sci. **4**(3), 905–930 (2011)
23. Decoste, D., Schölkopf, B.: Training invariant support vector machines. Mach. Learn. **46**(1), 161–190 (2002)
24. Ekeland, I., Téman, R.: Convex analysis and variational problems. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1999)
25. Elmoataz, A., Lezoray, O., Bougleux, S.: Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing. IEEE Trans. Image Process. **17**(7), 1047–1060 (2008)
26. Esser, J.E.: Primal dual algorithms for convex models and applications to image restoration, registration and nonlocal inpainting (Ph.D. thesis, UCLA CAM-report 10-31) (2010)
27. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton University Press (1962)
28. van Gennip, Y., Bertozzi, A.: Gamma-convergence of graph Ginzburg-Landau functionals. Advanced in Differential Equations **17**(11–12), 1115–1180 (2012)
29. van Gennip, Y., Guillen, N., Osting, B., Bertozzi, A.L.: Mean curvature, threshold dynamics, and phase field theory on finite graphs. Milan J. Math (2014)
30. Gilboa, G., Osher, S.: Nonlocal operators with applications to image processing. Multiscale Modeling & Simulation **7**(3), 1005–1028 (2008)
31. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum-flow problem. J. ACM **35**(4), 921–940 (1988)
32. Goldstein, T., Bresson, X., Osher, S.: Geometric applications of the split bregman method: Segmenta-

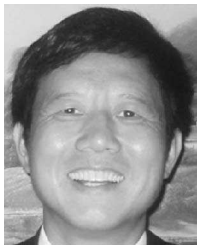
tion and surface reconstruction. J. Sci. Comput. **45**(1), 271–293 (2010)
33. Goldstein, T., Osher, S.: The split bregman method for l1-regularized problems. SIAM J. Imaging Sci. **2**(2), 323–343 (2009)
34. Grady, L.: Multilabel random walker image segmentation using prior models. In: Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 763–770 (2005)
35. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11), 1768–1783 (2006)
36. Grady, L., Polimeni, J.R.: Discrete Calculus: Applied Analysis on Graphs for Computational Science. Springer (2010)
37. Grady, L., Schiwietz, T., Aharon, S., Westermann, R.: Random walks for interactive alpha-matting. In: Proceedings of VIIP, pp. 423–429 (2005)
38. Hein, M., Audibert, J., Von Luxburg, U.: From graphs to manifolds - weak and strong pointwise consistency of graph laplacians. In: Proceedings of the 18th Conference on Learning Theory (COLT, pp. 470–485. Springer (2005)
39. Hein, M., Bühler, T.: An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA. Adv. Neural Inf. Process. Syst. **23**, 847–855 (2010)
40. Hu, H., Laurent, T., Porter, M.A., Bertozzi, A.L.: A method based on total variation for network modularity optimization using the MBO scheme. SIAM J. Appl. Math. **73**(6), 2224–2246 (2013)
41. Kégl, B., Busa-Fekete, R.: Boosting products of base classifiers. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 497–504 (2009)
42. Klodt, M., Cremers, D.: A convex framework for image segmentation with moment constraints. In: IEEE International Conference on Computer Vision (ICCV), pp. 2236 – 2243 (2011)
43. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **26**, 65–81 (2004)
44. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
45. LeCun, Y., Cortes, C.: The MNIST database of handwritten digits. URL http://yann.lecun.com/exdb/mnist/
46. Levin, A., Rav-Acha, A., Lischinski, D.: Spectral matting. IEEE Trans. Pattern Anal. Mach. Intell. **30**(10), 1699 –1712 (2008)
47. Lezoray, O., Elmoataz, A., Ta, V.T.: Nonlocal pdes on graphs for active contours models with applications to image segmentation and data clustering. In: Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on, pp. 873–876. IEEE (2012)
48. Merkurjev, E., Kostic, T., Bertozzi, A.: An MBO scheme on graphs for classification and image processing. SIAM J. Imaging Sci. **6**(4), 1903–1930 (2013)
49. Merriman, B., Bence, J.K., Osher, S.: Diffusion generated motion by mean curvature. AMS Selected Lectures in Mathematics Series: Computational Crystal Growers Workshop **8966**, 73–83 (1992)
50. Mohar, B.: The Laplacian spectrum of graphs. Graph Theory, Combin. Applicat. **2**, 871–898 (1991)
51. Mollenhoff, T., Nieuwenhuis, C., Toppe, E., Cremers, D.: Efficient convex optimization for minimal partition problems with volume constraints. In: Energy Minimization Methods in Computer Vision and Pattern Recognition, pp. 94–107 (2013)
52. Perona, P., Zelnik-Manor, L.: Self-tuning spectral clustering. Adv. Neural Inf. Process. Syst. **17**, 1601–1608 (2004)
53. Setzer, S.: Operator splittings, bregman methods and frame shrinkage in image processing. International Journal of Computer Vision **92**(3), 265–280 (2011)
54. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
55. Szlam, A., Bresson, X.: Total variation and cheeger cuts. In: J. Fürnkranz, T. Joachims (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 1039–1046. Omnipress, Haifa, Israel (2010). URL http://www.icml2010.org/papers/233.pdf
56. Szlam, A., Bresson, X.: A total variation-based graph clustering algorithm for Cheeger ratio cuts. In: Proceedings of the 27th International Conference on Machine Learning, pp. 1039–1046 (2010)
57. Szlam, A.D., Maggioni, M., Coifman, R.R.: Regularization on graphs with function-adapted diffusion processes. J. Mach. Learn. Res. **9**, 1711–1739 (2008)
58. Tai, X.C., Christiansen, O., Lin, P., Skjælaaen, I.: Image segmentation using some piecewise constant level set methods with mbo type of projection. International Journal of Computer Vision **73**(1), 61–76 (2007)
59. Wang, J., Jebara, T., Chang, S.: Graph transduction via alternating minimization. In: Proceedings of the 25th International Conference on Machine Learning, pp. 1144–1151 (2008)
60. Wu, C., Tai, X.C.: Augmented lagrangian method, dual methods, and split bregman iteration for rof, vectorial tv, and high order models. SIAM J. Imaging Sci. **3**(3), 300–339 (2010)
61. Yuan, J., Bae, E., Tai, X.C.: A study on continuous max-flow and min-cut approaches. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2217–2224 (2010)
62. Yuan, J., Bae, E., Tai, X.C., Boykov, Y.: A spatially continuous max-flow and min-cut framework for binary labeling problems. Numer. Math. **126**(3), 559–587 (2013)
63. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. Advances in Neural Information Processing Systems **16**, 321–328 (2004)
64. Zhou, D., Schölkopf, B.: A regularization framework for learning from graph data. In: Workshop on Statistical Relational Learning. International Conference on Machine Learning (2004)
65. Zhu, X.: Semi-supervised learning literature survey. Computer Sciences Technical Report 1530, University of Wisconsin-Madison (2005)



Ekaterina Merkurjev is a fifth 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 classification.

Egil Bae received the BS degree in mathematics in 2005 and the MS and PhD degrees in applied mathematics in 2007 and 2011 from the University of Bergen in Norway. He is currently a postdoctoral researcher in the mathematics Department at the University of California, Los Angeles. His research interests include optimization, variational, and PDE-based methods in image processing, computer vision, computer graphics and machine learning.

Andrea L. Bertozzi received her A. B., M. A. and Ph.D. degrees in Mathematics from Princeton University, Princeton NJ, in 1987, 1988, and 1991 respectively. After holding faculty positions at University of Chicago (1991-1995) and Duke University (1995-2003) she was appointed Professor of Mathematics at UCLA in 2003 and the Betsy Wood Knapp Chair for Innovation and Creativity at UCLA in 2012. Her research interests involve nonlinear PDE and pattern analysis applied to a range of problems in fluid dynamics, swarming, crime modeling, image processing, and now big data analysis. She has graduated 24 PhD students and mentored numerous postdocs. Her published work includes a graduate text and more than 160 scientific papers. She has served on the editorial boards of over ten journals and has served on the scientific advisory boards of three mathematics institutes.



Xue-Cheng Tai received the licenciate degree and the PhD degree in applied mathematics from Jyvaskalya University, Jyvaskalya, Finland in 1989 and 1991, respectively. After holding several research positions in Europe, he became an associate professor in 1994 at the University of Bergen, Bergen, Norway, and a professor since 1997. He was also with Nanyang Technological University, Singapore between 2007-2011. His research interests include numerical PDE for image processing, multigrid, and domain decomposition methods, iterative methods for linear and nonlinear PDE problems, and parallel computing. He has guided numerous masters and PhD students, and published more than 80 scientific papers. He has been a reviewer and an editor for a number of international journals.