Efficient Generalized Fused Lasso and Its Applications

BO XIN, Peking University
YOSHINOBU KAWAHARA, Osaka University
YIZHOU WANG, Peking University
LINGJING HU, Capital Medical University
WEN GAO, Peking University

Generalized fused lasso (GFL) penalizes variables with l_1 norms based both on the variables and their pairwise differences. GFL is useful when applied to data where prior information is expressed using a graph over the variables. However, the existing GFL algorithms incur high computational costs and do not scale to high-dimensional problems. In this study, we propose a fast and scalable algorithm for GFL. Based on the fact that fusion penalty is the Lovász extension of a cut function, we show that the key building block of the optimization is equivalent to recursively solving graph-cut problems. Thus, we use a parametric flow algorithm to solve GFL in an efficient manner. Runtime comparisons demonstrate a significant speedup compared to existing GFL algorithms. Moreover, the proposed optimization framework is very general; by designing different cut functions, we also discuss the extension of GFL to directed graphs. Exploiting the scalability of the proposed algorithm, we demonstrate the applications of our algorithm to the diagnosis of Alzheimer's disease (AD) and video background subtraction (BS). In the AD problem, we formulated the diagnosis of AD as a GFL regularized classification. Our experimental evaluations demonstrated that the diagnosis performance was promising. We observed that the selected critical voxels were well structured, i.e., connected, consistent according to cross validation, and in agreement with prior pathological knowledge. In the BS problem, GFL naturally models arbitrary foregrounds without predefined grouping of the pixels. Even by applying simple background models, e.g., a sparse linear combination of former frames, we achieved state-of-the-art performance on several public datasets.

Categories and Subject Descriptors: C.1.6 [**Optimization**]: Convex Programming

General Terms: Algorithms, Design, Performance

Additional Key Words and Phrases: Generalized fused lasso, parametric cut, Alzheimer's disease, background subtraction

ACM Reference Format:
Bo Xin, Yoshinobu Kawahara, Yizhou Wang, Lingjing Hu, and Wen Gao. 2016. Efficient generalized fused lasso and its applications. ACM Trans. Intell. Syst. Technol. 7, 4, Article 60 (May 2016), 22 pages.
DOI: http://dx.doi.org/10.1145/2847421

The authors were supported by the following grants: 2015CB351800, NSFC-61272027, NSFC-61231010, NSFC-61527804, NSFC-61421062, NSFC-61210005, the Okawa Foundation Research Grant, the Microsoft Research Asia Collaborative Research funding, JSPS KAKENHI 26280086 and 26120524, and Scientific Research Common Program of Beijing Municipal Commission of Education KM201610025013.
Authors' addresses: B. Xin, Y. Wang, and W. Gao, Nat'l Engineering Laboratory for Video Technology, Cooperative Medianet Innovation Center, Key Laboratory of Machine Perception (MoE), Sch'l of EECS, Peking University, Beijing, 100871, China; emails: {boxin, Yizhou.Wang, wgao}@pku.edu.cn; Y. Kawahara, Institute of Scientific and Industrial Research, Osaka University, Osaka 567-0047 Japan; email: ykawahara@sanken.osaka-u.ac.jp; L. Hu, Bioinformatics Office, Department of Foundational Education, Yan Jing Medical College, Capital Medical University, Beijing, 101300, China; email: hulingjing@gmail.com.

DOI: http://dx.doi.org/10.1145/2847421

1. INTRODUCTION

Sparse models, such as *lasso* [Tibshirani 1996], *basis pursuit* [Chen et al. 1998], and *compressive sensing* [Candès et al. 2006] have gained a great reputation in fields such as machine learning and signal processing. Many applications in artificial intelligence also benefit from similar strategies of pursuing sparsity, which are usually formulated as an optimization problem with constraints or regularization using l_1 or l_0 norm [Wright et al. 2009; Aharon et al. 2006; Candès et al. 2011]. Automatic selection of relevant variables by such formulations leads to high performance.

However, in sparse models, sparsity is encouraged with little regard for the underlying structural relationship between the variables. This can often result in overfitting of the noise and inconsistent variable selection across different experiment trials, especially when the dimension/data ratio is high. In this regard, sparse models were recently extended to explore structures of variables, and the problem of interest is often referred to as structured sparse learning. A variety of regularization for different structures and efficient algorithms solving the corresponding optimizations have been proposed [Huang et al. 2011; Bach et al. 2012]. Fused lasso [Tibshirani et al. 2005] is one of these variants, where pairwise differences between variables are penalized using the l_1 norm and thereafter selects sparse segments.

1.1. Generalized Fused Lasso

Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ be a set of samples, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. $\mathbf{X} \in \mathbb{R}^{d\times N}$ and $\mathbf{y} \in \mathbb{R}^N$ denote the concatenations of $\mathbf{x}_i$ and y_i, respectively. We start from the definition of (1D) fused lasso, which was first proposed by Tibshirani et al. [2005] and is formulated as

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2 + \lambda_1 \sum_{i=1}^{d} |\beta_i| + \lambda_2 \sum_{i=2}^{d} |\beta_i - \beta_{i-1}|, \tag{1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^d$ and $\lambda_1, \lambda_2 \geq 0$. Here, the variables, i.e., $\boldsymbol{\beta}$, are assumed to have a meaningful ordering, e.g., forming a chain structure. Due to the l_1 penalties on both single variables and consecutive pairs, solutions tend to be sparse and smooth—that is, consecutive variables tend to be similar. The third term is usually called the *fusion penalty*.

The preceding fused lasso method was proposed to pursue sparse segments on a chain of variables. Thus, a natural generalization of this conventional 1D fused lasso aims to promote smoothness between neighboring variables on a general/arbitrary graph structure. Suppose that we have a graph $G = (V, E)$ with nodes V and edges E, where each variable corresponds to a node on the graph; we can define such a generalization as

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\beta}\|_2^2 + \lambda_1 \sum_{i=1}^{d} |\beta_i| + \lambda_2 \sum_{(i,j)\in E} |\beta_i - \beta_j|. \tag{2}$$

Equation (2) is usually referred to as generalized fused lasso (GFL).

In the present study, we propose to solve the following more general problem,

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} l(\boldsymbol{\beta}) + \lambda_1 \sum_{i=1}^{d} |\beta_i| + \lambda_2 \sum_{(i,j)\in E} w_{ij}|\beta_i - \beta_j|, \tag{3}$$

where $l : \mathbb{R}^d \to \mathbb{R}$ is a smooth convex function of the type $C^{1,1}$ [Nesterov 2004] and $w_{ij} \in \mathbb{R}$ are weights defined on the edges of the graph. With a slight abuse of notation, we also refer this formulation to GFL in this article. The benefit of these generalizations is clear cut: (1) by applying a general loss term, nonlinear functions such as logistic

Table I. Relations of GFL to Existing Models

Models	$l(\cdot)$	$\lambda_1 \geq 0$	$\lambda_2 \geq 0$	Graph
Sparse models	$C^{1,1}$	$\forall$	0	None
TV	$\|\boldsymbol{\beta} - \mathbf{z}\|_2^2$	0	$\forall$	Undirected
Fused lasso	$\|\mathbf{X}^T\boldsymbol{\beta} - \mathbf{y}\|_2^2$	$\forall$	$\forall$	Chain
GFL	$C^{1,1}$	$\forall$	$\forall$	General

regression (LR; for classification problems) can be applied, and (2) by inducing weights w_{ij}, extra prior knowledge can be incorporated to adaptively control the strength of fusion. In Table I, we list some well-known models that can be viewed as special cases of this extended GFL model. Specifically, sparse models like lasso, compressive sensing, and so forth promote pure sparsity by solving the problem with $\lambda_2 = 0$ where the fusion term is not consider at all. The total variation (TV) formulation, often used in image processing literature as a robust smoother/denoiser, is another special case where the loss term is signal approximation, i.e., $\|\boldsymbol{\beta} - \mathbf{z}\|_2^2$, and it does not have the sparsity-inducing l_1 term. Therefore, the GFL can be viewed as using a combined prior of both sparsity inducing and (robust)[1] smoothness promoting, which ends up selecting sparse cliques of variables. To efficiently solve this general GFL problem is the focus of this article. Portions of this work have previously appeared in conference proceedings [Xin et al. 2014, 2015a, 2015b].

1.2. Existing Algorithms

The first algorithm for solving fused lasso, i.e., Equation (1), which was proposed by Tibshirani et al. [2005], is based on the two-phase active set algorithm SQOPT [Gill et al. 1999]. This algorithm can be extended to GFL and implemented using an off-the-shelf convex optimization solver. In general, however, it does not scale to high-dimensional problems. (Accelerated) proximal gradient methods such as the fast iterative shrinkage-thresholding algorithm (FISTA) [Beck and Teboulle 2009] solve convex problems whose objective comprises both smooth and nonsmooth parts. Using FISTA, Liu et al. proposed to solve Equation (1) by designing specific proximal operators [Liu et al. 2010]. Although their algorithm is efficient and scalable for the 1D case, it cannot be extended to GFL, i.e., on general graph structure, in principle. Friedman et al. proposed a pathwise coordinate descent algorithm for a special case of Equation (2) [Friedman et al. 2007], where the design matrix $\mathbf{X}$ is the identity matrix. The reported efficiency of the algorithm is impressive; however, as suggested in Friedman et al. [2007], this algorithm is not guaranteed to find exact solutions to general problems. In Tibshirani and Taylor [2011], a solution path algorithm is proposed for Equation (2). This algorithm solves for all possible parameters (λs) by finding critical changing points in a dual problem, which, however, tend to be very dense in large problems.

In the present study, we propose an efficient and scalable algorithm for solving GFL. Using proximal methods, the key building block of our algorithm is the fused lasso signal approximation (FLSA). Based on the fact that fusion penalty is the Lovász extension of a cut function, we apply a parametric flow algorithm and then the soft-thresholding method to solve the FLSA in an efficient manner. The proposed algorithm can find an exact solution (with respect to machine precision) to GFL, and it can also be implemented with a stable and efficient parametric flow solver. Our runtime experiments demonstrate that while solving the 1D fused lasso problem, the speed of the proposed algorithm is competitive compared to the state-of-the-art algorithms.

[1]Robustness is addressed as compared to graph Laplacian smoothness, i.e., $\sum_{(i,j)\in E} w_{ij}\|\beta_i - \beta_j\|_2^2$, where the quadratic smoothing often overpenalizes large differences. Note that in practice, graph Laplacian also tends to have a counter effect to sparsity inducing.

Table II. Definitions of Major Notations in This Article

Symbol	First Appearance	Definition
$\mathbf{X}$, $\mathbf{y}$	Eq. (1)	Data used by GFL, e.g., $\mathbf{X}$ are features and $\mathbf{y}$ are labels.
$\boldsymbol{\beta}$	Eq. (1)	Variables of GFL.
λ_1, λ_2	Eq. (1)	Tuning parameters of GFL, controlling the contribution of sparsity-inducing and fusion terms.
$l()$	Eq. (3)	Any smooth convex function of the type $C^{1,1}$.
w_{ij}	Eq. (3)	Weights associated with edge $(i, j) \in E$.
k	Eq. (4)	Iteration index, i.e., $\boldsymbol{\beta}_k$ is the value of $\boldsymbol{\beta}$ in the kth iteration of gradient descent.
L	Eq. (4)	Lipschitz constant of $\nabla l()$.
$\Omega_{gfl}(\boldsymbol{\beta})$	Eq. (5)	$\Omega_{gfl}(\boldsymbol{\beta}) = \lambda_1 \sum_{i=1}^{d} \lvert\beta_i\rvert + \lambda_2 \sum_{(i,j)\in E} w_{ij} \lvert\beta_i - \beta_j\rvert$.
$\mathbf{z}$	Eq. (7)	$\mathbf{z} = \boldsymbol{\beta}_k - \frac{1}{L}\nabla l(\boldsymbol{\beta}_k)$.
$\boldsymbol{\beta}_{\lambda_2}^{\lambda_1}$	Eq. (8)	$\boldsymbol{\beta}_{\lambda_2}^{\lambda_1} = \arg\min_{\boldsymbol{\beta}} f(\boldsymbol{\beta}, \lambda_1, \lambda_2)$, where $f(\boldsymbol{\beta}; \lambda_1, \lambda_2)$ is the objective of Equation (7).
$\mathcal{V}$	Eq. (10)	A finite set, in particular, corresponds to the index of set of $\boldsymbol{\beta}$, i.e., $\{i \lvert i \in 1, \ldots, d\}$.
$\mathcal{S}$	Eq. (10)	Any subset of $\mathcal{V}$.
$f_c()$	Eq. (10)	A cut function.
$f_c^*()$	Eq. (11)	Lovász extension of the cut function $f_c()$.
$B()$	Eq. (12)	Base polyhedron of a submodular function.
$\mathbf{t}$	Eq. (12)	An auxiliary variable, whose l_2 norm is to be minimized.
$g()$	Eq. (13)	Auxiliary submodular function.
α, γ	Eq. (14)	Auxiliary variables for deriving parametric graph cuts.
$\mathbf{x}_i$, y_i	Eq. (16)	AD features and labels.
c	Eq. (16)	Bias term in linear models.
$\mathbf{X}$, $\mathbf{y}$	Eq. (17)	BS data, i.e., $\mathbf{X}$ is the training frames and $\mathbf{y}$ is the testing frame.
$\mathbf{a}$	Eq. (17)	Sparse linear coefficient, for modeling the background relationship.
$\mathbf{e}$	Eq. (17)	Structured sparse foregrounds, whose prior is captured by the GFL regularization.

Nevertheless, when solving the GFL problem, it significantly outperforms all existing algorithms, especially with high-dimensional data.

The remainder of this article is organized as follows. In Section 2, we introduce two practical problems that motivated our work, namely the diagnosis of Alzheimer's disease (AD) problem and the background subtraction (BS) problem. In Section 3, we propose our algorithm for solving GFL, where the equivalence with parametric graph cuts are established and an efficient optimization via parametric flow is proposed. The efficiency of the proposed algorithm is validated in Section 4. In Sections 5.1 and 5.2, we formulate both the AD and BS problems as GFL, respectively, and demonstrate its promising performance. Section 6 concludes the article. Before proceeding, we summarize the definitions of the major notations in Table II.

2. MOTIVATION

In this section, we introduce two specific artificial intelligence problems that motivated our work in the first time. However, GFL can be applied to pursue structured sparse pattern wherever prior information is expressed using a graph over the variables. Its applications are not limited to the following ones.

2.1. Diagnosis of AD

A strong motivation of our work comes from the problem of the diagnosis of AD, which is a challenging real-world problem. This problem is usually formulated as a classification and/or prediction task, where structural magnetic resonance images (sMRIs) of human brains are used as the input features. Because of its practical benefit, this problem is

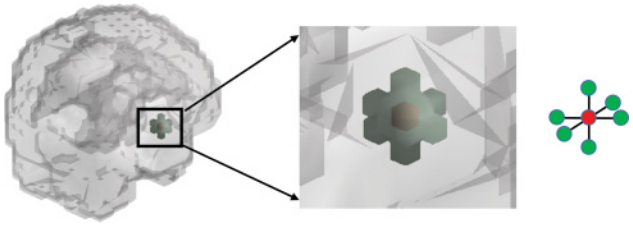


Fig. 1. A brain voxel (in red) and its adjacency (in green), and the corresponding graph representation.

increasingly attracting many researchers from various fields, such as computer vision and machine learning [Wan et al. 2012; Zhou et al. 2011].

In practice, the dimensionality of a brain image can be as high as millions, whereas the number of available samples is usually limited to hundreds. Thus, appropriate prior information is required. The critical brain voxels should be sparse and spatially assembled into several anatomical regions with early damage. Existing methods either assume independence between voxels, e.g., univariate selection [Dai et al. 2012], or they use the volume of interest (VOI) [Zhou et al. 2011] data as a processing unit, which loses much of the pathological information and might not be sufficiently sensitive for early diagnosis.

By considering the structure of a brain sMRI as a 3D grid graph (Figure 1), we propose to formulate the diagnosis of AD as GFL. However, the existing algorithms do not scale sufficiently well to solve this problem in feasible time. Thus, we demonstrate the effectiveness of the proposed algorithm, which solves the problem within limited memory and time, as well as yielding promising classification accuracy, which is competitive among the state-of-the-art methods. Perhaps more importantly from the pathological standpoint, the selected voxels are very well structured: being connected, consistent according to cross validation and in agreement with pathological prior knowledge.

2.2. Background Subtraction

BS is one of the key tasks for automatic video analysis. Usually, BS methods distinguish foreground pixels from the background in a video sequence by designing a background model and then comparing the current frame with the model.

Many methods for BS have been proposed in recent years. Comprehensive reviews can be found in Herrero and Bescós [2009] and Brutzer et al. [2011]. One series of successful work in this literature is to use Gaussian mixture models (GMMs) trained from previous frames to model the intensity distribution of each pixel, e.g., Stauffer and Grimson [1999] and Zivkovic and van der Heijden [2006]. Although nonparametric methods are also proposed for better efficiency, e.g., Barnich and Van Droogenbroeck [2011], most of these methods remain in the pixel-wise modeling framework and leave the task of utilizing interpixel structural information to delicate postprocessing [Brutzer et al. 2011; Haines and Xiang 2012]. Another series of work follows the celebrated *eigenbackground* [Oliver et al. 2000]. They build a background subspace via PCA and therefore are able to utilize background structure information. Perhaps not surprisingly, however, the extracted foregrounds are still rather scattered, and there exist both "holes" in the foregrounds and noisy falsely detected background pixels.

Quite recently, research such as that of Xu et al. [2013] and Mairal et al. [2011] apply group lasso (with overlap) regularization to explicitly promoted structured foregrounds. However, it is questionable whether predefined grouping of pixels is a good prior for arbitrary foreground structures. Our experimental results show that even with elaborately designed grouping, these models perform inferiorly to GFL, which naturally

models an arbitrary foreground shape (see Figure 12 in the Appendix). Nevertheless the efficiency of existing GFL algorithms prohibit its application to the BS problem, which usually deals with millions of variables. Therefore, we demonstrate the scalability of the proposed algorithm. Moreover, we show that the proposed algorithm can extend GFL using directed graphs. According to Boykov and Funka-Lea [2006], it is desirable to apply directed graph under certain conditions where asymmetric weights help to provide better segmentation at the boundaries.

3. EFFICIENT OPTIMIZATION FOR GFL

In this section, we propose an efficient and scalable optimization algorithm for GFL. First, we introduce FISTA, which is applied to solve GFL by iteratively calculating *proximal operators*. For GFL, we show that the computation of the proximal operator can be formulated as one of the FLSAs. We then propose a parametric optimization formulation to solve FLSA in an efficient manner, where we introduce a soft-thresholding strategy to discard the sparse term, transform the FLSA to a minimum-norm-point (MNP) problem under submodular constraints, prove its equivalence to recursively solving parametric graph-cut problems, and solve this problem using a parametric flow method.

3.1. Proximal Methods and FLSA

For smooth convex optimization problems, it was shown [Nesterov 2004] that there exists a gradient method with $O(1/k^2)$ (k is the iteration index) convergent, which is an "optimal" first order[2] method according to Nemirovsky and Yudin [1983]. By extending the method of Nesterov [2004] to the general case with nonsmooth terms, accelerated proximal methods like FISTA achieve the same convergence rate [Beck and Teboulle 2009]. The price paid here is that in each iteration, fast algorithms must be designed to solve a nonsmooth proximal operator. This idea has been applied to various sparse learning problems, e.g., Beck and Teboulle [2009] and Bach [2010], and to 1D fused lasso, e.g., Liu et al. [2010]. We also use FISTA to solve GFL in the present study.

Specifically, it is known that the optimization of any smooth objective function $l(\boldsymbol{\beta})$ can be achieved using a gradient method, where the updating rule of $\boldsymbol{\beta}$ can be viewed as minimizing an approximation of the linearization of $l()$ at the previous iteration $\boldsymbol{\beta}_k$, Polëiìak [1987], i.e.,

$$\boldsymbol{\beta}_{k+1} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ l(\boldsymbol{\beta}_k) \; + \langle \boldsymbol{\beta} - \boldsymbol{\beta}_k, \nabla l(\boldsymbol{\beta}_k) \rangle + \frac{L}{2} \|\boldsymbol{\beta} - \boldsymbol{\beta}_k\|_2^2 \right\}, \tag{4}$$

where $L > 0$ is the Lipschitz constant of $\nabla l()$.

Let us denote the regularization terms in Equation (3) as

$$\Omega_{gfl}(\boldsymbol{\beta}) = \lambda_1 \sum_{i=1}^{d} |\beta_i| + \lambda_2 \sum_{(i,j)\in E} w_{ij} |\beta_i - \beta_j|.$$

When there is a nonsmooth part $\Omega_{gfl}(\boldsymbol{\beta})$ in the objective function of GFL (Equation (3)), using FISTA changes the updating rule to

$$\boldsymbol{\beta}_{k+1} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ l(\boldsymbol{\beta}_k) + \langle \boldsymbol{\beta} - \boldsymbol{\beta}_k, \nabla l(\boldsymbol{\beta}_k) \rangle + \frac{L}{2} \|\boldsymbol{\beta} - \boldsymbol{\beta}_k\|_2^2 + \Omega_{gfl}(\boldsymbol{\beta}) \right\}. \tag{5}$$

[2]Second-order methods are seldom used for high-dimensional problems, mostly because the computation of the second-order Hessian matrix and its inverse is prohibitive when the feature dimensional is high.

ALGORITHM 1: FISTA Algorithm for GFL

Input: $L > 0$, $\boldsymbol{\beta}_0 \in \mathbb{R}^d$.
Output: $\boldsymbol{\beta}^*$.
$\mathbf{y}_1 = \boldsymbol{\beta}_0$, $t_1 = 1$.;
repeat
Call algorithms for proximal operator to solve

$$\boldsymbol{\beta}_k = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \Omega_{gfl}(\boldsymbol{\beta}) + \frac{L}{2} \left\| \boldsymbol{\beta} - \left(\mathbf{y}_k - \frac{1}{L} \nabla l(\mathbf{y}_k) \right) \right\|_2^2 \right\}.$$

Check stopping criteria, if true, return $\boldsymbol{\beta}_k$; else

$$t_{k+1} = \frac{1}{2} \sqrt{(1 + 4t_k^2)}, \mathbf{y}_{k+1} = \boldsymbol{\beta}_k + \frac{t_k - 1}{t_{k+1}} (\boldsymbol{\beta}_k - \boldsymbol{\beta}_{k-1}).$$

until *convergence*;

After some simple manipulations of Equation (5), e.g., ignoring constant terms of β_k, we have

$$\boldsymbol{\beta}_{k+1} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \Omega_{gfl}(\boldsymbol{\beta}) + \frac{L}{2} \left\| \boldsymbol{\beta} - \left(\boldsymbol{\beta}_k - \frac{1}{L} \nabla l(\boldsymbol{\beta}_k) \right) \right\|_2^2 \right\}. \tag{6}$$

Thus, the key to solving Equation (3) is how efficiently we can solve Equation (6), which can be rewritten as

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \frac{1}{2} \|\boldsymbol{\beta} - \mathbf{z}\|_2^2 + \lambda_1 \sum_{i=1}^{d} |\beta_i| + \lambda_2 \sum_{(i,j) \in E} w_{ij} |\beta_i - \beta_j|, \tag{7}$$

where $\mathbf{z} = \boldsymbol{\beta}_k - \frac{1}{L} \nabla l(\boldsymbol{\beta}_k)$ and λ_1 and λ_2 are scaled from Equation (3) by L. Problem (7) is the proximal operator of solving GFL using proximal methods. Actually, problem (7) is equivalent to the FLSA defined in Friedman et al. [2007] and Tibshirani and Taylor [2011], which is itself a very useful formulation.

3.2. An Efficient Solution to FLSA by Parametric Flow

To the best of our knowledge, there are no previous reports of an efficient method for solving the FLSA for high-dimensional problems. In the present study, we propose an efficient solution to the minimization problem (Equation (7)) by using a parametric flow method.

3.2.1. L_1 Soft Thresholding. First, let us denote the objective in Equation (7) by $f(\boldsymbol{\beta}; \lambda_1, \lambda_2)$ and $\boldsymbol{\beta}_{\lambda_2}^{\lambda_1} = \arg\min_{\boldsymbol{\beta}} f(\boldsymbol{\beta}, \lambda_1, \lambda_2)$. Then, we introduce the following lemma [Friedman et al. 2007; Liu et al. 2010].

LEMMA 3.1. *For any $\lambda_1, \lambda_2 \geq 0$, we have*

$$\boldsymbol{\beta}_{\lambda_2}^{\lambda_1} = \operatorname{sign}\left(\boldsymbol{\beta}_{\lambda_2}^0\right) \odot \max\left(|\boldsymbol{\beta}_{\lambda_2}^0| - \lambda_1, \mathbf{0}\right), \tag{8}$$

where $\odot$ is an element-wise product operator.

PROOF. The proof can be done by exploring the optimality condition of Equation (7). We give the sketch of the idea as follows. Note that since $\boldsymbol{\beta}_{\lambda_2}^0$ is the optimizer of $f(\boldsymbol{\beta}, 0, \lambda_2)$, it satisfies $\partial f(\boldsymbol{\beta}, 0, \lambda_2)/\partial \boldsymbol{\beta} = 0$ (subgradient is applied where nonsmooth). Since the additional $\|\boldsymbol{\beta}\|_1$ term in (7) is separable with respect to β_i, we can show that after applying the element-wise soft-thresholding defined in Equation (8), the resulted $\boldsymbol{\beta}_{\lambda_2}^{\lambda_1}$ satisfies $\partial f(\boldsymbol{\beta}, \lambda_1, \lambda_2)/\partial \boldsymbol{\beta} = 0$. □

According to Lemma 3.1, a solution to Equation (7) can be obtained using the soft-thresholding process. Therefore, based on this lemma, we will first solve the following problem:

$$\boldsymbol{\beta}^0_{\lambda_2} = \operatorname*{argmin}_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta} - \mathbf{z}\|_2^2 + \lambda_2 \sum_{(i,j)\in E} w_{ij}|\beta_i - \beta_j|. \tag{9}$$

Then, using Equation (8), a soft-thresholding process to $\boldsymbol{\beta}^0_{\lambda_2}$ with respect to λ_1, we obtain a solution to Equation (7).

3.2.2. MNP Problem under Submodular Constraints. Since the second term of Equation (9) is nonsmooth and nonseparable with respect to $\boldsymbol{\beta}$, its optimization is still nontrivial. Note that Equation (9) is known as a TV problem. Goldfarb and Yin [2009] used a parametric flow algorithm to solve (9), but we cannot extend the algorithm to exact GFL. This is because (1) the formulation gap between TV and GFL needs to be bridged, and (2) in Goldfarb and Yin [2009], they assumed that $\boldsymbol{\beta}, \mathbf{z} \in \mathbb{Z}^d_+$ (mostly for the benefit of image processing applications). However, in our problem, at each iteration, we often achieve continuous $\mathbf{z}$, and therefore by adopting the algorithm from Goldfarb and Yin [2009] will make the solution to GFL inexact.[3]

To develop an efficient algorithm for problem (9), we consider a transformation of problem (9) into an MNP problem under submodular[4] constraints. First, we propose the following lemma, which describes the relation between the fusion penalty and a cut function.

Let $\mathcal{V} := \{1, \ldots, d\}$ denote a finite set (corresponds to the index of each $\beta_i, i \in 1, \ldots, d$). Given a set of nonnegative weights $w : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_+$, a cut function of a set $\mathcal{S} \subseteq \mathcal{V}$ is defined by

$$f_c(\mathcal{S}) = \sum_{i\in\mathcal{S}, j\in\mathcal{V}\setminus\mathcal{S}} w_{ij}, \quad (\mathcal{S} \subseteq \mathcal{V}). \tag{10}$$

LEMMA 3.2. *The fusion term $\sum_{(i,j)\in E} w_{ij}|\beta_i - \beta_j|$ is equivalent to the Lovász extension of a cut function.*

The proof is provided in the Appendix.

COROLLARY 3.3. *When $w_{ij} \neq w_{ji}$, the Lovász extension becomes $\sum_{(i,j)\in E} \hat{w}_{ij}|\beta_i - \beta_j|$, where $\hat{w}_{ij} = \{ {}^{w_{ij} \quad \beta_i \geq \beta_j,}_{w_{ji} \quad \beta_i < \beta_j.}$. This defines a directed extension of the common fused lasso model.*

Remarks. Note that one cannot simply define directed fused lasso by applying $\sum_{(i,j)\in E} w_{ij}|\beta_i - \beta_j|$ with $w_{ij} \neq w_{ji}$. Because in this way, the fusion term is still undirected with $w'_{ij} = (w_{ij} + w_{ji})$. Therefore, to view the fused lasso as the Lovász extension of cut functions is necessary to derive directed fused lasso. Later, our experimental results show that by using directed weights, GFL can further improve the performance to certain applications.

[3]In Chambolle and Darbon [2009], a continuous TV is considered; however, the proposed algorithm is designed for calculating a sequence of solutions to their own problems. It is difficult to extend to other formulation. Nevertheless, the proposed algorithm is based on a specification of a more general theoretical framework (and the method of Chambolle and Darbon [2009] can be viewed implicitly as one special case). In other words, we derive to the parametric flow problem using the fact that the regularization term is an instance of the Lovász extension of the generalized graph-cut functions (a type of submodular function). This framework is easy to extend to other regularization terms.

[4]Some preliminary knowledge about submodular functions is provided in the Appendix.

With Lemma 3.2, we can rewrite Equation (9) as

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2}\|\boldsymbol{\beta}-\boldsymbol{z}\|_2^2 + \lambda_2 \cdot \hat{f}_c(\boldsymbol{\beta}). \tag{11}$$

Since a cut function is submodular, according to Bach [2010], this optimization problem can be transformed into an MNP problem under submodular constraints.

PROPOSITION 3.4. *Problem* (11) *is equivalent to the following problem:*

$$\min_{\mathbf{t}\in\mathbb{R}^d, \mathbf{t}\in B(f_c-\lambda_2^{-1}\mathbf{z})} \|\mathbf{t}\|_2^2, \tag{12}$$

where $B(\bullet)$ *is the base polyhedron of a submodular function* $\bullet$*. A minimizer* $\boldsymbol{\beta}^*$ *of problem* (11) *is obtained by* $\boldsymbol{\beta}^* = -\lambda_2 \mathbf{t}^*$*, where* $\mathbf{t}^*$ *is a minimizer of problem* (12)*.*

The proof is provided in the Appendix for the completeness of the article.

For general submodular functions, problem (12) is solvable using submodular minimization algorithms, such as the MNP algorithm [Fujishige et al. 2006]. However, the known fastest time complexity of submodular minimization is $O(d^5EO + d^6)$ [Orlin 2009], where EO is the cost for a function evaluation, and thus this approach to high-dimensional problems is infeasible in practice.

3.2.3. Parametric Graph Cut. To solve problem (12) in an efficient manner, we utilize a parametric property of our MNP problem and apply a parametric flow algorithm, which has a much less time complexity and can run very efficiently in practice.

The set function $g(\mathcal{S}) = f_c(\mathcal{S}) - \lambda_2^{-1}\mathbf{z}(\mathcal{S})$ in Equation (12) is the sum of a cut function and a modular function, which is still submodular (but not necessarily non-decreasing). Thus, problem (12) is a special case of a separable convex minimization problem under submodular constraints [Nagano and Aihara 2012], which can be solved by parametric optimization (if the submodular function is nondecreasing). Now let us first assume that $g(\mathcal{S})$ is nondecreasing. We will later describe how to satisfy this nondecreasing requirement in Lemmas 3.5 and 3.6.

For a parameter $\alpha \geq 0$, we define a set function $g_\alpha(\mathcal{S}) = g(\mathcal{S}) - \alpha \cdot \mathbf{1}(\mathcal{S})$. Now that we have assumed that g is a nondecreasing submodular function, there exists $l+1$ ($\leq d$) subsets

$$\mathcal{S}^* = \{(\emptyset =)\,\mathcal{S}_0 \subset \mathcal{S}_1 \subset \ldots \subset \mathcal{S}_l\,(=\mathcal{V})\},$$

and $l+1$ subintervals of

$$R_0 = [0, \alpha_1),\ R_1 = [\alpha_1, \alpha_2),\ \ldots,\ R_l = [\alpha_l, \infty),$$

such that, for each $j \in \{0, \ldots, l\}$, $\mathcal{S}_j$ is the unique maximal minimizer of $g_\alpha(\mathcal{S})$ for all $\alpha \in R_j$ [Nagano and Aihara 2012]. Then, the unique optimal solution $\mathbf{t}^* \in \mathbb{R}^d$ to problem (12) is determined by

$$t_i^* = \frac{g(\mathcal{S}_{j+1}) - g(\mathcal{S}_j)}{\mathbf{1}(\mathcal{S}_{j+1} \setminus \mathcal{S}_j)} \tag{13}$$

for each $i \in \mathcal{V}$ with $i \in \mathcal{S}_{j+1} \setminus \mathcal{S}_j\,(j \in \{1, \ldots, l-1\})$. Thus, by computing the unique maximal minimizer of g_α for some appropriately selected αs, we can find all $\mathcal{S}_j$ and therefore compute $\mathbf{t}^*$. A possible option for finding all "appropriate" αs would be to apply the decomposition algorithm [Fujishige 2005; Nagano and Aihara 2012], which recursively finds all $\mathcal{S}_j$s.

Recall that g has to be a nondecreasing function to apply the preceding procedure. However, this is not always the case for our choice $g = f - \lambda_2^{-1}\mathbf{z}$. Therefore, we introduce two lemmas from Fujishige [2005].

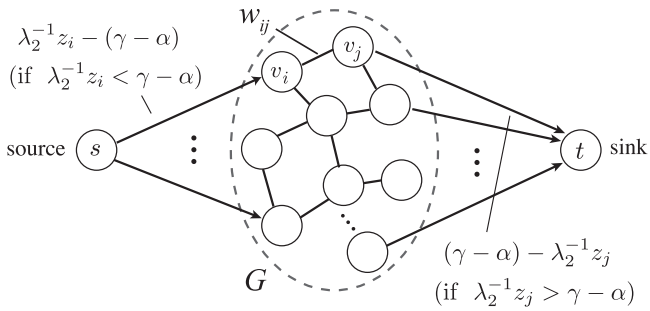


Fig. 2. Construction of an s-t graph for problem (14). Given a graph $G = (V, E)$ for GFL, the capacities on edges are defined as follows: $c(v_i, v_j) = w_{ij}$ $(i, j \in V)$, $c_{s,v_i} = \lambda_2^{-1}z_i - (\gamma - \alpha)$ if $\lambda_2^{-1}z_i < \gamma - \alpha$ or $c_{s,v_i} = 0$ and otherwise $(i \in V)$, and $c_{v_i,t} = (\gamma - \alpha) - \lambda_2^{-1}z_i$ if $\lambda_2^{-1}z_i > \gamma - \alpha$ or $c_{v_i,t} = 0$ and otherwise $(i \in V)$, where c_{s,v_i} and $c_{v_i,t}$ denote the capacities of the source-to-node and node-to-sink edges.

LEMMA 3.5. *For any $\gamma \in \mathbb{R}$ and a submodular function f, $\mathbf{t}^*$ is an optimal solution to $\min_{\mathbf{t}\in B(f)} \|\mathbf{t}\|_2^2$ if and only if $\mathbf{t}^* + \gamma\mathbf{1}$ is an optimal solution to $\min_{\mathbf{t}\in B(f+\gamma\mathbf{1})} \|\mathbf{t}\|_2^2$.*

LEMMA 3.6. *Set $\gamma = \max_{i=1,\ldots,d}\{0, f(V\backslash\{i\}) - f(V)\}$, then $f + \gamma\mathbf{1}$ is a nondecreasing submodular function.*

By applying Lemma 3.6 to our case with $f := g$, we have a nondecreasing submodular function $g' = g + \gamma\mathbf{1}$. Then, after we have found the minimizer of the MNP under the constraint of g', i.e., $\mathbf{t}^* + \gamma\mathbf{1}$, we can apply Lemma 3.5 to obtain a solution of the original problem.

Now that we have a nondecreasing submodular function, by applying the decomposition algorithm, in each recursion, we solve the following problem:

$$\min_{\mathcal{S}\subset\mathcal{V}} f_c(\mathcal{S}) - \lambda_2^{-1}\mathbf{z}(\mathcal{S}) + (\gamma - \alpha)\cdot\mathbf{1}(\mathcal{S}). \tag{14}$$

Owing to the specific form of problem (14), we can solve it as an easier problem as follows.

PROPOSITION 3.7. *For any cut function f_c, problem (14) is equivalent to an s-t cut problem with the s-t graph defined in Figure 2.*

PROOF. Problem (14) comprises modular terms and a submodular pairwise term. This is a typical $\mathcal{F}^2$ type energy function [Kolmogorov and Zabin 2004], which is known to be "graph representable" and can be minimized via graph-cut algorithms. Hence, by following the construction of an s-t graph according to Kolmogorov and Zabin [2004], we can solve problem (14) by solving an s-t cut on this graph. □

As a consequence, we can obtain a solution to Equation (12) by solving s-t cut problems for some different αs:

$$\text{Find minimum } s\text{-}t \text{ cuts with respect to Equation (14) for different } \alpha \geq 0, \tag{15}$$

each of which can be efficiently solved via a max-flow algorithm. However, since the parameter α only affects the edges from the source node or to the sink node, we do not need to search αs that yield different solutions. Specifically, as can be seen from the construction of the s-t graph, the capacities on source-to-node or node-to-sink edges have the following properties: (1) the capacities on source-to-node edges are nondecreasing functions of α, (2) the capacities on node-to-sink edges are nonincreasing functions of α, and (3) the capacities on node-to-node edges are constant with respect to α. For such cases, it is known that the parametric flow algorithm reported by Gallo et al. [1989]

ALGORITHM 2: Proximal Operator via Parametric Flow

Input: $\lambda_1, \lambda_2 > 0$ and $\mathbf{z} \in \mathbb{R}^d$.
Output: $\boldsymbol{\beta}^*$ for problem (7).
Compute γ from Lemma 3.6;
Set up an s-t graph as in Figure 2 and find S^* with parametric flow;
Compute from (13) and apply Lemma 3.5 to get $\mathbf{t}^*$;
$\boldsymbol{\beta}^0_{\lambda_2} = -\lambda_2 \mathbf{t}^*$.;
Apply Lemma 3.1 to get $\boldsymbol{\beta}^{\lambda_1}_{\lambda_2}$ and $\boldsymbol{\beta}^* = \boldsymbol{\beta}^{\lambda_1}_{\lambda_2}$;

(the GGT algorithm) can be applied to find all solutions for all $\alpha \in \mathbb{R}$. Thus, we can obtain the sequence of solutions to problem (14) for different αs by simply applying the GGT algorithm, which runs in $O(d|E| \log(d^2/|E|)$ as the worst case.

For better clarity, we summarize all of the steps in Algorithm 2, which is practically very scalable (for solving MNP) and serves as the building block of Algorithm 1 to finally solve GFL in an efficient manner.

4. RUNTIME COMPARISON

We investigated the efficiency of the proposed algorithm, i.e., fast generalized fused lasso (fGFL). All experiments were performed using an Intel Xeon E5-2687 CPU at 3.10GHz with 64G memory. Our implementation of FLSA was written in C++ and that of FISTA in Matlab.[5]

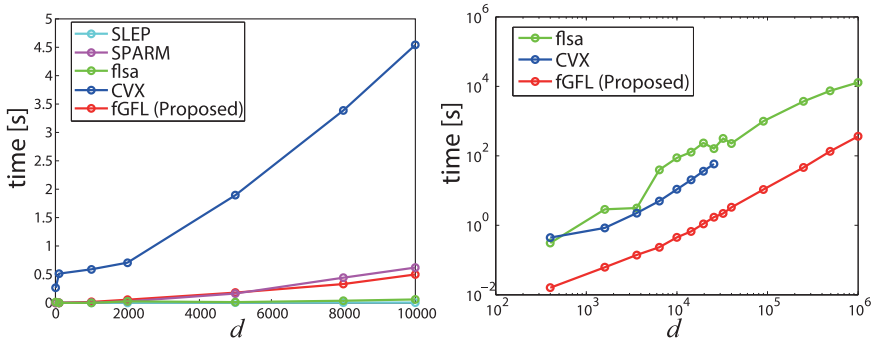
As mentioned in Section 1.2, several algorithms have been proposed for FLSA and GFL. Here we compare the proposed fGFL with the following state-of-the-art algorithms:

—*SLEP package* [Liu et al. 2009, 2010]: Implemented with Matlab and C for 1D fused lasso and 1D FLSA.
—*SPAMS* [Mairal et al. 2011]: Implemented with C for 1D fused lasso and 1D FLSA.
—*"flsa" R package*: Implemented with R for general FLSA, which includes accelerated implementations for 1D and 2D (grid) FLSA.
—*"genlasso" R package* [Tibshirani and Taylor 2011]: Implemented with R for GFL, which includes accelerated implementations for 1D and 2D (grid) fused lasso. (Note that it is limited to cases of $N \geq d$.)
—*CVX* [Grant et al. 2008]: This is a general convex optimization toolbox. We employed its general-use optimizer for GFL and FLSA.

We compared the application of the algorithms to 1D and 2D cases of FLSA defined in Equation (7) and GFL defined in Equation (2). (Note that the proposed fGFL can be applied to a more general case of Equation (3), for which most existing algorithms are not applicable. Later we demonstrate the advantage of Equation (3) and our solution based on their application to the AD problem.)

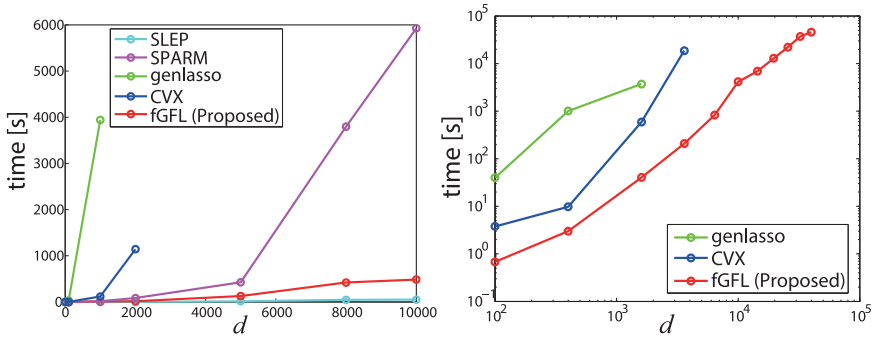
We generated data for the runtime comparison in the following manner. First, for 1D fused lasso, i.e., Equation (1), we set parameter $\boldsymbol{\beta}$ as $\beta_i = 0.5$ for $i \in \{d/2 - d/20, \ldots, d/2 + d/20\}$ and 0 otherwise. For 2D fused lasso, we set $\beta_{i,j} = 0.5$ for $i, j \in \{\sqrt{d}/2 - \sqrt{d}/20, \ldots, \sqrt{d}/2 + \sqrt{d}/20\}$ and 0 otherwise. For FLSA defined in Equation (7), we set $\mathbf{z} = \boldsymbol{\beta} + 0.05\mathbf{e}$, where $\mathbf{e}$ is a noise vector drawn from the standard normal distribution. For GFL as in Equation (2), we generated $N = d$ samples (because "genlasso" cannot solve Equation (2) when $N < d$): $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i = \boldsymbol{\beta}^T \mathbf{x}_i + 0.05e_i$, $\mathbf{x}_i$ and e_i for $i = 1, \ldots, N$ are drawn from the standard normal distribution. We fixed

[5]The codes can be found at https://sites.google.com/site/jimxinbo/.



(a) 1D FLSA (b) 2D FLSA

Fig. 3. FLSA runtime comparison (in seconds) using different algorithms with variable dimensionality d.



(a) 1D fused lasso (b) 2D fused lasso

Fig. 4. GFL runtime comparison (in seconds) using different algorithms with variable dimensionality d.

$\lambda_1, \lambda_2 = 0.1$ and applied the algorithm to a different dimension d to compare the runtime. The graphs in Figure 3 and 4 show the runtimes obtained using the algorithms.

The algorithm that used the standard optimizer, e.g., CVX, needs to handle the huge difference matrix $\mathbf{D} \in \mathbb{R}^{d \times |E|}$ for high-dimensional problems, which results in a memory shortage. The number of critical points found by "genlasso" significantly increases in high-dimensional problems, so we used the setting of "maxsteps=10,000," i.e., "genlasso" will find a maximum of 10,000 critical points. Both the explanations account for the missing plots in Figures 3 and 4. Nevertheless, as illustrated in the 1D cases, our algorithm was not the fastest, but it was competitive compared to the faster algorithms. In general cases of GFL, e.g., 2D, our algorithm was the fastest compared to existing ones. The speedup went to hundreds of times at the problem dimension around 5,000.

5. APPLICATIONS

5.1. Diagnosis of AD

In the diagnosis of AD, two fundamental issues are AD/NC (normal healthy controls) classification and MCI (mild cognitive impairment) conversion prediction, namely $\text{MCI}_C/\text{MCI}_S$ classification. Let $\mathbf{x}_i \in \mathbb{R}^d$ be the subject's sMRI features, and let $y_i = \{0, 1\}$ be the subject's disease status (AD/NC or $\text{MCI}_C/\text{MCI}_S$). Since our algorithm is applicable to general smooth convex loss terms, we used LR for the classification task and

Table III. Classification Accuracies (Acc.), Sensitivities (Sens.), and Specificities (Spec.) by Different Models Applied to AD

	ADNC			MCI		
	Acc.	Sens.	Spec.	Acc.	Sens.	Spec.
SVM	82.71%	80.65%	84.51%	67.38%	40.74%	**83.91%**
MLDA	**84.21%**	**84.51%**	83.87%	63.83%	**65.52%**	61.11%
Laplc	83.41%	79.03%	87.24%	70.21%	40.74%	88.51%
LR	80.45%	74.19%	85.92%	63.83%	50.00%	72.41%
L1	81.20%	75.81%	85.92%	68.79%	48.15%	81.61%
GFL	**84.21%**	80.65%	**87.32%**	**70.92%**	50.00%	**83.91%**

formulated the problem as GFL in the following manner:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d, c\in\mathbb{R}} \sum_{i=1}^{N} \log\left(1+\exp\left(-y_i(\boldsymbol{\beta}^T\mathbf{x}_i+c)\right)\right) + \lambda\Omega_{gfl}(\boldsymbol{\beta}). \tag{16}$$

We define the graph structure as illustrated in Figure 1. Problem (16) is an exact instance of problem (3), and thus the proposed algorithm can be directly applied. Note that other existing algorithms are not feasible in practice, even if we adopt the least square loss as in Equation (2).

Data. Data used in our experiments is obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (http://adni.loni.ucla.edu). We used 1.5T baseline MRI scan data. In the study, 62 AD patients, 71 NC, and 141 MCI patients (54 MCI_C and 87 MCI_S) are included. In the Appendix, we provide all subject IDs for easy repeating of the experiments. Data preprocessing is done following the DARTEL VBM pipeline [Ashburner 2007], as is commonly done in the literature. Here, 2,873 $8 \times 8 \times 8$ mm^3 size voxels that have values greater than 0.2 in the mean grey matter (GM) population template serve as the input features. We design experiments on both AD/NC classification (ADNC) and MCI prediction (MCI) tasks.

Performance. Ten-fold cross-validation evaluation is applied and classification accuracy for all tasks are summarized in Table III. Under the same experiment setup, we compare GFL with LR, SVM, sparse modeling, e.g., the lasso (l_1), and its graph Laplacian structured variants, i.e., Laplc, and the "MLDA" model [Dai et al. 2012], which applies a variant of Fisher discriminant analysis after univariate feature selection (via t-test). For each model, we used grid search to find the optimal parameters, respectively. Based on the accuracy, GFL outperforms LR, l_1, Laplc, and SVM on each task and achieves better results than MLDA on most tasks. MCI tasks are of more clinical importance and, in general, are more challenging than ADNC tasks. Notice that GFL obtains better performance gain in MCI tasks. These promising results justify that by inducing both sparsity and fusion priors, GFL captures useful information about AD.

Strictly speaking, it is hard to compare to other reported works on the AD problem. This is because most work removes samples whose MRI data are irregular—that is, outliers. In this way, different work ends up selecting different samples. Although we do not explicitly remove "outliers," our results still seem to be among the state of the art. For example, in Cheng et al. [2012], their best performance on MCI tasks is 69.4%, whereas our performance reached 70.92%. In Chu et al. [2012], our performance on ADNC tasks is comparable to or better than all of theirs (84.21% vs. 81% to 84%), and our performance on MCI tasks is much better (70.92% vs. 65%).

Feature selection. For each task, we perform feature selection using all data with the optimal parameters selected via cross validation. The selected features are those whose β are not zeros. In Figure 5, the result of ADNC is used to illustrate the feature selection by different models. In the top row, we illustrate all selected voxels. In the middle row,

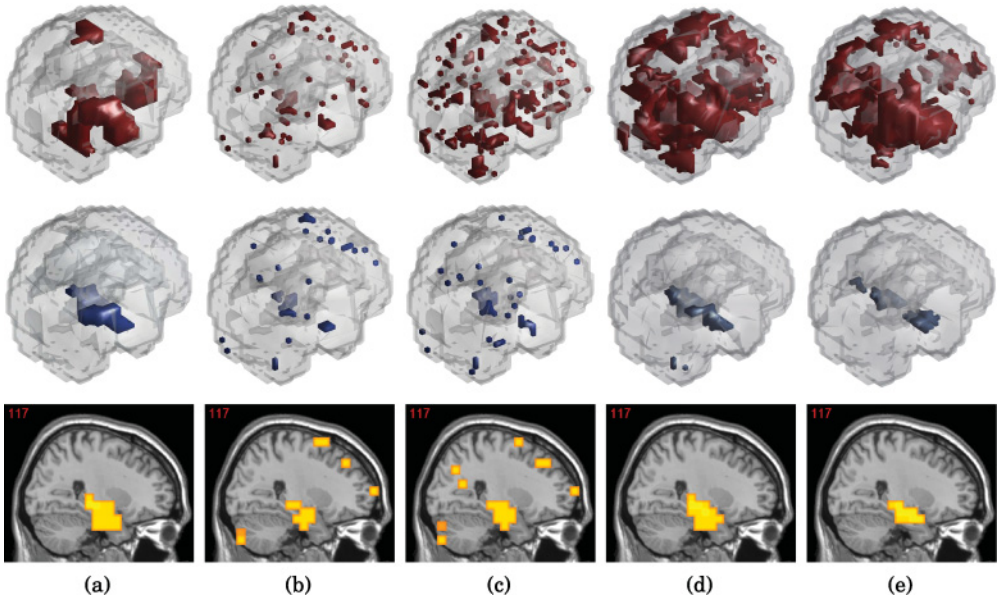


Fig. 5. Comparison of GFL with existing models. The top row illustrates selected voxels in a 3D brain model, the middle row illustrates the top 50 atrophied voxels, and the bottom row illustrates a projection onto one brain slice. (a) GFL (best accuracy 84.21%). (b) l_1 (best accuracy 81.20%). (c) l_1 (similar voxel number as (a)). (d) Laplc (best accuracy 83.41%). (e) t-Test (best accuracy 84.21%).

we illustrate the voxels corresponding to the top 50 negative β_i's (indicating the most atrophied voxels). We then project the most atrophied voxels onto a slice in the bottom row. We see that the selected voxels by GFL cluster into meaningful spatially connected regions while selected voxels by l_1 and t-test scatter around. The selected voxels by Laplc tend to be much more than necessary due to the l_2 regularization. Moreover, the selected voxels of GFL are concentrated in the hippocampus and parahippocampal gyrus (which are believed to be early damaged regions). On the other hand, l_1 either selects less critical voxels or probably selects noisy voxels not in early damaged regions (see Figures 5(b) and (c) for an illustration).

To further compare the behavior of GFL with l_1, in Figure 6 we illustrate the consistency of the most atrophied voxels in different folds of the cross validation. We see that the selected voxels by GFL have highly consistent spatial patterns. By comparing Figure 5 to Figure 6, we see that the consistent voxels are also pathologically meaningful, as they correspond to the early damaged regions. On the other hand, the selected voxels by l_1 change much over each fold of cross validation. This comparison is also quantitatively justified by the percentage of overlapped voxels: GFL (66%) versus l_1(22%). The inconsistency of selected voxels by l_1 indicates that rather than capturing meaningful information of the diagnosis, it is probably affected by sample-dependent information, e.g., noise.

In Figure 7, we demonstrate an interesting phenomenon by changing λ_2 while λ_1 is fixed. From Figure 7(a) through (d), we notice that as λ_2 increases, more voxels are selected and the connection between voxels becomes more obvious. However, the additional voxels from one to another are not randomly selected. They emerge along the boundary of the former selected voxels. Meanwhile, separated voxels either disappear or are connected to large regions. From the perspective of clinical usage, this is a

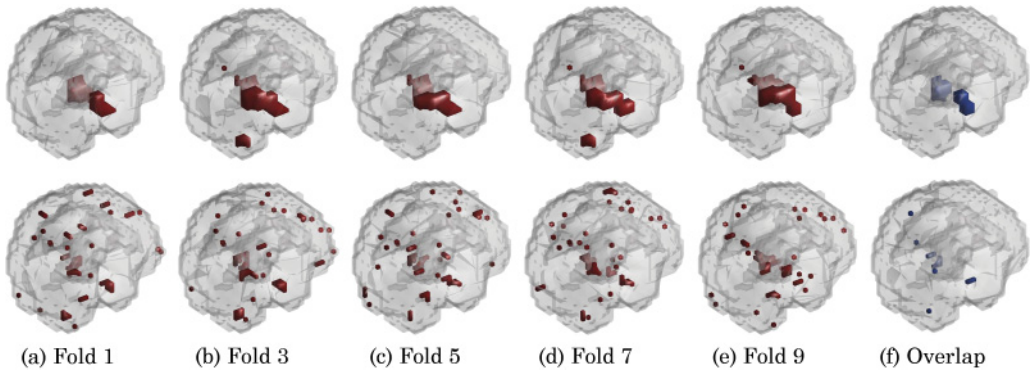


Fig. 6. Consistency of selected voxels over different folds of the cross validation. The results of 5 different folds are shown in (a) through (e), respectively, and the overlapped voxels of all 10 folds are shown in (f). The top row illustrates results from GFL, and the bottom row illustrates results from l_1. The percentage of the overlapped voxels are GFL(66%) versus l_1(22%).

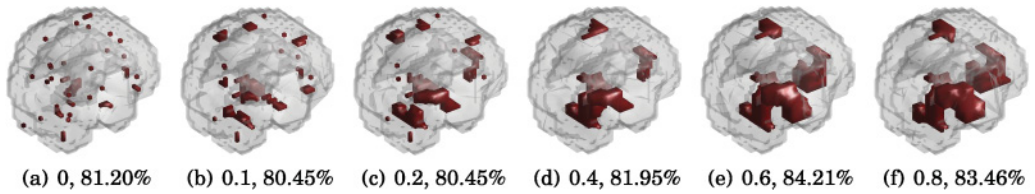


Fig. 7. Different levels of cohesion. We fix λ_1 and change λ_2 from left to right in an increasing order. The illustration is the selected voxels by our model applied to all data. The corresponding value of λ_2 and cross-validation accuracies are given in each subcaption.

desirable phenomenon. Moreover, the increased classification accuracy justifies that the graph-based cohesion behavior of GFL is consistent with the structure of critical voxels for AD. Otherwise, a balance between classification accuracy and that of voxel cohesion will have to be made, and the added voxels from one to another will end up being a more scattered pattern. As is also expected from a mathematical perspective, when too much emphasis is made on cohesion, useless or noisy voxels are probably selected and the performance begin to decrease, such as is shown in Figure 7(e) and (f).

5.2. Background Subtraction

Suppose that we are given a sequence of training video frames $\mathbf{X} \in \mathbb{R}^{N\times d}$ (d pixels) from a fixed camera and a test frame $\mathbf{y} \in \mathbb{R}^d$. We model $\mathbf{y}$ as a sparse linear combination of N training frames (the background model), plus an error term $\mathbf{e} \in \mathbb{R}^d$ (an additive foreground). We assume that $\mathbf{e}$ should be sparse and that the nonzero elements should be spatially (2D grid) connected. We formulate the BS problem as GFL in the following way:

$$\min_{\mathbf{a}\in\mathbb{R}^N \mathbf{e}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{a} - \mathbf{e}\|_2^2 + \lambda_1\|\mathbf{a}\|_1 + \lambda_2 \cdot \Omega_{gfl}(\mathbf{e}). \tag{17}$$

By defining $\mathbf{X}' = [\mathbf{X}, \mathbf{I}] \in \mathbb{R}^{N\times(d+N)}$, problem (17) can be viewed as an instance of problem (3). However, since the nonsmooth part of (17) is separable with respect to $\mathbf{a}$ and $\mathbf{e}$, we apply a different optimization method similar to that used in Mairal et al. [2011] by alternating the optimization with respect to $\mathbf{a}$ and $\mathbf{e}$. This method can be seen

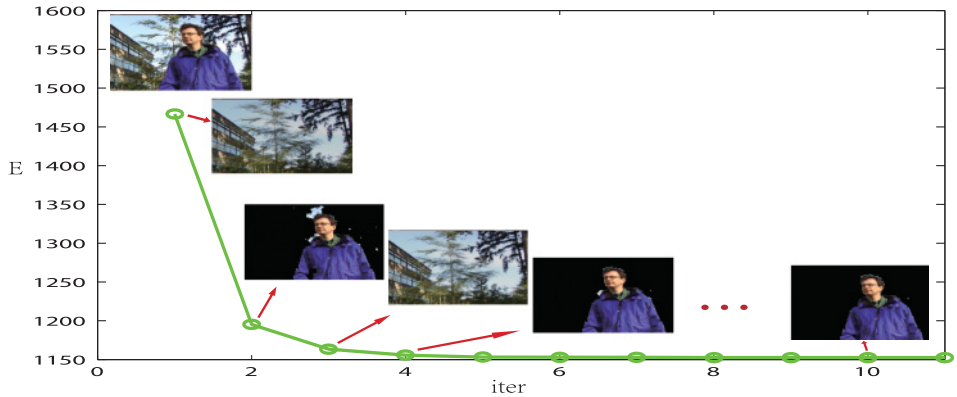


Fig. 8. An illustration of the alternating optimization of Algorithm 3. In each iteration (iter), either a (the background model) or e (the foreground) is updated, and the objective (E) is decreasing until convergence.

ALGORITHM 3: ADM Algorithm for Problem (17)

Input: $\lambda_1, \lambda_2 > 0$ and $\mathbf{y} \in \mathbb{R}^d$, $\mathbf{X} \in \mathbb{R}^{N \times d}$.
Output: $\mathbf{a}^* \in \mathbb{R}^N$ and $\mathbf{e}^* \in \mathbb{R}^d$ for problem (17).
Set $\mathbf{a}_0 = \mathbf{0}$, $\mathbf{e}_0 = \mathbf{0}$;
repeat
Call Algorithm 1 with $\lambda_2 = 0$ to solve

$$\mathbf{a}_{k+1} = \underset{\mathbf{a}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|(\mathbf{y} - \mathbf{e}_k) - \mathbf{X}\mathbf{a}\|_2^2 + \lambda_1 \|\mathbf{a}\|_1 \right\}.$$

Call Algorithm 2 to solve

$$\mathbf{e}_{k+1} = \underset{\mathbf{e}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|(\mathbf{y} - \mathbf{X}\mathbf{a}_k) - \mathbf{e}\|_2^2 + \lambda_2 \cdot \Omega_{gfl}(\mathbf{e}) \right\}.$$

Check stopping criteria, if true, return $\mathbf{a}_k$ and $\mathbf{e}_k$.
until *convergence*;

as a variant of the alternating direction methods (ADMs) [Boyd and Vandenberghe 2004]. This method is practically much faster, and the convergence is guaranteed, as (17) is convex with respect to the catenation of $\mathbf{a}$ and $\mathbf{e}$ (see Boyd and Vandenberghe [2004]). We summarize the ADM algorithm for problem (17) in Algorithm 3. In Figure 8, we illustrate the alternating effects and the fast convergence of this optimization via a practical example. In our experiments, the average number of iterations taken to converge is around 10.

We have shown that our algorithm also handles weighted GFL and, further, its directed graph variant. Therefore, we utilized the information from the image to adaptively adjust the strength of the fusion term. We define the weights by matrix $\mathbf{W}$ as follows:

$$\mathbf{w}_{ij}^{(1)} = \begin{cases} \exp \frac{-\|c(i)-c(j)\|_2^2}{2\sigma^2} & i, j \; connected, \\ 0 & otherwise, \end{cases}$$

where $c(i)$ is the color intensity of pixel i and $\sigma = 50$ is empirically set. As is discussed in Boykov and Funka-Lea [2006], in certain cases, directed costs can be applied to obtain more accurate object boundaries. Following this idea, we also define an asymmetric

Table IV. Results for the Wallflower Dataset, Given as the Number of Pixels That Have Been Mix Classified

Methods	MO	TD	LS	WT	CF	BS	FA
Frame difference	0	1358	2565	6789	10070	2175	4354
Mean+threshold	0	2593	16232	3285	1832	3236	2818
Block correlation	1200	1165	3802	3771	6670	2673	2402
Sta [Stauffer and Grimson 1999]	0	1028	15802	1664	3496	2091	2972
Oli [Oliver et al. 2000]	1065	895	1324	3084	1898	6433	2978
Toy [Toyama et al. 1999]	0	986	1322	2876	2935	2390	969
Hai [Haines and Xiang 2012]	0	**330**	3945	184	**384**	1236	1569
Can [Candès et al. 2011]	0	628	2016	1014		1465	2875
Jia [Xu et al. 2013]	0	912	1067	629		1779	1139
Ours	**0**	418	**686**	**166**		**795**	**192**

matrix $\mathbf{W}$:

$$\mathbf{w}_{ij}^{(2)} = \begin{cases} g(i)|sin(\theta(i))| & i \to j, \theta(i) \in \left(0, \frac{\pi}{2}\right) \cup \left(\frac{3\pi}{2}, 2\pi\right) \\ g(i)|cos(\theta(i))| & i \uparrow j, \theta(i) \in (0, \pi) \\ g(i)|sin(\theta(i))| & i \leftarrow j, \theta(i) \in \left(\frac{\pi}{2}, \frac{3\pi}{2}\right) \\ g(i)|cos(\theta(i))| & i \downarrow j, \theta(i) \in (\pi, 2\pi) \\ 0 & otherwise, \end{cases}$$

where $g(i)$, $\theta(i)$ are the magnitude and orientation (with respect to the x-axis) of the gradient of pixel i, and arrows illustrate the spatial relationship between adjacent pixels. We define $\mathbf{W} = \mathbf{W}^{(1)}$ and $\mathbf{W} = \eta\mathbf{W}^{(1)} + (1-\eta)\mathbf{W}^{(2)}$ for undirected and directed graph structures, respectively.

We apply GFL to data provided by two popular public BS datasets, namely the Wallflower dataset [Toyama et al. 1999] and the SABS dataset [Brutzer et al. 2011].

Wallflower. Wallflower is a pioneer dataset in the literature and is still actively used. This dataset provides manually labeled ground truth for natural video sequences of different challenges in the BS problem [Toyama et al. 1999]. In Table IV, we compare GFL to state-of-the-art models. Following the literature, we used "the number of pixels that have been mix classified" as an evaluation. These results demonstrate that GFL is a very competitive model for the BS problem. Specifically, on five out of seven sequences, GFL achieved the best performance, competing with both well-known, e.g., Stauffer and Grimson [1999] and Oliver et al. [2000], and very recent, e.g., Haines and Xiang [2012] and Xu et al. [2013], models. In practice, we notice that for the sequence of "CF," GFL performs very poorly. This is mainly because the foregrounds in CF occupied too many pixels, which violated our prior assumption of sparsity. The failure of RPCA ([Candès et al. 2011]) and the group lasso model ([Xu et al. 2013]) justifies this, where both models also assume sparse foregrounds. In Figure 9, we explicitly compare GFL to sparse models and illustrate our results on five out of seven sequences, where we also provide precision (p), recall (r), and F-score (F) as a supplemental evaluation. We see that the cut foreground by sparse models (l_1) have both much noise from the background and many holes in the foreground. On the other hand, by incorporating the fusion term, GFL cut much better in the foreground. From the comparison of the last two columns, we see that the directed graph sometimes results in more accurate object boundaries, although this is not always the case.

SABS. The SABS dataset provides artificial data and therefore high-quality ground truth [Brutzer et al. 2011]. Results of several state-of-the-art algorithms (including GMM, etc.) are provided on the related Web site. We apply GFL to the "Basic" setting and demonstrate the comparison with state of the arts in Table V. (The compared models are Mck [McKenna et al. 2000], Kim [Kim et al. 2004], McF [McFarlane and Schofield 1995], Oli [Oliver et al. 2000], Mad [Maddalena and Petrosino 2008], Bar

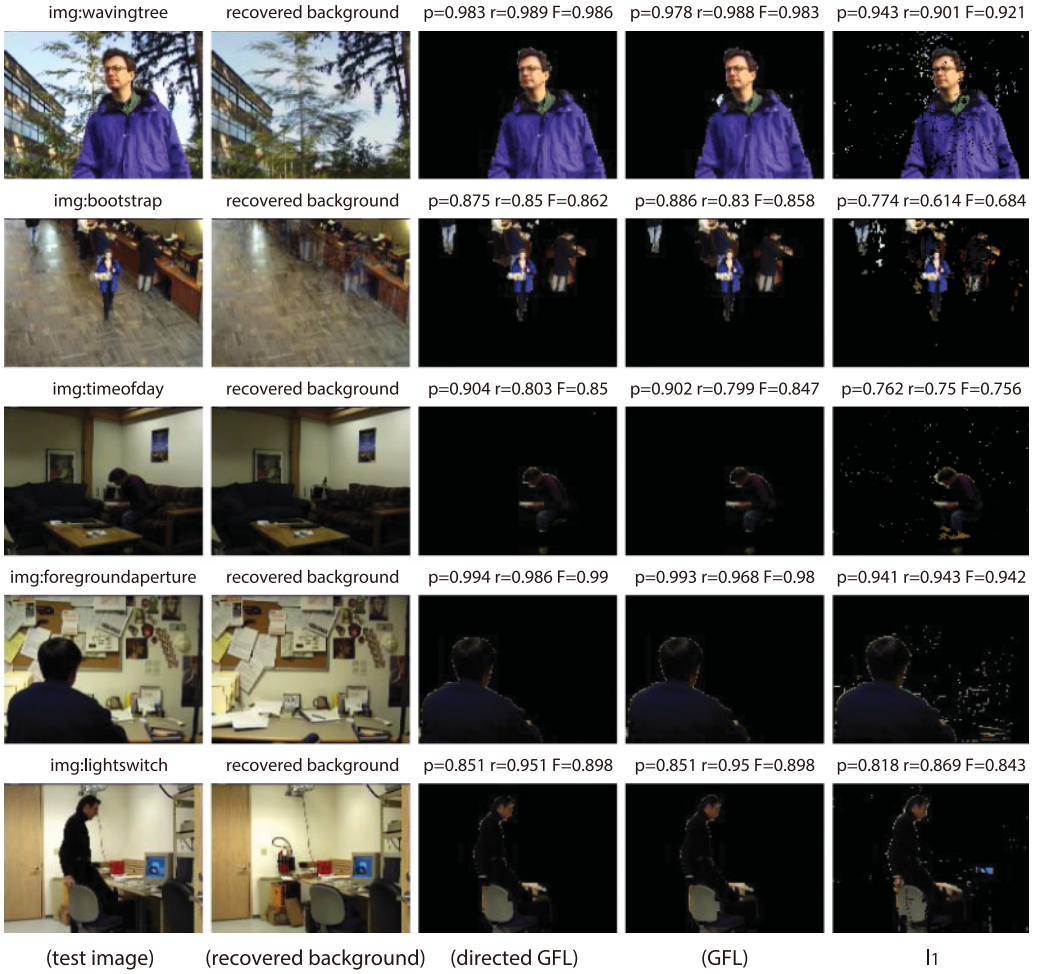


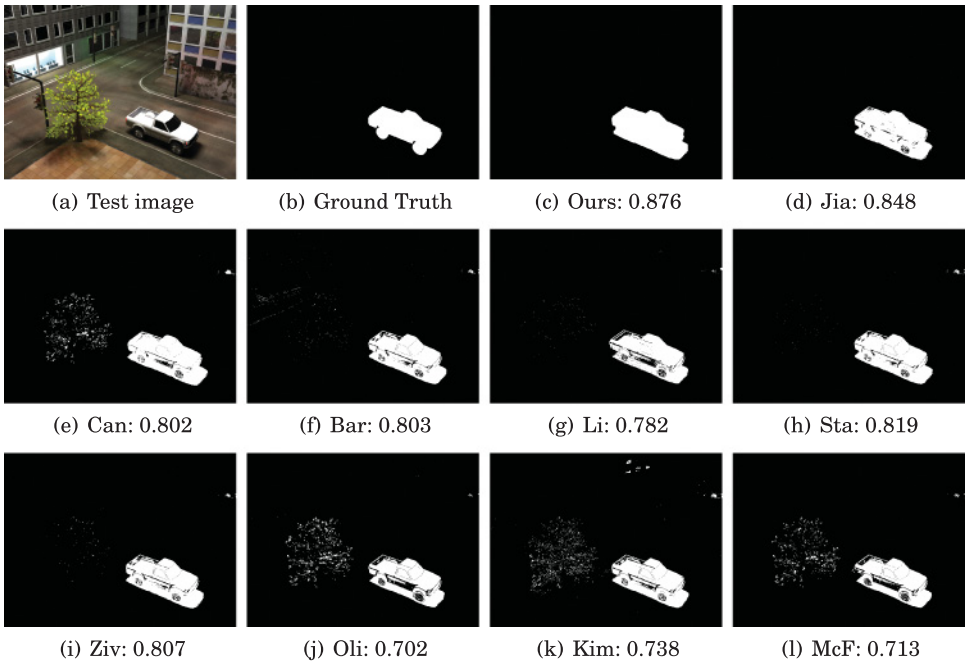
Fig. 9. BS results on the Wallflower dataset. From left to right: Test image, recovered background by directed GFL, BS by directed GFL, GFL, and l_1.

Table V. Results for the SABS Dataset, Given as F-Score

Mck	Kim	McF	Oli	Mad	Bar	Ziv	Sta	Li	Can	Jia	**Ours**
.3806	.5601	.5887	.5891	.6672	.7177	.7232	.7284	.7457	.6483	0.7326	**.7775**

[Barnich and Van Droogenbroeck 2011], Ziv [Zivkovic and van der Heijden 2006], Sta [Stauffer and Grimson 1999], Li [Li et al. 2004], Can [Candès et al. 2011], and Jia [Xu et al. 2013].) One example frame (448) is illustrated in Figure 10. GFL almost cut a perfect foreground as well as with its shadow. Notice that in the provided ground truth, the shadow is not included, which makes the precision value (thereafter, F-score) relatively low. However, this definition of foreground can be controversial depending on the actual situations. Nevertheless, GFL outperformed all other state-of-the-art BS methods on tested images. More results can be found in the Appendix and from our Web site.[6] Recall that in Xu et al. [2013] and Mairal et al. [2011], group structures such

[6]http://www.idm.pku.edu.cn/staff/wangyizhou/demo/BackgroundSubtraction/tist-ideo.mpg.

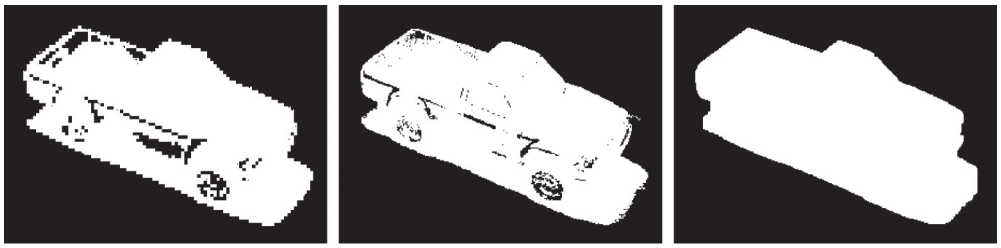


(a) Test image (b) Ground Truth (c) Ours: 0.876 (d) Jia: 0.848

(e) Can: 0.802 (f) Bar: 0.803 (g) Li: 0.782 (h) Sta: 0.819

(i) Ziv: 0.807 (j) Oli: 0.702 (k) Kim: 0.738 (l) McF: 0.713

Fig. 10. Results on the SABS dataset. F-scores are shown.



(a) 3×3 Block group lasso (b) Coarse-to-fine superpixel group lasso (c) GFL

Fig. 11. Comparison of different foreground regularizations.

as "3×3 blocks group" and "coarse-to-fine superpixel group" was used to model the foregrounds. However, in practice, these sophisticated groupings are still not flexible or trustworthy for representing arbitrary foregrounds (see Figure 11 for an example).

Recall that in Equation (17), the GFL assumption is made for the "additive foreground" only, and therefore background models other than "the sparse linear combination of training frames" can also be enhanced by GFL. With simple modification, we believe that most state-of-the-art background models can be improved by incorporating GFL.

6. CONCLUSION

In this study, we proposed an efficient and scalable algorithm for GFL. We demonstrated that the proposed algorithm runs significantly faster than existing algorithms. By exploiting the efficiency and scalability of the proposed algorithm, we formulated

both the diagnosis of AD and video BS problems as GFL. Our evaluations showed that for both problems, GFL achieved state-of-the-art performance. Whereas existing algorithms do not scale up to the dimensionality of these problems, the proposed algorithm solves both problems in feasible time. Note that the proposed algorithm solves GFL with arbitrary convex loss terms and general (directed) graph structures; these properties largely increase the usability of GFL in practice. Therefore, future extensions of the framework to other applications would be interesting.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. 2006. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing* 54, 11, 4311–4322.

John Ashburner. 2007. A fast diffeomorphic image registration algorithm. *Neuroimage* 38, 1, 95–113.

F. Bach. 2010. Structured sparsity-inducing norms through submodular functions. In *Advances in Neural Information Processing Systems (NIPS'10)*. Vol. 23. 118–126.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. 2012. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning* 4, 1, 1–106.

Olivier Barnich and Marc Van Droogenbroeck. 2011. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing* 20, 6, 1709–1724.

A. Beck and M. Teboulle. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* 2, 1, 183–202.

Stephen Poythress Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Yuri Boykov and Gareth Funka-Lea. 2006. Graph cuts and efficient ND image segmentation. *International Journal of Computer Vision* 70, 2, 109–131.

Sebastian Brutzer, Benjamin Hoferlin, and Gunther Heidemann. 2011. Evaluation of background subtraction techniques for video surveillance. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. IEEE, Los Alamitos, CA, 1937–1944.

Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *Journal of the ACM* 58, 3, 11.

Emmanuel J. Candès, Justin Romberg, and Terence Tao. 2006. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory* 52, 2, 489–509.

Antonin Chambolle and Jérôme Darbon. 2009. On total variation minimization and surface evolution using parametric maximum flows. *International Journal of Computer Vision* 84, 3, 288–307.

Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. 1998. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing* 20, 1, 33–61.

Bo Cheng, Daoqiang Zhang, and Dinggang Shen. 2012. Domain transfer learning for MCI conversion prediction. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2012*. Springer, 82–90.

Carlton Chu, Ai-Ling Hsu, Kun-Hsien Chou, Peter Bandettini, and ChingPo Lin. 2012. Does feature selection improve classification accuracy? Impact of sample size and feature selection on classification using anatomical magnetic resonance images. *Neuroimage* 60, 1, 59–70.

Zhengjia Dai, Chaogan Yan, Zhiqun Wang, Jinhui Wang, Mingrui Xia, Kuncheng Li, and Yong He. 2012. Discriminative analysis of early Alzheimer's disease using multi-modal imaging and multi-level characterization with multi-classifier (M3). *Neuroimage* 59, 3, 2187–2195.

J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. 2007. Pathwise coordinate optimization. *Annals of Applied Statistics* 1, 2, 302–332.

S. Fujishige. 2005. *Submodular Functions and Optimization* (2nd ed.). Elsevier.

S. Fujishige, T. Hayashi, and S. Isotani. 2006. *The Minimum-Norm-Point Algorithm Applied to Submodular Function Minimization and Linear Programming*. Technical Report RIMS-1571. Research Institute for Mathematical Sciences, Kyoto University.

G. Gallo, M. D. Grigoriadis, and R. E. Tarja. 1989. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing* 18, 1, 30–55.

P. E. Gill, W. Murray, and M. A. Saunders. 1999. *User's Guide for SNOPT 5.3: A Fortran Package for Large-scale Nonlinear Programming*. Technical Report. University of California, San Diego.

Donald Goldfarb and Wotao Yin. 2009. Parametric maximum flow algorithms for fast total variation minimization. *SIAM Journal on Scientific Computing* 31, 5, 3712–3743.

Michael Grant, Stephen Boyd, and Yinyu Ye. 2008. CVX: Matlab Software for Disciplined Convex Programming. Retrieved March 12, 2016, from http://cvxr.com/cvx/.

Tom S. F. Haines and Tao Xiang. 2012. Background subtraction with Dirichlet processes. In *Computer Vision—ECCV 2012*. Springer, 99–113.

Sonsoles Herrero and Jesús Bescós. 2009. Background subtraction techniques: Systematic evaluation and comparative analysis. In *Advanced Concepts for Intelligent Vision Systems*. Springer, 33–42.

J. Huang, T. Zhang, and D. Metaxas. 2011. Learning with structured sparsity. *Journal of Machine Learning Research* 12, 3371–3412.

Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood, and Larry Davis. 2004. Background modeling and subtraction by codebook construction. In *Proceedings of the 2004 International Conference on Image Processing (ICIP'04)*, Vol. 5. IEEE, Los Alamitos, CA, 3061–3064.

V. Kolmogorov and R. Zabin. 2004. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26, 2, 147–159.

Liyuan Li, Weimin Huang, Irene Yu-Hua Gu, and Qi Tian. 2004. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing* 13, 11, 1459–1472.

J. Liu, S. Ji, and J. Ye. 2009. SLEP: Sparse Learning with Efficient Projections. Retrieved March 12, 2016, from http://www.yelab.net/software/SLEP/.

J. Liu, L. Yuan, and J. Ye. 2010. An efficient algorithm for a class of fused lasso problems. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 323–332.

Lucia Maddalena and Alfredo Petrosino. 2008. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing* 17, 7, 1168–1177.

Julien Mairal, Rodolphe Jenatton, Guillaume Obozinski, and Francis Bach. 2011. Convex and network flow optimization for structured sparsity. *Journal of Machine Learning Research* 12, 2681–2720.

Nigel J. B. McFarlane and C. Paddy Schofield. 1995. Segmentation and tracking of piglets in images. *Machine Vision and Applications* 8, 3, 187–193.

Stephen J. McKenna, Sumer Jabri, Zoran Duric, Azriel Rosenfeld, and Harry Wechsler. 2000. Tracking groups of people. *Computer Vision and Image Understanding* 80, 1, 42–56.

K. Nagano and K. Aihara. 2012. Equivalent of convex minimization problems over base polytopes. *Japan Journal of Industrial and Applied Mathematics* 29, 519–534.

A. S. Nemirovsky and D. B. Yudin. 1983. *Problem Complexity and Method Efficiency in Optimization*. John Wiley & Sons.

Y. Nesterov. 2004. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer.

Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland. 2000. A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 8, 831–843.

J. B. Orlin. 2009. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming* 118, 237–251.

B. T. Polëìak. 1987. *Introduction to Optimization*. Optimization Software, Publications Division, New York, NY.

Chris Stauffer and W. Eric L. Grimson. 1999. Adaptive background mixture models for real-time tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2. IEEE, Los Alamitos, CA.

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 58, 1, 267–288.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. 2005. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B* 67, 1, 91–108.

R. J. Tibshirani and J. Taylor. 2011. The solution path of the generalized lasso. *Annals of Statistics* 39, 3, 1335–1371.

Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. 1999. Wallflower: Principles and practice of background maintenance. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, Vol. 1. IEEE, Los Alamitos, CA, 255–261.

Jing Wan, Zhilin Zhang, Jingwen Yan, Taiyong Li, Bhaskar D. Rao, Shiaofen Fang, Sungeun Kim, Shannon L. Risacher, Andrew J. Saykin, and Li Shen. 2012. Sparse Bayesian multi-task learning for predicting

cognitive outcomes from neuroimaging measures in Alzheimer's disease. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12)*. IEEE, Los Alamitos, CA, 940–947.

John Wright, Allen Y. Yang, Arvind Ganesh, Shankar S. Sastry, and Yi Ma. 2009. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 2, 210–227.

Bo Xin, Lingjing Hu, Yizhou Wang, and Wen Gao. 2015a. Stable feature selection from brain sMRI. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

Bo Xin, Yoshinobu Kawahara, Yizhou Wang, and Wen Gao. 2014. Efficient generalized fused lasso and its application to the diagnosis of Alzheimers disease. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 2163–2169.

Bo Xin, Yuan Tian, Yizhou Wang, and Wen Gao. 2015b. Background subtraction via generalized fused lasso foreground modeling. arXiv:1504.03707.

Jia Xu, Vamsi K. Ithapu, Lopamudra Mukherjee, James M. Rehg, and Vikas Singh. 2013. GOSUS: Grassmannian online subspace updates with structured-sparsity. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'13)*.

Jiayu Zhou, Lei Yuan, Jun Liu, and Jieping Ye. 2011. A multi-task learning formulation for predicting disease progression. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 814–822.

Zoran Zivkovic and Ferdinand van der Heijden. 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters* 27, 7, 773–780.

Received December 2014; revised July 2015; accepted November 2015