2013 IEEE 13th International Conference on Data Mining

Efficient Learning for Models with DAG-Structured Parameter Constraints

Leon Wenliang Zhong James T. Kwok
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
{*wzhong,jamesk*}*@cse.ust.hk*

***Abstract*—In high-dimensional models, hierarchical and structural relationships among features are often used to constrain the search for the more important interactions. These relationships may come from prior knowledge or traditional design principles, such as that low-order effects should have larger contributions than higher-order ones and should be included into the model earlier. However, these structural constraints also make the optimization problem more challenging. In this paper, we propose the use of the alternating direction method of multipliers (ADMM) and accelerated gradient methods. In particular, we show that ADMM can be used to either directly solve the problem or serve as a key building block. Experimental results on a number of synthetic and real-world data sets demonstrate that the proposed algorithm is efficient and flexible. Moreover, the use of the hierarchical relationships consistently improves generalization performance and parameter estimation.**

***Keywords*-Structural sparsity, Heredity, Alternating direction method of multipliers, Accelerated gradient methods**

I. Introduction

The linear model $f(\mathbf{x}) = \theta_0 + \sum_{j=1}^{d} x_j\theta_j$ has been widely used in classification and regression. Here, $\mathbf{x} = [x_1, \ldots, x_d]^T$ with $x_1, \ldots, x_d$ being the input variables (also called *main effects*), and θ_j's are the corresponding coefficients. Despite its popularity, this simple model does not capture the interactions that may exist among the inputs. To alleviate this problem, one often extends the model by incorporating all possible pairwise interactions (or effects), leading to

$$f(\mathbf{x}) = \theta_0 + \sum_{j=1}^{d} x_j\theta_j + \frac{1}{2}\sum_{j\neq k} x_j x_k \Theta_{jk}, \qquad (1)$$

where Θ_{jk} is the contribution from the pairwise interaction between x_j and x_k. More generally, higher-order interactions can be similarly incorporated.

However, this myriad of main effects and interactions may make the model very complicated, especially when d is large. Prior knowledge can be used to guide the search for important components. For example, in molecular genetics, it is estimated that about $5 - 10\%$ of the human genes contribute to oncogensis [1]. In training a cancer prediction model, it is thus reasonable to assume that the known cancer genes (also called *susceptibility genes*) should have larger contributions than the other genes. For factorial design and data analysis with interaction models, two design principles, namely *effect heredity* and *hierarchical ordering*, have also been commonly used to locate the more important interactions.

The *effect heredity* (or *heredity*) principle states that low-order effects should be included into the model before higher-order ones [2], [3], [4]. These dependencies among effects can be represented by a directed acyclic graph (DAG) $G(V, E)$, where V (resp. E) is the set of nodes (resp. edges). Each node in V corresponds to an effect, and there is an edge $e_{ab} \in E$ from $a \in V$ to $b \in V$ if and only if b depends on a. An example is shown in Figure 1. The heredity dependency comes in two popular flavors: (i) *strong heredity*, in which an effect can be included only after all its parents have been included; and (ii) *weak heredity*, in which an effect can be included if at least one of its parents has been included. For the model in (1), these imply

$$\begin{array}{ll} \text{(strong heredity)} & \Theta_{jk} \neq 0 \Rightarrow \theta_j \neq 0 \text{ and } \theta_k \neq 0, \\ \text{(weak heredity)} & \Theta_{jk} \neq 0 \Rightarrow \theta_j \neq 0 \text{ or } \theta_k \neq 0. \end{array} \qquad (2)$$

In general, heredity allows the model to have better physical interpretation. In particular, strong heredity, which will be the focus in this paper, also ensures the model to be invariant to linear transformations of the inputs. In a large meta-analysis of 113 data sets, Li *et. al.* [5] observed that the data strongly support heredity (especially strong heredity).

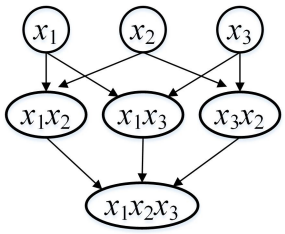


Figure 1. Example with three input variables x_1, x_2, x_3, and the two-factor, three-factor interactions.

The second, closely related, principle is *hierarchical ordering* (or *hierarchy*) [2]. It states that low-order effects

1550-4786/13 $31.00 © 2013 IEEE
DOI 10.1109/ICDM.2013.123

are more important than high-order ones. Specifically, the main effects are usually larger than two-factor interactions, which in turn are larger than three-factor interactions, and so on. In the DAG induced by heredity above, let w_a, w_b be coefficients corresponding to the nodes a and b. The hierarchy principle then implies the constraints

$$|w_a| \geq |w_b| \;\text{ if } e_{ab} \in E. \tag{3}$$

Again, this is also empirically supported by the meta-analysis in [5].

While the standard linear model can be trained with a variety of loss functions and regularizers [6], learning becomes more challenging with the additional constraints imposed by prior knowledge and/or heredity/hierarchy principles. For example, previous attempts are limited to ℓ_1-regularization. Bien *et. al.* [4] proposed two lasso-like procedures (called *strong/weak hierarchical lasso*) corresponding to strong/weak heredity. In particular, it replaces (2) with a convex relaxation of

$$\|\boldsymbol{\Theta}_j\|_1 \leq |\theta_j|, \;\; \text{for } j = 1, \ldots, d, \tag{4}$$

where $\|\boldsymbol{\Theta}_j\|_1 = \sum_{k=1}^{d} |\Theta_{jk}|$. Alternatively, by assuming that the regression coefficients are all non-negative, and that the heredity-induced DAG is indeed a tree, Liu *et. al.* [7] developed a novel solver called Atda for this restricted setup. A central operation in Atda is the following projection step:

$$\min_{\mathbf{w} \in \mathbb{R}^n_+} \; \frac{1}{2}\|\mathbf{w} - \mathbf{u}\|^2 \;:\; w_a \geq w_b \text{ if } e_{ab} \in E. \tag{5}$$

for some given $\mathbf{u}$. The efficiency of Atda stems from the observation that (5) admits an analytic solution.

While the existing approaches are limited to the ℓ_1-regularizer (and with further restrictions on the tree for [7]), we propose in this paper three solvers that can be used with a variety of structured sparsity regularizers Ω and with constraints ordered on a general DAG. The first solver is based on a combination of the *accelerated gradient method* FISTA [8] and the simple but powerful *alternating direction method of multipliers* (ADMM) [9]; while the other two solvers are based purely on ADMM. Empirically, the FISTA-ADMM hybrid is the most efficient and outperforms existing solvers.

The rest of this paper is organized as follows. Section II gives a brief review on the accelerated gradient method and ADMM. Sections III and IV then present the various proposed solvers. Experimental results are presented in Section V, and the last section gives some concluding remarks.

II. Related Work

A. Accelerated Gradient Methods

Accelerated gradient methods [8], [10] have been widely used for composite optimization [6], [11] of the form

$$\min_{\mathbf{w} \in \mathcal{W}} \ell(\mathbf{w}) + \Omega(\mathbf{w}), \tag{6}$$

where $\mathcal{W}$ is the domain, ℓ is a convex and smooth function, while Ω is also convex but possibly nonsmooth. In the tth iteration, it computes the following *proximal* step:

$$\min_{\mathbf{w} \in \mathcal{W}} \; (\mathbf{w} - \hat{\mathbf{w}}^{(t)})^T \nabla \ell(\hat{\mathbf{w}}^{(t)}) + \frac{L}{2}\|\mathbf{w} - \hat{\mathbf{w}}^{(t)}\|^2 + \Omega(\mathbf{w}), \tag{7}$$

where L is the Lipschitz constant[1] of $\nabla \ell$. The objective in (7) is a linear approximation of the smooth component $\ell(\mathbf{w})$ at some $\hat{\mathbf{w}}^{(t)}$, while leaving the nonsmooth component $\Omega(\mathbf{w})$ intact. Without loss of generality, we assume that $L = 1$ by absorbing L into $\Omega(\mathbf{w})$ with an appropriate scaling. It is well-known that (7) can then be rewritten as

$$\min_{\mathbf{w} \in \mathcal{W}} \; \frac{1}{2}\|\mathbf{w} - \mathbf{u}\|^2 + \Omega(\mathbf{w}), \tag{8}$$

where $\mathbf{u} = \mathbf{w}^{(t)} - \nabla \ell(\hat{\mathbf{w}}^{(t)})$.

In this paper, we adopt a popular accelerated gradient algorithm called FISTA (Algorithm 1) [8]. While standard gradient methods have a slow convergence rate of $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, where T is the number of iterations, FISTA is more advantageous in that it converges as $\mathcal{O}\left(\frac{1}{T^2}\right)$. However, for FISTA to be useful, the proximal step has to be solved efficiently. This is the case, for example, when $\mathcal{W} = \mathbb{R}^n$ and Ω is "simple" (e.g., $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$, $\|\mathbf{w}\|_2^2$, $\|\mathbf{w}\|_\infty$ and some mixed norms) [12].

Algorithm 1 The FISTA algorithm [8].

1: **Initialize:** $\hat{\mathbf{w}}^1 \leftarrow \mathbf{w}^0$, $\tau_1 \leftarrow 1, t \leftarrow 1$.
2: **repeat**
3: update $\mathbf{w}^{(t)}$ as in (8);
4: $\tau_{t+1} \leftarrow \frac{1+\sqrt{1+4\tau_t^2}}{2}$;
5: $\hat{\mathbf{w}}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \left(\frac{\tau_t - 1}{\tau_{t+1}}\right)(\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)})$;
6: $t \leftarrow t + 1$.
7: **until** convergence.
8: Output $\mathbf{w}^t$.

B. ADMM

In recent years, ADMM has been popularly used in diverse fields such as machine learning, data mining and image processing [9]. It can be used to solve optimization problems of the form

$$\min_{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2} \; \phi(\boldsymbol{\omega}_1) + \psi(\boldsymbol{\omega}_2) \;:\; \mathbf{A}\boldsymbol{\omega}_1 + \mathbf{B}\boldsymbol{\omega}_2 = \mathbf{c},$$

where ϕ, ψ are convex functions, and $\mathbf{A}, \mathbf{B}$ (resp. $\mathbf{c}$) are constant matrices (resp. vector) of appropriate sizes. As in the method of multipliers, ADMM considers the augmented Lagrangian

$$\begin{aligned}\mathcal{L}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\nu}) \;\; = \;\; & \phi(\boldsymbol{\omega}_1) + \psi(\boldsymbol{\omega}_2) + \boldsymbol{\nu}^T(\mathbf{A}\boldsymbol{\omega}_1 + \mathbf{B}\boldsymbol{\omega}_2 - \mathbf{c}) \\ & + \frac{\rho}{2}\|\mathbf{A}\boldsymbol{\omega}_1 + \mathbf{B}\boldsymbol{\omega}_2 - \mathbf{c}\|^2,\end{aligned}$$

[1] In other words, $\|\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)\| \leq L\|\mathbf{w}_1 - \mathbf{w}_2\|$ for every $\mathbf{w}_1, \mathbf{w}_2$.

where $\boldsymbol{\nu}$ is the vector of Lagrangian multipliers, and $\rho > 0$ is a penalty parameter. At the tth iteration, the values of $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ (denoted $\boldsymbol{\omega}_1^{(t)}$ and $\boldsymbol{\omega}_2^{(t)}$) are updated by minimizing $\mathcal{L}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\nu})$ w.r.t. $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ in an alternating manner. This allows ADMM to more easily decompose the optimization problem when ϕ and ψ are separable. Using the scaled dual variable $\mathbf{r} = \boldsymbol{\nu}/\rho$, the ADMM update can be expressed as:

$$\boldsymbol{\omega}_1^{(t+1)} = \arg\min_{\boldsymbol{\omega}_1} \phi(\boldsymbol{\omega}_1) + \frac{\rho}{2}\|\mathbf{A}\boldsymbol{\omega}_1 + \mathbf{B}\boldsymbol{\omega}_2^{(t)} - \mathbf{c} + \mathbf{r}^{(t)}\|^2, \quad (9)$$

$$\boldsymbol{\omega}_2^{(t+1)} = \arg\min_{\boldsymbol{\omega}_2} \psi(\boldsymbol{\omega}_2) + \frac{\rho}{2}\|\mathbf{A}\boldsymbol{\omega}_1^{(t+1)} + \mathbf{B}\boldsymbol{\omega}_2 - \mathbf{c} + \mathbf{r}^{(t)}\|^2, \quad (10)$$

$$\mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} + \mathbf{A}\boldsymbol{\omega}_1^{(t+1)} + \mathbf{B}\boldsymbol{\omega}_2^{(t+1)} - \mathbf{c},$$

In case (9) is difficult to solve, one may replace $\phi(\boldsymbol{\omega}_1)$ with its first-order approximation at $\boldsymbol{\omega}_1^{(t)}$. The update rule then becomes

$$\boldsymbol{\omega}_1^{(t+1)} = \arg\min_{\boldsymbol{\omega}_1} \nabla\phi(\boldsymbol{\omega}_1^{(t)})^T(\boldsymbol{\omega}_1 - \boldsymbol{\omega}_1^{(t)}) + \frac{\tilde{L}}{2}\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_1^{(t)}\|^2 + \frac{\rho}{2}\|\mathbf{A}\boldsymbol{\omega}_1 + \mathbf{B}\boldsymbol{\omega}_2^{(t)} - \mathbf{c} + \mathbf{r}^{(t)}\|^2, \quad (11)$$

where $\nabla\phi(\boldsymbol{\omega}_1^{(t)})$ is the gradient of ϕ at $\boldsymbol{\omega}_1^{(t)}$, and $\tilde{L}$ is the Lipschitz parameter of $\nabla\phi$. This linearized version has been recently studied in [13], [14].

Sometimes, (10) is also difficult to solve. In this case, one can replace $\frac{\rho}{2}\|\mathbf{A}\boldsymbol{\omega}_1^{(t+1)} + \mathbf{B}\boldsymbol{\omega}_2 - \mathbf{c} + \mathbf{r}^{(t)}\|^2$ by its Taylor approximation at the current iterate $\boldsymbol{\omega}_2^{(t)}$. This is called the split inexact Uzawa method [15] (or "prox-linear" in [16]), and has been very useful in image restoration problems [17]. It can be shown that (10) then becomes

$$\boldsymbol{\omega}_2^{(t+1)} = \arg\min_{\boldsymbol{\omega}_2} \psi(\boldsymbol{\omega}_2) + \frac{\rho\lambda_{\max}(\mathbf{B}^T\mathbf{B})}{2}\|\boldsymbol{\omega}_2 - \tilde{\boldsymbol{\omega}}_2^{(t)}\|^2, \quad (12)$$

where $\lambda_{\max}(\mathbf{B}^T\mathbf{B})$ is the largest eigenvalue of $\mathbf{B}^T\mathbf{B}$, and

$$\tilde{\boldsymbol{\omega}}_2^{(t)} = \boldsymbol{\omega}_2^{(t)} - \frac{\mathbf{B}^T(\mathbf{A}\boldsymbol{\omega}_1^{(t+1)} + \mathbf{B}\boldsymbol{\omega}_2^{(t)} - \mathbf{c} + \mathbf{r}^{(t)})}{\lambda_{\max}(\mathbf{B}^T\mathbf{B})}. \quad (13)$$

III. Algorithm Based on FISTA and ADMM

Let $\mathbf{w} \in \mathbb{R}^n$ be the vector that concatenates all the regression coefficients (θ_j's and Θ_{jk}'s) together. With the use of a convex and smooth loss function l, a convex but possibly nonsmooth regularizer Ω, and a set of samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, the training process involves the following composite optimization problem:

$$\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{N}\sum_{i=1}^{N} l(\mathbf{w}^T\mathbf{x}_i, y_i) + \Omega(\mathbf{w}), \quad (14)$$

where $\mathcal{W} \subseteq \mathbb{R}^n$ is the feasible region on $\mathbf{w}$ as defined by prior knowledge, heredity and/or hierarchy principles. In the sequel, given a DAG $G(V, E)$ that encodes the structural relationship among the elements of $\mathbf{w}$, we consider $\mathcal{W}$ of the form

$$\mathcal{W} = \left\{\mathbf{w} = [w_a] \in \mathbb{R}^n : \begin{array}{l} w_a \geq w_b \text{ if } e_{ab} \in E \\ w_a \in [B_1, B_2] \end{array}\right\}, \quad (15)$$

where $B_1 < B_2 \in \mathbb{R}$. Note that problem (14) is of the form in (6) (with $\ell(\mathbf{w}) \equiv \frac{1}{N}\sum_{i=1}^{N} l(f(\mathbf{x}_i), y_i)$) and hence can be solved by FISTA. As discussed in Section II-A, it is important to ensure that the proximal step in (8) can be efficiently computed.

In Sections III-A and III-B, we first show that the form in (15) is quite general. In particular, for the feasible region defined by hierarchy constraints in (3), the corresponding proximal step

$$\begin{array}{ll} \min_{\mathbf{w}\in\mathbb{R}^n} & \frac{1}{2}\|\mathbf{w} - \mathbf{u}\|^2 + \Omega(\mathbf{w}) \\ \text{s.t.} & |w_a| \geq |w_b| \text{ if } e_{ab} \in E, \end{array} \quad (16)$$

can be reduced to a proximal step (8) with the $\mathcal{W}$ in (15). The same also holds when the feasible region is defined by heredity constraints. Then, in Section III-C, we show that such a flexible $\mathcal{W}$ still allows the proximal step to be efficiently computed.

A. Proximal Step with Hierarchy Constraints (3)

Assume that $\Omega(\mathbf{w})$ in (16) is invariant to the sign of $\mathbf{w} = [w_a]$, i.e., $\Omega(\mathbf{w}) = \Omega(|\mathbf{w}|)$, where $|\mathbf{w}|$ is the vector with entries $|w_a|$'s. Many popular regularizers, such as $\|\mathbf{w}\|_1, \|\mathbf{w}\|^2$ and $\|\mathbf{w}\|_\infty$, satisfy this condition. As the constraints in (16) are also invariant to the sign, the optimal w_a must have the same sign as u_a (for $a = 1, \ldots, n$). Hence, we can first solve

$$\begin{array}{ll} \min_{\tilde{\mathbf{w}}\in\mathbb{R}^n} & \frac{1}{2}\sum_a (\tilde{w}_a - |u_a|)^2 + \Omega(\tilde{\mathbf{w}}) \\ \text{s.t.} & \tilde{w}_a \geq \tilde{w}_b \text{ if } e_{ab} \in E, \\ & \tilde{w}_a \geq 0, \;\; a = 1, \ldots, n, \end{array} \quad (17)$$

and then recover w_a as $\tilde{w}_a \text{sign}(u_a)$. Obviously, (17) is a proximal step and its feasible region is of the form in (15).

B. Proximal Step with Heredity Constraints

In the sequel, we focus on model (1) with pairwise interactions. Extension to models with higher-order interactions is straightforward. Recall that Bien *et al.* [4] enforces the heredity principle by using the constraints in (4). Here, instead of aggregating the interactions $\{|\Theta_{jk}|\}_{k=1}^d$ together into $\|\boldsymbol{\Theta}_j\|_1$, we enforce strong heredity by the constraints

$$|\Theta_{jk}| \leq |\theta_j|, \;\; |\Theta_{jk}| \leq |\theta_k|, \quad (18)$$

which are more direct and intuitive. Bien *et al.* also requires an additional constraint $\Theta = \Theta^T$ for strong heredity. We avoid this by simply redefining the model in (1) as $f(\mathbf{x}) = \theta_0 + \sum_{j=1}^d x_j\theta_j + \sum_{j<k} x_j x_k \Theta_{jk}$. With the feasible region defined by (18), the proximal step is then

$$\begin{array}{ll} \min_{\mathbf{w}=[\boldsymbol{\theta}^T, \boldsymbol{\Theta}^T]^T\in\mathbb{R}^n} & \frac{1}{2}\|\mathbf{w} - \mathbf{u}\|^2 + \Omega(\mathbf{w}) \\ \text{s.t.} & |\theta_j| \geq |\Theta_{jk}|, |\theta_k| \geq |\Theta_{jk}|. \end{array} \quad (19)$$

As in Section III-A, the absolute signs in (19) can be removed by reformulation, and the proximal step reduces to one with the feasible region defined in (15).

Algorithm 2 The RIA (Regularized Isotonic Regression with ADMM) algorithm for computing the proximal step with $\mathcal{W}$ in (15).

convert problem (14) to (21);
$t \leftarrow 0$; set $\mathbf{z}^0, \mathbf{w}^0, \mathbf{r}^0 \leftarrow \mathbf{0}$;
repeat
  $\mathbf{z}^{(t+1)} \leftarrow \arg\min_{\mathbf{z}\in\mathcal{Z}} \phi(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{z} - \mathbf{Q}\mathbf{w}^{(t)} + \mathbf{r}^{(t)}\|^2$, where the elements of $\mathbf{Q}$ are as defined in (20);
  $\mathbf{w}^{(t+1)} \leftarrow \arg\min_{\mathbf{w}\in\mathbb{R}^n} \psi(\mathbf{w}) + \frac{\rho}{2}\|\mathbf{z}^{(t+1)} - \mathbf{Q}\mathbf{w} + \mathbf{r}^{(t)}\|^2$;
  $\mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} + \mathbf{z}^{(t+1)} - \mathbf{Q}\mathbf{w}^{(t+1)}$;
  $t \leftarrow t + 1$;
until convergence;
return $\mathbf{w}^{(t)}$.

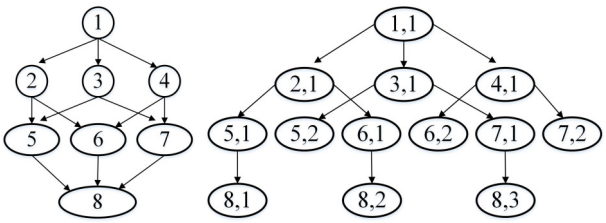
C. Computing the Proximal Step with $\mathcal{W}$ in (15)

When $\mathcal{W} = \mathbb{R}^n$ (i.e., no constraints are imposed on $\mathbf{w}$), efficient computation of the proximal step has been well-studied [12]. However, this becomes more challenging when constraints are present. In particular, $\mathcal{W}$ contains two types of constraints: (i) simple box constraints; and (ii) pairwise constraints $w_a \geq w_b$ that are defined w.r.t. a DAG. When $B_1 = 0$ and $B_2 = \infty$, $\mathcal{W}$ reduces to that in (5). When $\Omega(\mathbf{w}) = 0$, the proximal step, with $\mathcal{W}$ defined in (15), is a *(bounded) isotonic regression* problem [18] (the constraints $\{w_a \geq w_b\}$ are called *isotonic constraints*), and a number of solvers have been proposed [19]. However, the more interesting case with $\Omega(\mathbf{w}) \neq 0$ is much more difficult.

In this section, we propose a novel algorithm for solving this proximal step. We first convert the DAG (on which the pairwise constraints in $\mathcal{W}$ reside) to a tree, and then use the ADMM algorithm on the transformed problem. As reviewed in Section II-B, the key issue in ADMM is how to update the two variables $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$. We will show that one of these can be reduced to a standard isotonic regression problem, while the other can be reduced to a standard proximal step. The whole procedure, called RIA (Regularized Isotonic Regression with ADMM), is shown in Algorithm 2.

1) Convert the DAG to a Tree: Without loss of generality, we assume that the DAG $G(V, E)$ has a single root. Otherwise (as in Figure 1), we add a pseudo-root and connect it to all the original roots (Figure 2(a)). The conversion procedure is simple. It checks every node $v_a \in V$. If its number of parents $n_{\text{par}}(a)$ is greater than 1, v_a is duplicated $(n_{\text{par}}(a) - 1)$ times and edges are added such that each of its parents is connected to a copy of v_a. At the end, a tree T is formed (Figure 2(b)).

For any $\mathbf{w} \in \mathbb{R}^n$ defined on V, the corresponding vector defined on the nodes of T is denoted $\mathbf{z} = [z_{1,1}, \underbrace{z_{2,1}, z_{2,2}, \ldots, z_{2,n_{\text{par}}(2)}}_{n_{\text{par}}(2) \text{ times}}, \ldots, \underbrace{z_{n,1}, \ldots, z_{n,n_{\text{par}}(n)}}_{n_{\text{par}}(n) \text{ times}}]^T \in \mathbb{R}^{|E|+1}$. Here, we assume that the root has index 1.



(a) adding a pseudo-root. (b) tree obtained after the conversion.

Figure 2. Converting a DAG to a tree.

Moreover, for notational simplicity, we set $n_{\text{par}}(1) = 1$, and thus $z_{1,1} = w_1$. By construction, if $\mathbf{w}$ satisfies the isotonic constraints in G, $\mathbf{z}$ also satisfies the isotonic constraints in T. Moreover, it can be easily seen that $\mathbf{z}$ and $\mathbf{w}$ are related as $\mathbf{z} = \mathbf{Q}\mathbf{w}$, where $\mathbf{Q} \in \mathbb{R}^{(|E|+1)\times n}$ has rows indexed in the same order as $\mathbf{z}$, and

$$Q_{ec} = \begin{cases} 1 & b = c \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where $e = e_{ab} \in E$ is an edge from v_a to v_b.

With this reformulation, the proximal step, with $\mathcal{W}$ in (15), can be rewritten as

$$\min_{\mathbf{z}\in\mathcal{Z}, \mathbf{w}\in\mathbb{R}^n} \underbrace{\delta(\mathbf{z}) + \sum_{a=1}^{n}\sum_{p=1}^{n_{\text{par}}(a)} \frac{(z_{a,p} - u_a)^2}{2n_{\text{par}}(a)}}_{\phi(\mathbf{z})} + \underbrace{\Omega(\mathbf{w})}_{\psi(\mathbf{w})} \tag{21}$$
$$\text{s.t.} \quad \mathbf{z} = \mathbf{Q}\mathbf{w},$$

where $\mathcal{Z} = [B_1, B_2]^{|E|+1}$, and $\delta(\mathbf{z}) = 0$ if $\mathbf{z}$ satisfies the isotonic constraints in T; and ∞ otherwise.

2) Updating of $\mathbf{z}$*:* Using ADMM, we first show that the update of $\mathbf{z}$ reduces to a standard isotonic regression problem. Specifically, with ϕ as defined in (21), update rule (9) can be rewritten as

$$\begin{aligned} &\min_{\mathbf{z}\in\mathcal{Z}} \ \phi(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{z} - \mathbf{Q}\mathbf{w}^t + \mathbf{r}^{(t)}\|^2 \\ =\ &\min_{\mathbf{z}\in\mathcal{Z}} \sum_{a=1}^{n}\sum_{p=1}^{n_{\text{par}}(a)} \frac{(z_{a,p} - u_a)^2}{2n_{\text{par}}(a)} + \frac{\rho(z_{a,p} - w_a^{(t)} + r_{a,p}^{(t)})^2}{2} \\ &\quad + \delta(\mathbf{z}) \\ =\ &\min_{\mathbf{z}\in\mathcal{Z}} \sum_{a=1}^{n}\sum_{p=1}^{n_{\text{par}}(a)} \eta_{a,p}\left(z_{a,p} - \alpha_{a,p}\right)^2 \qquad (22) \\ &\quad \text{s.t.} \quad \mathbf{z} \text{ satisfies the isotonic constraint in } T, \end{aligned}$$

where $\eta_{a,p} = \frac{1+\rho n_{\text{par}}(a)}{2n_{\text{par}}(a)}$, $\alpha_{a,p} = \frac{u_a + n_{\text{par}}(a)\rho(w_a^{(t)} - r_{a,p}^{(t)})}{1+\rho n_{\text{par}}(a)}$, and the elements $r_{a,p}^{(t)}$'s of $\mathbf{r}^{(t)}$ are ordered in the same way as $\mathbf{z}$. This is a standard (bounded) isotonic regression problem with tree-ordered constraints. As shown in [18], its optimal solution can be obtained by first dropping the bound constraints in $\mathcal{Z}$, solve the unbounded isotonic regression problem with algorithms in [7], [20], and then project the solution back onto $\mathcal{Z}$.

3) Updating of $\mathbf{w}$*:* Next, we show that the update of $\mathbf{w}$ can be reformulated as a proximal step. Specifically, with ψ as defined in (21), the update rule (10) can be rewritten as

$$\begin{aligned}
&\min_{\mathbf{w}\in\mathbb{R}^n} \psi(\mathbf{w}) + \frac{\rho}{2}\|\mathbf{z}^{(t+1)} - \mathbf{Q}\mathbf{w} + \mathbf{r}^{(t)}\|^2 \\
&= \min_{\mathbf{w}\in\mathbb{R}^n} \Omega(\mathbf{w}) + \sum_{a=1}^{n}\sum_{p=1}^{n_{\text{par}}(a)} \frac{\rho}{2}(z_{a,p}^{(t+1)} - w_a^{(t)} + r_{a,p}^{(t)})^2 \\
&= \min_{\mathbf{w}\in\mathbb{R}^n} \Omega(\mathbf{w}) + \sum_{a=1}^{n} \frac{\rho n_{\text{par}}(a)}{2}(w_a - \beta_a)^2, \qquad (23)
\end{aligned}$$

where $\beta_a = \frac{\sum_{p=1}^{n_{\text{par}}(a)} z_{a,p}^{(t+1)} + r_{a,p}^{(t)}}{n_{\text{par}}(a)}$. This is of the same form as the proximal step in (8), and closed-form solutions are readily available for popular Ω's, including:

1) $\Omega(\mathbf{w}) = \lambda\|\mathbf{w}\|_1$, where λ is the regularization parameter: As the ℓ_1-norm is separable, the optimal $\mathbf{w}^*$ of (23) can be obtained by soft-thresholding at each dimension [21], i.e.,
$$w_a^* = \begin{cases} \text{sgn}(\beta_a)\left(|\beta_a| - \frac{\lambda}{\rho n_{\text{par}}(a)}\right), & |\beta_a| > \frac{\lambda}{\rho n_{\text{par}}(a)} \\ 0 & \text{otherwise.} \end{cases}$$
2) $\Omega(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$: By setting the gradient of the objective in (23) to zero, we obtain $w_a^* = \frac{\rho n_{\text{par}}(a)\beta_a}{\rho n_{\text{par}}(a)+\lambda}$.
3) $\Omega(\mathbf{w}) = \lambda\|\mathbf{w}\|_\infty$: Without loss of generality, we assume that $|\beta_1| \geq |\beta_2| \geq \cdots \geq |\beta_n|$ (otherwise, one can rearrange the indices). Using a derivation similar to that in [12], it can be shown that if $\rho\sum_{a=1}^{n} n_{\text{par}}(a)|\beta_a| \leq \lambda$, then the optimal $\mathbf{w}^* = \mathbf{0}$; otherwise,
$$w_a^* = \begin{cases} \text{sgn}(\beta_a)\frac{\rho\sum_{b\leq b^*} n_{\text{par}}(b)|\beta_b| - \lambda}{\rho\sum_{b\leq b^*} n_{\text{par}}(b)}, & a \leq b^* \\ 0, & a > b^*, \end{cases}$$
where b^* is the smallest $b \in \{1, 2, \ldots, n\}$ such that $\frac{\rho\sum_{c\leq b} n_{\text{par}}(c)|\beta_c| - \lambda}{\sum_{c\leq b}\rho n_{\text{par}}(c)} > |\beta_{b+1}|$.

For regularizers Ω that do not lead to an easy solution for (23), we use the split inexact Uzawa method [15] as discussed in Section II-B. Let $\mathbf{Q}_{\cdot a}$ be the ath column of $\mathbf{Q}$. Note from (20) that $\mathbf{Q}_{\cdot a}$ contains $n_{\text{par}}(a)$ 1's, and the n columns of $\mathbf{Q}$ are linearly independent. Hence, $\mathbf{Q}^T\mathbf{Q}$ is a diagonal matrix with the largest diagonal element $n_{\max} \equiv \max_{a=1,\ldots,n} n_{\text{par}}(a)$, and $\lambda_{\max}(\mathbf{Q}^T\mathbf{Q}) = n_{\max}$. From (12), the linearized problem is then

$$\min_{\mathbf{w}\in\mathbb{R}^n} \Omega(\mathbf{w}) + \frac{\rho n_{\max}}{2}\|\mathbf{w} - \boldsymbol{\gamma}\|^2, \qquad (24)$$

where

$$\begin{aligned}
\gamma_a &= w_a^{(t)} - \frac{\mathbf{Q}_{\cdot a}^T(\mathbf{Q}\mathbf{w}^{(t)} - \mathbf{z}^{(t+1)} - \mathbf{r}^{(t)})}{n_{\max}} \\
&= w_a^{(t)} - \frac{n_{\text{par}}(a)w_a^{(t)} - \sum_{p=1}^{n_{\text{par}}(a)}(z_{a,p}^{(t+1)} + r_{a,p}^{(t)})}{n_{\max}} \\
&= w_a^{(t)} - \frac{n_{\text{par}}(a)(w_a^{(t)} - \beta_a)}{n_{\max}}.
\end{aligned}$$

Note that (24) is now a proximal step, and thus can be solved efficiently if Ω is "simple".

In the following, we provide a concrete example, namely, group lasso with non-overlapping groups. Given a set of non-overlapping groups $\mathcal{G}$ defined on the dimensions of $\mathbf{w}$, we have $\Omega(\mathbf{w}) = \lambda\sum_{g\in\mathcal{G}}\|\mathbf{w}_g\|$, where $\mathbf{w}_g$ is the sub-vector of $\mathbf{w}$ restricted to a group $g \in \mathcal{G}$. With this Ω, solving (23) is difficult, while solving (24) is surprisingly easy. Specifically, as the groups in $\mathcal{G}$ are non-overlapping, we can decompose (24) into smaller subproblems, one for each group: $\min_{\mathbf{w}_g} \frac{1}{2}\|\mathbf{w}_g - \boldsymbol{\gamma}_g\|^2 + \frac{\lambda}{\rho n_{\max}}\|\mathbf{w}_g\|$. This admits a closed-form solution [22]:

$$\mathbf{w}_g^* = \begin{cases} \boldsymbol{\gamma}_g\left(1 - \frac{\lambda}{\rho n_{\max}\|\boldsymbol{\gamma}_g\|}\right), & \rho n_{\max}\|\boldsymbol{\gamma}_g\| > \lambda \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

4) Time Complexity: The conversion from DAG to tree takes $\mathcal{O}(|E|)$ time. Since ϕ is strongly convex (due to the quadratic term) and $\mathbf{Q}$ has full column rank, both the standard ADMM and its split inexact Uzawa variant converge linearly, i.e., it takes $O\left(\log\frac{1}{\epsilon}\right)$ iterations to obtain an ϵ-approximate solution[2] [16]. In each iteration, updating $\mathbf{z}$ takes $\mathcal{O}(|E|\log|E|)$ time for the unbounded isotonic regression problem with tree-ordered constraints [20], and $\mathcal{O}(|E|)$ time for the projection onto $\mathcal{Z}$. As for the update of $\mathbf{w}$, one has to first obtain the β_a's, which takes $\mathcal{O}(|E|)$ time (but is subsumed by the $\mathcal{O}(|E|\log|E|)$ time above). Next, the proximal step for $\|\mathbf{w}\|_1, \|\mathbf{w}\|_2^2$, or $\sum_{g\in\mathcal{G}}\|\mathbf{w}_g\|$ takes $t_{\text{prox}} = \mathcal{O}(n)$ time; while that for $\|\mathbf{w}\|_\infty$ takes $t_{\text{prox}} = \mathcal{O}(n\log n)$ time [12]. Hence, Algorithm 2 takes a total of $\mathcal{O}\left(\log\frac{1}{\epsilon}(|E|\log|E| + t_{\text{prox}})\right)$ time.

IV. Solvers Based on Only ADMM

Instead of using a combination of (outer) FISTA and (inner) ADMM iterations as in Section III, we show in this section that problem (14) can also be solved by using only ADMM. In Section IV-A, we first propose a naive approach, which is then improved by a more refined approach in Section IV-B.

A. Naive ADMM Approach

We rewrite problem (14) as

$$\begin{aligned}
\min_{\mathbf{w}\in\mathbb{R}^n, \boldsymbol{\omega}_2\in\mathbb{R}^{2n}} \quad & \underbrace{\frac{1}{N}\sum_{i=1}^{N} l(\mathbf{w}^T\mathbf{x}_i, y_i)}_{\phi(\mathbf{w})} + \underbrace{\tilde{\delta}(\mathbf{w}_1) + \Omega(\mathbf{w}_2)}_{\psi(\boldsymbol{\omega}_2)} \\
\text{s.t.} \quad & \mathbf{A}\mathbf{w} = \boldsymbol{\omega}_2, \qquad (25)
\end{aligned}$$

where $\boldsymbol{\omega}_2 \equiv \begin{bmatrix}\mathbf{w}_1 \\ \mathbf{w}_2\end{bmatrix}$, $\mathbf{A} \equiv \begin{bmatrix}\mathbf{I} \\ \mathbf{I}\end{bmatrix}$, $\mathbf{I}$ is the identity matrix, and $\tilde{\delta}(\mathbf{w}_1) = \begin{cases} 0 & \mathbf{w}_1 \in \mathcal{W} \\ \infty & \text{otherwise} \end{cases}$. From (10), the ADMM update

[2]Note that while the original works in [8], [10] require the proximal step to be solved exactly, this has been recently relaxed to allow the proximal step to be only approximately solved (see [23] and references therein).

Algorithm 3 Naive ADMM for solving (25).

$t \leftarrow 0$; set $\mathbf{w}^0, \boldsymbol{\omega}_2^0, \mathbf{r}^0 \leftarrow \mathbf{0}$;
repeat
  update $\mathbf{w}^{(t+1)}$ using (26);
  $\mathbf{w}_1^{(t+1)} \leftarrow \arg\min_{\mathbf{w}_1 \in \mathcal{W}} \left\| \mathbf{w}_1 - (\mathbf{w}^{(t+1)} + \mathbf{r}_1^{(t)}) \right\|^2$;
  $\mathbf{w}_2^{(t+1)} \leftarrow \arg\min_{\mathbf{w}_2 \in \mathbb{R}^n} \frac{\rho}{2} \left\| \mathbf{w}_2 - (\mathbf{w}^{(t+1)} + \mathbf{r}_2^{(t)}) \right\|^2 + \Omega(\mathbf{w}_2)$;
  $\mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} + \mathbf{A}\mathbf{w}^{(t+1)} - \boldsymbol{\omega}_2^{(t+1)}$;
  $t \leftarrow t + 1$;
until convergence;
return $\mathbf{w}^{(t)}$.

rule for $\mathbf{w}$ is

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w} \in \mathbb{R}^n} \frac{\sum_{i=1}^N l(\mathbf{w}^T \mathbf{x}_i, y_i)}{N} + \frac{\rho}{2} \left\| \mathbf{A}\mathbf{w} - \boldsymbol{\omega}_2^t + \mathbf{r}^{(t)} \right\|^2. \tag{26}$$

Depending on the form of l, this may require the use of a nonlinear solver such as L-BFGS. However, for the commonly used square loss $l(\mathbf{w}^T \mathbf{x}_i, y_i) = \frac{1}{2}(\mathbf{w}^T \mathbf{x}_i - y_i)^2$, a closed-form solution of (26) can be easily obtained as

$$\mathbf{w}^{(t+1)} = \left(\frac{\mathbf{X}^T \mathbf{X}}{N} + \rho \mathbf{A}^T \mathbf{A} \right)^{-1} \left(\frac{\mathbf{X}^T \mathbf{y}}{N} + \rho \mathbf{A}^T (\boldsymbol{\omega}_2^{(t)} - \mathbf{r}^{(t)}) \right), \tag{27}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ and $\mathbf{y} = [y_1, y_2, \ldots, y_N]^T$.

As for the update of $\boldsymbol{\omega}_2$, we have from (9),

$$\begin{aligned} \boldsymbol{\omega}_2^{(t+1)} = \; & \arg\min_{\boldsymbol{\omega}_2 \in \mathbb{R}^{2n}} \; \tilde{\delta}(\mathbf{w}_1) + \Omega(\mathbf{w}_2) \\ & + \frac{\rho}{2} \left\| \begin{bmatrix} \mathbf{w}^{(t+1)} \\ \mathbf{w}^{(t+1)} \end{bmatrix} - \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{r}_1^{(t)} \\ \mathbf{r}_2^{(t)} \end{bmatrix} \right\|^2 \end{aligned}$$

where $\mathbf{r}_1$ (resp. $\mathbf{r}_2$) is the subvector in $\mathbf{r}$ corresponding to $\mathbf{w}_1$ (resp. $\mathbf{w}_2$). Obviously, $\mathbf{w}_1$ and $\mathbf{w}_2$ can be optimized independently. The subproblem involving $\mathbf{w}_1$ can be rewritten as

$$\min_{\mathbf{w}_1 \in \mathcal{W}} \left\| \mathbf{w}_1 - (\mathbf{w}^{(t+1)} + \mathbf{r}_1^{(t)}) \right\|^2.$$

This is an isotonic regression problem with DAG-ordered constraints [18], and can be solved by algorithms in [24], [19]. On the other hand, the subproblem involving $\mathbf{w}_2$ is

$$\min_{\mathbf{w}_2 \in \mathbb{R}^n} \frac{\rho}{2} \left\| \mathbf{w}_2 - (\mathbf{w}^{(t+1)} + \mathbf{r}_2^{(t)}) \right\|^2 + \Omega(\mathbf{w}_2). \tag{28}$$

This is a proximal step, and can be solved efficiently in closed-form for "simple" Ω [12]. The whole procedure is shown in Algorithm 3.

From [15], [25], ADMM requires $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ iterations to obtain an ϵ-accurate solution. In each ADMM iteration, the computational cost is dominated[3] by the update of $\mathbf{w}_1$, which takes $\mathcal{O}(n^4)$ time [24], [19]. This is expensive even for moderate values of n. The total complexity of the algorithm is $\mathcal{O}\left(\frac{n^4}{\epsilon}\right)$.

[3] In general, the update of $\mathbf{w}$ in (26) can also be expensive. Nevertheless, for the square loss considered in this experiments, the update in (27) is cheap, assuming that $\left(\frac{\mathbf{X}^T \mathbf{X}}{N} + \rho \mathbf{A}^T \mathbf{A}\right)^{-1}$ can be pre-computed and stored.

B. Linearized ADMM with Transformed Tree Constraints

The previous ADMM approach involves solving an expensive isotonic regression problem on DAG ordering. In this section, we alleviate this problem by converting the DAG constraints to tree constraints as in Section III-C. Problem (14) can then be reformulated as

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^n, \tilde{\boldsymbol{\omega}}_2 \in \mathbb{R}^{|E|+1+n}} \quad & \underbrace{\frac{1}{N} \sum_{i=1}^N l(\mathbf{w}^T \mathbf{x}_i, y_i)}_{\phi(\mathbf{w})} + \underbrace{\delta(\mathbf{z}) + \Omega(\mathbf{w}_1)}_{\psi(\tilde{\boldsymbol{\omega}}_2)} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{w} = \tilde{\boldsymbol{\omega}}_2, \end{aligned} \tag{29}$$

where $\mathbf{A} \equiv \begin{bmatrix} \mathbf{Q} \\ \mathbf{I} \end{bmatrix}$ and $\tilde{\boldsymbol{\omega}}_2 \equiv \begin{bmatrix} \mathbf{z} \\ \mathbf{w}_1 \end{bmatrix}$, $\delta(\mathbf{z}) = 0$ if $\mathbf{z}$ satisfies the isotonic constraints in T and $z_i \in [B_1, B_2], \forall i$; and ∞ otherwise. Note the similarity with (25), except that (29) has $\delta(\mathbf{z})$ (defined w.r.t. the tree-ordered constraints) instead of $\tilde{\delta}(\mathbf{w})$ (defined w.r.t. the DAG-ordered constraints).

1) Updating of $\mathbf{w}$*:* Note that ϕ in (29) is the same as in (25). Hence, the update rule of $\mathbf{w}$ is also the same, which involves a matrix inversion even when l is the square function. In this section, we alleviate this problem by linearizing ϕ. Specifically, for (29), the linearized update rule in (11) can be rewritten as

$$\begin{aligned} & \min_{\mathbf{w} \in \mathbb{R}^n} \nabla \phi(\mathbf{w}^{(t)})^T \mathbf{w} + \frac{L \|\mathbf{w} - \mathbf{w}^{(t)}\|^2 + \rho \|\mathbf{A}\mathbf{w} - \tilde{\boldsymbol{\omega}}_2^t + \mathbf{r}^{(t)}\|^2}{2} \\ = \; & \min_{\mathbf{w}} \sum_{a=1}^n \eta_a (w_a - \tilde{\beta}_a)^2, \end{aligned} \tag{30}$$

where L is the Lipschitz constant of $\nabla \phi$,

$$\begin{aligned} \eta_a &= \frac{L + \rho(1 + n_{\text{par}}(a))}{2}, \\ \tilde{\beta}_a &= \frac{L w_a^{(t)} - [\nabla \phi(\mathbf{w}^{(t)})]_a + \rho([\mathbf{w}_1^{(t)}]_a - [\mathbf{r}_1^{(t)}]_a)}{L + \rho(1 + n_{\text{par}}(a))} \\ & \quad + \frac{\rho \sum_{p=1}^{n_{\text{par}}(a)} z_{a,p}^{(t)} - [\mathbf{r}_{\mathbf{z}}^{(t)}]_{a,p}}{L + \rho(1 + n_{\text{par}}(a))}, \end{aligned} \tag{31}$$

$\mathbf{r}_1^{(t)}$ (resp. $\mathbf{r}_{\mathbf{z}}^{(t)}$) is the subvector in $\mathbf{r}^{(t)}$ corresponding to $\mathbf{w}_1$ (resp. $\mathbf{z}$), and $[\mathbf{r}_{\mathbf{z}}^{(t)}]_{a,p}$'s are the elements of $\mathbf{r}_{\mathbf{z}}^{(t)}$ ordered in the same way as $\mathbf{z}$. Hence, $\mathbf{w}^{(t+1)}$, which is the optimal solution in (30), is simply given by $w_a^{(t+1)} = \tilde{\beta}_a$ for all a.

2) Updating of $\tilde{\boldsymbol{\omega}}_2$*:* From (9), we have

$$\begin{aligned} \tilde{\boldsymbol{\omega}}_2^{(t+1)} = \; & \arg\min_{\tilde{\boldsymbol{\omega}}_2} \delta(\mathbf{z}) + \Omega(\mathbf{w}_1) \\ & + \frac{\rho}{2} \left\| \begin{bmatrix} \mathbf{Q} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{w}^{(t+1)} \\ \mathbf{w}^{(t+1)} \end{bmatrix} - \begin{bmatrix} \mathbf{z} \\ \mathbf{w}_1 \end{bmatrix} + \begin{bmatrix} \mathbf{r}_{\mathbf{z}}^{(t)} \\ \mathbf{r}_1^{(t)} \end{bmatrix} \right\|^2. \end{aligned}$$

Algorithm 4 Linearized ADMM with transformed tree constraints.

convert problem (14) to (29);
$t \leftarrow 0$; set $\mathbf{w}^0, \boldsymbol{\omega}_2^0, \mathbf{r}^0 \leftarrow \mathbf{0}$;
repeat
  $w_a^{(t+1)} \leftarrow \tilde{\beta}_a$ for all a, where $\tilde{\beta}_a$ is given in (31);
  update $\mathbf{z}^{(t+1)}$ as in (32);
  $\mathbf{w}_1^{(t+1)} \leftarrow \arg\min_{\mathbf{w}_1 \in \mathbb{R}^n} \frac{\rho}{2} \left\| \mathbf{w}_1 - (\mathbf{w}^{(t+1)} + \mathbf{r}_1^{(t)}) \right\|^2 + \Omega(\mathbf{w}_1)$;
  $\mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} + \mathbf{A}\mathbf{w}^{(t+1)} - \tilde{\boldsymbol{\omega}}_2^{(t+1)}$;
  $t \leftarrow t + 1$;
until convergence;
return $\mathbf{z}^{(t)}$.

As in Section IV-A, $\mathbf{z}$ and $\mathbf{w}_1$ can be optimized independently. In particular, the update of $\mathbf{w}_1$ is exactly the same as in (28), whereas the update of $\mathbf{z}$ can be rewritten as

$$\begin{aligned} &\min_{\mathbf{z}} \ \delta(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{Q}\mathbf{w}^{(t+1)} - \mathbf{z} + \mathbf{r}_{\mathbf{z}}^{(t)}\|^2 \\ = \ & \min_{\mathbf{z}} \ \sum_{a=1}^{N} \sum_{p=1}^{n_{\text{par}}(a)} (z_{a,p} - \alpha_{a,p})^2 \\ & \text{s.t.} \quad \mathbf{z} \text{ satisfies the isotonic constraint in } T, \\ & \quad\ z_{a,p} \in [B_1, B_2], \end{aligned} \tag{32}$$

where $\alpha_{a,p} = w_a^{(t+1)} + [\mathbf{r}_{\mathbf{z}}^{(t)}]_{a,p}$. This is a bounded isotonic regression problem with tree-ordered constraints, and can be solved by the algorithms in [20], [7]. The whole procedure is shown in Algorithm 4.

3) Time Complexity: The conversion from DAG to tree takes $\mathcal{O}(|E|)$ time. In each ADMM iteration, updating $\mathbf{w}$ takes $\mathcal{O}(|E|)$ time to compute $\tilde{\beta}_a$'s; updating $\mathbf{w}_1$ takes t_{prox} time; while updating of $\mathbf{z}$ involves an isotonic regression problem with tree-ordered constraints, which takes $\mathcal{O}(|E| \log |E|)$ time [20]. The linearized ADMM algorithm has the same convergence rate as standard ADMM, i.e., $\mathcal{O}\left(\frac{1}{T}\right)$ [13]. Hence, the total complexity is $\mathcal{O}\left(\frac{|E| \log |E| + t_{\text{prox}}}{\epsilon}\right)$, which is much better than the naive ADMM formulation in Section IV-A.

While the original problem in (14) has only n unknowns, $|E| + 1$ auxiliary unknowns are introduced in (29). In the convergence results on ADMM (e.g., [13], [15]), the error bound involves $\frac{1}{t}\|\boldsymbol{\omega}_2^0 - \boldsymbol{\omega}_2^*\|^2$, where $\boldsymbol{\omega}_2^*$ is the optimal $\boldsymbol{\omega}_2$. Consequently, the more auxiliary variables there are, the higher the dimensionality of $\boldsymbol{\omega}_2$, and the larger the error. Indeed, in order to achieve the same accuracy, it needs $\mathcal{O}\left(\frac{|E|}{n}\right)$ times more iterations than the naive ADMM. This detrimental effect will be observed empirically in Section V-D. Interestingly, while Algorithm 2 also has as many auxiliary variables as Algorithm 4, it does not suffer from the same problem due to its linear convergence rate. Let the optimal solution be $(\boldsymbol{\omega}_1^*, \boldsymbol{\omega}_2^*, \mathbf{r}^*)$. Its error bound is of the form $\left\| \begin{matrix} \boldsymbol{\omega}_1^{(t)} - \boldsymbol{\omega}_1^* \\ \boldsymbol{\omega}_2^{(t)} - \boldsymbol{\omega}_2^* \\ \mathbf{r}^{(t)} - \mathbf{r}^* \end{matrix} \right\|^2 \leq \underbrace{\mu\mu\ldots\mu}_{t \text{ times}} \epsilon^0 \leq \epsilon$, where $\epsilon^0 \equiv \left\| \begin{matrix} \boldsymbol{\omega}_1^0 - \boldsymbol{\omega}_1^* \\ \boldsymbol{\omega}_2^0 - \boldsymbol{\omega}_2^* \\ \mathbf{r}^0 - \mathbf{r}^* \end{matrix} \right\|^2$, and $0 < \mu < 1$. Consequently, the number of iterations required to converge to an ϵ-approximate solution is $\mathcal{O}\left(\log \frac{\epsilon^0}{\epsilon}\right)$. When $|E|+1$ auxiliary unknowns are introduced, we have $\tilde{\epsilon}^0 = \mathcal{O}\left(\frac{|E|\epsilon^0}{n}\right)$, and the number of required iterations becomes $\mathcal{O}\left(\log \frac{|E|\epsilon^0}{n\epsilon}\right) = \mathcal{O}\left(\log \frac{1}{\epsilon}\right) + \mathcal{O}\left(\log \frac{|E|\epsilon^0}{n}\right)$. Hence, only $\mathcal{O}\left(\log \frac{|E|\epsilon^0}{n}\right)$ extra iterations, instead of $\mathcal{O}\left(\frac{|E|}{n}\right)$ times more iterations, are needed.

V. Experiments

In this section, experiments are performed on a number of synthetic and real-world data sets. The FISTA-based approach[4] proposed in Section III (which is faster) is used in Sections V-A, V-B and V-C. Comparison with the purely-ADMM-based approaches will be presented in Section V-D.

A. Prior Knowledge on Coefficients

In this experiment, we demonstrate that prior knowledge on the relative magnitudes of regression coefficients can improve performance. The setup is similar to that in [7]. The data is generated from the model $y = \mathring{\mathbf{w}}^T\mathbf{x} + \epsilon$, where $\mathbf{x} \in \mathbb{R}^{90}$, $\epsilon \sim \mathcal{N}(0,1)$ and $\mathring{\mathbf{w}}$ is the ground truth parameter. The $\mathring{\mathbf{w}}$ vector is divided into 3 equal-sized groups $\mathring{\mathbf{w}}_{g1}, \mathring{\mathbf{w}}_{g2}$ and $\mathring{\mathbf{w}}_{g3}$. Each element of $\mathring{\mathbf{w}}_{g1}$ is drawn from $\mathcal{N}(0,1)$, while $\mathring{\mathbf{w}}_{g2} = \mathring{\mathbf{w}}_{g3} = \mathbf{0}$. As for the generation of $\mathbf{x}$, we use two setups: (i) the features of $\mathbf{x}$ are sampled independently from $\mathcal{N}(0,1)$; (ii) the features of $\mathbf{x}$ are correlated with the correlation coefficient uniformly distributed drawn from the uniform distribution $\mathcal{U}[0,1]$.

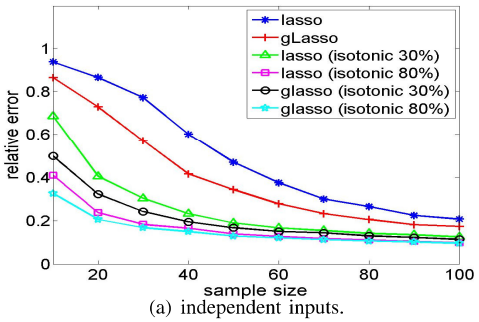
Prior knowledge is added in the form of isotonic constraints as follows. From each group in $\mathring{\mathbf{w}}$, we create a constraint $w_a \geq w_b$ on the parameter estimate $\mathbf{w}$ if (i) $\mathring{w}_a > \mathring{w}_b$; or (ii) $\mathring{w}_a = \mathring{w}_b$ and $a \leq b$. In general, the resultant set of constraints leads to a DAG. We vary the number of training samples from 10 to 100, and compare the standard lasso/group lasso with those using 30% or 80% of the isotonic constraints as side information. The regularization parameter λ is chosen using a validation set of size 100.

Note that though our setup is similar to [7], it is more general as (i) the w_i's are not constrained to be non-negative; (ii) there is a regularizer (namely, the ℓ_1-regularizer); and

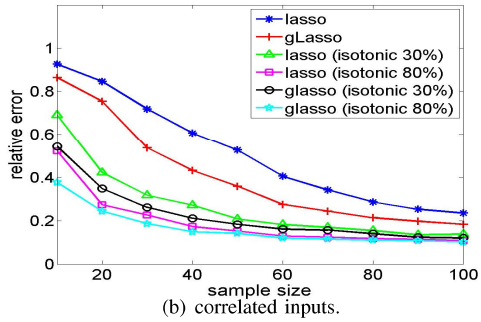
[4]The (outer) FISTA iteration is stopped when the relative change in the objectives from two consecutive iterations is less than 10^{-5}. The (inner) RIA iteration is stopped when both the primal and dual residuals are less than 10^{-5}.

(iii) the isotonic constraints are ordered on a DAG, not a tree. Hence, the Atda algorithm proposed in [7] cannot be used.

Figure 3 shows the relative model error $\frac{\|\mathbf{w}-\mathring{\mathbf{w}}\|}{\|\mathring{\mathbf{w}}\|}$ averaged over 10 runs. As expected, group lasso outperforms lasso, and the use of isotonic information helps both lasso and group lasso regardless of sample size. Moreover, increasing the amount of isotonic information further improves the accuracy of model parameter estimation.



(a) independent inputs.



(b) correlated inputs.

Figure 3. Relative errors obtained by the various models.

B. Hierarchical Interactions

Next, we perform experiments on six nucleoside reverse transcriptase inhibitors (NRTIs) data sets that have been used in [4] (Table I). The task is to predict the drug's susceptibility (a measure of drug resistance) based on the location of mutation. We compare the proposed algorithm with (i) the main-effect-lasso (MEL), which is standard lasso using the main effects only; (ii) all-pairs lasso (APL), which adds all the pairwise interactions to the lasso; (iii) weak hierarchical lasso (WHL); and (iv) strong hierarchical lasso (SHL), both of which are proposed in [4] to handle heredity relationships[5]. Using the notation in (1), isotonic constraints $|\theta_j| \geq |\Theta_{jk}|$ and $|\theta_k| \geq |\Theta_{jk}|$ are imposed for every (j,k). Given d original features, we have $\frac{1}{2}d(d-1)$ features for the second-order interaction, and $d(d-1)$ isotonic constraints. As not all pairwise interactions are likely to be useful, we also experiment with variants (suffixed with -S, for "subset") that only construct interactions from the 30 features having the largest regression coefficients in the MEL model. We use 50% of the data for training, 25% for validation and the remaining 25% for testing.

Table I
A SUMMARY OF THE NRTI DATA SETS.

data set	# samples	# original features	# 2nd-order features	# isotonic constraints
3TC	1,057	217	23,436	46,872
ABC	1,005	211	22,155	44,310
AZT	1,067	218	23,653	47,306
D4T	1,073	218	23,653	47,306
DDI	1,073	218	23,653	47,306
TDF	784	216	23,220	46,440

Table II shows the root mean square error (RMSE) averaged over 10 runs. As can be seen, on *D4T*, *DDI* and *TDF*, MEL performs similarly as the proposed model, indicating that there may be little pairwise interactions on these data sets. WHL performs even worse than MEL, which is also observed in [4]. On the other hand, the proposed models clearly outperform all others on *3TC*, *ABC* and *AZT*. Moreover, although the optimization problem is nonconvex (because of the constraints $|\theta_j| \geq |\Theta_{jk}|$ and $|\theta_k| \geq |\Theta_{jk}|$), we do not observe any empirical convergence problem for FISTA.

C. Constraints of the form $|w_a| > |w_b|$

In this section, we perform experiments on the breast cancer data set[6] [26], which contains 8,141 genes in 295 tumors. We use a varying number of genes that are most correlated to the output. From the supplementary information of [27], some of the genes are breast cancer susceptibility genes. As discussed in Section I, we assume that these genes have larger contributions than the others (i.e., $|w_a| \geq |w_b|$ if a is a cancer susceptibility gene and b is not). A summary of the number of genes, and the number of pairwise constraints generated, is shown in Table III. In particular, note that when all the genes are used, there are close to $485,000$ constraints.

Following [26], we reduce class imbalance by duplicating the positive samples twice. Different amounts (10%, 20% and 40%) of the data are then used for training, another 40% for testing, and the rest for validation. We compare three lasso models: (i) using the susceptibility genes only (denoted "susc-only"); (ii) using all the genes (denoted "without"); and (iii) using all the genes and also with the prior knowledge that susceptibility genes have larger contributions (denoted "with").

Table IV shows the classification accuracies averaged over 10 runs. As can be seen, the use of prior knowledge helps prediction in all cases. Using the susceptibility genes only

[5]The codes of WHL and SHL are provided by their authors.

[6]http://cbio.ensmp.fr/∼ljacob/

Table II
RMSE ON THE NRTI DATA SETS. THE BEST AND COMPARABLE RESULTS (ACCORDING TO THE PAIRWISE T-TEST WITH 95% CONFIDENCE) ARE HIGHLIGHTED.

data	MEL	APL	APL-S	WHL	SHL	ours	ours-S
3TC	0.256±0.021	0.195±0.024	0.192±0.021	0.209±0.040	0.201±0.021	0.201±0.023	**0.187±0.021**
ABC	0.218±0.017	0.210±0.019	0.215±0.023	0.222±0.024	0.213±0.018	**0.199±0.024**	0.209±0.020
AZT	0.606±0.043	0.592±0.039	0.585±0.043	**0.592±0.053**	0.628±0.029	**0.583±0.048**	**0.580±0.040**
D4T	**0.183±0.012**	0.195±0.012	0.190±0.012	0.196±0.025	0.191±0.012	**0.185±0.009**	**0.185±0.011**
DDI	**0.132±0.007**	0.143±0.014	0.138±0.011	0.142±0.011	0.138±0.010	**0.137±0.014**	**0.136±0.009**
TDF	**0.294±0.026**	0.326±0.028	0.307±0.023	0.306±0.023	0.314±0.026	**0.294±0.027**	**0.298±0.026**

Table III
A SUMMARY OF THE BREAST CANCER DATA SET.

#genes	#susceptibility genes	#pairwise constraints
300	6	1,764
500	9	4,419
1,000	10	9,900
8,141	60	484,860

(essentially performing feature selection) is also a useful way to constrain model complexity when the training data is limited and the available set of features is large.

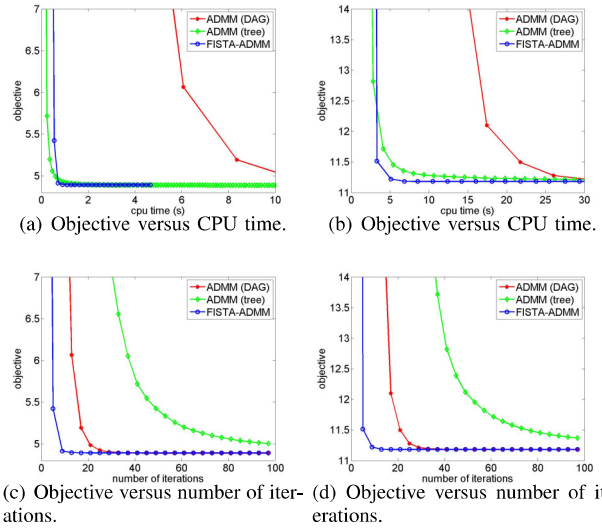
D. Comparison with the ADMM Approaches in Section IV

In this section, we compare the efficiencies of the solvers proposed in (i) Section III (denoted "FISTA-ADMM"); (ii) Section IV-A (denoted "ADMM(DAG)"); and (iii) Section IV-B (denoted "ADMM(tree)"). The setup is similar that in Section V-A. The data is generated from the model $y = \mathring{\mathbf{w}}^T\mathbf{x} + \epsilon$, where $\mathbf{x} \sim \mathcal{N}(0,1)$ and $\epsilon \sim \mathcal{N}(0,9)$. The vector $\mathring{\mathbf{w}}$ is divided into 3 equal-sized groups $\mathring{\mathbf{w}}_{g1}, \mathring{\mathbf{w}}_{g2}$ and $\mathring{\mathbf{w}}_{g3}$. Each element of $\mathring{\mathbf{w}}_{g1}$ is drawn from $\mathcal{N}(0,1)$, and $\mathring{\mathbf{w}}_{g2} = \mathring{\mathbf{w}}_{g3} = \mathbf{0}$. Moreover, 30% of the isotonic constraints (which are generated as in Section V-A) are used. We experimented with two settings: (i) 300 training samples, with $\mathbf{x} \in \mathbb{R}^{900}$; and (ii) 1000 training samples, with $\mathbf{x} \in \mathbb{R}^{3000}$.

Figures 4(a) and 4(b) show the optimization objective versus CPU time (averaged over 10 runs). As can be seen, the naive ADMM method ("ADMM(DAG)") is the slowest because of its high computational cost per iteration. The linearized ADMM with transformed tree constraints ("ADMM(tree)") is comparable with the FISTA-ADMM based approach ("FISTA-ADMM") for $\mathbf{x} \in \mathbb{R}^{900}$, but is slower for $\mathbf{x} \in \mathbb{R}^{3000}$. This is due to the large number of auxiliary variables introduced, and the larger number of iterations required (Figures 4(c) and 4(d)).

VI. CONCLUSION

In this paper, we considered models with hierarchical relationships among features, and three optimization solvers are proposed. In particular, we focus on the efficient computation of the proximal step, which serves as a core



(a) Objective versus CPU time.
(b) Objective versus CPU time.
(c) Objective versus number of iterations.
(d) Objective versus number of iterations.

Figure 4. Convergence of the various proposed solvers. Left: $\mathbf{x} \in \mathbb{R}^{900}$; Right: $\mathbf{x} \in \mathbb{R}^{3000}$.

building block in accelerated gradient methods. With the use of the ADMM, both update steps can be reduced to simple standard optimization problems. Experimental results on a number of data sets demonstrate the efficiency of the proposed solver, and the usefulness of the hierarchical relationships in improving generalization performance and parameter estimation.

ACKNOWLEDGMENT

This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant 614311).

REFERENCES

[1] R. Strausberg, A. Simpson, and R. Wooster, "Sequence-based cancer genomics: Progress, lessons and opportunities," *Nature Reviews Genetics*, vol. 4, pp. 409–418, 2003.

[2] M. Hamada and C. Wu, "Analysis of designed experiments with complex aliasing," *Journal of Quality Technology*, vol. 24, pp. 130–137, 1992.

Table IV
CLASSIFICATION ACCURACY ON THE BREAST CANCER DATA SET. THE BEST AND COMPARABLE RESULTS (ACCORDING TO THE PAIRWISE T-TEST WITH 95% CONFIDENCE) ARE HIGHLIGHTED.

	10% data for training			20% data for training			40% data for training		
#genes	susc-only	without	with	susc-only	without	with	susc-only	without	with
300	0.64±0.05	0.63±0.07	**0.67±0.06**	0.67±0.07	0.67±0.05	**0.72±0.05**	0.69±0.03	0.68±0.04	**0.72±0.04**
500	0.65±0.04	0.64±0.05	**0.68±0.03**	**0.67±0.05**	0.65±0.07	**0.70±0.05**	**0.68±0.05**	0.67±0.05	**0.70±0.06**
1000	0.62±0.06	0.61±0.06	**0.65±0.05**	0.65±0.05	0.67 ±0.04	**0.70±0.04**	**0.69±0.04**	0.67±0.04	**0.71±0.04**
8141	**0.60±0.06**	0.54±0.05	**0.57±0.07**	**0.61±0.07**	0.57±0.05	**0.63±0.04**	0.64±0.06	0.59±0.04	**0.66±0.06**

[3] N. Choi, W. Li, and J. Zhu, "Variable selection with the strong heredity constraint and its oracle property," *Journal of the American Statistical Association*, vol. 105, pp. 354–364, 2010.

[4] J. Bien, J. Taylor, and R. Tibshirani, "A lasso for hierarchical interactions," *Annals of Statistics.*, vol. 41, pp. 1111–1141, 2013.

[5] X. Li, N. Sudarsanam, and D. Frey, "Regularities in data from factorial experiments," *Complexity*, vol. 11, pp. 32–45, 2006.

[6] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Convex optimization with sparsity-inducing norms," in *Optimization for Machine Learning*. MIT, 2011, pp. 19–53.

[7] J. Liu, L. Sun, and J. Ye, "Projection onto a nonnegative max-heap," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 487–495.

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

[9] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 2010.

[10] Y. Nesterov, "Gradient methods for minimizing composite objective function," Catholic University of Louvain, Tech. Rep. 76, 2007.

[11] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach, "Network flow algorithms for structured sparsity," in *Advances in Neural Information Processing Systems 24*, 2010, pp. 1558–1566.

[12] J. Duchi and Y. Singer, "Efficient online and batch learning using forward backward splitting," *Journal of Machine Learning Research*, vol. 10, pp. 2873–2908, 2009.

[13] H. Ouyang, N. He, L. Tran, and A. Gray, "Stochastic alternating direction method of multipliers," in *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[14] T. Suzuki, "Dual averaging and proximal gradient descent for online alternating direction multiplier method," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 392–400.

[15] B. He and X. Yuan, "On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method," *SIAM Journal on Numerical Analysis*, vol. 50, pp. 700–709, 2012.

[16] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," Rice University, Tech. Rep. TR12-14, 2012.

[17] X. Zhang, M. Burger, X. Bresson, and S. Osher, "Bregmanized nonlocal regularization for deconvolution and sparse reconstruction," *SIAM Journal on Imaging Sciences*, vol. 3, pp. 253–276, 2010.

[18] R. Barlow, D. Bartholomew, J. Bremner, and H. Brunk, *Statistical Inference Under Order Restrictions*. New York: Wiley, 1972.

[19] Q. Stout, "Isotonic regression via partitioning," *Algorithmica*, vol. 66, pp. 93–112, 2013.

[20] P. Pardalos and G. Xue, "Algorithms for a class of isotonic regression problems," *Algorithmica*, vol. 23, pp. 211–222, 1999.

[21] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," in *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 272–279.

[22] J. Chen, J. Zhou, and J. Ye, "Integrating low-rank and group-sparse structures for robust multi-task learning," in *Proceedings of the 17th International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 42–50.

[23] M. Schmidt, N. L. Roux, and F. Bach, "Convergence rates of inexact proximal-gradient methods for convex optimization," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 1458–1466.

[24] R. Luss, S. Rosset, and M. Shahar, "Decomposing isotonic regression for efficiently solving large problems," in *Advances in Neural Information Processing Systems 23*, 2010, pp. 1513–1521.

[25] H. Wang and A. Banerjee, "Online alternating direction method," in *Proceedings of the 29th International Conference on Machine Learning*, 2012, pp. 1119–1126.

[26] L. Jacob, G. Obozinski, and J. Vert, "Group lasso with overlap and graph lasso," in *Proceedings of the 26th International Conference on Machine Learning*, 2009, pp. 433–440.

[27] H. Chuang, E. Lee, Y. Liu, D. Lee, and T. Ideker, "Network-based classification of breast cancer metastasis," *Molecular Systems Biology*, vol. 3, 2007.