Localized Lasso for High-Dimensional Regression

Makoto Yamada[1,2], **Koh Takeuchi**[3], **Tomoharu Iwata**[3], **John Shawe-Taylor**[4], **Samuel Kaski**[5]
[1]RIKEN AIP, [2]JST PRESTO, [3]NTT CS Labs [4]UCL, [5]Aalto University

Abstract

We introduce the localized Lasso, which learns models that both are interpretable and have a high predictive power in problems with high dimensionality d and small sample size n. More specifically, we consider a function defined by local sparse models, one at each data point. We introduce sample-wise network regularization to borrow strength across the models, and sample-wise exclusive group sparsity (a.k.a., $\ell_{1,2}$ norm) to introduce diversity into the choice of feature sets in the local models. The local models are interpretable in terms of similarity of their sparsity patterns. The cost function is convex, and thus has a globally optimal solution. Moreover, we propose a simple yet efficient iterative least-squares based optimization procedure for the localized Lasso, which does not need a tuning parameter, and is guaranteed to converge to a globally optimal solution. The solution is empirically shown to outperform alternatives for both simulated and genomic personalized/precision medicine data.

1 Introduction

A common problem in molecular medicine, shared by many other fields, is to learn predictions from data consisting of a large number of features (e.g., genes) and a small number of samples (e.g., drugs or patients). A key challenge is to tailor or "personalize" the predictions for each data sample, essentially solving a multi-task learning problem (Evgeniou and Pontil, 2007; Argyriou et al., 2008) where in each task $n = 1$. The features (genes) important for prediction can be different for different samples (patients or drugs), and reporting the important features is a key part of the data analysis, requiring models that are interpretable in addition to having high prediction accuracy. That is, the problem can be regarded as a *local* feature selection and prediction problem, which would be hard for existing multi-task learning approaches.

Sparse linear feature selection methods such as Lasso (Tibshirani, 1996) are useful for large p, small n problems. Standard feature selection methods select the same small set of features for all samples, which is too restrictive for the multi-task type of problems, where for instance effects of different drugs may be based on different features, and dimensionality needs to be minimized due to the very small sample size.

Recently, the network Lasso (Hallac et al., 2015a) method has been proposed for learning local functions $f(\boldsymbol{x}_i; \boldsymbol{w}_i), i = 1, \ldots, n$, by using network (graph) information between samples. In network Lasso, a group regularizer is introduced to the difference of the coefficient vectors between linked coefficients (i.e., $\boldsymbol{w}_i - \boldsymbol{w}_j$), making them similar. We can use this regularizer to make the local models borrow strength from linked models. In the network Lasso, sparsity has so far been used only for making the coefficient vectors similar instead of for feature selection, resulting in dense models.

We propose a sparse variant of the network Lasso, called the localized Lasso, which helps to choose interpretable features for each sample. More specifically, we propose to incorporate the sample-wise exclusive regularizer into the network Lasso framework. By imposing the network regularizer, we can borrow strength between samples neighboring in the graph, up to clustering or "stratifying" the samples according to how the predictions are made. Furthermore, by imposing a sample-wise exclusive group regularizer, each learned model is made sparse but the support remains non-empty, in contrast to what could happen with naive regularization. As a result, the sparsity pattern and the weights become similar for neighboring models. We propose an efficient iterative least squares algorithm and show that the algorithm will obtain a globally optimal solution. Through experiments on synthetic and real-world datasets, we show that the

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

proposed localized Lasso outperforms state-of-the-art methods even with a smaller number of features.

Contribution:

- We propose a *convex* local feature selection and prediction method. Specifically, we combine the exclusive regularizer and network regularizer to produce a locally defined model that gives accurate and interpretable predictions.
- We propose an efficient iterative least squares based optimization procedure, which does not need a tuning parameter and is guaranteed to converge to a globally optimal solution.

2 Proposed method

In this section, we first formulate the problem and then introduce the localized Lasso.

2.1 Problem Formulation

Let us denote an input vector by $\boldsymbol{x} = [x^{(1)}, \ldots, x^{(d)}]^\top \in \mathbb{R}^d$ and the corresponding output value $y \in \mathbb{R}$. The set of samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ has been drawn i.i.d. from a joint probability density $p(\boldsymbol{x}, y)$. We further assume a graph $\boldsymbol{R} \in \mathbb{R}^{n\times n}$, where $[\boldsymbol{R}]_{i,j} = r_{ij} \geq 0$ is the coefficient that represents the relatedness between the sample pair $(\boldsymbol{x}_i, y_i)$ and $(\boldsymbol{x}_j, y_j)$. In this paper, we assume that $\boldsymbol{R}$ is *undirected* (i.e., $\boldsymbol{R} = \boldsymbol{R}^\top$) and the diagonal elements of $\boldsymbol{R}$ are zero (i.e., $r_{11} = r_{22} = \ldots = r_{nn} = 0$).

The goal in this paper is to select multiple sets of features such that each set of features is locally associated with an individual data point or a cluster, from the training input-output samples and the graph information $\boldsymbol{R}$. In particular, we aim to learn a model with an interpretable sparsity pattern in the features.

2.2 Model

We employ the following model for each sample i:

$$y_i = \boldsymbol{w}_i^\top \boldsymbol{x}_i + e_i, \quad (1)$$

where e_i follows a normal distribution $N(0, \sigma^2)$. Here $\boldsymbol{w}_i \in \mathbb{R}^d$ contains the regression coefficients for sample $\boldsymbol{x}_i$ and $^\top$ denotes the transpose. Note that in regression problems the weight vectors are typically assumed to be equal, $\boldsymbol{w} = \boldsymbol{w}_1 = \ldots = \boldsymbol{w}_n$. Since we cannot assume the models to be based on the same features, and we want to interpret the support of the model for each sample, we use local models.

Since the number of unknown variables in Eq. (1) is the same as the number of observed variables, we need to regularize, for which we propose to use network Lasso type of regularization (Hallac et al., 2015a):

$$\rho(\boldsymbol{W}; \boldsymbol{R}, \lambda_1, \lambda_2) = \lambda_1 \sum_{i,j=1}^n r_{ij} \|\boldsymbol{w}_i - \boldsymbol{w}_j\|_2 + \lambda_2 \sum_{i=1}^n \|\boldsymbol{w}_i\|_1^2.$$

Here $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are the regularization parameters. By imposing the network regularization, we regularize the model parameters $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$ to be similar if $r_{ij} > 0$. If λ_1 is large, we will effectively cluster the samples according to how similar the $\boldsymbol{w}_i$s are, that is, according to the prediction criteria in the local models. More specifically, when $\|\boldsymbol{w}_i - \boldsymbol{w}_j\|_2$ is small (possibly zero), we can regard the i-th sample and j-th sample to belong to the same cluster. Moreover, outliers tend to form their own clusters, and thus, we can also detect outliers in addition to normal clusters; this would help in interpreting data.

The second regularization term is the $\ell_{1,2}$ regularizer (a.k.a., exclusive regularizer) (Kowalski, 2009; Zhou et al., 2010; Kong et al., 2014). By imposing the $\ell_{1,2}$ regularizer, we can select a small number of elements within each $\boldsymbol{w}_i$. Note that we regard each parameter vector $\boldsymbol{w}_i$ as a group (in total n groups), and are not treating each dimension as a group. Thanks to the square of ℓ_1 norm over the weight vector $\boldsymbol{w}_i$, the $\boldsymbol{w}_i$ remains non-zero (i.e., $\boldsymbol{w}_i \neq \boldsymbol{0}$). Similarities and differences in the sparsity patterns of the $\boldsymbol{w}_i$ are then easily interpretable, more easily than in dense vectors. Note that while simply imposing the ℓ_1 regularizer for all weights would induce sparsity too, for the heavy regularization required due to the small sample size, many of the $\boldsymbol{w}_i$ would be shrunk to zero. See Figure 1 for an example.

Our proposed regularizer can be seen as a (non-trivial) extension of network regularization (Hallac et al., 2015a), and hence it could be solved by a general alternating direction method of multipliers (ADMM) based solver. However, ADMM requires a tuning parameter for convergence (Nishihara et al., 2015). In this paper, we propose a simple yet effective iterative least-squares based optimization procedure, which does not need any tuning parameters, and is guaranteed to converge to a globally optimal solution.

2.3 Optimization problem

The optimization problem of the localized lasso[1] can be written as

$$\min_{\boldsymbol{W}} \; J(\boldsymbol{W}) = \sum_{i=1}^n (y_i - \boldsymbol{w}_i^\top \boldsymbol{x}_i)^2 + \rho(\boldsymbol{W}; \boldsymbol{R}, \lambda_1, \lambda_2), \quad (2)$$

[1] Code available at `http://www.makotoyamada-ml.com/localizedlasso.html`

Algorithm 1 Iterative Least-Squares Algorithm for solving Eq. (2)

```
Input: $\boldsymbol{Z} \in \mathbb{R}^{n\times(dn)}$, $\boldsymbol{y} \in \mathbb{R}^{n}$, $\boldsymbol{R} \in \mathbb{R}^{n\times n}$, $\lambda_1$, and $\lambda_2$.
Output: $\boldsymbol{W} \in \mathbb{R}^{n\times d}$.
Set $t = 0$, Initialize $\boldsymbol{F}_g^{(0)}$, $\boldsymbol{F}_e^{(0)}$.
repeat
    Compute $\mathrm{vec}(\boldsymbol{W}^{(t+1)}) = (\lambda_1\boldsymbol{F}_g^{(t)} + \lambda_2\boldsymbol{F}_e^{(t)})^{-1}\boldsymbol{Z}^\top(\boldsymbol{I}_n + \boldsymbol{Z}(\lambda_1\boldsymbol{F}_g^{(t)} + \lambda_2\boldsymbol{F}_e^{(t)})^{-1}\boldsymbol{Z}^\top)^{-1}\boldsymbol{y}$,
    Update $\boldsymbol{F}_g^{(t+1)}$, where $\boldsymbol{F}_g^{(t+1)} = \boldsymbol{I}_d \otimes \boldsymbol{C}^{(t+1)}$.
    Update $\boldsymbol{F}_e^{(t+1)}$, where $[\boldsymbol{F}_e^{(t+1)}]_{\ell,\ell} = \sum_{k=1}^{n} \frac{I_{k,\ell}\|\boldsymbol{w}_k^{(t+1)}\|_1}{[\mathrm{vec}(|\boldsymbol{W}^{(t+1)}|)]_\ell}$.
    $t = t + 1$.
until Converges
```

which is convex and hence has a globally optimal solution. Note that for classification problems the squared loss can be replaced by the logistic loss.

Let us denote $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]^\top$, $\boldsymbol{u}_i \in \mathbb{R}^n$, and $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n]^\top \in \mathbb{R}^{n\times d}$. We can alternatively write the objective function as

$$J(\boldsymbol{W}) = \|\boldsymbol{y} - \boldsymbol{Z}\mathrm{vec}(\boldsymbol{W})\|_2^2 + \rho(\boldsymbol{W}; \boldsymbol{R}, \lambda_1, \lambda_2), \quad (3)$$

where $\boldsymbol{Z} = [\mathrm{diag}(\boldsymbol{u}_1) \mid \mathrm{diag}(\boldsymbol{u}_2) \mid \ldots \mid \mathrm{diag}(\boldsymbol{u}_d)] \in \mathbb{R}^{n\times(dn)}$, $\mathrm{diag}\,(\boldsymbol{u}) \in \mathbb{R}^{n\times n}$ is the diagonal matrix whose diagonal elements are the $\boldsymbol{u}$, and $\mathrm{vec}(\cdot)$ is the vectorization operator such that

$$\mathrm{vec}(\boldsymbol{W}) = \Big[[\boldsymbol{W}]_{1,1}, [\boldsymbol{W}]_{2,1}, \ldots [\boldsymbol{W}]_{n,1}, \ldots, \\ [\boldsymbol{W}]_{1,d}, [\boldsymbol{W}]_{2,d}, \ldots [\boldsymbol{W}]_{n,d}\Big]^\top \in \mathbb{R}^{dn}.$$

Here we use the vectorization operator since it makes it possible to write the loss function and the two regularization terms as a function of $\mathrm{vec}(\boldsymbol{W})$, which is highly helpful for deriving a simple update formula for $\boldsymbol{W}$.

Taking the derivative of $J(\boldsymbol{W})$ with respect to $\mathrm{vec}(\boldsymbol{W})$ and using the Propositions 1 and 2 (See Supplementary material), the optimal solution is given as

$$\mathrm{vec}(\boldsymbol{W}) = (\boldsymbol{Z}^\top\boldsymbol{Z} + \lambda_1\boldsymbol{F}_g + \lambda_2\boldsymbol{F}_e)^{-1}\boldsymbol{Z}^\top\boldsymbol{y}, \quad (4)$$

where

$$\boldsymbol{F}_g = \boldsymbol{I}_d \otimes \boldsymbol{C},\ [\boldsymbol{F}_e]_{\ell,\ell} = \sum_{i=1}^{n} \frac{I_{i,\ell}\|\boldsymbol{w}_i\|_1}{[\mathrm{vec}(|\boldsymbol{W}|)]_\ell},$$

$$[\boldsymbol{C}]_{i,j} = \begin{cases} \sum_{j'=1}^{n} \frac{r_{ij'}}{\|\boldsymbol{w}_i - \boldsymbol{w}_{j'}\|_2} - \frac{r_{ij}}{\|\boldsymbol{w}_i - \boldsymbol{w}_j\|_2} & (i = j) \\ \frac{-r_{ij}}{\|\boldsymbol{w}_i - \boldsymbol{w}_j\|_2} & (i \neq j) \end{cases}.$$

Here $\boldsymbol{F}_e$ is diagonal, $\boldsymbol{I}_d \in \mathbb{R}^{d\times d}$ is the identity matrix, $\otimes$ is the Kronecker product, and the $I_{i,\ell} \in \{0,\ 1\}$ are group index indicators: $I_{i,\ell} = 1$ if the ℓ-th element $[\mathrm{vec}(\boldsymbol{W})]_\ell$ belongs to group i (i.e., $[\mathrm{vec}(\boldsymbol{W})]_\ell$ is the element of $\boldsymbol{w}_i$), otherwise $I_{i,\ell} = 0$.

Since the optimization problem Eq. (2) is convex, the $\boldsymbol{W}$ is a global optimum to the problem if and only if Eq. (4) is satisfied. However, the matrices $\boldsymbol{F}_g$ and $\boldsymbol{F}_e$ are dependent on $\boldsymbol{W}$ and are also unknown. Thus, we instead optimize the following objective function to solve Eq. (2):

$$\widetilde{J}(\boldsymbol{W}) = \|\boldsymbol{y} - \boldsymbol{Z}\mathrm{vec}(\boldsymbol{W})\|_2^2 \\ + \mathrm{vec}(\boldsymbol{W})^\top(\lambda_1\boldsymbol{F}_g^{(t)} + \lambda_2\boldsymbol{F}_e^{(t)})\mathrm{vec}(\boldsymbol{W}), \quad (5)$$

where $\boldsymbol{F}_g^{(t)} \in \mathbb{R}^{dn\times dn}$ is a block diagonal matrix and $\boldsymbol{F}_e^{(t)} \in \mathbb{R}^{dn\times dn}$ is a diagonal matrix whose diagonal elements are defined as[2]

$$\boldsymbol{F}_g^{(t)} = \boldsymbol{I}_d \otimes \boldsymbol{C}^{(t)},\ [\boldsymbol{F}_e]_{\ell,\ell}^{(t)} = \sum_{i=1}^{n} \frac{I_{i,\ell}\|\boldsymbol{w}_i^{(t)}\|_1}{[\mathrm{vec}(|\boldsymbol{W}^{(t)}|)]_\ell},$$

$$[\boldsymbol{C}^{(t)}]_{i,j} = \begin{cases} \sum_{j'=1}^{n} \frac{r_{ij'}}{\|\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_{j'}^{(t)}\|_2} - \frac{r_{ij}}{\|\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_j^{(t)}\|_2} & (i = j) \\ \frac{-r_{ij}}{\|\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_j^{(t)}\|_2} & (i \neq j) \end{cases}.$$

We propose to use the iterative least squares approach to optimize Eq. (5). With given $\boldsymbol{F}_g^{(t)}$ and $\boldsymbol{F}_e^{(t)}$, the optimal solution of $\boldsymbol{W}$ is obtained by solving $\frac{\partial\widetilde{J}(\boldsymbol{W})}{\partial\boldsymbol{W}} = \boldsymbol{0}$. The $\boldsymbol{W}$ is estimated as

$$\mathrm{vec}(\boldsymbol{W}^{(t+1)}) = {\boldsymbol{H}^{(t)}}^{-1}\boldsymbol{Z}^\top(\boldsymbol{I}_n + \boldsymbol{Z}{\boldsymbol{H}^{(t)}}^{-1}\boldsymbol{Z}^\top)^{-1}\boldsymbol{y}, \quad (6)$$

where $\boldsymbol{H}^{(t)} = \lambda_1\boldsymbol{F}_g^{(t)} + \lambda_2\boldsymbol{F}_e^{(t)}$, $\boldsymbol{F}_g^{(t)}$ is block diagonal and $\boldsymbol{F}_e^{(t)}$ diagonal. Here, we employ the Woodbury formula (Petersen et al., 2008). After we obtain $\boldsymbol{W}^{(t+1)}$, we update $\boldsymbol{F}_g^{(t+1)}$ and $\boldsymbol{F}_e^{(t+1)}$. We continue this two-step procedure until convergence. The algorithm is summarized in Algorithm 1.

Predicting for new test sample: For predicting on test sample $\boldsymbol{x}$, we use the estimated local models $\widehat{\boldsymbol{w}}_k$

[2] When $\boldsymbol{w}_i - \boldsymbol{w}_j = \boldsymbol{0}$, then $\boldsymbol{F}_g$ is the subgradient of $\sum_{i,j=1}^{n} r_{ij}\|\boldsymbol{w}_i - \boldsymbol{w}_j\|_2$. Also, $\boldsymbol{F}_e$ is the subgradient of $\sum_{i=1}^{n} \|\boldsymbol{w}_i\|_1^2$ when $[\mathrm{vec}(|\boldsymbol{W}|)]_\ell = 0$. However, we cannot set elements of $\boldsymbol{F}_g$ to 0 (i.e., when $\boldsymbol{w}_i - \boldsymbol{w}_j = \boldsymbol{0}$) or the element of $[\boldsymbol{F}_e]_{\ell,\ell} = 0$ (i.e., when $[\mathrm{vec}(|\boldsymbol{W}|)]_\ell = 0$), otherwise the Algorithm 1 cannot be guaranteed to converge. To deal with this issue, we can use $\sum_{i,j=1}^{n} r_{ij}\|\boldsymbol{w}_i - \boldsymbol{w}_j + \epsilon\|_2$ and $\sum_{i=1}^{n} \|\boldsymbol{w}_i + \epsilon\|_1^2$ ($\epsilon > 0$) instead (Kong et al., 2014; Nie et al., 2010).

Algorithm 2 Iterative Least-Squares Algorithm for solving Eq. (7)

```
Input: $\boldsymbol{x}$, $\boldsymbol{r}' \in \mathbb{R}^n$, and $\widehat{\boldsymbol{W}} \in \mathbb{R}^{d \times n}$.
Output: $\widehat{y} \in \mathbb{R}$ and $\widehat{\boldsymbol{w}} \in \mathbb{R}^d$.
Set $t = 0$, Initialize $\boldsymbol{f}_g \in \mathbb{R}^n$.
repeat
    Compute $\boldsymbol{w}^{(t+1)} = \frac{1}{\mathbf{1}_n^\top \boldsymbol{f}_g^{(t)}} \widehat{\boldsymbol{W}} \boldsymbol{f}_g^{(t)}$.
    Update $\boldsymbol{f}_g^{(t+1)}$, where $[\boldsymbol{f}_g^{(t+1)}]_i = \frac{[\boldsymbol{r}']_i}{2\|\widehat{\boldsymbol{w}}_i - \boldsymbol{w}^{(t+1)}\|_2}$.
    $t = t + 1$.
until Converges
$\widehat{y} = \widehat{\boldsymbol{w}}^\top \boldsymbol{x}$.
```

which are linked to the input $\boldsymbol{x}$. More specifically, we solve the Weber problem (Hallac et al., 2015a):

$$\min_{\boldsymbol{w}} \quad \sum_{i=1}^{n} r'_i \|\boldsymbol{w} - \widehat{\boldsymbol{w}}_i\|_2, \tag{7}$$

where $r'_i \geq 0$ is the link information between the test sample and the training sample $\boldsymbol{x}_i$. Since this problem is convex, we can solve it efficiently by an iterative update formula (see Algorithm 2). If there is no link information available, we simply average all $\widehat{\boldsymbol{w}}_i$s to estimate $\widehat{\boldsymbol{w}}$, and then predict as $\widehat{y} = \widehat{\boldsymbol{w}}^\top \boldsymbol{x}$.

2.4 Convergence analysis

Next, we prove the convergence of the algorithm.

Theorem 1 *The Algorithm 1 will monotonically decrease the objective function Eq.* (2) *in each iteration, and converge to the global optimum of the problem.*

Proof: Under the updating rule of Eq. (6)*, we have the following inequality using Lemma 4 and Lemma 8 (See Supplementary material):*

$$J(\boldsymbol{W}^{(t+1)}) - J(\boldsymbol{W}^{(t)}) \leq \widetilde{J}(\boldsymbol{W}^{(t+1)}) - \widetilde{J}(\boldsymbol{W}^{(t)}) \leq 0.$$

That is, the Algorithm 1 will monotonically decrease the objective function of Eq. (2)*. At convergence,* $\boldsymbol{F}_g^{(t)}$ *and* $\boldsymbol{F}_e^{(t)}$ *will satisfy Eq.* (4)*. Since the optimization problem Eq.* (2) *is convex, satisfying Eq.* (4) *means that* $\boldsymbol{W}$ *is a global optimum to the problem in Eq.* (2)*. Thus, the Algorithm 1 will converge to the global optimum of the problem Eq.* (2)*.* □

2.5 Sparse convex clustering

The proposed sparse regularization can be applied to convex clustering problems (Pelckmans et al., 2005; Hocking et al., 2011; Wang et al., 2016) by changing the objective function. The optimization problem is then

$$\min_{\boldsymbol{W}} \ J(\boldsymbol{W}) = \|\boldsymbol{X}^\top - \boldsymbol{W}\|_F^2 + \rho(\boldsymbol{W}; \boldsymbol{R}, \lambda_1, \lambda_2), \tag{8}$$

where

$$r_{ij} = \begin{cases} \delta_{ij} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2}\right) & (i \neq j) \\ 0 & \text{Otherwise} \end{cases}$$

is the Gaussian kernel. Here $\delta_{ij} = 1$ if $\boldsymbol{x}_j$ is included in the K-th neighbors of $\boldsymbol{x}_i$, otherwise $\delta_{ij} = 0$. Note that we define the neighborhood such that the $\boldsymbol{R}$ is always properly defined. The original convex clustering methods do not include the exclusive group sparsity regularization, and thus, the learned matrix $\boldsymbol{W}$ tends to be dense. Adding the sparsity makes the clusters more easily interpretable, even as biclusters or subspace clusters, still retaining convexity.

3 Related work

In this section, we review the existing regression methods and address the difference from the proposed method.

Sparsity-based global feature selection methods such as Lasso (Tibshirani, 1996) are useful for selecting genes. However, in personalized/precision medicine setups, we ultimately want to personalize the models for each patient (or drug), instead of assuming the same set of features (e.g., genes) for each.

The proposed method is also related to the fused Lasso (Tibshirani et al., 2005), which is widely used for analyzing spatial signals including brain signals (Xin et al., 2014; Ren et al., 2015). Both the fused Lasso and its generalizations (Takeuchi et al., 2015) operate on differences of scalars and are not suited for the differences of vectors we would need.

The generalized group fused Lasso (Pelckmans et al., 2005; Hocking et al., 2011) is a multivariate extension of the generalized fused Lasso, used for convex clustering problems. The key difference between the original convex clustering methods and our work is the exclusive regularization term, which enables us to select features in addition to clustering samples. Recently, a sparse convex clustering method has been proposed (Wang et al., 2016); its combination of feature-wise group regularization and sample-wise group fused regularization tends to select global features important for all samples, whereas we can choose features specific to each cluster and sample.

The network Lasso (Hallac et al., 2015a) is a general framework for solving regression problems having graph information, and our task can be categorized as a network Lasso problem. To our knowledge, ours is the first work to introduce feature-wise sparsity in the network Lasso problem. The additional central insight we bring is that by using the $\ell_{1,2}$ regularizer instead of the ℓ_1, we get non-obvious effects resulting in learning of different sparsity patterns for each local model, still borrowing strength according to the network.

Multi-task learning (Obozinski et al., 2006; Evgeniou and Pontil, 2007; Argyriou et al., 2008; Zhou et al., 2010; Sugiyama et al., 2014) is also relevant but does not solve our problem setup, since multi-task learning approaches assume the tasks (or clusters) to be known a priori. In contrast, in the localized lasso problem the clusters need to be found in addition to selecting features. It would be possible to first cluster based on the similarities in $\boldsymbol{R}$ and then apply multi-task learning for the resulting clusters. Convex multi-task learning methods which share inter-task similarity through low-rankness exist (Ando and Zhang, 2005; Jacob et al., 2009), but have not been designed to select a small number of features for each task. Recently, FORMULA, which both shares inter-task information using low-rankness, and enforces the low-rank matrices to be sparse, has been proposed (Xu et al., 2015), and we compare with it experimentally. However, FORMULA is a non-convex method, and it tends to perform poorly unless initialized very carefully. In particular in personalized medicine problems, since the data tend to be high-dimensional (i.e., the number of samples is much smaller than that of features), it tends to get trapped to poor local optima. Since our proposed method is *convex* and can effectively handle the joint feature selection and clustering problem, it is directly suited to such problem setups.

Local learning algorithms including local metric learning (Wang et al., 2012; Park et al., 2015) are also related to our work. By incorporating locality into metric learning, the embedding accuracy can be improved. However, existing local learning algorithms focus on learning local metrics, and not on local feature selection. Since we want to interpret the sets of features in local models, the current local metric algorithms are not sufficient for our task.

4 Experiments

In this section, we first illustrate our proposed method on synthetic data and then compare it with existing methods using a real-world dataset.

We compared our proposed method with Lasso (Tibshirani, 1996), Elastic Net (Zou and Hastie, 2005), FORMULA (Xu et al., 2015), and Network Lasso (Hallac et al., 2015a,b). For Lasso, Elastic Net, and FORMULA, we used the publicly available packages. For the network Lasso implementation, we set the regularization parameter to $\lambda_2 = 0$ in the localized Lasso. For supervised regression problems, all tuning parameters are determined by 3-fold nested cross validation. The experiments were run on a 3GHz AMD Opteron Processor with 48GB of RAM.

4.1 Synthetic experiments (high-dimensional regression)

We illustrate the behavior of the proposed method using a synthetic high-dimensional dataset.

We first generated the input variables as $x_{k,i} \sim \mathrm{Unif}(-1,1), k = 1,\ldots,10, i = 1,\ldots,30$. Then, we generated the corresponding output as

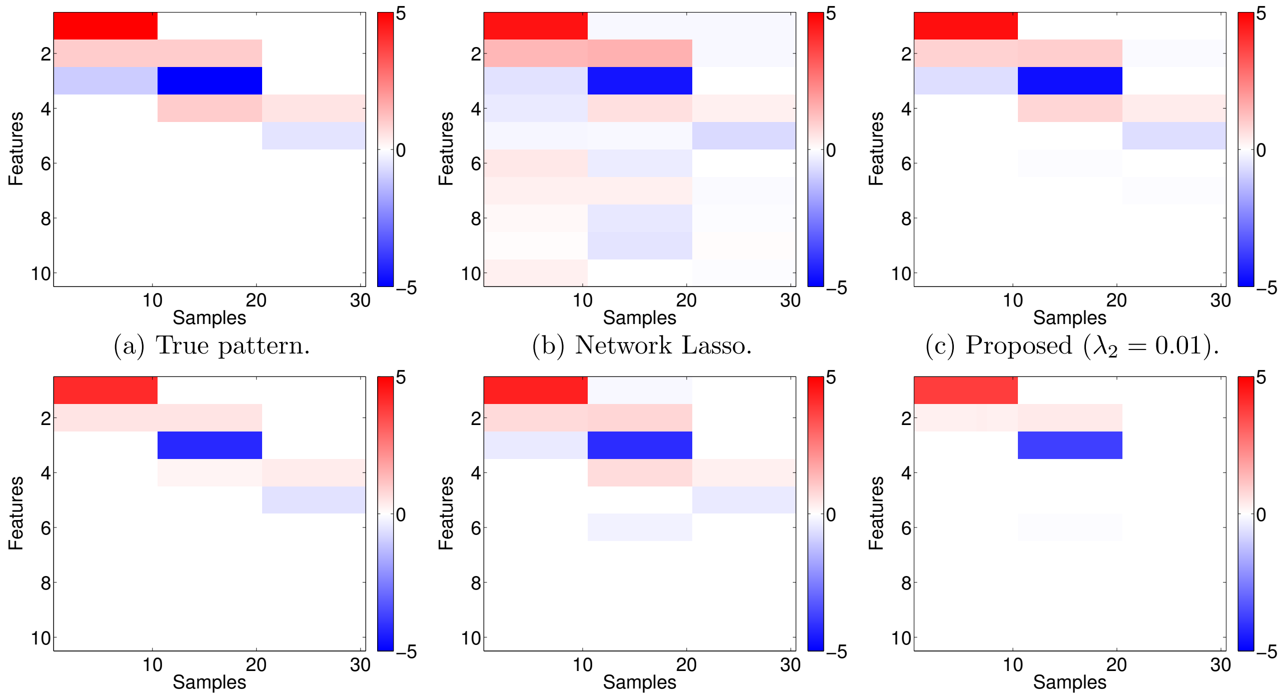
$$y_i = \begin{cases} 5x_{1,i} + x_{2,i} - x_{3,i} + 0.1e_i & (i = 1,\ldots,10) \\ x_{2,i} - 5x_{3,i} + x_{4,i} + 0.1e_i & (i = 11\ldots,20) \\ 0.5x_{4,i} - 0.5x_{5,i} + 0.1e_i & (i = 21\ldots30) \end{cases}, \quad (9)$$

where $x_{k,i}$ is the value of the k-th feature in the i-th sample and $e_i \sim N(0,1)$. In addition to the input-output pairs, we also randomly generated the link information matrix $\boldsymbol{R} \in \{0,1\}^{30\times30}$. In the link information matrix, only 40% of true links are observed.We experimentally set the regularization parameter for the proposed method to $\lambda_1 = 5$ and $\lambda_2 = \{0.01, 1, 10\}$. For the network Lasso, we used $\lambda_1 = 5$. Moreover, we compared the proposed method with the network Lasso + ℓ_1 regularizer, in which we used $\lambda_1 = 5$ and $\lambda_2 = \{0.05, 0.5\}$, where λ_2 is the regularization parameter for the ℓ_1 regularizer.

Figures 1(a)-(f) show the true coefficient pattern and the results of the learned coefficient matrices $\boldsymbol{W}$ by using the localized Lasso, the network Lasso, and[3] the network Lasso + ℓ_1. As can be seen, most of the unrelated coefficients of the proposed method are shrunk to zero. On the other hand, for the network Lasso, many unrelated coefficients take non-zero values. Thus, by incorporating the exclusive regularization in addition to the network regularization, we can learn sparse patterns in high-dimensional regression problems. Moreover, by setting the $\ell_{1,2}$ regularizer term to be stronger, we can obtain a sparser pattern within the $\boldsymbol{w}_i$. In contrast, Network Lasso + ℓ_1, which produces a similar pattern when the regularization is weak (Fig. 1(e)), shrinks many local models to zero if the regularization parameter λ_2 is large (Fig. 1(f)). This shows that the network Lasso + ℓ_1 is sensitive to the setting of the regularization parameter. Moreover, since we want to interpret features for each sample (or model), the $\ell_{1,2}$ norm is more suited than ℓ_1 for our tasks.

Figure 2 (a) shows the convergence of the proposed method. The objective score converges within 30 it-

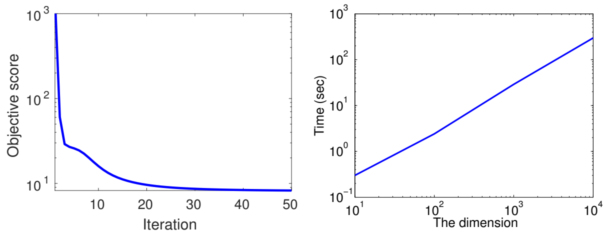
[3] Note that the combination of the network lasso and ℓ_1 is also new.



(a) True pattern. (b) Network Lasso. (c) Proposed ($\lambda_2 = 0.01$).

(d) Proposed ($\lambda_2 = 0.05$). (e) Network Lasso + ℓ_1 ($\lambda_2 = 0.1$). (f) Network Lasso + ℓ_1 ($\lambda_2 = 0.3$).

Figure 1: The learned coefficient matrix for the synthetic data for the different methods. Proposed = Localized Lasso. For Network Lasso + ℓ_1, we use the ℓ_1 regularizer instead of $\ell_{1,2}$ and $\lambda_2 \geq 0$ is the regularization parameter for the ℓ_1 term.



(a) Convergence. (b) Computation time.

Figure 2: (a): Objective score (Eq (2)) as a function of iteration ($d = 10^5$). (b): Computation time of the proposed method. We fixed the number of samples to $n = 100$ and the number of iterations to 10, and computed the results for $d = 10$, 10^2, 10^3, and 10^4.

erations for high-dimensional data ($d = 10^5$), without requiring tuning of step-size parameters as ADMM optimization would. Figure 2 (b) shows the computation time of the proposed method (implemented in MATLAB; the alternatives would perform similarly). As can be seen, the proposed method scales linearly with respect to the dimension.

4.2 Prediction in toxicogenomics (high-dimensional regression)

We evaluate our proposed method on the task of predicting toxicity of drugs on three cancer cell lines, based on gene expression measurements. The *Gene Expression* data includes the differential expression of 1106 genes in three different cancer types, for a collection of 53 drugs (i.e., $\boldsymbol{X}_l = [\boldsymbol{x}_1^{(l)}, \ldots, \boldsymbol{x}_{53}^{(l)}] \in \mathbb{R}^{1106 \times 53}, l = 1, 2, 3$). The learning data on *Toxicity* to be predicted contains three dose-dependent toxicity profiles of the corresponding 53 drugs over the three cancers (i.e., $\boldsymbol{Y}_l = [\boldsymbol{y}_1^{(l)}, \ldots, \boldsymbol{y}_{53}^{(l)}] \in \mathbb{R}^{3 \times 53}, l = 1, 2, 3$). The gene expression data of the three cancers (Blood, Breast and Prostate) comes from the Connectivity Map (Lamb et al., 2006) and was processed to obtain treatment vs. control differential expression. The toxicity screening data from the NCI-60 database (Shoemaker, 2006), summarizes the toxicity of drug treatments in three variables, GI50, LC50 and TGI, representing the 50% growth inhibition, 50% lethal concentration, and total growth inhibition levels. The data were confirmed to represent dose-dependent toxicity profiles for the doses used in the corresponding gene expression dataset.

In this experiment, we randomly split the data into training and test sets. The training set consisted of 48 drugs and the test set of 5 drugs. Moreover, we introduced a bias term in the proposed method (i.e., $[\boldsymbol{x}^\top \; 1]^\top \in \mathbb{R}^{d+1}$), and regularized the entire $\boldsymbol{w}_i$s in the network regularization term, and only $\boldsymbol{v}_i \in \mathbb{R}^{d-1}$ in the $\ell_{1,2}$ regularization term; here $\boldsymbol{w}_i = [\boldsymbol{v}_i^\top \; 1]^\top$. We

Table 1: Test root MSE on toxicogenomics data. The best method under sigfinicance level 5% (Wilcoxon signed-rank test) is reported in bold.

	Blood			Breast			Prostate			Average
	GI50	TGI	LC50	GI50	TGI	LC50	GI50	TGI	LC50	
Localized Lasso	1.030	0.622	0.529	1.129	0.627	0.562	1.297	0.518	0.539	**0.760**
Network Lasso	1.096	0.918	0.921	1.368	0.821	1.065	1.475	0.711	0.690	1.007
FORMULA	1.503	1.179	1.253	1.367	1.109	1.197	1.376	1.121	1.129	1.248
Lasso	1.201	1.006	0.514	1.435	0.879	0.560	1.455	0.763	0.523	0.926
Elastic Net	1.129	0.875	0.514	1.164	0.800	0.560	1.130	0.633	0.505	0.812
Kernel Regression	1.070	0.808	0.623	1.165	0.677	0.688	1.466	0.551	0.509	0.839

Table 2: The number of selected features (genes) on toxicogenomics data. For Localized Lasso, Network Lasso, FORMULA, and Elastic Net, we select features by checking $\|\boldsymbol{W}_{\cdot,i}\|_2 > 10^{-5}$, where $\boldsymbol{W}_{\cdot,i} \in \mathbb{R}^n$ is the i-th column of $\boldsymbol{W}$.

	Blood			Breast			Prostate			Average
	GI50	TGI	LC50	GI50	TGI	LC50	GI50	TGI	LC50	
Localized Lasso	32.7	33.4	92.5	92.6	125.9	58.2	35.1	53.9	43.4	63.4
Network Lasso	1039.6	1047.3	1052.2	1054.6	1051.5	1053.3	1060.5	1052.9	1053.1	1061.0
FORMULA	576.6	445.6	550.5	914.3	936.7	776.3	942.0	712.2	633.6	720.8
Lasso	29.6	12.0	1.0	12.0	1.9	1.0	12.5	4.4	3.8	8.7
Elastic Net	310.8	91.4	39.2	124.9	77.2	1.0	116.6	87.9	98.9	105.3

computed the graph information using the input $\boldsymbol{X}$ as

$$\boldsymbol{R} = \frac{\boldsymbol{S}^\top + \boldsymbol{S}}{2}, \; [\boldsymbol{S}]_{ij} = \begin{cases} 1 & \boldsymbol{x}_j \text{ is a 5-NN of } \boldsymbol{x}_i \\ 0 & \text{Otherwise} \end{cases}.$$

We repeated the experiments 20 times and report the average test RMSE scores in Table 1. We observed that the proposed localized Lasso outperforms state-of-the-art linear methods. Moreover, the proposed method also outperformed the *nonlinear* kernel regression with Gaussian kernel, which has high predictive power but cannot identify features. The kernel width of Gaussian kernel was tuned by cross-validation.

In Table 2, we report the number of selected features in each method. It is clear that the number of selected features in the proposed method is much smaller than that of the network Lasso. In some cases Lasso and Elastic net selected only one feature. This means that the features were shrunken to zero and only bias term remained. In summary, the proposed method is suited for producing interpretable sparse models in addition to having high predictive power.

4.3 Synthetic experiment (clustering)

Here, we illustrate the behavior of the proposed method for convex clustering using a synthetic dataset.

We generated the input variables as

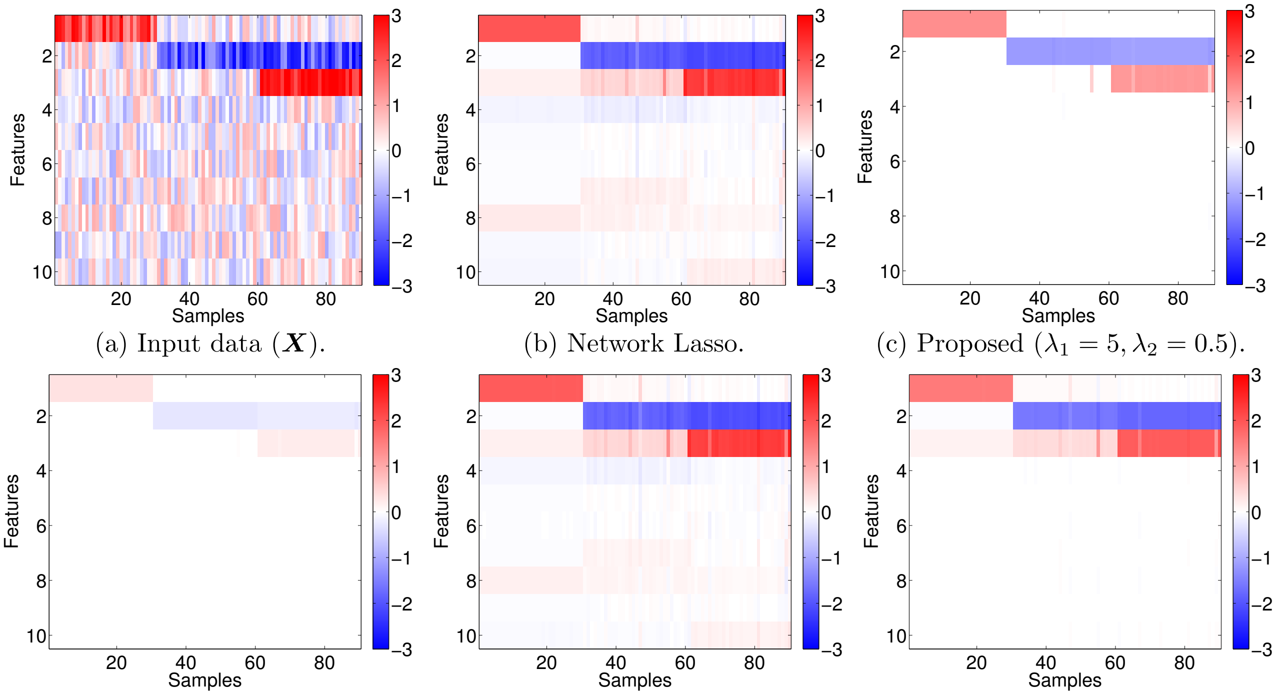
$$x_{ij} \sim \begin{cases} \text{Unif}(-3,-1) & (i=1, j=1,\ldots,30) \\ \text{Unif}(1,3) & (i=2, j=31,\ldots,60) \\ \text{Unif}(2,4) & (i=3, j=61,\ldots 90) \\ \text{Unif}(-1,1) & \text{Otherwise} \end{cases}, \quad (10)$$

where $x_{k,i}$ is the value of the k-th feature in the i-th sample. In this experiment, we compare our proposed method with the network lasso (i.e., $\lambda_2 = 0$) and sparse convex clustering (Wang et al., 2016), which employs the feature-wise group regularization (i.e., $\ell_{2,1}$-norm) in addition to the network regularization.

Figure 3 shows the learned coefficient matrices. As can be seen, the weight matrix of the network lasso is non-sparse. In contrast, it is possible to obtain the correct sparsity pattern using the proposed regularization. The sparse convex clustering method (Wang et al., 2016) can select the correct global set of features, but is less accurate in selecting the local feature sets than the proposed regularization.

4.4 Benchmark experiments (clustering)

We evaluated the proposed sparse convex clustering method on three benchmark datasets. We compared it with the convex clustering (Pelckmans et al., 2005; Hocking et al., 2011) and the sparse convex clustering + $\ell_{2,1}$ (Wang et al., 2016) algorithms. For all methods, we first ran the clustering algorithm which produced an estimate $\widehat{\boldsymbol{W}}$. Then, we applied an agglomerative clustering algorithm to threshold the $\widehat{\boldsymbol{W}}$ into a disjoint set of cluster indices. The clustering performance was evaluated by the *adjusted Rand index* (ARI) (Hubert and Arabie, 1985) between the estimated class labels and true labels. We ran each clustering method by multiple regularization parameter values and report the best ARI score. For all methods, the candidate lists of λ_1 and λ_2 were $\{0, 0.01, 0.1, 1, 2, \ldots, 15\}$ and



(a) Input data ($\boldsymbol{X}$).

(b) Network Lasso.

(c) Proposed ($\lambda_1 = 5, \lambda_2 = 0.5$).

(d) Proposed ($\lambda_1 = 5, \lambda_2 = 5$).

(e) Network Lasso + $\ell_{2,1}$ ($\lambda_1 = 5, \lambda_2 = 1$).

(f) Network Lasso + $\ell_{2,1}$ ($\lambda_1 = 5, \lambda_2 = 5$).

Figure 3: The learned coefficient matrix for the clustering data. For the Network Lasso + $\ell_{2,1}$, *feature-wise* group regularizer is used instead of $\ell_{1,2}$ and $\lambda_2 \geq 0$ is the regularization parameter for the $\ell_{2,1}$ term. (a): The input matrix $\boldsymbol{X}$. (b): Network Lasso ($\lambda_1 = 5, \lambda_2 = 0$). (c): Localized Lasso ($\lambda_1 = 5, \lambda_2 = 0.5$). (d): Localized Lasso ($\lambda_1 = 5, \lambda_2 = 10$). (e): Network Lasso + $\ell_{2,1}$ ($\lambda_1 = 5, \lambda_2 = 1$). (f): Network Lasso + $\ell_{2,1}$ ($\lambda_1 = 5, \lambda_2 = 5$). Note that the Network Lasso + $\ell_{2,1}$ is equivalent to sparse convex clustering (Wang et al., 2016).

Table 3: Experimental results (ARI) on real-world datasets. Larger ARI is better. K is the number of true clusters.

Data	d	n	K	Localized Lasso	Sparse Conv. Clust.	Conv. Clust.
LUNG	3312	203	5	**0.6316** ($\lambda_1 = 15, \lambda_2 = 1$)	0.5692 ($\lambda_1 = 9, \lambda_2 = 8$)	0.3715 ($\lambda_1 = 10$)
COIL20	1024	1440	20	**0.8048** ($\lambda_1 = 8, \lambda_2 = 0.1$)	0.7795 ($\lambda_1 = 15, \lambda_2 = 13$)	0.6991 ($\lambda_1 = 15$)
Lymphoma	4026	96	9	**0.6174** ($\lambda_1 = 5, \lambda_2 = 0.01$)	0.2673 ($\lambda_1 = 9, \lambda_2 = 0$)	0.2673 ($\lambda_1 = 9$)

$\{0, 0.01, 0.1, 1, 2, \ldots, 15\}$, respectively.

Table 3 shows the ARI results. As can be seen, the proposed method outperforms the existing state-of-the-art convex clustering methods for high-dimensional clustering problems. In other words, inducing the sample-wise exclusive sparsity is crucial to obtaining better clustering results.

5 Conclusion

In this paper, we proposed the localized Lasso method, which can produce sparse interpretable local models for high-dimensional problems. We proposed a simple yet efficient optimization approach by introducing structured sparsity: sample-wise network regularizer and sample-wise exclusive sparsity. Thanks to the structured sparsity, the proposed method had better regression performance with a smaller number of features than the alternatives. Moreover, the sparsity pattern in the learned models aids interpretation. We showed that the proposed method compares favorably with state-of-the-art methods on simulated data and molecular biological personalized medicine data.

Acknowledgment

MY was supported by the JST PRESTO program and partly supported by MEXT KAKENHI 16K16114. SK was supported by the Academy of Finland (Finnish Center of Excellence in Computational Inference COIN and grants 295503, 294238). We thank Dr. Suleiman A. Khan for providing the toxicity data set.

References

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853, 2005.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Theodoros Evgeniou and Massimiliano Pontil. Multitask feature learning. *NIPS*, 2007.

David Hallac, Jure Leskovec, and Stephen Boyd. Network lasso: Clustering and optimization in large graphs. In *KDD*, 2015a.

David Hallac, Christopher Wong, Steven Diamond, Rok Sosic, Stephen Boyd, and Jure Leskovec. Snapvx: A network-based convex optimization solver. *arXiv preprint arXiv:1509.06397*, 2015b.

Toby Dylan Hocking, Armand Joulin, Francis Bach, and Jean-Philippe Vert. Clusterpath an algorithm for clustering using convex fusion penalties. In *ICML*, 2011.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

Laurent Jacob, Jean-philippe Vert, and Francis R Bach. Clustered multi-task learning: A convex formulation. In *NIPS*, 2009.

Deguang Kong, Ryohei Fujimaki, Ji Liu, Feiping Nie, and Chris Ding. Exclusive feature learning on arbitrary structures via ℓ_{12}-norm. In *NIPS*, 2014.

Matthieu Kowalski. Sparse regression using mixed norms. *Applied and Computational Harmonic Analysis*, 27(3):303–324, 2009.

Justin Lamb et al. The connectivity map: Using gene-expression signatures to connect small molecules, genes, and disease. *Science*, 313(5795):1929–1935, 2006.

Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization. In *NIPS*, 2010.

Robert Nishihara, Laurent Lessard, Benjamin Recht, Andrew Packard, and Michael I Jordan. A general analysis of the convergence of ADMM. *arXiv:1502.02009*, 2015.

Guillaume Obozinski, Ben Taskar, and Michael Jordan. Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2, 2006.

Mijung. Park, Wittawat. Jitkrittum, Ahmad. Qamar, Zoltn. Szabó, Lars. Buesing, and Maneesh Sahani. Bayesian manifold learning: The locally linear latent variable model. In *NIPS*, 2015.

Kristiaan Pelckmans, Joseph De Brabanter, JAK Suykens, and B De Moor. Convex clustering shrinkage. In *PASCAL Workshop on Statistics and Optimization of Clustering*, 2005.

Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

Shaogang Ren, Shuai Huang, John Onofrey, Xenophon Papademetris, and Xiaoning Qian. A scalable algorithm for structured kernel feature selection. In *AISTATS*, 2015.

Robert H Shoemaker. The NCI60 human tumour cell line anticancer drug screen. *Nature Reviews Cancer*, 6(10):813–823, 2006.

Mahito Sugiyama, Chloé-Agathe Azencott, Dominik Grimm, Yoshinobu Kawahara, and Karsten M Borgwardt. Multi-task feature selection on multiple networks via maximum flows. In *SDM*, 2014.

Koh Takeuchi, Yoshinobu Kawahara, and Tomoharu Iwata. Higher order fused regularization for supervised learning with grouped parameters. In *ECML*. 2015.

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

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B*, 67(1):91–108, 2005.

Binhuan Wang, Yilong Zhang, Wei Sun, and Yixin Fang. Sparse convex clustering. *arXiv:1601.04586*, 2016.

Jun Wang, Alexandros Kalousis, and Adam Woznica. Parametric local metric learning for nearest neighbor classification. In *NIPS*, 2012.

Bo Xin, Yoshinobu Kawahara, Yizhou Wang, and Wen Gao. Efficient generalized fused lasso and its application to the diagnosis of alzheimer's disease. In *AAAI*, 2014.

Jianpeng Xu, Jiayu Zhou, and Pang-Ning Tan. FORMULA: FactORized MUlti-task LeArning for task discovery in personalized medical models. In *SDM*, 2015.

Yang Zhou, Rong Jin, and Steven CH Hoi. Exclusive lasso for multi-task feature selection. In *AISTATS*, 2010.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.