Nonconvex penalized regression using depth-based penalty functions: multitask learning and support union recovery in high dimensions

Subhabrata Majumdar, Snigdhansu Chatterjee

Abstract

We propose a new class of nonconvex penalty functions in the paradigm of multitask sparse penalized regression that are based on data depth-based inverse ranking. Focusing on a one-step sparse estimator of the coefficient matrix using local linear approximation of the penalty function, we derive its theoretical properties and provide the algorithm for its computation. For orthogonal design and independent responses, the resulting thresholding rule enjoys near-minimax optimal risk performance, similar to the adaptive lasso (**?**). A simulation study as well as real data analysis demonstrate its effectiveness compared to present methods that provide sparse solutions in multivariate regression.

Keywords: multivariate regression, nonconvex penalties, data depth, sparse regression, high-dimensional data

1 Introduction

Consider the multitask linear regression model:

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}$$

where $\mathbf{Y} \in \mathbb{R}^{n\times q}$ is the matrix of responses, and $\mathbf{E}$ is $n \times q$ the noise matrix: each row of which is drawn from $\mathcal{N}_q(\mathbf{0}_q, \mathbf{\Sigma})$ for a $q \times q$ positive definite matrix $\mathbf{\Sigma}$. We are interested in sparse estimates of the coefficient matrix $\mathbf{B}$ through solving penalized regression problems of the form

$$\min_{\mathbf{B}} \mathrm{Tr}\{(\mathbf{Y} - \mathbf{XB})^T(\mathbf{Y} - \mathbf{XB})\} + P_\lambda(\mathbf{B}). \tag{1}$$

The frequently studied classical linear model may be realized as a special of this for $q = 1$, where given a size-n sample of random responses $\mathbf{y} = (y_1, y_2, ..., y_n)^T$ and p-dimensional predictors $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)^T$, the above model may now be written as:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}; \quad \boldsymbol{\epsilon} = (\epsilon_1, ..., \epsilon_n)^T \sim \mathcal{N}_n(\mathbf{0}_n, \sigma^2\mathbf{I}_p).$$

Here the typical objective is to estimate the parameter vector $\boldsymbol{\beta}$ by minimizing $\sum_{i=1}^n \rho(y_i - \mathbf{x}_i^T\boldsymbol{\beta})$, for some loss function $\rho(.)$. Selecting important variables in this setup is often significant from an inferential and predictive perspective it is generally achieved by obtaining an estimate of $\boldsymbol{\beta}$ that minimizes a linear combination of the loss function and a 'penalty' term $P(\boldsymbol{\beta}) = \sum_{j=1}^p p(|\beta_j|)$, instead of only the loss function:

$$\hat{\boldsymbol{\beta}}_n = \arg\min_{\boldsymbol{\beta}} \left[\sum_{i=1}^n \rho(y_i - \mathbf{x}_i^T\boldsymbol{\beta}) + \lambda_n P(\boldsymbol{\beta})\right] \tag{2}$$

where λ_n is a tuning parameter depending on sample size. The penalty term is generally a measure of model complexity, providing a control against overfitting. Using a l_0 norm as penalty at this point, i.e. $p(z) = 1(z \neq 0)$, gives rise to the information criterion-based paradigm of statistical model selection, which goes back to the Akaike Information Criterion (AIC: **?**). Owing to the intractability of this problem due to an exponentially growing model space researchers have been exploring the use of functions that are non-differentiable at the origin as $p(.)$. This dates back to the celebrated LASSO

(**?**) which uses l_1 norm, adaptive LASSO (**?**) that reweights the coordinate-wise LASSO penalties based on Ordinary Least Square (OLS) estimate of $\boldsymbol{\beta}$, and **??** who used non-convex penalties to limit influence of large entries in the coefficient vector $\boldsymbol{\beta}$, resulting in improved estimation. Further, **?** and **?** provided efficient algorithms for computing solutions to the nonconvex penalized problems.

Two immediate extensions of the univariate-response penalized sparse regression paradigm are group-wise penalties and multivariate penalized regression. Applying penalties at variable group level instead of individual variables gives rise to Group LASSO (**?**). From an application perspective, this utilizes additional relevant information on the natural grouping of predictors: for example multiple correlated genes, or blockwise wavelet shrinkage (**?**). On the other hand, for multitask regression, penalizing at the coefficient matrix-level results in better estimation and prediction performance compared to performing q separate LASSO regressions to recover its corresponding columns (**?**).

Compared to sparse single-response regression where the penalty term can be broken down to elementwise penalties, in the multivariate response scenario we need to consider two levels of sparsity. The first level is recovering the set of predictors having non-zero effects on all the responses, as well as estimating their values. Assuming the coefficient matrix $\mathbf{B} \in \mathbb{R}^{p\times q}$ is made of rows $(\mathbf{b}_1, ..., \mathbf{b}_p)^T$, this means determining the set $\bigcup_k S_k$, with $S_k := \{k : b_{jk} \neq 0, j = 1, 2, ..., p\}$. This is called *support union recovery*, and is more effective in recovering non-zero elements of $\mathbf{B}$ compared to the naïve approach of performing q separate sparse regularized regressions and combining the results (**?**). The second level of sparsity is concerned with recovering non-zero elements *within* the non-zero rows obtained from the first step. The LARN algorithm addresses both of these issues.

Specifically, we perform support union recovery by considering penalties with row-wise decomposition: $P_\lambda(\mathbf{B}) = \sum_{j=1}^{p} p_\lambda(\|\mathbf{b}_j\|_2)$. In this paper, we shall concentrate on the scenario when $p_\lambda(\|\mathbf{b}_j\|_2)$ is a potentially nonconvex function of the row-norm. This automatically tempers the effects of large regression coefficients in the case of general q-dimensional response: this is not the case for methods based on l_1-norm penalization, e.g. Lasso. We also show that a simple corrective thresholding on elements of the first level row-sparse estimator ensures sparse recovery of within-row elements as well.

Our work provides a detailed treatment of using nonconvex penalties in the context of multivariate responses. We define the regularizing function in terms of *data depth* functions, which quantify the center-outward ranking of multivariate data (**?**). Inverse depth functions, or *peripherality functions* can be defined as some reverse ranking based on data depth, and we use such peripherality functions

as regularizers in this paper. In section 2 we discuss data-depth and illustrate some instances of peripherality functions, followed by detailed presentation of the LARN algorithm in section 3. Additional theoretical results in the orthogonal design case are discussed in section 4, and some simulation experiments are presented to compare the LARN algorithm with other methods in section 5. We present a data application of the LARN algorithm in section 6, followed by conclusions. The appendix contains proofs of the theoretical results.

2 Data depth and depth-based regularization

Given a data cloud or a probability distribution, a depth function is any real-valued function that measures the outlyingness of a point in feature space with respect to the data or its underlying distribution (figure 4 panel a). In order to formalize the notion of depth, we consider as data depth any scalar-valued function $D(\mathbf{x}, F_{\mathbf{X}})$ (where $\mathbf{x} \in \mathbb{R}^p$, and the random variable $\mathbf{X}$ has distribution F) that satisfies the following properties (**?**):

(P1) Affine invariance: $D(A\mathbf{x} + \mathbf{b}, F_{A\mathbf{X}+\mathbf{b}}) = D(\mathbf{x}, F_{\mathbf{X}})$ for any $p \times p$ non-singular matrix A and $p \times 1$ vector $\mathbf{b}$;

(P2) Maximality at center: When $F_{\mathbf{X}}$ has center of symmetry $\boldsymbol{\theta}$, $D(\boldsymbol{\theta}, F_{\mathbf{X}}) = \sup_{\mathbf{x}\in\mathbb{R}^p} D(\mathbf{x}, F_{\mathbf{X}})$. Here the symmetry can be central, angular or halfspace symmetry;

(P3) Monotonicity relative to deepest point: For any $p \times 1$ vector $\mathbf{x}$ and $\alpha \in [0, 1]$, $D(\mathbf{x}, F_{\mathbf{X}}) \leq D(\boldsymbol{\theta} + a(\mathbf{x} - \boldsymbol{\theta}))$;

(P4) Vanishing at infinity: As $\|\mathbf{x}\| \to \infty$, $D(\mathbf{x}, F_{\mathbf{X}}) \to 0$.

We incorporate measures of data depth as a row-level penalty function in 1. Specifically, we estimate the coefficient matrix $\mathbf{B}$ by solving the following constrained optimization problem:

$$\begin{aligned} \hat{\mathbf{B}} \;=\; & \arg\min_{\mathbf{B}} \Big[\mathrm{Tr}\{(\mathbf{Y} - \mathbf{XB})^T(\mathbf{Y} - \mathbf{XB})\} + \\ & \lambda_n \sum_{j=1}^{p} D^-(\mathbf{b}_j, F) \Big] \end{aligned} \tag{3}$$

where $D^-(\mathbf{x}, F)$ is an *inverse depth* function that measures the peripherality or outlyingness of the point $\mathbf{x}$ with respect to a fixed probability distribution F. Given a measure of data depth, any nonnegative-valued monotonically decreasing transformation on that depth function can be taken as a inverse depth function. Some examples include but are not limited to $D^-(\mathbf{x}, F) := \max_{\mathbf{x}} D(\mathbf{x}, F) - D(\mathbf{x}, F)$

and $D^-(\mathbf{x}, F) := \exp(-D(\mathbf{x}, F))$. This helps us obtain the nonconvex shape for our row-wise penalty function, where the penalty sharply increases for smaller entries inside the row but is bounded above for large values (see figure 4 panel b).

3 The LARN algorithm

3.1 Formulation

We first focus on the support union recovery problem in (3), starting with the first-order Taylor series approximation of $D^-(\mathbf{b}_j, F)$. At this point, we make the following assumptions:

(A1) The function $D^-(\mathbf{b}, F)$ is concave in $\mathbf{b}$, and continuously differentiable at every $\mathbf{b} \neq \mathbf{0}_q$ with bounded derivatives;

(A2) The distribution F is spherically symmetric.

Since F is spherically symmetric, due to affine invariance of $D(., F)$ hence $D^-(., F)$, depth at a point $\mathbf{b}$ becomes a function of $r = \|\mathbf{b}\|_2$ only. In this situation, several depth functions have closed-form expressions: e.g. when D is projection depth and F is a p-variate standard normal distribution, $D(\mathbf{b}_j, F) = c/(c + \|\mathbf{b}_j\|); c = \Phi^{-1}(3/4)$, while for halfspace depth and any F, $D(\mathbf{b}_j, F) = 1 - F_1(\|\mathbf{b}_j\|)$, F_1 being any univariate marginal of F. Hence, the computational burden of calculating depths becomes trivial.

Keeping the above in mind, we can write $D^-(\mathbf{b}_j, F) = p_F(r_j)$, and thus

$$\begin{aligned} P_{\lambda.F}(\mathbf{B}) &:= \lambda \sum_{j=1}^{p} p_F(r_j) \\ &\simeq \lambda \sum_{j=1}^{p} \left[p_F(r_j^*) + p_F'(r_j^*)(r_j - r_j^*) \right] \end{aligned} \tag{4}$$

for any $\mathbf{B}^*$ close to $\mathbf{B}$, and $r_j = \|\mathbf{b}_j\|_2, r_j^* = \|\mathbf{b}_j^*\|_2; j = 1, 2, ..., p$.

Thus, given a starting solution $\mathbf{B}^*$ close enough to the original coefficient matrix, $P_{\lambda.F}(\mathbf{B})$ is approximated by its conditional counterpart, say $P_{\lambda.F}(\mathbf{B}|\mathbf{B}^*)$. Following this a penalized maximum likelihood estimate for $\mathbf{B}$ can be obtained using the iterative algorithm below:

1. Take $\mathbf{B}^{(0)} = \mathbf{B}^* = (\mathbf{X}^T\mathbf{X})^-\mathbf{X}^T\mathbf{Y}$, i.e. the least square estimate of $\mathbf{B}$, set $k = 0$;

2. Calculate the next iterate by solving the penalized likelihood:

$$\mathbf{B}^{(k+1)} = \arg\min_{\mathbf{B}} \left[\mathrm{Tr}\left\{ (\mathbf{Y} - \mathbf{X}\mathbf{B}^{(k)})^T (\mathbf{Y} - \mathbf{X}\mathbf{B}^{(k)}) \right\} + \lambda \sum_{j=1}^{p} p'_F(r_j^{(k)}) r_j \right] \tag{5}$$

3. Continue until convergence.

We take the least square estimate as a starting value since it is within $O(1/\sqrt{n})$ distance of $\mathbf{B}$, and the upper bound on p'_F ensures that $P_{\lambda,F}(\mathbf{B}) = P_{\lambda,F}(\mathbf{B}|\mathbf{B}^*) + O(1/\sqrt{n})$ for fixed p. This algorithm approximates contours of the nonconvex penalty function using gradient planes at successive iterates, and is a multivariate generalization of the local linear approximation algorithm of **?**. We call this the *Local Approximation by Row-wise Norm* (LARN) algorithm.

LARN is a majorize-minimize (MM) algorithm where the actual objective function $Q(\mathbf{B})$ is being majorized by $R(\mathbf{B}|\mathbf{B}^{(k)})$, with

$$\begin{aligned} Q(\mathbf{B}) &= \mathrm{Tr}\left\{ (\mathbf{Y} - \mathbf{X}\mathbf{B})^T (\mathbf{Y} - \mathbf{X}\mathbf{B}) \right\} + P_{\lambda,F}(\mathbf{B}) \\ R(\mathbf{B}|\mathbf{B}^{(k)}) &= \mathrm{Tr}\left\{ (\mathbf{Y} - \mathbf{X}\mathbf{B})^T (\mathbf{Y} - \mathbf{X}\mathbf{B}) \right\} + P_{\lambda,F}(\mathbf{B}|\mathbf{B}^{(k)}) \end{aligned}$$

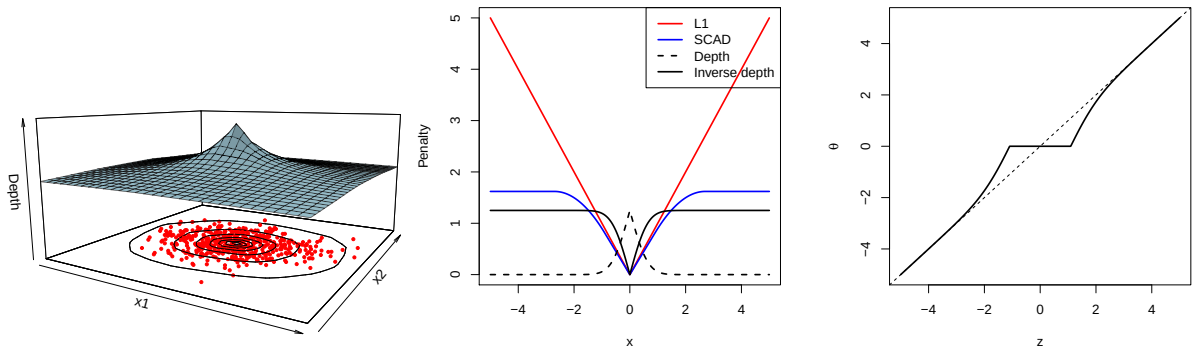
This is easy to see, because $Q(\mathbf{B}) - R(\mathbf{B}|\mathbf{B}^{(k)}) = \lambda \sum_{j=1}^{p} \left[p_F(r_j) - p_F(r_j^*) - p'_F(r_j^*)(r_j - r_j^*) \right]$. And since $p_F(.)$ is concave in its argument, we have $p_F(r_j) \leq p_F(r_j^*) + p'_F(r_j^*)(r_j - r_j^*)$. Thus $Q(\mathbf{B}) \leq R(\mathbf{B}|\mathbf{B}^{(k)})$. Also by definition $Q(\mathbf{B}) = R(\mathbf{B}^{(k)}|\mathbf{B}^{(k)})$.

Now notice that $\mathbf{B}^{(k+1)} = \arg\min_{\mathbf{B}} R(\mathbf{B}|\mathbf{B}^{(k)})$. Thus $Q(\mathbf{B}^{(k+1)}) \leq R(\mathbf{B}^{(k+1)}|\mathbf{B}^{(k)}) \leq R(\mathbf{B}^{(k)}|\mathbf{B}^{(k)}) = Q(\mathbf{B}^{(k)})$, i.e. the value of the objective function decreases in each iteration. At this point, we make the following assumption to enforce convergence to a local solution:

(A3) $Q(\mathbf{B}) = Q(M(\mathbf{B}))$ only for stationary points of Q, where M is the mapping from $\mathbf{B}^{(k)}$ to $\mathbf{B}^{(k+1)}$ defined in (5).

Since the sequence of penalized losses i.e. $\{Q(\mathbf{B}^{(k)}\}$ is bounded below (by 0) and monotone, it has a limit point, say $\hat{\mathbf{B}}$. Also the mapping $M(.)$ is continuous as ∇p_F is continuous. Further, we have $Q(\mathbf{B}^{(k+1)}) = Q(M(\mathbf{B}^{(k)})) \leq Q(\mathbf{B}^{(k)})$ which implies $Q(M(\hat{\mathbf{B}})) = Q(\hat{\mathbf{B}})$. It follows that $\hat{\mathbf{B}}$ is a local minimizer following assumption (A3).

Remark. Although the LARN algorithm guarantees convergence to a stationary point, that point may not be a local solution. However, local linear approximation has been found to be effective in approximating nonconvex penalties and obtaining oracle solutions for single-response regression (**?**) and support vector machines (**?**), and our method generalizes this concept for the multitask situation. We plan to elaborate on the presence and influence of saddle points in our scenario, in a future extended version of this work.



(a) (b) (c)

Figure 1: (a) Surface and contours of a data depth function for bivariate normal distribution; (b) Comparison of L1 and SCAD (**?**) penalty functions with depth at a scalar point: inverting the depth function helps obtain the nonconvex shape of the penalty function in the inverse depth; (c) Univariate thresholding rule for the LARN estimate (see section 4)

3.2 The one-step estimate and its oracle properties

Due to the row-wise additive structure of our penalty function, supports of each of the iterates in the LARN algorithm have the same set of singular points as the solution to the original optimization problem, say $\hat{\mathbf{B}}$. Consequently each of these iterates $\hat{\mathbf{B}}^{(k)}$ are capable of producing sparse solutions. In fact, the first iterate itself possesses oracle properties desirable of row-sparse estimates, namely consistent recovery of the support union $\bigcup_k S_k$ as well as the corresponding rows in $\mathbf{B}$. From our simulations there is little to differentiate between the first-step and multi-step estimates in terms of empirical efficiency. This is in line with the findings of **?** and **?**.

Given an initial solution $\mathbf{B}^*$, the first LARN iterate, say $\hat{\mathbf{B}}^{(1)}$, is a solution to the optimization

problem:

$$\arg\min_{\mathbf{B}} R(\mathbf{B}|\mathbf{B}^*) = \arg\min_{\mathbf{B}} \left[\mathrm{Tr}\left\{(\mathbf{Y}-\mathbf{XB})^T(\mathbf{Y}-\mathbf{XB})\right\} + \lambda \sum_{j=1}^{p} p'_F(r_j^{(k)}) r_j \right] \tag{6}$$

At this point, without loss of generality we assume that the true coefficient matrix $\mathbf{B}_0$ has the following decomposition: $\mathbf{B}_0 = (\mathbf{B}_{01}^T, \mathbf{0})^T, \mathbf{B}_1 \in \mathbb{R}^{p_1 \times q}$. Also denote the vectorized (i.e. stacked-column) version of a matrix $\mathbf{A}$ by $\mathrm{vec}(\mathbf{A})$. We are now in a position to to prove oracle properties of the one-step estimator in (6), in the sense that the estimator is able to consistently detect zero rows of $\mathbf{B}$ as well as estimate its non-zero rows for increasing sample size:

Theorem 3.1. *Assume that* $\mathbf{X}^T\mathbf{X}/n \to \mathbf{C}$ *for some positive definite matrix* $\mathbf{C}$*, and* $p'_F(r_j^*) = O((r_j^*)^{-s})$ *for* $1 \le j \le q$ *and some* $s > 0$*. Consider now a sequence of tuning parameters* λ_n *such that* $\lambda_n/\sqrt{n} \to 0$ *and* $\lambda_n n^{(s-1)/2} \to \infty$*. Then the following holds for the one-step estimate* $\hat{\mathbf{B}}^{(1)} = (\hat{\mathbf{B}}_{01}^T, \hat{\mathbf{B}}_{00}^T)^T$ *(with the component matrix having dimensions* $p_1 \times q$ *and* $p - p_1 \times q$*, respectively) as* $n \to \infty$*:*

- $vec(\hat{\mathbf{B}}_{00}) \to \mathbf{0}_{(p-p_1)q}$ *in probability;*
- $\sqrt{n}(vec(\hat{\mathbf{B}}_{01}) - vec(\mathbf{B}_{01})) \rightsquigarrow \mathcal{N}_p(\mathbf{0}_{p_1 q}, \mathbf{\Sigma} \otimes \mathbf{C}_{11}^{-1})$

where $\mathbf{C}_{11}$ *is the first* $p_1 \times p_1$ *block in* $\mathbf{C}$*.*

The assumption on the covariate matrix $\mathbf{X}$ is standard, and ensures uniqueness of the asymptotic covariance matrix of our estimator. Note that the restricted eigenvalue condition, which has been used in the literature to establish finite sample error bounds of penalized estimators **?** is a stronger version of this. With respect to the general framework of nonconvex penalized M-estimation in **?**, our modified form of p_F arising from assumption (A2) satisfies parts (i)-(iv) of Assumption 1 therein, and the conditions of theorem 3.1 adhere to part (v). Also note that the above oracle results depend on the assumption (A2), which simplifies depth as a function of the row-norm. We conjecture that similar oracle properties hold for weaker assumptions. From initial attempts into proving a broader result, we think it requires a more complex approach than the proof of theorem 3.1, and plan to work on this in the extended version of the work.

3.3 Recovering sparsity within a row

The set of variables with non-zero coefficients for each of the q univariate regressions may not be the same, and hence recovering the non-zero elements *within a row* is of interest as well. It turns out that consistent recovery at this level can be achieved by simply thresholding elements of the one-step estimate obtained in the preceding subsection. **?** have shown that a similar approach leads to consistent recovery of within-row supports in multivariate group lasso. The following result formalizes this in our scenario, provided that the non-zero signals in $\mathbf{B}$ are large enough:

Lemma 3.2. *Suppose the conditions of theorem 3.1 hold, and additionally all non-zero components of* $\mathbf{B}$ *have the following lower bound:*

$$|b_{jk}| \geq \sqrt{\frac{16 \log(qs)}{C_{min} n}}; \quad j \in S, 1 \leq k \leq q$$

where $C_{\min} > 0$ *is a lower bound for eigenvalues of* $\mathbf{C}_1$*. Then, for some constants* $c, c_0 > 0$*, the post-thresdolding estimator* $\mathbf{T}(\hat{\mathbf{B}}^{(1)})$ *defined by:*

$$t_{jk} = \begin{cases} 0 & \text{if } \hat{b}_{jk}^{(1)} \leq \sqrt{\frac{8 \log(q|\hat{S}|)}{C_{min} n}} \\ \hat{b}_{jk}^{(1)} & \text{otherwise} \end{cases}; \quad j \in \hat{S}, 1 \leq k \leq q$$

has the same set of non-zero supports within rows as $\mathbf{B}$ *with probability greater than* $1 - c_0 \exp(-cq \log s)$.

3.4 Computation

When the quantities $\mathbf{B}$ and $\mathbf{Y} - \mathbf{XB}$ are replaced with their corresponding vectorized versions, the optimization problem in (6) reduces to a weighted group lasso (**?**) setup, with group norms corresponding to l^2 norms of rows of $\mathbf{B}$ and inverse depths of corresponding rows of the initial estimate $\mathbf{B}^*$ acting as group weights. To solve this problem, we start from the following lemma, which gives necessary and sufficient conditions for the existence of a solution:

Lemma 3.3. *Given an initial value* $\mathbf{B}^*$*, a matrix* $\mathbf{B} \in \mathbb{R}^{p \times q}$ *is a solution to the optimization problem in (6) if and only if:*

1. $2\mathbf{x}_j^T(\mathbf{Y} - \mathbf{XB}) + \lambda p'_F(r_j^*)\mathbf{b}_j / r_j = \mathbf{0}_q$ *if* $\mathbf{b}_j \neq \mathbf{0}_q$*;*
2. $\|\mathbf{x}_j^T(\mathbf{Y} - \mathbf{XB})\|_2 \leq \lambda/2$ *if* $\mathbf{b}_j = \mathbf{0}_q$.

This lemma is a modified version of lemma 4.2 in chapter 4 of **?**, and can be proved in a similar fashion. Following the lemma, we can now use a block coordinate descent algorithm (**?**) to iteratively obtain $\hat{\mathbf{B}}^{(1)}$, given an appropriate starting value $\mathbf{B}_0$:

- Start with the OLS estimate $\mathbf{B}^*$, set $m = 1$ and $\hat{\mathbf{B}}^{(1,0)} = \mathbf{B}_0$;
- For $j = 1, 2, ..., p$ do:
  - If $\|\mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1,m-1)})\|_2 \leq (\lambda/2).p'_F(r_j^*)$, set $\hat{\mathbf{b}}_j^{(1,m)} = \mathbf{0}_q$;
  - Else update $\hat{\mathbf{b}}_j^{(1,m)}$ as

$$\hat{\mathbf{b}}_j^{(1,m)} = \frac{2\mathbf{s}_j^{(m-1)}}{2\|\mathbf{x}_j\|_2^2 + \lambda \frac{np'_F(r_j^*)}{\hat{r}_j^{(1,m-1)}} 1_{\hat{r}_j^{(1,m-1)} > 0}}$$

    where $\mathbf{s}_j^{(m-1)} = \mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}_{-j}^{(1,m-1)})$; $\hat{\mathbf{B}}_{-j}^{(1,m-1)}$ is the matrix obtained by replacing j^{th} row of $\hat{\mathbf{B}}^{(1,m-1)}$ by zeros.

- Set $m \leftarrow m + 1$, check for convergence and continue until convergence.
- Apply the thresholding from lemma 3.2 to recover within-row supports.

The parameter λ controls row-sparsity in $\hat{\mathbf{B}}^{(1)}$: a larger or smaller λ corresponding to higher number of zero rows in $\hat{\mathbf{B}}^{(1)}$, or an estimate closer to the ordinary least square solution, respectively. Since we use block coordinate descent, rows can drop in or out of the solution path, i.e. zero rows can reappear to be nonzero for a smaller λ.

Given a fixed λ, an easy choice of $\mathbf{B}_0$ is $\mathbf{B}^*$. We use k-fold cross-validation to choose the optimal λ. Also notice that in a sample setup the quantity $C_{\min}$ in lemma 3.2 is unknown. For this reason, we choose a best threshold for within-row sparsity through the above cross-validation procedure as well. Even though this means that the cross-validation has to be done over a two-dimensional grid, the thresholding step is actually done *after* estimation. Thus for any fixed λ only k models need to be calculated. Given a trained model for some value of λ we just cycle through the full range of thresholds to record their corresponding cross-validation errors. Also when optimizing over the range of tuning parameter values, say $\lambda_1 > ... > \lambda_m$, we use warm starts to speed up convergence. Denoting the solution corresponding to any tuning parameter λ as $\hat{\mathbf{B}}^{(1)}(\lambda)$, this means starting from the initial value $\mathbf{B}_0 = \hat{\mathbf{B}}^{(1)}(\lambda_{k-1})$ to obtain $\hat{\mathbf{B}}^{(1)}(\lambda_k)$, for $k = 2, ..., m$.

4 Orthogonal design and independent responses

We shed light on the workings of our penalty function by considering the simplified scenario when the predictor matrix $\mathbf{X}$ is orthogonal and all responses are independent. Independent responses make minimizing (3) equivalent to solving of q separate nonconvex penalized regression problems, while orthogonal predictors make the LARN estimate equivalent to a collection of coordinate-wise soft thresholding operators.

4.1 Thresholding rule

For the univariate thresholding rule, we are dealing with the simplified penalty function $p(|b_{jk}|) = D^-(b_{jk}, F)$, where D^- is a inverse depth function based on the univariate depth function D. In this case, depth calculation becomes simplified in exactly the same way as in subsection 3.1, only $|b_{jk}|$ replacing $\|\mathbf{b}_j\|$ therein, and $1 \leq k \leq q$.

Following **?**, a sufficient condition for the minimizer of the penalized least squares loss function

$$L(\theta; p_\lambda) = \frac{1}{2}(z-\theta)^2 + p_\lambda(|\theta|) \tag{7}$$

to be unbiased when the true parameter value is large is $p'_\lambda(|\theta|) = 0$ for large θ. In our formulation, this holds exactly when F has finite support, and approximately otherwise. A necessary condition for sparsity and continuity of the solution is $\min_{\theta \neq 0} |\theta| + p'_\lambda(|\theta|) > 0$. We ensure this by making a small assumption about the derivative of D^- (denoted by D_1^-):

(A4) $\lim_{\theta \to 0+} D_1^-(\theta, F) > 0$.

Subsequently we get the following thresholding rule as the solution to (7):

$$\begin{aligned} \hat{\theta}(F, \lambda) &= \text{sign}(z) \left[|z| - \lambda D_1^-(\theta, F) \right]_+ \\ &\simeq \text{sign}(z) \left[|z| - \lambda D_1^-(z, F) \right]_+ \end{aligned} \tag{8}$$

the approximation in the second step being due to **?**. A plot of the thresholding function in panel c of figure 4 demonstrates the unbiasedness and continuity properties of this estimator.

We note here that thresholding rules due to previously proposed nonconvex penalty functions can be obtained as special case of our rule. For example, the MCP penalty (**?**) corresponds to

$D_1^-(\theta, F) = |\theta| I(|\theta| < \lambda)$, while for the SCAD penalty (**?**):

$$D_1^-(\theta, F) = \begin{cases} c\lambda & \text{if } |\theta| < 2\lambda \\ \frac{c}{a-2}(a\lambda - |\theta|) & \text{if } 2\lambda \leq |\theta| < a\lambda \\ 0 & \text{if } |\theta| > a\lambda \end{cases}$$

with $c = 1/(2\lambda^2(a+2))$.

4.2 Minimax optimal performance

In the context of estimating the mean parameters μ_i of independent and identically distributed observations with normal errors: $z_i = \theta_i + v_i, v_i \sim N(0,1)$, the minimax risk is $2\log n$ times the ideal risk $R(\text{ideal}) = \sum_{i=1}^n \min(\theta_i^2, 1)$ (**?**). A major motivation of using lasso-type penalized estimators in linear regression is that they are able to approximately achieve this risk bound for large sample sizes (**??**). We now show that our thresholding rule in (8) also, in fact, replicates this performance.

Theorem 4.1. *Suppose the inverse depth function $D^-(., F)$ is twice continuously differentiable, except at the origin, with first and second derivatives bounded above by c_1 and c_2 respectively. Then for $\lambda = (\sqrt{.5\log n} - 1)/c_1$, we have*

$$\begin{aligned} R(\hat{\theta}(F,\lambda)) \quad &\leq \quad (2\log n - 3) \\ &\qquad \left[R(ideal) + \frac{c_1}{p_0(F)(\sqrt{.5\log n} - 1)} \right] \end{aligned} \tag{9}$$

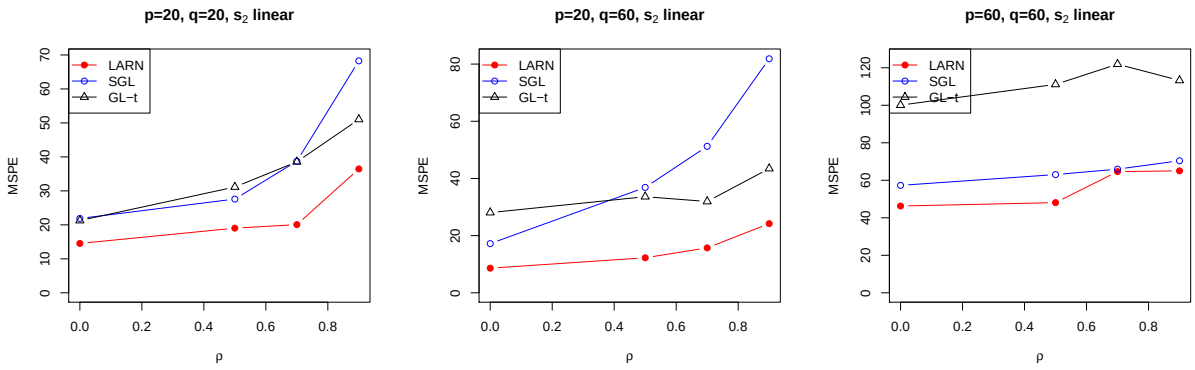
where $p_0(F) := \lim_{\theta \to 0+} D_1^-(\theta, F)$.

Following the theorem, we easily see that for large n the minimax risk of $\hat{\theta}(F, \lambda)$ approximately achieves the $2\log n$ multiple bound.

5 Simulation results

5.1 Methods and setup

We use the setup of **?** for our simulation study to compare the performance of LARN with other relevant methods. Specifically, we use performance metrics calculated after applying the following methods of predictor selection on simulated data for this purpose:



(a) (b) (c)

Figure 2: Mean squared testing errors for all three methods in different (p, q) settings

LARN: We use projection depth as our chosen depth function, take $D^-(\mathbf{x}, F) = \max_{\mathbf{x}} D(\mathbf{x}, F) - D(\mathbf{x}, F)$, and consider the set of tuning parameters $\lambda \in 10^{\{100, 99.5, \ldots, 0.5, 0\}}$ and use 5-fold cross-validation to get the optimal solution;

Sparse Graphical Lasso (SGL: ?): We adapt this method for single-response regression that uses group-level as well as element-level penalties on the coefficient vector in our scenario by taking vec($\mathbf{Y}$) as the response vector, $\mathbf{X} \otimes \mathbf{I}_q$ as the matrix of predictors, and then transforming back the pq-length coefficient estimate into a $p \times q$ matrix. Default options in the R package `SGL` are used while fitting the model;

Group Lasso with thresholding (GL-t): This has been proposed by **?**, and performs element-wise thresholding on a row-level group lasso estimator to get final estimate of $\mathbf{B}$. It can also be realized as a special case of LARN, with weights of all row-norms set as 1.

We generate rows of the model matrix $\mathbf{X}$ as $n = 50$ independent draws from $\mathcal{N}(\mathbf{0}_p, \mathbf{\Sigma}_X)$, where the positive $\mathbf{\Sigma}_X$ has an AR(1) covariance structure, with its $(i, j)^{\text{th}}$ element given by $0.7^{|i-j|}$. Rows of the random error matrix are generated as independent draws from $\mathcal{N}(\mathbf{0}_q, \mathbf{\Sigma})$: with $\mathbf{\Sigma}$ also having an AR(1) structure with correlation parameter $\rho \in \{0, 0.5, 0.7, 0.9\}$. Finally, to generate the coefficient matrix $\mathbf{B}$, we obtain the three $p \times q$ matrices: $\mathbf{W}$, whose elements are independent draws from $N(2, 1)$; $\mathbf{K}$, which has elements as independent draws from Bernoulli(0.3); and $\mathbf{Q}$ whose rows are made all 0 or all 1 according to p independent draws of another Bernoulli random variable with success probability 0.125. Following this, we multiply individual elements of these matrices (denoted by $(*)$) to obtain a sparse $\mathbf{B}$:

$$\mathbf{B} = \mathbf{W} * \mathbf{K} * \mathbf{Q}$$

Notice that the two levels of sparsity we consider: entire row and within-row, are imposed by the matrices $\mathbf{Q}$ and $\mathbf{K}$, respectively.

For a given value of ρ, we consider three settings of data dimensions for the simulations: (a) $p = 20, q = 20$, (b) $p = 20, q = 60$ and (c) $p = 60, q = 60$. Finally we replicate the full simulation 100 times for each set of (p, q, ρ).

5.2 Evaluation

To summarize the performance of an estimate matrix $\hat{\mathbf{B}}$ we use the following three performance metrics:

- *Mean Squared Testing Error (MSTE)*- Defined as $(1/pq)\text{Tr}\left[(\mathbf{Y}_{test} - \mathbf{X}_{test})(\mathbf{Y}_{test} - \mathbf{X}_{test})^T\right]$, with $(\mathbf{Y}_{test}, \mathbf{X}_{test})$ generated from the same simulation setup but using the same true $\mathbf{B}$;
- *True Positive Rate (TP)* - Defined as the proportion of non-zero entries in $\mathbf{B}$ detected as non-zero in $\hat{\mathbf{B}}$;
- *True Negative Rate (TN)* - Defined as the proportion of zero entries in $\mathbf{B}$ detected as zero in $\hat{\mathbf{B}}$.

A desirable estimate shall have low MSTE and high TP and TN proportions.

We summarize TP/TN rates of the three methods in table 4, and MSTE performances in figure 5. All across our method outperforms, GL-t, i.e. its unweighted version. Although its true negative detection is slightly worse than SGL, LARN makes up for that by a far superior signal detection ability (i.e. TP rate) for case (c), which has the highest feature and response space dimensions.

All simulations were run in parallel on 8 threads of an Intel Core i7 3770 3.4 GHz processor-run machine with 8 GB of RAM. As seen in table 2, LARN is the most computationally efficient of the three methods. This advantage becomes widest for case (c). Although SGL uses accelerated generalized gradient descent to speed up computation from block coordinate descent, its advantage is no longer observed in our case since we apply it on $\text{vec}(\mathbf{Y})$ and $\mathbf{X} \otimes \mathbf{I}_q$. Also note that GL-t is an unweighted version of LARN. In spite of that, LARN turns out to be faster than its unweighted counterpart: indicating faster convergence.

6 Real data example

We apply the LARN algorithm on a microarray dataset containing expression of several genes in the flowering plant *Arabidopsis thaliana* (**?**). Gene expressions are collected from $n = 118$ samples,

Table 1: Average true positive and true negative (TP/TN) rates for 3 methods, for $n = 50$ and AR1 covariance structure

ρ	GL-t	SGL	LARN
(a) $p = 20, q = 20$			
0.9	0.77/0.83	0.92/0.99	0.91/0.92
0.7	0.81/0.83	0.91/0.99	0.89/0.93
0.5	0.78/0.79	0.89/0.99	0.88/0.92
0.0	0.85/0.78	0.90/0.99	0.90/0.91
(b) $p = 20, q = 60$			
0.9	0.90/0.66	0.95/0.97	0.89/0.92
0.7	0.91/0.70	0.93/0.96	0.90/0.92
0.5	0.80/0.69	0.94/0.98	0.93/0.92
0.0	0.85/0.68	0.93/0.97	0.91/0.92
(c) $p = 60, q = 60$			
0.9	0.57/0.79	0.68/0.99	0.85/0.93
0.7	0.50/0.79	0.64/0.99	0.83/0.93
0.5	0.54/0.81	0.64/0.99	0.85/0.93
0.0	0.58/0.79	0.63/0.99	0.84/0.93

Table 2: Total runtimes in seconds for SGL and LARN algorithms for the three simulation settings

Setting	GL-t	SGL	LARN
(a)	332	490	209
(b)	676	52	328
(c)	4994	39760	3883

which are plants grown under different experimental conditions. We take the expressions of $q = 40$ genes in two pathways for biosynthesis of isoprenoid compounds, which are key compounds affecting plant metabolism as our multiple responses. Expressions of 795 other genes corresponding to 56 other pathways are taken as predictors.

Our objective here is to find out the extent of crosstalk between isoprenoid pathway genes and those in the other pathways. We apply LARN, as well as the two methods mentioned before, on the data and evaluate them based on predictive accuracy of 100 random splits with 90 training samples. All three methods have similar mean squared prediction error (MSPE) (LARN and GL-t have MSPE 0.45 and SGL has 0.44), but LARN produces more sparse solutions on average: the mean proportion of non-zero elements in the coefficient matrix are 0.15, 0.21 and 0.29 for LARN, GL-t and SGL, respectively. Focusing on the coefficient matrix estimated by LARN, we summarize the 10 largest coefficients (in absolute values) in table 5. We also visualize coefficients corresponding to genes in the 6 pathways in the table through a heatmap in figure 6.

All of the four largest coefficients correspond to interactions of one gene, DPPS2, with four dif-

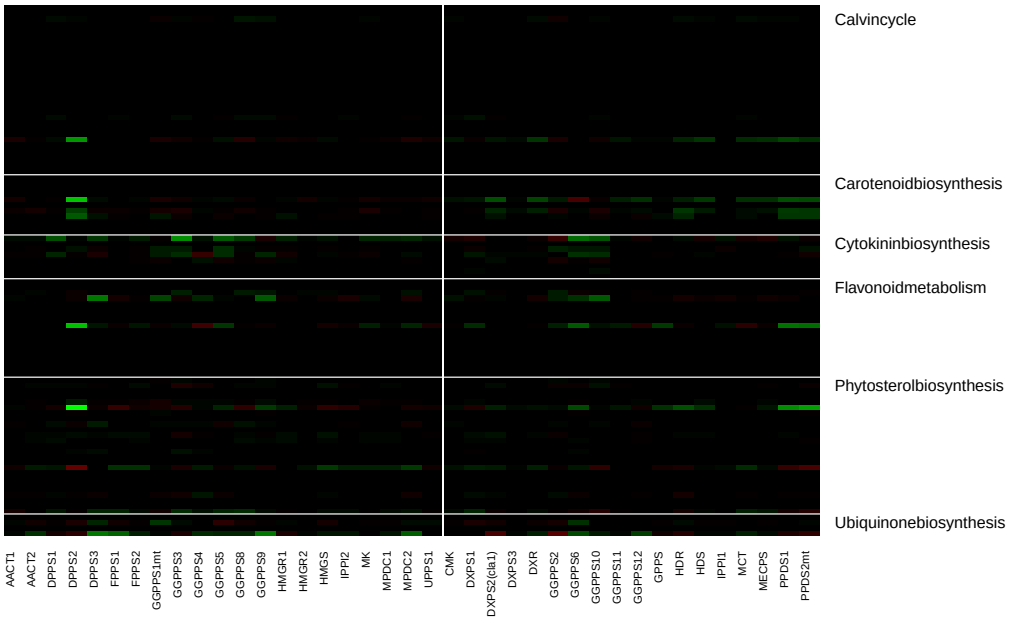


Figure 3: Estimated effects of different pathway genes on the activity of genes in Mevalonate and Non-mevalonate pathways (left and right of vertical line) in *A. thaliana*

Table 3: Top 10 gene-pathway connections in *A. thaliana* data found by LARN

Coeff	Gene	Pathway
0.18	DPPS2	Phytosterol biosynthesis
0.14	DPPS2	Carotenoid biosynthesis
0.14	DPPS2	Flavonoid metabolism
0.11	DPPS2	Calvin cycle
0.11	PPDS2mt	Phytosterol biosynthesis
0.10	GGPPS3	Cytokinin biosynthesis
0.10	PPDS1	Phytosterol biosynthesis
0.09	DPPS3	Flavonoid metabolism
0.09	DPPS3	Ubiquinone biosynthesis
0.09	GGPPS9	Ubiquinone biosynthesis

ferent pathways. Two of these pathways, Carotenoid and Phytosterol, directly use products from the isoprenoid pathways, and their connections with DPPS2 had been detected in **?**. The large Calvin Cycle-DPPS2 coefficient reveals that compounds synthesized in Carotenoid and Phytosterol pathways get used in Calvin Cycle. In the heatmap, Carotenoid biosynthesis seems to be connected mostly to the non-mevalonate pathway genes (right of the vertical line), while the activities of genes in Cytokinin and Ubiquinone synthesis pathways seem to be connected with those in the mevalonate pathway. These are consistent with the findings of **?**, **?** and **?**, respectively.

7 Conclusion

In the above sections we propose general nonconvex penalty functions based on data depth for performing support union recovery in multitask linear regression. Although several nonconvex penalties exist in the literature, the strength of our penalization scheme lies in its general nature and the instant extension to multitask regression. For the multitask case, we further show that a simple post-estimation thresholding recovers non-zero elements within rows in the coefficient matrix with good accuracy. It shares the weakness of all nonconvex penalties: small signals may go undetected or can be estimated in a biased fashion. Future studies in this direction include extending the setup to include generalized linear models, as well as exploring the use of more efficient algorithms for calculating the sparse solutions.

A Appendix

Proof of theorem 3.1. We shall prove a small lemma before going into the actual proof.

Lemma A.1. *For matrices* $\mathbf{K} \in \mathbb{R}^{l\times k}, \mathbf{L} \in \mathbb{R}^{l\times m}, \mathbf{M} \in \mathbb{R}^{m\times k}$,

$$\mathrm{Tr}(\mathbf{K}^T\mathbf{L}\mathbf{M}) = vec^T(\mathbf{K})(\mathbf{I}_k \otimes \mathbf{L})\, vec(\mathbf{M})$$

Proof of lemma A.1. From the property of Kronecker products, $(\mathbf{I}_k \otimes \mathbf{L})\,\mathrm{vec}(\mathbf{M}) = \mathrm{vec}(\mathbf{L}\mathbf{M})$. The lemma follows since $\mathrm{Tr}(\mathbf{K}^T\mathbf{L}\mathbf{M}) = \mathrm{vec}^T(\mathbf{K})\,\mathrm{vec}(\mathbf{L}\mathbf{M})$. □

Now, suppose $\mathbf{B} = \mathbf{B}_0 + \mathbf{U}/\sqrt{n}$, for some $\mathbf{U} \in \mathbb{R}^{p\times q}$, so that our objective function takes the form

$$\begin{aligned}
T_n(\mathbf{U}) &= \mathrm{Tr}\left[\left(\mathbf{Y} - \mathbf{X}\mathbf{B}_0 - \frac{1}{\sqrt{n}}\mathbf{X}\mathbf{U}\right)^T\left(\mathbf{Y} - \mathbf{X}\mathbf{B}_0 - \frac{1}{\sqrt{n}}\mathbf{X}\mathbf{U}\right)\right] \\
&\quad + \lambda_n \sum_{j=1}^{p} p'_F(r_j^*)\left\|\mathbf{b}_{0j} + \frac{\mathbf{u}_j}{\sqrt{n}}\right\|_2 \\
\Rightarrow T_n(\mathbf{U}) - T_n(\mathbf{0}_{p\times q}) &= \mathrm{Tr}\left[\frac{1}{n}\mathbf{U}^T\mathbf{X}^T\mathbf{X}\mathbf{U} - \frac{2}{\sqrt{n}}\mathbf{E}^T\mathbf{X}\mathbf{U}\right] \\
&\quad + \frac{\lambda_n}{\sqrt{n}} \sum_{j=1}^{p} p'_F(r_j^*)\left(\|\sqrt{n}\mathbf{b}_{0j} + \mathbf{u}_j\|_2 - \|\sqrt{n}\mathbf{b}_{0j}\|_2\right) \\
&= \mathrm{Tr}(\mathbf{V}_1 + \mathbf{V}_2) + V_3 \qquad (10)
\end{aligned}$$

Since $\mathbf{X}^T\mathbf{X}/n \to \mathbf{C}$ by assumption, we have $\mathrm{Tr}(\mathbf{V}_1) \to \mathrm{vec}^T(\mathbf{U})(\mathbf{I}_q \otimes \mathbf{C})\,\mathrm{vec}(\mathbf{U})$ using lemma A.1. Using the lemma we also get

$$\mathrm{Tr}(\mathbf{V}_2) = \frac{2}{\sqrt{n}}\mathrm{vec}^T(\mathbf{E})(\mathbf{I}_q \otimes \mathbf{X})\,\mathrm{vec}(\mathbf{U})$$

Now $\mathrm{vec}(\mathbf{E}) \sim \mathcal{N}_{nq}(\mathbf{0}_n, \mathbf{\Sigma}\otimes\mathbf{I}_q)$, so that $(\mathbf{I}_q \otimes \mathbf{X}^T)\,\mathrm{vec}(\mathbf{E})/\sqrt{n} \rightsquigarrow \mathbf{W} \equiv \mathcal{N}_{pq}(\mathbf{0}_{pq}, \mathbf{\Sigma}\otimes\mathbf{C})$ using properties of Kronecker products and Slutsky's theorem.

Let us look at V_3 now. Denote by V_{3j} the j^{th} summand of V_3. Now there are two scenarios. Firstly, when $\mathbf{b}_{0j} \neq \mathbf{0}_q$, we have $p'_F(r_j^*) \xrightarrow{P} p'_F(r_{0j})$. Since $\lambda_n/\sqrt{n} \to 0$, this implies $V_{3j} \xrightarrow{P} 0$ for any fixed $\mathbf{u}_j$. Secondly, when $\mathbf{b}_{0j} = \mathbf{0}_q$, we have

$$V_{3j} = \lambda_n n^{(s-1)/2}.(\sqrt{n}r_j^*)^{-s}.\frac{p'_F(r_j^*)\|\mathbf{u}_j\|_2}{(r_j^*)^{-s}}$$

We now have $\mathbf{b}_j^* = O_p(1/\sqrt{n})$, and also each term of the gradient vector is $D^-((r_j^*)^{-s})$ by assumption. Thus $V_{3j} = O_P(\lambda_n n^{(s-1)/2}\|\mathbf{u}_j\|_2)$. By assumption, $\lambda_n n^{(s-1)/2} \to \infty$ as $n \to \infty$, so $V_{3j} \xrightarrow{P} \infty$ unless $\mathbf{u}_j = \mathbf{0}_q$, in which case $V_{3j} = 0$.

Accumulating all the terms and putting them into (10) we see that

$$T_n(\mathbf{U}) - T_n(\mathbf{0}_{p\times q}) \rightsquigarrow \begin{cases} \mathrm{vec}^T(\mathbf{U}_1)[(\mathbf{I}_q \otimes \mathbf{C}_{11})\,\mathrm{vec}(\mathbf{U}_1) - 2\mathbf{W}_1] & \text{if } \mathbf{U}_0 = \mathbf{0}_{(p-p_1)q} \\ \infty & \text{otherwise} \end{cases} \tag{11}$$

where rows of $\mathbf{U}$ are partitioned into $\mathbf{U}_1$ and $\mathbf{U}_0$ according to the zero and non-zero rows of $\mathbf{B}_0$, respectively, and the random variable $\mathbf{W}$ is partitioned into $\mathbf{W}_1$ and $\mathbf{W}_0$ according to zero and non-zero *elements* of $\mathrm{vec}(\mathbf{B}_0)$. Applying epiconvergence results of **?** and **?** we now have

$$\mathrm{vec}(\hat{\mathbf{U}}_{1n}) \quad \rightsquigarrow \quad (\mathbf{I}_q \otimes \mathbf{C}_{11}^{-1})\mathbf{W}_1 \tag{12}$$

$$\mathrm{vec}(\hat{\mathbf{U}}_{0n}) \quad \rightsquigarrow \quad \mathbf{0}_{(p-p_1)q} \tag{13}$$

where $\hat{\mathbf{U}}_n = (\hat{\mathbf{U}}_{1n}^T, \hat{\mathbf{U}}_{0n}^T)^T := \arg\min_{\mathbf{U}} T_n(\mathbf{U})$.

The second part of the theorem, i.e. asymptotic normality of $\sqrt{n}(\mathrm{vec}(\hat{\mathbf{B}}_{1n}) - \mathrm{vec}(\hat{\mathbf{B}}_{1n})) = \hat{\mathbf{U}}_{1n}$ follows directly from (12). It is now sufficient to show that $P(\hat{\mathbf{b}}_j^{(1)} \neq \mathbf{0}_q | \mathbf{b}_{0j} = \mathbf{0}_q) \to 0$ to prove the oracle consistency part. For this notice that KKT conditions of the optimization problem for the one-step estimate indicate

$$2\mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)}) = -\lambda_n p'_F(r_j^*)\frac{\mathbf{b}_j^{(1)}}{r_j^{(1)}} \quad \Rightarrow \quad \frac{2\mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)})}{\sqrt{n}} = -\frac{\lambda_n p'_F(r_j^*)}{\sqrt{n}}.\frac{\mathbf{b}_j^{(1)}}{r_j^{(1)}} \tag{14}$$

for any $1 \leq j \leq p$ such that $\hat{\mathbf{b}}_j^{(1)} \neq \mathbf{0}_q$. Since $p'_F(r_j^*) = D^-((r_j^*)^{-s}) = O_P(\|(\mathbf{b}_{0j} + 1/\sqrt{n}\|^{-s})$ and $\lambda_n n^{(s-1)/2} \to \infty$, the right hand side goes to $-\infty$ in probability if $\mathbf{b}_{0j} = \mathbf{0}_q$. As for the left-hand side, it can be written as

$$\frac{2\mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)})}{\sqrt{n}} = \frac{2\mathbf{x}_j^T\mathbf{X}.\sqrt{n}(\mathbf{B}_0 - \hat{\mathbf{B}}^{(1)})}{n} + \frac{2\mathbf{x}_j^T\mathbf{E}}{\sqrt{n}} = \frac{2\mathbf{x}_j^T\mathbf{X}\hat{\mathbf{U}}_n}{n} + \frac{2\mathbf{x}_j^T\mathbf{E}}{\sqrt{n}}$$

Our previous derivations show that vectorized versions of $\hat{\mathbf{U}}_n$ and $\mathbf{E}$ have asymptotic and exact mul-

tivariate normal distributions, respectively. Hence

$$P\left[\hat{\mathbf{b}}_j^{(1)} \neq \mathbf{0}_q | \mathbf{b}_{0j} = \mathbf{0}_q\right] \leq P\left[2\mathbf{x}_j^T(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)}) = -\lambda_n p_F'(r_j^*)\frac{\mathbf{b}_j^{(1)}}{r_j^{(1)}}\right] \to 0$$

□

Proof of theorem 3.2. See the proof of corollary 2 of **?**in Appendix A therein. Our proof follows the same steps, only replacing $\mathbf{\Sigma}_{SS}$ with $\mathbf{\Sigma} \otimes \mathbf{C}_{11}$.

□

Proof of Lemma 4.1. We broadly proceed in a similar fashion as the proof of Theorem 3 in **?**. As a first step, we decompose the mean squared error:

$$\begin{aligned} E[\hat{\theta}(F,\lambda) - \theta]^2 &= E[\hat{\theta}(F,\lambda) - z]^2 + E(z-\theta)^2 + 2E[\hat{\theta}(F,\lambda)(z-\theta)] - 2E[z(z-\theta)] \\ &= E[\hat{\theta}(F,\lambda) - z]^2 + E\left[\frac{d\hat{\theta}(F,\lambda)}{dz}\right] - 1 \end{aligned}$$

by applying Stein's lemma (**?**). We now use Theorem 1 of **?** to approximate $\hat{\theta}(F,\lambda)$ in terms of y only. By part 2 of the theorem,

$$\hat{\theta}(F,\lambda) = \begin{cases} 0 & \text{if } |z| \leq \lambda p_0(F) \\ z - \text{sign}(z).\lambda D_1^-(\hat{\theta}(F,\lambda), F) & \text{if } |z| > \lambda p_0(F) \end{cases} \tag{15}$$

Moreover, applying part 5 of the theorem,

$$\hat{\theta}(F,\lambda) = z - \text{sign}(z).\lambda D_1^-(z,F) + o(D_1^-(z,F)) \tag{16}$$

for $|z| > \lambda p_0(F)$. Thus we get

$$[\hat{\theta}(F,\lambda) - z]^2 = \begin{cases} z^2 & \text{if } |z| \leq \lambda p_0(F) \\ \lambda^2 D_1^-(z,F)^2 + k_1(|z|) & \text{if } |z| > \lambda p_0(F) \end{cases} \tag{17}$$

and

$$\frac{d\hat{\theta}(F,\lambda)}{dz} = \begin{cases} 0 & \text{if } |z| \le \lambda p_0(F) \\ 1 + \lambda D_2^-(z,F) + k_1'(|z|) & \text{if } |z| > \lambda p_0(F) \end{cases} \tag{18}$$

where $k_1(|z|) = o(|z|)$, and $D_2^-(z,F) = d^2D^-(z,F)/dz^2$. Thus

$$\begin{aligned} E[\hat{\theta}(F,\lambda) - \theta]^2 &= E[z^2 1_{|z|\le\lambda p_0(F)}] + E\left[\left(\lambda^2 D_1^-(|z|,F)^2 + 2\lambda D_2^-(|z|,F) + 2 + \right.\right. \\ &\quad \left.\left. k_1(|z|) + k_1'(|z|)\right) 1_{|z|>\lambda p_0(F)}\right] - 1 \end{aligned} \tag{19}$$

Now

$$\begin{aligned} k_1(|z|) &= \lambda^2\left[D_1^-(z,F)^2 - D_1^-(\hat{\theta}(F,\lambda),F)^2\right] \quad \le \quad \lambda^2 c_1^2, \text{ and} \\ |k_1'(|z|)| &= \lambda\left|D_2^-(z,F) - \frac{dD_1^-(\hat{\theta}(F,\lambda),F)}{dz}\right| \quad \le \quad 2\lambda c_2 \end{aligned}$$

Substituting these in (19) above we get

$$\begin{aligned} E[\hat{\theta}(F,\lambda) - \theta]^2 &\le \lambda^2 p_0(F)^2 P[|z| \le \lambda p_0(F)] + E\left[\left(\lambda^2 f^2(|z|) + 2\lambda D_2^-(z,F)\right) 1_{|z|>\lambda p_0(F)}\right] \\ &\quad + \lambda^2 c_1^2 + 2\lambda c_2 + 1 \\ &\le 2\lambda^2 c_1^2 + 4\lambda c_2 + 1 \\ &\le 4\lambda^2 c_1^2 + 8\lambda c_2 + 1 \end{aligned} \tag{20}$$

Adding and subtracting $z^2 1_{|z|>\lambda p_0(F)}$ to the first and second summands of (19) above, we also have

$$\begin{aligned} E[\hat{\theta}(F,\lambda) - \theta]^2 &= Ez^2 + E\left[\left(\lambda^2 D_1^-(z,F)^2 + 2\lambda D_2^-(z,F) + 2 - y^2 + \lambda^2 c_1^2 + 2\lambda c_2\right) 1_{|z|>\lambda p_0(F)}\right] - 1 \\ &\le (2\lambda^2 c_1^2 + 4\lambda c_2) P[|z| > \lambda p_0(F)] + \theta^2 \end{aligned} \tag{21}$$

Following **?**, $P[|z| > \lambda p_0(F)] \le 2q(\lambda p_0(F)) + 2\theta^2$, with $q(x) = \exp[-x^2/2]/(\sqrt{2\pi}x)$. Thus

$$\begin{aligned} E[\hat{\theta}(F,\lambda) - \theta]^2 &\le 2(2\lambda^2 c_1^2 + 4\lambda c_2)[q(\lambda p_0(F)) + \theta^2] + \theta^2 \\ &\le (4\lambda^2 c_1^2 + 8\lambda c_2 + 1)[q(\lambda p_0(F)) + \theta^2] \end{aligned} \tag{22}$$

Combining this with (20) we get

$$E[\hat{\theta}(F,\lambda) - \theta]^2 \le [4(\lambda c_1 + 1)^2 - 3][q(\lambda p_0(F)) + \min(\theta^2, 1)] \tag{23}$$

assuming without loss of generality that $c_1 \geq c_2$. Since $R(\text{ideal}) = \min(\theta^2, 1)$ and $q(x) \leq (\sqrt{2\pi}x)^{-1} < 1/x$, we have the needed. $\square$

References

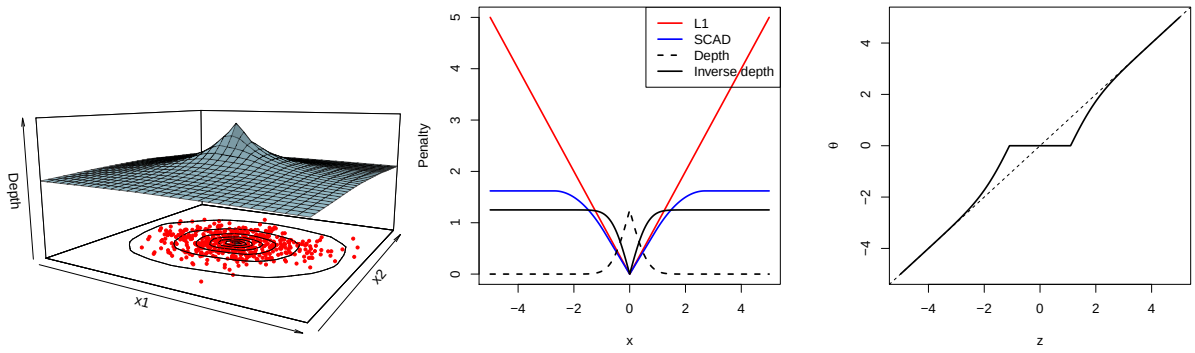
Figure and tables

ρ	GL-t	SGL	LARN
$p = 20, q = 20$			
0.9	0.90/0.99	0.95/0.99	0.97/0.97
0.7	0.91/0.99	0.96/0.99	0.96/0.97
0.5	0.93/0.99	0.96/0.99	0.98/0.98
0.0	0.92/0.99	0.96/0.99	0.96/0.97
$p = 20, q = 60$			
0.9	0.71/0.99	0.67/0.99	0.94/0.97
0.7	0.73/0.99	0.66/0.99	0.94/0.98
0.5	0.74/0.99	0.69/0.99	0.94/0.98
0.0	0.73/0.99	0.74/0.99	0.92/0.98
$p = 60, q = 60$			
0.9	0.59/0.99	0.71/0.99	0.91/0.98
0.7	0.62/0.99	0.73/0.99	0.92/0.98
0.5	0.65/0.99	0.73/0.99	0.91/0.98
0.0	0.62/0.99	0.73/0.99	0.92/0.98

Table 4: Average true positive and true negative (TP/TN) rates for 3 methods, for $n = 50$ and AR1 covariance structure

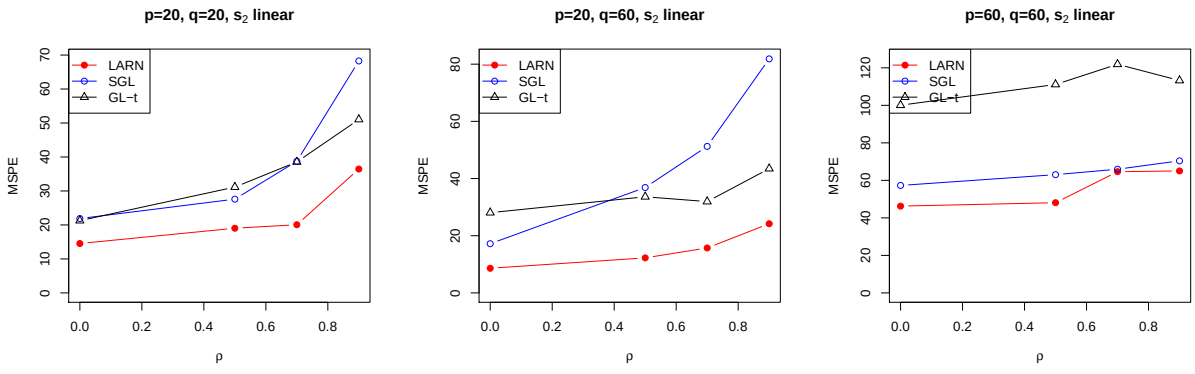
Coeff	Gene	Pathway
0.18	DPPS2	Phytosterol biosynthesis
0.14	DPPS2	Carotenoid biosynthesis
0.14	DPPS2	Flavonoid metabolism
0.11	DPPS2	Calvin cycle
0.11	PPDS2mt	Phytosterol biosynthesis
0.10	GGPPS3	Cytokinin biosynthesis
0.10	PPDS1	Phytosterol biosynthesis
0.09	DPPS3	Flavonoid metabolism
0.09	DPPS3	Ubiquinone biosynthesis
0.09	GGPPS9	Ubiquinone biosynthesis

Table 5: Top 10 gene-pathway connections in *A. thaliana* data found by LARN



(a) (b) (c)

Figure 4: (a) Surface and contours of a data depth function for bivariate normal distribution; (b) Comparison of L1 and SCAD penalty functions with depth at a scalar point: inverting the depth function helps obtain the nonconvex shape of the penalty function in the inverse depth; (c) Univariate thresholding rule for the LARN estimate (see section 4)



(a) (b) (c)

Figure 5: Mean absolute Estimation Errors for all three methods in different (p, q) settings

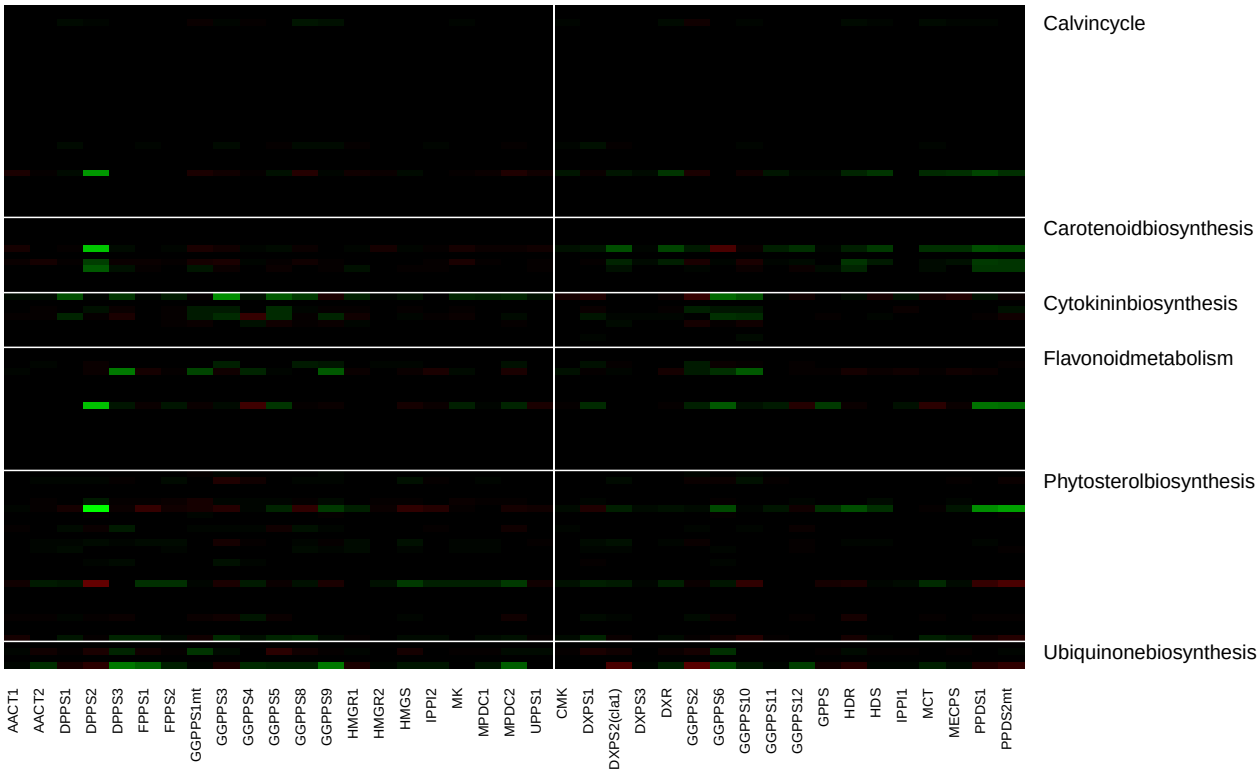


Figure 6: Estimated effects of different pathway genes on the activity of genes in Mevalonate and Non-mevalonate pathways (left and right of vertical line) in *A. thaliana*