Sparse Linear Models

Jingwen Zhang

Sun Yat-sen University, School of Mathematics

Variable selection

- Problem formulation and applications
- Different sparsity models
- Algorithms

l_1 regularization: basics

- Why does l_1 regularization yield sparse solutions?
- Optimality conditions for lasso
- Comparison of least squares, lasso, ridge and subset selection
- Regularization path
- Model selection
- Bayesian inference for linear models with Laplace priors

l_1 regularization: algorithms

- Coordinate descent
- LARS and other homotopy methods
- Proximal and gradient projection methods
- EM for lasso

13.2.1 Problem formulation

Let $\theta^* \in R^d$ be an unknown vector, referred to as the regression vector. Suppose that we observe a vector $y \in R^n$ and a matrix $X \in R^{n*d}$ that are linked via the standard linear model

$$y = X\theta^* + w$$

Where $w \in R^n$ is a vector of noise variables.

This model can also be written in a scalarized form: for each index $i = 1, 2, \ldots, n$, we have $y_i =< x_i, \theta^* > +w_i$. The quantity $< x_i, \theta^* >:= \sum_{j=1}^{d} x_{ij}\theta_j^*$ denotes the usual Euclidean inner product between the vector $xi \in R^d$ of predictors (or covariates), and the regression vector $\theta^* \in R^d$.

13.2.2 Different sparsity models

When $d > n$, it is impossible to obtain any meaningful estimates of θ^*.

One of the simplest kinds of structure in a linear model is a *hard sparsity* assumption, meaning that the set

$$S(\theta^*) := \{j \in \{1, 2, \ldots, d\} \mid \theta^* \neq 0\}$$

known as the *support set* of θ^*, has cardinality $s := |S(\theta^*)|$ substantially smaller than d.

Assuming that the model is exactly supported on *s* coefficients may be overly restrictive, in which case it is also useful to consider various relaxations of hard sparsity, which leads to the notion of weak sparsity.

13.2.2 Different sparsity models

There are different ways in which to formalize such an idea, one way being via the "ℓ_q −"norms". For a parameter $q \in [0, 1]$ and radius $R_q > 0$, consider the set

$$\mathbb{B}_q(R_q) = \left\{ \theta \in \mathbb{R}^d \,\middle|\, \sum_{j=1}^{d} |\theta_j|^q \leq R_q \right\}.$$

It is known as the " q-ball of radius R_q.

For $q \in [0, 1)$, it is not a ball in the strict sense of the word, since it is a non-convex set.

13.2.2 Different sparsity models

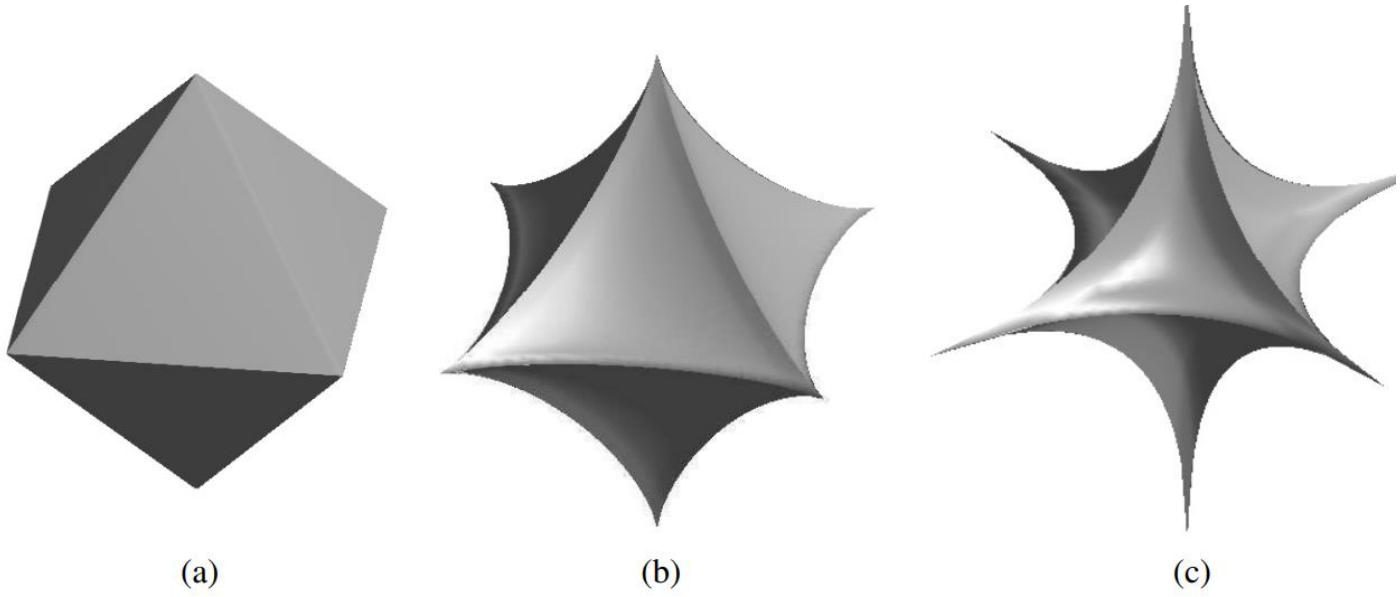


Figure 7.1 Illustrations of the ℓ_q-"balls" for different choices of the parameter $q \in (0, 1]$. (a) For $q = 1$, the set $\mathbb{B}_1(R_q)$ corresponds to the usual ℓ_1-ball shown here. (b) For $q = 0.75$, the ball is a non-convex set obtained by collapsing the faces of the ℓ_1-ball towards the origin. (c) For $q = 0.5$, the set becomes more "spiky", and it collapses into the hard sparsity constraint as $q \to 0^+$. As shown in Exercise 7.2(a), for all $q \in (0, 1]$, the set $\mathbb{B}_q(1)$ is star-shaped around the origin.

13.2.3 Algorithms

Algorithms - Greedy search

If we use the l_0 -regularized objective in Equation $f(w) = ||y - Xw||_2^2 + \lambda||w||_0$, we can exploit properties of least squares to derive various efficient greedy forwards search methods, some of which we summarize below.

- Single best replacement

Since we are expecting a sparse solution,
1. start with the empty set, $\gamma = 0$.
2. at each step, we define the neighborhood of the current model to be all models than can be reached by flipping a single bit of γ
3. continue adding or removing until no improvement is possible

13.2.3 Algorithms

- Orthogonal least squares

If we set $\lambda = 0$ in Equation 13.27, so there is no complexity penalty, there will be no reason to perform deletion steps. In this algorithm, we start with the empty set and add the best feature at each step. The error will go down monotonically with $||\gamma||_0$. We can pick the next best feature j^* to add to the current set γ^t by solving

$$j^* = \arg\min_{j \notin \boldsymbol{\gamma}_t} \min_{\mathbf{w}} ||\mathbf{y} - (\mathbf{X}_{\boldsymbol{\gamma}_t \cup j})\mathbf{w}||^2$$

13.2.3 Algorithms

- Orthogonal matching pursuits

Orthogonal least squares is somewhat expensive. A simplification is to "freeze" the current weights at their current value, and then to pick the next feature to add by solving

$$j^* = \arg\min_{j \notin \boldsymbol{\gamma}_t} \min_{\beta} ||\mathbf{y} - \mathbf{X}\mathbf{w}_t - \beta \mathbf{x}_{:,j}||^2$$

This only requires one least squares calculation per iteration and so is faster than orthogonal least squares, but is not quite as accurate.

13.3.1 Why does l_1 regularization yield sparse solutions?

We now explain why l_1 regularization results in sparse solutions, whereas l_2 regularization does not.

The objective is the following non-smooth objective function:

$$\min_{\mathbf{w}} \mathrm{RSS}(\mathbf{w}) + \lambda ||\mathbf{w}||_1$$

We can rewrite this as a constrained but smooth objective:

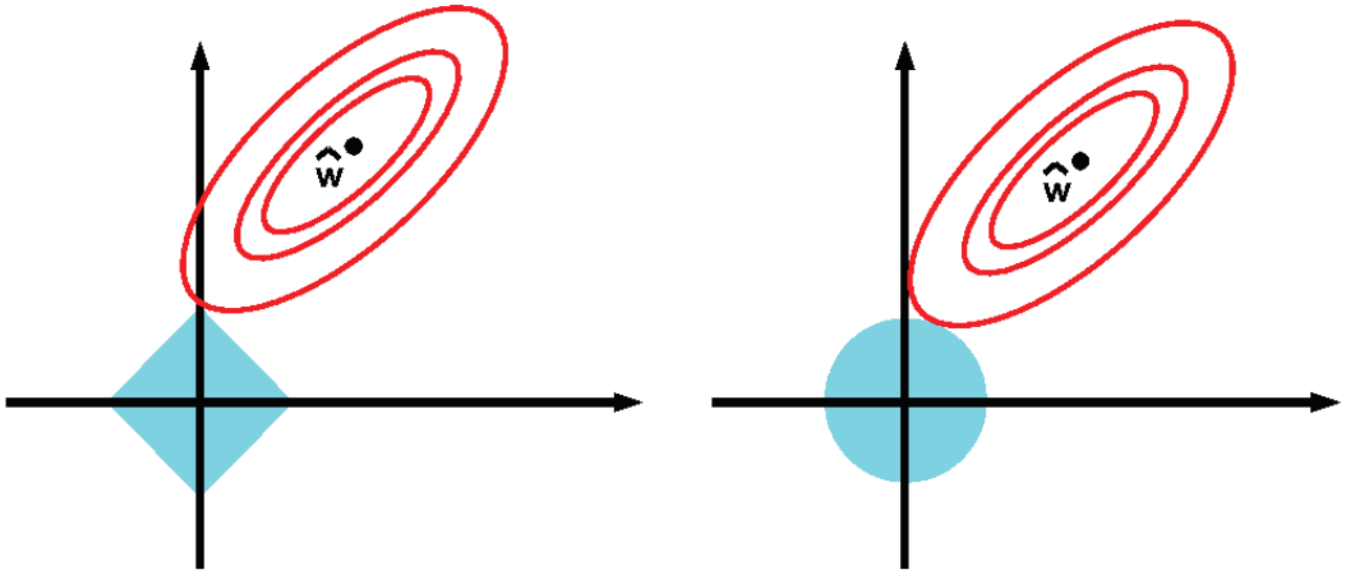
$$\min_{\mathbf{w}} \mathrm{RSS}(\mathbf{w}) \quad \text{s.t.} \quad ||\mathbf{w}||_1 \leq B$$

Similarly, we can write ridge regression

$$\min_{\mathbf{w}} \mathrm{RSS}(\mathbf{w}) + \lambda ||\mathbf{w}||_2^2$$

$$\min_{\mathbf{w}} \mathrm{RSS}(\mathbf{w}) \quad \text{s.t.} \quad ||\mathbf{w}||_2^2 \leq B$$

13.3.1 Why does l_1 regularization yield sparse solutions?



13.3.2 Optimality conditions for lasso

The lasso objective has the form

$$f(\boldsymbol{\theta}) = \mathrm{RSS}(\boldsymbol{\theta}) + \lambda ||\mathbf{w}||_1$$

Unfortunately, the $||w||_1$ term is not differentiable whenever $w_j = 0$. This is an example of a non-smooth optimization problem.

To handle non-smooth functions, we need to extend the notion of a derivative. We define a subderivative or subgradient of a (convex) function $f : I \rightarrow R$ at a point θ_0 to be a scalar g such that

$$f(\theta) - f(\theta_0) \geq g(\theta - \theta_0) \quad \forall \theta \in \mathcal{I}$$

where I is some interval containing θ_0 .

13.3.2 Optimality conditions for lasso

We define the set of subderivatives as the interval [a, b] where a and b are the one-sided limits.

$$a = \lim_{\theta \to \theta_0^-} \frac{f(\theta) - f(\theta_0)}{\theta - \theta_0}, \quad b = \lim_{\theta \to \theta_0^+} \frac{f(\theta) - f(\theta_0)}{\theta - \theta_0}$$

The set [a, b] of all subderivatives is called the subdifferential of the function f at θ_0 and is denoted $\partial f(\theta)|\theta_0$.

For example, in the case of the absolute value function f(θ) = |θ|, the subderivative is given by

$$\partial f(\theta) = \begin{cases} \{-1\} & \text{if } \theta < 0 \\ [-1, 1] & \text{if } \theta = 0 \\ \{+1\} & \text{if } \theta > 0 \end{cases}$$

13.3.2 Optimality conditions for lasso

Let us apply these concepts to the lasso problem. We ignore the non-smooth penalty term initially.

$$\begin{aligned} \frac{\partial}{\partial w_j}\mathrm{RSS}(\mathbf{w}) &= a_j w_j - c_j \\ a_j &= 2\sum_{i=1}^{n} x_{ij}^2 \\ c_j &= 2\sum_{i=1}^{n} x_{ij}(y_i - \mathbf{w}_{-j}^T \mathbf{x}_{i,-j}) \end{aligned}$$

where w_{-j} is w without component j.

13.3.2 Optimality conditions for lasso

Adding in the penalty term, we find that the subderivative is given by

$$\begin{aligned}\partial_{w_j} f(\mathbf{w}) &= (a_j w_j - c_j) + \lambda \partial_{w_j} ||\mathbf{w}||_1 \\ &= \begin{cases} \{a_j w_j - c_j - \lambda\} & \text{if } w_j < 0 \\ [-c_j - \lambda, -c_j + \lambda] & \text{if } w_j = 0 \\ \{a_j w_j - c_j + \lambda\} & \text{if } w_j > 0 \end{cases}\end{aligned}$$

If $c_j < -\lambda$, so the feature is strongly negatively correlated with the residual, then the subgradient is zero at $\hat{w}_j = \frac{c_j+\lambda}{a_j} < 0$.

If $c_j \in [-\lambda, \lambda]$, so the feature is only weakly correlated with the residual, then the subgradient is zero at $\hat{w}_j = 0$.

If $c_j > \lambda$, so the feature is strongly positively correlated with the residual, then the subgradient is zero at $\hat{w}_j = \frac{c_j-\lambda}{a_j} > 0$.

13.3.2 Optimality conditions for lasso

In summary, we have

$$\hat{w}_j(c_j) = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } c_j \in [-\lambda, \lambda] \\ (c_j - \lambda)/a_j & \text{if } c_j > \lambda \end{cases}$$

We can write this as follows:

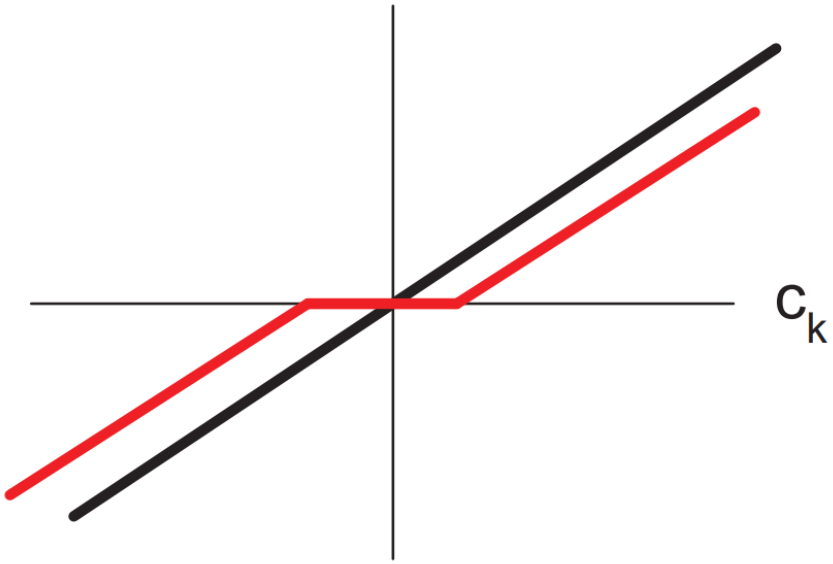
$$\hat{w}_j = \text{soft}(\frac{c_j}{a_j}; \frac{\lambda}{a_j})$$

where

$$\text{soft}\,(a; \delta) \triangleq \text{sign}(a)\,(|a| - \delta)_+$$

and $x^+ = \max(x, 0)$ is the positive part of x.

13.3.2 Optimality conditions for lasso



Least absolute selection and shrinkage operator(lasso) selects a subset of the variables, and shrinks all the coefficients by penalizing the absolute values.

13.3.3 Comparison of least squares, lasso, ridge and subset selection

For simplicity, assume all the features of X are orthonormal, so $X^T X = I$. In this case, the RSS is given by

$$\begin{aligned} \text{RSS}(\mathbf{w}) &= ||\mathbf{y} - \mathbf{X}\mathbf{w}||^2 = \mathbf{y}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} \\ &= \text{const} + \sum_k w_k^2 - 2\sum_k \sum_i w_k x_{ik} y_i \end{aligned}$$

Ridge One can show that the ridge estimate is given by

$$\hat{w}_k^{ridge} = \frac{\hat{w}_k^{OLS}}{1+\lambda}$$

13.3.3 Comparison of least squares, lasso, ridge and subset selection

Lasso: using the fact that $a_k = 2$ and $\hat{w}_k^{OLS} = \frac{c_k}{2}$, we have

$$\hat{w}_k^{lasso} = \text{sign}(\hat{w}_k^{OLS}) \left(|\hat{w}_k^{OLS}| - \frac{\lambda}{2} \right)_+$$

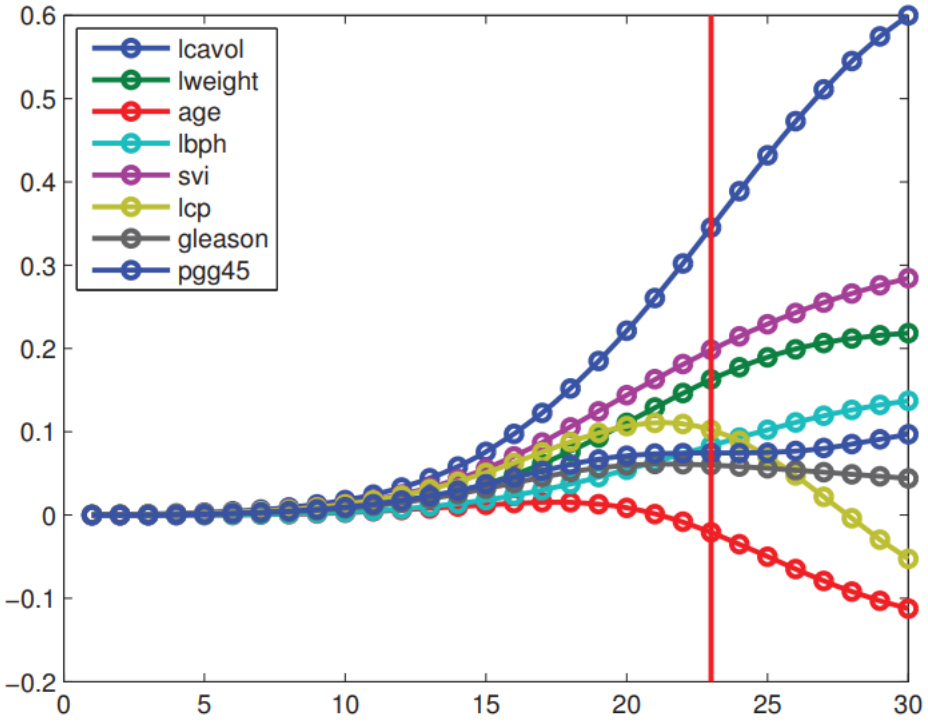
Subset selection: If we pick the best K features using subset selection, the parameter estimate is as follows

$$\hat{w}_k^{SS} = \begin{cases} \hat{w}_k^{OLS} & \text{if rank}(|w_k^{OLS}|) \leq K \\ 0 & \text{otherwise} \end{cases}$$

13.3.4 Regularization path

As we increase λ, the solution vector $\hat{w}_j(\lambda)$ will tend to get sparser, although not necessarily monotonically. We can plot the values $\hat{w}_j(\lambda)$ vs λ for each feature j; this is known as the regularization path.

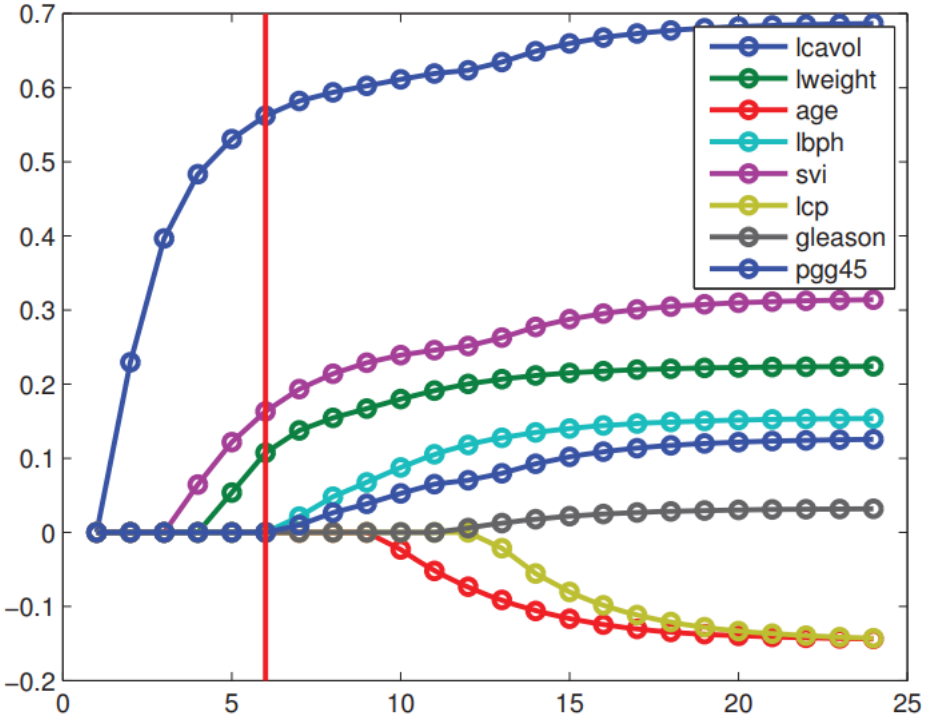
This is illustrated for ridge regression in Figure (a), where we plot $\hat{w}_j(\lambda)$ as the regularizer λ decreases. We see that when $\lambda = \infty$, all the coefficients are zero. But for any finite value of λ, all coefficients are non-zero; furthermore, they increase in magnitude as λ is decreased.



(a)

13.3.4 Regularization path

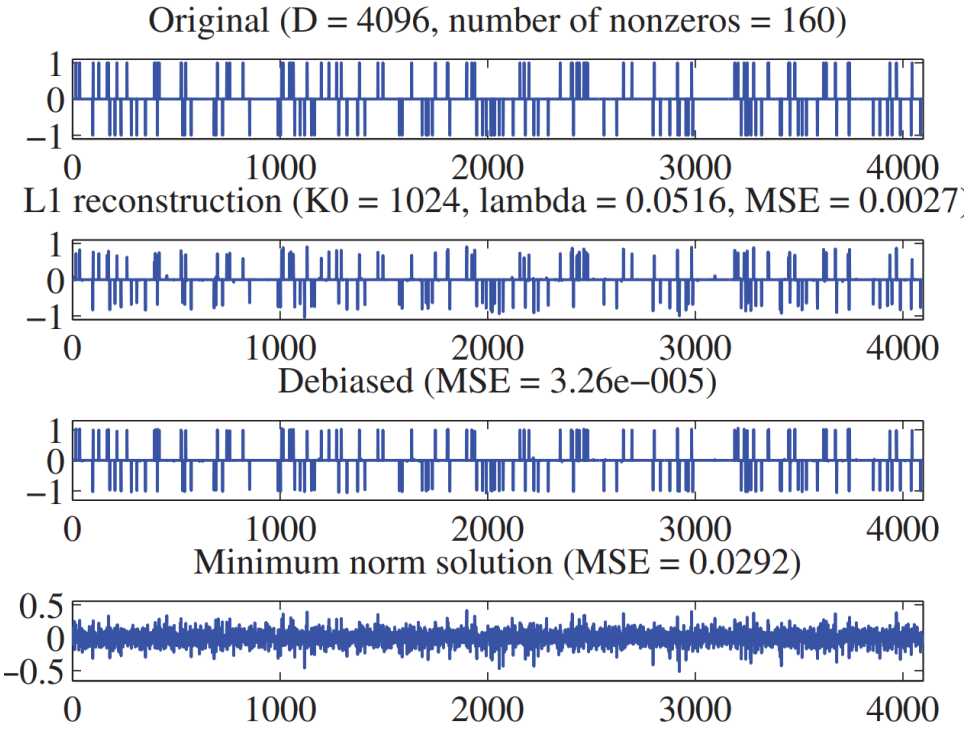
In Figure (b), we plot the analogous result for lasso. When B = 0, all the coefficients are zero. As we increase B, the coefficients gradually "turn on". But for any value between 0 and $B_{max} = ||\hat{w}_{OLS}||_1$, the solution is sparse.



(b)

13.3.5 Model selection

We first generate a sparse signal w^* of size $D = 4096$, consisting of 160 randomly placed ± 1 spikes. Next we generate a random design matrix X of size $N \times D$, where $N = 1024$. Finally we generate a noisy observation $y = Xw^* + \varepsilon$, where $\varepsilon_i \sim \mathcal{N}(0, 0.01^2)$. We then estimate w from y and X.



13.3.5 Model selection

The original w^* is shown in the first row of Figure 13.9.

The second row is the ℓ_1 estimate $\widehat{w}_{L1}$ using $\lambda = 0.1\lambda_{\max}$. We see that this has "spikes" in the right places, but they are too small.

The third row is the least squares estimate of the coefficients which are estimated to be non-zero based on supp($\widehat{w}_{L1}$). This is called debiasing, and is necessary because lasso shrinks the relevant coefficients as well as the irrelevant ones.

The last row is the least squares estimate for all the coefficients jointly, ignoring sparsity.

13.3.5 Model selection

A downside of using ℓ_1 regularization to select variables is that it can give quite different results if the data is perturbed slightly. A frequentist solution to this is to use bootstrap resampling, and to rerun the estimator on different versions of the data. By computing how often each variable is selected across different trials, we can approximate the posterior inclusion probabilities. This method is known as stability selection.

We can threshold the stability selection (bootstrap) inclusion probabilities at some level, say 90%. This is known as bootstrap lasso or bolasso . It will include a variable if it occurs in at least 90% of sets returned by lasso (for a fixed λ).

13.4.1 Coordinate descent

In this section, we give a brief review of some algorithms that can be used to solve ℓ_1 regularized estimation problems.

Sometimes it is hard to optimize all the variables simultaneously, but it easy to optimize them one by one. In particular, we can solve for the j'th coefficient with all the others held fixed:

$$w_j^* = \underset{z}{\operatorname{argmin}} f(\mathbf{w} + z\mathbf{e}_j) - f(\mathbf{w})$$

13.4.2 LARS

前向选择（Forward Selection）算法

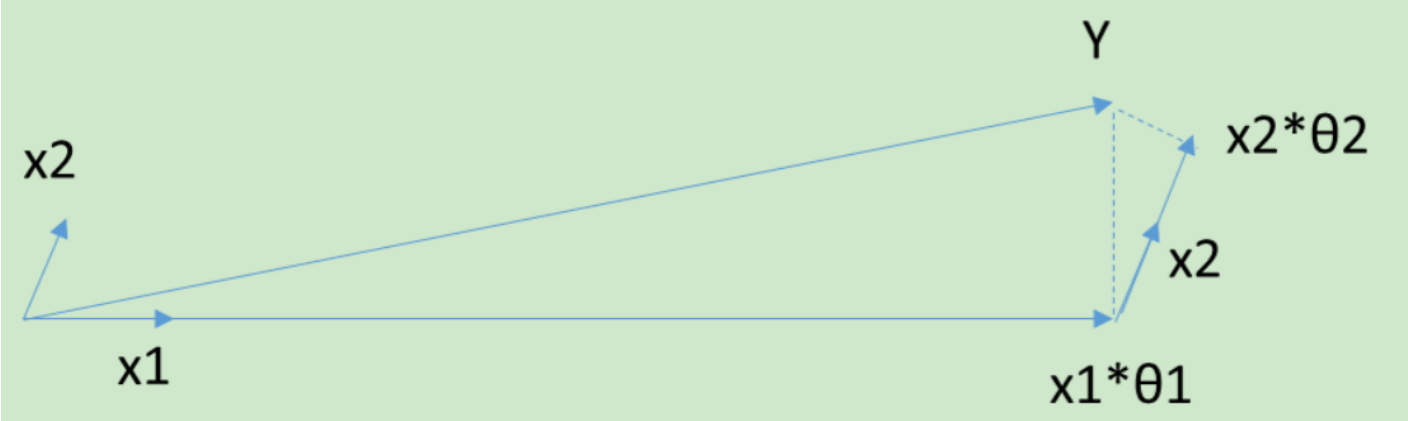
对于$Y = X\theta$这样的线性关系，求解稀疏向量θ。其中Y是$m \times 1$的向量，X为$m \times n$的矩阵，θ为$n \times 1$的向量，m是样本数量，n是特征维度。

则可把矩阵X看作n个$m \times 1$的向量$X_i(i = 1,2,\cdots,n)$，在所有X_i中选择和目标函数Y最为接近的一个变量X_k，用X_k来逼近Y，得到下式：

$$\bar{Y} = X_k\theta_k，\text{其中}\theta_k = \frac{<X_k,Y>}{||X_k||_2}$$

$\bar{Y}$是Y在X_k上的投影。残差为$Y_{yes} = Y - \bar{Y}$。因为是投影，所以Y_{yes}和X_k是正交的。再以Y_{yes}为新的因变量，去掉X_k后，剩下的自变量的集合为X_i为新的自变量集合，重复刚刚的投影和残差操作，直到残差为0，或者所有自变量均用完，才停止算法。

13.4.2 LARS

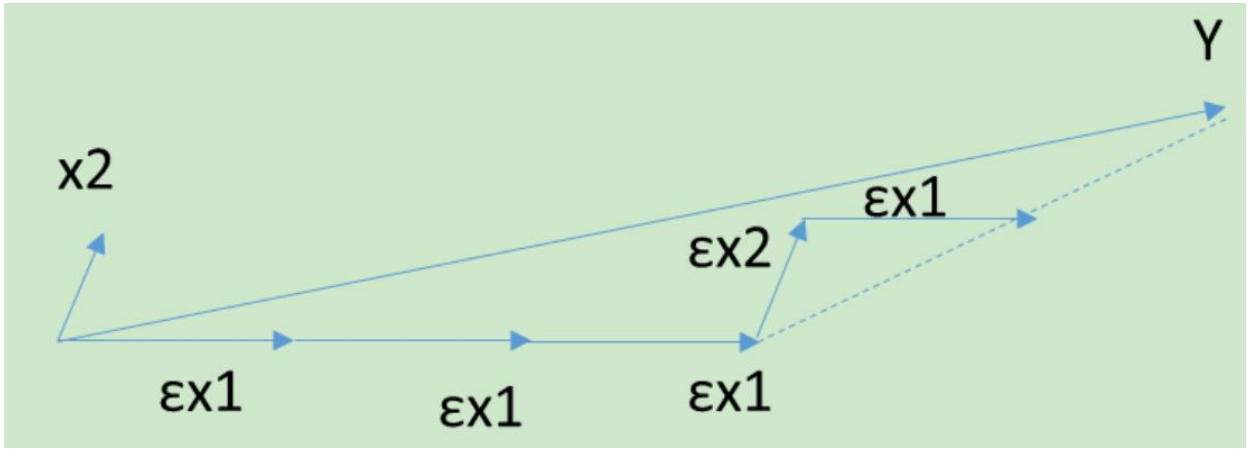


此算法对每个变量只需要执行一次操作，效率高，速度快。但也容易看出，当自变量不是正交的时候，由于每次都是在做投影，所有算法只能给出一个局部近似解。

13.4.2 LARS

前向梯度（Forward Stagewise）算法

在$\boldsymbol{X}$的变量X_i中，选择和目标函数Y最为接近（余弦距离最大）的一个变量X_k，用X_k来逼近Y，每次在最为接近的自变量X_k的方向移动一小步，然后再看残差和哪个X_i最为接近。再下一轮的选择中也有可能选X_k，因为有可能下面最接近的自变量还是X_k。如此下去，直到残差Y_{yes}减小到足够小，算法停止。



当算法在ε很小的时候，可以很精确的给出最优解。当然，其计算的迭代次数也是大大的增加。和前向选择算法相比，前向梯度算法更加精确，但是更加复杂。

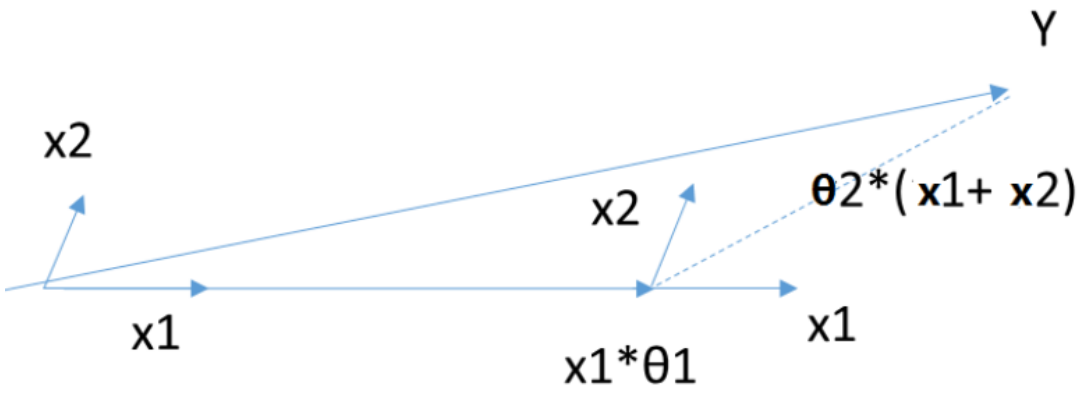
13.4.2 LARS

最小角回归(**Least Angle Regression， LARS**)算法

找到与因变量Y最接近的自变量X_k，沿着X_k的方向往前走，直到出现一个X_t，使得X_t和Y_{yes}的相关度与X_k和Y_{yes}的相关度一样，此时Y_{yes}沿着X_t和X_k的角平分线走，重复之前的步骤，直到Y_{yes}足够小，或者所有变量已经取完，则算法停止，此时对应的系数θ即为最终结果。

13.4.2 LARS

当θ只有二维时，例子可见下图，和Y最接近的是$\boldsymbol{X}_1$，首先在$\boldsymbol{X}_1$上走一段距离，一直到残差在$\boldsymbol{X}_1$和$\boldsymbol{X}_2$的角平分线上，此后沿着角平分线走，直到残差足够小时停止。



13.4.2 LARS

Algorithm 3.2 *Least Angle Regression.*

1. Standardize the predictors to have mean zero and unit norm. Start with the residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, $\beta_1, \beta_2, \ldots, \beta_p = 0$.

2. Find the predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$.

3. Move β_j from 0 towards its least-squares coefficient $\langle \mathbf{x}_j, \mathbf{r} \rangle$, until some other competitor $\mathbf{x}_k$ has as much correlation with the current residual as does $\mathbf{x}_j$.

4. Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor $\mathbf{x}_l$ has as much correlation with the current residual.

5. Continue in this way until all p predictors have been entered. After $\min(N-1, p)$ steps, we arrive at the full least-squares solution.

13.4.3 Proximal and gradient projection methods

Consider a convex objective of the form

$$f(\boldsymbol{\theta}) = L(\boldsymbol{\theta}) + R(\boldsymbol{\theta}) \tag{13.66}$$

where $L(\theta)$ (representing the loss) is convex and differentiable, and $R(\theta)$ (representing the regularizer) is convex but not necessarily differentiable.

The lasso problem can be formulated as follows: $L(\theta) = RSS(\theta)$ and $R(\theta) = I_C(\theta)$, where $C = \left\{\theta : ||\theta||_1 \leq B\right\}$, and $I_C(\theta)$ is the indicator function of a convex set C, defined as

$$I_C(\boldsymbol{\theta}) \triangleq \begin{cases} 0 & \boldsymbol{\theta} \in C \\ +\infty & \text{otherwise} \end{cases}$$

13.4.3 Proximal and gradient projection methods

In some cases, it is easy to optimize functions of the form in Equation 13.66. For example, suppose $L(\theta) = RSS(\theta)$, and the design matrix is simply $X = I$. Then the obective becomes $f(\theta) = \frac{1}{2}||\theta - y||_2^2 + R(\theta)$. The minimizer of this is given by $prox_R(\mathrm{y})$, which is the proximal operator for the convex function R, defined by

$$\mathrm{prox}_R(\mathbf{y}) = \underset{\mathbf{z}}{\mathrm{argmin}} \left(R(\mathbf{z}) + \frac{1}{2}||\mathbf{z} - \mathbf{y}||_2^2 \right)$$

Intuitively, we are returning a point that minimizes R but which is also close (proximal) to y.

13.4.3 Proximal and gradient projection methods

Proximal operators

If $R(\theta) = \lambda||\theta||_1$, the proximal operator is given by componentwise soft-thresholding:

$$\text{prox}_R(\boldsymbol{\theta}) = \text{soft}(\boldsymbol{\theta}, \lambda)$$

If $R(\theta) = \lambda||\theta||_0$, the proximal operator is given by componentwise hard-thresholding:

$$\text{prox}_R(\boldsymbol{\theta}) = \text{hard}(\boldsymbol{\theta}, \sqrt{2\lambda})$$

where $hard(u, a) \triangleq uI(|u| > a)$.

If $R(\theta) = I_C(\theta)$, the proximal operator is given by the projection onto the set C:

$$\text{prox}_R(\boldsymbol{\theta}) = \underset{\mathbf{z} \in C}{\text{argmin}} \, ||\mathbf{z} - \boldsymbol{\theta}||_2^2 = \text{proj}_C(\boldsymbol{\theta})$$

13.4.3 Proximal and gradient projection methods

Proximal gradient method

We now discuss how to use the proximal operator inside of a gradient descent routine. The basic idea is to minimize a simple quadratic approximation to the loss function, centered on the θ_k:

$$\theta_{k+1} = \text{argmin}_z R(z) + L(\theta_k) + g_k^T(z - \theta_k) + \frac{1}{2t_k}||z - \theta_k||_2^2$$

where $\mathrm{g}_k = \nabla L(\theta_k)$ is the gradient of the loss, $\nabla^2 L(\theta_k) \approx \frac{1}{t_k}\mathrm{I}$.

Dropping terms that are independent of z, and multiplying by t_k, we can rewrite the above expression in terms of a proximal operator as follows:

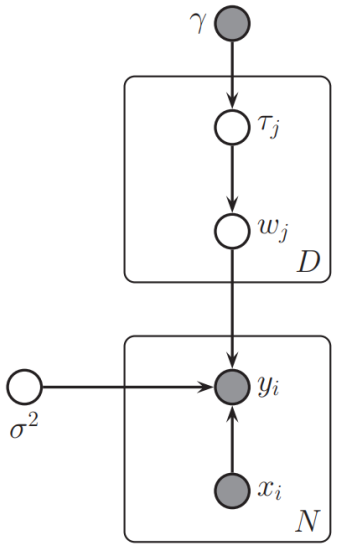
$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= \underset{\mathbf{z}}{\text{argmin}} \left[t_k R(\mathbf{z}) + \frac{1}{2}||\mathbf{z} - \mathbf{u}_k||_2^2 \right] = \text{prox}_{t_k R}(\mathbf{u}_k) \\ \mathbf{u}_k &= \boldsymbol{\theta}_k - t_k \mathbf{g}_k \\ \mathbf{g}_k &= \nabla L(\boldsymbol{\theta}_k) \end{aligned}$$

13.4.4 EM for lasso

The key insight is that we can represent the Laplace distribution as a **Gaussian scale mixture** as follows:

$$\text{Lap}(w_j|0,1/\gamma) = \frac{\gamma}{2}e^{-\gamma|w_j|} = \int \mathcal{N}(w_j|0,\tau_j^2)\text{Ga}(\tau_j^2|1,\frac{\gamma^2}{2})d\tau_j^2$$

Using this decomposition, we can represent the lasso model as shown in Figure 13.12.



13.4.4 EM for lasso

The corresponding joint distribution has the form

$$p(\mathbf{y}, \mathbf{w}, \boldsymbol{\tau}, \sigma^2|\mathbf{X}) = \mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N)\,\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{D}_\tau) \\ \mathrm{IG}(\sigma^2|a_\sigma, b_\sigma)\left[\prod_j \mathrm{Ga}(\tau_j^2|1, \gamma^2/2)\right]$$

Where $D_\tau = diag(\tau_j^2)$, and where we have assumed for notational simplicity that $\boldsymbol{X}$ is standardized and that $\boldsymbol{y}$ is centered. Expanding out, we get

$$p(\mathbf{y}, \mathbf{w}, \boldsymbol{\tau}, \sigma^2|\mathbf{X}) \propto (\sigma^2)^{-N/2}\exp\left(-\frac{1}{2\sigma^2}||\mathbf{y}-\mathbf{X}\mathbf{w}||_2^2\right)|\mathbf{D}_\tau|^{-\frac{1}{2}} \\ \exp\left(-\frac{1}{2}\mathbf{w}^T\mathbf{D}_\tau\mathbf{w}\right)(\sigma^2)^{-(a_\sigma+1)} \\ \exp(-b_\sigma/\sigma^2)\prod_j \exp(-\frac{\gamma^2}{2}\tau_j^2) \quad (13.88)$$

In brief, in the E step we infer τ_j^2 and σ^2, and in the M step we estimate w.

13.4.4 EM for lasso

The objective function

From Equation 13.88, the complete data penalized log likelihood is as follows (dropping terms that do not depend on w)

$$\ell_c(\mathbf{w}) = -\frac{1}{2\sigma^2}||\mathbf{y}-\mathbf{X}\mathbf{w}||_2^2 - \frac{1}{2}\mathbf{w}^T\mathbf{\Lambda}\mathbf{w} + \text{const}$$

where $\mathbf{\Lambda} = diag(\tau_j^2)$ is the precision matrix for w.

13.4.4 EM for lasso

The E step

The key is to compute $E\left[\frac{1}{\tau_j^2}|w_j\right]$.

$$P(w_j) = \int \mathcal{N}(w_j|0,\tau_j^2)P(\tau_j^2)d\tau_j^2$$

$$\frac{dP(w_j)}{d|w_j|} = P(w_j)\frac{dInP(w_j)}{d|w_j|} = \int \frac{d\mathcal{N}(w_j|0,\tau_j^2)}{d|w_j|}P(\tau_j^2)d\tau_j^2$$

$$= \int \frac{1}{\tau_j^2}\mathcal{N}(w_j|0,\tau_j^2)(-|w_j|)\,P(\tau_j^2)d\tau_j^2$$

$$E\left[\frac{1}{\tau_j^2}|w_j\right] = \int \frac{1}{\tau_j^2}\frac{P(w_j|\tau_j^2)P(\tau_j^2)}{P(w_j)}d\tau_j^2 = -\frac{1}{-|w_j|}\frac{dInP(w_j)}{d|w_j|}$$

13.4.4 EM for lasso

M step

The M step consists of computing

$$\hat{w} = argmax -\frac{1}{2\sigma^2}\left|\left|y - Xw\right|\right|_2^2 - \frac{1}{2}w^T \Lambda w$$

This is just MAP estimation under a Gaussian prior:

$$\hat{\mathbf{w}} = (\sigma^2\overline{\mathbf{\Lambda}} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Thanks !