CAP 5625: Computational Foundations for Artificial Intelligence

Advanced regularization techniques

Identifying optimal ridge regression parameters

Recall that with ridge regression, we were able to take the gradient with respect to the cost function

$$J(\beta, \lambda) = \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

where the gradient in dimension k is the partial derivative with respect to β_k, defined as

$$\frac{\partial}{\partial \beta_k} J(\beta, \lambda) = -2 \sum_{i=1}^{N} x_{ik} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right) + 2\lambda \beta_k$$

From this partial derivative we were able to generate a gradient descent approach (and even normal equations) to uncover the optimal parameter vector $\hat{\beta}$.

Identifying optimal lasso parameters not as easy

However, we cannot identify the optimal parameter vector $\hat{\beta}$ under lasso using the same approach, because the cost function

$$J(\beta, \lambda) = \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

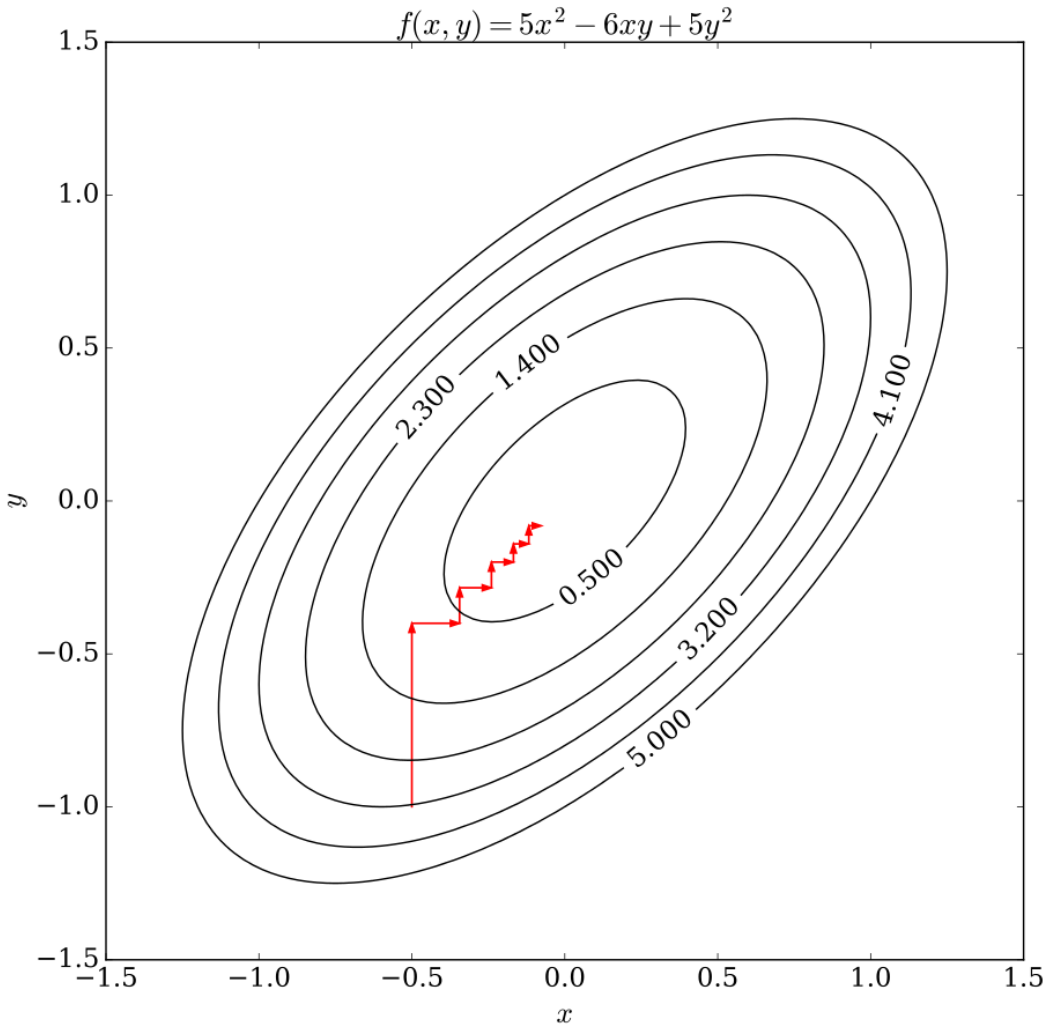
is not differentiable, because $|\beta_k|$ is not differentiable at 0.

The optimization algorithm we will consider for fitting the model parameters under the lasso penalty is **coordinate descent**, in which we iteratively find the optimal value for each coordinate (parameter), holding the other coordinates (parameters) fixed.

That is, we will find the optimal β_k, $k = 1,2, \ldots, p$, each in turn.

Illustration of coordinate descent

Each coordinate is optimized separately, and in turn.



https://en.wikipedia.org/wiki/Coordinate_descent#/media/File:Coordinate_descent.svg

Begin by taking partial derivative of $\mathrm{RSS}(\beta)$

Recall that the lasso cost function is

$$J(\beta, \lambda) = \mathrm{RSS}(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

We derived the partial derivative of $\mathrm{RSS}(\beta)$ with respect to β_k as

$$\frac{\partial}{\partial \beta_k} \mathrm{RSS}(\beta) = -2 \sum_{i=1}^{N} x_{ik} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)$$

$$= -2 \sum_{i=1}^{N} x_{ik} \left(y_i - \sum_{\substack{j=1 \\ j \neq k}}^{p} x_{ij} \beta_j - x_{ik} \beta_k \right)$$

$$= -2 \sum_{i=1}^{N} x_{ik} \left(y_i - \sum_{\substack{j=1 \\ j \neq k}}^{p} x_{ij} \beta_j \right) + 2 \beta_k \sum_{i=1}^{N} x_{ik}^2$$

Begin by taking partial derivative of $\mathrm{RSS}(\beta)$

$$\frac{\partial}{\partial \beta_k}\mathrm{RSS}(\beta) = -2\sum_{i=1}^{N} x_{ik}\left(y_i - \sum_{\substack{j=1 \\ j\neq k}}^{p} x_{ij}\beta_j\right) + 2\beta_k \sum_{i=1}^{N} x_{ik}^2$$

Define

$$a_k = \sum_{i=1}^{N} x_{ik}\left(y_i - \sum_{\substack{j=1 \\ j\neq k}}^{p} x_{ij}\beta_j\right)$$

$$b_k = \sum_{i=1}^{N} x_{ik}^2$$

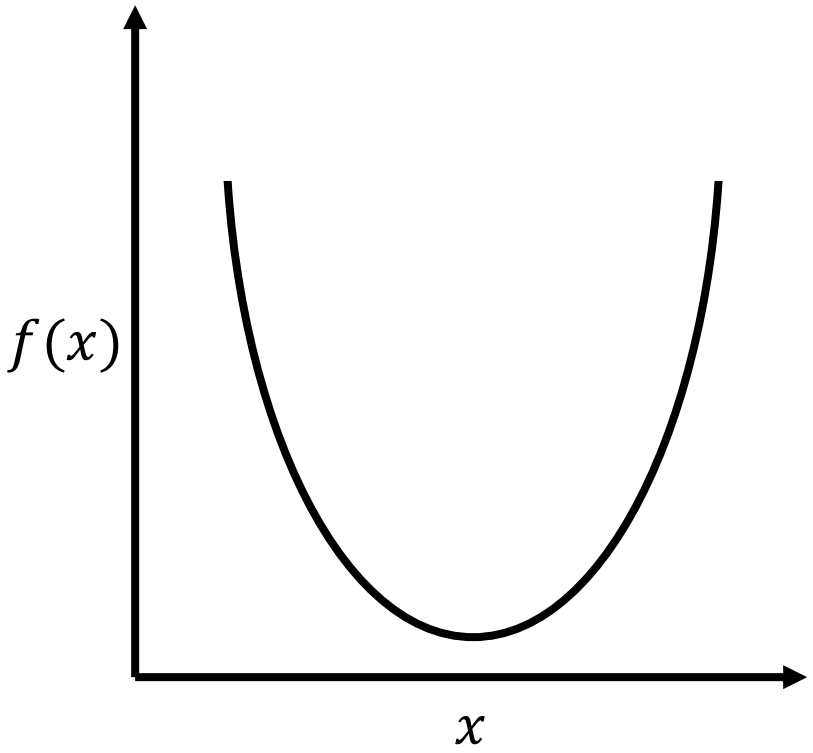
We then have

$$\frac{\partial}{\partial \beta_k}\mathrm{RSS}(\beta) = -2a_k + 2\beta_k b_k$$

Lower bound of convex functions through gradients

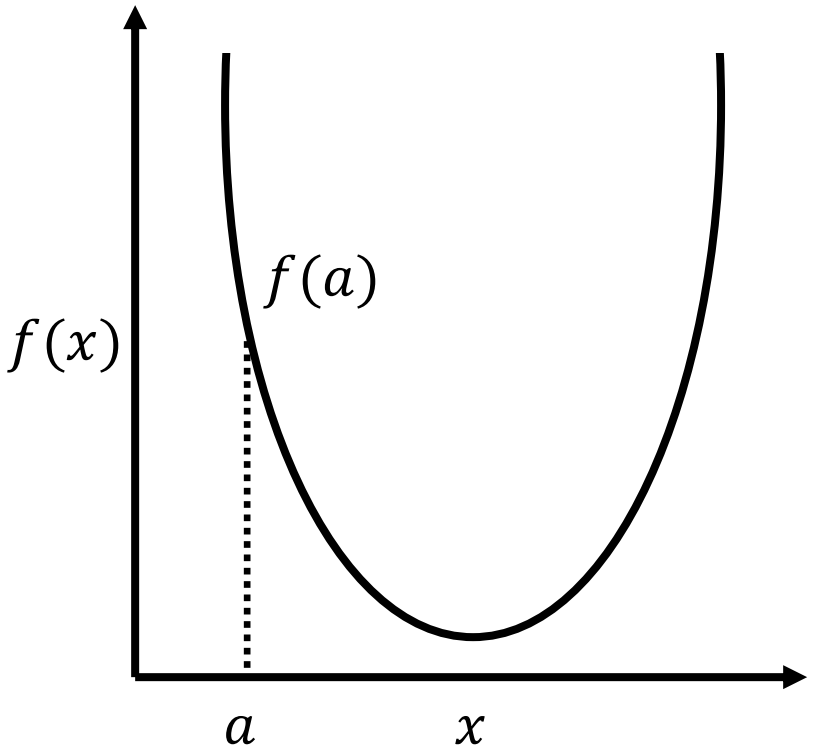
Consider the convex function $f(x)$ shown below.

The gradient (or derivative) of this function is a lower bound.



Lower bound of convex functions through gradients

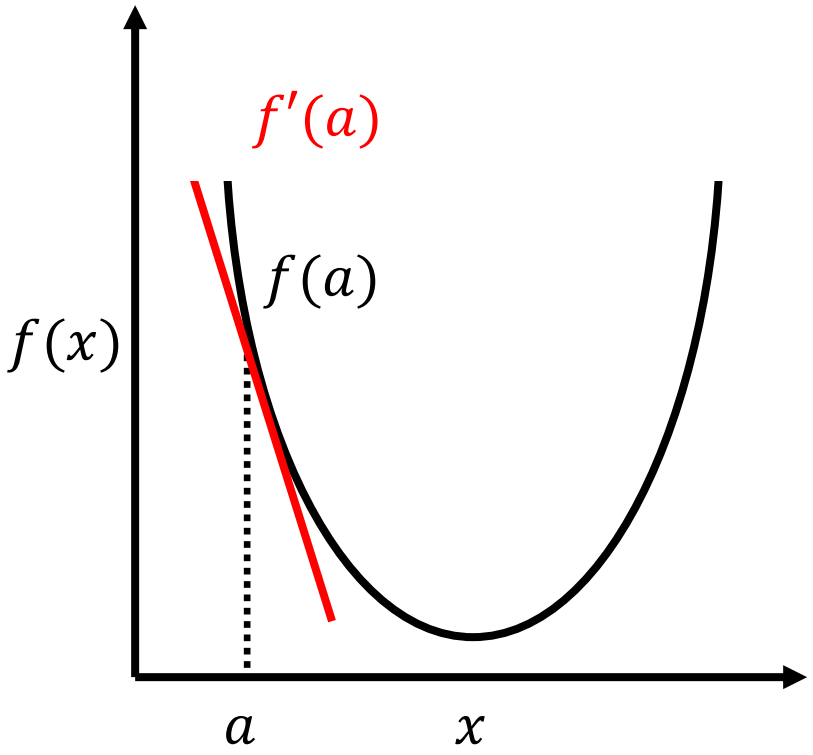
Consider the function evaluated at point a denoted $f(a)$.



Lower bound of convex functions through gradients

Consider the function evaluated at point a denoted $f(a)$.

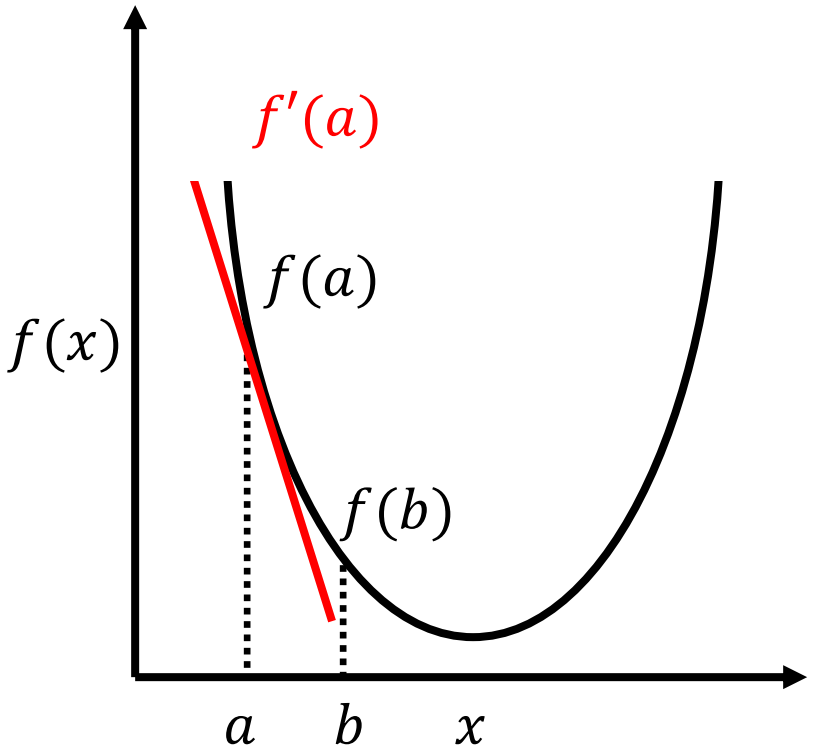
Also, consider the derivative $f'(a)$ of the function valuated at point a.



Lower bound of convex functions through gradients

Then for the function evaluated at another point b, denoted $f(b)$, we have

$$f(b) \geq f(a) + f'(a)(b - a)$$



Subgradients generalize for non-differentiable points

A **subgradient** (**subderivative** or **subdifferential**) $\partial f(x)$ of f at x generalizes the derivative to convex function that are non-differentiable, and is the slope of a line that touches $f(x)$ at a particular point.

If $f(x)$ is differentiable, then this slope is unique, and if not, then there are a set of such subgradients (or slopes) that form a lower bound on $f(x)$ while touching that point.

Subgradients generalize for non-differentiable points

A **subgradient** (**subderivative** or **subdifferential**) $\partial f(x)$ of f at x generalizes the derivative to convex function that are non-differentiable, and is the slope of a line that touches $f(x)$ at a particular point.

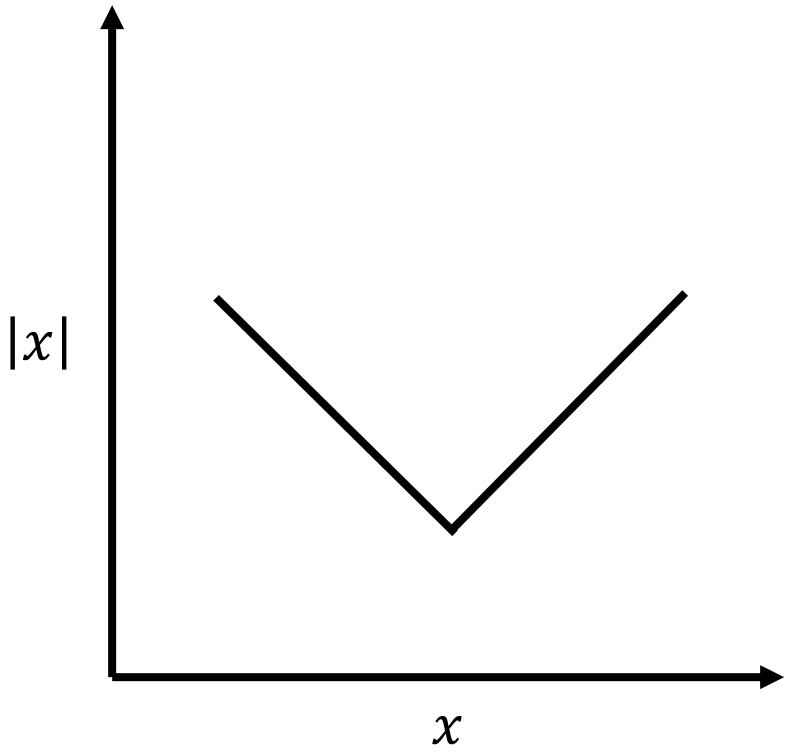
If $f(x)$ is differentiable, then this slope is unique, and if not, then there are a set of such subgradients (or slopes) that form a lower bound on $f(x)$ while touching that point.

That is $v \in \partial f(x)$ is a subgradient of f at a if

$$f(b) \geq f(a) + v(b - a)$$

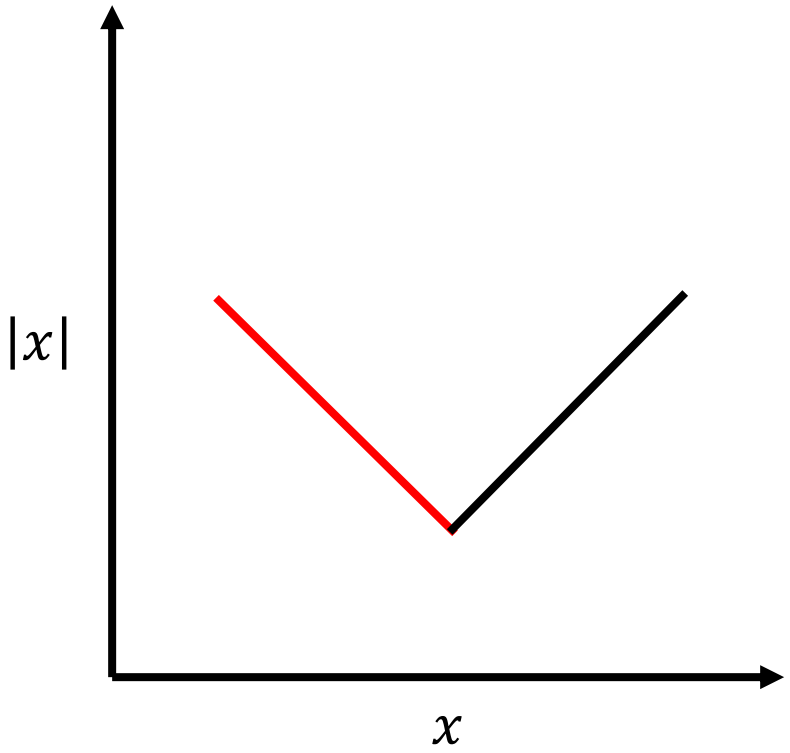
Subgradient of $|x|$

Consider the non-differentiable function $f(x) = |x|$ depicted below



Subgradient of $|x|$

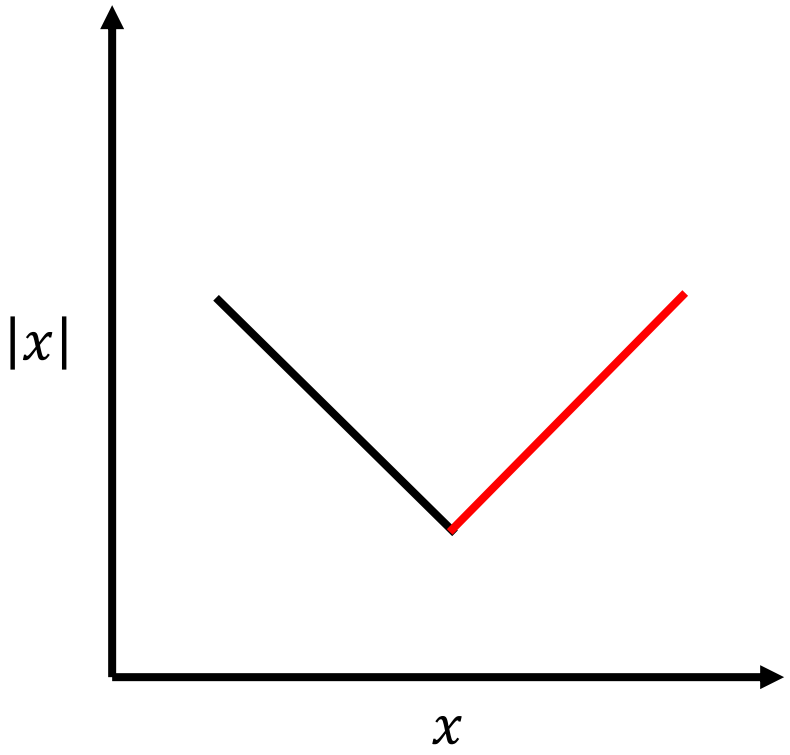
If $x < 0$, then the slope $f'(x) = -1$.



Subgradient of $|x|$

If $x < 0$, then the slope $f'(x) = -1$.

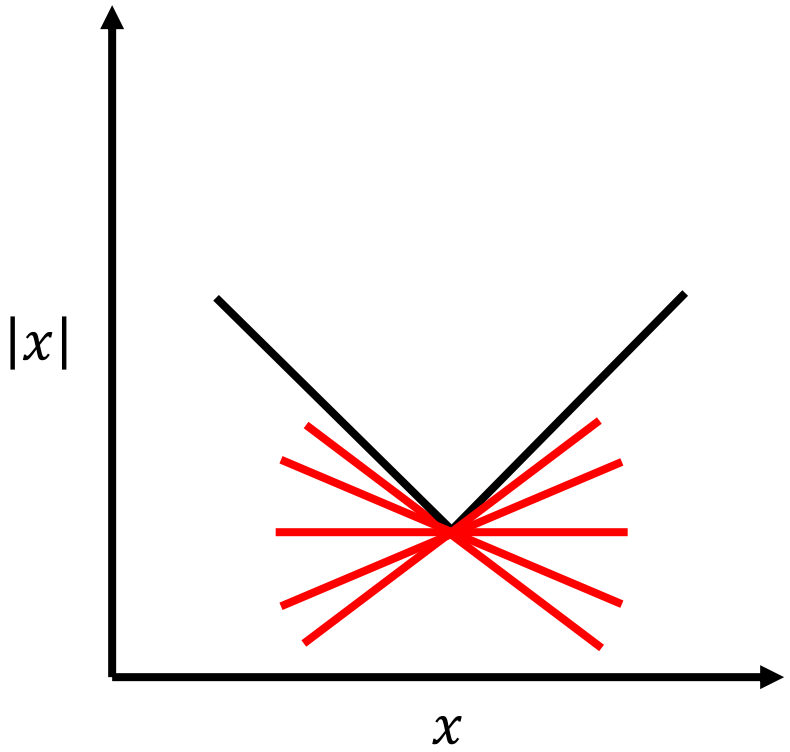
If $x > 0$, then the slope $f'(x) = 1$.



Subgradient of $|x|$

If $x = 0$, then the $f'(x)$ is undefined, and instead there are many possible lines with different slopes that are touching this point.

However, for $f(x) = |x|$, we have $v \in [-1,1]$ is a subgradient of $f(x)$ at $x = 0$.



Subgradient of the lasso penalty term

We were unable to previously differentiate the penalty term

$$\lambda \sum_{j=1}^{p} |\beta_j|$$

However, we can compute the subgradient with respect to β_k, which based on our previous discussion is

$$\lambda \partial_{\beta_k} |\beta_k| = \begin{cases} -\lambda & \text{if } \beta_k < 0 \\ [-\lambda, \lambda] & \text{if } \beta_k = 0 \\ \lambda & \text{if } \beta_k > 0 \end{cases}$$

The subgradient with respect to β_k of $\mathrm{RSS}(\beta)$ is

$$\partial_{\beta_k} \mathrm{RSS}(\beta) = \frac{\partial}{\partial \beta_k} \mathrm{RSS}(\beta) = -2a_k + 2\beta_k b_k$$

Subgradient of the lasso cost function $J(\beta, \lambda)$

The subgradient with respect to β_k of the lasso cost function is then

$$\begin{aligned}
\partial_{\beta_k} J(\beta, \lambda) &= \partial_{\beta_k} \mathrm{RSS}(\beta) + \lambda \partial_{\beta_k} |\beta_k| \\
&= -2a_k + 2\beta_k b_k + \lambda \partial_{\beta_k} |\beta_k| \\
&= -2a_k + 2\beta_k b_k + \begin{cases} -\lambda & \text{if } \beta_k < 0 \\ [-\lambda, \lambda] & \text{if } \beta_k = 0 \\ \lambda & \text{if } \beta_k > 0 \end{cases}
\end{aligned}$$

which is gives

$$\partial_{\beta_k} J(\beta, \lambda) = \begin{cases} -2a_k + 2\beta_k b_k - \lambda & \text{if } \beta_k < 0 \\ [-2a_k - \lambda, -2a_k + \lambda] & \text{if } \beta_k = 0 \\ -2a_k + 2\beta_k b_k + \lambda & \text{if } \beta_k > 0 \end{cases}$$

Optimize each coordinate in turn

In coordinate descent, we will take the gradient (subgradient) of the cost function with respect to each coordinate (parameter) and set this gradient (subgradient) to 0.

That is, we will solve $\partial_{\beta_k} J(\beta, \lambda) = 0$ for β_k to identify $\hat{\beta}_k$.

However, for each β_k, we have three cases to consider:

$\beta_k < 0$

$\beta_k = 0$

$\beta_k > 0$

The case when $\beta_k < 0$

When $\beta_k < 0$, setting $\partial_{\beta_k} J(\beta, \lambda) = 0$ gives

$$-2a_k + 2\beta_k b_k - \lambda = 0$$

Solving for β_k gives

$$\hat{\beta}_k = \frac{2a_k + \lambda}{2b_k} = \frac{a_k + \frac{\lambda}{2}}{b_k}$$

But we need $\beta_k < 0$, and therefore need $a_k < -\lambda/2$.

The case when $\beta_k = 0$

When $\beta_k = 0$, setting $\partial_{\beta_k} J(\beta, \lambda) = 0$ gives

$$0 \in [-2a_k - \lambda, -2a_k + \lambda]$$

Here we have that the optimum β_k is $\hat{\beta}_k = 0$ and so if $-2a_k + \lambda \geq 0$, then

$$a_k \leq \frac{\lambda}{2}$$

and if $-2a_k - \lambda \leq 0$, then

$$a_k \geq -\frac{\lambda}{2}$$

yielding

$$-\frac{\lambda}{2} \leq a_k \leq \frac{\lambda}{2}$$

The case when $\beta_k > 0$

When $\beta_k > 0$, setting $\partial_{\beta_k} J(\beta, \lambda) = 0$ gives

$$-2a_k + 2\beta_k b_k + \lambda = 0$$

Solving for β_k gives

$$\hat{\beta}_k = \frac{2a_k - \lambda}{2b_k} = \frac{a_k - \frac{\lambda}{2}}{b_k}$$

But we need $\beta_k > 0$, and therefore need $a_k > \lambda/2$.

Pulling it all together

Therefore if $a_k < -\lambda/2$, then set

$$\hat{\beta}_k = \frac{a_k + \frac{\lambda}{2}}{b_k}$$

If $a_k \in [-\lambda/2, \lambda/2]$, then set

$$\hat{\beta}_k = 0$$

If $a_k > \lambda/2$, then set

$$\hat{\beta}_k = \frac{a_k - \frac{\lambda}{2}}{b_k}$$

Note that following this procedure, the least squares estimates are

$$\hat{\beta}_k = \frac{a_k}{b_k}$$

Soft thresholding yields sparsity of **lasso**

We have that

$$\hat{\beta}_k = \begin{cases} \dfrac{a_k + \lambda/2}{b_k} & \text{if } a_k < -\dfrac{\lambda}{2} \\ 0 & \text{if } a_k \in \left[-\dfrac{\lambda}{2}, \dfrac{\lambda}{2}\right] \\ \dfrac{a_k - \lambda/2}{b_k} & \text{if } a_k > \dfrac{\lambda}{2} \end{cases}$$

or alternatively that

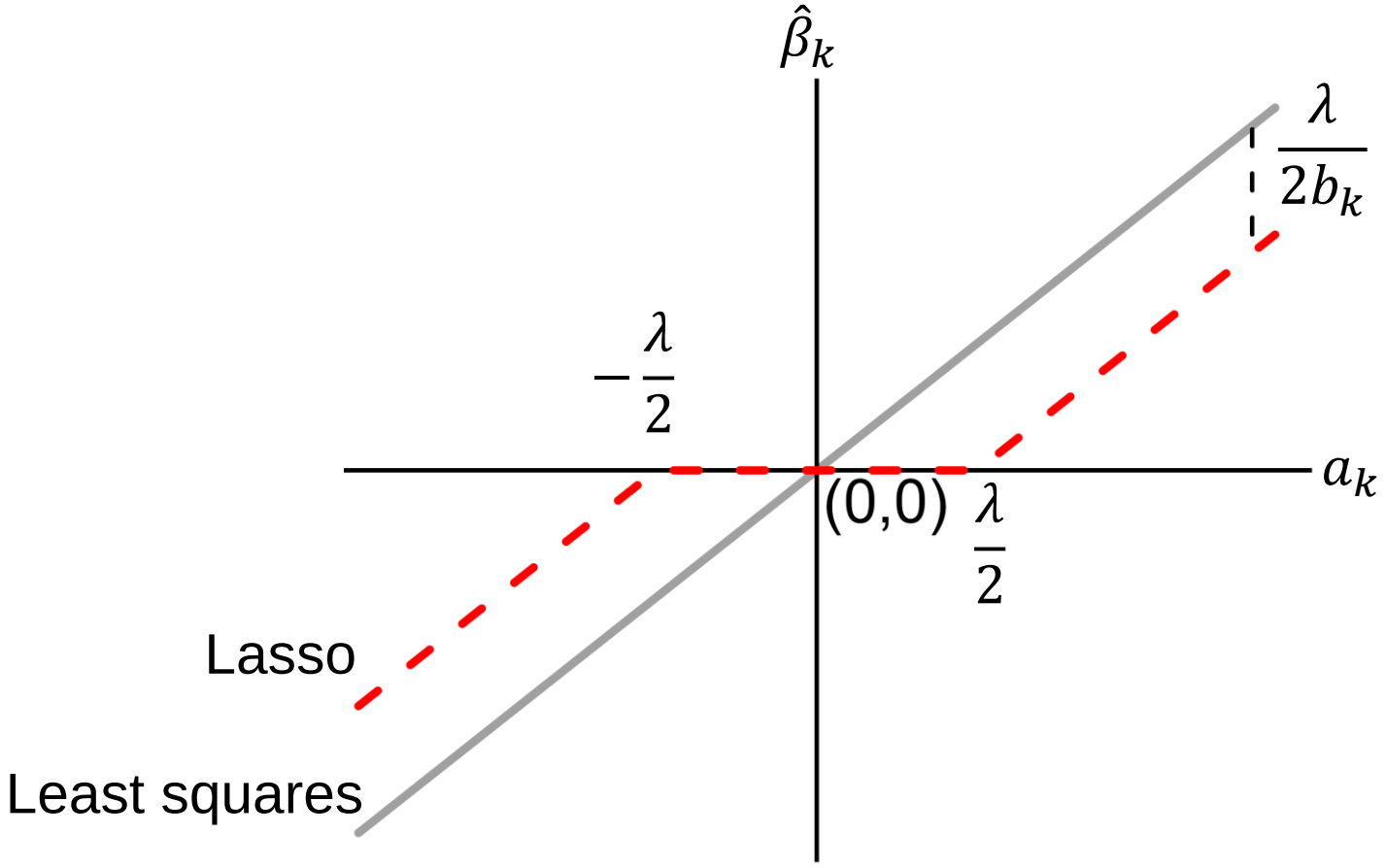
$$\hat{\beta}_k = \frac{\text{sign}(a_k)}{b_k}\left(|a_k| - \frac{\lambda}{2}\right)_+$$

where

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases} \qquad x_+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Soft thresholding of **lasso** illustrated

$$\hat{\beta}_k = \frac{\text{sign}(a_k)}{b_k}\left(|a_k| - \frac{\lambda}{2}\right)_+$$



Adapted from Hastie *et al* (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd Ed.

Coordinate descent algorithm for **lasso**

1. Precompute $b_j, j = 1,2,\dots,p$ as

$$b_j = \sum_{i=1}^{N} x_{ij}^2$$

2. Initialize $\hat{\beta} = \left[\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_p\right]^T$ with a random value.

3. While not converged:

For $k = 1,2,\dots,p$:

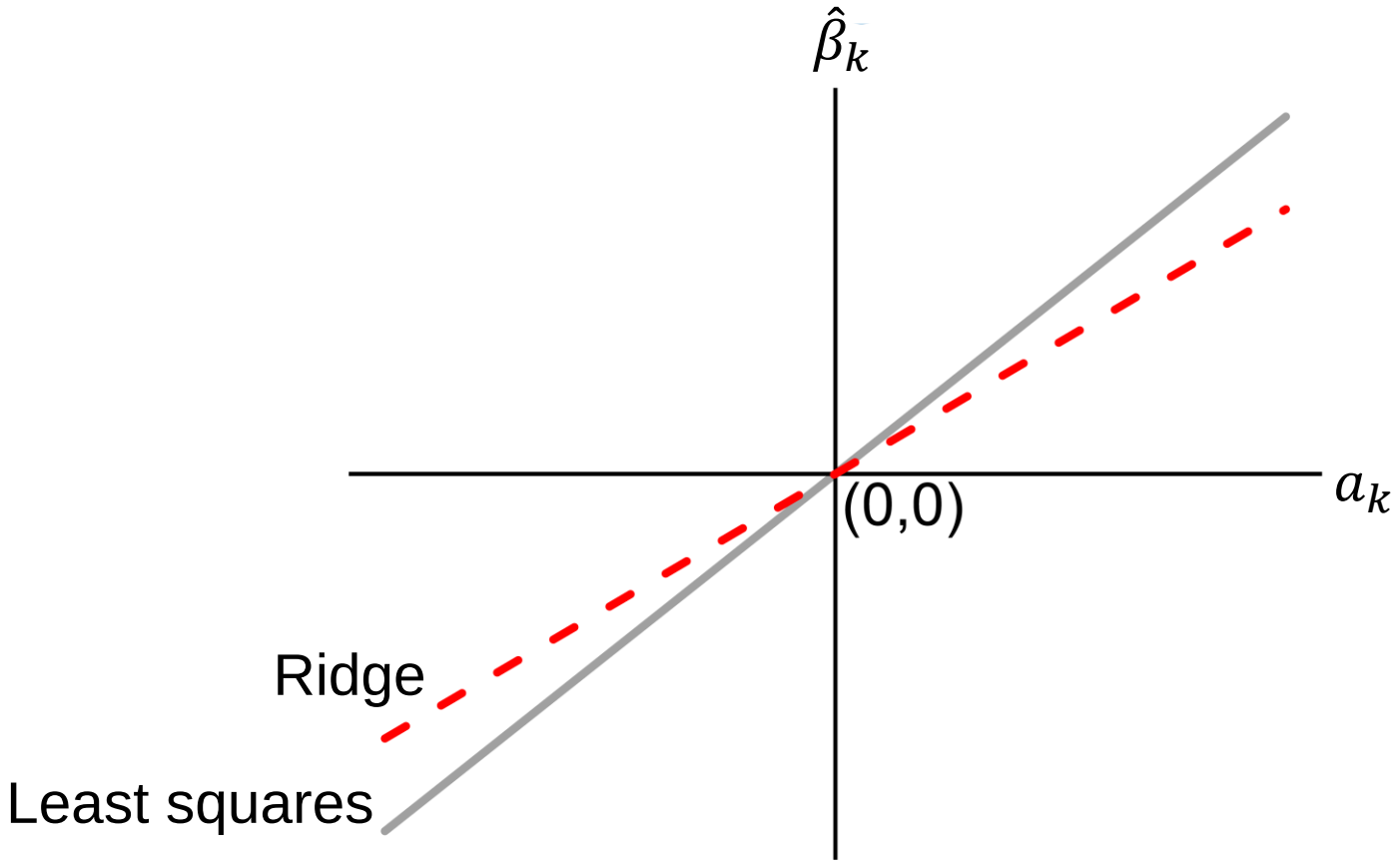
Compute a_k as

$$a_k = \sum_{i=1}^{N} x_{ik}\left(y_i - \sum_{\substack{j=1 \\ j\neq k}}^{p} x_{ij}\hat{\beta}_j\right)$$

Set $\hat{\beta}_k = \text{sign}(a_k)(|a_k| - \lambda/2)_+ / b_k$

Scaling of ridge regression illustrated

$$\hat{\beta}_k = \frac{a_k}{b_k + \lambda}$$



Adapted from Hastie *et al* (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd Ed.

Sparsity by soft thresholding and scaling with **elastic net**

We have that

$$\hat{\beta}_k = \begin{cases} \dfrac{a_k + \lambda(1-\alpha)/2}{b_k + \lambda\alpha} & \text{if } a_k < -\dfrac{\lambda(1-\alpha)}{2} \\ 0 & \text{if } a_k \in \left[-\dfrac{\lambda(1-\alpha)}{2}, \dfrac{\lambda(1-\alpha)}{2}\right] \\ \dfrac{a_k - \lambda(1-\alpha)/2}{b_k + \lambda\alpha} & \text{if } a_k > \dfrac{\lambda(1-\alpha)}{2} \end{cases}$$

or alternatively that

$$\hat{\beta}_k = \frac{\operatorname{sign}(a_k)}{b_k + \lambda\alpha}\left(|a_k| - \frac{\lambda(1-\alpha)}{2}\right)_+$$

where

$$\operatorname{sign}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases} \qquad x_+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Reduction to lasso, ridge regression, and least squares

$$\hat{\beta}_k = \frac{\text{sign}(a_k)}{b_k + \lambda\alpha}\left(|a_k| - \frac{\lambda(1-\alpha)}{2}\right)_+$$

When $\lambda = 0$, we have the **least squares** estimate

$$\hat{\beta}_k = \frac{a_k}{b_k}$$

When $\lambda > 0$ and $\alpha = 1$, we have the **ridge** estimate

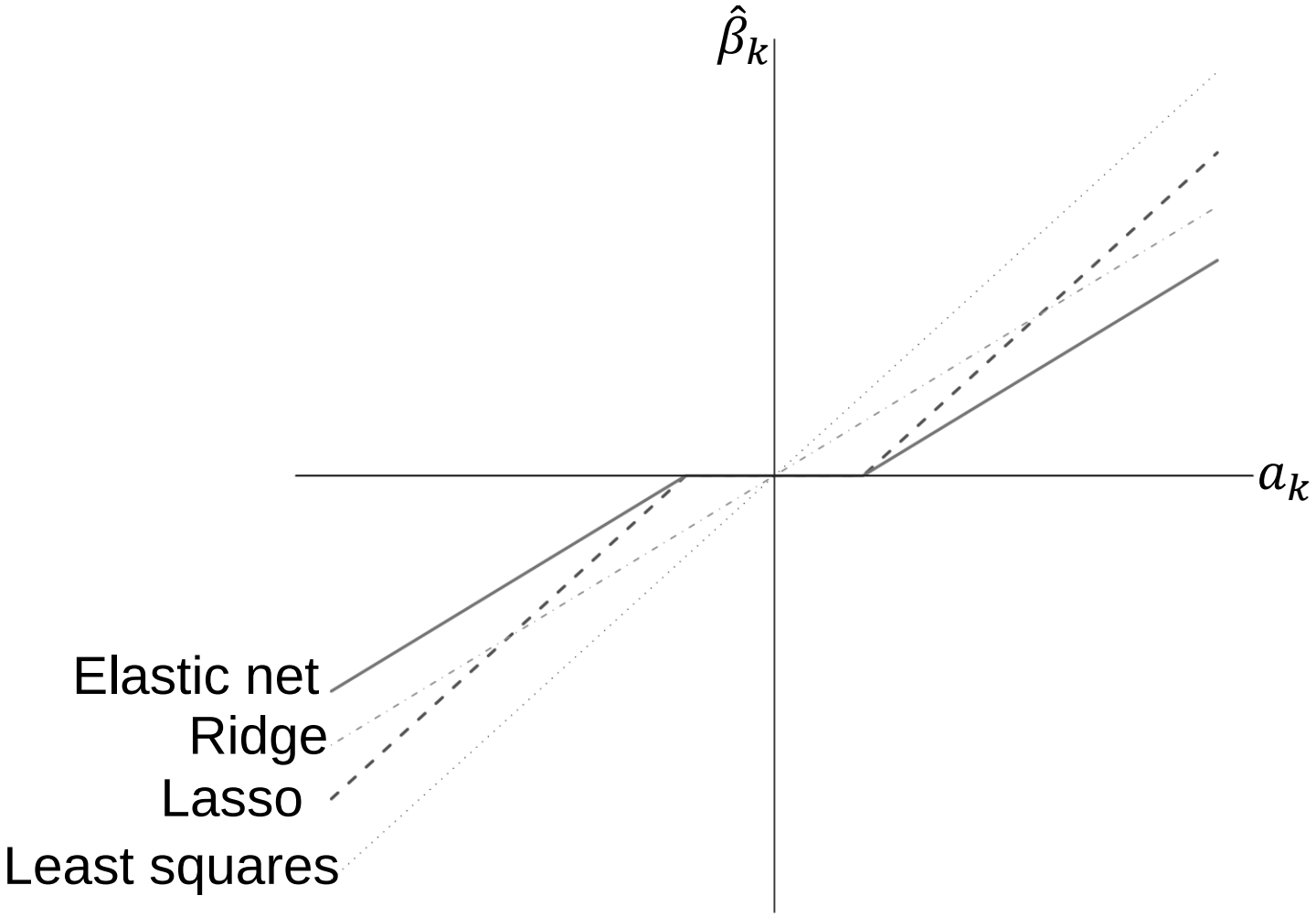
$$\hat{\beta}_k = \frac{a_k}{b_k + \lambda}$$

When $\lambda > 0$ and $\alpha = 0$, we have the **lasso** estimate

$$\hat{\beta}_k = \frac{\text{sign}(a_k)}{b_k}\left(|a_k| - \frac{\lambda}{2}\right)_+$$

Soft thresholding and scaling of **elastic net** illustrated

$$\hat{\beta}_k = \frac{\text{sign}(a_k)}{b_k + \lambda\alpha}\left(|a_k| - \frac{\lambda(1-\alpha)}{2}\right)_+$$



Zou and Hastie (2005) *J. R. Statist. Soc. B.* 67:301-320.

Coordinate descent algorithm for **elastic net**

1. Precompute $b_j, j = 1,2,\dots,p$ as

$$b_j = \sum_{i=1}^{N} x_{ij}^2$$

2. Initialized $\hat{\beta} = \left[\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_p\right]^T$ with a random value.

3. While not converged:

For $k = 1,2,\dots,p$:

Compute a_k as

$$a_k = \sum_{i=1}^{N} x_{ik}\left(y_i - \sum_{\substack{j=1 \\ j\neq k}}^{p} x_{ij}\hat{\beta}_j\right)$$

Set $\hat{\beta}_k = \text{sign}(a_k)(|a_k| - \lambda(1-\alpha)/2)_+/(b_k + \lambda\alpha)$

Trend filtered regression

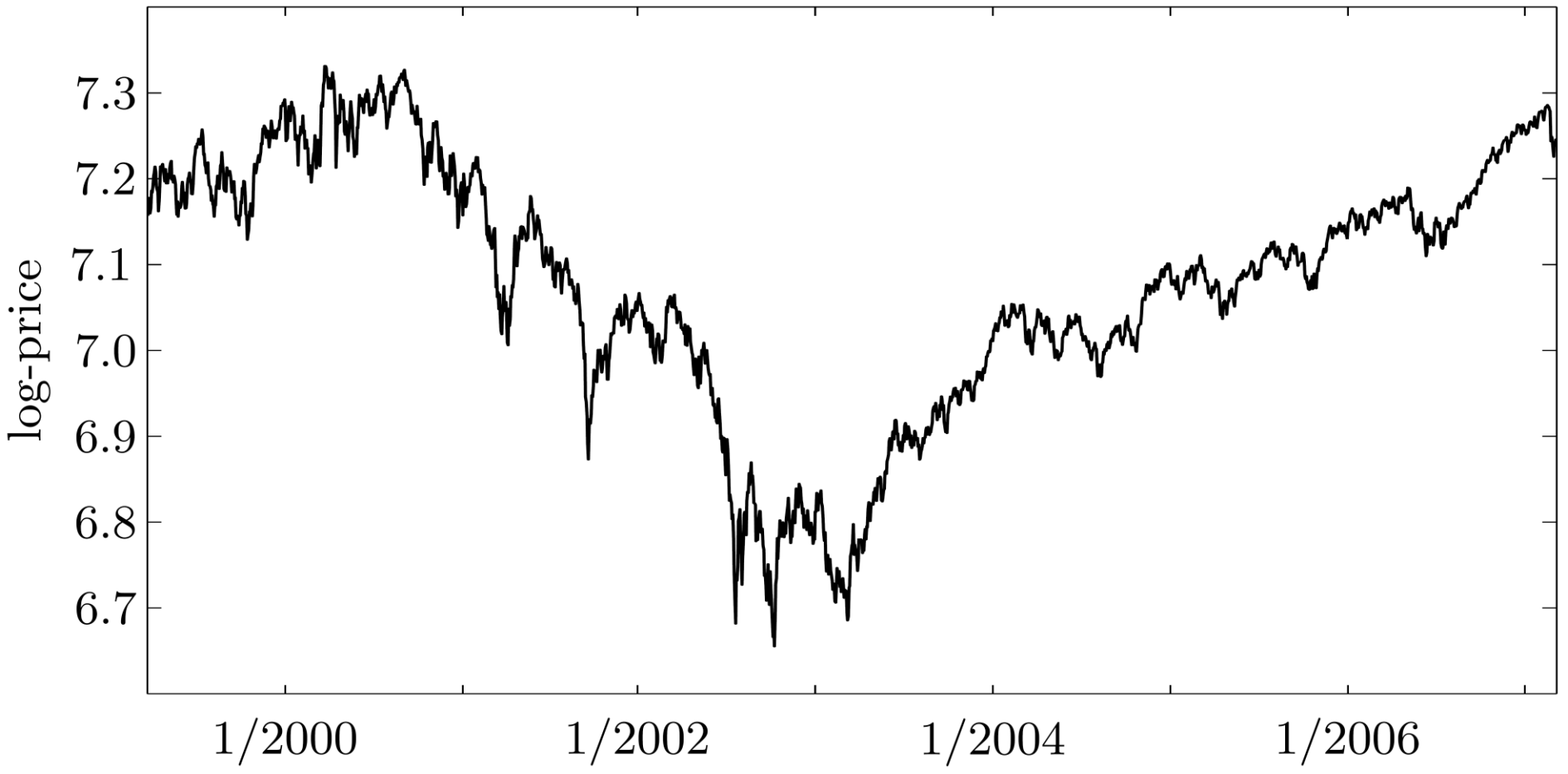
So far we have proposed penalized regression approaches (ridge regression, lasso, and elastic net) that penalize parameter values for getting too large.

These methods shrink individual parameter values associated with particular features to 0, with some of the formulations performing explicit feature selection by setting particular parameter values to 0 (lasso and elastic net).

However, for some problems, features are by construction highly correlated, and therefore it may be appropriate to penalize differences between parameters rather than the parameters themselves using an approach termed **trend filtering**.

One such problem is in the analysis of time-series data

Example: S&P 500 index prices from 1999 to 2007



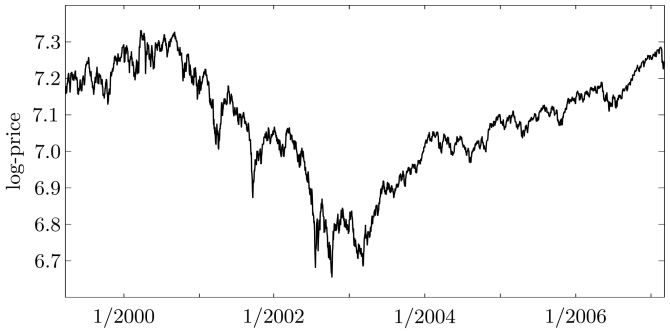
Kim *et al*. (2009) *SIAM Review* 51:339-360.

One such problem is in the analysis of time-series data

Suppose for such a dataset, we wanted to model the change in stock prices over time across a number of different stocks.

Then the input data for each stock $X = [X_1, X_2, \ldots, X_p]$ is a vector of p features measuring the price for this particular stock across a period of time, where feature j is a time point measurement prior to feature k for $j < k$.

Such data induces autocorrelation across X such that X_j will generally be more similar to X_{j-1} and X_{j+1} than to other features.

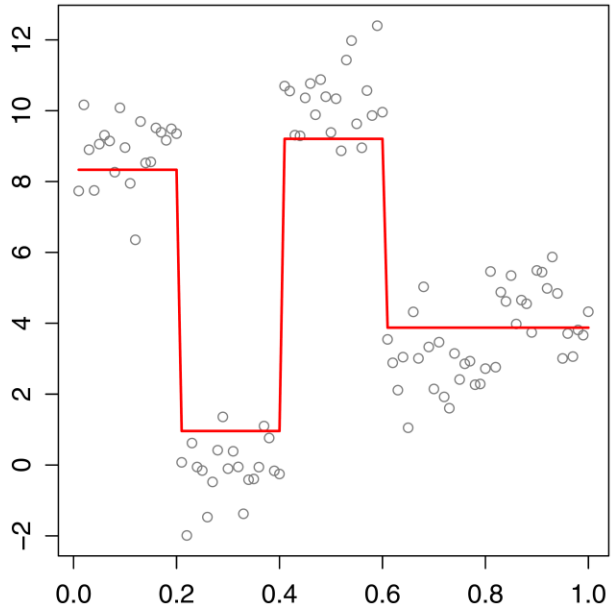


Choose to penalize differences in parameters of neighboring features.

Kim *et al*. (2009) *SIAM Review* 51:339-360.

L_2-norm piecewise constant trend filtered regression

Suppose we have features as depicted below for an observed data point that display an autocorrelation in their values.



We wish to fit a linear regression model while explicitly accounting for this autocorrelation.

We will model this autocorrelation using a piecewise constant penalty.

Tibshirani (2014) *Annals of Statistics* 42:285-323.

L_2-norm piecewise constant trend filtered regression

We will seek to minimize the cost function

$$J(\beta, \lambda) = \text{RSS}(\beta) + \lambda \sum_{j=1}^{p-1} \left(\beta_{j+1} - \beta_j\right)^2$$

where we are placing a penalty on large differences of parameter values for neighboring features.

If the features do indeed exhibit an autocorrelation, then we hope that the importance of feature $j + 1$ should be similar to the importance of feature j.

We will rewrite this in matrix notation as in prior lectures and try to find the parameter vector β, denoted $\hat{\beta}$, that minimizes $J(\beta, \lambda)$, assuming the response has been centered and inputs have been standardized.

Writing in matrix notation

Define the first-order difference matrix as $\mathbf{D} \in \mathbb{R}^{(p-1)\times p}$ as

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}$$

We have that $\mathbf{D}\beta \in \mathbb{R}^{p-1}$ is

$$\begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} = \begin{bmatrix} \beta_2 - \beta_1 \\ \beta_3 - \beta_2 \\ \vdots \\ \beta_p - \beta_{p-1} \end{bmatrix}$$

and so

$$\|\mathbf{D}\beta\|_2^2 = \sum_{j=1}^{p-1} \left(\beta_{j+1} - \beta_j\right)^2$$

L_2-norm piecewise constant trend filtered regression

We will seek to minimize the cost function

$$J(\beta, \lambda) = \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda\|\mathbf{D}\beta\|_2^2$$

where we are placing a penalty on large differences of parameter values for neighboring features.

We will return to the use of the matrices later, but for now will write the cost function in expanded form to derive a set of normal equations for L_2-norm piecewise constant trend filtered regression.

Finding the regression coefficients (normal equations)

As we performed for least squares regression and ridge regression, we can derive a set of normal equations for L_2-norm piecewise constant trend filtered regression by taking partial derivatives of the cost function $J(\beta, \lambda)$ with respect to the coefficient β_j of each feature j, and setting these derivatives to 0.

Recall that

$$J(\beta, \lambda) = \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p-1} \left(\beta_{j+1} - \beta_j \right)^2$$

Finding the regression coefficients (normal equations)

$$J(\beta,\lambda)=\sum_{i=1}^{N}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^{p-1}\left(\beta_{j+1}-\beta_j\right)^2$$

Taking the partial derivative with respect to β_1, we have

$$\frac{\partial}{\partial\beta_1}J(\beta,\lambda)=-2\sum_{i=1}^{N}x_{i1}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)+2\lambda(\beta_1-\beta_2)$$

$$=-2\mathrm{x}_1^T(\mathbf{y}-\mathbf{X}\beta)+2\lambda(\beta_1-\beta_2)$$

Taking the partial derivative with respect to β_p, we have

$$\frac{\partial}{\partial\beta_p}J(\beta,\lambda)=-2\sum_{i=1}^{N}x_{ip}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)+2\lambda\left(\beta_p-\beta_{p-1}\right)$$

$$=-2\mathrm{x}_p^T(\mathbf{y}-\mathbf{X}\beta)+2\lambda\left(\beta_p-\beta_{p-1}\right)$$

Finding the regression coefficients (normal equations)

$$J(\beta,\lambda)=\sum_{i=1}^{N}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^{p-1}\left(\beta_{j+1}-\beta_j\right)^2$$

Taking the partial derivative with respect to β_k, $1<k<p$, we have

$$\frac{\partial}{\partial\beta_k}J(\beta,\lambda)=-2\sum_{i=1}^{N}x_{ik}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)+2\lambda[\beta_k-\beta_{k-1}-(\beta_{k+1}-\beta_k)]$$

$$=-2\sum_{i=1}^{N}x_{ik}\left(y_i-\sum_{j=1}^{p}x_{ij}\beta_j\right)+2\lambda(-\beta_{k-1}+2\beta_k-\beta_{k+1})$$

$$=-2\mathrm{x}_k^T(\mathbf{y}-\mathbf{X}\beta)+2\lambda(-\beta_{k-1}+2\beta_k-\beta_{k+1})$$

Finding β (normal equations) that minimizes $J(\beta, \lambda)$

Putting this together, to identify the minimum, we take the gradient with respect to β, which is a p-dimensional vector, with dimension k the partial derivative of $J(\beta, \lambda)$ with respect to β_k, $k \in \{1,2, \dots, p\}$. The gradient is therefore

$$\frac{\partial}{\partial \beta} J(\beta, \lambda) = -2 \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{N1} \\ x_{12} & x_{22} & \cdots & x_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1p} & x_{2p} & \cdots & x_{Np} \end{bmatrix} (\mathbf{y} - \mathbf{X}\beta) + 2\lambda \begin{bmatrix} \beta_1 - \beta_2 \\ -\beta_1 + 2\beta_2 - \beta_3 \\ \vdots \\ \beta_p - \beta_{p-1} \end{bmatrix}$$

$$= -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) + 2\lambda \begin{bmatrix} \beta_1 - \beta_2 \\ -\beta_1 + 2\beta_2 - \beta_3 \\ \vdots \\ \beta_p - \beta_{p-1} \end{bmatrix}$$

How can we compactly write the last term?

Finding β (normal equations) that minimizes $J(\beta, \lambda)$

Recall the first-order difference matrix

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}$$

For simplicity, let's assume $p = 4$. Then $\mathbf{D}^T\mathbf{D} \in \mathbb{R}^{p \times p}$ is

$$\begin{bmatrix} -1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

Showing that the immediate sub- and super-diagonals are all -1s, and that the diagonal is all 2s, except for the first and last element, which are 1s.

Finding β (normal equations) that minimizes $J(\beta, \lambda)$

More generally we have

$$\mathbf{D}^T\mathbf{D} = \begin{bmatrix} 1 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 1 \end{bmatrix}$$

and we can see that the final term in the gradient can be written as

$$\mathbf{D}^T\mathbf{D}\beta = \begin{bmatrix} \beta_1 - \beta_2 \\ -\beta_1 + 2\beta_2 - \beta_3 \\ -\beta_2 + 2\beta_3 - \beta_4 \\ \vdots \\ \beta_p - \beta_{p-1} \end{bmatrix}$$

yielding

$$\frac{\partial}{\partial\beta}J(\beta, \lambda) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) + 2\lambda\mathbf{D}^T\mathbf{D}\beta$$

Finding β (normal equations) that minimizes $J(\beta, \lambda)$

Setting the gradient to the 0 vector, gives

$$-\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) + \lambda \mathbf{D}^T \mathbf{D}\beta = 0$$

which yields

$$-\mathbf{X}^T\mathbf{y} + \mathbf{X}^T\mathbf{X}\beta + \lambda \mathbf{D}^T \mathbf{D}\beta = -\mathbf{X}^T\mathbf{y} + (\mathbf{X}^T\mathbf{X} + \lambda \mathbf{D}^T \mathbf{D})\beta = 0$$

or

$$(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{D}^T \mathbf{D})\beta = \mathbf{X}^T\mathbf{y}$$

If $(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{D}^T \mathbf{D})$ is invertible, then the estimated coefficients are

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda \mathbf{D}^T \mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}$$

The “hat” or projection matrix under trend filtering

We have now shown that $\hat{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}$ is the L_2-norm piecewise constant trend filter estimate of the regression coefficients given training observations (x_i, y_i), $i = 1,2, \ldots, N$.

The fitted value vector (estimate of the output of the original training data), is given by

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}$$

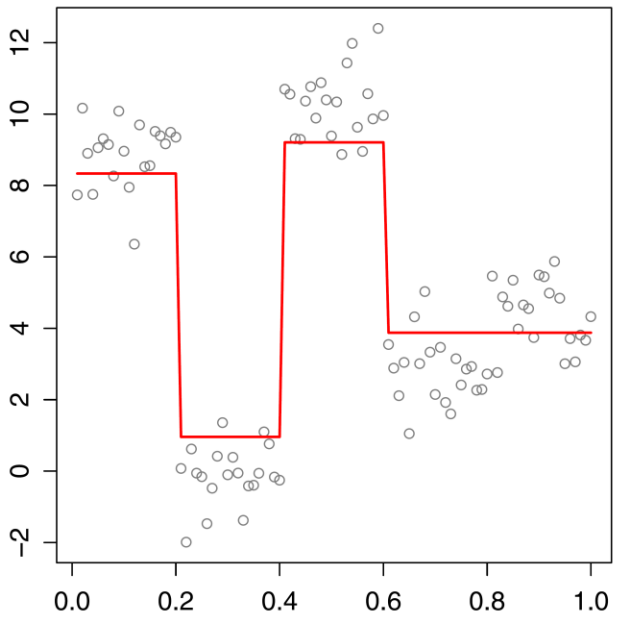
Define the hat (projection) matrix under piecewise constant trend filtered regression, denoted by $\mathbf{H}_\lambda$, as

$$\mathbf{H}_\lambda = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{X}^T$$

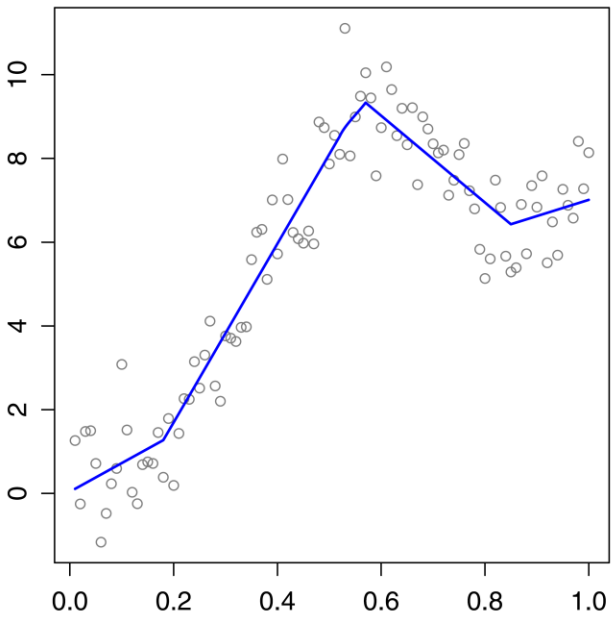
such that $\hat{\mathbf{y}} = \mathbf{H}_\lambda\mathbf{y}$.

L_2-norm trend filtering in general

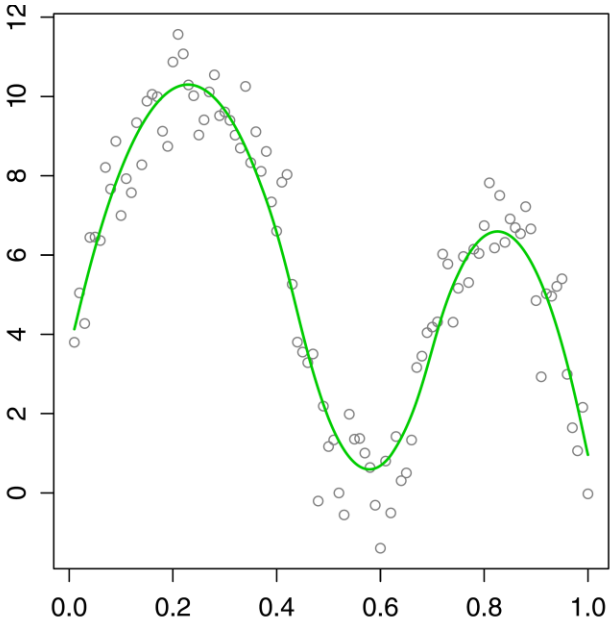
Piecewise constant



Piecewise linear



Piecewise quadratic



Different trend penalties represent finite difference approximations to a dth derivative:

Piecewise constant: first derivative

Piecewise linear: second derivative

Piecewise quadratic: third derivative

Tibshirani (2014) *Annals of Statistics* 42:285-323.

L_2-norm trend filtering in general

In general, an L_2-norm trend filtered regression that approximates the dth derivative is given by the cost function

$$J(\beta, \lambda) = \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda\|\mathbf{D}\beta\|_2^2$$

where $\mathbf{D} \in \mathbb{R}^{(p-d)\times p}$ is the dth-order difference matrix.

Parallel to constant trend filtering, the gradient is given by

$$\frac{\partial}{\partial\beta} J(\beta, \lambda) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) + 2\lambda\mathbf{D}^T\mathbf{D}\beta$$

and parameter estimates via normal equations are

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}$$

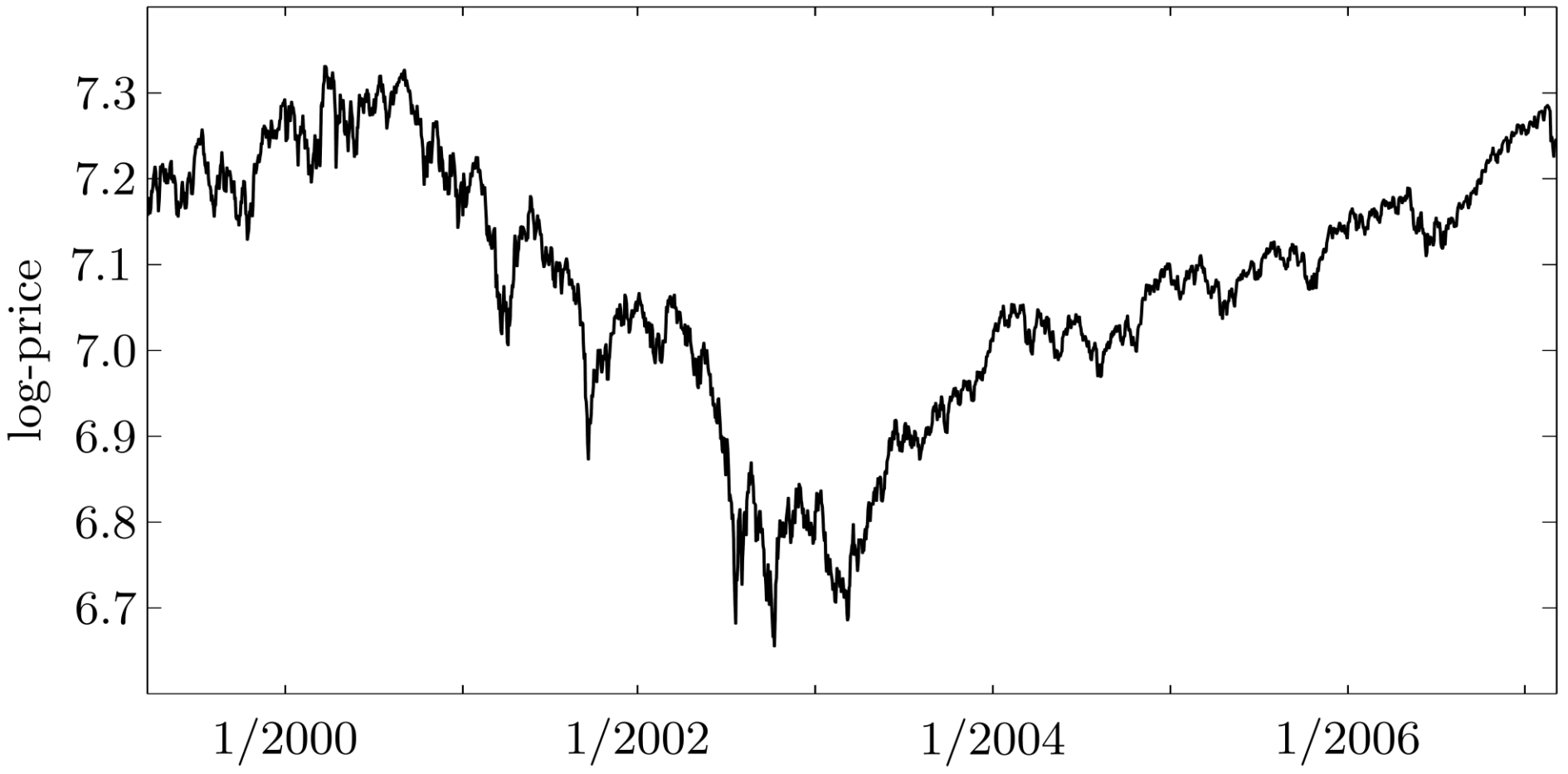
L_1-norm trend filtering used more often in general

L_1-norm trend filtering is used more often than L_2-norm, and seeks to find a model that is less smooth than L_2-norm trend filtering, but in a sense is simpler as many dth-order differences are exactly 0.

For example, for piecewise constant trend filtering, the L_1-norm penalty may lead to $\hat{\beta}_{j+1} - \hat{\beta}_j = 0$ for many j, whereas these terms will all be non-zero for L_2-norm penalties.

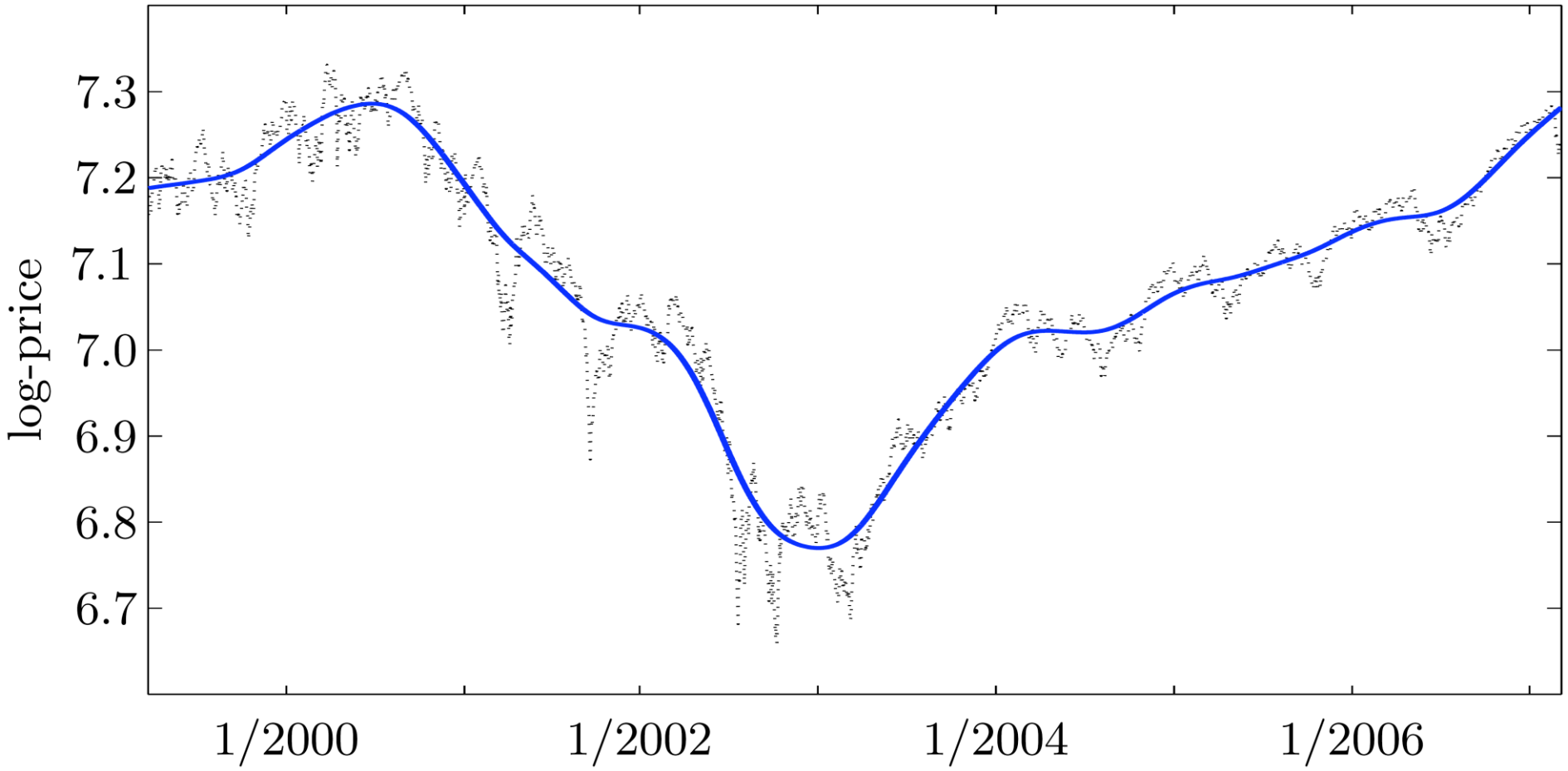
To illustrate, we will consider a second-order trend penalty using the L_1-norm and L_2-norm applied to data from S&P 500 Index from 1999 to 2007.

S&P 500 Index prices from 1999 to 2007



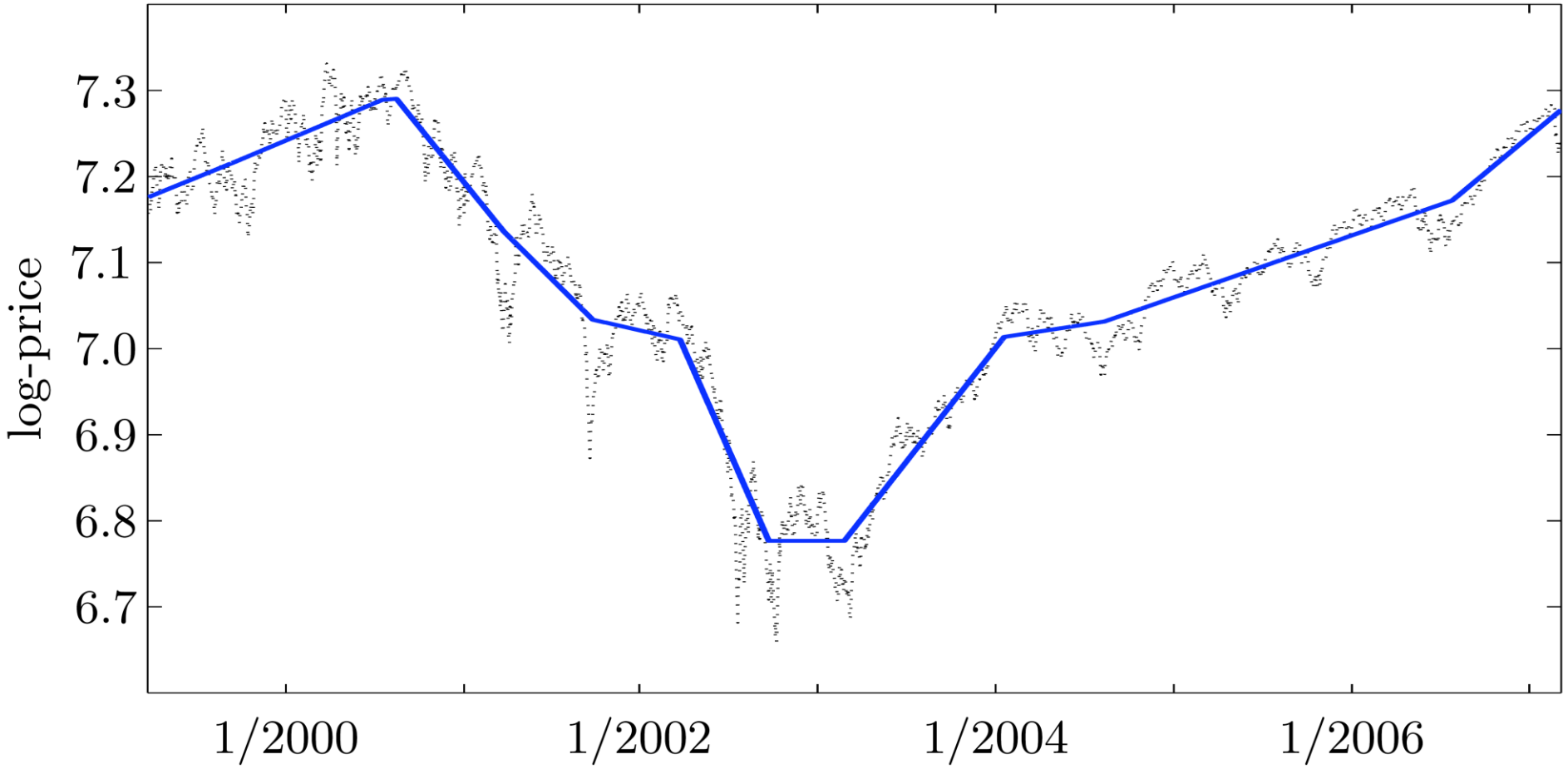
Kim *et al*. (2009) *SIAM Review* 51:339-360.

Applying piecewise linear L_2-norm filter (H-P filtering)



Kim *et al*. (2009) *SIAM Review* 51:339-360.

Applying piecewise linear L_1-norm filter



Kim *et al*. (2009) *SIAM Review* 51:339-360.

Fused lasso as a special case of L_1 trend filtering

A widely-employed approach for feature selection in which neighboring features are highly correlated is **fused lasso**, where we not only include an L_1-norm trend penalty, but also include an L_1-norm penalty on the parameter vector.

The cost function for fused lasso is given by

$$J(\beta, \lambda_1, \lambda_2) = \mathrm{RSS}(\beta) + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j|$$

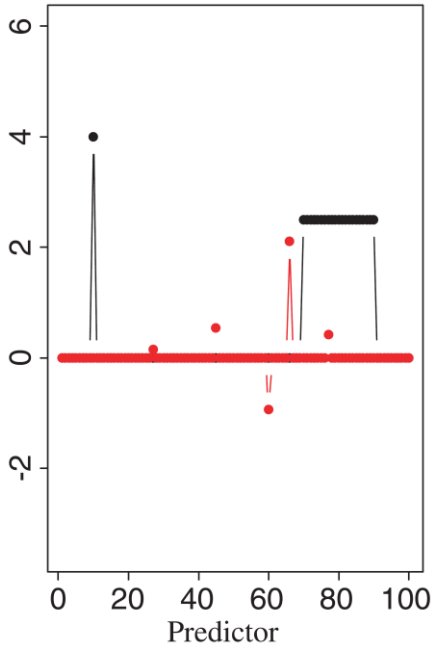
which can be rewritten compactly in matrix notation as

$$J(\beta, \lambda_1, \lambda_2) = \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\mathbf{D}\beta\|_1$$

where $\mathbf{D} \in \mathbb{R}^{(p-1)\times p}$ is the first-order difference matrix.

Example application of fused lasso

Simulated dataset (black) of $p = 100$ features with only two regions of non-zero coefficients (parameters).



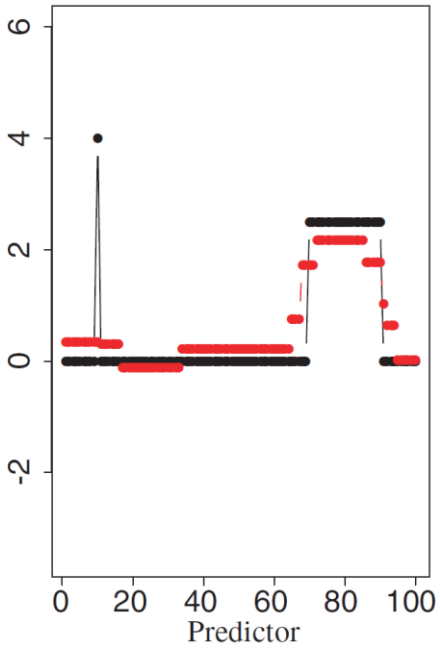
The lasso estimate (red) with the following cost function misses both true areas of non-zero parameters.

$$J(\beta, \lambda) = \mathrm{RSS}(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Example application of fused lasso

Simulated dataset (black) of $p = 100$ features with only two regions of non-zero coefficients (parameters).



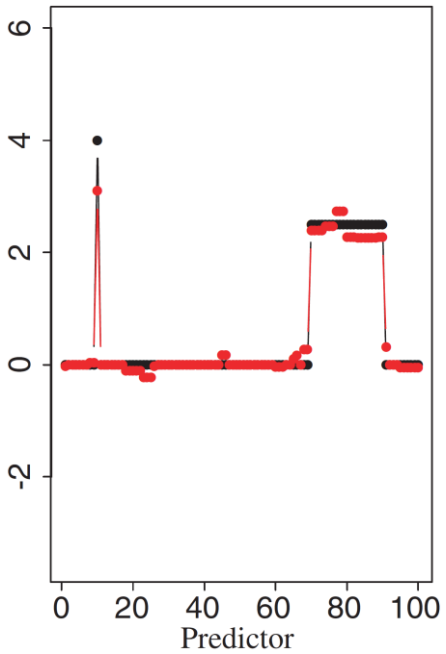
The L_1-norm trend filter (fusion) estimate (red) with the following cost function misses the smaller true area of non-zero parameter.

$$J(\beta, \lambda) = \mathrm{RSS}(\beta) + \lambda \sum_{j=1}^{p-1} \left| \beta_{j+1} - \beta_j \right|$$

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Example application of fused lasso

Simulated dataset (black) of $p = 100$ features with only two regions of non-zero coefficients (parameters).



The fused lasso estimate (red) with the following cost function identifies both true areas of non-zero parameters.

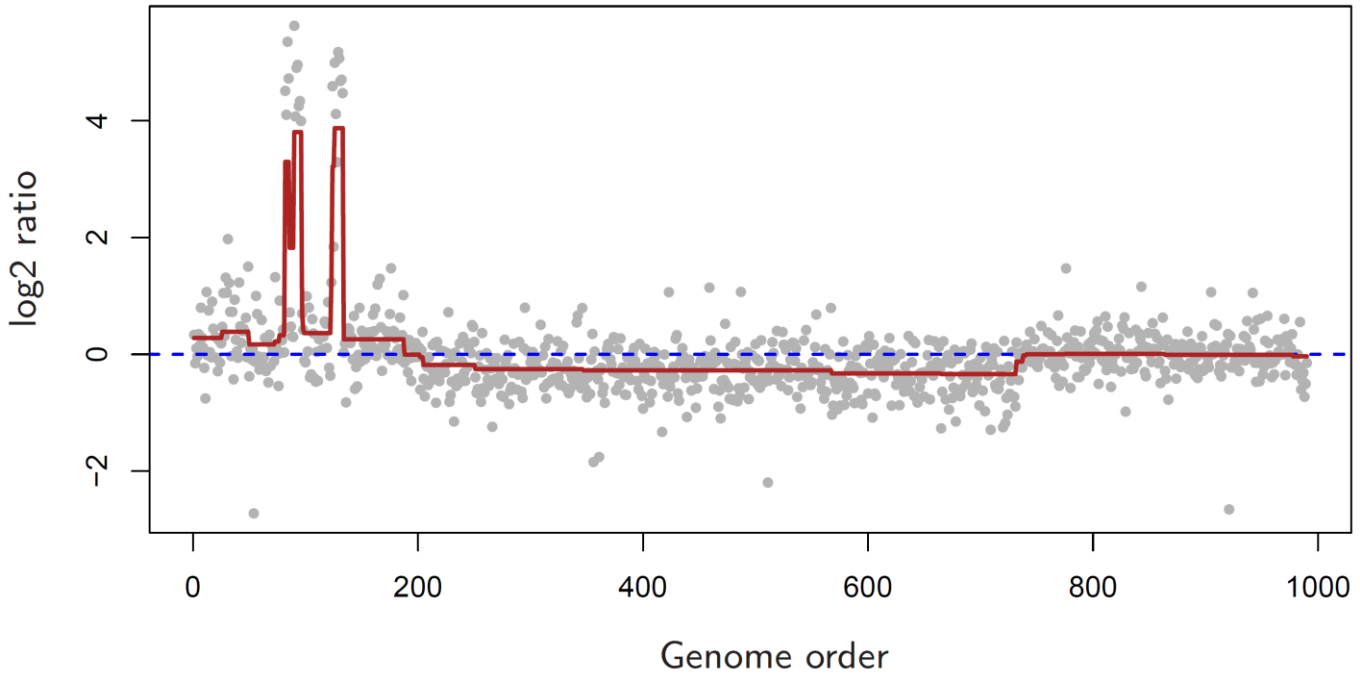
$$J(\beta, \lambda_1, \lambda_2) = \mathrm{RSS}(\beta) + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j|$$

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Example application of fused lasso on real data

Array-comparative genomic hybridization (aCGH) is an experimental technique that can detect whether an individual has fewer or greater number of copies of a particular genomic region than expected.

Deviant number of copies are based on the ratio of intensities measured by a target and a reference genome.



Hastie *et al* (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd Ed.

Alternate formulation of optimization problem

An alternate formulation to finding β that minimizes

$$J(\beta, \lambda_1, \lambda_2) = \mathrm{RSS}(\beta) + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j|$$

is to instead solve the constrained problem of finding β that minimizes

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2$$

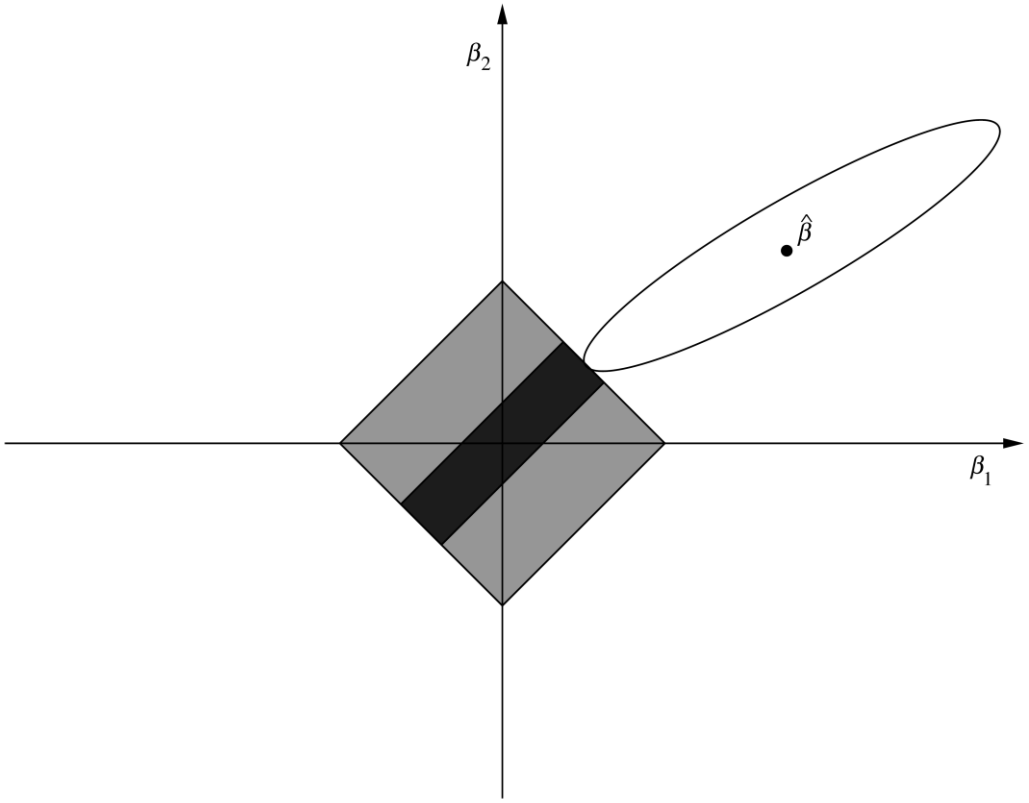
subject to the constraints that

$$\sum_{j=1}^{p} |\beta_j| \leq s_1 \quad \text{and} \quad \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j| \leq s_2$$

for some non-negative values s_1 and s_2.

Illustration based on alternate formulation

Contour of the $\mathrm{RSS}(\beta)$ and the constraint functions $|\beta_1| + |\beta_2| \leq s_1$ (gray) and $|\beta_2 - \beta_1| \leq s_2$ (black) for fused lasso with two features.

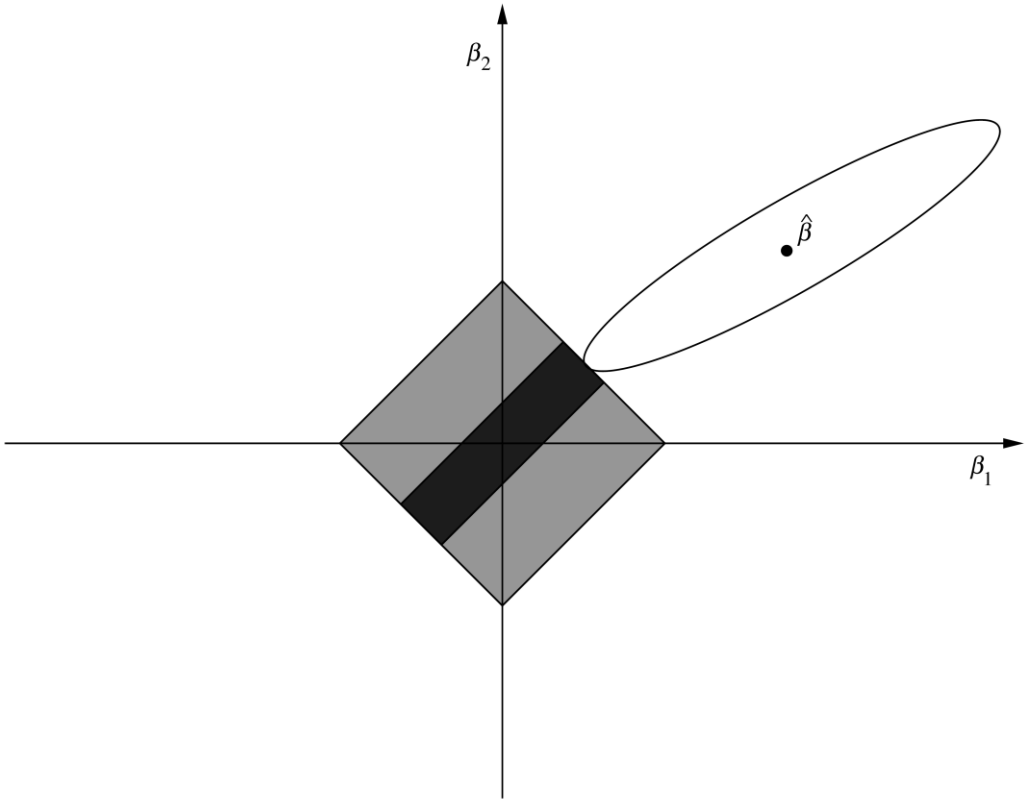


If s_1 and s_2 are large (like λ_1 and λ_2 near 0), then constraint region contains the least squares estimate $\hat{\beta}$.

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Illustration based on alternate formulation

Contour of the $\mathrm{RSS}(\beta)$ and the constraint functions $|\beta_1| + |\beta_2| \leq s_1$ (gray) and $|\beta_2 - \beta_1| \leq s_2$ (black) for fused lasso with two features.

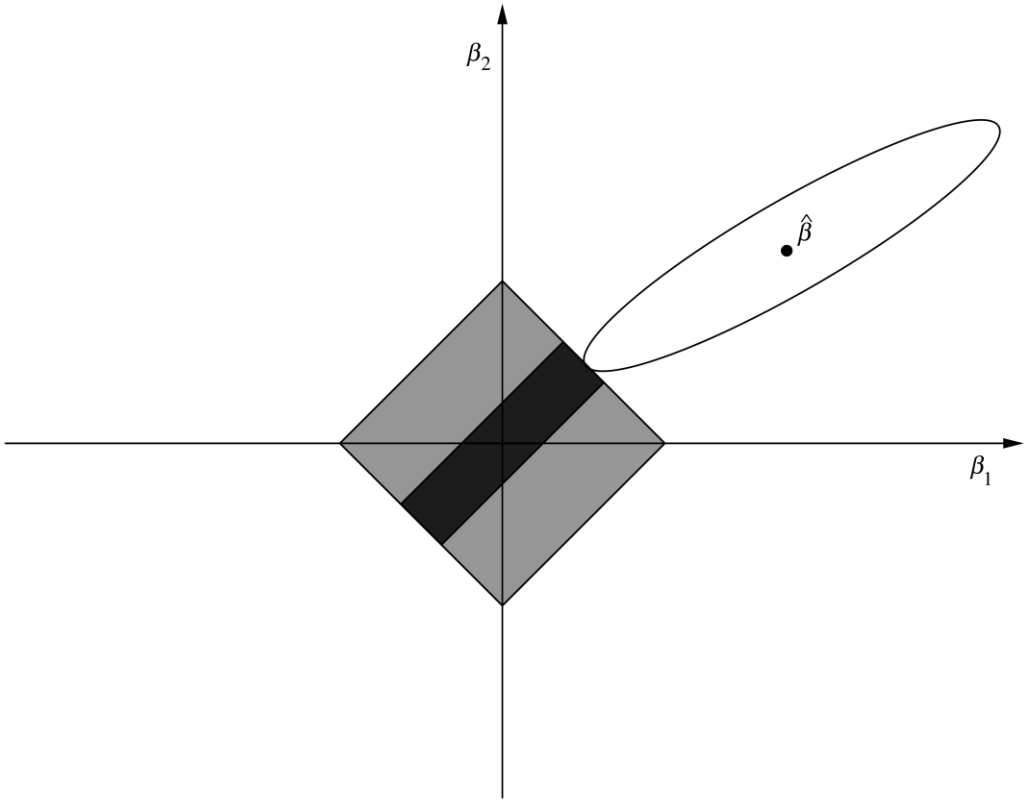


However, when least squares estimate resides outside the constraint regions, then fused lasso estimate is different.

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Illustration based on alternate formulation

Contour of the $\mathrm{RSS}(\beta)$ and the constraint functions $|\beta_1| + |\beta_2| \leq s_1$ (gray) and $|\beta_2 - \beta_1| \leq s_2$ (black) for fused lasso with two features.



The fused lasso estimate will lie on the boundary of the constraint regions, and the point it touches the $\mathrm{RSS}(\beta)$, may occur on the edge of the black band, leading to a coefficient difference close to 0.

Tibshirani *et al*. (2005) *J. R. Statist. Soc. B* 67:91-108.

Sometimes features belong to groups

Sometimes features belong to the same group.

For example, many genes belong to the same biological pathway, and so it may be reasonable that either all genes in a pathway are included in a model or that none of them are included.

In addition, recall that working with qualitative features a qualitative feature with m levels can be represented by $m - 1$ dummy variables. For example, we could represent a feature that takes the 3 values urban, suburban, and rural as the 2 dummy variables

$$X_1 = \begin{cases} 1 & \text{urban} \\ 0 & \text{not urban} \end{cases} \qquad X_2 = \begin{cases} 1 & \text{rural} \\ 0 & \text{not rural} \end{cases}$$

It would therefore be reasonable to shrink both β_1 and β_2 together.

Grouped lasso to penalize groups of features

Grouped lasso is one mechanism of shrinking coefficients for features in the same group together, while encouraging sparsity across groups.

Consider p features that can be divided into L groups, with p_ℓ the number of features in group ℓ, $\ell = 1,2, \ldots, L$.

Let $\mathbf{X}_\ell \in \mathbb{R}^{N \times p_\ell}$ be the design matrix for the p_ℓ features of group ℓ.

The complete design matrix is therefore

$$\mathbf{X} = [\mathbf{X}_1 \quad \mathbf{X}_2 \quad \cdots \quad \mathbf{X}_L]$$

Let $\beta_\ell \in \mathbb{R}^{p_\ell}$ be the coefficients associated with features of group ℓ.

Grouped lasso to penalize groups of features

We therefore get the complete coefficient vector as

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_L \end{bmatrix}$$

The cost function for grouped lasso is (**note not squared L_2 norm**)

$$J(\beta, \lambda) = \mathrm{RSS}(\beta) + \lambda \sum_{\ell=1}^{L} \sqrt{p_\ell} \cdot \|\beta_\ell\|_2$$

$$= \left\| \mathbf{y} - \sum_{\ell=1}^{L} \mathbf{X}_\ell \beta_\ell \right\|_2^2 + \lambda \sum_{\ell=1}^{L} \sqrt{p_\ell} \cdot \|\beta_\ell\|_2$$

where $\sqrt{p_\ell}$ accounts for varying group sizes

Reduction to lasso when $L = p$

When $L = p$, we have $p_\ell = 1$, $\|\beta_\ell\|_2 = \sqrt{\beta_\ell^2} = |\beta_\ell|$, and $\mathbf{X}_\ell = \mathrm{x}_\ell$.

Substituting into the cost function we have

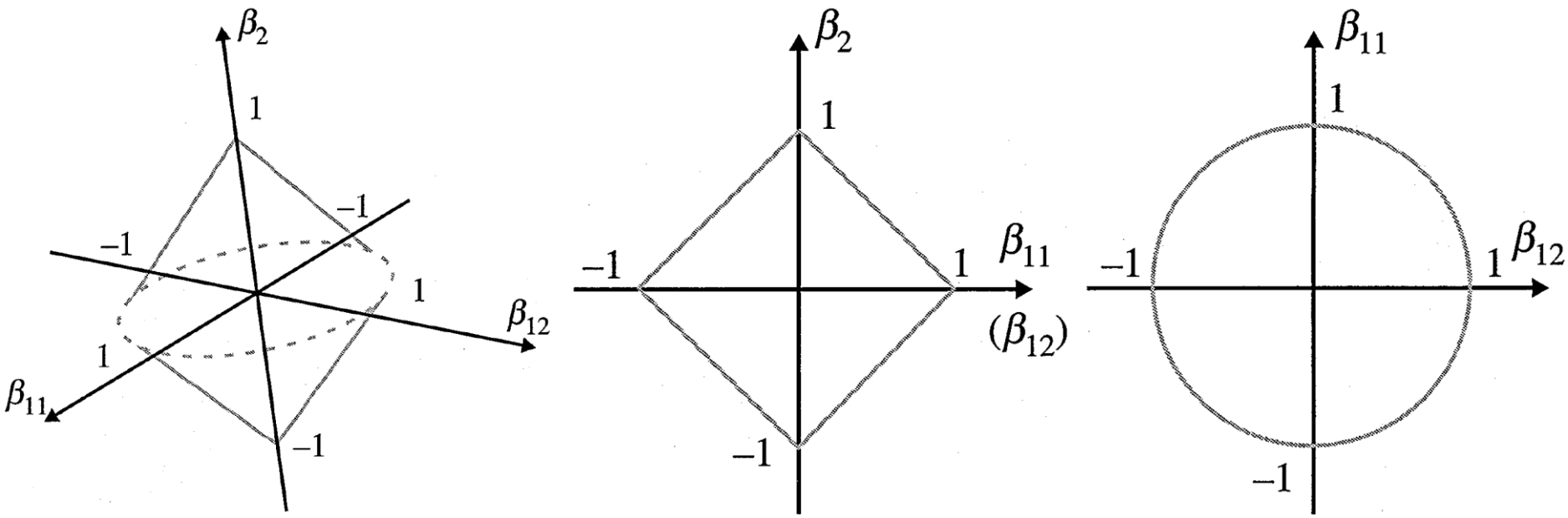
$$J(\beta, \lambda) = \left\| \mathbf{y} - \sum_{j=1}^{p} \mathrm{x}_j \beta_j \right\|_2^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

$$= \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \|\beta\|_1$$

$$= \mathrm{RSS}(\beta) + \lambda \|\beta\|_1$$

Illustration of group lasso

Example has $p = 3$ features, and 2 groups, with parameters of group 1 as $\beta_1 = [\beta_{11}, \beta_{12}]^T$ and group 2 as β_2, satisfying group lasso penalty $\|\beta_1\|_2 + |\beta_2| = 1$.

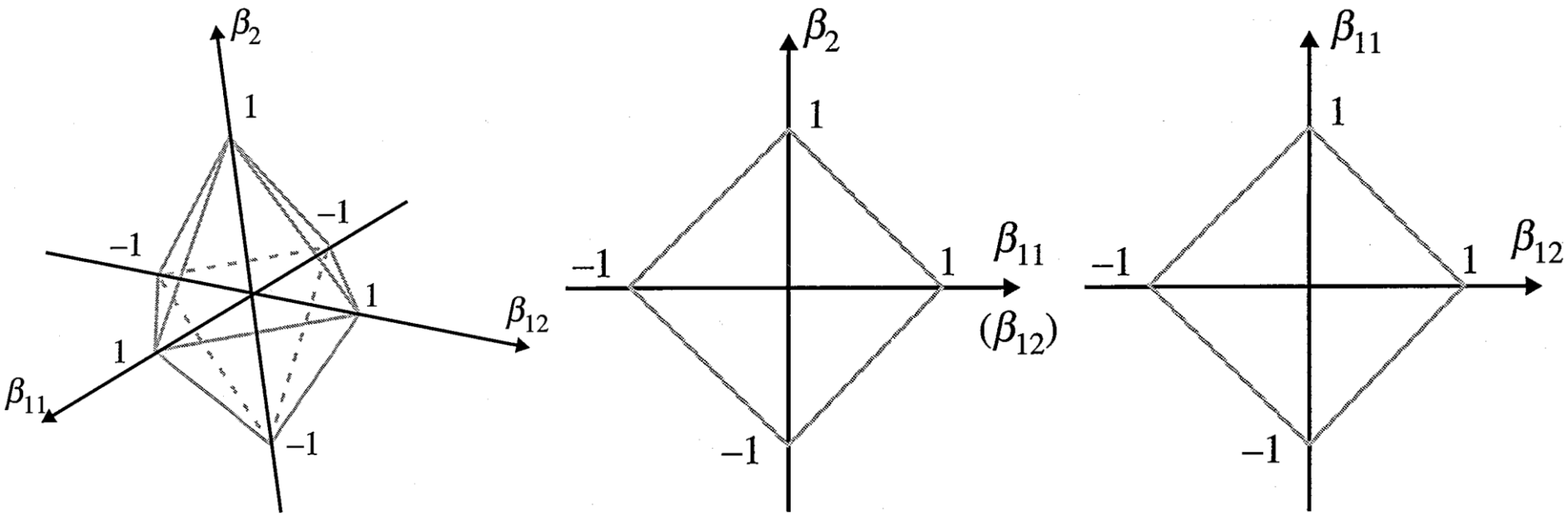


Group lasso encourages sparsity between groups, but not within groups, and shrinks all parameters to 0 within a group simultaneously.

Yuan, Li (2006) *J. R. Statist. Soc. B* 68:49-67.

Similar illustration, but with lasso penalty

Example has $p = 3$ features, and 2 groups, with parameters of group 1 as $\beta_1 = [\beta_{11}, \beta_{12}]^T$ and group 2 as β_2 satisfying lasso penalty $\|[\beta_1, \beta_2]^T\|_1 = 1$.

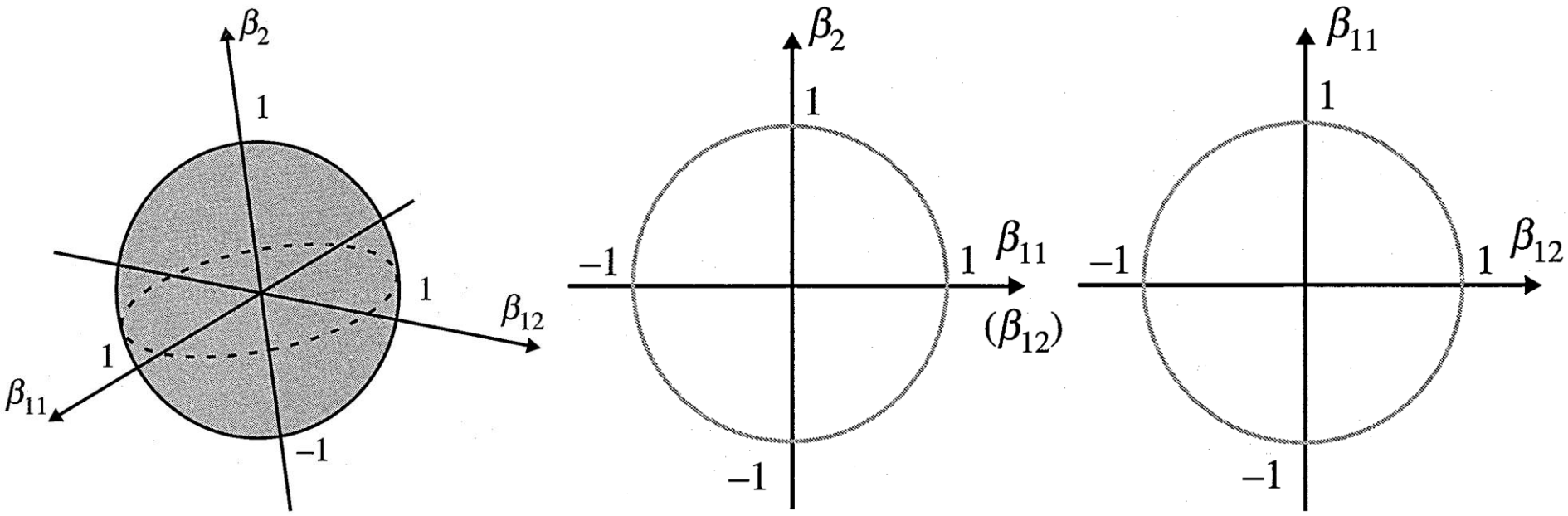


Lasso in contrast encourages sparsity across all parameters, regardless of their grouping.

Yuan, Li (2006) *J. R. Statist. Soc. B* 68:49-67.

Similar illustration, but with ridge penalty

Example has $p = 3$ features, and 2 groups, with parameters of group 1 as $\beta_1 = [\beta_{11}, \beta_{12}]^T$ and group 2 as β_2 satisfying ridge penalty $\|[\beta_1, \beta_2]^T\|_2^2 = 1$.



Ridge regression does not encourage sparsity and parameter or group levels, and instead shrinks all parameters to 0 simultaneously.

Yuan, Li (2006) *J. R. Statist. Soc. B* 68:49-67.

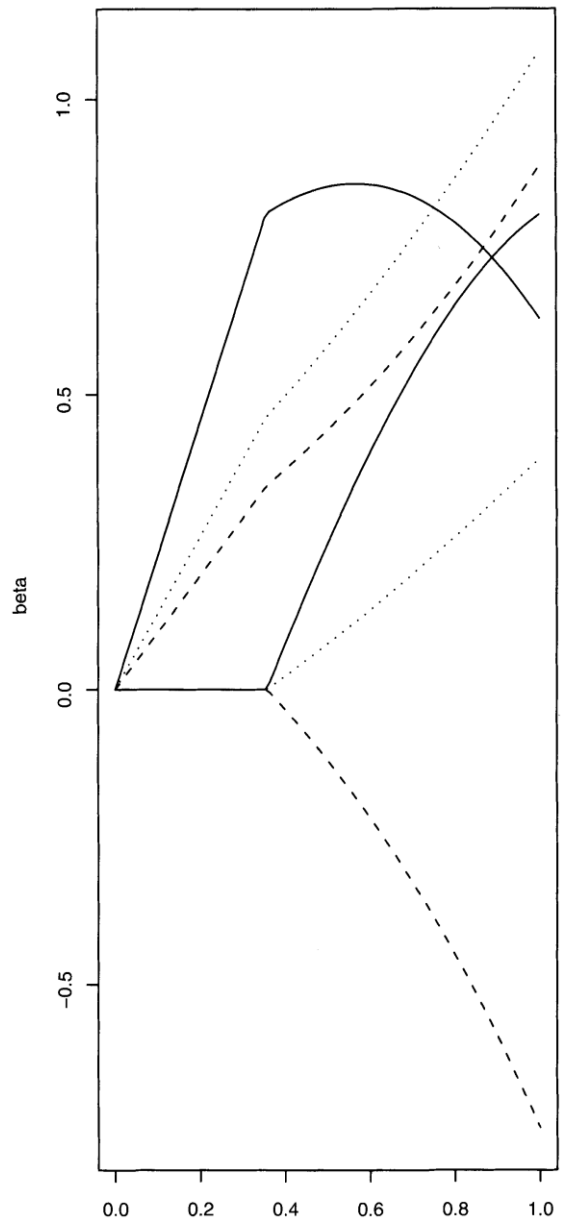
Parameter estimates as function of shrinkage

Here we plot estimated regression coefficients on the y-axis as a function of their shrinkage level.

Shrinkage level of 1 is associated with $\lambda = 0$, and shrinkage level tends to 0 as $\lambda \to \infty$.

This plots illustrates 2 groups each with 3 features (solid, dashed, and dotted curves).

One group shrinks to 0 and is eventually selected out, with the other group shrinking to 0 and becoming selected out near shrinkage level of 0.



Yuan, Li (2006) *J. R. Statist. Soc. B* 68:49-67.