Chapters 5 and 6: lasso and its extensions

Esa Ollila

Department of Signal Processing and Acoustics
Aalto University, Finland

Large Scale Data Analysis / Aalto University

Linear regression model recap

- Data: $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.
- output (response) $y_i \in \mathbb{R}$ is associated with inputs (predictors) $\mathbf{x}_i^\top = (x_{i1}, \ldots, x_{ip}) \in \mathbb{R}^p$
- Linear predictor function:

$$f(\mathbf{x}_i) = \beta_0 + \sum_{j=1}^p x_{ij}\beta_j$$

- Linear model:

$$y_i = \quad f(\mathbf{x}_i) \quad + \varepsilon_i$$

  where error terms ε_i, $i = 1, \ldots, N$ account for the modeling and measurement errors.
- Goal: estimate the vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\top \in \mathbb{R}^p$ of regression coefficients and the intercept $\beta_0 \in \mathbb{R}$ given $\mathcal{T}$.

Linear regression model recap

- Data: $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.
- output (response) $y_i \in \mathbb{R}$ is associated with inputs (predictors) $\mathbf{x}_i^\top = (x_{i1}, \ldots, x_{ip}) \in \mathbb{R}^p$
- Linear predictor function:

$$f(\mathbf{x}_i) = \beta_0 + \sum_{j=1}^p x_{ij}\beta_j$$

- Linear model:

$$y_i = \beta_0 + \sum_{j=1}^p x_{ij}\beta_j + \varepsilon_i$$

  where error terms ε_i, $i = 1, \ldots, N$ account for the modeling and measurement errors.
- Goal: estimate the vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\top \in \mathbb{R}^p$ of regression coefficients and the intercept $\beta_0 \in \mathbb{R}$ given $\mathcal{T}$.

In matrix-vector notations

$$
\begin{cases}
y_1 & = \beta_0 + x_{11}\beta_1 \ + \ldots + x_{1p}\beta_p \ + \varepsilon_1 \\
& \vdots \qquad\qquad\qquad \vdots \\
y_N & = \beta_0 + x_{N1}\beta_1 + \ldots + x_{Np}\beta_p + \varepsilon_N
\end{cases}
$$

$$
\boxed{\mathbf{y} = \beta_0 \mathbf{1} + \beta_1 \boldsymbol{x}_1 + \ldots + \beta_p \boldsymbol{x}_p + \boldsymbol{\varepsilon}}
$$

where

$\mathbf{1} =$ N-vector of 1's

$\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_N)^\top \in \mathbb{R}^N$ is the noise vector

$\mathbf{X} = \begin{pmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_p \end{pmatrix}$ is $N \times p$ matrix of inputs

Note: $\mathbf{x}_i \in \mathbb{R}^p$ denotes a (transposed) ith row-vector of $\mathbf{X}$ while $\boldsymbol{x}_i \in \mathbb{R}^N$ denotes the ith column $\boldsymbol{x}_i$.

In matrix-vector notations

$$\begin{cases} y_1 &= \beta_0 + \mathbf{x}_1^\top \boldsymbol{\beta} + \varepsilon_1 \\ &\vdots \\ y_N &= \beta_0 + \mathbf{x}_N^\top \boldsymbol{\beta} + \varepsilon_N \end{cases}$$

$$\boxed{\mathbf{y} = \beta_0 \mathbf{1} + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}}$$

where

$$\begin{aligned} \mathbf{1} &= N\text{-vector of } 1\text{'s} \\ \boldsymbol{\varepsilon} &= (\varepsilon_1, \ldots, \varepsilon_N)^\top \in \mathbb{R}^N \text{ is the noise vector} \\ \mathbf{X} &= \begin{pmatrix} \mathbf{x}_1^\top \\ \cdots \\ \mathbf{x}_N^\top \end{pmatrix} \qquad \text{is } N \times p \text{ matrix of inputs} \end{aligned}$$

Note: $\mathbf{x}_i \in \mathbb{R}^p$ denotes a (transposed) ith row-vector of $\mathbf{X}$ while $\boldsymbol{x}_i \in \mathbb{R}^N$ denotes the ith column $\boldsymbol{x}_i$.

Centering the data

- Sample means of inputs/outputs:

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \quad \text{and } \bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_p)^\top = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$$

- Centered responses/predictors:

$$\mathbf{y}_c = \mathbf{H}\mathbf{y} = \begin{pmatrix} y_1 - \bar{y} \\ \vdots \\ y_N - \bar{y} \end{pmatrix}$$

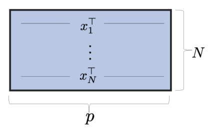
$$\mathbf{X}_c = \mathbf{H}\mathbf{X} = \begin{pmatrix} x_{11} - \bar{x}_1 & x_{12} - \bar{x}_2 & \ldots & x_{1p} - \bar{x}_p \\ x_{21} - \bar{x}_1 & x_{22} - \bar{x}_2 & \cdots & x_{2p} - \bar{x}_p \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} - \bar{x}_1 & x_{N2} - \bar{x}_2 & \cdots & x_{Np} - \bar{x}_p. \end{pmatrix}$$

where $\mathbf{H} = \mathbf{I} - (1/N)\mathbf{1}\mathbf{1}^\top$ is the centering matrix.

Menu

Big Data Challanges

$\mathbf{X}$ is flat-and-long ($p > N$ or $N \approx p$)



- Bias-variance tradeoff: infinitely many least squares (LS) solutions ($p > N$) or solution is subject to a large variance ($N \approx p$) $\Rightarrow$ introducing some bias to the estimate can reduce the variance.
- Model complexity vs parsimony (interpretation)

  among the large $\#$ of predictors, we would like to identify the ones that exhibit the strongest effects $\Rightarrow$ sparse $\hat{\boldsymbol{\beta}}$ is desired

Multicollinearity, i.e., high correlations between predictors

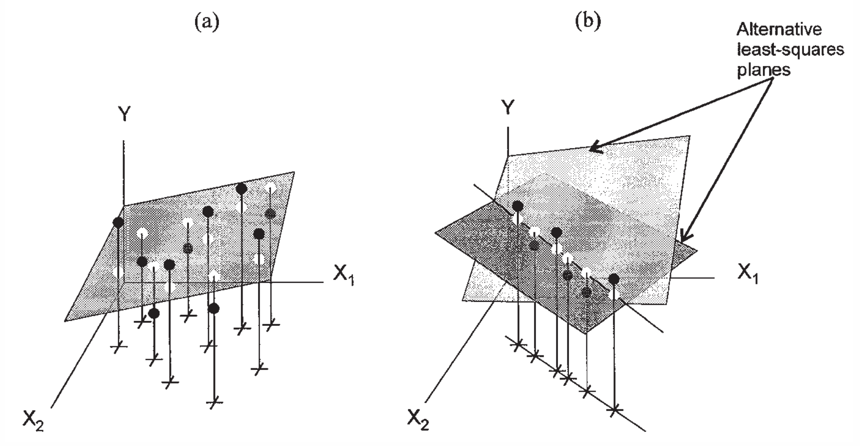
Huge variance: $\hat{\boldsymbol{\beta}}_{\text{LS}}$ can 'explode' as $\text{cond}(\mathbf{X}^\top\mathbf{X})^{-1}$ can grow very large.

Lack of robustness in heavy-tailed noise and/or in face of outliers

LSE is highly inefficient ($\neq$ MLE) and/or can become completely corrupted

Multicollinearity

- if $\mathbf{X}$ is not full rank ($\text{rank}(\mathbf{X}) < p$) the LSE is not unique and there are infinitely many solutions:



Menu

Penalized/Regularized regression

How to solve the problems above?

✓ Use regularization/penalization
regularize β_j's, i.e., we control how large they can grow.

Penalized regression problem

$$\min_{\beta_0, \boldsymbol{\beta}} \left\{ L(\beta_0, \boldsymbol{\beta}) + \lambda P(\boldsymbol{\beta}) \right\}$$

- Criterion function: $L(\beta_0, \boldsymbol{\beta}) : \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}_0^+$ depends on the data $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.
- Penalty (distance) function: $P : \mathbb{R}^p \to \mathbb{R}_0^+$ penalizes large values of $\boldsymbol{\beta}$, and can (when suitably chosen) enforce sparse solutions.
- Penalty parameter $\lambda > 0$ that controls trade-off between the two terms (data fidelity vs sparsity).

Regularized regression problem

$$\min_{\beta_0,\boldsymbol{\beta}} L(\beta_0,\boldsymbol{\beta}) \text{ subject to } P(\boldsymbol{\beta}) \leq t$$

where $L(\beta_0,\boldsymbol{\beta})$ and $P(\boldsymbol{\beta})$ are as earlier, and

- Constraint/regularization parameter $t > 0$, bounds the magnitude of the regression coefficients.

- For convex $L(\beta_0,\boldsymbol{\beta})$ and $P(\boldsymbol{\beta})$, the regularized and the penalized formulations are equivalent (1-to-1)

  This follows from Lagrangian duality. This equivalence holds since the criterion $L(\beta_0,\boldsymbol{\beta})$ is convex in $(\beta_0,\boldsymbol{\beta})$ with convex constraints $P(\boldsymbol{\beta}) \leq t$.

- In this lecture, we consider the ℓ_q-norm penalty, and its special cases: the *lasso* and *ridge regression* penalties.

ℓ_q-norm

$$\|\boldsymbol{\beta}\|_q = \sqrt[q]{\sum_{j=1}^{p} |\beta_j|^q}$$

- Consider using $P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_q^q$
- Convex for $q \geq 1$

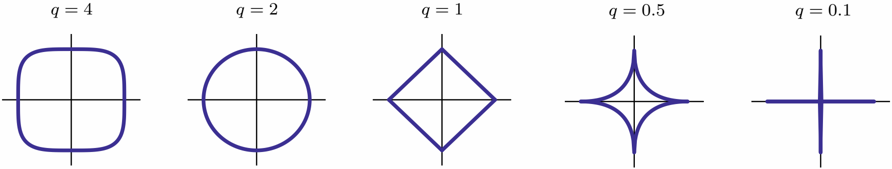


Figure: Constraint regions of unit $(\ell_q)^q$ balls: $\sum_{j=1}^{p} |\beta_j|^q \leq 1$

ℓ_q-norm

$$\|\boldsymbol{\beta}\|_q = \sqrt[q]{\sum_{j=1}^{p} |\beta_j|^q}$$

- Consider using $P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_q^q$
- Convex for $q \geq 1$ and non-convex for $0 \leq q < 1$

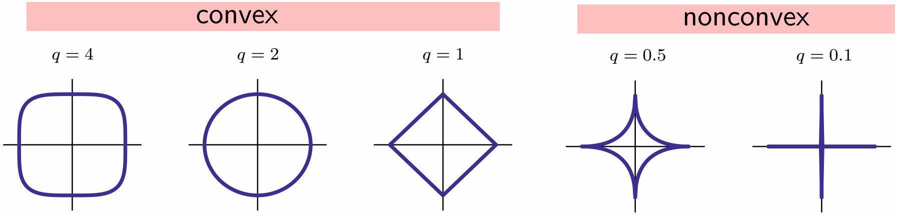


Figure: Constraint regions of unit $(\ell_q)^q$ balls: $\sum_{j=1}^{p} |\beta_j|^q \leq 1$

ℓ_q-norm

$$\|\boldsymbol{\beta}\|_q = \sqrt[q]{\sum_{j=1}^{p} |\beta_j|^q}$$

- Consider using $P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_q^q$
- Convex for $q \geq 1$ and non-convex for $0 \leq q < 1$

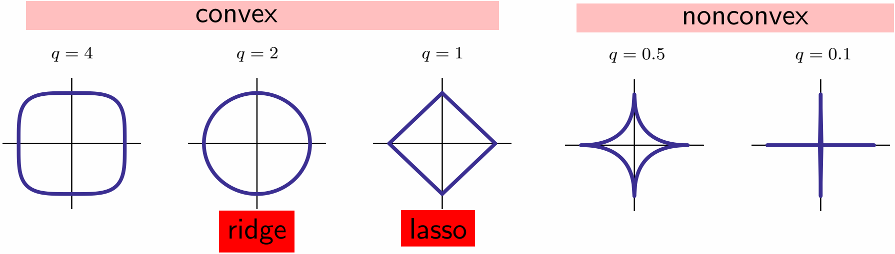


Figure: Constraint regions of unit $(\ell_q)^q$ balls: $\sum_{j=1}^{p} |\beta_j|^q \leq 1$

Menu

Ridge regression

- An older idea of regularizion in the regression model is ridge regression (RR) [Hoerl and Kennard, 1970]
- Uses RSS as the data fit term, but squared ℓ_2-norm as penalty.

ridge regression penalized/regularized forms:

$$\underset{\beta_o \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2} \| \mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta} \|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2,$$

$$\underset{\beta_0 \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2} \| \mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta} \|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_2^2 \le t$$

- **Theorem 5.1** Unique minimizer (for $\lambda > 0$) given in closed-form:

$$\hat{\beta}_{\mathrm{RR},0}(\lambda) = \bar{y} - \bar{\mathbf{x}}^\top \hat{\boldsymbol{\beta}}_{\mathrm{RR}}(\lambda),$$
$$\hat{\boldsymbol{\beta}}_{\mathrm{RR}}(\lambda) = (\mathbf{X}_c^\top \mathbf{X}_c + \lambda \mathbf{I})^{-1} \mathbf{X}_c^\top \mathbf{y}_c,$$

where $\mathbf{y}_c$ and $\mathbf{X}_c$ are centered response and feature matrix.

Discussion

$$\hat{\boldsymbol{\beta}}_{\mathrm{RR}}(\lambda) = (\mathbf{X}_c^\top \mathbf{X}_c + \lambda \mathbf{I})^{-1} \mathbf{X}_c^\top \mathbf{y}_c$$

Discuss the following special cases:

1. $\lambda \to 0$
2. $\lambda \to \infty$.
3. $\mathbf{X}_c$ is orthonormal (i.e., $\mathbf{X}_c^\top \mathbf{X}_c = \mathbf{I}_p$).

Note: often the benefits of Ridge regression are most striking when predictors are correlated.

Computation

- Make augmented data set

$$\mathbf{X}_\lambda = \begin{pmatrix} \mathbf{X}_c \\ \sqrt{\lambda}\mathbf{I}_p \end{pmatrix} \quad \text{and} \quad \mathbf{y}_\lambda = \begin{pmatrix} \mathbf{y}_c \\ \mathbf{0}_{p\times 1} \end{pmatrix}$$

  that is, append p zeros to $\mathbf{y}_c$ and a scaled $p \times p$ identity matrix to $\mathbf{X}_c$.

- Then observe that

$$\hat{\boldsymbol{\beta}}_{\mathsf{RR}}(\lambda) = (\mathbf{X}_\lambda^\top \mathbf{X}_\lambda)^{-1}\mathbf{X}_\lambda^\top \mathbf{y}_\lambda$$

$\Longrightarrow$ $\hat{\boldsymbol{\beta}}_{\mathsf{RR}}(\lambda)$ is nothing but a LS fit of $\mathbf{X}_\lambda$ to $\mathbf{y}_\lambda$.

- Columns of $\mathbf{X}$ are usually standardized, i.e., the predictors are also scaled so that they have a standard deviation equal to 1.
- The RR solution is then computed and retransformed back to the original scale. Same holds for lasso explained later.

Theorem 5.2

Bias-variance tradeoff of RR estimator

- The bias of RR estimator is

$$\begin{aligned}\text{bias}\big[\hat{\boldsymbol{\beta}}_{\mathrm{RR}}(\lambda)\big] &\triangleq \mathbb{E}\big[\hat{\boldsymbol{\beta}}_{\mathsf{RR}}(\lambda)\big] - \boldsymbol{\beta} \\ &= -\lambda \mathbf{R}_\lambda \boldsymbol{\beta},\end{aligned}$$

  where $\mathbf{R}_\lambda = (\mathbf{X}_c^\top \mathbf{X}_c + \lambda \mathbf{I})^{-1}$.
- Mean squared error (MSE) $=$ variance $+$ (bias)^2
- The MSE of RR estimator is (denoting $\sigma^2 = \text{var}(\varepsilon_i)$):

$$\begin{aligned}\mathrm{MSE}(\hat{\boldsymbol{\beta}}) &\triangleq \mathbb{E}[(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^\top] \\ &= \mathbf{R}_\lambda \{\sigma^2 \mathbf{X}_c^\top \mathbf{X}_c + \lambda^2 \boldsymbol{\beta}\boldsymbol{\beta}^\top\} \mathbf{R}_\lambda.\end{aligned}$$

- Moreover, there always exists λ such that total MSE of RR, defined as $\mathrm{Tr}\{\mathrm{MSE}(\hat{\boldsymbol{\beta}}_{\mathsf{RR}}(\lambda))\}$, is smaller than the total MSE of the LSE $\hat{\boldsymbol{\beta}}_{\mathsf{LS}}$.

Example 5.1

- Simple linear model with a single predictor ($p = 1$):

$$y_i = x_i\beta + \varepsilon_i, \quad i = 1, \ldots, N$$

where $\boldsymbol{x} = (x_1, \ldots, x_N)^\top$ standardized: $\boldsymbol{x}^\top \boldsymbol{x} = 1$.

- $\mathbb{E}[\varepsilon_i] = 0$ and $\text{var}(\varepsilon_i) = \sigma^2$.

$\Rightarrow$ The LSE and the RR estimator are

$$\hat{\beta}_{\mathsf{LS}} = \boldsymbol{x}^\top \mathbf{y} \quad \text{and} \quad \hat{\beta}_{\mathsf{RR}}(\lambda) = \frac{\hat{\beta}_{\mathsf{LS}}}{1+\lambda}.$$

- Based on Theorem 5.2, the MSE is

$$\mathrm{MSE}(\hat{\beta}_{\mathsf{RR}}(\lambda)) = \frac{\sigma^2 + \lambda^2\beta^2}{(1+\lambda)^2}.$$

- The optimal penalty parameter $\lambda^\star$ is

$$\lambda^\star = \arg\min_{\lambda} \mathrm{MSE}(\hat{\beta}_{\mathsf{RR}}(\lambda)) = \frac{\sigma^2}{\beta^2}.$$

Example 5.1 (cont'd)

- The minimum MSE:

$$\mathrm{MSE}(\hat{\beta}_{\mathsf{RR}}(\lambda^\star)) = \frac{\sigma^2 + (\lambda^\star)^2\beta^2}{(1+\lambda^\star)^2} = \sigma^2 \frac{1}{1+\frac{\sigma^2}{\beta^2}}$$

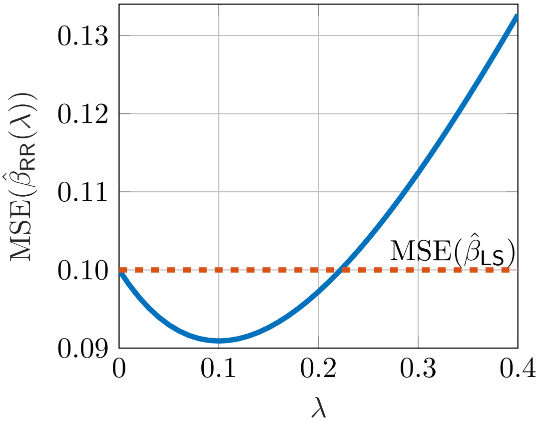
$$< \mathrm{MSE}(\hat{\beta}_{\mathsf{LS}}) = \sigma^2 \quad \forall \beta \in \mathbb{R}$$

Consider the case:

- $\sigma^2 = 0.1$
- $\beta = 1$

Optimal penalty parameter:

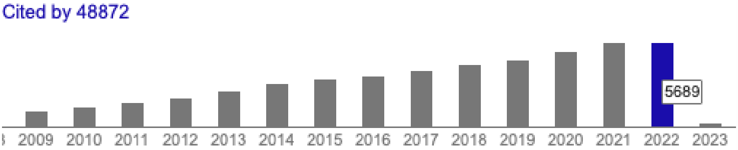
$$\lambda^\star = \frac{\sigma^2}{\beta^2} = 0.1$$



Menu

Lasso

- Lasso = "Least Absolute Shrinkage and Selection Operator" [Tibshirani, 1996]



Regression shrinkage and selection via the lasso
R Tibshirani - Journal of the Royal Statistical Society: Series B ..., 1996
Cited by 48863 Related articles All 49 versions

- Penalized regression method that uses residual sum of squares (RSS)

$$\mathrm{RSS}(\beta_0, \boldsymbol{\beta}) = \frac{1}{2N}\|\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\boldsymbol{\beta}\|_2^2$$

as the data fit and the ℓ_1-norm as penalty:

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

Lasso penalized/regularized problems

Lasso penalized/regularized optimization programs

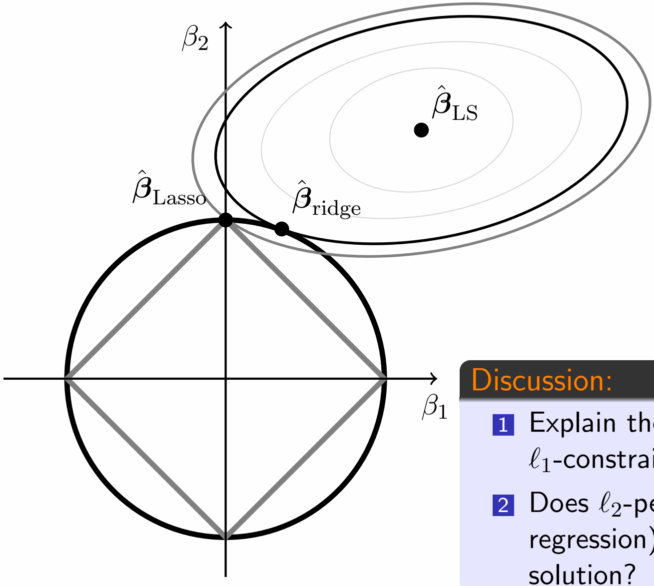
$$(\hat{\beta}_0(\lambda), \hat{\boldsymbol{\beta}}(\lambda)) = \underset{\beta_0 \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^p}{\arg\min} \quad \frac{1}{2N} \| \,\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta} \|_2^2 + \lambda \|\boldsymbol{\beta}\|_1$$

$$\underset{\beta_0 \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2N} \| \,\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta} \|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_1 \leq t,$$

$\lambda > 0$ is a shrinkage (penalty) parameter (1-to-1 with t):

- controls the (bias-variance) tradeoff between the penalty and minimization of the sum of squared residuals (fit).
- the bigger the λ the greater is the amount of shrinkage. Some of the coefficients can be shrunk all the way to zero.

Why does lasso promote sparse solutions?



Discussion:

1. Explain the figure. Why the ℓ_1-constraint promotes sparsity?
2. Does ℓ_2-penalty $\|\boldsymbol{\beta}\|_2^2$ (ridge regression) also give a sparse solution?

Lasso path

- Lasso solves:

$$(\hat{\beta}_0(\lambda), \hat{\boldsymbol{\beta}}(\lambda)) = \arg\min_{\beta_0, \boldsymbol{\beta}} \ \frac{1}{2N} \|\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1$$

  When $\lambda = 0$ we obtain the LSE, denoted $\hat{\boldsymbol{\beta}}(\lambda = 0) = \hat{\boldsymbol{\beta}}_{\text{LS}}$
- Solution is indexed by penalty $\lambda \geq 0$ or equivalently (1-to-1) by threshold $t \geq 0$: $\|\boldsymbol{\beta}\|_1 \leq t$.
- For each λ (or t), we have a solution and a set of λ's trace out a path of solutions.
- In many applications, one may wish to depict the whole solution path, i.e., the graph of $\hat{\beta}_j(\lambda)$ as a fnc of λ, $j = 1, \ldots, p$.

Lasso path (cont'd)

- In practice, one computs solutions on a grid of penalty values:

$$\begin{cases} [\lambda] = \{\lambda_0, \ldots, \lambda_L\}, & \lambda_0 > \lambda_1 > \ldots > \lambda_L, \\ & \lambda_0 = \max_j \frac{|\langle \boldsymbol{x}_j, \mathbf{y}\rangle|}{N} \end{cases}$$

  where $\lambda_0 =$ smallest λ such that $\hat{\boldsymbol{\beta}}(\lambda_0) = \mathbf{0}$.
- The sequence $\{\lambda_i\}$ is often chosen to be equispaced on log-scale:

$$\lambda_L = \epsilon\lambda_0 \quad \text{and } \lambda_j = \epsilon^{j/L}\lambda_0 = \epsilon^{1/L}\lambda_{j-1},$$

- Pathwise coordinate descent uses CCD algorithm to compute the whole lasso solution path efficiently.

Example: Prostate cancer data

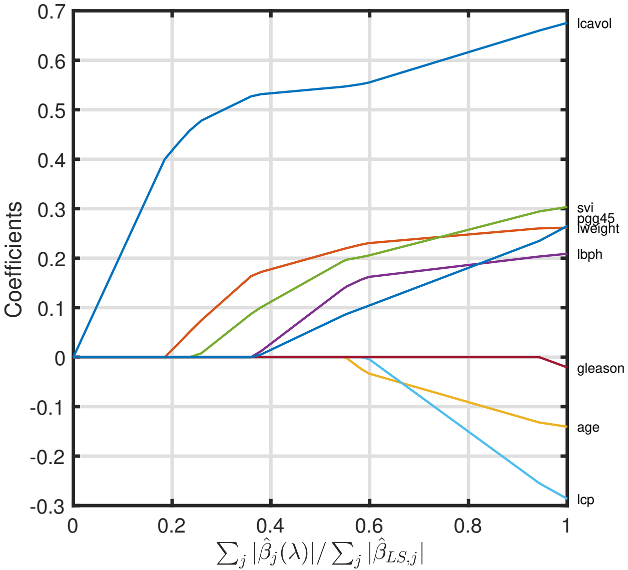
- Classic data set ($N = 97, p = 8$), also used in HW3.
- Interest is in measuring relationship between

$$y = \text{the level of prostate-specific antigen } (\texttt{lpsa})$$

in a number of clinical measures in men who were about to receive a radical prostatectomy:

x_1 = log cancer volume (`lcavol`)
x_2 = log prostate weight (`lweight`)
x_3 = age
x_4 = log of the amount of benign prostatic hyperplasia (`lbph`),
x_5 = seminal vesicle invasion (`svi`, binary)
x_6 = log of capsular penetration (`lcp`),
x_7 = Gleason score (`gleason`, ordered categorical)
x_8 = percent of Gleason scores 4 or 5 (`pgg45`)

Lasso coefficient paths



Menu

Computation of the lasso solution

- Consider the general problem of the form:

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\text{minimize}} \;\; L(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} P(\beta_j)$$

  Difficulties:

  ✗ *Non-smoothness*: objective function is not differentiable at $\beta_j = 0$ e.g., when using lasso penalty $P(\beta_j) = |\beta_j|$.

  ✗ *non-convexity*: e.g., if P is non-convex.

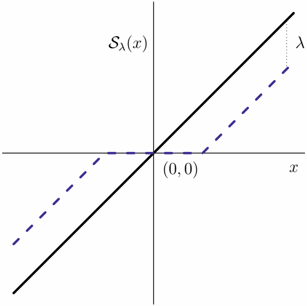
  ✗ *High-dimensionality*: p can large or huge... $(p > 10^6)$

- Cyclic coordinate descent algorithm offers a scalable method (when implemented carefully) to compute the lasso solution path.

Soft-thresholding operator

basic building block for computing penalized regression estimates

$$\mathcal{S}_\lambda(x) = \mathsf{sign}(x)(|x| - \lambda)_+$$
$$= \begin{cases} x - \lambda & \text{if } x > \lambda \\ 0 & \text{if } |x| \leq \lambda \\ x + \lambda & \text{if } x < -\lambda, \end{cases}$$



Notations:

- $(t)_+$ denotes the pos. part of $t \in \mathbb{R}$: $= t$ if $t > 0$ and 0 otherwise.
- sign(x) is the sign function: sign$(x) = +1, -1, 0$ if $x > 0$, < 0, $= 0$.

Theorem 2.3

(a) Given $y \in \mathbb{R}$, one has that

$$\begin{aligned}\hat{\beta}(\lambda) &= \arg\min_{\beta \in \mathbb{R}} \frac{1}{2}(y-\beta)^2 + \lambda|\beta| \\ &= \mathcal{S}_\lambda(y)\end{aligned}$$

(b) In the single predictor ($p = 1$) case, lasso has closed-form solution:

$$\begin{aligned}\hat{\beta}(\lambda) &= \arg\min_{\beta \in \mathbb{R}} \frac{1}{2N} \sum_{i=1}^{N} \big(y_i - \beta x_i\big)^2 + \lambda|\beta| \\ &= \mathcal{S}_\lambda\Big(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{N}\Big)\end{aligned}$$

where the predictor $\boldsymbol{x} = (x_1, \ldots, x_N)^\top$ is standardized such that $\|\boldsymbol{x}\|^2 = \boldsymbol{x}^\top \boldsymbol{x} = N$.

Proximal operator

- Proximal operator (proximal map) of convex function h is defined as

$$\mathsf{prox}_h(\mathbf{z}) = \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{z}-\boldsymbol{\beta}\|_2^2 + h(\boldsymbol{\beta}).$$

- By Theorem 2.3a and separability of lasso penalty, the proximal operator of $\lambda\|\boldsymbol{\beta}\|_1$ is

$$\begin{aligned}\mathsf{prox}_{\lambda\|\cdot\|_1}(\mathbf{z}) &= \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{z}-\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \\ &= \mathcal{S}_\lambda(\mathbf{z})\end{aligned}$$

Subgradient optimality conditions

- CCD for lasso solves:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \left\{ D(\boldsymbol{\beta}) = \frac{1}{2N} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \right\}$$

(where columns of $\mathbf{X}$ are standardized)

- For convex (subdifferentiable) function D (*cf.* Serigy's notes):

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} D(\boldsymbol{\beta}) \quad \Leftrightarrow \quad \mathbf{0} \in \partial D(\hat{\boldsymbol{\beta}}).$$

(where $\partial D(\boldsymbol{\beta})$ denotes the set of all subgradients of D at $\hat{\boldsymbol{\beta}}$)

- Subdifferential of $|\beta_j|$ is

$$\partial |\beta_j| = \begin{cases} \text{sign}(\beta_j), & \text{for } \beta_j \neq 0 \\ s & \text{for } \beta_j = 0 \end{cases}$$

where s is some number verifying $|s| \leq 1$.

Subgradient (estimating) equations for lasso

- A necessary and sufficient condition for $\hat{\boldsymbol{\beta}}$ to be the lasso solution is that it solves the zero subgradient equations:

$$\partial\Big(\frac{1}{2N}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1\Big) \in \mathbf{0}.$$

- Thus $\hat{\boldsymbol{\beta}}$ is a lasso solution iff

$$\frac{1}{N}\boldsymbol{x}_j^\top(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = \begin{cases} \lambda \text{sign}(\hat{\beta}_j), & \text{if } \hat{\beta}_j \neq 0 \\ \lambda s_j, & \text{if } \hat{\beta}_j = 0 \end{cases},$$

  where s_j is some number verifying $|s_j| \leq 1$, for $j = 1, \ldots, p$.
- You will check for this condition when you implement lasso in HW3.

Cyclic coordinate descent

- Consider a penalized objective function:

$$D(\beta_0, \beta_1, \ldots, \beta_p) = L(\beta_0, \beta_1, \ldots, \beta_p) + \lambda \sum_{j=1}^{p} P(\beta_j)$$

  - $L(\beta_0, \boldsymbol{\beta})$ is convex and differentiable
  - $P(\cdot)$ is convex (but not necessarily differentiable).
- Cyclic Coordinate descent (CCD): updates β_j by minimizing D in this coordinate while keeping others fixed:

$$\beta_j \leftarrow \arg\min_{\beta_j \in \mathbb{R}} D(\hat{\beta}_0, \ldots, \hat{\beta}_{j-1}, \beta_j, \hat{\beta}_{j+1}, \ldots, \hat{\beta}_p),$$

  and repeatedly cycles through the coefficients one at a time $(j = 0, 1, \ldots, p)$ until convergence.
- Tseng [2001] showed that any limit point of CCD is a minimizer of D.

- The benefits of CCD are:
  - ✓ CCD is a simple algorithm and very easy to implement.
  - ✓ Useful and general method for cases, when the single parameter (i.e., one coordinate at a time) problem is easy to solve.
  - ✓ Can be used to compute the whole lasso path.
- In the case of lasso, the single parameter problem is simple:

$$\text{"new estimate"} \leftarrow \mathcal{S}_\lambda(\text{"current estimate"} + \text{"correction"})$$

See details on next slides.

- The lasso objective function is separable in coordinates:

$$D(\beta_0, \boldsymbol{\beta}) = \frac{1}{2N} \sum_{i=1}^{N} \Big(y_i - \beta_0 - \sum_{k \neq j} x_{ik}\beta_k - x_{ij}\beta_j \Big)^2 \\ + \lambda|\beta_j| + \lambda \sum_{k \neq j} |\beta_k|.$$

- Update of β_0 (when holding β_js fixed at current estimates $\hat{\beta}_j$):

$$\begin{aligned} \hat{\beta}_0 &\leftarrow \frac{1}{N} \sum_{i=1}^{N} (y_i - \sum_j x_{ij}\hat{\beta}_j) \\ &= \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{\beta}_0 - \sum_j x_{ij}\hat{\beta}_j + \hat{\beta}_0) \\ &= \hat{\beta}_0 + \frac{1}{N} \sum_{i=1}^{N} \hat{r}_i, \end{aligned}$$

where $\hat{r}_i = y_i - \hat{\beta}_0 - \sum_{j=1}^{p} x_{ij}\hat{\beta}_j$ are full residuals before the update.

- Update for β_j (holding β_k-s fixed at current estimates $\hat{\beta}_k$, $k \neq j$):

$$
\begin{aligned}
\hat{\beta}_j &\leftarrow \arg\min_{\beta_j} \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{\beta}_0 - \sum_{k \neq j} x_{ik}\hat{\beta}_k - x_{ij}\beta_j)^2 + \lambda|\beta_j| \\
&= \arg\min_{\beta_j} \frac{1}{2N} \sum_{i=1}^{N} \Big(y_i - \hat{\beta}_0 - \sum_{k} x_{ik}\hat{\beta}_k + x_{ij}\hat{\beta}_j - x_{ij}\beta_j\Big)^2 + \lambda|\beta_j| \\
&= \arg\min_{\beta_j} \frac{1}{2N} \sum_{i=1}^{N} \Big(\hat{r}_i + x_{ij}\hat{\beta}_j - x_{ij}\beta_j\Big)^2 + \lambda|\beta_j| \\
&= \mathcal{S}_\lambda\Big(\hat{\beta}_j + \frac{1}{N}\langle \boldsymbol{x}_j, \hat{\mathbf{r}}\rangle\Big),
\end{aligned}
$$

where the last identity follows from Theorem 2.3b, $j = 1, \ldots, p$.

- Above $\hat{\mathbf{r}} = (\hat{r}_1, \ldots, \hat{r}_N)^\top$ is a vector of current residuals (before the update).

Some notes on CCD

- In practise cyclic updates of β_0 in CCD algorithm can be omitted if one simply runs the CCD algorithm (assuming a model with no intercept term, $\beta_0 = 0$) but for centered data $\mathbf{X}_c, \mathbf{y}_c$.
- Why? It can be shown, that solution $\hat{\boldsymbol{\beta}}(\lambda)$ for centered data $\mathbf{X}_c, \mathbf{y}_c$ is the same as for uncentered data $\mathbf{X}, \mathbf{y}$ (in a model with intercept).
- Thus the intercept is calculated in the last stage as

$$\hat{\beta}_0(\lambda) = \bar{y} - \bar{\mathbf{x}}^\top \hat{\boldsymbol{\beta}}(\lambda).$$

- Each coordinate update requires computing $\langle \mathbf{x}_j, \hat{\mathbf{r}} \rangle$ and then updating $\hat{\mathbf{r}} \leftarrow \hat{\mathbf{r}} + (\hat{\beta}_j^{\text{old}} - \hat{\beta}_j)\boldsymbol{x}_j$ which is of $O(N)$ flops.

Algorithm 5.1: `Lasso` algorithm that computes the lasso solution using CCD algorithm in a model with intercept

Input : $\mathbf{y} \in \mathbb{R}^N$, $\mathbf{X} = (\boldsymbol{x}_1 \ \cdots \ \boldsymbol{x}_p) \in \mathbb{R}^{N \times p}$, $\lambda > 0$, $\hat{\boldsymbol{\beta}}_{\text{init}} \in \mathbb{R}^p$.

1 Center the inputs and outputs:

$$\boldsymbol{x}_j \leftarrow \boldsymbol{x}_j - \bar{x}_j \mathbf{1} \quad \text{and} \quad \mathbf{y} \leftarrow \mathbf{y} - \bar{y}\mathbf{1}$$

$(j = 1, \ldots, p)$, where $\bar{x}_j = \frac{1}{N}\sum_{i=1}^N x_{ij}$ and $\bar{y} = \frac{1}{N}\sum_{i=1}^N y_i$.

2 Standardize the feature vectors:

$$\boldsymbol{x}_j \leftarrow \boldsymbol{x}_j / s_j \quad \text{for } j = 1, \ldots, p$$

where $s_j = \|\boldsymbol{x}_j\|_2 / \sqrt{N}$.

3 $\hat{\boldsymbol{\beta}}(\lambda) \leftarrow \text{ccdlasso}(\mathbf{y}, \mathbf{X}, \lambda, \hat{\boldsymbol{\beta}}_{\text{init}})$

4 Transform the regression coefficient back to the original scale:

$$\hat{\beta}_j(\lambda) \leftarrow \hat{\beta}_j(\lambda)/s_j \quad \text{for } j = 1, \ldots, p$$

5 $\hat{\beta}_0(\lambda) = \bar{y} - \bar{\mathbf{x}}^\top \hat{\boldsymbol{\beta}}(\lambda)$ `// Compute the intercept`

Output : $\hat{\beta}_0(\lambda), \hat{\boldsymbol{\beta}}(\lambda)$, minimizer of $\frac{1}{2N}\| \mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1$

Algorithm 5.2: `ccdlasso` computes lasso solution for standardized predictors in a model with no intercept.

Input : $\mathbf{y} \in \mathbb{R}^N$, $\mathbf{X} = (\boldsymbol{x}_1 \ \cdots \ \boldsymbol{x}_p) \in \mathbb{R}^{N \times p}$, warm start $\hat{\boldsymbol{\beta}}_{\text{init}} \in \mathbb{R}^p$, penalty $\lambda > 0$. Predictors are standardized s.t. $\boldsymbol{x}_j^\top \boldsymbol{x}_j = N$

Initialize: Max. # of iterations, $I_{max} = 10^4$; Convergence threshold, $\delta = 10^{-4}$

Set $\hat{\mathbf{r}} \leftarrow \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_{\text{init}}$, $\hat{\boldsymbol{\beta}}^{\text{old}} \leftarrow \hat{\boldsymbol{\beta}}_{\text{init}}$

for $i = 1, \ldots, I_{max}$ **do**

  for $j = 1$ **to** p **do**

    $\hat{\beta}_j \leftarrow \mathcal{S}_\lambda\big(\hat{\beta}_j + \frac{1}{N}\langle \boldsymbol{x}_j, \hat{\boldsymbol{r}} \rangle\big)$

    $\hat{\mathbf{r}} \leftarrow \hat{\mathbf{r}} + (\hat{\beta}_j^{\text{old}} - \hat{\beta}_j)\boldsymbol{x}_j$

  if $\|\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}^{\text{old}}\|_2 / \|\hat{\boldsymbol{\beta}}\|_2 < \delta$ **then**

    break

  $\hat{\boldsymbol{\beta}}^{\text{old}} \leftarrow \hat{\boldsymbol{\beta}}$

Output : $\hat{\boldsymbol{\beta}}(\lambda) = \hat{\boldsymbol{\beta}}$, the minimizer of $\frac{1}{2N}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1$

Pathwise coordinate descent

- Lasso solution path is computed over a grid $[\lambda]$ of penalty parameter values (recall slide 20).
- The algorithm starts from λ_0 that yields all zeros solution and then goes to next (smaller) value on the grid and uses the previous estimate as a warm start.
- This algorithm is called *pathwise coordinate descent* [Friedman et al., 2007]

Why CCD works for large-scale data?

CCD is scalable to large p, given that it is implemented with smart tricks:

✓ For large λ, most coordinates that are zero never become non-zero.

$\Rightarrow$ **active set** strategy updates active predictors (i.e., nonzero coefficients) until convergence and then check other variables. See Tibshirani and et al. [2012] for details.

✓ **warm starts**: move from large λ to smaller, using solutions at previous λ as initial value for next λ.

✓ CCD is easy to extent to generalized linear models (GLM)

✗ Coding in lower-level language (C/C++/Fortran) is necessary due to iterative nature of CCD.

✓ GLMnet (calls Fortran and uses tricks above) is fast.
https://hastie.su.domains/glmnet_python/
https://hastie.su.domains/glmnet_matlab/

Menu

Benefits of lasso

- ✓ Penalty (smart choice of λ) offers an automated variable selection (lasso performs estimation and variable selection simultaneously).
- ✓ Lasso does variable selection and shrinkage; ridge only shrinks.
- ✓ Depicting the whole lasso solution path (as λ grow) informs us when variables drop-out from the model.
- ✓ Works for underdetermined systems ($p > N$) which occur commonly in many applications.
- ✓ Growing importance of sparse representations and modelling.

Shortcoming of lasso

- ☺ Lasso solution is unique when the columns of $\mathbf{X}$ are in general position* and $\lambda > 0$. This holds true even when $N \leq p$
- ☹ When $N \leq p$, the number of nonzero coefficients in any lasso solution is at most N.
- ☹ If $\mathbf{X}$ is not full column rank, solution is not unique, and there can be infinitely many solutions.
- ☹ Lasso ignores possible structured sparsity (e.g., block sparsity, smoothness, etc).

*Columns $\{\mathbf{x}_j\}_{j=1}^{p}$ are in general position if any affine subspace $\mathbb{L} \subset \mathbb{R}^N$ of dimension $k < N$ contains at most $k+1$ elements of the set $\{\pm\mathbf{x}1, \pm\mathbf{x}_2, \ldots, \pm\mathbf{x}_p\}$.

Menu

Elastic net (EN)

- EN penalty is a convex combination of ridge and lasso penalties:

$$P_{\mathsf{EN}}(\boldsymbol{\beta};\alpha) = \alpha\|\boldsymbol{\beta}\|_1 + \frac{1}{2}(1-\alpha)\|\boldsymbol{\beta}\|_2^2 = \sum_{i=1}^{p}\left[\alpha|\beta_i| + \frac{1}{2}(1-\alpha)\beta_i^2\right],$$

  where $\alpha \in [0,1]$ is a tuning parameter that can be varied:
  - $\alpha = 1$ corresponds to lasso regression
  - $\alpha = 0$ corresponds to ridge regression
- α can be set on subjective grounds or cross-validation scheme on a grid of α values.
- The EN optimization problem proposed by [Zou and Hastie, 2005] is

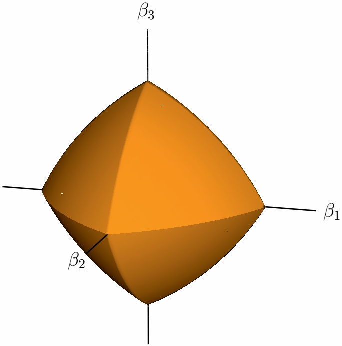
$$\underset{(\boldsymbol{\beta}_0,\boldsymbol{\beta})\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}}\left\{\frac{1}{2N}\sum_{i=1}^{N}(y_i-\beta_0-\mathbf{x}_i^\top\boldsymbol{\beta})^2 + \lambda\left[\alpha\|\boldsymbol{\beta}\|_1 + \frac{1}{2}(1-\alpha)\|\boldsymbol{\beta}\|_2^2\right]\right\}$$

  where $\lambda \geq 0$ is the penalty parameter.

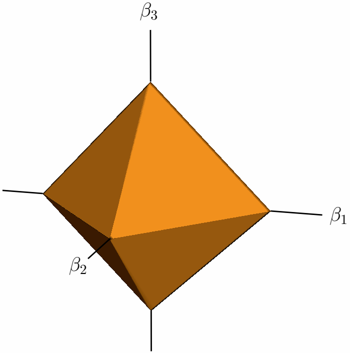
Benefits of elastic net

- removes the limitation on the number of selected variables
- encourages grouping effect
- stabilizes the coefficient paths

EN ($\alpha = 0.7$)



lasso ($\alpha = 1$)



Q: (a) what causes grouping effect? (b) Does EN still give sparse solution?

Theorem 6.1

(a) Given $y \in \mathbb{R}$, one has that

$$\hat{\beta}(\lambda, \alpha) = \arg\min_{\beta \in \mathbb{R}} \frac{1}{2}(y - \beta)^2 + \lambda\alpha|\beta| + \frac{\lambda(1-\alpha)}{2}\beta^2$$
$$= \frac{\mathcal{S}_{\lambda\alpha}(y)}{1 + \lambda(1-\alpha)}$$

where $\mathcal{S}_\lambda(x) = \text{sign}(x)(|x| - \lambda)_+$ is the soft-thresholding operator.

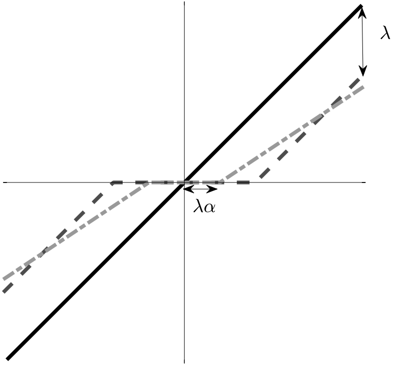
(b) In the single predictor ($p = 1$) case, EN admits closed-form solution:

$$\hat{\beta}(\lambda, \alpha) = \arg\min_{\beta \in \mathbb{R}} \frac{1}{2N} \sum_{i=1}^{N} \big(y_i - \beta x_i\big)^2 + \lambda\alpha|\beta| + \frac{\lambda(1-\alpha)}{2}\beta^2$$
$$= \frac{\mathcal{S}_{\lambda\alpha}\left(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{N}\right)}{1 + \lambda(1-\alpha)}$$

where the predictor $\boldsymbol{x} = (x_1, \ldots, x_N)^\top$ is standardized such that $\|\boldsymbol{x}\|^2 = \boldsymbol{x}^\top \boldsymbol{x} = N$).

Shrinkage in EN

$$\hat{\beta}(\lambda, \alpha) = \frac{\mathcal{S}_{\lambda\alpha}(y)}{1 + \lambda(1 - \alpha)}$$



solid line : $\lambda = 0$
dashed line : lasso $(\alpha = 1)$
dash dotted : EN $(\alpha = 0.5)$

Cyclic coordinate descent (CCD) for EN

- CCD update for j^{th} coefficient is

$$\hat{\beta}_j \leftarrow \frac{\mathcal{S}_{\alpha\lambda}\big(\hat{\beta}_j + \frac{1}{N}\langle \boldsymbol{x}_j, \hat{\mathbf{r}}\rangle\big)}{1+\lambda(1-\alpha)}$$

  where $\hat{\mathbf{r}}$ is the current residual and $\mathcal{S}_\lambda(x) = \mathrm{sign}(x)(|x|-\lambda)_+$.
- As for lasso, predictors are standardized ($\boldsymbol{x}_j^\top \boldsymbol{x}_j = N$).
- Only thing that changes in `ccdlasso` algorithm is the update of coefficient.
- The subgradient optimality condition is now

$$\frac{1}{N}\boldsymbol{x}_j^\top(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) - \lambda(1-\alpha)\hat{\beta}_j = \begin{cases} \lambda\alpha\mathrm{sign}(\hat{\beta}_j), & \text{if } \hat{\beta}_j \neq 0 \\ \lambda\alpha s_j, & \text{if } \hat{\beta}_j = 0 \end{cases},$$

  where s_j is a number verifying $|s_j| \leq 1$.

Menu

5 Lasso
- 5.1 Big Data Challenges
- 5.2 Penalized/Regularized regression
- 5.3 Ridge regression
- 5.4 Lasso
- 5.5 Computation of the lasso solution
- 5.6 Discussion

6 Lasso:extensions
- 6.1 Elastic net
- 6.2 Generalized lasso
  - Fused lasso
  - Trend filtering
- 6.3 Group lasso
- 6.4 Discussion

Generalized lasso

- *Generalized lasso* solves the problem

$$\underset{(\beta_0,\boldsymbol{\beta})\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda \|\mathbf{D}\boldsymbol{\beta}\|_1 \right\}$$

  where $\mathbf{D} \in \mathbb{R}^{m\times p}$ is a specified *penalty matrix*.
- Lasso is obtained when $\mathbf{D} = \mathbf{I}$.
- EX: neighboring coefficients β_j can be related (e.g., piecewise constant over neighboring values) and it makes sense to encourage both *block-sparsity* and *smoothness*.
- This can be achieved with proper choice of $\mathbf{D}$.

Fused lasso

- *Fused lasso (FL)* penalty is defined as

$$\|\boldsymbol{\beta}\|_{\mathsf{FL}} = \|\bar{\mathbf{D}}_p \boldsymbol{\beta}\|_1 = \sum_{j=1}^{p-1} |\beta_j - \beta_{j+1}|,$$

where $\bar{\mathbf{D}}_p$ is 1st order difference matrix, $\bar{\mathbf{D}}_p \in \mathbb{R}^{(p-1)\times p}$:

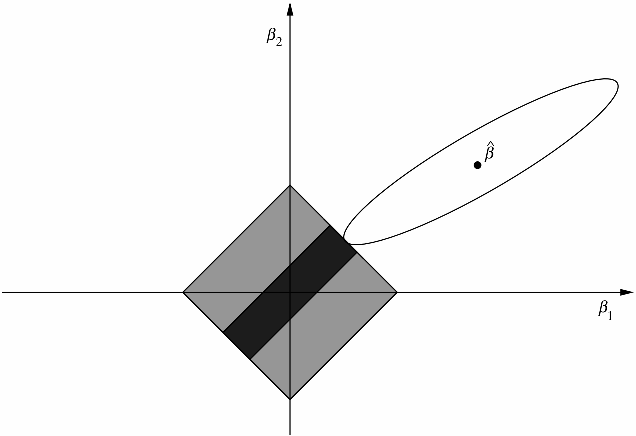
$$\bar{\mathbf{D}}_p = \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{pmatrix}.$$

- The fused lasso optimization problem is

$$\underset{(\beta_0, \boldsymbol{\beta}) \in \mathbb{R} \times \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p-1} |\beta_j - \beta_{j+1}| \right\}$$

Geometry of fused lasso

$$\underset{(\beta_0,\boldsymbol{\beta})\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}} \frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \mathbf{x}_i^\top \boldsymbol{\beta})^2 \quad \text{s.t.} \quad \sum_{j=1}^{p-1}|\beta_j - \beta_{j+1}| \le s$$



Sparse fused lasso

- Combining FL penalty with lasso yields the *sparse fused lasso (SFL)* penalty:
$$P_{\mathsf{SFL}}(\boldsymbol{\beta}; \lambda_1, \lambda_2) = \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{\beta}\|_{\mathsf{FL}}$$
where $\lambda_1, \lambda_2 \geq 0$ form a pair of fixed regularization parameters.
- SFL optimization problem is
$$\underset{(\beta_0, \boldsymbol{\beta}) \in \mathbb{R} \times \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \sum_{j=1}^{p-1} |\beta_j - \beta_{j+1}| \right\}.$$
- SFL was proposed for regression by Tibshirani et al. [2005] but it has longer history in image processing where it is called *total variation (TV)* penalty [Rudin et al., 1992].

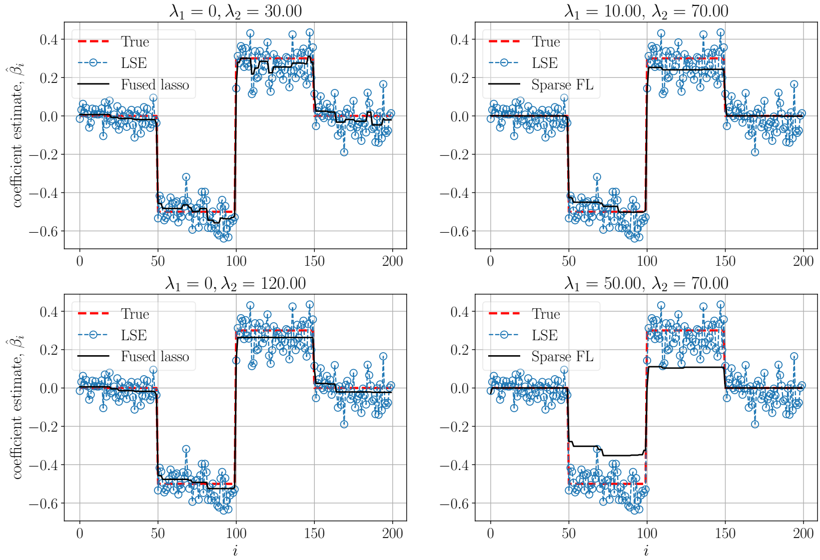
Example 6.1

- $N \times p$ predictor matrix $\mathbf{X}$ is of size $N = 1000$ and $p = 200$.
- Predictor variables have a joint multivariate Gaussian distribution with unit variance (var$(X_i) = 1$) and pairwise correlation between any two predictor variables being 0.7
- Coefficient profile is piecewise constant with 50% $\beta_i = 0$.
- The errors terms were unit variance Gaussian, $\varepsilon_i \sim \mathcal{N}(0, 1)$, and output was generated as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

- SFL solution was computed using different options for parameter pairs (λ_1, λ_2).

Example 6.1: result



Computation of FL/SFL regression

- The proximal operator of FL-penalty,

$$\text{prox}_{\lambda\|\cdot\|_{\mathsf{FL}}}(\mathbf{z}) = \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{z}-\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_{\mathsf{FL}},$$

  has no closed-form solution, but can be solved in in linear time via taut string method [Davies and Kovac, 2001] or dynamic programming (DP) [Johnson, 2013].
- Once having method for evaluating proximal operator, PGA iterations are

$$\hat{\boldsymbol{\beta}}^{(k)} = \text{prox}_{t_k\lambda\|\cdot\|_{\mathsf{FL}}}\big(\hat{\boldsymbol{\beta}}^{(k-1)} + t_k\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}^{(k-1)})\big).$$

  where t_k is the stepsize.
- The proximal operator of SFL penalty is [Friedman et al., 2007]:

$$\begin{aligned}\hat{\boldsymbol{\beta}}(\lambda_1,\lambda_2) &= \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{z}-\boldsymbol{\beta}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_{\mathsf{FL}} \\ &= \mathcal{S}_{\lambda_1}\big(\text{prox}_{\lambda_2\|\cdot\|_{\mathsf{FL}}}(\mathbf{z})\big).\end{aligned}$$

  where $\mathcal{S}_\lambda(x) = \text{sign}(x)(|x|-\lambda)_+$ is the soft-thresholding operator.

Fused lasso extensions

- FL can be generalized over a graph $\mathcal{G} = (\{1, \ldots, p\}, E)$ with p nodes and edge set E by defining the penalty matrix $\mathbf{D}$ as $|E| \times p$ matrix, whose ℓth row is defined as

$$\mathbf{d}_\ell^\top = (0, \ldots, \underset{\substack{\uparrow \\ i}}{-1}, \ldots, \underset{\substack{\uparrow \\ j}}{1}, \ldots, 0)$$

  when (i, j) is an edge in the graph, so $(i, j) \in E$.
- This yields

$$\|\mathbf{D}\boldsymbol{\beta}\|_1 = \sum_{(i,j) \in E} |\beta_i - \beta_j|.$$

- The regression solution using FL penalty over graph has $\hat{\beta}_i \approx \hat{\beta}_j$ across the edges in the graph (i.e., when $(i, j) \in E$).
- Another extension is Fused ridge (FR) penalty:

$$\|\boldsymbol{\beta}\|_{\mathsf{FR}}^2 = \sum_{i=1}^{p-1} (\beta_i - \beta_{i+1})^2$$

Trend filtering

- A special case of FL regression (with $\mathbf{X} = \mathbf{I}_{N \times N}$ and $p = N$) is *trend filtering* which is signal approximation problem that considers optimization problems of the form:

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_i)^2 + \lambda P(\boldsymbol{\beta}) \right\},$$

  where $P(\boldsymbol{\beta})$ is the penalty function and λ is the penalty parameter.
- $\{\beta_i\}_{i=1}^N$ is referred to as *signal*, since here we consider a classic signal-in-noise measurement model

$$y_i = \beta_i + \varepsilon_i, \ i = 1, \ldots, N.$$

  where only the corrupted measurements y_i-s are available but not the signal β_i itself.
- Applications are numerous in image or speech processing, or wireless comm. for example.

Trend filtering: choise of penalty

- The choice of the penalty depends on the assumed underlying signal shape.
- When the signal β_i is piecewise constant, $P(\boldsymbol{\beta})$ is commonly chosen to be FL or SFL penalty leading to solving

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_i)^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_{\mathsf{FL}} \right\}.$$

- Fused ridge penalty $\|\cdot\|_{\mathsf{FR}}^2$ works better when signal is smoother.
- Note: Trend filtering is tantamount to evaluating the proximal map of the penalty.

Example 6.2

- We consider two cases:
  (A) signal β_i is a piecewise constant signal.
  (B) signal is a superposition of two sine waves

$$\beta_i = \sin((i-1)2\pi f_1) + \sin((i-1)2\pi f_2)$$

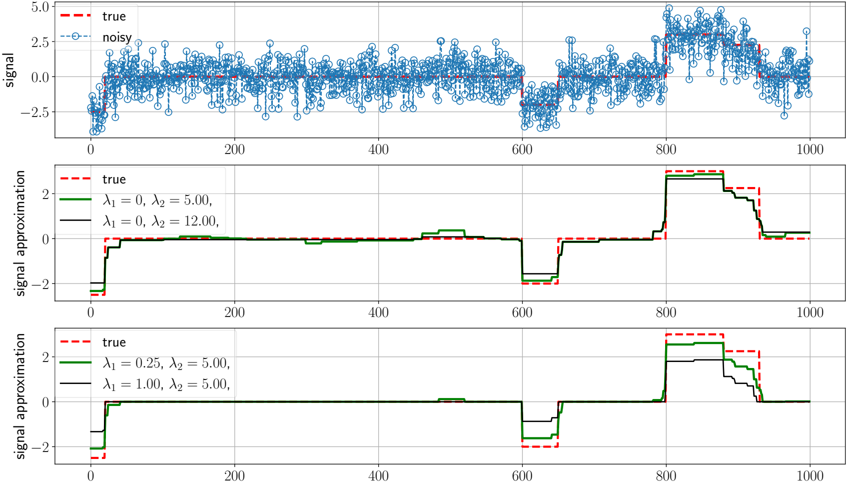
  with frequencies $f_1 = 0.15$ and $f_2 = f_1/10 = 0.015$.
- Signals are measured in additive white Gaussian noise: $\varepsilon \sim \mathcal{N}(0,1)$ for case (A) and $\varepsilon \sim \mathcal{N}(0,0.25)$ for case (B).
- Measurements y_i are then generated as

$$y_i = \beta_i + \varepsilon_i,\ i = 1, \ldots, N,$$

where the sample length is $N = 1000$.

Example 6.2: results for case (A)

- Results using SFL signal approximator with different (λ_1, λ_2).



Example 6.2: results for case (B)

- Results using FR signal approximator with different λ values.

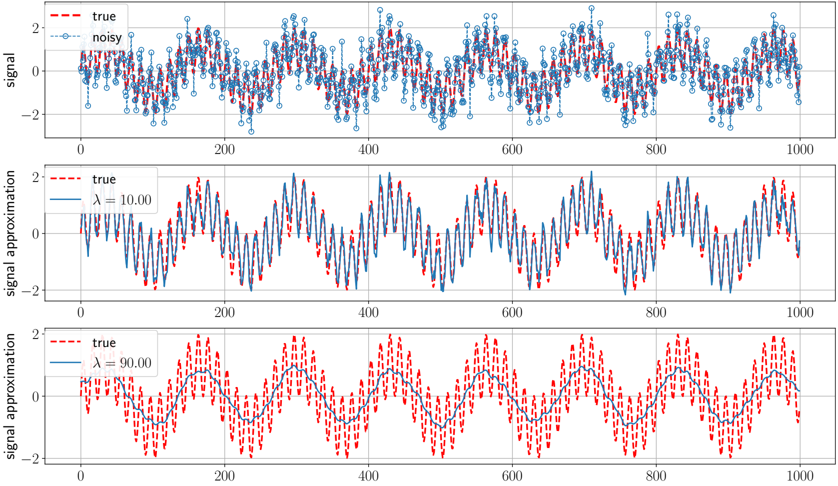


Image denoising

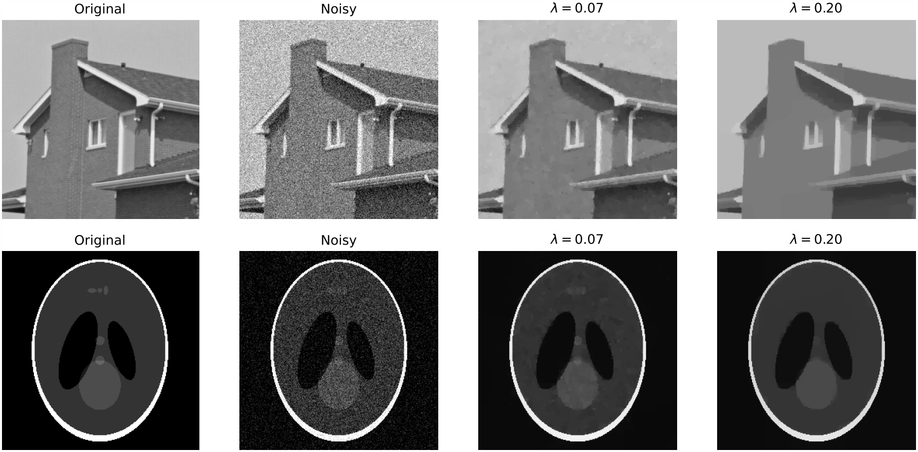
- FL was first used in image denoising where it is called *total variation (TV) denoising*, which solves

$$\underset{\mathcal{B}\in\mathbb{R}^{N_1\times N_2}}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_{i,j}-\beta_{i,j})^2 + \lambda\sum_{i=2}^{N_1}\sum_{j=1}^{N_2}|\beta_{i,j}-\beta_{i-1,j}| + \lambda\sum_{i=1}^{N_1}\sum_{j=2}^{N_2}|\beta_{i,j}-\beta_{i,j-1}| \right\}$$

  where
  - $\mathbf{Y} = (y_{i,j}) \in \mathbb{R}^{N_1\times N_2}$ is a $2D$-image
  - $\mathcal{B} = (\beta_{i,j})$ is the denoised image
  - λ is the penalty term.
- idea: enforce smoothness of neighborhood pixels both in horizontal and vertical directions of the image.

Image denoising

- Denoising the `house` and (Shepp-Logan) `phantom` image using total variation image denoiser (two choises of penalty λ).



Menu

Group lasso

- *Group lasso* is defined as the following optimization problem:

$$\underset{\boldsymbol{\beta}_g \in \mathbb{R}^{p_g}}{\text{minimize}} \ \frac{1}{2}\Big\|\mathbf{y} - \sum_{g=1}^{G} \mathbf{X}_g \boldsymbol{\beta}_g\Big\|_2^2 + \lambda \sum_{g=1}^{G} \sqrt{p_g}\|\boldsymbol{\beta}_g\|_2,$$

  where

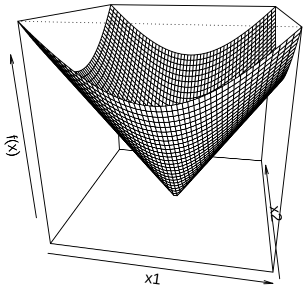
  - $\mathbf{X}_g \in \mathbb{R}^{N \times p_g}$ data matrix corresponding to covariates in group g
  - $\boldsymbol{\beta}_g$ regression coefficients corresponding to group g,
  - p_g dimensionality (number of covariates) of group g
  - G number of groups.

  and as earlier, $\mathbf{y} \in \mathbb{R}^N$ is the response, N is sample size, and $\lambda \geq 0$ the penalty parameter.

- Each group penalty is weighted according to their size, $\sqrt{p_g}$. This works well for orthogonal $\mathbf{X}_g$, but for general matrices, Frobeinus norm $\|\mathbf{X}_g\|_{\mathsf{F}}$ can be used.

Group lasso (cont'd)

- ℓ_2-norm penalty $\|\boldsymbol{\beta}_g\|_2$ is not differentiable at zero, making it have a sharp edge at 0.
- This leads it to have attributes that are similar to lasso



1. For large enough $\lambda > 0$, the entire vector $\boldsymbol{\beta}_g$ will be zero or all coefficients are nonzero.
2. if $p_q \equiv 1$ for all g, so we have a single covariate in each group, then the problem reduces to ordinary lasso.

Group lasso: usages

Some example applications where group lasso penalty is particularly useful:

1. The levels of qualitative factors are typically coded using a set of dummy variables and one would want to include or exclude this group of variables together.
2. In gene-expression arrays, genes from the same biological pathway can be highly correlated, and selecting them as a group corresponds to electing a pathway.

Computing the group lasso solution

- Subdifferential of $\|\boldsymbol{\beta}\|_2$ is

$$\partial\|\boldsymbol{\beta}\|_2 = \begin{cases} \boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2 & \text{for } \boldsymbol{\beta} \neq 0 \\ \{\mathbf{s} \in \mathbb{R}^p : \|\mathbf{s}\|_2 \leq 1\} & \text{for } \boldsymbol{\beta} = 0 \end{cases}$$

- For all but j^{th} block fixed, the zero subgradient equation is

$$-\mathbf{X}_j^\top(\boldsymbol{r}_j - \mathbf{X}_j\hat{\boldsymbol{\beta}}_j) + \lambda\sqrt{p_j}\hat{\mathbf{s}}_j = \mathbf{0}$$

where $\boldsymbol{r}_j = \mathbf{y} - \sum_{g \neq j}^{G} \mathbf{X}_g\hat{\boldsymbol{\beta}}_g$ is the jth partial residual and $\hat{\mathbf{s}}_j \in \mathbb{R}^{p_j}$ is an element of subdifferential of $\| \cdot \|_2$ evaluated at $\hat{\boldsymbol{\beta}}_j$.
- Has a simple closed-form solution when $\mathbf{X}_g$-s are orthonormal:

$$\hat{\boldsymbol{\beta}}_j = \left(1 - \frac{\lambda\sqrt{p_j}}{\|\mathbf{X}_j^\top\boldsymbol{r}_j\|_2}\right)_+ \mathbf{X}_j^\top\boldsymbol{r}_j$$

$\Rightarrow$ *block coordinate descent (BCD)* thus proves to be efficient approach for computing the group lasso

Menu

Discussion

- This chapter gave a quick look at some selected extensions/variations of lasso.
- Many important extensions were not discussed such as
  - Bayesian lasso [Park and Casella, 2008]
  - Adaptive lasso [Zou, 2006]
  - Lasso using nonconvex penalties such as smoothly clipped absolute deviation (SCAD) penalty [Fan and Li, 2001], minimax concave penalty [Zhang, 2010], etc,
  - Robust lasso
- Learn more about lasso and its variants from Hastie et al. [2015].

References

P Laurie Davies and Arne Kovac. Local extremes, runs, strings and multiresolution. *The Annals of Statistics*, 29(1):1–65, 2001.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *Ann. Appl. Stat.*, 1(2):302–332, 2007.

Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press, 2015.

A. E. Hoerl and R. W. Kennard. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Nicholas A Johnson. A dynamic programming algorithm for the fused lasso and l 0-segmentation. *Journal of Computational and Graphical Statistics*, 22(2):246–260, 2013.

Trevor Park and George Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103(482):681–686, 2008.

L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

M. Tibshirani, R. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *J. Royal Stat. Soc., Ser. B*, 67(1):91–108, 2005.

R. Tibshirani and et al. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2):245–266, 2012.

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

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.