Topics in High Dimensional Data Analysis

Quefeng Li

Ridge regression

$$\widehat{\boldsymbol{\beta}}^{\text{ridge}} = \underset{\boldsymbol{\beta}}{\text{argmin}} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{1.1}$$

- L_2 loss + L_2 penalty
- Ridge regression penalizes $\sum_{j=1}^{p} \beta_j^2$. Large values of β_j will be penalized. As will be seen, β_j will be shrunken towards zero. The intercept β_0 is not penalized.
- λ is the tuning parameter controlling the level of penalization. The choice of optimal λ will be discussed later.
- When $\lambda \to \infty$, $\widehat{\boldsymbol{\beta}}^{\text{ridge}} \to 0$. When $\lambda \to 0$, $\widehat{\boldsymbol{\beta}}^{\text{ridge}} \to \widehat{\boldsymbol{\beta}}^{\text{ls}}$.
- Common practice: centeralize y ($\sum_{i=1}^{N} y_i = 0$) and normalize X ($\sum_{i=1}^{N} x_{ij} = 0$ and $\frac{1}{N}\sum_{i=1}^{N} x_{ij}^2 = 1$ to ensure x_j has mean 0 and standard deviation 1) before solving problem (1.1).

An equivalent way to write the ridge regression is

$$\begin{aligned}\widehat{\boldsymbol{\beta}}^{\text{ridge}} &= \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2, \\ &\text{subject to } \sum_{j=1}^{p}\beta_j^2 \le t. \end{aligned} \tag{1.2}$$

(1.1) is the Lagrangian of (1.2). λ is the Lagrange multiplier. There is a one-to-one correspondence between the parameters λ and t such that the solutions to the two problems are identical.

Let $\boldsymbol{y} = (y_1, \ldots, y_N)^T$ and $\boldsymbol{X} = (x_{ij}) \in \mathbb{R}^{N \times p}$. Assume $\boldsymbol{y}$ has been centralized so that $\beta_0 = 0$. We write (1.1) as a matrix form

$$\widehat{\boldsymbol{\beta}}^{\text{ridge}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \, (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T\boldsymbol{\beta}.$$

It can be easily seen that the solution is given by (exercise)

$$\widehat{\boldsymbol{\beta}}^{\text{ridge}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{1.3}$$

Even when $p > n$, $\widehat{\boldsymbol{\beta}}^{\text{ridge}}$ exists and is a unique solution to (1.1). (1.3) shows the benefit of penalization. When $p > n$, $\boldsymbol{X}^T\boldsymbol{X}$ is not invertible, hence $\widehat{\boldsymbol{\beta}}^{\text{ls}}$ does not exist. However, $(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})$ is positive definite, hence it is invertible.
If the design matrix is orthonormal, $\widehat{\boldsymbol{\beta}}^{\text{ridge}} = \widehat{\boldsymbol{\beta}}^{\text{ls}}/(1+\lambda)$ (exercise).
Therefore, we see that $\widehat{\boldsymbol{\beta}}^{\text{ridge}}$ is a scaled version of $\widehat{\boldsymbol{\beta}}^{\text{ls}}$ (shrunken towards zero). $\underline{\text{However, } \widehat{\beta}_j^{\text{ridge}} \neq 0, \text{ for all } j \in \{1, \ldots, p\}.}$

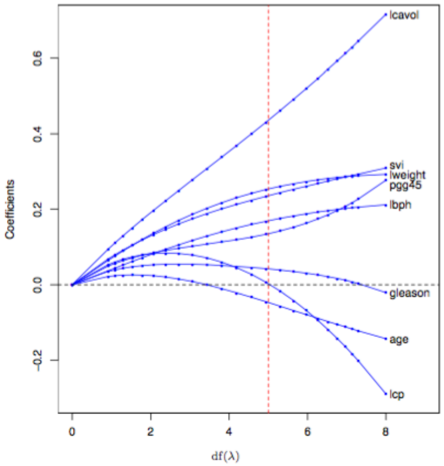


FIGURE 3.8. *Profiles of ridge coefficients for the prostate cancer example, as the tuning parameter λ is varied. Coefficients are plotted versus $\mathrm{df}(\lambda)$, the effective degrees of freedom. A vertical line is drawn at $\mathrm{df} = 5.0$, the value chosen by cross-validation.*

We prove that $\widehat{\boldsymbol{\beta}}^{\text{ridge}}$ is biased. Assume $\boldsymbol{X}$ is fixed. Let $\boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X}$. Then

$$\begin{aligned}
\widehat{\boldsymbol{\beta}}^{\text{ridge}} &= (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \\
&= (\boldsymbol{M} + \lambda\boldsymbol{I})^{-1}\boldsymbol{M}(\boldsymbol{M}^{-1}\boldsymbol{X}^T\boldsymbol{y}) \\
&= [\boldsymbol{M}(\boldsymbol{I} + \lambda\boldsymbol{M}^{-1})]^{-1}\boldsymbol{M}[\boldsymbol{M}^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\
&= (\boldsymbol{I} + \lambda\boldsymbol{M}^{-1})^{-1}\boldsymbol{M}^{-1}\boldsymbol{M}\widehat{\boldsymbol{\beta}}^{\text{ls}} \\
&= (\boldsymbol{I} + \lambda\boldsymbol{M}^{-1})^{-1}\widehat{\boldsymbol{\beta}}^{\text{ls}}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathrm{E}(\widehat{\boldsymbol{\beta}}^{\text{ridge}}) &= \mathrm{E}\{(\boldsymbol{I} + \lambda\boldsymbol{M}^{-1})^{-1}\widehat{\boldsymbol{\beta}}^{\text{ls}}\} \\
&= (\boldsymbol{I} + \lambda\boldsymbol{M}^{-1})^{-1}\boldsymbol{\beta} \\
&\neq \boldsymbol{\beta},
\end{aligned}$$

unless $\lambda = 0$.

Bayesian interpretation of ridge regression

- Suppose $y_i \sim N(\beta_0 + \boldsymbol{x}_i^T \boldsymbol{\beta}, \sigma^2)$.
- Suppose the prior distribution of $\boldsymbol{\beta}$ is $N(\mathbf{0}, \tau^2 \boldsymbol{I})$.
- Show that the negative log-posterior density of $\boldsymbol{\beta}$, with τ and σ being assumed known, is equal to the objective function in (1.1). What's the relationship between λ and (τ, σ)? (exercise).

Computation of $\widehat{\beta}^{\textbf{ridge}}$: Directly invert $(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})$ is computationally expensive. The computational cost of inverting an $p \times p$ matrix is $O(p^3)$. Rather, we'd use a singular value decomposition (SVD) method.

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T,$$

- $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p)$ is an $N \times p$ orthogonal matrix.
- $\boldsymbol{D} = \text{diag}(d_1, d_2, \ldots, d_p)$ is a $p \times p$ diagonal matrix consisting of the singular values $d_1 \geq d_2 \geq \cdots \geq d_p$.
- $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_p)$ is a $p \times p$ orthogonal matrix.
- Computational complexity of SVD: O(min$\{Np^2, N^2p\}$).
- R function: svd.

Using the SVD of $\boldsymbol{X}$, we have

$$\begin{aligned}\boldsymbol{X}^T\boldsymbol{X} &= (\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T)^T(\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T) = \boldsymbol{V}\boldsymbol{D}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T \\ &= \boldsymbol{V}\boldsymbol{D}^2\boldsymbol{V}^T\end{aligned}$$

Therefore,

$$\begin{aligned}\widehat{\boldsymbol{\beta}}^{\text{ridge}} &= (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{V}(\boldsymbol{D}^2 + \lambda\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{D}\boldsymbol{U}^T\boldsymbol{y}. \\ &= \boldsymbol{V}\text{diag}\left(\frac{d_j}{d_j^2 + \lambda}\right)\boldsymbol{U}^T\boldsymbol{y} \\ &= \sum_{j=1}^{p}\left(\frac{d_j}{d_j^2 + \lambda}\boldsymbol{u}_j^T\boldsymbol{y}\right)\boldsymbol{v}_j.\end{aligned}$$

Then, for the prediction given by ridge regression, we have

$$\widehat{\boldsymbol{y}}^{\text{ridge}} = \boldsymbol{X}\widehat{\boldsymbol{\beta}}^{\text{ridge}} = \sum_{j=1}^{p} \left(\frac{d_j^2}{d_j^2 + \lambda} \boldsymbol{u}_j^T \boldsymbol{y} \right) \boldsymbol{u}_j.$$

Interpretation:

- First, project $\boldsymbol{y}$ on the orthonormal basis $\{\boldsymbol{u}_j\}_{j=1}^p$.
- Then, shrink these coordinates by the factors $d_j^2/(d_j^2 + \lambda)$. Larger values of λ means more shrinkage.

For the Least Squares estimator, we have

$$\widehat{\boldsymbol{y}}^{\text{ls}} = \boldsymbol{X}\widehat{\boldsymbol{\beta}}^{\text{ls}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{y} = \sum_{j=1}^{p} (\boldsymbol{u}_j^T \boldsymbol{y})\boldsymbol{u}_j.$$

This further explains why $\widehat{\boldsymbol{\beta}}^{\text{ridge}}$ is regarded as a shrinkage version of $\widehat{\boldsymbol{\beta}}^{\text{ls}}$. The relationship between ridge regression and Principal Component Analysis (PCA). [read: page 66-67 of EoSL.]

Degrees of freedom for ridge regression

- A general **smoother matrix $\boldsymbol{S}$** is a linear operator satisfying:

$$\widehat{\boldsymbol{y}} = \boldsymbol{S}\boldsymbol{y}.$$

  The predicted value $\widehat{\boldsymbol{y}}$ can be regarded as a smoothing of original observations $\boldsymbol{y}$.
- In ordinary least squares, $\boldsymbol{S} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$. Suppose $\operatorname{rank}(\boldsymbol{X}) = p < N$. Then, $\operatorname{tr}(S) = p$, which is how many degrees of freedom are used in the model.
- By analogy, define the **effective degrees of freedom** for a general smoother matrix $\boldsymbol{S}$ to be

$$\operatorname{df}(\boldsymbol{S}) = \operatorname{tr}(\boldsymbol{S}).$$

In ridge regression, the predicted value

$$\widehat{\boldsymbol{y}}^{\text{ridge}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}.$$

Therefore, the smoother matrix

$$\boldsymbol{S}_\lambda = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T.$$

Hence, the degrees of freedom for ridge regression is given by

$$\mathrm{df}(\lambda) = \mathrm{tr}(\boldsymbol{S}_\lambda) = \mathrm{tr}(\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T) = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2 + \lambda}.$$

- $\mathrm{df}(\lambda)$ is monotone decreasing in λ.
- **Q:** what happens when $\lambda = 0$?

Summary of ridge regression

- Advantage of ridge regression: enable prediction even when $p > n$. It has a unique solution. It avoids singularity of $\boldsymbol{X}^T\boldsymbol{X}$ when $p > n$.
- Disadvantage of ridge regression: it is biased. It cannot conduct *variable selection*. The estimated covariate coefficients are *not* exactly zero.
- Properties of ridge regression: it can be regarded as an extension of least squares in many perspectives (solution, prediction, degrees of freedom).
- Bayesian interpretation of ridge regression.
- Computation: via SVD.

Lasso: Tibshirani (1996), JRSSB, Least Absolute Shrinkage and Selection Operator.
Lasso estimator is the solution to the following **convex** optimization problem:

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{ s.t. } \sum_{j=1}^{p} |\beta_j| \le t.$$

The equivalent Lagrangian form is

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|, \tag{1.4}$$

where $\lambda > 0$ is the Lagrangian multiplier.

Comparison of Lasso and Ridge estimator

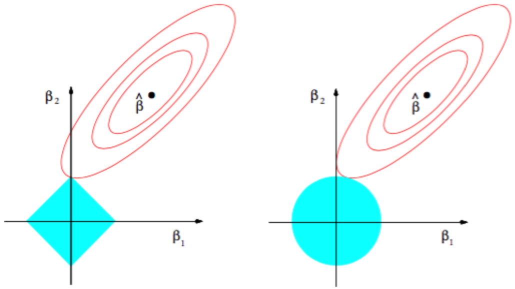


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

- Lasso uses an L_1 penalty; ridge estimator uses an L_2 penalty.
- Due to the singularity of L_1 penalty at 0, elements of Lasso estimator can be exactly 0. Ridge estimator does not have this property.
- In other words, Lasso can perform variable selection but not ridge estimator.

Generalization of Lasso and Ridge estimator

$$\widehat{\beta} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \right\} \text{ for some } q \geq 0. \tag{1.5}$$

- $q = 0$ penalizes the number of nonzero β_j's and gives the best subset estimator.
- $q = 1$ gives Lasso estimator.
- $q = 2$ gives ridge estimator.

- The L_q penalty is singular at 0 if and only if $q \leq 1$. Hence, to do variable selection, we can only use $q \leq 1$.
- Problem (1.5) is convex if and only if $q \geq 1$. When $q < 1$, the solution to (1.5) is not unique.
- Later we will see the L_q penalty will result in biased estimator.

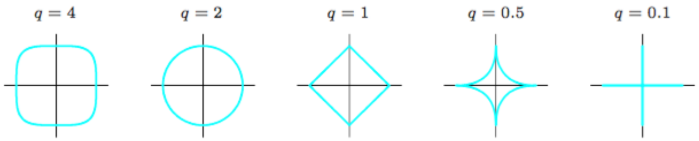


FIGURE 3.12. *Contours of constant value of $\sum_j |\beta_j|^q$ for given values of q.*

Solution path of Lasso and Ridge estimator

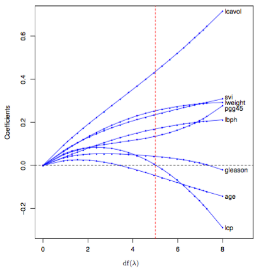


FIGURE 3.8. *Profiles of ridge coefficients for the prostate cancer example, as the tuning parameter λ is varied. Coefficients are plotted versus* $\mathrm{df}(\lambda)$*, the effective degrees of freedom. A vertical line is drawn at* $\mathrm{df} = 5.0$*, the value chosen by cross-validation.*

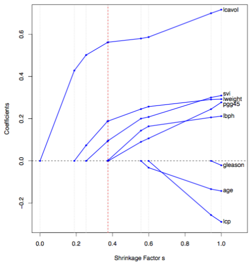


FIGURE 3.10. *Profiles of lasso coefficients, as the tuning parameter t is varied. Coefficients are plotted versus $s = t/\sum_1^p |\hat{\beta}_j|$. A vertical line is drawn at $s = 0.36$, the value chosen by cross-validation. Compare Figure 3.8 on page 65; the lasso profiles hit zero, while those for ridge do not. The profiles are piece-wise linear, and so are computed only at the points displayed; see Section 3.4.4 for details.*

- The solution path of ridge estimator is smooth.
- The solution path of Lasso is *piecewise linear*. [To see the proof, refer to Efron et al. (2004), AoS, “Least Angle Regression”.]

- When λ is large enough, all elements of Lasso estimator are 0. When λ decreases, some elements of Lasso estimator becomes nonzero.
- Later on, we will see how large λ needs to be in order to render a solution with all elements equal to 0. That value is the largest value we need to set when searching for optimal λ.
- Lasso gives a sparse solution. Ridge estimator cannot give a sparse solution.

The choice of optimal λ: via Cross Validation

- Recall that, λ controls the amount of regularization. A larger λ shrinks both Lasso and ridge estimators more towards 0.
- How do we choose the optimal λ? *We want to choose λ that minimizes the mean squared error.*
- **Cross validation:**
  - We divide the data into a *training* set and a *testing* set.
  - Using the training set, we solve Lasso(ridge) problem with a fixed λ.
  - We apply such a solution to the test set and calculate the mean squared error.
  - We choose the optimal λ as the one, which minimizes the mean squared error.

K-fold cross validation

- Partition N observations in the training data T into K separate sets of equal size. (Common choice of K is 5 or 10).
- For each $k = 1, 2, \ldots, K$ and a fixed λ, solve the Lasso or ridge problem by using data *not* contained in the k-th fold T_k to obtain $\widehat{\boldsymbol{\beta}}_{\lambda}^{(-k)}$.
- Compute the cross-validation (CV) error for the k-th fold data:

$$\mathrm{err}_k(\lambda) = |T_k|^{-1} \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in T_k} (y_i - \boldsymbol{x}_i^T \widehat{\boldsymbol{\beta}}_{\lambda}^{(-k)})^2,$$

  where $|T_k|$ is the sample size of T_k.
- Compute the overall cross validation error:

$$\mathrm{err}(\lambda) = K^{-1} \sum_{k=1}^{K} \mathrm{err}_k(\lambda).$$

- Choose the optimal tuning parameter $\lambda^* = \mathrm{argmin}_{\lambda}\, \mathrm{err}(\lambda)$.

Leave-one-out cross validation

- It's essentially K-fold cross validation with $K = N$.
- There is a convenient approximation to its cross validation error.
- Recall that a smoother matrix satisfies:

$$\widehat{\boldsymbol{y}} = \boldsymbol{S}\boldsymbol{y}.$$

- The leave-one-out cross-validated MSE has the form

$$\mathrm{err}_1(\lambda) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \boldsymbol{x}_i^T\widehat{\boldsymbol{\beta}}^{(-i)}(\lambda))^2$$

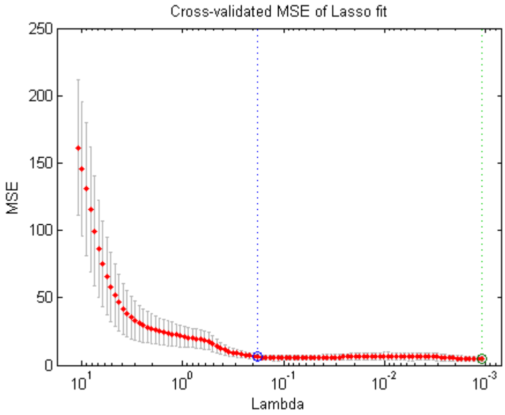


Figure: Cross validation error from a Lasso estimator

The choice of optimal λ: via information criteria

- Akaike information criterion (AIC): find the model such that AIC is minimized
$$\text{AIC} = -2\ell_n(\widehat{\boldsymbol{\beta}}) + 2k,$$
where $\ell_n(\widehat{\boldsymbol{\beta}})$ is the log-likelihood corresponding to the estimator $\widehat{\boldsymbol{\beta}}$ and k is the number of nonzero elements in $\widehat{\boldsymbol{\beta}}$.
- Bayesian information criterion (BIC): find the model such that AIC is minimized
$$\text{BIC} = -2\ell_n(\widehat{\boldsymbol{\beta}}) + k \cdot \log(n).$$
- Extended BIC: In high dimensional setting, the penalization in BIC turns out not to be enough. Chen and Chen proposed an extended BIC. See Chen and Chen (2008), Biometrika, “Extended Bayesian information criteria for model selection with large model spaces”.

How to solve Lasso problem for linear regression?

- R packages: `glmnet` (by Friedman, Hastie, Simon and Tibshirani), `lars`, `penalized`, etc.
- Algorithm: coordinate gradient descent algorithm (`glmnet`), least angle regression (`lars`)
- We will focus on coordinate gradient descent algorithm. For least angle regression, refer to Section 3.4.4 of EoSL.
- To simplify representation, we assume covariates x_{ij} are standardized: $\sum_{i=1}^N x_{ij} = 0$, $\frac{1}{N}\sum_{i=1}^N x_{ij}^2 = 1$, for $j = 1, \dots, p$. Such a standardization step is offered as a default option in `glmnet`.
- *Key of coordinate gradient descent algorithm:* solve the optimization problem one coordinate at a time. At each step, the problem has a **closed form** solution. Hence, the iterations are very fast.

Goodness of variable selection

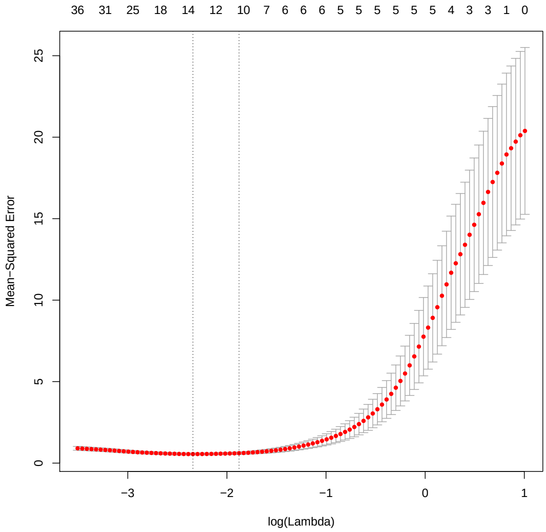
Let $\boldsymbol{\beta}^*$ be the true value $\boldsymbol{\beta}$. Denote $\mathcal{M} = \{j : \beta_j^* \neq 0\}$ and $\widehat{\mathcal{M}} := \{j : \widehat{\beta}_j \neq 0\}$, where $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_p)^T$ denotes an estimator of $\boldsymbol{\beta}^*$.

- **Variable selection consistency:** $\widehat{\mathcal{M}} = \mathcal{M}$.
- **False positives (FP):** the size of $\{j : \widehat{\beta}_j \neq 0 \text{ but } \beta_j^* = 0\}$, i.e. the number of elements being estimated as nonzero whose true value is actually 0.
- **False negatives (FN):** the size of $\{j : \widehat{\beta}_j = 0 \text{ but } \beta_j^* \neq 0\}$, i.e. the number of elements being estimated as nonzero whose true value is actually not 0.
- **Sensitivity:** $1 - (FN/|\mathcal{M}|_0)$, where $|\mathcal{M}|_0$ is the size of set $\mathcal{M}$.
- **Specificity:** $1 - (FP/|\mathcal{M}^c|_0)$, where $|\mathcal{M}^c|_0$ is the size of set $\mathcal{M}^c$.

An R example of running Lasso

```
library(glmnet)
## generate true model
N         <- 50
p         <- 100
nonzero   <- 5
X         <- matrix(rnorm(N * p), ncol = p)
beta.0    <- -1
beta.true <- c(rep(2, nonzero), rep(0, p - nonzero))
epsilon   <- rnorm(N, mean = 0, sd = 0.5)
y         <- beta.0 + X %*% beta.true + epsilon

## cross-validation to find the optimal lambda
cv  <- cv.glmnet(X, y, alpha = 1)
best_lambda <- cv$lambda.min
## plot the cross-validated mean squared error against
## lambda
plot(cv)
```


36 31 25 18 14 12 10 7 6 6 6 5 5 5 5 5 5 4 3 3 1 0
Mean−Squared Error
0
5
10
15
20
25
−3
−2
−1
0
1
log(Lambda)

```
## fit the model with optimal lambda
best.fit <- glmnet(X, y, lambda = best_lambda)

FN <- sum(as.vector(best.fit$beta)[beta.true != 0] == 0)
## FN
## [1] 0

FP <- sum(as.vector(best.fit$beta)[beta.true == 0] != 0)
## FP
## [1] 17

sensitivity <- 1 - FN / nonzero
## sensitivity
## [1] 1
specificity <- 1 - FP / (p - nonzero)
## specificity
## [1] 0.8210526
```

coordinate gradient descent algorithm

Let $R_\lambda(\beta_0, \boldsymbol{\beta}) = \frac{1}{2N}\sum_{i=1}^N (y_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta})^2$ and $p(\boldsymbol{\beta}) = \sum_{j=1}^p |\beta_j|$. We solve

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{(\beta_0, \boldsymbol{\beta})}{\operatorname{argmin}} \; R_\lambda(\beta_0, \boldsymbol{\beta}) + \lambda p(\beta).$$

Suppose at the current step, we have the solution $(\widetilde{\beta}_0, \ldots, \widetilde{\beta}_p)^T$. Now, we keep all other coordinates of $\widetilde{\boldsymbol{\beta}}$ and update $\widetilde{\beta}_j$ by solving.

$$\begin{aligned} \widetilde{\beta}_j^{\text{new}} &= \underset{\beta_j}{\operatorname{argmin}} \; R_\lambda(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}_{-j}, \beta_j) + \lambda \sum_{l \neq j} |\widetilde{\beta}_l| + \lambda|\beta_j| \\ &= \underset{\beta_j}{\operatorname{argmin}} \; \frac{1}{2N}\sum_{i=1}^N (y_i - \widetilde{y}_i^{(-j)} - x_{ij}\beta_j)^2 + \lambda|\beta_j|, \end{aligned} \tag{1.6}$$

where $\widetilde{y}_i^{(-j)} = \widetilde{\beta}_0 + \sum_{l \neq j} x_{il}\widetilde{\beta}_l$ is the predicted value of y_i excluding the contribution from x_{ij}.

Show that (exercise) the solution to (1.6) is given by

$$\widetilde{\beta}_j^{\text{new}} = S\Big(\frac{1}{N}\sum_{i=1}^N x_{ij}(y_i - \widetilde{y}_i^{(-j)}), \lambda\Big), \tag{1.7}$$

where $S(z, \lambda)$ is the soft-thresholding function defined by

$$S(z,\lambda) = \operatorname{sign}(z)(|z| - \lambda)_+ = \begin{cases} z - \lambda & \text{if } z > 0 \text{ and } \lambda < |z|; \\ z + \lambda & \text{if } z < 0 \text{ and } \lambda < |z|; \\ 0 & \text{if } \lambda \geq |z|. \end{cases}$$

[Hint] (1.6) is essentially a univariate Lasso problem. Observe that, if $\widetilde{\beta}_j > 0$, the gradient of objective function R_λ with respect to β_j has the form

$$\frac{\partial R_\lambda(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}_{-j}, \beta_j)}{\partial \beta_j} = -\frac{1}{N}\sum_{i=1}^N x_{ij}(y_i - \widetilde{\beta}_0 - \boldsymbol{x}_i^T \widetilde{\boldsymbol{\beta}}) \overset{(*)}{=} \widetilde{\beta}_j - \frac{1}{N}\sum_{i=1}^N x_{ij}(y_i - \widetilde{y}_i^{(-j)}),$$

where $(*)$ is due to standardization.
Think about how to handle the non-differentiable $|\beta_j|$.

Then, we have an important observation

$$\widetilde{\beta}_j^{\text{new}} = S\left(\widetilde{\beta}_j - \frac{\partial R_\lambda(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}_{-j}, \beta_j)}{\partial \beta_j}, \lambda\right).$$

This means that we first move along the direction of negative gradient with a step-size equal to 1, then we apply the soft-thresholding operation. This is where the name of "coordinate gradient descent" comes from. The soft-thresholding step is due to the L_1-penalty.
We don't need to compute $\widetilde{y}_i^{(-j)}$ for each j. Observe that

$$y_i - \widetilde{y}_i^{(-j)} = y_i - \widehat{y}_i + x_{ij}\widetilde{\beta}_j = r_i + x_{ij}\widetilde{\beta}_j,$$

where $\widehat{y}_i$ is the current fit of the model for observation i, and r_i is the current residual.

Thus, we have

$$\frac{1}{N}\sum_{i=1}^{N} x_{ij}(y_i - \widetilde{y}_i^{(-j)}) = \frac{1}{N}\sum_{i=1}^{N} x_{ij}r_i + \widetilde{\beta}_j. \tag{1.8}$$

To compute the gradient, it costs $O(N)$ operations. To cycle through all p variables, the computational cost is $O(pN)$. (1.8) is called the “naive update”.
A more efficient approach is called “covariance update”. Observe that

$$\sum_{i=1}^{N} x_{ij}r_i = \langle x_j, y\rangle - \sum_{k:|\widetilde{\beta}_k|>0} \langle x_j, x_k\rangle \widetilde{\beta}_k, \tag{1.9}$$

where $\langle x_j, y\rangle = \sum_{i=1}^{N} x_{ij}y_i$. We can store $\langle x_j, y\rangle$ in advance. Once x_k enters the model, we store its inner product with all the rest of the features.

Some other details

- If a coefficient is zero, or becomes zero after the soft-thresholding, it can be shown that it will always stay at zero. So there's no need to update them. It means that we don't need to update all p variables. We only need to update the nonzero coefficients.
- If the design matrix $\boldsymbol{X}$ is really sparse, we can use some sparse matrix object to save $\boldsymbol{X}$ in order to save storage space. In `glmnet`, they use "SparseMatrix".

- Weighted least square: solve

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{N} w_i (y_i - \beta_0 - \boldsymbol{x}_i^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} |\beta_j|,$$

  where $w_i > 0$ are the weight of the *i*th observation.
  Exercise: Show that the coordinate update has the form of

$$\widetilde{\beta}_j^{\text{new}} = \frac{S(\sum_{i=1}^{N} w_i x_{ij} (y_i - \widetilde{y}_i^{(-j)}), \lambda)}{\sum_{i=1}^{N} w_i x_{ij}^2}.$$

- The weighted least squares problem will shed light on solving Lasso problem for generalized linear regression. We will revisit it later.

- Comparison of Lasso and ridge solutions in the case of orthonomal columns of $\boldsymbol{X}$.
  - ridge: $\widehat{\beta}_j/(1+\lambda)$.
  - Lasso: $\text{sign}(\widehat{\beta}_j)(|\widehat{\beta}_j|-\lambda)_+$.

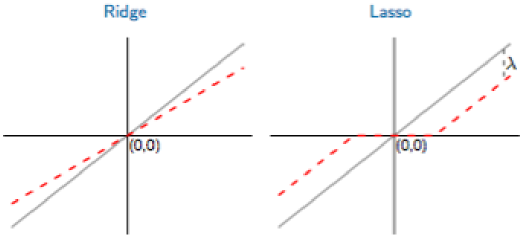


Figure: Comparison of univariate Lasso and ridge solutions.

- Pathwise coordinate gradient descent: we can always compute the largest λ ($\lambda_{\max}$) such that the entire vector $\widehat{\boldsymbol{\beta}}$ is shrunken to zero. When $\widetilde{\boldsymbol{\beta}} = \mathbf{0}$, we see that $\widetilde{\beta}_j$ will stay zero if

$$|\frac{1}{N}\sum_{i=1}^{N} x_{ij}y_i| < \lambda.$$

  Hence, $\lambda_{\max} = \max_{j \le p} |\frac{1}{N}\sum_{i=1}^{N} x_{ij}y_i|$.
- `glmnet` automatically computes $\lambda_{\max}$. You can set the minimum $\lambda_{\min} = \epsilon\lambda_{\max}$ by giving a value of ϵ. Then, `glmnet` automatically construct a sequence of K (specified by you) values of λ decreasing from $\lambda_{\max}$ to $\lambda_{\min}$ on the log scale.
- `glmnet` also allows users to define the searching grid of λ by themselves.

Summary of coordinate gradient descent algorithm for solving Lasso

- Update one coordinate at a time.
- Each update has a closed form solution. (This is the key for fast computation).
- Each update moves towards the direction of negative gradient followed by a *soft-thresholding* step. (This is a general scheme for using coordinate gradient descent algorithm to solve some optimization problem.)
- Soft-thresholding is due to the L_1 penalty. If we alter the penalty, we often need to figure out how the thresholding needs to be done.
- Since the whole problem is a convex problem, coordinate gradient descent algorithm guarantees to converge to the global minimizer.
- coordinate gradient descent algorithm will give a *sparse* solution.

Statistical properties of Lasso

- We study the statistical properties of Lasso in the setting that $p \gg n$, and p is allowed to grow with n.
- We provide a non-asymptotic statement regarding its variable selection property.
- Our model is
$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}.$$
For simplicity, we assume $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^T$ where ϵ_i's are i.i.d from $N(0, \sigma^2)$. We assume the design matrix $\boldsymbol{X}$ is fixed and its columns have been normalized to have mean 0 and standard deviation 1.
- Recall that the Lasso estimator is given by
$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda_n\|\boldsymbol{\beta}\|_1. \tag{1.10}$$
- Lasso problem is essentially a convex optimization problem.

Some general properties of convex function

- f is called convex if for any x_1 and x_2 and any $t \in [0, 1]$:

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2).$$

- If "$\leq$" is replaced by "$<$", f is called strictly convex.
- Recall that if the convex function f is differentiable

$$f(y) \geq f(x) + [\nabla f(x)]^T (y - x)$$

- What if f is not differentiable.

Subgradient of a function: g is a **subgradient** of f (not necessarily convex) at x if and only if

$$f(y) \geq f(x) + g^T(y-x) \quad \text{for all } y$$

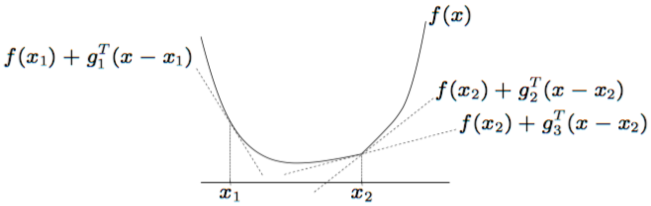
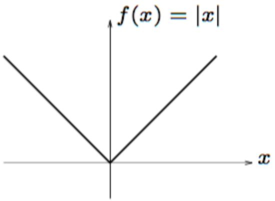

$f(x)$
$f(x_1) + g_1^T(x - x_1)$
$f(x_2) + g_2^T(x - x_2)$
$f(x_2) + g_3^T(x - x_2)$
x_1
x_2

Figure: g_2, g_3 are subgradients at x_2; g_1 is a subgradient at x_1

Subdifferential

- The set of all subgradients of f at x is called the **subdifferential** of f at x, denoted by $\partial f(x)$.
- $\partial f(x)$ is a set instead of a number. When f is differentiable and convex $\partial f(x) = \{\nabla f(x)\}$.
- See the subdifferential of $f(x) = |x|$.



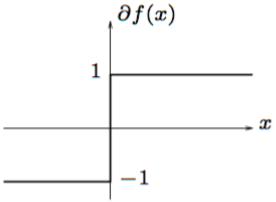


Figure: Righthand plot shows $\cup\{(x, g)|x \in \mathbb{R}, g \in \partial f(x)\}$

Some basic rules of Subdifferetial

- **Scaling:** $\partial(\alpha f) = \alpha \partial f$ (if $\alpha > 0$).
- **Addition:** $\partial(f_1 + f_2) = \partial f_1 + \partial f_2$ (RHS is addition of point-to-set mappings).
- **Affine transformation of variables:** if $g(x) = f(Ax + b)$, then $\partial g(x) = A^T \partial f(Ax + b)$.

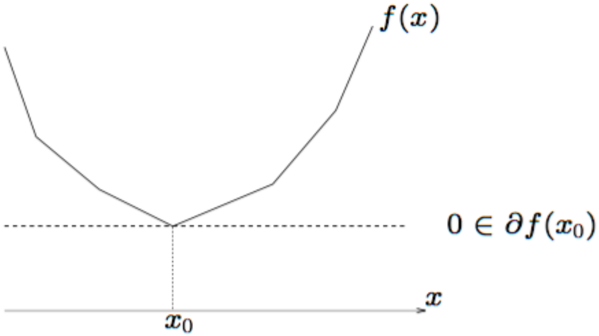
Why Subdifferential is important? It characterizes the minimizer of f.

- Recall if f is convex and differentiable

$$f(x^*) = \inf_x f(x) \text{ if and only if } \nabla f(x^*) = 0.$$

- Generalization to non-differentiable convex f:

$$f(x^*) = \inf_x f(x) \text{ if and only if } 0 \in \partial f(x^*).$$



Proof. By definition

$$f(y) \geq f(x_0) + 0^T(y - x_0) \text{ for all y if and only if } 0 \in \partial f(x_0).$$

We go back to the linear model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon},$$

- $\boldsymbol{y} \in \mathbb{R}^n$ is the response vector, $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ is the **fixed** design matrix, $\boldsymbol{\beta}^* \in \mathbb{R}^p$ is the vector of regression coefficients.
- Let $S = \{j : \beta_j^* \neq 0\}$, $S^c = \{1, \ldots, p\}\backslash S$, and $s = |S|$.
- We assume $\boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_n)$, where σ is assumed to be known.
- We assume $\boldsymbol{X}$ has been normalized such that $\sum_{i=1}^n x_{ij} = 0$ and $(1/n)\sum_{i=1}^n x_{ij}^2 = 1$.

We consider the Lasso estimator

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda_n \sum_{j=1}^p |\beta_j|. \tag{1.11}$$

We prove that $\widehat{\boldsymbol{\beta}}$ is model selection consistent (i.e. $\widehat{S} := \{j : \widehat{\beta}_j \neq 0\} = S$) and uniformly consistent to $\boldsymbol{\beta}^*$ (i.e. $\|\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*\|_\infty \to 0$).

For a matrix $\boldsymbol{M} \in \mathbb{R}^{m\times m}$, let $\|\boldsymbol{M}\|_\infty = \max_{1\le i\le m} \sum_{j=1}^m |m_{ij}|$, where m_{ij} is the (i,j)th element of $\boldsymbol{M}$. For a vector $\boldsymbol{X} \in R^p$, let $\|\boldsymbol{X}\|_\infty = \max_{1\le j\le p} |X_j|$. For two sequences a_n, b_n, denote $a_n \lesssim b_n$ if $a_n \le Cb_n$ for some $C > 0$. Let $\boldsymbol{X}_S$ be the subvector of $\boldsymbol{X}$ with indices in S.

Theorem: Suppose the following conditions hold
(C1) $\|(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\|_\infty = O(\sqrt{s}n^{-1})$;
(C2) $\|(\boldsymbol{X}_{S^c}^T\boldsymbol{X}_S)(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\|_\infty \le 1-\eta$, for some $0<\eta<1$;
(C3) $b_s \ge c_0$ for some $c_0 > 0$, where $b_s = \min_{j\in S} |\beta_j^*|$;
if we choose λ_n such that $\lambda_n = C_1 n\{(\log p)/n\}^{1/2-\alpha}$ for some $0<\alpha<1/2$ and $C_1 > 0$, there exist a constant $C > \sqrt{2}$ such that it holds with probability at least $1-p^{1-C^2/2}$ that
$[a]$ $\|\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*\|_\infty \lesssim \sqrt{s}\{(\log p)/n\}^{1/2-\alpha}$.
$[b]$ $\widehat{S} = S$.

Remark:

- The theorem is a **non-asymptotic** probabilistic statement, but implies that $\|\widehat{\beta}_S - \beta_S^*\|_\infty = o_P(1)$ and $\widehat{S} = S$ in probability. In fact, it provides more information by giving the explicit form of exception probability.
- The theorem allows both $s \to \infty$ and $p \to \infty$ (even when $p \gg n$). As long as $\sqrt{s}\{(\log p)/n\}^{1/2-\alpha} = o(1)$, consistency still exists. This is essentially a sparsity assumption (i.e. s cannot be too large) and also a requirement of the problem's dimension (i.e. we need $\log p = o(n)$).
- Condition (C1) requires that $\frac{1}{n}\boldsymbol{X}_S^T\boldsymbol{X}_S$ should be away from singular. $\frac{1}{n}\boldsymbol{X}_S^T\boldsymbol{X}_S$ is the Hessian matrix of the linear regression problem restricted to S. Even though $\frac{1}{n}\boldsymbol{X}^T\boldsymbol{X}$ is always singular, $\frac{1}{n}\boldsymbol{X}_S^T\boldsymbol{X}_S$ can still be positive definite when $s \ll n$.

Remark:

- Condition (C2) is called the "irrepresentable condition". It requires that the unimportant covariates (covariates in S^c) cannot be highly correlated with important covariates (covariates in S). To see why, if we regress $\boldsymbol{X}_j, j \in S^c$ on $\boldsymbol{X}_S$, the least square estimator is $(\boldsymbol{X}_S^T \boldsymbol{X}_S)^{-1} \boldsymbol{X}_S^T \boldsymbol{X}_j$. (C2) requires

$$\|(\boldsymbol{X}_S^T \boldsymbol{X}_S)^{-1} \boldsymbol{X}_S^T \boldsymbol{X}_j\|_1 < 1, \text{ for all } j \in S^c.$$

  Hence, this condition is restrictive. But it's known to be a necessary condition. We will see this is essentially due to the L_1 penalty we used, it can be relaxed when we choose some other penalty.
- Condition (C3) is the assumption on the minimal signal strength.

KKT conditions of Lasso

Any vector $\boldsymbol{\beta} \in \mathbb{R}^p$ satisfying the following conditions is a solution to (1.11), i.e. it is a Lasso estimator.

$$(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta})_j - \boldsymbol{X}_j^T\boldsymbol{y} + \lambda_n \mathrm{sign}(\beta_j) = 0 \quad \text{for } \beta_j \neq 0, \tag{1.12}$$

$$|(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta})_j - \boldsymbol{X}_j^T\boldsymbol{y}| < \lambda_n \quad \text{for } \beta_j = 0, \tag{1.13}$$

where $(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta})_j$ denotes the j-th element of $\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}$; the sign function is defined as $\mathrm{sign}(x) = 1$ if $x > 0$ and $\mathrm{sign}(x) = -1$ if $x < 0$. $\boldsymbol{X}_j$ is the jth column of $\boldsymbol{X}$.

- (1.12) and (1.13) are called Karush-Kuhn-Tucker (KKT) conditions of Lasso. They are sufficient conditions. For necessary conditions, just replace $<$ in (1.13) with $\leq$. There are indeed p conditions in total.
- We see that the KKT conditions essentially means

$$\mathbf{0} \in \nabla_{\boldsymbol{\beta}} \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda_n \partial \|\boldsymbol{\beta}\|_1 .$$

  In general, the KKT condition of claiming x^* is the optimizer of a function $f(x)$ ($f(x)$ can be nondifferentiable) is $\underline{0 \in \partial f(x^*)}$.
- KKT conditions play a key role in proving statistical properties of LASSO (and many other penalized estimators).
- KKT conditions give the sufficient and necessary conditions for a vector to be the minimizer of an optimization problem.

Proof: We prove the results through the following two steps.

1. Let $\mathcal{N} := \{\boldsymbol{\delta} \in \mathbb{R}^s, \|\boldsymbol{\delta} - \boldsymbol{\beta}_S^*\|_\infty \leq D\sqrt{s}\{(\log p)/n\}^{1/2-\alpha}\}$ for some $D > 0$. We show that with probability at least $1 - p^{1-C^2/2}$, there exists a solution $\widehat{\boldsymbol{\beta}}_S$ in set $\mathcal{N}$ to (1.12).
2. We expand $\widehat{\boldsymbol{\beta}}_S$ to $\widehat{\boldsymbol{\beta}} = (\widehat{\boldsymbol{\beta}}_S, \mathbf{0})^T$. We show that $\widehat{\boldsymbol{\beta}}$ satisfies

(1.13).
These two steps complete the proof.

[1] Note that, (1.12) is equivalent to

$$\boldsymbol{X}_S^T \boldsymbol{X}_S \boldsymbol{\beta}_S - \boldsymbol{X}_S^T \boldsymbol{y} + \lambda_n \text{sign}(\boldsymbol{\beta}_S) = \mathbf{0}. \tag{1.14}$$

Since $\boldsymbol{y} = \boldsymbol{X}_S \boldsymbol{\beta}_S^* + \boldsymbol{\epsilon}$, (1.14) can be written as

$$\boldsymbol{X}_S^T \boldsymbol{X}_S (\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^*) - \boldsymbol{X}_S^T \boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S) = \mathbf{0}.$$

Let event $A := \{\|\boldsymbol{X}_S^T \boldsymbol{\epsilon}\|_\infty \leq C\sigma\sqrt{n \log p}\}$, where $C > \sqrt{2}$. First, we prove that $P(A) \geq 1 - p^{-C^2/2}$.
Note that, $\sum_{i=1}^n x_{ij}\epsilon_i \sim N(0, n\sigma^2)$, since we normalize $\boldsymbol{X}_j$ such that $(1/n)\sum_{i=1}^n x_{ij}^2 = 1$. Then, we have

$$P\left(\frac{1}{\sqrt{n}\sigma}|\sum_{i=1}^n x_{ij}\epsilon_i| \geq t\right) \leq 2\Phi(-t) \overset{(i)}{\leq} \frac{2}{t}\exp(-t^2/2), \tag{1.15}$$

where $\Phi(\cdot)$ is the c.d.f of the standard normal distribution. [Exercise: prove (i).]

Let $t = C\sqrt{\log p}$ for some $C > \sqrt{2}$, we have

$$P(|\sum_{i=1}^{n} x_{ij}\epsilon_i| \geq C\sigma\sqrt{n\log p}) \leq \frac{2}{C\sqrt{\log p}}\exp\{-C^2(\log p)/2\} \leq p^{-C^2/2},$$

where we assume without loss of generality that $C(\log p)^{1/2} \geq 2$. By definition and the union bound, we have

$$\begin{aligned} P(\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_\infty \geq C\sigma\sqrt{n\log p}) &= P(\cup_{j=1}^{p}\{|\boldsymbol{X}_j^T\boldsymbol{\epsilon}| \geq C\sigma\sqrt{n\log p}\}) \\ &\leq p \cdot P(|\boldsymbol{X}_j^T\boldsymbol{\epsilon}| \geq C\sigma\sqrt{n\log p}) \\ &\leq p^{1-C^2/2}. \end{aligned}$$

All the following arguments are conditioning on the event A. Define functions

$$f(\boldsymbol{\beta}_S) = \boldsymbol{X}_S^T\boldsymbol{X}_S(\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^*) + \boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S) \tag{1.16}$$

$$g(\boldsymbol{\beta}_S) = (\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^*) + (\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S)\}. \tag{1.17}$$

By condition (C1),

$$\begin{aligned}
\|(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S)\}\|_\infty &\le \|(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\|_\infty \|\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S)\|_\infty \\
&\lesssim \sqrt{s} n^{-1}(\|\boldsymbol{X}_S^T\boldsymbol{\epsilon}\|_\infty + \lambda_n) \\
&\lesssim \sqrt{s} n^{-1}(\sqrt{n \log p} + \lambda_n) \\
&\lesssim \sqrt{s}\{(\log p)/n\}^{1/2-\alpha},
\end{aligned}$$

where in the last inequality, we use the fact that $\lambda_n/n = O(\{(\log p)/n\}^{1/2-\alpha})$.

Then, when n is sufficiently large, if $(\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^*)_j = D\sqrt{s}\{(\log p)/n\}^{1/2-\alpha}$ for some large $D > 0$,

$$\{g(\boldsymbol{\beta}_S)\}_j \geq D\sqrt{s}\{(\log p)/n\}^{1/2-\alpha} - [(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S)\}]_j \geq 0.$$

If $(\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^*)_j = -D\sqrt{s}\{(\log p)/n\}^{1/2-\alpha}$,

$$\{g(\boldsymbol{\beta}_S)\}_j \leq -D\sqrt{s}\{(\log p)/n\}^{1/2-\alpha} + [(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \text{sign}(\boldsymbol{\beta}_S)\}]_j \leq 0.$$

By condition (C3), $g(\boldsymbol{\beta})$ is continuous in $\mathcal{N}$. By Miranda's existence theorem, $g(\boldsymbol{\beta}) = \mathbf{0}$ has a solution $\widehat{\boldsymbol{\beta}}_S$ in $\mathcal{N}$. Obviously, it also solves (1.16).

[2] We let $\widehat{\boldsymbol{\beta}} = (\widehat{\boldsymbol{\beta}}_S, \mathbf{0})^T$. Note that, (1.13) requires that

$$\begin{aligned}(\boldsymbol{X}^T\boldsymbol{X}\widehat{\boldsymbol{\beta}})_{S^c} - \boldsymbol{X}_{S^c}^T\boldsymbol{Y} &= \boldsymbol{X}_{S^c}^T\boldsymbol{X}_S\widehat{\boldsymbol{\beta}}_S - \boldsymbol{X}_{S^c}^T(\boldsymbol{X}_S\boldsymbol{\beta}_S^* + \boldsymbol{\epsilon}) \\ &= \boldsymbol{X}_{S^c}^T\boldsymbol{X}_S(\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*) - \boldsymbol{X}_{S^c}^T\boldsymbol{\epsilon}.\end{aligned} \tag{1.18}$$

Since $\widehat{\boldsymbol{\beta}}_S$ solves (1.17), $\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^* = -(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \mathrm{sign}(\widehat{\boldsymbol{\beta}}_S)\}$.
Hence, (1.18) equals to

$$\boldsymbol{X}_{S^c}^T\boldsymbol{X}_S(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\{\boldsymbol{X}_S^T\boldsymbol{\epsilon} + \lambda_n \mathrm{sign}(\widehat{\boldsymbol{\beta}}_S)\}.$$

Hence,

$$\begin{aligned}\|(\boldsymbol{X}^T\boldsymbol{X}\widehat{\boldsymbol{\beta}})_{S^c} - \boldsymbol{X}_{S^c}^T\boldsymbol{Y}\|_\infty &\le \|\boldsymbol{X}_{S^c}^T\boldsymbol{X}_S(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\|_\infty(\|\boldsymbol{X}_S^T\boldsymbol{\epsilon}\|_\infty + \lambda_n) + \|\boldsymbol{X}_{S^c}^T\boldsymbol{\epsilon}\|_\infty \\ &< \|\boldsymbol{X}_S^T\boldsymbol{\epsilon}\|_\infty + (1-\eta)\lambda_n + \|\boldsymbol{X}_{S^c}^T\boldsymbol{\epsilon}\|_\infty \\ &< \lambda_n,\end{aligned} \tag{1.19}$$

since conditioning on event A, $\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_\infty = O(\sqrt{n\log p}) = o(\lambda_n)$.

Remark of the proof.

- The key of proof is KKT conditions + concentration inequality (1.15).
- Concentration inequality quantifies how a random variable deviates from its expected value. (Note that $E(X_j\epsilon) = 0$).
- The concentration result in (1.15) is critical for the convergence rate.
    - Wiki "concentration inequality" for a quick reference. A more comprehensive reference: Massart, P. (2007). Concentration inequalities and model selection. Berlin: Springer.
- The normality assumption can be further relaxed to sub-Gaussian assumption. The same type of concentration inequality can be established.
- The upper bound in the irrepresentable condition is solely determined by the L_1-penalty used by Lasso. Recall that the subdifferential of L_1-penalty at the original is $[-1, 1]$. If we choose other penalty, the upper bound can allow to grow.

High dimensional generalized linear regression model

- linear regression, logistic regression, Poisson regression, multinomial regression, etc.

Denote $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$ as the $N \times p$ design matrix with $\boldsymbol{x}_j = (x_{1j}, \ldots, x_{Nj})^T, j = 1, \ldots, p$ and $\boldsymbol{y} = (y_1, \ldots, y_N)^T$ as the response vector. With a canonical link, the conditional distribution of $\boldsymbol{y}|X$ is assumed to be belonged to the canonical exponential family and has the form

$$f(\boldsymbol{y}, \boldsymbol{X}; \boldsymbol{\beta}) = \prod_{i=1}^{N} \left\{ c(y_i) \exp\left[\frac{y_i\theta_i - b(\theta_i)}{\phi} \right] \right\},$$

where $(\theta_1, \ldots, \theta_n) = \boldsymbol{X}\boldsymbol{\beta}$, the function $b(\theta)$ is assumed to be known with twice continuously differentiable $b''(\theta) > 0$, and $\phi \in (0, \infty)$ is the dispersion parameter.
Our goal is to estimate $\boldsymbol{\beta}$ under the setting when $p \gg n$.

Penalized MLE

The log-likelihood, up to an affine transformation, is given by

$$\ell(\boldsymbol{\beta}) = \{\boldsymbol{y}^T \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{e}^T \boldsymbol{b}(\boldsymbol{X}\boldsymbol{\beta})\},$$

where $\boldsymbol{b}(\boldsymbol{\theta}) = (b(\theta_1), \ldots, b(\theta_n))^T$ and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)^T$.
Then, the L_1-regularized MLE is given by

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} -\ell(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} |\beta_j|.$$

Next, we discuss how to solve the above problem for some concrete example (logistic regression).

Regularized logistic regression

$$\log \frac{\Pr(y=1|\boldsymbol{x})}{\Pr(y=0|\boldsymbol{x})} = \beta_0 + \boldsymbol{x}^T\boldsymbol{\beta}.$$

The log-likelihood of the logistic model is given by

$$\ell(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{N} \{y_i(\beta_0 + \boldsymbol{x}_i^T\boldsymbol{\beta}) - \log(1 + \exp(\beta_0 + \boldsymbol{x}_i^T\boldsymbol{\beta}))\}. \tag{1.20}$$

Then, we solve

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{(\beta_0, \boldsymbol{\beta})}{\operatorname{argmin}} -\ell(\beta_0, \boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} |\beta_j|.$$

Local Quadratic Approximation (LQA)

Suppose $(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}})$ is the current solution. We do a local quadratic approximation to the negative log-likelihood. Let $\ell_i(\beta_0, \boldsymbol{\beta})$ be the i-th summand on the right hand side of (1.20). We have

$$\begin{aligned}-\ell_i(\beta_0, \boldsymbol{\beta}) \approx & -\ell_i(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}) - \ell_i'(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}})\{\beta_0 + \boldsymbol{x}_i^T\boldsymbol{\beta} - \widetilde{\beta}_0 - \boldsymbol{x}_i^T\widetilde{\boldsymbol{\beta}}\} \\ & - \frac{1}{2}\ell_i''(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}})\{\beta_0 + \boldsymbol{x}_i^T\boldsymbol{\beta} - \widetilde{\beta}_0 - \boldsymbol{x}_i^T\widetilde{\boldsymbol{\beta}}\}^2.\end{aligned}$$

After some algebra, we find

$$\ell_i'(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}) = y_i - \widetilde{p}(x_i), \text{ where } \widetilde{p}(x_i) = \frac{\exp(\widetilde{\beta}_0 + \boldsymbol{x}_i^T\widetilde{\boldsymbol{\beta}})}{1 + \exp(\widetilde{\beta}_0 + \boldsymbol{x}_i^T\widetilde{\boldsymbol{\beta}})}.$$

$$\ell_i''(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}) = -\widetilde{p}(x_i)\{1 - \widetilde{p}(x_i)\}.$$

Substitute them into (1.20), we have

$$-\ell(\beta_0, \boldsymbol{\beta}) \approx \frac{1}{2}\sum_{i=1}^{N} w_i(z_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta})^2 + C(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}),$$

where

$$\begin{aligned} z_i &= \widetilde{\beta}_0 + \boldsymbol{x}^T\widetilde{\boldsymbol{\beta}} + \frac{y_i - \widetilde{p}(x_i)}{\widetilde{p}(x_i)\{1-\widetilde{p}(x_i)\}}, \quad \text{(working response)} \\ w_i &= \widetilde{p}(x_i)\{1-\widetilde{p}(x_i)\}, \quad \text{(weights)}. \end{aligned}$$

Therefore, at the current step, we only need to solve the following weighted least squares problem.

$$\begin{aligned} (\widetilde{\beta}_0^{\text{new}}, \widetilde{\boldsymbol{\beta}}^{\text{new}}) &= \underset{(\beta_0,\boldsymbol{\beta})}{\text{argmin}}\ \frac{1}{2}\sum_{i=1}^{N} w_i(z_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta})^2 + C(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}}) + \lambda\sum_{j=1}^{p}|\beta_j| \\ &= \underset{(\beta_0,\boldsymbol{\beta})}{\text{argmin}}\ \frac{1}{2}\sum_{i=1}^{N} w_i(z_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta})^2 + \lambda\sum_{j=1}^{p}|\beta_j|. \end{aligned}$$

It can be solved by coordinate gradient descent algorithm.

The whole algorithm for solving (1.20) with a fixed λ is as follows.

- outer loop: Update the quadratic approximation ℓ_Q using the current parameters $(\widetilde{\beta}_0, \widetilde{\boldsymbol{\beta}})$.
- inner loop: Run the coordinate gradient descent algorithm on the penalized weighted least squares problem.
- Iterate between the two loops.

Implementation in `glmnet`

- `glmnet(X, y, family=binomial)`

More thoughts about the algorithm

- The algorithm minics the Iteratively Reweighted Least Squares (IRLS) for solving the ordinary logistic regression (without penalty). IRLS also uses local quadratic approximation to the log-likelihood function at each iteration. The extra effort brought by penalization is essentially the soft-thresholding.
- This algorithm sheds lights on how to solve a general optimization problem

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \; \ell(\boldsymbol{\beta}) + \lambda p(\boldsymbol{\beta}),$$

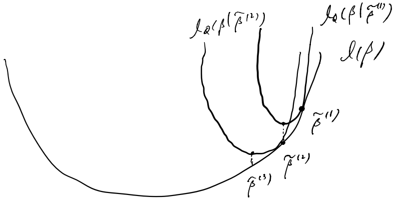
  where $\ell(\boldsymbol{\beta})$ is a general loss function and $p(\boldsymbol{\beta})$ is a general penalty function. The key is that if $\ell(\boldsymbol{\beta})$ is sufficiently smooth, we can do a second-order Taylor expansion and approximate $\ell(\boldsymbol{\beta})$ with a quadratic function $\ell_Q(\boldsymbol{\beta})$. Then, solve

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \; \ell_Q(\boldsymbol{\beta}) + \lambda p(\boldsymbol{\beta}),$$

  which will just be a weighted least squares problem.

Majorization-Minimization (MM) Algorithm:

- The key idea of MM algorithm is to find a function majorizing the objective function from the above and then minimize the majorization function.
- Here's an illustration of MM algorithm.



- Usually, the minimizer of the dominating function needs to be easy to be solved.
- For example, suppose $\ell(\boldsymbol{\beta})$ is twice continuously differentiable, we have

$$\begin{aligned}\ell(\boldsymbol{\beta}) &= \ell(\widetilde{\boldsymbol{\beta}}) + [\nabla \ell(\widetilde{\boldsymbol{\beta}})]^T(\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}}) + (1/2)(\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}})^T[\nabla^2 \ell(\bar{\boldsymbol{\beta}})](\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}}) \\ &\le \underbrace{\ell(\widetilde{\boldsymbol{\beta}}) + [\nabla \ell(\widetilde{\boldsymbol{\beta}})]^T(\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}}) + (c/2)(\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}})^T(\boldsymbol{\beta}-\widetilde{\boldsymbol{\beta}})}_{\ell_Q(\boldsymbol{\beta})},\end{aligned}$$

where $c \ge \sup_{\boldsymbol{\beta}} \lambda_{\max}(\nabla^2 \ell(\boldsymbol{\beta}))$. Then, we can update $\widetilde{\boldsymbol{\beta}}$ by

$$\widetilde{\boldsymbol{\beta}}^{(\text{new})} = \widetilde{\boldsymbol{\beta}} - \left\{\frac{1}{c}[\nabla \ell(\widetilde{\boldsymbol{\beta}})]\right\},$$

i.e., we move towards the direction of negative gradient of $\ell(\boldsymbol{\beta})$ with a *stepsize* of 1/c.

- There are studies showing that MM algorithm guarantees to converge to the global minimizer for convex optimization problem.
- Even if a non-differential penalty function exists, the above principles still apply. Usually we only need to add another thresholding step, depending on which penalty is used. If L_1-penalty is used, we only need to apply soft-thresholding to $\widetilde{\beta}^{(\text{new})}$.
- For more information, read Friedman, Hastie and Tibshirani (2010) “Regularization Paths for Generalized Linear Models via Coordinate Descent”. Journal of Statistical Software.

Variable selection consistency for penalized generalized linear regression model

- Similarly as the consistency result for Lasso under linear regression model, we can also give non-asymptotic result for penalized generalized linear regression model.
- The key of proof is again KKT conditions + concentration inequality.
- Reference: Fan and Lv (2011). Non-Concave Penalized Likelihood with NP-Dimensionality. IEEE Transactions on Information Theory.

Variation of Lasso

Revisit the penalized least squares problem

$$\widehat{\theta} = \underset{\theta}{\operatorname{argmin}} \ \frac{1}{2}(z-\theta)^2 + p_\lambda(|\theta|), \tag{1.21}$$

where $p_\lambda(|\theta|)$ is a general penalty function of θ and depends on λ, e.g. $p_\lambda(|\theta|) = \lambda|\theta|$ gives the Lasso penalty.

- If we choose the hard-thresholding penalty function

  $$p_\lambda(|\theta|) = \frac{1}{2}\{\lambda^2 - (|\theta|-\lambda)^2 I(|\theta| < \lambda)\},$$

  the solution is given by hard-thresholding rule $\widehat{\theta} = zI(|z| > \lambda)$ (exercise).

In general, we want to design a penalty function which should result in an estimator with the following three properties.

- **Unbiasedness:** the resulting estimator should be unbiased when the true unknown parameter is large to avoid unnecessary modeling bias.
- **Sparsity:** the resulting estimator is a thresholding rule, which automatically sets small estimated coefficient to zero to reduce model complexity.
- **Continuity:** the resulting estimator is continuous in data z to avoid instability in model prediction.

How to design the penalty function such that the above three requirements are met?

- The first order derivative of (1.21) with respect to θ is

$$\operatorname{sign}(\theta)\{|\theta| + p_\lambda'(|\theta|)\} - z.$$

- It's easy to see when $p_\lambda'(|\theta|) = 0$ for large $|\theta|$, the resulting estimator is z when $|z|$ is sufficiently large. Hence, the requirement of "unbiasedness" is met.
- To achieve "sparsity", *the minimum of the function* $|\theta| + p_\lambda'(|\theta|)$ should be positive.
    - When $|z| < \min_{\theta \neq 0} |\theta| + p_\lambda'(|\theta|)$, the derivative of (1.21) is positive for all positive θ's (and is negative for all negative θ's). Therefore, the penalized LSE is 0 in this case. Hence, it automatically shrinks the small estimated coefficients to be *exactly* zero.

- To achieve "continuity", *the minimum of the function* $|\theta| + p'_\lambda(|\theta|)$ should be attained at 0. Otherwise, the estimated coefficient won't be continuous where $|\theta| + p'_\lambda(|\theta|)$ attains its minimum.

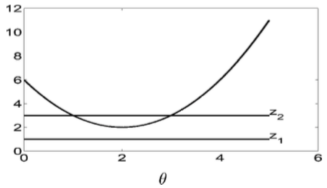


Figure 3. A Plot of $\theta + p'_\lambda(\theta)$ Against $\theta(\theta > 0)$.

To sum up, a good penalty function should have

- $p'_\lambda(|\theta|) = 0$ for large $|\theta|$. (unbiasedness)
- $p_\lambda(\cdot)$ should be singular at the origin. (sparsity)
- $|\theta| + p'_\lambda(|\theta|)$ attains its minimum at the origin. (continuity)

- For the L_q-penalty $p_\lambda(|\theta|) = \lambda|\theta|^q$, the solution is continuous only when $q \geq 1$. However, when $q > 1$, the solution is not sparse. When $q = 1$, the solution is not unbiased. Hence, L_q-penalty cannot meet all three requirements.
- The hard-thresholding penalty cannot result in a continuous estimator.

Fan and Li (2001) proposed the SCAD (Smoothly Clipped Absolute Deviation) penalty.

$$p'_\lambda(\theta) = \lambda \left\{ I(\theta \leq \lambda) + \frac{(a\lambda - \theta)_+}{(a-1)\lambda} I(\theta > \lambda) \right\}, \text{ for some } a > 2 \text{ and } \theta > 0.$$

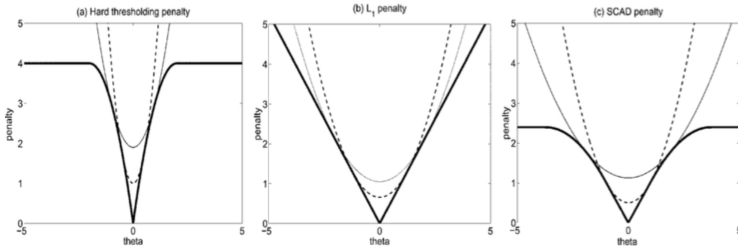


Figure 1. Three Penalty Functions $p_\lambda(\theta)$ and Their Quadratic Approximations. The values of λ are the same as those in Figure 5(c).

The SCAD penalty is essentially a quadratic spline function with knots at λ and $a\lambda$. a and λ are two tuning parameters. Fan and Li (2001) recommended to choose $a = 3.7$.
The solution of (1.21) with a SCAD penalty is given by the SCAD thresholding operator

$$\widehat{\theta} = \begin{cases} \operatorname{sign}(z)(|z| - \lambda)_+, & \text{when } |z| \leq 2\lambda, \\ \{(a-1)z - \operatorname{sign}(z)a\lambda\}/(a-2), & \text{when } 2\lambda < |z| \leq a\lambda, \\ z, & \text{when } |z| > a\lambda. \end{cases} \tag{1.22}$$

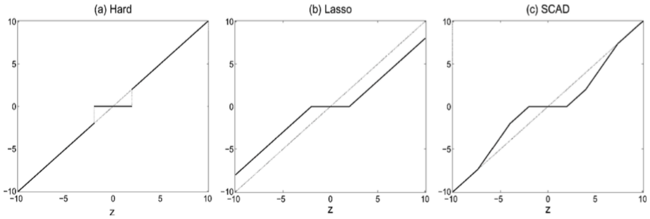


Figure 2. Plot of Thresholding Functions for (a) the Hard, (b) the Soft, and (c) the SCAD Thresholding Functions With $\lambda = 2$ and $a = 3.7$ for SCAD.

Notice that, since the SCAD penalty is not a convex function of θ. The whole problem of (1.21) is no longer a convex optimization problem. For a non-convex optimization problem, usually there exists multiple local optimizers. However, it is often hard to find the global optimizer. Usually we can only prove the theories for one local optimizer. But we cannot guarantee that the solution given by an algorithm will be that optimizer.

Computation of SCAD estimator

$$\widehat{\boldsymbol{\beta}}^{\text{scad}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} -L(\boldsymbol{\beta}) + \sum_{j=1}^{p} p_\lambda(|\beta_j|), \tag{1.23}$$

where $L(\boldsymbol{\beta})$ is the log-likelihood function and $p_\lambda(|\beta_j|)$ is the SCAD penalty.

- At each iteration, we do Local Quadratic Approximation (LQA) to $-L(\boldsymbol{\beta})$, which renders a weighted least squares problem.
- We can use coordinate gradient descent algorithm to solve the weighted least squares problem.
- For each coordinate, that will be a univariate weighted least squares problem, which is similar to (1.21). Then, we will have a closed form solution by using SCAD thresholding operation (1.22).
- We iterate between the above steps until the algorithm converges.
- R package: `ncvreg`

Fan and Li (2001) provided another algorithm to solve (1.23). The algorithm approximate the SCAD penalty by a quadratic function. As for the log-likelihood function, they also used LQA.

Notice that the problem (1.23) *is a non-convex optimization problem. Therefore, the above algorithm can only converge to a local minimizer, instead of the global minimizer. In recent years, there are studies regarding the statistical properties of the local minimizers.*

Variation of Lasso

Adaptive Lasso was proposed by Zou (2006), JASA, “Adaptive Lasso and its oracle properties”.
Adaptive Lasso is in the same spirit of SCAD: remove the bias caused by Lasso. Lasso imposes equal penalization on every coefficient. But the large coefficients should not be penalized. Only small coefficients should be shrunken to 0.
The key of adaptive Lasso is putting different weights on the penalization of different coefficients. Penalize less heavily for large coefficients.

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} w_j |\beta_j|.$$

Suppose that $\widehat{\boldsymbol{\beta}}$ is a root-n consistent estimator to $\boldsymbol{\beta}^*$, e.g. $\widehat{\boldsymbol{\beta}}^{\text{ls}}$. Pick a $\gamma > 0$, and define the weight vector $\widehat{\boldsymbol{w}} = 1/|\widehat{\boldsymbol{\beta}}^{\text{ls}}|^{\gamma}$. The adaptive Lasso estimator $\widehat{\boldsymbol{\beta}}^{\text{aLasso}}$ is given by

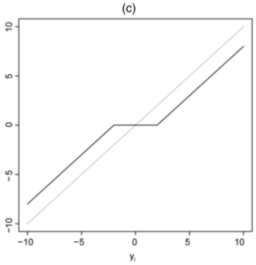
$$\widehat{\boldsymbol{\beta}}^{\text{aLasso}} = \operatorname*{argmin}_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} \widehat{w}_j |\beta_j|.$$

Adaptive Lasso has the oracle properties. See Theorem 2 of Zou (2006)

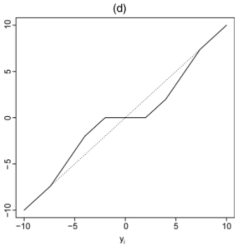
Some remarks of Adaptive Lasso

- It's essentially a two-step procedure and requires an initial *consistent* estimator $\widehat{\beta}$.
- The adaptive Lasso problem is convex, not like SCAD. Hence, it does not have the problem of multiple local minimizers.
- The initial estimator does not always need to be root-n consistent. As long as there exists $a_n \to \infty$ and $a_n(\widehat{\beta} - \beta^*) = O_P(1)$. The resulting estimator still has oracle properties, given that λ is chosen properly.
- The theorem established in Zou (2006) is for fixed p. When $p \gg n$, $\widehat{\beta}^{\text{ls}}$ can no longer be used as the initial estimator. There is a remedy in high-dimensional setting by using one-step technique. Refer to Fan et al. (2014). "Strong oracle optimality of folded concave penalized estimation." The Annals of Statistics, 42, 819-849.

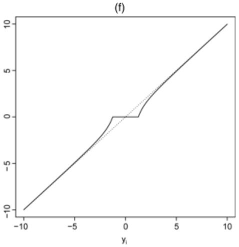
- The data-adaptive weights $\widehat{w}$ is the key for oracle properties. As the sample size grows, the weights for 0 coefficient predictors get inflated (to infinity), whereas the weights for nonzero coefficient predictors converge to a finite constant. This is the key to guarantee unbiasedness and sparsity simultaneously.



(a) Lasso



(b) SCAD



(c) Adaptive Lasso with $\gamma = 2$

Elastic Net

Limitations of Lasso

- For highly correlated variables, Lasso tends to select one from a group and ignore others.
- A desired method should have “grouping” effect for highly correlated variables, i.e. select these variables together.

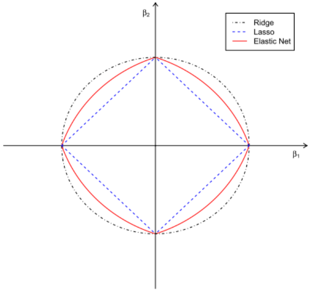
Elastic Net (Zou and Hastie (2005), JRSSB, “Regularization and variable selection via the elastic net”)

$$\widehat{\boldsymbol{\beta}}^{\text{Naive ENet}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{\beta}\|_2^2.$$

- The L_1 penalty generates a sparse model.
- The quadratic penalty encourages *grouping* effect and stabilizes the L_1 regularization path.

Geometry of the elastic net

2-dimensional illustration $\alpha = 0.5$



The equivalent form of elastic net

$$\min \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 \text{ s.t. } J(\boldsymbol{\beta}) \leq t,$$

where $J(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$, $\alpha = \lambda_1/(\lambda_1+\lambda_2)$.

- Singularities at the vertexes (necessary for sparsity)
- Strict convex edges. The strength of convexity varies with α (grouping).

Grouping effect: a regression method exhibits the grouping effect if the regression coefficients of a group of highly correlated variables equal (up to a change of sign if negatively correlated). In particular, when some variables are exactly identical, the regression method should assign identical coefficients to the identical variables.

Theorem 1 of Zou and Hastie (2005): Suppose $\boldsymbol{y}$ is centered and the predictors $\boldsymbol{X}$ are standardized. Let $\widehat{\boldsymbol{\beta}}(\lambda_1, \lambda_2)$ be the naive elastic net estimate. Suppose $\widehat{\beta}_i(\lambda_1, \lambda_2)\widehat{\beta}_j(\lambda_1, \lambda_2) > 0$. Define

$$D_{\lambda_1,\lambda_2} = \frac{1}{\|\boldsymbol{y}\|_1} |\widehat{\beta}_i(\lambda_1, \lambda_2) - \widehat{\beta}_j(\lambda_1, \lambda_2)|;$$

then

$$D_{\lambda_1,\lambda_2}(i,j) \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)},$$

where $\rho = \boldsymbol{x}_i^T \boldsymbol{x}_j$.

- $D_{\lambda_1,\lambda_2}(i,j)$ describes the difference between the coefficient paths of predictors i and j.
- If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are highly correlated, $\rho \approx 1$. Theorem 1 says that $D_{\lambda_1,\lambda_2}(i,j)$ is almost 0.
- The upper bound provides a quantitative description for the grouping effect of the naive elastic net.
- Lasso does not have grouping effect. Consider the linear model with $p = 2$, Tibshirani (1996) showed that $|\widehat{\beta}_1 - \widehat{\beta}_2| = |\cos(\theta)|$, where θ is the angle between $\boldsymbol{y}$ and $\boldsymbol{x}_1 - \boldsymbol{x}_2$. It is easy to construct examples such that $\rho = \text{corr}(\boldsymbol{x}_1, \boldsymbol{x}_2) \to 1$ but $\cos(\theta)$ does not vanish (exercise).

Connection between elastic net and Lasso

Lemma 1 of Zou and Hastie (2005) Given data set $(\boldsymbol{y}, \boldsymbol{X})$ and (λ_1, λ_2), define an artificial data set $(\boldsymbol{y}^*, \boldsymbol{X}^*)$ by

$$\boldsymbol{X}^*_{(n+p)\times p} = (1+\lambda_2)^{-1/2}\begin{pmatrix}\boldsymbol{X}\\ \sqrt{\lambda_2}\boldsymbol{I}\end{pmatrix}, \qquad \boldsymbol{y}^*_{(n+p)} = \begin{pmatrix}\boldsymbol{y}\\ 0\end{pmatrix}.$$

Let $\gamma = \lambda_1/\sqrt{1+\lambda_2}$ and $\boldsymbol{\beta}^* = \sqrt{1+\lambda_2}\boldsymbol{\beta}$. Then, the naive elastic net criterion can be written as

$$\widehat{\boldsymbol{\beta}}^* = \underset{\boldsymbol{\beta}^*}{\operatorname{argmin}} \|\boldsymbol{y}^* - \boldsymbol{X}^*\boldsymbol{\beta}^*\|_2^2 + \gamma\|\boldsymbol{\beta}^*\|_1.$$

And the naive elastic net estimator has

$$\widehat{\boldsymbol{\beta}}^{\text{Naive ENet}} = \frac{1}{\sqrt{1+\lambda_2}}\widehat{\boldsymbol{\beta}}^*$$

Elastic net with scaling correction

$$\widehat{\boldsymbol{\beta}}^{\text{ENet}} = (1+\lambda_2)\widehat{\boldsymbol{\beta}}^{\text{Naive ENet}}$$

- The rescaling keeps the grouping effect and overcome bias brought by the quadratic penalty. Recall that when $\boldsymbol{x}$ is orthonormal, $\widehat{\boldsymbol{\beta}}^{\text{ridge}} = (1+\lambda)^{-1}\widehat{\boldsymbol{\beta}}^{\text{ls}}$.
- Another perspective of Lasso and Elastic net.
    - Consider $\widehat{\boldsymbol{\Sigma}} = \boldsymbol{X}^T\boldsymbol{X}$ and $\widehat{\boldsymbol{\Sigma}}_{\lambda_2} = (1-\gamma)\widehat{\boldsymbol{\Sigma}} + \gamma\boldsymbol{I}$, $\gamma = \frac{\lambda_2}{1+\lambda_2}$. $\widehat{\boldsymbol{\Sigma}}_{\lambda_2}$ is a regularized estimate for the covariance matrix of predictors. Later in this course, we will see that the sample $\widehat{\boldsymbol{\Sigma}}$ won't be a good estimator of Σ^*.
    - We can show that (exercise)

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, \boldsymbol{\beta}^T\boldsymbol{\Sigma}^*\boldsymbol{\beta} - 2\rho\boldsymbol{\beta}, \text{ where } \rho = \mathrm{E}(y\boldsymbol{x}) \text{ and } \boldsymbol{\Sigma}^* = \mathrm{E}(\boldsymbol{x}\boldsymbol{x}^T).$$

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, \boldsymbol{\beta}^T\widehat{\boldsymbol{\Sigma}}\boldsymbol{\beta} - 2(\boldsymbol{y}^T\boldsymbol{X}/n)\boldsymbol{\beta} + \lambda\|\boldsymbol{\beta}\|_1.$$

$$\widehat{\boldsymbol{\beta}}^{\text{ENet}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, \boldsymbol{\beta}^T\widehat{\boldsymbol{\Sigma}}_{\lambda_2}\boldsymbol{\beta} - 2(\boldsymbol{y}^T\boldsymbol{X}/n)\boldsymbol{\beta} + \lambda_1\|\boldsymbol{\beta}\|_1.$$

- Elastic net can be regard to replace $\widehat{\boldsymbol{\Sigma}}$ with a regularized version $\widehat{\boldsymbol{\Sigma}}_{\lambda_2}$.

Computation of elastic net

- Due to its connection with Lasso, it can be transformed into a Lasso problem and use coordinate algorithm to solve.
- R implementation: `glmnet`.

Simulation

Simulation example 1: 50 data sets consisting of 20/20/200 observations and 8 predictors. $\beta = (3, 1.5, 0, 0, 2, 0, 0, 0)$ and $\sigma = 3$. $\text{cor}(\mathbf{x}_i, \mathbf{x}_j) = (0.5)^{|i-j|}$.

Simulation example 2: Same as example 1, except $\beta_j = 0.85$ for all j.

Simulation example 3: 50 data sets consisting of 100/100/400 observations and 40 predictors. $\beta = (\underbrace{0, \ldots, 0}_{10}, \underbrace{2, \ldots, 2}_{10}, \underbrace{0, \ldots, 0}_{10}, \underbrace{2, \ldots, 2}_{10})$ and $\sigma = 15$; $\text{cor}(x_i, x_j) = 0.5$ for all i, j.

Simulation example 4: 50 data sets consisting of 50/50/400 observations and 40 predictors. $\beta = (\underbrace{3, \ldots, 3}_{15}, \underbrace{0, \ldots, 0}_{25})$ and $\sigma = 15$.

$$\begin{aligned}
\mathbf{x}_i &= Z_1 + \epsilon_i^x, \quad && Z_1 \sim N(0,1), \quad && i = 1, \cdots, 5, \\
\mathbf{x}_i &= Z_2 + \epsilon_i^x, \quad && Z_2 \sim N(0,1), \quad && i = 6, \cdots, 10, \\
\mathbf{x}_i &= Z_3 + \epsilon_i^x, \quad && Z_3 \sim N(0,1), \quad && i = 11, \cdots, 15, \\
\mathbf{x}_i &\sim N(0,1), \quad && \mathbf{x}_i \quad \text{i.i.d} \quad && i = 16, \ldots, 40.
\end{aligned}$$

Simulation

Median MSE for the simulated examples

Method	Ex.1	Ex.2	Ex.3	Ex.4
Ridge	4.49 (0.46)	2.84 (0.27)	39.5 (1.80)	64.5 (4.78)
Lasso	3.06 (0.31)	3.87 (0.38)	65.0 (2.82)	46.6 (3.96)
Elastic Net	2.51 (0.29)	3.16 (0.27)	56.6 (1.75)	34.5 (1.64)
No re-scaling	5.70 (0.41)	2.73 (0.23)	41.0 (2.13)	45.9 (3.72)

Variable selection results

Method	Ex.1	Ex.2	Ex.3	Ex.4
Lasso	5	6	24	11
Elastic Net	6	7	27	16

Some other penalized estimators

Dantzig estimator:

- Candes and Tao (2007), AoS, “The Dantzig selector: statistical estimation when p is much larger than n.”
- Bickel et al (2009), AoS, “Simultaneous analysis of Lasso and Dantzig selector”.

MCP estimator:

- Zhang (2010), AoS, “Nearly unbiased variable selection under minimax concave penalty”.

Folded-concave estimator:

- Fan and Lv (2011), IEEE Transactions on Information Theory, “Nonconcave penalized likelihood with NP-dimensionality”.

Variable selection with pre-defined structures

- Sometimes, there are natural group structures among the covariates. How to build the group structure into variable selection?
    - Examples of group structures: pathways and genes.
- Aim: select groups (and possibly its members) that are associated with the outcome.
    - All-in-all-out selection: once one member in a group is selected, all other members will also be selected.
    - Within group selection: if one group is selected, its members are not necessarily all selected. We can further select which members are important.
- Group selection is different from the “grouping effect” of elastic net.
    - “grouping effect” is to simultaneously select variables that are highly correlated with each other. It’s not a group selection.
    - Group selection is to select the groups as a target.

Group Lasso: Yuan and Lin (2006). JRSSB. "Model selection and estimation in regression with grouped variables."
Suppose the p covariates can be partitioned into J groups, with G_j covariates in the j-th group, the group Lasso estimate is defined as

$$\begin{aligned}\widehat{\beta}^{\text{group Lasso}} &= \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \sum_{j=1}^{J}\sum_{l=1}^{G_j} \boldsymbol{X}_{j_l}\beta_{j_l}\|_2^2 + \lambda \sum_{j=1}^{J} (\sum_{l=1}^{G_j} \beta_{j_l}^2)^{1/2} \\ &= \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \sum_{j=1}^{J}\sum_{l=1}^{G_j} \boldsymbol{X}_{j_l}\beta_{j_l}\|_2^2 + \lambda \sum_{j=1}^{J} \|\boldsymbol{\beta}_j\|_2,\end{aligned}$$

where $\boldsymbol{\beta}_j$ is the subvector of $\boldsymbol{\beta}$ with indices in the j-th group.

- Group Lasso penalizes the magnitude of a group (i.e. $\|\boldsymbol{\beta}_j\|_2$). It imposes an L_1 penalty on $\|\boldsymbol{\beta}_j\|_2$. Groups with small magnitude will be shrunken to 0, thus it enables group selection.
- Within a group, the penalty is the sum of the squares of coefficients. Therefore, it cannot conduct within-group selections.
- In other words, it's an "all-in-all-out" selector. Meaning that either all elements within a group will be selected or none of them will be selected.
- The group Lasso problem is a convex optimization problem. Thus, it has a global minimizer.
- We can use blockwise gradient descent algorithm to solve group Lasso problem. It's in the same spirit of coordinate gradient descent. Each iteration has a closed-form solution given by block soft-thresholding.
- Group Lasso is group selection consistent given certain regularity conditions and a proper choice of λ.

Group bridge: Huang et al. (2009). Biometrika. “A group bridge approach for variable selection”.
Let $A_1, \ldots, A_J$ be subsets of $\{1, \ldots, p\}$ representing known groups of the covariates. Denote $\boldsymbol{\beta}_{A_j} = (\beta_k, k \in A_j)^T$ the vector of coefficients in the j-th group.

$$\begin{aligned}\widehat{\boldsymbol{\beta}}^{\text{group Bridge}} &= \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \sum_{k=1}^{p} \boldsymbol{X}_k \beta_k\|_2^2 + \lambda_n \sum_{j=1}^{J} c_j \Big(\sum_{j_l \in A_j} |\beta_{j_l}|\Big)^{\gamma} \\ &= \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \sum_{k=1}^{p} \boldsymbol{X}_k \beta_k\|_2^2 + \lambda_n \sum_{j=1}^{J} c_j \|\boldsymbol{\beta}_{A_j}\|_1^{\gamma},\end{aligned} \tag{1.24}$$

where $0 < \gamma < 1$, c_j is the weight for the j-th group. A simple choice is $c_j \approx |A_j|_0^{1-\gamma}$, where $|A_j|_0$ is the cardinality of A_j.

- Group bridge problem is not a convex optimization problem. Therefore, it can results in multiple local minimizers.
- Group bridge penalty is imposed on $\|\beta_{A_j}\|_1$. Due to the singularity of x^γ for $0 < \gamma < 1$. It enables group selection. Besides that, since the L_1 penalty is also singular, it also enables *within group* selection. This is the difference between group bridge and group Lasso.
- For asymptotic properties of group bridge, read Huang et al. (2009).

Computation of group bridge

Define

$$S_{1n}(\beta, \boldsymbol{\theta}) = \|\boldsymbol{y} - \sum_{j=1}^{p} \boldsymbol{X}_j \beta_j\|_2^2 + \sum_{j=1}^{J} \theta_j^{1-1/\gamma} c_j^{1/\gamma} \|\boldsymbol{\beta}_{A_j}\|_1 + \tau \sum_{j=1}^{J} \theta_j,$$

where τ is a penalty parameter.

Proposition 1 (Huang et al. (2009))

Suppose $0 < \gamma < 1$. If $\lambda_n = \tau^{1-\gamma}\gamma^{-\gamma}(1-\gamma)^{\gamma-1}$, the $\widehat{\beta}_n$ solves (1.24) if and only if $(\widehat{\beta}_n, \widehat{\boldsymbol{\theta}})$ solves

$$\min S_{1n}(\beta, \boldsymbol{\theta}) \text{ subject to } \boldsymbol{\theta} \geq 0,$$

for some $\widehat{\boldsymbol{\theta}} \geq 0$, where $\boldsymbol{\theta} \geq 0$ means $\theta_j \geq 0, j = 1, \ldots, J$.

- By Proposition 1, we can iteratively fix one component of $(\boldsymbol{\beta}_n, \boldsymbol{\theta})$ and solve for the other, i.e. $\widehat{\boldsymbol{\beta}}^{(0)} \to \widehat{\theta}^{(1)} \to \widehat{\boldsymbol{\beta}}^{(1)} \to \cdots$.
- The original nonconvex problem is decomposed into two simpler problems.
- The iteration guarantees to converge to a local minimizer as the objective function always decreases.

Fused Lasso: Tibshirani et. al (2005). JRSSB. "Sparsity and smoothness via the fused lasso".
One drawback of the Lasso is that it ignores ordering of the features. For this purpose, we consider the *fused lasso* defined by

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{subject to } \sum_{j=1}^{p} |\beta_j| \le s_1 \text{ and } \sum_{j=2}^{p} |\beta_j - \beta_{j-1}| \le s_2.$$

- The first constraint encourages sparsity in the coefficients.
- The second constraint encourages sparsity in their differences, i.e. flatness of the coefficient profiles β_j as a function of j.

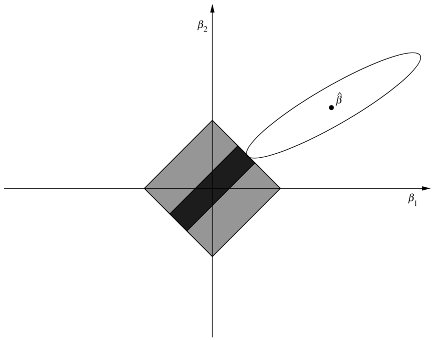


Fig. 2. Schematic diagram of the fused lasso, for the case $N > p = 2$: we seek the first time that the contours of the sum-of-squares loss function (⬭) satisfy $\Sigma_j |\beta_j| = s_1$ (◆) and $\Sigma_j |\beta_j - \beta_{j-1}| = s_2$ (⬋)

A simulated example: $y_i = \sum_j x_{ij}\beta_j + \epsilon_i$, where x_{ij} are standard normal, $\epsilon_i \sim N(0, \sigma^2)$ with $\sigma = 0.75$.

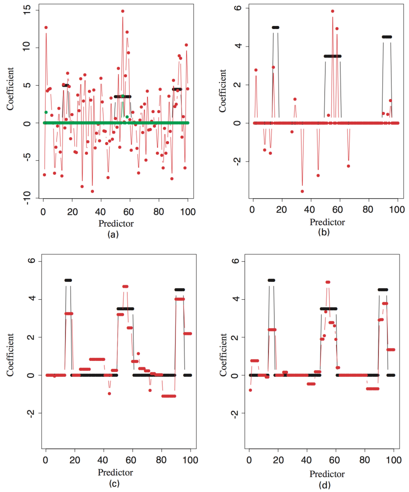


Fig. 3. Simulated example, with $p = 100$ predictors having coefficients shown by the black lines: (a) univariate regression coefficients (red) and a soft thresholded version of them (green); (b) lasso solution (red), using $s_1 = 35.6$ and $s_2 = \infty$; (c) fusion estimate, using $s_1 = \infty$ and $s_2 = 26$ (these values of s_1 and s_2 minimized the estimated test set error); (d) the fused lasso, using $s_1 = \Sigma_j |\beta_j|$ and $s_2 = \Sigma_j |\beta_j - \beta_{j-1}|$, where β is the true set of coefficients

Computation of fused Lasso: convert the original problem into quadratic programming with linear constraint, which is a standard optimization problem and can be solved by some optimization software (e.g `cvx`, `SQOPT`).

Let $\beta_j = \beta_j^+ - \beta_j^-$ with $\beta_j^+, \beta_j^- \geq 0$. Define $\theta_j = \beta_j - \beta_{j-1}$ for $j > 1$ and $\theta_1 = \beta_1$. Let $\theta_j = \theta_j^+ - \theta_j^-$ with $\theta_j^+, \theta_j^- \geq 0$. Let L be $p \times p$ matrix with $L_{ii} = 1$, $L_{i+1,i} = -1$ and $L_{ij} = 0$ otherwise so that $\theta = L\beta$. Let e be a column p-vector of 1s, and I be $p \times p$ identity matrix.

We re-write fused Lasso problem as

$$\widehat{\beta} = \operatorname{argmin}(y - X\beta)^T(y - X\beta)$$

subject to

$$\begin{pmatrix} -a_0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \leq \begin{pmatrix} L & 0 & 0 & -I & I \\ I & -I & I & 0 & 0 \\ 0 & e^T & e^T & 0 & 0 \\ 0 & 0 & 0 & e^T & e^T \end{pmatrix} \begin{pmatrix} \beta \\ \beta^+ \\ \beta^- \\ \theta^+ \\ \theta^- \end{pmatrix} \leq \begin{pmatrix} a_0 \\ 0 \\ s_1 \\ s_2 \end{pmatrix},$$

and $\beta^+, \beta^-, \theta^+, \theta^- \geq 0$.

- Fused Lasso problem can also be solved by coordinate gradient descent algorithm with some modifications.
- What if we replace the constraint $\sum_{j=2}^{p} |\beta_j - \beta_{j-1}| \leq s_2$ with $\sum_{j=2}^{p} (\beta_j - \beta_{j-1})^2 \leq s_2$?
- If we have prior information on how the predictors are connected with each other (e.g. a network structure), we can build that prior information into the fused Lasso penalty. For example,

$$\sum_{j} \sum_{k \sim j} w(j,k) |\beta_j - \beta_k| \leq s_2,$$

  where $k \sim j$ means k is connected with j, and $w(j,k)$ is the weight.
- For more details, see Li and Li (2012). AoAS. “Variable Selection and Regression Analysis for Graph-structured Covariates with an Application to Genomics”.

Regularization of Cox's proportional hazards model

- Bradic, J., Fan, J., & Jiang, J. (2011). Regularization for Cox's proportional hazards model with NP-dimensionality. Annals of Statistics, 39(6), 3092-3120.

Model setup:

- Consider i.i.d. sample $\{(\boldsymbol{X}_i, T_i)\}_{i=1}^n$ form the population $(\boldsymbol{X}, \boldsymbol{T})$, where $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{ip})^T$ is a column vector of covariates, T_i is the survival time which are not fully observable.
- Consider right censoring scheme: for certain subjects, we only know their survival time is beyond certain time point, but not know the exact survival time.
- Let $\{C_i\}_{i=1}^n$ be the censoring time, which are assumed to be conditionally independent of survival times given covariates $\{\boldsymbol{X}_i\}_{i=1}^n$.
- We observe the i.i.d sample $\{(\boldsymbol{X}_i, Z_i, \delta_i)\}_{i=1}^n$, where $Z_i = \min(T_i, C_i)$ and $\delta_i = I(T_i \leq C_i)$.

Cox's proportional hazards model

- Define the survival function $S(t) = P(T > t)$, where t is a fixed time point. $S(t)$ is the probability that the survival will be beyond time t.
- Define the hazard function

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(t < T \leq t + \Delta t | T > t)}{\Delta t} = \frac{-S'(t)}{S(t)}.$$

- The Cox's proportional hazards model assumes that

$$\lambda(t|\boldsymbol{X}) = \lambda_0(t) \exp(\boldsymbol{\beta}^T \boldsymbol{X}),$$

where the baseline hazard rate $\lambda_0(t)$ is a nuisance function.

When $p \gg n$, in order to estimate $\boldsymbol{\beta}$ in the Cox model, we consider the following penalized partial likelihood.

$$\boldsymbol{\beta} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \; -\frac{1}{n} Q_n(\boldsymbol{\beta}) + \sum_{j=1}^{p} p_{\lambda_n}(|\beta_j|),$$

where p_{λ_n} is a penalty function and λ_n is the tuning parameter. $Q_n(\boldsymbol{\beta}) = \sum_{j=1}^{N} \{\boldsymbol{\beta}^T \boldsymbol{X}_j - \log(\sum_{i \in \mathcal{R}_j} \exp(\boldsymbol{\beta}^T \boldsymbol{X}_i))\}$, where $\boldsymbol{X}_j$ is the jth column of $\boldsymbol{X}$ and $\mathcal{R}_j = \{i \in \{1, \ldots, n\} : Z_i \geq t_j\}$ denotes the risk set at time t_j.

- $-Q_n(\boldsymbol{\beta})$ is a convex function of $\boldsymbol{\beta}$.
- For $p_{\lambda_n}(\boldsymbol{\beta})$, we can choose L_1, SCAD, etc.
- The problem can be efficiently solved by gradient descent.
- Variable selection properties can be established. Key to proof: KKT conditions + concentration inequality. See more details in Bradic et al. (2011).

A general penalized estimator

$$\widehat{\boldsymbol{\beta}} = \operatorname*{argmin}_{\boldsymbol{\beta}\in\mathbb{R}^p} \ell(\mathbf{Z},\boldsymbol{\beta}) + \sum_{j=1}^{p} p_{j,\lambda_n}(\beta_j), \tag{1.25}$$

where $\ell(\mathbf{Z},\boldsymbol{\beta})$ is a general loss function, $\boldsymbol{Z}$ is the data and $\boldsymbol{\beta}$ is the parameter of interest, $p_{j,\lambda_n}(\cdot)$ is the penalty function. In particular $p_{j,\lambda_n}(\cdot)$ can be different for different covariates.

- The solution of optimization problem (1.25) can usually be done by gradient descent algorithm, or alternating direction method of multipliers (ADMM). [A good source to learn: Dr. Stephen Boyd's book, Convex optimization, and his course website, http://web.stanford.edu/class/ee364a/courseinfo.html]

- Theoretical properties
  - Model selection consistency: the proof can be done via concentration inequality + KKT conditions
  - Estimation consistency: $\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2 = o_P(1)$, $\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_1 = o_P(1)$. [See Negahban, S. N., Ravikumar, P., Wainwright, M. J., & Yu, B. (2012). A Unified Framework for High-Dimensional Analysis of *M*-Estimators with Decomposable Regularizers. Statistical Science, 27(4), 538-557.]
- Many problems can be written in the form of (1.25).
  - Generalized linear regression, generalized linear mixed model, Cox's proportional hazards model, generalized additive model, matrix completion problem, discriminant analysis, support vector machine, etc.

Matrix Estimation

- The estimation of covariance matrix and its inverse (precision matrix) plays a key role in many statistical problems (e.g linear discriminant analysis, factor analysis, principal component analysis)
- Suppose $\boldsymbol{x} \in \mathbb{R}^p$ is a random vector with $\mathrm{E}(\boldsymbol{x}) = \mathbf{0}$ and $\mathrm{Var}(\boldsymbol{x}) = \boldsymbol{\Sigma}$. $\{\boldsymbol{X}_i\}_{i=1}^n$ are i.i.d samples drawn from the distribution of $\boldsymbol{x}$.
- Denote the sample covariance matrix

$$\boldsymbol{S} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{X}_i - \bar{\boldsymbol{X}})(\boldsymbol{X}_i - \bar{\boldsymbol{X}})^T$$

- When p is fixed and $n \to \infty$, $\boldsymbol{S}$ is a consistent estimator of $\boldsymbol{\Sigma}$ and $\boldsymbol{S}^{-1}$ is a consistent estimator of $\boldsymbol{\Sigma}^{-1}$, in terms for various matrix norms (to be shown).
- When p grows with n, $\boldsymbol{S}$ is no longer a good estimate of $\boldsymbol{\Sigma}$. In addition, when $p > n$, $\boldsymbol{S}$ will not be invertible.
- Need to develop some regularized estimator.

Basic facts of matrix: Let $\boldsymbol{A} \in \mathbb{R}^{p \times p}$.
Induced norms: let $\boldsymbol{x} \in \mathbb{R}^p$. The matrix q-norm is defined as

$$||\boldsymbol{A}||_q = \sup_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{||\boldsymbol{A}\boldsymbol{x}||_q}{||\boldsymbol{x}||_q}.$$

Then, we have

- $\|\boldsymbol{A}\|_1 = \max_{1 \le j \le p} \sum_{i=1}^p |a_{ij}|$, where a_{ij} is the (i,j)-th element of $\boldsymbol{A}$. ($\|\boldsymbol{A}\|_1$ is the maximal column sums of $\boldsymbol{A}$.)
- $\|\boldsymbol{A}\|_\infty = \max_{1 \le i \le p} \sum_{j=1}^p |a_{ij}|$. ($\|\boldsymbol{A}\|_\infty$ is the maximal row sums of $\boldsymbol{A}$).
- When $\boldsymbol{A}$ is symmetric, $\|\boldsymbol{A}\|_1 = \|\boldsymbol{A}\|_\infty$.
- $\|\boldsymbol{A}\|_2 = \{\lambda_{\max}(\boldsymbol{A}^T\boldsymbol{A})\}^{1/2}$, where $\lambda_{\max}(\boldsymbol{A}^T\boldsymbol{A})$ is the maximal eigenvalue of $\boldsymbol{A}^T\boldsymbol{A}$.

- Frobenius norm: it is the Euclidean norm of $\text{vec}(\boldsymbol{A})$, where $\text{vec}(\boldsymbol{A})$ transforms $\boldsymbol{A}$ into a p^2 dimensional vector.

$$\|\boldsymbol{A}\|_F = \left(\sum_{i=1}^{p}\sum_{j=1}^{p} a_{ij}^2\right)^{1/2} = \left\{\text{tr}(\boldsymbol{A}^T\boldsymbol{A})\right\}^{1/2}$$

- Elementwise max norm: $||\boldsymbol{A}||_{\max} = \max_{1\le i,j\le p} |a_{ij}|$.
- Relationship of various matrix norms (exercise)
    - $\|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_F \le \sqrt{p}\|\boldsymbol{A}\|_2$.
    - $\|\boldsymbol{A}\|_2 \le \{\|\boldsymbol{A}\|_1\|\boldsymbol{A}\|_\infty\}^{1/2}$.
    - $||\boldsymbol{A}||_{\max} \le \|\boldsymbol{A}\|_2 \le p||\boldsymbol{A}||_{\max}$.

Consistency of $\boldsymbol{S}$ when p is fixed. For simplicity we assume $\boldsymbol{X} \sim N_p(\boldsymbol{0}, \boldsymbol{\Sigma})$.

- We can show that $\widehat{\sigma}_{ij} - \sigma_{ij} = O_P(n^{-1/2})$, where $\widehat{\sigma}_{ij}$ the (i,j)-th element of $\boldsymbol{S}$ and σ_{ij} is the (i,j)-th element of $\boldsymbol{\Sigma}$.
    - For a sequence of random variables Z_n, $Z_n = O_P(a_n)$ means for any $\epsilon > 0$ there is a constant $M_\epsilon > 0$ such that $P(|Z_n| > M_\epsilon a_n) < \epsilon$.
- By definition, $\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_1 = \max_i \sum_{j=1}^{p} |\widehat{\sigma}_{ij} - \sigma_{ij}| = O_P(p^2 n^{-1/2}) = O_P(n^{-1/2})$, when p is fixed.
- $\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_2 \leq \|\boldsymbol{S} - \boldsymbol{\Sigma}\|_1 = O_P(n^{-1/2})$.
- $\boldsymbol{S}$ is also consistent to $\boldsymbol{\Sigma}$ in Frobenius norm and elementwise max norm.
- However, when $p^2/n^{1/2} \to \infty$, the upper bound no longer converges to 0. In this case, $\boldsymbol{S}$ is no longer a consistent estimator of $\boldsymbol{\Sigma}$.
- One observation: when estimating each element in $\boldsymbol{\Sigma}$, there is an error of $O_P(n^{-1/2})$. If there are many elements in $\boldsymbol{\Sigma}$ (i.e. p is very large), these elementwise estimation errors can be built up and finally ruin the consistency.

Marchenko-Pastur Law

Suppose $\{\boldsymbol{X}_i\}_{i=1}^n$ are i.i.d samples with $\mathrm{E}(\boldsymbol{X}_i) = \boldsymbol{0}$ and $\mathrm{Var}(\boldsymbol{X}_i) = \boldsymbol{I}_p$. If $p/n \to c \in (0, 1)$ as $n \to \infty$, the empirical distribution of the eigenvalues of $\boldsymbol{S}$ converges to a distribution with density

$$\frac{1}{2c\pi x}\sqrt{(b-x)(x-a)}I(a \le x \le b),$$

where $a = (1-\sqrt{c})^2$, $b = (1+\sqrt{c})^2$.

- We expect the eigenvalues to be 1. But it is spread out between $(1-\sqrt{c})^2$ and $(1+\sqrt{c})^2$.
- In this situation, $\boldsymbol{S}$ is no longer a good estimate of $\boldsymbol{\Sigma}$.

Regularized Covariance Matrix Estimator
Bickel and Levina (2008). AoS. Covariance regularization by thresholding.
A class of sparse covariance matrices

$$\mathcal{U}(q, c_0(p)) = \{\mathbf{\Sigma} : \max_{1 \le i \le p} \sum_{j=1}^{p} |\sigma_{ij}|^q \le c_0(p)\} \text{ for } 0 \le q < 1.$$

When $q = 0$,

$$\mathcal{U}(q, c_0(p)) = \{\mathbf{\Sigma} : \max_{1 \le i \le p} \sum_{j=1}^{p} I(\sigma_{ij} \neq 0) \le c_0(p)\}.$$

In other words, for each row, there are at most $c_0(p)$ nonzero elements.

Define the hard-thresholding operator on a matrix $\boldsymbol{A}$ by

$$T_s(\boldsymbol{A}) = [a_{ij} I(|a_{ij}| \geq s)].$$

Suppose $\{\boldsymbol{X}_i\}_{i=1}^n$ are i.i.d normal with $\mathrm{E}(\boldsymbol{X}_i) = \boldsymbol{0}$ and $\mathrm{Var}(\boldsymbol{X}_i) = \boldsymbol{\Sigma}$ and $\max_{1 \leq j \leq p} \sigma_{jj} \leq M$.
Theorem 1: Uniformly on $\mathcal{U}(q, c_0(p))$, if

$$t_n = M' \sqrt{\frac{\log p}{n}},$$

and $\log p = o(n)$, then

$$\|T_{t_n}(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_2 = O_P\left(c_0(p)\{(\log p)/n\}^{(1-q)/2}\right).$$

If in addition, $\lambda_{\min}(\boldsymbol{\Sigma}) \geq \epsilon_0 > 0$,

$$\|(T_{t_n}(\boldsymbol{S}))^{-1} - \boldsymbol{\Sigma}^{-1}\|_2 = O_P\left(c_0(p)\{(\log p)/n\}^{(1-q)/2}\right).$$

- Theorem 1 says the thresholding estimator $T_{t_n}(\boldsymbol{S})$ can still be consistent to $\boldsymbol{\Sigma}$ if $c_0(p)\{(\log p)/n\}^{(1-q)/2} \to 0$.
- This requires that $c_0(p)$ cannot be too large, i.e. $\boldsymbol{\Sigma}$ needs to be sparse (many of its elements should be small or exactly equal to 0).
- The key of the proof is first obtain that

$$\max_{i,j} |\widehat{\sigma}_{ij} - \sigma_{ij}| = O_P\left(\{(\log p)/n\}^{1/2}\right). \tag{2.26}$$

- Then use $\|T_{t_n}(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_2 \leq \|T_{t_n}(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_1$.
- Their proof essentially shows that

$$\|T_{t_n}(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_1 = O_P\left(c_0(p)\{(\log p)/n\}^{(1-q)/2}\right).$$

Theorem 2: Under the same conditions as in Theorem 1, if $t = M'\{(\log p)/n\}^{1/2}$ and M' is sufficiently large,

$$\frac{1}{p}\|T_t(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_F^2 = O_P\left(c_0(p)\{(\log p)/n\}^{1-q/2}\right).$$

The non-Gaussian case

- Exponential tail: For some $\eta > 0$,

$$\mathrm{E}e^{tX_j^2} \leq K < \infty \text{ for all } |t| \leq \eta, \text{ for all } j.$$

  In this case,

$$\max_{1 \leq i,j \leq p} |\widehat{\sigma}_{ij} - \sigma_{ij}| = O_P\left(\{(\log p)/n\}^{1/2}\right). \qquad (2.27)$$

  (See proof in the paper). Then, Theorem 1 still holds.
- Polynomial tail: For some $\gamma > 0$,

$$\mathrm{E}|X_j|^{2(1+\gamma)} \leq K \text{ for all } j.$$

  In this case

$$\max_{1 \leq i,j \leq p} |\widehat{\sigma}_{ij} - \sigma_{ij}| = O_P\left(\frac{p^{2/(1+\gamma)}}{n^{1/2}}\right). \qquad (2.28)$$

By taking $t_n = Mp^{2/(1+\gamma)}/n^{1/2}$, we have

$$\|T_{t_n}(\boldsymbol{S}) - \boldsymbol{\Sigma}\|_2 = O_P\left(c_0(p)\left(\frac{p^{2/(1+\gamma)}}{n^{1/2}}\right)^{1-q}\right).$$

- An observation: we always take the threshold at the "noise level", meaning that if the true $\sigma_{ij} = 0$, how large $\widehat{\sigma}_{ij}$ can be. Those $\widehat{\sigma}_{ij}$'s below "noise level" are more likely to be false positives. In a sparse matrix, a lot of $\widehat{\sigma}_{ij}$'s are false positives. Once we threshold them, the accumulated errors brought by these entries will be deleted so that we can still have consistent estimator of $\boldsymbol{\Sigma}$.

Summary of the concentration result

- The concentration results (2.27) and (2.28) are critical for matrix estimation problem. It often serves as the starting point of theoretical derivations.
- The rate of concentration depends on the tail assumption.

Regularized sample covariance matrix by banding.
Bickel and Levina (2008). AoS. Regularized estimation of large covariance matrices.
Banding the sample covariance matrix: For any matrix $\boldsymbol{A} = [a_{ij}]_{p\times p}$ and any $0 \leq k < p$, define

$$B_k(\boldsymbol{A}) = [a_{ij} I(|i-j| \leq k)].$$

For covariance matrix estimation, we can band the sample covariance matrix $\widehat{\boldsymbol{\Sigma}}_{k,p} = B_k(\boldsymbol{S})$.

- This method is ideal for $\boldsymbol{\Sigma} = [\sigma_{ij}]$ that

$$|i-j| > k \Rightarrow \sigma_{ij} = 0.$$

  i.e $\boldsymbol{\Sigma}$ is a banded matrix.

A class of sparse positive definite symmetric matrices

$$\mathcal{U}(\varepsilon_0, \alpha, C) = \left\{ \mathbf{\Sigma} : \max_{1 \le j \le p} \sum_{i=1}^{p} \{|\sigma_{ij}| : |i-j| > k\} \le C k^{-\alpha} \text{ for all } k > 0, \right.$$
$$\left. \text{and } 0 < \varepsilon_0 \le \lambda_{\min}(\mathbf{\Sigma}) \le \lambda_{\max}(\mathbf{\Sigma}) \le 1/\varepsilon_0 \right\}.$$

- A special case: If $\mathbf{\Sigma}$ is a k band matrix, then $\mathbf{\Sigma} \in \mathcal{U}(\varepsilon_0, \alpha, C)$.

Theorem 1. Suppose that $\boldsymbol{X}$ is Gaussian and $\mathcal{U}(\varepsilon_0, \alpha, C)$ is the class of covariance matrices defined above, Then, if $\kappa_n \asymp (n^{-1}\log p)^{-1/(2(\alpha+1))}$,

$$\|\widehat{\boldsymbol{\Sigma}}_{k_n,p} - \boldsymbol{\Sigma}\|_2 = O_P\left(\{(\log p)/n\}^{\alpha/(2(\alpha+1))}\right) = \|\widehat{\boldsymbol{\Sigma}}_{k_n,p}^{-1} - \boldsymbol{\Sigma}^{-1}\|_2,$$

uniformly on $\boldsymbol{\Sigma} \in \mathcal{U}$.

- Here, $a_n \asymp b_n$ means $a_n = O(b_n)$ and $b_n = O(a_n)$.
- The proof follows a similar argument as in the proof for the thresholding estimator.
- First, obtain the bound for $\max_{ij} |\widehat{\sigma}_{ij} - \sigma_{ij}|$.
- Then, use $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_1$.

Summary of thresholding and banding estimator

- The key assumption is that $\boldsymbol{\Sigma}$ is sparse.
- The key reason why they are consistent is that they threshold small elements of $\boldsymbol{S}$ where it's likely that in these positions $\sigma_{ij} = 0$ or very small. By doing so, in these places, the estimation error is 0 or very small. Hence, such errors won't accumulate to ruin the whole estimation of $\boldsymbol{\Sigma}$.

What if $\boldsymbol{\Sigma}$ is not sparse? In many real applications, $\boldsymbol{\Sigma}$ can be dense, meaning that most of its elements are not small. In this case, how do we estimate $\boldsymbol{\Sigma}$?

- If no assumption is made on $\boldsymbol{\Sigma}$, it's hard to find a consistent estimator to $\boldsymbol{\Sigma}$.
- If we know some information of $\boldsymbol{\Sigma}$, we can still consistently estimate $\boldsymbol{\Sigma}$.
- For example, if $\boldsymbol{\Sigma}$ can be decomposed as a low rank matrix + a sparse matrix.

$$\boldsymbol{\Sigma} = \boldsymbol{\Theta} + \boldsymbol{U},$$

  where $\mathrm{rank}(\boldsymbol{\Theta}) \leq r$ and $\boldsymbol{U}$ is a sparse matrix, consistent estimators of $\boldsymbol{\Sigma}$ still exist.
- Some reference
  - Agarwal, Negahban and Wainwright (2011). AoS. Noisy matrix decomposition via convex relaxation: Optimal rates in high dimension.
  - Chandrasekaran, et al. (2011) SIAM Journal of Optimization. Rank-sparsity incoherence for matrix decomposition.

Estimation of Precision matrix

- Graphical Lasso estimator (glasso)
- Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3), 432-441.

The graphical lasso minimizes a L_1-regularized negative log-likelihood from Wishart distribution

$$\widehat{\boldsymbol{\Theta}}^{\text{glasso}} = \underset{\boldsymbol{\Theta} \succ 0}{\operatorname{argmin}} - \log\det(\boldsymbol{\Theta}) + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \lambda \|\boldsymbol{\Theta}\|_{\text{one}}, \tag{2.29}$$

where $\boldsymbol{\Theta} \succ 0$ means $\boldsymbol{\Theta}$ is positive definite, $\det(\boldsymbol{\Theta})$ stands for the determinant of $\boldsymbol{\Theta}$, $\boldsymbol{S}$ is the sample covariance matrix, $\|\boldsymbol{\Theta}\|_{\text{one}} := \sum_{i=1}^{p} \sum_{j=1}^{p} |\theta_{ij}|$.

Computation

The KKT condition of problem (2.29) is

$$-\boldsymbol{\Theta}^{-1} + \boldsymbol{S} + \lambda\boldsymbol{\Gamma} = \mathbf{0}, \tag{2.30}$$

where $\boldsymbol{\Gamma}$ is a matrix of component-wise signs of $\boldsymbol{\Theta}$:

$$\gamma_{ij} = \text{sign}(\theta_{ij}), \text{ if } \theta_{ij} \neq 0;$$
$$\gamma_{ij} \in [-1, 1], \text{ if } \theta_{ij} = 0.$$

We can transform (2.30) into a Lasso problem and solve it by gradient descent algorithm.

Consider a partition of $\mathbf{\Theta}$ and $\mathbf{\Gamma}$:

$$\mathbf{\Theta} = \begin{pmatrix} \mathbf{\Theta}_{11} & \boldsymbol{\theta}_{12} \\ \boldsymbol{\theta}_{21} & \theta_{22} \end{pmatrix}, \qquad \mathbf{\Gamma} = \begin{pmatrix} \mathbf{\Gamma}_{11} & \boldsymbol{\gamma}_{12} \\ \boldsymbol{\gamma}_{21} & \gamma_{22} \end{pmatrix}$$

where $\mathbf{\Theta}_{11}$ is $(p-1)\times(p-1)$, $\boldsymbol{\theta}_{12}$ is $(p-1)\times 1$ and θ_{22} is scalar. Let $\boldsymbol{W} := \mathbf{\Theta}^{-1}$ and $\boldsymbol{S}$ be partitioned in the same way. Using properties of the inverse of block matrices, we can show that $\boldsymbol{W}$ can be written as

$$\begin{pmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} (\boldsymbol{\Theta}_{11} - \frac{\boldsymbol{\theta}_{12}\boldsymbol{\theta}_{21}}{\theta_{22}})^{-1} & -\boldsymbol{W}_{11}\frac{\boldsymbol{\theta}_{12}}{\theta_{22}} \\ \cdot & \frac{1}{\theta_{22}} - \frac{\boldsymbol{\theta}_{21}\boldsymbol{W}_{11}\boldsymbol{\theta}_{12}}{\theta_{22}^2} \end{pmatrix}. \tag{2.31}$$

We can solve (2.30) column by column, holding the rest fixed. Considering the *p*th column of (2.30), we get

$$-\boldsymbol{w}_{12} + \boldsymbol{s}_{12} + \lambda\boldsymbol{\gamma}_{12} = \boldsymbol{0}. \tag{2.32}$$

Reading off $\boldsymbol{w}_{12}$ from (2.31), we have

$$\boldsymbol{w}_{12} = -\boldsymbol{W}_{11}\boldsymbol{\theta}_{12}/\theta_{22} \tag{2.33}$$

and plug it into (2.32), we have

$$\boldsymbol{W}_{11}\frac{\boldsymbol{\theta}_{12}}{\theta_{22}} + \boldsymbol{s}_{12} + \lambda\boldsymbol{\gamma}_{12} = \boldsymbol{0} \tag{2.34}$$

We can solve (2.34) for $\boldsymbol{\beta} := \boldsymbol{\theta}_{12}/\theta_{22}$, that is

$$\boldsymbol{W}_{11}\boldsymbol{\beta} + \boldsymbol{s}_{12} + \lambda\boldsymbol{\gamma}_{12} = \boldsymbol{0}, \tag{2.35}$$

where $\boldsymbol{\gamma}_{12} = \text{sign}(\boldsymbol{\beta})$, since $\theta_{22} > 0$.
Notice that, (2.35) is the KKT condition for the following regularized quadratic problem

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{p-1}}{\text{argmin}}\ \frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{W}_{11}\boldsymbol{\beta} + \boldsymbol{\beta}^T\boldsymbol{s}_{12} + \lambda\|\boldsymbol{\beta}\|_1, \tag{2.36}$$

where $\boldsymbol{W}_{11} \succ 0$ and is assumed to be fixed. This is analogous to a lasso regression problem of the last variable on the rest, except the cross-product matrix $\boldsymbol{S}_{11}$ is replaced by its current estimate $\boldsymbol{W}_{11}$. It can be efficiently solved using elementwise gradient descent, exploiting the sparsity in $\boldsymbol{\beta}$.

From $\widehat{\boldsymbol{\beta}}$, it is easy to obtain $\widehat{\boldsymbol{w}}_{12}$ from (2.33). Using the lower-right element of (2.31), $\widehat{\theta}_{22}$ is obtained by

$$\frac{1}{\widehat{\theta}_{22}} = w_{22} - \widehat{\boldsymbol{\beta}}^T \widehat{\boldsymbol{w}}_{12}. \tag{2.37}$$

Finally, $\widehat{\boldsymbol{\theta}}_{12}$ can now be recovered from $\widehat{\boldsymbol{\beta}}$ and $\widehat{\theta}_{22}$.
The final piece of puzzle, we need to show that $\boldsymbol{W}_{11} \succ 0$ along iterations. (See Lemma 3 of Mazumder, R., & Hastie, T. (2012). The graphical lasso: New insights and alternatives. Electronic journal of statistics, 6, 2125.)

Graphical Lasso Algorithm:

- Initialize $\boldsymbol{W} = \boldsymbol{S} + \lambda \boldsymbol{I}$.
- Cycle around the columns repeatedly until convergence:
  - Rearrange the columns so that the target column is last (implicitly).
  - Solve the Lasso problem (2.36).
  - Update the column of the covariance using $\widehat{\boldsymbol{w}}_{12}$ in (2.33).
  - Save $\widehat{\beta}$ for this column in the matrix $\boldsymbol{B}$.
- Finally, for every column, compute the diagonal entries $\widehat{\theta}_{jj}$ using (2.37), and convert the $\boldsymbol{B}$ matrix to $\boldsymbol{\Theta}$.

R package: `glasso`.
Some other methods of computing graphical Lasso estimator:
Mazumder, R., & Hastie, T. (2012). The graphical lasso: New insights and alternatives. Electronic journal of statistics, 6, 2125.

Theoretical properties of graphical Lasso

- Ravikumar, P., Wainwright, M. J., Raskutti, G., & Yu, B. (2011). High-dimensional covariance estimation by minimizing L_1-penalized log-determinant divergence. Electronic Journal of Statistics, 5, 935-980.
  - $\|\widehat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}^*\|_\infty = o_P(1)$.
  - Model selection consistency: asymptotically $\text{sign}(\widehat{\Theta}_{ij}) = \text{sign}(\Theta^*_{ij})$.

Estimation of Precision matrix

- Constrained L_1-minimization for Inverse Matrix Estimation (CLIME): Cai, Liu and Luo. (2011). JASA. A constrained L_1 minimization approach to sparse precision matrix estimation.

For a matrix $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, define $\|\boldsymbol{A}\|_{\text{one}} = \sum_{i=1}^{p}\sum_{j=1}^{p}|a_{ij}|$. Define CLIME estimator as

$$\begin{aligned} \widehat{\boldsymbol{\Omega}}_1 &= \operatorname{argmin} \|\boldsymbol{\Omega}\|_{\text{one}} \\ \text{subject to } & \|\boldsymbol{S\Omega} - \boldsymbol{I}\|_{\max} \le \lambda_n \end{aligned} \tag{2.38}$$

The solution $\widehat{\boldsymbol{\Omega}}_1$ is not symmetric. The final CLIME estimator is obtained by symmetrizing $\widehat{\boldsymbol{\Omega}}_1$. Write $\widehat{\boldsymbol{\Omega}}_1 = (\widehat{\boldsymbol{\omega}}_1^1, \ldots, \widehat{\boldsymbol{\omega}}_p^1)$. The CLIME estimator $\widehat{\boldsymbol{\Omega}}$ of $\boldsymbol{\Omega}_0$ is defined as

$$\widehat{\boldsymbol{\Omega}}^{\text{CLIME}} = (\widehat{\omega}_{ij}), \text{ where } \widehat{\omega}_{ij} = \widehat{\omega}_{ji} = \widehat{\omega}_{ij}^1 I(|\widehat{\omega}_{ij}^1| \le |\widehat{\omega}_{ji}^1|) + \widehat{\omega}_{ji}^1 I(|\widehat{\omega}_{ij}^1| > |\widehat{\omega}_{ji}^1|).$$

In problem (2.38), $\widehat{\Omega}_1$ finds the sparest solution such that $\boldsymbol{S}\widehat{\boldsymbol{\Omega}} - \boldsymbol{I}$ is sufficiently close.

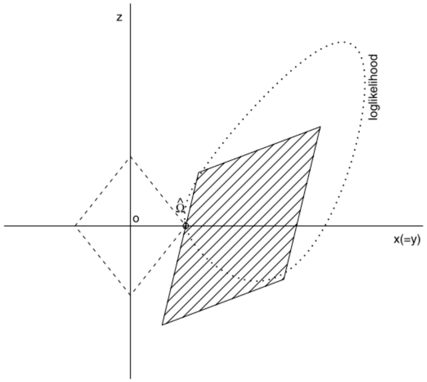


Figure 1. Plot of the elementwise ℓ_∞ constrained feasible set (shaded polygon) and the elementwise ℓ_1 norm objective (dashed diamond near the origin) from CLIME. The log-likelihood function as in Glasso is represented by the dotted line.

The convex problem (2.38) can be further decomposed into p vector minimization problems. Let $\boldsymbol{e}_i \in \mathbb{R}^p$ be a standard unit vector with the ith coordinate being 1 and all other coordinates being 0. For $1 \leq i \leq p$, let $\widehat{\boldsymbol{\beta}}_i$ be the solution of the following convex problem:

$$\min \|\boldsymbol{\beta}\|_1 \text{ subject to } \|\boldsymbol{S}\boldsymbol{\beta} - \boldsymbol{e}_i\|_{\max} \leq \lambda_n. \tag{2.39}$$

Lemma 1. Let $\{\widehat{\boldsymbol{\Omega}}_1\}$ be the solution set of (2.38), and let $\{\widehat{\boldsymbol{B}}\} := \{(\widehat{\boldsymbol{\beta}}_1, \ldots, \widehat{\boldsymbol{\beta}}_p)\}$, where $\widehat{\boldsymbol{\beta}}_i$ are solutions to (2.39) for $i = 1, \ldots, p$. Then, $\{\widehat{\boldsymbol{\Omega}}_1\} = \{\widehat{\boldsymbol{B}}\}$.

Computation

- Problem (2.39) can be transformed into a linear programming problem (see section 5 of the paper).
- R implementation: `fastclime`, `flare`.

Theoretical properties

- (C1) *Exponential-tails:* Suppose that there exist some $0 < \eta < 1/4$ such that $(\log p)/n \leq \eta$ and

$$\mathrm{E} e^{t(X_i - \mu_i)^2} \leq K < \infty \text{ for all } |t| \leq \eta, \text{ for all } i,$$

  where K is a bounded constant.

- (C2) *Polynomial-type tails:* Suppose that for some $\gamma, c_1 > 0$, $p \leq c_1 n^{\gamma}$, and for some $\delta > 0$,

$$\mathrm{E}|X_i - \mu_i|^{4\gamma+4+\delta} \leq K, \text{ for all } i.$$

Class of sparsity precision matrices:

$$\mathcal{U} := \mathcal{U}(q, s_0(p)) = \left\{ \boldsymbol{\Omega} : \boldsymbol{\Omega} > 0, \|\boldsymbol{\Omega}\|_1 \leq M, \max_{1 \leq i \leq p} \sum_{j=1}^{p} |\omega_{ij}|^p \leq s_0(p) \right\},$$

for $0 \leq q < 1$, where $\boldsymbol{\Omega} = (\omega_{ij}) = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_p)$.

Theorem 1. Suppose that $\mathbf{\Omega}_0 \in \mathcal{U}(q, s_0(p))$.
(a) Assume (C1) holds. Let $\lambda_n = C_0 M \sqrt{(\log p)/n}$ and $\tau > 0$. Then,

$$\|\widehat{\mathbf{\Omega}} - \mathbf{\Omega}_0\|_2 \leq C_1 M^{2-2q} s_0(p) \left(\frac{\log p}{n}\right)^{(1-q)/2},$$

with probability greater than $1 - 4p^{-\tau}$.
(b) Assume (C2) holds. Let $\lambda_n = C_2 M \sqrt{(\log p)/n}$. Then

$$\|\widehat{\mathbf{\Omega}} - \mathbf{\Omega}_0\|_2 \leq C_3 M^{2-2q} s_0(p) \left(\frac{\log p}{n}\right)^{(1-q)/2},$$

with probability greater than $1 - O(n^{-\delta/8} + p^{-\tau/2})$.

Theorem 4.
(a) Under the conditions of Theorem 1(a), we have

$$\|\widehat{\boldsymbol{\Omega}} - \boldsymbol{\Omega}_0\|_{\max} \le 4C_0M^2\sqrt{\frac{\log p}{n}},$$
$$\frac{1}{p}\|\widehat{\boldsymbol{\Omega}} - \boldsymbol{\Omega}_0\|_F^2 \le 4C_1M^{4-2q}s_0(p)\left(\frac{\log p}{n}\right)^{1-q/2},$$

with probability greater than $1 - 4p^{-\tau}$.
(b) Under the conditions of Theorem 1(b), we have

$$\|\widehat{\boldsymbol{\Omega}} - \boldsymbol{\Omega}_0\|_{\max} \le 4C_2M^2\sqrt{\frac{\log p}{n}},$$
$$\frac{1}{p}\|\widehat{\boldsymbol{\Omega}} - \boldsymbol{\Omega}_0\|_F^2 \le 4C_3M^{4-2q}s_0(p)\left(\frac{\log p}{n}\right)^{1-q/2},$$

with probability greater than $1 - O(n^{-\delta/8} + p^{-\tau/2})$.

Outline of proof.
The key is to prove that with high probability

$$\max_{ij} |\widehat{\sigma}_{ij} - \sigma_{ij}^0| \leq C_0 \sqrt{(\log p)/n}, \tag{2.40}$$

where $\widehat{\sigma}_{ij}$ is the (i,j)-th element of $\boldsymbol{S}$ and σ_{ij} is the (i,j)-th element of $\boldsymbol{\Sigma} = (\boldsymbol{\Omega})^{-1}$.

- Similar results have been shown to be critical for covariance matrix estimation. (Bickel and Levina 2008a, 2008b).
- The probability essentially depends on the tail assumption.
- These are very standard concentration inequality results.

Theorem 6. Suppose that $\mathbf{\Omega}_0 \in \mathcal{U}(q, s_0(p))$ and $\rho \geq 0$. If $\lambda_n \geq \|\mathbf{\Omega}_0\|_1 (\max_{ij} |\widehat{\sigma}_{ij} - \sigma^0_{ij}|)$, then we have

$$\begin{aligned} \|\widehat{\mathbf{\Omega}} - \mathbf{\Omega}_0\|_{\max} &\leq 4\|\mathbf{\Omega}_0\|_1 \lambda_n, \\ \|\widehat{\mathbf{\Omega}} - \mathbf{\Omega}_0\|_2 &\leq C_4 s_0(p) \lambda_n^{1-q}, \\ \frac{1}{p}\|\widehat{\mathbf{\Omega}} - \mathbf{\Omega}_0\|_F^2 &\leq C_5 s_0(p) \lambda_n^{2-q}, \end{aligned}$$

where C_4 and C_5 are some positive constants.

Gaussian Graphical Model

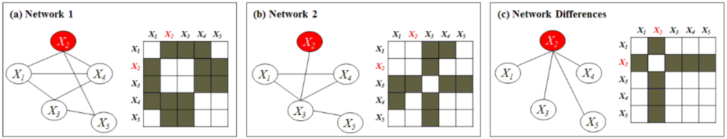
Suppose $X \sim N_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a d-dim random vector following multivariate normal distribution, i.e. its density has the form of

$$f(\boldsymbol{x}) = (2\pi)^{-d/2}(\det\boldsymbol{\Omega})^{1/2}\exp\{-(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Omega}(\boldsymbol{x}-\boldsymbol{\mu})/2\},$$

where $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$.

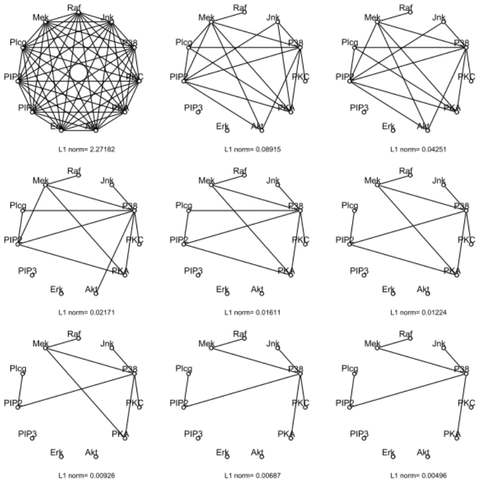
- From theory of multivariate normal random variables, we know that $X_i \perp X_j | \boldsymbol{X}_{-(i,j)}$ if and only if $\omega_{ij} = 0$, where ω_{ij} is the (i,j)-th element of $\boldsymbol{\Omega}$.

- Every multivariate Gaussian distribution can be presented by a *pairwise* Gaussian Markov Random Field (GMRF).
- GMRF can be represented by an undirected graph $G = (V, E)$ with
    - Vertex set: $V = \{1, \ldots, p\}$ corresponding to p elements of $\boldsymbol{X}$.
    - Edge set: $E = \{(i,j) \in V | i \neq j, \omega_{ij} \neq 0\}$, i.e. X_i and X_j are independent conditioning on the other variables.
- There is one-to-one correspondence between $\boldsymbol{\Omega}$ and the graph G.



- The main task of Gaussian graphical model is to infer the underlying network structure using observations $\{\boldsymbol{X}_i\}_{i=1}^n$.
- From the above arguments, we see that it's equivalent to estimate the precision matrix $\boldsymbol{\Omega}$.
- The model selection consistency result in Ravikumar, et al. (2011). implies that the graphical Lasso estimator can asymptotically recover the true network.
- If the dimension d is much larger than n, we need to assume that the graph is sparse, in the sense that there are not too many edges within the graph.
- The applications of Gaussian graphical model: network analysis (e.g. gene regulatory network, social network, citation network).

Friedman, et al. (2008) used the graphical Lasso estimator to analyze a flow cytometry dataset on $p = 11$ proteins and $n = 7466$ cells. Here's the inferred graphs using different penalty parameter.



- Gaussian graphical model assumes that the observation $\boldsymbol{X}_i$ follows a multivariate normal distribution.
- If $\boldsymbol{X}$ follows some other distributions, it leads to other graphical model, e.g Ising model.
- If the distribution is not normal, what's the relationship between conditional independence and the structure of the precision matrix? (*It's still an open problem. There are some partial answers.*)
- Some reference
  - Liu et al. (2012). High dimensional semiparametric Gaussian copula graphical models.
  - Liu et al. (2009). The nonparanormal: Semiparametric estimation of high dimensional undirected graphs.

Linear Methods for Classification

Problem setup: Suppose there are K classes. We have i.i.d samples $\{\boldsymbol{X}_i\}_{i=1}^n$ randomly collected from the K classes with class identifier being available (supervised learning). We'd like to learn a classification rule $G(\cdot)$ based on these observations so that when a new observation x come, we classify it into one of the K classes based on $G(\boldsymbol{x})$.

- $\delta_k(\boldsymbol{x})$: discriminant function.

$$G(\boldsymbol{x}) = k_0, \text{ if } k_0 = \underset{k}{\operatorname{argmax}}\, \delta_k(\boldsymbol{x})$$

  An example of $\delta_k(\boldsymbol{x})$, $\delta_k(\boldsymbol{x}) = \Pr(G = k|X = x)$.
- Decision boundary: $\delta_k(\boldsymbol{x}) = \delta_l(\boldsymbol{x})$.
- Linear methods: $\delta_k(\boldsymbol{x})$ or a monotone transformation of $\delta_k(\boldsymbol{x})$ is a linear function, i.e. $g(\delta_k(\boldsymbol{x})) = \boldsymbol{h}(\boldsymbol{x})^T\boldsymbol{\beta}$.
- We will cover three examples
  - Linear regression using indicator matrix.
  - Linear Discriminant Analysis.
  - Logistic regression.
- We will also see their extension in the high dimension setting.

Linear regression using indicator matrix

- Let $Y(\boldsymbol{x}) = (Y_1, Y_2, \ldots, Y_K)$, where $Y_k = 1$ if $G(\boldsymbol{x}) = k$ and $Y_k = 0$ if $G(\boldsymbol{x}) \neq k$. Indicator response matrix $\boldsymbol{Y}$ is a $N \times K$ matrix formed by the N training data point.
- Example: the response vector $g = (g_1, \ldots, g_N)^T$ and the response matrix

$$g = \begin{pmatrix} 3 \\ 1 \\ 4 \\ \vdots \\ 2 \end{pmatrix}, \qquad Y = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \vdots & & & \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Multi-task linear regression formula

- Let $\boldsymbol{X}$ be the design matrix with $p+1$ columns (including the intercept column). The multi-task linear regression model is

  $$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B} + Error.$$

  The least squares is

  $$\operatorname*{argmin}_{\boldsymbol{B}} \sum_{i=1}^{N} \|\boldsymbol{y}_i - \boldsymbol{B}^T \boldsymbol{x}_i\|_2^2,$$

  where $\boldsymbol{y}_i$, $\boldsymbol{x}_i$ are ith rows of $\boldsymbol{Y}$ and $\boldsymbol{X}$.
- Coefficient estimate: $\widehat{\boldsymbol{B}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$. Estimation of $\boldsymbol{Y}$: $\widehat{\boldsymbol{Y}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$.

- A new observation with input $\boldsymbol{x}$ is classified as follows:
  - Compute the prediction $\widehat{f}(\boldsymbol{x}) = \widehat{\boldsymbol{B}}^T \boldsymbol{x}$.
  - Identify the largest component and classify accordingly:

$$\widehat{G}(\boldsymbol{x}) = \underset{k \in \mathcal{G}}{\operatorname{argmax}} \ \widehat{f}_k(\boldsymbol{x}),$$

  where $\widehat{f}_k(\boldsymbol{x})$ is the kth element of $\widehat{f}(\boldsymbol{x})$.
- The variable Y_k follows binomial distribution $Y_k = 1$ if $G(\boldsymbol{x}) = k$ and 0 otherwise.

$$E(Y_k|X = x) = \Pr(Y_k = 1|X = x) = \Pr(G = k|X = x)$$

  Therefore $E[\widehat{\boldsymbol{Y}}]_{(i,j)} = \Pr(G_i = j|X = x_i)$.
- Correspondingly, it is easy to verify that $\sum_{k \in \mathcal{G}} \widehat{f}_k(\boldsymbol{x}) = 1$. [exercise] Note that, here we assume $\boldsymbol{X}$ includes a column of intercept and $\boldsymbol{x}$ includes an element of 1.

- But there is one problem: $\widehat{f}_k(x)$ can be larger than 1 or negative.
- However, this doesn't affect the prediction. The estimate of this method can be as accurate as some other methods if more basis functions of $\boldsymbol{X}$ are included.
- The extension of linear regression of indicator matrix in high dimension. Use penalized linear regression.
    - Reference: Mai, Q., Zou, H., & Yuan, M. (2012). A direct approach to sparse discriminant analysis in ultra-high dimensions. Biometrika, 29-42.
    - Because of the rigid nature of the regression model, classes can be *masked* by others.

- We project data onto the line joining the three centroids (no information in the orthogonal direction).
- The three regression lines for Y_1, Y_2 and Y_3 are shown in the left panel.
- The line corresponding to Y_2 never dominates. Therefore, observations from class 2 are classified either as class 1 or class 3.
- The right panel uses quadratic regression rather than linear regression. For this example, it solves the problem.
- A lose but general rule is that if $K \geq 3$ classes are lined up, polynomials terms up to degree $K - 1$ might be needed to resolve them.

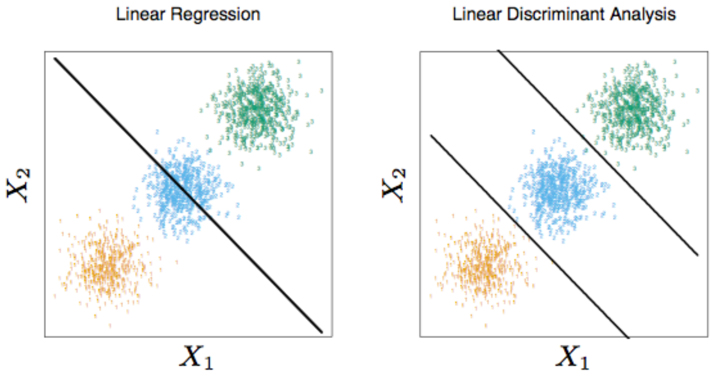


Figure: The data come from three classes in $\mathbb{R}^2$ and are easily separated by linear decision boundaries. The right plot shows the boundaries found by linear discriminant analysis. The left plot shows the boundaries found by linear regression of the indicator response variables. The middle class is completely masked.

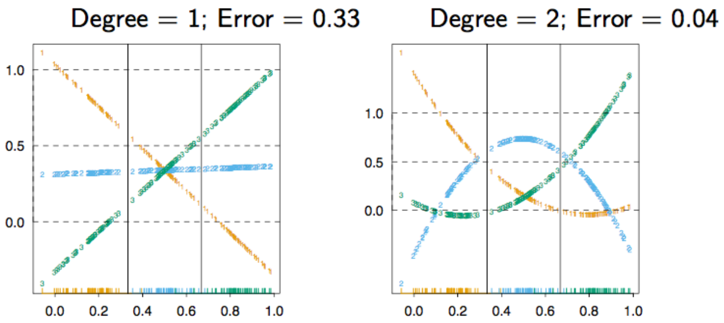


Figure: The rug plot at the base indicates the positions and class membership of each observation. The three curves in each panel are the fitted regressions to the three-class indicator variables. Above each plot is the training error rate. The Bayes error rate is 0.025 for this problem.

Classification via multinomial regression

- Multiple logit model:

$$\log \frac{\Pr(G=1|\boldsymbol{X}=\boldsymbol{x})}{\Pr(G=K|\boldsymbol{X}=\boldsymbol{x})} = \beta_{10} + \boldsymbol{\beta}_1^T\boldsymbol{x},$$

$$\vdots$$

$$\log \frac{\Pr(G=K-1|\boldsymbol{X}=\boldsymbol{x})}{\Pr(G=K|\boldsymbol{X}=\boldsymbol{x})} = \beta_{(K-1)0} + \boldsymbol{\beta}_{K-1}^T\boldsymbol{x},$$

which is equivalent to

$$\Pr(G=k|\boldsymbol{X}=\boldsymbol{x}) = \frac{\exp(\beta_{k0}+\boldsymbol{\beta}_k^T\boldsymbol{x})}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\boldsymbol{\beta}_l^T\boldsymbol{x})}, \quad k=1,\ldots,K-1.$$

$$\Pr(G=K|\boldsymbol{X}=\boldsymbol{x}) = \frac{1}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\boldsymbol{\beta}_l^T\boldsymbol{x})}.$$

- The computation can be done by iterative reweighted least squares.
- In the high dimensional setting where $p \gg n$, we can use penalized likelihood.

Linear Discriminant Analysis

- Assume that in each class $G = k$, $\boldsymbol{X}$ follows multivariate normal distribution.

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)\right\}.$$

- One more assumption for LDA: $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$ for all k, i.e. common covariance matrix.
- Comparing any two classes k and l, it is sufficient to look at their log likelihood ratio

$$\begin{aligned}&\log \frac{\Pr(G = k|\boldsymbol{X} = \boldsymbol{x})}{\Pr(G = l|\boldsymbol{X} = \boldsymbol{x})}\\ &= \log \frac{f_k(\boldsymbol{x})}{f_l(\boldsymbol{x})} + \log \frac{\pi_k}{\pi_l}\\ &= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\boldsymbol{\mu}_k + \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k - \boldsymbol{\mu}_l) + \boldsymbol{x}^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k - \boldsymbol{\mu}_l).\end{aligned}$$

For two classes, we classify $\boldsymbol{x}$ to class 1 if and only if

$$\boldsymbol{x}^T \widehat{\boldsymbol{\Sigma}}^{-1}(\widehat{\boldsymbol{\mu}}_1 - \widehat{\boldsymbol{\mu}}_2) > \frac{1}{2}(\widehat{\boldsymbol{\mu}}_1 + \widehat{\boldsymbol{\mu}}_2)^T \widehat{\boldsymbol{\Sigma}}^{-1}(\widehat{\boldsymbol{\mu}}_1 - \widehat{\boldsymbol{\mu}}_2) - \log \frac{N_1}{N_2}, \qquad (3.41)$$

where N_1, N_2 are numbers of observations in each class.

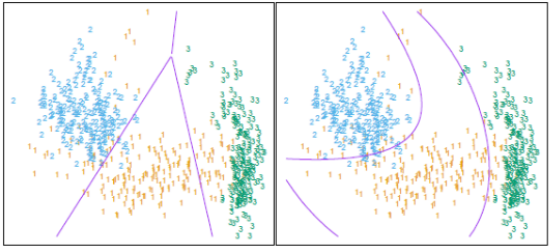


Figure: The left plot shows some data form three classes, with linear decision boundaries found by linear discriminant analysis. The right plot shows quadratic decision boundaries. These were obtained by finding linear boundaries in the five-dimensional space $X_1, X_2, X_1X_2, X_1^2, X_2^2$. Linear inequalities in this space are quadratic inequalities in the original space.

- Discriminant functions

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \log \pi_k.$$

- Estimate
  - $\widehat{\pi}_k = N_k/N$, where N_k is the number of observations of class-k.
  - $\widehat{\boldsymbol{\mu}}_k = \sum_{g_i=k} \boldsymbol{x}_i/N_k$.
  - $\widehat{\boldsymbol{\Sigma}} = \sum_{k=1}^K \sum_{g_i=k} (\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)^T/(N-K)$.

Quadratic Discriminant Analysis

- If $\boldsymbol{\Sigma}_k$ are not assumed to be equal,

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2}\log|\boldsymbol{\Sigma}_k| - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \log\pi_k.$$

- Number of parameters to be estimated: only need the difference $\delta_k(\boldsymbol{x}) - \delta_K(\boldsymbol{x})$,

$$\{1(\pi_k)+p(\mu_k)+\frac{p(p+1)}{2}(\boldsymbol{\Sigma})\}\times(K-1) = \left\{\frac{p(p+3)}{2}+1\right\}\times(K-1).$$

- Both LDA and QDA have good records of performance even if the underlying distribution differs from Normal distribution.

Illustration of QDA

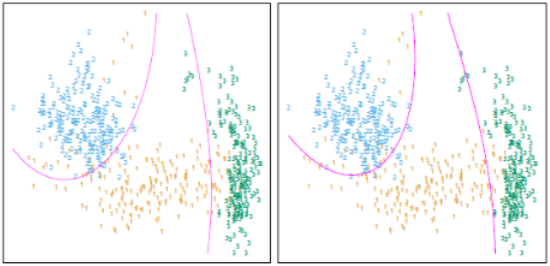


Figure: Two methods for fitting quadratic boundaries. [Left] Quadratic decision boundaries, obtained using LDA in the five-dimensional “quadratic” space. [Right] Quadratic decision boundaries found by QDA. The differences are small, as is usually the case.

Regularized Discriminant Analysis (RDA)

- A compromise between LDA and QDA. The regularized covariance matrices have the form,

$$\widehat{\mathbf{\Sigma}}_k(\alpha) = \alpha\widehat{\mathbf{\Sigma}}_k + (1-\alpha)\widehat{\mathbf{\Sigma}}, \quad \alpha \in [0,1],$$

  where $\widehat{\mathbf{\Sigma}}$ is the pooled sample covariance.
- Similar modification allow $\widehat{\mathbf{\Sigma}}$ itself to be shrunk toward the scalar covariance

$$\widehat{\mathbf{\Sigma}}(\gamma) = \gamma\widehat{\mathbf{\Sigma}} + (1-\gamma)\widehat{\sigma}^2\boldsymbol{I}, \quad \gamma \in [0,1].$$

- They together can lead to $\widehat{\mathbf{\Sigma}}(\alpha,\gamma)$.

Computations for LDA and QDA

- We can simplify the computation by matrix decomposition.
- Let $\widehat{\boldsymbol{\Sigma}}_k = \boldsymbol{U}_k \boldsymbol{D}_k \boldsymbol{U}_k^T$ be the SVD of $\widehat{\boldsymbol{\Sigma}}_k$, where $\boldsymbol{U}_k$ is the $p \times p$ orthogonal matrix. $\boldsymbol{D}_k$ is a diagonal matrix of positive eigenvalues d_{lk}. Then,

$$(\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_k)^T \widehat{\boldsymbol{\Sigma}}_k^{-1} (\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_k) = [\boldsymbol{U}_k^T (\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_k)]^T \boldsymbol{D}_k^{-1} [\boldsymbol{U}_k^T (\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_k)].$$

$$\log |\widehat{\boldsymbol{\Sigma}}_k| = \sum_l \log(d_{kl}).$$

- In light of the computational steps outlined above, the LDA can be implemented by the following steps:
  - *Sphere* the data with respect to the common covariance estimate $\widehat{\boldsymbol{\Sigma}} : X^* \leftarrow \boldsymbol{D}^{-1/2} \boldsymbol{U}^T X$, where $\widehat{\boldsymbol{\Sigma}} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$. The common covariance estimate of X^* will now be the identity.
  - Classify to the closest class centroid in the transformed space, modulo the effect of the class prior probabilities π_k.

Fisher’s Perspective of Discriminant Analysis

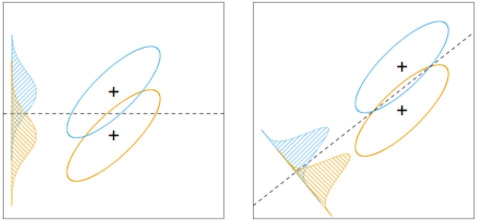


Figure: Although the line joining the centroids defines the direction of greatest centroid spread, the projected data overlap because of covariance (left panel). The discriminant direction minimizes this overlap for Gaussian data (right panel).

Fisher's Perspective of Discriminant Analysis

Find the linear combination $z = \boldsymbol{a}^T \boldsymbol{x}$ such the between-class variance is maximized relative to the within-class variance, i.e.,

$$\max_{\boldsymbol{a}} \frac{\boldsymbol{a}^T \boldsymbol{B} \boldsymbol{a}}{\boldsymbol{a}^T \boldsymbol{W} \boldsymbol{a}},$$

where the between class variance $\boldsymbol{B} = \sum_{k=1}^{K} N_k(\widehat{\boldsymbol{\mu}}_k - \bar{\boldsymbol{\mu}})(\widehat{\boldsymbol{\mu}}_k - \bar{\boldsymbol{\mu}})^T$, where $\bar{\boldsymbol{\mu}} = \sum_{i=1}^{N} \boldsymbol{x}_i / N$ and the within class variance $\boldsymbol{W} = \widehat{\boldsymbol{\Sigma}} = \sum_{k=1}^{K} \sum_{i=1}^{N_k} (\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)^T$.
What is $\boldsymbol{B} + \boldsymbol{W}$?
Exercise: Find the solution to the above problem, when $K = 2$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$ and $N_1 = N_2 = N/2$. Compare it with the LDA rule. What can you find?

R example of LDA and QDA

```
library(MASS)
SAheart<-read.table('SAheart.txt',sep=',',header=T)
model_lda <-lda(chd~sbp+tobacco+ldl+adiposity+
typea+obesity+alcohol+age,data=SAheart)
model_qda <-qda(chd~sbp+tobacco+ldl+adiposity+
typea+obesity+alcohol+age,data=SAheart)
```

- See data from EoSL.

LDA in the high dimensional setting

Recall that LDA classifies $\boldsymbol{x}$ to class 1 if and only if

$$\boldsymbol{x}^T\widehat{\boldsymbol{\Sigma}}^{-1}(\widehat{\boldsymbol{\mu}}_1-\widehat{\boldsymbol{\mu}}_2) > \frac{1}{2}(\widehat{\boldsymbol{\mu}}_1+\widehat{\boldsymbol{\mu}}_2)^T\widehat{\boldsymbol{\Sigma}}^{-1}(\widehat{\boldsymbol{\mu}}_1-\widehat{\boldsymbol{\mu}}_2) - \log\frac{N_1}{N_2}, \qquad (3.42)$$

Compared it with the Bayes rule, we see that it essentially replace the unknown parameters $(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}, \pi_1, \pi_2)$ with the MLE.
In the low dimensional case $p = o(\sqrt{n})$, MLEs are consistent. When $p \gg n$, MLE no longer works. $\widehat{\boldsymbol{\Sigma}}$ is not even invertible.

- Bickel and Levina (2004) proves that when $p > n$, LDA can be as bad as random guessing, i.e misclassification error $\to 1/2$.
  - See Bickel, P. J., & Levina, E. (2004). Some theory for Fisher's linear discriminant function,'naive Bayes', and some alternatives when there are many more variables than observations. Bernoulli, 989-1010.

- The reason is that the errors of estimating these parameters can accumulate and finally ruin the classification when $p \gg n$.
- We will see three remedies of LDA in the high dimensional setting.

Method 1: Sparse LDA.

- Shao, J., Wang, Y., Deng, X., & Wang, S. (2011). Sparse linear discriminant analysis by thresholding for high dimensional data. The Annals of statistics, 39(2), 1241-1265.
- Key idea: using regularized estimator of $\widehat{\boldsymbol{\Sigma}}$, $\widehat{\boldsymbol{\mu}}_1$ and $\widehat{\boldsymbol{\mu}}_2$.
    - $\widehat{\boldsymbol{\Sigma}}$ can be the thresholding estimator we learned before.
    - $\widehat{\boldsymbol{\mu}}_1$ and $\widehat{\boldsymbol{\mu}}_2$ can also be the thresholding estimator, i.e. the MLEs being thresholded at certain level.
    - As long as $\boldsymbol{\Sigma}$, $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$ are sparse, the thresholding estimators will be consistent to the true parameters.
    - The resulting sparse LDA rule is still error-rate consistent, i.e. its error rate converges to the error rate of the Bayes rule.

- Key of proof: For simplicity, we assume $N_1 = N_2 = N/2$. The error-rate consistency can be proved through the following steps.
- The optimal Bayes rule classifies $\boldsymbol{x}$ to class 1 if and only if

$$\boldsymbol{\delta}^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \bar{\boldsymbol{\mu}}) > 0, \tag{3.43}$$

  where $\boldsymbol{\delta} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$ and $\bar{\boldsymbol{\mu}} = (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)/2$. [exercise]
- The error rate of the Bayes rule is $\Phi(-\sqrt{\Delta_p}/2)$, where $\Delta_p = \boldsymbol{\delta}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta}$ and $\Phi(\cdot)$ is the cumulative distribution function of standard normal distribution. [exercise]
- The error rate of the LDA rule in (3.42) is [exercise]

$$\frac{1}{2} \sum_{k=1}^{2} \Phi \left(\frac{(-1)^k \widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{\mu}_k - \widehat{\boldsymbol{\mu}}_k) - \widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1} \widehat{\boldsymbol{\delta}}/2}{\sqrt{\widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma} \widehat{\boldsymbol{\Sigma}}^{-1} \widehat{\boldsymbol{\delta}}}} \right).$$

- We can show that for both $k = 1, 2$, [refer to the paper for convergence rates]

$$\frac{(-1)^k \widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{\mu}_k - \widehat{\boldsymbol{\mu}}_k) - \widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1}\widehat{\boldsymbol{\delta}}/2}{\sqrt{\widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\Sigma}\widehat{\boldsymbol{\Sigma}}^{-1}\widehat{\boldsymbol{\delta}}}} \xrightarrow{P} -\sqrt{\Delta_p}/2.$$

- Since $\Phi(\cdot)$ is a continuous function, the result follows. A key inequality to use

$$\frac{x}{1+x^2}\phi(x) \le \Phi(-x) \le \frac{1}{x}\phi(x),$$

for $x > 0$, where $\phi(x)$ is the p.d.f of the standard normal distribution. This inequality shows that the tail probability $\Phi(-x)$ is in the same rate of $(1/x)e^{-x^2/2}$.

Method 2: Linear Programming Discriminant (LPD) rule

- Note that, from the Bayes rule (3.43), we don't need to estimate $\boldsymbol{\Sigma}$ (or its inverse) and $\boldsymbol{\delta}$ separately. We only need to estimate $\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$. Let $\boldsymbol{\beta}^* := \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$.
- Therefore, $\boldsymbol{\beta}^*$ solves $\boldsymbol{\Sigma}\boldsymbol{\beta} = \boldsymbol{\delta}$.
- In practice, we solve

$$\widehat{\boldsymbol{\beta}} = \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \|\boldsymbol{\beta}\|_1$$
$$\text{subject to } \|\boldsymbol{S}\boldsymbol{\beta} - (\widehat{\boldsymbol{\mu}}_1 - \widehat{\boldsymbol{\mu}}_2)\|_{\max} \leq \lambda_n,$$

  where $\boldsymbol{S}$ is the pooled sample covariance matrix, $\widehat{\boldsymbol{\mu}}_1$ and $\widehat{\boldsymbol{\mu}}_2$ are sample means from the two class and λ_n is a tuning parameter.
- It means that "we find the sparest solution $\widehat{\boldsymbol{\beta}}$ such that $\boldsymbol{S}\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\delta}}$ is sufficiently small."

- $\|\boldsymbol{\beta}\|_1$ regulates the sparsity level of $\boldsymbol{\beta}$ and λ_n regulates how close $\boldsymbol{S}\boldsymbol{\beta}$ is to $\widehat{\boldsymbol{\delta}}$.
- In order to establish error-rate consistence, we need to let $\lambda_n \to 0$ at a good rate.
- Once we have $\widehat{\boldsymbol{\beta}}$, we can classify a new observation $\boldsymbol{x}$ to class 1 if and only if
$$(\boldsymbol{x} - \widehat{\boldsymbol{\mu}})^T \widehat{\boldsymbol{\beta}} \geq 0,$$
where $\widehat{\boldsymbol{\mu}} = (\widehat{\boldsymbol{\mu}}_1 + \widehat{\boldsymbol{\mu}}_2)/2$.
- This is called “linear programming discriminant” (LPD) rule by the authors.
- Cai, T., & Liu, W. (2011). A Direct Estimation Approach to Sparse Linear Discriminant Analysis. Journal of the American Statistical Association, 106(496), 1566-1577.

- In order to solve $\widehat{\beta}$, we can transform it to standard linear programming problem, which can be solved by some standard optimization software.
- See details in section 5.1 of the paper.
- The paper proves that if $\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$ is sparse, the LPD rule's error rate is consistent to Bayes error rate.
- The proofs follow similar arguments as in Shao et al.

- Indeed, the LPD problem is very similar to CLIME problem. It also teaches us how to solve a more general problem.
- Suppose we want to estimate $\boldsymbol{\beta}^*$, which is the solution to a general equation $\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \mathbf{0}$, where $\boldsymbol{f}$ is a multivariate function and $\boldsymbol{\theta}$ is some other unknown population parameters.
- We can always solve

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \|\boldsymbol{\beta}\|_1$$
$$\text{subject to } \|\boldsymbol{f}(\boldsymbol{\beta}, \widehat{\boldsymbol{\theta}})\|_{\max} < \lambda_n,$$

where $\widehat{\boldsymbol{\theta}}$ is the MLE of $\boldsymbol{\theta}$ and λ_n is a tuning parameter.

- Indeed, we find the sparsest $\widehat{\boldsymbol{\beta}}$ such that $\boldsymbol{f}(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\theta}})$ is sufficiently close to 0.

Method 3: Regularized Optimal Affine Discriminant (ROAD)

- Fan, J., Feng, Y., & Tong, X. (2012). A road to classification in high dimensional space: the regularized optimal affine discriminant. Journal of the Royal Statistical Society: Series B, 74(4), 745-771.
- We consider a general linear discriminant rule that classifies $\boldsymbol{x}$ to class 1 if and only if
$$\boldsymbol{w}^T(\boldsymbol{x} - \bar{\boldsymbol{\mu}}) > 0,$$
where $\bar{\boldsymbol{\mu}} = (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)/2$ and $\boldsymbol{w}$ is the classification direction vector that needs to be determined.
- It can be shown that the error rate of this rule is
$$1 - \Phi(-\boldsymbol{w}^T\boldsymbol{\mu}_d/(\boldsymbol{w}^T\Sigma\boldsymbol{w})^{1/2}), \tag{3.44}$$
where $\boldsymbol{\mu}_d := (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)/2$.

- Naturally, we want to minimize the rate in (3.44), which is equivalent to minimize $\boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w}$ subject to $\boldsymbol{w}^T \boldsymbol{\mu}_d = 1$. Moreover, we want a sparse rule. Therefore, we control the L_1-norm of $\boldsymbol{w}$ and solve

$$\boldsymbol{w}_c = \operatorname*{argmin}_{\|\boldsymbol{w}\|_1 \le c, \boldsymbol{w}^T \boldsymbol{\mu}_d = 1} \boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w}.$$

- In practice, we can replace $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}_d$ with their MLEs and solve

$$\widehat{\boldsymbol{w}}_c = \operatorname*{argmin}_{\|\boldsymbol{w}\|_1 \le c, \boldsymbol{w}^T \widehat{\boldsymbol{\mu}}_d = 1} \boldsymbol{w}^T \widehat{\boldsymbol{\Sigma}} \boldsymbol{w}. \tag{3.45}$$

- The augmented Lagrangian form is

$$\operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{2} \boldsymbol{w}^T \widehat{\boldsymbol{\Sigma}} \boldsymbol{w} + \lambda \|\boldsymbol{w}\|_1 + \frac{1}{2} \gamma (\boldsymbol{w}^T \widehat{\boldsymbol{\mu}}_d - 1)^2,$$

  where λ and γ are Lagrangian multipliers.
- This is a standard convex optimization problem, which can be solved by coordinate gradient descent algorithm.
- Read section 4 of the paper to see the details of the algorithm.
- ROAD's error rate is also consistent to Bayes error rate given some sparsity assumptions.

Logistic regression vs LDA

- LDA:

$$\begin{aligned}
&\log \frac{\Pr(G=k|\boldsymbol{X}=\boldsymbol{x})}{\Pr(G=l|\boldsymbol{X}=\boldsymbol{x})}\\
&= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\boldsymbol{\mu}_k+\boldsymbol{\mu}_l)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k-\boldsymbol{\mu}_l) + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k-\boldsymbol{\mu}_l)\\
&= \alpha_{j0} + \boldsymbol{\alpha}_j^T\boldsymbol{x}
\end{aligned}$$

- LR:

$$\log \frac{\Pr(G=k|\boldsymbol{X}=\boldsymbol{x})}{\Pr(G=l|\boldsymbol{X}=\boldsymbol{x})} = \beta_{j0} + \boldsymbol{\beta}_j^T\boldsymbol{x}.$$

- Assumptions: LDA needs normality assumption on $\boldsymbol{X}$, LR does not have such an assumption.

- LDA uses the full joint distribution $Pr(X, G = j) = Pr(X)Pr(G = j|X)$. LR uses the conditional distribution of Y given x to estimate parameters.
- Based on some empirical experience, if normality holds, LDA is up to 30% more efficient; otherwise LR can be more robust.

Comparison between LDA and Logistic Regression

Properties	LDA	Logistic Regression
Model form (odds-ratio)	$\alpha_{k0} + \boldsymbol{\alpha}_k^T \boldsymbol{x}$	$\beta_{k0} + \boldsymbol{\beta}_k^T \boldsymbol{x}$
Assumption	$\boldsymbol{X}$ is Gaussian	No
Likelihood	unconditional	conditional

Virtues of LDA

- It is a simple prototype classifier.
- LDA is the estimated Bayes classifier if the observations are multivariate Gaussian, with a common covariance matrix.
- The decision boundaries are linear, leading to simple decision rules.
- Even for some complicated data, LDA often produces acceptable classification results, because of its simplicity and low variance.

Drawbacks of LDA

- Linear decision boundaries often do not adequately separate the classes.
- A single prototype per class is insufficient.
- If we have too many correlated predictors, the performance of LDA suffers.

Different Ways to Extend LDA

- Recast LDA as a linear regression problem. Apply nonparametric forms of regression, which in turn leads to more flexible forms of discriminant analysis (FDA).
- In case of too many predictors, we penalize the coefficients in LDA model to be smooth (PDA).
- Model each class by a mixture of two or more Gaussians with different centroids, but with every component Gaussian, both within and between classes, sharing the same covariance matrix (MDA).

Flexible Discriminant Analysis There are K classes $\mathcal{G} = \{1, \ldots, K\}$, each having measured features X. Suppose $\theta : \mathcal{G} \mapsto \mathcal{R}^1$ assigning scores to the classes, such that we solve

$$\min_{\beta,\theta} \sum_{i=1}^{N} \left(\theta(g_i) - x_i^T \beta\right)^2$$

with restrictions on θ to avoid a trivial solution (See Exercise 12.6 of EoSL). More generally, we can find up to $L \leq K - 1$ sets of independent scorings $\theta_1, \theta_2, \ldots, \theta_L$, and L corresponding linear maps $\eta_l(X) = X^T \beta_l$, to minimize

$$ASR = \frac{1}{N} \sum_{l=1}^{L} \left[\sum_{i=1}^{N} \left(\theta_l(g_i) - x_i^T \beta_l\right)^2 \right].$$

More generally, we can replace $\eta_l(x) = x^T\beta_l$ by more flexible nonparametric fits, such as spline functions, etc. The regression problem is

$$ASR(\{\theta_l, \eta_l\}_{l=1}^{L}) = \frac{1}{N}\sum_{l=1}^{L}\left[\sum_{i=1}^{N}(\theta_l(g_i) - \eta_l(x_i))^2 + \lambda J(\eta_l)\right],$$

where J is an appropriate regularizer on the smoothness of $\eta_l(x)$.

- It can be shown that LDA can be performed by a sequence of linear regressions, followed by classification to the closest class centroid in the space of fits.
- For more details, read Section 12.5 of EoSL.

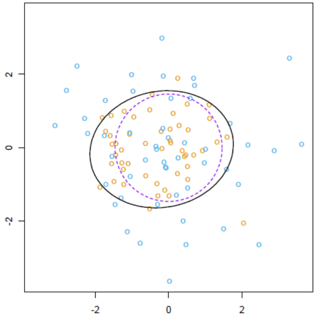


Figure: $N(0, \boldsymbol{I})$ vs $N(0, 9/4\boldsymbol{I})$. The ellipse is the decision boundary found by FDA. The circle is Bayes rule boundary.

Suppose we use degree–2 polynomial regression for each η_l, the FDA boundary is *identical* to the decision boundary that we augment our original predictors with their squares and cross–products, perform an LDA in the enlarged space and map down to the quadratic boundaries in the original space.

The Bayes decision boundary is the sphere, which is a linear boundary in the enlarged space.

Computing the FDA Estimates

$$\begin{array}{c|ccccc} & C_1 & C_2 & C_3 & C_4 & C_5 \\ \hline g_1 = 2 & 0 & 1 & 0 & 0 & 0 \\ g_2 = 1 & 1 & 0 & 0 & 0 & 0 \\ g_3 = 1 & 1 & 0 & 0 & 0 & 0 \\ g_4 = 5 & 0 & 0 & 0 & 0 & 1 \\ g_5 = 4 & 0 & 0 & 0 & 1 & 0 \\ \vdots & & & \vdots & & \\ g_N = 3 & 0 & 0 & 1 & 0 & 0 \end{array}$$

We create an $N \times K$ *indicator response matrix* $\boldsymbol{Y}$ from the responses g_i, s.t. $y_{ik} = 1$ if $g_i = k$, otherwise $y_{ik} = 0$. For a five–class problem $\boldsymbol{Y}$ might look like the left matrix.

- *Multivariate nonparametric regression.* Fit a multiresponse, adaptive nonparametric regression of $\boldsymbol{Y}$ on $\boldsymbol{X}$, giving fitted values $\widehat{\boldsymbol{Y}}$. Let $\boldsymbol{S}_\lambda$ be the linear operator that first the final chosen model, and $\eta^*(x)$ be the vector of fitted regression functions.
- *Optimal scores.* Compute the eigen–decomposition of $\boldsymbol{Y}^T\widehat{\boldsymbol{Y}} = \boldsymbol{Y}^T\boldsymbol{S}_\lambda\boldsymbol{Y}$, where the eigenvectors $\boldsymbol{\Theta}$ are normalized: $\boldsymbol{\Theta}^T\boldsymbol{D}_\pi\boldsymbol{\Theta} = \boldsymbol{I}$. Here $\boldsymbol{D}_\pi = \boldsymbol{Y}^T\boldsymbol{Y}/N$ is a diagonal matrix of the estimated class prior probabilities.
- *Update* the model from step 1 using the optimal scores: $\eta(x) = \boldsymbol{\Theta}^T\eta^*(x)$.

Penalized Discriminant Analysis

Suppose the regression procedure used in FDA amounts to a linear regression onto a basis expansion $h(X)$, with a quadratic penalty on the coefficients:

$$ASR = \frac{1}{N} \sum_{l=1}^{L} \left[\sum_{i=1}^{N} \left(\theta_l(g_i) - h(x_i)^T \beta_l \right)^2 + \lambda \beta_l^T \Omega \beta_l \right].$$

The choice of Ω depends on the problem.

- If $\eta_l(x) = h(x)\beta_l$ is an expansion on spline basis functions, Ω constrain η_l to be smooth.
- In the case of additive splines, there are N spline basis functions for each coordinate; Ω is $Np \times Np$ block diagonal.

The steps in FDA can be viewed as a generalized form of LDA, which we call penalized discriminant analysis, or PDA:

- Enlarge the set of predictors X via a basis expansion $h(X)$.
- Use (penalized) LDA in the enlarged space, where the penalized Mahalanobis distance is given by

$$D(x, \mu) = (h(x) - h(\mu))^T (\Sigma_W + \lambda\Omega)^{-1} (h(x) - h(\mu)),$$

  where Σ_W is the within–class covariance matrix of the derived variable $h(x_i)$.
- Decompose the classification subspace using a penalized metric:

$$\max u^T \Sigma_{\text{Bet}} u \quad \text{subject to } u^T (\Sigma_W + \lambda\Omega) u = 1.$$

Mixture Discriminant Analysis

In many situations a single prototype is not sufficient to represent inhomogeneous classes, and mixture models are more appropriate.
A Gaussian mixture model for the *k*th class has density

$$P(X|G=k) = \sum_{r=1}^{R_k} \pi_{kr}\phi(X;\mu_{kr},\Sigma),$$

where the *mixing proportions* π_{kr} sum to one.
The class posterior probabilities are given by

$$P(G=k|X=x) = \frac{\sum_{r=1}^{R_k} \pi_{kr}\phi(X;\mu_{kr},\Sigma)\Pi_k}{\sum_{l=1}^{K}\sum_{r=1}^{R_k} \pi_{lr}\phi(X;\mu_{lr},\Sigma)\Pi_l}$$

Computation

Directly obtaining MLE from the joint log–likelihood

$$\sum_{k=1}^{K}\sum_{g_i=k}\log\left[\sum_{r=1}^{R_k}\pi_{kr}\phi(x_i;\mu_{kr},\Sigma)\Pi_k\right]$$

is not easy. Instead, use EM.

- E–step: Given the current parameters, compute the *responsibility* of sub–class c_{kr} within class k for each of the class–k obs. $(g_i = k)$:

$$W(c_{kr}|x_i,g_i) = \frac{\pi_{kr}\phi(x_i;\mu_{kr},\Sigma)}{\sum_{l=1}^{R_k}\pi_{kl}\phi(x_i;\mu_{kl},\Sigma)}.$$

- M–step: Compute the weighted MLEs for the parameters of each of the component Gaussians within each of the classes, using the weights from the E–step.

Support Vector Machine

Separable case

Training data: $(x_1, y_1), \dots, (x_N, y_N)$ with $x_i \in \mathbb{R}^d$ and $y \in \{-1, 1\}$ is the class label. This is a *supervised* learning problem, where the class label is known.

Task: find the best hyperplane $\{x : f(x) = \beta^T x + \beta_0 = 0\}$ to separate the two classes.

The corresponding classification rule is

$$G(x) = \mathrm{sign}(f(x)) = \begin{cases} 1 & \text{if } f(x) \geq 0, \\ -1 & \text{if } f(x) < 0. \end{cases}$$

$f(x) = \beta^T x + \beta_0 = 0$ defines a hyperplane (affine set) L.

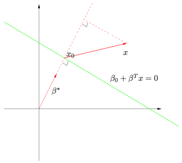


Figure: The linear algebra of a hyperplane (affine set).

Here are some properties of L.

- For any two points x_1 and x_2 lying in L, $\beta^T(x_1 - x_2) = 0$, hence $\beta^* = \beta/\|\beta\|_2$ is the vector normal to the surface of L.
- For any points x_0 in L, $\beta^T x_0 = -\beta_0$.
- The signed distance of any point x to L is

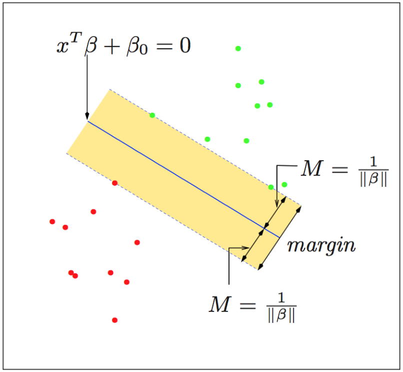
$$\|x - x_0\|_2 \frac{\beta^T(x - x_0)}{\|\beta\|_2 \|x - x_0\|_2} = \frac{1}{\|\beta\|_2}(\beta^T x + \beta_0) = \frac{f(x)}{\|f'(x)\|_2}.$$

- $f(x)$ is proportional to the signed distance from x to the hyperplane defined by $f(x) = 0$.

Optimal Separating Hyperplane

$yf(x)$ gives the distance from x to the hyperplane defined by $f(x) = 0$. We call it *margin*. Consider the following optimization problem

$$\max_{\beta,\beta_0,\|\beta\|_2=1} M \qquad (4.46)$$
$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq M, \text{ for } i = 1, \dots, N.$$



- The optimal separating hyperplane maximizes the margin of all data points to that plane.
- The constraint $\|\beta\|_2 = 1$ is used here to avoid the identifiability issue, since $\text{sign}(\beta^T x + \beta_0) = \text{sign}(c(\beta^T x + \beta_0))$ for $c > 0$.
- We further simplify our problem (4.46). To remove the constraint that $\|\beta\|_2 = 1$, Notice that, (4.46) is equivalent to

$$y_i(x_i^T \beta + \beta_0) \geq M\|\beta\|_2.$$

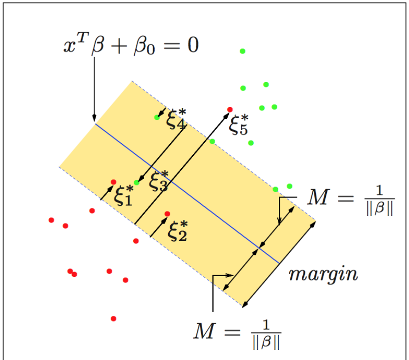
- Since for any β and β_0 satisfy these inequalities, any positively scaled multiple satisfies them too. Then, we can set $M = 1/\|\beta\|_2$. (4.46) becomes

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|_2^2$$
$$\text{subject to } y_i(x_i^T \beta + \beta_0) \geq 1, \text{ for } i = 1, \ldots, N.$$

Overlap case: To deal with overlapping, we still maximize M, but allow for some points on the wrong side of the margin. Define the slack variables $\xi = (\xi_1, \dots, \xi_N)$. Two ways for relaxing the constraint in (4.46) are

$$y_i(x_i^T\beta + \beta_0) \geq M - \xi_i \text{ or } y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i),$$

$$\forall i, \xi_i \geq 0, \sum_{i=1}^{N} \xi_i \leq \text{constant}.$$



- We control the sum $\sum_{i=1}^{N} \xi_i$ by a constant to ensure the total offset cannot be too large.
- The constant plays the role a tuning parameter, whose optimal value can be chosen by cross validation.
- The first relaxation measures the overlap in actual distance from the margin; the second relaxation measures the relative distance. Usually we use the second choice as it leads to a convex problem.
- By the same argument as in the separating case, we can transform it into the standard SVM problem.

$$
\begin{aligned}
&\min \frac{1}{2}\|\beta\|_2^2 + C\sum_{i=1}^{N} \xi_i \\
&\text{subject to } \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \quad \forall i. \\ \xi_i \geq 0. \end{cases}
\end{aligned}
\tag{4.47}
$$

- The solution of β to the SVM problem has the form

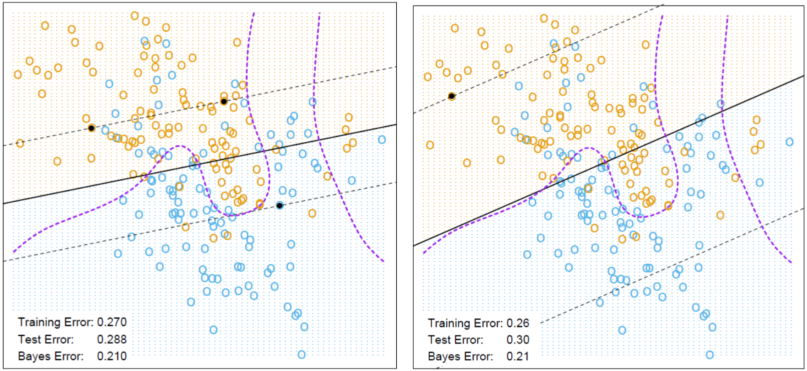
$$\widehat{\beta} = \sum_{i=1}^{N} \widehat{\alpha}_i y_i x_i,$$

  where $\widehat{\alpha}_i$ is nonzero only for points(vectors) close to the classification boundary, which are called "support vectors". $\widehat{\beta}_0$ can be found accordingly. The resulting classification rule is $\widehat{G}(x) = \mathrm{sign}(x^T\widehat{\beta} + \widehat{\beta}_0)$.
- A point(vector) i well inside the boundary has $\widehat{\alpha}_0 = 0$. Therefore, it does not contribute to the classification rule.
- Details of the solution can be found in Section 12.2.1 of EoSL.
- R package: `e1071`

Mixture Example

Setup:

- Generate 10 means m_k from $N((1,0)^T, \boldsymbol{I})$ and label as blue.
- 10 more drawn from $N((0,1)^T, \boldsymbol{I})$ and label as orange.
- For each class generate 100 observations as follows: picked an m_k at random w.p 1/10, and then generate a $N(m_k, \boldsymbol{I}/5)$.



$C = 10000$ $C = 0.01$

Figure: The linear support vector boundary for mixture data example, for two different values of C. The broken lines indicate the margins, where $f(x) = \pm 1$. The support points are all points on the wrong side of their margin. The black solid dots are those support points falling exactly on the margin. In the left panel, 62% are support points. In the right panel, 85% are support points. The purple line is the Bayes decision boundary.

- Points on the wrong side of the boundary are support vectors.
- Points on the correct side of the boundary but close to it (in the margin), are also support vectors.
- Points well inside the correct side of the boundary are not support vectors.
- Larger values of C focus attention more on (correctly classified) points near the boundary.
- Misclassified points are given weight, no matter how far away.

Support Vector Machines and Kernels

- The SVM classifier described so far gives a linear boundary. How to get a nonlinear boundary?
- Enlarge the feature space using basis expansions such as polynomials or splines.
- Linear boundaries in the enlarged space achieve better training–class separation, and translate to nonlinear boundaries in the original space.
- Fit SVM classifier using input features $h(x_i) = (h_1(x_i), h_2(x_i), \ldots, h_M(x_i))$ and produce $\widehat{f}(x) = h(x)^T\widehat{\beta} + \widehat{\beta}_0$. The classifier is $\widehat{G}(x) = \text{sign}(\widehat{f}(x))$.

There are two concerns.

- If the dimension of the enlarged space is very large, computations would become prohibitive.
- With sufficient basis functions, the data is separable, and overfitting would occur.

Computing the SVM for Classification

The solution function has the form

$$\begin{aligned} f(x) &= h(x)^T\beta + \beta_0 \\ &= \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i)\rangle + \beta_0. \end{aligned}$$

- As in the linear case, nonzero α_i's only depend on support points. No need to specify $h(x)$ at all, but require only knowledge of the kernel function

$$K(x, x') = \langle h(x), h(x_i)\rangle,$$

where K is a positive (semi–) definite function.

Three popular choices for K in the SVM literature.

- dth–Degree polynomial: $K(x,x') = (1 + \langle x, x' \rangle)^d$,
- Radial basis: $K(x,x') = \exp(-\gamma \|x - x'\|^2)$,
- Neural network: $K(x,x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$.

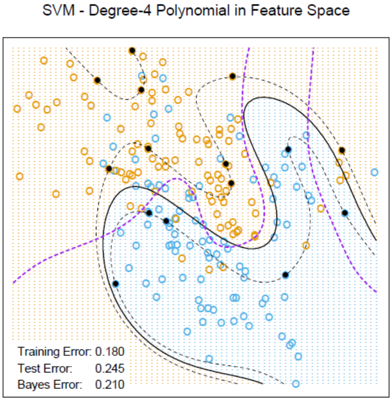
Consider a feature space with two input points X_1 and X_2, and a polynomial kernel of degree 2. Then,

$$\begin{aligned} K(X,X') &= (1 + \langle X, X' \rangle)^2 = (1 + X_1X_1' + X_2X_2')^2 \\ &= 1 + 2X_1X_1' + 2X_2X_2' + (X_1X_1')^2 + (X_2X_2')^2 + 2X_1X_1'X_2X_2', \end{aligned}$$

which is a linear function in the expanded space but a quadratic function in the original space. For more details of kernel functions, see Section 5.8.1 of EoSL.

The role of parameter C

- A large value of C will lead to and overfit wiggly boundary;
- A small value of C will encourage a smoother boundary.



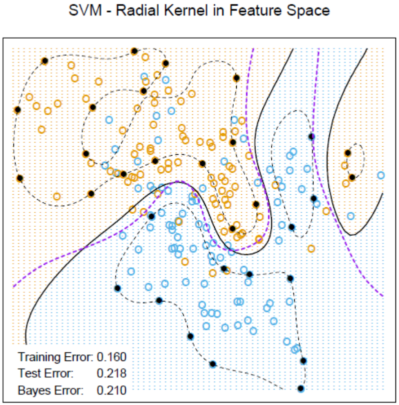


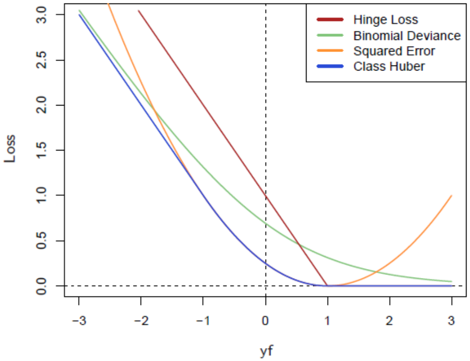
Figure: Two nonlinear SVMs for the mixture data. The left panel uses a 4th degree polynomial kernel. The right panel uses a radial basis kernel with $\gamma = 1$. It performs closely to the Bayes optimal, as might be expected given the data arise from mixtures of Gaussians. The purple line is the Bayes decision boundary.

The SVM as a Penalization Method

With $f(x) = x^T\beta + \beta_0$, consider

$$\min_{\beta_0,\beta} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2}\|\beta\|^2. \tag{4.48}$$

Setting $\lambda = 1/C$, it can be shown that the solution to (4.48) is the same as the solution to SVM problem (4.47). [exercise]



The SVM as a Penalization Method

- Hinge loss gives zero penalty to points inside its margin and a linear penalty to other points.
- Binomial deviance (i.e., - log(Binomial likelihood)) gives a small penalty to points well inside its margin and a linear penalty far away.
- Squared-error gives a quadratic penalty to all points.
- Huberised square hinge loss (by Rosset and Zhu) shares attractive properties of logistic regression and SVM hinge loss. It produces a robust classifier (i.e. a classifier that is insensitive to outliers) by penalizing less heavily than a quadratic loss.

We can characterize these loss functions in terms of what they are estimating at the population level. Consider minimizing $\mathrm{E}L(Y,f(X))$. Whereas the hinge loss estimates the classifier $G(x)$ itself, all the others estimate a transformation of the class posterior probabilities.

Loss Function	$L[y,f(x)]$	Minimizing Function
Binomial Deviance	$\log[1+e^{-yf(x)}]$	$f(x)=\log\frac{\Pr(Y=+1\|x)}{\Pr(Y=-1\|x)}$
SVM Hinge Loss	$[1-yf(x)]_+$	$f(x)=\mathrm{sign}[\Pr(Y=+1\|x)-\frac{1}{2}]$
Squared Error	$[y-f(x)]^2=[1-yf(x)]^2$	$f(x)=2\Pr(Y=+1\|x)-1$
“Huberised” Square Hinge Loss	$-4yf(x)$, $\quad yf(x)<-1$ $[1-yf(x)]_+^2$ $\quad$ otherwise	$f(x)=2\Pr(Y=+1\|x)-1$

More generally, consider

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \lambda J(f), \tag{4.49}$$

where $\mathcal{H}$ is a space of functions, and $J(f)$ a regularizer on that space, e.g., $\mathcal{H}$ is the space of additive functions $f(x) = \sum_{j=1}^{p} f_j(x_j)$, and $J(f) = \sum_j \int \{f_j''(x_j)\}^2 dx_j$ penalizes the roughness of function f. It can be shown that the solution to this problem is an additive cubic spline, and has a kernel representation.

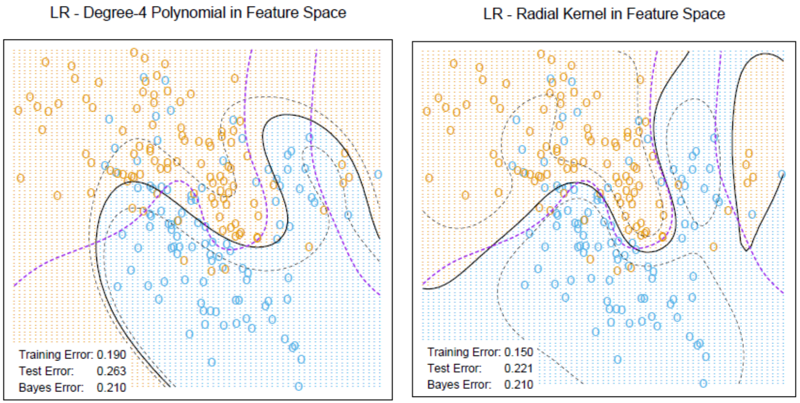


Figure: The same kernels but log-likelihood loss instead of SVM loss. The two broken contours correspond to posterior probabilities of 0.75 and 0.25 for +1 class. The purple curve is the Bayes decision boundary.

Support Vector Machine for Regression

Consider linear regression model

$$f(x) = x^T\beta + \beta_0.$$

To estimate β, consider minimization of

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V_\epsilon(y_i - f(x_i)) + \frac{\lambda}{2}\|\beta\|^2,$$

where

$$V_\epsilon(r) = \begin{cases} 0 & \text{if } |r| < \epsilon, \\ |r| - \epsilon, & \text{otherwise.} \end{cases}$$

is an "ϵ-insensitive" error measure.

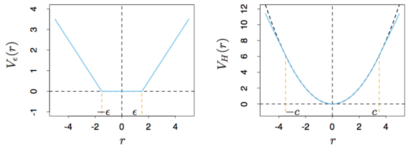


Figure: The left panel shows the ϵ-insensitive error function used by support vector regression machine. The right panel shows the error function used in Huber's robust regression. Beyond $|c|$, the function changes from quadratic to linear.

The loss function $V_\epsilon(r)$ mimics the Huber loss for robust regression. Huber loss has the form

$$V_H(r) = \begin{cases} r^2/2 & \text{if } |r| \le c, \\ c|r| - c^2/2 & \text{if } |r| > c. \end{cases}$$

Huber loss is a hybrid of quadratic loss and absolute error loss. It penalizes less heavily on outliers. Hence, it yields a robust estimator.

If $\widehat{\beta}$, $\widehat{\beta}_0$ are the minimizer of H, the solution function has the form

$$\widehat{\beta} = \sum_{i=1}^{N} (\widehat{\alpha}_i^* - \widehat{\alpha}_i) x_i,$$

$$\widehat{f}(x) = \sum_{i=1}^{N} (\widehat{\alpha}_i^* - \widehat{\alpha}_i) \langle x, x_i \rangle + \beta_0,$$

where $\widehat{\alpha}_i, \widehat{\alpha}_i^*$ are solution to certain optimization problem. (See (12.41) of EoSL)

- The solution depends on the input values only through the inner products $\langle x, x_i \rangle$;
- Only a subset of solution values $(\widehat{\alpha}_i^* - \widehat{\alpha}_i)$ are nonzero, and the associated data values are called the support vectors.
- The regularization parameter λ can be estimated by CV.

Regression and Kernels

More generally, consider approximation of the regression function in terms of a set of basis functions $\{h_m(x)\}$, $m = 1, 2, \dots, M$:

$$f(x) = \sum_{m=1}^{M} \beta_m h_m(x) + \beta_0.$$

To estimate β and β_0, we minimize

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2} \sum \beta_m^2,$$

for some error measure $V(r)$. The solution $\widehat{f}(x) = \sum \widehat{\beta}_m h_m(x) + \widehat{\beta}_0$ has the form

$$\widehat{f}(x) = \sum_{i=1}^{N} \widehat{a}_i K(x, x_i),$$

with $K(x, y) = \sum_{m=1}^{M} h_m(x) h_m(y)$.

Let's work out the case $V(r) = r^2$. Denote $\boldsymbol{H}$ as the $N \times M$ basis matrix with (i, m)th element $h_m(x_i)$ and suppose $M > N$ is large. For simplicity, assume $\beta_0 = 0$.
We estimate β by minimizing

$$H(\boldsymbol{\beta}) = (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|^2.$$

The solution is

$$\widehat{\boldsymbol{y}} = \boldsymbol{H}\widehat{\boldsymbol{\beta}}$$

with $\widehat{\boldsymbol{\beta}}$ determined by

$$\boldsymbol{H}^T(\boldsymbol{y} - \boldsymbol{H}\widehat{\boldsymbol{\beta}}) + \lambda\widehat{\boldsymbol{\beta}} = 0.$$

From this it appears that we need to evaluate the $M \times M$ matrix of inner products in the transformed space. However, we can premultiply by $\boldsymbol{H}$ to give

$$\boldsymbol{H}\widehat{\boldsymbol{\beta}} = (\boldsymbol{H}\boldsymbol{H}^T + \lambda\boldsymbol{I})^{-1}\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{y}.$$

Then, we only need to evaluate the inner product kernel $\{\boldsymbol{H}\boldsymbol{H}^T\}_{i,i'} = K(x_i, x_{i'})$.

Then, we can show that the predicted values at an arbitrary x satisfy

$$\widehat{f}(x) = h(x)^T \widehat{\boldsymbol{\beta}} = \sum_{i=1}^{N} \widehat{\alpha}_i K(x, x_i),$$

where $\widehat{\alpha} = (\boldsymbol{H}\boldsymbol{H}^T + \lambda \boldsymbol{I})^{-1}\boldsymbol{y}$. As before, we do not need to evaluate the large set of functions $h_1(x), \ldots, h_M(x)$. Only the inner product kernel $K(x_i, x_{i'})$ need to be evaluated at the N training points for each i, i' and at points x for predictions there.